Non-Proprietary Security Policy

Marvell LS2 HSM Family

October 29, 2025

Revision History

Revision	Date	Author	Change Description
1.0	October 18, 2021	Phanikumar Kancharla	FW v1.0 build 01 CMVP initial submission.
1.0.1	April 6, 2022	Phanikumar Kancharla and Girish Kumar Yerra	FW 10.0 build 03 update.
1.0.1a	January 12, 2023	Girish Kumar Yerra	Addressing CMVP comments and adding bug fixes.
1.0.1b	August 16, 2023	Rajendar Kalwa	Addressed comments from NIST. Updated FW build with changes to address CMVP comments and bug fixes to 10.01-07
1.0.1c	December 27, 2023	Rajendar Kalwa	Addressed comments from NIST. Updated FW build with changes to address algorithm transitions and bug fixes to 10.02-1101
1.0.1d	May 13, 2024	Rajendar Kalwa Vikash Kumar	Addressed comments from NIST
1.0.1e	June 4, 2024	Rajendar Kalwa	Addressed final comments from NIST
1.0.2	March 7, 2025	Vikash Kumar Deepanshu Tyagi	NSRL submission changes
1.0.3	October 29, 2025	Vikash Kumar Deepanshu Tyagi	NSRL submission changes

Notice

Table of Contents

List of Tables

List of Figures

1 General

1.1 Module Overview

The LS2 HSM Family module (hereafter referred to as the module or HSM) by Marvell is a high-performance purpose-built security solution for key management and crypto acceleration. The module provides a FIPS 140-3 overall Level 3 security solution. The module is deployed in a PCIe slot to provide crypto and protocol acceleration in a secure manner to the system host. It is typically deployed in a server or an appliance to provide crypto offload for the keys stored on the HSM. The module's functions are accessed over the PCIe interface via opcode defined by the module.

The HSM is a hardware multi-chip embedded cryptographic module with firmware programmed on the HSM. The module provides cryptographic primitives to accelerate approved, allowed and non-approved, allowed (only in non-approved mode of operation) algorithms to support use cases including PKI, code signing, document signing, Root Of Trust, and TLS. The cryptographic functionality includes asymmetric (RSA/EC) and symmetric (AES and Triple-DES) ciphers, signatures, and random number generation, along with protocol-specific complex instructions to support TLS 1.2. The module implements password-based single-factor authentication at FIPS 140-3 Level 3 security and an optional public-key-based authentication. The physical boundary of the module is the outer perimeter of the PCIe card itself as depicted in section 2.5 .

1.2 Security Level

The cryptographic module meets the overall requirements applicable to Level 3 security of FIPS 140-3.

Table 1 Security Levels

ISO/IEC 24759 Section 6 [Number Below]	FIPS 140-3 Section Title	Security Level
1	General	3
2	Cryptographic Module Specification	3
3	Cryptographic Module Interfaces	3
4	Roles, Services, and Authentication	3
5	Software/Firmware Security	3
6	Operational Environment	N/A
7	Physical Security	3
8	Non-Invasive Security	N/A
9	Sensitive Security Parameter Management	3
10	Self-Tests	3
11	Life-Cycle Assurance	3
12	Mitigation of Other Attacks	N/A

2 Cryptographic Module Specification

The configuration of hardware and firmware for this validation is as described in the following table:

Table 2 Cryptographic Module Tested Configuration

Model	Hardware (Part Number and Version)	Firmware Version	Distinguishing Features
LiquidSecurity 2 (LS2)	LS2-G-A050-B0	FW: MARVELL-LS2-FW-10.02-1102 Bootloader: MARVELL-LS2-UBOOT-10.01-10 Bootloader: MARVELL-LS2-UBOOT-10.02-1200 Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R01-SB Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R02-SB FW: MARVELL-LS2-FW-10.23-1107 Bootloader: MARVELL-LS2-UBOOT-10.01-10 Bootloader: MARVELL-LS2-UBOOT-10.02-1200 Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R01-SB Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R02-SB HPS APP: PIN-App:10.23-1107 FW: MARVELL-LS2-FW-10.23-1150 Bootloader: MARVELL-LS2-UBOOT-10.01-10 Bootloader: MARVELL-LS2-UBOOT-10.02-1200-SB Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R01-SB Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R02-SB HPS APP: PIN-App:10.23-1107	HW Version B0
LiquidSecurity 2 (LS2)	LS2-G-A100-B0	FW: MARVELL-LS2-FW-10.02-1102 Bootloader: MARVELL-LS2-UBOOT-10.01-10 Bootloader: MARVELL-LS2-UBOOT-10.02-1200 Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R01-SB Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R02-SB FW: MARVELL-LS2-FW-10.23-1107 Bootloader: MARVELL-LS2-UBOOT-10.01-10 Bootloader: MARVELL-LS2-UBOOT-10.02-1200 Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R01-SB Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R02-SB HPS APP: PIN-App:10.23-1107 FW: MARVELL-LS2-FW-10.23-1150 Bootloader: MARVELL-LS2-UBOOT-10.01-10 Bootloader: MARVELL-LS2-UBOOT-10.02-1200-SB Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R01-SB Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R02-SB HPS APP: PIN-App:10.23-1107	HW Version B0

LiquidSecurity 2 (LS2)	LS2-G-A200-B0	FW: MARVELL-LS2-FW-10.02-1102 Bootloader: MARVELL-LS2-UBOOT-10.01-10 Bootloader: MARVELL-LS2-UBOOT-10.02-1200 Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R01-SB Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R02-SB FW: MARVELL-LS2-FW-10.23-1107 Bootloader: MARVELL-LS2-UBOOT-10.01-10 Bootloader: MARVELL-LS2-UBOOT-10.02-1200 Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R01-SB Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R02-SB HPS APP: PIN-App:10.23-1107	HW Version B0
LiquidSecurity 2 (LS2)	LS2-G-A300-B0	FW: MARVELL-LS2-FW-10.02-1102 Bootloader: MARVELL-LS2-UBOOT-10.01-10 Bootloader: MARVELL-LS2-UBOOT-10.02-1200 Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R01-SB Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R02-SB FW: MARVELL-LS2-FW-10.23-1107 Bootloader: MARVELL-LS2-UBOOT-10.01-10 Bootloader: MARVELL-LS2-UBOOT-10.02-1200 Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R01-SB Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R02-SB HPS APP: PIN-App:10.23-1107 FW: MARVELL-LS2-FW-10.23-1150 Bootloader: MARVELL-LS2-UBOOT-10.01-10 Bootloader: MARVELL-LS2-UBOOT-10.02-1200-SB Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R01-SB Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R02-SB HPS APP: PIN-App:10.23-1107	HW Version B0
LiquidSecurity 2 (LS2)	LS2-G-A400-B0	FW: MARVELL-LS2-FW-10.02-1102 Bootloader: MARVELL-LS2-UBOOT-10.01-10 Bootloader: MARVELL-LS2-UBOOT-10.02-1200 Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R01-SB Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R02-SB FW: MARVELL-LS2-FW-10.23-1107 Bootloader: MARVELL-LS2-UBOOT-10.01-10 Bootloader: MARVELL-LS2-UBOOT-10.02-1200 Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R01-SB Bootloader: MARVELL-LS2-UBOOT-10.23-1107 -R01-SB HPS APP: PIN-App:10.23-1107	HW Version B0

The part number printed on the product label will have additional information to optionally identify a manufacturing process such as key

ceremony, packaging, and shipping per customer requirements.

For example, the part number shown on the product label is in the format “LS2-G-A300-XX-Y-B0”. Here XX represents the key ceremony process and “Y” represents the size of the bracket (short or full) fitted to the module (PCIe adapter).

LS2-HSM-G Firmware

MARVELL-LS2-FW-10.02-1102

MARVELL-LS2-UBOOT-10.01-10

MARVELL-LS2-UBOOT-10.02-1200

MARVELL-LS2-FW-10.23-1107

MARVELL-LS2-UBOOT-10.23-1107 -R01-SB

MARVELL-LS2-UBOOT-10.23-1107 -R02-SB

PIN-App:10.23-1107

MARVELL-LS2-FW-10.23-1150

MARVELL-LS2-UBOOT-10.01-10

MARVELL-LS2-UBOOT-10.02-1200-SB

MARVELL-LS2-UBOOT-10.23-1107 -R01-SB
MARVELL-LS2-UBOOT-10.23-1107 -R02-SB
PIN-App:10.23-1107

Note: These binaries do not have extensions.

The LS2 HSM module is a multi-chip PCIe adapter with firmware. It consists of multiple components, including an operating system, applications exposing services and interfaces related to secure key management, crypto operations, and policy management of the module.

Important hardware components of the module are: general purpose, ARM-based control processor, ARM-based microcontroller, RAM memory, NOR and eMMC flash for persistent storage, USB interfaces, and RJ45 and PCIe gen-4 x8 with SMBus interfaces.

The following components are excluded from the cryptographic boundary as they do not belong to either data/control interface.

- Top side:
    - RJ45 connector (J901): This is blocked in firmware.
- Bottom side:
    - None

2.1 Partitions

The LS2 HSM adapter is an SR-IOV enabled intelligent PCIe adapter with one physical function and 64 virtual functions. The physical device is referred to as Physical Function (PF) while the virtual devices are referred to as Virtual Functions (VF). Allocation of the VF can be dynamically controlled by the PF via registers encapsulated in the capability. Each VF's PCI configuration space can be accessed by its own Bus, Device and Function Number (Routing ID). And each VF also has PCI Memory Space, which is used to map its register set. VF device driver operates on the register set so it can be functional and appear as a real existing PCI device. In addition to crypto offloads, this adapter can provide secure key storage with up to 64 logical partitions, including a master partition. Each partition will have dedicated resources that are logically and securely isolated from other partitions. A partition will have its own specific policies and controls, and its own user accounts to manage what can be done with a partition and by a user of a partition. Consequently, each partition is treated as a virtual HSM and referred as a pHSM (or HSM Partition). An LS2 HSM always has one default partition called the master partition, which exposes the interfaces to create, delete, and update the remaining partitions.

LS2 HSM can either be in host virtualization mode or non-virtualization mode. One or more applications on a host/virtual machine/container can make use of different partitions available on the HSM. Each partition will have dedicated communication channels; these channels are linked to the PCIe devices. In virtualization mode, channels will be linked to PCIe VFs, otherwise linked to PCIe PF. Communication channels have basic checks to identify that an application is communicating with an intended partition on

the HSM.

HSM FW provides the following mechanisms to ensure only an authorized user or application of a specific partition can communicate with it. It is optional to use either mechanism based on the deployment scenarios and risk assessment.

2.1.1 Basic Channel

These channels depend on the Host OS security for device binding and application isolation. In this case, HSM does basic session validation.

- To run any authorized operation on an HSM partition, a user needs to login, get a random session handle, and use the session handle in all future communications to bind the authorization information.
- Security critical operations (user management, key management, backup/restore, etc.) are all protected by encryption using keys that are (optionally derived from) set up during key ceremony.

2.1.2 Encrypted Communication Channels

The end-to-end encryption feature in the module allows an application to initiate a TLS connection with the firmware to ensure the confidentiality of the data communicated to the HSM.

The connection is based on TLS v1.2 with the cipher-suite TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 and TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384(known to OpenSSL as ECDHE_RSA_AES128-GCM-SHA256

and ECDHE_RSA_AES256-GCM-SHA384). The module will act as server, and the host application will act as client. The server private key will be the partition private key (PAK), which is generated for each pHSM when it is created. The server certificate used for the SSL connection is the partition certificate (PAC). The complete chain will be validated by host applications before establishing the TLS connection.

The end-to-end encryption feature is enabled using the initialization configuration parameters. Once this feature is enabled, all commands except initialize and open session are encrypted.

2.1.3 HSM Master Partition

This is the default partition with only one user, called the Master Crypto Officer (MCO). This partition will expose authorized services to manage (create, delete, update, backup/restore) other partitions and set policies or conditions for them to operate. Additionally, it exposes anonymous services like fetching module information, etc. Refer to section 4.3 Roles, Services, and CSP Access for details about the services supported by the master partition.

The master partition must be initialized and the MCO logged in to execute any authorized service.

2.1.4 HSM Partition

Each partition will have a different set of users to manage it and a dedicated key storage and crypto resources associated with it. A partition will have default configuration or policies supplied by the master partition. Some policies can be changed by the partition administrator or PCO. When a partition is created by the MCO, it will be in a zeroized state and must be initialized to do any key management or crypto operations. Partition initialization will create the Partition Crypto Officer (PCO). The PCO can later create up to 1024 users (PCO or PCU) on demand. Each user will have a unique username to identify themselves. The user has to log in to the partition/vHSM to issue any authorized commands. Users are authenticated using passwords submitted during the user creation.

2.2 Modes of Operation

The module supports the following modes of operation:

1. Uninitialized/zeroized mode.
2. Non-approved mode of operation.
3. Approved mode of operation.

The module is initialized into one of the modes specified above during the module initialization period as mentioned in Section 11. The value of the parameter `fipsState` passed into the call specifies the mode. The following values are allowed for the `fipsState` parameter:

0 - Non-Approved mode.

2 - Approved mode with single-factor authentication mechanism.

3 - Approved mode with certificate-based dual-factor authentication mechanism.

The indicator of Approved mode is obtained by using the Get Status service. The `fipsState` field of the Get Status service indicates the mode. CSPs are not shared between the approved and non-approved modes of operation.

2.2.1 Uninitialized/Zeroized Mode of Operation

When received by the end user, the module is not configured or initialized; this state is called uninitialized/zeroized mode. In this mode, the user can query for basic information about the module, such as version details and vendor information, using an unauthenticated role.

2.2.2 Approved Mode of Operation

The module provides an Approved mode of operation, comprising all services described in Table 10. In this mode, the module allows only Approved or allowed algorithms. Request for any non-approved/allowed algorithm is rejected.

2.2.3 Non-Approved Mode of Operation

The module supports a non-Approved mode implementing the non-Approved algorithms listed in Table 6. In this mode, the module also allows Approved or allowed algorithms.

2.3 Supported Cryptographic Algorithms

This section provides the list of supported cryptographic algorithms segregated based on the operating mode.

2.3.1 Approved Algorithms

The cryptographic module supports the following Approved algorithms. There are algorithms, modes, and key/moduli sizes that have been CAVP-tested but are not used by any approved service of the module. Only the algorithms, modes/methods, and key lengths/curves/moduli used by the module in approved mode are listed in the following table.

Note: All symmetric key sizes represent the key strength.

Table 3 Approved Algorithms

CAVP Cert	Algorithm and Standard	Mode/Method	Description/ Key Size(s) / Key Strength(s)	Use / Function
A1946	RSA SigVer (FIPS186-4)	RSA SigVer (FIPS186-4)	RSA 2048 with SHA2-256 Uses SHA2-256 (FIPS 180-4) (#A1946) as the underlying digest algorithm	Firmware integrity verification by U-boot
A1946	SHA2-256 (FIPS 180-4)	SHA2	SHA 256	Firmware integrity verification by U-boot
A1947	AES-CBC (SP 800-38A)	AES	Encrypt/Decrypt: 128, 192, and 256-bit	Data encryption, decryption, key-wrap and key-unwrap
A1947	AES-CCM (SP 800-38C)	AES	Encrypt/Decrypt: 128, 192, and 256-bit Uses AES-ECB (SP 800-38A) (#A1947) as the underlying block cipher	Authenticated Data encryption, decryption
A1947	AES-CMAC (SP 800-38B)	AES	CMAC generate and verify: 128, 192, and 256-bit Uses AES-CBC (SP 800-38A) (#A1947) as the underlying block cipher	Message authentication code generation and verification
A1947	AES-CTR (SP 800-38A)	AES	Encrypt/Decrypt: 128, 192, and 256-bit Uses AES-ECB (SP 800-38A)	Data encryption, decryption

CAVP Cert	Algorithm and Standard	Mode/Method	Description/ Key Size(s) / Key Strength(s)	Use / Function
			(#A1947) as the underlying block cipher	
A1947	AES-ECB (SP 800-38A)	AES	Encrypt/Decrypt: 128, 192, and 256-bit	Data encryption, decryption, key-wrap and key-unwrap
A1947	AES-GCM (SP 800-38D)	AES	Encrypt/Decrypt: 128, 192, and 256-bit Uses AES-ECB (SP 800-38A) (#A1947) as the underlying block cipher	Authenticated Data encryption, decryption, key wrap, and key unwrap
A1947	AES-GMAC (SP 800-38D)	AES	GMAC generation: 128, 192, and 256-bitUses AES-ECB (SP 800-38A) (#A1947) as the underlying block cipher	Message Authentication
A1947	ECDSA SigGen (FIPS186-4) (CVL)	ECDSA SigGen (FIPS186-4)	SigGen Component: P-224, P-256, P-384, P-521 (SHA-256, 384, and 512) Uses Hash DRBG (SP 800-90Ar1) (#A1947) as underlying Random Generator P-224, P-256, P-384, and P-521 curves providing 112, 128, 192, or 256 bits of encryption strength respectively	Signature generation
A1947	Hash DRBG (SP 800-90Ar1)	Hash DRBG	SHA-512 based with security strength of 256-bit. No prediction resistance Uses SHA2-512 (#A1947) as the underlying digest algorithm	Random number generation for user, internal IVs, and salt
A1947	HMAC-SHA2-256 (FIPS 198-1)	HMAC	HMAC-SHA2-256 Uses SHA2-256 (#A1947) as the underlying digest algorithm	MAC generation, verify, KAS and KDF
A1947	HMAC-SHA2-384 (FIPS 198-1)	HMAC	HMAC-SHA2-384 Uses SHA2-384 (#A1947) as the underlying digest algorithm	MAC generation, verify, KAS and KDF
A1947	HMAC-SHA2-512 (FIPS 198-1)	HMAC	HMAC-SHA2-512 Uses SHA2-512 (#A1947) as the underlying digest algorithm	MAC generation, verify, KAS and KDF
A1947	KAS-ECC-SSC SP800-56Ar3	KAS	ECC CDH staticUnified: P-224, P-256, P-384, P-521, P-224, P-256, P384 and P-521 providing 112 bits,128 bits. 192 bits and 256 bits of encryption strength respectively Uses Hash DRBG (SP800-90Ar1) (#A1947) as underlying Random Generator for the ECDSA KeyGen key pair Uses ECDSA KeyVer (FIPS	Shared secret computation

CAVP Cert	Algorithm and Standard	Mode/Method	Description/ Key Size(s) / Key Strength(s)	Use / Function
			186-4) (#A1948) and ECDSA KeyGen (FIPS 186-4) (#A2393) underlying verification for the key pair Uses SHA2-256 (FIPS 180-4) (#A1947), SHA2-384 (FIPS 180-4) (#A1947) and SHA2-512 (FIPS 180-4) (#A1947) as underlying digest algorithm	
A1947	KDF TLS (CVL) (SP 800-135r1)	KDF TLS	TLS-KDF (v1.2) v1.2: SHA2-256, SHA2-384, SHA2-512 Uses HMAC-SHA2-256 (FIPS 180-4) (#A1947), HMAC-SHA2-384 (FIPS 180-4) (#A1947) and HMAC-SHA2-512 (FIPS 180-4) (#A1947) as underlying MAC algorithm	TLS handshake
A1947	RSA Decryption Primitive (SP 800-56Br2) (CVL)	RSA Decryption Primitive	2048 bit, 3072 bit and 4096-bit 2048, 3072, and 4096-bit modulus providing 112, 128, and 150 bits of encryption strength respectively.	Decryption primitive
A1947	RSA Signature Primitive (CVL) (FIPS 186-4)	RSA Signature Primitive	2048 bit, 3072 bit and 4096-bit 2048, 3072, and 4096-bit modulus providing 112, 128, and 150 bits of encryption strength respectively.	Signature primitive
A1947	SHA-1 (FIPS 180-4)	SHA-1	SHA-1	Message digest
A1947	SHA2-256 (FIPS 180-4)	SHA2	SHA2256	Message digest
A1947	SHA2-384 (FIPS 180-4)	SHA2	SHA2-384	Message digest
A1947	SHA2-512 (FIPS 180-4)	SHA2	SHA2-512	Message digest
A1947	SHA3-224 (FIPS 202)	SHA3	SHA3-224	Message digest
A1947	SHA3-256 (FIPS 202)	SHA3	SHA3-256	Message digest
A1947	SHA3-384 (FIPS 202)	SHA3	SHA3-384	Message digest
A1947	SHA3-512 (FIPS 202)	SHA3	SHA3-512	Message digest
A1947	SHAKE-128 (FIPS 202)	SHAKE-128	SHAKE-128	Message digest
A1947	SHAKE-256 (FIPS 202)	SHAKE-256	SHAKE-256	Message digest
A1947	TDES-CBC (SP 800-38A)	TDES	3-key Triple-DES decrypt (supports only 192-bit size)	Data decryption * Legacy use only
A1947	TDES-ECB (SP 800-38A)	TDES	3-key Triple-DES decrypt (supports only 192-bit size)	Data decryption * Legacy use only
A1948	AES-CBC (SP 800-38A)	AES	Encrypt/Decrypt, 128 and 256-bit	Data encryption, decryption, and key unwrap

CAVP Cert	Algorithm and Standard	Mode/Method	Description/ Key Size(s) / Key Strength(s)	Use / Function
A1948	AES-CMAC (SP 800-38B)	AES	CMAC generate and verify: 128, 192, and 256-bit Uses AES-CBC (SP 800-38A) (#A1948) as the underlying block cipher	MAC generation and verification
A1948	AES-GCM (KTS) (SP800-38D)	SP 800-38D and SP 800-38F. KTS (key wrapping and unwrapping) per IG D.G.	128, 192, and 256- bit keys providing 128, 192, or 256 bits of encryption strength	Data encryption, decryption, key-wrap, and key-unwrap
A1948	AES-GCM (SP 800-38D)	AES	GCM mode: Authenticated encrypt/decrypt; 128, 192, and 256-bitUses AES-CBC (SP 800-38A) (#A1948) as the underlying block cipher	Authenticated data encryption and decryption, key unwrap
A1948	AES-KW (KTS) (SP 800-38F)	SP 800-38F. KTS (key wrapping and unwrapping) per IG D.G.	128, 192 and 256-bit keys providing 128-bit, 192-bit, or 256 bits of encryption strength	Key wrapping/unwrapping
A1948	AES-KW (SP 800-38F)	AES	KW, 128, 192, and 256-bit Uses AES-CBC (#A1948) as underlying block cipher	Key wrapping/unwrapping
A1948	AES-KWP (KTS) (SP 800-38F)	SP 800-38F. KTS (key wrapping and unwrapping) per IG D.G.	128, 192 and 256-bit keys providing 128-bit, 192-bit, or 256 bits of encryption strength	Key wrapping/unwrapping
A1948	AES-KWP (SP 800-38F)	AES	KWP, 128, 192, and 256-bit Uses AES-CBC (#A1948) as underlying block cipher	Key wrapping/unwrapping
A1948	Counter DRBG (SP 800-90Ar1)	Counter DRBG	AES 256 with df No prediction resistance Uses AES-CBC (SP 800-38A) (#A1948) as the underlying block cipher	Random number generation for user, internal IVs, and salt
A1948	ECDSA KeyGen (FIPS186-4)	Key Gen	Key Gen: P-224, P-256, P-384, P-521 Uses Counter DRBG (SP800-90Ar1) (#A1948) as underlying Random Generator	Key generation
A1948	ECDSA KeyVer (FIPS186-4)	ECDSA KeyVer (FIPS186-4)	Key Ver; P-224, P-256, P-384 and P-521	Key Verification
A1948	ECDSA SigGen (FIPS186-4)	ECDSA SigGen (FIPS186-4)	Signature generation: P-224, P-256, P-384, P-521 (SHA2-256, SHA2-384, SHA2-512) Uses Hash DRBG (SP800-90Ar1) (#A1947) as underlying Random generator Uses SHA2-256 (FIPS 180-4) (#A1948), SHA2-384 (FIPS 180-4) (#A1948) and SHA2-512 (FIPS 180-4) (#A1948) as underlying digest algorithm	Signature Generation
A1948	ECDSA SigVer (FIPS186-4)	ECDSA SigVer(FIPS186-4)	Signature verify: P-224, P-256, P-384, P-521 (SHA- 1, SHA2-256, SHA2-384, and SHA2-512) Uses SHA-1 (FIPS 180-4) (#A1948), SHA2-256 (FIPS 180-4) (#A1948) and SHA2-	Signature Verification

CAVP Cert	Algorithm and Standard	Mode/Method	Description/ Key Size(s) / Key Strength(s)	Use / Function
			384 (FIPS 180-4) (#A1948) and SHA2-512 (FIPS 180-4) (#A1948) as underlying digest algorithm	
A1948	HMAC-SHA-1 (FIPS 198-1)	HMAC	HMAC-SHA-1 Uses SHA-1 (FIPS 180-4) (#A1948) as underlying digest algorithm	MAC generation, Verify and KDF
A1948	HMAC-SHA2-256 (FIPS 198-1)	HMAC	HMAC-SHA2-256 Uses SHA2-256 (FIPS 180-4) (#A1948) as underlying digest algorithm	MAC generation, verify, KAS, and KDF
A1948	HMAC-SHA2-384 (FIPS 198-1)	HMAC	HMAC-SHA2-384 Uses SHA2-384 (FIPS 180-4) (#A1948) as underlying digest algorithm	MAC generation, verify, KAS, and KDF
A1948	HMAC-SHA2-512 (FIPS 198-1)	HMAC	HMAC-SHA2-512 Uses SHA2-512((FIPS 180-4) (#A1948) as underlying digest algorithm	MAC generation, verify, KAS, and KDF
A1948	KAS-ECC (KAS) (SP 800-56Ar3)	SP 800-56Arev3. KAS-ECC per IG D.F Scenario 2 path (2).	P-521 curve providing 256 bits of encryption strength	Cloning protocol
A1948	KAS-ECC SP800-56Ar3	KAS	ECC CDH Ephemeral Unified P-521 with OneStep KDF (HMAC-SHA2-512) Uses Counter DRBG (#A1948) as underlying Random number Uses ECDSA KeyGen (FIPS186-4) (#A1948) as underlying ECC KeyGen and ECDSA KeyVer (#A1948) as underlying ECDSA Key Verification algorithm.\ Uses SHA2-512) (FIPS 180-4) as underlying digest algorithm	Cloning protocol

CAVP Cert	Algorithm and Standard	Mode/Method	Description/ Key Size(s) / Key Strength(s)	Use / Function
A1948	KAS-ECC-SSC SP800-56Ar3	KAS	ECC CDH Static Unified: P-224, P-256, P384, P-521, B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571 P-224, P-256, P384 and P-521 providing 112 bits,128 bits. 192 bits and 256 bits of encryption strength respectively Uses Counter DRBG (SP 800-90Ar1) (#A1948) as underlying Random Generator for the ECDSA KeyGen key pair Uses ECDSA KeyVer (FIPS 186-4) (#A1948) and ECDSA KeyGen (FIPS 186-4) (#A1948) underlying verification and generation for the key pair Uses SHA2-256 (FIPS 180-4) (#A1948), SHA2-384 (FIPS 180-4) (#A1948), SHA2-512 (FIPS 180-4) (#A1948) as the underlying digest algorithm	Shared secret computation
A1948	KAS-IFC-SSC (SP 800-56Br2)	KAS	RSA-based key exchange scheme KAS1 and KAS2 RSA 2048, 3072, and 4096-bit 2048, 3072, and 4096-bit modulus providing 112, 128, and 150 bits of encryption strength respectively Uses Counter DRBG (SP 800-90Ar1) (#A1948) as underlying Random Generator for the RSA Key Pair Generation Uses SHA2-256 (#A1948), SHA2-384 (#A1948) and SHA2-512 (#A1948) as underlying hash algorithm Uses RSA KeyGen (FIPS 186-4) [#A1948] as underlying RSA key generation algorithm	PEK and KLK generation and certificate authentication
A1948	KDA HKDF SP800-56Cr1	KDA HKDF SP800-56Cr1	HKDF Uses Counter DRBG (SP 800-90Ar1) (#A1948) as underlying Random Generator	ECDH key derivation and ECDH key wrap

CAVP Cert	Algorithm and Standard	Mode/Method	Description/ Key Size(s) / Key Strength(s)	Use / Function
			Uses HMAC-SHA2-256 (#A1948), HMAC-SHA2-384 (#A1948) and HMAC-SHA2-512 (#A1948) as underlying MAC algorithm	
A1948	KDA OneStep SP800-56Cr1	KDA OneStep SP800-56Cr1	Hash-based KDF Uses Counter DRBG (SP 800-90Ar1) (#A1948) as underlying Random Generator Uses HMAC-SHA2-256 (#A1948), HMAC-SHA2-384 (#A1948) and HMAC-SHA2-512 (#A1948) as underlying MAC algorithm	ECDH key derivation and ECDH key wrap
A1948	KDA TwoStep SP800-56Cr1	KDA TwoStep SP800-56Cr1	HMAC/CMAC-based extract-expand KDFs Uses Counter DRBG (SP 800-90Ar1) (#A1948) as underlying Random Generator Uses HMAC-SHA2-256 (#A1948), HMAC-SHA2-384 (#A1948) and HMAC-SHA2-512 (#A1948) as underlying MAC algorithm Uses AES-CBC (#A1948) as underlying Block cipher for AES-CMAC	ECDH key derivation and ECDH key wrap
A1948	KDF ANS 9.63 (CVL) (SP 800-135r1)	KDF ANS 9.63	SHA-2 224, 256, 384, and 512 Uses SHA2-256 (#A1948), SHA2-384 (#A1948) and SHA2-512 (#A1948) as underlying digest algorithm	Key derivation and key agreement schemes
A1948	KDF SP800-108	KDF SP800-108	HMAC-SHA-1, SHA2-224, SHA2-256, 384, and 512 KDF Uses HMAC-SHA-1 (#A1948), HMAC-SHA2-256 (#A1948), HMAC-SHA2-384 (#A1948) and HMAC-SHA2-512 (#A1948) as underlying MAC algorithm	Key derivation
A1948	KTS-IFC (KTS) (SP 800-56Br2)	SP 800-56Brev2. KTS-IFC (key encapsulation and un-encapsulation) per IG D.G.	2048, 3072, or 4096-bit modulus providing 112, 128, or 150 bits of encryption strength	Asymmetric key encapsulation and un-encapsulation
A1948	KTS-IFC (SP 800-56Br2)	KTS	KTS-OAEP-basic RSA 2048, 3072, and 4096-bit Uses counter DRBG (SP800-90Ar1) (#A1948) as underlying Random	Asymmetric key encapsulation and un-encapsulation

CAVP Cert	Algorithm and Standard	Mode/Method	Description/ Key Size(s) / Key Strength(s)	Use / Function
			Generator Uses HMAC-SHA2-256 (FIPS 180-4) (#A1948), HMAC-SHA2-384 (FIPS 180-4) (#A1948), HMAC-SHA2-512 (FIPS 180-4) (#A1948) as underlying digest algorithm Uses RSA KeyGen (FIPS 186-4) (#A1948) as underlying key generation algorithm	
A1948	PBKDF (SP 800-132)	PBKDF	HMAC with SHA-1, SHA-2 256, 384, and 512 Uses HMAC-SHA-1 (#A1948), HMAC-SHA2-256 (#A1948), HMAC-SHA2-384 (#A1948) and HMAC-SHA2-512 (#A1948) as underlying MAC algorithm	User credentials storage. Master key derived from PBKDF is not used for deriving keys for data encryption/protection, but instead used only for storing passwords as hashes.
A1948	RSA Decryption Primitive (SP 800-56Br2) (CVL)	RSA Decryption Primitive	Modulus sizes: 2048, 3072, and 4096-bit	RSA key transport
A1948	RSA KeyGen (FIPS186-4)	RSA KeyGen (FIPS186-4)	Key generation: 2048, 3072, and 4096-bit Uses Counter DRBG (SP800-90Ar1) (#A1948) as underlying Random Generator	RSA key generation
A1948	RSA SigGen (FIPS186-4)	RSA SigGen (FIPS186-4)	FIPS 186-4 PKCS #1 1.5 and PSS Sig Gen: 2048, 3096, and 4096-bit (SHA-2 256, 384, and 512) Uses SHA2-256 (FIPS 180-4) (#A1948), SHA2-384 (FIPS 180-4) (#A1948), SHA2-512 (#A1948) as underlying digest algorithm 2048, 3072, and 4096-bit modulus providing 112, 128, and 150 bits of encryption strength respectively.	Signature generation
A1948	RSA SigVer (FIPS186-4)	RSA SigVer (FIPS186-4)	FIPS 186-4 PKCS #1 1.5 and PSS SigVer: 1024, 2048, 3072, and 4096-bit (SHA-1, SHA-2 256, 384, and 512) Uses SHA-1 (FIPS 180-4)(#A1947) SHA2-256 (FIPS 180-4) (#A1948), SHA2-384 (FIPS 180-4) (#A1948), SHA2-512 (#A1948) as underlying digest algorithm 2048, 3072, and 4096-bit modulus providing 112, 128, and 150 bits of encryption strength respectively.	Signature verification User authentication Firmware update verification
A1948	SHA-1 (FIPS 180-4)	SHA-1	SHA-1	Digests, HMAC, and KDFs
A1948	SHA2-256 (FIPS 180-4)	SHA2	SHA2-256	Digests, HMAC, signature generation, and KDFs

CAVP Cert	Algorithm and Standard	Mode/Method	Description/ Key Size(s) / Key Strength(s)	Use / Function
A1948	SHA2-384 (FIPS 180-4)	SHA2	SHA2-384	Digests, HMAC, signature generation, and KDFs
A1948	SHA2-512 (FIPS 180-4)	SHA2	SHA2-512	Digests, HMAC, signature generation, and KDFs
A1948	TDES-ECB	TDES	Triple-DES 3-key decrypt.	Prerequisite for TDES-KW. * Legacy use only
A1948	TDES-KW (KTS) (SP 800-38F)	SP 800-38F. KTS (key unwrapping) per IG D.G	192-bit key providing 112-bit encryption strength	Key unwrapping * Legacy use only
A1948	TDES-KW (SP 800-38F)	TDES	TKW 3-key DES decrypt Uses TDES-ECB (#A1948) as underlying block cipher	Key unwrapping * Legacy use only
A2393	ECDSA KeyGen (FIPS186-4)	ECDSA KeyGen (FIPS186-4)	Key Gen: P-224, P-256, P-384, P-521 Uses Hash DRBG (SP800-90Ar1) (#A1947) as underlying Random Generator for the ECDSA KeyGen key pair	Key Generation
A2393	KTS-IFC (KTS) (SP 800-56Br2)	SP 800-56Brev2. KTS-IFC (key encapsulation and un-encapsulation) per IG D.G.	2048, 3072, or 4096-bit modulus providing 112, 128, or 150 bits of encryption strength	Asymmetric key encapsulation and un-encapsulation in hybrid environment
A2393	KTS-IFC (SP 800-56Br2)	KTS	RSA key wrap and unwrap of symmetric keys in KTS-OAEP-Basic padding. 2048, 3072, 4096-bit modulus Uses Counter DRBG (SP800-90Ar1) (#A1948) as underlying Random Generator Uses HMAC-SHA2-256 (FIPS 180-4) (#A1947), HMAC-SHA2-384 (FIPS 180-4) (#A1947), HMAC-SHA2-512 (FIPS 180-4) (#A1947) as underlying digest algorithm Uses RSA KeyGen (FIPS 186-4) (#A2393) as underlying key generation algorithm	Asymmetric key encapsulation and un-encapsulation in hybrid environment
A2393	RSA KeyGen (FIPS186-4)	RSA KeyGen (FIPS186-4)	Key generation: 2048, 3072, and 4096-bit Uses Hash DRBG (SP800-90Ar1) (#A1947) as underlying Random Generator for the RSA KeyGen key pair	RSA key generation in hybrid environment
KAS-ECC-SSC SP800-56Ar3 (#A1947) and KDF TLS (CVL) (#A1947)	KAS TLS (SP 800-56Ar3)	SP 800-56Arev3. KAS-ECC per IG D.F Scenario 2 path (2)	P-224, P-256, P-384, P-521 curves providing 112, 128, 192, or 256 bits of encryption strength	TLS
KAS-ECC-SSC SP800-56Ar3 (#A1948) KDF ANS 9.63 (CVL) (SP 800-135r1) (#A1948)	KAS ANS 9.63 (SP 800-56Ar3)	SP 800-56Arev3. KAS-ECC per IG D.F Scenario 2 path (2).	P-224, P-256, P-384, and P-521 curves providing 112, 128, 192, or 256 bits of encryption strength	ECDH key derivation and ECDH-AES key wrap

CAVP Cert	Algorithm and Standard	Mode/Method	Description/ Key Size(s) / Key Strength(s)	Use / Function
KAS-ECC-SSC SP800-56Ar3 (#A1948), KDA HKDF SP800-56Cr1 (#A1948)	KAS KDA HKDF (SP 800-56Ar3)	SP 800-56Arev3. KAS-ECC per IG D.F Scenario 2 path (2).	P-224, P-256, P-384, and P-521 curves providing 112, 128, 192, or 256 bits of encryption strength	ECDH key derivation and ECDH-AES key wrap
KAS-ECC-SSC SP800-56Ar3 (#A1948), KDA OneStep SP800-56Cr1 (#A1948)	KAS KDA ONESTEP (SP 800-56Ar3)	SP 800-56Arev3. KAS-ECC per IG D.F Scenario 2 path (2).	P-224, P-256, P-384, and P-521 curves providing 112, 128, 192, or 256 bits of encryption strength	ECDH key derivation and ECDH-AES key wrap
KAS-ECC-SSC SP800-56Ar3 (#A1948), KDA TwoStep SP800-56Cr1 (#A1948)	KAS KDA TWOSTEP (SP 800-56Ar3)	SP 800-56Arev3. KAS-ECC per IG D.F Scenario 2 path (2).	P-224, P-256, P-384, and P-521 curves providing 112, 128, 192, or 256 bits of encryption strength	ECDH key derivation and ECDH-AES key wrap
KAS-IFC-SSC (SP 800-56Br2) (#A1948) KDA HKDF SP800-56Cr1 (#A1948)	KAS-IFC HKDF (SP800-56Br2)	SP 800-56Brev2. KAS-IFC per IG D.F Scenario 1 path (2).	2048-bit, 3072-bit and 4096-bit modulus providing between 112 and 150 bits of encryption strength	PEK and KLK generation and certificate authentication
KAS-IFC-SSC (SP 800-56Br2) (#A1948) KDA OneStep SP800-56Cr1 (#A1948)	KAS-IFC OneStep (SP800-56Br2)	SP 800-56Brev2. KAS-IFC per IG D.F Scenario 1 path (2).	2048-bit, 3072-bit and 4096-bit modulus providing between 112 and 150 bits of encryption strength	PEK and KLK generation and certificate authentication
KAS-IFC-SSC (SP 800-56Br2) (#A1948) KDA TwoStep SP800-56Cr1 (#A1948)	KAS-IFC TwoStep (SP800-56Br2)	SP 800-56Brev2. KAS-IFC per IG D.F Scenario 1 path (2).	2048-bit, 3072-bit and 4096-bit modulus providing between 112 and 150 bits of encryption strength	PEK and KLK generation and certificate authentication
N/A	ENT (P) (SP 800-90B)	N/A	SP 800-90B entropy source	Entropy Source
Vendor Affirmed	CKG SP 800-133Rev2	CKG	Please refer to section 2.3.2 Algorithm Specific Information	Cryptographic Key Generation; SP 800-133Rev2 and IG D.H

2.3.2 Algorithm Specific Information

- AES-GCM (#A1947)
  - IG C.H Scenario #1:
    - TLS 1.2 or other applications can offload GCM operations.
    - For TLS-1.2 protocol, IV constructed as described in RFC 5288.
    - Refer Section 2.4 for the TLS 1.2 AES GCM supported cipher suites
    - IV is generated internally to the cryptographic module.
    - The module triggers a handshake to establish new encryption keys and IVs when the IV exhausts the maximum possible values for the given session key.
    - SP 800-38D §8.2.2 is used for GCM IV construction.
  - IG C.H Scenario #2:
    - IVs are generated randomly and IG C.H Option #2 applies.
    - IV's free field is a 4-byte counter
    - IV's random field is a 96-bit random number.

        - IV's random field is incremented by 1. IV's random field wouldn't overflow 96-bits in the lifetime of the module.
        - For IV restoration conditions guidance, refer to section 11.5 User Guidance.
        - Internal Approved RBG (Hash DRBG #A1947): SP 800-90A DRBG, HASH_DRBG SHA2-512.
- AES-GCM (#A1948)
    - IG C.H Notes:
        - IVs are generated randomly, and IG C.H Option #2 applies.
        - IV is generated internally to the cryptographic module.
        - SP 800-38D §8.2.2 is used for GCM IV construction.
        - IV's random field is a 128-bit random number.
        - For IV restoration conditions guidance, refer to section 11.5 User Guidance.
        - Approved RBG (Hash DRBG #A1947): SP 800-90Ar1 DRBG, HASH_DRBG SHA2-512
- CKG SP 800-133Rev2 (Vendor affirmed)
    - IG D.H
        - SP 800-133Rev2 Section 5.1 Asymmetric signature key generation using unmodified DRBG output
        - SP 800-133Rev2 Section 5.2 Asymmetric key establishment key generation using unmodified DRBG output
        - SP 800-133Rev2 Section 6.1 Direct symmetric key generation using unmodified DRBG output
        - SP 800-133Rev2 Section 6.2.1 Derivation of symmetric keys from a key-agreement shared secret.
        - SP 800-133Rev2 Section 6.2.2 Derivation of symmetric keys from a pre-shared key
- PBKDF (SP 800-132) (#A1948)
    - PBKDF with HMAC password strength
        - The password is a minimum of 8 characters, case-sensitive alpha-numeric. As such there are (26*2+10)^8 = 62^8 possible minimum-length passwords, and the false acceptance rate is 1 in 62^8 which is less than 1 in 1,000,000.
        - A maximum of 20 password attempts are possible before permanent lockout. Therefore the probability of false authentication over any timeframe is 20 in 62^8, which is less than 1 in 100,000. (The number of allowed login attempts prior to lockout is configured during module initialization but cannot exceed 20.)
        - Lockout of MCO automatically zeroizes the module in the next reboot. In all other cases, lockout can be unset by destroying the partition.
    - PBKDF with HMAC Iteration Count and Justification
        - Iteration count should be at least 1000, following the recommendation in SP800-132r2.
        - Salt length is 128 to 4096 bits.

2.3.3 Non-Approved Allowed Algorithms Used in the Module

The cryptographic module supports the following non-Approved algorithms, which are allowed for use in Approved mode.

Table 4 Non-Approved Algorithms Allowed in the Approved Mode of Operation

Algorithm	Caveat	Use/Function
AES	Cert A1947, Key unwrapping. Per IG D.G.	Legacy Key unwrap only • ECB mode: Decrypt; 128, 192, and 256-bit • CBC mode: Decrypt; 128, 192, and 256-bit
AES	Cert A1948, Key unwrapping. Per IG D.G	Legacy Key unwrap only • ECB mode: Decrypt; 128, 192, and 256-bit • CBC mode: Decrypt; 128, 192, and 256-bit

Algorithm	Caveat	Use/Function
EC Diffie-Hellman with non-NIST recommended curves	Cert A1947, Provides between 112 and 256 bits of encryption strength. Per IGs D.F and C.A.	EC-DH Secp224k1(112 bits), Secp256K1 (128 bits) brainpoolP224r1(112 bits), brainpoolP256r1(128 bits), brainpoolP320r1(160 bits), brainpoolP384r1(192 bits), brainpoolP512r1(256 bits) FRP256v1 (128 bits) • Prime order curve, generated as per FIPS 186-4 Section 6.1.1 (SHA-1*, SHA2-224, SHA2-256, SHA2-384, SHA2-512)
ECDSA with non-NIST recommended curves	Cert A1947, Provides between 112 and 256 bits of encryption strength. Per IG C.A.	EC Key generation, sign Secp256K1 (128 bits) brainpoolP224r1(112 bits), brainpoolP256r1(128 bits), brainpoolP320r1(160 bits), brainpoolP384r1(192 bits), brainpoolP512r1(256 bits) FRP256v1 (128 bits) • Prime order curve, generated as per FIPS 186-4 Section 6.1.1 (SHA-1*, SHA2-224, SHA2-256, SHA2-384, SHA2-512)

- This functionality of the module is used by the user of the module as part of TLS protocol negotiation. The TLS protocol has not been reviewed or tested by the CAVP or CMVP.

2.3.4 Non-Approved Algorithms Allowed in Approved Mode of Operation with No Security Claimed

The cryptographic module supports the following non-Approved algorithms; they are allowed in the Approved mode of operation with no security claimed.

Table 5 Non-Approved Algorithms Allowed in the Approved Mode of Operation with No Security Claimed

Algorithm	Caveat	Use/Function
SHA-1	No security claimed per IG 2.4.A	Fingerprints
Triple-DES SP 800-38B	No security claimed per IG 2.4.A	Fingerprints Key Sizes • 192-bit (Generation, Verify)

2.3.5 Non-Approved, Non-Allowed Algorithms

The cryptographic module supports the following non-Approved algorithms available only in non-Approved mode of operation.

Table 6 Non-Approved Algorithms Not Allowed in the Approved Mode of Operation

Algorithm	Use/Function
AES (non-compliant)	Key wrap (TR31/TR34/AES-CBC/AES-GCM, wrap/unwrap), DecimalTable/Data/PIN Encryption/Decryption. FF1/FF3-1 Data Encryption/Decryption • In Non-Approved mode, AES GCM supports the IV length from 1 byte to 16 bytes

Algorithm	Use/Function
DES	Derive unique key per transaction (DUKPT) EMV key derivation. Derive PIN from Offset Derive Offset from PIN PIN Verification PVV generation and Verification CVV generation and verification Export Symmetric key/Export Asymmetric key pair using TR31 wrapping. Import/Export using TR34. Import Decimal Table EMV script. EMV ARQC/ARPC Data/PIN encryption/decryption
DES MAC	MAC generation and Verification
Double-DES	Derive unique key per transaction (DUKPT), EMV key derivation. Derive PIN from Offset Derive Offset from PIN PIN Verification PVV generation and Verification CVV generation and verification Export Symmetric key/Export Asymmetric key pair using TR31 wrapping. Import/Export using TR34 Import Decimal Table EMV script. EMV ARQC/ARPC Data/PIN encryption/decryption
EC-AES	EC-AES wrap/unwrap (EC BYOK)
ECDH KDF	Key derivation using ECDH followed by HMAC/CMAC counter KDF
ECDSA (non-compliant)	Key generation, Sign, Verify P192, Secp192k1, brainpoolP160r1, brainpool192r1, K-163 and B-163 (SHA-1, SHA-224, SHA-256, SHA-384, SHA-512)
EDDSA (non-compliant)	Key generation, Sign, Verify
KAS-ECC (non-compliant)	EC Key generation and ECDH Curve25519 (128 bits), Curve448 (224 bits)
ML-DSA (non-compliant)	Key generation, sign, verify
ML-KEM (non-compliant)	Key generation, Asymmetric key encapsulation and un-encapsulation
PBE	Key generation
RSA (non-compliant)	TR34 Import TR34 Export PIN block decryption BYOK Encrypt/Decrypt Asymmetric key encapsulation and un-encapsulation using PKCS#1-v1.5 padding with modulus size 2048, 3072 and 4096 bits Key generation, Sign, Verify (1024-bit)
Shamir’s Key Share	Key share
Triple-DES (non-compliant)	Derive unique key per transaction (DUKPT) EMV key derivation. Derive PIN from Offset

Algorithm	Use/Function
	Derive Offset from PIN PIN Verification PVV generation and verification CVV generation and verification Export Symmetric key/Export Asymmetric key pair using TR31 wrapping Import/Export using TR34 Import Decimal Table EMV script. EMV ARQC/ARPC Data/PIN encryption/decryption

2.4 TLS 1.2 Cipher Suites

The module supports the algorithms for the following cipher suites using Approved and allowed algorithms and key sizes:

- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

For cipher suites using GCM, the IV is generated per RFC 5288. The module supports GCM cipher suites compatible with SP 800-52 Rev2.

2.5 Module Photograph

The following images depict the module's physical and logical cryptographic boundary.

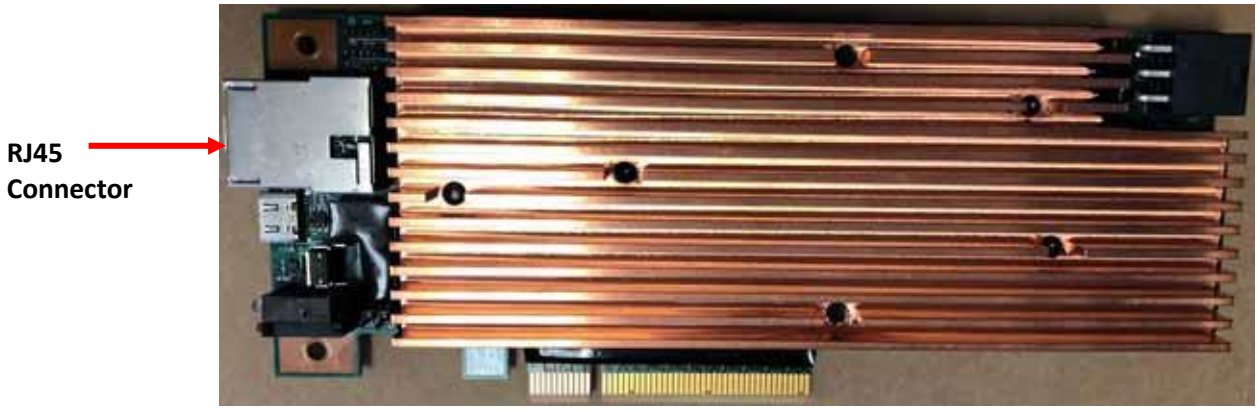


Figure 1 Top View of Cryptographic Module

Figure 2 Bottom View of Cryptographic Module

3 Cryptographic Module Interfaces

The module ports and interfaces are described in the below table.

Table 7 Ports and Interfaces

Physical Port	Logical Interface*	Data Passing Over Port/Interface
USB FTDI USB to Multi-Channel Serial SPI/I2C bridge *FTDI dual port USB endpoint* Channel A: OCTEON UART0 Channel B: MCU UART2 OR *FTDI quad port USB end point* Channel A: OCTEON UART0 Channel B: not connected Channel C: MCU UART2 Channel D: not connected	Status output	OCTEON UART • Log messages MCU UART, SMBus • Diagnostics • Log messages
PCIe	Control input	PCIe configuration is read and written; no other.
	Data input, data output, and status output	Primary interface to communicate with the module. Provides APIs for the software on the host to communicate with the module.
	Power	N/A
PCIe SMBus	Status output	Diagnostics, log messages System management and log reading
UART	Status output	Log messages
LED	Status output	Status output
Tamper PIN	Control input	No data; only a signal from high to low
Zeroize push button	Control input	No data;
Power connector	Power	No data; external power connector
Battery interfaces	Power	No data; external power connector (only to MCU)

* The module does not contain any control output interface.

* The RJ45 Connector depicted in Figure 1 is disabled in firmware.

3.1 PCIe Data Interface

The PCIe data interface is the only interface that accepts the security services always accessible from the module; the host system cannot read or write data over this interface. The module will start reading commands from the Host system only after the power-on self-tests are run and firmware is loaded.

After the module is zeroized, the firmware enforces that the PCIe data interface only provides basic versioning and informational output.

When the module enters the error state, the PCIe data interface will report the error state and no other data.

3.2 Other Interfaces

Other interfaces are all meant for informational services to read temperature, logs, and diagnostic information but no other data.

4 Roles, Services, and Authentication

The module supports different operator roles. One identity is allowed for each role, per partition. Username is used as the identity of a user. This means for a given partition, each username needs to be unique per predefined roles.

Master Partition Roles:

- **Master Crypto Officer (MCO) -** The Master Crypto Officer role (MCO) is allowed only on the master partition, which is mandatory to use the HSM. This role has access to administrative services offered by the module or HSM. This role is used to configure non-master partitions (create, provision, resize, and delete) but cannot access their resources (e.g., cannot manage or use non-master partition keys).

Master partition supports only MCO role in addition to UN-AUTH operator roles as described in Table 8.

Non-Master Partition Roles:

- **Pre-Crypto Officer (Pre-CO)** - This role is an optional role with limited functionality; eventually transitions into PCO. During partition initialization, default credentials are used to create a Pre-CO or a PCO. The Pre-CO is a restricted role primarily for configuring certificates and setting up a PCO. After a PCO is set up for a partition, the Pre-CO role is no longer accessible.
- **Partition Crypto Officer (PCO) -** This role has access to administrative services of the partition and can configure PCU and AU identities. The HSM supports more than one Crypto Officer role with a requirement there shall be at least one.
- **Partition Crypto User (PCU) -** This role has access to all cryptographic services offered by the partition; its purpose is operational use of the module.
- **Appliance User (AU) -** This role has access to partition audit logs and can create end-to-end encrypted channels. It is used to set up and synchronize clusters.

Each non-master partition supports these four (4) distinct operator roles (Pre-CO, PCO, PCU, and AU) in addition to UN-AUTH as described in Table 8. The module enforces the separation of roles using identity-based authentication. Re-authentication is required to change roles.

Except for Pre-CO, concurrent operators are allowed; however, only one operator is allowed per login session.

4.1 Roles, Services, and CSP Access

Service interface documentation details specific service inputs and outputs:

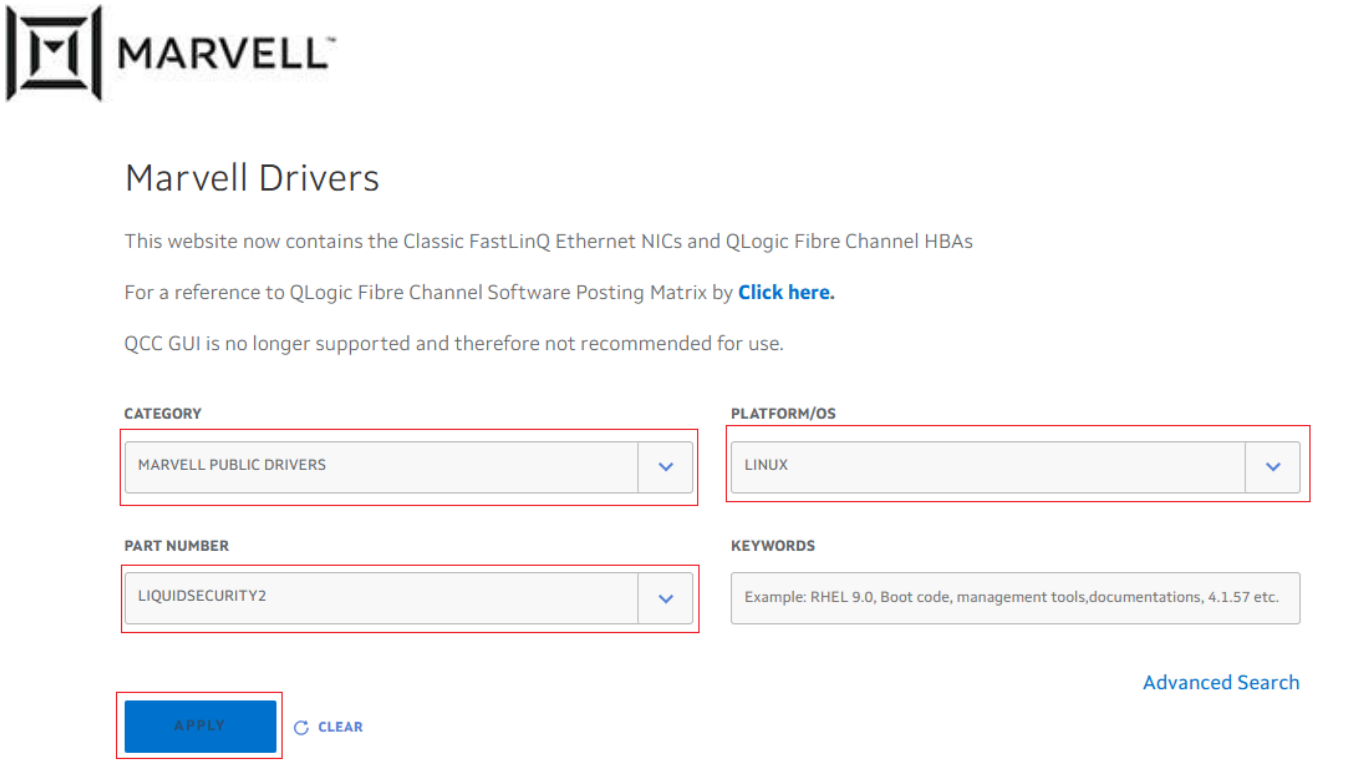
Document name: LiquidSecurity2-10.23-1150-Driver-APIs.zip
Version: 10.23-1150
Release date: 09/26/2025

To access the document, complete the below steps.

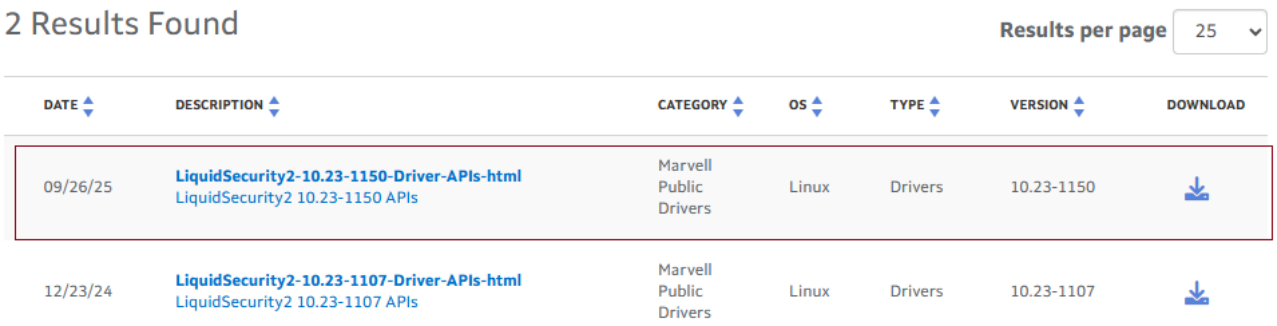
1. Open the following link to open the Marvell Public Driver Downloads page:

   https://www.marvell.com/support/downloads.html

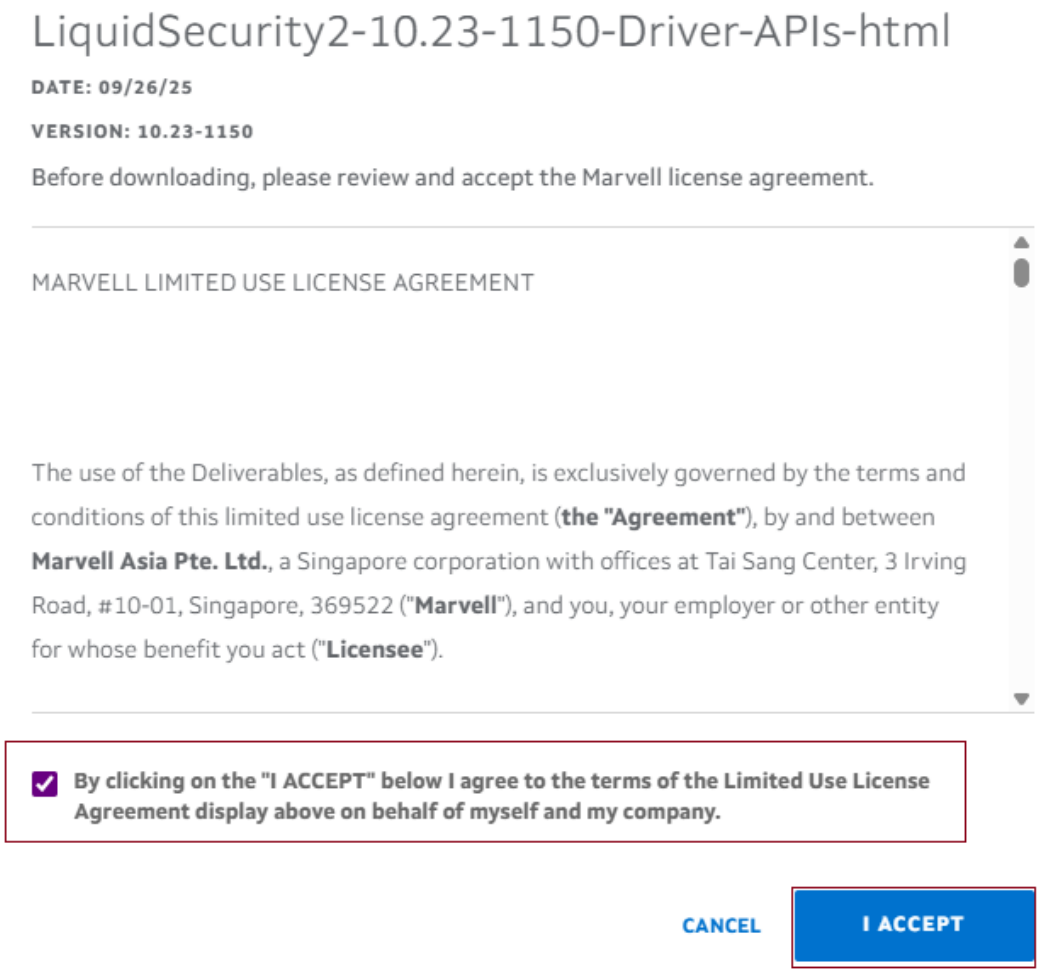
2. Choose CATEGORY, PLATFORM, and PART NUMBER as shown in the following screenshot; then click the **APPLY** button.



Select “LiquidSecurity2-10.23-1150-Driver-APIs-html" to download.

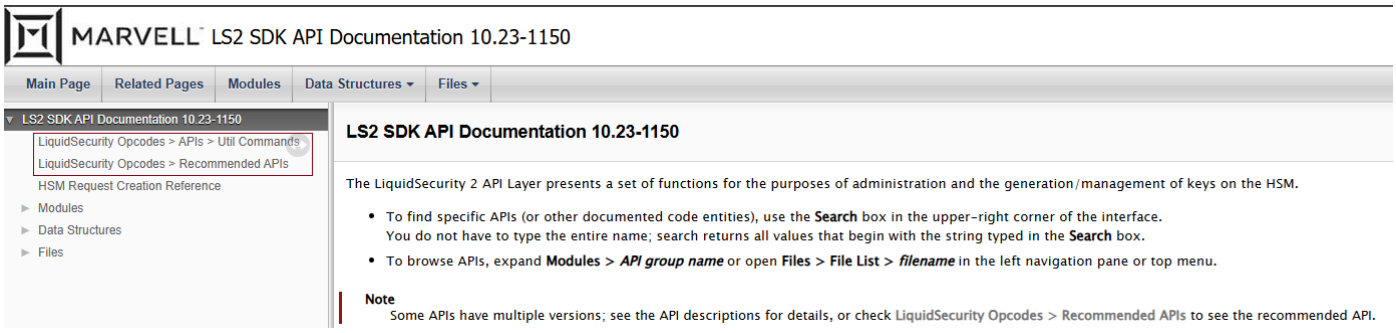


3. This pops up a window to accept the “MARVELL LIMITED USE LICENSE AGREEMENT”.



Click **I ACCEPT** to accept the terms and conditions; the Service Interface document will be downloaded.

4. After the Interface document is downloaded, extract the archive with the password “LS-FIPS-140-3”.
5. To access the Services, open the `index.html` file; then select LiquidSecurity Opcodes > Recommended API(s) from the left-pane. The following interface displays.



The module's primary service inputs are opcodes. An API is provided to the operator on the host system to invoke these opcodes.

The host API is called, which packetizes services and calls the PCIe host driver to send the request from the host to the HSM. The HSM processes the service requests and passes the response from HSM to the host through PCIe host Driver and API.

NOTE: The new API documentation 10.23-1150 also covers the API documentation version 10.02-1102 and 10.23-1107.

Table 8 Roles, Service Commands, Input and Output

Role	Service	Input*	Output*
MCO/PCO/PCU/AU/UN-AUTH	CN_ZEROIZE	Opcode inputs	Opcode outputs
MCO	CN_VENDOR_ZEROIZE	Opcode inputs	Opcode outputs
UN-AUTH	CN_APP_INITIALIZE	Opcode inputs	Opcode outputs
MCO/PCO/PCU/AU/UN-AUTH	CN_APP_FINALIZE	Opcode inputs	Opcode outputs
MCO/PCO/PCU/AU/UN-AUTH	CN_OPEN_SESSION	Opcode inputs	Opcode outputs
MCO/PCO/PCU/AU/UN-AUTH	CN_CLOSE_SESSION	Opcode inputs	Opcode outputs
MCO/PCO/PCU/AU/UN-AUTH	CN_GET_SESSION_INFO	Opcode inputs	Opcode outputs
MCO/PCO/PCU/AU/UN-AUTH	CN_CLOSE_ALL_SESSIONS	Opcode inputs	Opcode outputs
MCO/PCO	CN_CLOSE_PARTITION_SESSIONS	Opcode inputs	Opcode outputs
UN-AUTH	CN_ENCRYPT_SESSION	Opcode inputs	Opcode outputs
UN-AUTH	CN_AUTHORIZE_SESSION	Opcode inputs	Opcode outputs
UN-AUTH	CN_LOGIN	Opcode inputs	Opcode outputs
MCO/PCO/PCU/AU	CN_LOGOUT	Opcode inputs	Opcode outputs
PCU	CN_ALWAYS_AUTHORIZE_USER	Opcode inputs	Opcode outputs
PCO/Pre-CO	CN_UPDATE_USER_DETAILS	Opcode inputs	Opcode outputs
PCO	CN_SET_USER_ATTR	Opcode inputs	Opcode outputs
MCO/PCO/PCU/AU/UN-AUTH	CN_TOKEN_INFO	Opcode inputs	Opcode outputs
MCO/PCO/PCU/AU/UN-AUTH	CN_PARTITION_INFO	Opcode inputs	Opcode outputs
MCO/PCO/PCU/AU/UN-AUTH	CN_GET_HSM_LABEL	Opcode inputs	Opcode outputs
MCO	CN_ALL_PARTITION_INFO	Opcode inputs	Opcode outputs
MCO/PCO/PCU/AU/UN-AUTH	CN_GET_POLICY_SET	Opcode inputs	Opcode outputs
MCO/PCO/PCU/AU/UN-AUTH	CN_GET_VERSION	Opcode inputs	Opcode outputs
Manufacturer	CN_STORE_VENDOR_PRE_SHARED_KEY (CN_STORE_KBK_SHARE)	Opcode inputs	Opcode outputs
MCO	CN_LOAD_RECOVERY_KEY_INIT CN_LOAD_RECOVERY_KEY_FINISH	Opcode inputs	Opcode outputs
MCO	CN_SHUTDOWN	Opcode inputs	Opcode outputs
MCO	CN_SET_INIT_TIME	Opcode inputs	Opcode outputs
MCO	CN_SET_VENDOR_TIME	Opcode inputs	Opcode outputs
MCO	CN_GET_TIME	Opcode inputs	Opcode outputs
MCO	CN_SYNC_TIME	Opcode inputs	Opcode outputs
MCO	CN_UCD_CMD	Opcode inputs	Opcode outputs
MCO	CN_GET_HSM_LOGGER_INFO	Opcode inputs	Opcode outputs
MCO/PCO/Pre-CO	CN_INIT_TOKEN	Opcode inputs	Opcode outputs
MCO/PCO/Pre-CO	CN_GEN_PSWD_ENC_KEY	Opcode inputs	Opcode outputs
MCO/PCO/Pre-CO	CN_CREATE_CO	Opcode inputs	Opcode outputs

Role	Service	Input*	Output*
MCO/PCO/Pre-CO	CN_INIT_DONE	Opcode inputs	Opcode outputs
MCO	CN_FW_UPDATE_BEGIN, CN_FW_UPDATE, CN_FW_UPDATE_END	Opcode inputs	Opcode outputs
MCO	CN_STORE_FW_SIGNING_KEY	Opcode inputs	Opcode outputs
MCO	CN_ALLOW_FW_UPDATE	Opcode inputs	Opcode outputs
MCO	CN_INVOKE_FIPS	Opcode inputs	Opcode outputs
PCO	CN_SET_HSM_CONFIG	Opcode inputs	Opcode outputs
MCO	CN_CREATE_PARTITION	Opcode inputs	Opcode outputs
MCO	CN_DELETE_PARTITION	Opcode inputs	Opcode outputs
MCO/PCO	CN_BACKUP	Opcode inputs	Opcode outputs
MCO/PCO	CN_RESTORE	Opcode inputs	Opcode outputs
MCO	CN_BACKUP_OBJECT	Opcode inputs	Opcode outputs
MCO	CN_RESTORE_OBJECT	Opcode inputs	Opcode outputs
PCO	CN_SET_NODEID	Opcode inputs	Opcode outputs
PCO	CN_CREATE_USER	Opcode inputs	Opcode outputs
PCO	CN_DELETE_USER	Opcode inputs	Opcode outputs
PCO/PCU	CN_LIST_USERS	Opcode inputs	Opcode outputs
PCO	CN_GET_LOGIN_FAILURE_CNT	Opcode inputs	Opcode outputs
Pre-CO	CN_CREATE_PRE_OFFICER	Opcode inputs	Opcode outputs
PCO	CN_CREATE_APPLIANCE_USER	Opcode inputs	Opcode outputs
UN-AUTH	CN_OPEN_SESSION_V2	Opcode inputs	Opcode outputs
UN-AUTH	CN_ENCRYPT_SESSION_V2	Opcode inputs	Opcode outputs
UN-AUTH	CN_GET_SERVER_PARAMS	Opcode inputs	Opcode outputs
PCO/PCU/AU	CN_GET_USER_INFO	Opcode inputs	Opcode outputs
PCU	CN_CREATE_OBJECT	Opcode inputs	Opcode outputs
PCO	CN_GEN_KEY_ENC_KEY	Opcode inputs	Opcode outputs
PCO/PCU/AU	CN_EXTRACT_MASKED_OBJECT	Opcode inputs	Opcode outputs
PCO/PCU/AU	CN_INSERT_MASKED_OBJECT	Opcode inputs	Opcode outputs
PCU	CN_DESTROY_OBJECT	Opcode inputs	Opcode outputs
PCU	CN_TOMBSTONE_OBJECT	Opcode inputs	Opcode outputs
PCO/PCU/AU	CN_DELETE_TOMBSTONED_OBJECT	Opcode inputs	Opcode outputs
PCU	CN_GET_ATTRIBUTE_VALUE	Opcode inputs	Opcode outputs
PCU	CN_GET_ATTRIBUTE_SIZE	Opcode inputs	Opcode outputs
PCU	CN_GET_ALL_ATTRIBUTE_SIZE	Opcode inputs	Opcode outputs
PCU	CN_GET_ALL_ATTRIBUTE_VALUE	Opcode inputs	Opcode outputs
PCO/PCU	CN_MODIFY_OBJECT	Opcode inputs	Opcode outputs
PCO	CN_MODIFY_KEY_OWNER	Opcode inputs	Opcode outputs
PCU	CN_GENERATE_KEY	Opcode inputs	Opcode outputs
PCU	CN_GENERATE_KEY_PAIR	Opcode inputs	Opcode outputs

Role	Service	Input*	Output*
PCU	CN_EXPORT_PUB_KEY	Opcode inputs	Opcode outputs
PCU	CN_GET_OBJECT_INFO	Opcode inputs	Opcode outputs
PCU	CN_UNWRAP_KEY/CN_UNWRAP_KEY2	Opcode inputs	Opcode outputs
PCU	CN_WRAP_KEY/CN_WRAP_KEY2	Opcode inputs	Opcode outputs
PCU	CN_NIST_AES_WRAP_UNWRAP/ CN_NIST_AES_WRAP_UNWRAP2	Opcode inputs	Opcode outputs
PCU	CN_DERIVE_KEY	Opcode inputs	Opcode outputs
MCO/PCO/PCU/AU	CN_FIND_OBJECTS_USING_COUNT/ CN_FIND_ALL_OBJECTS_IN_RANGE/ CN_FIND_ALL_OBJECTS/ CN_FIND_ALL_OBJECTS_USING_COUNT/ CN_FIND_OBJECTS/ CN_FIND_OBJECTS_FROM_INDEX	Opcode inputs	Opcode outputs
PCO/AU	CN_ADMIN_GET_PARTN_KEYHANDLES_HASH	Opcode inputs	Opcode outputs
PCO/AU/PCU	CN_GET_PARTN_SINGLE_KEYHANDLE_HASH	Opcode inputs	Opcode outputs
PCU	CN_ DESTROY _OBJECT	Opcode inputs	Opcode outputs
PCO/AU	CN_PARTN_GET_AUDIT_DETAILS	Opcode inputs	Opcode outputs
PCO/AU	CN_PARTN_GET_AUDIT_LOGS	Opcode inputs	Opcode outputs
PCO/AU	CN_PARTN_GET_AUDIT_SIGN	Opcode inputs	Opcode outputs
PCO/AU	CN_PARTN_ACK_AUDIT_SIGN	Opcode inputs	Opcode outputs
MCO	CN_FINALIZE_LOGS	Opcode inputs	Opcode outputs
MCO/PCO	CN_SET_POLICY	Opcode inputs	Opcode outputs
PCU	CN_NIST_AES_WRAP	Opcode inputs	Opcode outputs
PCU	CN_ALLOC_SSL_CTX	Opcode inputs	Opcode outputs
PCU	CN_FREE_SSL_CTX	Opcode inputs	Opcode outputs
PCU	CN_FIPS_RAND	Opcode inputs	Opcode outputs
PCU	CN_ME_PKCS_LARGE	Opcode inputs	Opcode outputs
PCU	CN_ME_PKCS	Opcode inputs	Opcode outputs
PCU	CN_FECC	Opcode inputs	Opcode outputs
PCU	CN_HASH	Opcode inputs	Opcode outputs
PCU	CN_SHA3	Opcode inputs	Opcode outputs
PCU	CN_HMAC	Opcode inputs	Opcode outputs
PCU	MAJOR_OP_AES_CMAC	Opcode inputs	Opcode outputs
PCU	CN_ENCRYPT_DECRYPT	Opcode inputs	Opcode outputs
PCU	MAJOR_OP_DECRYPT_AND_ENCRYPT_COMMAND	Opcode inputs	Opcode outputs
PCU	MAJOR_OP_ENCRYPT_DECRYPT_RECORD	Opcode inputs	Opcode outputs
PCO	CN_CERT_AUTH_GET_SOURCE_RANDOM	Opcode inputs	Opcode outputs
PCO	CN_CERT_AUTH_VALIDATE_PEER_CERTS/ CN_CERT_AUTH_VALIDATE_TARGET_CERTS	Opcode inputs	Opcode outputs
PCO	CN_CERT_AUTH_SOURCE_KEY_EXCHANGE	Opcode inputs	Opcode outputs
PCO	CN_CERT_AUTH_TARGET_KEY_EXCHANGE	Opcode inputs	Opcode outputs
PCO	CN_CLONE_SOURCE_INIT	Opcode inputs	Opcode outputs

Role	Service	Input*	Output*
PCO	CN_CLONE_SOURCE_STAGE1	Opcode inputs	Opcode outputs
PCO	CN_CLONE_TARGET_INIT	Opcode inputs	Opcode outputs
PCO	CN_CLONE_TARGET_STAGE1	Opcode inputs	Opcode outputs
PCO/PCU/AU/UN-AUTH	CN_LIST_AUTH_PUB_KEYS	Opcode inputs	Opcode outputs
PCO/PCU/AU/UN-AUTH	CN_GET_M_VALUE	Opcode inputs	Opcode outputs
PCO/PCU	CN_GET_TOKEN	Opcode inputs	Opcode outputs
PCO/PCU	CN_APPROVE_TOKEN	Opcode inputs	Opcode outputs
PCO/PCU	CN_LIST_TOKENS	Opcode inputs	Opcode outputs
PCO/PCU	CN_TOKEN_TIMEOUT	Opcode inputs	Opcode outputs
PCO/PCU	CN_DELETE_TOKEN	Opcode inputs	Opcode outputs
UN-AUTH	CN_FRAMLOG_CMD	None	Opcode outputs
MCO/PCO	CN_GET_CFG_PREGEN_CACHE_SZ	Opcode inputs	Opcode outputs
MCO/PCO	CN_GET_CFG_PREGEN_CACHE_VAL	Opcode inputs	Opcode outputs
PCO	CN_LIST_UNLINKED_OBJECTS	Opcode inputs	Opcode outputs
MCO/PCU	CN_GET_PARTN_FINGERPRINT	Opcode inputs	Opcode outputs
MCO/PCO/PCU/AU/UN-AUTH	CN_CERT_AUTH_GET_CERT	Opcode inputs	Opcode outputs
MCO/PCO	CN_CERT_AUTH_STORE_CERT	Opcode inputs	Opcode outputs
MCO/PCO	CN_GET_KBK_SLOT_INFO	Opcode inputs	Opcode outputs
MCO/PCO	CN_SET_KBK_PRIMARY	Opcode inputs	Opcode outputs
PCU	CN_SHARE_OBJECT	Opcode inputs	Opcode outputs
UN-AUTH	CN_UNLOCK_CO	Opcode inputs	Opcode outputs
MCO/PCO	CN_UNLOCK_USER	Opcode inputs	Opcode outputs
UN-AUTH	CN_GET_CORE_DUMP	Opcode inputs	Opcode outputs
MCO	CN_MODULE_INFO	Opcode inputs	Opcode outputs
PCO	CN_WRAP_KBK (Modes: KBK_WRAP_WITH_KEK, KBK_WRAP_WITH_CERT_AUTH_DERIVED_KEY, KBK)	Opcode inputs	Opcode outputs
PCO	CN_UNWRAP_KBK (Modes: KBK_WRAP_WITH_KEK, KBK_WRAP_WITH_CERT_AUTH_DERIVED_KEY)	Opcode inputs	Opcode outputs
UN-AUTH	CN_GET_CHALLENGE_CO	Opcode inputs	Opcode outputs
PCO	CN_SET_M_VALUE	Opcode inputs	Opcode outputs
MCO/PCO/PCU/AU/UN-AUTH	CN_CERT_AUTH_GET_CERT_REQ	Opcode inputs	Opcode outputs
PCO	CN_CERT_AUTH_REMOVE_CERT	Opcode inputs	Opcode outputs
MCO	CN_UPDATE_LICENSE	Opcode inputs	Opcode outputs
MCO/UN-AUTH	CN_GET_LICENSE_INFO	Opcode inputs	Opcode outputs
MCO/UN-AUTH	CN_GET_DIAGLOG	Opcode inputs	Opcode outputs
MCO/UN-AUTH	CN_NOR_HUK_OP	Opcode inputs	Opcode outputs
MCO/PCO/PCU/AU/UN-AUTH	CN_APP_CLEANUP	Opcode inputs	Opcode outputs
MCO/PCO/PCU/AU/UN-AUTH	CN_APP_CLEANUP_V2	Opcode inputs	Opcode outputs
UN-AUTH	CN_SAFE_REBOOT	Opcode inputs	Opcode outputs

Role	Service	Input*	Output*
UN-AUTH	CN_GET_ALL_PARTITION_INFO	Opcode inputs	Opcode outputs
PCU	CN_GENERATE_PBE_KEY	Opcode inputs	Opcode outputs
PCU	LSPAY_GENERATE_ASYMM_KEY	Opcode inputs	Opcode outputs
PCU	LSPAY_GENERATE_SYMM_KEY	Opcode inputs	Opcode outputs
PCU	LSPAY_EXPORT_PUBLIC_KEY	Opcode inputs	Opcode outputs
PCU	LSPAY_IMPORT_KPK	Opcode inputs	Opcode outputs
PCU	LSPAY_IMPORT_KEY	Opcode inputs	Opcode outputs
PCU	LSPAY_IMPORT_TR34_KEY	Opcode inputs	Opcode outputs
PCU	LSPAY_EXPORT_KEY	Opcode inputs	Opcode outputs
PCU	LSPAY_EXPORT_TR34_KEY	Opcode inputs	Opcode outputs
PCU	LSPAY_TRANSLATE_KEY	Opcode inputs	Opcode outputs
PCU	LSPAY_IMPORT_CERTIFICATE	Opcode inputs	Opcode outputs
PCU	LSPAY_IMPORT_DECIMAL_TABLE	Opcode inputs	Opcode outputs
PCU	LSPAY_GENERATE_CSR	Opcode inputs	Opcode outputs
PCU	LSPAY_DERIVE_KEY	Opcode inputs	Opcode outputs
PCU	LSPAY_ENCRYPT	Opcode inputs	Opcode outputs
PCU	LSPAY_DECRYPT	Opcode inputs	Opcode outputs
PCU	LSPAY_DECRYPT_THEN_ENCRYPT	Opcode inputs	Opcode outputs
PCU	LSPAY_MAC_GEN	Opcode inputs	Opcode outputs
PCU	LSPAY_MAC_VERIFY	Opcode inputs	Opcode outputs*
PCU	LSPAY_MAC_TRANSLATE	Opcode inputs	Opcode outputs*
PCU	LSPAY_FPE_ENCRYPT	Opcode inputs	Opcode outputs
PCU	LSPAY_FPE_DECRYPT	Opcode inputs	Opcode outputs
PCU	LSPAY_SIGN	Opcode inputs	Opcode outputs
PCU	LSPAY_SIGN_VERIFY	Opcode inputs	Opcode outputs
PCU	LSPAY_PINBLK_TRANSLATE	Opcode inputs	Opcode outputs
PCU	LSPAY_DERIVE_PIN_FROM_OFFSET	Opcode inputs	Opcode outputs
PCU	LSPAY_DERIVE_OFFSET_FROM_PIN	Opcode inputs	Opcode outputs
PCU	LSPAY_VERIFY_PIN	Opcode inputs	Opcode outputs
PCU	LSPAY_PVV_GEN	Opcode inputs	Opcode outputs
PCU	LSPAY_PVV_VERIFY	Opcode inputs	Opcode outputs
PCU	LSPAY_EMV_GENVERIFY	Opcode inputs	Opcode outputs
PCU	LSPAY_EMV_SECURE_MSG_GEN	Opcode inputs	Opcode outputs
PCU	LSPAY_CVV_GEN	Opcode inputs	Opcode outputs
PCU	LSPAY_CVV_VERIFY	Opcode inputs	Opcode outputs
PCU	LSPAY_KEY_SHARE_CREATE	Opcode inputs	Opcode outputs
PCU	LSPAY_KEY_SHARE_EXPORT_COMPONENT	Opcode inputs	Opcode outputs
PCU	LSPAY_KEY_SHARE_IMPORT_COMPONENT	Opcode inputs	Opcode outputs

Role	Service	Input*	Output*
PCU	LSPAY_KEY_SHARE_COMBINE_INIT	Opcode inputs	Opcode outputs
PCU	LSPAY_KEY_SHARE_COMBINE_KEY	Opcode inputs	Opcode outputs
PCU	LSPAY_KEY_SHARE_ZEROIZE	Opcode inputs	Opcode outputs
PCO	LSPAY_MFK_GENERATE	Opcode inputs	Opcode outputs
PCO	LSPAY_MFK_GET_INFO	Opcode inputs	Opcode outputs
PCO	LSPAY_MFK_SET_PRIMARY	Opcode inputs	Opcode outputs
PCO	LSPAY_MFK_DELETE	Opcode inputs	Opcode outputs
PCO	LSPAY_FUNCTIONALITY_GET	Opcode inputs	Opcode outputs
PCO	LSPAY_FUNCTIONALITY_SET	Opcode inputs	Opcode outputs
PCU	LSPAY_EXPORT_KPK	Opcode inputs	Opcode outputs
PCU	LSPAY_IMPORT_PUBLIC_KEY	Opcode inputs	Opcode outputs
PCU	LSPAY_VALIDATE_PUBLIC_KEY	Opcode inputs	Opcode outputs
PCU	CN_GENERATE_KEY_PAIR (non-compliant)	Opcode inputs	Opcode outputs
PCU	CN_GENERATE_KEY (non-compliant)	Opcode inputs	Opcode outputs
PCU	CN_CREATE_OBJECT (non-compliant)	Opcode inputs	Opcode outputs
PCU	CN_UNWRAP_KEY (non-compliant)	Opcode inputs	Opcode outputs
PCU	CN_WRAP_KEY (non-compliant)	Opcode inputs	Opcode outputs
PCU/PCO/AU	CN_EXTRACT_MASKED_OBJECT (non-compliant)	Opcode inputs	Opcode outputs
MCO	CN_STORE_FW_SIGNING_KEY (non-compliant)	Opcode inputs	Opcode outputs
PCU	CN_ME_PKCS_LARGE (non-compliant) CN_ME_PKCS (non-compliant)	Opcode inputs	Opcode outputs
Manufacturer	CN_STORE_VENDOR_PRE_SHARED_KEY (CN_STORE_KBK_SHARE) (non-compliant)	Opcode inputs	Opcode outputs
PCU/PCO/AU	CN_INSERT_MASKED_OBJECT (non-compliant)	Opcode inputs	Opcode outputs
UN-AUTH	CN_ENCRYPT_SESSION (non-compliant)	Opcode inputs	Opcode outputs
PCU	CN_DERIVE_KEY (non-compliant)	Opcode inputs	Opcode outputs
PCU	CN_PQC_GENERATE_KEY_PAIR	Opcode inputs	Opcode outputs
PCU	CN_PQC_CRYPTO_SIG_GEN	Opcode inputs	Opcode outputs
PCU	CN_PQC_CRYPTO_SIG_VERIFY	Opcode inputs	Opcode outputs
PCU	CN_PQC_CRYPTO_MULTICALL_SIG_GEN	Opcode inputs	Opcode outputs
PCU	CN_PQC_CRYPTO_MULTICALL_SIG_VERIFY	Opcode inputs	Opcode outputs
PCU	CN_PQC_CRYPTO_HYBRID_SIG_GEN	Opcode inputs	Opcode outputs
PCU	CN_PQC_CRYPTO_HYBRID_SIG_VERIFY	Opcode inputs	Opcode outputs
MCO	HPS_CREATE_PARTITION	Opcode inputs	Opcode outputs
MCO	HPS_PART_FW_UPDATE_BEGIN HPS_PART_FW_UPDATE HPS_PART_FW_UPDATE_END	Opcode inputs	Opcode outputs
MCO	HPS_PARTITION_MGMT	Opcode inputs	Opcode outputs
MCO	HPS_PARTITION_INFO	Opcode inputs	Opcode outputs
MCO	HPS_DELETE_PARTITION	Opcode inputs	Opcode outputs

Role	Service	Input*	Output*
MCO	CN_UPDATE_LICENSE (non-compliant)	Opcode inputs	Opcode outputs
UN-AUTH	CN_GET_LICENSE_INFO (non-compliant)	Opcode inputs	Opcode outputs

* Please refer to the API documentation for specific Opcode inputs and Opcode outputs.

* The below generic services represent a set of related services:

- CN_BACKUP includes the opcodes CN_BACKUP_BEGIN, CN_BACKUP_CONFIG, CN_BACKUP_END, CN_BACKUP_KEY, and CN_BACKUP_USERS.
- CN_RESTORE includes the opcodes CN_RESTORE_BEGIN, CN_ RESTORE_CONFIG, CN_ RESTORE_END, CN_ RESTORE_KEY, and CN_ RESTORE USERS.
- CN_INSERT_MASKED_OBJECT includes the opcode the CN_INSERT_MASKED_OBJECT_USER.
- CN_CERT_AUTH_GET_CERT includes the opcode the CN_CERT_AUTH_GET_CERT_CHAIN.
- CN_CERT_AUTH_STORE_CERT includes the opcode the CN_CERT_AUTH_STORE_CERT_CHAIN .
- CN_UPDATE_LICENSE, CN_GET_LICENSE_INFO, CN_GET_DIAGLOG, and CN_NOR_HUK_OP services are only applicable for MARVELL-LS2-FW-10.23-1107.

4.1.1 Authentication

The module enforces identity-based authentication. A role is explicitly selected at authentication; the MCO role is associated with the master partition and the PCO and PCU roles are associated with user partitions (see section 4 Roles, Services, and Authentication for details). The module allows one identity per role, per partition.

Table 9 Roles and Authentication

Role	Authentication Method	Authentication Strength
MCO/Pre-CO/PCO/PCU/ AU	Username and password	The password is a minimum of 8 characters (case-sensitive, alpha-numeric. As such, there are (26*2+10)^8 = 62^8 possible minimum-length passwords, and the false acceptance rate is 1 in 62^8. A maximum of 20 password attempts are possible before a user is permanently locked out. Therefore, the probability of false authentication over any time frame is 20 in 62^8. (The number of allowed login attempts prior to lockout is configured during module initialization but cannot exceed 20.) Lockout of MCO automatically zeroizes the module. In all other cases, lockout can be unset by destroying the partition.
MCO/PCO /PCU	Digital signature	Authentication is performed using SHA-256 based RSA 2048-bit PKCS#1-v1.5 signatures (provides 112 bits of strength). Corresponding public key is associated with the identity. The probability that a random attempt will succeed or a false acceptance will occur is approximately 1 in 2^112. For each failed signature verification, the module will block for 2 seconds. Based on this maximum rate, the probability that a random attempt will succeed in a one minute period is approximately 30 in 2^112.

In the table below, “UN-AUTH” refers to all users defined in section 4.2 Assumption of Roles – Non-Master Partition Roles other than the Pre-CO.

Table 10 Approved Services

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
CN_ZEROIZE	Zeroize all master partition and user partitions. Can be configured to be allowed by CO only, Factory-reset the master partition. zeroizes all the data except vendor- programmed certificates and SSPs. (Perform zeroization)	None	User keys MMEK PMEK PAK KLK Partition Masking Key PAC 2FAMofNPubK CAPubK AOAPubK Login Passwords PEK KBK POTAC POKBK POAC	MCO PCO PCU AU UN-AUTH	Z	Success with fips_state = 2 or 3
CN_VENDOR_ZEROIZE	Zeroize all master and user partitions. Zeroizes vendor- programmed certificates and SSPs. Zeroize types: Vendor/factory reset (Perform zeroization)	None	User keys FMAK FMAEK MARC MARECFMAEC MMEK MFDEK PAK PAC AOTAC POTAC POKBK POAC AOTAC OKBK AOAC SecureBootAuth Public Key	MCO	Z	Success with fips_state = 2 or 3

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
			On vendor zeroize: MFDEK FMAK MFKBK			
CN_APP_INITIALIZE	Registers an application with HSM.	Hash DRBG (SP 800-90Ar1) (#A1947)	DRBG Entropy/HASH_DRBG Internal State	UN-AUTH	E R	Success with fips_state = 2 or 3
CN_APP_FINALIZE	Unregisters an application from HSM.	None	User keys	MCO PCO PCU AU UN-AUTH	Z	Success with fips_state = 2 or 3
CN_OPEN_SESSION	Management services for open, status of sessions.	Hash DRBG (SP 800-90Ar1) (#A1947)	DRBG Entropy/HASH_DRBG Internal State	MCO PCO PCU AU UN-AUTH	None	Success with fips_state = 2 or 3
CN_CLOSE_SESSION	Closes the session.	None	User keys	MCO PCO PCU AU UN-AUTH	Z	Success with fips_state = 2 or 3
CN_GET_SESSION_INFO	Gets the session information.	None	None	MCO PCO PCU AU UN-AUTH	None	Success with fips_state = 2 or 3
CN_CLOSE_ALL_SESSIONS	Management services for closing all sessions.	None	User keys	MCO PCO PCU AU UN-AUTH	Z	Success with fips_state = 2 or 3
CN_CLOSE_PARTITION_SESSIONS	Closes sessions of all applications tied to a partition.	None	User keys	PCO MCO	Z	Success with fips_state = 2 or 3
CN_ENCRYPT_SESSION	Enables encrypted communication channel.	Hash DRBG (SP 800-90Ar1) (#A1947) RSA Signature Primitive (CVL) (FIPS 186-4) (#A1947) KDF TLS (CVL) (SP 800-135r1) (#A1947) ECDSA SigVer (FIPS186-4)	POAC PAK TLS pre-master secret TLS master secret	UN-AUTH	R E G, E G, E	Success with fips_state = 2 or 3

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
		(#A1948)	TLS session symmetric key set DRBG Entropy/HASH_DRBG Internal State		G E	
CN_AUTHORIZE_SESSION	Authorizes the sessions to be used under E2E and do login.	None	None	UN-AUTH	None	Success with fips_state = 2 or 3
CN_LOGIN	Allows login to a session. Public key is used to verify user signatures, optionally in 2-factor authentication.	AES-CBC (SP 800-38A) (#A1947) RSA SigVer (FIPS186-4) (#A1948) PBKDF (SP 800-132) (#A1948) Hash DRBG (SP 800-90Ar1) (#A1947)	PEK Login passwords 2FAMofNPubK CAPubK DRBG Entropy/HASH_DRBG Internal State	UN-AUTH	E W E E E	Success with fips_state = 2 or 3
CN_LOGOUT	Allows logout of a session.	None	None	MCO PCO PCU AU	None	Success with fips_state = 2 or 3
CN_ALWAYS_AUTHORIZE_USER	Context specific explicit user authorization service for CKA_ALWAYS_AUTHENTICATE keys	AES-CBC (SP 800-38A) (#A1947) RSA SigVer (FIPS186-4) (#A1948) PBKDF (SP 800-132) (#A1948)	PEK 2FAMofNPubK CAPubK Login passwords	PCU	E E E E	Success with fips_state = 2 or 3
CN_UPDATE_USER_DETAILS	Requires user to be logged in. Updates passwords and public key for 2-factor authentication.	RSA SigVer (FIPS186-4) (#A1948) AES-CBC (SP 800-38A) (#A1947) Hash DRBG (SP 800-90Ar1) (#A1947) PBKDF (SP 800-132) (#A1948)	PMEK 2FAMofNPubK PEK Login passwords DRBG Entropy/HASH_DRBG Internal State	PCO Pre-CO	E W E W E	Success with fips_state = 2 or 3
CN_SET_USER_ATTR	Set user attributes that control the functionality of a crypto user.	None	None	PCO	None	Success with fips_state =

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
						2 or 3
CN_TOKEN_INFO	Gets token/module information. (Show module's versioning information, Show status)	None	None	MCO PCO PCU AU UN-AUTH	None	Success with fips_state = 2 or 3
CN_PARTITION_INFO	Returns partition Information. (Show module's versioning information, Show status)	None	Partition Owner KBK (POKBK)	MCO PCO PCU AU UN-AUTH	E	Success with fips_state = 2 or 3
CN_GET_HSM_LABEL	Returns HSM label.	None	None	MCO PCO PCU AU UN-AUTH	None	Success with fips_state = 2 or 3
CN_ALL_PARTITION_INFO	Gets information for all Partitions. (Show module's versioning information, Show status)	None	Partition Owner KBK (POKBK)	MCO	E	Success with fips_state = 2 or 3
CN_GET_POLICY_SET	Gets the current policy settings. This operation does not need authentication.	None	None	MCO PCO PCU AU UN-AUTH	None	Success with fips_state = 2 or 3
CN_GET_VERSION	Obtains firmware version. (Show module's versioning information)	None	None	MCO PCO PCU AU UN-AUTH	None	Success with fips_state = 2 or 3
CN_STORE_VENDOR_PRE_SHARED_KEY (CN_STORE_KBK_SHARE)	Stores fixed keys (KBK) for backup.	RSA SigVer (FIPS186-4) (#A1948) KTS-IFC (KTS) (SP 800-56Br2) (#A1948) AES-KWP (SP 800-38F) (#A1948)	MFKBK PDEK MARC MAREC AOTAC, POTAC, OKBK, POKBK, FMAK, PAK	Manufacturer PCO	W E E E E E W W E E	Success with fips_state = 2 or 3
CN_LOAD_RECOVERY_KEY_INIT CN_LOAD_RECOVERY_KEY_FINISH	Manages loading recovery key into HSM	KAS-ECC-SSC SP800-56Ar3 (#A1948) KAS-ECC SP800-56Ar3 (#A1948) ECDSA KeyVer (FIPS186-4) (#A1948) KDA HKDF	MCO_RK HSMEK HSMEKC	MCO	W E E	Success with fips_state = 2 or 3

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
		SP800-56Cr1 (#A1948) KDA OneStep SP800-56Cr1 (#A1948) KDA TwoStep SP800-56Cr1 (#A1948) AES-KW (KTS) (SP 800-38F) (#A1948) KAS KDA HKDF (SP 800-56Ar3) KAS KDA ONESTEP (SP 800-56Ar3) KAS KDA TWOSTEP (SP 800-56Ar3)				
CN_SHUTDOWN	Closes and cleans the Cfm library and driver. Unregisters an application from HSM. Deletes all the sessions created within this application. Closes the device file.	None	None	MCO	None	Success with fips_state = 2 or 3
CN_SET_INIT_TIME	Sets the user's initial time upon receiving the HSM	None	None	MCO	None	Success with fips_state = 2 or 3
CN_SET_VENDOR_TIME	Sets the vendor time on the HSM	None	None	MCO	None	Success with fips_state = 2 or 3
CN_GET_TIME	Gets the RTC and System time from HSM (Show status)	None	None	MCO	None	Success with fips_state = 2 or 3
CN_SYNC_TIME	Sets the user's time on the HSM. Also used for drift calculation and configuration. Returns useful information such as the drift between previous configured time and new time configured and the lifetime average drift observed.	None	None	MCO	None	Success with fips_state = 2 or 3
CN_UCD_CMD	Gets the logs from UCD related to voltage rail monitoring and faults in the system.	None	None	MCO	None	Success with fips_state = 2 or 3
CN_GET_HSM_LOGGER_INFO	Gets the pending logs of FRAM from the HSM. (Show status)	None	None	MCO	None	Success with fips_state = 2 or 3
CN_INIT_TOKEN	Initializes the HSM and sets its policies and boundaries to the values specified in config_file	Hash DRBG (SP 800-90Ar1) (#A1947)	MMEK PMEK DRBG Entropy/HASH_DRBG Internal	MCO PCO Pre-CO	G G E	Success with fips_state = 2 or 3

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
			State			
CN_GEN_PSWD_ENC_KEY	Generates a Password Encryption Key (PEK), which is used to wrap the user password while sending it over the FIPS boundary.	KAS-IFC-SSC (SP 800-56Br2) (#A1948) KDA HKDF SP800-56Cr1 (#A1948) KDA OneStep SP800-56Cr1 (#A1948) KDA TwoStep SP800-56Cr1 (#A1948) Hash DRBG (SP 800-90Ar1) (#A1947) KAS-IFC HKDF (SP800-56Br2) KAS-IFC OneStep (SP800-56Br2) KAS-IFC TwoStep (SP800-56Br2)	PEK DRBG Entropy/HASH_DRBG Internal State Host PswdEncKeyPublic Key	MCO PCO Pre-CO	G E E	Success with fips_state = 2 or 3
CN_CREATE_CO	Creates Crypto Officer	RSA SigVer (FIPS186-4) (#A1948) AES-CBC (SP 800-38A) (#A1947) PBKDF (SP 800-132) (#A1948)	PMEK 2FAMofNPubK PEK Login passwords DRBG Entropy/HASH_DRBG Internal State	MCO PCO Pre-CO	E W E W E	Success with fips_state = 2 or 3
CN_INIT_DONE	Completes initialization of HSM/ partition. Successful initialization of HSM will reboot the HSM.	None	None	MCO PCO Pre-CO	None	Success with fips_state = 2 or 3
CN_FW_UPDATE_BEGIN, CN_FW_UPDATE, CN_FW_UPDATE_END	Performs firmware update: uploads the signed firmware to the module.	RSA SigVer (FIPS186-4) (#A1948)	Manufacturer firmware update validation key MFUVK	MCO	E E	Success with fips_state = 2 or 3

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
CN_STORE_FW_SIGNING_KEY	Configure an RSA or EC public key into HSM as AO attestation key. These keys can be of modulus 1024, 2048, 3072, and 4096 or a supported EC curve.	RSA SigVer (FIPS186-4) (#A1948) ECDSA SigVer (FIPS186-4) (#A1948)	AOAPubK AOAC	MCO	W E	Success with fips_state = 2 or 3
CN_ALLOW_FW_UPDATE	Configure a lower version of FW to be allowed to be updated for certain time period and on certain HSMs	RSA SigVer (FIPS186-4) (#A1948) ECDSA SigVer (FIPS186-4) (#A1948)	AOAPubK	MCO	E	Success with fips_state = 2 or 3
CN_INVOKE_FIPS	Performs Self tests.	None	None	MCO	None	Success with fips_state = 2 or 3
CN_SET_HSM_CONFIG	Sets the HSM configuration parameters	None	None	PCO	None	Success with fips_state = 2 or 3
CN_CREATE_PARTITION	Creates a partition with the given name and size.	RSA KeyGen (FIPS186-4) (#A2393) RSA SigGen (FIPS186-4) (#A1948)	PAK FMAC FMAK MARC PAC	MCO	G E E E G	Success with fips_state = 2 or 3
CN_DELETE_PARTITION	Deletes a partition and all associated keys.	None	None	MCO	Z	Success with fips_state = 2 or 3
CN_BACKUP	Backs up config, users, keys, and data objects.	KDF SP800-108 (#A1948) Hash DRBG (SP 800-90Ar1) (#A1947) HMAC-SHA2-256 (FIPS 198-1) (#A1947) SHA2-512 (FIPS 180-4) (#A1948)	Backup session key 2FAMofNPubK PEK CAPubK User Keys DRBG Entropy/HASH_DRBG Internal State	MCO PCO	G, E R R R R E	Success with fips_state = 2 or 3
CN_RESTORE	Restores partition configuration, users, and keys.	AES-KW (KTS) (SP 800-38F) (#A1948) AES-CBC (SP 800-38A) (#A1948)	Backup session key 2FAMofNPubK	MCO PCO	W, E W	Success with fips_state = 2 or 3

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
		SHA2-512 (FIPS 180-4) (#A1948) Hash DRBG (SP 800-90Ar1) (#A1947) CKG SP 800-133Rev2 (Vendor affirmed)	PEK Login Passwords CAPubK User Keys DRBG Entropy/HASH_DRBG Internal State		W W W W E	
CN_BACKUP_OBJECT	Backs up partition key, partition CSR, PO cert, partition cert signed by PO.	AES-KW (KTS) (SP 800-38F) (#A1948)	Backup PAK PAC Partition masking key	MCO	E R R R	Success with fips_state = 2 or 3
CN_RESTORE_OBJECT	Restores the backed-up object and object details.	AES-KW (KTS) (SP 800-38F) (#A1948)	Backup session key PAK PAC Partition masking key	MCO	E W W W	Success with fips_state = 2 or 3
CN_SET_NODEID	Sets the cluster node ID for a partition.	None	None	PCO	None	Success with fips_state = 2 or 3
CN_CREATE_USER	Creates a new CU, CO, or AU user with the provided name and password.	AES-CBC (SP 800-38A) (#A1947) PBKDF (SP 800-132) (#A1948) RSA SigVer (FIPS186-4) (#A1948)	PMEK 2FAMofNPubK PEK Login passwords DRBG internal states	PCO	E W E W E	Success with fips_state = 2 or 3
CN_DELETE_USER	Deletes the user with the given name.	None	None	PCO	Z	Success with

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
						fips_state = 2 or 3
CN_LIST_USERS	Lists all users of the current partition.	None	None	PCO/ PCU	None	Success with fips_state = 2 or 3
CN_GET_LOGIN_FAILURE_CNT	Gets login failure count. (Show status)	None	None	PCO	None	Success with fips_state = 2 or 3
CN_CREATE_PRE_OFFICER	Creates a pre-officer.	AES-CBC (SP 800-38A) (#A1947) PBKDF (SP 800-132) (#A1948)	PMEK	Pre-CO	E	Success with fips_state = 2 or 3
CN_CREATE_APPLIANCE_USER	Creates an Appliance User.	AES-CBC (SP 800-38A) (#A1947) PBKDF (SP 800-132) (#A1948)	PMEK	PCO	E	Success with fips_state = 2 or 3
CN_OPEN_SESSION_V2	Opens a session in HSM and returns the session handle.	None	None	UN-AUTH	None	Success with fips_state = 2 or 3
CN_ENCRYPT_SESSION_V2	Establishes E2E connection with/without client- authentication, between HSM and the CavClient/CavMgmt Util.	Hash DRBG (SP 800-90Ar1) (#A1947) RSA Decryption Primitive (SP 800-56Br2) (CVL) (#A1947) KAS-ECC-SSC SP800-56Ar3 (#A1947) ECDSA KeyVer (FIPS186-4) (#A1948) KDF TLS (CVL) (SP 800-135r1) (#A1947) ECDSA SigVer (FIPS186-4) (#A1948) KAS TLS (SP 800-56Ar3)	CAPubK POAC TLS pre-master secret TLS master secret TLS session symmetric key set DRBG Entropy/HASH_DRBG Internal State	UN-AUTH	E E G, E G, E G E	Success with fips_state = 2 or 3
CN_GET_SERVER_PARAMS	Gets the server parameters used in Cav-server for the server handshake messages	Hash DRBG (SP 800-90Ar1) (#A1947) ECDSA KeyGen	TLS session (E2E) ECDH key	UN-AUTH	G E	Success with fips_state = 2 or 3

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
		(FIPS186-4) (#A2393)	POAC PAK DRBG Entropy/HASH_DRBG Internal State		E E	
CN_GET_USER_INFO	Gets user info and user attributes in TLV format for a crypto user. Memory should be allocated for user attribute template by the application calling this opcode.	None	None	PCO PCU AU	None	Success with fips_state = 2 or 3
CN_CREATE_OBJECT	Imports a public key into HSM.	None	User Public Keys	PCU	W	Success with fips_state = 2 or 3
CN_GEN_KEY_ENC_KEY	Generates KLK or key encryption key. The type of key is determined by the kek_method parameter in the hsm_config file. The KLK is always a global key;	ECDSA KeyGen (FIPS186-4) (#A2393) KAS-IFC-SSC (SP 800-56Br2) (#A1948) KAS-ECC (KAS) (SP 800-56Ar3) ECDSA KeyVer (FIPS186-4) (#A1948) KAS-IFC HKDF (SP800-56Br2) KAS-IFC OneStep (SP800-56Br2) KAS-IFC TwoStep (SP800-56Br2) Hash DRBG (SP 800-90Ar1) (#A1947) CKG SP 800-133Rev2 (Vendor affirmed)	KLK Partition key loading private key KLSZ DRBG Entropy/HASH_DRBG Internal State	PCO	G,E G, E G, E	Success with fips_state = 2 or 3
CN_EXTRACT_MASKED_OBJECT	Extracts a masked object; i.e., retrieves an object by wrapping it with a masking key shared by the process of cloning.	AES-KW (KTS) (SP 800-38F) (#A1948)	User keys PEK KLK	PCO PCU AU	R R R	Success with fips_state = 2 or 3

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
			Partition masking key		E	
CN_INSERT_MASKED_OBJECT	Inserts a masked object into an HSM that Is extracted from another HSM.	AES-KW (KTS) (SP 800-38F) (#A1948) Triple-DES SP 800-38B, no security claimed per IG 2.4.A AES-CMAC (SP 800-38B) (#A1948) SHA-1, no security claimed per IG 2.4.A	User keys PEK KLK Partition masking key	PCU PCO AU	R R R E	Success with fips_state = 2 or 3
CN_DESTROY_OBJECT	Destroys key object.	None	User keys	PCU	Z	Success with fips_state = 2 or 3
CN_TOMBSTONE_OBJECT	Marks the key to be deleted and makes it unusable.	None	User keys	PCU	Z	Success with fips_state = 2 or 3
CN_DELETE_TOMBSTONED_OBJECT	Destroys tombstoned keys.	None	User keys	PCO PCU AU	Z	Success with fips_state = 2 or 3
CN_GET_ATTRIBUTE_VALUE	Retrieves single key attribute/ metadata.	None	User keys	PCU	E	Success with fips_state = 2 or 3
CN_GET_ATTRIBUTE_SIZE	Retrieves an attribute or its size from an object.	None	User keys	PCU	E	Success with fips_state = 2 or 3
CN_GET_ALL_ATTRIBUTE_SIZE	Retrieves an attribute or its size from an object.	None	User keys	PCU	E	Success with fips_state = 2 or 3
CN_GET_ALL_ATTRIBUTE_VALUE	Retrieves all attributes or their size from an object.	None	User keys	PCU	E	Success with fips_state = 2 or 3
CN_MODIFY_OBJECT	Uses the setAttribute command to modify object attributes.	None	User keys	PCO PCU	W	Success with fips_state = 2 or 3
CN_MODIFY_KEY_OWNER	Modifies object attributes.	None	User keys	PCO	W	Success with fips_state = 2 or 3
CN_GENERATE_KEY	Generates a symmetric key of given key type and length.	Hash DRBG (SP 800-90Ar1) (#A1947)	Symmetric User Keys	PCU	G E	Success with fips_state = 2 or 3

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
		CKG SP 800-133Rev2 (Vendor affirmed) Triple-DES SP 800-38B, no security claimed per IG 2.4.A AES-CMAC (SP 800-38B) (#A1948) SHA-1, no security claimed per IG 2.4.A	DRBG Entropy/HASH_DRBG Internal State			
CN_GENERATE_KEY_PAIR	Generates asymmetric keys (RSA/ ECC). Updates the public and private key handles in the output on return.	RSA KeyGen (FIPS186-4) (#A2393) ECDSA KeyGen (FIPS186-4) (#A2393) ECDSA SigGen (FIPS186-4) (#A1948) ECDSA SigVer (FIPS186-4) (#A1948) Hash DRBG (SP 800-90Ar1) (#A1947)	Asymmetric User Keys DRBG Entropy/HASH_DRBG Internal State	PCU	G E	Success with fips_state = 2 or 3
CN_EXPORT_PUB_KEY	Exports a public key in PEM- encoded format.	None	User keys	PCU	R	Success with fips_state = 2 or 3
CN_GET_OBJECT_INFO	Obtains Key details like shared sessions, shared users ,and m_values of USE_KEY, MANAGE_KEY services.	None	None	PCU	None	Success with fips_state = 2 or 3
CN_UNWRAP_KEY/ CN_UNWRAP_KEY2	Unwraps a key with an AES/ Triple-DES/RSA private key existing on HSM or KLK. Takes the output wrapped data of wrapKey2 command.	AES-KW (KTS) (SP 800-38F) (#A1948) KAS-ECC-SSC SP800-56Ar3 (#A1947) ECDSA KeyVer (FIPS186-4) (#A1948) KDA HKDF SP800-56Cr1 (#A1948) KDA OneStep SP800-56Cr1	User keys KLK	PCU	W E	Success with fips_state = 2 or 3

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
		(#A1948) KDA TwoStep SP800-56Cr1 (#A1948) KTS-IFC (SP 800-56Br2) (#A1948) AES-CBC (SP 800-38A) (#A1947) KTS-IFC (KTS) (SP 800-56Br2) (#A2393) TDES-KW (KTS) (SP 800-38F) (#A1948) * Legacy use only TDES-KW (SP 800-38F) (#A1948) * Legacy use only AES-KW (KTS) (SP 800-38F) (#A1948) Triple-DES SP 800-38B, no security claimed per IG 2.4.A SHA-1, no security claimed per IG 2.4.A AES-CMAC (SP 800-38B) (#A1948) KAS ANS 9.63 (SP 800-56Ar3) KAS KDA HKDF (SP 800-56Ar3) KTS-IFC (KTS) (SP 800-56Br2) (#A1948) AES-KWP (KTS) (SP 800-38F) (#A1948) AES-GCM (KTS) (SP800-38D) (#A1948) KAS KDA				

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
		HKDF (SP 800-56Ar3) KAS KDA ONESTEP (SP 800-56Ar3) KAS KDA TWOSTEP (SP 800-56Ar3) AES [#A1947], allowed per IG D.G AES [#A1948], allowed per IG D.G				
CN_WRAP_KEY/CN_WRAP_KEY2	Wraps sensitive (private and symmetric) keys from the HSM to the host.	AES-KW (KTS) (SP 800-38F) (#A1948) KAS-ECC-SSC SP800-56Ar3 (#A1947) ECDSA KeyVer (FIPS186-4) (#A1948) KDA HKDF SP800-56Cr1 (#A1948) KDA OneStep SP800-56Cr1 (#A1948) KDA TwoStep SP800-56Cr1 (#A1948) KTS-IFC (SP 800-56Br2) (#A1948) KTS-IFC (KTS) (SP 800-56Br2) (#A2393) AES-KW (KTS) (SP 800-38F) (#A1948) KTS-IFC (KTS) (SP 800-56Br2) (#A1948) AES-KWP (KTS) (SP 800-38F) (#A1948) AES-GCM (KTS) (SP800-38D)	User keys KLK	PCU	R E	Success with fips_state = 2 or 3

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
		(#A1948) KAS KDA HKDF (SP 800-56Ar3) KAS KDA ONESTEP (SP 800-56Ar3) KAS KDA TWOSTEP (SP 800-56Ar3) KAS ANS 9.63 (SP 800-56Ar3)				
CN_NIST_AES_WRAP_UNWRAP/ CN_NIST_AES_WRAP_UNWRAP2	Wraps/unwraps data with a specified AES key.	AES-KW (KTS) (SP 800-38F) (#A1948)	Symmetric User Keys	PCU	E	Success with fips_state = 2 or 3
CN_DERIVE_KEY	Derives a key using a supported KDF mechanism with the params given by the user.	KDF SP800-108 (#A1948) ECDSA KeyVer (FIPS186-4) (#A1948) KDA HKDF SP800-56Cr1 (#A1948) HMAC-SHA2-256 (FIPS 198-1) (#A1948) HMAC-SHA2-384 (FIPS 198-1) (#A1948) HMAC-SHA2-512 (FIPS 198-1) (#A1948) HMAC-SHA-1 (FIPS 198-1) (#A1948) KAS KDA HKDF (SP 800-56Ar3) KAS KDA ONESTEP (SP 800-56Ar3) KAS KDA TWOSTEP (SP 800-56Ar3) KAS ANS 9.63 (SP 800-56Ar3)	User keys	PCU	G, E	Success with fips_state = 2 or 3
CN_FIND_OBJECTS_USING_COUNT/ CN_FIND_ALL_OBJECTS_IN_RANGE/ CN_FIND_ALL_OBJECTS/ CN_FIND_ALL_OBJECTS_USING_COUNT/	Finds all key(s) in the partition based on input criteria. An array of key handles will be returned for the keys that match the input criteria specified, key class, key label, etc. Search can be requested from an index.	None	None	MCO PCO PCU AU	None	Success with fips_state = 2 or 3

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
CN_FIND_OBJECTS/ CN_FIND_OBJECTS_FROM_INDEX						
CN_ADMIN_GET_PARTN_KEYHANDLES_HASH	Gets hash of all keys for a partition.	SHA-1 (FIPS 180-4) (#A1948) SHA2-256 (FIPS 180-4) (#A1948) SHA2-384 (FIPS 180-4) (#A1948) SHA2-512 (FIPS 180-4) (#A1948)	User keys	PCO AU	E R	Success with fips_state = 2 or 3
CN_GET_PARTN_SINGLE_KEY HANDLE_HASH	Gets hash of single key for a partition	SHA-1 (FIPS 180-4) (#A1948) SHA2-256 (FIPS 180-4) (#A1948) SHA2-384 (FIPS 180-4) (#A1948) SHA2-512 (FIPS 180-4) (#A1948)	User keys	PCO PCU AU	E	Success with fips_state = 2 or 3
CN_ DESTROY _OBJECT	Deletes the specified object stored on the HSM.	None	User keys	PCU	Z	Success with fips_state = 2 or 3
CN_PARTN_GET_AUDIT_DETAILS	Gets audit logs details	None	None	PCO AU	None	Success with fips_state = 2 or 3
CN_PARTN_GET_AUDIT_LOGS	Gets audit logs	SHA2-256 (FIPS 180-4) (#A1947)	None	PCO AU	None	Success with fips_state = 2 or 3
CN_PARTN_GET_AUDIT_SIGN	Gets audit logs hash or RSA signature	SHA2-256 (FIPS 180-4) (#A1947) ECDSA SigGen (FIPS186-4) (CVL) (#A1947)	PAK	PCO AU	E	Success with fips_state = 2 or 3
CN_PARTN_ACK_AUDIT_SIGN	Acks previously retrieved signature. Either hash or signature needs to match with the values stored by HSM for firmware to accept the signature acknowledgment	None	None	PCO AU	None	Success with fips_state = 2 or 3
CN_FINALIZE_LOGS	Finalizes logs by inserting end marker. No more loggable commands are allowed on the partition after this command is run.	None	None	MCO	None	Success with fips_state = 2 or 3
CN_SET_POLICY	Sets an HSM policy	None	None	MCO PCO	None	Success with fips_state =

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
						2 or 3
CN_NIST_AES_WRAP	Wraps data with a specified AES key.	AES-KW (KTS) (SP 800-38F) (#A1948)	POKBK	PCU	E	Success with fips_state = 2 or 3
CN_ALLOC_SSL_CTX	Allocates a context segment in the HSM memory and returns its handle, which will be passed later to the processor in the final 8 bytes of the request as Cptr.	None	None	PCU	None	Success with fips_state = 2 or 3
CN_FREE_SSL_CTX	Frees a context segment for use by another SSL connection.	None	None	PCU	None	Success with fips_state = 2 or 3
CN_FIPS_RAND	Generates FIPS random number of given lengths.	Hash DRBG (SP 800-90Ar1) (#A1947)	DRBG Entropy/HASH_DRBG Internal State	PCU	E	Success with fips_state = 2 or 3
CN_ME_PKCS_LARGE	ModExp and PKCS#1v1.5 Sign and verify	RSA Signature Primitive (CVL) (FIPS 186-4) (#A1947) RSA Decryption Primitive (SP 800-56Br2) (CVL) (#A1947) KTS-IFC (KTS) (SP800-56Br2) (#A2393)	Asymmetric User Keys	PCU	E	Success with fips_state = 2 or 3
CN_ME_PKCS	PKCS#1v2.2 sign and verify.	RSA Signature Primitive (CVL) (FIPS 186-4) (#A1947) RSA Decryption Primitive (SP 800-56Br2) (CVL) (#A1947) KTS-IFC (KTS) (SP800-56Br2) (#A2393)	Asymmetric User Keys	PCU	E	Success with fips_state = 2 or 3
CN_FECC	ECDSA sign and point add/ double/mul operation.	ECDSA SigGen (FIPS186-4) (CVL) (#A1947) KAS-ECC-SSC SP800-56Ar3 (#A1948) KDF ANS 9.63 (CVL) (SP 800-135r1) (#A1948) EC Diffie-Hellman with	Asymmetric User Keys	PCU	E	Success with fips_state = 2 or 3

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
		non-NIST recommended curves [#A1947] allowed per IGs D.F and C.A ECDSA with non-NIST recommended curves [#A1947] allowed per IG C.A				
CN_HASH	Computes SHA hash.	SHA-1 (FIPS 180-4) (#A1947) SHA2-256 (FIPS 180-4) (#A1947) SHA2-384 (FIPS 180-4) (#A1947) SHA2-512 (FIPS 180-4) (#A1947)	None	PCU	None	Success with fips_state = 2 or 3
CN_SHA3	Computes SHA3 Hash	SHA3-224 (FIPS 202) (#A1947) SHA3-256 (FIPS 202) (#A1947) SHA3-384 (FIPS 202) (#A1947) SHA3-512 (FIPS 202) (#A1947) SHAKE-128 (FIPS 202) (#A1947) SHAKE-256 (FIPS 202) (#A1947)	None	PCU	None	Success with fips_state = 2 or 3
CN_HMAC	Computes/verifies the MAC of a complete message. HMAC max message length supported will vary based on hash type.	HMAC-SHA2-256 (FIPS 198-1) (#A1947) HMAC-SHA2-384 (FIPS 198-1) (#A1947) HMAC-SHA2-512 (FIPS 198-1) (#A1947) AES-CMAC (SP 800-38B) (#A1947)	Symmetric and HMAC User Keys	PCU	E	Success with fips_state = 2 or 3
MAJOR_OP_AES_CMAC	Computes/verifies the MAC of a complete message using AES as PRF.	AES-CMAC (SP 800-38B)	Symmetric and HMAC	PCU	E	Success with fips_state =

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
	Minor Opcodes are MINOR_OP_START, MINOR_OP_UPDATE and MINOR_OP_FINISH	(#A1948)	User Keys			2 or 3
CN_ENCRYPT_DECRYPT	AES/Triple-DES(* legacy use only)/AES-GCM encryption and decryption Minor Ops: GCM_INIT, GCM_UPDATE, GCM_FINAL (GCM_MINOR_OP_DEC)	AES-CBC (SP 800-38A) (#A1947) AES-ECB (SP 800-38A) (#A1947) TDES-ECB (SP 800-38A) (#A1947) TDES-CBC (SP 800-38A) (#A1947) AES-CTR (SP 800-38A) (#A1947) AES-CCM (SP 800-38C) (#A1947) AES-GCM (SP 800-38D) (#A1947) TDES-ECB (SP 800-38A) *legacy use only (#A1947) TDES-CBC (SP 800-38A) *legacy use only (#A1947) Hash DRBG (SP 800-90Ar1) (#A1947) AES-GMAC (SP 800-38D) (#A1947)	Symmetric User Keys DRBG Entropy/HASH_DRBG Internal State	PCU	E E	Success with fips_state = 2 or 3
MAJOR_OP_DECRYPT_AND_ENCRYPT_COMMAND	AES-GCM encryption and decryption Minor Ops: GCM_INIT, GCM_UPDATE, GCM_FINAL (GCM_MINOR_OP_DEC)	AES-GCM (SP 800-38D) (#A1947) Hash DRBG (SP 800-90Ar1) (#A1947) AES-GMAC (SP 800-38D) (#A1947)	Symmetric User Keys DRBG Entropy/HASH_DRBG Internal State	PCU	E	Success with fips_state = 2 or 3
MAJOR_OP_ENCRYPT_DECRYPT_RECORD	Encrypts/decrypts records. Sends encrypted E2E request to the FW.	Hash DRBG (SP 800-90Ar1) (#A1947) AES-GCM (SP 800-38D)	Symmetric User Keys DRBG Entropy/H	PCU	E E	Success with fips_state = 2 or 3

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
		(#A1947)	ASH_DRBG Internal State			
CN_CERT_AUTH_GET_SOURCE_RANDOM	Gets the source random number required for mutual trust protocol.	Hash DRBG (SP 800-90Ar1) (#A1947)	DRBG Entropy/HASH_DRBG Internal State	PCO	E R	Success with fips_state = 2 or 3
CN_CERT_AUTH_VALIDATE_PEER_CERTS/ CN_CERT_AUTH_VALIDATE_TARGET_CERTS	Validates the peer certificates as part of the mutual trust protocol.	Hash DRBG (SP 800-90Ar1) (#A1947) RSA SigVer (FIPS186-4) (#A1948) RSA Decryption Primitive (SP 800-56Br2) (CVL) (#A1948)	DRBG Entropy/HASH_DRBG Internal State MARC MAC PAC AOTAC AOAC POTAC POAC	PCO	E E E E E E E E	Success with fips_state = 2 or 3
CN_CERT_AUTH_SOURCE_KEY_EXCHANGE	Generate source key exchange message from the HSM.	Hash DRBG (SP 800-90Ar1) (#A1947) RSA SigVer (FIPS186-4) (#A1948)	SAZ PAK DRBG Entropy/HASH_DRBG Internal State	PCO	G E E	Success with fips_state = 2 or 3
CN_CERT_AUTH_TARGET_KEY_EXCHANGE	Validate key exchange message from peer. Used in cert-based cloning	KAS-IFC HKDF (SP800-56Br2) KAS-IFC OneStep (SP800-56Br2) KAS-IFC TwoStep (SP800-56Br2)	PAK SAZ	PCO	E G	Success with fips_state = 2 or 3
CN_CLONE_SOURCE_INIT	Fetch the value for the clone target initialization.	Hash DRBG (SP 800-90Ar1) (#A1947) RSA KeyGen	PCPK DRBG Entropy/HASH_DRB	PCO	G E	Success with fips_state = 2 or 3

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
		(FIPS186-4) (#A1948) ECDSA KeyGen (FIPS186-4) (#A1948)	G Internal State Partition Cloning Initiator Public Key		E	
CN_CLONE_SOURCE_STAGE1	Push clone target output into clone source.	KAS-ECC (KAS) (SP 800-56Ar3) ECDSA KeyVer (FIPS186-4) (#A1948) HMAC-SHA2-512 (FIPS 198-1) (#A1948) AES-KW (KTS) (SP 800-38F) (#A1948) KAS-IFC HKDF (SP800-56Br2) KAS-IFC OneStep (SP800-56Br2) KAS-IFC TwoStep (SP800-56Br2)	CSSZ PCSK PCSMK Partition masking key Partition Cloning Responder Public Key	PCO	G G G R E	Success with fips_state = 2 or 3
CN_CLONE_TARGET_INIT	Push clone source output into clone target.	Hash DRBG (SP 800-90Ar1) (#A1947) RSA KeyGen (FIPS186-4) (#A1948) ECDSA KeyGen (FIPS186-4) (#A1948)	PCPK DRBG Entropy/HASH_DRBG Internal State Partition Cloning Responder Public Key Partition Cloning Initiator Public Key	PCO	G E E E	Success with fips_state = 2 or 3
CN_CLONE_TARGET_STAGE1	Fetch the value for clone target end.	KAS-ECC (KAS) (SP 800-56Ar3) ECDSA KeyVer (FIPS186-4) (#A1948) HMAC-SHA2-512 (FIPS 198-1) (#A1947) AES-KW	CSSZ PCSK PCSMK Partition masking key	PCO	G G G W	Success with fips_state = 2 or 3

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
		(KTS) (SP 800-38F) (#A1948) KAS-IFC HKDF (SP800-56Br2) KAS-IFC OneStep (SP800-56Br2) KAS-IFC TwoStep (SP800-56Br2)				
CN_LIST_AUTH_PUB_KEYS	List all registered user auth pub keys.	None	None	PCO PCU AU UN-AUTH	None	Success with fips_state = 2 or 3
CN_GET_M_VALUE	Gets minimum number of quorum approvals needed for a service in a partition	None	None	PCO PCU AU UN-AUTH	None	Success with fips_state = 2 or 3
CN_GET_TOKEN	Gets a token from the partition for a given service.	Counter DRBG (SP 800-90Ar1) (#A1948)	DRBG Entropy/CTR_DRBG Internal State	PCO PCU	E	Success with fips_state = 2 or 3
CN_APPROVE_TOKEN	Submit approvals on token, approval could be on a single or multiple blobs.	RSA SigVer (FIPS186-4) (#A1948)	2FA MofNPubkey	PCO PCU	E	Success with fips_state = 2 or 3
CN_LIST_TOKENS	List all MofN tokens in the current partition.	None	None	PCO PCU	None	Success with fips_state = 2 or 3
CN_TOKEN_TIMEOUT	Get the timeout values of the tokens in the partition.	None	None	PCO PCU	None	Success with fips_state = 2 or 3
CN_DELETE_TOKEN	Deletes the existing MxN tokens based on the token-delete options.	None	None	PCO PCU	None	Success with fips_state = 2 or 3
CN_FRAMLOG_CMD	It is status output of the module for critical messages	None	None	UN-AUTH	None	Success with fips_state = 2 or 3
CN_GET_CFG_PREGEN_CACHE_SZ	Set configured pre generated keys cache size	None	None	MCO PCO	None	Success with fips_state = 2 or 3
CN_GET_CFG_PREGEN_CACHE_VAL	Returns the key count in pre generated key cache	None	None	MCO PCO	None	Success with fips_state = 2 or 3
CN_LIST_UNLINKED_OBJECTS	Return the total tombstone sessions, keys and contexts	None	None	PCO	None	Success with fips_state = 2 or 3

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
CN_GET_PARTN_FINGERPRINT	Returns the fingerprint of partition keys, users and objects	SHA2-256 (FIPS 180-4) (#A1948)	User keys, PAC, POTAC, PAK	MCO PCO	E	Success with fips_state = 2 or 3
CN_CERT_AUTH_GET_CERT	Fetches certificates stored on the HSM. Certificates like VENDOR_CERT HSM_CERT PARTITION_OWNER_CERT PARTITION_CERT PARTITION_CERT_ISSUED_BY_HSM HSM_OWNER_CERT HSM_CERT_ISSUED_BY_HO	None	MARC FMAC POAC POTAC PAC AOAC AOTAC	MCO/PCO/ PCU/AU/UN -AUTH	R R R R R R R	Success with fips_state = 2 or 3
CN_CERT_AUTH_STORE_CERT	Store the partition owner certificate or the partition certificate signed by the partition owner or HSM certificate signed by vendor, HSM owner certificate and HSM certificate signed by HSM owner.	RSA SigVer (FIPS 186-4) [#A1948] ECDSA SigVer(FIPS186-4) [#A1948]	AOTAC POTAC AOAC POAC	MCO PCO	W W W W	Success with fips_state = 2 or 3
CN_GET_KBK_SLOT_INFO	Get the stored fixed (KBK) keys ekcv information	None	None	MCO PCO	None	Success with fips_state = 2 or 3
CN_SET_KBK_PRIMARY	Set the latest stored fixed (KBK) key as the primary key for backup.	None	None	MCO PCO	None	Success with fips_state = 2 or 3
CN_SHARE_OBJECT	Share an object between users.	None	User Keys	PCU	W	Success with fips_state = 2 or 3
CN_UNLOCK_CO	On providing response(signature) over the challenge thrown by HSM/Partition during CN_GET_CHALLENGE_CO (CfmGetChallengeCO), session will be marked with "unlock" privileges, allowing the user to generate PEK, zeroizeHSM and change the CO's self password. Session will remain in unlocked state only for 120 seconds.	RSA SigVer (FIPS 186-4) [#A1948]	POAC AOAC	UN-AUTH	E	Success with fips_state = 2 or 3
CN_UNLOCK_USER	Unlock CU or AU user that got locked up due to invalid login attempts	None	None	MCO PCO	None	Success with fips_state = 2 or 3
CN_GET_CORE_DUMP	To retrieve the binary core dump	None	None	UN-AUTH	None	Success with fips_state = 2 or 3
CN_MODULE_INFO	To retrieve board names and hardware versions (how module's versioning information, Show Status)	None	None	MCO	None	Success with fips_state = 2 or 3
CN_WRAP_KBK (Modes: KBK_WRAP_WITH_KEK, KBK_WRAP_WITH_CERT_AUTH_DERIVED_KEY, KBK)	Wrap KBK out of the HSM	AES-KW (KTS) (SP 800-38F) (#A1948), AES-KWP (KTS) (SP 800-38F) (#A1948)	KBK	PCO	R	Success with fips_state = 2 or 3

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
CN_UNWRAP_KBK (Modes: KBK_WRAP_WITH_KEK, KBK_WRAP_WITH_CERT_AUTH_DERIVED_KEY)	Unwraps KBK into the HSM	AES-KW (KTS) (SP 800-38F) (#A1948), AES-KWP (KTS) (SP 800-38F) (#A1948)	KBK	PCO	W	Success with fips_state = 2 or 3
CN_GET_CHALLENGE_CO	Gets a challenge to be signed by either HSM/Partition's owner to move the session to "unlocked" state using "unlockco" command.	RSA SigGen (FIPS186-4) (#A1948) Counter DRBG (SP 800-90Ar1) (#A1948)	POAC AOAC PAK FMAC FMAK DRBG Entropy/CTR_DRBG Internal State	UN-AUTH	R R E R E E	Success with fips_state = 2 or 3
CN_SET_M_VALUE	Set the current M value for a CO service.	None	None	PCO	None	Success with fips_state = 2 or 3
CN_CERT_AUTH_GET_CERT_REQ	Get the partition or HSM Certificate Signing Request (CSR).	None	AOAC POAC	MCO/PCO/ PCU/AU/ UN-AUTH	R R	Success with fips_state = 2 or 3
CN_CERT_AUTH_REMOVE_CERT	Stores or removes the Partition TA cert	None	None	PCO	None	Success with fips_state = 2 or 3
CN_UPDATE_LICENSE	Performs license update: uploads the license file and its signature file.	RSA SigVer (FIPS 186-4) [#A1948]	Manufacturer License Validation Key (MLVK)	MCO	E	Success with fips_state = 2 or 3
CN_GET_LICENSE_INFO	Gets data from license file	None	None	MCO	None	Success with fips_state = 2 or 3
CN_GET_DIAGLOG	Status output of the module for Critical, System and FRAM logs	None	None	MCO/UN-AUTH	None	Success with fips_state = 2 or 3
CN_NOR_HUK_OP	Shows the status of HUK in NOR Flash	None	None	MCO/UN-AUTH	None	Success with fips_state = 2 or 3
CN_APP_CLEANUP	Closes an application from HSM Partition	None	User Keys	MCO/PCO/ PCU/AU/UN-AUTH	Z	Success with fips_state = 2 or 3
CN_APP_CLEANUP_V2	Closes multiple applications from HSM Partition	None	User Keys	MCO/PCO/ PCU/AU/UN-AUTH	Z	Success with fips_state = 2 or 3
CN_SAFE_REBOOT	Restarts the HSM or specific partition	None	User Keys	UN-AUTH	Z	Success with

Service	Description	Algorithm and Standard	Keys and/or SSPs	Roles	Access Rights to Keys and/or SSPs	Indicator
						fips_state = 2 or 3
CN_GET_ALL_PARTITION_INFO	Diagnostic information for all the HSM Partition	None	None	None	None	Success with fips_state = 2 or 3

Indicator for Approved services:

The indicator is success for all Approved services when the partition is operated in the approved mode. The following codes are used in the Access Rights to Keys and/or SSPs column in Table 10 above.

G = Generate: The module generates or derives the SSP.

R = Read: The SSP is read from the module (e.g., the SSP is output).

W = Write: The SSP is updated, imported, or written to the module.

E = Execute: The module uses the SSP in performing a cryptographic operation.

Z = Zeroize: The module zeroizes the SSP.

All Approved services can be executed in the Non-Approved services mode with indicator of “success with fips_state=0”.

Approved services that also support the use of non-approved security functions are enumerated in the Non-Approved Services table below with their supported non-approved security functions.

Table 11 Non-Approved Services

Service	Description	Algorithms Accessed	Role	Indicator*
CN_GENERATE_PBE_KEY	Generate PBE DES3 key with the given password, salt, and iteration count. Make sure that HSM is initialized with fips_state=0. The fips_state parameter can be found in the hsm_config file.	Counter DRBG Allowed Per IG 2.4.A/ PBE	PCU	Success with fips_state = 0
LSPAY_GENERATE_ASYMM_KEY	Generates RSA KEY Pair (mod_len>= 2048-bit). Generates EC KEY PAIR (Curves: Nist P256, 224, 384, 521, Brain pool, x25519/448 and Decp256K1 and FRP)	RSA (non-compliant) ECDSA (non-compliant) KAS-ECC (non-compliant)	PCU	Success with fips_state = 0
LSPAY_GENERATE_SYMM_KEY	Generates symmetric key AES/ TDEA keys used for LSPay operations	Counter DRBG Allowed Per IG 2.4.A	PCU	Success with fips_state = 0
LSPAY_EXPORT_PUBLIC_KEY	Exports public key for RSA BYOK.	N/A	PCU	Success with fips_state = 0
LSPAY_IMPORT_KPK	Imports OAEP wrapped or ECDH_AES_PAD wrapped symmetric key.	RSA (non-compliant), EC-AES / ECDH KDF	PCU	Success with fips_state = 0
LSPAY_IMPORT_KEY	Import symmetric or asymmetric keys. Wrap mech: TR31, AES_CBC, AES_CBC_PAD.	AES (non-compliant) Triple-DES (non-compliant).	PCU	Success with fips_state = 0
LSPAY_IMPORT_TR34_KEY	Import symmetric keys using TR-34 unwrap.	RSA (non-compliant)	PCU	Success with fips_state = 0

Service	Description	Algorithms Accessed	Role	Indicator*
LSPAY_EXPORT_KEY	Exports symmetric key wrapped with TR31/AES_CBC/ AES_CBC_PAD.	AES (non-compliant) Triple-DES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_EXPORT_TR34_KEY	Exports symmetric keys wrapped with TR34 mechanism	RSA (non-compliant)	PCU	Success with fips_state = 0
LSPAY_TRANSLATE_KEY	Translates wrapped key from on KPK to other KPK.	AES (non-compliant) Triple-DES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_IMPORT_CERTIFICATE	Imports peer's certificate to read public key required in TR34. Import X901 certificate into HSM.	N/A	PCU	Success with fips_state = 0
LSPAY_IMPORT_DECIMAL_TABLE	Imports encrypted decimal table to be used in PIN APIs to decimalize native PIN.	AES (non-compliant) Triple-DES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_GENERATE_CSR	Create CSR with given key pair.	RSA (non-compliant) ECDSA (non-compliant)	PCU	Success with fips_state = 0
LSPAY_DERIVE_KEY	Derives DUKPT working key from the BDK.	AES (non-compliant) DES Triple-DES (non-compliant))	PCU	Success with fips_state = 0
LSPAY_ENCRYPT	Encrypts input data or PIN.	AES (non-compliant) Triple-DES (non-compliant) DES Double-DES	PCU	Success with fips_state = 0
LSPAY_DECRYPT	Decrypts input data or PIN.	AES (non-compliant) Triple-DES (non-compliant) DES Double-DES	PCU	Success with fips_state = 0
LSPAY_DECRYPT_THEN_ENCRYPT	Decrypts the input cipher text with one key and encrypts with another key.	AES (non-compliant) Triple-DES (non-compliant) DES Double-DES	PCU	Success with fips_state = 0
LSPAY_MAC_GEN	Computes MAC on input data. Algorithm used: DES/Triple-DES	DES MAC Triple-DES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_MAC_VERIFY	Verifies MAC with calculated AMC on input data.	DES MAC Triple-DES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_MAC_TRANSLATE	Translates MAC by using new key on input data.	DES MAC Triple-DES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_FPE_ENCRYPT	Performs FPE FF1/FF3-1 encrypt operation on input data.	AES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_FPE_DECRYPT	Performs FPE FF1/FF3-1 decrypt operation on input data.	AES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_SIGN	Performs sign and verify on input data.	RSA (non-compliant) EDDSA (non-compliant)	PCU	Success with fips_state = 0
LSPAY_SIGN_VERIFY	Verifies sign on input data.	RSA (non-compliant) EDDSA (non-compliant)	PCU	Success with fips_state = 0
LSPAY_PINBLK_TRANSLATE	Decrypts the input PIN using decryption key, translates to given PIN format and encrypts with another key.	AES (non-compliant) Triple-DES (non-compliant) RSA (non-compliant)	PCU	Success with fips_state = 0

Service	Description	Algorithms Accessed	Role	Indicator*
LSPAY_DERIVE_PIN_FROM_OFFSET	Derive PIN from given offset. Encrypts validation data with DES EDE. Derives native PIN, then offset will be added to derive IBM PIN. PIN will be encoded in given ISO format. Encrypt encoded PIN with PIN encryption key.	AES (non-compliant) Triple-DES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_DERIVE_OFFSET_FROM_PIN	Generates IBM offset from given PIN Decrypt and decode received PIN. Generate native from given validation data. Subtract decoded PIN from native PIN t to get PIN offset.	AES (non-compliant) Triple-DES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_VERIFY_PIN	Verifies given PIN . Decrypt and decode received PIN. Generate native from given validation data. Add offset to native PIN. Compare resultant PIN with received PIN.	AES (non-compliant) Triple-DES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_PVV_GEN	Perform PVV generation on PIN and PAN data.	AES (non-compliant) Triple-DES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_PVV_VERIFY	Verifies given PVV.	AES (non-compliant) Triple-DES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_EMV_GENVERIFY	Perform EMV crypto operations. Generate ARPC. Generate or Verify ARQC.	AES (non-compliant) Triple-DES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_EMV_SECURE_MSG_GEN	Generates MAC over secure message.	AES (non-compliant) Triple-DES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_CVV_GEN	Generates CVV, CVV2, iCVV on given card details.	AES (non-compliant) Triple-DES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_CVV_VERIFY	Verifies CVV with given card details	Triple-DES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_KEY_SHARE_CREATE	Creates components of Key	Shamir's key share algorithm	PCU	Success with fips_state = 0
LSPAY_KEY_SHARE_EXPORT_COMPONENT	Exports created components in encrypted format	AES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_KEY_SHARE_IMPORT_COMPONENT	Imports component of the key.	AES (non-compliant)	PCU	Success with fips_state = 0
LSPAY_KEY_SHARE_COMBINE_INIT	Starts combine key init.	N/A	PCU	Success with fips_state = 0
LSPAY_KEY_SHARE_COMBINE_KEY	Combines all components of the key.	Shamir's key share algorithm	PCU	Success with fips_state = 0
LSPAY_KEY_SHARE_ZEROIZE	Erases all components of the key.	N/A	PCU	Success with fips_state = 0
LSPAY_MFK_GENERATE	Generates MFK key.	Counter DRBG Allowed Per IG 2.4.A	PCO	Success with fips_state = 0
LSPAY_MFK_GET_INFO	Returns MFK information for partition.	N/A	PCO	Success with fips_state = 0
LSPAY_MFK_SET_PRIMARY	Sets MFK as primary.	N/A	PCO	Success with fips_state = 0
LSPAY_MFK_DELETE	Deletes MFK.	N/A	PCO	Success with fips_state = 0
LSPAY_FUNCTIONALITY_GET	Gets status of enabled/disabled LSPay services	N/A	PCO	Success with fips_state = 0
LSPAY_FUNCTIONALITY_SET	Enable/Disable services	N/A	PCO	Success with fips_state = 0

Service	Description	Algorithms Accessed	Role	Indicator*
LSPAY_EXPORT_KPK	Export a Key Block Protection Key (KBPK)	RSA (non-compliant), EC-AES / ECDH KDF	PCU	Success with fips_state = 0
LSPAY_IMPORT_PUBLIC_KEY	Import an RSA public key	RSA (non-compliant)	PCU	Success with fips_state = 0
LSPAY_VALIDATE_PUBLIC_KEY	Validates the RSA public key	RSA (non-compliant)	PCU	Success with fips_state = 0
CN_GENERATE_KEY_PAIR (non-compliant)	Generates asymmetric keys (RSA/ ECC). Updates the public and private key handles in the output on return. Caveats in Non Approved apart from approved mode: • RSA 1024 bits allowed along with all odd public exponent i.e. even lesser than 65537. • NID_X9_62_prime192v 1/NID_sect163k1/NID_ED255 19/ • NID_sect163r2 /NID_secp192k1/NID_brainpo olP160r1/ • NID_brainpoolP192r1 /NID_X25519/ • NID_X448	RSA (non-compliant) ECDSA (non-compliant) KAS-ECC (non-compliant) ML-KEM (non-compliant) ML-DSA (non-compliant)	PCU	Success with fips_state = 0
CN_GENERATE_KEY (non-compliant)	Generates a symmetric key of given key type and length. Caveats in Non-Approved apart from approved mode: DES token key is allowed.	Counter DRBG Allowed Per IG 2.4.A	PCU	Success with fips_state = 0
CN_CREATE_OBJECT (non-compliant)	Imports a public key into HSM. Caveats in Non Approved apart from approved mode: • RSA 1024 bits allowed • NID_ED25519/ NID_secp192k1/ NID_brainpoolP160r1/ • NID_brainpoolP192r1/ NID_X25519/NID_X448	None	PCU	Success with fips_state = 0
CN_UNWRAP_KEY (non-compliant)	Unwraps a key with an AES/ Triple-DES/RSA private key existing on HSM or KLK. Takes the output wrapped data of wrapKey2 command. Caveats in Non Approved apart from approved mode: • RSA 1024 bit • RSA PKCS1V1.5 Unwrap • NID_X9_62_prime192v1/ NID_sect163k1/ NID_ED25519/ • NID_sect163r2 / NID_secp192k1/ NID_brainpoolP160r1/ • NID_brainpoolP192r1 / NID_X25519/ • NID_X448 • Triple-DES	RSA (non-compliant), EC-AES / ECDH KDF AES (non-compliant) Triple-DES (non-compliant) ML-KEM (non-compliant)	PCU	Success with fips_state = 0
CN_WRAP_KEY (non-compliant)	Wraps sensitive (private and symmetric) keys from the HSM to the host.	AES (non-compliant) Triple-DES (non-compliant)	PCU	Success with fips_state = 0

Service	Description	Algorithms Accessed	Role	Indicator*
	Caveats in Non Approved apart from approved mode: • AES-ECB mode • AES-CBC mode • AES-CBC-PAD mode • Triple-DES ECB mode • Triple-DES CBC mode • Triple-DES NIST Wrap mode • RSA-PKCS1V1.5 Wrap	RSA (non-compliant) ML-KEM (non-compliant)		
CN_EXTRACT_MASKED_OBJECT (non-compliant)	Extracts a masked object; i.e., retrieves an object by wrapping it with a masking key shared by the process of cloning.	AES (non-compliant)	PCU PCO AU	Success with fips_state = 0
CN_STORE_FW_SIGNING_KEY (non-compliant)	Configure an RSA or EC public key into HSM as AO attestation key. These keys can be of modulus 1024, 2048, 3072, and 4096 or a supported 256 bits, 384 bits or 521 bits EC curve . Caveats in Non Approved is 192-bit curves supported	RSA (non-compliant) ECDSA (non-compliant)	MCO	Success with fips_state = 0
CN_ME_PKCS_LARGE (non-compliant) CN_ME_PKCS (non-compliant)	ModExp and PKCS#1v1.5 and PKCS#1v2.2 Sign and verify. PKCS#1v1.5 and PKCS#1v2.2 encrypt and decrypt	RSA (non-compliant)	PCU	Success with fips_state = 0
CN_STORE_VENDOR_PRE_SHARED_KEY (CN_STORE_KBK_SHARE) (non-compliant)	Stores fixed keys (KBK) for backup. Including PKCS#1v1.5	RSA (non-compliant) AES (non-compliant)	Manufacturer	Success with fips_state = 0
CN_INSERT_MASKED_OBJECT (non-compliant)	Inserts a masked object into an HSM that Is extracted from another HSM.	AES (non-compliant) Triple-DES (non-compliant)	PCU PCO AU	Success with fips_state = 0
CN_ENCRYPT_SESSION (non-compliant)	Enables encrypted communication channel. Caveat is Non Approved mode allow the additional Cipher suite E2E_RSA_AES128_GCM_SHA256 , E2E_RSA_AES128_GCM_SHA384, OQSKEM_DEFAULT_ECDHE_RSA_AES128_GCM_SHA256 and OQSKEM_DEFAULT_ECDHE_RSA_AES256_GCM_SHA384	RSA (non-compliant), EC-AES / ECDH KDF ML-KEM (non-compliant)	UN-AUTH	Success with fips_state = 0
CN_DERIVE_KEY (non-compliant)	Derives a key using a supported KDF mechanism with the params given by the user.	AES (non-compliant) Triple-DES (non-compliant) RSA (non-compliant) EC-AES / ECDH KDF ML-KEM (non-compliant)	PCU	Success with fips_state = 0
CN_PQC_GENERATE_KEY_PAIR	Generates the Post quantum asymmetric key pair (ML-KEM and ML-DSA).	ML-KEM (non-compliant) ML-DSA (non-compliant)	PCU	Success with fips_state = 0
CN_PQC_CRYPTO_SIG_GEN	Generates a signature using ML-DSA algorithm.	ML-DSA (non-compliant) Hash DRBG (non-compliant)	PCU	Success with fips_state = 0
CN_PQC_CRYPTO_SIG_VERIFY	Verifies a signature using ML-DSA algorithm.	ML-DSA (non-compliant)	PCU	Success with fips_state = 0

Service	Description	Algorithms Accessed	Role	Indicator*
CN_PQC_CRYPTO_MULTICALL_SIG_GEN	Generates a signature using ML-DSA algorithm. Minor Ops: MINOR_OP_START, MINOR_OP_UPDATE, and MINOR_OP_FINISH	ML-DSA (non-compliant)	PCU	Success with fips_state = 0
CN_PQC_CRYPTO_MULTICALL_SIG_VERIFY	Verifies a signature using ML-DSA algorithm. Minor Ops: MINOR_OP_START, MINOR_OP_UPDATE, and MINOR_OP_FINISH	ML-DSA (non-compliant)	PCU	Success with fips_state = 0
CN_PQC_CRYPTO_HYBRID_SIG_GEN	Generates a signature using ML-DSA and ECDSA sign algorithm.	ML-DSA (non-compliant) ECDSA (non-compliant)	PCU	Success with fips_state = 0
CN_PQC_CRYPTO_HYBRID_SIG_VERIFY	Verifies a signature using ML-DSA and ECDSA sign algorithm.	ML-DSA (non-compliant) ECDSA (non-compliant)	PCU	Success with fips_state = 0
HPS_CREATE_PARTITION	Creates a Platform partition with the given name.	AES (non-compliant) ECDSA (non-compliant)	MCO	Success with fips_state = 0
HPS_PART_FW_UPDATE_BEGIN HPS_PART_FW_UPDATE HPS_PART_FW_UPDATE_END	Update the Platform firmware for the given partition.	RSA (non-compliant) ECDSA (non-compliant)	MCO	Success with fips_state = 0
HPS_PARTITION_MGMT	Handles the commands for Platform partition firmware life cycle such as START, STOP and STATUS	None	MCO	Success with fips_state = 0
HPS_PARTITION_INFO	Gets the Platform partition firmware information such as version, status, memory, CPU, etc.	None	MCO	Success with fips_state = 0
HPS_DELETE_PARTITION	Deletes a Platform partition and all associated data.	None	MCO	Success with fips_state = 0
CN_UPDATE_LICENSE (non-compliant)	Performs license update: uploads the license file and its signature file.	RSA (non-compliant)	MCO	Success with fips_state = 0
CN_GET_LICENSE_INFO (non-compliant)	Gets data from license file	None	UN-AUTH	Success with fips_state = 0

Indicator for non-Approved services:

The indicator is success for all non-approved services only when the partition is operated in the non-approved mode. The non-approved services will fail FIPS POLICY MISMATCH when partition is operated in the approved mode.

5 Firmware Security

Firmware integrity is verified by the bootloader during the boot up using EDC (CRC32) for itself and using RSA 2048 signatures and SHA-256 for next level image. The integrity test can be run on-demand by resetting the module by power-cycling it or by PCIe function reset.

As part of firmware load operation, the new firmware's integrity and authenticity is verified by the active firmware using RSA 2048 signatures and SHA-256.

6 Operational Environment

The module implements a limited operational environment running SMP Linux operating system on 64-bit Armv8.2-based control processor. FIPS 140-3 Area 6 Operational Environment requirements do not apply to the module in this validation.

Any firmware loaded into this module that is not shown on the module certificate is out of the scope of this validation and requires a separate FIPS 140-3 validation.

The module is a hardware module with a limited operational environment and Physical Security Level 3.

7 Physical Security

7.1 Physical Security Mechanisms

The module's cryptographic boundary is defined to be the outer perimeter of the hard epoxy enclosure containing the hardware and firmware components. The module is opaque and completely conceals the internal components of the cryptographic module. The epoxy enclosure of the module prevents physical access to any of the internal components without having to destroy the module. There are no operator required actions.

Note: The module's hardness testing was only performed at ambient temperature (23°C); no assurance is provided for Level 3 hardness conformance at any other temperature.

7.2 Tamper Evidence

The module is coated in hard epoxy, such that any physical breach attempt leaves behind evidence of tamper. This is shown in the figure below.

Figure 3 Cryptographic Module Showing Tamper Evidence

While the module is designed to prevent successful tampering (any physical breach to module circuitry is likely to destroy the module, as per FIPS 140-3 Level 3 Physical Security requirements), the module should still be checked periodically for attempts. Guidelines are provided in the table below.

Table 12 Physical Security Inspection Guidelines

Physical Security Mechanism	Recommended Frequency of Inspection/Test	Inspection/Test Guidance Details
Epoxy Coating	12 Months	Examine surface of module for scratched or damaged epoxy, especially if circuitry shows.
Battery life	6 Months	If the HSM has not been powered on in the last six months, then power it on for at least an hour.

If the module is found to be meaningfully damaged or tampered with (e.g., circuitry is showing or other significant damage has occurred), it should be removed from use and destroyed.

Table 13 EFP/EFT

	Temperature or voltage measurement	Specify EFP or EFT	Specify if This Condition Results in a Shutdown or Zeroization
Low Temperature	-13C (Junction temperature)	EFP	Shutdown
High Temperature	100C (Junction temperature)	EFP	Shutdown
Low Voltage	3.01V (PCIe 3.3V Aux) 3.08V (PCIe 3.3V Rail) 10.9V (PCIe 12V Rail)	EFP	Shutdown

	Temperature or voltage measurement	Specify EFP or EFT	Specify if This Condition Results in a Shutdown or Zeroization
High Voltage	3.68V (PCIe 3.3V Aux) 3.75V (PCIe 3.3V Rail) 13.5V (PCIe 12V Rail)	EFP	Shutdown

Table 14 Hardness Testing Temperature Ranges

	Hardness-Tested Temperature Measurements
Low Temperature	-13C (Junction temperature)
High Temperature	100C (Junction temperature)

8 Non-Invasive Security

N/A due to the module not claiming to implement any non-invasive security protection mechanisms.

9 Sensitive Security Parameter Management

9.1 Definition of Critical Security Parameters (CSPs)

The Manufacturer FIPS Data Encryption Key (MFDEK) and HSM Master Partition Master Encryption Key are stored in plaintext form in eMMC Flash. The Partition Master Encryption Key (PMEK) is stored encrypted under the HSM Master Partition Master Encryption Key. All other keys and CSPs stored in the persistent memory are encrypted by the MFDEK, HSM Master Partition Master Encryption Key, or PMEK. All general purpose user CSPs are generated/created by the PCU, and these CSPs can be shared between multiple PCUs.

In Table 15, the following notations are used to indicate the type of zeroization:

D = Manually zeroized (via CN_DESTROY_OBJECT service)

E = Zeroized right after use (Memory is wiped with zeros, immediately after use)

MFZ = Zeroize all Partitions' SSPs, including the HSM adapter owner programmed ones (Brings HSM to factory state. Factory reset via CN_ZEROIZE service with factory-reset as argument with MCO credentials)

MZ = Zeroize all Partitions' SSPs, except vendor programmed ones (Master Partition Regular CN_ZEROIZE)

PD = Zeroize all SSPs in the Partition and then delete the User Partition (via CN_DELETE_PARTITION service)

PFZ = Zeroize all SSPs in the Partition (Factory reset via CN_ZEROIZE service with factory-reset as argument with PCO credentials)

PZ = Zeroize all User SSPs in the Partition Regular p (Equivalent to User Partition Regular zeroize via CN_ZEROIZE service)

S = Zeroized on session close (Session Close via CN_CLOSE_SESSION, CN_APP_FINALIZE and CN_CLOSE_PARTITION_SESSIONS)

VZ = Vendor zeroize (Zeroizes all SSPs including vendor programmed configuration and CSPs. Makes the module unusable; the module must be sent back to the vendor for re-programming.) via CN_VENDOR_ZEROIZE service.

* The SSPs are zeroized securely by writing zeros to memory.
* Private/Secret keys are always encrypted when doing export/import, and key encapsulation mechanisms are listed in the respective CSP row.
* All CSPs from Table 15 and Table 16 are entered through automated electronic entry mechanisms.

Table 15 SSPs

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import/ Export	Establish-ment	Storage	Zeroiza-tion	Use & Related Keys
DRBG Entropy (OCTEON HW RBG)	256-bit	ENT (P) (SP 800-90B)	ENT (P) (SP 800-90B)	N/A	SSP Generation	N/A	PZ MZ MFZ PFZ PD VZ	The entropy input string and seed for the Approved DRBG. Each version of the DRBG has its own DRBG Entropy CSP.
CTR_DRBG Internal State (V and Key)	256-bit	Counter DRBG (SP 800-90Ar1) (#A1948)	(Derived Entropy from HW)	N/A	SSP Generation	N/A	PZ MZ MFZ PFZ PD VZ	The internal state (V, Key) for the Counter DRBG.
HASH_DRBG Internal State (V and C)	256-bit	Hash DRBG (SP 800-90Ar1) (#A1947)	Derived Entropy from Counter DRBG (SP800-90Ar1) (#A1948))	N/A	SSP Generation	N/A	PZ MZ MFZ PFZ PD VZ	The internal state (V,C) for the Hash DRBG.
Manufacturer FIPS Data Encryption Key (MFDEK)	256-bit	AES-CBC (SP 800-38A) (#A1948)	CKG SP 800-133Rev2 Section 6.1 Direct symmetric key generation using unmodified DRBG output (SP800-90Ar1) (#A1948)	No	SSP generation	NOR (plaintext)	VZ	Use: AES 256-bit key used to encrypt manufacturer keys stored in persistent storage of the HSM. Related keys: FMAK, PAK, MFKBK, OKBK,POKBK, FMAEC, FMAEK
HSM Master Partition Master Encryption Key (MMEK)	256-bit	AES-CBC (SP 800-38A) (#A1948)	CKG SP 800-133Rev2 Section 6.1 Direct symmetric key generation using unmodified DRBG output (SP800-90Ar1) (#A1948)	No	SSP generation	MCU (plaintext)	MZ	Use: AES 256-bit key used to encrypt Master Partition CSPs and authentication data stored in persistent storage of the HSM. Related keys:PMEK
Partition Master Encryption Key (PMEK)	256-bit	AES-CBC (SP 800-38A) (#A1948)	CKG SP 800-133Rev2 Section 6.1 Direct symmetric key generation using unmodified DRBG output (SP800-90Ar1) (#A1948)	No	SSP Generation	eMMC flash (Encrypted by MMEK, AES-CBC #A1948)	PZ	Use: AES 256-bit key used to encrypt partition CSPs and authentication data stored in persistent storage of the HSM
Partition Data Encryption Key (PDEK)	256-bit	AES-KWP (SP 800-38F) (#A1948)	CKG SP 800-133Rev2 Section 6.1 Direct symmetric key generation using unmodified DRBG output (SP800-90Ar1) (#A1948)	No	SSP Generation	eMMC flash (Encrypted by MMEK, AES-CBC #A1948)	PD	Use: AES 256-bit key used to wrap PAK and POKBK
HSM FIPS Master Authentication Key (FMAK)	150-bit	RSA SigGen (FIPS186-4) (#A1948) RSA SigVer (FIPS186-4) (#A1948) KTS-IFC (KTS) (SP 800-56Br2) (#A1948)	CKG SP 800-133Rev2 Section 6.1 Direct symmetric key generation using unmodified DRBG output (SP800-90Ar1) (#A1948)	No	SSP generation	NOR flash (Encrypted by MMEK, AES-CBC #A1948)	VZ	Use: A unique 4096-bit RSA private key. Used to identify the HSM when in the FIPS operating mode. Related keys: PAC, FMAC
HSM FIPS Master Authentication ECC Key (FMAEK)	256-bit	ECDSA SigVer (FIPS186-4) (#A1948) ECDSA SigGen (FIPS186-4) (#A1948)	CKG SP 800-133Rev2 Section 6.1 Direct symmetric key generation using unmodified DRBG output (SP800-90Ar1) (#A1948)	No	SSP generation	NOR flash (Encrypted by MMEK, AES-CBC #A1948)	VZ	Use: A unique ECC P521 private key. Used to identify the HSM when in the FIPS operating mode. Related keys: PAC, FMAC
HSM Endorsement ECC Key	128-bit	ECDSA SigVer (FIPS186-4)	CKG SP 800-133Rev2 Section 6.1 Direct	No	SSP generation	NOR flash (Encrypted by MMEK,	VZ	Use: A unique ECC P256 private key.

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import/ Export	Establish-ment	Storage	Zeroiza-tion	Use & Related Keys
(HSMEK)		(#A1948) ECDSA SigGen (FIPS186-4) (#A1948)	symmetric key generation using unmodified DRBG output (SP800-90Ar1) (#A1948)			AES-CBC #A1948)		Used to identify the HSM when in the FIPS operating mode. Related keys: PAC, FMAC
Partition Authentication Key (PAK)	112-bit	RSA SigGen (FIPS186-4) (#A1948) RSA SigVer (FIPS186-4) (#A1948)	CKG SP 800-133Rev2 Section 6.1 Direct symmetric key generation using unmodified DRBG output (SP800-90Ar1) (#A1948)	No	SSP generation	eMMC flash (Encrypted by PDEK, AES-CBC #A1948)	PD	A unique 2048-bit RSA private key used to identify the HSM partition
Secure Auth Shared Secret (SAZ)	112-bit	KDF SP800-108 (#A1948)	KAS (SP800- 56Br2) KAS-IFC HKDF (SP800-56Br2) KAS-IFC OneStep (SP800-56Br2) KAS-IFC TwoStep (SP800-56Br2)	No	Key agreement	In memory (plaintext)	S	Shared secret Z for SP 800-56Br2 KAS2, using PAK and POAC
PswdEncKey (PEK)	256-bit	AES Cert A1948, Key unwrapping, per IG D.G AES-CBC (SP 800- 38A) (#A1948)	KAS-IFC HKDF (SP800-56Br2) KAS-IFC OneStep (SP800-56Br2) KAS-IFC TwoStep (SP800-56Br2)	No	Key agreement	In memory (Encrypted by PMEK, AES-CBC #A1948)	PZ	AES-256 key for encrypting user passwords during user creation and authentication.
Login Passwords	8 to 32 Characters	PBKDF (SP 800-132) (#A1948)	N/A	Yes (import) (Encrypted by PEK, AES-CBC #A1948)	Key Transport	In eMMC Flash (Encrypted by PMEK, AES-CBC #A1948)	PD	String of 8 to 32 alphanumeric characters.
Partition Key Loading Private Key	256-bit	KAS-ECC SP800-56Ar3 (#A1948) KAS-IFC-SSC (SP 800-56Br2) (#A1948) KAS-IFC HKDF (SP800-56Br2) KAS-IFC OneStep (SP800-56Br2) KAS-IFC TwoStep (SP800-56Br2)	CKG SP 800-133Rev2 Section 6.1 Direct symmetric key generation using unmodified DRBG output (SP800-90Ar1) (#A1948) ECDSA KeyGen (FIPS186-4) [#A2393] RSA KeyGen (FIPS186-4) (#A2393) KAS-IFC HKDF (SP800-56Br2) KAS-IFC OneStep (SP800-56Br2) KAS-IFC TwoStep (SP800-56Br2)	No	SSP generation	In memory (plaintext)	E	ECC 521-bit or RSA 2048-bit key used in SP 800-56Ar3 C (2,0, ECC DH) or SP 800-56Br2 KAS2 to agree on Z during key loading.
Partition Key Loading Shared Secret (KLSZ)	256-bit	KDF SP800-108 (#A1948)	KAS KDA HKDF (SP 800-56Ar3) KAS-IFC HKDF (SP800-56Br2) KAS-IFC OneStep (SP800-56Br2) KAS-IFC TwoStep (SP800-56Br2)	No	Key agreement	In memory (plaintext)	E	Shared secret Z for SP 800-56Ar3 C (2,0, ECC DH) or SP 800-56Br2 KAS2.
Partition Key Loading Key (KLK)	256-bit	AES-CBC (SP 800-38A) (#A1948) AES-KW (KTS) (SP 800-38F)	KAS KDA HKDF (SP 800-56Ar3) KAS-IFC HKDF (SP800-56Br2)	No	Key agreement	eMMC flash (encrypted by PMEK, AES-CBC	PZ	A 256-bit AES key derived from Z; used to decrypt the imported CSPs.

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import/ Export	Establish-ment	Storage	Zeroiza-tion	Use & Related Keys
		(#A1948) AES-KWP (KTS) (SP 800-38F) (#A1948) AES-GCM (SP 800-38D) (#A1947)	KAS-IFC OneStep (SP800-56Br2) KAS-IFC TwoStep (SP800-56Br2)			#A1948)		
Manufacturer FIPS Key Backup Key (MFKBK)	256-bit	KDF SP800-108 (#A1948)	No	Yes (Import) KTS-IFC (KTS) (SP800-56Br2) (#A1948)	Key transport	eMMC flash (encrypted by MMEK, AES-KWP #A1948)	VZ	AES 256-bit key used to derive KBK.
HSM Owner KBK (OKBK)	256-bit	KDF SP800-108 (#A1948)	No	Yes (Import) KTS-IFC (KTS) (SP800-56Br2) (#A1948)	Key transport	eMMC flash (encrypted by MMEK AES-KWP #A1948)	MFZ	AES 256-bit key used to derive KBK.
MCO Recovery Key (MCO_RK)	256-bit	AES-KW (KTS) (SP 800-38F) (#A1948)	No	Yes (Import) AES-KW (KTS) (SP 800-38F) (#A1948)	Key transport	MCU (plaintext)	MFZ VZ	AES 256-bit key used to double encrypt the manufacturer keys.
Partition Owner KBK (POKBK)	256-bit	KDF SP800-108 (#A1948)	No	Yes (Import) KTS-IFC (KTS) (SP800-56Br2) (#A1948)	Key transport	eMMC flash (encrypted by PDEK AES-KWP (#A1948)	MZ	AES 256-bit key used to derive KBK.
HSM Key Backup Key (KBK)	256-bit	AES-CBC (SP 800-38A) (#A1948) AES-KW (KTS) (SP 800-38F) (#A1948) AES-KWP (KTS) (SP 800-38F) (#A1948)	KDF SP800-108 (#A1948)	No	Key derivation/ SSP generation	eMMC flash (Encrypted by MMEK AES-KW #A1948)	MZ	Key used to encrypt/decrypt the backup session key.
Backup session key	256-bit	AES-CBC (SP 800-38A) (#A1948) AES-KW (KTS) (SP 800-38F) (#A1948) AES-KWP (KTS) (SP 800-38F) (#A1948)	CKG SP 800-133Rev2 Section 6.1 Direct symmetric key generation using unmodified DRBG output Counter DRBG (SP800-90Ar1) (#A1948)	No	SSP generation	In memory (plaintext)	E	Key used to backup and restore partition data.
Partition masking key	256-bit	AES-KW (KTS) (SP 800-38F) (#A1948)	CKG SP 800-133Rev2 Section 6.1 Direct symmetric key generation using unmodified DRBG output Counter DRBG (SP800-90Ar1) (#A1948)	No	SSP generation	eMMC flash (encrypted by PMEK, AES-CBC #A1948)	PZ	AES-256 key for key wrapping. Used to import/export CSPs and masked objects.
Partition Cloning Private Key (PCPK)	256-Bit	KAS-ECC (KAS) (SP 800-56Ar3) KAS-IFC HKDF (SP800-56Br2) KAS-IFC OneStep (SP800-56Br2) KAS-IFC TwoStep (SP800-56Br2)	CKG SP 800-133Rev2 Section 6.1 Direct symmetric key generation using unmodified DRBG output (SP800-90Ar1) (#A1948) ECDSA KeyGen (FIPS186-4) (#A1948) RSA KeyGen (FIPS186-4) (#A1948)	No	SSP generation	In memory (plaintext)	E	ECC 521-bit or RSA 2048-bit ephemeral Private Key used in SP 800-56Ar3 C (2,0, ECC DH) or SP 800 -56Br2 KAS2 -bilateral - confirmation key agreement to generate shared secret Z. At HSM partition level, used to establish secure channel for cloning process (to export partition masking key).
Partition Cloning Shared	256-Bit	KDF SP800-108	KAS-ECC (KAS) (SP	No	Key	In memory	E	Shared secret Z for SP 800-56Ar3 C (2,0, ECC DH) or SP

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import/ Export	Establish-ment	Storage	Zeroiza-tion	Use & Related Keys
Secret (CSSZ)		(#A1948)	800-56Ar3)		agreement	(plaintext)		800-56Br2 KAS2 -bilateral -confirmation scheme.
Partition Cloning Session Key (PCSK)	256-Bit	AES-KW (KTS) (SP 800-38F) (#A1948) AES-KWP (KTS) (SP 800-38F) (#A1948)	KAS-ECC (KAS) (SP 800-56Ar3)	No	Key agreement	In memory (plaintext)	E	AES 256 key for encryption and decryption of partition masking key.
Partition Cloning Session MAC Key (PCSMK)	256-bit	HMAC-SHA2-256 (FIPS 198-1) (#A1947)	KAS-ECC (KAS) (SP 800-56Ar3)	No	Key agreement	In memory (plaintext)	E	HMAC SHA256 key used for key confirmation during SP 800-56Ar3 key agreement.
Asymmetric private keys (user keys)	112-256 bit	ECDSA KeyVer (FIPS186-4) (#A1948) ECDSA SigGen (FIPS186-4) (CVL) (#A1947) RSA Decryption Primitive (SP 800-56Br2) (CVL) (#A1948) RSA SigGen (FIPS186-4) (#A1948) RSA SigVer (FIPS186-4) (#A1948) RSA KeyGen (FIPS186-4) (#A2393) KAS-ECC-SSC SP800-56Ar3 (#A1947) KTS-IFC (SP 800-56Br2) (#A2393) RSA Decryption Primitive (SP 800-56Br2) (CVL) (#A1947) RSA Signature Primitive (CVL) (FIPS 186-4) (#A1947) KTS-IFC (KTS) (SP 800-56Br2) (#A2393) KTS-IFC (KTS) (SP 800-56Br2) (#A1948) KAS-ECC (KAS) (SP 800-56Ar3) KDA HKDF SP800-56Cr1 (#A1948) KAS-ECC-SSC SP800-56Ar3 (#A1948) KDF ANS 9.63 (CVL) (SP 800-135r1) (#A1948) KAS KDA HKDF (SP 800-56Ar3) KAS KDA	CKG SP 800-133Rev2 Section 6.1 Direct symmetric key generation using unmodified DRBG output (SP800-90Ar1) (#A1948) ECDSA KeyGen (FIPS186-4) (#A2393) RSA KeyGen (FIPS186-4) (#A2393)	Yes (Import/Export) AES-KW (KTS) (SP 800-38F) (#A1948) AES-KWP (KTS) (SP 800-38F) (#A1948) AES-GCM (KTS) (SP 800-38D) (#A1948) KTS-IFC (KTS) (SP 800-56Br2) (#A2393) KTS-IFC (KTS) (SP 800-56Br2) (#A1948)	SSP generation/ Key transport	In memory (plaintext)/ eMMC flash (encrypted by PMEK, AES-CBC #A1948)	D, S	RSA/ECDSA/ECDH general purpose keys.

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import/ Export	Establish-ment	Storage	Zeroiza-tion	Use & Related Keys
		ONESTEP (SP 800-56Ar3) KAS KDA TWOSTEP (SP 800-56Ar3) KAS ANS 9.63 (SP 800-56Ar3) KTS-IFC (KTS) (SP 800-56Br2) (#A2393) EC Diffie-Hellman with non-NIST recommended curves [#A1947] allowed per IGs D.F and C.A ECDSA with non-NIST recommended curves [#A1947] allowed per IG C.A						
Asymmetric private session keys (user keys)	112-256 bit	ECDSA SigGen (FIPS186-4) (CVL) (#A1947) RSA Decryption Primitive (SP 800-56Br2) (CVL) (#A1948) RSA Signature Primitive (CVL) (FIPS 186-4) (#A1947) RSA SigGen (FIPS186-4) (#A1948) RSA SigVer (FIPS186-4) (#A1948) KAS-ECC-SSC SP800-56Ar3 (#A1947) KTS-IFC (SP 800-56Br2) (#A2393) RSA Decryption Primitive (SP 800-56Br2) (CVL) (#A1947) KTS-IFC (KTS) (SP 800-56Br2) (#A2393) KTS-IFC (KTS) (SP 800-56Br2) (#A1948) KAS-ECC (KAS) (SP 800-56Ar3) KDA HKDF SP800-56Cr1 (#A1948) KAS-ECC-SSC SP800-56Ar3 (#A1948) KDF ANS 9.63 (CVL) (SP 800-135r1) (#A1948)	CKG SP 800-133Rev2 Section 6.1 Direct symmetric key generation using unmodified DRBG output (SP800-90Ar1) (#A1948) ECDSA KeyGen (FIPS186-4) (#A2393) RSA KeyGen (FIPS186-4) (#A2393)	Yes (Import/Export) AES-KW (KTS) (SP 800-38F) (#A1948) AES-KWP (KTS) (SP 800-38F) (#A1948) AES-GCM (KTS) (SP 800-38D) (#A1948) KTS-IFC (KTS) (SP 800-56Br2) (#A2393) KTS-IFC (KTS) (SP 800-56Br2) (#A1948)	SSP generation/ Key transport	In memory (plaintext)	D, S	RSA/DSA/ECDSA/ECDH general purpose session keys.

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import/ Export	Establish-ment	Storage	Zeroiza-tion	Use & Related Keys
		KAS KDA HKDF (SP 800-56Ar3) KAS KDA ONESTEP (SP 800-56Ar3) KAS KDA TWOSTEP (SP 800-56Ar3) KAS ANS 9.63 (SP 800-56Ar3)KTS-IFC (KTS) (SP 800-56Br2) (#A2393) EC Diffie-Hellman with non-NIST recommended curves [#A1947] allowed per IGs D.F and C.A ECDSA with non-NIST recommended curves [#A1947] allowed per IG C.A						
Symmetric keys (user keys)	112-256 bit	AES-CBC (SP 800-38A) (#A1947) AES-ECB (SP 800-38A) (#A1947) AES-CTR (SP 800-38A) (#A1947) TDES-ECB (SP 800-38A) (#A1947) TDES-CBC (SP 800-38A) (#A1947) AES-CCM (SP 800-38C) (#A1947) AES-GCM (SP 800-38D) (#A1947) AES-GMAC (SP 800-38D) (#A1947) AES-CMAC (SP 800-38B) (#A1947) AES-KW (KTS) (SP 800-38F) (#A1948) AES-CMAC (SP 800-38B) (#A1948) TDES-KW (KTS) (SP 800-38F) (#A1948) AES-GCM (KTS) (SP 800-38D) (#A1948) AES-GCM (SP 800-38D) (#A1947) TDES-KW (SP 800-38F) (#A1948)	CKG SP 800-133Rev2 Section 6.1 Direct symmetric key generation using unmodified DRBG output OUNTER DRBG (SP800-90Ar1) (#A1948)	Yes (Import/Export) AES-KW (KTS) (SP 800-38F) (#A1948) AES-KWP (KTS) (SP 800-38F) (#A1948) AES-GCM (KTS) (SP 800-38D) (#A1948) KTS-IFC (KTS) (SP 800-56Br2) (#A2393) KTS-IFC (KTS) (SP 800-56Br2) (#A1948)	SSP generation/ Key transport	In memory (plaintext)/ eMMC flash (encrypted by PMEK, AES-CBC #A1948)	D, S	Triple-DES or AES general purpose keys.

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import/ Export	Establish-ment	Storage	Zeroiza-tion	Use & Related Keys
Symmetric session keys (user keys)	112-256 bit	AES-CBC (SP 800-38A) (#A1947) AES-ECB (SP 800-38A) (#A1947) TDES-ECB (SP 800-38A) (#A1947) TDES-CBC (SP 800-38A) (#A1947) AES-CCM (SP 800-38C) (#A1947) AES-GCM (SP 800-38D) (#A1947) AES-GMAC (SP 800-38D) (#A1947) AES-CMAC (SP 800-38B) (#A1947) AES-KW (KTS) (SP 800-38F) (#A1948) AES-CMAC (SP 800-38B) (#A1948) TDES-KW (KTS) (SP 800-38F) (#A1948) AES-GCM (KTS) (SP 800-38D) (#A1948) AES-GCM (SP 800-38D) (#A1947) TDES-KW (SP 800-38F) (#A1948)	CKG SP 800-133Rev2 Section 6.1 Direct symmetric key generation using unmodified DRBG output OUNTER DRBG (SP800-90Ar1) (#A1948)	Yes (Import/Export) AES-KW (KTS) (SP 800-38F) (#A1948) AES-KWP (KTS) (SP 800-38F) (#A1948) AES-GCM (KTS) (SP 800-38D) (#A1948) KTS-IFC (KTS) (SP 800-56Br2) (#A2393) KTS-IFC (KTS) (SP 800-56Br2) (#A1948)	SSP generation/ Key transport	In memory (plaintext)	D, S	Triple-DES or AES general purpose session keys.
HMAC keys (user keys)	112-256 bit	HMAC-SHA2-256 (FIPS 198-1) (#A1947) HMAC-SHA2-384 (FIPS 198-1) (#A1947) HMAC-SHA2-512 (FIPS 198-1) (#A1947) HMAC-SHA-1 (FIPS 198-1) (#A1948) HMAC-SHA2-256 (FIPS 198-1) (#A1948) HMAC-SHA2-384 (FIPS 198-1) (#A1948) HMAC-SHA2-512 (FIPS 198-1) (#A1948)	CKG SP 800-133Rev2 Section 6.1 Direct symmetric key generation using unmodified DRBG output OUNTER DRBG (SP800-90Ar1) (#A1948)	Yes (Import/Export) AES-KW (KTS) (SP 800-38F) (#A1948) AES-GCM (KTS) (SP 800-38D) (#A1948) KTS-IFC (KTS) (SP 800-56Br2) (#A2393) KTS-IFC (KTS) (SP 800-56Br2) (#A1948)	SSP generation/ Key transport	In memory (plaintext)/ eMMC flash (encrypted by PMEK, AES-CBC #A1948)	D S	HMAC general purpose keys (minimum key size of 160 bits).
HMAC session keys (user keys)	112-256 bit	HMAC-SHA2-256 (FIPS 198-1) (#A1947) HMAC-SHA2-384 (FIPS 198-1) (#A1947) HMAC-SHA2-512 (FIPS 198-1) (#A1947)	CKG SP 800-133Rev2 Section 6.1 Direct symmetric key generation using unmodified DRBG output OUNTER DRBG (SP800-90Ar1) (#A2393)	Yes (Import/Export) AES-KW (KTS) (SP 800-38F) (#A1948) AES-GCM (KTS) (SP 800-38D) (#A1948)	SSP generation/ Key transport	In memory (plaintext)	D S	HMAC session general purpose keys (minimum key size of 160 bits

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import/ Export	Establish-ment	Storage	Zeroiza-tion	Use & Related Keys
		HMAC-SHA-1 (FIPS 198-1) (#A1948) HMAC-SHA2-256 (FIPS 198-1) (#A1948) HMAC-SHA2-384 (FIPS 198-1) (#A1948) HMAC-SHA2-512 (FIPS 198-1) (#A1948)		KTS-IFC (KTS) (SP 800-56Br2) (#A2393) KTS-IFC (KTS) (SP 800-56Br2) (#A1948)				
TLS session (E2E) ECDH key	112-256 bit	KAS-ECC-SSC SP800-56Ar3 (#A1947) KAS TLS (SP 800-56Ar3)	Yes (CKG SP 800-133r2 using unmodified output of DRBG) DRBG SP 800-90Ar1 (#A1947) ECDSA KEYGEN (FIPS186-4) (#A1948)	No	SSP Generation	In memory (plaintext)	E	Used for key agreement as part of TLS-1.2 handshake protocol.
TLS session symmetric key set	112-256 bit	AES-GCM (SP 800-38D) (#A1947)	KDF TLS (SP800-135r1) (#A1947)	No	Key Derivation	In memory (plaintext)	E	AES 128, 192, 256, or Triple-DES keys used for encrypting TLS sessions.
TLS pre-master secret	112-256 bit	KDF TLS (CVL) (SP 800-135r1) (#A1947)	No	No	Yes KAS-ECC-SSC SP800-56Ar3 (#A1947) RSA Decryption Primitive (SP 800-56Br2) (CVL) (#A1947)	In memory (plaintext)	E	pre-master secret, used to derive master secret.
TLS master secret	384-bit	KDF TLS (CVL) (SP 800-135r1) (#A1947)	No	No	Yes CVL SP 800-135r1 (#A1947)	In memory (plaintext)	E	TLS master secret of size 384-bit used to derive session keys
Manufacturer Firmware Integrity Check Keys	112-bit	RSA SigVer (FIPS186-4) (#A1948)	No	No	Pre-loading of a key	eMMC flash (plaintext)	VZ	RSA 2048-bit public keys used to check the integrity of the SW images booted. The SW image is signed by the manufacturer using ECDSA private key. **Note**: This is not an SSP but is included for completeness of the parameters used by the module
Manufacturer Firmware Update Validation Key (MFUVK)	112-bit	RSA SigVer (FIPS186-4) (#A1948)	No	No	Pre-loading of a key	eMMC flash (plaintext)	NA	RSA 2048-bit public key used to authenticate new FW images uploaded into the module. The FW image is signed by the manufacturer using an RSA private key and the signature is verified before upgrading to the new image using the public key.
Manufacturer License	112-bit	RSA SigVer (FIPS186-4)	No	No	Pre-loading	eMMC flash	NA	RSA 2048-bit public key used to authenticate the manufacturer

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import/ Export	Establish-ment	Storage	Zeroiza-tion	Use & Related Keys
Validation Key (MLVK)		(#A1948)			of a key	(plaintext)		role.
Manufacturer Authentication Root Cert. (MARC)	150-bit	RSA SigVer (FIPS186-4) (#A1948)	No	No	Pre-loading of a key	eMMC flash (plaintext)	VZ	RSA 4096-bit public key certificate used to issue FMAC certificates.
HSM FIPS Master Authentication Certificate (FMAC)	112-bit	RSA SigVer (FIPS186-4) (#A1948)	No	No	Pre-loading of a key	eMMC flash (plaintext)	VZ	RSA 4096-bit public key certificate of FMAK used to identify the HSM FIPS operating mode.
Manufacturer Authentication Root ECC Cert. (MAREC)	256-bit	ECDSA KeyVer (FIPS186-4) (#A1948) ECDSA SigVer (FIPS186-4) (#A1948) ECDSA SigGen (FIPS186-4) (#A1948)	No	No	Pre-loading of a key	eMMC flash (plaintext)	VZ	ECC P521 public key certificate used to issue FMAEC certificates.
HSM FIPS Master Authentication ECC Certificate (FMAEC)	256-bit	ECDSA KeyVer (FIPS186-4) (#A1948) ECDSA SigVer (FIPS186-4) (#A1948) ECDSA SigGen (FIPS186-4) (#A1948)	No	No	Pre-loading of a key	eMMC flash (plaintext)	VZ	ECC P521 public key certificate of FMAEK used to identify the HSM FIPS operating mode.
HSM Endorsement ECC Certificate (HSMEKC)	128-bit	ECDSA KeyVer (FIPS186-4) (#A1948) ECDSA SigVer (FIPS186-4) (#A1948) ECDSA SigGen (FIPS186-4) (#A1948)	No	No	Pre-loading of a key	eMMC flash (plaintext)	VZ	ECC P256 public key certificate of HSMEK used to identify the each HSM independently
HSM/Adapter Owner Trust Anchor Certificate (AOTAC)	112-bit	RSA SigVer (FIPS186-4) (#A1948)	No	No	SSP Entry	eMMC flash (plaintext)	MFZ	RSA 2048-bit public key certificate used as trust anchor of MCO.
HSM/Adapter Owner Authentication Certificate (AOAC)	112-bit	RSA SigVer (FIPS186-4) (#A1948)	No	No	SSP Entry	eMMC flash (plaintext)	MFZ	RSA 2048-bit public key certificate of FMAK used to identify the HSM owner.
Partition Authentication Certificate (PAC)	112-bit	RSA SigVer (FIPS186-4) (#A1948)	CKG SP 800-133Rev2 Section 5.2 Asymmetric key establishment, key generation using unmodified DRBG output RSA KEYGEN (FIPS186-4) (#A1948)	No	SSP Entry/ SSP generation	eMMC flash (plaintext)	PD	RSA 2048-bit public key certificate of PAK used to identify the partition.
Partition Owner Trust Anchor Certificate (POTAC)	112-bit	RSA SigVer (FIPS186-4) (#A1948)	No	No	SSP Entry	eMMC flash (plaintext)	PFZ	RSA 2048-bit public key certificate used as trust anchor of PCO.
Partition Owner Authentication Certificate (POAC)	112-bit	RSA SigVer (FIPS186-4) (#A1948)	No	No	SSP Entry	eMMC flash (plaintext)	PFZ	RSA 2048-bit public key certificate of PAK used to identify the partition owner.

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import/ Export	Establish-ment	Storage	Zeroiza-tion	Use & Related Keys
Partition Cloning Initiator Public Key	256-bit	ECDSA SigVer (FIPS186-4) (#A1948) KAS-ECC SP800-56Ar3 (#A1948)	No	No	SSP Entry	In memory (plaintext)	E	ECC 521-bit ephemeral public key used in SP 800-56Ar3 C (2,0, ECC DH) key agreement or RSA 2048- bit ephemeral public key used in SP 800-56Br2 KAS2 - bilateral - confirmation key agreement to generate shared secret Z.
E2E Client Authentication Public key (CAPubK)	112-256 Bit	RSA SigVer (FIPS186-4) (#A1948) ECDSA SigVer (FIPS186-4) (#A1948)	No	No	SSP Entry	eMMC flash (Plaintext)	PZ	RSA or EC public key of approved modulus or curveId to allow E2E/ TLS client authentication in E2E/ TLS handshake.
Adapter Owner Attestation Public key (AOAPubK)	112-256 Bit	RSA SigVer (FIPS186-4) (#A1948) ECDSA SigVer (FIPS186-4) (#A1948)	No	No	SSP Entry	eMMC flash (Plaintext)	PZ	RSA or EC public key of approved modulus or curveId to allow HSM/ Adapter Owner to authenticated with signature verification with this public key and perform FW image update
Partition Cloning Responder Public Key	256-bit	ECDSA SigVer (FIPS186-4) (#A1948) KAS-ECC SP800-56Ar3 (#A1948) KAS-IFC HKDF (SP800-56Br2) KAS-IFC OneStep (SP800-56Br2) KAS-IFC TwoStep (SP800-56Br2)	No	No	SSP Entry	In memory (plaintext)	E	ECC 521-bit ephemeral public key used in SP 800-56Ar3 C (2, 0, ECC DH) key agreement or RSA 2048- bit ephemeral public key used in SP 800-56Br2 KAS2 - bilateral - confirmation key agreement to generate shared secret Z.
Host PswdEncKeyPublic Key	112-bit	RSA SigVer (FIPS186-4) (#A1948) KAS-IFC HKDF (SP800-56Br2) KAS-IFC OneStep (SP800-56Br2) KAS-IFC TwoStep (SP800-56Br2)	No	No	SSP Entry	In memory (plaintext)	E	RSA 2048-bit public key loaded by the host used for SP 800-56Br2 key agreement to generate PswdEncKey.
Two-Factor Authentication Public Key (2FAMofNPubK)	112-bit	RSA SigVer (FIPS186-4) (#A1948)	No	No	SSP Entry	eMMC flash (Encrypted by PMEK, AES-CBC #A1948)	PZ	RSA 2048-bit public key used to verify signature on encrypted passwords during user creation and login and/or to verify signatures on MofN authentication tokens.
User Public Keys (User Keys)	112-256 bit	ECDSA SigVer (FIPS186-4) (#A1948) RSA Decryption Primitive (SP 800-56Br2) (CVL) (#A1948) RSA Signature Primitive (CVL) (FIPS 186-4) RSA SigGen (FIPS186-4) (#A1948) RSA SigVer (FIPS186-4)	CKG SP 800-133Rev2 Section 5.2 Asymmetric key establishment, key generation using unmodified DRBG output. ECDSA KEYGEN (FIPS186-4) (#A2393) RSA KEYGEN (FIPS186-4) (#A2393)	Yes (Import Plaintext)	SSP entry/ SSP generation	eMMC flash (Encrypted by PMEK, AES-CBC #A1948)	D	RSA/ECDSA/ECDH public keys.

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import/ Export	Establish-ment	Storage	Zeroiza-tion	Use & Related Keys
		(#A1948) ECDSA SigGen (FIPS186-4) (CVL) (#A1947) KAS-ECC-SSC SP800-56Ar3 (#A1947) RSA Decryption Primitive (SP 800-56Br2) (CVL) (#A1947) KTS-IFC (SP 800-56Br2) (#A2393) KTS-IFC (KTS) (SP 800-56Br2) (#A2393) KAS-ECC-SSC SP800-56Ar3 (#A1948) KDF ANS 9.63 (CVL) (SP 800-135r1) (#A1948) KAS KDA HKDF (SP 800-56Ar3) KAS KDA ONESTEP (SP 800-56Ar3) KAS KDA TWOSTEP (SP 800-56Ar3) KAS ANS 9.63 (SP 800-56Ar3) KTS-IFC (KTS) (SP 800-56Br2) (#A1948)					

Key/SSP Name/ Type	Strength	Security Function and Cert. Number	Generation	Import/ Export	Establish-ment	Storage	Zeroiza-tion	Use & Related Keys
User Public Session Keys (User Keys)	112-256 bit	ECDSA SigVer (FIPS186-4) (#A1948) RSA Decryption Primitive (SP 800-56Br2) (CVL) (#A1948) RSA Signature Primitive (CVL)(#A1947) (FIPS 186-4) RSA SigGen (FIPS186-4) (#A1948) RSA SigVer (FIPS186-4) (#A1948) ECDSA SigGen (FIPS186-4) (CVL) (#A1947) KAS-ECC-SSC SP800-56Ar3 (#A1947) RSA Decryption Primitive (SP 800-56Br2) (CVL) (#A1947) KTS-IFC (SP 800-56Br2) (#A2393) KTS-IFC (KTS) (SP 800-56Br2) (#A2393) KAS KDA HKDF (SP 800-56Ar3) KAS KDA ONESTEP (SP 800-56Ar3) KAS KDA TWOSTEP (SP 800-56Ar3) KAS ANS 9.63 (SP 800-56Ar3) KTS-IFC (KTS) (SP 800-56Br2) (#A1948)	CKG SP 800-133Rev2 Section 5.2 Asymmetric key establishment, key generation using unmodified DRBG output. ECDSA KEYGEN (FIPS186-4) (#A2393) RSA KEYGEN (FIPS186-4) (#A2393)	No	SSP Entry/SSP Generation	In Memory (Plaintext)	D S	RSA/ECDSA/ECDH public session keys.

Table 16 Non-Deterministic Random Number Generation Specification

Entropy sources	Minimum number of bits of entropy	Details
OCTEON HW RBG	Gathers 256-bit security strength of entropy (which is 1584 bytes).	The OCTEON TX2 HW unit generates random bits from the 8-free running oscillators from a total of 128-free running oscillators. The generated random bits are already run through HW-level health tests (APT and RCT). OCTEON TX2 produces 2.673 bits of minimum entropy in each 16 bytes of output of its HW RBG.

9.2 Definition of Session Keys

The cryptographic module supports the generation/import/export of user keys that are bound to a session and are termed as session keys. The following points apply to session keys:

- Session keys are stored in RAM and are lost across reboots.
- Session key access is restricted to the application in which it is created.
- PCU can share the session keys with other users so that other sessions can use it.
- Every session in an application will have access to the keys created by every other session in the same application.
- When a session is closed, the session keys created by that session are destroyed.
- If a session key is shared, then it will be deleted only after closing all the sessions sharing the key.

10 Self-Tests

This section documents the security rules enforced by the cryptographic module to implement the self-test requirements of this FIPS 140-3 Level-3 module.

The module always executes the self-tests without operator intervention regardless of approved or non-approved mode or any other configuration.

Failure of any of the self-tests causes the module to go into an error state. If the failure happens during periodic execution of Cryptographic Algorithm Self-Tests (CASTs), then module will reject all future commands received.

The module need to be reset to recover from the situation. Data output except for status log messages is inhibited during self-tests, zeroization, and error states. Status information does not contain CSPs or sensitive data. The conditional cryptographic algorithm self-tests (CASTs) run periodically.

The periodicity is configurable by the MCO and by default runs every 24 hours. The execution of CASTs causes a momentary (less than a second) service interruption.

The voltage monitoring happens continuously by the module which samples the voltage rails for every 400 micro-seconds.

The temperature monitoring happens continuously by the module for every 30 seconds.

Module performs the pre-operational Firmware integrity tests (#A1946) RSA 2048-bit SHA-256 signature verification every 24 hours.

The cryptographic module performs the pre-operational and conditional self-tests:

- Pre-operational software/firmware integrity test
  - CRC-32 integrity test
  - Firmware integrity tests (#A1946) RSA 2048-bit SHA-256 signature verification.
- Pre-operational bypass test (7.10.2.3)
  - None
- Pre-operational critical functions tests: The module runs the following critical functions tests, which are required to ensure the correct functioning of the device.
  - Temperature monitor test
  - Voltage monitor test
- Conditional self-tests
  - Conditional cryptographic algorithm test: The operator is capable of commanding the module to perform the power-up self-test by cycling power or resetting the module. Power-up self-tests do not require any operator action.
    - CPT Library:
      - AES-CBC (SP 800-38A) Decrypt KAT (#A1947, 128-bit key)
      - AES-CBC (SP 800-38A) Encrypt KAT (#A1947, 128-bit key)
      - AES-CCM (SP 800-38C) Decrypt KAT (#A1947, AES 128-bit key)
      - AES-CCM (SP 800-38C) Encrypt KAT (#A1947, AES 128-bit key)

- AES-CMAC (SP 800-38B) Sign KAT (#A1947, 128-bit key)
- AES-CMAC (SP 800-38B) Verify KAT (#A1947, 128-bit key)
- AES-GCM (SP 800-38D) Decrypt KAT (#A1947, 128-bit key)
- AES-GCM (SP 800-38D) Encrypt KAT (#A1947, 128-bit key)
- ECDSA SigGen (FIPS186-4) (CVL) KAT (#A1947, P-256 using SHA-1, SHA-256, SHA-384, SHA-512)
- Hash DRBG (SP 800-90Ar1) (instantiate, generate, reseed) KAT (#A1947, SHA512)
- KAS-ECC-SSC SP800-56Ar3 KAT (#A1947, P-256)
- *KDF SP800-108 CMAC in Counter mode KAT (#A1947, AES 128-bit key)
- *KDF SP800-108 HMAC in Counter mode KAT (#A1947, HMAC-SHA-256)
- KDF TLS (CVL) (SP 800-135r1) TLSv1.2 KDF KAT (#A1947, HMAC-SHA-256)
- RSA (SP 800-56Br2) Encrypt KAT (2048-bit)
- RSA Decryption Primitive (SP 800-56Br2) (CVL) KAT (#A1947, 2048-bit)
- RSA Signature Primitive (CVL) (FIPS 186-4) KAT (#A1947, 2048-bit)
- TDES-CBC (SP 800-38A) Decrypt KAT (#A1947, Triple-DES)
- TDES-CBC (SP 800-38A) Encrypt KAT (#A1947, Triple-DES)

– LS2_SW_Crypto Library:
  - AES-CBC (SP 800-38A) Decrypt KAT (#A1948, 128-bit key)
  - AES-CBC (SP 800-38A) Encrypt KAT (#A1948, 128-bit key)
  - AES-CMAC (SP 800-38B) Sign KAT (#A1948, AES 128-bit)
  - AES-CMAC (SP 800-38B) Verify KAT (#A1948, AES 128-bit)
  - *AES-GCM (SP 800-38D) Encrypt KAT (#A1948, AES 128-bit)
  - AES-KW (SP 800-38F) Unwrap KAT (#A1948, AES 128-bit)
  - AES-KW (SP 800-38F) Wrap KAT (#A1948, AES 128-bit)
  - Counter DRBG (SP 800-90Ar1) (instantiate/generate/reseed) KAT (#A1948, AES 256-bit)
  - *ECDSA KeyGen (FIPS186-4) KAT (#A1948, P-256)
  - ECDSA KeyVer (FIPS186-4) KAT (#A1948, P-256)
  - ECDSA SigGen (FIPS186-4) KAT (#A1948, P-256 with SHA-1, SHA-256, SHA-384, SHA-512)
  - ECDSA SigVer (FIPS186-4) KAT (#A1948, P-256 with SHA-1, SHA-256, SHA-384, SHA-512)
  - HMAC-SHA-1 (FIPS-198-1) KAT (#A1948, HMAC-SHA-1)
  - HMAC-SHA2-256 (FIPS 198-1) KAT (#A1948, HMAC-SHA-256)
  - HMAC-SHA2-512 (FIPS 198-1) KAT (#A1948, HMAC-SHA-512)
  - KAS-ECC SP800-56Ar3 KAT (#A1948, P-521 and HMAC-SHA-512)
  - KAS-ECC-SSC SP800-56Ar3 KAT (#A1948, P-256)
  - KAS-IFC-SSC (SP 800-56Br2) KAT (#A1948, RSA 2048-bit)
  - KDA OneStep SP800-56Cr1 KAT (#A1948, SHA-224)
  - KDA TwoStep SP800-56Cr1 KAT (#A1948, SHA-224)
  - KDF ANS 9.63 (SP 800-135r1) KAT (#A1948, SHA-256)
  - KDF SP800-108 CMAC in Counter mode KAT (#A1948, AES 128-bit)
  - KDF SP800-108 HMAC in Counter mode KAT (#A1948, HMAC-SHA-256)
  - *KDF TLS (SP 800-135r1) TLSv1.2 KDF KAT (#A1948, HMAC-SHA-256)
  - PBKDF (SP 800-132) KAT (#A1948, HMAC-SHA-256)
  - RSA (SP 800-56Br2) Encrypt KAT (2048-bit)
  - RSA Decryption Primitive (SP 800-56Br2) (CVL) KAT (#A1948, 2048-bit)
  - RSA SigGen (FIPS186-4) KAT (#A1948, 2048-bit)
  - RSA SigGen (FIPS186-4) KAT (#A1948, 4096-bit)
  - RSA SigVer (FIPS186-4) KAT (#A1948, 2048-bit)
  - RSA SigVer (FIPS186-4) KAT (#A1948, 4096-bit)

    - *RSA Signature Primitive (CVL) (FIPS 186-4) KAT (#A1948, 2048-bit)
    - TDES-CMAC (SP 800-38B) Sign KAT (#A1948)
    - TDES-CMAC (SP 800-38B) Verify KAT (#A1948)
    - TDES-KW (SP 800-38F) Unwrap KAT (#A1948)
  - LS2_UBOOT Library:
    - RSA SigVer (FIPS186-4) KAT (#A1946, 2048-bit with SHA-256)
  - OCTEON HW RBG:
    - SP 800-90B startup health tests (RCT and APT)
    - SP 800-90B (HW RBG) continuous health tests (RCT and APT)
  - DRBG, SP 800-90A Health Tests (instantiate, generate, reseed) (#A1948, AES CTR 256-bit), (#A1947, Hash SHA512)
  - SP 800-56A rev3 required assurances (All SP 800-56Ar3 implementations).
  - SP 800-56Brev2 required assurances (All SP 800-56Br2 implementations).

Note: CN_INVOKE_FIPS" will execute all the listed CASTs.

- Conditional pair-wise consistency test
  - ECDSA/ECDH Pairwise Consistency Test (ECDSA FIPS 186-4 & KAS-ECC-SSC) at the time of key generation and import
  - RSA Pairwise Consistency Test (RSA FIPS 186-4 & SP 800-56Br2) at the time of key generation and import
- Conditional software/firmware load test (7.10.3.4)
  - Firmware load test (RSA Signature Verification) – RSA 2048-SHA-512 (#A1948)
- Conditional manual entry test (7.10.3.5)
  - N/A
- Conditional bypass test (7.10.3.6)
  - N/A
- Conditional critical functions test (7.10.3.7)
  - Temperature monitor test
  - Voltage monitor test
  - PKCS Sign and Verify Mod exp CRT with private key and verify with public key at every Sign Operation
- Periodic Conditional self-tests
  - Module performs periodic self-tests CASTs every 24 hours or as configured by MCO.
  - Temperature monitor test: every 30 Seconds
  - Voltage monitor test: every 400 micro seconds

Note: Algorithms preceded with * marks that the algorithm is only utilized for self-tests.

11 Life-Cycle Assurance

11.1 Secure Installation, Initialization, Startup, and Operation of the Module

Refer to the sections related to HSM installation and configuration in the user guide (LS2-HSM-NFBE-Driver-SDK- UserGuide; version 1.0 or later).

Before installing the HSM, the customer verifies the following:

- The ElectroStatic Discharge (ESD) bag in which the HSM is placed in the shipping container is sealed and has not been tampered with.
- The part number and other information included in the label on the HSM matches the label on the shipping bag.

- There is no evidence of physical tampering on the HSM itself.

After this is verified, the customer can physically insert the HSM into the PCIe on the host server.

The host must meet the following requirements:

- Support low-profile PCIe Gen 3/Gen 4 (x4 or x8 slot)
- SR-IOV support enabled.

After the HSM is physically installed, the LS2 driver and utilities that communicate with the HSM are installed on the host using standard Linux/Operating system tools. The user must be logged in the host as root/Administrator to perform the installation.

The HSM owner then completes the following steps to claim ownership of the HSM and enable the approved mode:

1. Loads the driver (command: insmod <driver.ko>).
2. Invokes Cfm2Master Utility and logs in as default crypto officer (CO) and initializes the HSM.

For example:

```
Command: initHSM -p <CO password> -sO <CO user name> -fips_state [2|3]
```

As part of initializing the HSM:

- The Master Crypto Officer is created with username/password (see Table 8 and Table 9 for a description of this role, authentication requirements, and service access).
- The fips_state flag is set on the HSM (non-Approved, Approved with single- or dual-factor authentication, or Approved with dual-factor authentication required).

As a final step, the HSM owner claims the HSM by loading the adapter owner certificates (AOTAC and AOAC) on the HSM and import the HSM owner fixed backup key (OKBK). These steps are taken by the MCO using Cfm2MasterUtil.

Example command syntax:

```
Command: storeCert -s <cert-type> -f <Adapter Owner Trust Anchor Cert (AOTAC)>.crt
Command: storeCert -s <cert-type> -f <Adapter Owner Auth Cert (AOAC)>.crt
Command: storeMCOFixedKey -f <path>/<OKBK file>
```

11.2 Maintenance Requirements

N/A

11.3 Administrative and Non-Administrative Guidance

The specific tasks that can be performed by users on the HSM are strictly limited by their user role (see Table 8 for details).

A user of the module can query the module's device information, operational parameters, operating mode by invoking the command getHSMInfo from the Cfm2MasterUtil. "FIPS state" member of the output will indicate the module to be in one of the following modes:

- Approved mode with 1FA ("2")
- Approved mode with 2FA ("3")
- Non-Approved mode ("0")
- Zeroized ("1")

Example command syntax:

```
Command: getHSMInfo
```

The following compliance-specific conditions are applicable to the HSM module:

- Crypto Officer must maintain control of the module while the zeroization process is executing.
- There are no restrictions on which keys or CSPs are zeroized by the zeroization service.
- The module does not support a maintenance interface or role.
- The module does not support bypass capabilities.

- The module does not support manual key entry.
- The module has no CSP feedback to operators.
- The module does not enter or output plaintext CSPs.
- The module does not output intermediate key values part of any operation.
- The module is delivered to the users using the following secure distribution mechanism.
- The module is attached with a specific part number and serial number labels. And, kept in a tamper evident ESD bag with the same labels.
- Then the ESD bags are put inside a shipping box and delivered to customers using any preferred shipping carriers.
- Optionally the modules are locked to a given customer using the AOTAC cert during the manufacturing process and is locked to not run any other commands unless AOAC loaded as per the steps mentioned above.

HSM Zeroization

The instances of zeroization mentioned below are executed through the utilities.

There are different types of zeroization which can be executed through utilities:

- zeroizeHSM --> example command: Cfm2MasterUtil singlecmd zeroizeHSM.
  This zeroizeHSM will delete all the Partition which in turn will zeroize all the partition CSPs, including cleanup all the temporary SSPs.
  - This maps to CN_ZEROIZE service.
- zeroizeHSM along with option (-factory_reset) can be used by operator to zeroize all non-vendor specific CSSPs.
  - This maps to CN_ZEROIZE service with –factory_reset option
- VENDOR zeroizeHSM can be used to zeroize all SSPs.
  To operate a VENDOR zeroizeHSM the operator must log-in as Crypto Officer with its credentials. Without the credentials of the crypto officer, the vendor zeroize can't be executed.
  - This maps to CN_VENDOR_ZEROIZE service.

Notes:

- Temporary SSPs (like in this case: session keys and Integrity test values) are forcefully memset to 0 during session close, application close, partition deletion and zeroization of the partition or the HSM.
- Reboot is a power cycle operation which will lead to the zeroization of temporary SSPs like session Keys.
- Zeroization of a partition or HSM execution can take a few minutes to complete. The zeroization request command is blocked until the execution is completed only after zeroization of required CSPs is completed. User is notified about delay with a notification log as depicted below. The Operator must remain in control of the module while the zeroization process is executing.

  For example: (the below output is executed through Marvell provided driver utilities):

```
Cfm2MasterUtil singlecmd zeroizeHSM
 Version info, Driver Version: 10.02.11.01, SDK API Version: 10.02.11.01

 Cfm2AppInitWithExtNonce () returned app id : 000e0000

 Cfm2OpenSession2() returned 0x00 : HSM Return: SUCCESS
 Command: zeroizeHSM

 Successful zeroization of HSM will reboot the HSM and
 Host-HSM handshake will be re-done.
 Please wait, this may take few minutes.

 Cfm2ZeroizeHSM returned: 0x00 : HSM Return: SUCCESS

 Current FIPS mode is: ffffffff
```

Note: Unsigned `ffffffff` indicates zeroized, which is –1.

- DRBG context : On zeroizeHSM command DRBG related contexts are reset. This means that the internal state of the context with respect to DRBG become unusable.
- In case the power is lost unexpectedly and if the TDES keys are permanent keys, then the module can continue to use the TDES keys for encryption after the module restart as well, until the encryption limit is reached.
- On a routine basis, the MCO can verify that the HSM is correctly operating in approved mode by providing MCO credentials and invoking the command “fipsTest” from the Cfm2MasterUtil utility. The fipsTest utility invokes CN_INVOKE_FIPS service to perform the CAST.

  Example command syntax:

  ```
  Command: fipsTest
  ```

- The following compliance-specific conditions are applicable to the HSM module:
  - There are no restrictions on which keys or CSPs are zeroized by the vendor zeroization service (CN_VENDOR_ZEROIZE).
  - The module does not support a maintenance interface or role.
  - The module does not support bypass capabilities.
  - The module does not support manual key entry.
  - The module does not enter or output plaintext CSPs.
  - The module has no CSP feedback to operators. The module does not output intermediate key values part of any operation.
  - The cryptographic module clears previous authentications on power cycle. The module does not let access SSPs between approved/non-approved and requires zeroization of the HSM/partition.
  - When the module has not been placed in a valid role, the operator shall not have access to any cryptographic services.

11.4 LED Error Pattern for FIPS Failure

On successful completion of the FIPS tests, the DA Green LED remains in the “ON” state. If any other LED remains in the permanent glow, the card’s is in ERROR state.

Any of these fatal errors will cause the module to reject all future commands received. Resetting the module is required to recover from the situation.

Bootloader-level self-test failures will reset the board automatically. There is no separate LED indication for this error.

All these fatal errors are shown on UART when the issue is observed. In addition, the latest errors are reported on UART on bootup. Because the logs are maintained within the module’s flash memory, which has no direct access to external users/ applications, they are not vulnerable to any tampering.

Table 17 LED Flash Pattern for Errors

FIPS Test	Red	Green	Blue	Pattern
Any FIPS self-test failure	Y			On
Hardware RBG for failure			Y	On
Pairwise consistency			Y	On
Boot success and healthy state		Y		On

11.5 User Guidance

AES GCM IV restoration upon Power Cycle of Module:

If the module’s power is lost and then restored, the module will establish new sessions, which will generate new IVs. For an E2E (TLS) session, this will result in fresh handshake; new AES-GCM keys and IVs will be established for the new session.

12 Mitigation of Other Attacks

No mitigation of other attacks is implemented by the module.

13 References

1. NIST Key Wrap Specification SP 800-38F, December 2012.
2. NIST Special Publication 800-38D November 2007.
3. NIST Special Publication 800-56A rev3 , April 2018.
4. NIST Special Publication 800-56B rev2, March 2019.
5. NIST Special Publication 800-56C rev2, August 2020.
6. NIST Special Publication 800-52 rev2, August 2019.
7. NIST Special Publication 800-57 Part-1 rev5, May 2020.
8. FIPS PUB 186-4, Digital Signature Standard (DSS), July 2013.
9. FIPS PUB 140-3, FIPS Publication 140-3 Security Requirements for Cryptographic Modules.
10. NIST Special Publication 800-90A rev1, June 2015.
11. NIST Special Publication 800-90B, January 2018.
12. Implementation Guidance for FIPS PUB 140-3 and the Cryptographic Module Validation Program.
13. NIST Special Publication 800-131Ar2, March 2019.
14. NIST Special Publication 800-133r2 rev2, June 2020.
15. NIST Special Publication 800-108, October 2009.
16. NIST Special Publication 800-135 Revision 1, December 2011.
17. LS2-HSM-NFBE-Driver-SDK-UserGuide-1.0.

14 Definitions and Acronyms

ATF Arm Trusted Firmware

CAST Cryptographic Algorithm Self-Tests

HSM Hardware Security Module

KAS Key Agreement Scheme

KAT Known Answer Test

KBK Key Backup Key

KLK Key Loading Key

MCO Master Crypto Officer

PCO Partition Crypto Officer

PCU Partition Crypto User

SR-IOV Single Root I/O Virtualization

2FA 2-Factor Authentication

Marvell first revolutionized the digital storage industry by moving information at speeds never thought possible. Today, that same breakthrough innovation remains at the heart of the company's storage, networking and connectivity solutions. With leading intellectual property and deep system-level knowledge, Marvell semiconductor solutions continue to transform the enterprise, cloud, automotive, industrial, and consumer markets. For more information, visit www.marvell.com.

Doc. No. LS2-HSM-SP Revised: January 14, 2025