SafeLogic Inc.

CryptoComply 140-3 FIPS Provider

FIPS 140-3 Non-Proprietary Security Policy

Software Versions 3.0.0-FIPS 140-3, 3.0.1-FIPS 140-3

Document Version 1.1a

June 27, 2024

SafeLogic Inc.
530 Lytton Ave, Suite 200
Palo Alto, CA 94301
www.safelogic.com

Table of Contents

List of Tables

List of Figures

1 General

1.1 Overview

This document provides a non-proprietary FIPS 140-3 Security Policy for CryptoComply 140-3 FIPS Provider.

SafeLogic Inc.'s CryptoComply 140-3 FIPS Provider is designed to provide FIPS 140-3 validated cryptographic functionality and is available for licensing. For more information, visit www.safelogic.com/cryptocomply.

1.1.1 About FIPS 140

Federal Information Processing Standards Publication 140-3, Security Requirements for Cryptographic Modules, (FIPS 140-3) specifies the latest requirements for cryptographic modules utilized to protect sensitive but unclassified information. The National Institute of Standards and Technology (NIST) and Canadian Centre for Cyber Security (CCCS) collaborate to run the Cryptographic Module Validation Program (CMVP), which assesses conformance to FIPS 140. NIST (through NVLAP) accredits independent testing labs to perform FIPS 140 testing. The CMVP reviews and validates modules tested against FIPS 140 criteria. *Validated* is the term given to a module that has successfully gone through this FIPS 140 validation process. Validated modules receive a validation certificate that is posted on the CMVP's website.

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

1.1.2 About this Document

This non-proprietary cryptographic module Security Policy for CryptoComply 140-3 FIPS Provider from SafeLogic Inc. (SafeLogic) provides an overview of the product and a high-level description of how it meets the security requirements of FIPS 140-3. This document includes details on the module's cryptographic capabilities, services, sensitive security parameters, and self-tests. This Security Policy also includes guidance on operating the module while maintaining compliance with FIPS 140-3.

CryptoComply 140-3 FIPS Provider may also be referred to as the "module" in this document.

1.1.3 External Resources

The SafeLogic website (www.safelogic.com) contains information on SafeLogic services and products. The CMVP website maintains all FIPS 140 certificates for SafeLogic's FIPS 140 validations. These certificates also include SafeLogic contact information.

1.1.4 Notices

This document may be freely reproduced and distributed, but only in its entirety and without modification.

1.2 Security Levels

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

Table 1 - Security Levels

ISO/IEC 24759 Section 6. [Number Below]	FIPS 140-3 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

2 Cryptographic Module Specification

2.1 Description

2.1.1 Purpose and Use

CryptoComply 140-3 FIPS Provider is a standards-based “Drop-in Compliance™” cryptographic engine. The module delivers core cryptographic functions to applications such as servers, personal computers, mobile devices, and appliances. The module features robust algorithm support, including CNSA algorithms.

The module delivers cryptographic services to host applications through a C language Application Programming Interface (API).

2.1.2 Module Type

Software

2.1.3 Module Embodiment

Multichip Standalone

2.1.4 Module Characteristics

None

2.1.5 Cryptographic Boundary

The module's cryptographic boundary is delimited by the module’s components, as well as the instantiation of the cryptographic module saved in memory and executed by the processor. The executable files that constitute the cryptographic module are listed in Table 4 - Executable Code Sets. Additionally, the module’s integrity value is included inside the boundary.

Refer to the block diagram in Figure 1 (Security Policy Section 2.7 - Module Block Diagram) for additional detail.

2.1.5.1 Tested Operational Environment’s Physical Perimeter (TOEPP)

As a software cryptographic module, the module operates within the Tested Operational Environment’s Physical Perimeter (TOEPP). The TOEPP consists of the Operating System (OS) and the physical perimeter of the General Purpose Computer (GPC). This TOEPPe comprises the Operational Environment (OE) that the module operates in, the module itself, and all other applications that operate within the OE, including the host application for the module.

Refer to the block diagram in Figure 1 (Security Policy Section 2.7 - Module Block Diagram) for additional detail.

2.2 Version Information

Table 2 - Version Information

Type	Versions
Hardware	N/A
Software	3.0.0-FIPS 140-3, 3.0.1-FIPS 140-3
Firmware	N/A

Note: versions 3.0.0-FIPS 140-3 and 3.0.1-FIPS 140-3 have identical functionality and APIs, however an internal modification permits the module to be built as a dynamic or static build.

2.3 Operating Environments

2.3.1 Hardware Operating Environments

Not applicable

2.3.2 Software, Firmware, Hybrid Tested Operating Environments

The module operates in a modifiable operational environment under the FIPS 140-3 definitions. The module operates on a general purpose computer (GPC) running a general purpose operating system (GPOS).

The module was tested in the following operating environments specified in the table below. Testing on iOS 16 and iPadOS 16 was performed with module version 3.0.1-FIPS 140-3 and testing on all other OEs was performed with module version 3.0.0-FIPS 140-3.

Table 3 – Tested Operational Environments

#	Operating System	Hardware Platform	Processor	PAA/Acceleration
1.	AlmaLinux 9	Dell PowerEdge R830	Intel Xeon E5-4667v4	AES_NI
2.	AlmaLinux 9	Dell PowerEdge R830	Intel Xeon E5-4667v4	No
3.	Android 13	Google Pixel 7	Google Tensor G2	No
4.	Debian 11	Dell PowerEdge R830	Intel Xeon E5-4667v4	AES_NI
5.	Debian 11	Dell PowerEdge R830	Intel Xeon E5-4667v4	No
6.	FreeBSD 13	Dell PowerEdge R830	Intel Xeon E5-4667v4	AES_NI

#	Operating System	Hardware Platform	Processor	PAA/Acceleration
7.	FreeBSD 13	Dell PowerEdge R830	Intel Xeon E5-4667v4	No
8.	iOS 16	iPhone 13 Mini	Apple A15 Bionic	No
9.	iPadOS 16	iPad Air (2022)	Apple M1	No
10.	macOS 13 (Ventura)	Mac Mini M2	Apple M2	No
11.	Oracle Solaris 11.4	Dell PowerEdge R830	Intel Xeon E5-4667v4	AES_NI
12.	Oracle Solaris 11.4	Dell PowerEdge R830	Intel Xeon E5-4667v4	No
13.	Red Hat Enterprise Linux 9	Dell PowerEdge R830	Intel Xeon E5-4667v4	AES_NI
14.	Red Hat Enterprise Linux 9	Dell PowerEdge R830	Intel Xeon E5-4667v4	No
15.	Rocky Linux 9	Dell PowerEdge R830	Intel Xeon E5-4667v4	AES_NI
16.	Rocky Linux 9	Dell PowerEdge R830	Intel Xeon E5-4667v4	No
17.	SUSE Linux Enterprise Server 15	Dell PowerEdge R830	Intel Xeon E5-4667v4	AES_NI
18.	SUSE Linux Enterprise Server 15	Dell PowerEdge R830	Intel Xeon E5-4667v4	No
19.	Ubuntu 22.04	Dell PowerEdge R830	Intel Xeon E5-4667v4	AES_NI
20.	Ubuntu 22.04	Dell PowerEdge R830	Intel Xeon E5-4667v4	No
21.	Windows 10	Dell PowerEdge R830	Intel Xeon E5-4667v4	AES_NI
22.	Windows 10	Dell PowerEdge R830	Intel Xeon E5-4667v4	No
23.	Windows 11	Dell PowerEdge R830	Intel Xeon E5-4667v4	AES_NI
24.	Windows 11	Dell PowerEdge R830	Intel Xeon E5-4667v4	No
25.	Windows Server 2019	Dell PowerEdge R830	Intel Xeon E5-4667v4	AES_NI
26.	Windows Server 2019	Dell PowerEdge R830	Intel Xeon E5-4667v4	No
27.	Windows Server 2022	Dell PowerEdge R830	Intel Xeon E5-4667v4	AES_NI
28.	Windows Server 2022	Dell PowerEdge R830	Intel Xeon E5-4667v4	No

2.3.3 Executable Code Sets

Table 4 - Executable Code Sets

Package/File Names	Software Version	Non-Security Relevant Distinguishing Features	Integrity Test Implemented
fips.so	3.0.0-FIPS 140-3	Compiled for Linux, Unix, Android	HMAC-SHA-256
fips.dll	3.0.0-FIPS 140-3	Compiled for Windows	HMAC-SHA-256
fips.dylib	3.0.0-FIPS 140-3	Compiled for MacOS	HMAC-SHA-256
fips.a	3.0.1-FIPS 140-3	Compiled as a static library, tested on iOS and iPadOS	HMAC-SHA-256

2.3.4 Vendor Affirmed Operating Environments

Porting guidance is defined in the FIPS 140-3 CMVP Management Manual Section 7.9. FIPS 140-3 validation compliance can be maintained when the following requirements are met:

- No source code modifications are required to recompile the module and port it to another operating environment
- The module is operating on any general-purpose platform/processor that supports the specified operating system as listed on the validation entry or another compatible operating system.

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

The module, when compiled from the same unmodified source code, is vendor affirmed to be FIPS 140-3 compliant when compiled as a static library for the operating environments listed in Table 3 for which the module was tested as a shared object or dynamically loaded library.

Table 5 - Vendor Affirmed Operational Environments

#	Operating System	Hardware Platform
1.	AlmaLinux 9	Any general-purpose platform that supports this OS
2.	Android 13	Any general-purpose mobile platform that supports this OS
3.	Debian 11	Any general-purpose platform that supports this OS
4.	FreeBSD 13	Any general-purpose platform that supports this OS
5.	iOS 16	Any general-purpose mobile platform that supports this OS
6.	iPadOS 16	Any general-purpose mobile platform that supports this OS
7.	macOS 13 (Ventura)	Any general-purpose platform that supports this OS
8.	Oracle Solaris 11.4	Any general-purpose platform that supports this OS

#	Operating System	Hardware Platform
9.	Red Hat Enterprise Linux 9	Any general-purpose platform that supports this OS
10.	Rocky Linux 9	Any general-purpose platform that supports this OS
11.	SUSE Linux Enterprise Server 15	Any general-purpose platform that supports this OS
12.	Ubuntu 22.04	Any general-purpose platform that supports this OS
13.	Windows 10	Any general-purpose platform that supports this OS
14.	Windows 11	Any general-purpose platform that supports this OS
15.	Windows Server 2019	Any general-purpose platform that supports this OS
16.	Windows Server 2022	Any general-purpose platform that supports this OS

2.4 Excluded Components

Not applicable.

2.5 Modes of Operation

2.5.1 Modes List and Description

Table 6 - Modes of Operation

Name	Description	FIPS	Status Indicator
Approved mode	Single approved mode of operation. No non-approved mode is implemented in the module.	FIPS	In alignment with IG 2.4.C example scenario 2, the module only provides approved services. The module provides a global indicator that services are approved. Additionally, the module provides a status code indicating the completion of each service, as indicated in Security Policy Section 4.3 - Approved Services. The successful completion of a service is an implicit indicator for the use of an approved service.

2.5.2 Mode change instructions and status indicators

No instructions are needed to invoke the Approved mode in the module. The module only supports this mode of operation and will operate in this mode once the module is powered on.

To confirm that the module is operating in Approved mode, the operator should:

- Obtain the global indicator by calling EVP_default_properties_is_fips_enabled() and confirming that this returns as true.

    - Returning as true indicates that the module is configured in Approved mode
- Confirm that the service that is called successfully completes. The successful completion of a service is an implicit indicator for the use of an approved service.

2.5.3 Degraded Mode Description

Not applicable.

2.6 Algorithms

2.6.1 Approved Algorithms

The module implements the following approved algorithms that have been tested by the Cryptographic Algorithm Validation Program (CAVP).

Table 7 - Approved Algorithms

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A4593, A5173	AES-CBC SP 800-38A	AES-CBC	Strength: 128, 192, 256 bits Direction: Decrypt, Encrypt Key Length: 128, 192, 256	Encryption, Decryption
A4593, A5173	AES-CBC-CS1 SP 800-38A-Add	AES-CBC-CS1	Strength: 128, 192, 256 bits Direction: decrypt, encrypt Key Length: 128, 192, 256 Payload Length: 128-65536 Increment 8	Encryption, Decryption
A4593, A5173	AES-CBC-CS2 SP 800-38A-Add	AES-CBC-CS2	Strength: 128, 192, 256 bits Direction: decrypt, encrypt Key Length: 128, 192, 256 Payload Length: 128-65536 Increment 8	Encryption, Decryption
A4593, A5173	AES-CBC-CS3 SP 800-38A-Add	AES-CBC-CS3	Strength: 128, 192, 256 bits Direction: decrypt, encrypt Key Length: 128, 192, 256 Payload Length: 128-65536 Increment 8	Encryption, Decryption
A4593, A5173	AES-CCM SP 800-38C	AES-CCM	Strength: 128, 192, 256 bits Key Length: 128, 192, 256 Tag Length: 32, 48, 64, 80, 96, 112, 128 IV Length: 56-104 Increment 8 Payload Length: 0-256 Increment 8 AAD Length: 0-524288 Increment 8	Encryption, Decryption
A4593, A5173	AES-CFB1 SP 800-38A	AES-CFB1	Strength: 128, 192, 256 bits Direction: Decrypt, Encrypt Key Length: 128, 192, 256	Encryption, Decryption
A4593, A5173	AES-CFB8 SP 800-38A	AES-CFB8	Strength: 128, 192, 256 bits Direction: Decrypt, Encrypt Key Length: 128, 192, 256	Encryption, Decryption

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A4593, A5173	AES-CFB128 SP 800-38A	AES-CFB128	Strength: 128, 192, 256 bits Direction: Decrypt, Encrypt Key Length: 128, 192, 256	Encryption, Decryption
A4593, A5173	AES-CMAC SP 800-38B	AES-CMAC	Strength: 128, 192, 256 bits Direction: Generation, Verification Key Length: 128, 192, 256 MAC Length: 128 Message Length: 0-524288 Increment 8	Generation, Verification
A4593, A5173	AES-CTR SP 800-38A	AES-CTR	Strength: 128, 192, 256 bits Direction: Decrypt, Encrypt Key Length: 128, 192, 256 Payload Length: 8-128 Increment 8 Incremental Counter Counter Tests Performed	Encryption, Decryption
A4593, A5173	AES-ECB SP 800-38A	AES-ECB	Strength: 128, 192, 256 bits Direction: Decrypt, Encrypt Key Length: 128, 192, 256	Encryption, Decryption
A4593, A5173	AES-GCM SP 800-38D	AES-GCM	Strength: 128, 192, 256 bits Direction: Decrypt, Encrypt IV Generation: Internal IV Generation Mode: 8.2.1 Key Length: 128, 192, 256 Tag Length: 32, 64, 96, 104, 112, 120, 128 IV Length: 96-1024 Increment 8 Payload Length: 0-65536 Increment 8 AAD Length: 0-65536 Increment 8	Encryption, Decryption
A4593, A5173	AES-GCM SP 800-38D	AES-GCM	Strength: 128, 192, 256 bits Direction: Decrypt, Encrypt IV Generation: External IV Generation Mode: 8.2.2 Key Length: 128, 192, 256 Tag Length: 32, 64, 96, 104, 112, 120, 128 IV Length: 96-1024 Increment 8 Payload Length: 8-65536 Increment 8 AAD Length: 0-65536 Increment 8	Encryption, Decryption

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A4593, A5173	AES-GMAC SP 800-38D	AES-GMAC	Strength: 128, 192, 256 bits Direction: Decrypt, Encrypt IV Generation: Internal IV Generation Mode: 8.2.1 Key Length: 128, 192, 256 Tag Length: 32, 64, 96, 104, 112, 120, 128 IV Length: 96-1024 Increment 8 AAD Length: 0-65536 Increment 8	Encryption, Decryption, Generation, Verification
A4593, A5173	AES-GMAC SP 800-38D	AES-GMAC	Strength: 128, 192, 256 bits Direction: Decrypt, Encrypt IV Generation: External IV Generation Mode: 8.2.2 Key Length: 128, 192, 256 Tag Length: 32, 64, 96, 104, 112, 120, 128 IV Length: 96-1024 Increment 8 AAD Length: 0-65536 Increment 8	Encryption, Decryption, Generation, Verification
A4593, A5173	AES-KW SP 800-38F	AES-KW	Strength: 128, 192, 256 bits Direction: Decrypt, Encrypt Cipher: Cipher, Inverse Key Length: 128, 192, 256 Payload Length: 128-524288 Increment 128	Encryption, Decryption
A4593, A5173	AES-KWP SP 800-38F	AES-KWP	Strength: 128, 192, 256 bits Direction: Decrypt, Encrypt Cipher: Cipher, Inverse Key Length: 128, 192, 256 Payload Length: 8-524288 Increment 8	Encryption, Decryption
A4593, A5173	AES-OFB SP 800-38A	AES-OFB	Strength: 128, 192, 256 bits Direction: Decrypt, Encrypt Key Length: 128, 192, 256	Encryption, Decryption
A4593, A5173	AES-XTS Testing Revision 2.0 SP 800-38E	AES-XTS Testing Revision 2.0	Strength: 128, 256 bits Direction: Decrypt, Encrypt Key Length: 128, 256 Payload Length: 128-65536 Increment 128 Tweak Mode: Hex Data Unit Length Matches Payload	Encryption, Decryption

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
Vendor affirmed	CKG (SP 800-133Rev2)	CKG (SP 800-133Rev2)	The module uses the direct output of its approved DRBGs for key generation Key Generation per SP 800-133r2: • Section 4: Using the Output of a Random Bit Generator. The module uses the direct output of its approved DRBGs. ○ As per Section 5, this method is used to supply random values used in the Generation of Key Pairs for Asymmetric-Key Algorithms. • Section 6.2: Derivation of Symmetric Keys. The module also supports key derivation via KDF and SSP agreement.	Cryptographic Key Generation; SP 800-133 and IG D.H.
A4593, A5173	Counter DRBG SP 800-90A	Counter DRBG	Strength: 128, 192, 256 bits Prediction Resistance: Yes Supports Reseed Capabilities: Mode: AES-128 Derivation Function Enabled: Yes Additional Input: 0-256 Increment 256 Entropy Input: 128-256 Increment 128 Nonce: 128 Personalization String Length: 0-256 Increment 256 Returned Bits: 256 Capabilities: Mode: AES-192 Derivation Function Enabled: Yes Additional Input: 0-256 Increment 256 Entropy Input: 256-512 Increment 128 Nonce: 128 Personalization String Length: 0-256 Increment 256 Returned Bits: 256 Capabilities: Mode: AES-256 Derivation Function Enabled: Yes Additional Input: 0-256 Increment 256 Entropy Input: 256-512 Increment 128 Nonce: 128 Personalization String Length: 0-256 Increment 256 Returned Bits: 256	Random Number Generation

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A4593, A5173	DSA KeyGen FIPS 186-4	DSA KeyGen	Strength: 112 bits Capabilities: L: 2048 N: 224 Capabilities: L: 2048 N: 256	Key Generation for Key Agreement
A4593, A5173	DSA PQGGen FIPS 186-4	DSA PQGGen	Strength: 112 bits Capabilities: P/Q Generation Methods: Probable G Generation Methods: Canonical, Unverifiable L: 2048 N: 224 Hash Algorithm: SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256 Capabilities: P/Q Generation Methods: Probable G Generation Methods: Canonical, Unverifiable L: 2048 N: 256 Hash Algorithm: SHA2-256, SHA2-384, SHA2-512, SHA2-512/256	Key Generation for Key Agreement

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A4593, A5173	DSA PQGVer FIPS 186-4	DSA PQGVer	Strength: 80, 112 bits Capabilities: P/Q Generation Methods: Probable G Generation Methods: Canonical, Unverifiable L: 1024 N: 160 Hash Algorithm: SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256 Capabilities: P/Q Generation Methods: Probable G Generation Methods: Canonical, Unverifiable L: 2048 N: 224 Hash Algorithm: SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256 Capabilities: P/Q Generation Methods: Probable G Generation Methods: Canonical, Unverifiable L: 2048 N: 256 Hash Algorithm: SHA2-256, SHA2-384, SHA2-512, SHA2-512/256	Legacy Key Verification

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A4593, A5173	DSA SigVer FIPS 186-4	DSA SigVer	Strength: 80, 112, 128 bits Capabilities: L: 1024 N: 160 Hash Algorithm: SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256 Capabilities: L: 2048 N: 224 Hash Algorithm: SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256 Capabilities: L: 2048 N: 256 Hash Algorithm: SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256 Capabilities: L: 3072 N: 256 Hash Algorithm: SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256	Legacy Signature Verification
A4593, A5173	ECDSA KeyGen FIPS 186-4	ECDSA KeyGen	Strength: 112-256 bits Curve: B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 Secret Generation Mode: Testing Candidates	Key Generation
A4593, A5173	ECDSA KeyVer FIPS 186-4	ECDSA KeyVer	Strength: 80 bits (Legacy), 112-256 bits Curve: B-163, B-233, B-283, B-409, B-571, K-163, K-233, K-283, K-409, K-571, P-192, P-224, P-256, P-384, P-521	Key Verification
A4593, A5173	ECDSA SigGen FIPS 186-4	ECDSA SigGen	Strength: 112-256 bits Curve: B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 Hash Algorithm: SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	Signature Generation
A4593, A5173	ECDSA SigVer FIPS 186-4	ECDSA SigVer	Strength: 80 bits (Legacy), 112-256 bits Curve: B-163, B-233, B-283, B-409, B-571, K-163, K-233, K-283, K-409, K-571, P-192, P-224, P-256, P-384, P-521 Hash Algorithm: SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	Signature Verification

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A4593, A5173	EDDSA keygen FIPS 186-5	EDDSA keygen	Strength: 128, 224 bits Curve: ED-25519, ED-448	Key Generation
A4593, A5173	EDDSA keyVer FIPS 186-5	EDDSA keyVer	Strength: 128, 224 bits Curve: ED-25519, ED-448	Key Verification
A4593, A5173	EDDSA SigGen FIPS 186-5	EDDSA SigGen	Strength: 128, 224 bits Curve: ED-25519, ED-448	Signature Generation
A4593, A5173	EDDSA SigVer FIPS 186-5	EDDSA SigVer	Strength: 128, 224 bits Curve: ED-25519, ED-448	Signature Verification
A4593, A5173	Hash DRBG SP 800-90A	Hash DRBG	Strength: 128 bits for SHA-1, 256 bits for SHA-256 and SHA-512 Prediction Resistance: Yes Supports Reseed Capabilities: Mode: SHA-1 Entropy Input: 128-256 Increment 64 Nonce: 96-128 Increment 32 Personalization String Length: 0-256 Increment 128 Additional Input: 0-256 Increment 128 Returned Bits: 160 Capabilities: Mode: SHA2-256 Entropy Input: 256-320 Increment 64 Nonce: 128-160 Increment 32 Personalization String Length: 0-256 Increment 128 Additional Input: 0-256 Increment 128 Returned Bits: 256 Capabilities: Mode: SHA2-512 Entropy Input: 256-320 Increment 64 Nonce: 128-160 Increment 32 Personalization String Length: 0-256 Increment 128 Additional Input: 0-256 Increment 128 Returned Bits: 512	Random Number Generation

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A4593, A5173	HMAC DRBG SP 800-90A	HMAC DRBG	Strength: 128 bits for SHA-1, 256 bits for SHA-256 and SHA-512 Prediction Resistance: Yes Supports Reseed Capabilities: Mode: SHA-1 Entropy Input: 160-256 Increment 32 Nonce: 64 Personalization String Length: 0-256 Increment 128 Additional Input: 0-256 Increment 128 Returned Bits: 160 Capabilities: Mode: SHA2-256 Entropy Input: 256-512 Increment 64 Nonce: 128 Personalization String Length: 0-256 Increment 128 Additional Input: 0-256 Increment 128 Returned Bits: 256 Capabilities: Mode: SHA2-512 Entropy Input: 512-1024 Increment 64 Nonce: 128 Personalization String Length: 0-256 Increment 128 Additional Input: 0-256 Increment 128 Returned Bits: 512	Random Number Generation
A4593, A5173	HMAC-SHA-1 FIPS 198-1	HMAC-SHA-1	Strength: 128 bits MAC: 32-160 Increment 8 Key Length: 8-524288 Increment 8	Message Authentication
A4593, A5173	HMAC-SHA2-224 FIPS 198-1	HMAC-SHA2-224	Strength: 192 bits MAC: 32-224 Increment 8 Key Length: 8-524288 Increment 8	Message Authentication
A4593, A5173	HMAC-SHA2-256 FIPS 198-1	HMAC-SHA2-256	Strength: 256 bits MAC: 32-256 Increment 8 Key Length: 8-524288 Increment 8	Message Authentication
A4593, A5173	HMAC-SHA2-384 FIPS 198-1	HMAC-SHA2-384	Strength: 256 bits MAC: 32-384 Increment 8 Key Length: 8-524288 Increment 8	Message Authentication

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A4593, A5173	HMAC-SHA2-512 FIPS 198-1	HMAC-SHA2-512	Strength: 256 bits MAC: 32-512 Increment 8 Key Length: 8-524288 Increment 8	Message Authentication
A4593, A5173	HMAC-SHA2-512/224 FIPS 198-1	HMAC-SHA2-512/224	Strength: 192 bits MAC: 32-224 Increment 8 Key Length: 8-524288 Increment 8	Message Authentication
A4593, A5173	HMAC-SHA2-512/256 FIPS 198-1	HMAC-SHA2-512/256	Strength: 256 bits MAC: 32-256 Increment 8 Key Length: 8-524288 Increment 8	Message Authentication
A4593, A5173	HMAC-SHA3-224 FIPS 198-1	HMAC-SHA3-224	Strength: 192 bits MAC: 32-224 Increment 8 Key Length: 8-524288 Increment 8	Message Authentication
A4593, A5173	HMAC-SHA3-256 FIPS 198-1	HMAC-SHA3-256	Strength: 256 bits MAC: 32-256 Increment 8 Key Length: 8-524288 Increment 8	Message Authentication
A4593, A5173	HMAC-SHA3-384 FIPS 198-1	HMAC-SHA3-384	Strength: 256 bits MAC: 32-384 Increment 8 Key Length: 8-524288 Increment 8	Message Authentication
A4593, A5173	HMAC-SHA3-512 FIPS 198-1	HMAC-SHA3-512	Strength: 256 bits MAC: 32-512 Increment 8 Key Length: 8-524288 Increment 8	Message Authentication
A4593, A5173	KAS-ECC-SSC SP 800-56Ar3	KAS-ECC-SSC	Strength: 112-256 bits Domain Parameter Generation Methods: B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 Scheme: ephemeralUnified: KAS Role: initiator, responder	Key Agreement
A4593, A5173	KAS-FFC-SSC SP 800-56Ar3	KAS-FFC-SSC	Strength: 112 bits (FB, FC), 112-200 bits (Safe Primes) Domain Parameter Generation Methods: FB, FC, ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192 Scheme: dhEphem: KAS Role: initiator, responder	Key Agreement

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A4593, A5173	KAS-IFC-SSC SP 800-56Br2	KAS-IFC-SSC	Strength: 112-256 bits Modulo: 2048, 3072, 4096, 6144, 8192 Key Generation Methods: rsakpg1-basic, rsakpg1-crt, rsakpg1-prime-factor, rsakpg2-basic, rsakpg2-crt, rsakpg2-prime-factor Scheme: KAS1: KAS Role: initiator, responder KAS2: KAS Role: initiator, responder Fixed Public Exponent: 010001	Key Agreement
A4593, A5173	KDA HKDF SP 800-56Cr2	KDA HKDF	Strength: 128, 192, 256 bits Fixed Info Pattern: algorithmId||l||uPartyInfo||vPartyInfo Fixed Info Encoding: concatenation Derived Key Length: 2048 Shared Secret Length: 224-8192 Increment 8 HMAC Algorithm: SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	Key Derivation

A4593, A5173	KDA OneStep SP 800-56Cr2	KDA OneStep	Strength: 128, 192, 256 bits Auxiliary Function Methods: Auxiliary Function Name: SHA-1 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: SHA2-224 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: SHA2-256 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: SHA2-384 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: SHA2-512 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: SHA2-512/224 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: SHA2-512/256 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: SHA3-224 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: SHA3-256 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: SHA3-384 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: SHA3-512 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: HMAC-SHA-1 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: HMAC-SHA2-224 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: HMAC-SHA2-256 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: HMAC-SHA2-384 MAC Salting Methods: default, random	Key Derivation

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
			Auxiliary Function Methods: Auxiliary Function Name: HMAC-SHA2-512 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: HMAC-SHA2-512/224 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: HMAC-SHA2-512/256 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: HMAC-SHA3-224 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: HMAC-SHA3-256 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: HMAC-SHA3-384 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: HMAC-SHA3-512 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: KMAC-128 MAC Salting Methods: default, random Auxiliary Function Methods: Auxiliary Function Name: KMAC-256 MAC Salting Methods: default, random Fixed Info Pattern: algorithmId||||uPartyInfo||vPartyInfo Fixed Info Encoding: concatenation Derived Key Length: 2048 Shared Secret Length: 224-8192 Increment 8	
A4593, A5173	KDA TwoStep SP 800-56Cr2	KDA TwoStep	Strength: 128, 192, 256 bits Fixed Info Pattern: algorithmId||||uPartyInfo||vPartyInfo Fixed Info Encoding: concatenation KDF Mode: feedback MAC Modes: HMAC-SHA-1, HMAC-SHA2-224, HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512, HMAC-SHA2-512/224, HMAC-SHA2-512/256, HMAC-SHA3-224, HMAC-SHA3-256, HMAC-SHA3-384, HMAC-SHA3-512 Fixed Data Order: after fixed data Counter Lengths: 8 The KDF supports an empty IV The KDF requires an empty IV Supported Lengths: 2048 Derived Key Length: 2048 Shared Secret Length: 224-8192 Increment 8	Key Derivation

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A4593, A5173	KDF ANS 9.42 SP 800-135r1 CVL	KDF ANS 9.42	Strength: 128, 192, 256 bits KDF Type: DER Hash Algorithm: SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512 Other Info Length: 0-4096 Increment 8 zz Length: 8-4096 Increment 8 Key Data Length: 8-4096 Increment 8 Supplemental Information Length: 0-120 Increment 8 OID: AES-128-KW, AES-192-KW, AES-256-KW	Key Derivation
A4593, A5173	KDF ANS 9.63 SP 800-135r1 CVL	KDF ANS 9.63	Strength: 192, 256 bits Hash Algorithm: SHA2-224, SHA2-256, SHA2-384, SHA2-512 Field Size: 224, 571 Shared Info Length: 0, 1024 Key Data Length: 128, 4096	Key Derivation
A4593, A5173	KDF KMAC SP 800-108r1	KBKDF KMAC	Strength: 128, 256 bits Key Derivation Key Length: 112-4096 Increment 8 Context Length: 8-4096 Increment 8 Label Length: 8-4096 Increment 8 Derived Key Length: 112-4096 Increment 8 MAC Modes: KMAC-128, KMAC-256	Key Derivation
A4593, A5173	KDF SP 800-108r1	KBKDF	Strength: 128, 192, 256 bits Capabilities: KDF Mode: Counter MAC Mode: CMAC-AES128, CMAC-AES192, CMAC-AES256, HMAC-SHA-1, HMAC-SHA2-224, HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512, HMAC-SHA2-512/224, HMAC-SHA2-512/256, HMAC-SHA3-224, HMAC-SHA3-256, HMAC-SHA3-384, HMAC-SHA3-512 Supported Lengths: 8, 72, 128, 776, 3456, 4096 Fixed Data Order: Before Fixed Data Counter Length: 32 Custom Key In Length: 0 Capabilities: KDF Mode: Feedback MAC Mode: CMAC-AES128, CMAC-AES192, CMAC-AES256, HMAC-SHA-1, HMAC-SHA2-224, HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512, HMAC-SHA2-512/224, HMAC-SHA2-512/256, HMAC-SHA3-224, HMAC-SHA3-256, HMAC-SHA3-384, HMAC-SHA3-512 Supported Lengths: 8, 72, 128, 776, 3456, 4096 Fixed Data Order: Before Fixed Data Counter Length: 32 Custom Key In Length: 0	Key Derivation

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A4593, A5173	KDF SSH SP 800-135r1 CVL	KDF SSH	Strength: 128, 192, 256 bits Cipher: AES-128, AES-192, AES-256 Hash Algorithm: SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	Key Derivation
A4593, A5173	KMAC-128 SP 800-185	KMAC-128	Strength: 128 bits Message Length: 0-65536 Increment 8 MAC Length: 32-65536 Increment 8 Key Data Length: 128-1024 Increment 8 Supports eXtendable-Output Functions: Yes, No	Key Derivation
A4593, A5173	KMAC-256 SP 800-185	KMAC-256	Strength: 256 bits Message Length: 0-65536 Increment 8 MAC Length: 32-65536 Increment 8 Key Data Length: 128-1024 Increment 8 Supports eXtendable-Output Functions: Yes, No	Key Derivation
A4593, A5173	KTS (AES) SP 800-38F	KTS (AES)	Strength: 128, 192, 256 bits AES-KW, AES-KWP Key Length: 128, 192, 256 Additional detail provided in Table 9 - Security Function Implementations in the entry for the KeyWrapping function	Key Transport
A4593, A5173	KTS-IFC SP 800-56Br2	KTS-IFC	Strength: 112-200 bits Modulo: 2048, 3072, 4096, 6144 Key Generation Methods: rsakpg1-basic, rsakpg1-crt, rsakpg1-prime-factor, rsakpg2-basic, rsakpg2-crt, rsakpg2-prime-factor Fixed Public Exponent: 010001 Scheme: KTS-OAEP-basic: KAS Role: initiator, responder Key Transport Method: Hash Algorithms: SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512 Supports Null Associated Data Associated Data Encoding: concatenation Key Length: 1024 Additional detail provided in Table 9 - Security Function Implementations in the entry for the KeyTransport function	Key Transport

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A4593, A5173	PBKDF SP 800-132	PBKDF	Strength: 128, 192, 256 bits Iteration Count: 1-10000 Increment 1 HMAC Algorithm: SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512 Password Length: 8-128 Increment 8 Salt Length: 128-4096 Increment 8 Key Data Length: 112-4096 Increment 8	Key Derivation
A4593, A5173	RSA KeyGen FIPS 186-4	RSA KeyGen	Strength: 112, 128, 150 bits Capabilities: Key Generation Mode: B.3.3 Properties: Modulo: 2048 Primality Tests: Table C.2 Properties: Modulo: 3072 Primality Tests: Table C.2 Properties: Modulo: 4096 Primality Tests: Table C.2 Info Generated By Server Public Exponent Mode: Random Private Key Format: Standard	Key Generation

A4593, A5173	RSA SigGen FIPS 186-4	RSA SigGen	Strength: 112, 128, 150 bits Capabilities: Signature Type: PKCS 1.5 Properties: Modulo: 2048 Hash Pair: Hash Algorithm: SHA2-224 Hash Pair: Hash Algorithm: SHA2-256 Hash Pair: Hash Algorithm: SHA2-384 Hash Pair: Hash Algorithm: SHA2-512 Hash Pair: Hash Algorithm: SHA2-512/224 Hash Pair: Hash Algorithm: SHA2-512/256 Properties: Modulo: 3072 Hash Pair: Hash Algorithm: SHA2-224 Hash Pair: Hash Algorithm: SHA2-256 Hash Pair: Hash Algorithm: SHA2-384 Hash Pair: Hash Algorithm: SHA2-512 Hash Pair: Hash Algorithm: SHA2-512/224 Hash Pair: Hash Algorithm: SHA2-512/256 Properties: Modulo: 4096 Hash Pair: Hash Algorithm: SHA2-224 Hash Pair: Hash Algorithm: SHA2-256 Hash Pair: Hash Algorithm: SHA2-384 Hash Pair: Hash Algorithm: SHA2-512 Hash Pair: Hash Algorithm: SHA2-512/224 Hash Pair:	Signature Generation

			Hash Algorithm: SHA2-512/256 Capabilities: Signature Type: PKCSPSS Properties: Modulo: 2048 Hash Pair: Hash Algorithm: SHA2-224 Salt Length: 28 Hash Pair: Hash Algorithm: SHA2-256 Salt Length: 32 Hash Pair: Hash Algorithm: SHA2-384 Salt Length: 48 Hash Pair: Hash Algorithm: SHA2-512 Salt Length: 64 Hash Pair: Hash Algorithm: SHA2-512/224 Salt Length: 28 Hash Pair: Hash Algorithm: SHA2-512/256 Salt Length: 32 Properties: Modulo: 3072 Hash Pair: Hash Algorithm: SHA2-224 Salt Length: 28 Hash Pair: Hash Algorithm: SHA2-256 Salt Length: 32 Hash Pair: Hash Algorithm: SHA2-384 Salt Length: 48 Hash Pair: Hash Algorithm: SHA2-512 Salt Length: 64 Hash Pair: Hash Algorithm: SHA2-512/224 Salt Length: 28 Hash Pair: Hash Algorithm: SHA2-512/256 Salt Length: 32 Properties: Modulo: 4096 Hash Pair:	

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
			Hash Algorithm: SHA2-224 Salt Length: 28 Hash Pair: Hash Algorithm: SHA2-256 Salt Length: 32 Hash Pair: Hash Algorithm: SHA2-384 Salt Length: 48 Hash Pair: Hash Algorithm: SHA2-512 Salt Length: 64 Hash Pair: Hash Algorithm: SHA2-512/224 Salt Length: 28 Hash Pair: Hash Algorithm: SHA2-512/256 Salt Length: 32	

A4593, A5173	RSA SigVer FIPS 186-4	RSA SigVer	Strength: 80 bits (Legacy), 112, 128, 150 bits Capabilities: Signature Type: PKCS 1.5 Properties: Modulo: 1024 Hash Pair: Hash Algorithm: SHA-1 Hash Pair: Hash Algorithm: SHA2-224 Hash Pair: Hash Algorithm: SHA2-256 Hash Pair: Hash Algorithm: SHA2-384 Hash Pair: Hash Algorithm: SHA2-512 Hash Pair: Hash Algorithm: SHA2-512/224 Hash Pair: Hash Algorithm: SHA2-512/256 Properties: Modulo: 2048 Hash Pair: Hash Algorithm: SHA-1 Hash Pair: Hash Algorithm: SHA2-224 Hash Pair: Hash Algorithm: SHA2-256 Hash Pair: Hash Algorithm: SHA2-384 Hash Pair: Hash Algorithm: SHA2-512 Hash Pair: Hash Algorithm: SHA2-512/224 Hash Pair: Hash Algorithm: SHA2-512/256 Properties: Modulo: 3072 Hash Pair: Hash Algorithm: SHA-1 Hash Pair: Hash Algorithm: SHA2-224 Hash Pair: Hash Algorithm: SHA2-256 Hash Pair: Hash Algorithm: SHA2-384 Hash Pair:	Signature Verification

			Hash Algorithm: SHA2-512 Hash Pair: Hash Algorithm: SHA2-512/224 Hash Pair: Hash Algorithm: SHA2-512/256 Properties: Modulo: 4096 Hash Pair: Hash Algorithm: SHA-1 Hash Pair: Hash Algorithm: SHA2-224 Hash Pair: Hash Algorithm: SHA2-256 Hash Pair: Hash Algorithm: SHA2-384 Hash Pair: Hash Algorithm: SHA2-512 Hash Pair: Hash Algorithm: SHA2-512/224 Hash Pair: Hash Algorithm: SHA2-512/256 Capabilities: Signature Type: ANSI X9.31 Properties: Modulo: 1024 Hash Pair: Hash Algorithm: SHA-1 Hash Pair: Hash Algorithm: SHA2-256 Hash Pair: Hash Algorithm: SHA2-384 Hash Pair: Hash Algorithm: SHA2-512 Properties: Modulo: 2048 Hash Pair: Hash Algorithm: SHA-1 Hash Pair: Hash Algorithm: SHA2-256 Hash Pair: Hash Algorithm: SHA2-384 Hash Pair: Hash Algorithm: SHA2-512 Properties: Modulo: 3072 Hash Pair:	

			Hash Algorithm: SHA-1 Hash Pair: Hash Algorithm: SHA2-256 Hash Pair: Hash Algorithm: SHA2-384 Hash Pair: Hash Algorithm: SHA2-512 Properties: Modulo: 4096 Hash Pair: Hash Algorithm: SHA-1 Hash Pair: Hash Algorithm: SHA2-256 Hash Pair: Hash Algorithm: SHA2-384 Hash Pair: Hash Algorithm: SHA2-512 Capabilities: Signature Type: PKCSPSS Properties: Modulo: 1024 Hash Pair: Hash Algorithm: SHA-1 Salt Length: 20 Hash Pair: Hash Algorithm: SHA2-224 Salt Length: 28 Hash Pair: Hash Algorithm: SHA2-256 Salt Length: 32 Hash Pair: Hash Algorithm: SHA2-384 Salt Length: 48 Hash Pair: Hash Algorithm: SHA2-512 Salt Length: 62 Hash Pair: Hash Algorithm: SHA2-512/224 Salt Length: 24 Hash Pair: Hash Algorithm: SHA2-512/256 Salt Length: 32 Properties: Modulo: 2048 Hash Pair: Hash Algorithm: SHA-1	

			Salt Length: 20 Hash Pair: Hash Algorithm: SHA2-224 Salt Length: 28 Hash Pair: Hash Algorithm: SHA2-256 Salt Length: 32 Hash Pair: Hash Algorithm: SHA2-384 Salt Length: 48 Hash Pair: Hash Algorithm: SHA2-512 Salt Length: 64 Hash Pair: Hash Algorithm: SHA2-512/224 Salt Length: 24 Hash Pair: Hash Algorithm: SHA2-512/256 Salt Length: 32 Properties: Modulo: 3072 Hash Pair: Hash Algorithm: SHA-1 Salt Length: 20 Hash Pair: Hash Algorithm: SHA2-224 Salt Length: 28 Hash Pair: Hash Algorithm: SHA2-256 Salt Length: 32 Hash Pair: Hash Algorithm: SHA2-384 Salt Length: 48 Hash Pair: Hash Algorithm: SHA2-512 Salt Length: 64 Hash Pair: Hash Algorithm: SHA2-512/224 Salt Length: 24 Hash Pair: Hash Algorithm: SHA2-512/256 Salt Length: 32 Properties: Modulo: 4096 Hash Pair: Hash Algorithm: SHA-1 Salt Length: 20	

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
			Hash Pair: Hash Algorithm: SHA2-224 Salt Length: 28 Hash Pair: Hash Algorithm: SHA2-256 Salt Length: 32 Hash Pair: Hash Algorithm: SHA2-384 Salt Length: 48 Hash Pair: Hash Algorithm: SHA2-512 Salt Length: 64 Hash Pair: Hash Algorithm: SHA2-512/224 Salt Length: 24 Hash Pair: Hash Algorithm: SHA2-512/256 Salt Length: 32 Public Exponent Mode: Random	
A4593, A5173	Safe Primes Key Generation SP 800-56Ar3	Safe Primes Key Generation	Strength: 112-200 bits Safe Prime Groups: ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192	Key Generation
A4593, A5173	Safe Primes Key Verification SP 800-56Ar3	Safe Primes Key Verification	Strength: 112-200 bits Safe Prime Groups: ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192	Key Verification
A4593	SHA-1 FIPS 180-4	SHA-1	Strength: 80 bits (Legacy) Collision Resistance 128 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2, 4, 8gigabytes	Hashing
A5173	SHA-1 FIPS 180-4	SHA-1	Strength: 80 bits (Legacy) Collision Resistance 128 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2gigabytes	Hashing
A4593	SHA2-224 FIPS 180-4	SHA2-224	Strength: 112 bits Collision Resistance 192 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2, 4, 8gigabytes	Hashing

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A5173	SHA2-224 FIPS 180-4	SHA2-224	Strength: 112 bits Collision Resistance 192 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2gigabytes	Hashing
A4593	SHA2-256 FIPS 180-4	SHA2-256	Strength: 128 bits Collision Resistance 256 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2, 4, 8gigabytes	Hashing
A5173	SHA2-256 FIPS 180-4	SHA2-256	Strength: 128 bits Collision Resistance 256 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2gigabytes	Hashing
A4593	SHA2-384 FIPS 180-4	SHA2-384	Strength: 192 bits Collision Resistance 256 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2, 4, 8gigabytes	Hashing
A5173	SHA2-384 FIPS 180-4	SHA2-384	Strength: 192 bits Collision Resistance 256 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2gigabytes	Hashing
A4593	SHA2-512 FIPS 180-4	SHA2-512	Strength: 256 bits Collision Resistance 256 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2, 4, 8gigabytes	Hashing
A5173	SHA2-512 FIPS 180-4	SHA2-512	Strength: 256 bits Collision Resistance 256 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2gigabytes	Hashing

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A4593	SHA2-512/224 FIPS 180-4	SHA2-512/224	Strength: 112 bits Collision Resistance 192 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2, 4, 8gigabytes	Hashing
A5173	SHA2-512/224 FIPS 180-4	SHA2-512/224	Strength: 112 bits Collision Resistance 192 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2gigabytes	Hashing
A4593	SHA2-512/256 FIPS 180-4	SHA2-512/256	Strength: 128 bits Collision Resistance 256 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2, 4, 8gigabytes	Hashing
A5173	SHA2-512/256 FIPS 180-4	SHA2-512/256	Strength: 128 bits Collision Resistance 256 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2gigabytes	Hashing
A4593	SHA3-224 FIPS 202	SHA3-224	Strength: 112 bits Collision Resistance 192 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2, 4, 8gigabytes	Hashing
A5173	SHA3-224 FIPS 202	SHA3-224	Strength: 112 bits Collision Resistance 192 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2gigabytes	Hashing
A4593	SHA3-256 FIPS 202	SHA3-256	Strength: 128 bits Collision Resistance 256 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2, 4, 8gigabytes	Hashing

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A5173	SHA3-256 FIPS 202	SHA3-256	Strength: 128 bits Collision Resistance 256 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2gigabytes	Hashing
A4593	SHA3-384 FIPS 202	SHA3-384	Strength: 192 bits Collision Resistance 256 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2, 4, 8gigabytes	Hashing
A5173	SHA3-384 FIPS 202	SHA3-384	Strength: 192 bits Collision Resistance 256 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2gigabytes	Hashing
A4593	SHA3-512 FIPS 202	SHA3-512	Strength: 256 bits Collision Resistance 256 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2, 4, 8gigabytes	Hashing
A5173	SHA3-512 FIPS 202	SHA3-512	Strength: 256 bits Collision Resistance 256 bits Pre-Image Resistance Message Length: 0-65536 Increment 8 Large Message Sizes: 1, 2gigabytes	Hashing
A4593, A5173	SHAKE-128 FIPS 202	SHAKE-128	Strength: 128 bits Supports Empty Message Output Length: 16-65536 Increment 8	Hashing
A4593, A5173	SHAKE-256 FIPS 202	SHAKE-256	Strength: 256 bits Supports Empty Message Output Length: 16-65536 Increment 8	Hashing
A4593, A5173	TDES-CBC SP 800-67r2	TDES-CBC	Strength: 112 bits Direction: Decrypt Keying Option: 1	Legacy Decryption
A4593, A5173	TDES-ECB SP 800-67r2	TDES-ECB	Strength: 112 bits Direction: Decrypt Keying Option: 1	Legacy Decryption

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A4593, A5173	TLS v1.2 KDF RFC7627 SP 800-135r1 CVL	TLS v1.2 KDF RFC7627	Strength: 256 bits Hash Algorithm: SHA2-256, SHA2-384, SHA2-512 Key Block Length: 1024	Key Derivation
A4593, A5173	TLS v1.3 KDF RFC 8446 CVL	TLS v1.3 KDF	Strength: 256 bits HMAC Algorithm: SHA2-256, SHA2-384 KDF Running Modes: DHE, PSK, PSK-DHE	Key Derivation

2.6.2 Vendor Affirmed Algorithms

The module implements the vendor affirmed algorithms that are approved for use in Approved mode. Specifically, the module implements CKG per SP 800-133r2 for generation of symmetric keys and asymmetric keys. Refer to the CKG entries in Table 7 for additional details.

2.6.3 Non-Approved, Allowed Algorithms

The module implements the following algorithms that are allowed for use in Approved mode. These are the brainpool curves as listed in SP 800-186 Appendix H.1.

Table 8 - Non-Approved Algorithms Allowed in the Approved Mode of Operation

Algorithm	Caveat	Use / Function
EC Diffie-Hellman with non-NIST recommended curves	Provides 112, 128, 160, 192, or 256 bits of encryption strength. Per IGs D.F and C.A.	Shared secret computation using non-NIST curves: brainpoolP224r1, brainpoolP256r1, brainpoolP320r1, brainpoolP384r1, brainpoolP512r1, with strengths 112 bits, 128 bits, 160 bits, 192 bits, and 256 bits
ECDSA with non-NIST recommended curves	Provides 112, 128, 160, 192, or 256 bits of encryption strength. Per IG C.A.	Key pair generation, digital signature generation, digital signature verification using non-NIST curves: brainpoolP224r1, brainpoolP256r1, brainpoolP320r1, brainpoolP384r1, brainpoolP512r1, with strengths 112 bits, 128 bits, 160 bits, 192 bits, and 256 bits

2.6.4 Non-Approved, Allowed Algorithms with No Security Claimed

Not applicable.

The module does not implement any non-approved algorithms with no security claimed.

2.6.5 Non-Approved, Not Allowed Algorithms

Not applicable.

The module does not implement any non-approved, not allowed algorithms.

2.7 Module Block Diagram

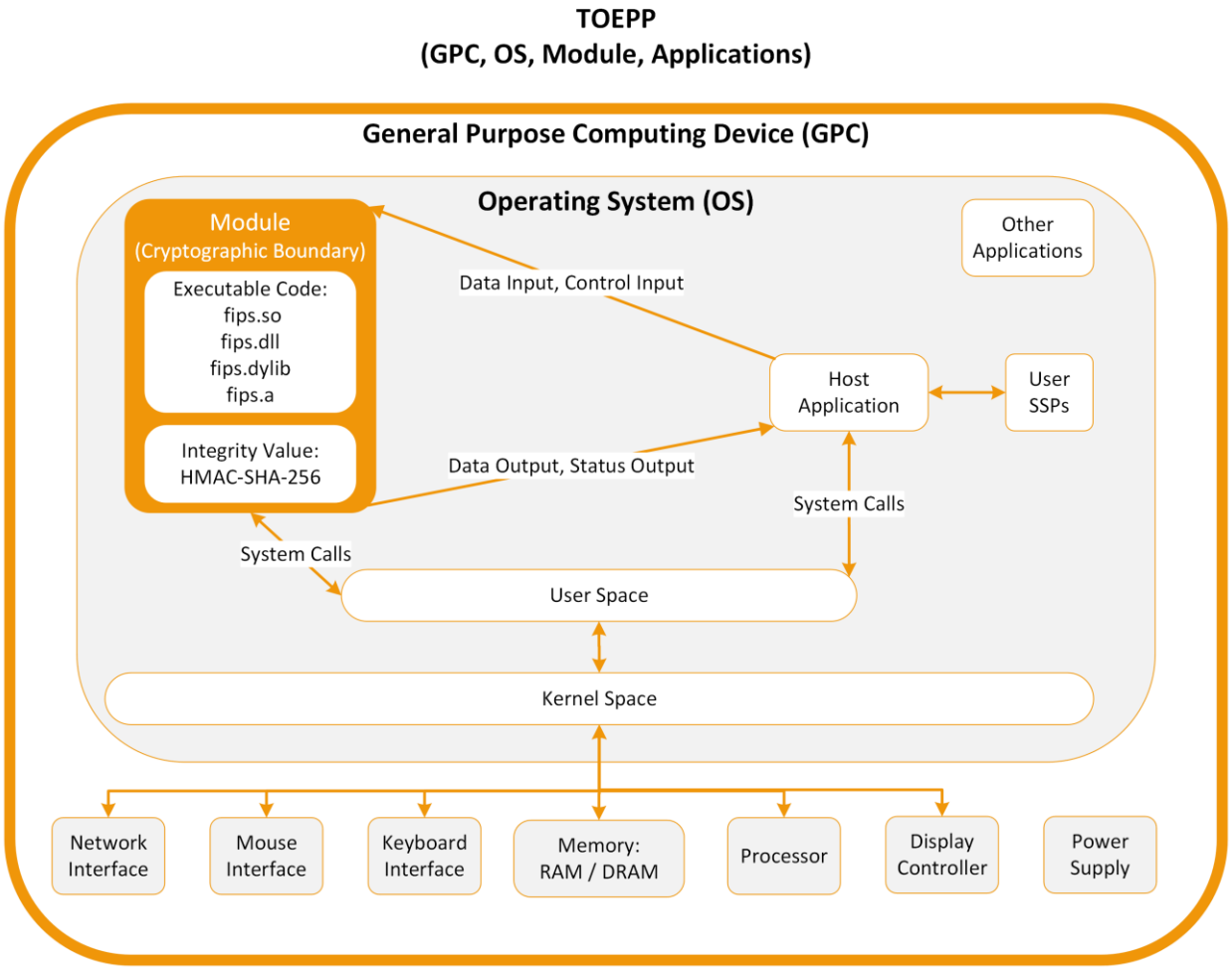

Figure 1 - Module Block Diagram and Cryptographic Boundary

The module's block diagram depicts the cryptographic boundary, TOEPP, and the components of each. Additionally, it depicts the data flow between these components. The module's logical interfaces are defined by its API. These interfaces are used by the host application to interact with the module. All input to the module occurs through the data input interface or control input interface. All output from the module occurs through the data output interface or status output interface. Refer also to Security Policy Section 3 - Cryptographic Module Interfaces and Section 9.2 - SSP Input-Output Methods.

The module executes within the operating environments specified in Security Policy Section 2.3 - Operating Environments.

2.8 Security Function Implementations

Security function implementations (SFIs) are defined by the table below. The module is a software library, therefore the SFIs map directly to the module's services. Refer also to Security Policy Section 4.3 - Approved Services for a description of the module's services and SSP access.

The SFIs below are used to encrypt or decrypt a key value on behalf of the calling application. SSPs are passed in by the calling application. Established SSPs are passed out to the calling application.

Table 9 - Security Function Implementations

Name	Type	Description	SF Properties	Algorithms /CAVP Cert
KeyTransport	KTS	SP 800-56Brev2. KTS-IFC (key encapsulation and un-encapsulation) per IG D.G.	2048 to 16384-bit modulus providing 112 to 256 bits of encryption strength	KTS-IFC A4593, A5173
KeyWrapping	KTS	SP 800-38F. KTS (key wrapping and unwrapping) per IG D.G.	128, 192, and 256-bit keys providing 128, 192, or 256 bits of encryption strength	AES-KW, AES-KWP A4593, A5173

2.9 Algorithm Specific Information

2.9.1 AES-GCM (IG C.H conformance)

The module is compatible with TLS 1.2 and supports AES-GCM IV construction in alignment with IG C.H scenario 1. The module does not implement the TLS 1.2 protocol itself, however, it provides the cryptographic functions required for implementing the protocol, including for AES-GCM cipher suites specified in Section 3.3.1 of SP 800-52r2. AES GCM encryption is used in the context of the TLS 1.2 protocol and the mechanism for IV generation is compliant with RFC 5288. The counter portion of the AES GCM IV is set by the module within its cryptographic boundary. The counter portion of the IV is strictly increasing. When the IV exhausts the maximum number of possible values for a given session key, encryption will fail. A handshake to establish a new encryption key is required. It is the responsibility of the user of the module (i.e., the first party to encounter this condition, either the client or the server) to trigger this handshake in accordance with RFC 5246.

The module supports internal IV generation by the module's approved DRBGs, in alignment with IG C.H scenario 2. The IV is at least 96 bits in length per NIST SP 800-38D, Section 8.2.2.

The module is compatible with TLS 1.3 and supports AES-GCM IV construction in alignment with IG C.H scenario 5. The module does not implement the TLS 1.3 protocol itself, however, it provides the cryptographic functions required for implementing the protocol. AES GCM encryption is used in the context of the TLS 1.3 protocol. When used in the context of TLS 1.3, the GCM IV is constructed in accordance with RFC 8446.

2.9.2 AES-XTS

Per SP 800-38E, AES-XTS should only be used for storage applications.

2.9.3 DSA

DSA KeyGen (FIPS186-4) and DSA PQGGen (FIPS 186-4) are only implemented for use as a part of an approved SP 800-56Ar3 FFC scheme. In accordance with this, only the FIPS 186-type parameter sets FB (2048, 224) and FC (2048, 256) from SP 800-56Arev3 are supported by the module.

For DSA signatures, only DSA PQGVer (FIPS186-4) and DSA SigVer (FIPS186-4) are only implemented.

Refer to Security Policy Section 9.5 - Transitions for additional context.

2.9.4 Edwards Curves

Per FIPS 186-5, Edwards curves are only used for digital signatures using EdDSA. Per FIPS 186-5, only SHA-512 is supported with curve Edwards25519 and only SHAKE256 is supported with curve Edwards448.

2.9.5 PBKDF (IG D.N Conformance)

The PBKDF aligns with Option 1a in Section 5.4 of SP 800-132. Keys derived from passwords using the PBKDF may only be used in storage applications.

The PBKDF function can be called using the Key Derivation service, but it does not establish keys into the module. The PBKDF function supports passwords from 8 to 128 bytes and iteration counts from 1 to 10,000. SP 800-132 Section 5.2 recommends a minimum iteration count of 1,000. Operators should select an appropriate password length and iteration count for their use case, bearing in mind that both should be as large as is feasible for the application.

2.9.6 RSA

RSA SigVer (FIPS186-4) ANSI X9.31 functionality is only implemented for legacy support. Refer to Security Policy Section 9.5 - Transitions for additional context.

The module supports the following even RSA modulus sizes that are not testable by the CAVP:

- For RSA signature generation: 2048-16384
- For RSA signature verification: 1024-16384
- For RSA KAS and RAS KTS per SP 800-56Br2: 2048-16384

All conformance requirements from IG C.F have been met. All implemented modulus sizes for which CAVP testing is available have been validated under CAVP certificates A4593 and A5173. The minimum number of Miller-Rabin tests used in primality testing is conformant with both FIPS 186-4 and FIPS 186-5.

2.9.7 RSA KTS (IG D.G conformance)

For the RSA KTS (KTS-IFC) algorithm, the module supports the KTS-OAEP-basic scheme.

As indicated in Security Policy Section 2.9.6, the module supports even RSA modulus sizes that are not testable by the CAVP. The module supports moduli 2048-16384 for RSA KTS. This is conformant to IG C.F.

2.9.8 TLS 1.2 KDF (IG D.Q conformance)

As indicated under CAVP certificates A4593 and A5173, the module supports TLS 1.2 KDF per RFC 7627, i.e. using the extended master secret.

2.9.9 Triple-DES

TDES-CBC and TDES-ECB Decryption functionality is only implemented for legacy support. Refer to Security Policy Section 9.5 - Transitions for additional context.

2.10 RNG and Entropy

Not applicable

The module does not include an entropy source. The module aligns with IG 9.3.A, scenario 2b, therefore the module's certificate includes the caveat "No assurance of the minimum strength of generated SSPs (e.g., keys)."

2.10.1 Entropy Information

The module accepts input from entropy sources external to the cryptographic boundary for use as seed material for the module's approved DRBG implementations.

Entropy is supplied to the module by means of callback functions. Those functions return an error if the minimum entropy strength is not met. Entropy strength requirements are per NIST Special Publication 800-90A Table 2 (Hash_DRBG, HMAC_DRBG) and Table 3 (CTR_DRBG). At a minimum, the entropy source shall provide at least 128 bits of entropy to the DRBG.

All random values used by the module for approved algorithms are provided by the module's approved DRBGs.

2.10.2 RNG Information

The module does not include an entropy source.

The module includes Counter DRBG, Hash DRBG, and HMAC DRBG, all of which are approved RBGs. The output of these approved RBGs is used to generate random data, symmetric keys, and asymmetric keys, as indicated in Security Policy Section 2.6.2 - Vendor Affirmed Algorithms.

2.11 Key Generation

Any generated SSPs are passed out to the calling application and are not stored in the module. Additional detail is provided in Security Policy Section 4.3 - Approved Services.

Random values for key generation are provided by the module's approved DRBGs.

The output of the module's approved DRBGs may be used to generate symmetric keys per SP 800-133r2 using the Random Number Generation service, as indicated in Security Policy Section 2.6.2 - Vendor Affirmed Algorithms.

The output of the module's approved DRBGs may be used to generate asymmetric keys per FIPS 186-4 and per FIPS 186-5 (for EdDSA only) using the Asymmetric Key Generation service.

2.12 Key Establishment

SSPs used for services are passed in by the calling application. Established SSPs are passed out to the calling application and are not stored in the module. Additional detail is provided in Security Policy Section 4.3 - Approved Services.

The module provides ECC and FFC shared secret computation that is conformant to SP 800-56Ar3 in alignment with IG D.F scenario 2 (path 1) via the Key Agreement service. For ECC, the module supports the (Cofactor) Ephemeral Unified Model, C(2e, 0s, ECC CDH) Scheme described in SP 800-56Ar3 Section 6.1.2.2. For FFC, the module supports the dhEphem, C(2e, 0s, FFC DH) Scheme described in SP 800-56Ar3 Section 6.1.2.1. The module also provides ECC key agreement using the allowed curves specified in Table 8 - Non-Approved Algorithms Allowed in the Approved Mode of Operation in alignment with IG D.F scenario 3 via the Key Agreement service. The appropriate public key validation assurances are implemented. For ECC, full public key validation is implemented (SP 800-56Ar3 Section 5.6.2.3.3). For FFC, both full public key validation (per SP 800-56Ar3 Section 5.6.2.3.1) and partial public key validation (per SP 800-56Ar3 Section 5.6.2.3.2) are implemented.

The module provides RSA shared secret computation that is conformant to SP 800-56Br2 in alignment with IG D.F scenario 1 (path 1) via the Key Agreement service. The module supports the KAS1 basic and KAS2 basic schemes.

The module supports various key derivation functions separately via the Key Derivation service. Supported KDFs are conformant to SP 800-108r1 (KBKDF), SP 800-132 (PBKDF), SP 800-56Cr2 (HKDF, KDA OneStep KDA, TwoStep KDA), SP 800-135r1 (ANSI 9.42 KDF, ANSI 9.63 KDF, SSH KDF, TLS 1.2 KDF), and RFC 8446 (TLS 1.3 KDF).

The module provides RSA key encapsulation that is conformant to SP 800-56Br2 via the Key Transport service (this is the KeyTransport SFI).

The module provides AES key wrapping (AES KW, AES KWP) that is conformant to SP 800-38F via the Key Wrapping service (this is the KeyWrapping SFI).

2.13 Industry Protocols

The module implements KDFs from SP 800-135r1 (Recommendation for Existing Application-Specific Key Derivation Functions) and the TLS 1.3 KDF. These KDFs have been validated by the CAVP and received CVL certificates (A4593, A5173). No parts of these protocols, other than the approved cryptographic algorithms and the KDFs, have been tested by the CAVP and CMVP.

2.14 Design and Rules

The module is designed to meet the applicable requirements of FIPS 140-3. The module initializes when powered on, then performs the pre-operational self-tests and CASTs as specified in Security Policy

Section 10 - Self-Tests. After successfully passing these self-tests, the module automatically transitions to the operational state and awaits service requests.

2.15 Initialisation

The Module Initialization service is executed when the module is powered on.

3 Cryptographic Module Interfaces

3.1 Ports and Interfaces

As a software cryptographic module, the module supports logical interfaces only and not physical ports. All access to the module is through the module's API. The API provides and defines the module's logical interfaces. The API provides functions that may be called by a host application (refer to Security Policy Section 4.3 - Approved Services).

Table 10 - Ports and Interfaces

Physical Port	Logical Interface	Data that passes over port/interface
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 status outputs (return codes, error messages)

The following interfaces are omitted from the table above because they are not applicable to the module: Control Output (not implemented), Power Input (N/A for software modules).

3.2 Additional Information

All interfaces are logically separated by the module's API.

The data output path is inhibited during pre-operational self-tests, zeroisation, and when the module is in an error state.

4 Roles, Services, and Authentication

4.1 Roles

Table 11 - Roles

Name	Type	Operator Type
Crypto Officer	Role	CO

Crypto Officer is the only role supported by the module. The module does not support a User role or a Maintenance role. The Crypto Officer role is implicitly selected by calling the module's services.

Table 12 below describes the service inputs and outputs for the CO role and should be reviewed in conjunction with the service descriptions in Table 14.

Table 12 – Roles, Service Commands, Input and Output

Role	Service	Input	Output
CO	Module Initialization	External dispatch (function pointer) table	Internal dispatch (function pointer) table
CO	Self-Test	None	Module State (queried via Show Status) changes to Running (FIPS_STATE_RUNNING)
CO	Integrity Test	Expected HMAC	Module State (queried via Show Status) changes to Running (FIPS_STATE_RUNNING)
CO	Show Status	None	Module Status: Running (FIPS_STATE_RUNNING), or Error (FIPS_STATE_ERROR)
CO	Output ID/ Version Information	None	name: 140-3 FIPS Provider version: 3.0.0-FIPS 140-3 or 3.0.1-FIPS 140-3 buildinfo: 3.0.0-FIPS 140-3 or 3.0.1-FIPS 140-3
CO	Random Number Generation	Desired security strength in bits, entropy input	Random data
CO	Symmetric Encryption/ Decryption	AES EDK, AES XTS key, TDES DK, IV, ciphertext data, plaintext data	Ciphertext data or plaintext data

Role	Service	Input	Output
CO	Authenticated Symmetric Encryption/ Decryption	AES CMAC/CCM key, AES GMAC/GCM key, ciphertext data, plaintext data	Ciphertext data or plaintext data
CO	Symmetric Digest	Digest or message, AES CMAC/CCM key, AES GMAC/GCM key	Digest or verification result
CO	Asymmetric Key Generation	Desired security strength in bits, entropy input, prediction resistance, parameters and values for FFC, ECC, RSA key generation	ECDSA SGK, ECDSA SVK, RSA SGK, RSA SVK, EdDSA SGK, EdDSA SVK, DH Private, DH Public, ECDH Private, ECDH Public, RSA KAK Private, RSA KAK Public, RSA KDK Private, RSA KEK Public
CO	Digital Signatures	DSA SVK, ECDSA SGK, ECDSA SVK, RSA SGK, RSA SVK, EdDSA SGK, EdDSA SVK	Digital signature or verification result
CO	Keyed Hash	HMAC key, KMAC key	Keyed hash or verification result
CO	Message Digest	Message data	Digest
CO	Key Agreement	DH Private, DH Public, ECDH Private, ECDH Public, RSA KAK Private, RSA KAK Public	DH Private, DH Public, ECDH Private, ECDH Public, RSA KAK Private, RSA KAK Public, KDF secret
CO	Key Derivation	KDF secret, salt, iteration count, MAC, digest, cipher, key	Generic Secret
CO	Key Transport	RSA KEK Public and key to be encapsulated (Generic Secret), or RSA KDK Private and encapsulated key (Generic Secret)	Encapsulated key (Generic Secret), or unencapsulated key (Generic Secret)
CO	Key Wrapping	AES key wrapping key, key to be wrapped or unwrapped (Generic Secret)	Wrapped key or unwrapped key (Generic Secret)
CO	Zeroise	Memory to be cleansed (pointer and length)	The completion of a zeroisation routine indicates that the zeroisation procedure succeeded. Zeroisation can be confirmed via EVP_RAND_verify_zeroization: 1 for success (i.e. the DRBG CSPs have been zeroised), 0 for failure.
CO	Utility	None	None

4.2 Authentication Methods

Table 13 – Roles and Authentication

Role	Authentication Methods	Authentication Strength
Crypto Officer	N/A	N/A

The module does not support authentication.

4.3 Approved Services

The following table describes the services the module provides and the access to SSPs by each service. Additional details on each service are available in the module's user guidance documentation.

SSP Access is divided into the following access types:

- Generate: The module generates or derives the SSP.
- Read: The SSP is read from the module (e.g. the SSP is output).
- Write: The SSP is updated, imported, or written to the module.
- Execute: The module uses the SSP in performing a cryptographic operation.
- Zeroise: The module zeroises the SSP.

In alignment with IG 2.4.C example scenario 2, the module provides a global indicator that services are approved and a status code indicating the completion of each service, as specified in the "Indicator" column of the table below. The module only provides approved services. The successful completion of a service is an implicit indicator for the use of an approved service. Additional detail is provided in Security Policy Section 2.5 - Modes of Operation.

Table 14 - Approved Services

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
Module Initialization	Initialize the FIPS module when it is loaded (calls pre-operational self-tests and CASTs).	None	N/A	CO	N/A	API return value from OSSL_provider_init: 1 for success, 0 for failure
Self-Test	Performs pre-operational self-tests and CASTs on demand.	None	N/A	CO	N/A	API return value code from SELF_TEST_post(): 1 for success, 0 for failure
Integrity Test	Performs the integrity test on demand.	HMAC-SHA-256	N/A	CO	N/A	API return value from verify_integrity(): 1 for verified, 0 for failure
Show Status	Provides status information by querying the “status” parameter.	None	N/A	CO	N/A	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for query operation completed successfully, 0 for failure to query the parameter

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
Output ID/ Version Information	Displays FIPS module version by querying the "version," "name," and "buildinfo" parameters, with the results specified in the output column. The version aligns with the FIPS certificate and Security Policy Section 2.2 - Version Information.	None	N/A	CO	N/A	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for query operation completed successfully, 0 for failure to query the parameter
Random Number Generation	Used to seed/reseed a DRBG instance (including determining the security strength) or obtain random data that is passed out to the calling application.	Counter DRBG Hash DRBG HMAC DRBG	DRBG Entropy Input CTR_DRBG Seed, CTR_DRBG V, CTR_DRBG Key, Hash_DRBG Seed, Hash_DRBG V, Hash_DRBG C, HMAC_DRBG Seed, HMAC_DRBG V, HMAC_DRBG Key	CO	Write, Execute, Zeroise (W, E, Z) Generate, Execute, Zeroise (G, E, Z)	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for operation completed successfully, 0 for failure

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
Symmetric Encryption/ Decryption	Used to encrypt or decrypt data. SSPs are passed in by the calling application.	AES-CBC AES-CBC-CS1 AES-CBC-CS2 AES-CBC-CS3 AES-CFB1 AES-CFB8 AES-CFB128 AES-CTR AES-ECB AES-OFB AES-XTS TDES-CBC TDES-ECB	AES EDK, AES XTS key, TDES DK	CO	Write, Execute, Zeroise (W, E, Z)	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for operation completed successfully, 0 for failure
Authenticated Symmetric Encryption/ Decryption	Used to encrypt or decrypt data or keys. SSPs are passed in by the calling application. Any established SSPs are passed out to the calling application.	AES-CCM AES-GCM	AES CMAC/CCM key, AES GMAC/GCM key AES GMAC/GCM IV	CO	Write, Execute, Zeroise (W, E, Z) Generate, Execute, Zeroise (G, E, Z)	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for operation completed successfully, 0 for failure

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
Symmetric Digest	Used to generate or verify data integrity with CMAC or GMAC. SSPs are passed in by the calling application.	AES-CMAC AES-GMAC	AES CMAC/CCM key, AES GMAC/GCM key AES GMAC/GCM IV	CO	Write, Execute, Zeroise (W, E, Z) Generate, Execute, Zeroise (G, E, Z)	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for operation completed successfully, 0 for failure

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
Asymmetric Key Generation	Used to generate asymmetric keys using the DRBG. Established SSPs are passed out to the calling application.	Prerequisites: Counter DRBG Hash DRBG HMAC DRBG CKG With DSA KeyGen DSA PQGGen DSA PQGVer ECDSA KeyGen ECDSA KeyVer ECDSA with non-NIST recommended curves EDDSA keyGen EDDSA keyVer RSA KeyGen Safe Primes Key Generation Safe Primes Key Verification	DRBG Entropy Input CTR_DRBG Seed, CTR_DRBG V, CTR_DRBG Key, Hash_DRBG Seed, Hash_DRBG V, Hash_DRBG C, HMAC_DRBG Seed, HMAC_DRBG V, HMAC_DRBG Key ECDSA SGK, ECDSA SVK, RSA SGK, RSA SVK, EdDSA SGK, EdDSA SVK, DH Private, DH Public, ECDH Private, ECDH Public, RSA KAK Private, RSA KAK Public, RSA KDK Private, RSA KEK Public	CO	Write, Execute, Zeroise (W, E, Z) Generate, Execute, Zeroise (G, E, Z) Generate, Read, Zeroise (G, R, Z)	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for operation completed successfully, 0 for failure

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
Digital Signatures	Used to generate or verify digital signatures. SSPs are passed in by the calling application.	DSA SigVer ECDSA SigGen ECDSA SigVer ECDSA with non-NIST recommended curves EDDSA SigGen EDDSA SigVer RSA SigGen RSA SigVer	DSA SVK, ECDSA SGK, ECDSA SVK, RSA SGK, RSA SVK, EdDSA SGK, EdDSA SVK	CO	Write, Execute, Zeroise (W, E, Z)	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for operation completed successfully, 0 for failure
Keyed Hash	Used to generate or verify data integrity. SSPs are passed in by the calling application.	HMAC-SHA-1 HMAC-SHA2-224 HMAC-SHA2-256 HMAC-SHA2-384 HMAC-SHA2-512 HMAC-SHA2-512/224 HMAC-SHA2-512/256 HMAC-SHA3-224 HMAC-SHA3-256 HMAC-SHA3-384 HMAC-SHA3-512 KMAC-128 KMAC-256	HMAC key, KMAC key	CO	Write, Execute, Zeroise (W, E, Z)	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for operation completed successfully, 0 for failure

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
Message Digest	Used to generate a SHA-1, SHA-2, SHA-3, or SHAKE message digest.	SHA-1 SHA2-224 SHA2-256 SHA2-384 SHA2-512 SHA2-512/224 SHA2-512/256 SHA3-224 SHA3-256 SHA3-384 SHA3-512 SHAKE-128 SHAKE-256	N/A	CO	N/A	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for operation completed successfully, 0 for failure
Key Agreement	Used to perform key agreement primitives on behalf of the calling application (does not establish keys into the module). SSPs are passed in by the calling application. Established SSPs are passed out to the calling application.	KAS-ECC-SSC EC Diffie-Hellman with non-NIST recommended curves KAS-FFC-SSC KAS-IFC-SSC	DH Private, DH Public, ECDH Private, ECDH Public, RSA KAK Private, RSA KAK Public KDF secret	CO	Write, Read, Execute, Zeroise (W, R, E, Z) Generate, Read, Zeroise (G, R, Z)	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for operation completed successfully, 0 for failure

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
Key Derivation	Used to derive keys using KBKDF, PBKDF, HKDF, SP 800-56Cr2 One-Step KDF (KDA), SP 800-56Cr2 Two-Step KDF (KDA), ANSI X9.42-2001 KDF, ANSI X9.63-2001 KDF, SSHv2 KDF, TLS 1.2 KDF, TLS 1.3 KDF (does not establish keys into the module). SSPs are passed in by the calling application. Established SSPs are passed out to the calling application.	KDA HKDF SP800-56Cr2 KDA OneStep SP800-56Cr2 KDA TwoStep SP800-56Cr2 KDF ANS 9.42 KDF ANS 9.63 KDF KMAC Sp800-108r1 KDF SP800-108 KDF SSH PBKDF TLS v1.2 KDF RFC7627 TLS v1.3 KDF	KDF Secret Generic Secret	CO	Write, Execute, Zeroise (W, E, Z) Generate, Read, Zeroise (G, R, Z)	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for operation completed successfully, 0 for failure

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
Key Transport	Used to encrypt or decrypt a key value on behalf of the calling application (does not establish keys into the module). SSPs are passed in by the calling application. Established SSPs are passed out to the calling application.	KTS-IFC KeyTransport SFI	RSA KDK Private, RSA KEK Public Generic Secret	CO	Write, Execute, Zeroise (W, E, Z) Write, Read, Zeroise (W, R, Z)	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for operation completed successfully, 0 for failure
Key Wrapping	Used to encrypt or decrypt a key value on behalf of the calling application (does not establish keys into the module). SSPs are passed in by the calling application. Established SSPs are passed out to the calling application.	AES-KW AES-KWP KeyWrapping SFI	AES key wrapping key Generic Secret	CO	Write, Execute, Zeroise (W, E, Z) Write, Read, Zeroise (W, R, Z)	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for operation completed successfully, 0 for failure

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
Zeroise	All services automatically overwrite SSPs stored in allocated memory. The module does not store any SSP persistently (beyond the lifetime of an API call), except for DRBG state values (stored for the lifetime of the DRBG instance). Stack cleanup is the responsibility of the calling application.	None	DRBG Entropy Input, CTR_DRBG Seed, CTR_DRBG V, CTR_DRBG Key, Hash_DRBG Seed, Hash_DRBG V, Hash_DRBG C, HMAC_DRBG Seed, HMAC_DRBG V, HMAC_DRBG Key	CO	Zeroise (Z)	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for operation completed successfully, 0 for failure
Utility	Miscellaneous helper functions.	None	N/A	CO	N/A	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for operation completed successfully, 0 for failure

4.4 Non-Approved Services

Not applicable.

The module does not implement any non-approved, not allowed algorithms; therefore, it also does not provide any non-approved services.

4.5 External Software/Firmware Loaded

Not applicable.

The module does not support this functionality.

5 Software/Firmware Security

5.1 Integrity Techniques

As specified in Security Policy Section 2.3.3 - Executable Code Sets, the module implements integrity techniques for all executable code sets. The integrity technique used by the module is HMAC-SHA-256. The integrity technique has received CAVP certificates A4593 and A5173. The integrity technique is implemented by the module itself.

5.2 Initiate on Demand

The Integrity Test can be performed on demand via the “Integrity Test” service.

6 Operational Environment

6.1 Operational Environment Type and Requirements

Refer to Security Policy Section 2.3.4 for vendor affirmed operating environment porting guidance.

6.1.1 Type of Operating Environment

Modifiable

6.1.2 How Requirements are Satisfied

Supported operational environments are indicated in Security Policy Section 2.3 - Operating Environments.

The operating environments ensure that every application using the module operates in its own private and isolated environment (memory, I/O, etc.) and that user processes are segregated into separate process spaces. The module does not spawn any processes.

6.2 Configuration Settings and Restrictions

The module must be installed, and the correct installation confirmed, as described in Security Policy Section 11.1 – Startup Procedures.

No specific configuration options are required for the operational environments. No security rules, settings, or restrictions to the configuration of the operational environment are needed for the module to function in a FIPS-conformant manner.

It is advised to restrict write access to the module and its related configuration file to the administrator role in the operational environment.

7 Physical Security

The requirements of this section are not applicable to the module. The module is a software module and does not implement any physical security mechanisms.

8 Non-Invasive Security

The requirements of this section are not applicable to the module.

9 Sensitive Security Parameter Management

9.1 Storage Areas

Table 15 – Storage Areas

Name	Description	Persistence Type
RAM / DRAM	Memory that only holds data during power on of the operating environment	Dynamic

9.2 SSP Input-Output Methods

Table 16 – SSP Input-Output Methods

Name	From	To	Format Type	Distribution Type	Entry Type	SFI or Algorithm
API Input via TOEPP path	App	CM Software	Plaintext	Manual	Electronic	
Encrypted API Input via TOEPP path	App	CM Software	Encrypted	Manual	Electronic	KeyTransport (KTS-IFC) KeyWrapping (AES KW, AES KWP)
API Output via TOEPP path	CM Software	App	Plaintext	Manual	Electronic	
Encrypted API Output via TOEPP path	CM Software	App	Encrypted	Manual	Electronic	KeyTransport (KTS-IFC) KeyWrapping (AES KW, AES KWP)

The information in the table above aligns with IG 9.5.A. IG 9.5.A indicates that SSPs established by a software cryptographic module to or from a general purpose application that operates outside the module's boundary but within the TOEPP are classified as Manually Distributed using Electronic Entry.

The module does not support any other methods of SSP input or output. Specifically, the module does not support Automated Distribution, Wireless Distribution, or Direct Entry.

The module outputs CSPs in plaintext unless a KeyTransport (RSA) or KeyWrapping (AES) Security Function Implementation (refer to Security Policy Section 2.8 - Security Function Implementations) is used to encrypt the output CSP.

9.3 SSP Zeroisation Methods

Table 17 – SSP Zeroisation Methods

Method	Description	Rationale	Operator Initiation Capability
Zeroise service	Calls OPENSSL_cleanse to zeroise the DRBG CSPs	DRBG CSPs are the only SSPs stored by the module beyond the lifetime of an API call. The Zeroise service zeroises SSPs by overwriting zeroes to the memory location occupied by the SSP and further deallocating that area.	Function provided via API
Call a service that creates or uses the SSP	Services include appropriate APIs (OPENSSL_free or OPENSSL_cleanse) to automatically zeroise the SSPs created or used by the services. This zeroises the context structures that contain the SSP.	SSPs are zeroised by overwriting zeroes to the memory location occupied by the SSP and further deallocating that area.	Function provided via API

As indicated in Table 14 - Approved Services , the completion of a zeroisation routine indicates that the zeroisation procedure succeeded. Zeroisation can be confirmed via EVP_RAND_verify_zeroization: 1 for success (i.e. the DRBG CSPs have been zeroised), 0 for failure.

9.4 SSPs

The following two tables define the module's Sensitive Security Parameters (SSPs). Access to SSPs is defined under Security Policy Section 4.3 - Approved Services.

9.4.1 SSPs

Table 18 – SSPs

Key/SSP Name/Type	Strength	Security Function Cert. #	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
Generic Secret Type: Key or other SSP	112 – 256 bits	N/A	Key Derivation Service (A4593 and A5173): KDA HKDF SP800-56Cr2 KDA OneStep SP800-56Cr2 KDA TwoStep SP800-56Cr2 KDF ANS 9.42 KDF ANS 9.63 KDF KMAC Sp800-108r1 KDF SP800-108 KDF SSH PBKDF TLS v1.2 KDF RFC7627 TLS v1.3 KDF or External	Key Derivation Service: plaintext export Key Transport Service (A4593 and A5173): import or export, plaintext or encrypted with KTS-IFC Key Wrapping Service (A4593 and A5173), import or export, plaintext or encrypted with AES-KW or AES-KWP	Key Transport Service (A4593 and A5173): KTS-IFC Key Wrapping Service (A4593 and A5173): AES-KW AES-KWP	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Note: Used only as an input or output **Related SSPs:** May be derived from KDF Secret May be wrapped or unwrapped by RSA KDK Private, RSA KEK Public, or AES key wrapping key

Key/SSP Name/Type	Strength	Security Function Cert. #	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
AES EDK Type: Symmetric Key	128, 192, 256 bits	A4593 and A5173: AES-CBC AES-CBC-CS1 AES-CBC-CS2 AES-CBC-CS3 AES-CFB1 AES-CFB8 AES-CFB128 AES-CTR AES-ECB AES-OFB	External	Symmetric Encryption/ Decryption service: plaintext import	N/A	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Symmetric Encryption/ Decryption
AES CMAC/CCM key Type: Symmetric Key	128, 192, 256 bits	A4593 and A5173: AES-CCM AES-CMAC	External	Authenticated Symmetric Encryption/ Decryption service: plaintext import Symmetric Digest service: plaintext import	N/A	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Authenticated Symmetric Encryption/ Decryption, Symmetric Digest
AES GMAC/GCM key Type: Symmetric Key	128, 192, 256 bits	A4593 and A5173: AES-GCM AES-GMAC	External	Authenticated Symmetric Encryption/ Decryption service: plaintext import Symmetric Digest service: plaintext import	N/A	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Authenticated Symmetric Encryption/ Decryption, Symmetric Digest **Related SSPs:** AES GMAC/GCM IV: Used with
AES GMAC/GCM IV Type: IV	96-1024 bits	A4593 and A5173: AES-GCM AES-GMAC	Random Number Generation service (A4593 and A5173): Counter DRBG Hash DRBG HMAC DRBG	N/A	N/A	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Authenticated Symmetric Encryption/ Decryption, Symmetric Digest **Related SSPs:** AES GMAC/GCM key: Used with

Key/SSP Name/Type	Strength	Security Function Cert. #	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
AES XTS key Type: Symmetric Key	128, 256 bits	A4593 and A5173: AES-XTS	External	Symmetric Encryption/ Decryption service: plaintext import	N/A	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Symmetric Encryption/ Decryption
AES key wrapping key Type: Symmetric Key	128, 192, 256 bits	A4593 and A5173: AES-KW AES-KWP (KeyWrapping SFI)	External	Key Wrapping service: plaintext import	N/A	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Key Wrapping (using KeyWrapping SFI) **Related SSPs:** Wraps or unwraps Generic Secret
TDES DK Type: Symmetric Key	112 bits	A4593 and A5173: TDES-CBC TDES-ECB	External	Symmetric Encryption/ Decryption service: plaintext import	N/A	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Symmetric Encryption/ Decryption
DRBG Entropy Input Type: RBG	128-256 bits	A4593 and A5173: Counter DRBG Hash DRBG HMAC DRBG	External	Random Number Generation service: plaintext import Asymmetric Key Generation service: plaintext import	N/A	RAM / DRAM All DRBG SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the DRBG instance.	Zeroise service (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Random Number Generation, Asymmetric Key Generation **Related SSPs:** CTR_DRBG Seed, CTR_DRBG V, CTR_DRBG Key, Hash_DRBG Seed, Hash_DRBG V, Hash_DRBG C, HMAC_DRBG Seed, HMAC_DRBG V, HMAC_DRBG Key: Used with

Key/SSP Name/Type	Strength	Security Function Cert. #	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
CTR_DRBG Seed Type: RBG	128-256 bits	A4593 and A5173: Counter DRBG	Random Number Generation service (A4593 and A5173): Counter DRBG Asymmetric Key Generation service (A4593 and A5173): Counter DRBG	N/A	N/A	RAM / DRAM All DRBG SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the DRBG instance.	Zeroise service (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Random Number Generation, Asymmetric Key Generation **Related SSPs:** DRBG Entropy Input, CTR_DRBG V, CTR_DRBG Key: Used with
CTR_DRBG V Type: RBG	128 bits	A4593 and A5173: Counter DRBG	Random Number Generation service (A4593 and A5173): Counter DRBG Asymmetric Key Generation service (A4593 and A5173): Counter DRBG	N/A	N/A	RAM / DRAM All DRBG SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the DRBG instance.	Zeroise service (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Random Number Generation, Asymmetric Key Generation **Related SSPs:** DRBG Entropy Input, CTR_DRBG Seed, CTR_DRBG Key: Used with
CTR_DRBG Key Type: RBG	128, 192, 256 bits	A4593 and A5173: Counter DRBG	Random Number Generation service (A4593 and A5173): Counter DRBG Asymmetric Key Generation service (A4593 and A5173): Counter DRBG	N/A	N/A	RAM / DRAM All DRBG SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the DRBG instance.	Zeroise service (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Random Number Generation, Asymmetric Key Generation **Related SSPs:** DRBG Entropy Input, CTR_DRBG Seed, CTR_DRBG V: Used with

Key/SSP Name/Type	Strength	Security Function Cert. #	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
Hash_DRBG Seed Type: RBG	128-256 bits	A4593 and A5173: Hash DRBG	Random Number Generation service (A4593 and A5173): Hash DRBG Asymmetric Key Generation service (A4593 and A5173): Hash DRBG	N/A	N/A	RAM / DRAM All DRBG SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the DRBG instance.	Zeroise service (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Random Number Generation, Asymmetric Key Generation **Related SSPs:** DRBG Entropy Input, Hash_DRBG V, Hash_DRBG C: Used with
Hash_DRBG V Type: RBG	128, 256 bits	A4593 and A5173: Hash DRBG	Random Number Generation service (A4593 and A5173): Hash DRBG Asymmetric Key Generation service (A4593 and A5173): Hash DRBG	N/A	N/A	RAM / DRAM All DRBG SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the DRBG instance.	Zeroise service (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Random Number Generation, Asymmetric Key Generation **Related SSPs:** DRBG Entropy Input, Hash_DRBG Seed, Hash_DRBG C: Used with
Hash_DRBG C Type: RBG	128, 256 bits	A4593 and A5173: Hash DRBG	Random Number Generation service (A4593 and A5173): Hash DRBG Asymmetric Key Generation service (A4593 and A5173): Hash DRBG	N/A	N/A	RAM / DRAM All DRBG SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the DRBG instance.	Zeroise service (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Random Number Generation, Asymmetric Key Generation **Related SSPs:** DRBG Entropy Input, Hash_DRBG Seed, Hash_DRBG V: Used with

Key/SSP Name/Type	Strength	Security Function Cert. #	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
HMAC_DRBG Seed Type: RBG	128-256 bits	A4593 and A5173: HMAC DRBG	Random Number Generation service (A4593 and A5173): HMAC DRBG Asymmetric Key Generation service (A4593 and A5173): HMAC DRBG	N/A	N/A	RAM / DRAM All DRBG SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the DRBG instance.	Zeroise service (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Random Number Generation, Asymmetric Key Generation **Related SSPs:** DRBG Entropy Input, HMAC_DRBG V, HMAC_DRBG Key: Used with
HMAC_DRBG V Type: RBG	160, 256, 512 bits	A4593 and A5173: HMAC DRBG	Random Number Generation service (A4593 and A5173): HMAC DRBG Asymmetric Key Generation service (A4593 and A5173): HMAC DRBG	N/A	N/A	RAM / DRAM All DRBG SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the DRBG instance.	Zeroise service (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Random Number Generation, Asymmetric Key Generation **Related SSPs:** DRBG Entropy Input, HMAC_DRBG Seed, HMAC_DRBG Key: Used with
HMAC_DRBG Key Type: RBG	160, 256, 512 bits	A4593 and A5173: HMAC DRBG	Random Number Generation service (A4593 and A5173): HMAC DRBG Asymmetric Key Generation service (A4593 and A5173): HMAC DRBG	N/A	N/A	RAM / DRAM All DRBG SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the DRBG instance.	Zeroise service (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Random Number Generation, Asymmetric Key Generation **Related SSPs:** DRBG Entropy Input, HMAC_DRBG Seed, HMAC_DRBG Key: Used with

Key/SSP Name/Type	Strength	Security Function Cert. #	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
ECDSA SGK Type: Signature	112 – 256 bits	A4593 and A5173: ECDSA SigGen ECDSA with non-NIST recommended curves (allowed)	Asymmetric Key Generation service (A4593 and A5173): Counter DRBG Hash DRBG HMAC DRBG CKG (Vendor affirmed) ECDSA KeyGen ECDSA KeyVer or External	Digital Signatures service: plaintext import Asymmetric Key Generation service: plaintext export	N/A	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Digital Signatures **Related SSPs:** May be paired with ECDSA SVK
RSA SGK Type: Signature	112 – 256 bits	A4593 and A5173: RSA SigGen	Asymmetric Key Generation service (A4593 and A5173): Counter DRBG Hash DRBG HMAC DRBG CKG (Vendor affirmed) RSA KeyGen or External	Digital Signatures service: plaintext import Asymmetric Key Generation service: plaintext export	N/A	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Digital Signatures **Related SSPs:** May be paired with RSA SVK
EdDSA SGK Type: Signature	128, 224 bits	A4593 and A5173: EDDSA SigGen	Asymmetric Key Generation service (A4593 and A5173): Counter DRBG Hash DRBG HMAC DRBG CKG (Vendor affirmed) EDDSA keyGen EDDSA keyVer or External	Digital Signatures service: plaintext import Asymmetric Key Generation service: plaintext export	N/A	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Digital Signatures **Related SSPs:** May be paired with EdDSA SVK

Key/SSP Name/Type	Strength	Security Function Cert. #	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
DH Private Type: Key Agreement	112 bits 112-200 bits	A4593 and A5173: KAS-FFC-SSC	Asymmetric Key Generation service (A4593 and A5173): Counter DRBG Hash DRBG HMAC DRBG CKG (Vendor affirmed) DSA KeyGen DSA PQGGen DSA PQGVer Safe Primes Key Generation Safe Primes Key Verification or External	Key Agreement service: plaintext import/export Asymmetric Key Generation service: plaintext export	Key Agreement service (A4593 and A5173): KAS-FFC-SSC	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Key Agreement **Related SSPs:** May be paired with DH Public Used to establish KDF Secret
EC DH Private Type: Key Agreement	112 – 256 bits	A4593 and A5173: KAS-ECC-SSC EC Diffie-Hellman with non-NIST recommended curves (allowed)	Asymmetric Key Generation service (A4593 and A5173): Counter DRBG Hash DRBG HMAC DRBG CKG (Vendor affirmed) ECDSA KeyGen ECDSA KeyVer or External	Key Agreement service: plaintext import/export Asymmetric Key Generation service: plaintext export	Key Agreement service (A4593 and A5173): KAS-ECC-SSC EC Diffie-Hellman with non-NIST recommended curves (allowed)	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Key Agreement **Related SSPs:** May be paired with EC DH Public Used to establish KDF Secret
RSA KAK Private Type: Key Agreement	112 – 256 bits	A4593 and A5173: KAS-IFC-SSC	Asymmetric Key Generation service (A4593 and A5173): Counter DRBG Hash DRBG HMAC DRBG CKG (Vendor affirmed) RSA KeyGen or External	Key Agreement service: plaintext import/export Asymmetric Key Generation service: plaintext export	Key Agreement service (A4593 and A5173): KAS-IFC-SSC	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Key Agreement **Related SSPs:** May be paired with RSA KAK Public Used to establish KDF Secret

Key/SSP Name/Type	Strength	Security Function Cert. #	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
RSA KDK Private Type: Key Transport	112 – 256 bits	A4593 and A5173: KTS-IFC (KeyTransport SFI)	Asymmetric Key Generation service (A4593 and A5173): Counter DRBG Hash DRBG HMAC DRBG CKG (Vendor affirmed) RSA KeyGen or External	Key Transport service: plaintext import Asymmetric Key Generation service: plaintext export	N/A	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Key Transport (using KeyTransport SFI) **Related SSPs:** May be paired with RSA KEK Public Unwraps Generic Secret
HMAC Key Type: Authentication	128, 192 256 bits	A4593 and A5173: HMAC-SHA-1 HMAC-SHA2-224 HMAC-SHA2-256 HMAC-SHA2-384 HMAC-SHA2-512 HMAC-SHA2-512/224 HMAC-SHA2-512/256 HMAC-SHA3-224 HMAC-SHA3-256 HMAC-SHA3-384 HMAC-SHA3-512	External	Keyed Hash service: plaintext import	N/A	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Keyed Hash
KMAC Key Type: Authentication	128, 256 bits	A4593 and A5173: KMAC-128 KMAC-256	External	Keyed Hash service: plaintext import	N/A	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Keyed Hash

Key/SSP Name/Type	Strength	Security Function Cert. #	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
KDF Secret Type: Key Derivation Function	112 – 512 bits	A4593 and A5173: KDA HKDF SP800-56Cr2 KDA OneStep SP800-56Cr2 KDA TwoStep SP800-56Cr2 KDF ANS 9.42 KDF ANS 9.63 KDF KMAC Sp800-108r1 KDF SP800-108 KDF SSH PBKDF TLS v1.2 KDF RFC7627 TLS v1.3 KDF	N/A	Key Derivation service: plaintext import Key Agreement service: plaintext export	Key Agreement Service (A4593 and A5173): KAS-ECC-SSC EC Diffie-Hellman with non-NIST recommended curves (allowed) KAS-FFC-SSC KAS-IFC-SSC or External	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Key Derivation **Related SSPs:** May be derived from DH Private, DH Public, or EC DH Private, EC DH Public, or RSA KAK Private, RSA KAK Public May be used to derive Generic Secret
DSA SVK Type: Signature	80 – 128	A4593 and A5173: DSA SigVer	External	Digital Signatures service: plaintext import	N/A	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Digital Signatures
ECDSA SVK Type: Signature	80 – 256 bits	A4593 and A5173: ECDSA SigVer ECDSA with non-NIST recommended curves (allowed)	Asymmetric Key Generation service (A4593 and A5173): Counter DRBG Hash DRBG HMAC DRBG CKG (Vendor affirmed) ECDSA KeyGen ECDSA KeyVer or External	Digital Signatures service: plaintext import Asymmetric Key Generation service: plaintext export	N/A	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Digital Signatures **Related SSPs:** May be paired with ECDSA SGK

Key/SSP Name/Type	Strength	Security Function Cert. #	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
RSA SVK Type: Signature	80 – 256 bits	A4593 and A5173: RSA SigVer	Asymmetric Key Generation service (A4593 and A5173): Counter DRBG Hash DRBG HMAC DRBG CKG (Vendor affirmed) RSA KeyGen or External	Digital Signatures service: plaintext import Asymmetric Key Generation service: plaintext export	N/A	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Digital Signatures **Related SSPs:** May be paired with RSA SGK
EdDSA SVK Type: Signature	128, 224 bits	A4593 and A5173: EDDSA SigVer	Asymmetric Key Generation service (A4593 and A5173): Counter DRBG Hash DRBG HMAC DRBG CKG (Vendor affirmed) EDDSA keyGen EDDSA keyVer or External	Digital Signatures service: plaintext import Asymmetric Key Generation service: plaintext export	N/A	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Digital Signatures **Related SSPs:** May be paired with EdDSA SGK
DH Public Type: Key Agreement	112 bits 112-200 bits	A4593 and A5173: KAS-FFC-SSC	Asymmetric Key Generation service (A4593 and A5173): Counter DRBG Hash DRBG HMAC DRBG CKG (Vendor affirmed) DSA KeyGen DSA PQGGen DSA PQGVer Safe Primes Key Generation Safe Primes Key Verification or External	Key Agreement service: plaintext import/export Asymmetric Key Generation service: plaintext export	Key Agreement service (A4593 and A5173): KAS-FFC-SSC	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Key Agreement **Related SSPs:** May be paired with DH Private Used to establish KDF Secret

Key/SSP Name/Type	Strength	Security Function Cert. #	Generation	Import / Export	Establishment	Storage	Zeroisation	Use & related keys
EC DH Public Type: Key Agreement	112 – 256 bits	A4593 and A5173: KAS-ECC-SSC EC Diffie-Hellman with non-NIST recommended curves (allowed)	Asymmetric Key Generation service (A4593 and A5173): Counter DRBG Hash DRBG HMAC DRBG CKG (Vendor affirmed) ECDSA KeyGen ECDSA KeyVer or External	Key Agreement service: plaintext import/export Asymmetric Key Generation service: plaintext export	Key Agreement service (A4593 and A5173): KAS-ECC-SSC EC Diffie-Hellman with non-NIST recommended curves (allowed)	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Key Agreement **Related SSPs:** May be paired with EC DH Private Used to establish KDF Secret
RSA KAK Public Type: Key Agreement	112 – 256 bits	A4593 and A5173: KAS-IFC-SSC	Asymmetric Key Generation service (A4593 and A5173): Counter DRBG Hash DRBG HMAC DRBG CKG (Vendor affirmed) RSA KeyGen or External	Key Agreement service: plaintext import/export Asymmetric Key Generation service: plaintext export	Key Agreement service (A4593 and A5173): KAS-IFC-SSC	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Key Agreement **Related SSPs:** May be paired with RSA KAK Private Used to establish KDF Secret
RSA KEK Public Type: Key Transport	112 – 256 bits	A4593 and A5173: KTS-IFC (KeyTransport SFI)	Asymmetric Key Generation service (A4593 and A5173): Counter DRBG Hash DRBG HMAC DRBG CKG (Vendor affirmed) RSA KeyGen or External	Key Transport service: plaintext import Asymmetric Key Generation service: plaintext export	N/A	RAM / DRAM All SSPs are temporarily stored in plaintext. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP (Refer to Table 17 – SSP Zeroisation Methods)	**Use:** Key Transport (using KeyTransport SFI) **Related SSPs:** May be paired with RSA KDK Private Wraps Generic Secret

9.4.1 SSPs, Additional Details

The table below provides additional detail for the SSPs listed in Table 18 – SSPs.

Table 19 – SSPs, Additional Details

Key/SSP Name/Type	Description	Size	Category
Generic Secret Type: Key or other SSP	SSPs generated by key derivation and directly output by the module, or generic keys that are wrapped or transported and directly output by the module. Note: when the encrypted item is not a key or other SSP, it is denoted as “data” instead of as the Generic Secret SSP.	112 – 512 bits	CSP
AES EDK Type: Symmetric Key	AES encrypt/ decrypt key	128, 192, 256 bits	CSP
AES CMAC/CCM key Type: Symmetric Key	AES CMAC/CCM key for encrypt/ decrypt or generate/ verify	128, 192, 256 bits	CSP
AES GMAC/GCM key Type: Symmetric Key	AES GMAC/GCM key for encrypt/ decrypt or generate/ verify	128, 192, 256 bits	CSP
AES GMAC/GCM IV Type: IV	AES GMAC/GCM IV for encrypt/ decrypt or generate/ verify	96-1024 bits	CSP
AES XTS key Type: Symmetric Key	AES XTS encrypt/ decrypt key	128, 256 bits	CSP
AES key wrapping key Type: Symmetric Key	AES KW, KWP key	128, 192, 256 bits	CSP
TDES DK Type: Symmetric Key	3-key Triple-DES decrypt key	192 bits	CSP

Key/SSP Name/Type	Description	Size	Category
DRBG Entropy Input Type: RBG	Entropy Input	128-1024 bits (length is dependent on the requested security strength, per SP 800-90A Table 2 and Table 3)	CSP
CTR_DRBG Seed Type: RBG	CTR_DRBG seed, constructed from entropy input and other inputs per SP 800-90A Sections 7.2, 8.6	256-896 bits	CSP
CTR_DRBG V Type: RBG	V, internal state	128 bits	CSP
CTR_DRBG Key Type: RBG	Key (AES), internal state	128, 192, 256 bits	CSP
Hash_DRBG Seed Type: RBG	Hash_DRBG seed, constructed from entropy input and other inputs per SP 800-90A Sections 7.2, 8.6	224-736 bits	CSP
Hash_DRBG V Type: RBG	V, internal state	440, 888 bits	CSP
Hash_DRBG C Type: RBG	C, internal state	440, 888 bits	CSP
HMAC_DRBG Seed Type: RBG	HMAC_DRBG seed, constructed from entropy input and other inputs per SP 800-90A Sections 7.2, 8.6	224-1408 bits	CSP
HMAC_DRBG V Type: RBG	V, internal state	160, 256, 512 bits	CSP
HMAC_DRBG Key Type: RBG	Key (HMAC), internal state	160, 256, 512 bits	CSP
ECDSA SGK Type: Signature	ECDSA signature generation key (P, B, K curves and brainpool)	224 – 512 bits	CSP
RSA SGK Type: Signature	RSA signature generation key	2048 – 16384 bits	CSP
EdDSA SGK Type: Signature	Ed25519 or Ed448 signature generation key	256, 456 bits	CSP

Key/SSP Name/Type	Description	Size	Category
DH Private Type: Key Agreement	Diffie-Hellman private key agreement key (186-4-type and safe primes)	For 186-4 type key generation: 224, 256 bits For safe primes key generation: ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192	CSP
EC DH Private Type: Key Agreement	Elliptic Curve Diffie-Hellman private key agreement key (P, B, K curves and brainpool)	224 – 512 bits	CSP
RSA KAK Private Type: Key Agreement	RSA private key agreement key	2048 – 16384 bits	CSP
RSA KDK Private Type: Key Transport	RSA private key decryption key	2048 – 16384 bits	CSP
HMAC Key Type: Authentication	Keyed hash key for HMAC	160, 224, 256, 384, 512 bits	CSP
KMAC Key Type: Authentication	Keyed hash key for KMAC	128-1024 bits	CSP
KDF Secret Type: Key Derivation Function	Secret value used by KDFs	112 – 512 bits	CSP
DSA SVK Type: Signature	DSA signature verification key (legacy only)	DSA (L, N) = (512 ≤ L < 2048, 160 ≤ N < 224) (2048, 224) (2048, 256) (3072, 256)	PSP
ECDSA SVK Type: Signature	ECDSA signature verification key (P, B, K curves and brainpool)	160 – 512 bits	PSP

Key/SSP Name/Type	Description	Size	Category
RSA SVK Type: Signature	RSA signature verification key	1024 – 16384 bits	PSP
EdDSA SVK Type: Signature	Ed25519 or Ed448 signature verification key	256, 456 bits	PSP
DH Public Type: Key Agreement	Diffie-Hellman public key agreement key (186-4-type and safe primes)	For 186-4 type key generation: 2048 bits For safe primes key generation: ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192	PSP
EC DH Public Type: Key Agreement	Elliptic Curve Diffie-Hellman public key agreement key (P, B, K curves and brainpool)	224 – 512 bits	PSP
RSA KAK Public Type: Key Agreement	RSA public key agreement key	2048 – 16384 bits	PSP
RSA KEK Public Type: Key Transport	RSA public key encryption key	2048 – 16384 bits	PSP

9.5 Transitions

All algorithms implemented by the module are approved for FIPS 140-3 and will not be impacted by the transitions specified below.

The information below provides context for the algorithms not supported by the module due to algorithm transitions. Refer also to Security Policy Section 2.9 – Algorithm Specific Information.

After December 31, 2023, Triple-DES transitioned to non-approved (refer to SP 800-131Ar2.). After December 31, 2023, the following functionality remains approved for legacy use. Because this functionality remains approved, this is the only Triple-DES functionality supported by the module.

- Triple-DES decryption remains approved for legacy use

After February 3, 2024, DSA and RSA X9.31 transitioned to non-approved for all new FIPS module submissions (refer to FIPS 186-4 and IG C.K). Although this validation was submitted to the CMVP before February 3, 2024, the module only implements the DSA and RSA X9.31 functionality that remains approved for submissions after this transition:

- DSA primes and group generators used exclusively in a SP 800-56Ar3-compliant scheme remain approved
- DSA verification remains approved
- RSA X9.31 verification remains approved

After December 31, 2030, SHA-1 transitions to non-approved (refer to NIST announcements). Because this functionality remains approved currently, this functionality is still supported by the module.

10 Self-Tests

10.1 Pre-Operational Self-Tests

Table 20 – Pre-Operational Self-Tests

Algorithm	Implementation	Test Properties	Test Method	Type	Indicator	Details
HMAC-SHA2-256	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	HMAC-SHA-256[1]	Compare to pre-computed HMAC	SW/FW Integrity	The Module State (queried via Show Status) changes to Running (FIPS_STATE_RUNNING)	Verify

The module only performs one pre-operational self-test, which is the software/firmware integrity test. The module does not implement any other pre-operational self-tests, including pre-operational self-tests for bypass or critical functions, because the module does not implement corresponding functions.

The pre-operational self-tests are executed automatically by the Module Initialization service when the module is powered on. Automatic execution of the pre-operational self-tests relies on use of the default entry point (DEP); no operator intervention is required.

For the pre-operational self-tests, the module performs an HMAC-SHA-256 CAST, and then verifies the integrity of the runtime executable using a HMAC-SHA-256 digest computed at build time. If the digests match, the CAST tests are then performed. The integrity technique (HMAC-SHA-256) has received CAVP certificates A4593 and A5173.

10.2 Conditional Self-Tests

The module mainly performs two types of conditional self-tests, which are Cryptographic Algorithm Self-Tests (CASTs) and Pairwise Consistency Tests (PCTs). The module also performs one critical function test for AES-XTS, per IG C.I. The module does not implement any other conditional

[1] Please refer also to the HMAC-SHA-256 CAST, which is performed before the pre-operational SW integrity test

self-tests, including conditional self-tests for software/firmware loading, manual entry, or bypass, because the module does not implement corresponding functions.

The CAST tests below are executed automatically by the Module Initialization service when the module is powered on. Automatic execution of the CASTs relies on use of the default entry point (DEP); no operator intervention is required.

The CASTs execute before the module transitions to the operational state. If the CASTs are successful, the Module State (queried via Show Status) is updated to indicate that the module is in the Running state (FIPS_STATE_RUNNING).

All other conditional self-tests are executed when the relevant condition occurs, as specified in the table below.

Table 21 - Conditional Self-Tests

Algorithm	Implementation	Test Properties	Test Method	Type	Indicator	Details	Condition
AES-GCM	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	256-bit AES	KAT	CAST	The Module State (queried via Show Status) changes to Running (FIPS_STATE_RUNNING)	Authenticated Encrypt (forward cipher)	Initialisation
AES-GCM	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	256-bit AES	KAT	CAST	The Module State changes to Running	Decrypt (forward cipher)	Initialisation
AES-ECB	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	256-bit AES	KAT	CAST	The Module State changes to Running	Decrypt (inverse cipher)	Initialisation
Counter DRBG	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	128-bit AES with df	KAT	CAST	The Module State changes to Running	Instantiate, Reseed, Generate (per IG 10.3.A, 7)	Initialisation
Hash DRBG	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	SHA-256	KAT	CAST	The Module State changes to Running	Instantiate, Reseed, Generate (per IG 10.3.A, 7)	Initialisation

Algorithm	Implementation	Test Properties	Test Method	Type	Indicator	Details	Condition
HMAC DRBG	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	HMAC-SHA-1	KAT	CAST	The Module State changes to Running	Instantiate, Reseed, Generate (per IG 10.3.A, 7)	Initialisation
KDF ANS 9.42	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	SHA-1	KAT	CAST	The Module State changes to Running	Derive	Initialisation
KDF ANS 9.63	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	SHA-256	KAT	CAST	The Module State changes to Running	Derive	Initialisation
KDF SSH	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	SHA-1	KAT	CAST	The Module State changes to Running	Derive	Initialisation
TLS v1.2 KDF RFC7627	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	HMAC-SHA-256	KAT	CAST	The Module State changes to Running	Derive	Initialisation
TLS v1.3 KDF	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	SHA-256	KAT	CAST	The Module State changes to Running	Derive	Initialisation
DSA SigVer (FIPS186-4)	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	2048, SHA-256	KAT	CAST	The Module State changes to Running	Verify	Initialisation
ECDSA SigGen (FIPS186-4)	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	P-224, SHA-512	KAT	CAST	The Module State changes to Running	Sign	Initialisation

Algorithm	Implementation	Test Properties	Test Method	Type	Indicator	Details	Condition
ECDSA SigGen (FIPS186-4)	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	K-233, SHA-512	KAT	CAST	The Module State changes to Running	Sign	Initialisation
Brainpool curves for ECDSA signatures	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	brainpoolP224r1, SHA-512	KAT	CAST	The Module State changes to Running	Sign	Initialisation
ECDSA SigVer (FIPS186-4)	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	P-224, SHA-512	KAT	CAST	The Module State changes to Running	Verify	Initialisation
ECDSA SigVer (FIPS186-4)	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	K-233, SHA-512	KAT	CAST	The Module State changes to Running	Verify	Initialisation
Brainpool curves for ECDSA signatures	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	brainpoolP224r1, SHA-512	KAT	CAST	The Module State changes to Running	Verify	Initialisation
EDDSA sigGen	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	Ed25519	KAT	CAST	The Module State changes to Running	Sign	Initialisation
EDDSA sigGen	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	Ed448	KAT	CAST	The Module State changes to Running	Sign	Initialisation
EDDSA sigVer	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	Ed25519	KAT	CAST	The Module State changes to Running	Verify	Initialisation

Algorithm	Implementation	Test Properties	Test Method	Type	Indicator	Details	Condition
EDDSA sigVer	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	Ed448	KAT	CAST	The Module State changes to Running	Verify	Initialisation
HMAC-SHA2-256	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	HMAC-SHA-256[2]	KAT	CAST	The Module State changes to Running	Verify	Initialisation, performed before pre-operational integrity test
KAS-ECC-SSC Sp800-56Ar3	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	P-256	KAT	CAST	The Module State changes to Running	Verify computation of shared secret Z in Ephemeral Unified scheme[3]	Initialisation
KAS-FFC-SSC Sp800-56Ar3	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	FB (2048, 224)	KAT	CAST	The Module State changes to Running	Verify computation of shared secret Z in dhEphem scheme[4]	Initialisation
KAS-IFC-SSC	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	2048-bit key	KAT	CAST	The Module State changes to Running	RSA Primitive Computation[5]	Initialisation

[2] Also serves as a self-test for SHA-256 and covers the self-test requirement for SHA-224
[3] Per Scenario 2 of IG D.F and Section 6 of SP 800-56Ar3. Also covers the self-test requirement for EC Diffie-Hellman with non-NIST recommended curves.
[4] Per Scenario 2 of IG D.F and Section 6 of SP 800-56Ar3
[5] Per Scenario 1 of IG D.F and Section 8.2.2 in SP 800-56Br2

Algorithm	Implementation	Test Properties	Test Method	Type	Indicator	Details	Condition
KDA OneStep SP800-56Cr2	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	SHA-224	KAT	CAST	The Module State changes to Running	Derive	Initialisation
KDA TwoStep SP800-56Cr2[6]	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	SHA-256	KAT	CAST	The Module State changes to Running	Derive	Initialisation
KDF SP800-108	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	Counter Mode with HMAC-SHA-256	KAT	CAST	The Module State changes to Running	Derive	Initialisation
KTS-IFC	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	2048-bit key	KAT	CAST	The Module State changes to Running	Encrypt for KTS-OAEP-Basic[7]	Initialisation
KTS-IFC	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	2048-bit key	KAT	CAST	The Module State changes to Running	Decrypt for KTS-OAEP-Basic[8]	Initialisation
KTS-IFC	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	2048-bit key	KAT	CAST	The Module State changes to Running	Decrypt for CRT[9]	Initialisation

[6] Also serves as the CAST for KDA HKDF SP800-56Cr2
[7] Per IG D.G and SP 800-56Br2
[8] Per IG D.G and SP 800-56Br2
[9] Per IG D.G and SP 800-56Br2

Algorithm	Implementation	Test Properties	Test Method	Type	Indicator	Details	Condition
PBKDF	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	HMAC-SHA-1	KAT	CAST	The Module State changes to Running	Derivation of the Master Key (MK)[10]	Initialisation
RSA SigGen (FIPS186-4)	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	2048-bit key, SHA-256, PKCS#1	KAT	CAST	The Module State changes to Running	Sign	Initialisation
RSA SigVer (FIPS186-4)	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	2048-bit key, SHA-256, PKCS#1	KAT	CAST	The Module State changes to Running	Verify	Initialisation
SHA3-256	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	SHA3-256[11]	KAT	CAST	The Module State changes to Running	Hash	Initialisation
SHA-1	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	SHA-1	KAT	CAST	The Module State changes to Running	Hash	Initialisation
SHA2-512	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	SHA-512[12]	KAT	CAST	The Module State changes to Running	Hash	Initialisation
TDES-CBC	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	CBC mode, 3-key	KAT	CAST	The Module State changes to Running	Decrypt	Initialisation

[10] Per Section 5.3 of SP 800-132
[11] Also serves as a self-test for KMAC and SHAKE since it utilizes the same Keccak-p permutation
[12] Also covers the self-test requirements for SHA-384, SHA-512/224, and SHA-512/256. SHA-256 is tested by the HMAC-SHA-256 self-test.

Algorithm	Implementation	Test Properties	Test Method	Type	Indicator	Details	Condition
DSA KeyGen (FIPS186-4), Safe Primes Key Generation, Safe Primes Key Verification	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	All supported parameters for KAS-FFC	PCT	PCT	Return value for the relevant API call (i.e. for key pair generation or key pair import): 1 for success, 0 for failure	Sign/Verify for Key Agreement[13]	Key Pair Generation, Key Pair Import
ECDSA KeyGen (FIPS186-4)	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	All supported curves	PCT	PCT	Return value for the relevant API call (i.e. for key pair generation): 1 for success, 0 for failure	Sign/Verify for Digital Signatures[14]	Key Pair Generation
ECDSA KeyGen (FIPS186-4)	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	All supported curves	PCT	PCT	Return value for the relevant API call (i.e. for key pair import): 1 for success, 0 for failure	Sign/Verify for Key Agreement[15]	Key Pair Import
EDDSA keyGen	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	All supported curves (Ed25519, Ed448)	PCT	PCT	Return value for the relevant API call (i.e. for key pair generation or key pair import): 1 for success, 0 for failure	Sign/Verify for Digital Signatures[16]	Key Pair Generation, Key Pair Import

[13] Per VE10.35.03

[14] Per VE10.35.02. At the time of key pair generation, the keys' intended usage is not known (key pairs may be used for digital signatures or key agreement); per IG 10.3.A comment 1, any of the AS10.35 PCTs is acceptable.

[15] Per VE10.35.03. At the time of key pair import, the keys' intended usage is not known (key pairs may be used for digital signatures or key agreement); per IG 10.3.A comment 1, any of the AS10.35 PCTs is acceptable.

[16] Per VE10.35.02. EdDSA keys can only be used for digital signatures.

Algorithm	Implementation	Test Properties	Test Method	Type	Indicator	Details	Condition
RSA KeyGen (FIPS186-4)	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	All supported moduli	PCT	PCT	Return value for the relevant API call (i.e. for key pair generation or key pair import): 1 for success, 0 for failure	Sign/Verify for Key Agreement[17]	Key Pair Generation, Key Pair Import
AES-XTS	CryptoComply 140-3 FIPS Provider (CAVP Cert. #A4593 and Cert. #A5173)	All supported sizes (128-bit, 256-bit)	Other	Critical Function	Return value for the relevant API call (i.e. for symmetric encryption/decryption with AES-XTS): 1 for success, 0 for failure	Test that Key_1 ≠ Key_2 [18]	Symmetric Encryption/ Decryption

[17] Per VE10.35.03. At the time of key pair generation or import, the keys' intended usage is not known (key pairs may be used for key transport, digital signatures, or key agreement); per IG 10.3.A comment 1, any of the AS10.35 PCTs is acceptable.
[18] Per IG C.I

10.3 Periodic Self-Tests

The module provides several methods for the operator to perform periodic self-tests on demand.

Table 22 - Periodic Information

Period	Periodic Method
On demand. It is recommended to run the periodic tests at least annually.	Pre-Operational Periodic tests are called by power cycling the module, calling the Integrity Test service, or calling the Self-Test service (which calls the module's integrity test and all CASTs). Conditional Periodic tests are called by power cycling the module or calling the Self-Test service (which calls the module's integrity test and all CASTs).

10.4 Error States

Table 23 - Error States

State Name	Description	Conditions	Recovery Method	Indicator
FIPS_STATE_ERROR	The module has entered an error state. All cryptographic APIs will return an error when called.	The error state is triggered if any pre-operational or CAST self-tests fail to pass (either on power on or when called by operator)	Restart the module	Module State (queried via Show Status) changes to Error (FIPS_STATE_ERROR)
Temporary Error	The module enters a temporary error state when a PCT test fails or when the AES-XTS critical function test fails. Keys that fail the tests are disabled and the module returns to the Running state.	The error state is triggered if a conditional PCT test or conditional AES-XTS critical function test fails to pass.	The module will reject the tested key or key pair and then return automatically to the Running state (FIPS_STATE_RUNNING).	The return value for the relevant API call (i.e. for key pair generation, key pair import, or symmetric encryption/ decryption with AES-XTS) returns 0 for failure

The module supports two error states, both triggered by failures of the module's self-tests. The module must be restarted to recover from a failure of the pre-operational or CAST self-test, but it recovers automatically from a failure in the other conditional self-tests.

10.5 Operator Initiation

Self-tests can be called on demand using the Self-Test service. This service calls the module's integrity test and all CASTs (i.e. KATs). PCTs are not called by this service; PCTs are only called under the conditions specified in Section 10.2 - Conditional Self-Tests.

The integrity test is automatically called as part of the Pre-Operational Self-Tests and can also be manually called by the Integrity Test service (or the Self-Test service).

11 Life-Cycle Assurance

11.1 Startup Procedures

Failure to follow initialization instructions for the module (provided below) will result in the module being in a non-compliant state.

The module is only provided to the end user in the form of a compiled binary file. Its source code is not provided.

The module is provided to the end user by the vendor as a binary archive and an associated hash value. The end user should validate the integrity of the binary archive against the SHA-256 hash value provided with the binary archive. If the integrity value for the archive is correct, the archive should be extracted, and the binaries should be installed.

The module is a FIPS-validated cryptographic provider for use by OpenSSL 3.x. OpenSSL 3.x should be installed per its documentation prior to module installation. The FIPS module may be installed using the following procedure:

1. If the module is provided as a dynamic library:
    a. Copy the module binary and configuration files to the OpenSSL Provider Directory, e.g. [OPENSSL_INSTALL_LOCATION]/lib/ossl-modules/
2. If the module is provided as a static library:
    a. Copy the module library archive and configuration file to the OpenSSL directory, e.g. [OPENSSL_INSTALL_LOCATION]/lib/ and [OPENSSL_INSTALL_LOCATION]/conf/
3. If the module is provided as part of an XCFramework bundle:
    a. Integrate the module framework into an XCode application and install the application on an iOS device.
    b. Note, when provided as a XCFramework bundle (fips.xcframework), the OpenSSL framework (openssl.xcframework) should also be integrated into the application project. The OpenSSL framework is a general implementation of the OpenSSL 3.x API (common and crypto).
4. To initialize and start up the module, use the OSSL_PROVIDER_load API call from OpenSSL. An example is specified below:

```
int main(int argc, char **argv) {
    OSSL_PROVIDER *fips_provider;

    fips_provider = OSSL_PROVIDER_load(NULL, "fips");
    if (fips_provider == NULL) {
        printf("Could not load FIPS provider\n");
        return 1;
    }
    printf("Provider %s loaded \n", OSSL_PROVIDER_get0_name(fips_provider));
```

```
        //Execute commands for the FIPS module

        if (fips_provider != NULL)
            OSSL_PROVIDER_unload(fips_provider);
        return 0;
    }
```

After the module starts up, the operator should confirm that the module outputs the Approved mode status indicator (refer to Security Policy Section 2.5 - Modes of Operation) and verify the module's version using the "Output ID/ Version Information" service (refer to Security Policy Section 4.3 - Approved Services).

11.2 Administrator Guidance

Additional administrator guidance is provided separately in other operator documentation, including the User Manual.

11.3 Non-Administrator Guidance

If the module power is lost and restored, the operator shall establish a new key for use with AES-GCM encryption/decryption. Refer also to Security Policy Section 2.9.1 - AES-GCM (IG C.H conformance).

Additional guidance is provided separately in other operator documentation, including the User Manual.

11.4 End of Life

The vendor documentation (User Guide) specifies the procedures for the removal of the FIPS module and secure sanitization of the device that the module was installed on.

12 Mitigation of Other Attacks

12.1 Attack List

The module implements two types of mitigations of other attacks, which are constant-time implementations and numeric blinding.

Constant-time implementations protect cryptographic implementations in the module against timing analysis. With this mitigation, variations in execution time cannot be traced back to an SSP, key, or secret data.

Numeric Blinding protects RSA, DSA, and ECDSA from timing attacks, where attackers measure the time of signature operations or RSA decryption. To mitigate this attack, the module generates a random blinding factor that is provided as an input to the decryption/signature operation and is discarded once the operation has completed. With this mitigation, the execution time cannot be correlated to the RSA, DSA, or ECDSA key via a timing attack because the attacker does not know the blinding factor.

12.2 Mitigation Effectiveness

These mitigations should make the timing of the encryption, decryption, and signing operations independent of the key material or the input data. This should prevent an attacker from recovering information by measuring the timing of these operations.

12.3 Guidance and Constraints

While the module implements countermeasures to prevent timing analysis and timing attacks, other side-channel attacks may be possible. As a Level 1, software-based module, the module is limited in its ability to prevent access at the hardware level; power analysis attacks may be possible for an attacker with physical access. Users of software-based modules should be aware of these limitations and incorporate this information into their threat model.