Minimus

Minimus Cryptographic Module

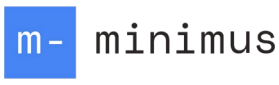
FIPS 140-3 Non-Proprietary Security Policy

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

Document Version 1.0

July 18, 2025

Prepared For:



Minimus
10202 Perkins Rowe, Suite E 160
Baton Rouge, LA 70810
USA
https://www.minimus.io

Prepared By:

SafeLogic, Inc.
8300 Boone Blvd., Suite 500
Vienna, VA 22182
USA
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 Minimus Cryptographic Module.

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 Minimus Cryptographic Module from Minimus (Minimus) 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.

Minimus Cryptographic Module may also be referred to as the "module" in this document.

1.1.3 External Resources

The Minimus website (https://www.minimus.io) contains information on Minimus services and products. The CMVP website maintains all FIPS 140 certificates for Minimus's FIPS 140 validations. These certificates also include Minimus contact information.

1.1.4 Notices

1.2 Security Levels

The following table lists the module's level of validation for each area in 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

Table 1: Security Levels

2 Cryptographic Module Specification

2.1 Description

Purpose and Use:

Minimus FIPS Cryptographic Module is a standards-based cryptographic engine for containers, VMs, and apps that run on cloud native platforms like Docker and Kubernetes.

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

Module Type: Software

Module Embodiment: Multi-Chip Standalone

Module Characteristics:

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 Security Policy Section 2.2 - Tested and Vendor Affirmed Module Version and Identification. Additionally, the module's integrity value is included inside the boundary.

Refer to the block diagram in

Figure 1 for additional detail.

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 TOEPP 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 for additional detail.

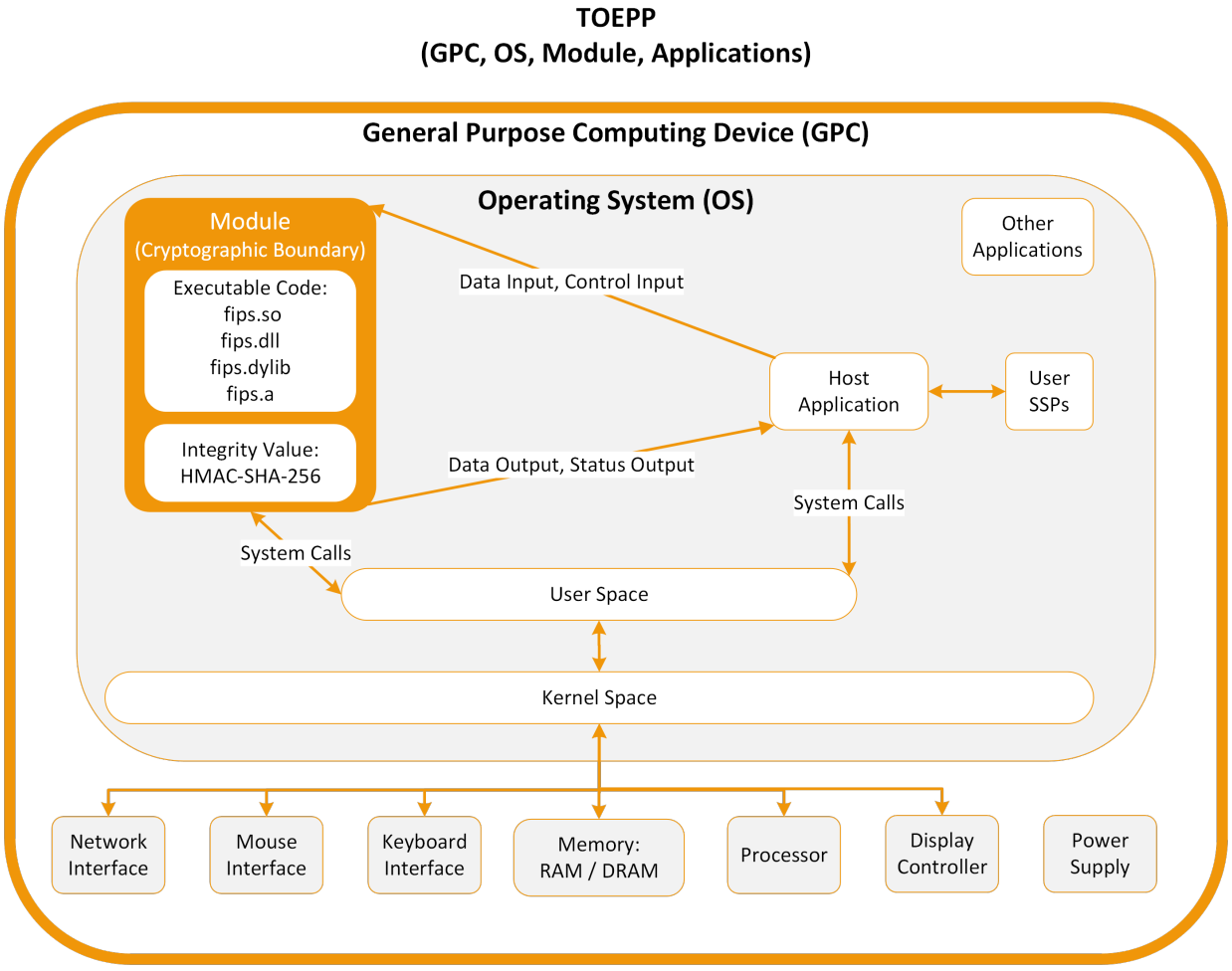


Figure 1: Block Diagram

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 2.2 - Tested and Vendor Affirmed Module Version and Identification.

2.2 Tested and Vendor Affirmed Module Version and Identification

Tested Module Identification – Hardware:

N/A for this module.

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

Package or File Name	Software/ Firmware Version	Features	Integrity Test
fips.a	3.0.1-FIPS 140-3	Compiled as a static library, tested on iOS and iPadOS	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.so	3.0.0-FIPS 140-3	Compiled for Linux, Unix, Android	HMAC-SHA-256

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

Note: module versions 3.0.0-FIPS 140-3, 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 in the latter version.

Tested Module Identification – Hybrid Disjoint Hardware:

N/A for this module.

Tested Operational Environments - Software, Firmware, Hybrid:

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.

Operating System	Hardware Platform	Processors	PAA/PAI	Hypervisor or Host OS	Version(s)
AlmaLinux 9	Dell PowerEdge R830	Intel Xeon E5-4667v4	Yes		3.0.0-FIPS 140-3
AlmaLinux 9	Dell PowerEdge R830	Intel Xeon E5-4667v4	No		3.0.0-FIPS 140-3
Android 13	Google Pixel 7	Google Tensor G2	No		3.0.0-FIPS 140-3
Debian 11	Dell PowerEdge R830	Intel Xeon E5-4667v4	Yes		3.0.0-FIPS 140-3

Operating System	Hardware Platform	Processors	PAA/PAI	Hypervisor or Host OS	Version(s)
Debian 11	Dell PowerEdge R830	Intel Xeon E5-4667v4	No		3.0.0-FIPS 140-3
FreeBSD 13	Dell PowerEdge R830	Intel Xeon E5-4667v4	Yes		3.0.0-FIPS 140-3
FreeBSD 13	Dell PowerEdge R830	Intel Xeon E5-4667v4	No		3.0.0-FIPS 140-3
iOS 16	iPhone 13 Mini	Apple A15 Bionic	No		3.0.1-FIPS 140-3
iPadOS 16	iPad Air (2022)	Apple M1	No		3.0.1-FIPS 140-3
macOS 13 (Ventura)	Mac Mini M2	Apple M2	No		3.0.0-FIPS 140-3
Oracle Solaris 11.4	Dell PowerEdge R830	Intel Xeon E5-4667v4	Yes		3.0.0-FIPS 140-3
Oracle Solaris 11.4	Dell PowerEdge R830	Intel Xeon E5-4667v4	No		3.0.0-FIPS 140-3
Red Hat Enterprise Linux 9	Dell PowerEdge R830	Intel Xeon E5-4667v4	Yes		3.0.0-FIPS 140-3
Red Hat Enterprise Linux 9	Dell PowerEdge R830	Intel Xeon E5-4667v4	No		3.0.0-FIPS 140-3
Rocky Linux 9	Dell PowerEdge R830	Intel Xeon E5-4667v4	Yes		3.0.0-FIPS 140-3
Rocky Linux 9	Dell PowerEdge R830	Intel Xeon E5-4667v4	No		3.0.0-FIPS 140-3
SUSE Linux Enterprise Server 15	Dell PowerEdge R830	Intel Xeon E5-4667v4	Yes		3.0.0-FIPS 140-3
SUSE Linux Enterprise Server 15	Dell PowerEdge R830	Intel Xeon E5-4667v4	No		3.0.0-FIPS 140-3
Ubuntu 22.04	Dell PowerEdge R830	Intel Xeon E5-4667v4	Yes		3.0.0-FIPS 140-3

Operating System	Hardware Platform	Processors	PAA/PAI	Hypervisor or Host OS	Version(s)
Ubuntu 22.04	Dell PowerEdge R830	Intel Xeon E5-4667v4	No		3.0.0-FIPS 140-3
Windows 10	Dell PowerEdge R830	Intel Xeon E5-4667v4	Yes		3.0.0-FIPS 140-3
Windows 10	Dell PowerEdge R830	Intel Xeon E5-4667v4	No		3.0.0-FIPS 140-3
Windows 11	Dell PowerEdge R830	Intel Xeon E5-4667v4	Yes		3.0.0-FIPS 140-3
Windows 11	Dell PowerEdge R830	Intel Xeon E5-4667v4	No		3.0.0-FIPS 140-3
Windows Server 2019	Dell PowerEdge R830	Intel Xeon E5-4667v4	Yes		3.0.0-FIPS 140-3
Windows Server 2019	Dell PowerEdge R830	Intel Xeon E5-4667v4	No		3.0.0-FIPS 140-3
Windows Server 2022	Dell PowerEdge R830	Intel Xeon E5-4667v4	Yes		3.0.0-FIPS 140-3
Windows Server 2022	Dell PowerEdge R830	Intel Xeon E5-4667v4	No		3.0.0-FIPS 140-3

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

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

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 tested operating environments listed above for which the module was tested as a shared object or dynamically loaded library.

Operating System	Hardware Platform
AlmaLinux 9	Any general-purpose platform that supports this OS
Android 13	Any general-purpose platform that supports this OS
Debian 11	Any general-purpose platform that supports this OS
FreeBSD 13	Any general-purpose platform that supports this OS
iOS 16	Any general-purpose platform that supports this OS

Operating System	Hardware Platform
iPadOS 16	Any general-purpose platform that supports this OS
macOS 13 (Ventura)	Any general-purpose platform that supports this OS
Oracle Solaris 11.4	Any general-purpose platform that supports this OS
Red Hat Enterprise Linux 9	Any general-purpose platform that supports this OS
Rocky Linux 9	Any general-purpose platform that supports this OS
SUSE Linux Enterprise Server 15	Any general-purpose platform that supports this OS
Ubuntu 22.04	Any general-purpose platform that supports this OS
Windows 10	Any general-purpose platform that supports this OS
Windows 11	Any general-purpose platform that supports this OS
Windows Server 2019	Any general-purpose platform that supports this OS
Windows Server 2022	Any general-purpose platform that supports this OS

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

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.

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

2.3 Excluded Components

Not applicable.

2.4 Modes of Operation

Modes List and Description:

Mode Name	Description	Type	Status Indicator
Approved Mode	Single approved mode of operation. No non-approved mode is implemented in the module.	Approved	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.

Table 5: Modes List and Description

Mode Change Instructions and Status:

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 Algorithms

2.5.1 Approved Algorithms

Approved Algorithms:

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

Algorithm	CAVP Cert	Properties	Reference
AES-CBC	A4593	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC	A5173	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS1	A4593	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A

Algorithm	CAVP Cert	Properties	Reference
AES-CBC-CS1	A5173	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS2	A4593	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS2	A5173	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS3	A4593	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS3	A5173	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CCM	A4593	Key Length - 128, 192, 256	SP 800-38C
AES-CCM	A5173	Key Length - 128, 192, 256	SP 800-38C
AES-CFB1	A4593	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CFB1	A5173	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CFB128	A4593	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CFB128	A5173	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CFB8	A4593	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CFB8	A5173	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CMAC	A4593	Direction - Generation, Verification Key Length - 128, 192, 256	SP 800-38B
AES-CMAC	A5173	Direction - Generation, Verification Key Length - 128, 192, 256	SP 800-38B
AES-CTR	A4593	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A

Algorithm	CAVP Cert	Properties	Reference
AES-CTR	A5173	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-ECB	A4593	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-ECB	A5173	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-GCM	A4593	Direction - Decrypt, Encrypt IV Generation - External, Internal IV Generation Mode - 8.2.1, 8.2.2 Key Length - 128, 192, 256	SP 800-38D
AES-GCM	A5173	Direction - Decrypt, Encrypt IV Generation - External, Internal IV Generation Mode - 8.2.1, 8.2.2 Key Length - 128, 192, 256	SP 800-38D
AES-GMAC	A4593	Direction - Decrypt, Encrypt IV Generation - External, Internal IV Generation Mode - 8.2.1, 8.2.2 Key Length - 128, 192, 256	SP 800-38D
AES-GMAC	A5173	Direction - Decrypt, Encrypt IV Generation - External, Internal IV Generation Mode - 8.2.1, 8.2.2 Key Length - 128, 192, 256	SP 800-38D
AES-KW	A4593	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-KW	A5173	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-KWP	A4593	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-KWP	A5173	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-OFB	A4593	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A

Algorithm	CAVP Cert	Properties	Reference
AES-OFB	A5173	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-XTS Testing Revision 2.0	A4593	Direction - Decrypt, Encrypt Key Length - 128, 256	SP 800-38E
AES-XTS Testing Revision 2.0	A5173	Direction - Decrypt, Encrypt Key Length - 128, 256	SP 800-38E
Counter DRBG	A4593	Prediction Resistance - Yes Mode - AES-128, AES-192, AES-256 Derivation Function Enabled - Yes	SP 800-90A Rev. 1
Counter DRBG	A5173	Prediction Resistance - Yes Mode - AES-128, AES-192, AES-256 Derivation Function Enabled - Yes	SP 800-90A Rev. 1
DSA KeyGen (FIPS186-4)	A4593	L - 2048 N - 224, 256	FIPS 186-4
DSA KeyGen (FIPS186-4)	A5173	L - 2048 N - 224, 256	FIPS 186-4
DSA PQGGen (FIPS186-4)	A4593	L - 2048 N - 224, 256 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256	FIPS 186-4
DSA PQGGen (FIPS186-4)	A5173	L - 2048 N - 224, 256 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256	FIPS 186-4
DSA PQGVer (FIPS186-4)	A4593	L - 1024, 2048 N - 160, 224, 256 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256	FIPS 186-4
DSA PQGVer (FIPS186-4)	A5173	L - 1024, 2048 N - 160, 224, 256 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256	FIPS 186-4
DSA SigVer (FIPS186-4)	A4593	L - 1024, 2048, 3072 N - 160, 224, 256	FIPS 186-4

Algorithm	CAVP Cert	Properties	Reference
		Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256	
DSA SigVer (FIPS186-4)	A5173	L - 1024, 2048, 3072 N - 160, 224, 256 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256	FIPS 186-4
ECDSA KeyGen (FIPS186-4)	A4593	Curve - B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 Secret Generation Mode - Testing Candidates	FIPS 186-4
ECDSA KeyGen (FIPS186-4)	A5173	Curve - B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 Secret Generation Mode - Testing Candidates	FIPS 186-4
ECDSA KeyVer (FIPS186-4)	A4593	Curve - B-163, B-233, B-283, B-409, B-571, K-163, K-233, K-283, K-409, K-571, P-192, P-224, P-256, P-384, P-521	FIPS 186-4
ECDSA KeyVer (FIPS186-4)	A5173	Curve - B-163, B-233, B-283, B-409, B-571, K-163, K-233, K-283, K-409, K-571, P-192, P-224, P-256, P-384, P-521	FIPS 186-4
ECDSA SigGen (FIPS186-4)	A4593	Component - No Curve - B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	FIPS 186-4
ECDSA SigGen (FIPS186-4)	A5173	Component - No Curve - B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	FIPS 186-4
ECDSA SigVer (FIPS186-4)	A4593	Component - No Curve - B-163, B-233, B-283, B-409, B-571, K-163, K-233, K-283, K-409, K-571, P-192, P-224, P-256, P-384, P-521 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	FIPS 186-4

Algorithm	CAVP Cert	Properties	Reference
ECDSA SigVer (FIPS186-4)	A5173	Component - No Curve - B-163, B-233, B-283, B-409, B-571, K-163, K-233, K-283, K-409, K-571, P-192, P-224, P-256, P-384, P-521 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	FIPS 186-4
EDDSA KeyGen	A4593	Curve - ED-25519, ED-448	FIPS 186-5
EDDSA KeyGen	A5173	Curve - ED-25519, ED-448	FIPS 186-5
EDDSA KeyVer	A4593	Curve - ED-25519, ED-448	FIPS 186-5
EDDSA KeyVer	A5173	Curve - ED-25519, ED-448	FIPS 186-5
EDDSA SigGen	A4593	Curve - ED-25519, ED-448 PreHash - Yes	FIPS 186-5
EDDSA SigGen	A5173	Curve - ED-25519, ED-448 PreHash - Yes	FIPS 186-5
EDDSA SigVer	A4593	Curve - ED-25519, ED-448 PreHash - No Pure - Yes	FIPS 186-5
EDDSA SigVer	A5173	Curve - ED-25519, ED-448 PreHash - No Pure - Yes	FIPS 186-5
Hash DRBG	A4593	Prediction Resistance - Yes Mode - SHA-1, SHA2-256, SHA2-512	SP 800-90A Rev. 1
Hash DRBG	A5173	Prediction Resistance - Yes Mode - SHA-1, SHA2-256, SHA2-512	SP 800-90A Rev. 1
HMAC DRBG	A4593	Prediction Resistance - Yes Mode - SHA-1, SHA2-256, SHA2-512	SP 800-90A Rev. 1
HMAC DRBG	A5173	Prediction Resistance - Yes Mode - SHA-1, SHA2-256, SHA2-512	SP 800-90A Rev. 1
HMAC-SHA-1	A4593	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA-1	A5173	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1

Algorithm	CAVP Cert	Properties	Reference
HMAC-SHA2-224	A4593	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-224	A5173	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-256	A4593	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-256	A5173	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-384	A4593	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-384	A5173	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512	A4593	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512	A5173	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512/224	A4593	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512/224	A5173	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512/256	A4593	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512/256	A5173	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA3-224	A4593	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA3-224	A5173	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA3-256	A4593	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1

Algorithm	CAVP Cert	Properties	Reference
HMAC-SHA3-256	A5173	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA3-384	A4593	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA3-384	A5173	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA3-512	A4593	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA3-512	A5173	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
KAS-ECC-SSC Sp800-56Ar3	A4593	Domain Parameter Generation Methods - B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 Scheme - ephemeralUnified - KAS Role - initiator, responder	SP 800-56A Rev. 3
KAS-ECC-SSC Sp800-56Ar3	A5173	Domain Parameter Generation Methods - B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 Scheme - ephemeralUnified - KAS Role - initiator, responder	SP 800-56A Rev. 3
KAS-FFC-SSC Sp800-56Ar3	A4593	Domain Parameter Generation Methods - FB, FC, ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192 Scheme - dhEphem - KAS Role - initiator, responder	SP 800-56A Rev. 3
KAS-FFC-SSC Sp800-56Ar3	A5173	Domain Parameter Generation Methods - FB, FC, ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192 Scheme - dhEphem - KAS Role - initiator, responder	SP 800-56A Rev. 3
KAS-IFC-SSC	A4593	Modulo - 2048, 3072, 4096, 6144, 8192 Key Generation Methods - rsakpg1-basic, rsakpg1-crt, rsakpg1-prime-factor, rsakpg2-basic, rsakpg2-crt, rsakpg2-prime-factor Scheme -	SP 800-56A Rev. 3

Algorithm	CAVP Cert	Properties	Reference
		KAS1 - KAS Role - initiator, responder KAS2 - KAS Role - initiator, responder	
KAS-IFC-SSC	A5173	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	SP 800-56A Rev. 3
KDA HKDF SP800-56Cr2	A4593	Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-8192 Increment 8 HMAC Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	SP 800-56C Rev. 2
KDA HKDF SP800-56Cr2	A5173	Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-8192 Increment 8 HMAC Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	SP 800-56C Rev. 2
KDA OneStep SP800-56Cr2	A4593	Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-8192 Increment 8	SP 800-56C Rev. 2
KDA OneStep SP800-56Cr2	A5173	Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-8192 Increment 8	SP 800-56C Rev. 2
KDA TwoStep SP800-56Cr2	A4593	MAC Salting Methods - default, random KDF Mode - feedback Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-8192 Increment 8	SP 800-56C Rev. 2
KDA TwoStep SP800-56Cr2	A5173	MAC Salting Methods - default, random KDF Mode - feedback Derived Key Length - 2048	SP 800-56C Rev. 2

Algorithm	CAVP Cert	Properties	Reference
		Shared Secret Length - Shared Secret Length: 224-8192 Increment 8	
KDF ANS 9.42 (CVL)	A4593	KDF Type - DER Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512 Key Data Length - Key Data Length: 8-4096 Increment 8	SP 800-135 Rev. 1
KDF ANS 9.42 (CVL)	A5173	KDF Type - DER Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512 Key Data Length - Key Data Length: 8-4096 Increment 8	SP 800-135 Rev. 1
KDF ANS 9.63 (CVL)	A4593	Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512 Key Data Length - Key Data Length: 128, 4096	SP 800-135 Rev. 1
KDF ANS 9.63 (CVL)	A5173	Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512 Key Data Length - Key Data Length: 128, 4096	SP 800-135 Rev. 1
KDF KMAC Sp800-108r1	A4593	Derived Key Length - Derived Key Length: 112-4096 Increment 8	SP 800-108 Rev. 1
KDF KMAC Sp800-108r1	A5173	Derived Key Length - Derived Key Length: 112-4096 Increment 8	SP 800-108 Rev. 1
KDF SP800-108	A4593	KDF Mode - Counter, Feedback Supported Lengths - Supported Lengths: 8, 72, 128, 776, 3456, 4096	SP 800-108 Rev. 1
KDF SP800-108	A5173	KDF Mode - Counter, Feedback Supported Lengths - Supported Lengths: 8, 72, 128, 776, 3456, 4096	SP 800-108 Rev. 1
KDF SSH (CVL)	A4593	Cipher - AES-128, AES-192, AES-256 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
KDF SSH (CVL)	A5173	Cipher - AES-128, AES-192, AES-256 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
KMAC-128	A4593	Message Length - Message Length: 0-65536 Increment 8 Key Data Length - Key Data Length: 128-1024 Increment 8	SP 800-185

Algorithm	CAVP Cert	Properties	Reference
KMAC-128	A5173	Message Length - Message Length: 0-65536 Increment 8 Key Data Length - Key Data Length: 128-1024 Increment 8	SP 800-185
KMAC-256	A4593	Message Length - Message Length: 0-65536 Increment 8 Key Data Length - Key Data Length: 128-1024 Increment 8	SP 800-185
KMAC-256	A5173	Message Length - Message Length: 0-65536 Increment 8 Key Data Length - Key Data Length: 128-1024 Increment 8	SP 800-185
KTS-IFC	A4593	Modulo - 2048, 3072, 4096, 6144 Key Generation Methods - rsakpg1-basic, rsakpg1-crt, rsakpg1-prime-factor, rsakpg2-basic, rsakpg2-crt, rsakpg2-prime-factor Scheme - KTS-OAEP-basic - KAS Role - initiator, responder Key Transport Method - Key Length - 1024	SP 800-56B Rev. 2
KTS-IFC	A5173	Modulo - 2048, 3072, 4096, 6144 Key Generation Methods - rsakpg1-basic, rsakpg1-crt, rsakpg1-prime-factor, rsakpg2-basic, rsakpg2-crt, rsakpg2-prime-factor Scheme - KTS-OAEP-basic - KAS Role - initiator, responder Key Transport Method - Key Length - 1024	SP 800-56B Rev. 2
PBKDF	A4593	Iteration Count - Iteration Count: 1-10000 Increment 1 Password Length - Password Length: 8-128 Increment 8	SP 800-132
PBKDF	A5173	Iteration Count - Iteration Count: 1-10000 Increment 1 Password Length - Password Length: 8-128 Increment 8	SP 800-132
RSA KeyGen (FIPS186-4)	A4593	Key Generation Mode - B.3.3 Modulo - 2048, 3072, 4096 Primality Tests - Table C.2 Private Key Format - Standard	FIPS 186-4
RSA KeyGen (FIPS186-4)	A5173	Key Generation Mode - B.3.3 Modulo - 2048, 3072, 4096 Primality Tests - Table C.2 Private Key Format - Standard	FIPS 186-4

Algorithm	CAVP Cert	Properties	Reference
RSA SigGen (FIPS186-4)	A4593	Signature Type - PKCS 1.5, PKCSPSS Modulo - 2048, 3072, 4096	FIPS 186-4
RSA SigGen (FIPS186-4)	A5173	Signature Type - PKCS 1.5, PKCSPSS Modulo - 2048, 3072, 4096	FIPS 186-4
RSA SigVer (FIPS186-4)	A4593	Signature Type - ANSI X9.31, PKCS 1.5, PKCSPSS Modulo - 1024, 2048, 3072, 4096	FIPS 186-4
RSA SigVer (FIPS186-4)	A5173	Signature Type - ANSI X9.31, PKCS 1.5, PKCSPSS Modulo - 1024, 2048, 3072, 4096	FIPS 186-4
Safe Primes Key Generation	A4593	Safe Prime Groups - ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192	SP 800-56A Rev. 3
Safe Primes Key Generation	A5173	Safe Prime Groups - ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192	SP 800-56A Rev. 3
Safe Primes Key Verification	A4593	Safe Prime Groups - ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192	SP 800-56A Rev. 3
Safe Primes Key Verification	A5173	Safe Prime Groups - ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192	SP 800-56A Rev. 3
SHA-1	A4593	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA-1	A5173	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2	FIPS 180-4
SHA2-224	A4593	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-224	A5173	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2	FIPS 180-4
SHA2-256	A4593	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-256	A5173	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2	FIPS 180-4

Algorithm	CAVP Cert	Properties	Reference
SHA2-384	A4593	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-384	A5173	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2	FIPS 180-4
SHA2-512	A4593	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-512	A5173	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2	FIPS 180-4
SHA2-512/224	A4593	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-512/224	A5173	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2	FIPS 180-4
SHA2-512/256	A4593	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-512/256	A5173	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2	FIPS 180-4
SHA3-224	A4593	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHA3-224	A5173	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2	FIPS 202
SHA3-256	A4593	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHA3-256	A5173	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2	FIPS 202
SHA3-384	A4593	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHA3-384	A5173	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2	FIPS 202
SHA3-512	A4593	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202

Algorithm	CAVP Cert	Properties	Reference
SHA3-512	A5173	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2	FIPS 202
SHAKE-128	A4593	Output Length - Output Length: 16-65536 Increment 8	FIPS 202
SHAKE-128	A5173	Output Length - Output Length: 16-65536 Increment 8	FIPS 202
SHAKE-256	A4593	Output Length - Output Length: 16-65536 Increment 8	FIPS 202
SHAKE-256	A5173	Output Length - Output Length: 16-65536 Increment 8	FIPS 202
TDES-CBC	A4593	Direction - Decrypt	SP 800-67 Rev. 2
TDES-CBC	A5173	Direction - Decrypt	SP 800-67 Rev. 2
TDES-ECB	A4593	Direction - Decrypt	SP 800-67 Rev. 2
TDES-ECB	A5173	Direction - Decrypt	SP 800-67 Rev. 2
TLS v1.2 KDF RFC7627 (CVL)	A4593	Hash Algorithm - SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
TLS v1.2 KDF RFC7627 (CVL)	A5173	Hash Algorithm - SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
TLS v1.3 KDF (CVL)	A4593	HMAC Algorithm - SHA2-256, SHA2-384 KDF Running Modes - DHE, PSK, PSK-DHE	SP 800-135 Rev. 1
TLS v1.3 KDF (CVL)	A5173	HMAC Algorithm - SHA2-256, SHA2-384 KDF Running Modes - DHE, PSK, PSK-DHE	SP 800-135 Rev. 1

Table 6: Approved Algorithms

2.5.2 Vendor-Affirmed Algorithms

Vendor-Affirmed Algorithms:

The module implements the following vendor affirmed algorithms that are approved for use in Approved mode.

Name	Properties	Implementation	Reference
CKG	Key Type:Symmetric and Asymmetric	N/A	SP 800-133r2 and IG D.H: Per Section 4, example 1
CKG (XTS)	Key Type:Symmetric	N/A	SP 800-133r2 and IG D.H: Per Section 6.3, approved method 1. Applicable to AES-XTS compliant to IG C.I because Key_1 and Key_2 are concatenated prior to usage.

Table 7: Vendor-Affirmed Algorithms

2.5.3 Non-Approved, Allowed Algorithms

Non-Approved, Allowed Algorithms:

The module implements the following algorithms that are allowed for use in Approved mode.

Name	Properties	Implementation	Reference
EC Diffie-Hellman with non-NIST recommended curves	Curves: brainpoolP224r1 (strength 112 bits) brainpoolP256r1 (strength 128 bits) brainpoolP320r1 (strength 160 bits) brainpoolP384r1 (strength 192 bits) brainpoolP512r1 (strength 256 bits) : SSP Agreement	CryptoComply 140-3 FIPS Provider	Allowed per IG D.F, scenario 3 (per IG C.A, category 1a and SP 800-186 Appendix H.1)
ECDSA with non-NIST recommended curves	Curves: brainpoolP224r1 (strength 112 bits) brainpoolP256r1 (strength 128 bits) brainpoolP320r1 (strength 160 bits) brainpoolP384r1 (strength 192 bits) brainpoolP512r1 (strength 256 bits) : Signature Generation, Signature Verification, Key Generation, Key Verification	CryptoComply 140-3 FIPS Provider	Allowed per IG C.A, category 1a (per SP 800-186 Appendix H.1)

Table 8: Non-Approved, Allowed Algorithms

2.5.4 Non-Approved, Allowed Algorithms with No Security Claimed

Non-Approved, Allowed Algorithms with No Security Claimed:

N/A for this module.

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

2.5.5 Non-Approved, Not Allowed Algorithms

Non-Approved, Not Allowed Algorithms:

N/A for this module.

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

2.6 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.

Name	Type	Description	Properties	Algorithms
AsymmetricKeyGen	AsymKeyPair-DomPar AsymKeyPair-KeyGen AsymKeyPair-KeyVer AsymKeyPair-PubKeyVal	Used to generate asymmetric keys using the DRBG for CKG per SP 800-133r2. Established SSPs are passed out to the calling application.	ECDSA Allowed Curves:brainpoolP224r1 (strength 112 bits) brainpoolP256r1 (strength 128 bits) brainpoolP320r1 (strength 160 bits) brainpoolP384r1 (strength 192 bits) brainpoolP512r1 (strength 256 bits)	Counter DRBG: (A4593, A5173) Hash DRBG: (A4593, A5173) HMAC DRBG: (A4593, A5173) CKG, asymmetric keys: () DSA KeyGen (FIPS186-4): (A4593, A5173) DSA PQGGen (FIPS186-4): (A4593, A5173) DSA PQGVer (FIPS186-4): (A4593, A5173) EDDSA KeyGen: (A4593, A5173) EDDSA KeyVer: (A4593, A5173) RSA KeyGen (FIPS186-4): (A4593, A5173) Safe Primes Key Generation: (A4593, A5173) Safe Primes Key Verification: (A4593, A5173) ECDSA KeyGen (FIPS186-4): (A4593, A5173)

Name	Type	Description	Properties	Algorithms
				Approved Curves: B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 ECDSA KeyVer (FIPS186-4): (A4593, A5173) Approved Curves: B-163, B-233, B-283, B-409, B-571, K-163, K-233, K-283, K-409, K-571, P-192, P-224, P-256, P-384, P-521
AuthSymmetric Encrypt/Decrypt	BC-Auth	Used to encrypt or decrypt data. SSPs are passed in by the calling application.		AES-CCM: (A4593, A5173) AES-GCM: (A4593, A5173)
CKG	CKG	Direct output of DRBGs may be used for symmetric key generation per SP 800-133r2.		Counter DRBG: (A4593, A5173) CKG: () Key Type: Symmetric and Asymmetric Hash DRBG: (A4593, A5173) HMAC DRBG: (A4593, A5173)
CKG (XTS)	CKG	AES-XTS key concatenation		CKG: () Key Type: Symmetric
DigitalSig	DigSig-SigGen DigSig-SigVer	Used to generate or verify digital signatures. SSPs are passed in by the calling application.	ECDSA Allowed Curves:brainpoolP224r1 (strength 112 bits) brainpoolP256r1 (strength 128 bits) brainpoolP320r1 (strength 160 bits)	EDDSA SigGen: (A4593, A5173) EDDSA SigVer: (A4593, A5173) RSA SigGen (FIPS186-4): (A4593, A5173)

Name	Type	Description	Properties	Algorithms
			brainpoolP384r1 (strength 192 bits) brainpoolP512r1 (strength 256 bits)	RSA SigVer (FIPS186-4): (A4593, A5173) ECDSA SigGen (FIPS186-4): (A4593, A5173) Approved Curves: B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 ECDSA SigVer (FIPS186-4): (A4593, A5173) Approved Curves: B-163, B-233, B-283, B-409, B-571, K-163, K-233, K-283, K-409, K-571, P-192, P-224, P-256, P-384, P-521
DigitalSig (Legacy)	AsymKeyPair-DomPar AsymKeyPair-KeyVer DigSig-SigVer	Used to verify digital signatures. SSPs are passed in by the calling application.		DSA PQGVer (FIPS186-4): (A4593, A5173) ECDSA KeyVer (FIPS186-4): (A4593, A5173) DSA SigVer (FIPS186-4): (A4593, A5173) ECDSA SigVer (FIPS186-4): (A4593, A5173) RSA SigVer (FIPS186-4): (A4593, A5173)
IntegrityTest	MAC	Integrity Test		HMAC-SHA2-256: (A4593, A5173)
KeyAgreement (ECC)	KAS-SSC	Used to perform key agreement primitives on	IG:IG D.F Scenario 2, path 1 Key Confirmation:No	KAS-ECC-SSC Sp800-56Ar3: (A4593, A5173)

Name	Type	Description	Properties	Algorithms
		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.	Key Derivation:No Caveat:Key establishment methodology provides between 112 and 256 bits of security strength	Approved Curves: B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 Allowed Curves: EC Diffie-Hellman with non-NIST recommended curves
KeyAgreement (FFC)	KAS-SSC	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.	IG:IG D.F Scenario 2, path 1 Key Confirmation:No Key Derivation:No Caveat:Key establishment methodology provides between 112 and 200 bits of security strength	KAS-FFC-SSC Sp800-56Ar3: (A4593, A5173)
KeyAgreement (RSA)	KAS-SSC	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.	IG:IG D.F Scenario 1, path 1 Key Confirmation:No Key Derivation:No Caveat:Key establishment methodology provides between 112 and 200 bits of security strength	KAS-IFC-SSC: (A4593, A5173)
KeyDerivation	KAS-135KDF KAS-56CKDF	Used to derive keys using KBKDF, PBKDF, HKDF, SP		KDA HKDF SP800-56Cr2: (A4593, A5173)

Name	Type	Description	Properties	Algorithms
	KBKDF PBKDF	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 OneStep SP800-56Cr2: (A4593, A5173) KDA TwoStep SP800-56Cr2: (A4593, A5173) KDF ANS 9.42: (A4593, A5173) KDF ANS 9.63: (A4593, A5173) KDF KMAC Sp800-108r1: (A4593, A5173) KDF SP800-108: (A4593, A5173) KDF SSH: (A4593, A5173) PBKDF: (A4593, A5173) TLS v1.2 KDF RFC7627: (A4593, A5173) TLS v1.3 KDF: (A4593, A5173)
KeyedHash	MAC XOF	Used to generate or verify data integrity. SSPs are passed in by the calling application.		HMAC-SHA-1: (A4593, A5173) HMAC-SHA2-224: (A4593, A5173) HMAC-SHA2-256: (A4593, A5173) HMAC-SHA2-384: (A4593, A5173) HMAC-SHA2-512: (A4593, A5173) HMAC-SHA2-512/224: (A4593, A5173) HMAC-SHA2-512/256: (A4593, A5173) HMAC-SHA3-224: (A4593, A5173) HMAC-SHA3-256:

Name	Type	Description	Properties	Algorithms
				(A4593, A5173) HMAC-SHA3-384: (A4593, A5173) HMAC-SHA3-512: (A4593, A5173) KMAC-128: (A4593, A5173) KMAC-256: (A4593, A5173)
KeyTransport	BC-AuthDecrypt BC-AuthEncrypt	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.	Standard:SP 800-56Br2 Key Confirmation:No Caveat:Key establishment methodology provides between 112 and 256 bits of security strength IG D.G:Approved key encapsulation	KTS-IFC: (A4593, A5173)
KeyWrapping	KTS-Unwrap KTS-Wrap	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.	Standard:SP 800-38F IG D.G:Approved key wrapping Key Confirmation:No Caveat:Key establishment methodology provides between 128 and 256 bits of encryption strength	AES-KW: (A4593, A5173) AES-KWP: (A4593, A5173)
MessageDigest	SHA	Used to generate a SHA-1, SHA-2, SHA-3, or SHAKE message digest.		SHA-1: (A4593, A5173) SHA2-224: (A4593, A5173) SHA2-256: (A4593, A5173) SHA2-384: (A4593,

Name	Type	Description	Properties	Algorithms
				A5173) SHA2-512: (A4593, A5173) SHA2-512/224: (A4593, A5173) SHA2-512/256: (A4593, A5173) SHA3-224: (A4593, A5173) SHA3-256: (A4593, A5173) SHA3-384: (A4593, A5173) SHA3-512: (A4593, A5173) SHAKE-128: (A4593, A5173) SHAKE-256: (A4593, A5173)
RNG	DRBG	Random Number Generation from the DRBG. Random data or established SSPs are passed out to the calling application.		Counter DRBG: (A4593, A5173) Hash DRBG: (A4593, A5173) HMAC DRBG: (A4593, A5173) CKG: () Key Type: Symmetric and Asymmetric
Symmetric Decrypt (Legacy)	BC-UnAuth	Symmetric decryption SSPs are passed in by the calling application.		TDES-CBC: (A4593, A5173) TDES-ECB: (A4593, A5173)
Symmetric Encrypt/Decrypt	BC-UnAuth	Symmetric encryption or decryption SSPs are passed in by the calling application.		AES-CBC: (A4593, A5173) AES-CBC-CS1: (A4593, A5173) AES-CBC-CS2: (A4593, A5173) AES-CBC-CS3: (A4593, A5173)

Name	Type	Description	Properties	Algorithms
				AES-CFB1: (A4593, A5173) AES-CFB8: (A4593, A5173) AES-CFB128: (A4593, A5173) AES-CTR: (A4593, A5173) AES-ECB: (A4593, A5173) AES-OFB: (A4593, A5173) AES-XTS Testing Revision 2.0: (A4593, A5173)
SymmetricDigest	MAC	Used to generate or verify data integrity with CMAC or GMAC. SSPs are passed in by the calling application.		AES-CMAC: (A4593, A5173) AES-GMAC: (A4593, A5173)

Table 9: Security Function Implementations

2.7 Algorithm Specific Information

2.7.1 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, the module provides the cryptographic functions required for implementing the TLS 1.2 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, the module provides the cryptographic functions required for implementing the TLS 1.3 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.

2.7.2 AES-XTS

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

2.7.3 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.

2.7.4 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.

2.7.5 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.

2.7.6 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 even RSA modulus sizes that are not testable by the CAVP in the following ranges:

- For RSA signature generation: 2048-16384
- For RSA signature verification: 2048-16384

- 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.

2.7.7 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 2.7.6, 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.

Refer also to Security Policy Section 2.7.10 - SP 800-140Br1 SSP Establishment for more information on the SSP establishment by the module.

2.7.8 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.

2.7.9 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.

2.7.10 SP 800-140Br1 SSP Establishment

As a cryptographic software library, 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. Accordingly, the following statements are required for conformance. For additional details, see also Security Policy Section 2.10 - Key Establishment.

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

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

2.8 RBG and Entropy

Entropy Certificates:

N/A for this module.

Entropy Sources:

N/A for this module.

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)."

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 Rev. 1 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.

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 2.5.2 - Vendor-Affirmed Algorithms.

2.9 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 2.6 - Security Function Implementations and 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 and asymmetric keys per SP 800-133r2, as indicated in Security Policy Section 2.5.2 - Vendor-Affirmed Algorithms. The module is a software library that provides a service (called Random Number Generation) for direct output of the approved DRBG (U). This output is approved for generating keys or SSPs.

Symmetric keys are generated per SP 800-133r2 Section 6.1 using the Random Number Generation service; additionally, Section 6.3 is applicable for AES-XTS keys. Asymmetric keys are generated per SP

800-133r2 Section 5 per FIPS 186-4 and per FIPS 186-5 (for EdDSA only) using the Asymmetric Key Generation service.

2.10 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 2.6 - Security Function Implementations and 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 (ECC/FFC) 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 Security Policy Section 2.5.3 - Non-Approved, Allowed Algorithms in alignment with IG D.F scenario 3 via the Key Agreement (ECC/FFC) 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 (RSA) 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.

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

Refer also to Security Policy Section 2.7.10 - SP 800-140Br1 SSP Establishment for more information on the SSP establishment by the module.

2.11 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.

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).

Physical Port	Logical Interface(s)	Data That Passes
N/A	Data Input	API input parameters for data
N/A	Data Output	API output parameters for data
N/A	Control Input	API function calls
N/A	Status Output	API status outputs (return codes, error messages)

Table 10: Ports and Interfaces

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 Authentication Methods

N/A for this module.

4.2 Roles

Name	Type	Operator Type	Authentication Methods
Crypto Officer	Role	CO	None

Table 11: Roles

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.

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 2.4 - Modes of Operation.

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Module Initialisation	Initialize the FIPS module when it is loaded	API return value from OSSL_provider_init: 1 for success, 0 for failure	External dispatch (function	Internal dispatch (function pointer) table	None	Crypto Officer

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	(calls pre-operational self-tests and CASTs).		pointer) table			
Self-Test	Performs pre-operational self-tests and CASTs on demand.	API return value code from SELF_TEST_post(): 1 for success, 0 for failure	None	Module State (queried via Show Status) changes to Running (FIPS_STATE_RUNNING)	None	Crypto Officer
Integrity Test	Performs the integrity test on demand.	API return value from verify_integrity(): 1 for verified, 0 for failure	Expected HMAC	Module State (queried via Show Status) changes to Running (FIPS_STATE_RUNNING)	IntegrityTest	Crypto Officer
Show Status	Provides status information by querying the "status" parameter.	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	None	Module Status: Running (FIPS_STATE_RUNNING), or Error (FIPS_STATE_ERROR)	None	Crypto Officer
Output ID/ Version Information (Show Version)	Displays FIPS module version by querying the "version," " "name," and "buildinf	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	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	None	Crypto Officer

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	o" parameters, with the results specified in the output column. The version aligns with the FIPS certificate and Security Policy Section 2.2 - Tested and Vendor Affirmed Module Version and Identification.					
Random Number Generation	Used to seed/reseed a DRBG instance (including determining the security strength) or obtain random	Implied if EVP_default_properties_is_fips_enabled() returns true. API return value: 1 for operation completed successfully, 0 for failure	Desired security strength in bits, entropy input	Random data	RNG CKG	Crypto Officer - DRBG Entropy Input: W,E,Z - CTR_DRBG Seed: G,E,Z - CTR_DR

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	data. Random data may be used for CKG per SP 800-133r2. Established SSPs are passed out to the calling application.					BG V: G,E,Z - CTR_DRBG Key: G,E,Z - Hash_DRBG Seed: G,E,Z - Hash_DRBG V: G,E,Z - Hash_DRBG C: G,E,Z - HMAC_DRBG Seed: G,E,Z - HMAC_DRBG V: G,E,Z - HMAC_DRBG Key: G,E,Z - Generic Secret: G,R,Z
Symmetric Encryption/	Used to encrypt or decrypt data.	Implied if EVP_default_properties_is_fips_enabled() returns true. API return value: 1 for	AES EDK, AES XTS key, IV, cipherte	Ciphertext data or plaintext data	Symmetric Encrypt/Decrypt CKG (XTS)	Crypto Officer - AES EDK: W,E,Z

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Decryption	SSPs are passed in by the calling application.	operation completed successfully, 0 for failure	xt data, plaintext data			- AES XTS key: W,E,Z
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.	Implied if EVP_default_properties_is_fips_enabled() returns true. API return value: 1 for operation completed successfully, 0 for failure	AES CMAC/CCM key, AES GMAC/ GCM key, ciphertext data, plaintext data	Ciphertext data or plaintext data	AuthSymmetric Encrypt/Decrypt	Crypto Officer - AES CMAC/CCM key: W,E,Z - AES GMAC/ GCM key: W,E,Z - AES GMAC/ GCM IV: G,E,Z
Symmetric Digest	Used to generate or verify data integrity with CMAC or GMAC. SSPs are passed in by the calling application.	Implied if EVP_default_properties_is_fips_enabled() returns true. API return value: 1 for operation completed successfully, 0 for failure	Digest or message, AES CMAC/CCM key, AES GMAC/ GCM key	Digest or verification result	SymmetricDigest	Crypto Officer - AES CMAC/CCM key: W,E,Z - AES GMAC/ GCM key: W,E,Z - AES GMAC/

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						GCM IV: G,E,Z
Asymmetric Key Generation	Used to generate asymmetric keys using the DRBG. Established SSPs are passed out to the calling application.	Implied if EVP_default_properties_is_fips_enabled() returns true. API return value: 1 for operation completed successfully, 0 for failure	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	Asymmetric KeyGen	Crypto Officer - DRBG Entropy Input: W,E,Z - CTR_DRBG Seed: G,E,Z - CTR_DRBG V: G,E,Z - CTR_DRBG Key: G,E,Z - Hash_DRBG Seed: G,E,Z - Hash_DRBG V: G,E,Z - Hash_DRBG C: G,E,Z - HMAC_DRBG Seed: G,E,Z - HMAC_DRBG V:

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						G,E,Z - HMAC_ DRBG Key: G,E,Z - ECDSA SGK: G,R,Z - ECDSA SVK: G,R,Z - RSA SGK: G,R,Z - RSA SVK: G,R,Z - EdDSA SGK: G,R,Z - EdDSA SVK: G,R,Z - DH Private: G,R,Z - DH Public: G,R,Z - EC DH Private: G,R,Z - EC DH Public: G,R,Z - RSA KAK Private: G,R,Z - RSA KAK Public:

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						G,R,Z - RSA KDK Private: G,R,Z - RSA KEK Public: G,R,Z
Digital Signatures	Used to generate or verify digital signatures. SSPs are passed in by the calling application.	Implied if EVP_default_properties_is_fips_enabled() returns true. API return value: 1 for operation completed successfully, 0 for failure	DSA SVK, ECDSA SGK, ECDSA SVK, RSA SGK, RSA SVK, EdDSA SGK, EdDSA SVK	Digital signature or verification result	DigitalSig	Crypto Officer - ECDSA SGK: W,E,Z - ECDSA SVK: W,E,Z - RSA SGK: W,E,Z - RSA SVK: W,E,Z - EdDSA SGK: W,E,Z - EdDSA SVK: W,E,Z
Keyed Hash	Used to generate or verify data integrity. SSPs are passed in by the calling application.	Implied if EVP_default_properties_is_fips_enabled() returns true. API return value: 1 for operation completed successfully, 0 for failure	HMAC key, KMAC key	Keyed hash or verification result	KeyedHash	Crypto Officer - HMAC Key: W,E,Z - KMAC Key: W,E,Z

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Message Digest	Used to generate a SHA-1, SHA-2, SHA-3, or SHAKE message digest.	Implied if EVP_default_properties_is_fips_enabled() returns true. API return value: 1 for operation completed successfully, 0 for failure	Message data	Digest	MessageDigest	Crypto Officer
Key Agreement (ECC/FFC)	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.	Implied if EVP_default_properties_is_fips_enabled() returns true. API return value: 1 for operation completed successfully, 0 for failure	DH Private, DH Public, ECDH Private, ECDH Public	DH Private, DH Public, ECDH Private, ECDH Public, KDF secret	KeyAgreement (ECC) KeyAgreement (FFC)	Crypto Officer - DH Private: R,W,E,Z - DH Public: R,W,E,Z - EC DH Private: R,W,E,Z - EC DH Public: R,W,E,Z - KDF Secret: G,R,Z

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Key Agreement (RSA)	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	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for operation completed successfully, 0 for failure	RSA KAK Private, RSA KAK Public	RSA KAK Private, RSA KAK Public, KDF secret	KeyAgreement (RSA)	Crypto Officer - RSA KAK Private: R,W,E,Z - RSA KAK Public: R,W,E,Z - KDF Secret: G,R,Z
Key Derivation	Used to derive keys using KBKDF, PBKDF, HKDF, SP 800-56Cr2 One-	Implied if EVP_default_properties_is_fips_enabled() returns true. API return value: 1 for operation completed successfully, 0 for failure	KDF secret, salt, iteration count, MAC, digest, cipher, key	Generic Secret	KeyDerivation	Crypto Officer - KDF Secret: W,E,Z - Generic Secret: G,R,Z

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	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.					

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
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.	Implied if EVP_default_properties_is_fips_enabled() returns true. API return value: 1 for operation completed successfully, 0 for failure	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)	KeyTransport	Crypto Officer - RSA KDK Private: W,E,Z - RSA KEK Public: W,E,Z - Generic Secret: R,W,Z
Key Wrapping	Used to encrypt or decrypt a key value on behalf of the calling application (does	Implied if EVP_default_properties_is_fips_enabled() returns true. API return value: 1 for operation completed successfully, 0 for failure	AES key wrapping key, key to be wrapped or unwrapped	Wrapped key or unwrapped key (Generic Secret)	KeyWrapping	Crypto Officer - AES key wrapping key: W,E,Z - Generic

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	not establish keys into the module). SSPs are passed in by the calling application. Established SSPs are passed out to the calling application.		(Generic Secret)			Secret: R,W,Z
Zeroise	All services automatically overwrite SSPs stored in allocated memory (zeroise). The module does not store any SSP persistently (beyond the lifetime of an API call),	Implied if EVP_default_properties_is_fips_enabled() returns true. API return value: 1 for operation completed successfully, 0 for failure	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.	None	Crypto Officer - DRBG Entropy Input: Z - CTR_DRBG Seed: Z - CTR_DRBG V: Z - CTR_DRBG Key: Z - Hash_DRBG Seed: Z - Hash_D

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	except for DRBG state values (stored for the lifetime of the DRBG instance). Stack cleanup is the responsibility of the calling application.					RBG V: Z - Hash_DRBG C: Z - HMAC_DRBG Seed: Z - HMAC_DRBG V: Z - HMAC_DRBG Key: Z
Utility	Miscellaneous helper functions.	Implied if EVP_default_properties_is_fips_enabled() returns true. API return value: 1 for operation completed successfully, 0 for failure	None	None	None	Crypto Officer
Symmetric Decryption (Legacy)	Used to decrypt data. SSPs are passed in by the calling application.	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for operation completed successfully, 0 for failure	TDES DK, ciphertext data	Plaintext data	Symmetric Decrypt (Legacy)	Crypto Officer - TDES DK: W,E,Z
Digital Signatures (Legacy)	Used to verify digital signatures. SSPs are passed	Implied if EVP_default_properties_is_fips_enabled() returns true API return value: 1 for operation completed successfully, 0 for failure	DSA SVK, ECDSA SVK (Legacy), RSA	Verification result	DigitalSig (Legacy)	Crypto Officer - DSA SVK: W,E,Z - ECDSA SVK

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	in by the calling application.		SVK (Legacy)			(Legacy) : W,E,Z - RSA SVK (Legacy) : W,E,Z

Table 12: Approved Services

4.4 Non-Approved Services

N/A for this module.

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 for this module.

5 Software/Firmware Security

5.1 Integrity Techniques

As specified in the executable code sets table in Security Policy Section 2.2 - Tested and Vendor Affirmed Module Version and Identification, 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.

The integrity authentication key for the integrity technique is an HMAC-SHA-256 key with a key length of 256 bits. It is integrated into the module during compilation and cannot be changed afterwards. The module is only provided to the end user in the form of a compiled binary (refer to Security Policy Section 11.1). Note, per ISO 19790:2012 Section 7.5, this key is not considered a SSP.

The installation process generates the HMAC digest for the module using this key and the module. For dynamic libraries, the HMAC digest is generated from the module file and is then stored in the module's configuration file. For static and iOS libraries, the HMAC digest is generated from the memory the module is loaded at and is then stored in the executable the module is linked into. To verify the module's integrity (for the pre-operational self-test or on demand), the module generates a new HMAC digest and compares it with the corresponding stored value. The test passes if the values match.

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

Type of Operational Environment: Modifiable

How Requirements are Satisfied:

Supported operational environments are indicated in Security Policy Section 2.2 - Tested and Vendor Affirmed Module Version and Identification. Refer also to that section for vendor affirmed operating environment porting guidance.

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 - Installation, Initialization, and 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 an approved 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 Parameters Management

9.1 Storage Areas

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

Table 13: Storage Areas

9.2 SSP Input-Output Methods

Name	From	To	Format Type	Distribution Type	Entry Type	SFI or Algorithm
API Input via TOEPP path	Other Applications (App per IG 9.5.A)	RAM / DRAM	Plaintext	Manual	Electronic	
Encrypted API Input using Key Transport via TOEPP path	Other Applications (App per IG 9.5.A)	RAM / DRAM	Encrypted	Manual	Electronic	KeyTransport
Encrypted API Input using Key Wrapping via TOEPP path	Other Applications (App per IG 9.5.A)	RAM / DRAM	Encrypted	Manual	Electronic	KeyWrapping
API Output via TOEPP path	RAM / DRAM	Other Applications (App per IG 9.5.A)	Plaintext	Manual	Electronic	
Encrypted API Output using Key Transport via TOEPP path	RAM / DRAM	Other Applications (App per IG 9.5.A)	Encrypted	Manual	Electronic	KeyTransport
Encrypted API Output using Key Wrapping via TOEPP path	RAM / DRAM	Other Applications (App per IG 9.5.A)	Encrypted	Manual	Electronic	KeyWrapping

Table 14: SSP Input-Output Methods

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 2.6 - Security Function Implementations) is used to encrypt the output CSP.

9.3 SSP Zeroization Methods

Zeroization Method	Description	Rationale	Operator Initiation
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 contains 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

Table 15: SSP Zeroization Methods

As indicated in Security Policy Section 4.3 - 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 1 of 2)

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
Generic Secret	SSPs generated from the direct DRBG output (CKG) or 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 - 112 - 256 bits	Key or other SSP - CSP	KeyDerivation CKG	KeyTransport KeyWrapping	KeyTransport KeyWrapping
AES EDK	AES encrypt/ decrypt key	128, 192, 256 bits - 128, 192, 256 bits	Symmetric Key - CSP			Symmetric Encrypt/Decrypt

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
AES CMAC/CCM key	AES CMAC/CCM key for encrypt/ decrypt or generate/ verify	128, 192, 256 bits - 128, 192, 256 bits	Symmetric Key - CSP			AuthSymmetric Encrypt/Decrypt SymmetricDigest
AES GMAC/GCM key	AES GMAC/GCM key for encrypt/ decrypt or generate/ verify	128, 192, 256 bits - 128, 192, 256 bits	Symmetric Key - CSP			AuthSymmetric Encrypt/Decrypt SymmetricDigest
AES GMAC/GCM IV	AES GMAC/GCM IV for encrypt/ decrypt or generate/ verify	96-1024 bits - 96-1024 bits	IV - CSP	RNG		AuthSymmetric Encrypt/Decrypt SymmetricDigest
AES XTS key	AES XTS encrypt/ decrypt key	128, 256 bits - 128, 256 bits	Symmetric Key - CSP	CKG (XTS)		Symmetric Encrypt/Decrypt
AES key wrapping key	AES KW, KWP key	128, 192, 256 bits - 128, 192, 256 bits	Symmetric Key - CSP			KeyWrapping
TDES DK	3-key Triple-DES decrypt key	192 bits - 112 bits	Symmetric Key - CSP			Symmetric Decrypt (Legacy)
DRBG Entropy Input	Entropy Input	128-1024 bits (length is dependent on the requested security	RBG - CSP			RNG AsymmetricKeyGen

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
		strength, per SP 800-90A Rev. 1 Table 2 and Table 3) - 128-256 bits				
CTR_DRBG Seed	CTR_DRBG seed, constructed from entropy input and other inputs per SP 800-90A Rev. 1 Sections 7.2, 8.6	256-896 bits - 128-256 bits	RBG - CSP	RNG AsymmetricKeyGen		RNG AsymmetricKeyGen
CTR_DRBG V	V, internal state	128 bits - 128 bits	RBG - CSP	RNG AsymmetricKeyGen		RNG AsymmetricKeyGen
CTR_DRBG Key	Key (AES), internal state	128, 192, 256 bits - 128, 192, 256 bits	RBG - CSP	RNG AsymmetricKeyGen		RNG AsymmetricKeyGen
Hash_DRBG Seed	Hash_DRBG seed, constructed from entropy input and other inputs per SP 800-90A Rev 1 Sections 7.2, 8.6	224-736 bits - 128-256 bits	RBG - CSP	RNG AsymmetricKeyGen		RNG AsymmetricKeyGen

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
Hash_DRBG V	V, internal state	440, 888 bits - 128, 256 bits	RBG - CSP	RNG AsymmetricKeyGen		RNG AsymmetricKeyGen
Hash_DRBG C	C, internal state	440, 888 bits - 128, 256 bits	RBG - CSP	RNG AsymmetricKeyGen		RNG AsymmetricKeyGen
HMAC_DRBG Seed	HMAC_DRBG seed, constructed from entropy input and other inputs per SP 800-90A Rev. 1 Sections 7.2, 8.6	224-1408 bits - 128, 256 bits	RBG - CSP	RNG AsymmetricKeyGen		RNG AsymmetricKeyGen
HMAC_DRBG V	V, internal state	160, 256, 512 bits - 160, 256, 512 bits	RBG - CSP	RNG AsymmetricKeyGen		RNG AsymmetricKeyGen
HMAC_DRBG Key	Key (HMAC), internal state	160, 256, 512 bits - 160, 256, 512 bits	RBG - CSP	RNG AsymmetricKeyGen		RNG AsymmetricKeyGen
ECDSA SGK	ECDSA signature generation key (P, B, K curves and brainpool)	224 - 512 bits - 112 - 256 bits	Signature - CSP	AsymmetricKeyGen		DigitalSig
RSA SGK	RSA signature generation key	2048 - 16384 bits - 112 - 256 bits	Signature - CSP	AsymmetricKeyGen		DigitalSig

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
EdDSA SGK	Ed25519 or Ed448 signature generation key	256, 456 bits - 128, 224 bits	Signature - CSP	AsymmetricKeyGen		DigitalSig
DH Private	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 - 112 bits 112-200 bits	Key Agreement - CSP	AsymmetricKeyGen		KeyAgreement (FFC)
EC DH Private	Elliptic Curve Diffie-Hellman	224 - 512 bits - 112 - 256 bits	Key Agreement - CSP	AsymmetricKeyGen		KeyAgreement (ECC)

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
	private key agreement key (P, B, K curves and brainpool)					
RSA KAK Private	RSA private key agreement key	2048 - 16384 bits - 112 - 256 bits	Key Agreement - CSP	AsymmetricKeyGen		KeyAgreement (RSA)
RSA KDK Private	RSA private key decryption key	2048 - 16384 bits - 112 - 256 bits	Key Transport - CSP	AsymmetricKeyGen		KeyTransport
HMAC Key	Keyed hash key for HMAC	160, 224, 256, 384, 512 bits - 128, 192 256 bits	Authentication - CSP			KeyedHash
KMAC Key	Keyed hash key for KMAC	128-1024 bits - 128, 256 bits	Authentication - CSP			KeyedHash
KDF Secret	Secret value used by KDFs	112 - 512 bits - 112 - 512 bits	Key Derivation Function - CSP	KeyAgreement (ECC) KeyAgreement (FFC) KeyAgreement (RSA)		KeyDerivation
DSA SVK	DSA signature verification key (legacy only)	DSA (L, N) = (512 L < 2048, 160 N < 224) (2048, 224) (2048, 256) (3072, 256) - 80 - 128 bits	Signature - PSP			DigitalSig (Legacy)

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
ECDSA SVK	ECDSA signature verification key (P, B, K curves and brainpool)	224 - 512 bits - 112 - 256 bits	Signature - PSP	AsymmetricKeyGen		DigitalSig
RSA SVK	RSA signature verification key	2048 - 16384 bits - 112 - 256 bits	Signature - PSP	AsymmetricKeyGen		DigitalSig
EdDSA SVK	Ed25519 or Ed448 signature verification key	256, 456 bits - 128, 224 bits	Signature - PSP	AsymmetricKeyGen		DigitalSig
DH Public	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-	Key Agreement - PSP	AsymmetricKeyGen		KeyAgreement (FFC)

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
		6144, MODP-8192 - 112-200 bits				
EC DH Public	Elliptic Curve Diffie-Hellman public key agreement key (P, B, K curves and brainpool)	224 - 512 bits - 112 - 256 bits	Key Agreement - PSP	AsymmetricKeyGen		KeyAgreement (ECC)
RSA KAK Public	RSA public key agreement key	2048 - 16384 bits - 112 - 256 bits	Key Agreement - PSP	AsymmetricKeyGen		KeyAgreement (RSA)
RSA KEK Public	RSA public key encryption key	2048 - 16384 bits - 112 - 256 bits	Key Transport - PSP	AsymmetricKeyGen		KeyTransport
ECDSA SVK (Legacy)	ECDSA signature verification key (P, B, K curves) for legacy curves	163 - 192 bits - 80 bits	Signature - PSP			DigitalSig (Legacy)
RSA SVK (Legacy)	RSA signature verification key for legacy key lengths or legacy SHA-1	1024 - 16384 bits - 80 bits	Signature - PSP			DigitalSig (Legacy)

Table 16: SSP Table 1

9.4.2 SSPs (Table 2 of 2)

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
Generic Secret	API Input via TOEPP path Encrypted API Input using Key Transport via TOEPP path Encrypted API Input using Key Wrapping via TOEPP path API Output via TOEPP path Encrypted API Output using Key Transport via TOEPP path Encrypted API Output using Key Wrapping via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	KDF Secret:Derived From RSA KDK Private:May be Wrapped or Unwrapped by RSA KEK Public:May be Wrapped or Unwrapped by AES key wrapping key:May be Wrapped or Unwrapped by
AES EDK	API Input via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	
AES CMAC/CCM key	API Input via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	
AES GMAC/GCM key	API Input via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is	Call a service that creates or uses the SSP	AES GMAC/GCM IV:Used With

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
			for the lifetime of the API call.		
AES GMAC/GCM IV		RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	AES GMAC/GCM key:Used With
AES XTS key	API Input via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	
AES key wrapping key	API Input via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	Generic Secret:Wraps or Unwraps
TDES DK	API Input via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	
DRBG Entropy Input	API Input via TOEPP path	RAM / DRAM:Plaintext	All DRBG SSPs are temporarily stored. Storage duration is for the lifetime of the DRBG instance.	Zeroise service	CTR_DRBG Seed:Used With CTR_DRBG V:Used With CTR_DRBG Key:Used With Hash_DRBG Seed:Used With Hash_DRBG V:Used With Hash_DRBG C:Used With HMAC_DRBG Seed:Used With HMAC_DRBG V:Used With

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
					HMAC_DRBG Key:Used With
CTR_DRBG Seed		RAM / DRAM:Plaintext	All DRBG SSPs are temporarily stored. Storage duration is for the lifetime of the DRBG instance.	Zeroise service	DRBG Entropy Input:Used With CTR_DRBG V:Used With CTR_DRBG Key:Used With
CTR_DRBG V		RAM / DRAM:Plaintext	All DRBG SSPs are temporarily stored. Storage duration is for the lifetime of the DRBG instance.	Zeroise service	DRBG Entropy Input:Used With CTR_DRBG Seed:Used With CTR_DRBG Key:Used With
CTR_DRBG Key		RAM / DRAM:Plaintext	All DRBG SSPs are temporarily stored. Storage duration is for the lifetime of the DRBG instance.	Zeroise service	DRBG Entropy Input:Used With CTR_DRBG Seed:Used With CTR_DRBG V:Used With
Hash_DRBG Seed		RAM / DRAM:Plaintext	All DRBG SSPs are temporarily stored. Storage duration is for the lifetime of the DRBG instance.	Zeroise service	DRBG Entropy Input:Used With Hash_DRBG V:Used With Hash_DRBG C:Used With
Hash_DRBG V		RAM / DRAM:Plaintext	All DRBG SSPs are temporarily stored. Storage duration is for the lifetime of the DRBG instance.	Zeroise service	DRBG Entropy Input:Used With Hash_DRBG Seed:Used With Hash_DRBG C:Used With
Hash_DRBG C		RAM / DRAM:Plaintext	All DRBG SSPs are temporarily stored. Storage duration is for the lifetime of the DRBG instance.	Zeroise service	DRBG Entropy Input:Used With Hash_DRBG Seed:Used With Hash_DRBG V:Used With

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
HMAC_DRBG Seed		RAM / DRAM:Plaintext	All DRBG SSPs are temporarily stored. Storage duration is for the lifetime of the DRBG instance.	Zeroise service	DRBG Entropy Input:Used With HMAC_DRBG V:Used With HMAC_DRBG Key:Used With
HMAC_DRBG V		RAM / DRAM:Plaintext	All DRBG SSPs are temporarily stored. Storage duration is for the lifetime of the DRBG instance.	Zeroise service	DRBG Entropy Input:Used With HMAC_DRBG Seed:Used With HMAC_DRBG Key:Used With
HMAC_DRBG Key		RAM / DRAM:Plaintext	All DRBG SSPs are temporarily stored. Storage duration is for the lifetime of the DRBG instance.	Zeroise service	DRBG Entropy Input:Used With HMAC_DRBG Seed:Used With HMAC_DRBG V:Used With
ECDSA SGK	API Input via TOEPP path API Output via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	ECDSA SVK:Paired With
RSA SGK	API Input via TOEPP path API Output via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	RSA SVK:Paired With
EdDSA SGK	API Input via TOEPP path API Output via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	EdDSA SVK:Paired With
DH Private	API Input via TOEPP path API Output via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	DH Public:Paired With KDF Secret:Used to establish

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
EC DH Private	API Input via TOEPP path API Output via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	EC DH Public:Paired With KDF Secret:Used to establish
RSA KAK Private	API Input via TOEPP path API Output via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	RSA KAK Public:Paired With KDF Secret:Used to establish
RSA KDK Private	API Input via TOEPP path API Output via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	RSA KEK Public:Paired With Generic Secret:Unwraps
HMAC Key	API Input via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	
KMAC Key	API Input via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	
KDF Secret	API Input via TOEPP path API Output via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	DH Private:Derived From DH Public:Derived From EC DH Private:Derived From EC DH Public:Derived From RSA KAK Private:Derived From RSA KAK

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
					Public:Derived From Generic Secret:Used to derive
DSA SVK	API Input via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	
ECDSA SVK	API Input via TOEPP path API Output via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	ECDSA SGK:Paired With
RSA SVK	API Input via TOEPP path API Output via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	RSA SGK:Paired With
EdDSA SVK	API Input via TOEPP path API Output via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	EdDSA SGK:Paired With
DH Public	API Input via TOEPP path API Output via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	DH Private:Paired With KDF Secret:Used to establish
EC DH Public	API Input via TOEPP path API Output via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	EC DH Private:Paired With KDF Secret:Used to establish
RSA KAK Public	API Input via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is	Call a service that creates	RSA KAK Private:Paired With

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
	API Output via TOEPP path		for the lifetime of the API call.	or uses the SSP	KDF Secret:Used to establish
RSA KEK Public	API Input via TOEPP path API Output via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	RSA KDK Private:Paired With Generic Secret:Wraps
ECDSA SVK (Legacy)	API Input via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	
RSA SVK (Legacy)	API Input via TOEPP path	RAM / DRAM:Plaintext	All SSPs are temporarily stored. Storage duration is for the lifetime of the API call.	Call a service that creates or uses the SSP	

Table 17: SSP Table 2

9.5 Transitions

All algorithms implemented by the module are approved for FIPS 140-3 and their approval status 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 2.7 - 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, SHA-1 functionality is still supported by the module.

10 Self-Tests

10.1 Pre-Operational Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details
HMAC-SHA2-256 (A4593)	HMAC-SHA-256 (Cert. A4593)	Compare to pre-computed HMAC	SW/FW Integrity	The Module State (queried via Show Status) changes to Running (FIPS_STATE_RUNNING)	Verify
HMAC-SHA2-256 (A5173)	HMAC-SHA-256 (Cert. A5173)	Compare to pre-computed HMAC	SW/FW Integrity	The Module State (queried via Show Status) changes to Running (FIPS_STATE_RUNNING)	Verify

Table 18: Pre-Operational Self-Tests

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.

Note, please refer also to the HMAC-SHA-256 CAST, which is performed before the pre-operational software integrity test.

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 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.

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-GCM Authenticated Encrypt KAT (Forward Cipher) (A4593)	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 Authenticated Encrypt KAT (Forward Cipher) (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 Decrypt KAT (Forward Cipher) (A4593)	256-bit AES	KAT	CAST	The Module State (queried via Show Status) changes to Running (FIPS_STATE_RUNNING)	Decrypt (forward cipher)	Initialisation
AES-GCM Decrypt KAT (Forward Cipher) (A5173)	256-bit AES	KAT	CAST	The Module State (queried via Show Status) changes to Running (FIPS_STATE_RUNNING)	Decrypt (forward cipher)	Initialisation
AES-ECB Decrypt KAT (Inverse Cipher) (A4593)	256-bit AES	KAT	CAST	The Module State changes to Running	Decrypt (inverse cipher)	Initialisation

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-ECB Decrypt KAT (Inverse Cipher) (A5173)	256-bit AES	KAT	CAST	The Module State changes to Running	Decrypt (inverse cipher)	Initialisation
Counter DRBG (A4593)	128-bit AES with df	KAT	CAST	The Module State changes to Running	Instantiate, Reseed, Generate (per IG 10.3.A, 6)	Initialisation
Counter DRBG (A5173)	128-bit AES with df	KAT	CAST	The Module State changes to Running	Instantiate, Reseed, Generate (per IG 10.3.A, 6)	Initialisation
Hash DRBG (A4593)	SHA-256	KAT	CAST	The Module State changes to Running	Instantiate, Reseed, Generate (per IG 10.3.A, 6)	Initialisation
Hash DRBG (A5173)	SHA-256	KAT	CAST	The Module State changes to Running	Instantiate, Reseed, Generate (per IG 10.3.A, 6)	Initialisation
HMAC DRBG (A4593)	HMAC-SHA-1	KAT	CAST	The Module State changes to Running	Instantiate, Reseed, Generate (per IG 10.3.A, 6)	Initialisation
HMAC DRBG (A5173)	HMAC-SHA-1	KAT	CAST	The Module State changes to Running	Instantiate, Reseed, Generate (per IG 10.3.A, 6)	Initialisation

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
KDF ANS 9.42 (A4593)	SHA-1	KAT	CAST	The Module State changes to Running	Derive	Initialisation
KDF ANS 9.42 (A5173)	SHA-1	KAT	CAST	The Module State changes to Running	Derive	Initialisation
KDF ANS 9.63 (A4593)	SHA-256	KAT	CAST	The Module State changes to Running	Derive	Initialisation
KDF ANS 9.63 (A5173)	SHA-256	KAT	CAST	The Module State changes to Running	Derive	Initialisation
KDF SSH (A4593)	SHA-1	KAT	CAST	The Module State changes to Running	Derive	Initialisation
KDF SSH (A5173)	SHA-1	KAT	CAST	The Module State changes to Running	Derive	Initialisation
TLS v1.2 KDF RFC7627 (A4593)	HMAC-SHA-256	KAT	CAST	The Module State changes to Running	Derive	Initialisation
TLS v1.2 KDF RFC7627 (A5173)	HMAC-SHA-256	KAT	CAST	The Module State changes to Running	Derive	Initialisation
TLS v1.3 KDF (A4593)	SHA-256	KAT	CAST	The Module State changes to Running	Derive	Initialisation
TLS v1.3 KDF (A5173)	SHA-256	KAT	CAST	The Module State changes to Running	Derive	Initialisation
DSA Verify (A4593)	2048, SHA-256	KAT	CAST	The Module State changes to Running	Verify	Initialisation
DSA Verify (A5173)	2048, SHA-256	KAT	CAST	The Module State changes to Running	Verify	Initialisation
ECDSA Sign KAT for Prime Curves (A4593)	P-224, SHA-512	KAT	CAST	The Module State changes to Running	Sign	Initialisation

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
ECDSA Sign KAT for Prime Curves (A5173)	P-224, SHA-512	KAT	CAST	The Module State changes to Running	Sign	Initialisation
ECDSA Sign KAT for Binary Curves (A4593)	K-233, SHA-512	KAT	CAST	The Module State changes to Running	Sign	Initialisation
ECDSA Sign KAT for Binary Curves (A5173)	K-233, SHA-512	KAT	CAST	The Module State changes to Running	Sign	Initialisation
ECDSA Sign KAT for Brainpool Curves (A4593)	brainpoolP224r1, SHA-512	KAT	CAST	The Module State changes to Running	Sign	Initialisation
ECDSA Sign KAT for Brainpool Curves (A5173)	brainpoolP224r1, SHA-512	KAT	CAST	The Module State changes to Running	Sign	Initialisation
ECDSA Verify KAT for Prime Curves (A4593)	P-224, SHA-512	KAT	CAST	The Module State changes to Running	Verify	Initialisation
ECDSA Verify KAT for Prime Curves (A5173)	P-224, SHA-512	KAT	CAST	The Module State changes to Running	Verify	Initialisation
ECDSA Verify KAT for Binary Curves (A4593)	K-233, SHA-512	KAT	CAST	The Module State changes to Running	Verify	Initialisation

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
ECDSA Verify KAT for Binary Curves (A5173)	K-233, SHA-512	KAT	CAST	The Module State changes to Running	Verify	Initialisation
ECDSA Verify KAT for Brainpool Curves (A4593)	brainpoolP224r1, SHA-512	KAT	CAST	The Module State changes to Running	Verify	Initialisation
ECDSA Verify KAT for Brainpool Curves (A5173)	brainpoolP224r1, SHA-512	KAT	CAST	The Module State changes to Running	Verify	Initialisation
EDDSA Sign KAT for Ed25519 (A4593)	Ed25519	KAT	CAST	The Module State changes to Running	Sign	Initialisation
EDDSA Sign KAT for Ed25519 (A5173)	Ed25519	KAT	CAST	The Module State changes to Running	Sign	Initialisation
EDDSA Sign KAT for Ed448 (A4593)	Ed448	KAT	CAST	The Module State changes to Running	Sign	Initialisation
EDDSA Sign KAT for Ed448 (A5173)	Ed448	KAT	CAST	The Module State changes to Running	Sign	Initialisation
EDDSA Verify KAT for Ed25519 (A4593)	Ed25519	KAT	CAST	The Module State changes to Running	Verify	Initialisation

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
EDDSA Verify KAT for Ed25519 (A5173)	Ed25519	KAT	CAST	The Module State changes to Running	Verify	Initialisation
EDDSA Verify KAT for Ed448 (A4593)	Ed448	KAT	CAST	The Module State changes to Running	Verify	Initialisation
EDDSA Verify KAT for Ed448 (A5173)	Ed448	KAT	CAST	The Module State changes to Running	Verify	Initialisation
HMAC-SHA2-256 (A4593)	HMAC-SHA-256	KAT	CAST	The Module State changes to Running	Verify	Initialisation, performed before pre-operational integrity test
HMAC-SHA2-256 (A5173)	HMAC-SHA-256	KAT	CAST	The Module State changes to Running	Verify	Initialisation, performed before pre-operational integrity test
KAS-ECC-SSC Sp800-56Ar3 (A4593)	P-256	KAT	CAST	The Module State changes to Running	Verify computation of shared secret Z in Ephemeral Unified scheme, per Scenario 2 of IG D.F and Section 6 of SP 800-56Ar3	Initialisation

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
KAS-ECC-SSC Sp800-56Ar3 (A5173)	P-256	KAT	CAST	The Module State changes to Running	Verify computation of shared secret Z in Ephemeral Unified scheme, per Scenario 2 of IG D.F and Section 6 of SP 800-56Ar3	Initialisation
KAS-FFC-SSC Sp800-56Ar3 (A4593)	FB (2048, 224)	KAT	CAST	The Module State changes to Running	Verify computation of shared secret Z in dhEphem scheme, per Scenario 2 of IG D.F and Section 6 of SP 800-56Ar3	Initialisation
KAS-FFC-SSC Sp800-56Ar3 (A5173)	FB (2048, 224)	KAT	CAST	The Module State changes to Running	Verify computation of shared secret Z in dhEphem scheme, per Scenario 2 of IG D.F and Section 6 of SP 800-56Ar3	Initialisation
KAS-IFC-SSC (A4593)	2048-bit key	KAT	CAST	The Module State changes to Running	RSA Primitive Computation, per Scenario 1 of IG D.F and Section 8.2.2 in SP 800-56Br2	Initialisation

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
KAS-IFC-SSC (A5173)	2048-bit key	KAT	CAST	The Module State changes to Running	RSA Primitive Computation, per Scenario 1 of IG D.F and Section 8.2.2 in SP 800-56Br2	Initialisation
KDA OneStep SP800-56Cr2 (A4593)	SHA-224	KAT	CAST	The Module State changes to Running	Derive	Initialisation
KDA OneStep SP800-56Cr2 (A5173)	SHA-224	KAT	CAST	The Module State changes to Running	Derive	Initialisation
KDA TwoStep SP800-56Cr2 (A4593)	SHA-256	KAT	CAST	The Module State changes to Running	Derive	Initialisation
KDA TwoStep SP800-56Cr2 (A5173)	SHA-256	KAT	CAST	The Module State changes to Running	Derive	Initialisation
KDF SP800-108 (A4593)	Counter Mode with HMAC-SHA-256	KAT	CAST	The Module State changes to Running	Derive	Initialisation
KDF SP800-108 (A5173)	Counter Mode with HMAC-SHA-256	KAT	CAST	The Module State changes to Running	Derive	Initialisation
KTS-IFC Encrypt KAT for KTS-OAEP-Basic (A4593)	2048-bit key	KAT	CAST	The Module State changes to Running	Encrypt for KTS-OAEP-Basic, per IG D.G and SP 800-56Br2	Initialisation
KTS-IFC Encrypt KAT for KTS-OAEP-Basic (A5173)	2048-bit key	KAT	CAST	The Module State changes to Running	Encrypt for KTS-OAEP-Basic, per IG D.G and SP 800-56Br2	Initialisation

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
KTS-IFC Decrypt KAT for KTS-OAEP-Basic (A4593)	2048-bit key	KAT	CAST	The Module State changes to Running	Decrypt for KTS-OAEP-Basic, per IG D.G and SP 800-56Br2	Initialisation
KTS-IFC Decrypt KAT for KTS-OAEP-Basic (A5173)	2048-bit key	KAT	CAST	The Module State changes to Running	Decrypt for KTS-OAEP-Basic, per IG D.G and SP 800-56Br2	Initialisation
KTS-IFC Decrypt KAT for CRT (A4593)	2048-bit key	KAT	CAST	The Module State changes to Running	Decrypt for CRT, per IG D.G and SP 800-56Br2	Initialisation
KTS-IFC Decrypt KAT for CRT (A5173)	2048-bit key	KAT	CAST	The Module State changes to Running	Decrypt for CRT, per IG D.G and SP 800-56Br2	Initialisation
PBKDF (A4593)	HMAC-SHA-1	KAT	CAST	The Module State changes to Running	Derivation of the Master Key (MK), per Section 5.3 of SP 800-132	Initialisation
PBKDF (A5173)	HMAC-SHA-1	KAT	CAST	The Module State changes to Running	Derivation of the Master Key (MK), per Section 5.3 of SP 800-132	Initialisation
RSA Sign KAT (A4593)	2048-bit key, SHA-256, PKCS#1	KAT	CAST	The Module State changes to Running	Sign	Initialisation
RSA Sign KAT (A5173)	2048-bit key, SHA-256, PKCS#1	KAT	CAST	The Module State changes to Running	Sign	Initialisation

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
RSA Verify KAT (A4593)	2048-bit key, SHA-256, PKCS#1	KAT	CAST	The Module State changes to Running	Verify	Initialisation
RSA Verify KAT (A5173)	2048-bit key, SHA-256, PKCS#1	KAT	CAST	The Module State changes to Running	Verify	Initialisation
SHA3 KAT for Keccak-p Permutation (A4593)	SHA3-256	KAT	CAST	The Module State changes to Running	Hash	Initialisation
SHA3 KAT for Keccak-p Permutation (A5173)	SHA3-256	KAT	CAST	The Module State changes to Running	Hash	Initialisation
SHA-1 (A4593)	SHA-1	KAT	CAST	The Module State changes to Running	Hash	Initialisation
SHA-1 (A5173)	SHA-1	KAT	CAST	The Module State changes to Running	Hash	Initialisation
SHA2-512 (A4593)	SHA-512	KAT	CAST	The Module State changes to Running	Hash	Initialisation
SHA2-512 (A5173)	SHA-512	KAT	CAST	The Module State changes to Running	Hash	Initialisation
TDES-CBC (A4593)	CBC mode, 3-key	KAT	CAST	The Module State changes to Running	Decrypt	Initialisation
TDES-CBC (A5173)	CBC mode, 3-key	KAT	CAST	The Module State changes to Running	Decrypt	Initialisation
DSA (FFC) PCT for Key Agreement (A4593)	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, per VE10.35.03	Key Pair Generation, Key Pair Import

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
DSA (FFC) PCT for Key Agreement (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, per VE10.35.03	Key Pair Generation, Key Pair Import
ECC PCT for Key Pair Generation (A4593)	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, 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.	Key Pair Generation
ECC PCT for Key Pair Generation (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, per VE10.35.02. At the time of key pair generation, the keys' intended usage is not	Key Pair Generation

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
					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.	
ECC PCT for Key Pair Import (A4593)	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, 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	Key Pair Import
ECC PCT for Key Pair Import (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, per VE10.35.03.	Key Pair Import

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
					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	
EdDSA PCT (A4593)	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, per VE10.35.02. EdDSA keys can only be used for digital signatures.	Key Pair Generation, Key Pair Import
EdDSA PCT (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, per VE10.35.02. EdDSA keys can only be used for digital signatures.	Key Pair Generation, Key Pair Import

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
RSA PCT (A4593)	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, 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.	Key Pair Generation, Key Pair Import
RSA PCT (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, 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,	Key Pair Generation, Key Pair Import

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
					digital signatures, or key agreement); per IG 10.3.A comment 1, any of the AS10.35 PCTs is acceptable.	
AES-XTS Key Test (A4593)	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, per IG C.I	Symmetric Encryption/ Decryption
AES-XTS Key Test (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, per IG C.I	Symmetric Encryption/ Decryption

Table 19: Conditional Self-Tests

10.3 Periodic Self-Test Information

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

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
HMAC-SHA2-256 (A4593)	Compare to pre-computed HMAC	SW/FW Integrity	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

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				integrity test and all CASTs).
HMAC-SHA2-256 (A5173)	Compare to pre-computed HMAC	SW/FW Integrity	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).

Table 20: Pre-Operational Periodic Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
AES-GCM Authenticated Encrypt KAT (Forward Cipher) (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
AES-GCM Authenticated Encrypt KAT (Forward Cipher) (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
AES-GCM Decrypt KAT (Forward Cipher) (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	Conditional Periodic tests are called by power cycling the module or calling the Self-Test service (which calls the

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				module's integrity test and all CASTs).
AES-GCM Decrypt KAT (Forward Cipher) (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
AES-ECB Decrypt KAT (Inverse Cipher) (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
AES-ECB Decrypt KAT (Inverse Cipher) (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
Counter DRBG (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
Counter DRBG (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	Conditional Periodic tests are called by power cycling the module or calling the Self-Test service (which calls the

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				module's integrity test and all CASTs).
Hash DRBG (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
Hash DRBG (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
HMAC DRBG (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
HMAC DRBG (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KDF ANS 9.42 (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	Conditional Periodic tests are called by power cycling the module or calling the Self-Test service (which calls the

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				module's integrity test and all CASTs).
KDF ANS 9.42 (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KDF ANS 9.63 (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KDF ANS 9.63 (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KDF SSH (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KDF SSH (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	Conditional Periodic tests are called by power cycling the module or calling the Self-Test service (which calls the

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				module's integrity test and all CASTs).
TLS v1.2 KDF RFC7627 (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
TLS v1.2 KDF RFC7627 (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
TLS v1.3 KDF (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
TLS v1.3 KDF (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
DSA Verify (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	Conditional Periodic tests are called by power cycling the module or calling the Self-Test service (which calls the

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				module's integrity test and all CASTs).
DSA Verify (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
ECDSA Sign KAT for Prime Curves (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
ECDSA Sign KAT for Prime Curves (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
ECDSA Sign KAT for Binary Curves (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
ECDSA Sign KAT for Binary Curves (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	Conditional Periodic tests are called by power cycling the module or calling the Self-Test service (which calls the

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				module's integrity test and all CASTs).
ECDSA Sign KAT for Brainpool Curves (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
ECDSA Sign KAT for Brainpool Curves (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
ECDSA Verify KAT for Prime Curves (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
ECDSA Verify KAT for Prime Curves (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
ECDSA Verify KAT for Binary Curves (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	Conditional Periodic tests are called by power cycling the module or calling the Self-Test service (which calls the

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				module's integrity test and all CASTs).
ECDSA Verify KAT for Binary Curves (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
ECDSA Verify KAT for Brainpool Curves (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
ECDSA Verify KAT for Brainpool Curves (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
EDDSA Sign KAT for Ed25519 (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
EDDSA Sign KAT for Ed25519 (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	Conditional Periodic tests are called by power cycling the module or calling the Self-Test service (which calls the

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				module's integrity test and all CASTs).
EDDSA Sign KAT for Ed448 (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
EDDSA Sign KAT for Ed448 (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
EDDSA Verify KAT for Ed25519 (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
EDDSA Verify KAT for Ed25519 (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
EDDSA Verify KAT for Ed448 (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	Conditional Periodic tests are called by power cycling the module or calling the Self-Test service (which calls the

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				module's integrity test and all CASTs).
EDDSA Verify KAT for Ed448 (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
HMAC-SHA2-256 (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
HMAC-SHA2-256 (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KAS-ECC-SSC Sp800-56Ar3 (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KAS-ECC-SSC Sp800-56Ar3 (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	Conditional Periodic tests are called by power cycling the module or calling the Self-Test service (which calls the

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				module's integrity test and all CASTs).
KAS-FFC-SSC Sp800-56Ar3 (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KAS-FFC-SSC Sp800-56Ar3 (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KAS-IFC-SSC (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KAS-IFC-SSC (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KDA OneStep SP800-56Cr2 (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	Conditional Periodic tests are called by power cycling the module or calling the Self-Test service (which calls the

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				module's integrity test and all CASTs).
KDA OneStep SP800-56Cr2 (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KDA TwoStep SP800-56Cr2 (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KDA TwoStep SP800-56Cr2 (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KDF SP800-108 (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KDF SP800-108 (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	Conditional Periodic tests are called by power cycling the module or calling the Self-Test service (which calls the

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				module's integrity test and all CASTs).
KTS-IFC Encrypt KAT for KTS-OAEP-Basic (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KTS-IFC Encrypt KAT for KTS-OAEP-Basic (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KTS-IFC Decrypt KAT for KTS-OAEP-Basic (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KTS-IFC Decrypt KAT for KTS-OAEP-Basic (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
KTS-IFC Decrypt KAT for CRT (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	Conditional Periodic tests are called by power cycling the module or calling the Self-Test service (which calls the

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				module's integrity test and all CASTs).
KTS-IFC Decrypt KAT for CRT (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
PBKDF (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
PBKDF (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
RSA Sign KAT (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
RSA Sign KAT (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	Conditional Periodic tests are called by power cycling the module or calling the Self-Test service (which calls the

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				module's integrity test and all CASTs).
RSA Verify KAT (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
RSA Verify KAT (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
SHA3 KAT for Keccak-p Permutation (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
SHA3 KAT for Keccak-p Permutation (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
SHA-1 (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	Conditional Periodic tests are called by power cycling the module or calling the Self-Test service (which calls the

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				module's integrity test and all CASTs).
SHA-1 (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
SHA2-512 (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
SHA2-512 (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
TDES-CBC (A4593)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	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).
TDES-CBC (A5173)	KAT	CAST	On demand. It is recommended to run the periodic tests at least annually.	Conditional Periodic tests are called by power cycling the module or calling the Self-Test service (which calls the

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				module's integrity test and all CASTs).
DSA (FFC) PCT for Key Agreement (A4593)	PCT	PCT		
DSA (FFC) PCT for Key Agreement (A5173)	PCT	PCT		
ECC PCT for Key Pair Generation (A4593)	PCT	PCT		
ECC PCT for Key Pair Generation (A5173)	PCT	PCT		
ECC PCT for Key Pair Import (A4593)	PCT	PCT		
ECC PCT for Key Pair Import (A5173)	PCT	PCT		
EdDSA PCT (A4593)	PCT	PCT		
EdDSA PCT (A5173)	PCT	PCT		
RSA PCT (A4593)	PCT	PCT		
RSA PCT (A5173)	PCT	PCT		
AES-XTS Key Test (A4593)	Other	Critical Function		
AES-XTS Key Test (A5173)	Other	Critical Function		

Table 21: Conditional Periodic Information

10.4 Error States

Name	Description	Conditions	Recovery Method	Indicator
FIPS_STATE_ERROR	The module has entered an error state. All cryptographic	Pre-operational self-test failure CAST	Restart the module	Module State (queried via Show Status)

Name	Description	Conditions	Recovery Method	Indicator
	APIs will return an error when called.	self-test failure		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.	Conditional PCT test failure Conditional AES-XTS critical function test failure	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

Table 22: Error States

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 of Self-Tests

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 Installation, Initialization, and Startup Procedures

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;
    }
```

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

After the module starts up, the operator should confirm that the module outputs the Approved mode status indicator (refer to Security Policy Section 2.4 - Modes of Operation) and verify the module's version using the "Output ID/ Version Information (Show Version)" 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 2.7.1 - AES-GCM (IG C.H conformance).

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

11.4 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.

11.5 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.

Prepared By:

SafeLogic, Inc.

Website: www.safelogic.com
Email: sales@safelogic.com
Phone: 844-436-2797

8300 Boone Blvd., Suite 500
Vienna, VA 22182