Amazon Web Services, Inc.

Amazon Linux 2023 OpenSSL FIPS Provider

FIPS 140-3 Non-Proprietary Security Policy

Document Version 1.2
Last update: 2025-05-14

Prepared by:

atsec information security corporation

4516 Seton Center Pkwy, Suite 250

Austin, TX 78759

www.atsec.com

Table of Contents

1 General 6
1.1 Overview 6
1.1.1 How this Security Policy was Prepared 6
1.2 Security Levels 6
2 Cryptographic Module Specification 7
2.1 Description 7
2.2 Tested and Vendor Affirmed Module Version and Identification 8
2.3 Excluded Components 9
2.4 Modes of Operation 9
2.5 Algorithms 9
2.6 Security Function Implementations 14
2.7 Algorithm Specific Information 22
2.7.1 AES GCM IV 22
2.7.2 AES XTS 22
2.7.3 Key Derivation using SP 800-132 PBKDF2 22
2.7.4 SP 800-56Ar3 Assurances 23
2.7.5 SHA-3 23
2.7.6 RSA Signatures 23
2.7.7 Key Wrapping 23
2.8 RBG and Entropy 23
2.9 Key Generation 24
2.10 Key Establishment 24
2.11 Industry Protocols 24
3 Cryptographic Module Interfaces 26
3.1 Ports and Interfaces 26
4 Roles, Services, and Authentication 27
4.1 Authentication Methods 27
4.2 Roles 27
4.3 Approved Services 27
4.4 Non-Approved Services 34
5 Software/Firmware Security 36
5.1 Integrity Techniques 36
5.2 Initiate on Demand 36
6 Operational Environment 37
6.1 Operational Environment Type and Requirements 37
6.2 Configuration Settings and Restrictions 37
7 Physical Security 38
8 Non-Invasive Security 39
9 Sensitive Security Parameters Management 40
9.1 Storage Areas 40

9.2 SSP Input-Output Methods 40
9.3 SSP Zeroization Methods 40
9.4 SSPs 41
9.5 Transitions 47
10 Self-Tests 48
10.1 Pre-Operational Self-Tests 48
10.2 Conditional Self-Tests 48
10.3 Periodic Self-Test Information 58
10.4 Error States 64
10.5 Operator Initiation of Self-Tests 64
11 Life-Cycle Assurance 65
11.1 Installation, Initialization, and Startup Procedures 65
11.2 Administrator Guidance 65
11.3 Non-Administrator Guidance 65
11.6 End of Life 65
12 Mitigation of Other Attacks 66
12.1 Attack List 66
Appendix A. Glossary and Abbreviations 67
Appendix B. References 69

List of Tables

Table 1: Security Levels 6
Table 2: Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets) 8
Table 3: Tested Operational Environments - Software, Firmware, Hybrid 9
Table 4: Modes List and Description 9
Table 5: Approved Algorithms 13
Table 6: Vendor-Affirmed Algorithms 13
Table 7: Non-Approved, Not Allowed Algorithms 14
Table 8: Security Function Implementations 22
Table 9: Entropy Certificates 23
Table 10: Entropy Sources 24
Table 11: Ports and Interfaces 26
Table 12: Roles 27
Table 13: Approved Services 34
Table 14: Non-Approved Services 35
Table 15: Storage Areas 40
Table 16: SSP Input-Output Methods 40
Table 17: SSP Zeroization Methods 41
Table 18: SSP Table 1 44
Table 19: SSP Table 2 47
Table 20: Pre-Operational Self-Tests 48
Table 21: Conditional Self-Tests 58
Table 22: Pre-Operational Periodic Information 59
Table 23: Conditional Periodic Information 63
Table 24: Error States 64

List of Figures

Figure 1: Block Diagram 8

1 General

1.1 Overview

This document is the non-proprietary FIPS 140-3 Security Policy for version 3.0.8- d694bfa693b76001 of the Amazon Linux 2023 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.

1.1.1 How this Security Policy was Prepared

The vendor has provided the non-proprietary Security Policy of the cryptographic module, which was further consolidated into this document by atsec information security together with other vendor-supplied documentation. 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.

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

2 Cryptographic Module Specification

2.1 Description

Purpose and Use:

The Amazon Linux 2023 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”, which implements the FIPS requirements and the cryptographic functionality provided to the operator.

Module Type: Software

Module Embodiment: MultiChipStand

Cryptographic Boundary:

The cryptographic boundary of the module is defined as the fips.so shared library, which contains the compiled code implementing the FIPS provider.

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.

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. In this diagram, the physical perimeter of the operational environment (a general-purpose computer on which the module is installed) is indicated by a purple dashed line. The cryptographic boundary is represented by the components painted in orange blocks, which consists only of the shared library implementing the FIPS provider (fips.so).

Green lines indicate the flow of data between the cryptographic module and its operator application, through the logical interfaces defined in Section 3 Cryptographic Module Interfaces.

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). For example, the kernel is responsible for managing system calls issued by the module itself, as well as other applications using the module for cryptographic services.

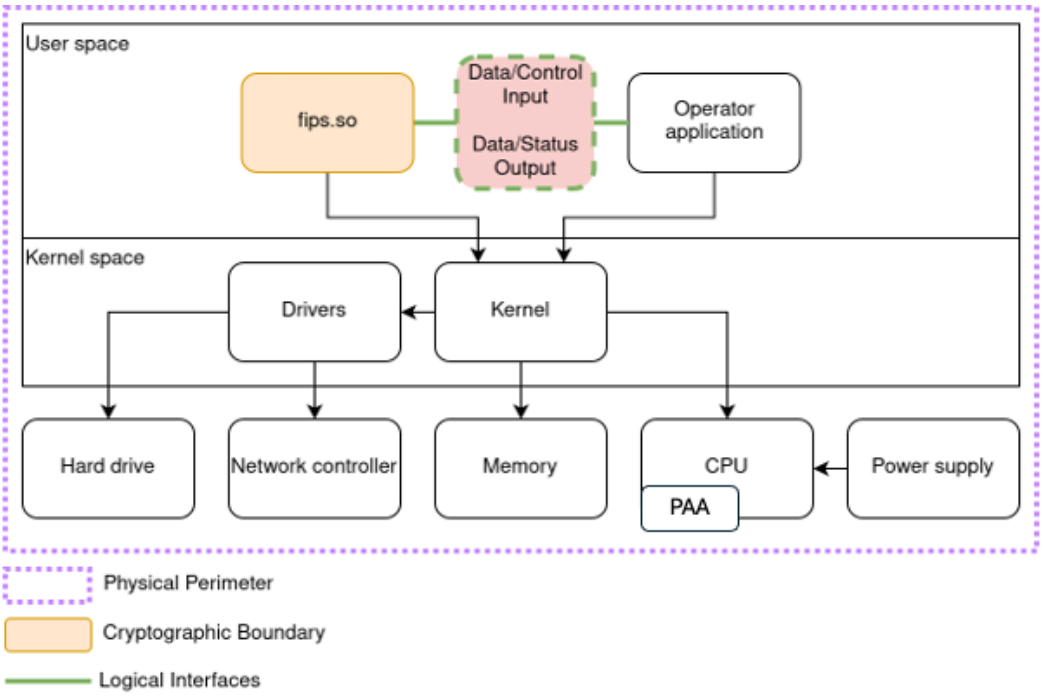

Figure 1: Block Diagram

2.2 Tested and Vendor Affirmed Module Version and Identification

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

Package or File Name	Software/ Firmware Version	Features	Integrity Test
fips.so on Amazon Linux 2023 with AWS Graviton3	3.0.8-d694bfa693b76001	N/A	HMAC-SHA-256
fips.so on Amazon Linux 2023 with Intel Xeon Platinum 8375C	3.0.8-d694bfa693b76001	N/A	HMAC-SHA-256
fips.so on Amazon Linux 2023 with AMD EPYC 7702	3.0.8-d694bfa693b76001	N/A	HMAC-SHA-256

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

Tested Operational Environments - Software, Firmware, Hybrid:

Operating System	Hardware Platform	Processors	PAA/PAI	Hypervisor or Host OS	Version(s)
Amazon Linux 2023	EC2 c7g.metal	AWS Graviton3	Yes	N/A	3.0.8-d694bfa693b76001
Amazon Linux 2023	EC2 c6i.metal	Intel Xeon Platinum 8375C	Yes	N/A	3.0.8-d694bfa693b76001
Amazon Linux 2023	AWS Snowball	AMD EPYC 7702	Yes	N/A	3.0.8-d694bfa693b76001
Amazon Linux 2023	EC2 c7g.metal	AWS Graviton3	No	N/A	3.0.8-d694bfa693b76001

Operating System	Hardware Platform	Processors	PAA/PAI	Hypervisor or Host OS	Version(s)
Amazon Linux 2023	EC2 c6i.metal	Intel Xeon Platinum 8375C	No	N/A	3.0.8-d694bfa693b76001
Amazon Linux 2023	AWS Snowball	AMD EPYC 7702	No	N/A	3.0.8-d694bfa693b76001

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

2.3 Excluded Components

There are no components excluded from the requirements of the FIPS 140-3 standard.

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 (specified in Section 4.3) of the requested service
Non-approved mode	Automatically entered whenever a non-approved service is requested	Non-Approved	Equivalent to the indicator (specified in Section 4.3) of the requested service

Table 4: Modes List and Description

After passing all pre-operational self-tests and cryptographic algorithm self-tests executed on start-up, the module automatically transitions to the approved mode. No operator intervention is required to reach this point.

In the operational state, the module accepts service requests from calling applications through its logical interfaces. At any point in the operational state, a calling application can end its process, causing the module to end its operation.

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	A4605, A4606, A4607, A4609, A4610, A4611	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS1	A4605, A4606, A4607, A4609, A4610, A4611	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS2	A4605, A4606, A4607, A4609, A4610, A4611	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS3	A4605, A4606, A4607, A4609, A4610, A4611	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CCM	A4605, A4606, A4607, A4609, A4610, A4611	Key Length - 128, 192, 256	SP 800-38C
AES-CFB1	A4605, A4606, A4607, A4609, A4610, A4611	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CFB128	A4605, A4606, A4607, A4609, A4610, A4611	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A

Algorithm	CAVP Cert	Properties	Reference
AES-CFB8	A4605, A4606, A4607, A4609, A4610, A4611	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CMAC	A4605, A4606, A4607, A4609, A4610, A4611	Direction - Generation, Verification Key Length - 128, 192, 256	SP 800-38B
AES-CTR	A4605, A4606, A4607, A4609, A4610, A4611	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-ECB	A4605, A4606, A4607, A4609, A4610, A4611, A4635, A4636, A4637, A4638, A4639	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-GCM	A4614, A4615, A4616, A4617, A4620, A4621, A4622, A4623, A4624, A4625, A4626, A4627, A4628	Direction - Decrypt, Encrypt IV Generation - External, Internal IV Generation Mode - 8.2.1, 8.2.2 Key Length - 128, 192, 256	SP 800-38D
AES-GMAC	A4614, A4615, A4616, A4617, A4620, A4621, A4622, A4623, A4624, A4625, A4626, A4627, A4628	Direction - Decrypt, Encrypt IV Generation - External IV Generation Mode - 8.2.1 Key Length - 128, 192, 256	SP 800-38D
AES-KW	A4605, A4606, A4607, A4609, A4610, A4611	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-KWP	A4605, A4606, A4607, A4609, A4610, A4611	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-OFB	A4605, A4606, A4607, A4609, A4610, A4611	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-XTS Testing Revision 2.0	A4605, A4606, A4607, A4609, A4610, A4611	Direction - Decrypt, Encrypt Key Length - 128, 256	SP 800-38E
Counter DRBG	A4604	Prediction Resistance - No, Yes Mode - AES-128, AES-192, AES-256 Derivation Function Enabled - No, Yes	SP 800-90A Rev. 1
ECDSA KeyGen (FIPS186-5)	A4612, A4618, A4629, A4630, A4631, A4632	Curve - P-224, P-256, P-384, P-521 Secret Generation Mode - testing candidates	FIPS 186-5
ECDSA KeyVer (FIPS186-5)	A4612, A4618, A4629, A4630, A4631, A4632	Curve - P-224, P-256, P-384, P-521	FIPS 186-5
ECDSA SigGen (FIPS186-5)	A4612, A4618, A4629, A4630, A4631, A4632	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	FIPS 186-5
ECDSA SigGen (FIPS186-5)	A4613, A4619	Curve - P-224, P-256, P-384, P-521 Hash Algorithm - SHA3-224, SHA3-256, SHA3-384, SHA3-512 Component - No	FIPS 186-5
ECDSA SigVer (FIPS186-5)	A4612, A4618, A4629, A4630, A4631, A4632	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)	A4613, A4619	Curve - P-224, P-256, P-384, P-521 Hash Algorithm - SHA3-224, SHA3-256, SHA3-384, SHA3-512	FIPS 186-5
Hash DRBG	A4604	Prediction Resistance - No, Yes Mode - SHA-1, SHA2-256, SHA2-512	SP 800-90A Rev. 1
HMAC DRBG	A4604	Prediction Resistance - No, Yes Mode - SHA-1, SHA2-256, SHA2-512	SP 800-90A Rev. 1
HMAC-SHA-1	A4612, A4618, A4629, A4630, A4631, A4632	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1

Algorithm	CAVP Cert	Properties	Reference
HMAC-SHA2-224	A4612, A4618, A4629, A4630, A4631, A4632	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA2-256	A4608, A4612, A4618, A4629, A4630, A4631, A4632	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA2-384	A4612, A4618, A4629, A4630, A4631, A4632	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512	A4612, A4618, A4629, A4630, A4631, A4632	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512/224	A4612, A4618, A4629, A4630, A4631, A4632	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512/256	A4612, A4618, A4629, A4630, A4631, A4632	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA3-224	A4613, A4619	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA3-256	A4613, A4619	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA3-384	A4613, A4619	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA3-512	A4613, A4619	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
KAS-ECC-SSC Sp800-56Ar3	A4612, A4618, A4629, A4630, A4631, A4632	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	A4642	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
KDA HKDF Sp800-56Cr1	A4603	Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-2048 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	SP 800-56C Rev. 2
KDA OneStep SP800-56Cr2	A4641	Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-2048 Increment 8	SP 800-56C Rev. 2
KDA TwoStep SP800-56Cr2	A4641	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)	A4612, A4618, A4629, A4630, A4631, A4632	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: 8-4096 Increment 8	SP 800-135 Rev. 1
KDF ANS 9.42 (CVL)	A4613, A4619	KDF Type - DER Hash Algorithm - SHA3-224, SHA3-256, SHA3-384, SHA3-512 Key Data Length - Key Data Length: 8-4096 Increment 8	SP 800-135 Rev. 1

Algorithm	CAVP Cert	Properties	Reference
KDF ANS 9.63 (CVL)	A4612, A4618, A4629, A4630, A4631, A4632	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)	A4613, A4619	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	A4608, A4640	KDF Mode - Counter, Feedback Supported Lengths - Supported Lengths: 8, 72, 128, 776, 3456, 4096	SP 800-108 Rev. 1
KDF SSH (CVL)	A4635, A4636, A4637, A4638, A4639	Cipher - AES-128, AES-192, AES-256 Hash Algorithm - SHA-1, SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
PBKDF	A4612, A4613, A4618, A4619, A4629, A4630, A4631, A4632	Iteration Count - Iteration Count: 1000-10000 Increment 1 Password Length - Password Length: 8-128 Increment 1	SP 800-132
RSA KeyGen (FIPS186-5)	A4612, A4618, A4629, A4630, A4631, A4632	Key Generation Mode - probableWithProbableAux Modulo - 2048, 3072, 4096 Primality Tests - 2powSecStr Private Key Format - standard	FIPS 186-5
RSA SigGen (FIPS186-5)	A4612, A4613, A4618, A4619, A4629, A4630, A4631, A4632	Modulo - 2048, 3072, 4096 Signature Type - pkcs1v1.5, pss	FIPS 186-5
RSA SigVer (FIPS186-4)	A4612, A4618, A4629, A4630, A4631, A4632	Signature Type - PKCS 1.5, PKCSPSS Modulo - 1024	FIPS 186-4
RSA SigVer (FIPS186-5)	A4612, A4613, A4618, A4619, A4629, A4630, A4631, A4632	Modulo - 2048, 3072, 4096 Signature Type - pkcs1v1.5, pss	FIPS 186-5
Safe Primes Key Generation	A4642	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	A4642	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	A4612, A4618, A4629, A4630, A4631, A4632	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-224	A4612, A4618, A4629, A4630, A4631, A4632	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-256	A4608, A4612, A4618, A4629, A4630, A4631, A4632	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-384	A4612, A4618, A4629, A4630, A4631, A4632	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-512	A4612, A4618, A4629, A4630, A4631, A4632	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-512/224	A4612, A4618, A4629, A4630, A4631, A4632	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-512/256	A4612, A4618, A4629, A4630, A4631, A4632	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4

Algorithm	CAVP Cert	Properties	Reference
SHA3-224	A4613, A4619	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHA3-256	A4613, A4619	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHA3-384	A4613, A4619	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHA3-512	A4613, A4619	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHAKE-128	A4613, A4619	Output Length - Output Length: 16-65536 Increment 8	FIPS 202
SHAKE-256	A4613, A4619	Output Length - Output Length: 16-65536 Increment 8	FIPS 202
TLS v1.2 KDF RFC7627 (CVL)	A4612, A4618, A4629, A4630, A4631, A4632	Hash Algorithm - SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
TLS v1.3 KDF (CVL)	A4603	HMAC Algorithm - SHA2-256, SHA2-384 KDF Running Modes - DHE, PSK, PSK-DHE	SP 800-135 Rev. 1

Table 5: Approved Algorithms

The table above lists all approved cryptographic algorithms of the module, including specific key lengths employed for approved services (see Approved Services table in Section 4.3 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 800-133Rev2 section 4, example 1

Table 6: Vendor-Affirmed Algorithms

Non-Approved, Allowed Algorithms:

N/A for this module.

The module does not implement non-approved algorithms that are allowed in the approved mode of operation.

Non-Approved, Allowed Algorithms with No Security Claimed:

N/A for this module.

The module does not implement non-approved algorithms that are allowed in the approved mode of operation with no security claimed.

Non-Approved, Not Allowed Algorithms:

Name	Use and Function
AES GCM (external IV)	Authenticated encryption
HMAC (< 112-bit keys)	Message authentication
KBKDF, KDA OneStep, KDA TwoStep, HKDF, ANS X9.42 KDF, ANS X9.63 KDF (< 112-bit keys)	Key derivation
KDA OneStep, KDA TwoStep (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

Name	Use and Function
TLS 1.2 KDF (SHA-1, SHA-224, SHA-512/224, SHA-512/256, SHA-3)	Key derivation
TLS 1.3 KDF (SHA-1, SHA-224, SHA-512, SHA-512/224, SHA-512/256, SHA-3)	Key derivation
PBKDF2 (< 8 characters password; < 128 salt length; < 1000 iterations; < 112-bit keys)	Password-based key derivation
RSA (KAS1, KAS2 schemes)	Shared secret computation
RSA and ECDSA (pre-hashed message)	Signature generation; Signature verification
RSA-PSS (invalid salt length)	Signature generation; Signature verification
RSA-OAEP	Asymmetric encryption; Asymmetric decryption

Table 7: Non-Approved, Not Allowed Algorithms

The table above lists all non-approved cryptographic algorithms of the module employed by the non-approved services of the Non-Approved Services table in Section 4.4 Non-Approved Services.

2.6 Security Function Implementations

Name	Type	Description	Properties	Algorithms
Encryption with AES	BC-UnAuth	Encryption using AES	Key:128, 192, 256 bit keys with 128, 192, 256 bits of key strength, respectively	AES-ECB: (A4605, A4606, A4607, A4609, A4610, A4611, A4635, A4636, A4637, A4638, A4639) AES-CBC: (A4605, A4606, A4607, A4609, A4610, A4611) AES-CBC-CS1: (A4605, A4606, A4607, A4609, A4610, A4611) AES-CBC-CS2: (A4605, A4606, A4607, A4609, A4610, A4611) AES-CBC-CS3: (A4605, A4606, A4607, A4609, A4610, A4611) AES-CFB1: (A4605, A4606, A4607, A4609, A4610, A4611) AES-CFB8: (A4605, A4606, A4607, A4609, A4610, A4611) AES-CFB128: (A4605, A4606, A4607, A4609, A4610, A4611) AES-CTR: (A4605, A4606, A4607, A4609, A4610, A4611)

Name	Type	Description	Properties	Algorithms
				AES-OFB: (A4605, A4606, A4607, A4609, A4610, A4611) AES-XTS Testing Revision 2.0: (A4605, A4606, A4607, A4609, A4610, A4611)
Decryption with AES	BC-UnAuth	Decryption using AES	Key:128, 192, 256 bit keys with 128, 192, 256 bits of key strength, respectively	AES-ECB: (A4605, A4606, A4607, A4609, A4610, A4611, A4635, A4636, A4637, A4638, A4639) AES-CBC: (A4605, A4606, A4607, A4609, A4610, A4611) AES-CBC-CS1: (A4605, A4606, A4607, A4609, A4610, A4611) AES-CBC-CS2: (A4605, A4606, A4607, A4609, A4610, A4611) AES-CBC-CS3: (A4605, A4606, A4607, A4609, A4610, A4611) AES-CFB1: (A4605, A4606, A4607, A4609, A4610, A4611) AES-CFB8: (A4605, A4606, A4607, A4609, A4610, A4611) AES-CFB128: (A4605, A4606, A4607, A4609, A4610, A4611) AES-CTR: (A4605, A4606, A4607, A4609, A4610, A4611) AES-OFB: (A4605, A4606, A4607, A4609, A4610, A4611) AES-XTS Testing Revision 2.0: (A4605, A4606, A4607, A4609, A4610, A4611)
Authenticated Encryption with AES	BC-Auth	Authenticated encryption using AES	Key:128, 192, 256 bit keys with 128, 192, 256 bits of key strength, respectively	AES-CCM: (A4605, A4606, A4607, A4609, A4610, A4611)

Name	Type	Description	Properties	Algorithms
				AES-KW: (A4605, A4606, A4607, A4609, A4610, A4611) AES-KWP: (A4605, A4606, A4607, A4609, A4610, A4611) AES-GCM: (A4614, A4615, A4616, A4617, A4620, A4621, A4622, A4623, A4624, A4625, A4626, A4627, A4628)
Authenticated Decryption with AES	BC-Auth	Authenticated decryption using AES	Key:128, 192, 256 bit keys with 128, 192, 256 bits of key strength, respectively	AES-CCM: (A4605, A4606, A4607, A4609, A4610, A4611) AES-KW: (A4605, A4606, A4607, A4609, A4610, A4611) AES-KWP: (A4605, A4606, A4607, A4609, A4610, A4611) AES-GCM: (A4614, A4615, A4616, A4617, A4620, A4621, A4622, A4623, A4624, A4625, A4626, A4627, A4628)
Key wrapping using AES KW	KTS-Wrap	Key wrapping using AES KW	Security strength:128, 192, 256 bits	AES-KW: (A4605, A4606, A4607, A4609, A4610, A4611)
Key unwrapping using AES KW	KTS-Wrap	Key unwrapping using AES KW	Security strength:128, 192, 256 bits	AES-KW: (A4605, A4606, A4607, A4609, A4610, A4611)
Key wrapping using AES KWP	KTS-Wrap	Key wrapping using AES KW with padding	Security strength:128, 192, 256 bits	AES-KWP: (A4605, A4606, A4607, A4609, A4610, A4611)
Key unwrapping using AES KWP	KTS-Wrap	Key unwrapping using AES KW with padding	Security strength:128, 192, 256 bits	AES-KWP: (A4605, A4606, A4607, A4609, A4610, A4611)
Key wrapping using AES CCM	KTS-Wrap	Key wrapping using AES CCM	Security strength:128, 192, 256 bits	AES-CCM: (A4605, A4606, A4607, A4609, A4610, A4611)
Key unwrapping using AES CCM	KTS-Wrap	Key unwrapping using AES CCM	Security strength:128, 192, 256 bits	AES-CCM: (A4605, A4606, A4607, A4609, A4610, A4611)
Key wrapping using AES GCM	KTS-Wrap	Key wrapping using AES GCM	Key:128, 192, 256 bit keys with 128, 192,	AES-GCM: (A4614, A4615, A4616,

Name	Type	Description	Properties	Algorithms
			256 bits of key strength, respectively IV method:Internally generated	A4617, A4620, A4621, A4622, A4623, A4624, A4625, A4626, A4627, A4628)
Key unwrapping using AES GCM	KTS-Wrap	Key unwrapping using AES GCM	Key:128, 192, 256 bit keys with 128, 192, 256 bits of key strength, respectively IV method:Provided externally	AES-GCM: (A4614, A4615, A4616, A4617, A4620, A4621, A4622, A4623, A4624, A4625, A4626, A4627, A4628)
Shared secret computation using Diffie-Hellman	KAS-SSC	Compute shared secret using DH	Security strength:112-220 bits Compliance:SP 800-56Arev3, FIPS 140-3 IG D.F. Scenario 2 (1) Scheme:dpEphem Roles:initiator, responder Groups:MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192, ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192	KAS-FFC-SSC Sp800-56Ar3: (A4642)
Shared secret computation using EC Diffie-Hellman	KAS-SSC	Compute shared secret using ECDH	Security strength:112, 128, 192, 256 bits Compliance:SP 800 56Arev3, FIPS 140-3 IG D.F. Scenario 2 (1) Scheme:Ephemeral Unified Model Roles:initiator, responder Curves:P-224, P-256, P-384, P-521	KAS-ECC-SSC Sp800-56Ar3: (A4612, A4618, A4629, A4630, A4631, A4632)
Hashing	SHA	Compute message digest using SHA		SHA-1: (A4612, A4618, A4629, A4630, A4631, A4632) SHA2-224: (A4612, A4618, A4629, A4630, A4631, A4632) SHA2-256: (A4608, A4612, A4618, A4629, A4630, A4631, A4632) SHA2-384: (A4612, A4618, A4629, A4630, A4631, A4632) SHA2-512: (A4612, A4618, A4629, A4630, A4631, A4632) SHA2-512/224: (A4612, A4618,

Name	Type	Description	Properties	Algorithms
				A4629, A4630, A4631, A4632) SHA2-512/256: (A4612, A4618, A4629, A4630, A4631, A4632) SHA3-224: (A4613, A4619) SHA3-256: (A4613, A4619) SHA3-384: (A4613, A4619) SHA3-512: (A4613, A4619)
Message authentication	MAC	Compute MAC tags using HMAC or AES-based CMAC, GMAC	HMAC hashes:SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512 AES key:128, 192, 256 bits	AES-CMAC: (A4605, A4606, A4607, A4609, A4610, A4611) AES-GMAC: (A4614, A4615, A4616, A4617, A4620, A4621, A4622, A4623, A4624, A4625, A4626, A4627, A4628) HMAC-SHA-1: (A4612, A4618, A4629, A4630, A4631, A4632) HMAC-SHA2-224: (A4612, A4618, A4629, A4630, A4631, A4632) HMAC-SHA2-256: (A4608, A4612, A4618, A4629, A4630, A4631, A4632) HMAC-SHA2-384: (A4612, A4618, A4629, A4630, A4631, A4632) HMAC-SHA2-512: (A4612, A4618, A4629, A4630, A4631, A4632) HMAC-SHA2-512/224: (A4612, A4618, A4629, A4630, A4631, A4632) HMAC-SHA2-512/256: (A4612, A4618, A4629, A4630, A4631, A4632) HMAC-SHA3-224: (A4613, A4619) HMAC-SHA3-256:

Name	Type	Description	Properties	Algorithms
				(A4613, A4619) HMAC-SHA3-384: (A4613, A4619) HMAC-SHA3-512: (A4613, A4619)
Key Pair Generation with RSA	AsymKeyPair-KeyGen	Generate a key pair for RSA	Mode:A.1.6 Probable Primes Based on Auxiliary Probable Primes Modulus:2048-16384 bits	RSA KeyGen (FIPS186-5): (A4612, A4618, A4629, A4630, A4631, A4632)
Key Pair Generation with ECDSA	AsymKeyPair-KeyGen	Generate a key pair for ECDSA	Mode:A.2.2 Rejection Sampling Curves:P-224, P-256, P-384, P-521	ECDSA KeyGen (FIPS186-5): (A4612, A4618, A4629, A4630, A4631, A4632)
Public Key Verification with ECDSA	AsymKeyPair-KeyVer	Verify public key for ECDSA	Mode:A.2.2 Rejection Sampling Curves:P-224, P-256, P-384. P-521	ECDSA KeyVer (FIPS186-5): (A4612, A4618, A4629, A4630, A4631, A4632)
Signature Generation with RSA	DigSig-SigGen	Generate a signature using RSA	Padding:PKCS#1 v1.5, PSS Hashes:SHA-224, SHA- 256, SHA-384, SHA-512, SHA-512/224, SHA-512/256, SHA3-224, SHA3- 256, SHA3-384, SHA3-512 Modulus:2048-16384 bits	RSA SigGen (FIPS186-5): (A4612, A4613, A4618, A4619, A4629, A4630, A4631, A4632)
Signature Verification with RSA	DigSig-SigVer	Verify a signature using RSA	Padding:PKCS#1 v1.5, PSS Hashes:SHA-224, SHA- 256, SHA-384, SHA-512, SHA-512/224, SHA-512/256, SHA3-224, SHA3- 256, SHA3-384, SHA3-512 Modulus:1024-16384 bits	RSA SigVer (FIPS186-5): (A4612, A4613, A4618, A4619, A4629, A4630, A4631, A4632) RSA SigVer (FIPS186-4): (A4612, A4618, A4629, A4630, A4631, A4632)
Signature Generation with ECDSA	DigSig-SigGen	Generate a signature using ECDSA	Curves:P-224, P-256, P-384, P-521 Hashes:SHA-224, SHA- 256, SHA-384, SHA-512, SHA-512/224, SHA-512/256, SHA3-224, SHA3- 256, SHA3-384, SHA3-512	ECDSA SigGen (FIPS186-5): (A4612, A4613, A4618, A4619, A4629, A4630, A4631, A4632)
Signature Verification with ECDSA	DigSig-SigVer	Verify a signature using ECDSA	Curves:P-224, P-256, P-384, P-521 Hashes:SHA-224, SHA- 256, SHA-384, SHA-512, SHA-512/224, SHA-512/256, SHA3-224,	ECDSA SigVer (FIPS186-5): (A4612, A4613, A4618, A4619, A4629, A4630, A4631, A4632)

Name	Type	Description	Properties	Algorithms
			SHA3- 256, SHA3-384, SHA3-512	
Extendable Output Function	XOF	Compute message digests from XOFs		SHAKE-128: (A4613, A4619) SHAKE-256: (A4613, A4619)
Random Number Generation with DRBG	DRBG	Generate random numbers using DRBGs	CTR_DRBG:AES-128, AES-192, AES-256, with/without derivation function, with/without prediction resistance Hash_DRBG:SHA-1, SHA-256, SHA-512, with/without prediction resistance HMAC_DRBG:SHA-1, SHA-256, SHA-512, with/without prediction resistance Compliance:Compliant with SP 800-90Arev1	Counter DRBG: (A4604) Hash DRBG: (A4604) HMAC DRBG: (A4604)
Key derivation with KBKDF	KBKDF	Derive keys from a key materials	Modes:Counter, Feedback MACs:CMAC and HMAC SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	KDF SP800-108: (A4608, A4640)
Key derivation with HKDF	KAS-56CKDF	Derive keys using HKDF	Mode:Feedback MACs:HMAC SHA-1, SHA-224, SHA-256, SHA-384, SHA- 512, SHA-512/224, SHA-512/256, SHA3-224, SHA3- 256, SHA3-384, SHA3-512	KDA HKDF Sp800-56Cr1: (A4603)
Key derivation with TLS 1.2 KDF	KAS-135KDF	Derive keys from TLS 1.2 KDF	Hashes:SHA-256, SHA- 384, SHA-512 Support:Extended Master Secret	TLS v1.2 KDF RFC7627: (A4612, A4618, A4629, A4630, A4631, A4632)
Key derivation with TLS 1.3 KDF		Derive keys from TLS 1.3 KDF	Modes:DHE, PSK, PSK-DHE Hashes:SHA-256, SHA-384	TLS v1.3 KDF: (A4603)
Key derivation with SSH KDF	KAS-135KDF	Derive keys from SSH KDF	Ciphers:AES-128, AES-192, AES-256 Hashes:SHA-1, SHA-224, SHA-256, SHA-384, SHA-512	KDF SSH: (A4635, A4636, A4637, A4638, A4639)
Key derivation with X9.63 KDF	KAS-135KDF	Derive keys from ANS X9.63 KDF	Hashes:SHA-224, SHA- 256, SHA-384, SHA-512, SHA-512/224, SHA-512/256, SHA3-224,	KDF ANS 9.63: (A4612, A4613, A4618, A4619, A4629, A4630, A4631, A4632)

Name	Type	Description	Properties	Algorithms
			SHA3- 256, SHA3-384, SHA3-512	
Key derivation with X9.42 KDF	KAS-135KDF	Derive keys from ANS X9.42 KDF	Hashes:SHA-1, SHA-224, SHA-256, SHA-384, SHA- 512, SHA-512/224, SHA-512/256, SHA3-224, SHA3- 256, SHA3-384, SHA3-512 OID:AWS-KW 128, 192, 256	KDF ANS 9.42: (A4612, A4613, A4618, A4619, A4629, A4630, A4631, A4632)
Key derivation with PBKDF	PBKDF	Derive keys from PBKDF	Option:1a Password length:8-128 characters Salt length:128-4096 bytes Iteration count:1000-10000 Hashes:SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	PBKDF: (A4612, A4613, A4618, A4619, A4629, A4630, A4631, A4632)
Key Pair Generation with Safe Primes	AsymKeyPair-KeyGen	Generate a key pair from safe primes	Mode:Testing Candidates (SP 800-56Arev3 Appendix 5.6.1.1.4) Groups:MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192, ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192	Safe Primes Key Generation: (A4642)
Key Pair Verification with Safe Primes	AsymKeyPair-KeyVer	Verify a key pair using safe primes	Mode:Testing Candidates (SP 800-56Arev3 Appendix 5.6.1.1.4) Groups:MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192, ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192	Safe Primes Key Verification: (A4642)
Key derivation using KDA OneStep	KAS-56CKDF	Key derivation using KDA OneStep	MACs:(HMAC) SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	KDA OneStep SP800-56Cr2: (A4641)
Key derivation using KDA TwoStep	KAS-56CKDF	Key derivation using KDA TwoStep	Modes:Feedback MACs:HMAC SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256, SHA3-224,	KDA TwoStep SP800-56Cr2: (A4641)

Name	Type	Description	Properties	Algorithms
			SHA3-256, SHA3-384, SHA3-512	

Table 8: Security Function Implementations

2.7 Algorithm Specific Information

2.7.1 AES GCM IV

For TLS 1.2, the module offers the AES GCM implementation and uses the context of Scenario 1 of FIPS 140-3 IG C.H. OpenSSL 3 is compliant with SP 800-52r2 Section 3.3.1 and the mechanism for IV generation is compliant with RFC 5288 and 8446.

The module does not implement the TLS protocol. The module's implementation of AES GCM is used together with an application that runs outside the module's cryptographic boundary. The design of the TLS protocol implicitly ensures that the counter (the nonce_explicit part of the IV) does not exhaust the maximum number of possible values for a given session key. If the counter exhaustion condition is observed, the module returns an error indication to the calling application, which will then need to either abort the connection, or trigger a handshake to establish a new encryption key.

In the event the module's power is lost and restored, the consuming application must ensure that a new key for use with the AES GCM key encryption or decryption under this scenario shall be established.

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.

The module also provides a non-approved AES GCM encryption service which accepts arbitrary external IVs from the operator. This service can be requested by invoking the EVP_EncryptInit_ex2 API function with a non-NULL iv value. When this is the case, the API will set a non-approved service indicator.

Finally, for TLS 1.3, the AES GCM implementation uses the context of Scenario 5 of FIPS 140-3 IG C.H. 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 SP800-52r2. TLS 1.3 employs separate 64-bit sequence numbers, one for protocol records that are received, and one for protocol records that are sent to a peer. These sequence numbers are set at zero at the beginning of a TLS 1.3 connection and each time when the AES-GCM key is changed. After reading or writing a record, the respective sequence number is incremented by one. The protocol specification determines that the sequence number should not wrap, and if this condition is observed, then the protocol implementation must either trigger a re-key of the session (i.e., a new key for AES-GCM), or terminate the connection.

2.7.2 AES XTS

The length of a single data unit encrypted or decrypted with AES XTS shall not exceed 2^{20} AES blocks, that is 16MB, of data per XTS instance. An XTS instance is defined in Section 4 of SP 800-38E.

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.

The XTS mode shall only be used for the cryptographic protection of data on storage devices. It shall not be used for other purposes, such as the encryption of data in transit.

2.7.3 Key Derivation using SP 800-132 PBKDF2

The module provides password-based key derivation (PBKDF2), compliant with SP 800-132. The module supports option 1a from Section 5.4 of SP 800-132, in which the Master Key (MK) or a segment of it is used directly as the Data Protection Key (DPK). In accordance to SP 800-132 and FIPS 140-3 IG D.N, the following requirements are met:

- Derived keys shall be used only for storage applications, and shall not be used for any other purposes. The length of the MK or DPK is 112 bits or more (this is verified by the module).
- Passwords or passphrases, used as an input for the PBKDF2, shall not be used as cryptographic keys.
- The length of the password or passphrase is at least 8 characters (this is verified by the module), and may consist of lowercase, uppercase, and numeric characters. Assuming the worst-case scenario of all digits, the probability is estimated to be at most 10^{-8}. Combined with the minimum iteration count as described in the fifth bullet point, this provides an acceptable trade-off between user experience and security against brute-force attacks.
- A portion of the salt shall be generated randomly using the SP 800-90Ar1 DRBG provided by the module. The minimum length required is 128 bits (this is verified by the module).
- The iteration count is 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 is 1000 (this is verified by the module).

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 800-56Ar3 Assurances

To comply with the assurances found in Section 5.6.2 of SP 800-56Ar3, the operator must use the module together with an application that implements the TLS protocol. Additionally, the module's approved key pair generation service (see Approved Services table in Section 4.3 Approved Services) must be used to generate ephemeral Diffie-Hellman or EC Diffie-Hellman 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 full public key validation of the generated public key.

The module's shared secret computation service will internally perform the full public key validation of the peer public key, complying with Sections 5.6.2.2.1 and 5.6.2.2.2 of SP 800-56Ar3.

2.7.5 SHA-3

To meet the requirement stated in IG C.C, the module implements the SHA-3 algorithms as both standalone and part of higher-level algorithms. 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, HKDF, X9.63 KDF, X9.42 KDF, PBKDF, OneStep KDA, TwoStep 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 Signatures

To meet the requirement stated in IG C.F, the module implements only the approved modulus sizes of 2048, 3072, and 4096 bits for signature generation. For signature verification, the module implements only the approved module sizes of 1024, 2048, 3072, and 4096 bits. Each algorithm was tested, and corresponding certificates can be found detailed in Section 2.6 Security Function Implementations.

2.7.7 Key Wrapping

To meet the requirement stated in IG D.G, the module implements AES KW, KWP, GCM, and CCM as approved key wrapping algorithms. In addition, AES KW and AES KWP meets the requirements of SP 800-38F. Each approve key wrapping algorithm was tested, and corresponding certificates can be found detailed in Section 2.6 Security Function Implementations.

2.8 RBG and Entropy

Cert Number	Vendor Name
E125	Amazon Web Services, Inc.

Table 9: Entropy Certificates

Name	Type	Operational Environment	Sample Size	Entropy per Sample	Conditioning Component
Amazon OpenSSL CPU Time Jitter RNG Entropy Source	Non-Physical	Amazon Linux 2023 on EC2 c7g.metal; Amazon Linux 2023 on EC2 c6i.metal; Amazon Linux 2023 on AWS Snowball	256 bits	256 bits	SHA-3 (A4551); HMAC-SHA-512 DRBG (A4551); AES-256 CTR DRBG (A4604)

Table 10: Entropy Sources

The module employs two Deterministic Random Bit Generator (DRBG) implementations based on SP 800-90Ar1. These DRBGs are used internally by the module (e.g. to generate seeds for asymmetric key pairs and random numbers for security functions). They can also be accessed using the specified API functions. The following parameters are used:

1. Private DRBG: AES-256 CTR_DRBG with derivation function. This DRBG is used to generate secret random values (e.g. during asymmetric key pair generation). It can be accessed using RAND_priv_bytes .
2. Public DRBG: AES-256 CTR_DRBG with derivation function. This DRBG is used to generate general purpose random values that do not need to remain secret (e.g. initialization vectors). It can be accessed using RAND_bytes .

These DRBGs will always employ prediction resistance. More information regarding the configuration and design of these DRBGs can be found in the module's manual pages.

2.9 Key Generation

The module implements Cryptographic Key Generation (CKG, vendor affirmed), compliant with SP 800-133r2. When random values are required, they are obtained from the SP 800-90Ar1 approved DRBG, compliant with Section 4 of SP 800-133r2 (without XOR).

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 SSP transport methods as listed in the table above. Additionally, the module implements the following key derivation methods:

- KBKDF: compliant with SP 800-108r1. This implementation can be used to generate secret keys from a pre-existing key-derivation-key.
- KDA OneStep, KDA TwoStep, HKDF: compliant with SP 800-56Cr1 (HKDF) and SP 800-56Cr2 (KDA OneStep, KDA TwoStep). These implementations shall only be used to generate secret keys in the context of an SP 800-56Ar3 key agreement scheme.
- ANS X9.42 KDF, ANS X9.63 KDF: compliant with SP 800-135r1. These implementations shall only be used to generate secret keys in the context of an ANS X9.42-2001 resp. ANS X9.63-2001 key agreement scheme.
- SSH KDF, TLS 1.2 KDF, TLS 1.3 KDF: compliant with SP 800-135r1. These implementations shall only be used to generate secret keys in the context of the SSH, TLS 1.2, or TLS 1.3 protocols, respectively.
- PBKDF2: compliant with option 1a of SP 800-132. This implementation shall only be used to derive keys for use in storage applications.

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.

For Diffie-Hellman, the module supports the use of the following safe primes:

- IKE (RFC 3526):
  - MODP-2048 (ID = 14)
  - MODP-3072 (ID = 15)
  - MODP-4096 (ID = 16)
  - MODP-6144 (ID = 17)
  - MODP-8192 (ID = 18)

- TLS (RFC 7919)
  - ffdhe2048 (ID = 256)
  - ffdhe3072 (ID = 257)
  - ffdhe4096 (ID = 258)
  - ffdhe6144 (ID = 259)
  - ffdhe8192 (ID = 260)

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 11: 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 12: Roles

The module supports the Crypto Officer role only. This sole role is implicitly and always assumed by the operator of the module when performing a service. The module does not support multiple concurrent operators.

4.3 Approved Services

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Message digest	Compute a message digest	EVP_DigestFinal_ex returns 1	Message	Digest value	Hashing	Crypto Officer
XOF	Compute output of XOF	EVP_DigestFinalXOF_ex returns 1	Message, output length	Digest value	Extendable Output Function	Crypto Officer
Encryption	Encrypt a plaintext	EVP_EncryptFinal_ex returns 1	Plaintext, AES key	Ciphertext	Encryption with AES	Crypto Officer - AES key: W,E
Decryption	Decrypt a plaintext	EVP_DecryptFinal_ex returns 1	Ciphertext, AES key	Plaintext	Decryption with AES	Crypto Officer - AES key: W,E
Authenticated Encryption	Encrypt and authenticate a plaintext	AES GCM: EVP_CIPHER_REDHAT_FIPS_INDICATOR_APPROVED; Others: EVP_EncryptFinal_ex returns 1	AES key, IV (only CCM and GCM), plaintext	Ciphertext, MAC tag (only CCM and GCM)	Authenticated Encryption with AES Key wrapping using AES KW Key wrapping using AES KWP Key wrapping using AES CCM Key wrapping using AES GCM	Crypto Officer - AES key: W,E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Authenticated Decryption	Decrypt and authenticate a ciphertext	AES GCM: EVP_CIPHER_REDHAT_FIPS_INDICATOR_APPROVED; Others: EVP_DecryptFinal_ex returns 1	AES key, ciphertext, MAC tag (only CCM and GCM), IV (only CCM and GCM)	Plaintext or failure	Authenticated Decryption with AES Key unwrapping using AES KW Key unwrapping using AES KWP Key unwrapping using AES CCM Key unwrapping using AES GCM	Crypto Officer - AES key: W,E
Message Authentication	Computes a MAC tag	HMAC: EVP_MAC_REDHAT_FIPS_INDICATOR_APPROVED; Others: EVP_MAC_final returns 1	Message, AES or HMAC key	MAC tag	Message authentication	Crypto Officer - AES key: W,E - HMAC key: W,E
Key derivation with KBKDF	Derive a key from a key-derivation key using KBKDF	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
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 - HKDF Derived key: G,R - Shared secret: W,E
Key derivation with TLS KDF	Derive a key from a shared secret using TLS KDF	EVP_KDF_REDHAT_FIPS_INDICATOR_APPROVED	Shared secret	TLS Derived key	Key derivation with TLS 1.2 KDF Key derivation with TLS 1.3 KDF	Crypto Officer - Shared secret: W,E - TLS 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

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
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 using X9.42 KDF	Derive a key from a shared secret using X9.42 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 derivation with KDA OneStep	Derive a key from a shared secret using KDA OneStep	EVP_KDF_REDHAT_FIPS_INDICATOR_APPROVED	Shared secret	KDA OneStep Derived key	Key derivation using KDA OneStep	Crypto Officer - KDA OneStep Derived key: G,R - Shared secret: W,E
Key derivation with KDA TwoStep	Derive a key from a shared secret using KDA OneStep	EVP_KDF_REDHAT_FIPS_INDICATOR_APPROVED	Shared secret	KDA TwoStep Derived key	Key derivation using KDA TwoStep	Crypto Officer - KDA TwoStep Derived key: G,R - Shared secret: W,E
Password-based key derivation	Derive a key from a password	EVP_KDF_REDHAT_FIPS_INDICATOR_APPROVED	Password, salt, iteration count	PBKDF Derived key	Key derivation with PBKDF	Crypto Officer - Password: W,E - PBKDF Derived key: G,R
Random number generation	Generate random bytes	EVP_RAND_generate returns 1	Seed, Output length	Random bytes	Random Number Generation with DRBG	Crypto Officer - Entropy input: W,E - DRBG seed: G,E - Internal state (V, Key): G,W,E - Internal state (V, C): G,W,E
Shared secret computation	Compute a shared secret	EVP_PKEY_derive returns 1	DH private key, DH public key; EC private key, EC public key	Shared secret	Shared secret computation using Diffie-Hellman Shared secret computati	Crypto Officer - Shared secret: G,R - DH private key: W,E - DH public key: W,E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
					on using EC Diffie-Hellman	- EC private key: W,E - EC public key: W,E
Signature generation	Generate a signature	RSA: OSSL_RH_FIPSINDICATOR_APPROVED and EVP_SIGNATURE_REDHAT_FIPS_INDICATOR_APPROVED; ECDSA: OSSL_RH_FIPSINDICATOR_APPROVED	Message, RSA or EC private key	Signature	Signature Generation with RSA Signature Generation with ECDSA	Crypto Officer - RSA private key: W,E - EC private key: W,E
Signature verification	Verify a sigantur e	RSA: OSSL_RH_FIPSINDICATOR_APPROVED and EVP_SIGNATURE_REDHAT_FIPS_INDICATOR_APPROVED; ECDSA: OSSL_RH_FIPSINDICATOR_APPROVED	Message, signature, RSA or EC public key	Pass/fail	Signature Verification with RSA Signature Verification with ECDSA	Crypto Officer - RSA public key: W,E - EC public key: W,E
Key pair generation	Generate a key pair	EVP_PKEY_generate returns 1	Group; Curve; Modulus bits	DH key pair; EC key pair; RSA key pair	Key Pair Generation with RSA Key Pair Generation with ECDSA Key Pair Generation with Safe Primes	Crypto Officer - DH private key: G,R - DH public key: G,R - RSA private key: G,R - RSA public key: G,R - EC private key: G,R - EC public key: G,R - Intermediate key generation value: G,E,Z
Public key verification	Verify an EC public key	EVP_PKEY_public_check or EVP_PKEY_private_check or EVP_PKEY_check returns 1	EC public key	Pass/fail	Public Key Verification with ECDSA	Crypto Officer - EC public key: W,E
Key pair verification	Verify a DH key pair	EVP_PKEY_public_check or EVP_PKEY_private_check or EVP_PKEY_check returns 1	DH public key; DH private key	Pass/fail	Key Pair Verification with Safe Primes	Crypto Officer - DH private key: W,E - DH public key: W,E
Show version	Return the name and version	None	N/A	Name and version information	None	Unauthenticated

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	information					
Show status	Return the module status	None	N/A	Module status	None	Unauthenticated
Self-tests	Perform CASTs and integrity test	None	N/A	Pass/fail results of self-tests	Encryption with AES Decryption with AES Authenticated Encryption with AES Authenticated Decryption with AES Shared secret computation using Diffie-Hellman Shared secret computation using EC Diffie-Hellman Hashing Message authentication Key Pair Generation with RSA Key Pair Generation with ECDSA Public Key Verification with ECDSA Signature Generation with RSA Signature Verification with RSA	Unauthenticated

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
					Signature Generation with ECDSA Signature Verification with ECDSA Random Number Generation with DRBG Key derivation with KBKDF Key derivation with HKDF Key derivation with TLS 1.2 KDF Key derivation with TLS 1.3 KDF Key derivation with SSH KDF Key derivation with X9.63 KDF Key derivation with X9.42 KDF Key derivation with PBKDF Key Pair Generation with Safe Primes Key derivation using KDA OneStep Key derivation using KDA TwoStep	

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Zeroization	Zeroize any SSP	None	Any SSP	None	None	Unauthenticated - AES key: Z - HMAC key: Z - Key-derivation key: Z - Shared secret: Z - Password: Z - KBKDF Derived key: Z - HKDF Derived key: Z - X9.63 Derived key: Z - X9.42 Derived key: Z - SSH Derived key: Z - KDA OneStep Derived key: Z - KDA TwoStep Derived key: Z - TLS Derived key: Z - PBKDF Derived key: Z - Entropy input: Z - DRBG seed: Z - Internal state (V, Key): Z - Internal state (V, C): Z - DH private key: Z - DH public key: Z - EC private

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						key: Z - EC public key: Z - RSA private key: Z - RSA public key: Z - Intermediate key generation value: Z

Table 13: 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.
- **N/A**: The module does not access any SSP or key during its operation.

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. After a cryptographic service was performed by the module, the API context (listed in the left column of the bullets below) associated with this request can contain a parameter (listed in the right column of the bullets below) which represents the approved service indicator:

- EVP_CIPHER_CTX: OSSL_CIPHER_PARAM_REDHAT_FIPS_INDICATOR
- EVP_MAC_CTX: OSSL_MAC_PARAM_REDHAT_FIPS_INDICATOR
- EVP_KDF_CTX: OSSL_KDF_PARAM_REDHAT_FIPS_INDICATOR
- EVP_PKEY_CTX: OSSL_KEM_PARAM_REDHAT_FIPS_INDICATOR
- EVP_PKEY_CTX: OSSL_SIGNATURE_PARAM_REDHAT_FIPS_INDICATOR
- EVP_PKEY_CTX: OSSL_ASYM_CIPHER_PARAM_REDHAT_FIPS_INDICATOR

The exact process to use these functions and parameters are described in the module's manual pages.

4.4 Non-Approved Services

Name	Description	Algorithms	Role
Encryption	Encrypt a plaintext	AES GCM (external IV)	CO
Message authentication	Compute a MAC tag	HMAC (< 112-bit keys)	CO
Key derivation	Derive a key from a key-derivation key or a shared secret	KBKDF, KDA OneStep, KDA TwoStep, HKDF, ANS X9.42 KDF, ANS X9.63 KDF (< 112-bit keys) KDA OneStep, KDA TwoStep (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-	CO

Name	Description	Algorithms	Role
		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)	
Password-based key derivation	Derive a key from a password	PBKDF2 (< 8 characters password; < 128 salt length; < 1000 iterations; < 112-bit keys)	CO
Shared secret computation	Compute a shared secret	RSA (KAS1, KAS2 schemes)	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
Asymmetric encryption	Encrypt a plaintext	RSA-OAEP	CO
Asymmetric decryption	Decrypt a ciphertext	RSA-OAEP	CO

Table 14: Non-Approved Services

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

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.

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 provided by the Linux kernel, 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 15: 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 16: SSP Input-Output Methods

9.3 SSP Zeroization Methods

Zeroization Method	Description	Rationale	Operator Initiation
Free cipher handle	Zeroizes the SSPs contained within the cipher handle	Memory occupied by SSPs is overwritten with zeroes, which renders the SSP values irretrievable. The completion of the zeroization routine indicates that the zeroization procedure succeeded.	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 Derived key: EVP_KDF_CTX_free; SSH Derived key: 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 seed: EVP_RAND_CTX_free; Internal state: 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
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
Remove power from the module	De-allocates the volatile memory	Volatile memory used by the module is overwritten within nanoseconds when	By unloading the module

Zeroization Method	Description	Rationale	Operator Initiation
	used to store SSPs	the module is unloaded. The successful completion of the removal of power from the module indicates that zeroization has completed.	

Table 17: SSP Zeroization Methods

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	XTS: 256, 512 bits; Other modes: 128, 192, 256 bits - XTS: 128, 256 bits; Other modes: 128, 192, 256 bits	Symmetric key - CSP			Encryption with AES Decryption with AES Key wrapping using AES KW Key unwrapping using AES KW Key wrapping using AES KWP Key unwrapping using AES KWP Key wrapping using AES CCM Key unwrapping using AES CCM Key wrapping using AES GCM Key unwrapping using AES GCM
HMAC key	HMAC key used for computing MAC tag	112-524288 bits - 112-256 bits	Symmetric key - CSP			Message authentication
Shared secret	Shared secret generated by (EC) Diffie-Hellman	224-8192 bits - 112-256 bits	Shared secret - CSP		Shared secret computation using Diffie-Hellman Shared secret computation using EC Diffie-Hellman	Shared secret computation using Diffie-Hellman Shared secret computation using EC Diffie-Hellman Key derivation using KDA OneStep Key derivation using KDA TwoStep

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
Key-derivation key	Symmetric key used to derive symmetric keys	112-4096 bits - 112-256 bits	Symmetric key - CSP			Key derivation with KBKDF
Password	Password used to derive symmetric keys	8-128 characters - N/A	Password - CSP			Key derivation with PBKDF
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
TLS Derived key	Symmetric key derived from a shared secret	112-4096 bits - 112-256 bits	Symmetric key - CSP	Key derivation with TLS 1.2 KDF Key derivation with TLS 1.3 KDF		Key derivation with TLS 1.2 KDF Key derivation with TLS 1.3 KDF
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
PBKDF Derived key	Symmetric key derived from a password	112-4096 bits - 112-256 bits	Symmetric key - CSP	Key derivation with PBKDF		Key derivation with PBKDF
KDA OneStep Derived key	Symmetric key derived from a shared secret	112-4096 bits - 112-256 bits	Symmetric key - CSP	Key derivation using KDA OneStep		Key derivation using KDA OneStep
KDA TwoStep Derived key	Symmetric key derived from a shared secret	112-4096 bits - 112-256 bits	Symmetric key - CSP	Key derivation using KDA TwoStep		Key derivation using KDA TwoStep
Entropy input	Entropy input used to seed the DRBGs	128-384 bits - 128-256 bits	Entropy input - CSP			Random Number Generation with DRBG
DRBG seed	DRBG seed derived from entropy input	CTR_DRBG: 256, 320, 384 bits; Hash_DRBG: 440, 888 bits;	Seed - CSP	Random Number Generation with DRBG		Random Number Generation with DRBG

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
		HMAC_DRBG: 160, 256, 512 bits - CTR_DRBG: 128, 192, 256 bits; Hash_DRBG: 128, 256 bits; HMAC_DRBG: 128, 256 bits				
Internal state (V, Key)	Internal state of CTR_DRBG and HMAC_DRBG instances	CTR_DRBG: 256, 320, 348 bits; HMAC_DRBG: 320, 512, 1024 bits - CTR_DRBG: 128, 192, 256 bits; HMAC_DRBG: 128, 256 bits	Internal state - CSP	Random Number Generation with DRBG		Random Number Generation with DRBG
Internal state (V, C)	Internal state of Hash_DRBG	880, 1776 bits - 128, 256 bits	Internal state - CSP	Random Number Generation with DRBG		Random Number Generation with DRBG
DH private key	Private key used for Diffie-Hellman	2048-8192 bits - 112-200 bits	Private key - CSP	Key Pair Generation with Safe Primes		Shared secret computation using Diffie-Hellman Key Pair Generation with Safe Primes Key Pair Verification with Safe Primes
DH public key	Public key used for Diffie-Hellman	2048-8192 bits - 112-200 bits	Public key - PSP	Key Pair Generation with Safe Primes		Shared secret computation using Diffie-Hellman Key Pair Generation with Safe Primes Key Pair Verification with Safe Primes
EC private key	Private key used for ECDH and ECDSA	P-224, P-256, P-384, P-521 - 112, 128, 192, 256 bits	Private key - CSP	Key Pair Generation with ECDSA		Shared secret computation using EC Diffie-Hellman Key Pair Generation with ECDSA Public Key Verification with ECDSA Signature Generation with ECDSA

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
EC public key	Public key used for ECDH and ECDSA	P-224, P-256, P-384, P-521 - 112, 128, 192, 256 bits	Public key - PSP	Key Pair Generation with ECDSA		Shared secret computation using EC Diffie-Hellman Key Pair Generation with ECDSA Public Key Verification with ECDSA Signature Verification with ECDSA
RSA private key	Private key used for RSA signature generation	2048-16384 bits - 112-256 bits	Private key - CSP	Key Pair Generation with RSA		Key Pair Generation with RSA Signature Generation with RSA
RSA public key	Public key used for RSA signature verification	Signature verification: 1024-16384 bits; Key pair generation: 2048-16384 bits - Signature verification: 80-256 bits; Key pair generation: 112-256 bits	Public key - PSP	Key Pair Generation with RSA		Key Pair Generation with RSA Signature Verification with RSA
Intermediate key generation value	Temporary value generated during key pair generation services	2048-16384 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 RSA Key Pair Generation with ECDSA Key Pair Generation with Safe Primes

Table 18: SSP Table 1

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
AES key	API input parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	
HMAC key	API input parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	
Shared secret	API input parameters API output parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	DH private key:Established By DH public key:Established By EC private

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
					key:Established By EC public key:Established By HKDF Derived key:Derives KDA OneStep Derived 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
Key-derivation key	API input parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	KBKDF Derived key:Derives
Password	API input parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	PBKDF Derived key:Derives
KBKDF Derived key	API output parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	Key-derivation key:Derived From
HKDF Derived key	API output parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	Shared secret:Derived From
TLS Derived key	API output parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	Shared secret:Derived From
SSH Derived key	API output parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	Shared secret:Derived From
X9.63 Derived key	API output parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	Shared secret:Derived From
X9.42 Derived key	API output parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	Shared secret:Derived From
PBKDF Derived key	API output parameters	RAM:Plaintext	For the duration of the service	Free cipher handle	Password:Derived From

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
				Remove power from the module	
KDA OneStep Derived key	API output parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	Shared secret:Derived From
KDA TwoStep Derived key	API output parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	Shared secret:Derived From
Entropy input		RAM:Plaintext	From generation until DRBG seed is created	Automatic Remove power from the module	DRBG seed:Derives
DRBG seed		RAM:Plaintext	While the DRBG is instantiated	Automatic Remove power from the module	Entropy input:Derived From Internal state (V, Key):Generates Internal state (V, C):Generates
Internal state (V, Key)		RAM:Plaintext	From DRBG instantiation until DRBG termination	Free cipher handle Remove power from the module	DRBG seed:Generated From
Internal state (V, C)		RAM:Plaintext	From DRBG instantiation until DRBG termination	Free cipher handle Remove power from the module	DRBG seed:Generated From
DH private key	API input parameters API output parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	DH public key:Paired With Intermediate key generation value:Generated From
DH public key	API input parameters API output parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	DH private key:Paired With Intermediate key generation value:Generated From
EC private key	API input parameters API output parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	EC public key:Paired With Intermediate key generation value:Generated From
EC public key	API input parameters API output parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	EC private key:Paired With Intermediate key generation value:Generated From
RSA private key	API input parameters API output parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	RSA public key:Paired With Intermediate key generation value:Generated From

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
RSA public key	API input parameters API output parameters	RAM:Plaintext	For the duration of the service	Free cipher handle Remove power from the module	RSA private key:Paired With Intermediate key generation value:Generated From
Intermediate key generation value		RAM:Plaintext	For the duration of the service	Automatic	DH private key:Generates DH public key:Generates EC private key:Generates EC public key:Generates RSA private key:Generates RSA public key:Generates

Table 19: 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 (A4608)	256-bit key	Message authentication	SW/FW Integrity	Module becomes operational	Integrity test for fips.so
HMAC-SHA2-256 (A4612)	256-bit key	Message authentication	SW/FW Integrity	Module becomes operational	Integrity test for fips.so
HMAC-SHA2-256 (A4618)	256-bit key	Message authentication	SW/FW Integrity	Module becomes operational	Integrity test for fips.so
HMAC-SHA2-256 (A4629)	256-bit key	Message authentication	SW/FW Integrity	Module becomes operational	Integrity test for fips.so
HMAC-SHA2-256 (A4630)	256-bit key	Message authentication	SW/FW Integrity	Module becomes operational	Integrity test for fips.so
HMAC-SHA2-256 (A4631)	256-bit key	Message authentication	SW/FW Integrity	Module becomes operational	Integrity test for fips.so
HMAC-SHA2-256 (A4632)	256-bit key	Message authentication	SW/FW Integrity	Module becomes operational	Integrity test for fips.so

Table 20: 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.

10.2 Conditional Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
SHA-1 (A4612)	24-bit message	KAT	CAST	Module is operational	Message digest	Module initialization (before integrity test)
SHA-1 (A4618)	24-bit message	KAT	CAST	Module is operational	Message digest	Module initialization (before integrity test)
SHA-1 (A4629)	24-bit message	KAT	CAST	Module is operational	Message digest	Module initialization (before integrity test)
SHA-1 (A4630)	24-bit message	KAT	CAST	Module is operational	Message digest	Module initialization (before integrity test)
SHA-1 (A4631)	24-bit message	KAT	CAST	Module is operational	Message digest	Module initialization (before integrity test)
SHA-1 (A4632)	24-bit message	KAT	CAST	Module is operational	Message digest	Module initialization (before integrity test)
SHA2-512 (A4612)	24-bit message	KAT	CAST	Module is operational	Message digest	Module initialization

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
						(before integrity test)
SHA2-512 (A4618)	24-bit message	KAT	CAST	Module is operational	Message digest	Module initialization (before integrity test)
SHA2-512 (A4629)	24-bit message	KAT	CAST	Module is operational	Message digest	Module initialization (before integrity test)
SHA2-512 (A4630)	24-bit message	KAT	CAST	Module is operational	Message digest	Module initialization (before integrity test)
SHA2-512 (A4631)	24-bit message	KAT	CAST	Module is operational	Message digest	Module initialization (before integrity test)
SHA2-512 (A4632)	24-bit message	KAT	CAST	Module is operational	Message digest	Module initialization (before integrity test)
SHA3-256 (A4613)	32-bit message	KAT	CAST	Module is operational	Message digest	Module initialization (before integrity test)
SHA3-256 (A4619)	32-bit message	KAT	CAST	Module is operational	Message digest	Module initialization (before integrity test)
AES-GCM (A4614)	256-bit key, 96-bit IV, 128-bit plaintext, 128-bit additional data	KAT	CAST	Module is operational	Encryption, Decryption (Separately)	Module initialization (before integrity test)
AES-GCM (A4615)	256-bit key, 96-bit IV, 128-bit plaintext, 128-bit additional data	KAT	CAST	Module is operational	Encryption, Decryption (Separately)	Module initialization (before integrity test)
AES-GCM (A4616)	256-bit key, 96-bit IV, 128-bit plaintext, 128-bit additional data	KAT	CAST	Module is operational	Encryption, Decryption (Separately)	Module initialization (before integrity test)
AES-GCM (A4617)	256-bit key, 96-bit IV, 128-bit plaintext, 128-bit additional data	KAT	CAST	Module is operational	Encryption, Decryption (Separately)	Module initialization (before integrity test)
AES-GCM (A4620)	256-bit key, 96-bit IV, 128-bit plaintext, 128-bit additional data	KAT	CAST	Module is operational	Encryption, Decryption (Separately)	Module initialization (before integrity test)
AES-GCM (A4621)	256-bit key, 96-bit IV, 128-bit plaintext, 128-bit additional data	KAT	CAST	Module is operational	Encryption, Decryption (Separately)	Module initialization (before integrity test)
AES-GCM (A4622)	256-bit key, 96-bit IV, 128-bit plaintext, 128-bit additional data	KAT	CAST	Module is operational	Encryption, Decryption (Separately)	Module initialization (before integrity test)

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-GCM (A4623)	256-bit key, 96-bit IV, 128-bit plaintext, 128-bit additional data	KAT	CAST	Module is operational	Encryption, Decryption (Separately)	Module initialization (before integrity test)
AES-GCM (A4624)	256-bit key, 96-bit IV, 128-bit plaintext, 128-bit additional data	KAT	CAST	Module is operational	Encryption, Decryption (Separately)	Module initialization (before integrity test)
AES-GCM (A4625)	256-bit key, 96-bit IV, 128-bit plaintext, 128-bit additional data	KAT	CAST	Module is operational	Encryption, Decryption (Separately)	Module initialization (before integrity test)
AES-GCM (A4626)	256-bit key, 96-bit IV, 128-bit plaintext, 128-bit additional data	KAT	CAST	Module is operational	Encryption, Decryption (Separately)	Module initialization (before integrity test)
AES-GCM (A4627)	256-bit key, 96-bit IV, 128-bit plaintext, 128-bit additional data	KAT	CAST	Module is operational	Encryption, Decryption (Separately)	Module initialization (before integrity test)
AES-GCM (A4614)	256-bit key, 96-bit IV, 128-bit plaintext, 128-bit additional data	KAT	CAST	Module is operational	Encryption, Decryption (Separately)	Module initialization (before integrity test)
AES-ECB (A4605)	128-bit key; 128-bit ciphertext	KAT	CAST	Module is operational	Decryption	Module initialization (before integrity test)
AES-ECB (A4606)	128-bit key; 128-bit ciphertext	KAT	CAST	Module is operational	Decryption	Module initialization (before integrity test)
AES-ECB (A4607)	128-bit key; 128-bit ciphertext	KAT	CAST	Module is operational	Decryption	Module initialization (before integrity test)
AES-ECB (A4609)	128-bit key; 128-bit ciphertext	KAT	CAST	Module is operational	Decryption	Module initialization (before integrity test)
AES-ECB (A4610)	128-bit key; 128-bit ciphertext	KAT	CAST	Module is operational	Decryption	Module initialization (before integrity test)
AES-ECB (A4611)	128-bit key; 128-bit ciphertext	KAT	CAST	Module is operational	Decryption	Module initialization (before integrity test)
AES-ECB (A4635)	128-bit key; 128-bit ciphertext	KAT	CAST	Module is operational	Decryption	Module initialization (before integrity test)
AES-ECB (A4636)	128-bit key; 128-bit ciphertext	KAT	CAST	Module is operational	Decryption	Module initialization (before integrity test)
AES-ECB (A4637)	128-bit key; 128-bit ciphertext	KAT	CAST	Module is operational	Decryption	Module initialization

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
						(before integrity test)
AES-ECB (A4638)	128-bit key; 128-bit ciphertext	KAT	CAST	Module is operational	Decryption	Module initialization (before integrity test)
AES-ECB (A4639)	128-bit key; 128-bit ciphertext	KAT	CAST	Module is operational	Decryption	Module initialization (before integrity test)
KDF SP800-108 (A4640)	Counter mode; HMAC-SHA-256; 128-bit input key	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDA OneStep SP800-56Cr2 (A4641)	SHA-224; 392-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDA HKDF Sp800-56Cr1 (A4603)	SHA-256, 48-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDF ANS 9.42 (A4613)	SHA-1 with AES-128 KW; 160-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDF ANS 9.42 (A4612)	SHA-1 with AES-128 KW; 160-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDF ANS 9.42 (A4618)	SHA-1 with AES-128 KW; 160-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDF ANS 9.42 (A4619)	SHA-1 with AES-128 KW; 160-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDF ANS 9.42 (A4629)	SHA-1 with AES-128 KW; 160-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDF ANS 9.42 (A4630)	SHA-1 with AES-128 KW; 160-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDF ANS 9.42 (A4631)	SHA-1 with AES-128 KW; 160-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDF ANS 9.42 (A4632)	SHA-1 with AES-128 KW; 160-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDF ANS 9.63 (A4618)	SHA-256; 192-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
KDF SSH (A4635)	SHA-1; 1056-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDF SSH (A4636)	SHA-1; 1056-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDF SSH (A4637)	SHA-1; 1056-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDF SSH (A4638)	SHA-1; 1056-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDF SSH (A4639)	SHA-1; 1056-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
TLS v1.2 KDF RFC7627 (A4612)	SHA-256; 384-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
TLS v1.2 KDF RFC7627 (A4618)	SHA-256; 384-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
TLS v1.2 KDF RFC7627 (A4629)	SHA-256; 384-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
TLS v1.2 KDF RFC7627 (A4630)	SHA-256; 384-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
TLS v1.2 KDF RFC7627 (A4631)	SHA-256; 384-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
TLS v1.2 KDF RFC7627 (A4632)	SHA-256; 384-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
TLS v1.3 KDF (A4603)	Extract and expand modes; SHA-256	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
PBKDF (A4612)	SHA-256; 24 character password; 288-bit salt; Iteration count: 4096	KAT	CAST	Module is operational	Password-based key derivation	Module initialization (before integrity test)
PBKDF (A4613)	SHA-256; 24 character password; 288-bit salt; Iteration count: 4096	KAT	CAST	Module is operational	Password-based key derivation	Module initialization (before integrity test)

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
PBKDF (A4618)	SHA-256; 24 character password; 288-bit salt; Iteration count: 4096	KAT	CAST	Module is operational	Password-based key derivation	Module initialization (before integrity test)
PBKDF (A4619)	SHA-256; 24 character password; 288-bit salt; Iteration count: 4096	KAT	CAST	Module is operational	Password-based key derivation	Module initialization (before integrity test)
PBKDF (A4629)	SHA-256; 24 character password; 288-bit salt; Iteration count: 4096	KAT	CAST	Module is operational	Password-based key derivation	Module initialization (before integrity test)
PBKDF (A4630)	SHA-256; 24 character password; 288-bit salt; Iteration count: 4096	KAT	CAST	Module is operational	Password-based key derivation	Module initialization (before integrity test)
PBKDF (A4631)	SHA-256; 24 character password; 288-bit salt; Iteration count: 4096	KAT	CAST	Module is operational	Password-based key derivation	Module initialization (before integrity test)
PBKDF (A4632)	SHA-256; 24 character password; 288-bit salt; Iteration count: 4096	KAT	CAST	Module is operational	Password-based key derivation	Module initialization (before integrity test)
Counter DRBG (A4604)	AES-128 with prediction resistance	KAT	CAST	Module is operational	Instantiate; Generate; Reseed (compliant to SP 800-90Arev1 Section 11.3)	Module initialization (before integrity test)
Hash DRBG (A4604)	SHA-256 with prediction resistance	KAT	CAST	Module is operational	Instantiate; Generate; Reseed (compliant to SP 800-90Arev1 Section 11.3)	Module initialization (before integrity test)
HMAC DRBG (A4604)	SHA-1 with prediction resistance	KAT	CAST	Module is operational	Instantiate; Generate; Reseed (compliant to SP 800-90Arev1 Section 11.3)	Module initialization (before integrity test)
KAS-FFC-SSC Sp800-56Ar3 (A4642)	ffdhe2048	KAT	CAST	Module is operational	Shared Secret Computation	Module initialization (before integrity test)
KAS-ECC-SSC Sp800-56Ar3 (A4612)	P-256	KAT	CAST	Module is operational	Shared Secret Computation	Module initialization (before integrity test)
KAS-ECC-SSC Sp800-56Ar3 (A4618)	P-256	KAT	CAST	Module is operational	Shared Secret Computation	Module initialization (before integrity test)

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
KAS-ECC-SSC Sp800-56Ar3 (A4629)	P-256	KAT	CAST	Module is operational	Shared Secret Computation	Module initialization (before integrity test)
KAS-ECC-SSC Sp800-56Ar3 (A4630)	P-256	KAT	CAST	Module is operational	Shared Secret Computation	Module initialization (before integrity test)
KAS-ECC-SSC Sp800-56Ar3 (A4631)	P-256	KAT	CAST	Module is operational	Shared Secret Computation	Module initialization (before integrity test)
KAS-ECC-SSC Sp800-56Ar3 (A4632)	P-256	KAT	CAST	Module is operational	Shared Secret Computation	Module initialization (before integrity test)
RSA SigGen (FIPS186-5) (A4612)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature generation	Module initialization (before integrity test)
RSA SigGen (FIPS186-5) (A4613)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature generation	Module initialization (before integrity test)
RSA SigGen (FIPS186-5) (A4618)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature generation	Module initialization (before integrity test)
RSA SigGen (FIPS186-5) (A4619)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature generation	Module initialization (before integrity test)
RSA SigGen (FIPS186-5) (A4629)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature generation	Module initialization (before integrity test)
RSA SigGen (FIPS186-5) (A4630)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature generation	Module initialization (before integrity test)
RSA SigGen (FIPS186-5) (A4631)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature generation	Module initialization (before integrity test)
RSA SigGen (FIPS186-5) (A4632)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature generation	Module initialization (before integrity test)
RSA SigVer (FIPS186-5) (A4612)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
RSA SigVer (FIPS186-5) (A4613)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
RSA SigVer (FIPS186-5) (A4618)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature verification	Module initialization

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
						(before integrity test)
RSA SigVer (FIPS186-5) (A4619)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
RSA SigVer (FIPS186-5) (A4629)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
RSA SigVer (FIPS186-5) (A4630)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
RSA SigVer (FIPS186-5) (A4631)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
RSA SigVer (FIPS186-5) (A4632)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
RSA SigVer (FIPS186-4) (A4612)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
RSA SigVer (FIPS186-4) (A4618)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
RSA SigVer (FIPS186-4) (A4629)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
RSA SigVer (FIPS186-4) (A4630)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
RSA SigVer (FIPS186-4) (A4631)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
RSA SigVer (FIPS186-4) (A4632)	PKCS#1 v1.5 with SHA-256; 2048-bit key	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
ECDSA SigGen (FIPS186-5) (A4612)	SHA-256; P-224, P-256, P-384, P-521	KAT	CAST	Module is operational	Signature generation	Module initialization (before integrity test)
ECDSA SigGen (FIPS186-5) (A4613)	SHA-256; P-224, P-256, P-384, P-521	KAT	CAST	Module is operational	Signature generation	Module initialization (before integrity test)
ECDSA SigGen (FIPS186-5) (A4618)	SHA-256; P-224, P-256, P-384, P-521	KAT	CAST	Module is operational	Signature generation	Module initialization (before integrity test)

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
ECDSA SigGen (FIPS186-5) (A4619)	SHA-256; P-224, P-256, P-384, P-521	KAT	CAST	Module is operational	Signature generation	Module initialization (before integrity test)
ECDSA SigGen (FIPS186-5) (A4629)	SHA-256; P-224, P-256, P-384, P-521	KAT	CAST	Module is operational	Signature generation	Module initialization (before integrity test)
ECDSA SigGen (FIPS186-5) (A4630)	SHA-256; P-224, P-256, P-384, P-521	KAT	CAST	Module is operational	Signature generation	Module initialization (before integrity test)
ECDSA SigGen (FIPS186-5) (A4631)	SHA-256; P-224, P-256, P-384, P-521	KAT	CAST	Module is operational	Signature generation	Module initialization (before integrity test)
ECDSA SigGen (FIPS186-5) (A4632)	SHA-256; P-224, P-256, P-384, P-521	KAT	CAST	Module is operational	Signature generation	Module initialization (before integrity test)
ECDSA SigVer (FIPS186-5) (A4618)	SHA-256; P-224, P-256, P-384, P-521	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
ECDSA SigVer (FIPS186-5) (A4619)	SHA-256; P-224, P-256, P-384, P-521	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
ECDSA SigVer (FIPS186-5) (A4629)	SHA-256; P-224, P-256, P-384, P-521	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
ECDSA SigVer (FIPS186-5) (A4630)	SHA-256; P-224, P-256, P-384, P-521	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
ECDSA SigVer (FIPS186-5) (A4631)	SHA-256; P-224, P-256, P-384, P-521	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
ECDSA SigVer (FIPS186-5) (A4632)	SHA-256; P-224, P-256, P-384, P-521	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
Safe Primes Key Generation (A4642)	N/A	PCT	PCT	Key pair generation is successful	SP 800-56Arev3 Section 5.6.2.1.4	Key pair generation
RSA KeyGen (FIPS186-5) (A4612)	N/A	PCT	PCT	Key pair generation is sucessful	Signature generation & verification	Key pair generation
RSA KeyGen (FIPS186-5) (A4618)	N/A	PCT	PCT	Key pair generation is sucessful	Signature generation & verification	Key pair generation
RSA KeyGen (FIPS186-5) (A4629)	N/A	PCT	PCT	Key pair generation is sucessful	Signature generation & verification	Key pair generation

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
RSA KeyGen (FIPS186-5) (A4630)	N/A	PCT	PCT	Key pair generation is sucessful	Signature generation & verification	Key pair generation
RSA KeyGen (FIPS186-5) (A4631)	N/A	PCT	PCT	Key pair generation is sucessful	Signature generation & verification	Key pair generation
RSA KeyGen (FIPS186-5) (A4632)	N/A	PCT	PCT	Key pair generation is sucessful	Signature generation & verification	Key pair generation
ECDSA KeyGen (FIPS186-5) (A4612)	SHA-256	PCT	PCT	Key pair generation successful	Signature generation & verification	Key pair generation
ECDSA KeyGen (FIPS186-5) (A4618)	SHA-256	PCT	PCT	Key pair generation successful	Signature generation & verification	Key pair generation
ECDSA KeyGen (FIPS186-5) (A4629)	SHA-256	PCT	PCT	Key pair generation successful	Signature generation & verification	Key pair generation
ECDSA KeyGen (FIPS186-5) (A4630)	SHA-256	PCT	PCT	Key pair generation successful	Signature generation & verification	Key pair generation
ECDSA KeyGen (FIPS186-5) (A4631)	SHA-256	PCT	PCT	Key pair generation successful	Signature generation & verification	Key pair generation
ECDSA KeyGen (FIPS186-5) (A4632)	SHA-256	PCT	PCT	Key pair generation successful	Signature generation & verification	Key pair generation
ECDSA SigVer (FIPS186-5) (A4613)	SHA-256; P-224, P-256, P-384, P-521	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
KDF ANS 9.63 (A4613)	SHA-256; 192-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
ECDSA SigVer (FIPS186-5) (A4612)	SHA-256; P-224, P-256, P-384, P-521	KAT	CAST	Module is operational	Signature verification	Module initialization (before integrity test)
KDF ANS 9.63 (A4612)	SHA-256; 192-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDF ANS 9.63 (A4619)	SHA-256; 192-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDF ANS 9.63 (A4629)	SHA-256; 192-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
KDF ANS 9.63 (A4630)	SHA-256; 192-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDF ANS 9.63 (A4631)	SHA-256; 192-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDF ANS 9.63 (A4632)	SHA-256; 192-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDA TwoStep SP800-56Cr2 (A4641)	SHA-256, 48-bit input secret	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)
KDF SP800-108 (A4608)	Counter mode; HMAC-SHA-256; 128-bit input key	KAT	CAST	Module is operational	Key derivation	Module initialization (before integrity test)

Table 21: Conditional Self-Tests

The module performs self-tests on all approved cryptographic algorithms as part of the approved services supported in the approved mode of operation, using the tests shown in the table above.

Upon generation of a DH, RSA, or EC key pair, the module will perform a pair-wise consistency test (PCT) as shown in the Conditional Self-tests table in Section 10.2 Conditional Self-Tests, which provides some assurance that the generated key pair is well formed. For DH key pairs, this test consists of the PCT described in Section 5.6.2.1.4 of SP 800-56Ar3. For RSA and EC key pairs, this test consists of a signature generation and a signature verification operation.

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 (A4608)	Message authentication	SW/FW Integrity	On demand	Unload and re-initialize the module
HMAC-SHA2-256 (A4612)	Message authentication	SW/FW Integrity	On demand	Unload and re-initialize the module
HMAC-SHA2-256 (A4618)	Message authentication	SW/FW Integrity	On demand	Unload and re-initialize the module
HMAC-SHA2-256 (A4629)	Message authentication	SW/FW Integrity	On demand	Unload and re-initialize the module
HMAC-SHA2-256 (A4630)	Message authentication	SW/FW Integrity	On demand	Unload and re-initialize the module
HMAC-SHA2-256 (A4631)	Message authentication	SW/FW Integrity	On demand	Unload and re-initialize the module
HMAC-SHA2-256 (A4632)	Message authentication	SW/FW Integrity	On demand	Unload and re-initialize the module

Table 22: Pre-Operational Periodic Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
SHA-1 (A4612)	KAT	CAST	On demand	Manually
SHA-1 (A4618)	KAT	CAST	On demand	Manually
SHA-1 (A4629)	KAT	CAST	On demand	Manually
SHA-1 (A4630)	KAT	CAST	On demand	Manually
SHA-1 (A4631)	KAT	CAST	On demand	Manually
SHA-1 (A4632)	KAT	CAST	On demand	Manually
SHA2-512 (A4612)	KAT	CAST	On demand	Manually
SHA2-512 (A4618)	KAT	CAST	On demand	Manually
SHA2-512 (A4629)	KAT	CAST	On demand	Manually
SHA2-512 (A4630)	KAT	CAST	On demand	Manually
SHA2-512 (A4631)	KAT	CAST	On demand	Manually
SHA2-512 (A4632)	KAT	CAST	On demand	Manually
SHA3-256 (A4613)	KAT	CAST	On demand	Manually
SHA3-256 (A4619)	KAT	CAST	On demand	Manually
AES-GCM (A4614)	KAT	CAST	On demand	Manually
AES-GCM (A4615)	KAT	CAST	On demand	Manually
AES-GCM (A4616)	KAT	CAST	On demand	Manually
AES-GCM (A4617)	KAT	CAST	On demand	Manually
AES-GCM (A4620)	KAT	CAST	On demand	Manually
AES-GCM (A4621)	KAT	CAST	On demand	Manually
AES-GCM (A4622)	KAT	CAST	On demand	Manually
AES-GCM (A4623)	KAT	CAST	On demand	Manually
AES-GCM (A4624)	KAT	CAST	On demand	Manually
AES-GCM (A4625)	KAT	CAST	On demand	Manually
AES-GCM (A4626)	KAT	CAST	On demand	Manually
AES-GCM (A4627)	KAT	CAST	On demand	Manually
AES-GCM (A4614)	KAT	CAST	On demand	Manually
AES-ECB (A4605)	KAT	CAST	On demand	Manually
AES-ECB (A4606)	KAT	CAST	On demand	Manually
AES-ECB (A4607)	KAT	CAST	On demand	Manually
AES-ECB (A4609)	KAT	CAST	On demand	Manually
AES-ECB (A4610)	KAT	CAST	On demand	Manually
AES-ECB (A4611)	KAT	CAST	On demand	Manually
AES-ECB (A4635)	KAT	CAST	On demand	Manually
AES-ECB (A4636)	KAT	CAST	On demand	Manually
AES-ECB (A4637)	KAT	CAST	On demand	Manually
AES-ECB (A4638)	KAT	CAST	On demand	Manually
AES-ECB (A4639)	KAT	CAST	On demand	Manually
KDF SP800-108 (A4640)	KAT	CAST	On demand	Manually
KDA OneStep SP800-56Cr2 (A4641)	KAT	CAST	On demand	Manually
KDA HKDF Sp800-56Cr1 (A4603)	KAT	CAST	On demand	Manually
KDF ANS 9.42 (A4613)	KAT	CAST	On demand	Manually
KDF ANS 9.42 (A4612)	KAT	CAST	On demand	Manually
KDF ANS 9.42 (A4618)	KAT	CAST	On demand	Manually
KDF ANS 9.42 (A4619)	KAT	CAST	On demand	Manually
KDF ANS 9.42 (A4629)	KAT	CAST	On demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
KDF ANS 9.42 (A4630)	KAT	CAST	On demand	Manually
KDF ANS 9.42 (A4631)	KAT	CAST	On demand	Manually
KDF ANS 9.42 (A4632)	KAT	CAST	On demand	Manually
KDF ANS 9.63 (A4618)	KAT	CAST	On demand	Manually
KDF SSH (A4635)	KAT	CAST	On demand	Manually
KDF SSH (A4636)	KAT	CAST	On demand	Manually
KDF SSH (A4637)	KAT	CAST	On demand	Manually
KDF SSH (A4638)	KAT	CAST	On demand	Manually
KDF SSH (A4639)	KAT	CAST	On demand	Manually
TLS v1.2 KDF RFC7627 (A4612)	KAT	CAST	On demand	Manually
TLS v1.2 KDF RFC7627 (A4618)	KAT	CAST	On demand	Manually
TLS v1.2 KDF RFC7627 (A4629)	KAT	CAST	On demand	Manually
TLS v1.2 KDF RFC7627 (A4630)	KAT	CAST	On demand	Manually
TLS v1.2 KDF RFC7627 (A4631)	KAT	CAST	On demand	Manually
TLS v1.2 KDF RFC7627 (A4632)	KAT	CAST	On demand	Manually
TLS v1.3 KDF (A4603)	KAT	CAST	On demand	Manually
PBKDF (A4612)	KAT	CAST	On demand	Manually
PBKDF (A4613)	KAT	CAST	On demand	Manually
PBKDF (A4618)	KAT	CAST	On demand	Manually
PBKDF (A4619)	KAT	CAST	On demand	Manually
PBKDF (A4629)	KAT	CAST	On demand	Manually
PBKDF (A4630)	KAT	CAST	On demand	Manually
PBKDF (A4631)	KAT	CAST	On demand	Manually
PBKDF (A4632)	KAT	CAST	On demand	Manually
Counter DRBG (A4604)	KAT	CAST	On demand	Manually
Hash DRBG (A4604)	KAT	CAST	On demand	Manually
HMAC DRBG (A4604)	KAT	CAST	On demand	Manually
KAS-FFC-SSC Sp800-56Ar3 (A4642)	KAT	CAST	On demand	Manually
KAS-ECC-SSC Sp800-56Ar3 (A4612)	KAT	CAST	On demand	Manually
KAS-ECC-SSC Sp800-56Ar3 (A4618)	KAT	CAST	On demand	Manually
KAS-ECC-SSC Sp800-56Ar3 (A4629)	KAT	CAST	On demand	Manually
KAS-ECC-SSC Sp800-56Ar3 (A4630)	KAT	CAST	On demand	Manually
KAS-ECC-SSC Sp800-56Ar3 (A4631)	KAT	CAST	On demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
KAS-ECC-SSC Sp800-56Ar3 (A4632)	KAT	CAST	On demand	Manually
RSA SigGen (FIPS186-5) (A4612)	KAT	CAST	On demand	Manually
RSA SigGen (FIPS186-5) (A4613)	KAT	CAST	On demand	Manually
RSA SigGen (FIPS186-5) (A4618)	KAT	CAST	On demand	Manually
RSA SigGen (FIPS186-5) (A4619)	KAT	CAST	On demand	Manually
RSA SigGen (FIPS186-5) (A4629)	KAT	CAST	On demand	Manually
RSA SigGen (FIPS186-5) (A4630)	KAT	CAST	On demand	Manually
RSA SigGen (FIPS186-5) (A4631)	KAT	CAST	On demand	Manually
RSA SigGen (FIPS186-5) (A4632)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-5) (A4612)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-5) (A4613)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-5) (A4618)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-5) (A4619)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-5) (A4629)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-5) (A4630)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-5) (A4631)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-5) (A4632)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-4) (A4612)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-4) (A4618)	KAT	CAST	On demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
RSA SigVer (FIPS186-4) (A4629)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-4) (A4630)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-4) (A4631)	KAT	CAST	On demand	Manually
RSA SigVer (FIPS186-4) (A4632)	KAT	CAST	On demand	Manually
ECDSA SigGen (FIPS186-5) (A4612)	KAT	CAST	On demand	Manually
ECDSA SigGen (FIPS186-5) (A4613)	KAT	CAST	On demand	Manually
ECDSA SigGen (FIPS186-5) (A4618)	KAT	CAST	On demand	Manually
ECDSA SigGen (FIPS186-5) (A4619)	KAT	CAST	On demand	Manually
ECDSA SigGen (FIPS186-5) (A4629)	KAT	CAST	On demand	Manually
ECDSA SigGen (FIPS186-5) (A4630)	KAT	CAST	On demand	Manually
ECDSA SigGen (FIPS186-5) (A4631)	KAT	CAST	On demand	Manually
ECDSA SigGen (FIPS186-5) (A4632)	KAT	CAST	On demand	Manually
ECDSA SigVer (FIPS186-5) (A4618)	KAT	CAST	On demand	Manually
ECDSA SigVer (FIPS186-5) (A4619)	KAT	CAST	On demand	Manually
ECDSA SigVer (FIPS186-5) (A4629)	KAT	CAST	On demand	Manually
ECDSA SigVer (FIPS186-5) (A4630)	KAT	CAST	On demand	Manually
ECDSA SigVer (FIPS186-5) (A4631)	KAT	CAST	On demand	Manually
ECDSA SigVer (FIPS186-5) (A4632)	KAT	CAST	On demand	Manually
Safe Primes Key Generation (A4642)	PCT	PCT	On demand	Manually
RSA KeyGen (FIPS186-5) (A4612)	PCT	PCT	On demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
RSA KeyGen (FIPS186-5) (A4618)	PCT	PCT	On demand	Manually
RSA KeyGen (FIPS186-5) (A4629)	PCT	PCT	On demand	Manually
RSA KeyGen (FIPS186-5) (A4630)	PCT	PCT	On demand	Manually
RSA KeyGen (FIPS186-5) (A4631)	PCT	PCT	On demand	Manually
RSA KeyGen (FIPS186-5) (A4632)	PCT	PCT	On demand	Manually
ECDSA KeyGen (FIPS186-5) (A4612)	PCT	PCT	On demand	Manually
ECDSA KeyGen (FIPS186-5) (A4618)	PCT	PCT	On demand	Manually
ECDSA KeyGen (FIPS186-5) (A4629)	PCT	PCT	On demand	Manually
ECDSA KeyGen (FIPS186-5) (A4630)	PCT	PCT	On demand	Manually
ECDSA KeyGen (FIPS186-5) (A4631)	PCT	PCT	On demand	Manually
ECDSA KeyGen (FIPS186-5) (A4632)	PCT	PCT	On demand	Manually
ECDSA SigVer (FIPS186-5) (A4613)	KAT	CAST	On demand	Manually
KDF ANS 9.63 (A4613)	KAT	CAST	On demand	Manually
ECDSA SigVer (FIPS186-5) (A4612)	KAT	CAST	On demand	Manually
KDF ANS 9.63 (A4612)	KAT	CAST	On demand	Manually
KDF ANS 9.63 (A4619)	KAT	CAST	On demand	Manually
KDF ANS 9.63 (A4629)	KAT	CAST	On demand	Manually
KDF ANS 9.63 (A4630)	KAT	CAST	On demand	Manually
KDF ANS 9.63 (A4631)	KAT	CAST	On demand	Manually
KDF ANS 9.63 (A4632)	KAT	CAST	On demand	Manually
KDA TwoStep SP800-56Cr2 (A4641)	KAT	CAST	On demand	Manually
KDF SP800-108 (A4608)	KAT	CAST	On demand	Manually

Table 23: Conditional Periodic Information

10.4 Error States

Name	Description	Conditions	Recovery Method	Indicator
Error	The module immediately stops functioning	Software integrity test failure CAST failure PCT failure	Re-initialization of the module	Module will not load; Module is aborted for PCT failure

Table 24: 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

The software integrity tests and cryptographic algorithm self-tests can be invoked on demand by unloading and subsequently re-initializing the module. 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

The module is distributed as a part of the Amazon Linux 2023 packages in the form of the openssl-3.0.8-1.amzn2023.0.17 RPM package.

Before the openssl-3.0.8-1.amzn2023.0.17 RPM package is installed, the Amazon Linux 2023 systems must operate in the FIPS validated configuration. This can be achieved by switching the system into the FIPS validated configuration after the installation. Execute the openssl list -providers command. Restart the system. More information can be found at the vendor documentation.

The Crypto Officer must verify the Amazon Linux 2023 systems operates in the FIPS validated configuration by executing the fips-mode-setup –check command, which should output “FIPS mode is enabled.”

11.2 Administrator Guidance

After installation of the openssl-3.0.8-1.amzn2023.0.17 RPM package, the Crypto Officer must verify the module name and version by executing the openssl list -providers command. The Crypto Officer must ensure that the fips provider is listed in the output as follows:

```
fips
  name: Amazon Linux 2023 - OpenSSL FIPS Provider
  version: 3.0.8-d694bfa693b76001
  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 administrator guidance.

11.6 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-3.0.8-1.amzn2023.0.9 RPM package can be uninstalled from the Amazon Linux 2023 systems.

12 Mitigation of Other Attacks

12.1 Attack List

Certain cryptographic subroutines and algorithms are vulnerable to timing analysis. The module mitigates this vulnerability by using constant-time implementations. This includes, but is not limited to:

- Big number operations: computing GCDs, modular inversion, multiplication, division, and modular exponentiation (using Montgomery multiplication)
- Elliptic curve point arithmetic: addition and multiplication (using the Montgomery ladder)
- Vector-based AES implementations

RSA, ECDSA, ECDH, and DH employ blinding techniques to further impede timing and power analysis.

Appendix A. Glossary and Abbreviations

AES	Advanced Encryption Standard
AES-NI	Advanced Encryption Standard New Instructions
API	Application Programming Interface
CAST	Cryptographic Algorithm Self-Test
CAVP	Cryptographic Algorithm Validation Program
CBC	Cipher Block Chaining
CCM	Counter with Cipher Block Chaining-Message Authentication Code
CE	Cryptography Extensions
CFB	Cipher Feedback
CKG	Cryptographic Key Generation
CMAC	Cipher-based Message Authentication Code
CMVP	Cryptographic Module Validation Program
CPACF	CP Assist for Cryptographic Functions
CSP	Critical Security Parameter
CTR	Counter
CTS	Ciphertext Stealing
DH	Diffie-Hellman
DRBG	Deterministic Random Bit Generator
ECB	Electronic Code Book
ECC	Elliptic Curve Cryptography
ECDH	Elliptic Curve Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
EVP	Envelope
FFC	Finite Field Cryptography
FIPS	Federal Information Processing Standards
GCM	Galois Counter Mode
GMAC	Galois Counter Mode Message Authentication Code
HKDF	HMAC-based Key Derivation Function
HMAC	Keyed-Hash Message Authentication Code
IKE	Internet Key Exchange
KAS	Key Agreement Scheme
KAT	Known Answer Test
KBKDF	Key-based Key Derivation Function
KW	Key Wrap
KWP	Key Wrap with Padding
MAC	Message Authentication Code
NIST	National Institute of Science and Technology
OFB	Output Feedback

PAA	Processor Algorithm Acceleration
PCT	Pair-wise Consistency Test
PBKDF2	Password-based Key Derivation Function v2
PSS	Probabilistic Signature Scheme
RSA	Rivest, Shamir, Adleman
SHA	Secure Hash Algorithm
SSC	Shared Secret Computation
SSH	Secure Shell
SSP	Sensitive Security Parameter
TLS	Transport Layer Security
XOF	Extendable Output Function
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 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** January 2024 https://csrc.nist.gov/Projects/cryptographic-module-validation-program/fips-140-3-ig-announcements
FIPS 180-4	**Secure Hash Standard (SHS)** March 2012 https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf
FIPS 186-4	**Digital Signature Standard (DSS)** July 2013 https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf
FIPS 186-5	**Digital Signature Standard (DSS)** February 2023 https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-5.pdf
FIPS 197	**Advanced Encryption Standard** November 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 5288	**AES Galois Counter Mode (GCM) Cipher Suites for TLS** August 2008 https://www.ietf.org/rfc/rfc5288.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 800-38A	**Recommendation for Block Cipher Modes of Operation Methods and Techniques** December 2001 https://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.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://csrc.nist.gov/publications/nistpubs/800-38B/SP_800-38B.pdf
SP 800-38C	**Recommendation for Block Cipher Modes of Operation: the CCM Mode for Authentication and Confidentiality** May 2004 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://csrc.nist.gov/publications/nistpubs/800-38E/nist-sp-800-38E.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 800-52r2	**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 800-56Ar3	**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 800-56Cr1	**Recommendation for Key-Derivation Methods in Key-Establishment Schemes** April 2018 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Cr1.pdf
SP 800-56Cr2	**Recommendation for Key-Derivation Methods in Key-Establishment Schemes** August 2020 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Cr2.pdf
SP 800-90Ar1	**Recommendation for Random Number Generation Using Deterministic Random Bit Generators** June 2015 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf
SP 800-90B	**Recommendation for the Entropy Sources Used for Random Bit Generation** January 2018 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90B.pdf

SP 800-108r1	**NIST Special Publication 800-108 - Recommendation for Key Derivation Using Pseudorandom Functions** August 2022 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-108r1.pdf
SP 800-132	**Recommendation for Password-Based Key Derivation - Part 1: Storage Applications** December 2010 https://csrc.nist.gov/publications/nistpubs/800-132/nist-sp800-132.pdf
SP 800-133r2	**Recommendation for Cryptographic Key Generation** June 2020 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-133r2.pdf
SP 800-135r1	**Recommendation for Existing Application-Specific Key Derivation Functions** December 2011 https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-135r1.pdf