| Vendor name |
| --- |
| DigiCert, Inc. |

| Module Name |
| --- |
| DigiCert TrustCore Cryptographic Suite B Module |

# FIPS 140-3 Non-Proprietary Security Policy

Software Version: 7.0.0f_u1

Document Version: 7.0.0f_u1_1.4
Date: 10/20/2025

*DigiCert, Inc.*

2801 North Thanksgiving Way
Suite 500
Lehi, UT 84043
+1 800-896-7973

# Table of Contents

## List of Tables

## List of Figures

# 1 General

## 1.1 Overview

This document is the non-proprietary FIPS 140-3 Security Policy for version 7.0.0f_u1 of the DigiCert TrustCore Cryptographic Suite B Module. It contains the security rules under which the module must operate and describes how this module meets the requirements as specified in FIPS PUB 140-3 (Federal Information Processing Standards Publication 140-3) for an overall Security Level 1 module.

## 1.2 Security Levels

The FIPS 140-3 security levels for the Module are as follows from the table below:

| 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 | N/A |
| | Overall Level | 1 |

Table 1: Security Levels

# 2 Cryptographic Module Specification

This DigiCert, Inc. (DigiCert) DigiCert TrustCore Cryptographic Suite B Module, is hereafter denoted as the Module. The Module is the cryptographic engine of DigiCert's TrustCore development platform. The trust and crypto abstraction layer reduce costs with extensibility across secure elements, a modular architecture for H/W acceleration, and plug-ins to comply with export/import controls.

## 2.1 Description

**Purpose and Use:**

The primary purpose of the Module is to provide approved cryptographic routines to consuming applications via an Application Programming Interface (API). The Module is intended for use by US Federal agencies or other markets that require FIPS 140-3 validated Security Level 1 software modules. The Module is intended to be used in dedicated purpose IOT (Internet of Things) devices and general-purpose computer systems.

**Module Type**: Software

**Module Embodiment**: Multi-Chip Standalone

**Cryptographic Boundary:**

The physical form of the Module is depicted in Figure 1. The Module is software, with a multi-chip standalone embodiment. The cryptographic boundary is comprised of the shared library files (libmss.so) and the integrity check signature file (libmss.so.sig), the POST status file (mssp.bin), and the CPU when PAA is enabled.

**Tested Operational Environment's Physical Perimeter (TOEPP):**

The TOEPP is bound by the General Purpose Computer and includes the DigiCert TrustCore Cryptographic Suite B Module, the CPU with PAA when PAA is enabled, and API calls from calling applications running within the same process as the Module.

Figure 1 shows the module, interfaces with the Tested Operational Environment (TOEPP), and the delimitation of its cryptographic boundary, shown shaded in blue. The Cryptographic Boundary components are described in Table 1.

| Component* | Description |
|---|---|
| libmss.so | Shared Library for cryptographic algorithms |
| libmss.so.sig | Integrity Check HMAC value for the libmss shared library |
| mssp.bin | Status file that contains the persistent results from the first run of POST. The file is protected from modification with a HMAC-SHA2-256. |

Table 2: Cryptographic Module Components

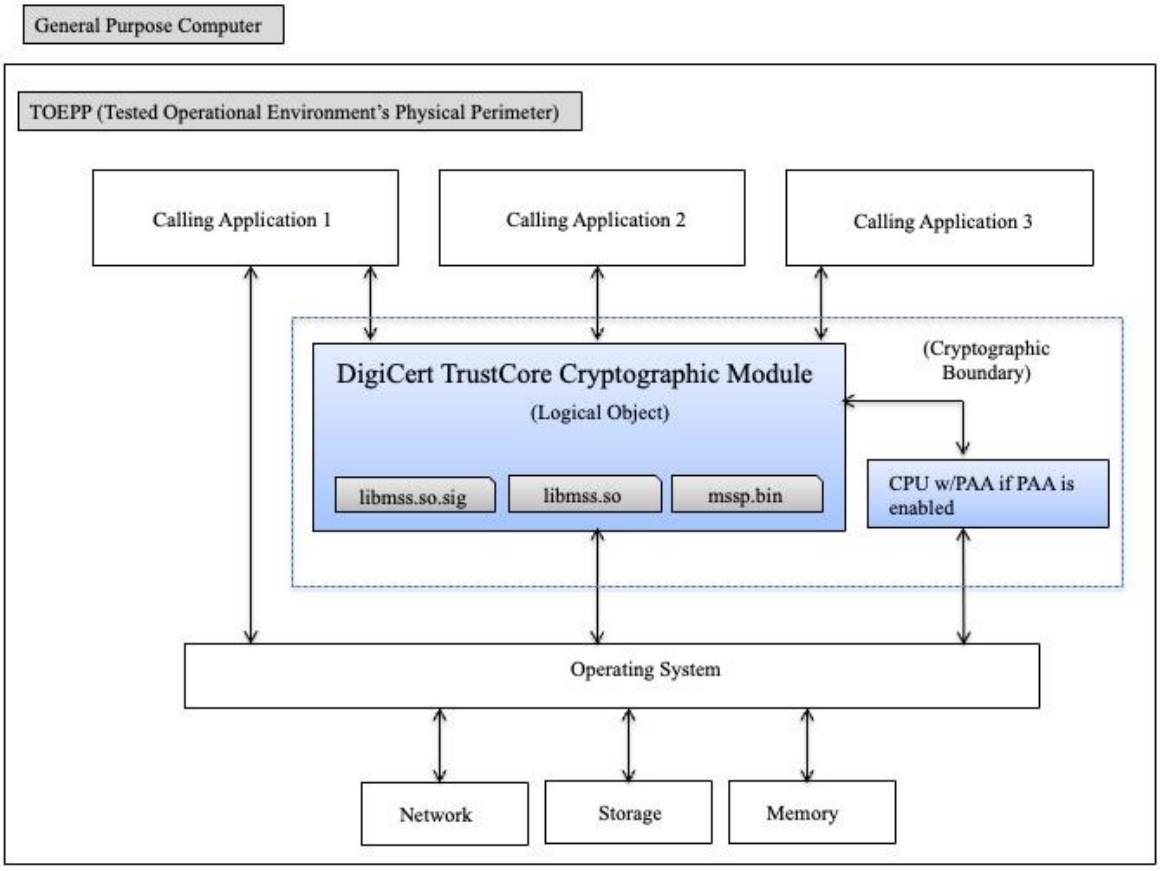


Figure 1: Logical [cryptographic] boundary [and physical perimeter if combined]

## 2.2 Tested and Vendor Affirmed Module Version and Identification

| Package or File Name | Software/ Firmware Version | Features | Integrity Test |
|---|---|---|---|
| E3950-libmss.so | 7.0.0f_u1 | Xerox Explorer 6.5 with Intel Atom E3950 with and without PAA | HMAC-SHA2-256 |
| x6413E-libmss.so | 7.0.0f_u1 | Xerox Explorer 8.0 with Intel Atom x6413E with and without PAA | HMAC-SHA2-256 |
| ARM-A53-libmss.so | 7.0.0f_u1 | Xerox Alexandra Platform with ARM Cortex A53 without PAA | HMAC-SHA2-256 |
| X1600E-libmss.so | 7.0.0f_u1 | Ultra Charge & Communication Base | HMAC-SHA2-256 |

| Package or File Name | Software/ Firmware Version | Features | Integrity Test |
|---|---|---|---|
| | | with Ingenic X1600E without PAA | |
| X2000-libmss.so | 7.0.0f_u1 | Honeywell Cordless Ultra with Ingenic X2000 without PAA | HMAC-SHA2-256 |

Table 3: Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets)

**Tested Operational Environments - Software, Firmware, Hybrid:**

The DigiCert TrustCore Cryptographic Suite B Module is tested on the following operational environments:

| Operating System | Hardware Platform | Processors | PAA/PAI | Hypervisor or Host OS | Version(s) |
|---|---|---|---|---|---|
| Yocto Linux 3.1 (32-bit) | Xerox Explorer 6.5 | Intel Atom E3950 | Yes | N/A | 7.0.0f_u1 |
| Yocto Linux 3.1 (32-bit) | Xerox Explorer 6.5 | Intel Atom E3950 | No | N/A | 7.0.0f_u1 |
| Yocto Linux 3.1 (32-bit) | Xerox Explorer 8.0 | Intel Atom x6413E | Yes | N/A | 7.0.0f_u1 |
| Yocto Linux 3.1 (32-bit) | Xerox Explorer 8.0 | Intel Atom x6413E | No | N/A | 7.0.0f_u1 |
| Yocto Linux 3.1 (32-bit) | Xerox Alexandra Platform | ARM Cortex A53 | No | N/A | 7.0.0f_u1 |
| Buildroot Linux 2020.02.01 (32-bit) | Ultra Charge & Communication Base | Ingenic X1600E | No | N/A | 7.0.0f_u1 |
| Buildroot Linux 2020.02.01 (32-bit) | Honeywell Cordless Ultra | Ingenic X2000 | No | N/A | 7.0.0f_u1 |

Table 4: Tested Operational Environments - Software, Firmware, Hybrid

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

The DigiCert TrustCore Cryptographic Suite B Module is tested on the following operational environments.

| Operating System | Hardware Platform |
|---|---|
| Ubuntu Linux 4.15 (64-bit) | Intel NUC with i7-8650U processor with and without PAA |
| Yocto Linux 4.4.11 | DVF99 using ARM926EJ-S |
| Yocto Linux 4.4.11 | DVF101 using ARM Cortex-A9 with Neon |
| Wind River Linux 2.6.27.18 | BCM111XX using ARM1176JZF-S |

Table 5: Vendor-Affirmed Operational Environments - Software, Firmware, Hybrid

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

## 2.3 Excluded Components

No components are excluded from [140-3] requirements.

## 2.4 Modes of Operation

**Modes List and Description:**

| Mode Name | Description | Type | Status Indicator |
|---|---|---|---|
| Approved Mode | Only approved or allowed security functions with sufficient key security strength can be used | Approved | FIPS_eventLog, FIPS_EventType |
| Non-Approved Mode | When non-approved security functions or approved security functions with insufficient key security strength are used. | Non-Approved | FIPS_eventLog, FIPS_EventType |

Table 6: Modes List and Description

The Module enters approved mode after pre-operational self-tests have successfully completed. Once the Module is operational, the mode of operation is implicitly assumed depending on the security function invoked and the security strength of the cryptographic keys. To provide an indicator of the current mode of operation, an asynchronous callout event mechanism is provided.

The application using the Module can register for a FIPS_eventLog call-out function to be used as the approved-mode/non-approved-mode indicator. The Module uses the scenario of a shared indicator for multiple services, per IG 2.4.C example #3. It uses a dedicated status output interface that is a software callback function. The calling application using the Module registers a callback function to be called at the beginning and end of all services and algorithm implementations. The application's FIPS_eventLog function will be called from the Module at the beginning and end of each approved or non-approved service or algorithm. This FIPS_eventLog and the FIPS_EventType enumeration value provided as a parameter serves as a thread-safe status indicator of the current approved or non-approved mode of operation.

**Mode Change Instructions and Status:**
The approved mode of operation is configured at instantiation of the Module by the Cryptographic Officer role by execution of an application or protocol operating system process that uses the Module's cryptographic functions.

The Module transitions to the non-approved mode of operation when one of the non-approved security functions is utilized. The Module can transition back to the approved mode of operation by utilizing an approved security function.

Keys and CSPs are not shared between the approved and non-approved mode of operation.

**Degraded Mode Description:**
N/A

## 2.5 Algorithms

**Approved Algorithms:**
The Module implements the approved cryptographic algorithms listed the table below.

| Algorithm | CAVP Cert | Properties | Reference |
|---|---|---|---|
| AES-CBC | A4653 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CCM | A4653 | Key Length - 128, 192, 256<br>Tag Length - 112, 128, 32, 48, 64, 80, 96<br>IV Length - IV Length: 56-104 Increment 8<br>Payload Length - Payload Length: 0-256 Increment 8<br>AAD Length - AAD Length: 0-524288 Increment 8 | SP 800-38C |
| AES-CFB128 | A4653 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CMAC | A4653 | Direction - Generation, Verification<br>Key Length - 128, 192, 256<br>MAC Length - MAC Length: 32-128 Increment 8<br>Message Length - Message Length: 0-65536 Increment 8 | SP 800-38B |
| AES-CTR | A4653 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256<br>Payload Length - Payload Length: 8-128 Increment 8<br>Supports Counter larger than maximum value - No<br>Incremental Counter - Yes<br>Counter Tests Performed - Yes | SP 800-38A |
| AES-ECB | A4653 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-GCM | A4654 | Direction - Decrypt, Encrypt<br>IV Generation - External<br>IV Generation Mode - 8.2.1<br>Key Length - 128, 192, 256<br>Tag Length - 104, 112, 120, 128, 32, 64, 96<br>IV Length - IV Length: 8-1024 Increment 8<br>Payload Length - Payload Length: 8-65536 Increment 8<br>AAD Length - AAD Length: 0-65536 Increment 8 | SP 800-38D |
| AES-GCM | A4655 | Direction - Decrypt, Encrypt<br>IV Generation - External<br>IV Generation Mode - 8.2.1<br>Key Length - 128, 192, 256<br>Tag Length - 104, 112, 120, 128, 32, 64, 96<br>IV Length - IV Length: 8-1024 Increment 8<br>Payload Length - Payload Length: 8-65536 Increment 8<br>AAD Length - AAD Length: 0-65536 Increment 8 | SP 800-38D |
| AES-GMAC | A4654 | Direction - Decrypt, Encrypt<br>IV Generation - External<br>Key Length - 128, 192, 256<br>Tag Length - 104, 112, 120, 128, 32, 64, 96 | SP 800-38D |

| Algorithm | CAVP Cert | Properties | Reference |
|---|---|---|---|
| | | IV Length - IV Length: 8-1024 Increment 8<br>AAD Length - AAD Length: 0-4096 Increment 8 | |
| AES-GMAC | A4655 | Direction - Decrypt, Encrypt<br>IV Generation - External<br>Key Length - 128, 192, 256<br>Tag Length - 104, 112, 120, 128, 32, 64, 96<br>IV Length - IV Length: 8-1024 Increment 8<br>AAD Length - AAD Length: 0-4096 Increment 8 | SP 800-38D |
| AES-OFB | A4653 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-XTS Testing Revision 2.0 | A4653 | Direction - Decrypt, Encrypt<br>Key Length - 128, 256<br>Payload Length - Payload Length: 128-65536 Increment 128<br>Tweak Mode - Hex<br>Data Unit Length Matches Payload Length - Yes | SP 800-38E |
| Counter DRBG | A4653 | Prediction Resistance - No, Yes<br>Supports Reseed - Yes<br>Mode - AES-128, AES-192, AES-256<br>Derivation Function Enabled - No, Yes<br>Additional Input - Additional Input: 0-16777216 Increment 8, Additional Input: 0-256 Increment 8, Additional Input: 0-320 Increment 8, Additional Input: 0-384 Increment 8<br>Entropy Input - Entropy Input: 128-16777216 Increment 8, Entropy Input: 192-16777216 Increment 8, Entropy Input: 256, Entropy Input: 256-16777216 Increment 8, Entropy Input: 320, Entropy Input: 384<br>Nonce - Nonce: 0, Nonce: 128-16777216 Increment 8, Nonce: 64-16777216 Increment 8, Nonce: 96-16777216 Increment 8<br>Personalization String Length - Personalization String Length: 0-16777216 Increment 8, Personalization String Length: 0-256 Increment 8, Personalization String Length: 0-320 Increment 8, Personalization String Length: 0-384 Increment 8<br>Returned Bits - 512 | SP 800-90A Rev. 1 |
| DSA KeyGen (FIPS186-4) | A4653 | L - 2048, 3072<br>N - 224, 256 | FIPS 186-4 |
| DSA PQGGen (FIPS186-4) | A4653 | P/Q Generation Methods - Probable<br>G Generation Methods - Unverifiable<br>L - 2048, 3072<br>N - 224, 256<br>Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512 | FIPS 186-4 |
| DSA PQGVer (FIPS186-4) | A4653 | P/Q Generation Methods - Probable<br>G Generation Methods - Unverifiable<br>L - 1024, 2048, 3072<br>N - 160, 224, 256 | FIPS 186-4 |

| Algorithm | CAVP Cert | Properties | Reference |
|---|---|---|---|
| | | Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512 | |
| DSA SigGen (FIPS186-4) | A4653 | L - 2048, 3072<br>N - 224, 256<br>Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512 | FIPS 186-4 |
| DSA SigVer (FIPS186-4) | A4653 | L - 1024, 2048, 3072<br>N - 160, 224, 256<br>Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512 | FIPS 186-4 |
| ECDSA KeyGen (FIPS186-4) | A4653 | Curve - P-224, P-256, P-384, P-521<br>Secret Generation Mode - Testing Candidates | FIPS 186-4 |
| ECDSA KeyVer (FIPS186-4) | A4653 | Curve - P-192, P-224, P-256, P-384, P-521 | FIPS 186-4 |
| ECDSA SigGen (FIPS186-4) | A4653 | Component - No<br>Curve - P-224, P-256, P-384, P-521<br>Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512 | FIPS 186-4 |
| ECDSA SigVer (FIPS186-4) | A4653 | Component - No<br>Curve - P-192, P-224, P-256, P-384, P-521<br>Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512 | FIPS 186-4 |
| EDDSA KeyGen | A4653 | Curve - ED-25519, ED-448 | FIPS 186-5 |
| EDDSA KeyVer | A4653 | Curve - ED-25519, ED-448 | FIPS 186-5 |
| EDDSA SigGen | A4653 | Curve - ED-25519, ED-448<br>PreHash - Yes | FIPS 186-5 |
| EDDSA SigVer | A4653 | Curve - ED-25519, ED-448<br>PreHash - No<br>Pure - Yes | FIPS 186-5 |
| HMAC-SHA-1 | A4653 | MAC - MAC: 32-160 Increment 8<br>Key Length - Key Length: 112-65536 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-224 | A4653 | MAC - MAC: 32-224 Increment 8<br>Key Length - Key Length: 112-65536 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-256 | A4653 | MAC - MAC: 32-256 Increment 8<br>Key Length - Key Length: 112-65536 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-384 | A4653 | MAC - MAC: 32-384 Increment 8<br>Key Length - Key Length: 112-65536 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-512 | A4653 | MAC - MAC: 32-512 Increment 8<br>Key Length - Key Length: 112-65536 Increment 8 | FIPS 198-1 |
| HMAC-SHA3-224 | A4653 | MAC - MAC: 32-224 Increment 8<br>Key Length - Key Length: 112-65536 Increment 8 | FIPS 198-1 |
| HMAC-SHA3-256 | A4653 | MAC - MAC: 32-256 Increment 8<br>Key Length - Key Length: 112-65536 Increment 8 | FIPS 198-1 |

| Algorithm | CAVP Cert | Properties | Reference |
|---|---|---|---|
| HMAC-SHA3-384 | A4653 | MAC - MAC: 32-384 Increment 8<br>Key Length - Key Length: 112-65536 Increment 8 | FIPS 198-1 |
| HMAC-SHA3-512 | A4653 | MAC - MAC: 32-512 Increment 8<br>Key Length - Key Length: 112-65536 Increment 8 | FIPS 198-1 |
| KAS-ECC Sp800-56Ar3 | A4653 | Domain Parameter Generation Methods - P-224, P-256, P-384, P-521<br>Function - Full Validation, Key Pair Generation<br>iutId - 123456ABCD4242<br>Scheme -<br>ephemeralUnified -<br>KAS Role - Initiator, Responder<br>KDF Methods -<br>twoStepKdf -<br>MAC Salting Methods - default<br>Supported Lengths - Supported Lengths: 8-4096 Increment 8<br>Fixed Info Pattern - label\|\|literal[00]\|\|context\|\|l<br>Fixed Info Encoding - Concatenation<br>KDF Mode - Feedback<br>MAC Modes - HMAC-SHA-1, HMAC-SHA2-224, HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512<br>Fixed Data Order - After Fixed Data<br>Counter Lengths - 8<br>The KDF supports an empty IV - Yes<br>The KDF requires an empty IV - No<br>Key Length - 256 | SP 800-56A Rev. 3 |
| KAS-FFC Sp800-56Ar3 | A4653 | Domain Parameter Generation Methods - MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192<br>Function - Full Validation, Key Pair Generation<br>iutId - 123456ABCD4242<br>Scheme -<br>dhEphem -<br>KAS Role - Initiator, Responder<br>KDF Methods -<br>twoStepKdf -<br>MAC Salting Methods - default<br>Supported Lengths - Supported Lengths: 8-4096 Increment 8<br>Fixed Info Pattern - label\|\|literal[00]\|\|context\|\|l<br>Fixed Info Encoding - Concatenation<br>KDF Mode - Feedback<br>MAC Modes - HMAC-SHA-1, HMAC-SHA2-224, HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512<br>Fixed Data Order - After Fixed Data<br>Counter Lengths - 8<br>The KDF supports an empty IV - Yes<br>The KDF requires an empty IV - No<br>Key Length - 256 | SP 800-56A Rev. 3 |

| Algorithm | CAVP Cert | Properties | Reference |
|---|---|---|---|
| KDF SP800-108 | A4653 | KDF Mode - Feedback<br>MAC Mode - HMAC-SHA-1, HMAC-SHA2-224, HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512, HMAC-SHA3-224, HMAC-SHA3-256, HMAC-SHA3-384, HMAC-SHA3-512<br>Supported Lengths - Supported Lengths: 8-4096 Increment 8<br>Fixed Data Order - After Fixed Data<br>Counter Length - 8<br>Supports Empty IV - Yes<br>Requires Empty IV - No<br>Custom Key In Length - 0 | SP 800-108 Rev. 1 |
| RSA KeyGen (FIPS186-4) | A4653 | Key Generation Mode - B.3.3, B.3.5<br>Modulo - 2048, 3072, 4096<br>Primality Tests - Table C.3<br>Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512<br>Info Generated By Server - Yes<br>Public Exponent Mode - Fixed<br>Fixed Public Exponent - 010001<br>Private Key Format - Standard | FIPS 186-4 |
| RSA SigGen (FIPS186-4) | A4653 | Signature Type - PKCS 1.5, PKCSPSS<br>Modulo - 2048, 3072, 4096<br>Hash Pair -<br>Hash Algorithm - SHA2-224 | FIPS 186-4 |
| RSA SigVer (FIPS186-4) | A4653 | Signature Type - PKCS 1.5, PKCSPSS<br>Modulo - 1024, 2048, 3072, 4096<br>Hash Pair -<br>Hash Algorithm - SHA2-512<br>Public Exponent Mode - Fixed<br>Fixed Public Exponent - 010001 | FIPS 186-4 |
| SHA-1 | A4653 | Message Length - Message Length: 160, 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-224 | A4653 | Message Length - Message Length: 224, 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-256 | A4653 | Message Length - Message Length: 256, 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-384 | A4653 | Message Length - Message Length: 384, 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-512 | A4653 | Message Length - Message Length: 512, 0-65536 Increment 8 | FIPS 180-4 |
| SHA3-224 | A4653 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 202 |
| SHA3-256 | A4653 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 202 |
| SHA3-384 | A4653 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 202 |
| SHA3-512 | A4653 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 202 |
| SHAKE-128 | A4653 | Supports Bit-Oriented Messages - No<br>Supports Empty Message - Yes<br>Supports Bit-Oriented Output - No<br>Output Length - Output Length: 16-65536 Increment 8 | FIPS 202 |
| SHAKE-256 | A4653 | Supports Bit-Oriented Messages - No<br>Supports Empty Message - Yes | FIPS 202 |

| Algorithm | CAVP Cert | Properties | Reference |
|---|---|---|---|
| | | Supports Bit-Oriented Output - No<br>Output Length - Output Length: 16-65536 Increment 8 | |

Table 7: Approved Algorithms

**Vendor-Affirmed Algorithms:**

The Module implements the FIPS Vendor Affirmed cryptographic algorithms listed.

| Name | Properties | Implementation | Reference |
|---|---|---|---|
| CKG 1 | ECDSA:Asymmetric<br>EDDSA:Asymmetric<br>RSA:Asymmetric<br>DSA:Asymmetric | NA | SP800-133rev2 Section 4 Example 1 and IG D.H |
| CKG 2 | KAS-ECC: Asymmetric<br>KAS-FFC:Asymmetric | NA | SP800-133rev2 Section 4 Example 1 and IG D.H |

Table 8: Vendor-Affirmed Algorithms

Note:
CKG does not have a certificate as IG D.H. states: “The module’s validation certificate shall have a CKG entry only if the module is generating keys for the symmetric-key algorithms.”

**Non-Approved, Allowed Algorithms:**

The module does not implement any Non-Approved, but Allowed Algorithms in the Approved Mode of Operation

N/A for this module.

**Non-Approved, Allowed Algorithms with No Security Claimed:**

The module does not implement any Non-Approved, Algorithms Allowed with No Security Claimed in the Approved Mode of Operation

N/A for this module.

**Non-Approved, Not Allowed Algorithms:**

The Module implements the Non-Approved, Not Allowed cryptographic algorithms listed.

| Name | Use and Function |
|---|---|
| AES-EAX (NC) | Authentication and Encryption |
| AES GCM 256-bit (NC) | 256-bit Encryption/Decryption for 256-bit state table implementation |
| AES GMAC 256-bit (NC) | 256-bit Message Authentication for 256-bit state table implementation |

| Name | Use and Function |
|---|---|
| AES XCBC (NC) | Message Authentication |
| DES (NC) | Encryption/Decryption |
| DH (NC) | Key Agreement: Key establishment methodology provides less than 112 bits of encryption strength |
| DSA (NC) | SigGen using L=2048, N=224 using SHA-1 |
| ECC CDH (NC) | Key Agreement: Key establishment methodology provides less than 112 bits of encryption strength |
| EDDH (NC) | Curve 25519; Curve 448 |
| HMAC (NC) | HMAC generation with key size less than 112 bits |
| HMAC-MD5 (NC) | Message Authentication |
| MD2, MD4, MD5 (NC) | Message Digest |
| RNG (NC) | FIPS 186-2 Random Number Generation |
| RSA (NC) | Key Wrapping: Key establishment methodology provides less than 112 bits of encryption strength |
| RSA-OAEP (NC) | PKCS#1 v2.1 RSAES-OAEP Encryption/Decryption |
| RSA (Key Wrapping) (NC) | Key establishment methodology provides between 112 and 138 bits of encryption strength. Per IG D.G, the module wraps data sent by the requesting application via an API call. Data being wrapped is unknown. PKCS non-approved padding. |
| Triple-DES (NC) | Encryption/Decryption |

Table 9: Non-Approved, Not Allowed Algorithms

Note: All the various AES modes (e.g., EAX, XCBC, XTS, etc.) use the same underlying AES implementation as the approved AES cert.

## 2.6 Security Function Implementations

The table below shows the Security Function Implementations that the module implements:

| Name | Type | Description | Properties | Algorithms |
|---|---|---|---|---|
| SFI-AES-UnAuth-Encrypt | BC-UnAuthEncrypt | Block Cipher Encryption | | AES-CBC: (A4653)<br>AES-ECB: (A4653)<br>AES-CTR: (A4653)<br>AES-CFB128: (A4653)<br>AES-OFB: (A4653)<br>AES-XTS Testing Revision 2.0: (A4653) |
| SFI-AES-UnAuth-Decrypt | BC-UnAuthDecrypt | Block Cipher Decryption | | AES-CBC: (A4653)<br>AES-ECB: (A4653)<br>AES-CTR: (A4653)<br>AES-CFB128: |

| Name | Type | Description | Properties | Algorithms |
|---|---|---|---|---|
| | | | | (A4653)<br>AES-OFB: (A4653)<br>AES-XTS Testing Revision 2.0: (A4653) |
| SFI-AES-CCM-Encrypt | BC-AuthEncrypt | Block Cipher Encryption | | AES-CCM: (A4653) |
| SFI-AES-CCM-Decrypt | BC-AuthDecrypt | Block Cipher Decryption | | AES-CCM: (A4653) |
| SFI-AES-GCM-Encrypt | BC-AuthEncrypt | Block Cipher Encryption | | AES-GCM: (A4654, A4655) |
| SFI-AES-GCM-Decrypt | BC-AuthDecrypt | Block Cipher Decryption | | AES-GCM: (A4654, A4655) |
| SFI-SHS | SHA | Secure Hash Standard | IG C.B: | SHA-1: (A4653)<br>SHA2-224: (A4653)<br>SHA2-256: (A4653)<br>SHA2-384: (A4653)<br>SHA2-512: (A4653) |
| SFI-SHA3 | SHA | Secure Hash Standard | Publications: IG C.B, IG C.C | SHA3-224: (A4653)<br>SHA3-256: (A4653)<br>SHA3-384: (A4653)<br>SHA3-512: (A4653) |
| SFI-SHAKE | SHA | SHAKE Extendable Output Function | Publications: IG C.C | SHAKE-128: (A4653)<br>SHAKE-256: (A4653) |
| SFI-DSA-KeyGen | AsymKeyPair-KeyGen | Asymmetric Key-Pair Generation | | DSA KeyGen (FIPS186-4): (A4653)<br>Counter DRBG: (A4653) |
| SFI-DSA-PQGGen | AsymKeyPair-DomPar | AsymKeyPair-Domain Parameter Generation | | DSA PQGGen (FIPS186-4): (A4653)<br>Counter DRBG: (A4653)<br>SHA2-224: (A4653) |

| Name | Type | Description | Properties | Algorithms |
|---|---|---|---|---|
| | | | | SHA2-256: (A4653)<br>SHA2-384: (A4653)<br>SHA2-512: (A4653) |
| SFI-DSA-PQGVer | AsymKeyPair-DomPar | AsymKeyPair-Domain Parameter Verification | | DSA PQGVer (FIPS186-4): (A4653)<br>Counter DRBG: (A4653)<br>SHA2-224: (A4653)<br>SHA2-256: (A4653)<br>SHA2-384: (A4653)<br>SHA2-512: (A4653) |
| SFI-DSA-SigGen | DigSig-SigGen | Digital Signature Generation | | DSA SigGen (FIPS186-4): (A4653)<br>Counter DRBG: (A4653)<br>SHA2-224: (A4653)<br>SHA2-256: (A4653)<br>SHA2-384: (A4653)<br>SHA2-512: (A4653) |
| SFI-DSA-SigVer | DigSig-SigVer | Digital Signature Verification | | DSA SigVer (FIPS186-4): (A4653)<br>Counter DRBG: (A4653)<br>SHA2-224: (A4653)<br>SHA2-256: (A4653)<br>SHA2-384: (A4653)<br>SHA2-512: (A4653) |

| Name | Type | Description | Properties | Algorithms |
|---|---|---|---|---|
| SFI-ECDSA-KeyGen | AsymKeyPair-KeyGen | Asymmetric Key-Pair Generation | | ECDSA KeyGen (FIPS186-4): (A4653) Counter DRBG: (A4653) |
| SFI-ECDSA-KeyVer | AsymKeyPair-KeyVer | Asymmetric Key-Pair Verification | | ECDSA KeyVer (FIPS186-4): (A4653) Counter DRBG: (A4653) |
| SFI-ECDSA-SigGen | DigSig-SigGen | Digital Signature Generation | | ECDSA SigGen (FIPS186-4): (A4653) Counter DRBG: (A4653) SHA2-224: (A4653) SHA2-256: (A4653) SHA2-384: (A4653) SHA3-512: (A4653) |
| SFI-ECDSA-SigVer | DigSig-SigVer | Digital Signature Verification | | ECDSA SigVer (FIPS186-4): (A4653) Counter DRBG: (A4653) SHA-1: (A4653) SHA2-224: (A4653) SHA2-256: (A4653) SHA2-384: (A4653) SHA2-512: (A4653) |
| SFI-EdDSA-KeyGen | AsymKeyPair-KeyGen | Asymmetric Key-Pair Generation | | EDDSA KeyGen: (A4653) Counter DRBG: (A4653) |
| SFI-EdDSA-KeyVer | AsymKeyPair-KeyVer | Asymmetric Key-Pair Verification | | EDDSA KeyVer: (A4653) Counter DRBG: (A4653) |

| Name | Type | Description | Properties | Algorithms |
|---|---|---|---|---|
| SFI-EdDSA-SigGen | DigSig-SigGen | Digital Signature Generation | | EDDSA SigGen: (A4653) SHAKE-256: (A4653) SHA2-512: (A4653) Counter DRBG: (A4653) |
| SFI-EdDSA-SigVer | DigSig-SigVer | Digital Signature Verification | | EDDSA SigVer: (A4653) SHAKE-256: (A4653) SHA3-512: (A4653) Counter DRBG: (A4653) |
| SFI-RSA-KeyGen | AsymKeyPair-KeyGen | Asymmetric Key-Pair Generation | [IG C.E]: | RSA KeyGen (FIPS186-4): (A4653) Counter DRBG: (A4653) |
| SFI-RSA-SigGen | DigSig-SigGen | Digital Signature Generation using PKCS1v1.5 | | RSA SigGen (FIPS186-4): (A4653) SHA2-224: (A4653) SHA2-256: (A4653) SHA2-384: (A4653) SHA2-512: (A4653) Counter DRBG: (A4653) |
| SFI-RSA-PSS-SigGen | DigSig-SigGen | Digital Signature Generation using PSS | | RSA SigGen (FIPS186-4): (A4653) SHA2-224: (A4653) SHA2-256: (A4653) SHA2-384: (A4653) SHA2-512: (A4653) |

| Name | Type | Description | Properties | Algorithms |
|---|---|---|---|---|
| | | | | Counter DRBG: (A4653) |
| SFI-RSA-SigVer | DigSig-SigVer | Digital Signature Verification | | RSA SigVer (FIPS186-4): (A4653)<br>SHA-1: (A4653)<br>SHA2-224: (A4653)<br>SHA2-256: (A4653)<br>SHA2-384: (A4653)<br>SHA2-512: (A4653)<br>Counter DRBG: (A4653) |
| SFI-RSA-PSS-SigVer | DigSig-SigVer | Digital Signature Verification using PSS | | RSA SigVer (FIPS186-4): (A4653)<br>SHA-1: (A4653)<br>SHA2-224: (A4653)<br>SHA2-256: (A4653)<br>SHA2-384: (A4653)<br>SHA2-512: (A4653)<br>Counter DRBG: (A4653) |
| SFI-CKG-AsymKeyGen | AsymKeyPair-KeyGen | Asymmetric Key Generation | Publications:[IG D.H] | ECDSA KeyGen (FIPS186-4): (A4653)<br>EDDSA KeyGen: (A4653)<br>RSA KeyGen (FIPS186-4): (A4653)<br>CKG 1: ()<br>ECDSA: Asymmetric<br>EDDSA: Asymmetric<br>RSA: Asymmetric |

| Name | Type | Description | Properties | Algorithms |
|---|---|---|---|---|
| SFI-CKG-AsymKeyGen KAS-ECC | KAS-KeyGen | Asymmetric Key Generation | Publications:[IG D.H] | KAS-ECC Sp800-56Ar3: (A4653) CKG 2: () KAS-ECC: Asymmetric |
| SFI-CKG-AsymKeyGen KAS-FFC | KAS-KeyGen | Asymmetric Key Generation | Publications:[IG D.H] | KAS-FFC Sp800-56Ar3: (A4653) CKG 2: () KAS-FFC: Asymmetric |
| SFI-AES-CMAC | MAC | Message Authentication Generation | | AES-CMAC: (A4653) |
| SFI-AES-GMAC | MAC | Message Authentication Generation | | AES-GMAC: (A4654, A4655) |
| SFI-HMAC | MAC | Message Authentication Generation | | HMAC-SHA-1: (A4653) HMAC-SHA2-224: (A4653) HMAC-SHA2-256: (A4653) HMAC-SHA2-384: (A4653) HMAC-SHA2-512: (A4653) HMAC-SHA3-224: (A4653) HMAC-SHA3-256: (A4653) HMAC-SHA3-384: (A4653) HMAC-SHA3-512: (A4653) |
| SFI-HMAC-KDF | KBKDF | Key-Based Key Derivation | Publications: [IG D.F] | KDF SP800-108: (A4653) HMAC-SHA-1: (A4653) HMAC-SHA2-224: (A4653) HMAC-SHA2-256: (A4653) HMAC-SHA2-384: (A4653) HMAC-SHA2-512: (A4653) |

| Name | Type | Description | Properties | Algorithms |
|---|---|---|---|---|
| | | | | HMAC-SHA3-224: (A4653)<br>HMAC-SHA3-256: (A4653)<br>HMAC-SHA3-384: (A4653)<br>HMAC-SHA3-512: (A4653) |
| SFI-DRBG-Generate | DRBG | Random Number Generation | | AES-CTR: (A4653) |
| SFI-DRBG-ReSeed | DRBG | Random Number ReSeed | | Counter DRBG: (A4653) |
| SFI-KAS-ECC | KAS-Full | Key Agreement | | KAS-ECC Sp800-56Ar3: (A4653)<br>HMAC-SHA-1: (A4653)<br>HMAC-SHA2-224: (A4653)<br>HMAC-SHA2-256: (A4653)<br>HMAC-SHA2-384: (A4653)<br>HMAC-SHA2-512: (A4653)<br>Counter DRBG: (A4653) |
| SFI-KAS-FFC | KAS-Full | Key Agreement | | KAS-FFC Sp800-56Ar3: (A4653)<br>HMAC-SHA-1: (A4653)<br>HMAC-SHA2-224: (A4653)<br>HMAC-SHA2-256: (A4653)<br>HMAC-SHA2-384: (A4653)<br>HMAC-SHA2-512: (A4653)<br>Counter DRBG: (A4653) |

Table 10: Security Function Implementations

## 2.7 Algorithm Specific Information

**AES GCM IV Uniqueness: FIPS 140-3 IG C.H., Option 1**

The AES GCM implementation generates GCM IVs deterministically as specified in SP800-38D Section 8.2.1 using the following protocols:

**TLS 1.2 Protocol IV generation for GCM Cipher Suites:** The Module supports TLS 1.2 GCM Cipher Suites for TLS, as described in RFCs 5116, 5246, 5288 and 5289 and shall only be used for the TLS protocol version 1.2 to be compliant with FIPS140-3 IG C.H, Option 1.

Per [IG] C.H. technique 1.a. TLS 1.2 GCM Cipher Suites for TLS method was used for testing during operational testing. During operational testing, the Module was tested against an independent version of TLS and found to behave correctly.

The counter portion of the IV is set by the Module within its cryptographic boundary.

The nonce explicit part of the IV is incremented each time an AES GCM computation is performed. The Module establishes a new session key when the nonce-explicit part of the IV exhausts the maximum number of possible values ($2^{32}$-1).

In case the Module's power is lost and then restored, a new key for use with the AES GCM encryption/decryption shall be established.

**Protocol specific KDF listed in SP 800-135rev1: FIPS-140-3 IG D.C Option 4**
The cryptographic module supports HMAC-KDF used by upper-level protocols, but does not implement protocol specific KDFs as defined in SP-800-135rev1. This module does not implement, use or depend upon protocols listed in SP 800-135rev1.

**AES XTS Requirements on the Key: FIPS 140-3 IG C.I.**

Per [IG] C.I. the XTS algorithm implementation includes a check to ensure Key_1 ≠ Key_2

**KAS Requirements on the Key: FIPS 140-3 IG D.F., Scenario 2 path (2)**

KAS [56Ar3] - Per [IG] D.F Scenario 2 path (2), compliant key agreement scheme where testing is performed end-to-end for the shared secret computation and a KDF compliant with SP800-56Cr2. The Module has both a KAS-ECC and KAS-FFC certificate with no key confirmation.

## 2.8 RBG and Entropy

The Module does not have a specific entropy source. Entropy must be provided by the calling application through the API.

The Module implements a CTR-based DRBG per SP800-90Ar1 for creation of symmetric and asymmetric keys.

The Module accepts input from entropy sources external to the cryptographic boundary for use as seed material for the Module's approved DRBGs. External entropy can be added via several APIs available to the cryptographic module client application.

The calling application of the Module shall use entropy sources that meet the security strength required for the random bit generation mechanism as shown in NIST SP800-90Ar1 Table 3 (CTR_DRBG). A

minimum of 384 bits of entropy must be provided by the calling application. The calling application shall provide full entropy for 256-bit keys. When the CTR_DRBG is used without a derivation function full entropy must be provided, per SP 800.90Ar1, IG D.L. Due to the entropy being provided by an external source, the following caveat applies: There is no assurance of the minimum strength of generated SSPs (e.g. keys).

The Module performs DRBG health tests (Instantiate, Generate, ReSeed) as defined in Section 11.3 of SP800-90Ar1.

## 2.9 Key Generation

For Key Generation, see Section 2.5 and Section 2.6 above.

## 2.10 Key Establishment

**Key Agreement Information**

For Key Agreement, see Section 2.5 and Section 2.6 above.

**Key Transport Information**

N/A. The module does not implement key transport.

## 2.11 Industry Protocols

The module does not implement any Industry Protocols

# 3 Cryptographic Module Interfaces

## 3.1 Ports and Interfaces

The Module's ports and associated defined logical interface categories are listed below. The module's logical interface (API) provides logical separation of the input and output interfaces.

| Physical Port | Logical Interface(s) | Data That Passes |
|---|---|---|
| N/A | Data Input | Input parameters of API function calls |
| N/A | Data Output | Output parameters of API function calls |
| N/A | Control Input | API Function Calls |
| N/A | Control Output | Output of the FIPS_EventType enumeration value and an indication of approved or non-approved mode of operation. |
| N/A | Status Output | For Approved mode, function calls returning status information and return code provided by API function calls |
| N/A | Power | None |

Table 11: Ports and Interfaces

# 4 Roles, Services, and Authentication

## 4.1 Authentication Methods

Note: The module is Level 1 and does not implement any Authentication techniques.

N/A for this module.

## 4.2 Roles

The Module supports one distinct operator role, Cryptographic Officer (CO).

The Roles Table below lists all operator roles supported by the Module.

The Module does not support concurrent operators, bypass capability, or a maintenance role. The Cryptographic Officer role is implicitly identified by the service that is requested.

| Name | Type | Operator Type | Authentication Methods |
|---|---|---|---|
| CO | Role | Cryptographic Officer | None |

Table 12: Roles

## 4.3 Approved Services

All approved services implemented by the Module are listed in the table below:

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| AES Encrypt | Perform encryption on a block of data using the shared key Modes: AES-CBC AES-ECB AES-CTR AES-CFB128 AES-OFB AES-XTS Testing Revision 2 | Approved | Encrypt command with AES, input data, input size, key, key size, mode of operation | Ciphertext and return status of OK or error condition | SFI-AES-UnAuth-Encrypt | CO<br>- AES Keys: W,E |
| AES Decrypt | Perform decryption on a block of data using the shared key Modes: AES- | Approved | Decrypt command with AES, input data, input size, key, key size, | Plaintext and return status of OK or error condition | SFI-AES-UnAuth-Decrypt | CO<br>- AES Keys: W,E |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| | CBC AES-ECB AES-CTR AES-CFB128 AES-OFB AES-XTS Testing Revision 2 | | mode of operation | | | |
| AES-CCM Encrypt | Perform encryption on a block of data using the shared key and CCM message authentication code | Approved | Encrypt command with AES, input data, input size, key, key size, mode of operation | Ciphertext and return status of OK or error condition | SFI-AES-CCM-Encrypt | CO<br>- AES Keys: W,E |
| AES-CCM Decrypt | Perform decryption on a block of data using the shared key and CCM message authentication code | Approved | Decrypt command with AES, input data, input size, key, key size, mode of operation | Plaintext and return status of OK or error condition | SFI-AES-CCM-Decrypt | CO<br>- AES Keys: W,E |
| AES-GCM Encrypt | Perform encryption on a block of data using the shared key and GCM message authentication code | Approved | Encrypt command with AES, input data, input size, key, key size, mode of operation | Ciphertext and return status of OK or error condition | SFI-AES-GCM-Encrypt | CO<br>- AES Keys: W,E |
| AES-GCM Decrypt | Perform decryption on a block of data using the shared key and GCM message authentication code | Approved | Decrypt command with AES, input data, input size, key, key size, mode of operation | Plaintext and return status of OK or error condition | SFI-AES-GCM-Decrypt | CO<br>- AES Keys: W,E |
| SHS | Generation a SHA-1 or SHA- | Approved | Message | Message Digest | SFI-SHS | CO |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| | 2 message digest | | | | | |
| SHA3 | Generation a SHA-3 message digest | Approved | Message | Message Digest | SFI-SHA3 | CO |
| SHA3-SHAKE | Generation a SHA-3 Extendable Output Function (XOF) message digest | Approved | Message | Message Digest | SFI-SHAKE | CO |
| ECDSA KeyGen | Generate Public/Private Asymmetric key pairs. | Approved | Generate Key command with random number and key size | Key (public and private) and return status of OK or error condition | SFI-ECDSA-KeyGen SFI-CKG-AsymKeyGen | CO<br>- ECDSA Private Key: G,R<br>- ECDSA Public Key: G,R |
| ECDSA KeyVer | Verify Public/Private Asymmetric key pairs. | Approved | ECDSA public/private key pairs | Return status of OK or error condition | SFI-ECDSA-KeyVer | CO<br>- ECDSA Private Key: W,E<br>- ECDSA Public Key: W,E |
| ECDSA SigGen | Perform digital signature generation | Approved | Signature command with ECDSA private key and message | Digital Signature and return status of OK or error condition | SFI-ECDSA-SigGen | CO<br>- ECDSA Private Key: W,E |
| ECDSA SigVer | Perform digital signature verification | Approved | Verify command with ECDSA public key and signature | Valid Signature Indicator (True/False) and return status of OK or error condition | SFI-ECDSA-SigVer | CO<br>- ECDSA Public Key: W,E |
| EdDSA KeyGen | Generate Public/Private | Approved | Generate Key command with random | Key (public and private) and return | SFI-EdDSA-KeyGen SFI-CKG- | CO<br>- EdDSA Private Key: |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| | Asymmetric key pairs. | | number and key size | status of OK or error condition | AsymKeyGen | G,R<br>- EdDSA Public Key: G,R |
| EdDSA KeyVer | Verify Public/Private Asymmetric key pairs. | Approved | EdDSA public/private key pairs | Return status of OK or error condition | SFI-EdDSA-KeyVer | CO<br>- EdDSA Private Key: W,E<br>- EdDSA Public Key: W,E |
| EdDSA SigGen | Perform digital signature generation | Approved | Signature command with EdDSA private key and message | Digital Signature and return status of OK or error condition | SFI-EdDSA-SigGen | CO<br>- EdDSA Private Key: W,E |
| EdDSA SigVer | Perform digital signature verification | Approved | Verify command with EdDSA public key and signature | Valid Signature Indicator (True/False) and return status of OK or error condition | SFI-EdDSA-SigVer | CO<br>- EdDSA Public Key: W,E |
| RSA KeyGen | Generate Public/Private RSA Asymmetric key pairs | Approved | Generate Key command with random number and key size | Key (public and private) and return status of OK or error condition | SFI-RSA-KeyGen<br>SFI-CKG-AsymKeyGen | CO<br>- RSA Private Key: G,R<br>- RSA Public Key: G,R |
| RSA SigGen | Perform digital signature generation | Approved | Signature command with RSA private key and message | Digital Signature and return status of OK or error condition | SFI-RSA-SigGen<br>SFI-RSA-PSS-SigGen | CO<br>- RSA Private Key: W,E |
| RSA SigVer | Perform digital signature verification | Approved | Verify command with RSA public key and signature | Valid Signature Indicator (True/False) and return status of OK or error condition | SFI-RSA-SigVer<br>SFI-RSA-PSS-SigVer | CO<br>- RSA Public Key: W,E |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| CMAC MACGen | Generate a keyed-hash message authentication code with AES-CMAC | Approved | AES key, message | Keyed hash with return status of OK or error condition | SFI-AES-CMAC | CO<br>- AES Keys: W,E |
| GMAC MACGen | Generate a keyed-hash message authentication code with AES-GCM | Approved | AES key, message | Keyed hash with return status of OK or error condition | SFI-AES-GMAC | CO<br>- AES Keys: W,E |
| HMAC MACGen | Generate a keyed-hash message authentication code | Approved | HMAC key, message | Keyed hash with return status of OK or error condition | SFI-SHS<br>SFI-SHA3<br>SFI-HMAC | CO<br>- HMAC Keys: W,E |
| KDF-HMAC | Extract input key material and expand into additional keys | Approved | Pseudorandom key material | Key material and return status of OK or error condition | SFI-SHS<br>SFI-SHA3<br>SFI-HMAC<br>SFI-HMAC-KDF | CO<br>- KBKDF Pseudorandom Keys: W,E<br>- KBKDF Output Key: G,R |
| AES-CTR-DRBG Gen | Generate Psuedo random numbers | Approved | Generate command | Random number with status of OK or error condition | SFI-DRBG-Generate | CO<br>- DRBG Entropy Input: W<br>- Nonce Values: W<br>- DRBG Values: G<br>- DRBG V: R<br>- DRBG Key: R<br>- DRBG Reseed Counter: G |
| AES-CTR-DRBG Reseed | Re-seed the DRBG | Approved | Reseed command with input of entropy | Status of OK or error condition | SFI-DRBG-ReSeed | CO<br>- DRBG Entropy Input: W<br>- Nonce Values: W<br>- DRBG |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| | | | | | | Reseed Counter: G |
| KAS-FFC KeyGen | Generate a local asymmetric key for an approved group to be used as part of the key-agreement protocol (KAS-FFC Key Generation) | Approved | Approved Security Group | Local Public/Private key-pair | SFI-CKG-AsymKeyGen KAS-FFC SFI-KAS-FFC | CO<br>- FFC Private Key: G,R,E<br>- FFC Public Key: G,R,E |
| KAS-FFC KeyExchange | Generate a shared secret key based on local asymmetric key and remote public key parameters to be used between two or more parties based on the key-agreement protocol | Approved | Generate command with random number, Local and remote Asymmetric keys | Shared secret symmetric key and return status of OK or error condition | SFI-CKG-AsymKeyGen KAS-FFC SFI-KAS-FFC | CO<br>- FFC Private Key: W,E<br>- FFC Public Key: W,E<br>- FFC Shared Secret: G,R<br>- FFC Derived KeyMaterial: G,R |
| KAS-ECC KeyGen | Generate a secret key to be used between two or more parties based on the key-agreement protocol, ECC Key Generation | Approved | Approved ECC curves | Local Public/Private key-pair | SFI-CKG-AsymKeyGen KAS-ECC SFI-KAS-ECC | CO<br>- ECC Private Key: G,R,E<br>- ECC Public Key: G,R,E |
| KAS-ECC KeyExchange | Generate a secret key to be used between two | Approved | Generate command with random number, | Shared secret symmetric key and | SFI-CKG-AsymKeyGen KAS-ECC | CO<br>- ECC Private Key: W,E<br>- ECC Public |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| | or more parties based on the key-agreement protocol | | Local and remote Asymmetric keys | return status of OK or error condition | SFI-KAS-ECC | Key: W,E<br>- ECC Shared Secret: G,R<br>- ECC Derived KeyMaterial: G,R |
| Integrity Verify | Perform integrity test and return the status | None | Integrity Command | Return status of OK or error condition | None | CO |
| Self-tests | Initiate self-tests (Software Integrity Check, DRBG KAT, SHA2-256 KAT, HMAC-SHA2-256 KAT) | Approved | Command with list of CASTs to be performed | Return Code - OK or error condition | None | CO |
| Show Status | Return the status of the module state, exit codes, kernel log (dmesg) | None | Status Command | Return Code - OK or error condition | None | CO |
| Show Version | Return module version information | None | Version Command | SW Version | None | CO |
| Zeroize | Destroy/Zeroize all SSPs | Approved | Zeroize command | Return status of OK or error condition | None | CO<br>- FFC Shared Secret: Z<br>- FFC Private Key: Z<br>- FFC Public Key: Z<br>- ECC Private Key: Z<br>- ECC Shared Secret: Z<br>- ECC Public Key: Z<br>- DRBG Entropy Input: Z |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| | | | | | | - Nonce Values: Z<br>- DRBG Values: Z<br>- RSA Private Key: Z<br>- RSA Public Key: Z<br>- ECDSA Private Key: Z<br>- ECDSA Public Key: Z<br>- EdDSA Private Key: Z<br>- EdDSA Public Key: Z<br>- AES Keys: Z<br>- HMAC Keys: Z<br>- KBKDF Pseudorandom Keys: Z<br>- DRBG V: Z<br>- DRBG Reseed Counter: Z<br>- DRBG Key: Z<br>- FFC Derived KeyMaterial: Z<br>- ECC Derived KeyMaterial: Z<br>- DSA Private Key: Z<br>- DSA Public Key: Z<br>- HMAC SHA2-256: Z |
| DSA KeyGen | Generate Public/Private DSA Asymmetric key pairs | Approved | Generate Key command with random number and key size | Key (public and private) and return status of OK or error condition | SFI-DSA-KeyGen<br>SFI-DSA-PQGGen<br>SFI-DSA-PQGVer<br>SFI-CKG- | CO<br>- DSA Private Key: G,R<br>- DSA Public Key: G,R |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| | | | | | AsymKeyGen | |
| DSA SigGen | Perform digital signature generation | Approved | Signature command with DSA private key and message | Digital Signature and return status of OK or error condition | SFI-DSA-SigGen | CO<br>- DSA Private Key: W,E |
| DSA SigVer | Perform digital signature verification | Approved | Verify command with DSA public key and signature | Valid Signature Indicator (True/False) and return status of OK or error condition | SFI-DSA-SigVer | CO<br>- DSA Public Key: W,E |

Table 13: Approved Services

## 4.4 Non-Approved Services

All approved services implemented by the Module are listed in the table below:

| Name | Description | Algorithms | Role |
|---|---|---|---|
| NC Digital Signature | DSA (FIPS 186-4) | DSA (NC) | CO |
| NC Key Agreement DH | Key agreement; key establishment methodology provides less than 112 bits of encryption strength | DH (NC) | CO |
| NC Key Agreement ECC_CDH | Key agreement; key establishment methodology provides less than 112 bits of encryption strength | ECC CDH (NC) | CO |
| NC Key Agreement EDDH | Generate a secret key to be used between 2 or more parties based on the key-agreement protocol | EDDH (NC) | CO |
| NC Keyed Message Digest GCM/GMAC | AES-GCM or AES-GMAC encryption and decryption for 256-bit state table implementations | AES GCM 256-bit (NC)<br>AES GMAC 256-bit (NC) | CO |
| NC Keyed Message Digest HMAC | HMAC generation with key size less than 112 bits | HMAC (NC) | CO |
| NC Keyed Message Digest HMAC MD5 | HMAC generation with MD5 | HMAC-MD5 (NC) | CO |
| NC Key Wrapping | Key wrapping; key establishment methodology | RSA (NC)<br>RSA (Key Wrapping) (NC) | CO |

| Name | Description | Algorithms | Role |
|---|---|---|---|
| NC OAEP Encryption | PKCS#1 v2.1 RSAES-OAEP encryption/decryption | RSA-OAEP (NC) | CO |
| NC Message Digest | Generate an MD2, MD4, or MD5 message digest | MD2, MD4, MD5 (NC) | CO |
| NC Random Number Generation | FIPS 186-2 Random Number Generation | RNG (NC) | CO |
| NC Symmetric Encryption/Decryption AES | Compute the cipher for encryption and decryption | AES-EAX (NC) AES XCBC (NC) | CO |
| NC Symmetric Encryption/Decryption DES | Compute the cipher for encryption and decryption | DES (NC) Triple-DES (NC) | CO |

Table 14: Non-Approved Services

## 4.5 External Software/Firmware Loaded

NOTE: There is no External Software/Firmware loaded

# 5 Software/Firmware Security

## 5.1 Integrity Techniques

The Module is composed of the following software component(s):

- libmss.so: executable - binary - The shared library that contains the cryptographic module code, data, and constants
- libmss.so.sig: - data - The integrity check signature file that contains an HMAC-SHA2-256 of the cryptographic module.
- mssp.bin: - data - The POST status file that contains persistent results of the first run of POST

The software components are protected with the HMAC-SHA2-256 authentication technique.

The HMAC for the Shared Library is calculated during the manufacturing (build) process of the Shared Library. This HMAC value is stored either within the resulting “libmss.so” shared or as a separate “libmss.so.sig” file dependent upon the development tools and target operating system constraints.

During the load of the shared object, the integrity check of the library code and constants occurs in the module startup function. It verifies the integrity of the shared library by executing the HMAC-SHA2-256 fingerprint algorithm on the libmss.so file and comparing the result with the signature. This integrity check is performed as part of the function FIPS_powerupSelfTest(). This function is called automatically by the host O/S upon loading the shared object into memory as shown below.

```
#ifdef __ENABLE_MOCANA_FIPS_LIB_CONSTRUCTOR__
static void FIPS_constructor() __attribute__((constructor));
void FIPS_constructor()
{
        FIPS_powerupSelfTest();
}
#endif
```

## 5.2 Initiate on Demand

The operator can initiate the integrity test on demand by reloading the Module or by calling the API function: FIPS_StartupSelftestIntegrity(void).

# 6 Operational Environment

## 6.1 Operational Environment Type and Requirements

**Type of Operational Environment**: Modifiable

The Module has a modifiable operational environment under the FIPS 140-3 definitions. The tested operational environments are listed in Section 2.2 Tested and Vendor Affirmed Module Version and Identification above. In addition, DigiCert claims that the Module can be ported on the Vendor Affirmed Operational Environment(s); no statement is made regarding the correct operation of the Module on the Vendor Affirmed Operational Environments.

For each process, session management through the operating system provides role association, process and session isolation, and memory protection. Each process has control over its own data while the operating system prevents uncontrolled access to the data and other processes. The module does not support concurrent operators.

A software handle between the consuming application (i.e., entity) and the cryptographic module's key structure provides the key to entity association in the module. The software handle is specific to the consuming application and is contained within its own process (e.g., handles are not shared between multiple consuming applications).

## 6.2 Configuration Settings and Restrictions

No operational environment restrictions are required for the operation of the Module. The operating system of the host device prevents unauthorized access to SSPs during execution of the Module. The Module allows access to SSPs only through specific APIs.

# 7 Physical Security

The module is a software module at level 1.

# 8 Non-Invasive Security

## 8.1 Mitigation Techniques

The Module does not implement any mitigation method against non-invasive attack.

# 9 Sensitive Security Parameters Management

## 9.1 Storage Areas

| Storage Area Name | Description | Persistence Type |
|---|---|---|
| Memory (S1) | Only stored in volatile memory (RAM) | Dynamic |

Table 15: Storage Areas

## 9.2 SSP Input-Output Methods

| Name | From | To | Format Type | Distribution Type | Entry Type | SFI or Algorithm |
|---|---|---|---|---|---|---|
| Input in plaintext (IO2) | Memory (S1) | Memory (S1) | Plaintext | Manual | Electronic | |
| Output in plaintext (IO3) | Memory (S1) | Memory (S1) | Plaintext | Manual | Electronic | |

Table 16: SSP Input-Output Methods

## 9.3 SSP Zeroization Methods

| Zeroization Method | Description | Rationale | Operator Initiation |
|---|---|---|---|
| Z1 | Zeroized by the zeroization service by overwriting with a fixed pattern of zeros. | The application is responsible for calling the appropriate destruction functions from the API. These functions overwrite the memory with zeros and de-allocate the memory. In case of abnormal termination, the Linux kernel overwrites the keys in physical memory before the physical memory is allocated to another process. | "Zeroize" service. |

Table 17: SSP Zeroization Methods

## 9.4 SSPs

All usage of these SSPs by the Module are described in the services detailed in Section 4.3

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| FFC Shared Secret | Shared secret computation established as part of FFC key agreement scheme | 2048 to 4096 bits - 112-200 bits | Private - CSP | | SFI-KAS-FFC | SFI-KAS-FFC |
| FFC Private Key | Used to derive the secret key during the FFC key agreement protocol | MODP-2048, MODP-3072, MODP-4096, MODP-6144, and MODP-8192 - 112-200 bits | Private - CSP | SFI-CKG-AsymKeyGen KAS-FFC | | SFI-KAS-FFC |
| FFC Public Key | Used to derive the secret key during FFC key agreement protocol | MODP-2048, MODP-3072, MODP-4096, MODP-6144, and MODP-8192 - 112-200 bits | Public - PSP | SFI-CKG-AsymKeyGen KAS-FFC | | SFI-KAS-FFC |
| ECC Private Key | Used to derive the secret session key during ECC key agreement protocol | P-224, P-256, P-384, P-521 - 112-256 bits | Private - CSP | SFI-CKG-AsymKeyGen KAS-ECC | | SFI-KAS-ECC |
| ECC Shared Secret | Shared secret computation | P-224, P-256, P-384, P-521 - 112-256 bits | Shared Secret - CSP | | SFI-KAS-ECC | SFI-KAS-ECC |
| ECC Public Key | Used to derive the secret session key during ECC key agreement protocol | P-224, P-256, P-384, P-521 - 112-256 bits | Public - PSP | SFI-CKG-AsymKeyGen KAS-ECC | | SFI-KAS-ECC |
| DRBG Entropy Input | Used to seed the DRBG for key generation | 1282^24 bits - 128 s 256 | Entropy - CSP | | | SFI-DRBG-ReSeed |
| Nonce Values | Used to seed the DRBG for key generation | 02^24 bits - N/A | Entropy - CSP | | | SFI-DRBG-ReSeed |
| DRBG Values | Random number | 256 bits - 256 bits | RBG - CSP | SFI-DRBG-Generate | | SFI-DSA-KeyGen SFI-DSA-SigGen |

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| | | | | | | SFI-DSA-SigVer<br>SFI-ECDSA-KeyGen<br>SFI-ECDSA-SigGen<br>SFI-ECDSA-SigVer<br>SFI-EdDSA-KeyGen<br>SFI-EdDSA-SigGen<br>SFI-EdDSA-SigVer<br>SFI-RSA-KeyGen<br>SFI-RSA-SigGen<br>SFI-RSA-PSS-SigGen<br>SFI-RSA-SigVer<br>SFI-RSA-PSS-SigVer<br>SFI-CKG-AsymKeyGen<br>SFI-CKG-AsymKeyGen KAS-ECC<br>SFI-CKG-AsymKeyGen KAS-FFC<br>SFI-KAS-ECC<br>SFI-KAS-FFC |
| RSA Private Key | Used to create RSA digital signatures | 2048 to 4096 bits - 112 to 148 bits | Private - CSP | SFI-RSA-KeyGen | | SFI-RSA-KeyGen |

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| RSA Public Key | Used to verify RSA signatures | 2048 to 4096 bits - 112 to 148 bits | Public - PSP | SFI-RSA-KeyGen | | SFI-RSA-KeyGen |
| ECDSA Private Key | Used to create ECDSA digital signatures | P-224, P-256, P-384, P-521 - 112 to 256 bits | Private - CSP | SFI-ECDSA-KeyGen | | SFI-ECDSA-KeyGen |
| ECDSA Public Key | Used to verify ECDSA signatures | P-224, P-256, P-384, P-521 - 112 to 256 bits | Public - PSP | SFI-ECDSA-KeyGen | | SFI-ECDSA-KeyGen |
| EdDSA Private Key | Used to create EdDSA digital signatures | ED-25519, ED-448 - 128 to 224 bits | Private - CSP | SFI-EdDSA-KeyGen | | SFI-EdDSA-KeyGen |
| EdDSA Public Key | Used to verify EdDSA signatures | ED-25519, ED-448 - 128 to 224 bits | Public - PSP | SFI-EdDSA-KeyGen | | SFI-EdDSA-KeyGen |
| AES Keys | Used during AES Encryption, Decryption, CMAC, and GMAC operations | 128, 192, 256 bits - 128 to 256 bits | Symmetric - CSP | | | SFI-AES-UnAuth-Encrypt<br>SFI-AES-UnAuth-Decrypt<br>SFI-AES-CCM-Encrypt<br>SFI-AES-CCM-Decrypt<br>SFI-AES-GCM-Encrypt<br>SFI-AES-GCM-Decrypt<br>SFI-AES-CMAC<br>SFI-AES-GMAC |
| HMAC Keys | Used during HMAC-SHA operations | 112-65536 - 128 to 256 bits | Symmetric - CSP | | | SFI-HMAC |
| KBKDF Pseudorandom Keys | Used in deriving other keys per SP800-108 | 112-4096 - 128 to 256 bits | Symmetric - CSP | | | SFI-HMAC-KDF |
| DRBG V | Internal State Value of V | 128 to 256 bits - 128 s 256 bits | Internal State Critical Value - CSP | SFI-DRBG-Generate | | SFI-DRBG-Generate |

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| DRBG Reseed Counter | Internal State Value of Reseed Counter | 64 bits - N/A | Internal State Critical Value - CSP | SFI-DRBG-Generate SFI-DRBG-ReSeed | | SFI-DRBG-Generate SFI-DRBG-ReSeed |
| DRBG Key | Internal State Value of Key | 128 to 256 bits - 128 s 256 bits | Internal State Critical Value - CSP | SFI-DRBG-ReSeed | | SFI-DRBG-Generate |
| KBKDF Output Key | Output key derived per SP800-108 | 128 to 256 bits - 128 to 256 bits | Symmetric - CSP | SFI-HMAC-KDF | | SFI-HMAC-KDF |
| FFC Derived KeyMaterial | Key Material derived from shared secret part of FFC key agreement scheme | 160 bits to 512 bits - 160 to 512 bits | Private - CSP | | SFI-KAS-FFC | SFI-KAS-FFC |
| ECC Derived KeyMaterial | Key Material derived from shared secret part of ECC key agreement scheme | 160 bits to 512 bits - 160 to 512 bits | Private - CSP | | SFI-KAS-ECC | SFI-KAS-ECC |
| DSA Private Key | Used to create DSA digital signatures | 2048 bits to 3072 bits - 112 to 128 bits | Private - CSP | SFI-DSA-KeyGen | | SFI-DSA-KeyGen |
| DSA Public Key | Used to verify DSA signatures | 2048 bits to 3072 bits - 112 to 128 bits | Public - PSP | SFI-DSA-KeyGen | | SFI-DSA-KeyGen |
| HMAC SHA2-256 | Used during the integrity check | 256 bits - 256 bits | Symmetric - Neither | | | SFI-HMAC |

Table 18: SSP Table 1

| Name | Input - Output | Storage | Storage Duration | Zeroization | Related SSPs |
|---|---|---|---|---|---|
| FFC Shared Secret | Output in plaintext (IO3) | Memory (S1):Plaintext | Call lifetime | Z1 | FFC Private Key:Derived From FFC Public Key:Derived From FFC Derived KeyMaterial:Used to derive |

| Name | Input - Output | Storage | Storage Duration | Zeroization | Related SSPs |
|---|---|---|---|---|---|
| FFC Private Key | Input in plaintext (IO2)<br>Output in plaintext (IO3) | Memory (S1):Plaintext | Call lifetime | Z1 | FFC Public Key:Paired With<br>FFC Shared Secret:Used to derive |
| FFC Public Key | Input in plaintext (IO2)<br>Output in plaintext (IO3) | Memory (S1):Plaintext | Call lifetime | Z1 | FFC Private Key:Paired With<br>FFC Shared Secret:Used to derive |
| ECC Private Key | Input in plaintext (IO2)<br>Output in plaintext (IO3) | Memory (S1):Plaintext | Call lifetime | Z1 | ECC Public Key:Paired With<br>ECC Shared Secret:Used to derive |
| ECC Shared Secret | Output in plaintext (IO3) | Memory (S1):Plaintext | Call lifetime | Z1 | ECC Private Key:Derived From<br>ECC Public Key:Derived From<br>ECC Derived KeyMaterial:Used to derive |
| ECC Public Key | Input in plaintext (IO2)<br>Output in plaintext (IO3) | Memory (S1):Plaintext | Call lifetime | Z1 | ECC Private Key:Paired With<br>ECC Shared Secret:Used to derive |
| DRBG Entropy Input | Input in plaintext (IO2) | Memory (S1):Plaintext | Call lifetime | Z1 | Nonce Values:Used With<br>DRBG V:Derived From<br>DRBG Key:Derived From |
| Nonce Values | Input in plaintext (IO2) | Memory (S1):Plaintext | Call lifetime | Z1 | DRBG Entropy Input:Used With |
| DRBG Values | Output in plaintext (IO3) | Memory (S1):Plaintext | Call lifetime | Z1 | DRBG Entropy Input:Derived From<br>Nonce Values:Derived From |
| RSA Private Key | Input in plaintext (IO2)<br>Output in plaintext (IO3) | Memory (S1):Plaintext | Call lifetime | Z1 | RSA Public Key:Paired With |

| Name | Input - Output | Storage | Storage Duration | Zeroization | Related SSPs |
|---|---|---|---|---|---|
| RSA Public Key | Input in plaintext (IO2)<br>Output in plaintext (IO3) | Memory (S1):Plaintext | Call lifetime | Z1 | RSA Private Key:Paired With |
| ECDSA Private Key | Input in plaintext (IO2)<br>Output in plaintext (IO3) | Memory (S1):Plaintext | Call lifetime | Z1 | ECDSA Public Key:Paired With |
| ECDSA Public Key | Input in plaintext (IO2)<br>Output in plaintext (IO3) | Memory (S1):Plaintext | Call lifetime | Z1 | ECDSA Private Key:Paired With |
| EdDSA Private Key | Input in plaintext (IO2)<br>Output in plaintext (IO3) | Memory (S1):Plaintext | Call lifetime | Z1 | EdDSA Public Key:Paired With |
| EdDSA Public Key | Input in plaintext (IO2)<br>Output in plaintext (IO3) | Memory (S1):Plaintext | Call lifetime | Z1 | EdDSA Private Key:Paired With |
| AES Keys | Input in plaintext (IO2) | Memory (S1):Plaintext | Call lifetime | Z1 | |
| HMAC Keys | Input in plaintext (IO2) | Memory (S1):Plaintext | Call lifetime | Z1 | |
| KBKDF Pseudorandom Keys | Input in plaintext (IO2) | Memory (S1):Plaintext | Call lifetime | Z1 | KBKDF Output Key:Used to derive |
| DRBG V | | Memory (S1):Plaintext | Call lifetime (module up time for internal DRBG) | Z1 | DRBG Entropy Input:Used With<br>Nonce Values:Used With<br>DRBG Key:Used With<br>DRBG Reseed Counter:Used With |

| Name | Input - Output | Storage | Storage Duration | Zeroization | Related SSPs |
|---|---|---|---|---|---|
| DRBG Reseed Counter | | Memory (S1):Plaintext | Call lifetime (module up time for internal DRBG) | Z1 | DRBG Entropy Input:Used With<br>Nonce Values:Used With<br>DRBG V:Used With<br>DRBG Key:Used With |
| DRBG Key | | Memory (S1):Plaintext | Call lifetime (module up time for internal DRBG) | Z1 | DRBG Entropy Input:Used With<br>Nonce Values:Used With<br>DRBG V:Used With<br>DRBG Reseed Counter:Used With |
| KBKDF Output Key | Output in plaintext (IO3) | Memory (S1):Plaintext | Call lifetime | Z1 | KBKDF Pseudorandom Keys:Derived From |
| FFC Derived KeyMaterial | Output in plaintext (IO3) | Memory (S1):Plaintext | Call lifetime | Z1 | FFC Shared Secret:Derived From |
| ECC Derived KeyMaterial | Output in plaintext (IO3) | Memory (S1):Plaintext | Call lifetime | Z1 | ECC Shared Secret:Derived From |
| DSA Private Key | Input in plaintext (IO2)<br>Output in plaintext (IO3) | Memory (S1):Plaintext | Call lifetime | Z1 | DSA Public Key:Paired With |
| DSA Public Key | Input in plaintext (IO2)<br>Output in plaintext (IO3) | Memory (S1):Plaintext | Call lifetime | Z1 | DSA Private Key:Paired With |
| HMAC SHA2-256 | Input in plaintext (IO2) | Memory (S1):Plaintext | Call lifetime | Z1 | |

Table 19: SSP Table 2

# 10 Self-Tests

## 10.1 Pre-Operational Self-Tests

The Module performs self-tests to ensure the proper operation of the Module. Per FIPS 140-3 these are categorized as either pre-operational self-tests or conditional self-tests.

Pre-operational self–tests are available on demand by power cycling the Module or reloading the Module into memory. The Module is available to perform services only after successfully completing the pre-operational self-tests.

The Module performs the following pre-operational self-tests in table below

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details |
|---|---|---|---|---|---|
| HMAC-SHA2-256 (A4653) | Key Length = 256 | KAT | SW/FW Integrity | Crypto module enabled upon return status of OK. ES1 error status upon KAT failure | HMAC-SHA2-256 integrity check is performed. Result is compared against the hash value in the signature .sig file |

Table 20: Pre-Operational Self-Tests

## 10.2 Conditional Self-Tests

The Module performs the conditional self-tests listed in the table below

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| AES-CBC-Enc (A4653) | Key length = 256 bits | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Encryption | Before first use of AES-CBC |
| AES-CBC-Dec (A4653) | Key length = 256 bits | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Decryption | Before first use of AES-CBC |

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| AES-CCM-Enc (A4653) | Key length = 128 bits | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Encryption | Before first use of AES-CCM |
| AES-CCM-Dec (A4653) | Key length = 128 bits | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Decryption | Before first use of AES-CCM |
| AES-CFB128-Enc (A4653) | Key length = 256 bits | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Encryption | Before first use of AES-CFB128 |
| AES-CFB128-Dec (A4653) | Key length = 256 bits | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Decryption | Before first use of AES-CFB128 |
| AES-CTR-Enc (A4653) | Key length = 256 bits | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Encryption | Before first use of AES-CTR |
| AES-CTR-Dec (A4653) | Key length = 256 bits | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Decryption | Before first use of AES-CTR |
| AES-ECB-Enc (A4653) | Key length = 256 bits | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Encryption | Before first use of AES-ECB |
| AES-ECB-Dec (A4653) | Key length = 256 bits | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Decryption | Before first use of AES-ECB |

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| AES-OFB-Enc (A4653) | Key length = 256 bits | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Encryption | Before first use of AES-OFB |
| AES-OFB-Dec (A4653) | Key length = 256 bits | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Decryption | Before first use of AES-OFB |
| AES-XTS-Enc Testing Revision 2.0 (A4653) | Key length = 128 bits | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Encryption | Before first use of AES-XTS |
| AES-XTS-Dec Testing Revision 2.0 (A4653) | Key length = 128 bits | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Decryption | Before first use of AES-XTS |
| AES-CMAC-Gen (A4653) | Key length = 128 bits | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Generate | Before first use of AES-CMAC |
| AES-GCM-Enc (A4654) | Key length = 128 bits for 4k modules | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Encryption | Before first use of AES-GCM |
| AES-GCM-Dec (A4654) | Key length = 128 bits for 4k modules | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Decryption | Before first use of AES-GCM |
| AES-GCM-Enc (A4655) | Key length = 128 bits for 64k modules | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Encryption | Before first use of AES-GCM |

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| AES-GCM-Dec (A4655) | Key length = 128 bits for 64k modules | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Decryption | Before first use of AES-GCM |
| AES-GMAC-Gen (A4654) | Key length = 128 bits for 4k modules | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Generate | Before first use of AES-GMAC |
| AES-GMAC-Gen (A4655) | Key length = 128 bits for 64k modules | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Generate | Before first use of AES-GMAC |
| KAS-FFC Sp800-56Ar3 (A4653) | Group-14 2k bit prime with SHA-224 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Verify computation of shared secret Z in dhEphem scheme | Before first use of algorithm: KAS-FFC |
| KAS-ECC Sp800-56Ar3 (A4653) | P224 curve curve with SHA-224 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Verify computation of shared secret Z in Ephemeral Unified scheme | Before first use of algorithm: KAS-ECC |
| Counter DRBG (A4653) | 256 Bits with and without df | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure. ES3 error status upon critical function test failure | Instantiation, Generation, Reseed | Initialization |
| ECDSA SigGen (FIPS186-4) (A4653) | P-224 curve with SHA-224 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Sign | Before first use of algorithm: ECDSA |

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| ECDSA SigVer (FIPS186-4) (A4653) | P-224 curve with SHA-224 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Verify | Before first use of algorithm: ECDSA |
| ECDSA KeyGen (FIPS186-4) (A4653) | P-224, P-256, P-384, P-521 | PCT | PCT | Crypto module enabled upon return status of OK. ES3 error status upon PCT failure | Sign and Verify signature with generated Key | For each Key pair generation |
| EDDSA SigGen (A4653) | Ed25519 with SHA-512 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Sign | Before first use of algorithm: EDDSA |
| EDDSA SigVer (A4653) | Ed25519 with SHA-512 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Verify | Before first use of algorithm: EDDSA |
| EDDSA KeyGen (A4653) | Ed25519 with SHA-512 | PCT | PCT | Crypto module enabled upon return status of OK. ES3 error status upon PCT failure | Sign and Verify signature with generated Key | For each Key pair generation |
| HMAC-SHA-1 (A4653) | SHA-1 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash-based Authentication | Before first use of algorithm: HMAC-SHA-1 |
| HMAC-SHA2-224 (A4653) | SHA2-224 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash-based Authentication | Before first use of algorithm: HMAC-SHA2-224 |
| HMAC-SHA2-256 (A4653) | SHA2-256 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash-based Authentication | Before first use of algorithm: HMAC-SHA2-256. Before pre- |

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| | | | | | | operational integrity test. |
| HMAC-SHA2-384 (A4653) | SHA2-384 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash-based Authentication | Before first use of algorithm: HMAC-SHA2-384 |
| HMAC-SHA2-512 (A4653) | SHA2-512 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash-based Authentication | Before first use of algorithm: HMAC-SHA2-512 |
| HMAC-SHA3-224 (A4653) | SHA3-224 Feedback Mode | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash-based Authentication | Before first use of algorithm: HMAC-SHA3-224 |
| HMAC-SHA3-256 (A4653) | SHA3-256 Feedback Mode | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash-based Authentication | Before first use of algorithm: HMAC-SHA3-256 |
| HMAC-SHA3-384 (A4653) | SHA3-384 Feedback Mode | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash-based Authentication | Before first use of algorithm: HMAC-SHA3-384 |
| HMAC-SHA3-512 (A4653) | SHA3-512 Feedback Mode | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash-based Authentication | Before first use of algorithm: HMAC-SHA3-512 |
| KDF SP800-108-SHA2-224 | HMAC-SHA2-224 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Key Derivation | Before first use of algorithm: KDF HMAC-SHA2-224 |

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| KDF SP800-108-SHA2-256 | HMAC-SHA2-256 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Key Derivation | Before first use of algorithm: KDF HMAC-SHA2-256 |
| KDF SP800-108-SHA2-384 | HMAC-SHA2-384 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Key Derivation | Before first use of algorithm: KDF HMAC-SHA2-384 |
| KDF SP800-108-SHA2-512 | HMAC-SHA2-512 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Key Derivation | Before first use of algorithm: KDF HMAC-SHA2-512 |
| KDF SP800-108-SHA3-224 | HMAC-SHA3-224 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Key Derivation | Before first use of algorithm: KDF HMAC-SHA3-224 |
| KDF SP800-108-SHA3-256 | HMAC-SHA3-256 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Key Derivation | Before first use of algorithm: KDF HMAC-SHA3-256 |
| KDF SP800-108-SHA3-384 | HMAC-SHA3-384 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Key Derivation | Before first use of algorithm: KDF HMAC-SHA3-384 |
| KDF SP800-108-SHA3-512 | HMAC-SHA3-512 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Key Derivation | Before first use of algorithm: KDF HMAC-SHA3-512 |
| SHA-1 (A4653) | SHA-1 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash | Before first use of algorithm: SHA-1 |

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| SHA2-224 (A4653) | SHA2-224 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash | Before first use of algorithm: SHA2-224 |
| SHA2-256 (A4653) | SHA2-256 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash | Before first use of algorithm: SHA2-256 |
| SHA2-384 (A4653) | SHA2-384 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash | Before first use of algorithm: SHA2-384 |
| SHA2-512 (A4653) | SHA2-512 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash | Before first use of algorithm: SHA2-512 |
| SHA3-224 (A4653) | SHA3-224 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash | Before first use of algorithm: SHA3-224 |
| SHA3-256 (A4653) | SHA3-256 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash | Before first use of algorithm: SHA3-256 |
| SHA3-384 (A4653) | SHA3-384 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash | Before first use of algorithm: SHA3-384 |
| SHA3-512 (A4653) | SHA3-512 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash | Before first use of algorithm: SHA3-512 |

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| SHAKE-128 (A4653) | SHAKE-128 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash | Before first use of algorithm: SHAKE-128 |
| SHAKE-256 (A4653) | SHAKE-256 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Hash | Before first use of algorithm: SHAKE-256 |
| RSA SigGen (FIPS186-4) (A4653) | 2048 bit | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Sign | Before first use of algorithm: RSA |
| RSA SigVer (FIPS186-4) (A4653) | 2048 bit | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Verify | Before first use of algorithm: RSA |
| RSA KeyGen (FIPS186-4) (A4653) | 2048, 3072, and 4096 bit | PCT | PCT | Crypto module enabled upon return status of OK. ES3 error status upon KAT failure | Sign and Verify signature with generated Key | For each key pair generation |
| KDF SP800-108-SHA-1 | HMAC-SHA-1 | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Key Derivation | Before first use of algorithm: KDF HMAC-SHA-1 |
| DSA KeyGen (FIPS186-4) (A4653) | 2048, 3072 bit | PCT | CAST | Crypto module enabled upon return status of OK. ES3 error status upon KAT failure | Sign and Verify signature with generated Key | For each Key pair generation |
| DSA SigGen (FIPS186-4) (A4653) | 2048, 3072 bit DSA | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Sign | Before first use of algorithm: DSA |

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| DSA SigVer (FIPS186-4) (A4653) | 1024, 2048, and 3072 bit DSA | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Verify | Before first use of algorithm: DSA |
| AES-GMAC-Ver (A4654) | Key length = 128 bits for 4k modules | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Verify | Before first use of AES-GMAC |
| AES-GMAC-Ver (A4655) | Key length = 128 bits for 4k modules | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Verify | Before first use of AES-GMAC |
| AES-CMAC-Ver (A4653) | Key length = 128 bits | KAT | CAST | Crypto module enabled upon return status of OK. ES2 error status upon KAT failure | Verify | Before first use of AES-CMAC |
| AES-XTS Key Comparison | Key length = 128 bits | AES-XTS key comparison | Critical Function | Crypto module enabled upon return status of OK. ES3 error status upon KAT failure | IG C.I key comparison test is performed to verify Key1 != Key2. | Before first use of AES-XTS |

Table 21: Conditional Self-Tests

The intended usage of the generation of DSA, ECDSA, EdDSA, or RSA key pairs is not known at the time when the key pair is generated. A pair-wise consistency test (PCT) is performed, providing assurance for the generated key pair. These tests are conducted for the testing of signature generation and signature verification. Upon failure of a test, the module transitions into an error state, as shown in Section 10.4.

## 10.3 Periodic Self-Test Information

| Algorithm or Test | Test Method | Test Type | Period | Periodic Method |
|---|---|---|---|---|
| HMAC-SHA2-256 (A4653) | KAT | SW/FW Integrity | On demand | By power cycling or reloading the Module into memory |

Table 22: Pre-Operational Periodic Information

| Algorithm or Test | Test Method | Test Type | Period | Periodic Method |
|---|---|---|---|---|
| AES-CBC-Enc (A4653) | KAT | CAST | On demand | Manually |
| AES-CBC-Dec (A4653) | KAT | CAST | On demand | Manually |
| AES-CCM-Enc (A4653) | KAT | CAST | On demand | Manually |
| AES-CCM-Dec (A4653) | KAT | CAST | On demand | Manually |
| AES-CFB128-Enc (A4653) | KAT | CAST | On demand | Manually |
| AES-CFB128-Dec (A4653) | KAT | CAST | On demand | Manually |
| AES-CTR-Enc (A4653) | KAT | CAST | On demand | Manually |
| AES-CTR-Dec (A4653) | KAT | CAST | On demand | Manually |
| AES-ECB-Enc (A4653) | KAT | CAST | On demand | Manually |
| AES-ECB-Dec (A4653) | KAT | CAST | On demand | Manually |
| AES-OFB-Enc (A4653) | KAT | CAST | On demand | Manually |
| AES-OFB-Dec (A4653) | KAT | CAST | On demand | Manually |
| AES-XTS-Enc Testing Revision 2.0 (A4653) | KAT | CAST | On demand | Manually |
| AES-XTS-Dec Testing Revision 2.0 (A4653) | KAT | CAST | On demand | Manually |
| AES-CMAC-Gen (A4653) | KAT | CAST | On demand | Manually |
| AES-GCM-Enc (A4654) | KAT | CAST | On demand | Manually |
| AES-GCM-Dec (A4654) | KAT | CAST | On demand | Manually |
| AES-GCM-Enc (A4655) | KAT | CAST | On demand | Manually |
| AES-GCM-Dec (A4655) | KAT | CAST | On demand | Manually |
| AES-GMAC-Gen (A4654) | KAT | CAST | On demand | Manually |
| AES-GMAC-Gen (A4655) | KAT | CAST | On demand | Manually |
| KAS-FFC Sp800-56Ar3 (A4653) | KAT | CAST | On demand | Manually |
| KAS-ECC Sp800-56Ar3 (A4653) | KAT | CAST | On demand | Manually |
| Counter DRBG (A4653) | KAT | CAST | Initialization and on demand | Manually |

| Algorithm or Test | Test Method | Test Type | Period | Periodic Method |
|---|---|---|---|---|
| ECDSA SigGen (FIPS186-4) (A4653) | KAT | CAST | On demand | Manually |
| ECDSA SigVer (FIPS186-4) (A4653) | KAT | CAST | On demand | Manually |
| ECDSA KeyGen (FIPS186-4) (A4653) | PCT | PCT | On demand | Manually |
| EDDSA SigGen (A4653) | KAT | CAST | On demand | Manually |
| EDDSA SigVer (A4653) | KAT | CAST | On demand | Manually |
| EDDSA KeyGen (A4653) | PCT | PCT | On demand | Manually |
| HMAC-SHA-1 (A4653) | KAT | CAST | On demand | Manually |
| HMAC-SHA2-224 (A4653) | KAT | CAST | On demand | Manually |
| HMAC-SHA2-256 (A4653) | KAT | CAST | On demand | Manually |
| HMAC-SHA2-384 (A4653) | KAT | CAST | On demand | Manually |
| HMAC-SHA2-512 (A4653) | KAT | CAST | On demand | Manually |
| HMAC-SHA3-224 (A4653) | KAT | CAST | On demand | Manually |
| HMAC-SHA3-256 (A4653) | KAT | CAST | On demand | Manually |
| HMAC-SHA3-384 (A4653) | KAT | CAST | On demand | Manually |
| HMAC-SHA3-512 (A4653) | KAT | CAST | On demand | Manually |
| KDF SP800-108-SHA2-224 | KAT | CAST | On demand | Manually |
| KDF SP800-108-SHA2-256 | KAT | CAST | On demand | Manually |
| KDF SP800-108-SHA2-384 | KAT | CAST | On demand | Manually |
| KDF SP800-108-SHA2-512 | KAT | CAST | On demand | Manually |
| KDF SP800-108-SHA3-224 | KAT | CAST | On demand | Manually |
| KDF SP800-108-SHA3-256 | KAT | CAST | On demand | Manually |
| KDF SP800-108-SHA3-384 | KAT | CAST | On demand | Manually |
| KDF SP800-108-SHA3-512 | KAT | CAST | On demand | Manually |
| SHA-1 (A4653) | KAT | CAST | On demand | Manually |
| SHA2-224 (A4653) | KAT | CAST | On demand | Manually |
| SHA2-256 (A4653) | KAT | CAST | On demand | Manually |
| SHA2-384 (A4653) | KAT | CAST | On demand | Manually |
| SHA2-512 (A4653) | KAT | CAST | On demand | Manually |
| SHA3-224 (A4653) | KAT | CAST | On demand | Manually |

| Algorithm or Test | Test Method | Test Type | Period | Periodic Method |
|---|---|---|---|---|
| SHA3-256 (A4653) | KAT | CAST | On demand | Manually |
| SHA3-384 (A4653) | KAT | CAST | On demand | Manually |
| SHA3-512 (A4653) | KAT | CAST | On demand | Manually |
| SHAKE-128 (A4653) | KAT | CAST | On demand | Manually |
| SHAKE-256 (A4653) | KAT | CAST | On demand | Manually |
| RSA SigGen (FIPS186-4) (A4653) | KAT | CAST | On demand | Manually |
| RSA SigVer (FIPS186-4) (A4653) | KAT | CAST | On demand | Manually |
| RSA KeyGen (FIPS186-4) (A4653) | PCT | PCT | On demand | Manually |
| KDF SP800-108-SHA-1 | KAT | CAST | On demand | Manually |
| DSA KeyGen (FIPS186-4) (A4653) | PCT | CAST | On demand | Manually |
| DSA SigGen (FIPS186-4) (A4653) | KAT | CAST | On demand | Manually |
| DSA SigVer (FIPS186-4) (A4653) | KAT | CAST | On demand | Manually |
| AES-GMAC-Ver (A4654) | KAT | CAST | On demand | Manually |
| AES-GMAC-Ver (A4655) | KAT | CAST | On demand | Manually |
| AES-CMAC-Ver (A4653) | KAT | CAST | On demand | Manually |
| AES-XTS Key Comparison | AES-XTS key comparison | Critical Function | On demand | Manually |

Table 23: Conditional Periodic Information

## 10.4 Error States

| Name | Description | Conditions | Recovery Method | Indicator |
|---|---|---|---|---|
| ES1 | The Module enters the disable Crypto Module "Error State" | The Module fails the software integrity pre-operational self-test. | Reboot/Power cycle the module | Outputs status of ERR_FIPS_INTEGRITY_FAIL, otherwise it indicates successful completion by returning the OK status. |
| ES2 | The Module enters the disable Crypto Module "Error State" | The Module fails the software CAST test with a specified error number. | Reboot/Power cycle the module | Outputs a specific error status; otherwise, it indicates successful completion by enabling the Crypto Module with OK status. |
| ES3 | The Module enters the disable Crypto Module "Error State" | The Module fails all other self-tests not listed above. | Reboot/Power cycle the module | Outputs a specific error status; otherwise, it indicates successful completion by enabling the Crypto Module with OK status. |

Table 24: Error States

# 11 Life-Cycle Assurance

## 11.1 Installation, Initialization, and Startup Procedures

Installation is performed by placing the module in the target file system during the OEM or ISV's manufacturing process. The module initialization is performed automatically by the operating system's loader when a calling application is loaded into memory. Operation of the module is controlled by the calling application's use of the module's API functions.

**Installation and Initialization:**

The following steps must be performed in order to securely install, initialize, and start up the DigiCert TrustCore Cryptographic Suite B Module in the FIPS 140-3 Approved mode of operation:

The Module shall be installed within the operating system confines and structures consistent with DigiCert's operating environment specific documentation. The installation may be specified in the operating environment's specific documentation. The Cryptographic Officer will install the Module and associated signature of the Module into the proper location within the computer system. For example, the shared memory library and signature file may be installed in the /usr/local/lib directory, which is protected by Linux access control mechanisms. The Module is protected from modification by the integrity self-test performed during start-up. The Module is initialized by the operating system upon loading the Module into memory for use by calling applications.

The Module must be operated in the approved mode to ensure that FIPS 140-3 validated cryptographic algorithms and security functions are used.

In addition, the security rules defined in Section Rules of Operation shall apply to the operating system.

## 11.2 Administrator Guidance

The module is provided with supporting documentation which includes an API Reference document and Operating Environment document.

## 11.3 Non-Administrator Guidance

The Module supports the Cryptographic Officer (CO) operator role and does not support non-administrators or non-administrative roles.

## 11.4 Design and Rules

**(RSA)**
The calling application of the Module must generate RSA key pairs of at least 2048 bits to operate in approved mode.

**(ECC)**
The calling application of the module must generate ECC keys using a P-Curve with a security strength of at least 112 bits to operate in the approved mode of operation.

**(Random Number Generation)**
The Module implements a CTR-based DRBG per SP800-90Ar1 for creation of symmetric and asymmetric keys.
The Module accepts input from entropy sources external to the cryptographic boundary for use as seed material for the Module's approved DRBG. External entropy can be added via the AES-CTR-DRBG Gen and the AES-CTR-DRBG ReSeed service available to the cryptographic module client application.
The calling application of the Module shall use entropy sources that meet the security strength required for the random bit generation mechanism as shown in NIST SP 800-90Ar1 Table 3 (CTR_DRBG). A minimum of 384 bits of entropy must be provided by the calling application. The calling application shall provide full entropy for 256-bit keys. Due to the entropy being provided by an external source, the following caveat applies: There is no assurance of the minimum strength of generated SSPs (e.g., keys).
The Module performs DRBG health tests (Instantiate, Generate, ReSeed) as defined in Section 11.3 of SP800-90Ar1.

**(Key Management)**
The application that uses the module is responsible for appropriate destruction and zeroization of the keys. The Module provides API calls for key allocation and destruction. These API calls overwrite the memory occupied by the key information with zeros before that memory is de-allocated. See Key Destruction Service below.

**(Key/CSP Authorized Access and Use)**
An authorized application has access to all key data generated during the operation of the Module.

**(Key/CSP Storage)**
Private and public keys are provided to the module by the calling process and are destroyed when released by the appropriate API function calls. The module does not perform persistent storage of keys.

**(Key/CSP Zeroization)**
The application is responsible for calling the appropriate destruction functions from the API. These functions overwrite the memory with zeros and de-allocate the memory. In case of abnormal termination, the Linux kernel overwrites the keys in physical memory before the physical memory is allocated to another process.

**(Key Destruction Service)**
A context structure is associated with every cryptographic algorithm available in the Module. Context structures hold sensitive information such as cryptographic keys. These context structures must be zeroized when the application software no longer needs to use a specific algorithm. This API call will zeroize all sensitive information before freeing the dynamically allocated memory. This will occur while the application process is still in memory, but no longer needs the specific algorithm, which protects the sensitive information from compromise. See the *Cryptographic API Reference* for additional information.

## Rules of Operation

1. The Module provides one operator role: Cryptographic Officer.
2. The Module does not provide any operator authentication.
3. An operator does not have access to any cryptographic services prior to assuming an authorized role.

4. The Module allows the operator to initiate power-up self-tests by power cycling power or reloading the Module into memory.
5. All self-tests do not require any operator action.
6. Data and Control outputs are inhibited during key generation, self-tests, zeroization, and error states. Because the logical interface is defined as the API of the Module and the API of the Module is single-threaded, key generation or zeroization must be complete before the API returns control to the calling application.
7. Status information does not contain CSPs or sensitive data that if misused could lead to a compromise of the Module.
8. There are no restrictions on which keys or SSPs are zeroized by the zeroization service.
9. The Module does not support concurrent operators.
10. The Module does not support a maintenance interface or role.
11. The Module does not support a manual SSP establishment method.
12. The Module does not have any proprietary external input/output devices used for entry/output of data.
13. The Module does not enter or output plaintext SSPs, except to/from the calling application via API parameters. The module does not support the entry or output of encrypted SSPs.
14. The Module does not store any plaintext SSPs. SSPs provided to the Module by the calling processes are destroyed when released by the appropriate API function calls.
15. The Module does not output intermediate key values.
16. The Module does not provide bypass services or ports/interfaces.
17. AES GCM IV uniqueness: The AES GCM implementation meets Option 1 of IG C.H. The Module supports TLS 1.2 GCM Cipher Suites for TLS, as described in RFCs 5116, 5288 and 5289. The counter portion of the IV is set by the Module within its cryptographic boundary.
18. When the nonce explicit (counter) part of the IV exhausts the maximum number of possible values for a given session key this condition triggers a handshake to establish a new encryption key per RFC 5246. During operational testing, the Module was tested against an independent version of TLS and found to behave correctly.
    AES GCM keys are zeroized when the Module is power-cycled and for each new TLS session, a new AES GCM key is established.
19. AES XTS is to be used only for storage purposes, per SP800-38E.

## 11.5 Maintenance Requirements

The module currently does not have any maintenance requirements.

## 11.6 End of Life

For secure sanitization all SSPs shall first be zeroized and the calling application shall be closed. SSP zeroization is performed through the Key Destruction service that is described below. API calls will

overwrite the memory occupied by the key information with zeros before that memory is de-allocated. If the calling application is terminated prior to zeroization, the Linux kernel overwrites the keys in physical memory before the physical memory is allocated to another process. The key zeroization process is performed in a sufficient time to prevent compromise of SSPs, taking only a few milliseconds.

Then, for actual deprecation, the module shall be upgraded to a newer version that is FIPS 140-3 validated. Since the module does not possess persistent storage of SSPs, no further sanitization steps are needed.

# 12 Mitigation of Other Attacks

The Module does not implement any mitigation method against other attacks.

# References and Definitions

The following standards are referred to in this Security Policy.

Table 25: References

| Abbreviation* | Full Specification Name |
|---|---|
| [FIPS140-3] | *Security Requirements for Cryptographic Modules*, March 22, 2019 |
| [ISO19790] | *International Standard, ISO/IEC 19790, Information technology — Security techniques — Test requirements for cryptographic modules, Third edition, March 2017* |
| [ISO24759] | *International Standard, ISO/IEC 24759, Information technology — Security techniques — Test requirements for cryptographic modules, Second and Corrected version, 15 December 2015* |
| [IG] | *Implementation Guidance for FIPS PUB 140-3 and the Cryptographic Module Validation Program, April 18, 2025* |
| [108] | *NIST Special Publication 800-108, Recommendation for Key Derivation Using Pseudorandom Functions (Revised), August 2022* |
| [131A] | *Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths, Revision 2, March 2019* |
| [132] | *NIST Special Publication 800-132, Recommendation for Password-Based Key Derivation, Part 1: Storage Applications, December 2010* |
| [133] | *NIST Special Publication 800-133r2, Recommendation for Cryptographic Key Generation, Revision 2, June 2020* |
| [135] | *National Institute of Standards and Technology, Recommendation for Existing Application-Specific Key Derivation Functions, Special Publication 800-135rev1, December 2011.* |
| [186] | *National Institute of Standards and Technology, Digital Signature Standard (DSS), Federal Information Processing Standards Publication 186-4, July 2013.* |
| [197] | *National Institute of Standards and Technology, Advanced Encryption Standard (AES), Federal Information Processing Standards Publication 197, November 26, 2001* |
| [198] | *National Institute of Standards and Technology, The Keyed-Hash Message Authentication Code (HMAC), Federal Information Processing Standards Publication 198-1, July, 2008* |
| [180] | *National Institute of Standards and Technology, Secure Hash Standard, Federal Information Processing Standards Publication 180-4, August, 2015* |
| [202] | *FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION, SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions, FIPS PUB 202, August 2015* |
| [38A] | *National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation, Methods and Techniques, Special Publication 800-38A, December 2001* |

| Abbreviation* | Full Specification Name |
|---|---|
| [38B] | *National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication, Special Publication 800-38B, May 2005* |
| [38C] | *National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality, Special Publication 800-38C, May 2004* |
| [38D] | *National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC, Special Publication 800-38D, November 2007* |
| [38E] | *National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation: The XTS-AES Mode for Confidentiality on Storage Devices, Special Publication 800-38E, January 2010* |
| [38F] | *National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping, Special Publication 800-38F, December 2012* |
| [56Ar3] | *NIST Special Publication 800-56A Revision 3, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography, April 2018* |
| [56Br2] | *NIST Special Publication 800-56B Revision 2, Recommendation for Pair-Wise Key Establishment Schemes Using Finite Field Cryptography, March 2019* |
| [56Cr2] | *NIST Special Publication 800-56C Revision 2, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography, August 2020* |
| [67] | *National Institute of Standards and Technology, Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher, Special Publication 800-67, May 2004* |
| [90A] | *National Institute of Standards and Technology, Recommendation for Random Number Generation Using Deterministic Random Bit Generators, Special Publication 800-90A, Revision 1, June 2015.* |
| [90B] | *National Institute of Standards and Technology, Recommendation for the Entropy Sources Used for Random Bit Generation, Special Publication 800-90B, January 2018.* |

Table 26: Acronyms and Definitions

| Acronym* | Definition |
|---|---|
| AES | Advanced Encryption Standard |
| AES-NI | Advanced Encryption Standard New Instructions |
| API | Application Program Interface |
| CBC | Cipher Block Chaining |
| CCM | Counter with Cipher Block Chaining-Message Authentication Code |
| CMAC | Cipher-based Message Authentication Code |

| Acronym* | Definition |
|---|---|
| CMVP | Cryptographic Module Validation Program |
| CSP | Critical Security Parameter |
| CTR | Counter Mode |
| DES | Data Encryption Standard |
| DH | Diffie-Hellman |
| DRBG | Deterministic Random Bit Generator |
| DSA | Digital Signature Algorithm |
| ECC CDH | Elliptic Curve Cryptography Cofactor Diffie-Hellman |
| ECDSA | Elliptic Curve Digital Signature Algorithm |
| EdDSA | Edwards-curve Digital Signature Algorithm |
| EMC | Electromagnetic Compatibility |
| EMI | Electromagnetic Interference |
| FIPS | Federal Information Processing Standard |
| GCM | Galois Counter Mode |
| HMAC | Hash Message Authentication Code |
| IG | Implementation Guidance |
| KAT | Known Answer Test |
| KDF | Key Derivation Function |
| KVM | Kernel-based Virtual Machine |
| NC | Non-Compliant |
| PAA | Processor Algorithm Acceleration |
| PCT | Pair-wise Consistency Test |
| RNG | Random Number Generator |
| RSA | Rivest, Shamir and Adleman Algorithm |
| SHA | Secure Hash Algorithm |
| SHS | Secure Hash Standard |
| SO | Shared Object |
| TDES | Triple-DES |
| XTS | XEX-based Tweaked-codebook mode with ciphertext Stealing |