Amazon Web Services, Inc

AWS-LC Cryptographic Module (dynamic)

FIPS 140-3 Non-Proprietary Security Policy

Document version: 1.2

Last update: 2026-01-20

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

Table of Contents

List of Tables

List of Figures

1 General

1.1 Overview

This document is the non-proprietary FIPS 140-3 Security Policy for version AWS-LC FIPS 1.29.1 of the AWS-LC Cryptographic Module (dynamic). 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.

1.2 Security Levels

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

Table 1: Security Levels

1.3 Additional Information

This Security Policy describes the features and design of the module named AWS-LC Cryptographic Module (dynamic) using the terminology contained in the FIPS 140-3 specification. The FIPS 140-3 Security Requirements for Cryptographic Module specifies the security requirements that will be satisfied by a cryptographic module utilized within a security system protecting sensitive but unclassified information. The NIST/CCCS Cryptographic Module Validation Program (CMVP) validates cryptographic module to FIPS 140-3. Validated products are accepted by the Federal agencies of both the USA and Canada for the protection of sensitive or designated information.

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.

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

2 Cryptographic Module Specification

2.1 Description

Purpose and Use: The AWS-LC Cryptographic Module (dynamic) (hereafter referred to as "the module") provides cryptographic services to applications running in the user space of the underlying operating system through a C language Application Program Interface (API).

Module Type: Software

Module Embodiment: MultiChipStand

Cryptographic Boundary: The block diagram in Figure 1 shows the cryptographic boundary of the module, its interfaces with the operational environment and the flow of information between the module and operator (depicted through the arrows).

The cryptographic boundary is defined as the AWS-LC Cryptographic Module (dynamic) which is a cryptographic library consisting of the bcm.o file (version AWS-LC FIPS 1.29.1). This file is dynamically linked to the userspace application during the compilation process.

Tested Operational Environment's Physical Perimeter (TOEPP):

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

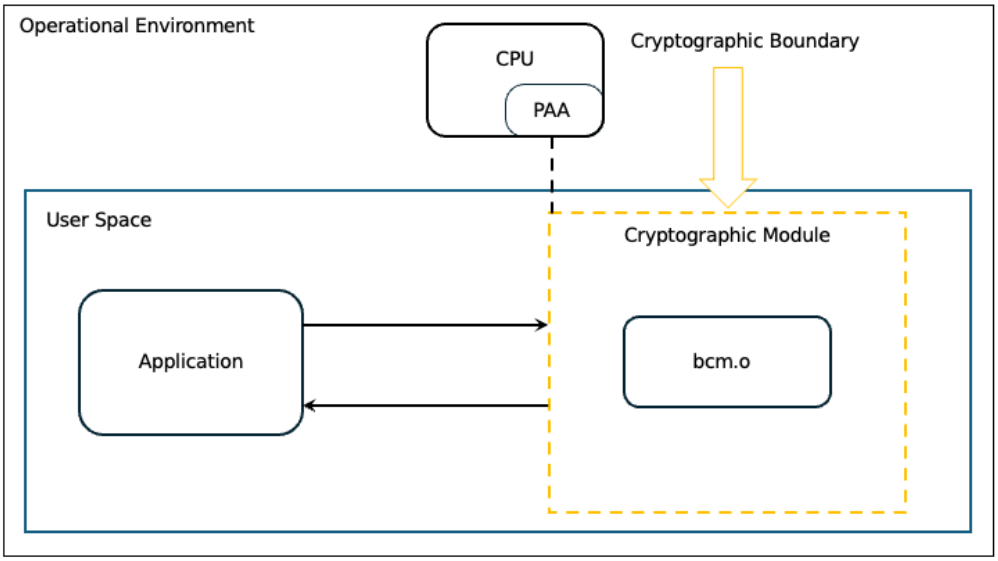

Figure 1: Block Diagram

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
bcm.o on NetOS 2024 on AS7772 on NXP T-Series T2080	AWS-LC FIPS 1.29.1	N/A	HMAC-SHA2-256

Package or File Name	Software/ Firmware Version	Features	Integrity Test
bcm.o on NetOS 2024 on CS8274 on NXP Layerscape LX2080	AWS-LC FIPS 1.29.1	N/A	HMAC-SHA2-256
bcm.o on NetOS 2024 v1.1 on AZ3324 on NXP Layerscape LX2080	AWS-LC FIPS 1.29.1	N/A	HMAC-SHA2-256
bcm.o on NetOS 2024 on CS8320 on Annapurna K2X-N	AWS-LC FIPS 1.29.1	N/A	HMAC-SHA2-256

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

Tested Operational Environments - Software, Firmware, Hybrid:

Operating System	Hardware Platform	Processors	PAA/PAI	Hypervisor or Host OS	Version(s)
NetOS 2024	CS8274	NXP Layerscape LX2080 (ARMv8)	Yes	N/A	AWS-LC FIPS 1.29.1
NetOS 2024	CS8320	Annapurna K2X-N (ARMv8)	Yes	N/A	AWS-LC FIPS 1.29.1
NetOS 2024 v1.1	AZ3324	NXP Layerscape LX2080 (ARMv8)	Yes	N/A	AWS-LC FIPS 1.29.1
NetOS 2024	AS7772	NXP T-Series T2080 (Power 7)	No	N/A	AWS-LC FIPS 1.29.1
NetOS 2024	CS8274	NXP Layerscape LX2080 (ARMv8)	No	N/A	AWS-LC FIPS 1.29.1
NetOS 2024	CS8320	Annapurna K2X-N (ARMv8)	No	N/A	AWS-LC FIPS 1.29.1
NetOS 2024 v1.1	AZ3324	NXP Layerscape LX2080 (ARMv8)	No	N/A	AWS-LC FIPS 1.29.1

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

2.3 Excluded Components

Not applicable.

2.4 Modes of Operation

Modes List and Description:

Mode Name	Description	Type	Status Indicator
Approved Mode	Automatically entered whenever an approved service is requested.	Approved	Equivalent to the indicator of the requested service as defined in section 4.3
Non-approved Mode	Automatically entered whenever a non-approved service is requested.	Non-Approved	Equivalent to the indicator of the requested service as defined in section 4.3

Table 4: Modes List and Description

Mode Change Instructions and Status:

When the module starts up successfully, after passing a set of cryptographic algorithms self-tests (CASTs) and the pre-operational self-test, the module is operating in the approved mode of operation by default and can only be transitioned into the non-approved mode by calling one of the non-approved services listed in the *Non-Approved Services* table. The module will transition back to approved mode when approved service is called. Section 4 provides details on the service indicator implemented by the module. The service indicator identifies when an approved service is called.

2.5 Algorithms

Approved Algorithms:

Algorithm	CAVP Cert	Properties	Reference
AES-CBC	A5422, A5427, A5429, A5431	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CCM	A5422, A5427, A5429, A5431	Key Length - 128	SP 800-38C
AES-CMAC	A5422, A5427, A5429, A5431	Direction - Generation, Verification Key Length - 128, 256	SP 800-38B
AES-CTR	A5422, A5427, A5429, A5431	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-ECB	A5422, A5423, A5427, A5428, A5429, A5430, A5431, A5432, A5435	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-GCM	A5423, A5428, A5430, A5432, A5435	Direction - Decrypt, Encrypt IV Generation - External, Internal Key Length - 128, 256 IV Generation Mode - 8.2.1, 8.2.2	SP 800-38D
AES-GMAC	A5423, A5428, A5430, A5432, A5435	Direction - Decrypt, Encrypt IV Generation - External, Internal Key Length - 128, 256 IV Generation Mode - 8.2.1, 8.2.2	SP 800-38D
AES-KW	A5422, A5427, A5429, A5431	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-KWP	A5422, A5427, A5429, A5431	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-XTS Testing Revision 2.0	A5422, A5427, A5429, A5431	Direction - Decrypt, Encrypt Key Length - 256	SP 800-38E
Counter DRBG	A5422, A5427, A5429, A5431	Prediction Resistance - No Mode - AES-256 Derivation Function Enabled - No	SP 800-90A Rev. 1
ECDSA KeyGen (FIPS186-5)	A5425, A5426, A5433, A5434	Curve - P-224, P-256, P-384, P-521 Secret Generation Mode - testing candidates	FIPS 186-5
ECDSA KeyVer (FIPS186-5)	A5425, A5426, A5433, A5434	Curve - P-224, P-256, P-384, P-521	FIPS 186-5
ECDSA SigGen (FIPS186-5)	A5425, A5426, A5433, A5434	Curve - P-224, P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512 Component - No	FIPS 186-5
ECDSA SigVer (FIPS186-4)	A5425, A5426, A5433, A5434	Component - No Curve - P-224, P-256, P-384, P-521 Hash Algorithm - SHA-1	FIPS 186-4
ECDSA SigVer (FIPS186-5)	A5425, A5426, A5433, A5434	Curve - P-224, P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512	FIPS 186-5
HMAC-SHA-1	A5425, A5426, A5433, A5434	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1

Algorithm	CAVP Cert	Properties	Reference
HMAC-SHA2-224	A5425, A5433, A5434	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA2-256	A5425, A5433, A5434	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA2-384	A5425, A5433, A5434	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512	A5425, A5433, A5434	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512/224	A5425, A5433, A5434	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512/256	A5425, A5433, A5434	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
KAS-ECC-SSC Sp800-56Ar3	A5425, A5426, A5433, A5434	Domain Parameter Generation Methods - P-224, P-256, P-384, P-521 Scheme - ephemeralUnified - KAS Role - initiator, responder	SP 800-56A Rev. 3
KDA HKDF Sp800-56Cr1	A5425, A5426, A5433, A5434	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	SP 800-56C Rev. 2
KDF SSH (CVL)	A5425, A5426, A5433, A5434	Cipher - AES-128, AES-192, AES-256 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
KDF TLS (CVL)	A5425, A5426, A5433, A5434	TLS Version - v1.0/1.1, v1.2 Hash Algorithm - SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
PBKDF	A5425, A5426, A5433, A5434	Iteration Count - Iteration Count: 1000-10000 Increment 1 Password Length - Password Length: 14-128 Increment 1	SP 800-132
RSA KeyGen (FIPS186-5)	A5425, A5426, A5433, A5434	Key Generation Mode - probable Modulo - 2048, 3072, 4096 Primality Tests - 2powSecStr Private Key Format - standard	FIPS 186-5
RSA SigGen (FIPS186-5)	A5425, A5426, A5433, A5434	Modulo - 2048, 3072, 4096 Signature Type - pkcs1v1.5, pss	FIPS 186-5
RSA SigVer (FIPS186-4)	A5425, A5426, A5433, A5434	Signature Type - PKCS 1.5, PKCSPSS Modulo - 1024, 2048, 3072, 4096	FIPS 186-4
RSA SigVer (FIPS186-5)	A5425, A5426, A5433, A5434	Modulo - 2048, 3072, 4096 Signature Type - pkcs1v1.5, pss	FIPS 186-5
SHA-1	A5425, A5426, A5433, A5434	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-224	A5425, A5433, A5434	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4

Algorithm	CAVP Cert	Properties	Reference
SHA2-256	A5425, A5433, A5434	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-384	A5425, A5433, A5434	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-512	A5425, A5433, A5434	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-512/224	A5425, A5433, A5434	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-512/256	A5425, A5433, A5434	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA3-224	A5424	Message Length - Message Length: 0-65536 Increment 8	FIPS 202
SHA3-256	A5424	Message Length - Message Length: 0-65536 Increment 8	FIPS 202
SHA3-384	A5424	Message Length - Message Length: 0-65536 Increment 8	FIPS 202
SHA3-512	A5424	Message Length - Message Length: 0-65536 Increment 8	FIPS 202
SHAKE-128	A5424	Output Length - Output Length: 16-65536 Increment 8	FIPS 202
SHAKE-256	A5424	Output Length - Output Length: 16-65536 Increment 8	FIPS 202

Table 5: Approved Algorithms

Vendor-Affirmed Algorithms:

Name	Properties	Implementation	Reference
Cryptographic Key Generation (CKG)	Key Type:Asymmetric RSA (FIPS 186-5):2048, 3072, 4096 bits with 112, 128, 150 bits of key strength. EC (FIPS 186-5):P-224, P-256, P-384, P-521 elliptic curves with 112-256 bits of key strength	N/A	SP 800-133Rev2 section 4, example 1

Table 6: Vendor-Affirmed Algorithms

Non-Approved, Allowed Algorithms:

N/A for this module.

Non-Approved, Allowed Algorithms with No Security Claimed:

Name	Caveat	Use and Function
MD5	Allowed per IG 2.4.A	Message Digest used in TLS 1.0/1.1 KDF only

Table 7: Non-Approved, Allowed Algorithms with No Security Claimed

Non-Approved, Not Allowed Algorithms: Name	Use and Function
AES with OFB or CFB1, CFB8 modes	Encryption, Decryption (not CAVP tested)
AES GCM, GMAC, XTS with keys not listed in Table 5	Encryption, Decryption
AES using aes_*_generic function	Encryption, Decryption (not CAVP tested)

Non-Approved, Not Allowed Algorithms: Name	Use and Function
AES GMAC using aes_*_generic	Message Authentication Generation (not CAVP tested)
Curve secp256k1	Signature Generation, Signature Verification, Shared Secret Computation
Diffie Hellman	Shared Secret Computation (not CAVP tested)
HMAC-MD4, HMAC-MD5, HMAC-SHA-3, HMAC-RIPEMD-160	Message Authentication Generation (not CAVP tested)
MD4	Message Digest
MD5 (outside of TLS)	Message Digest
RSA using RSA_generate_key_ex	Key Generation (not complaint with FIPS186-5)
ECDSA using EC_KEY_generate_key	Key Generation (not complaint with FIPS186-5)
RSA using keys less than 2048 bits	Signature Generation
RSA using keys less than 1024 bits	Signature Verification
RSA without hashing	Sign/Verify primitive operations
RSA encryption primitive with PKCS#1 v1.5 and OAEP padding	Encryption
SHA-1, SHA-3	Signature Generation (not CAVP tested)
RIPEMD-160	Message Digest
TLS KDF using any SHA algorithms other than SHA2-256, SHA2-384, SHA2-512; or TLS KDF using non-extended master secret	Key Derivation
RSA	Key Encapsulation/Un-encapsulation (not compliant with SP 800-56BRev2)

Table 8: Non-Approved, Not Allowed Algorithms

2.6 Security Function Implementations

Name	Type	Description	Properties	Algorithms
Shared Secret Computation with EC Diffie-Hellman	KAS-SSC	SP800-56Arev3. KAS-ECC-SSC per IG D.F Scenario 2 path (1).		KAS-ECC-SSC Sp800-56Ar3: (A5425, A5426, A5433, A5434)
Authenticated Encryption/Decryption with AES KW, AES-KWP	BC-Auth	SP800-38F. Authenticated encryption, Authenticated decryption		AES-KW: (A5422, A5427, A5429, A5431) AES-KWP: (A5422, A5427, A5429, A5431)
Encryption/Decryption with AES	BC-UnAuth	SP800-38A and SP 800-38E. Encryption and Decryption		AES-CBC: (A5422, A5427, A5429, A5431) AES-CTR: (A5422, A5427, A5429, A5431) AES-ECB: (A5422, A5423, A5427, A5428, A5429, A5430, A5431,

Name	Type	Description	Properties	Algorithms
				A5432, A5435) AES-XTS Testing Revision 2.0: (A5422, A5427, A5429, A5431)
Signature Generation with RSA	DigSig-SigGen	FIPS186-5. Digital signature generation		RSA SigGen (FIPS186-5): (A5425, A5426, A5433, A5434)
Signature Generation with ECDSA	DigSig-SigGen	FIPS186-5. Digital signature generation		ECDSA SigGen (FIPS186-5): (A5425, A5426, A5433, A5434)
Key Generation with RSA	AsymKeyPair-KeyGen CKG	FIPS186-5. Key generation		RSA KeyGen (FIPS186-5): (A5425, A5426, A5433, A5434)
Key Generation with ECDSA	AsymKeyPair-KeyGen CKG	FIPS186-5. Key generation		ECDSA KeyGen (FIPS186-5): (A5425, A5426, A5433, A5434)
Signature Verification with ECDSA	DigSig-SigVer	FIPS186-5. Digital signature verification		ECDSA SigVer (FIPS186-5): (A5425, A5426, A5433, A5434)
Signature Verification with RSA	DigSig-SigVer	FIPS186-5. Digital signature verification		RSA SigVer (FIPS186-5): (A5425, A5426, A5433, A5434)
Key Verification with ECDSA	AsymKeyPair-KeyVer	FIPS186-5. Key verification		ECDSA KeyVer (FIPS186-5): (A5425, A5426, A5433, A5434)
Key Derivation with TLS KDF	KAS-135KDF	SP800-135rev1. Key derivation		KDF TLS: (A5425, A5426, A5433, A5434)
Key Derivation with SSH KDF	KAS-135KDF	SP800-135rev1. Key derivation		KDF SSH: (A5425, A5426, A5433, A5434)
Key Derivation with KDA HKDF	KAS-56CKDF	SP800-56Crev1. Key derivation		KDA HKDF Sp800-56Cr1: (A5425, A5426, A5433, A5434)
Key Derivation with PBKDF	PBKDF	SP800-132. Key derivation		PBKDF: (A5425, A5426, A5433, A5434)
Message Digest with SHA	SHA	FIPS180-4 and FIPS202. Message digest using SHA		SHA-1: (A5425, A5426, A5433, A5434) SHA2-224: (A5425, A5433, A5434) SHA2-256: (A5425, A5433, A5434) SHA2-384: (A5425, A5433, A5434) SHA2-512: (A5425, A5433, A5434) SHA2-512/224: (A5425, A5433, A5434) SHA2-512/256: (A5425, A5433, A5434)

Name	Type	Description	Properties	Algorithms
				SHA3-224: (A5424) SHA3-256: (A5424) SHA3-384: (A5424) SHA3-512: (A5424)
Random Number Generation with DRBG	DRBG	SP800-90ARev1. Random number generation		Counter DRBG: (A5422, A5427, A5429, A5431)
Message Authentication Generation with HMAC	MAC	FIPS198-1. Message authentication generation		HMAC-SHA-1: (A5425, A5426, A5433, A5434) HMAC-SHA2-224: (A5425, A5433, A5434) HMAC-SHA2-256: (A5425, A5433, A5434) HMAC-SHA2-384: (A5425, A5433, A5434) HMAC-SHA2-512: (A5425, A5433, A5434) HMAC-SHA2-512/256: (A5425, A5433, A5434) HMAC-SHA2-512/224: (A5425, A5433, A5434)
Message Authentication Generation with AES	MAC	SP800-38B and SP800-38D Message authentication generation		AES-CMAC: (A5422, A5427, A5429, A5431) AES-GMAC: (A5423, A5428, A5430, A5432, A5435)
Authenticated Encryption/Decryption with AES CCM	BC-Auth	SP800-38C. Authenticated encryption, Authenticated decryption		AES-CCM: (A5422, A5427, A5429, A5431)
Authenticated Encryption/Decryption with AES GCM	BC-Auth	SP800-38D. Authenticated encryption, Authenticated decryption		AES-GCM: (A5423, A5428, A5430, A5432, A5435)
Message Digest with SHAKE	XOF	FIPS202. Message digest		SHAKE-128: (A5424) SHAKE-256: (A5424)
Signature Verification with RSA (legacy)	DigSig-SigVer	FIPS186-4. Legacy digital signature verification	Publications:FIPS 140-3 IG C.M legacy algorithms	RSA SigVer (FIPS186-4): (A5425, A5426, A5433, A5434)
Signature Verification with ECDSA (legacy)	DigSig-SigVer	FIPS186-4. Legacy digital signature verification	Publications:FIPS 140-3 IG C.M legacy algorithms	ECDSA SigVer (FIPS186-4): (A5425, A5426, A5433, A5434)

Table 9: Security Function Implementations

2.7 Algorithm Specific Information

2.7.1 GCM IV

The module offers three AES GCM implementations. The GCM IV generation for these implementations complies respectively with IG C.H under Scenario 1, Scenario 2, and Scenario 5. The GCM shall only be used in the context of the AES-GCM encryption executing under each scenario, and using the referenced APIs explained next.

Scenario 1, TLS 1.2

For TLS 1.2, the module offers the GCM implementation via the functions EVP_aead_aes_128_gcm_tls12() and EVP_aead_aes_256_gcm_tls12(), and uses the context of Scenario 1 of IG C.H. The module is compliant with SP800-52rev2 and the mechanism for IV generation is compliant with RFC5288. The module supports acceptable AES-GCM ciphersuites from Section 3.3.1 of SP800-52rev2.

The module explicitly ensures that the counter (the nonce_explicit part of the IV) does not exhaust the maximum number of possible values of 2^{64-1} for a given session key. If this 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.

Scenario 2, Random IV

In this implementation, the module offers the interfaces EVP_aead_aes_128_gcm_randnonce() and EVP_aead_aes_256_gcm_randnonce() for compliance with Scenario 2 of IG C.H and SP800-38D Section 8.2.2. The module generates the IV and then performs AES GCM encryption without outputting the IV to the calling application. The 96-bit AES-GCM IV, is generated randomly internal to the module using module's approved DRBG.

Scenario 5, TLS 1.3

For TLS 1.3, the module offers the AES-GCM implementation via the functions EVP_aead_aes_128_gcm_tls13() and EVP_aead_aes_256_gcm_tls13(), and uses the context of Scenario 5 of IG C.H. The protocol that provides this compliance is TLS 1.3, defined in RFC8446 of August 2018, using the ciphersuites that explicitly select AES-GCM as the encryption/decryption cipher (Appendix B.4 of RFC8446). The module supports acceptable AES-GCM ciphersuites from Section 3.3.1 of SP800-52rev2.

The module implements, within its boundary, an IV generation unit for TLS 1.3 that keeps control of the 64-bit counter value within the AES-GCM IV. If the exhaustion condition is observed, the module will return an error indication to the calling application, who will then need to either trigger a re-key of the session (i.e., a new key for AES-GCM), or terminate the connection.

In the event the module's power is lost and restored, the consuming application must ensure that new AES-GCM keys encryption or decryption under this scenario are established. TLS 1.3 provides session resumption, but the resumption procedure derives new AES-GCM encryption keys.

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. 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. To meet the requirement stated in IG C.I, the module implements a check to ensure that the two AES keys used in AES XTS mode are not identical.

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 with SP 800-132 and FIPS 140-3 IG D.N, the following requirements shall be met:

- Derived keys shall only be used in storage applications. The MK shall not be used for other purposes. The module accepts a minimum length of 112 bits for the MK or DPK.
- Passwords or passphrases, used as an input for the PBKDF2, shall not be used as cryptographic Keys.
- The minimum length of the password or passphrase accepted by the module is 14 characters. This results in the estimated probability of guessing the password to be at most 10^{-14}. Combined with the minimum iteration count as described below, this provides an acceptable trade-off between user experience and security against brute-force attacks.
- A portion of the salt, with a length of at least 128 bits (this is verified by the module to determine the service is approved), shall be generated randomly using the SP 800-90Ar1 DRBG provided by the module.
- The iteration count shall be selected as large as possible, if the time required to generate the key using the entered password is acceptable for the users. The module restricts the minimum iteration count to be 1000.

2.7.4 Compliance to SP 800-56ARev3 assurances

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

- The entity using the module, must use the module's "Key Pair Generation" service for generating ECDH ephemeral keys. This meets the assurances required by key pair owner defined in the section 5.6.2.1 of SP 800-56ARev3.
- As part of the module's shared secret computation (SSC) service, the module internally performs the public key validation on the peer's public key passed in as input to the SSC function. This meets the public key validity assurance required by the sections 5.6.2.2.1/5.6.2.2.2 of SP 800-56Arev3.
- The module does not support static keys therefore the "assurance of peer's possession of private key" is not applicable.

2.7.5 Approved Modulus Sizes for RSA Digital Signature

RSA SigGen (FIPS 186-5) has been CAVP tested with all the supported RSA modulus lengths (i.e., 2048, 3072, 4096). This is documented in the Approved Algorithms table of the Security Policy. All modulus sizes for SigVer have also been CAVP tested. There is no RSA signature with keys for which CAVP testing is not available. The minimum number of the Miller-Rabin tests used in primality testing complies with Table B.1 in FIPS 186-5. The RSA SigVer (FIPS 186-4) and (FIPS 186-5) have been CAVP tested with the modulus sizes of

1024 (FIPS 186-4) and 2048, 3072, 4096 (FIPS 186-5). All modulus sizes in which testing is available have been tested by the CAVP.

2.7.6 Legacy Algorithms

The cryptographic module implements the following cryptographic algorithms for legacy use. Algorithms designated as “Legacy” can only be used on data that was generated prior to the Legacy Date specified in FIPS 140-3 IG C.M:

- RSA SigVer (FIPS 186-4) with 1024-bit keys.
- RSA SigVer (FIPS 186-4) with SHA-1.
- ECDSA SigVer (FIPS 186-4) with SHA-1.

SHA-1 is only approved when used for message digest and for signature verification as a legacy option. The use of SHA-1 for digital signature generation is non-approved. Starting January 1 of 2031 SHA-1 will be non-approved for all purposes.

2.7.7 Authenticated Encryption/Decryption

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

2.7.8 KAS-SSC

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

2.8 RBG and Entropy

The module provides an SP800-90Arev1-compliant Deterministic Random Bit Generator (DRBG) using CTR_DRBG mechanism with AES-256, with a derivation function, for generation of key components of asymmetric keys, and random number generation. The DRBG is seeded with 256-bit of entropy input provided from an external entity to the module. This corresponds to scenario 2 (b) of IG 9.3.A i.e., the DRBG that receives a LOAD command from external entropy source outside of module's cryptographic boundary. The calling application shall use an entropy source that meets the security strength required for the CTR_DRBG as shown in NIST SP 800-90Arev1, Table 3 and should return an error if minimum strength cannot be met.

Per the IG 9.3.A requirement, the module includes the caveat "No assurance of the minimum strength of generated keys (e.g., keys)".

2.9 Key Generation

The key generation methods implemented by the module are specified in the *Vendor-Affirmed Algorithms* table. The key derivation methods implemented by the module are specified in the *Security Function Implementations* table.

2.10 Key Establishment

The key establishment methods implemented by the module are specified in the *Security Function Implementations* table.

2.11 Industry Protocols

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

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

No parts of the SSH, TLS, 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 for data.
N/A	Data Output	API output parameters for data.
N/A	Control Input	API function calls.
N/A	Status Output	API return codes, error message.

Table 10: Ports and Interfaces

The module does not implement the Control Output interface.

4 Roles, Services, and Authentication

4.1 Authentication Methods

N/A for this module.

4.2 Roles

Name	Type	Operator Type	Authentication Methods
Crypto Officer	Role	CO	None

Table 11: Roles

The module does not support concurrent operators.

4.3 Approved Services

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Encryption	Encryption	Return value 1 from the function: FIPS_service_indicator_check_approved()	AES key, plaintext	Ciphertext	Encryption/Decryption with AES	Crypto Officer - AES Key: W,E
Decryption	Decryption	Return value 1 from the function: FIPS_service_indicator_check_approved()	AES key, ciphertext	Plaintext	Encryption/Decryption with AES	Crypto Officer - AES Key: W,E
Authenticated Encryption	Authenticated Encryption	Return value 1 from the function: FIPS_service_indicator_check_approved()	AES key, plaintext, IV	Ciphertext, MAC tag	Authenticated Encryption/Decryption with AES KW, AES-KWP Authenticated Encryption/Decryption with AES CCM Authenticated Encryption/Decryption with AES GCM	Crypto Officer - AES Key: W,E
Authenticated Decryption	Authenticated Decryption	Return value 1 from the function: FIPS_service_indicator_check_approved()	AES key, ciphertext, IV, MAC tag	Plaintext or fail	Authenticated Encryption/Decryption with AES KW, AES-KWP Authenticated Encryption/Decryption with AES CCM Authenticated Encryption/Decryption with AES GCM	Crypto Officer - AES Key: W,E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Message Authentication Generation	MAC computation	Return value 1 from the function: FIPS_service_indicator_check_approved()	AES key or HMAC key, message	MAC tag	Message Authentication Generation with HMAC Message Authentication Generation with AES	Crypto Officer - HMAC Key: W,E - AES Key: W,E
Message Digest	Generating message digest	Return value 1 from the function: FIPS_service_indicator_check_approved()	Message	Message digest	Message Digest with SHA Message Digest with SHAKE	Crypto Officer
Random Number Generation	Generating random numbers	Return value 1 from the function: FIPS_service_indicator_check_approved()	Output length	Random bytes	Random Number Generation with DRBG	Crypto Officer - Entropy Input: W,E - DRBG Seed: G,W,E - DRBG Internal State (V, Key): G,W,E
Key Generation	Generating a key pair	Return value 1 from the function: FIPS_service_indicator_check_approved()	Modulus size / Curve	RSA public key, RSA private key / EC public key, EC private key	Key Generation with RSA Key Generation with ECDSA	Crypto Officer - RSA Public Key: G,R - RSA Private Key: G,R - EC Public Key: G,R - EC Private Key: G,R - Intermediate Key Generation Value: G,E,Z
Key Verification	Verifying the public key	Return value 1 from the function: FIPS_service_indicator_check_approved()	Public key	Success/ error	Key Verification with ECDSA	Crypto Officer - EC Public Key: W,E
Signature Generation	Generating signature	Return value 1 from the function: FIPS_	Message, EC private	Digital signature	Signature Generation with RSA	Crypto Officer - RSA

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
		service_indicator_check_approved( )	key or RSA private key, hash algorithm		Signature Generation with ECDSA	Private Key: W,E - EC Private Key: W,E
Signature Verification	Verifying signature	Return value 1 from the function: FIPS_ service_indicator_check_approved( )	Signature, EC public key or RSA public key, hash algorithm	Digital signature verification result	Signature Verification with ECDSA Signature Verification with RSA Signature Verification with RSA (legacy) Signature Verification with ECDSA (legacy)	Crypto Officer - RSA Public Key: W,E - EC Public Key: W,E
Shared Secret Computation	Calculating the Shared Secret	Return value 1 from the function: FIPS_ service_indicator_check_approved( )	EC public key, EC private key	Shared Secret	Shared Secret Computation with EC Diffie-Hellman	Crypto Officer - EC Public Key: W,E - EC Private Key: W,E - Shared Secret: G,R
Key Derivation with TLS KDF	Deriving Keys	Return value 1 from the function: FIPS_ service_indicator_check_approved( )	TLS Pre-Master Secret, key length	TLS Derived Key (AES/HMAC)	Key Derivation with TLS KDF	Crypto Officer - TLS Pre-Master Secret: W,E - TLS Master Secret: G,E,Z - TLS Derived Key (AES/HMAC): G,R
Key Derivation with PBKDF	Deriving Keys	Return value 1 from the function: FIPS_ service_indicator_check_approved( )	Password, salt, iteration count, key length	PBKDF Derived Key	Key Derivation with PBKDF	Crypto Officer - PBKDF Derived Key: G,R - Password: W,E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Key Derivation with KDA HKDF	Deriving Keys	Return value 1 from the function: FIPS_service_indicator_check_approved()	Shared Secret, Key Length	HKDF Derived Key	Key Derivation with KDA HKDF	Crypto Officer - HKDF Derived Key: G,R - Shared Secret: W,E - TLS Master Secret: W,E,Z
Zeroization	Zeroize SSP in volatile memory	N/A	SSP	N/A	None	Crypto Officer - AES Key: Z - HMAC Key: Z - Entropy Input: Z - DRBG Seed: Z - DRBG Internal State (V, Key): Z - RSA Public Key: Z - RSA Private Key: Z - EC Public Key: Z - EC Private Key: Z - Shared Secret: Z - TLS Pre-Master Secret: Z - TLS Master Secret: Z - TLS Derived Key (AES/HMAC): Z - HKDF Derived Key: Z

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						- SSH Derived Key: Z - PBKDF Derived Key: Z - Password: Z - Intermediate Key Generation Value: Z
Key Derivation with SSH KDF	Deriving Keys	Return value 1 from the function: FIPS_service_indicator_check_approved()	Shared Secret, Key Length	SSH Derived Key	Key Derivation with SSH KDF	Crypto Officer - Shared Secret: W,E - SSH Derived Key: G,R
Show Status	Show status of the module state	N/A	N/A	Module status	None	Crypto Officer
Show Version	Show the version of the module using awslc_version_string	N/A	N/A	Module name and version	None	Crypto Officer
On-Demand Self-test	Initiate cryptographic algorithms self-tests and integrity test on-demand.	N/A	N/A	Pass or fail	Shared Secret Computation with EC Diffie-Hellman Authenticated Encryption/Decryption with AES KW, AES-KWP Encryption/Decryption with AES Signature Generation with RSA Signature Generation with ECDSA Key Generation with RSA Key Generation with ECDSA Signature Verification with ECDSA Signature Verification with	Crypto Officer

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
					RSA Key Verification with ECDSA Key Derivation with TLS KDF Key Derivation with SSH KDF Key Derivation with KDA HKDF Key Derivation with PBKDF Message Digest with SHA Random Number Generation with DRBG Message Authentication Generation with HMAC Message Authentication Generation with AES Authenticated Encryption/Decryption with AES CCM Authenticated Encryption/Decryption with AES GCM Message Digest with SHAKE	

Table 12: Approved Services

For the above table, the convention below applies when specifying the access permissions (types) that the service has for each SSP.

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

For the role, CO indicates “Crypto Officer”.

The module implements a service indicator that indicates whether the invoked service is approved. The service indicator is a return value 1 from the FIPS_service_indicator_check_approved function. This function is used together with two other functions. The usage is as follows:

- STEP 1: Should be called before invoking the service.
  int before = FIPS_service_indicator_before_call();
- STEP 2: Make a service call i.e., API function for performing a service.
  Func();
- STEP 3: Should be called after invoking the service.
  int after = FIPS_service_indicator_after_call();
- STEP 4: Return value 1 indicates approved service was invoked.
  int ret = FIPS_service_indicator_check_approved(before, after);

Alternatively, all the above steps can be done by using a single call using the function CALL_SERVICE_AND_CHECK_APPROVED(approved, func).

4.4 Non-Approved Services

Name	Description	Algorithms	Role
Encryption	Encryption	AES with OFB or CFB1, CFB8 modes AES GCM, GMAC, XTS with keys not listed in Table 5 AES using aes_*_generic function AES GMAC using aes_*_generic RSA encryption primitive with PKCS#1 v1.5 and OAEP padding	CO
Decryption	Decryption	AES with OFB or CFB1, CFB8 modes AES GCM, GMAC, XTS with keys not listed in Table 5 AES using aes_*_generic function AES GMAC using aes_*_generic	CO
Message Authentication Generation	MAC computation	AES using aes_*_generic function HMAC-MD4, HMAC-MD5, HMAC-SHA-3, HMAC-RIPEMD-160	CO
Message Digest	Generating message digest	MD4 MD5 (outside of TLS) RIPEMD-160	CO
Signature Generation	Generating signatures	RSA using keys less than 2048 bits RSA without hashing SHA-1, SHA-3	CO
Signature Verification	Verifying signatures	RSA using keys less than 1024 bits RSA without hashing	CO
Key Generation	Generating key pair	RSA using RSA_generate_key_ex ECDSA using EC_KEY_generate_key	CO
Shared Secret Computation	Calculating shared secret	Curve secp256k1 Diffie Hellman	CO
Key Derivation	Deriving TLS keys	TLS KDF using any SHA algorithms other than SHA2-256, SHA2-384, SHA2-512; or TLS KDF using non-extended master secret	CO
Key Encapsulation	Encrypting a key	RSA	CO
Key Un-encapsulation	Decrypting a key	RSA	CO

Table 13: Non-Approved Services

4.5 External Software/Firmware Loaded

Not applicable.

5 Software/Firmware Security

5.1 Integrity Techniques

The integrity of the module is verified by comparing a HMAC value calculated at run time on the bcm.o file, with the HMAC-SHA2-256 value stored within the module that was computed at build time. The HMAC key for the integrity verification is embedded in the module.

5.2 Initiate on Demand

The module provides on-demand integrity test. The integrity test can be performed on demand by reloading the module. Additionally, the integrity test can be performed using the On-Demand Integrity Test service, which calls the BORINGSSL_integrity_test function.

6 Operational Environment

6.1 Operational Environment Type and Requirements

Type of Operational Environment: Modifiable

How Requirements are Satisfied:

The module runs on a commercially available general-purpose operating system executing on the hardware specified in section 2.

The module shall be compiled and installed as stated in section 11. The Crypto Officer shall confirm that the module is installed correctly by following steps listed in section 11.1.

6.2 Configuration Settings and Restrictions

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 software only; therefore, this section is not applicable.

8 Non-Invasive Security

The module does not implement any non-invasive security mechanisms; 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. The module does not perform persistent storage of SSPs	Dynamic

Table 14: Storage Areas

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 15: SSP Input-Output Methods

The module does not support entry and output of SSPs beyond the physical perimeter of the operational environment. The SSPs are provided to the module via API input parameters in the plaintext form and output via API output parameters in the plaintext form to and from the calling application running on the same operational environment.

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 zeros, which renders the SSP values irretrievable. The successful completion of the zeroization routine indicates that the zeroization procedure succeeded.	By calling the appropriate zeroization functions: OpenSSL_cleanse, EVP_CIPHER_CTX_cleanup, EVP_AEAD_CTX_zero, HMAC_CTX_cleanup, CTR_DRBG_clear, RSA_free, EC_KEY_free
Module Reset	De-allocates the volatile memory used to store SSPs	Volatile memory used by the module is overwritten within nanoseconds when power is removed. The successful completion of the removal of power from the module indicates that zeroization has completed.	By unloading and reloading the module.
Automatically	Automatically zeroized when no longer needed	Memory occupied by SSPs is overwritten with zeros, which renders the SSP values irretrievable. The successful completion of the running service indicates that zeroization has completed.	N/A

Table 16: 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	128-256 bits - 128-256 bits	Symmetric key - CSP			Authenticated Encryption/Decryption with AES KW, AES-KWP Encryption/Decryption with AES Message Authentication Generation with AES Authenticated Encryption/Decryption with AES CCM Authenticated Encryption/Decryption with AES GCM
HMAC Key	HMAC key for Message Authentication Generation	112-524288 bits - 112-256 bits	Authentication key - CSP			Message Authentication Generation with HMAC
Entropy Input	Entropy input used to seed the DRBGs	256 bits - 256 bits	Entropy - CSP			Random Number Generation with DRBG
DRBG Seed	DRBG seed derived from entropy input as defined in SP 800-90Ar1	256 bits - 256 bits	DRBG seed - CSP	Random Number Generation with DRBG		Random Number Generation with DRBG
DRBG Internal State (V, Key)	Internal state of CTR_DRBG	V: 128 bits, Key: 256 bits - 256 bits	Internal state - CSP	Random Number Generation with DRBG		Random Number Generation with DRBG
RSA Public Key	RSA public key used for RSA key generation, signature verification	1024, 2048, 3072, 4096 bits - 80-150 bits	Public key - PSP	Key Generation with RSA		Signature Verification with RSA Signature Verification with RSA (legacy)
RSA Private Key	RSA private key used for RSA key generation, signature generation	2048, 3072, 4096 bits - 112-150 bits	Private key - CSP	Key Generation with RSA		Signature Generation with RSA
EC Public Key	EC public key used for EC key generation, key verification,	P-224, P-256, P-384, P-521 - 112-256 bits	Public key - PSP	Key Generation with ECDSA		Shared Secret Computation with EC Diffie-Hellman Signature Verification

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
	signature verification, shared secret computation					with ECDSA Key Verification with ECDSA Signature Verification with ECDSA (legacy)
EC Private Key	EC private key used for EC key generation, key verification, signature generation, shared secret computation	P-224, P-256, P-384, P-521 - 112-256 bits	Private key - CSP	Key Generation with ECDSA		Shared Secret Computation with EC Diffie-Hellman Signature Generation with ECDSA
Shared Secret	Shared Secret generated by KAS-ECC-SSC	P-224, P-256, P-384, P-521 - 112-256 bits	Shared secret - CSP		Shared Secret Computation with EC Diffie-Hellman	Key Derivation with TLS KDF Key Derivation with SSH KDF Key Derivation with KDA HKDF
TLS Pre-Master Secret	TLS Pre-Master secret used for deriving the TLS Master Secret	P-224, P-256, P-384, P-521 - 112-256 bits	TLS pre-master secret - CSP			Key Derivation with TLS KDF Key Derivation with KDA HKDF
TLS Master Secret	TLS Master secret used for deriving the TLS Derived Key	384 bits - 112-256 bits	TLS master secret - CSP	Key Derivation with TLS KDF Key Derivation with KDA HKDF		Key Derivation with TLS KDF Key Derivation with KDA HKDF
TLS Derived Key (AES/HMAC)	TLS Derived Key from TLS Master Secret	AES: 128-256 bits HMAC: 112 to 256 bits - AES: 128-256 bits HMAC: 112 to 256 bits	Symmetric key - CSP	Key Derivation with TLS KDF		
HKDF Derived Key	KDA HKDF derived key	2048 bits - 112-256 bits	Symmetric key - CSP	Key Derivation with KDA HKDF		
SSH Derived Key	SSH KDF derived key	128 to 512 bits - 112 to 256 bits	Symmetric key - CSP	Key Derivation with SSH KDF		
PBKDF Derived Key	PBKDF derived key	128 to 4096 bits - N/A	Symmetric key - CSP	Key Derivation with PBKDF		

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
Password	Password for PBKDF	14-128 characters - N/A	Password - CSP			Key Derivation with PBKDF
Intermediate Key Generation Value	Intermediate key generation value	224-4096 bits - 112-256 bits	Intermediate value - CSP	Key Generation with RSA Key Generation with ECDSA		Key Generation with RSA Key Generation with ECDSA

Table 17: SSP Table 1

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
AES Key	API input parameters	RAM:Plaintext	From service invocation to service completion	Free Cipher Handle Module Reset	
HMAC Key	API input parameters	RAM:Plaintext	From service invocation to service completion	Free Cipher Handle Module Reset	
Entropy Input	API input parameters	RAM:Plaintext	From service invocation to service completion	Module Reset Automatically	DRBG Seed:Generation OF
DRBG Seed		RAM:Plaintext	From service invocation to service completion	Module Reset Automatically	Entropy Input:Derived From DRBG Internal State (V, Key):Generation Of
DRBG Internal State (V, Key)		RAM:Plaintext	From service invocation to service completion	Free Cipher Handle Module Reset	DRBG Seed:Derived From
RSA Public Key	API input parameters API output parameters	RAM:Plaintext	From service invocation to service completion	Free Cipher Handle Module Reset	RSA Private Key:Paired With Intermediate Key Generation Value:Generated From
RSA Private Key	API input parameters API output parameters	RAM:Plaintext	From service invocation to service completion	Free Cipher Handle Module Reset	RSA Public Key:Paired With Intermediate Key Generation Value:Generated From
EC Public Key	API input parameters API output parameters	RAM:Plaintext	From service invocation to service completion	Free Cipher Handle Module Reset	EC Private Key:Paired With Shared Secret:Generation Of Intermediate Key Generation Value:Generated From
EC Private Key	API input parameters API output parameters	RAM:Plaintext	From service invocation to service completion	Free Cipher Handle Module Reset	EC Public Key:Paired With Shared Secret:Generation Of Intermediate Key Generation Value:Generated From

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
Shared Secret	API input parameters API output parameters	RAM:Plaintext	From service invocation to service completion	Free Cipher Handle Module Reset	EC Public Key:Derived From EC Private Key:Derived From
TLS Pre-Master Secret	API input parameters	RAM:Plaintext	From service invocation to service completion	Free Cipher Handle Module Reset	TLS Master Secret:Derivation Of
TLS Master Secret		RAM:Plaintext	From service invocation to service completion	Free Cipher Handle Module Reset	TLS Pre-Master Secret:Derived From TLS Derived Key (AES/HMAC):Derivation Of
TLS Derived Key (AES/HMAC)	API output parameters	RAM:Plaintext	From service invocation to service completion	Free Cipher Handle Module Reset	TLS Master Secret:Derived From
HKDF Derived Key	API output parameters	RAM:Plaintext	From service invocation to service completion	Free Cipher Handle Module Reset	Shared Secret:Derived From
SSH Derived Key	API output parameters	RAM:Plaintext	From service invocation to service completion	Free Cipher Handle Module Reset	Shared Secret:Derived From
PBKDF Derived Key	API output parameters	RAM:Plaintext	From service invocation to service completion	Free Cipher Handle Module Reset	Password:Derived From
Password	API input parameters	RAM:Plaintext	From service invocation to service completion	Free Cipher Handle Module Reset	PBKDF Derived Key:Derivation Of
Intermediate Key Generation Value		RAM:Plaintext	From service invocation to service completion	Module Reset Automatically	RSA Public Key:Generation Of RSA Private Key:Generation Of EC Public Key:Generation Of EC Private Key:Generation Of

Table 18: 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 (A5425)	SHA2-256	Message Authentication	SW/FW Integrity	Module becomes operational	Integrity test for bcm.o
HMAC-SHA2-256 (A5433)	SHA2-256	Message Authentication	SW/FW Integrity	Module becomes operational	Integrity test for bcm.o
HMAC-SHA2-256 (A5434)	SHA2-256	Message Authentication	SW/FW Integrity	Module becomes operational	Integrity test for bcm.o

Table 19: Pre-Operational Self-Tests

The module performs the pre-operational self-test automatically when the module is loaded into memory; the pre-operational self-test is the software integrity test that ensures that the module is not corrupted. While the module is executing the pre-operational self-test, services are not available, and input and output are inhibited. If the pre-operational self-test fails, the module transitions to the Error state.

The software integrity test is performed after a set of conditional cryptographic algorithm self-tests (CASTs). The set of CASTs includes the self-test for HMAC-SHA2-256 algorithm used in the pre-operational self-test.

10.2 Conditional Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-CBC (A5422)	128, 192, 256 bit keys, encrypt/decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Power up
AES-CBC (A5427)	128, 192, 256 bit keys, encrypt/decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Power up
AES-CBC (A5429)	128, 192, 256 bit keys, encrypt/decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Power up
AES-CBC (A5431)	128, 192, 256 bit keys, encrypt/decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Power up
AES-GCM (A5423)	128, 192, 256 bit keys, 96-bit (internal/external IV), encrypt/decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Power up
AES-GCM (A5428)	128, 192, 256 bit keys, 96-bit (internal/external IV), encrypt/decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Power up
AES-GCM (A5430)	128, 192, 256 bit keys, 96-bit (internal/external IV), encrypt/decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Power up

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-GCM (A5432)	128, 192, 256 bit keys, 96-bit (internal/external IV), encrypt/decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Power up
AES-GCM (A5435)	128, 192, 256 bit keys, 96-bit (internal/external IV), encrypt/decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Power up
SHA-1 (A5425)	SHA-1	KAT	CAST	Module becomes operational	Message digest	Power up
SHA-1 (A5426)	SHA-1	KAT	CAST	Module becomes operational	Message digest	Power up
SHA-1 (A5433)	SHA-1	KAT	CAST	Module becomes operational	Message digest	Power up
SHA-1 (A5434)	SHA-1	KAT	CAST	Module becomes operational	Message digest	Power up
SHA2-256 (A5425)	SHA2-256	KAT	CAST	Module becomes operational	Message digest	Power up
SHA2-256 (A5433)	SHA2-256	KAT	CAST	Module becomes operational	Message digest	Power up
SHA2-256 (A5434)	SHA2-256	KAT	CAST	Module becomes operational	Message digest	Power up
SHA2-512 (A5425)	SHA2-512	KAT	CAST	Module becomes operational	Message digest	Power up
SHA2-512 (A5433)	SHA2-512	KAT	CAST	Module becomes operational	Message digest	Power up
SHA2-512 (A5434)	SHA2-512	KAT	CAST	Module becomes operational	Message digest	Power up
HMAC-SHA2-256 (A5425)	SHA2-256	KAT	CAST	Module becomes operational	Message authentication	Power up
HMAC-SHA2-256 (A5433)	SHA2-256	KAT	CAST	Module becomes operational	Message authentication	Power up
HMAC-SHA2-256 (A5434)	SHA2-256	KAT	CAST	Module becomes operational	Message authentication	Power up

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
Counter DRBG (A5422)	256 bit keys, with PR	KAT	CAST	Module becomes operational	Health test per section 11.3 of SP 800-90Ar1	Power up
Counter DRBG (A5427)	256 bit keys, with PR	KAT	CAST	Module becomes operational	Health test per section 11.3 of SP 800-90Ar1	Power up
Counter DRBG (A5429)	256 bit keys, with PR	KAT	CAST	Module becomes operational	Health test per section 11.3 of SP 800-90Ar1	Power up
Counter DRBG (A5431)	256 bit keys, with PR	KAT	CAST	Module becomes operational	Health test per section 11.3 of SP 800-90Ar1	Power up
ECDSA SigGen (FIPS186-5) (A5425)	SHA2-256, P-256 curve	KAT	CAST	Module becomes operational	Digital signature generation	Signature Generation or Key Generation service request
ECDSA SigGen (FIPS186-5) (A5426)	SHA2-256, P-256 curve	KAT	CAST	Module becomes operational	Digital signature generation	Signature Generation or Key Generation service request
ECDSA SigGen (FIPS186-5) (A5433)	SHA2-256, P-256 curve	KAT	CAST	Module becomes operational	Digital signature generation	Signature Generation or Key Generation service request
ECDSA SigGen (FIPS186-5) (A5434)	SHA2-256, P-256 curve	KAT	CAST	Module becomes operational	Digital signature generation	Signature Generation or Key Generation service request
ECDSA SigVer (FIPS186-5) (A5425)	SHA2-256, P-256 curve	KAT	CAST	Module becomes operational	Digital signature verification	Signature verification or Key Generation service request
ECDSA SigVer (FIPS186-5) (A5426)	SHA2-256, P-256 curve	KAT	CAST	Module becomes operational	Digital signature verification	Signature verification or Key Generation service request
ECDSA SigVer (FIPS186-5) (A5433)	SHA2-256, P-256 curve	KAT	CAST	Module becomes operational	Digital signature verification	Signature verification or Key Generation service request
ECDSA SigVer (FIPS186-5) (A5434)	SHA2-256, P-256 curve	KAT	CAST	Module becomes operational	Digital signature verification	Signature verification or Key Generation service request
RSA SigGen (FIPS186-5) (A5425)	PKCS#1 v1.5 with 2048 bit key and SHA2-256	KAT	CAST	Module becomes operational	Digital signature generation	Signature Generation or Key Generation service request
RSA SigGen (FIPS186-5) (A5426)	PKCS#1 v1.5 with 2048 bit key and SHA2-256	KAT	CAST	Module becomes operational	Digital signature generation	Signature Generation or Key Generation service request
RSA SigGen (FIPS186-5) (A5433)	PKCS#1 v1.5 with 2048 bit key and SHA2-256	KAT	CAST	Module becomes operational	Digital signature generation	Signature Generation or Key Generation service request

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
RSA SigGen (FIPS186-5) (A5434)	PKCS#1 v1.5 with 2048 bit key and SHA2-256	KAT	CAST	Module becomes operational	Digital signature generation	Signature Generation or Key Generation service request
RSA SigVer (FIPS186-5) (A5425)	PKCS#1 v1.5 with 2048 bit key and SHA2-256	KAT	CAST	Module becomes operational	Digital signature verification	Signature Verification or Key Generation service request
RSA SigVer (FIPS186-5) (A5426)	PKCS#1 v1.5 with 2048 bit key and SHA2-256	KAT	CAST	Module becomes operational	Digital signature verification	Signature Verification or Key Generation service request
RSA SigVer (FIPS186-5) (A5433)	PKCS#1 v1.5 with 2048 bit key and SHA2-256	KAT	CAST	Module becomes operational	Digital signature verification	Signature Verification or Key Generation service request
RSA SigVer (FIPS186-5) (A5434)	PKCS#1 v1.5 with 2048 bit key and SHA2-256	KAT	CAST	Module becomes operational	Digital signature verification	Signature Verification or Key Generation service request
KAS-ECC-SSC Sp800-56Ar3 (A5425)	P-256 curve	KAT	CAST	Module becomes operational	Shared secret computation	Shared secret computation service request
KAS-ECC-SSC Sp800-56Ar3 (A5426)	P-256 curve	KAT	CAST	Module becomes operational	Shared secret computation	Shared secret computation service request
KAS-ECC-SSC Sp800-56Ar3 (A5433)	P-256 curve	KAT	CAST	Module becomes operational	Shared secret computation	Shared secret computation service request
KAS-ECC-SSC Sp800-56Ar3 (A5434)	P-256 curve	KAT	CAST	Module becomes operational	Shared secret computation	Shared secret computation service request
KDF TLS (A5425)	SHA2-256	KAT	CAST	Module becomes operational	Key derivation	Power up
KDF TLS (A5426)	SHA2-256	KAT	CAST	Module becomes operational	Key derivation	Power up
KDF TLS (A5433)	SHA2-256	KAT	CAST	Module becomes operational	Key derivation	Power up
KDF TLS (A5434)	SHA2-256	KAT	CAST	Module becomes operational	Key derivation	Power up
KDA HKDF Sp800-56Cr1 (A5425)	SHA2-256	KAT	CAST	Module becomes operational	Shared secret key derivation	Power up
KDA HKDF Sp800-56Cr1 (A5426)	SHA2-256	KAT	CAST	Module becomes operational	Shared secret key derivation	Power up

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
KDA HKDF Sp800-56Cr1 (A5433)	SHA2-256	KAT	CAST	Module becomes operational	Shared secret key derivation	Power up
KDA HKDF Sp800-56Cr1 (A5434)	SHA2-256	KAT	CAST	Module becomes operational	Shared secret key derivation	Power up
PBKDF (A5425)	SHA2-256	KAT	CAST	Module becomes operational	Password-based key derivation	Power up
PBKDF (A5426)	SHA2-256	KAT	CAST	Module becomes operational	Password-based key derivation	Power up
PBKDF (A5433)	SHA2-256	KAT	CAST	Module becomes operational	Password-based key derivation	Power up
PBKDF (A5434)	SHA2-256	KAT	CAST	Module becomes operational	Password-based key derivation	Power up
ECDSA KeyGen (FIPS186-5) (A5425)	SHA2-256	PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
ECDSA KeyGen (FIPS186-5) (A5426)	SHA2-256	PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
ECDSA KeyGen (FIPS186-5) (A5433)	SHA2-256	PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
ECDSA KeyGen (FIPS186-5) (A5434)	SHA2-256	PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
RSA KeyGen (FIPS186-5) (A5425)	SHA2-256	PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
RSA KeyGen (FIPS186-5) (A5426)	SHA2-256	PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
RSA KeyGen (FIPS186-5) (A5433)	SHA2-256	PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
RSA KeyGen (FIPS186-5) (A5434)	SHA2-256	PCT	PCT	Successful key pair generation	Signature generation & verification	Key pair generation
SHA3-256 (A5424)	SHA3-256	KAT	CAST	Module becomes operational	Message digest	Power up

Table 20: Conditional Self-Tests

Data output through the data output interface is inhibited during the self-tests.

10.3 Periodic Self-Test Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
HMAC-SHA2-256 (A5425)	Message Authentication	SW/FW Integrity	On demand	Manually
HMAC-SHA2-256 (A5433)	Message Authentication	SW/FW Integrity	On demand	Manually
HMAC-SHA2-256 (A5434)	Message Authentication	SW/FW Integrity	On demand	Manually

Table 21: Pre-Operational Periodic Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
AES-CBC (A5422)	KAT	CAST	On Demand	Manually
AES-CBC (A5427)	KAT	CAST	On Demand	Manually
AES-CBC (A5429)	KAT	CAST	On Demand	Manually
AES-CBC (A5431)	KAT	CAST	On Demand	Manually
AES-GCM (A5423)	KAT	CAST	On Demand	Manually
AES-GCM (A5428)	KAT	CAST	On Demand	Manually
AES-GCM (A5430)	KAT	CAST	On Demand	Manually
AES-GCM (A5432)	KAT	CAST	On Demand	Manually
AES-GCM (A5435)	KAT	CAST	On Demand	Manually
SHA-1 (A5425)	KAT	CAST	On Demand	Manually
SHA-1 (A5426)	KAT	CAST	On Demand	Manually
SHA-1 (A5433)	KAT	CAST	On Demand	Manually
SHA-1 (A5434)	KAT	CAST	On Demand	Manually
SHA2-256 (A5425)	KAT	CAST	On Demand	Manually
SHA2-256 (A5433)	KAT	CAST	On Demand	Manually
SHA2-256 (A5434)	KAT	CAST	On Demand	Manually
SHA2-512 (A5425)	KAT	CAST	On Demand	Manually
SHA2-512 (A5433)	KAT	CAST	On Demand	Manually
SHA2-512 (A5434)	KAT	CAST	On Demand	Manually
HMAC-SHA2-256 (A5425)	KAT	CAST	On Demand	Manually
HMAC-SHA2-256 (A5433)	KAT	CAST	On Demand	Manually
HMAC-SHA2-256 (A5434)	KAT	CAST	On Demand	Manually
Counter DRBG (A5422)	KAT	CAST	On Demand	Manually
Counter DRBG (A5427)	KAT	CAST	On Demand	Manually
Counter DRBG (A5429)	KAT	CAST	On Demand	Manually
Counter DRBG (A5431)	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
ECDSA SigGen (FIPS186-5) (A5425)	KAT	CAST	On Demand	Manually
ECDSA SigGen (FIPS186-5) (A5426)	KAT	CAST	On Demand	Manually
ECDSA SigGen (FIPS186-5) (A5433)	KAT	CAST	On Demand	Manually
ECDSA SigGen (FIPS186-5) (A5434)	KAT	CAST	On Demand	Manually
ECDSA SigVer (FIPS186-5) (A5425)	KAT	CAST	On Demand	Manually
ECDSA SigVer (FIPS186-5) (A5426)	KAT	CAST	On Demand	Manually
ECDSA SigVer (FIPS186-5) (A5433)	KAT	CAST	On Demand	Manually
ECDSA SigVer (FIPS186-5) (A5434)	KAT	CAST	On Demand	Manually
RSA SigGen (FIPS186-5) (A5425)	KAT	CAST	On Demand	Manually
RSA SigGen (FIPS186-5) (A5426)	KAT	CAST	On Demand	Manually
RSA SigGen (FIPS186-5) (A5433)	KAT	CAST	On Demand	Manually
RSA SigGen (FIPS186-5) (A5434)	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-5) (A5425)	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-5) (A5426)	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-5) (A5433)	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-5) (A5434)	KAT	CAST	On Demand	Manually
KAS-ECC-SSC Sp800-56Ar3 (A5425)	KAT	CAST	On Demand	Manually
KAS-ECC-SSC Sp800-56Ar3 (A5426)	KAT	CAST	On Demand	Manually
KAS-ECC-SSC Sp800-56Ar3 (A5433)	KAT	CAST	On Demand	Manually
KAS-ECC-SSC Sp800-56Ar3 (A5434)	KAT	CAST	On Demand	Manually
KDF TLS (A5425)	KAT	CAST	On Demand	Manually
KDF TLS (A5426)	KAT	CAST	On Demand	Manually
KDF TLS (A5433)	KAT	CAST	On Demand	Manually
KDF TLS (A5434)	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
KDA HKDF Sp800-56Cr1 (A5425)	KAT	CAST	On Demand	Manually
KDA HKDF Sp800-56Cr1 (A5426)	KAT	CAST	On Demand	Manually
KDA HKDF Sp800-56Cr1 (A5433)	KAT	CAST	On Demand	Manually
KDA HKDF Sp800-56Cr1 (A5434)	KAT	CAST	On Demand	Manually
PBKDF (A5425)	KAT	CAST	On Demand	Manually
PBKDF (A5426)	KAT	CAST	On Demand	Manually
PBKDF (A5433)	KAT	CAST	On Demand	Manually
PBKDF (A5434)	KAT	CAST	On Demand	Manually
ECDSA KeyGen (FIPS186-5) (A5425)	PCT	PCT	On Demand	Manually
ECDSA KeyGen (FIPS186-5) (A5426)	PCT	PCT	On Demand	Manually
ECDSA KeyGen (FIPS186-5) (A5433)	PCT	PCT	On Demand	Manually
ECDSA KeyGen (FIPS186-5) (A5434)	PCT	PCT	On Demand	Manually
RSA KeyGen (FIPS186-5) (A5425)	PCT	PCT	On Demand	Manually
RSA KeyGen (FIPS186-5) (A5426)	PCT	PCT	On Demand	Manually
RSA KeyGen (FIPS186-5) (A5433)	PCT	PCT	On Demand	Manually
RSA KeyGen (FIPS186-5) (A5434)	PCT	PCT	On Demand	Manually
SHA3-256 (A5424)	KAT	CAST	On Demand	Manually

Table 22: Conditional Periodic Information

10.4 Error States

Name	Description	Conditions	Recovery Method	Indicator
Error	The library is aborted with SIGABRT signal. Module is no longer operational the data output interface is inhibited	Pre-operational test failure; CAST failure	Module reset	Pre-operational test failure: an error message is output (i.e., "FIPS integrity test failed.") on the stderr and then the module is aborted. Conditional test failure: an error message indicating which KAT failed (e.g., "* KAT failed") is output on the stderr and then the module is aborted.
PCT Error	The library is aborted with SIGABRT signal. Module is no longer	PCT failure	Module reset	An error message is output in the error queue (e.g., EC_R_PUBLIC_KEY_VALIDATION_FAILED, RSA_R_PUBLIC_KEY_VALIDATION_FAILED) and then the module

Name	Description	Conditions	Recovery Method	Indicator
	operational the data output interface is inhibited			generates a new key, if the PCT still does not pass, eventually the module will be aborted after 5 tries.

Table 23: Error States

In the error states, the output interface is inhibited. If the module fails any of the self-tests, the module enters an error state. To recover from any error state, the module must be rebooted.

10.5 Operator Initiation of Self-Tests

The software integrity tests and the CASTs for AES, SHA, SHA3, DRBG, TLS KDF, KDA HKDF, PBKDF2 can be invoked by unloading and subsequently re-initializing the module. The CASTs for ECDSA, KAS-ECC-SSC and RSA can be invoked by requesting the corresponding Key Generation, Shared Secret Computation or Digital Signature services. Additionally, all the CASTs can be invoked by calling the BORINGSSL_self_test function. The PCTs can be invoked on demand by requesting the Key Generation service.

11 Life-Cycle Assurance

11.1 Installation, Initialization, and Startup Procedures

The module bcm.o is embedded into the shared library libcrypto.so which can be obtained by building the source code at the following location [1]. The set of files specified in the archive constitutes the complete set of source files of the validated module. There shall be no additions, deletions, or alterations of this set as used during module build.

[1] https://github.com/aws/aws-lc/archive/refs/heads/fips-NetOS-2024-06-11.zip

The downloaded zip file can be verified by issuing the "sha256sum aws-lc-fips-NetOS-2024-06-11.zip" command. The expected SHA2-256 digest value is:

952c4a23cea54e2ada37dde18d4d95228736f1d2c303a056c9bb39de55c9cd89

After the zip file is extracted, the cmake flags listed below must be used to compile the module. The compilation must be executed separately for each platform. Due to four possible combinations of OS/processor, the module count is four (i.e., there are four separate binaries generated, one for each entry listed in the *Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets)* table).

```
-DFIPS=1 \
-Dlibdir=${libdir} \
-Dbindir=${bindir} \ -DENABLE_EXPERIMENTAL_BIG_ENDIAN_SUPPORT=ON \
-DCMAKE_BUILD_TYPE=Release \
-GNinja \
```

Upon completion of the build process, the module's status can be verified by the command below. If the value obtained is "1" then the module has been installed and configured to operate in FIPS compliant manner.

./tool/bssl isfips

Lastly, the user can call the "show version" service using awslc_version_string function and the expected output is "AWS-LC FIPS 1.29.1" which is the module version. This will confirm that the module is in the operational mode. Additionally, the "AWS-LC FIPS" also acts as the module identifier and the verification of the "dynamic" part can be done using following command with an application that was used for dynamic linking. The "U" in the output confirms that the module is dynamically linked.

Command: *nm <application_name> | grep awslc_version_string*

Example Output: *" U awslc_version_string"*

11.2 Administrator Guidance

The approved and non-approved modes of operation are specified in section 2.4. The approved services that include the administrative functions are specified in section 4.3. All the logical interfaces are specified in section 3.1. The procedures of installation, initialization, startup, that are specified in section 11.1, and the operational environment requirements, that are specified in section 6, must be followed.

11.3 Non-Administrator Guidance

The approved and non-approved modes of operation are specified in section 2.4. The approved and non-approved cryptographic algorithms are specified in section 2.5. The approved security functions are specified in

section 2.6. The algorithm-specific information is specified in section 2.7. The approved services are specified in section 4.3, the non-approved services are specified in section 4.4. The logical interfaces available to users are specified in section 3.1. The user is responsible for following the procedures of installation, initialization, startup that are specified in section 11.1. The configuration settings and restrictions regarding the operational environment are specified in section 6.2.

11.4 End of Life

When the module is at end of life, for the GitHub repo, the README will be modified to mark the library as deprecated. After a 6-month window, more restrictive branch permissions will be added such that only administrators can read from the FIPS branch. The module does not possess persistent storage of SSPs. The SSP value only exists in volatile memory and that value vanishes when the module is powered off. So as a first step for the secure sanitization, the module needs to be powered off. Then for actual deprecation, the module will be upgraded to newer version that is approved. This upgrade process will uninstall/remove the old/terminated module and provide a new replacement.

12 Mitigation of Other Attacks

12.1 Attack List

RSA timing attacks: RSA is vulnerable to timing attacks. In a setup where attackers can measure the time of RSA decryption or signature operations, blinding must be used to protect the RSA operation from that attack.

12.2 Mitigation Effectiveness

The module provides the mechanism to use the blinding for RSA. When the blinding is on, the module generates a random value to form a blinding factor in the RSA key before the RSA key is used in the RSA cryptographic operations.