Ezurio

# Summit Linux FIPS Core Crypto Module

# FIPS 140-3 Non-Proprietary Security Policy

Document Version: 1.1

Last Modified: 08/19/2025

Prepared by:

atsec information security corporation

4516 Seton Center Pkwy, Suite 250

Austin, TX 78759

www.atsec.com

# Table of Contents

# List of Tables



# List of Figures

# 1 General

## 1.1 Overview

This document is the non-proprietary FIPS 140-3 Security Policy for the Summit Linux FIPS Core Crypto Module, firmware version 11.1, hardware version ATSAMA5D31, ATSAMA5D36. It contains the security rules under which the module must be operated and describes how this module meets the requirements as specified in FIPS 140-3 (Federal Information Processing Standards Publication 140-3) for a Security Level 1 module.

This Security Policy contains non-proprietary information. All other documentation submitted for FIPS 140-3 conformance testing and validation is proprietary and is releasable only under appropriate non-disclosure agreements.

### 1.1.1 How this Security Policy was prepared

The vendor has provided the non-proprietary Security Policy of the cryptographic module, which was further consolidated into this document by atsec information security together with other vendor-supplied documentation. In preparing the Security Policy document, the laboratory formatted the vendor-supplied documentation for consolidation without altering the technical statements therein contained. The further refining of the Security Policy document was conducted iteratively throughout the conformance testing, wherein the Security Policy was submitted to the vendor, who would then edit, modify, and add technical contents. The vendor would also supply additional documentation, which the laboratory formatted into the existing Security Policy, and resubmitted to the vendor for their final editing.

## 1.2 Security Levels

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

Table 1: Security Levels

# 2 Cryptographic Module Specification

## 2.1 Description

### Purpose and Use:

The Summit Linux FIPS Core Crypto Module (hereafter referred to as the "module") is a Firmware-Hybrid module supporting FIPS 140-3 Approved cryptographic algorithms. The module is composed by a hardware component, the ARM-based Microchip/Atmel microprocessor, and firmware components comprised of a kernel and OpenSSL library, and fipscheck binary. The firmware components provide a C language application program interface (API) for use by other processes that require cryptographic functionality.

The module offers approved cryptographic functions in the Approved mode for, among other uses:

- Algorithms for use in the Wi-Fi protocols CCMP and GCMP.
- Algorithms for use in the TLS protocol.
- Encryption and decryption for data at rest.

**Module Type**: Firmware-hybrid

**Module Embodiment**: MultiChipStand

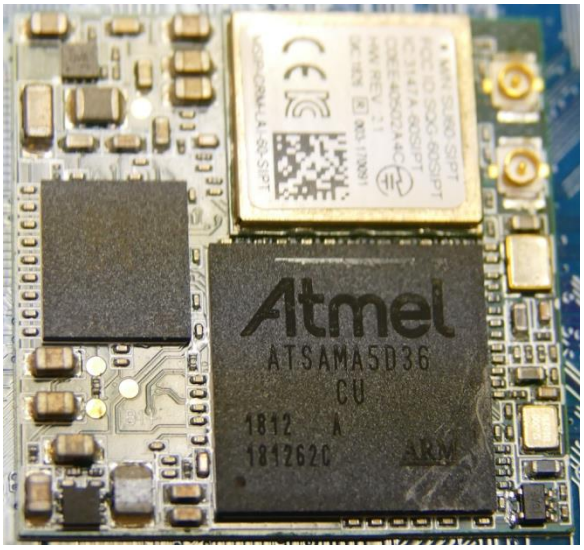
### Cryptographic Boundary:

The Summit Linux FIPS Core Crypto Module is defined as a Firmware-Hybrid, Multi-chip Standalone module per the requirements within FIPS 140-3. The cryptographic boundary of the module consists of the embedded hardware AES cryptographic engine within the Microchip/Atmel microprocessor, the firmware component files (Kernel, OpenSSL FIPS provider, and the fipscheck integrity test tool) and the integrity test HMAC files. For easier readability in the rest of the document "OpenSSL FIPS provider" has been shortened to "FIPS provider".

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

Figure 1 below shows the physical representations of the tested platforms as the top view of the circuit boards. The tested environments are further described in Section 6.


(a) WB50NBT with Microchip/Atmel ATSAMA5D31.




(b) SU60-SOMC 60 Series SOM (System on Module) with Microchip/Atmel ATSAMA5D36

Figure 1: Tested Operational Environment Physical Perimeter

Figure 2 below shows the block diagram of the module. The cryptographic boundary is indicated with yellow blocks, distributed among hardware and firmware components. Blocks of another color do not belong to the cryptographic boundary. Users of the module interact through the API that are the logical interfaces data input, data output, control input, status output. A dotted line encompasses the module's components that interface through the API.

In Figure 2, users of the module are exemplified by applications. These applications may reside within the NAND Flash memory or may reside outside (but still within the physical perimeter), always interacting with the module's API.

The physical perimeter of the module is defined as the perimeter of the circuit board on which the module is installed. The filesystem and operating system reside on NAND Flash memory within the physical perimeter.

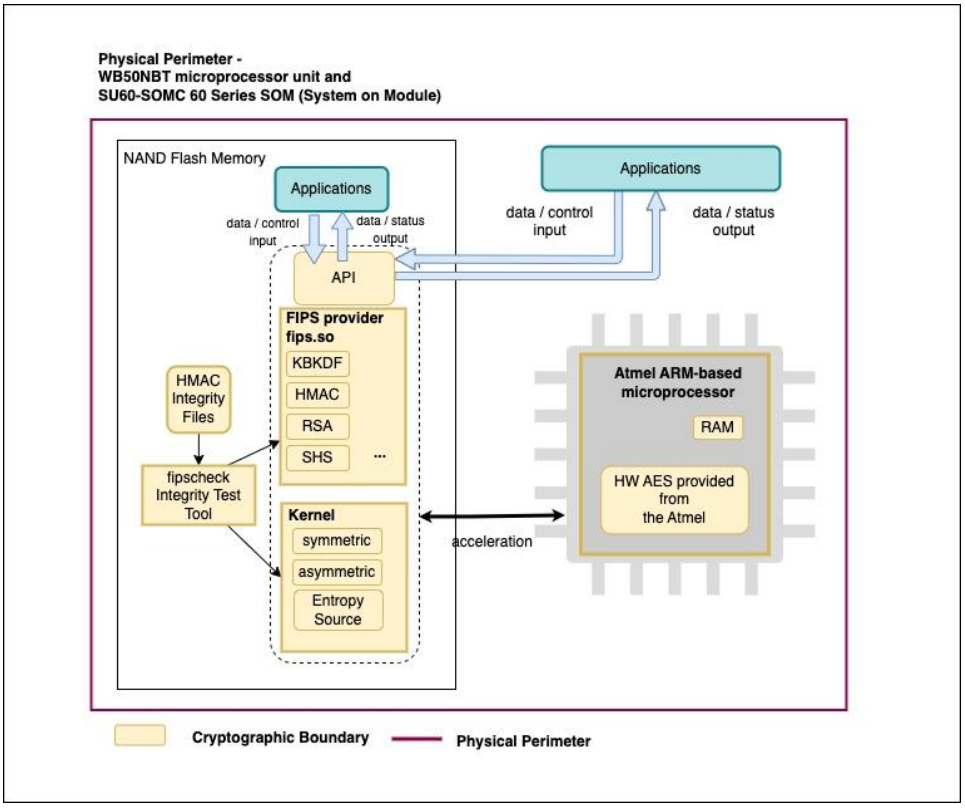



Figure 2: Block Diagram

## 2.2 Tested and Vendor Affirmed Module Version and Identification

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

| Package or File Name | Software/ Firmware Version | Features | Integrity Test |
|---|---|---|---|
| Image.gz, fips.so and fipscheck (application and library) | 11.1 | N/A | HMAC-SHA-256 |

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

### Tested Module Identification – Hybrid Disjoint Hardware:

| Model and/or Part Number | Hardware Version | Firmware Version | Processors | Features |
|---|---|---|---|---|
| ATSAMA5D31 | ATSAMA5D31 | N/A | N/A | N/A |
| ATSAMA5D36 | ATSAMA5D36 | N/A | N/A | N/A |

Table 3: Tested Module Identification – Hybrid Disjoint Hardware

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

| Operating System | Hardware Platform | Processors | PAA/PAI | Hypervisor or Host OS | Version(s) |
|---|---|---|---|---|---|
| Summit Linux 11.1 | Ezurio WB50NBT System-On-Module | Microchip/Atmel ATSAMA5D31, ARM Cortex A5-based (ARMv7) | No | N/A | 11.1 |
| Summit Linux 11.1 | Ezurio 60 Series SOM (System on Module) | Microchip/Atmel ATSAMA5D36, ARM Cortex A5-based (ARMv7) | No | N/A | 11.1 |

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

## 2.3 Excluded Components

There are no components within the cryptographic boundary excluded from the FIPS 140-3 requirements.

## 2.4 Modes of Operation

### Modes List and Description:

| Mode Name | Description | Type | Status Indicator |
|---|---|---|---|
| Approved mode | Automatically entered whenever an approved service is requested | Approved | Equivalent to the indicator of the requested service as defined in section 4.2 |
| Non-approved mode | Automatically entered whenever a non-approved service is requested | Non-Approved | Equivalent to the indicator of the requested service as defined in section 4.3 |

Table 5: Modes List and Description

After passing all pre-operational self-tests and cryptographic algorithm self-tests executed on start-up, the module automatically transitions to the approved mode.

### Mode Change Instructions and Status:

The module automatically switches between the approved and non-approved modes depending on the services requested by the operator. The status indicator of the mode of operation is equivalent to the indicator of the service that was requested.

## 2.5 Algorithms

### Approved Algorithms:

| Algorithm | CAVP Cert | Properties | Reference |
|---|---|---|---|
| AES-CBC | A4712 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CBC | A4716 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CBC | A4719 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CBC | A4721 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CBC | A5004 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CBC-CS1 | A5004 | Direction - decrypt, encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CBC-CS2 | A5004 | Direction - decrypt, encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CBC-CS3 | A4714 | Direction - decrypt, encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CBC-CS3 | A4718 | Direction - decrypt, encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CBC-CS3 | A4720 | Direction - decrypt, encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CBC-CS3 | A4722 | Direction - decrypt, encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CBC-CS3 | A5004 | Direction - decrypt, encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CCM | A4712 | Key Length - 128, 192, 256 | SP 800-38C |
| AES-CCM | A4716 | Key Length - 128, 192, 256 | SP 800-38C |
| AES-CCM | A4719 | Key Length - 128, 192, 256 | SP 800-38C |
| AES-CCM | A4721 | Key Length - 128, 192, 256 | SP 800-38C |
| AES-CCM | A5004 | Key Length - 128, 192, 256 | SP 800-38C |
| AES-CFB1 | A5004 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CFB128 | A4724 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CFB128 | A5004 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CFB8 | A4724 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CFB8 | A5004 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CMAC | A4712 | Direction - Generation, Verification<br>Key Length - 128, 192, 256 | SP 800-38B |

| Algorithm | CAVP Cert | Properties | Reference |
|---|---|---|---|
| AES-CMAC | A4716 | Direction - Generation, Verification<br>Key Length - 128, 192, 256 | SP 800-38B |
| AES-CMAC | A4719 | Direction - Generation, Verification<br>Key Length - 128, 192, 256 | SP 800-38B |
| AES-CMAC | A4721 | Direction - Generation, Verification<br>Key Length - 128, 192, 256 | SP 800-38B |
| AES-CMAC | A5004 | Direction - Generation, Verification<br>Key Length - 128, 192, 256 | SP 800-38B |
| AES-CTR | A4712 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CTR | A4716 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CTR | A4719 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CTR | A4721 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-CTR | A5004 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-ECB | A4711 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-ECB | A4712 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-ECB | A4715 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-ECB | A4716 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-ECB | A4717 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-ECB | A4719 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-ECB | A4721 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-ECB | A5004 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-ECB | A5019 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-GCM | A4712 | Direction - Decrypt, Encrypt<br>IV Generation - External<br>IV Generation Mode - 8.2.1<br>Key Length - 128, 192, 256 | SP 800-38D |
| AES-GCM | A4715 | Direction - Decrypt, Encrypt<br>IV Generation - External | SP 800-38D |

| Algorithm | CAVP Cert | Properties | Reference |
|---|---|---|---|
| | | IV Generation Mode - 8.2.1<br>Key Length - 128, 192, 256 | |
| AES-GCM | A4717 | Direction - Decrypt, Encrypt<br>IV Generation - External<br>IV Generation Mode - 8.2.1<br>Key Length - 128, 192, 256 | SP 800-38D |
| AES-GCM | A4719 | Direction - Decrypt, Encrypt<br>IV Generation - External<br>IV Generation Mode - 8.2.1<br>Key Length - 128, 192, 256 | SP 800-38D |
| AES-GCM | A4721 | Direction - Decrypt, Encrypt<br>IV Generation - External<br>IV Generation Mode - 8.2.1<br>Key Length - 128, 192, 256 | SP 800-38D |
| AES-GCM | A5008 | Direction - Decrypt, Encrypt<br>IV Generation - External, Internal<br>IV Generation Mode - 8.2.1, 8.2.2<br>Key Length - 128, 192, 256 | SP 800-38D |
| AES-GMAC | A4712 | Direction - Decrypt, Encrypt<br>IV Generation - External<br>IV Generation Mode - 8.2.1<br>Key Length - 128, 192, 256 | SP 800-38D |
| AES-GMAC | A4719 | Direction - Decrypt, Encrypt<br>IV Generation - External<br>IV Generation Mode - 8.2.1<br>Key Length - 128, 192, 256 | SP 800-38D |
| AES-GMAC | A4721 | Direction - Decrypt, Encrypt<br>IV Generation - External<br>IV Generation Mode - 8.2.1<br>Key Length - 128, 192, 256 | SP 800-38D |
| AES-GMAC | A5008 | Direction - Decrypt, Encrypt<br>IV Generation - External<br>IV Generation Mode - 8.2.1<br>Key Length - 128, 192, 256 | SP 800-38D |
| AES-KW | A5004 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38F |
| AES-KWP | A5004 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38F |
| AES-OFB | A4723 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |
| AES-OFB | A5004 | Direction - Decrypt, Encrypt<br>Key Length - 128, 192, 256 | SP 800-38A |

| Algorithm | CAVP Cert | Properties | Reference |
|---|---|---|---|
| AES-XTS Testing Revision 2.0 | A4712 | Direction - Decrypt, Encrypt<br>Key Length - 128, 256 | SP 800-38E |
| AES-XTS Testing Revision 2.0 | A4716 | Direction - Decrypt, Encrypt<br>Key Length - 128, 256 | SP 800-38E |
| AES-XTS Testing Revision 2.0 | A4719 | Direction - Decrypt, Encrypt<br>Key Length - 128, 256 | SP 800-38E |
| AES-XTS Testing Revision 2.0 | A4721 | Direction - Decrypt, Encrypt<br>Key Length - 128, 256 | SP 800-38E |
| AES-XTS Testing Revision 2.0 | A5004 | Direction - Decrypt, Encrypt<br>Key Length - 128, 256 | SP 800-38E |
| Counter DRBG | A4711 | Prediction Resistance - No, Yes<br>Mode - AES-128, AES-192, AES-256<br>Derivation Function Enabled - Yes | SP 800-90A Rev. 1 |
| Counter DRBG | A4712 | Prediction Resistance - No, Yes<br>Mode - AES-128, AES-192, AES-256<br>Derivation Function Enabled - Yes | SP 800-90A Rev. 1 |
| Counter DRBG | A4715 | Prediction Resistance - No, Yes<br>Mode - AES-128, AES-192, AES-256<br>Derivation Function Enabled - Yes | SP 800-90A Rev. 1 |
| Counter DRBG | A4717 | Prediction Resistance - No, Yes<br>Mode - AES-128, AES-192, AES-256<br>Derivation Function Enabled - Yes | SP 800-90A Rev. 1 |
| Counter DRBG | A4719 | Prediction Resistance - No, Yes<br>Mode - AES-128, AES-192, AES-256<br>Derivation Function Enabled - Yes | SP 800-90A Rev. 1 |
| Counter DRBG | A4721 | Prediction Resistance - No, Yes<br>Mode - AES-128, AES-192, AES-256<br>Derivation Function Enabled - Yes | SP 800-90A Rev. 1 |
| Counter DRBG | A5015 | Prediction Resistance - No, Yes<br>Mode - AES-128, AES-192, AES-256<br>Derivation Function Enabled - No, Yes | SP 800-90A Rev. 1 |
| ECDSA KeyGen (FIPS186-5) | A4711 | Curve - P-256, P-384<br>Secret Generation Mode - testing candidates | FIPS 186-5 |
| ECDSA KeyGen (FIPS186-5) | A5018 | Curve - P-224, P-256, P-384, P-521<br>Secret Generation Mode - testing candidates | FIPS 186-5 |
| ECDSA KeyVer (FIPS186-5) | A5018 | Curve - P-224, P-256, P-384, P-521 | FIPS 186-5 |

| Algorithm | CAVP Cert | Properties | Reference |
|---|---|---|---|
| ECDSA SigGen (FIPS186-5) | A5018 | Curve - P-224, P-256, P-384, P-521<br>Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256<br>Component - No | FIPS 186-5 |
| ECDSA SigGen (FIPS186-5) | A5020 | Curve - P-224, P-256, P-384, P-521<br>Hash Algorithm - SHA3-224, SHA3-256, SHA3-384, SHA3-512<br>Component - No | FIPS 186-5 |
| ECDSA SigVer (FIPS186-5) | A5018 | Curve - P-224, P-256, P-384, P-521<br>Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256 | FIPS 186-5 |
| ECDSA SigVer (FIPS186-5) | A5020 | Curve - P-224, P-256, P-384, P-521<br>Hash Algorithm - SHA3-224, SHA3-256, SHA3-384, SHA3-512 | FIPS 186-5 |
| EDDSA KeyGen | A5016 | Curve - ED-25519, ED-448 | FIPS 186-5 |
| EDDSA SigGen | A5016 | Curve - ED-25519, ED-448 | FIPS 186-5 |
| EDDSA SigVer | A5016 | Curve - ED-25519, ED-448 | FIPS 186-5 |
| Hash DRBG | A5015 | Prediction Resistance - No, Yes<br>Mode - SHA-1, SHA2-256, SHA2-512 | SP 800-90A Rev. 1 |
| HMAC DRBG | A5015 | Prediction Resistance - No, Yes<br>Mode - SHA-1, SHA2-256, SHA2-512 | SP 800-90A Rev. 1 |
| HMAC-SHA-1 | A5018 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-224 | A4711 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-224 | A4712 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-224 | A4716 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-224 | A5018 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-256 | A4711 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-256 | A4712 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-256 | A4716 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-256 | A5018 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-384 | A4711 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-384 | A4712 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-384 | A4716 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |

| Algorithm | CAVP Cert | Properties | Reference |
|---|---|---|---|
| HMAC-SHA2-384 | A5018 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-512 | A4711 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-512 | A4712 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-512 | A4716 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-512 | A5018 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-512/224 | A5018 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA2-512/256 | A5018 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA3-224 | A4713 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA3-224 | A5020 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA3-256 | A4713 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA3-256 | A5020 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA3-384 | A4713 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA3-384 | A5020 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA3-512 | A4713 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| HMAC-SHA3-512 | A5020 | Key Length - Key Length: 112-524288 Increment 8 | FIPS 198-1 |
| KAS-ECC-SSC Sp800-56Ar3 | A4711 | Domain Parameter Generation Methods - P-256, P-384<br>Scheme -<br>ephemeralUnified -<br>KAS Role - initiator, responder | SP 800-56A Rev. 3 |
| KAS-ECC-SSC Sp800-56Ar3 | A5018 | Domain Parameter Generation Methods - P-224, P-256, P-384, P-521<br>Scheme -<br>ephemeralUnified -<br>KAS Role - initiator, responder | SP 800-56A Rev. 3 |
| KAS-FFC-SSC Sp800-56Ar3 | A5014 | Domain Parameter Generation Methods - ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192<br>Scheme - | SP 800-56A Rev. 3 |

| Algorithm | CAVP Cert | Properties | Reference |
|---|---|---|---|
| | | dhEphem -<br>KAS Role - initiator, responder | |
| KAS-IFC-SSC | A5018 | Modulo - 2048, 3072, 4096, 6144, 8192<br>Key Generation Methods - rsakpg1-basic, rsakpg1-crt, rsakpg1-prime-factor, rsakpg2-basic, rsakpg2-crt, rsakpg2-prime-factor<br>Scheme -<br>KAS1 -<br>KAS Role - initiator, responder | SP 800-56A Rev. 3 |
| KDA HKDF Sp800-56Cr1 | A5013 | Derived Key Length - 2048<br>Shared Secret Length - Shared Secret Length: 224-2048 Increment 8<br>HMAC Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512 | SP 800-56C Rev. 2 |
| KDA OneStep SP800-56Cr2 | A5012 | Derived Key Length - 2048<br>Shared Secret Length - Shared Secret Length: 224-2048 Increment 8 | SP 800-56C Rev. 2 |
| KDA TwoStep SP800-56Cr2 | A5012 | MAC Salting Methods - default, random<br>KDF Mode - feedback<br>Derived Key Length - 2048<br>Shared Secret Length - Shared Secret Length: 224-2048 Increment 8 | SP 800-56C Rev. 2 |
| KDF ANS 9.42 (CVL) | A5018 | KDF Type - DER<br>Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256<br>Key Data Length - Key Data Length: 8-4096 Increment 8 | SP 800-135 Rev. 1 |
| KDF ANS 9.42 (CVL) | A5020 | KDF Type - DER<br>Hash Algorithm - SHA3-224, SHA3-256, SHA3-384, SHA3-512<br>Key Data Length - Key Data Length: 8-4096 Increment 8 | SP 800-135 Rev. 1 |
| KDF ANS 9.63 (CVL) | A5018 | Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256<br>Key Data Length - Key Data Length: 128-4096 Increment 8 | SP 800-135 Rev. 1 |
| KDF KMAC Sp800-108r1 | A5017 | Derived Key Length - Derived Key Length: 112-4096 Increment 8 | SP 800-108 Rev. 1 |
| KDF SP800-108 | A5017 | KDF Mode - Counter, Feedback<br>Supported Lengths - Supported Lengths: 112, 128, 776, 3456, 4096 | SP 800-108 Rev. 1 |
| KDF SSH (CVL) | A5019 | Cipher - AES-128, AES-192, AES-256, TDES<br>Hash Algorithm - SHA-1, SHA2-256, SHA2-384, SHA2-512 | SP 800-135 Rev. 1 |
| KDF TLS (CVL) | A5018 | TLS Version - v1.0/1.1 | SP 800-135 Rev. 1 |
| KMAC-128 | A5020 | Message Length - Message Length: 0-65536 Increment 8<br>Key Data Length - Key Data Length: 128-1024 Increment 8 | SP 800-185 |

| Algorithm | CAVP Cert | Properties | Reference |
|---|---|---|---|
| KMAC-256 | A5020 | Message Length - Message Length: 0-65536 Increment 8<br>Key Data Length - Key Data Length: 128-1024 Increment 8 | SP 800-185 |
| KTS-IFC | A5018 | Modulo - 2048, 3072, 4096, 6144, 8192<br>Key Generation Methods - rsakpg1-basic, rsakpg1-crt, rsakpg1-prime-factor, rsakpg2-basic, rsakpg2-crt, rsakpg2-prime-factor<br>Scheme -<br>KTS-OAEP-basic -<br>KAS Role - initiator, responder<br>Key Transport Method -<br>Key Length - 768 | SP 800-56B Rev. 2 |
| PBKDF | A5018 | Iteration Count - Iteration Count: 1000-10000 Increment 1<br>Password Length - Password Length: 14-128 Increment 1 | SP 800-132 |
| PBKDF | A5020 | Iteration Count - Iteration Count: 1000-10000 Increment 1<br>Password Length - Password Length: 14-128 Increment 1 | SP 800-132 |
| RSA KeyGen (FIPS186-5) | A5018 | Key Generation Mode - probableWithProbableAux<br>Modulo - 2048, 3072, 4096<br>Primality Tests - 2powSecStr<br>Private Key Format - standard | FIPS 186-5 |
| RSA SigGen (FIPS186-5) | A5018 | Modulo - 2048, 3072, 4096<br>Signature Type - pkcs1v1.5, pss | FIPS 186-5 |
| RSA SigVer (FIPS186-5) | A5018 | Modulo - 2048, 3072, 4096<br>Signature Type - pkcs1v1.5, pss | FIPS 186-5 |
| Safe Primes Key Generation | A5014 | Safe Prime Groups - ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192 | SP 800-56A Rev. 3 |
| Safe Primes Key Verification | A5014 | Safe Prime Groups - ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192 | SP 800-56A Rev. 3 |
| SHA-1 | A5018 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-224 | A4711 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-224 | A4712 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-224 | A4716 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-224 | A5018 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-256 | A4711 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-256 | A4712 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-256 | A4716 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-256 | A5018 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-384 | A4711 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-384 | A4712 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-384 | A4716 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-384 | A5018 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-512 | A4711 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-512 | A4712 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |

| Algorithm | CAVP Cert | Properties | Reference |
|---|---|---|---|
| SHA2-512 | A4716 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-512 | A5018 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-512/224 | A5018 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |
| SHA2-512/256 | A5018 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 180-4 |
| SHA3-224 | A4713 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 202 |
| SHA3-224 | A5020 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 202 |
| SHA3-256 | A4713 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 202 |
| SHA3-256 | A5020 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 202 |
| SHA3-384 | A4713 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 202 |
| SHA3-384 | A5020 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 202 |
| SHA3-512 | A4713 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 202 |
| SHA3-512 | A5020 | Message Length - Message Length: 0-65536 Increment 8 | FIPS 202 |
| SHAKE-128 | A5020 | Output Length - Output Length: 16-65536 Increment 8 | FIPS 202 |
| SHAKE-256 | A5020 | Output Length - Output Length: 16-65536 Increment 8 | FIPS 202 |
| TLS v1.2 KDF RFC7627 (CVL) | A5018 | Hash Algorithm - SHA2-256, SHA2-384, SHA2-512 | SP 800-135 Rev. 1 |
| TLS v1.3 KDF (CVL) | A5013 | HMAC Algorithm - SHA2-256, SHA2-384<br>KDF Running Modes - DHE, PSK, PSK-DHE | SP 800-135 Rev. 1 |

Table 6: Approved Algorithms

## Vendor-Affirmed Algorithms:

| Name | Properties | Implementation | Reference |
|---|---|---|---|
| Asymmetric keygen (CKG) | Type:asymmetric | N/A | Section 4 example 1 per SP 800-133rev2 |

Table 7: Vendor-Affirmed Algorithms

## Non-Approved, Allowed Algorithms:

N/A for this module.

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

N/A for this module.

## Non-Approved, Not Allowed Algorithms:

| Name | Use and Function |
|---|---|
| FIPS provider PBKDF with salt length less than 128 bits | Key derivation |

| Name | Use and Function |
|---|---|
| FIPS provider TLSv1.0 and TLSv1.1 KDF using EMS | Key derivation |
| FIPS provider TLSv1.2 KDF without using EMS | Key derivation |
| FIPS provider AES GCM using externally generated IV | Encryption/Decryption |

Table 8: Non-Approved, Not Allowed Algorithms

## 2.6 Security Function Implementations

| Name | Type | Description | Properties | Algorithms |
|---|---|---|---|---|
| Kernel AES-CCM (KTS-Wrap) | KTS-Wrap | Key Unwrapping, Key Unwrapping | Keys: 128, 192, 256-bit keys with 128, 192, 256 bits of key strength, respectively Compliance:Compliant with IG D.G | AES-CCM: (A4712, A4716, A4719, A4721) |
| Kernel AES-GCM (KTS-Wrap) | KTS-Wrap | Key Wrapping, Key Unwrapping | Keys:128, 192, 256-bit keys with 128, 192, 256 bits of key strength, respectively Compliance:Compliant with IG D.G | AES-GCM: (A4712, A4715, A4717, A4719, A4721) |
| Kernel AES CBC with HMAC | KTS-Wrap | Key Wrapping, Key Unwrapping | Keys:128, 192, 256-bit keys with 128, 192, 256 bits of key strength, respectively Compliance:Compliant with IG D.G | AES-CBC: (A4712, A4716, A4719, A4721) HMAC-SHA2-256: (A4711, A4712, A4716) HMAC-SHA2-384: (A4711, A4712, A4716) HMAC-SHA2-512: (A4711, A4712, A4716) |
| Kernel AES CTR with HMAC | KTS-Wrap | Key Wrapping, Key Unwrapping | Keys:128, 192, 256-bit keys with 128, 192, 256 bits of key strength, respectively Compliance:Compliant with IG D.G | AES-CTR: (A4712, A4716, A4719, A4721) HMAC-SHA2-256: (A4711, A4712, A4716) HMAC-SHA2-384: (A4711, A4712, A4716) HMAC-SHA2-512: (A4711, A4712, A4716) |

| Name | Type | Description | Properties | Algorithms |
|---|---|---|---|---|
| Kernel KAS-ECC-SSC | KAS-SSC | Shared Secret Computation | Curves:Curves : P-256, P-384 elliptic curves with 128 and 192 bits of key strength Compliance : Compliant with IG D.F scenario 2(1) | KAS-ECC-SSC Sp800-56Ar3: (A4711) |
| Kernel AES-ECB | BC-UnAuth | Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-ECB: (A4711, A4712, A4715, A4716, A4717, A4719, A4721) |
| Kernel AES-CTR | BC-UnAuth | Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-CTR: (A4712, A4716, A4719, A4721) |
| Kernel AES-CBC | BC-UnAuth | Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-CBC: (A4712, A4716, A4719, A4721) |
| Kernel AES-CBC-CS3 | BC-UnAuth | Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-CBC-CS3: (A4714, A4718, A4720, A4722) |
| Kernel AES-CFB8 | BC-UnAuth | Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-CFB8: (A4724) |
| Kernel AES-CFB128 | BC-UnAuth | Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-CFB128: (A4724) |
| Kernel AES-XTS | BC-UnAuth | Encryption/Decryption | Keys:128, 256 bits with 128 and 256 bits of key strength | AES-XTS Testing Revision 2.0: (A4712, A4716, A4719, A4721) |
| Kernel AES-CCM (BC-Auth) | BC-Auth | Authenticated Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-CCM: (A4712, A4716, A4719, A4721) |
| Kernel AES-GCM (BC-Auth) | BC-Auth | Authenticated Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-GCM: (A4712, A4715, A4717, A4719, A4721) |
| Kernel AES-OFB | BC-Auth | Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-OFB: (A4723) |
| Kernel AES-CMAC | MAC | Message authentication code (MAC) | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-CMAC: (A4712, A4716, A4719, A4721) |

| Name | Type | Description | Properties | Algorithms |
|---|---|---|---|---|
| Kernel AES-GMAC | MAC | Message authentication code (MAC) | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-GMAC: (A4712, A4719, A4721) |
| Kernel Counter DRBG | DRBG | Random Number Generation | Compliance:Compliant with SP800-90ARev1 | Counter DRBG: (A4711, A4712, A4715, A4717, A4719, A4721) |
| Kernel ECDSA Key Generation | CKG | Key Generation | | ECDSA KeyGen (FIPS186-5): (A4711) Asymmetric keygen (CKG): () Type: asymmetric |
| Kernel HMAC | MAC | Message authentication code (MAC) | Keys:112-256 bits with 112-256 bits of key strength | HMAC-SHA2-224: (A4711, A4712, A4716) HMAC-SHA2-256: (A4711, A4712, A4716) HMAC-SHA2-384: (A4711, A4712, A4716) HMAC-SHA2-512: (A4711, A4712, A4716) HMAC-SHA3-224: (A4713) HMAC-SHA3-256: (A4713) HMAC-SHA3-384: (A4713) HMAC-SHA3-512: (A4713) |
| Kernel Hashes | SHA | Hashing | | SHA2-224: (A4711, A4712, A4716) SHA2-256: (A4711, A4712, A4716) SHA2-384: (A4711, A4712, A4716) SHA2-512: (A4711, A4712, |

| Name | Type | Description | Properties | Algorithms |
|---|---|---|---|---|
| | | | | A4716)<br>SHA3-224: (A4713)<br>SHA3-256: (A4713)<br>SHA3-384: (A4713)<br>SHA3-512: (A4713) |
| FIPS provider AES-CCM (KTS-Wrap) | KTS-Wrap | Key Unwrapping, Key Unwrapping | Keys: 128, 192, 256-bit keys with 128, 192, 256 bits of key strength, respectively<br>Compliance:Compliant with IG D.G | AES-CCM: (A5004) |
| FIPS provider AES-GCM (KTS-Wrap) | KTS-Wrap | Key Wrapping, Key Unwrapping | Keys:128, 192, 256-bit keys with 128, 192, 256 bits of key strength, respectively<br>Compliance:Compliant with IG D.G | AES-GCM: (A5008) |
| FIPS provider KAS-IFC-SSC | KAS-SSC | Shared Secret Computation | Keys:2048, 3072, 4096, 6144, 8192-bit keys with 112-200 bits of key strength<br>Compliance : Compliant with IG D.F scenario 1(1) | KAS-IFC-SSC: (A5018) |
| FIPS provider KTS-IFC | KTS-Encap | Key encapsulation, Key unencapsulation | Keys:2048, 3072, 4096, 6144, 8192-bit keys with 112-200 bits of key strength respectively<br>Compliance:Compliant with IG D.G | KTS-IFC: (A5018) |
| FIPS provider Safe Primes Key Generation | CKG | Key Generation | | Safe Primes Key Generation: (A5014)<br>Asymmetric keygen (CKG): ()<br>Type: asymmetric |
| FIPS provider Safe Primes Key Verification | AsymKeyPair-KeyVer | Key Verification | Groups:MODP-2048, MODP-3072, MODP-4096, MODP-6144, | Safe Primes Key Verification: (A5014) |

| Name | Type | Description | Properties | Algorithms |
|---|---|---|---|---|
| | | | MODP-8192, ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192 | |
| FIPS provider KAS-FFC-SSC | KAS-SSC | Shared Secret Computation | Keys:2048, 3072, 4096, 6144, 8192-bit keys with 112-200 bits of key strength Compliance : Compliant with IG D.F scenario 2(1) | KAS-FFC-SSC Sp800-56Ar3: (A5014) |
| FIPS provider KAS-ECC-SSC | KAS-SSC | Shared Secret Computation | Curves:P-224, P-256, P-384, P-521 elliptic curves with 112-256 bits of key strength Compliance : Compliant with IG D.F scenario 2(1) | KAS-ECC-SSC Sp800-56Ar3: (A5018) |
| FIPS provider AES KW | KTS-Wrap | Key Wrapping, Key Unwrapping | Keys:128, 192, 256-bit keys with 128, 192, 256 bits of key strength, respectively Compliance:Compliant with IG D.G | AES-KW: (A5004) |
| FIPS provider AES KWP | KTS-Wrap | Key Wrapping, Key Unwrapping | Keys:128, 192, 256-bit keys with 128, 192, 256 bits of key strength, respectively Compliance:Compliant with IG D.G | AES-KWP: (A5004) |
| FIPS provider AES-ECB | BC-UnAuth | Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-ECB: (A5004, A5019) |
| FIPS provider AES-CTR | BC-UnAuth | Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-CTR: (A5004) |
| FIPS provider AES-CBC | BC-UnAuth | Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-CBC: (A5004) |
| FIPS provider AES-CBC-CS1 | BC-UnAuth | Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-CBC-CS1: (A5004) |
| FIPS provider AES-CBC-CS2 | BC-UnAuth | Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-CBC-CS2: (A5004) |

| Name | Type | Description | Properties | Algorithms |
|---|---|---|---|---|
| FIPS provider AES-CBC-CS3 | BC-UnAuth | Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-CBC-CS3: (A5004) |
| FIPS provider AES-CFB1 | BC-UnAuth | Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-CFB1: (A5004) |
| FIPS provider AES-CFB8 | BC-UnAuth | Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-CFB8: (A5004) |
| FIPS provider AES-CFB128 | BC-UnAuth | Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-CFB128: (A5004) |
| FIPS provider AES-XTS | BC-UnAuth | Encryption/Decryption | Keys:128, 256 bits with 128 and 256 bits of key strength | AES-XTS Testing Revision 2.0: (A5004) |
| FIPS provider AES-CCM (BC-Auth) | BC-Auth | Authenticated Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-CCM: (A5004) |
| FIPS provider AES-GCM (BC-Auth) | BC-Auth | Authenticated Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-GCM: (A5008) |
| FIPS provider AES-OFB | BC-UnAuth | Encryption/Decryption | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-OFB: (A5004) |
| FIPS provider AES-CMAC | MAC | Message authentication code (MAC) | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-CMAC: (A5004) |
| FIPS provider AES-GMAC | MAC | Message authentication code (MAC) | Keys:128, 192, 256 bits with 128-256 bits of key strength | AES-GMAC: (A5008) |
| FIPS provider Counter DRBG | DRBG | Random Number Generation | Compliance:Compliant with SP800-90ARev1 | Counter DRBG: (A5015) |
| FIPS provider Hash DRBG | DRBG | Random Number Generation | Compliance:Compliant with SP800-90ARev1 | Hash DRBG: (A5015) |
| FIPS provider HMAC DRBG | DRBG | Random Number Generation | Compliance:Compliant with SP800-90ARev1 | HMAC DRBG: (A5015) |
| FIPS provider ECDSA Key Generation | CKG | Key Generation | | ECDSA KeyGen (FIPS186-5): (A5018) Asymmetric keygen (CKG): () Type: asymmetric |

| Name | Type | Description | Properties | Algorithms |
|---|---|---|---|---|
| FIPS provider ECDSA Key Verification | AsymKeyPair-KeyVer | Key Verification | Curves:P-224, P-256, P-384, P-521 | ECDSA KeyVer (FIPS186-5): (A5018) |
| FIPS provider ECDSA Signature Generation | DigSig-SigGen | Signature Generation | Curves:P-224, P-256, P-384, P-521 | ECDSA SigGen (FIPS186-5): (A5018, A5020) |
| FIPS provider ECDSA Signature Verification | DigSig-SigVer | Signature Verification | Curves:P-224, P-256, P-384, P-521 | ECDSA SigVer (FIPS186-5): (A5018, A5020) |
| FIPS provider EDDSA Key Generation | CKG | Key Generation | | EDDSA KeyGen: (A5016) Asymmetric keygen (CKG): () Type: asymmetric |
| FIPS provider EDDSA Signature Generation | DigSig-SigGen | Signature Generation | Curves:Ed25519, Ed448 | EDDSA SigGen: (A5016) |
| FIPS provider EDDSA Signature Verification | DigSig-SigVer | Signature Verification | Curves:Ed25519, Ed448 | EDDSA SigVer: (A5016) |
| FIPS provider RSA Key Generation | CKG | Key Generation | | RSA KeyGen (FIPS186-5): (A5018) Asymmetric keygen (CKG): () Type: asymmetric |
| FIPS provider RSA Signature Generation | DigSig-SigGen | Signature Generation | Keys:2048, 3072, 4096 keys with 112-150 bits of key strength respectively | RSA SigGen (FIPS186-5): (A5018) |
| FIPS provider RSA Signature Verification (Legacy) | DigSig-SigVer | Signature Verification using SHA-1 message digest | Keys:2048, 3072, 4096 keys with 112-150 bits of key strength respectively | RSA SigGen (FIPS186-5): (A5018) |
| FIPS provider RSA Signature Verification | DigSig-SigVer | Signature Verification | Keys:2048, 3072, 4096 keys with 112-150 bits of key strength respectively | RSA SigVer (FIPS186-5): (A5018) |

| Name | Type | Description | Properties | Algorithms |
|---|---|---|---|---|
| FIPS provider HMAC | MAC | Message authentication code (MAC) | Keys:112-256 bits with 112-256 bits of key strength | HMAC-SHA-1: (A5018) HMAC-SHA2-224: (A5018) HMAC-SHA2-256: (A5018) HMAC-SHA2-384: (A5018) HMAC-SHA2-512: (A5018) HMAC-SHA2-512/224: (A5018) HMAC-SHA2-512/256: (A5018) HMAC-SHA3-224: (A5020) HMAC-SHA3-256: (A5020) HMAC-SHA3-384: (A5020) HMAC-SHA3-512: (A5020) |
| FIPS provider KMAC | MAC | Message authentication code (MAC) | Keys:112-256 bits with 112-256 bits of key strength | KMAC-128: (A5020) KMAC-256: (A5020) |
| FIPS provider Hashes | SHA | Hashing | | SHA-1: (A5018) SHA2-224: (A5018) SHA2-256: (A5018) SHA2-384: (A5018) SHA2-512: (A5018) SHA2-512/224: (A5018) SHA2-512/256: (A5018) SHA3-224: (A5020) SHA3-256: (A5020) SHA3-384: (A5020) |

| Name | Type | Description | Properties | Algorithms |
|---|---|---|---|---|
| | | | | SHA3-512: (A5020)<br>SHAKE-128: (A5020)<br>SHAKE-256: (A5020) |
| FIPS provider ANS 9.42 Key Derivation (CVL) | KAS-135KDF | Key Derivation | OID:AES-128-KW, AES-192-KW, AES-256-KW with 128, 192, 256 bits of key strength, respectively | KDF ANS 9.42: (A5018, A5020) |
| FIPS provider ANS 9.63 Key Derivation (CVL) | KAS-135KDF | Key Derivation | Key data length:128-4096 bits | KDF ANS 9.63: (A5018) |
| FIPS provider TLS 1.0 and 1.1 Key Derivation (CVL) | KAS-135KDF | Key Derivation | Derived key:112-256 bits with 112-256 bits of key strength | KDF TLS: (A5018) |
| FIPS provider TLS 1.2 Key Derivation (CVL) | KAS-135KDF | Key Derivation | Derived key:112-256 bits with 112-256 bits of key strength | TLS v1.2 KDF RFC7627: (A5018) |
| FIPS provider TLS 1.3 Key Derivation (CVL) | KAS-135KDF | Key Derivation | Derived key:112-256 bits with 112-256 bits of key strength | TLS v1.3 KDF: (A5013) |
| FIPS provider HKDF Key Derivation | KAS-56CKDF | Key Derivation | Derived key:112-256 bits with 112-256 bits of key strength | KDA HKDF Sp800-56Cr1: (A5013) |
| FIPS provider Password-based Key Derivation | PBKDF | Key Derivation | Derived key:112-4096 bits with 112-150 bits of key strength | PBKDF: (A5018, A5020) |
| FIPS provider OneStep Key Derivation | KAS-56CKDF | Key Derivation | Derived key:2048 bits with 112 bits of key strength | KDA OneStep SP800-56Cr2: (A5012) |
| FIPS provider TwoStep Key Derivation | KAS-56CKDF | Key Derivation | Derived key:2048 bits with 112 bits of key strength | KDA TwoStep SP800-56Cr2: (A5012) |
| FIPS provider KMAC Key Derivation | KBKDF | Key Derivation | Derived key:112-4096 bits with 112-150 bits of key strength | KDF KMAC Sp800-108r1: (A5017) |
| FIPS provider KBKDF Key Derivation | KBKDF | Key Derivation. | Derived key:112-4096 bits with 112-150 bits of key strength | KDF SP800-108: (A5017) |

| Name | Type | Description | Properties | Algorithms |
|---|---|---|---|---|
| FIPS provider SSH Key Derivation | KAS-135KDF | Key Derivation | Keys:128, 192, 256 bits with 128-256 bits of key strength | KDF SSH: (A5019) |

Table 9: Security Function Implementations

## 2.7 Algorithm Specific Information

### 2.7.1 AES GCM IV

AES-GCM encryption and decryption are used in the context of the TLS protocol version 1.2 and 1.3 using the FIPS provider component (corresponding to Scenario 1 and 5 of IG C.H), and in the context of IEEE 802.11 GCMP using the kernel/hardware components (corresponding to Scenario 5 of IG C.H). For IPsec, the module offers the AES GCM implementation and uses the context of Scenario 1 of FIPS 140-3 IG C.H. The mechanism for IV generation is compliant with RFC 4106. IVs generated using this mechanism may only be used in the context of AES GCM encryption within the IPsec protocol.

Alternatively, the Crypto Officer can use the module's API to perform AES GCM encryption using internal IV generation. These IVs are always 96 bits and generated using the approved DRBG internal to the module's boundary, compliant with Scenario 2 of FIPS 140-3 IG C.H.

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

#### 2.7.1.1 TLS version 1.2

For TLS v1.2, the module uses the context of Scenario 1 of IG C.H. The module is compliant with SP 800-52rev2 section 3.3.1, and the mechanism for IV generation is compliant with RFC5288. For this compliance, the module's implementation of the AES-GCM shall be used together with an application that negotiates the protocol session's keys and the 32-bit nonce value of the IV. The setting of the counter portion of the IV is performed within the cryptographic boundary.

The nonce explicit part of the IV does not exhaust the maximum number of possible values for a given session key. This condition is implicitly ensured by the design of the TLS protocol, in which the nonce_explicit is denied exhaustion by the control exerted by the protocol's management logic (wherein the nonce_explicit is incremented per each TLS record). This management logic also implies that the probability of an exhaustion of all $2^{64} - 1$ values of the nonce_explicit for the same TLS session in a realistic time frame is not significant.

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

#### 2.7.1.2 TLS version 1.3

For TLS 1.3, the AES GCM implementation uses the context of Scenario 5 of FIPS 140-3 IG C.H. The protocol that provides this compliance is TLS 1.3, defined in RFC8446 of August 2018, using the cipher-suites that

explicitly select AES GCM as the encryption/decryption cipher (Appendix B.4 of RFC8446). The module supports acceptable AES GCM cipher suites from Section 3.3.1 of SP800-52r2. TLS 1.3 employs separate 64-bit sequence numbers, one for protocol records that are received, and one for protocol records that are sent to a peer. These sequence numbers are set at zero at the beginning of a TLS 1.3 connection and each time when the AES-GCM key is changed. After reading or writing a record, the respective sequence number is incremented by one. The protocol specification determines that the sequence number should not wrap, and if this condition is observed, then the protocol implementation must either trigger a re-key of the session (i.e., a new key for AES-GCM) or terminate the connection. The module implements, within its boundary, an IV generation unit for TLS 1.3 that keeps control of the 64-bit counter value within the AES-GCM IV.

The module explicitly ensures that the 64-bit counter is monotonically increasing at each invocation of the AES-GCM for the same encryption key, and that this counter does not exhaust all its possible values. If this exhaustion condition is observed, the module will return an error indication to the calling application who will then need to either trigger a re-key of the session (i.e., a new key for AES-GCM) or terminate the connection. The module will refuse a new AES-GCM encryption for the same key and IV under this scenario.

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

#### 2.7.1.3 IEEE 802.11 GCMP

The kernel component is in compliance with FIPS 140-3 IG C.H scenario 5 for the WPA2 protocol. Specifically, GCMP is defined in IEEE 802.11ac-2013. For IEEE 802.11 GCMP, the module implements an internal production unit logic that constructs the IV deterministically upon the initialization of a GCMP connection, and therefore the initialization of a GCM encryption context.

In case the module's power is lost and then restored, the key used for AES GCM encryption or decryption shall be re-distributed.

### 2.7.2 AES XTS

The length of a single data unit encrypted or decrypted with AES XTS shall not exceed $2^{20}$ AES blocks, that is 16MB, of data per XTS instance. An XTS instance is defined in Section 4 of SP 800-38E. To meet the requirement stated in IG C.I, the module implements a check to ensure that the two AES keys used in AES XTS mode are not identical.

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

### 2.7.3 Key derivation using SP 800-132 PBKDF2

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

- Derived keys shall only be used in storage applications. The MK shall not be used for other purposes. The length of the MK or DPK shall be of 112 bits or more.
- Passwords or passphrases, used as an input for the PBKDF2, shall not be used as cryptographic keys.
- The length of the password or passphrase shall be at least 8 characters. The probability of guessing the value, assuming a worst-case scenario of all digits, is estimated to be at most $10^{-8}$. Combined with the minimum iteration count as described below, this provides an acceptable trade-off between user experience and security against brute-force attacks.
- A portion of the salt, with a length of at least 128 bits, shall be generated randomly using the SP 800-90Ar1 DRBG provided by the module.
- The iteration count shall be selected as large as possible, as long as the time required to generate the key using the entered password is acceptable for the users. The minimum value is 1000.

### 2.7.4 SP 800-56Ar3 Assurances

**Kernel Component:** The module offers ECDH shared secret computation services compliant to the SP 800-56ARev3. In order to meet the required assurances listed in section 5.6 of SP 800-56ARev3, the module shall be used together with an application that implements the "IPsec protocol" and the following steps shall be performed.

The entity using the module, must use the module's "key pair generation" service for generating ECDH ephemeral keys. The key generation service performs full public key validation. This meets the assurances required by key pair owner defined in the section 5.6.2.1 of SP 800-56ARev3.
The consumer using the module doesn't need to obtain assurance of the peer's possession of private key as the module only makes use of ephemeral keys.
As part of the module's shared secret computation service, the module internally performs the public key validation on the peer's public key passed in as input to the SSC function. This meets the public key validity assurance required by the sections 5.6.2.2.1/5.6.2.2.2 of SP 800-56ARev3.

**FIPS provider Component:** The module offers DH and ECDH shared secret computation services compliant to the SP 800-56ARev3. To comply with the assurances found in Section 5.6.2 of SP 800-56Ar3, the operator must use the module together with an application that implements the TLS protocol. Additionally, the module's approved key pair generation service must be used to generate ephemeral Diffie-Hellman or EC Diffie-Hellman key pairs, or the key pairs must be obtained from another FIPS-validated module. As part of this service, the module will internally perform the full public key validation of the generated public key. The module's shared secret computation service will internally perform the full public key validation of the peer public key, complying with Sections 5.6.2.2.1 and 5.6.2.2.2 of SP 800-56Ar3.

### 2.7.5 RSA Key Encapsulation

To comply with SP800-56Br2 assurances found in its Section 6 (specifically SP800-56Br2 Section 6.4 Required Assurances) the entity using the module must obtain required assurances listed in section 6.4 of SP 800-56Br2 by performing the following steps:

1. The entity requesting the RSA key un-encapsulation service from the module, shall only use an RSA private key that was generated by an active FIPS validated module that implements FIPS 186-5 compliant RSA key generation service and performs the key pair validity and the pairwise consistency as stated in

section 6.4.1.1 of the SP 800-56Br2. Additionally, the entity shall renew these assurances over time by using any method described in section 6.4.1.5 of the SP 800-56Br2.

2. For use of an RSA key encapsulation service in the context of key transport per IG D.G the entity using the module shall:
    a. verify the validity of the peer's public key using the public key validation service of the module (EVP_PKEY_check() API).
    b. confirm the peer's possession of private key by using any method specified in section 6.4.2.3 of the SP 800-56Br2.

Only after the above assurances are successfully met, shall the entity use the peer's public key to perform the RSA key encapsulation service of the module.

### 2.7.6 RSA Key Agreement

To comply with the assurances found in Section 6.4 of SP 800-56Br2, the module's approved RSA key pair generation service must be used to generate the RSA key pairs, or the key pairs must be obtained from another FIPS-validated module. As part of this service, the module will internally perform the key pair validity and the pairwise consistency according to section 6.4.1.1 of SP 800-56Br2.

Additionally, the entity requesting the shared secret computation service shall verify the validity of the peer's public key using the public key validation service of the module (EVP_PKEY_check() API). This service will perform the full public key validation of the peer's public key, complying with Section 6.4.2.1 of SP 800-56Br2.

### 2.7.7 RSA SigGen and SigVer compliance

The module provides RSA signature generation and signature verification compliant with IG C.F.

The module supports RSA modulus lengths of 2048, 3072, and 4096 bits for signature generation and 1024, 2048, 3072, and 4096 for signature verification.

The RSA signature generation and signature verification implementations have been tested for all implemented RSA modulus lengths.

The number of Miller-Rabin tests is consistent with the bit sizes of p and q from Table B.1 of FIPS 186-4.

### 2.7.8 SHA-3 compliance

The module provides SHA-3 and SHAKE hash functions compliant with IG C.C. Every implementation of each SHA-3 and SHAKE functions were tested and validated on all of the module's operating environments. SHA-3 hash functions are also used as part of a higher-level algorithm for HMAC. SHAKE functions are only used a standalone algorithms.

### 2.7.9 SHA-1 compliance to SP 800-131A rev2

SHA-1 from FIPS provider Message Digest service is only approved for non-digital-signature uses. SHA-1 used within Digital Signature Verification is considered Legacy (approved) per IG C.M. Algorithms designated as "Legacy" can only be used on data that was generated prior to the Legacy Date specified in FIPS 140-3 IG C.M.

## 2.8 RBG and Entropy

| Cert Number | Vendor Name |
|---|---|
| E119 | Ezurio |

Table 10: Entropy Certificates

| Name | Type | Operational Environment | Sample Size | Entropy per Sample | Conditioning Component |
|---|---|---|---|---|---|
| Summit CPU Time Jitter RNG Entropy Source | Non-Physical | Summit Linux 11.1 on Microchip SAMA5D3 ATSAMA5D31 and Linux 11.1 on Microchip SAMA5D3 ATSAMA5D36 | 256 bits | full entropy | A4713 (SHA3-256) |

Table 11: Entropy Sources

The module implements multiple DRBGs compliant with SP800-90A for random number generation and the creation of key components of asymmetric keys. The kernel component of the module implements a CTR_DRBG while the FIPS provider component of the module implements a CTR_DRBG, Hash_DRBG and HMAC_DRBG. Each of these DRBG is seeded with full entropy using an entropy source listed in the above table. For internal usage, module uses an SP800-90Ar1 CTR_DRBG with AES-256 as the default DRBG in both the Kernel and the FIPS Provider components.

Note: Per FIPS 140-3 IG C.L please make sure to select the appropriate hash function when instantiating HMAC or Hash DRBG based on the minimum-security strength required for the generated random bits.

## 2.9 Key Generation

For generating RSA, ECDSA, Diffie-Hellman, EC Diffie-Hellman keys for the FIPS provider component and ECDSA keys for Kernel component, the module implements asymmetric key generation services compliant with FIPS186-5 or SP800-56Arev3 as applicable and using a DRBG compliant with SP800-90A. The random value used in asymmetric key generation is obtained from the DRBG. In accordance with FIPS 140-3 IG D.H, the cryptographic module performs Cryptographic Key Generation (CKG) for asymmetric keys as per Section 4 of SP800-133rev2 (vendor affirmed).

Additionally, the module implements the following key derivation methods according to section 6.2 of SP 800-133r2:

- HKDF compliant with SP 800-56Cr1: derivation of secret keys in the context of SP 800-56Ar3 and SP 800-56Br2 key agreement schemes.

- TLS KDF compliant with SP 800-135r1: derivation of secret keys in the context of TLS 1.0/1.1, TLS 1.2, TLS 1.3.
- PBKDF2: compliant with option 1a of SP 800-132: derivation of keys for use in storage applications.
- ANS X9.42 KDF compliant with SP 800-135r1: derivation of secret keys in the context of ANS X9.42-2001 key agreement schemes
- ANS X9.63 KDF compliant with SP 800-135r1: derivation of secret keys in the context of ANS X9.63-2001 key agreement schemes.
- OneStep KDF compliant with SP 800-56Cr2: derivation of secret keys in the context of SP 800-56Ar3 and SP 800-56Br2 key agreement schemes.
- TwoStep KDF compliant with SP 800-56Cr2: derivation of secret keys in the context of SP 800-56Ar3 and SP 800-56Br2 key agreement schemes.
- KBKDF compliant with SP800-108r1: derivation of secret keys
- SSH KDF compliant with SP 800-135r1: derivation of secret keys in the context of SSH.

## 2.10 Key Establishment

The module implements following key establishments methods that are listed in the Security Function Implementations table:
- shared secret computation for KAS-IFC-SSC, KAS-FFC-SSC KAS-ECC-SSC
- key transport for KTS-IFC and KTS-Wrap

## 2.11 Industry Protocols

Only the Key Derivation Functions have been validated by the CAVP No other part of the SSH, IKE or TLS protocols are implemented or have been tested by the CAVP and CMVP.

For DH, the module supports the use of the safe primes defined in RFC 3526 (IKE) and RFC 7919 (TLS) as listed in Approved Services table. Note that the module only implements key pair generation, key pair verification, and shared secret computation.

SSH KDF, TLS 1.0/1.1 KDF, TLS 1.2 KDF (RFC 7627), TLS 1.3 KDF implementations shall only be used to generate secret keys in the context of the SSH, TLS 1.0/1.1, TLS 1.2, or TLS 1.3 protocols, respectively. Note that TLS 1.2 KDF must be compliant with RFC 7627 to be considered approved.

ANS X9.42 KDF and ANS X9.63 KDF implementations shall only be used to generate secret keys in the context of an ANS X9.42-2001 resp. ANS X9.63-2001 key agreement scheme.

# 3 Cryptographic Module Interfaces

## 3.1 Ports and Interfaces

| Physical Port | Logical Interface(s) | Data That Passes |
|---|---|---|
| N/A | Data Input | API data input parameters, AF_ALG type sockets (kernel component) |
| N/A | Data Output | API output parameters, AF_ALG type sockets (kernel component) |
| N/A | Control Input | API function calls, API control input parameters, AF_ALG type sockets (kernel component), kernel command line (kernel component) |
| N/A | Status Output | API return values, error queue (FIPS provider component), AF_ALG type sockets (kernel component), kernel logs (kernel component) |
| N/A | Power | The hardware component of the module receives power from the circuit board on which the module is installed. The power input is not applicable for the firmware components. |

Table 12: Ports and Interfaces

The logical interfaces are the APIs through which the applications request services. These logical interfaces are logically separated from each other by the API design.

# 4 Roles, Services, and Authentication

## 4.1 Roles

| Name | Type | Operator Type | Authentication Methods |
|---|---|---|---|
| Crypto Officer | Role | Crypto Officer | None |

Table 13: Roles

The module supports the Crypto Officer role only. This sole role is implicitly and always assumed by the operator of the module. No support is provided for multiple concurrent operators.

## 4.2 Approved Services

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| Kernel Encryption | Encryption | crypto_skcipher_setkey returns 0 | AES key, plaintext | ciphertext | Kernel AES-ECB<br>Kernel AES-CTR<br>Kernel AES-CBC<br>Kernel AES-CBC-CS3<br>Kernel AES-CFB8<br>Kernel AES-CFB128<br>Kernel AES-XTS<br>Kernel AES-OFB | Crypto Officer - Kernel AES key: W,E |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| Kernel Decryption | Decryption | crypto_skcipher_setkey returns 0 | AES key, ciphertext | plaintext | Kernel AES-ECB Kernel AES-CTR Kernel AES-CBC Kernel AES-CBC-CS3 Kernel AES-CFB8 Kernel AES-CFB128 Kernel AES-XTS Kernel AES-OFB | Crypto Officer - Kernel AES key: W,E |
| Kernel Authenticated Encryption | Encryption | crypto_aead_setkey returns 0 | AES key, IV, plaintext | ciphertext | Kernel AES-CCM (BC-Auth) Kernel AES-GCM (BC-Auth) | Crypto Officer - Kernel AES key: W,E |
| Kernel Authenticated Decryption | Decryption | crypto_aead_setkey returns 0 | AES key, IV, MAC tag, ciphertext | plaintext | Kernel AES-CCM (BC-Auth) Kernel AES-GCM | Crypto Officer - Kernel AES key: W,E |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| | | | | | (BC-Auth) | |
| Kernel key wrapping | Wrap a key | crypto_skcipher_setkey returns 0; crypto_aead_setkey returns 0; crypto_shash_init returns 0 | AES key, key to be wrapped | wrapped key | Kernel AES-CCM (KTS-Wrap) Kernel AES-GCM (KTS-Wrap) Kernel AES CBC with HMAC Kernel AES CTR with HMAC | Crypto Officer - Kernel AES key: W,E |
| Kernel key unwrapping | unwrap a key | crypto_skcipher_setkey returns 0; crypto_aead_setkey returns 0; crypto_shash_init returns 0 | AES key, key to be unwrapped | unwrapped key | Kernel AES-CCM (KTS-Wrap) Kernel AES-GCM (KTS-Wrap) Kernel AES CBC with HMAC Kernel AES CTR with HMAC | Crypto Officer - Kernel AES key: W,E |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| Kernel AES Message Authentication | compute a MAC tag | crypto_shash_init returns 0 | AES key, message | MAC tag | Kernel AES-CMAC Kernel AES-GMAC | Crypto Officer - Kernel AES key: W,E |
| Kernel HMAC Message Authentication | compute a MAC tag | crypto_shash_init returns 0 | HMAC key, message | MAC tag | Kernel HMAC | Crypto Officer - Kernel HMAC key: W,E |
| Kernel Message Digest | compute a message digest | crypto_shash_init returns 0 | message | digest value | Kernel Hashes | Crypto Officer |
| Kernel ECC Shared Secret Computation | compute a shared secret | crypto_kpp_compute_shared_secret returns 0 | EC public key, EC private key | Shared Secret | Kernel KAS-ECC-SSC | Crypto Officer - Kernel EC public key: W,E - Kernel EC private key: W,E - Kernel shared secret: W,E |
| Kernel Random Number Generation | generate random bytes | crypto_rng_get_bytes returns 0 | output length | random data | Kernel Counter DRBG | Crypto Officer - Entropy input: W,E - DRBG seed: G,E - Internal state (V, C): G,W,E |
| Kernel EC Key generation | generate key pair | crypto_kpp_set_secret and crypto_kpp_generate_public_key return 0 | Curve | EC keys | Kernel ECDSA Key Generation | Crypto Officer - Kernel EC public key: G,R - Kernel EC private key: G,R - Kernel |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| | | | | | | Intermediate Key Generation Value: G,E,Z |
| FIPS provider Message Digest | compute a message digest | _SUMMIT_FIPS_INDICATOR_APPROVED | message | digest value | FIPS provider Hashes | Crypto Officer |
| FIPS provider Encryption | Encrypt plaintext | _SUMMIT_FIPS_INDICATOR_APPROVED | AES key, plaintext | ciphertext | FIPS provider AES-CTR FIPS provider AES-CBC FIPS provider AES-ECB FIPS provider AES-CBC-CS1 FIPS provider AES-CBC-CS2 FIPS provider AES-CBC-CS3 FIPS provider AES-CFB1 FIPS provider AES- | Crypto Officer - FIPS provider AES Key: W,E |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| | | | | | CFB8 FIPS provider AES-CFB128 FIPS provider AES-XTS FIPS provider AES-OFB | |
| FIPS provider Decryption | Decrypt ciphertext | _SUMMIT_FIPS_INDICATOR_APPROVED | AES key, ciphertext | plaintext | FIPS provider AES-CTR FIPS provider AES-CBC FIPS provider AES-ECB FIPS provider AES-CBC-CS1 FIPS provider AES-CBC-CS2 FIPS provider AES-CBC-CS3 FIPS provider AES- | Crypto Officer - FIPS provider AES Key: W,E |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| | | | | | CFB1 FIPS provider AES-CFB8 FIPS provider AES-CFB128 FIPS provider AES-XTS FIPS provider AES-OFB | |
| FIPS provider Authenticated Encryption | Encrypt plaintext | _SUMMIT_FIPS_INDICATOR_APPROVED | AES key, IV, plaintext | ciphertext | FIPS provider AES-CCM (BC-Auth) FIPS provider AES-GCM (BC-Auth) | Crypto Officer - FIPS provider AES Key: W,E |
| FIPS provider Authenticated Decryption | Decrypt ciphertext | _SUMMIT_FIPS_INDICATOR_APPROVED | AES key, IV, MAC tag, ciphertext | plaintext | FIPS provider AES-CCM (BC-Auth) FIPS provider AES-GCM (BC-Auth) | Crypto Officer - FIPS provider AES Key: W,E |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| FIPS provider AES Message Authentication | compute a MAC tag | _SUMMIT_FIPS_INDICATOR_APPROVED | AES key, message | MAC tag | FIPS provider AES-CMAC FIPS provider AES-GMAC | Crypto Officer - FIPS provider AES Key: W,E |
| FIPS provider HMAC Message Authentication | compute a MAC tag | _SUMMIT_FIPS_INDICATOR_APPROVED | HMAC key, message | MAC tag | FIPS provider HMAC | Crypto Officer - FIPS provider HMAC key: W,E |
| FIPS provider FFC Shared Secret Computation | compute a shared secret | _SUMMIT_FIPS_INDICATOR_APPROVED | DH private key, DH public key | Shared Secret | FIPS provider KAS-FFC-SSC | Crypto Officer - FIPS provider DH public key: W,E - FIPS provider DH private key: W,E |
| FIPS provider ECC Shared Secret Computation | compute a shared secret | _SUMMIT_FIPS_INDICATOR_APPROVED | EC public key, EC private key | Shared Secret | FIPS provider KAS-ECC-SSC | Crypto Officer - FIPS provider EC public key: W,E - FIPS provider EC private key: W,E - FIPS provider shared secret: W,E |
| FIPS provider IFC Shared Secret | compute a shared secret | _SUMMIT_FIPS_INDICATOR_APPROVED | RSA public key, RSA private key | Shared Secret | FIPS provider KAS-IFC-SSC | Crypto Officer - FIPS provider RSA public |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| Computation | | | | | | key: W,E<br>- FIPS provider<br>RSA private key: W,E<br>- FIPS provider<br>shared secret: W,E |
| FIPS provider Key Derivation | derive a key | _SUMMIT_FIPS_INDICATOR_APPROVED | Shared secret | derived key | FIPS provider ANS 9.42 Key Derivation (CVL)<br>FIPS provider ANS 9.63 Key Derivation (CVL)<br>FIPS provider HKDF Key Derivation<br>FIPS provider OneStep Key Derivation<br>FIPS provider TwoStep Key Derivation | Crypto Officer<br>- FIPS provider<br>shared secret: W,E<br>- FIPS provider<br>derived key: G,R<br>- FIPS provider<br>key-derivation key: W,E<br>- FIPS provider<br>AES Derived Key: G,R<br>- FIPS provider<br>HMAC Derived Key: G,R<br>- FIPS provider<br>802.11 Pre-shared key (PSK): W,E<br>- FIPS provider<br>802.11 Pairwise |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| | | | | | FIPS provider KMAC Key Derivation FIPS provider KBKDF Key Derivation FIPS provider SSH Key Derivation | Master Key (PMK): W,E<br>- FIPS provider 802.11 KDF Internal State: R<br>- FIPS provider 802.11 Temporal Keys: W,E<br>- FIPS provider 802.11 MIC keys (KCK): W,E<br>- FIPS provider 802.11 Key Encryption Key (KEK): W,E<br>- FIPS provider 802.11 Group Temporal Key (GTK): W,E |
| FIPS provider Key Derivation (FIPS provider TLS master secret) | derive a TLS master secret | _SUMMIT_FIPS_INDICATOR_APPROVED | FIPS provider TLS pre-master secret | | FIPS provider TLS 1.0 and 1.1 Key Derivation (CVL) FIPS provider TLS 1.2 Key | Crypto Officer<br>- FIPS provider TLS pre-master secret: W,E<br>- FIPS provider TLS master secret: G,R |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| | | | | | Derivation (CVL) FIPS provider TLS 1.3 Key Derivation (CVL) | |
| FIPS provider Key Derivation (FIPS provider derived key) | derive a key used for session establishment | _SUMMIT_FIPS_INDICATOR_APPROVED | FIPS provider TLS master secret | FIPS provider derived key | FIPS provider TLS 1.0 and 1.1 Key Derivation (CVL) FIPS provider TLS 1.2 Key Derivation (CVL) FIPS provider TLS 1.3 Key Derivation (CVL) | Crypto Officer - FIPS provider TLS master secret: W,E - FIPS provider derived key: G,R |
| FIPS provider Password-based key derivation | derive a key from a password | _SUMMIT_FIPS_INDICATOR_APPROVED | password | derived key | FIPS provider Password-based Key Derivation | Crypto Officer - FIPS provider derived key: G,R - FIPS provider Password: W,E |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| FIPS provider SafePrime key generation | generate a key pair | _SUMMIT_FIPS_INDICATOR_APPROVED | DH-Group | Module generated Dh private key, Module generated DH public key | FIPS provider Safe Primes Key Generation | Crypto Officer - FIPS provider module generated DH public key: G,R - FIPS provider module generated DH private key: G,R - FIPS provider Intermediate Key Generation Value: G,R |
| FIPS provider EC Key generation | generate a key pair | _SUMMIT_FIPS_INDICATOR_APPROVED | Curve | Module Generated EC Private Key, Module Generated EC Public Key | FIPS provider ECDSA Key Generation FIPS provider EDDSA Key Generation | Crypto Officer - FIPS provider module generated EC public key: G,R - FIPS provider module generated EC private key: G,R - FIPS provider Intermediate Key Generation Value: G,R |
| FIPS provider | generate a key pair | _SUMMIT_FIPS_INDICATOR_APPROVED | Modulus | Module Generated RSA | FIPS provider RSA | Crypto Officer - FIPS |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| RSA key generation | | | | Private Key, Module Generated RSA Public Key | Key Generation | provider module generated RSA private key: G,R<br>- FIPS provider module generated RSA public key: G,R<br>- FIPS provider Intermediate Key Generation Value: G,R |
| FIPS provider SafePrime Key Verification | verify key pair | _SUMMIT_FIPS_INDICATOR_APPROVED | DH Private key, DH public key | Pass/fail | FIPS provider Safe Primes Key Verification | Crypto Officer<br>- FIPS provider DH public key: W<br>- FIPS provider DH private key: W |
| FIPS provider EC Key Verification | verify key pair | _SUMMIT_FIPS_INDICATOR_APPROVED | EC public key, EC private key | Pass/fail | FIPS provider ECDSA Key Verification | Crypto Officer<br>- FIPS provider EC public key: W<br>- FIPS provider EC private key: W |
| FIPS provider Key wrapping | wrap a key | _SUMMIT_FIPS_INDICATOR_APPROVED | AES key, key to be wrapped | wrapped key | FIPS provider AES-CCM (KTS-Wrap) | Crypto Officer<br>- FIPS provider AES Key: W,E |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| | | | | | FIPS provider AES-GCM (KTS-Wrap) FIPS provider AES KW FIPS provider AES KWP | |
| FIPS provider Key unwrapping | unwrap a key | _SUMMIT_FIPS_INDICATOR_APPROVED | AES key, key to be unwrapped | unwrapped key | FIPS provider AES-CCM (KTS-Wrap) FIPS provider AES-GCM (KTS-Wrap) FIPS provider AES KW FIPS provider AES KWP | Crypto Officer - FIPS provider AES Key: W,E |
| FIPS provider RSA Signature Verification | verify digital signature | _SUMMIT_FIPS_INDICATOR_APPROVED | RSA public key, signature, hash algorithm | Pass/fail | FIPS provider RSA Signature Verification | Crypto Officer - FIPS provider RSA public key: W,E |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| FIPS provider EC Signature Verification | verify digital signature | _SUMMIT_FIPS_INDICATOR_APPROVED | Message, EC public key, signature, hash algorithm (ECDSA only) | Pass/fail | FIPS provider ECDSA Signature Verification FIPS provider EDDSA Signature Verification | Crypto Officer - FIPS provider EC public key: W,E |
| FIPS provider EC Signature Generation | generate digital signature | _SUMMIT_FIPS_INDICATOR_APPROVED | Message, EC public key, signature, hash algorithm (ECDSA only) | signature | FIPS provider ECDSA Signature Generation FIPS provider EDDSA Signature Generation | Crypto Officer - FIPS provider EC private key: W,E |
| FIPS provider RSA Signature Generation | generate digital signature | _SUMMIT_FIPS_INDICATOR_APPROVED | Message, RSA public key, signature, hash algorithm | signature | FIPS provider RSA Signature Generation | Crypto Officer - FIPS provider RSA private key: W,E |
| FIPS provider RSA Signature Verification (Legacy) | verify a digital signature using SHA-1 message digest | _SUMMIT_FIPS_INDICATOR_APPROVED | Message, RSA public key, signature, hash | signature | FIPS provider RSA Signature Verifica | Crypto Officer - FIPS provider RSA public key: W,E |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| | | | algorithm | | tion (Legacy) | |
| FIPS provider Random Number Generation | generate random bytes | _SUMMIT_FIPS_INDICATOR_APPROVED | output length | random bytes | FIPS provider Counter DRBG FIPS provider Hash DRBG FIPS provider HMAC DRBG | Crypto Officer - Entropy input: W,E - DRBG seed: G,E - Internal State (V, Key): G,W,E - Internal state (V, C): G,W,E |
| FIPS provider key encapsulation | KTS | _SUMMIT_FIPS_INDICATOR_APPROVED | RSA public key | Encapsulated key | FIPS provider KTS-IFC | Crypto Officer - FIPS provider RSA public key: W,E |
| FIPS provider key decapsulation | KTS | _SUMMIT_FIPS_INDICATOR_APPROVED | RSA private key | Decapsulated key | FIPS provider KTS-IFC | Crypto Officer - FIPS provider RSA private key: W,E |
| FIPS provider KMAC Message Authentication | MAC | _SUMMIT_FIPS_INDICATOR_APPROVED | KMAC key | Mac tag | FIPS provider KMAC | Crypto Officer |
| Show version | Return the module name and version information | None | N/A | module name and version | None | Unauthenticated |
| Show status | return module status | None | N/A | module status | None | Unauthenticated |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| Self-Tests | perform CASTs and integrity test | None | N/A | Pass/Fail | None | Unauthenticated |
| Zeroization | zeroize all SSPs | None | Any SSP | N/A | None | Crypto Officer<br>- Kernel AES key: Z<br>- FIPS provider AES Key: Z<br>- Kernel HMAC key: Z<br>- Kernel shared secret: Z<br>- FIPS provider shared secret: Z<br>- Entropy input: Z<br>- DRBG seed: Z<br>- Internal State (V, Key): Z<br>- Internal state (V, C): Z<br>- FIPS provider DH public key: Z<br>- FIPS provider DH private key: Z<br>- Kernel EC public key: Z |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| | | | | | | - Kernel EC private key: Z<br>- FIPS provider EC public key: Z<br>- FIPS provider EC private key: Z<br>- FIPS provider module generated DH public key: Z<br>- FIPS provider module generated DH private key: Z<br>- FIPS provider module generated EC public key: Z<br>- FIPS provider RSA public key: Z<br>- FIPS provider RSA private key: Z<br>- FIPS provider module generated RSA public key: Z |

| Name | Description | Indicator | Inputs | Outputs | Security Functions | SSP Access |
|---|---|---|---|---|---|---|
| | | | | | | - FIPS provider module generated RSA private key: Z<br>- FIPS provider Intermediate Key Generation Value: Z<br>- FIPS provider derived key: Z<br>- Kernel Intermediate Key Generation Value: Z<br>- FIPS provider key-derivation key: Z<br>- FIPS provider Password: Z |

Table 14: Approved Services

The table above lists the approved services. The following convention is used to specify access rights to SSPs:
**Generate (G):** The module generates or derives the SSP.
**Read (R):** The SSP is read from the module (e.g. the SSP is output).
**Write (W):** The SSP is updated, imported, or written to the module.
**Execute (E):** The module uses the SSP in performing a cryptographic operation.
**Zeroize (Z):** The module zeroizes the SSP.

To interact with the FIPS provider component of the module, a calling application must use the EVP API layer provided by OpenSSL. This layer will delegate the request to the FIPS provider, which will in turn perform the requested service. The EVP_KDF_CTX_get_params() function can be used to determine whether an EVP API

function is approved. After a cryptographic service was performed by the module, the API context associated with this request can contain a parameter (listed below) which represents the approved service indicator.

- _SUMMIT_FIPS_INDICATOR_APPROVED
- _SUMMIT_FIPS_INDICATOR_NOT_APPROVED

The security function implementation “FIPS provider KBKDF Key Derivation” listed for the “FIPS provider Key Derivation” approved service in the table above is intended to derive keys for use of 802.11 protocols.

## 4.3 Non-Approved Services

| Name | Description | Algorithms | Role |
|---|---|---|---|
| FIPS provider PBKDF with salt length less than 128 bits | Key derivation | FIPS provider PBKDF with salt length less than 128 bits | CO |
| FIPS provider TLSv1.0 and TLSv1.1 KDF using EMS | Key derivation | FIPS provider TLSv1.0 and TLSv1.1 KDF using EMS | CO |
| FIPS provider TLSv1.2 KDF without using EMS | Key derivation | FIPS provider TLSv1.2 KDF without using EMS | CO |
| FIPS provider AES-GCM using EVP_EncryptInit_ex2 | Encryption/Decryption using AES-GCM and externally generated IV | FIPS provider AES GCM using externally generated IV | CO |

Table 15: Non-Approved Services

# 5 Software/Firmware Security

## 5.1 Integrity Techniques

The integrity of the module's firmware components (the kernel, the FIPS provider components and fipscheck application and library) is individually verified by the fipscheck integrity test tool using an HMAC-SHA2-256 implemented by the FIPS provider. The HMAC value of each firmware component is computed at build time and stored in the .hmac file for each component. The value is recalculated at runtime for the image of the kernel, for the FIPS provider binary and the fipscheck application and library, and then compared against the stored value in the file. If the comparison succeeds, then the remaining Known Answer Tests (KATs) for FIPS provider are performed. Then the kernel component executes its algorithm-specific Known Answer Tests. If the integrity test fails the module will enter the error state. Please see section 10.4 for details

## 5.2 Initiate on Demand

Integrity tests are performed as part of the pre-operational self-tests, which are executed when the module is initialized. The integrity tests can be invoked on demand by unloading and subsequently re-initializing the module, which will perform (among others) the firmware integrity tests. The Self-Tests service can also be used to invoke the integrity test on-demand.

# 6 Operational Environment

## 6.1 Operational Environment Type and Requirements

**Type of Operational Environment**: Limited

**How Requirements are Satisfied:**

The firmware components of this module are executed in the Microchip/Atmel ATSAMA5D31 (Microprocessor Unit) and Microchip/Atmel ATSAMA5D36 (Microprocessor Unit), ARM Cortex A5-based (ARMv7) operational environments.

## 6.2 Configuration Settings and Restrictions

The module shall be installed as stated in Section 11.1.

There are no security rules, settings or restrictions to the configuration of the operational environment.

# 7 Physical Security

## 7.1 Mechanisms and Actions Required

N/A for this module.

The module is a firmware-hardware hybrid module. The module contains standard integrated circuits with a uniform exterior material and standard connectors. The module is enclosed within a production-grade enclosure with components that include standard passivation techniques (e.g., a conformal coating applied over the module's circuitry to protect against environmental or other physical damage) conformant to the Level 1 requirements for physical security.

The physical security requirements do not apply to the firmware components of the module.

# 8 Non-Invasive Security

## 8.1 Mitigation Techniques

This module does not implement any non-invasive security mechanism and therefore this section is not applicable.

# 9 Sensitive Security Parameters Management

## 9.1 Storage Areas

| Storage Area Name | Description | Persistence Type |
|---|---|---|
| RAM | Temporary storage for SSPs used by the module as part of service execution. The module does not perform persistent storage of SSPs. | Dynamic |

Table 16: Storage Areas

The module does not perform persistent storage of SSPs. The SSPs are temporarily stored in the RAM in plaintext form. SSPs are provided to the module by the calling process and are destroyed when released by the appropriate zeroization function calls.

## 9.2 SSP Input-Output Methods

| Name | From | To | Format Type | Distribution Type | Entry Type | SFI or Algorithm |
|---|---|---|---|---|---|---|
| API input parameters | Operating calling application (TOEPP) | Cryptographic module | Plaintext | Manual | Electronic | |
| Kernel AF_ALG_type sockets (input) | Operating calling application (TOEPP) | Cryptographic module | Plaintext | Manual | Electronic | |
| API output parameters | Cryptographic module | Operator calling application (TOEPP) | Plaintext | Manual | Electronic | |
| Kernel AF_ALG type sockets (output) | Cryptographic module | Operator calling application (TOEPP) | Plaintext | Manual | Electronic | |

Table 17: SSP Input-Output Methods

## 9.3 SSP Zeroization Methods

| Zeroization Method | Description | Rationale | Operator Initiation |
|---|---|---|---|
| Kernel free cipher handle | Zeroizes the SSPs contained | Memory occupied by SSPs is overwritten with zeroes, which renders the | By calling the appropriate zeroization functions:- AES key: crypto_free_skcipher and crypto_free_aead; - HMAC key: |

| Zeroization Method | Description | Rationale | Operator Initiation |
|---|---|---|---|
| | within the cipher handle | SSP values irretrievable. The completion of the zeroization routine(s) indicate that the zeroization procedure succeeded | crypto_free_shash and crypto_free_ahash; - DRBG Internal state: crypto_free_rng; - EC public & private key: crypto_free_kpp and crypto_free_akcipher |
| FIPS provider calling the zeroization API | Zeroizes the SSPs | Memory occupied by SSPs is overwritten with zeroes, which renders the SSP values irretrievable. All data output is inhibited during zeroization. The completion of the zeroization routine(s) indicate that the zeroization procedure succeeded | By calling the appropriate zeroization functions: - EVP_CIPHER_CTX_free(): clears and frees symmetric cipher context; - EVP_MAC_CTX_free(): clears and frees MAC context; -EVP_KDF_CTX_free(): clears and frees KDF context; - EVP_RAND_CTX_free(): clears and frees DRBG context; - EVP_PKEY_free(): clears and frees asymmetric key pair structures |
| FIPS provider Automatic | Zeroizes the SSPs | Memory occupied by SSPs is overwritten with zeroes, which renders the SSP values irretrievable. All data output is inhibited during zeroization. | Intermediate key generation value: zeroized automatically by the module (after the requested service completed) |
| Remove power from the module | De-allocates the volatile memory used to store SSPs | Volatile memory used by the module is overwritten within nanoseconds when power is removed | By removing power |

Table 18: SSP Zeroization Methods

All data output is inhibited during zeroization.

## 9.4 SSPs

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| Kernel AES key | AES key used for encryption, decryption, and computing MAC tags | 128, 192, 256 bits - 128, 192, 256 bits | Symmetric Key - CSP | | | Kernel AES-CCM (KTS-Wrap) |

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| | | | | | | Kernel AES-GCM (KTS-Wrap)<br>Kernel AES CBC with HMAC<br>Kernel AES CTR with HMAC<br>Kernel AES-ECB<br>Kernel AES-CTR<br>Kernel AES-CBC<br>Kernel AES-CBC-CS3<br>Kernel AES-CFB8<br>Kernel AES-CFB128<br>Kernel AES-XTS<br>Kernel AES-CCM (BC-Auth)<br>Kernel AES-GCM (BC-Auth)<br>Kernel AES-OFB<br>Kernel AES- |

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| | | | | | | CMAC Kernel AES-GMAC |
| Kernel HMAC key | HMAC key | 112-256 bits - 112-256 bits | Authentication key - CSP | | | Kernel AES CBC with HMAC Kernel AES CTR with HMAC Kernel HMAC |
| Kernel Intermediate Key Generation Value | Intermediate key generation value | P-256, P-384 - 128, 192 bits | Intermediate value - CSP | Kernel ECDSA Key Generation | | Kernel ECDSA Key Generation |
| Kernel shared secret | Shared secret generated by ECDH | P-256, P-384 - 128 and 192 bits | Shared secret - CSP | | Kernel KAS-ECC-SSC | Kernel KAS-ECC-SSC |
| DRBG seed | DRBG seed derived from entropy input | CTR_DRBG:256,320,384 bits; HMAC or HASH DRBG: 440,888 bits - CTR_DRBG: 128,192,256 bits; HMAC or HASH DRBG: 128,256 bits | Seed - CSP | Kernel Counter DRBG FIPS provider Counter DRBG FIPS provider Hash DRBG FIPS provider HMAC DRBG | | Kernel Counter DRBG FIPS provider Counter DRBG FIPS provider Hash DRBG FIPS provider HMAC DRBG |
| Kernel EC public key | Public key used for ECDH | P-256, P-384 - 128, 192 bits | Public key - PSP | | | Kernel KAS-ECC-SSC |
| Kernel EC private key | Private key used for ECDH | P-256, P-384 - 128, 192 bits | Private key - CSP | | | Kernel KAS-ECC-SSC |

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| Entropy input | Entropy input used to seed the DRBGs | CTR_DRBG:192,288,384 bits; HMAC or HASH DRBG:240,384 bits - CTR_DRBG:192,288,384 bits; HMAC or HASH DRBG:240,384 bits | Entropy input - CSP | | | Kernel Counter DRBG FIPS provider Counter DRBG FIPS provider Hash DRBG FIPS provider HMAC DRBG |
| Internal State (V, Key) | Internal state of Counter DRBG and HMAC DRBG | CTR_DRBG: 256,320,384 bits; HMAC DRBG: 320,512,1024 bits - CTR_DRBG: 128,192,256 bits; HMAC DRBG: 128,256 bits | DRBG Internal state - CSP | Kernel Counter DRBG FIPS provider Counter DRBG | | Kernel Counter DRBG |
| Internal state (V, C) | Internal state of Hash DRBG | HASH DRBG:888,1776 bits - HASH DRBG:128,256 bits | DRBG Internal state - CSP | FIPS provider Hash DRBG | | FIPS provider Hash DRBG |
| FIPS provider AES Key | AES key used for encryption, decryption, and computing MAC tags | 128, 192, 256 bits - 128, 192, 256 bits | Symmetric Key - CSP | | | FIPS provider AES-CCM (KTS-Wrap) FIPS provider AES-GCM (KTS-Wrap) FIPS provider AES KWP FIPS |

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| | | | | | | provider AES-ECB FIPS provider AES-CTR FIPS provider AES-CBC FIPS provider AES-CBC-CS1 FIPS provider AES-CBC-CS2 FIPS provider AES-CBC-CS3 FIPS provider AES-CFB1 FIPS provider AES-CFB8 FIPS provider AES-XTS FIPS provider AES-CCM (BC-Auth) FIPS provider AES-GCM (BC-Auth) FIPS |

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| | | | | | | provider AES-OFB FIPS provider AES-CMAC FIPS provider AES-GMAC FIPS provider ANS 9.42 Key Derivation (CVL) FIPS provider KBKDF Key Derivation |
| FIPS provider HMAC key | HMAC key | 112-256 bits - 112-256 bits | Authentication key - CSP | | | FIPS provider HMAC |
| FIPS provider shared secret | Shared secret generated by DH/ECDH | 224-8192 bits - 112-256 bits | Shared secret - CSP | | FIPS provider KAS-FFC-SSC FIPS provider KAS-ECC-SSC | FIPS provider ANS 9.42 Key Derivation (CVL) FIPS provider ANS 9.63 Key Derivation (CVL) FIPS provider TLS 1.0 and 1.1 Key Derivatio |

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| | | | | | | n (CVL) FIPS provider TLS 1.2 Key Derivation (CVL) FIPS provider TLS 1.3 Key Derivation (CVL) FIPS provider HKDF Key Derivation FIPS provider OneStep Key Derivation FIPS provider TwoStep Key Derivation FIPS provider SSH Key Derivation |
| FIPS provider TLS pre-master secret | Shared secret used for deriving TLS master secret | 224-8192 bits - 112-256 bits | Shared secret - CSP | | FIPS provider KAS-FFC-SSC FIPS provider KAS-ECC-SSC | FIPS provider TLS 1.0 and 1.1 Key Derivation (CVL) FIPS |

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| | | | | | | provider TLS 1.2 Key Derivation (CVL) FIPS provider TLS 1.3 Key Derivation (CVL) |
| FIPS provider TLS master secret | Shared secret used for the establishment of encrypted session | 256 bits - 112-256 bits based on the TLS pre-master secret used | Shared secret - CSP | | FIPS provider TLS 1.0 and 1.1 Key Derivation (CVL) FIPS provider TLS 1.2 Key Derivation (CVL) FIPS provider TLS 1.3 Key Derivation (CVL) | FIPS provider TLS 1.0 and 1.1 Key Derivation (CVL) FIPS provider TLS 1.2 Key Derivation (CVL) FIPS provider TLS 1.3 Key Derivation (CVL) |
| FIPS provider DH public key | Public key used for DH | 2048, 3072, 4096, 6144, 8192 bits - 112-200 bits | Public key - PSP | | | FIPS provider KAS-FFC-SSC |
| FIPS provider DH private key | Private key used for DH | 2048, 3072, 4096, 6144, 8192 bits - 112-200 bits | Private key - CSP | | | FIPS provider KAS-FFC-SSC |
| FIPS provider EC public key | Public key used for ECDH and ECDSA | P-224, P-256, P-384, P-521; Ed25519, Ed448 - 112, 128, 192, 256 bits | Public key - PSP | | | FIPS provider KAS-ECC-SSC FIPS |

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| | | | | | | provider ECDSA Key Verification<br>FIPS provider ECDSA Signature Verification<br>FIPS provider EDDSA Signature Verification |
| FIPS provider EC private key | Private key used for ECDH and ECDSA | P-224, P-256, P-384, P-521; Ed25519, Ed448 - 112, 128, 192, 256 bits | Private key - CSP | | | FIPS provider KAS-ECC-SSC<br>FIPS provider ECDSA Signature Generation<br>FIPS provider EDDSA Signature Generation |
| FIPS provider module generated DH public key | DH public key generated by the module | 2048, 3072, 4096, 6144, 8192 bits - 112-200 bits | Public key - PSP | FIPS provider Safe Primes Key Generation | | FIPS provider KAS-FFC-SSC |
| FIPS provider module generated | DH private key generated by the module | 2048, 3072, 4096, 6144, 8192 bits - 112-200 bits | Private key - CSP | FIPS provider Safe Primes | | FIPS provider KAS-FFC-SSC |

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| DH private key | | | | Key Generation | | |
| FIPS provider module generated EC public key | EC public key generated by the module | P-224, P-256, P-384, P-521; Ed25519, Ed448 - 128-256 bits | Public key - PSP | FIPS provider ECDSA Key Generation<br>FIPS provider EDDSA Key Generation | | FIPS provider KAS-ECC-SSC<br>FIPS provider ECDSA Signature Verification<br>FIPS provider EDDSA Signature Verification |
| FIPS provider module generated EC private key | EC private key generated by the module | P-224, P-256, P-384, P-521; Ed25519, Ed448 (128, 192 bits) - 128-256 bits | Private key - CSP | FIPS provider ECDSA Key Generation<br>FIPS provider EDDSA Key Generation | | FIPS provider KAS-ECC-SSC<br>FIPS provider ECDSA Signature Generation<br>FIPS provider EDDSA Signature Generation |
| FIPS provider RSA public key | Public key used for RSA signature generation | 2048, 3072, 4096 bits - 112, 128, 150 bits | Public key - PSP | | | FIPS provider RSA Signature Verification |
| FIPS provider RSA | Private key used for RSA signature generation | 2048, 3072, 4096 bits - 112, 128, 150 bits | Private key - CSP | | | FIPS provider RSA |

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| private key | | | | | | Signature Generation |
| FIPS provider module generated RSA public key | RSA public key generated by the module | 2048, 3072, 4096 bits - 112, 128, 150 bits | Public key - PSP | FIPS provider RSA Key Generation | | FIPS provider RSA Key Generation |
| FIPS provider module generated RSA private key | RSA private key generated by the module | 2048, 3072, 4096 bits - 112, 128, 150 bits | Private key - CSP | FIPS provider RSA Key Generation | | FIPS provider RSA Key Generation |
| FIPS provider Intermediate Key Generation Value | Intermediate key generation value | 224-4096 bits - 112-256 bits | Intermediate value - CSP | FIPS provider Safe Primes Key Generation<br>FIPS provider ECDSA Key Generation<br>FIPS provider EDDSA Key Generation<br>FIPS provider RSA Key Generation | | FIPS provider Safe Primes Key Generation<br>FIPS provider ECDSA Key Generation<br>FIPS provider EDDSA Key Generation<br>FIPS provider RSA Key Generation |
| FIPS provider derived key | Symmetric key derived from a key-derivation | 112-4096 bits - 112-256 bits | Symmetric key - CSP | | FIPS provider ANS 9.42 Key | FIPS provider ANS 9.42 Key |

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| | key, shared secret, or password | | | | Derivation (CVL) FIPS provider ANS 9.63 Key Derivation (CVL) FIPS provider TLS 1.0 and 1.1 Key Derivation (CVL) FIPS provider TLS 1.2 Key Derivation (CVL) FIPS provider TLS 1.3 Key Derivation (CVL) FIPS provider HKDF Key Derivation FIPS provider Password-based Key Derivation FIPS provider OneStep Key | Derivation (CVL) FIPS provider ANS 9.63 Key Derivation (CVL) FIPS provider TLS 1.0 and 1.1 Key Derivation (CVL) FIPS provider TLS 1.2 Key Derivation (CVL) FIPS provider TLS 1.3 Key Derivation (CVL) FIPS provider HKDF Key Derivation FIPS provider Password-based Key Derivation FIPS provider OneStep Key |

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| | | | | | Derivation FIPS provider TwoStep Key Derivation FIPS provider KMAC Key Derivation FIPS provider KBKDF Key Derivation FIPS provider SSH Key Derivation | Derivation FIPS provider TwoStep Key Derivation FIPS provider KMAC Key Derivation FIPS provider KBKDF Key Derivation FIPS provider SSH Key Derivation |
| FIPS provider key-derivation key | Symmetric key used to derive symmetric keys | 112-4096 bits - 112-256 bits | Symmetric key - CSP | | | FIPS provider KMAC Key Derivation FIPS provider KBKDF Key Derivation |
| FIPS provider Password | Password used to derive symmetric keys | 8-128 characters - N/A | Password - CSP | | | FIPS provider Password-based Key Derivation |

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| FIPS provider AES Derived Key | AES key used for encryption, decryption, and computing MAC tags | 128, 192, 256 bits - 128, 192, 256 bits | Symmetric Key - CSP | | FIPS provider TLS 1.0 and 1.1 Key Derivation (CVL) FIPS provider TLS 1.2 Key Derivation (CVL) FIPS provider TLS 1.3 Key Derivation (CVL) FIPS provider KBKDF Key Derivation | FIPS provider TLS 1.0 and 1.1 Key Derivation (CVL) FIPS provider TLS 1.2 Key Derivation (CVL) FIPS provider TLS 1.3 Key Derivation (CVL) FIPS provider KBKDF Key Derivation |
| FIPS provider HMAC Derived Key | HMAC key | 112-256 bits - 112-256 bits | Authentication Key - CSP | | FIPS provider TLS 1.0 and 1.1 Key Derivation (CVL) FIPS provider TLS 1.2 Key Derivation (CVL) FIPS provider TLS 1.3 Key Derivation (CVL) | FIPS provider TLS 1.0 and 1.1 Key Derivation (CVL) FIPS provider TLS 1.2 Key Derivation (CVL) FIPS provider TLS 1.3 Key Derivation (CVL) |

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| | | | | | FIPS provider KBKDF Key Derivation | FIPS provider KBKDF Key Derivation |
| FIPS provider 802.11 Pre-shared key (PSK) | Used for pre-shared key authentication and session key establishment, as well as for 802.11 KDF | Up to 256 bits of length - Up to 256 bits | Pre-shared key - CSP | | | FIPS provider KBKDF Key Derivation |
| FIPS provider 802.11 Pairwise Master Key (PMK) | Used for pre-shared key authentication and session key establishment, as well as for 802.11 KDF | 256 or 384 bits - 256 bits | Pairwise Master Key - CSP | | | FIPS provider KBKDF Key Derivation |
| FIPS provider 802.11 KDF Internal State | Used for SP800-108 KDF to calculate the WPA2 session keys | N/A - N/A | Internal state - CSP | FIPS provider KBKDF Key Derivation | | FIPS provider KBKDF Key Derivation |
| FIPS provider 802.11 Temporal Keys | AES-CCM or AES-GCM keys used for session encryption/decryption | 128 or 256 bits - 128 or 256 bits | Temporal Keys - CSP | FIPS provider KBKDF Key Derivation | | Kernel AES-CCM (BC-Auth) Kernel AES-GCM (BC-Auth) |
| FIPS provider 802.11 MIC keys (KCK) | Key confirmation keys (KCK) used for message authentication during session establishment | 128 or 192 bits - 128 or 192 bits | MIC keys - CSP | | | FIPS provider KBKDF Key Derivation |

| Name | Description | Size - Strength | Type - Category | Generated By | Established By | Used By |
|---|---|---|---|---|---|---|
| FIPS provider 802.11 Key Encryption Key (KEK) | Used for AES Key Wrapping of the 802.11 Group Temporal Key (GTK) | 128 or 256 bits - 128 or 256 bits | Key Encryption Key - CSP | | Kernel AES-CBC Kernel AES-CCM (BC-Auth) Kernel AES-GCM (BC-Auth) | Kernel AES-CBC Kernel AES-CCM (BC-Auth) Kernel AES-GCM (BC-Auth) |
| FIPS provider 802.11 Group Temporal Key (GTK) | 802.11 session key for broadcast communications | 128 to 256 bits - 128 to 256 bits | Group Temporal Key - CSP | | Kernel AES-CBC Kernel AES-CCM (BC-Auth) Kernel AES-GCM (BC-Auth) | Kernel AES-CBC Kernel AES-CCM (BC-Auth) Kernel AES-GCM (BC-Auth) |

Table 19: SSP Table 1

| Name | Input - Output | Storage | Storage Duration | Zeroization | Related SSPs |
|---|---|---|---|---|---|
| Kernel AES key | API input parameters Kernel AF_ALG_type sockets (input) | RAM:Plaintext | For the duration of the service | Kernel free cipher handle Remove power from the module | |
| Kernel HMAC key | API input parameters Kernel AF_ALG_type sockets (input) | RAM:Plaintext | For the duration of the service | Kernel free cipher handle Remove power from the module | |

| Name | Input - Output | Storage | Storage Duration | Zeroization | Related SSPs |
|---|---|---|---|---|---|
| Kernel Intermediate Key Generation Value | | RAM:Plaintext | For the duration of the service | Kernel free cipher handle Remove power from the module | Kernel EC public key:Generates Kernel EC private key:Generates |
| Kernel shared secret | API output parameters Kernel AF_ALG type sockets (output) | RAM:Plaintext | For the duration of the service | Kernel free cipher handle Remove power from the module | Kernel EC public key:Used With Kernel EC private key:Used With |
| DRBG seed | | RAM:Plaintext | While the DRBG is being instantiated | Kernel free cipher handle FIPS provider calling the zeroization API Remove power from the module | Entropy input:Derived From Internal State (V, Key):Generates Internal state (V, C):Generates |
| Kernel EC public key | API input parameters Kernel AF_ALG_type sockets (input) API output parameters Kernel AF_ALG type sockets (output) | RAM:Plaintext | For the duration of the service | Kernel free cipher handle Remove power from the module | Kernel EC private key:Paired With Kernel Intermediate Key Generation Value:Generated from |
| Kernel EC private key | API input parameters Kernel AF_ALG_type sockets (input) API output parameters Kernel AF_ALG type sockets (output) | RAM:Plaintext | For the duration of the service | Kernel free cipher handle Remove power from the module | Kernel EC public key:Paired With Kernel Intermediate Key Generation Value:Generated from |

| Name | Input - Output | Storage | Storage Duration | Zeroization | Related SSPs |
|---|---|---|---|---|---|
| Entropy input | | RAM:Plaintext | From generation until DRBG seed is created | Kernel free cipher handle FIPS provider calling the zeroization API Remove power from the module | DRBG seed:Derives |
| Internal State (V, Key) | | RAM:Plaintext | From DRBG instantiation until DRBG termination | Kernel free cipher handle FIPS provider calling the zeroization API Remove power from the module | DRBG seed:Generated from |
| Internal state (V, C) | | RAM:Plaintext | From DRBG instantiation until DRBG termination | FIPS provider calling the zeroization API Remove power from the module | DRBG seed:Generated from |
| FIPS provider AES Key | API input parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API Remove power from the module | |
| FIPS provider HMAC key | API input parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API Remove | |

| Name | Input - Output | Storage | Storage Duration | Zeroization | Related SSPs |
|---|---|---|---|---|---|
| | | | | power from the module | |
| FIPS provider shared secret | API output parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API Remove power from the module | FIPS provider DH public key:Established by FIPS provider DH private key:Established by FIPS provider EC public key:Established by FIPS provider EC private key:Established by FIPS provider derived key:Derives |
| FIPS provider TLS pre-master secret | API input parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API Remove power from the module | FIPS provider DH public key:Established by FIPS provider DH private key:Established by FIPS provider EC public key:Established by FIPS provider EC private key:Established by FIPS provider TLS master secret:Derives |
| FIPS provider TLS master secret | API input parameters API output parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API Remove power from the module | FIPS provider TLS pre-master secret:Derived From FIPS provider derived key:Derives |
| FIPS provider DH public key | API input parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API | FIPS provider DH private key:Paired With FIPS provider Intermediate Key |

| Name | Input - Output | Storage | Storage Duration | Zeroization | Related SSPs |
|---|---|---|---|---|---|
| | | | | | Generation Value:Generated from |
| FIPS provider DH private key | API input parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API | FIPS provider DH public key:Paired With<br>FIPS provider Intermediate Key Generation Value:Generated from |
| FIPS provider EC public key | API input parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API<br>Remove power from the module | FIPS provider EC private key:Paired With<br>FIPS provider Intermediate Key Generation Value:Generated from |
| FIPS provider EC private key | API input parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API<br>Remove power from the module | FIPS provider EC public key:Paired With<br>FIPS provider Intermediate Key Generation Value:Generated from |
| FIPS provider module generated DH public key | API output parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API<br>Remove power from the module | FIPS provider module generated DH private key:Paired With<br>FIPS provider Intermediate Key Generation Value:Generated from |
| FIPS provider module generated DH private key | API output parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API<br>Remove power from the module | FIPS provider module generated DH public key:Paired With<br>FIPS provider Intermediate Key Generation |

| Name | Input - Output | Storage | Storage Duration | Zeroization | Related SSPs |
|---|---|---|---|---|---|
| | | | | | Value:Generated from |
| FIPS provider module generated EC public key | API output parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API Remove power from the module | FIPS provider module generated EC private key:Paired With FIPS provider Intermediate Key Generation Value:Generated from |
| FIPS provider module generated EC private key | API output parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API Remove power from the module | FIPS provider module generated EC public key:Paired With FIPS provider Intermediate Key Generation Value:Generated from |
| FIPS provider RSA public key | API input parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API Remove power from the module | FIPS provider RSA private key:Paired With FIPS provider Intermediate Key Generation Value:Generated from |
| FIPS provider RSA private key | API input parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API Remove power from the module | FIPS provider RSA public key:Paired With FIPS provider Intermediate Key Generation Value:Generated from |
| FIPS provider module generated RSA public key | API output parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API Remove power from the module | FIPS provider module generated RSA private key:Paired With FIPS provider Intermediate Key Generation |

| Name | Input - Output | Storage | Storage Duration | Zeroization | Related SSPs |
|---|---|---|---|---|---|
| | | | | | Value:Generated from |
| FIPS provider module generated RSA private key | API output parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API Remove power from the module | FIPS provider module generated RSA public key:Paired With FIPS provider Intermediate Key Generation Value:Generated from |
| FIPS provider Intermediate Key Generation Value | | RAM:Plaintext | For the duration of the service | FIPS provider Automatic | FIPS provider DH public key:Generates FIPS provider DH private key:Generates FIPS provider module generated DH public key:Generates FIPS provider module generated DH private key:Generates FIPS provider EC public key:Generates FIPS provider EC private key:Generates FIPS provider module generated EC public key:Generates FIPS provider module generated EC private key:Generates FIPS provider RSA public key:Generates FIPS provider RSA private key:Generates FIPS provider module generated |

| Name | Input - Output | Storage | Storage Duration | Zeroization | Related SSPs |
|---|---|---|---|---|---|
| | | | | | RSA public key:Generates<br>FIPS provider module generated RSA private key:Generates |
| FIPS provider derived key | API output parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API<br>Remove power from the module | FIPS provider key-derivation key:Derived From<br>FIPS provider shared secret:Derived From<br>FIPS provider password:Derived From |
| FIPS provider key-derivation key | API input parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API<br>Remove power from the module | FIPS provider derived key:Derives |
| FIPS provider Password | API input parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API<br>Remove power from the module | FIPS provider derived key:Derives |
| FIPS provider AES Derived Key | API output parameters | RAM:Plaintext | For the duration of the service | Kernel free cipher handle<br>FIPS provider calling the zeroization API<br>Remove power from the module | FIPS provider derived key:Derives |
| FIPS provider HMAC Derived Key | API output parameters | RAM:Plaintext | For the duration of the service | Kernel free cipher handle | FIPS provider derived key:Derives |

| Name | Input - Output | Storage | Storage Duration | Zeroization | Related SSPs |
|---|---|---|---|---|---|
| | | | | FIPS provider calling the zeroization API Remove power from the module | |
| FIPS provider 802.11 Pre-shared key (PSK) | API input parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API Remove power from the module | FIPS provider derived key:Used With |
| FIPS provider 802.11 Pairwise Master Key (PMK) | API input parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API Remove power from the module | FIPS provider derived key:Used With |
| FIPS provider 802.11 KDF Internal State | | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API Remove power from the module | |
| FIPS provider 802.11 Temporal Keys | API input parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization API Remove power from the module | Kernel AES key:Encrypts Kernel AES key:Decrypts |
| FIPS provider 802.11 MIC keys (KCK) | API input parameters | RAM:Plaintext | For the duration of the service | FIPS provider calling the zeroization | |

| Name | Input - Output | Storage | Storage Duration | Zeroization | Related SSPs |
|---|---|---|---|---|---|
| | | | | API Remove power from the module | |
| FIPS provider 802.11 Key Encryption Key (KEK) | API input parameters | RAM:Plaintext | For the duration of the service | Kernel free cipher handle Remove power from the module | Kernel AES key:Encrypts |
| FIPS provider 802.11 Group Temporal Key (GTK) | API output parameters | RAM:Plaintext | For the duration of the service | Kernel free cipher handle Remove power from the module | Kernel AES key:Encrypts Kernel AES key:Decrypts |

Table 20: SSP Table 2

## 9.5 Transitions

The SHA-1 algorithm as implemented by the module will be non-approved for all purposes, starting January 1, 2031.

# 10 Self-Tests

## 10.1 Pre-Operational Self-Tests

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details |
|---|---|---|---|---|---|
| HMAC-SHA2-256 (A5018) | 256-bit key | Message Authentication | SW/FW Integrity | Module becomes operational and services are available for use | Integrity test for fips.so; Integrity test for kernel binary; Integrity test for fipscheck binary; Integrity test for fipscheck library |

Table 21: Pre-Operational Self-Tests

The pre-operational firmware integrity tests are performed automatically when the module is powered on, before the module transitions into the operational state. The algorithm used for the integrity test (i.e., HMAC-SHA2-256) is self-tested before the firmware integrity test is performed. While the module is executing the self-tests, services are not available, and data output (via the data output interface) is inhibited until the tests are successfully completed. The module transitions to the operational state only after the pre-operational self-tests are passed successfully.

## 10.2 Conditional Self-Tests

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| ECDSA KeyGen (FIPS186-5) (A4711) | N/A | PCT | PCT | crypto_kpp_generate_public_key returns 0 | SP 800-56Ar3 Section 5.6.2.1.4 | Key pair generation |
| HMAC-SHA2-256 (A4711) | 0-8184 bit messages | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA2-256 (A4712) | 0-8184 bit messages | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA2-256 (A4716) | 0-8184 bit messages | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA2-384 (A4711) | 0-8184 bit messages | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA2-384 (A4712) | 0-8184 bit messages | KAT | CAST | Module is operational | Message Authentication | Module initialization |

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| HMAC-SHA2-384 (A4716) | 0-8184 bit messages | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA2-512 (A4711) | 0-8184 bit messages | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA2-512 (A4712) | 0-8184 bit messages | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA2-512 (A4716) | 0-8184 bit messages | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA3-224 (A4713) | 0-8184 bit messages | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA3-256 (A4713) | 0-8184 bit messages | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA3-384 (A4713) | 0-8184 bit messages | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA3-512 (A4713) | 0-8184 bit messages | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA-1 (A5018) | 24-bit message | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA2-512 (A5018) | 24-bit message | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| AES-ECB (A4711) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| AES-ECB (A4712) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| AES-ECB (A4715) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| AES-ECB (A4716) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| AES-ECB (A4717) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| AES-ECB (A4719) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| AES-ECB (A4721) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| AES-OFB (A4723) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| AES-CFB128 (A4724) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| AES-CCM (A4719) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| AES-CCM (A4712) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| AES-CCM (A4716) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| AES-CCM (A4721) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| AES-GCM (A4712) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Message authentication | Module initialization |
| AES-GCM (A4715) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Message authentication | Module initialization |
| AES-GCM (A4717) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Message authentication | Module initialization |
| AES-GCM (A4719) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Message authentication | Module initialization |
| AES-GCM (A4721) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Message authentication | Module initialization |
| AES-CMAC (A4712) | 128 and 256 bit keys | KAT | CAST | Module is operational | Message authentication | Module initialization |
| AES-CMAC (A4716) | 128 and 256 bit keys | KAT | CAST | Module is operational | Message authentication | Module initialization |
| AES-CMAC (A4719) | 128 and 256 bit keys | KAT | CAST | Module is operational | Message authentication | Module initialization |

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| AES-CMAC (A4721) | 128 and 256 bit keys | KAT | CAST | Module is operational | Message authentication | Module initialization |
| KAS-ECC-SSC Sp800-56Ar3 (A4711) | P-256, P-384 | KAT | CAST | Module is operational | Shared secret computation | Module initialization |
| Counter DRBG (A4711) | 128, 192, 256 bit keys with/without PR; Health test per section 11.3 of SP 800-90A | KAT | CAST | Module is operational | Seed Generate | Module initialization |
| Counter DRBG (A4712) | 128, 192, 256 bit keys with/without PR; Health test per section 11.3 of SP 800-90A | KAT | CAST | Module is operational | Seed Generate | Module initialization |
| Counter DRBG (A4715) | 128, 192, 256 bit keys with/without PR; Health test per section 11.3 of SP 800-90A | KAT | CAST | Module is operational | Seed Generate | Module initialization |
| Counter DRBG (A4717) | 128, 192, 256 bit keys with/without PR; Health test per section 11.3 of SP 800-90A | KAT | CAST | Module is operational | Seed Generate | Module initialization |
| Counter DRBG (A4719) | 128, 192, 256 bit keys with/without PR; Health test per section 11.3 of SP 800-90A | KAT | CAST | Module is operational | Seed Generate | Module initialization |
| Counter DRBG (A4721) | 128, 192, 256 bit keys with/without | KAT | CAST | Module is operational | Seed Generate | Module initialization |

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| | PR; Health test per section 11.3 of SP 800-90A | | | | | |
| ECDSA KeyGen (FIPS186-5) (A5018) | SHA2-256 | PCT | PCT | Successful key generation | Signature generation and verification | EC key pair generation |
| RSA KeyGen (FIPS186-5) (A5018) | PKCS#1 v1.5 with SHA2-256 | PCT | PCT | Successful key generation | Signature generation and verification | RSA key pair generation |
| Safe Primes Key Generation (A5014) | N/A | PCT | PCT | Successful key generation | Public key re-computation and comparison with the existing public key (per SP 800-56Ar3 Section 5.6.2.1.4) | Safe Primes key pair generation |
| EDDSA KeyGen (A5016) | ED25519 and ED448 | PCT | PCT | Successful key generation | Signature generation and verification | EDDSA key pair generation |
| AES-ECB (A5019) | 128-bit keys, 128-bit ciphertext | KAT | CAST | Module is operational | Decryption | Module initialization |
| AES-GCM (A5008) | 256-bit keys, 96-bit IVs, 128-bit plaintext, 128-bit additional data | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| KDF SP800-108 (A5017) | Counter mode, HMAC-SHA2-256, 128-bit input key | KAT | CAST | Module is operational | Key Derivation | Module initialization |
| KDA OneStep SP800-56Cr2 (A5012) | SHA-224, 392-bit input secret | KAT | CAST | Module is operational | Key Derivation | Module initialization |
| KDA HKDF Sp800-56Cr1 (A5013) | SHA-256, 48-bit input secret | KAT | CAST | Module is operational | Key Derivation | Module initialization |

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| KDF ANS 9.42 (A5018) | SHA-1 with AES-128, KW, 160-bit input secret | KAT | CAST | Module is operational | Key Derivation | Module initialization |
| KDF ANS 9.42 (A5020) | SHA-1 with AES-128, KW, 160-bit input secret | KAT | CAST | Module is operational | Key Derivation | Module initialization |
| KDF ANS 9.63 (A5018) | SHA-256, 192-bit input secret | KAT | CAST | Module is operational | Key Derivation | Module initialization |
| KDF SSH (A5019) | SHA-1, 1056-bit input secret | KAT | CAST | Module is operational | Key Derivation | Module initialization |
| TLS v1.2 KDF RFC7627 (A5018) | SHA-256, 84-bit input secret | KAT | CAST | Module is operational | Key Derivation | Module initialization |
| TLS v1.3 KDF (A5013) | Extract and expand modes, SHA-256 | KAT | CAST | Module is operational | Key Derivation | Module initialization |
| PBKDF (A5018) | SHA-256, 24-character password, 288-bit salt, Iteration count: 4096 | KAT | CAST | Module is operational | Key Derivation | Module initialization |
| PBKDF (A5020) | SHA-256, 24-character password, 288-bit salt, Iteration count: 4096 | KAT | CAST | Module is operational | Key Derivation | Module initialization |
| Counter DRBG (A5015) | AES-128 with prediction resistance | KAT | CAST | Module is operational | Instantiate, Generate, Reseed, Generate (compliant with SP 800-90Ar1 Section 11.3) | Module initialization |
| HMAC DRBG (A5015) | SHA-1 with prediction resistance | KAT | CAST | Module is operational | Instantiate, Generate, Reseed, Generate (compliant with SP 800-90Ar1 Section 11.3) | Module initialization |

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| Hash DRBG (A5015) | SHA-256 with prediction resistance | KAT | CAST | Module is operational | Instantiate, Generate, Reseed, Generate (compliant with SP 800-90Ar1 Section 11.3) | Module initialization |
| KAS-FFC-SSC Sp800-56Ar3 (A5014) | ffdhe2048 | KAT | CAST | Module is operational | Shared Secret Computation | Module initialization |
| KAS-ECC-SSC Sp800-56Ar3 (A5018) | P-256 | KAT | CAST | Module is operational | Shared Secret Computation | Module initialization |
| RSA SigGen (FIPS186-5) (A5018) | PKCS#1 v1.5 with SHA-256 and 2048-bit key | KAT | CAST | Module is operational | Signature Generation | Module initialization |
| ECDSA SigGen (FIPS186-5) (A5018) | SHA-256 and P-224, P-256, P-384, and P-521 | KAT | CAST | Module is operational | Signature Generation | Module initialization |
| ECDSA SigGen (FIPS186-5) (A5020) | SHA-256 and P-224, P-256, P-384, and P-521 | KAT | CAST | Module is operational | Signature Generation | Module initialization |
| EDDSA SigGen (A5016) | ED25519 and ED448 | KAT | CAST | Module is operational | Signature Generation | Module initialization |
| KTS-IFC (A5018) | SHA-256 with no padding | KAT | CAST | Module is operational | RSA Primitive Computation | Module initialization |
| AES-CMAC (A5004) | 128 and 256 bit keys | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| AES-CBC (A5004) | 128 and 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| AES-CCM (A5004) | 128, 192, 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| HMAC-SHA2-224 (A4711) | 24-bit message | KAT | CAST | Module is operational | Message Authentication | Module initialization |

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| HMAC-SHA2-224 (A4712) | 24-bit message | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA2-224 (A4716) | 24-bit message | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA2-224 (A5018) | 24-bit message | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA2-256 (A5018) | 24-bit message | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA2-384 (A5018) | 24-bit message | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA3-224 (A5020) | 24-bit message | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA3-384 (A5020) | 24-bit message | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA3-512 (A5020) | 24-bit message | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| HMAC-SHA3-256 (A5020) | 24-bit message | KAT | CAST | Module is operational | Message Authentication | Module initialization |
| AES-CBC-CS3 (A4714) | 128 and 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| AES-CBC-CS3 (A4718) | 128 and 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| AES-CBC-CS3 (A4720) | 128 and 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| AES-CBC-CS3 (A4722) | 128 and 256 bit keys | KAT | CAST | Module is operational | Encryption, Decryption (Separately) | Module initialization |
| SHAKE-128 (A5020) | 0-8184 bit messages | KAT | CAST | Module is operational | Message digest | Module initialization |

| Algorithm or Test | Test Properties | Test Method | Test Type | Indicator | Details | Conditions |
|---|---|---|---|---|---|---|
| SHAKE-256 (A5020) | 0-8184 bit messages | KAT | CAST | Module is operational | Message digest | Module initialization |
| KDF KMAC Sp800-108r1 (A5017) | Counter mode, HMAC-SHA2-256, 128-bit input key | KAT | CAST | Module is operational | Key Derivation | Module initialization |
| KDA TwoStep SP800-56Cr2 (A5012) | SHA-224, 392-bit input secret | KAT | CAST | Module is operational | Key Derivation | Module initialization |
| KMAC-128 (A5020) | 0-8184 bit messages | KAT | CAST | Module is operational | Message digest | Module initialization |
| KMAC-256 (A5020) | 0-8184 bit messages | KAT | CAST | Module is operational | Message digest | Module initialization |
| KAS-IFC-SSC (A5018) | SHA-256 with no padding | KAT | CAST | Module is operational | Shared Secret Computation | Module initialization |

Table 22: Conditional Self-Tests

The module performs self-tests on all approved cryptographic algorithms as part of the approved services supported in the approved mode of operation, using the tests shown in the table above. Services are not available, and data output (via the data output interface) is inhibited during the self-tests. If any of these tests fails, the module transitions to the error state.

## 10.3 Periodic Self-Test Information

| Algorithm or Test | Test Method | Test Type | Period | Periodic Method |
|---|---|---|---|---|
| HMAC-SHA2-256 (A5018) | Message Authentication | SW/FW Integrity | On demand | Manually |

Table 23: Pre-Operational Periodic Information

| Algorithm or Test | Test Method | Test Type | Period | Periodic Method |
|---|---|---|---|---|
| ECDSA KeyGen (FIPS186-5) (A4711) | PCT | PCT | On demand | Manually |
| HMAC-SHA2-256 (A4711) | KAT | CAST | On demand | Manually |

| Algorithm or Test | Test Method | Test Type | Period | Periodic Method |
|---|---|---|---|---|
| HMAC-SHA2-256 (A4712) | KAT | CAST | On demand | Manually |
| HMAC-SHA2-256 (A4716) | KAT | CAST | On demand | Manually |
| HMAC-SHA2-384 (A4711) | KAT | CAST | On demand | Manually |
| HMAC-SHA2-384 (A4712) | KAT | CAST | On demand | Manually |
| HMAC-SHA2-384 (A4716) | KAT | CAST | On demand | Manually |
| HMAC-SHA2-512 (A4711) | KAT | CAST | On demand | Manually |
| HMAC-SHA2-512 (A4712) | KAT | CAST | On demand | Manually |
| HMAC-SHA2-512 (A4716) | KAT | CAST | On demand | Manually |
| HMAC-SHA3-224 (A4713) | KAT | CAST | On demand | Manually |
| HMAC-SHA3-256 (A4713) | KAT | CAST | On demand | Manually |
| HMAC-SHA3-384 (A4713) | KAT | CAST | On demand | Manually |
| HMAC-SHA3-512 (A4713) | KAT | CAST | On demand | Manually |
| HMAC-SHA-1 (A5018) | KAT | CAST | On demand | Manually |
| HMAC-SHA2-512 (A5018) | KAT | CAST | On demand | Manually |
| AES-ECB (A4711) | KAT | CAST | On demand | Manually |
| AES-ECB (A4712) | KAT | CAST | On demand | Manually |
| AES-ECB (A4715) | KAT | CAST | On demand | Manually |
| AES-ECB (A4716) | KAT | CAST | On demand | Manually |
| AES-ECB (A4717) | KAT | CAST | On demand | Manually |
| AES-ECB (A4719) | KAT | CAST | On demand | Manually |
| AES-ECB (A4721) | KAT | CAST | On demand | Manually |
| AES-OFB (A4723) | KAT | CAST | On demand | Manually |
| AES-CFB128 (A4724) | KAT | CAST | On demand | Manually |
| AES-CCM (A4719) | KAT | CAST | On demand | Manually |
| AES-CCM (A4712) | KAT | CAST | On demand | Manually |
| AES-CCM (A4716) | KAT | CAST | On demand | Manually |
| AES-CCM (A4721) | KAT | CAST | On demand | Manually |
| AES-GCM (A4712) | KAT | CAST | On demand | Manually |
| AES-GCM (A4715) | KAT | CAST | On demand | Manually |

| Algorithm or Test | Test Method | Test Type | Period | Periodic Method |
|---|---|---|---|---|
| AES-GCM (A4717) | KAT | CAST | On demand | Manually |
| AES-GCM (A4719) | KAT | CAST | On demand | Manually |
| AES-GCM (A4721) | KAT | CAST | On demand | Manually |
| AES-CMAC (A4712) | KAT | CAST | On demand | Manually |
| AES-CMAC (A4716) | KAT | CAST | On demand | Manually |
| AES-CMAC (A4719) | KAT | CAST | On demand | Manually |
| AES-CMAC (A4721) | KAT | CAST | On demand | Manually |
| KAS-ECC-SSC Sp800-56Ar3 (A4711) | KAT | CAST | On demand | Manually |
| Counter DRBG (A4711) | KAT | CAST | On demand | Manually |
| Counter DRBG (A4712) | KAT | CAST | On demand | Manually |
| Counter DRBG (A4715) | KAT | CAST | On demand | Manually |
| Counter DRBG (A4717) | KAT | CAST | On demand | Manually |
| Counter DRBG (A4719) | KAT | CAST | On demand | Manually |
| Counter DRBG (A4721) | KAT | CAST | On demand | Manually |
| ECDSA KeyGen (FIPS186-5) (A5018) | PCT | PCT | On demand | Manually |
| RSA KeyGen (FIPS186-5) (A5018) | PCT | PCT | On demand | Manually |
| Safe Primes Key Generation (A5014) | PCT | PCT | On demand | Manually |
| EDDSA KeyGen (A5016) | PCT | PCT | On demand | Manually |
| AES-ECB (A5019) | KAT | CAST | On demand | Manually |
| AES-GCM (A5008) | KAT | CAST | On demand | Manually |
| KDF SP800-108 (A5017) | KAT | CAST | On demand | Manually |
| KDA OneStep SP800-56Cr2 (A5012) | KAT | CAST | On demand | Manually |

| Algorithm or Test | Test Method | Test Type | Period | Periodic Method |
|---|---|---|---|---|
| KDA HKDF Sp800-56Cr1 (A5013) | KAT | CAST | On demand | Manually |
| KDF ANS 9.42 (A5018) | KAT | CAST | On demand | Manually |
| KDF ANS 9.42 (A5020) | KAT | CAST | On demand | Manually |
| KDF ANS 9.63 (A5018) | KAT | CAST | On demand | Manually |
| KDF SSH (A5019) | KAT | CAST | On demand | Manually |
| TLS v1.2 KDF RFC7627 (A5018) | KAT | CAST | On demand | Manually |
| TLS v1.3 KDF (A5013) | KAT | CAST | On demand | Manually |
| PBKDF (A5018) | KAT | CAST | On demand | Manually |
| PBKDF (A5020) | KAT | CAST | On demand | Manually |
| Counter DRBG (A5015) | KAT | CAST | On demand | Manually |
| HMAC DRBG (A5015) | KAT | CAST | On demand | Manually |
| Hash DRBG (A5015) | KAT | CAST | On demand | Manually |
| KAS-FFC-SSC Sp800-56Ar3 (A5014) | KAT | CAST | On demand | Manually |
| KAS-ECC-SSC Sp800-56Ar3 (A5018) | KAT | CAST | On demand | Manually |
| RSA SigGen (FIPS186-5) (A5018) | KAT | CAST | On demand | Manually |
| ECDSA SigGen (FIPS186-5) (A5018) | KAT | CAST | On demand | Manually |
| ECDSA SigGen (FIPS186-5) (A5020) | KAT | CAST | On demand | Manually |
| EDDSA SigGen (A5016) | KAT | CAST | On demand | Manually |
| KTS-IFC (A5018) | KAT | CAST | On demand | Manually |
| AES-CMAC (A5004) | KAT | CAST | On demand | Manually |
| AES-CBC (A5004) | KAT | CAST | On demand | Manually |
| AES-CCM (A5004) | KAT | CAST | On demand | Manually |

| Algorithm or Test | Test Method | Test Type | Period | Periodic Method |
|---|---|---|---|---|
| HMAC-SHA2-224 (A4711) | KAT | CAST | On demand | Manually |
| HMAC-SHA2-224 (A4712) | KAT | CAST | On demand | Manually |
| HMAC-SHA2-224 (A4716) | KAT | CAST | On demand | Manually |
| HMAC-SHA2-224 (A5018) | KAT | CAST | On demand | Manually |
| HMAC-SHA2-256 (A5018) | KAT | CAST | On demand | Manually |
| HMAC-SHA2-384 (A5018) | KAT | CAST | On demand | Manually |
| HMAC-SHA3-224 (A5020) | KAT | CAST | On demand | Manually |
| HMAC-SHA3-384 (A5020) | KAT | CAST | On demand | Manually |
| HMAC-SHA3-512 (A5020) | KAT | CAST | On demand | Manually |
| HMAC-SHA3-256 (A5020) | KAT | CAST | On demand | Manually |
| AES-CBC-CS3 (A4714) | KAT | CAST | On demand | Manually |
| AES-CBC-CS3 (A4718) | KAT | CAST | On demand | Manually |
| AES-CBC-CS3 (A4720) | KAT | CAST | On demand | Manually |
| AES-CBC-CS3 (A4722) | KAT | CAST | On demand | Manually |
| SHAKE-128 (A5020) | KAT | CAST | On demand | Manually |
| SHAKE-256 (A5020) | KAT | CAST | On demand | Manually |
| KDF KMAC Sp800-108r1 (A5017) | KAT | CAST | On demand | Manually |
| KDA TwoStep SP800-56Cr2 (A5012) | KAT | CAST | On demand | Manually |
| KMAC-128 (A5020) | KAT | CAST | On demand | Manually |
| KMAC-256 (A5020) | KAT | CAST | On demand | Manually |
| KAS-IFC-SSC (A5018) | KAT | CAST | On demand | Manually |

Table 24: Conditional Periodic Information

## 10.4 Error States

| Name | Description | Conditions | Recovery Method | Indicator |
|---|---|---|---|---|
| Error State | The module immediately stops functioning due to a self-test failure | Firmware integrity test failure<br>CAST Failure<br>PCT Failure | Successful completion of self-tests after reboot | Module will reboot. |

Table 25: Error States

In the error state, the output interface is inhibited, and the module accepts no more inputs or requests (as the module is no longer running).

## 10.5 Operator Initiation of Self-Tests

All self-tests, with the exception of the health tests, can be invoked on demand by unloading and subsequently re-initializing the module. The entropy health tests are run during DRBG seeding and reseeding. Similarly, a Pair-wise Consistency Test (PCT) it is run for keygen operations.

# 11 Life-Cycle Assurance

## 11.1 Installation, Initialization, and Startup Procedures

Before deploying the module for usage, the Crypto Officer shall employ the following steps:

1. Verify the HMAC values of each component of the module as listed in section 2.2.
2. Verify that the kernel component command line is configured to run fipsInit.sh before any user mode application or init system.
3. Verify that 'fips=1' parameter is present on the kernel command line for Approved mode operation.

## 11.2 Administrator Guidance

The Crypto Officer must execute the "cat /proc/sys/crypto/fips_name" command. The Crypto Officer must ensure that the proper name is listed in the output as follows:

*Summit Linux*

This output maps to the module name "Summit Linux FIPS Core Crypto Module".

Next the Crypto Officer must execute "cat /proc/sys/crypto/fips_version". This command must output the following:

*11.1*

The hardware component of the module for both OE's can be identified by executing the "cat /proc/cpuinfo" command which outputs:

```
processor   : 0
model name  : ARMv7 Processor rev 1 (v7l)
BogoMIPS    : 33.00
Features    : half thumb fastmult vfp edsp thumbee vfpv3 vfpv3d16 tls vfpv4
CPU implementer   : 0x41
CPU architecture: 7
CPU variant : 0x0
CPU part    : 0xc05
CPU revision      : 1

Hardware    : Atmel SAMA5
Revision    : 0000
Serial            : 0000000000000000
```

The following are the HMAC values for each of the components for each platform:

- WB50NBT MPU /usr/lib/fipscheck/Image.gz.hmac: a1eb02e51646193b56a0391677293df724cf5a75a424812a1872842049b830fa

- WB50NBT MPU /usr/lib/fipscheck/fips.so.hmac: c246bfcce5880bebd46dae090270fff4f06d3ba3482e5b1ac4c7000d73b2372d
- WB50NBT MPU /usr/lib/fipscheck/fipscheck.hmac: 5c986e37c11276eb300df8ee113f28b3129276f90b2b8099aa08c2c74fc89616
- WB50NBT MPU /usr/lib/fipscheck/libfipscheck.so.1.hmac: baae25b3d5faaedf74376e8e4ce729ec7009f1038d4f8851bcf3095491dded00
- SU60-SOMC 60 Series SOM /usr/lib/fipscheck/Image.gz.hmac: 013763a8e1ddd8395cf3e1f8d5d6e171a1486a350c66124fea88aaaf8f105c5d
- SU60-SOMC 60 Series SOM /usr/lib/fipscheck/fips.so.hmac: c246bfcce5880bebd46dae090270fff4f06d3ba3482e5b1ac4c7000d73b2372d
- SU60-SOMC 60 Series SOM /usr/lib/fipscheck/fipscheck.hmac: 5c986e37c11276eb300df8ee113f28b3129276f90b2b8099aa08c2c74fc89616
- SU60-SOMC 60 Series SOM /usr/lib/fipscheck/libfipscheck.so.1.hmac: baae25b3d5faaedf74376e8e4ce729ec7009f1038d4f8851bcf3095491dded00

## 11.3 Non-Administrator Guidance

There is no non-administrator guidance.

## 11.4 End of Life

As the module does not persistently store SSPs, secure sanitization of the module consists of unloading the module. This will zeroize all SSPs in volatile memory.

# 12 Mitigation of Other Attacks

## 12.1 Attack List

For the FIPS Provider component, certain cryptographic subroutines and algorithms are vulnerable to timing analysis. The FIPS Provider component mitigates this vulnerability by using constant-time implementations. This includes, but is not limited to:

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

In addition, RSA, ECDSA, ECDH, and DH employ blinding techniques to further impede timing and power analysis. No configuration is needed to enable the countermeasures.

# Appendix A. Glossary and Abbreviations

| | |
|---|---|
| AES | Advanced Encryption Standard |
| API | Application Programming Interface |
| CAST | Cryptographic Algorithm Self-Test |
| CAVP | Cryptographic Algorithm Validation Program |
| CBC | Cipher Block Chaining |
| CCM | Counter with Cipher Block Chaining-Message Authentication Code |
| CFB | Cipher Feedback |
| CMAC | Cipher-based Message Authentication Code |
| CMVP | Cryptographic Module Validation Program |
| CSP | Critical Security Parameter |
| CTR | Counter |
| CTS | Ciphertext Stealing |
| DH | Diffie-Hellman |
| DRBG | Deterministic Random Bit Generator |
| ECB | Electronic Code Book |
| ECC | Elliptic Curve Cryptography |
| ECDH | Elliptic Curve Diffie-Hellman |
| ECDSA | Elliptic Curve Digital Signature Algorithm |
| EMS | Extended Master Secret |
| ENT (NP) | Non-physical Entropy Source |
| FFC | Finite Field Cryptography |
| FIPS | Federal Information Processing Standards |
| GCM | Galois Counter Mode |
| GMAC | Galois Counter Mode Message Authentication Code |
| HKDF | HMAC-based Key Derivation Function |
| HMAC | Keyed-Hash Message Authentication Code |
| IPsec | Internet Protocol Security |
| KAT | Known Answer Test |
| KBKDF | Key-based Key Derivation Function |
| MAC | Message Authentication Code |
| NIST | National Institute of Science and Technology |
| PAA | Processor Algorithm Acceleration |
| PBKDF2 | Password-based Key Derivation Function v2 |
| PKCS | Public-Key Cryptography Standards |
| RSA | Rivest, Shamir, Addleman |
| SFI | Security Function Implementation |
| SHA | Secure Hash Algorithm |
| SSC | Shared Secret Computation |
| SSP | Sensitive Security Parameter |
| TOEPP | Test Operational Environment's Physical Perimeter |
| XTS | XEX-based Tweaked-codebook mode with cipher text Stealing |

# Appendix B. References


| | |
|---|---|
| ANS X9.42-2001 | **Public Key Cryptography for the Financial Services Industry: Agreement of Symmetric Keys Using Discrete Logarithm Cryptography**<br>2001<br>https://webstore.ansi.org/standards/ascx9/ansix9422001 |
| ANS X9.63-2001 | **Public Key Cryptography for the Financial Services Industry, Key Agreement and Key Transport Using Elliptic Curve Cryptography**<br>2001<br>https://webstore.ansi.org/standards/ascx9/ansix9632001 |
| FIPS 140-3 | FIPS PUB 140-3 - Security Requirements For Cryptographic Modules<br>March 2019<br>https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.140-3.pdf |
| FIPS 140-3 IG | Implementation Guidance for FIPS PUB 140-3 and the Cryptographic Module Validation Program<br>https://csrc.nist.gov/Projects/cryptographic-module-validation-program/fips-140-3-ig-announcements |
| FIPS 180-4 | **Secure Hash Standard (SHS)**<br>March 2012<br>https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf |
| FIPS 186-5 | **Digital Signature Standard (DSS)**<br>February 3, 2023<br>https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-5.pdf |
| FIPS 197 | **Advanced Encryption Standard**<br>November 2001<br>https://csrc.nist.gov/publications/fips/fips197/fips-197.pdf |
| FIPS 198-1 | **The Keyed Hash Message Authentication Code (HMAC)**<br>July 2008<br>https://csrc.nist.gov/publications/fips/fips198-1/FIPS-198-1_final.pdf |
| FIPS 202 | **SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions**<br>August 2015<br>https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf |
| PKCS#1 | **Public Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1**<br>February 2003<br>https://www.ietf.org/rfc/rfc3447.txt |
| RFC 3526 | **More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)**<br>May 2003<br>https://www.ietf.org/rfc/rfc3526.txt |

| | |
|---|---|
| RFC 5288 | **AES Galois Counter Mode (GCM) Cipher Suites for TLS**<br>August 2008<br>https://www.ietf.org/rfc/rfc5288.txt |
| RFC 7919 | **Negotiated Finite Field Diffie-Hellman Ephemeral Parameters for Transport Layer Security (TLS)**<br>August 2016<br>https://www.ietf.org/rfc/rfc7919.txt |
| RFC 8446 | **The Transport Layer Security (TLS) Protocol Version 1.3**<br>August 2018<br>https://www.ietf.org/rfc/rfc8446.txt |
| SP 800-38A | **Recommendation for Block Cipher Modes of Operation Methods and Techniques**<br>December 2001<br>https://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf |
| SP 800-38A Addendum | **Recommendation for Block Cipher Modes of Operation: Three Variants of Ciphertext Stealing for CBC Mode**<br>October 2010<br>https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38a-add.pdf |
| SP 800-38B | **Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication**<br>May 2005<br>https://csrc.nist.gov/publications/nistpubs/800-38B/SP_800-38B.pdf |
| SP 800-38C | **Recommendation for Block Cipher Modes of Operation: the CCM Mode for Authentication and Confidentiality**<br>May 2004<br>https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38c.pdf |
| SP 800-38D | Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC<br>November 2007<br>https://csrc.nist.gov/publications/nistpubs/800-38D/SP-800-38D.pdf |
| SP 800-38E | **Recommendation for Block Cipher Modes of Operation: The XTS AES Mode for Confidentiality on Storage Devices**<br>January 2010<br>https://csrc.nist.gov/publications/nistpubs/800-38E/nist-sp-800-38E.pdf |
| SP 800-38F | **Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping**<br>December 2012<br>https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38F.pdf |
| SP 800-52r2 | **Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations**<br>August 2019<br>https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-52r2.pdf |

| | |
|---|---|
| SP 800-56Ar3 | **Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography**<br>April 2018<br>https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Ar3.pdf |
| SP 800-56Cr2 | **Recommendation for Key-Derivation Methods in Key-Establishment Schemes**<br>August 2020<br>https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Cr2.pdf |
| SP 800-90Ar1 | **Recommendation for Random Number Generation Using Deterministic Random Bit Generators**<br>June 2015<br>https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf |
| SP 800-90B | **Recommendation for the Entropy Sources Used for Random Bit Generation**<br>January 2018<br>https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90B.pdf |
| SP 800-108r1 | **NIST Special Publication 800-108 - Recommendation for Key Derivation Using Pseudorandom Functions**<br>August 2022<br>https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-108r1.pdf |
| SP 800-131Ar2 | **Transitioning the Use of Cryptographic Algorithms and Key Lengths**<br>March 2019<br>https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-131Ar2.pdf |
| SP 800-132 | **Recommendation for Password-Based Key Derivation - Part 1: Storage Applications**<br>December 2010<br>https://csrc.nist.gov/publications/nistpubs/800-132/nist-sp800-132.pdf |
| SP 800-133r2 | **Recommendation for Cryptographic Key Generation**<br>June 2020<br>https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-133r2.pdf |
| SP 800-135r1 | **Recommendation for Existing Application-Specific Key Derivation Functions**<br>December 2011<br>https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-135r1.pdf |
| SP 800-140B | CMVP Security Policy Requirements<br>March 2020<br>https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-140B.pdf |