

Hitachi Vantara, Ltd.

Hitachi Embedded Storage Manager Kernel Crypto API Cryptographic Module  
**FIPS 140-3 Non-Proprietary Security Policy**

Version 1.0

# Table of Contents

1 General .....	5
1.1 Overview .....	5
1.2 Security Levels .....	5
2 Cryptographic Module Specification .....	5
2.1 Description .....	5
2.2 Tested and Vendor Affirmed Module Version and Identification .....	7
2.3 Excluded Components .....	8
2.4 Modes of Operation .....	8
2.5 Algorithms .....	8
2.6 Security Function Implementations .....	10
2.7 Algorithm Specific Information .....	22
2.8 RBG and Entropy .....	22
2.9 Key Generation .....	22
2.10 Key Establishment .....	22
2.11 Industry Protocols .....	22
3 Cryptographic Module Interfaces .....	23
3.1 Ports and Interfaces .....	23
4 Roles, Services, and Authentication .....	23
4.1 Authentication Methods .....	23
4.2 Roles .....	23
4.3 Approved Services .....	23
4.4 Non-Approved Services .....	27
4.5 External Software/Firmware Loaded .....	27
5 Software/Firmware Security .....	27
5.1 Integrity Techniques .....	27
5.2 Initiate on Demand .....	28
6 Operational Environment .....	28
6.1 Operational Environment Type and Requirements .....	28
6.2 Configuration Settings and Restrictions .....	28
7 Physical Security .....	28
8 Non-Invasive Security .....	28
9 Sensitive Security Parameters Management .....	28
9.1 Storage Areas .....	28
9.2 SSP Input-Output Methods .....	28
9.3 SSP Zeroization Methods .....	29

9.4 SSPs .....	29
9.5 Transitions.....	31
10 Self-Tests.....	31
10.1 Pre-Operational Self-Tests .....	31
10.2 Conditional Self-Tests.....	32
10.3 Periodic Self-Test Information.....	43
10.4 Error States .....	48
11 Life-Cycle Assurance .....	49
11.1 Installation, Initialization, and Startup Procedures.....	49
11.2 Administrator Guidance .....	49
11.3 Non-Administrator Guidance.....	49
11.4 Design and Rules .....	49
12 Mitigation of Other Attacks .....	50

## List of Tables

Table 1: Security Levels.....	5
Table 2: Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets)....	7
Table 3: Tested Operational Environments - Software, Firmware, Hybrid .....	8
Table 4: Modes List and Description .....	8
Table 5: Approved Algorithms .....	9
Table 6: Non-Approved, Not Allowed Algorithms.....	10
Table 7: Security Function Implementations.....	22
Table 8: Ports and Interfaces .....	23
Table 9: Roles.....	23
Table 10: Approved Services .....	27
Table 11: Non-Approved Services.....	27
Table 12: Storage Areas .....	28
Table 13: SSP Input-Output Methods.....	29
Table 14: SSP Zeroization Methods..	29
Table 15: SSP Table 1 .....	31
Table 16: SSP Table 2 .....	31
Table 17: Pre-Operational Self-Tests .....	31
Table 18: Conditional Self-Tests .....	43
Table 19: Pre-Operational Periodic Information.....	43
Table 20: Conditional Periodic Information.....	48
Table 21: Error States .....	48

## List of Figures

Figure 1: Block Diagram.....	7
------------------------------	---

# 1 General

## 1.1 Overview

This document defines the Security Policy for the Hitachi Embedded Storage Manager Kernel Crypto API Cryptographic Module, hereafter denoted as the module. The module meets FIPS 140-3 overall Level 1 requirements.

## 1.2 Security Levels

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

Table 1: Security Levels

# 2 Cryptographic Module Specification

## 2.1 Description

### Purpose and Use:

The module provides general purpose cryptographic services for the Hitachi Embedded Storage Manager. The module works in kernel space and provides cryptographic services to other kernel functions through C language interfaces and to user space applications through the AF\_ALG socket.

**Module Type:** Software

**Module Embodiment:** MultiChipEmbed

### Cryptographic Boundary:

The cryptographic boundary for the module consists of the static kernel binary, the cryptographic kernel object files, the self-test program, the integrity test program, the Conditional Cryptographic Algorithm Self-Tests (CAST) result check program, the version printout program, and its integrity hash files. The components are enumerated below.

- Static kernel binary:  
`/boot/Image-5.10.212-cip45`
- Cryptographic kernel object files:  
`/lib/modules/5.10.212-cip45/kernel/crypto/*.ko`  
`/lib/modules/5.10.212-cip45/kernel/arch/arm64/crypto/*.ko`
- Self-test program:  
`/usr/local/sbin/pltf-kernel-fips.sh`
- Integrity test utility:  
`/usr/local/sbin/sha256sum-fips`
- Integrity test hash file:  
`/usr/local/sbin/fips-checksums.txt`
- Conditional Cryptographic Algorithm Self-Tests (CAST) result check program:  
`/usr/local/sbin/checkcrypto`
- Version printout program:  
`/usr/local/sbin/showversion-fips`

The green solid line in Figure 1 shows the delimitation of the module's cryptographic boundary. The module is designed to be able to utilize Processor Algorithm Acceleration(PAA) functions, Arm Neon instructions and ARMv8 Crypto Extensions, from the processor and assembly code for AES and SHA operations to accelerate the cryptographic calculations of the module.

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

The tested operational environment hardware for the module is dedicated hardware for the Hitachi Storage System, Storage Management Controller. The enclosure of the Storage Management Controller is TOEPP. The Storage Management Controller implements a processor, Layerscape® 1046A. An operating system, EMLinux®, works on the processor. The module and the Embedded Storage Manager (ESM) applications work within the operating system.

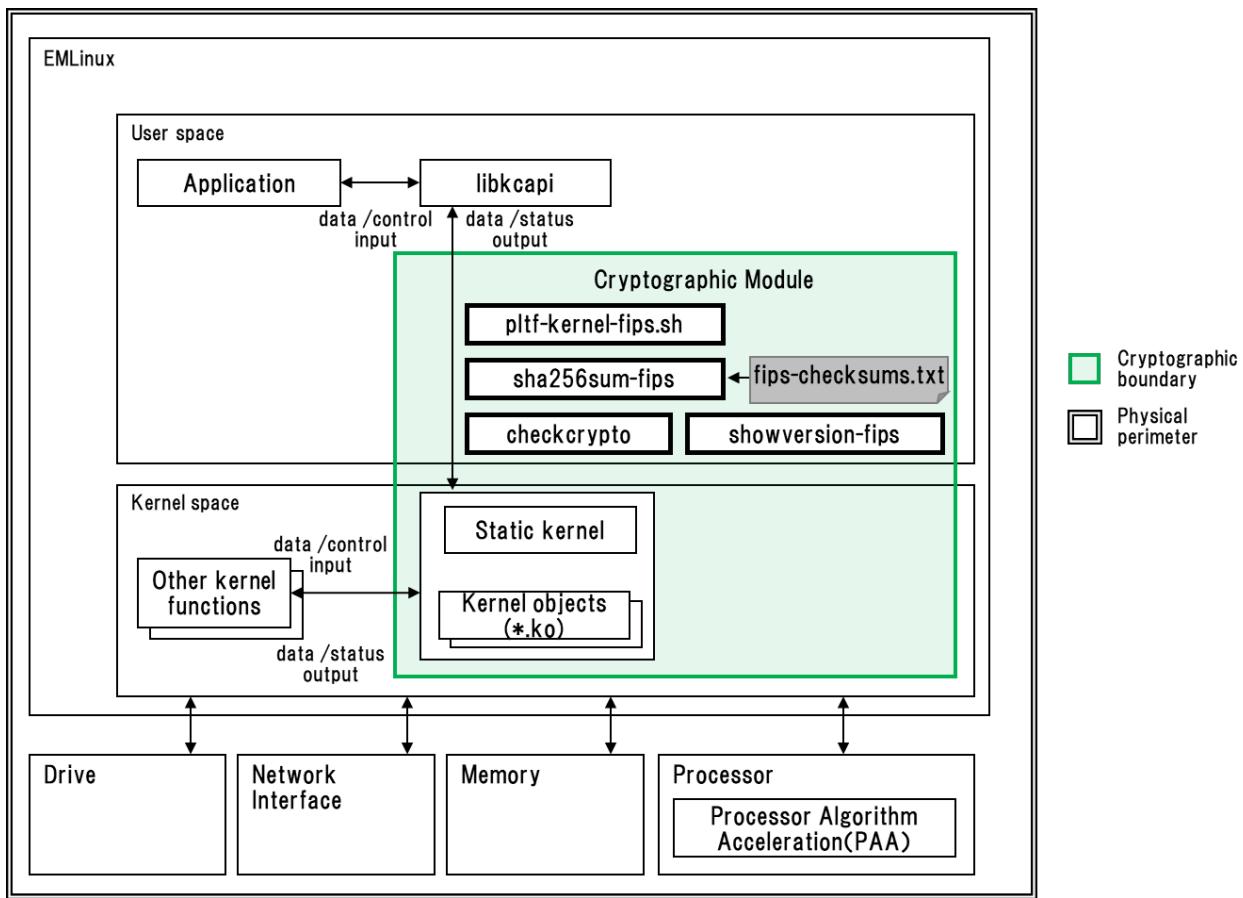


Figure 1: Block Diagram

## 2.2 Tested and Vendor Affirmed Module Version and Identification

### Tested Module Identification – Hardware:

N/A for this module.

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

Package or File Name	Software/ Firmware Version	Features	Integrity Test
Embedded Storage Manager Kernel Crypto API Cryptographic Module	1.1	N/A	SHA2-256

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

### Tested Module Identification – Hybrid Disjoint Hardware:

N/A for this module.

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

<b>Operating System</b>	<b>Hardware Platform</b>	<b>Processors</b>	<b>PAA/PAI</b>	<b>Hypervisor or Host OS</b>	<b>Version(s)</b>
EMLinux 2.9	Storage Management Controller	Layerscape® 1046A	Yes	N/A	1.1
EMLinux 2.9	Storage Management Controller	Layerscape® 1046A	No	N/A	1.1

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

The Layerscape® 1046A processor integrates quad 64-bit Arm® Cortex®-A72 cores.

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

N/A for this module.

### **2.3 Excluded Components**

The module has no excluded components.

### **2.4 Modes of Operation**

#### **Modes List and Description:**

<b>Mode Name</b>	<b>Description</b>	<b>Type</b>	<b>Status Indicator</b>
Approved	Only approved or allowed security functions with sufficient security strength can be used.	Approved	The indicator of the service as defined in Section 4.3.
Non-approved	Only non-approved security functions can be used.	Non-Approved	The indicator of the return value of the indicator function.

Table 4: Modes List and Description

When the operating system starts, the module automatically enters the approved mode of operation. No special API calls or settings are required to place the module in the approved mode of operation.

Once the module is operational, if the use of the non-approved service is started, the module implicitly enters the non-approved mode. When the use of the non-approved service is ended, the module implicitly and immediately enters the approved mode.

### **2.5 Algorithms**

#### **Approved Algorithms:**

<b>Algorithm</b>	<b>CAVP Cert</b>	<b>Properties</b>	<b>Reference</b>
AES-CBC	A5827, A5828, A5829, A5830, A5831, A5834	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A

<b>Algorithm</b>	<b>CAVP Cert</b>	<b>Properties</b>	<b>Reference</b>
AES-CBC-CS3	A5827, A5828, A5829, A5830, A5831, A5833	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CMAC	A5827, A5828, A5829, A5830, A5831, A5833	Direction - Generation, Verification Key Length - 128, 192, 256	SP 800-38B
AES-CTR	A5827, A5828, A5829, A5830, A5831, A5834	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-ECB	A5827, A5828, A5829, A5830, A5831, A5834	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-KW	A5827, A5828, A5830, A5831	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-XTS Testing Revision 2.0	A5827, A5828, A5829, A5830, A5831, A5834	Direction - Decrypt, Encrypt Key Length - 128, 256	SP 800-38E
HMAC-SHA-1	A5830, A5831	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA2-256	A5828, A5830, A5831, A5833	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA2-384	A5828, A5831	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512	A5828, A5831	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA3-256	A5831	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA3-384	A5831	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA3-512	A5831	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
SHA-1	A5830, A5831	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-256	A5828, A5830, A5831, A5832, A5833	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-384	A5828, A5831	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA2-512	A5828, A5831	Message Length - Message Length: 0-65536 Increment 8	FIPS 180-4
SHA3-256	A5831	Message Length - Message Length: 0-65536 Increment 8	FIPS 202
SHA3-384	A5831	Message Length - Message Length: 0-65536 Increment 8	FIPS 202
SHA3-512	A5831	Message Length - Message Length: 0-65536 Increment 8	FIPS 202

Table 5: Approved Algorithms

#### **Vendor-Affirmed Algorithms:**

N/A for this module.

#### **Non-Approved, Allowed Algorithms:**

N/A for this module.

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

N/A for this module.

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

Name	Use and Function
RSA-sigGen (pkcs1pad(rsa-generic,sha256/sha384/sha512))	Used for RSA signature generation.
RSA-sigVer (pkcs1pad(rsa-generic,sha256/sha384/sha512))	Used for RSA signature verification.
RSA-signaturePrimitive (rsa-generic)	Used for RSA signature primitive operation.
RSA-decryptionPrimitive (rsa-generic)	Used for RSA decrypt primitive operation.

Table 6: Non-Approved, Not Allowed Algorithms

## 2.6 Security Function Implementations

Name	Type	Description	Properties	Algorithms
sha1-generic	SHA	Used to generate SHA1 hash value using generic C implementation.		SHA-1: (A5831)
sha1-ce	SHA	Used to generate SHA1 hash value using ARMv8 Crypto Extensions (CE).		SHA-1: (A5830)
sha256-generic	SHA	Used to generate SHA2-256 hash value using generic C implementation.		SHA2-256: (A5831)
sha256-ce	SHA	Used to generate SHA2-256 hash value using ARMv8 Crypto Extensions (CE).		SHA2-256: (A5830)
sha256-arm64	SHA	Used to generate SHA2-256 hash value using assembler.		SHA2-256: (A5828)
sha256-arm64-neon	SHA	Used to generate SHA2-		SHA2-256: (A5833)

Name	Type	Description	Properties	Algorithms
		256 hash value using Arm NEON instructions.		
Integrity Check Utility	SHA	Used to generate SHA2-256 hash value for integrity check of the module.		SHA2-256: (A5832)
sha384-generic	SHA	Used to generate SHA2-384 hash value using generic C implementation.		SHA2-384: (A5831)
sha384-arm64	SHA	Used to generate SHA2-384 hash value using assembler.		SHA2-384: (A5828)
sha512-generic	SHA	Used to generate SHA2-512 hash value using generic C implementation.		SHA2-512: (A5831)
sha512-arm64	SHA	Used to generate SHA2-512 hash value using assembler.		SHA2-512: (A5828)
sha3-256-generic	SHA	Used to generate SHA3-256 hash value using generic C implementation.		SHA3-256: (A5831)
sha3-384-generic	SHA	Used to generate SHA3-384 hash value using generic C implementation.		SHA3-384: (A5831)
sha3-512-generic	SHA	Used to generate SHA3-512 hash value using generic C implementation.		SHA3-512: (A5831)
hmac(sha1-generic)	MAC	Used to generate MAC based on SHA1 function using		HMAC-SHA-1: (A5831)

Name	Type	Description	Properties	Algorithms
		generic C implementation.		
hmac(sha1-ce)	MAC	Used to generate MAC based on SHA1 function using ARMv8 Crypto Extensions (CE).		HMAC-SHA-1: (A5830)
hmac(sha256-generic)	MAC	Used to generate MAC based on SHA2-256 function using generic C implementation.		HMAC-SHA2-256: (A5831)
hmac(sha256-ce)	MAC	Used to generate MAC based on SHA2-256 function using ARMv8 Crypto Extensions (CE).		HMAC-SHA2-256: (A5830)
hmac(sha256-arm64)	MAC	Used to generate MAC based on SHA2-256 function using assembler.		HMAC-SHA2-256: (A5828)
hmac(sha256-arm64-neon)	MAC	Used to generate MAC based on SHA2-256 function using NEON instructions.		HMAC-SHA2-256: (A5833)
hmac(sha384-generic)	MAC	Used to generate MAC based on SHA2-384 function using generic C implementation.		HMAC-SHA2-384: (A5831)
hmac(sha384-arm64)	MAC	Used to generate MAC based on SHA2-384 function using assembler.		HMAC-SHA2-384: (A5828)
hmac(sha512-generic)	MAC	Used to generate MAC based on SHA2-512 function		HMAC-SHA2-512: (A5831)

Name	Type	Description	Properties	Algorithms
		using generic C implementation.		
hmac(sha512-arm64)	MAC	Used to generate MAC based on SHA2-512 function using assembler.		HMAC-SHA2-512: (A5828)
hmac(sha3-256-generic)	MAC	Used to generate MAC based on SHA3-256 function using generic C implementation.		HMAC-SHA3-256: (A5831)
hmac(sha3-384-generic)	MAC	Used to generate MAC based on SHA3-384 function using generic C implementation.		HMAC-SHA3-384: (A5831)
hmac(sha3-512-generic)	MAC	Used to generate MAC based on SHA3-512 function using generic C implementation.		HMAC-SHA3-512: (A5831)
ecb(aes-generic)	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-ECB composed of AES using generic C implementation and ECB mode using generic C implementation.		AES-ECB: (A5831)
ecb(aes-ce)	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-ECB composed of AES using ARMv8 Crypto Extensions (CE) and ECB mode using generic C implementation.		AES-ECB: (A5830)

Name	Type	Description	Properties	Algorithms
ecb-aes-ce	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-ECB composed of AES using ARMv8 Crypto Extensions (CE) and ECB mode using assembler.		AES-ECB: (A5829)
ecb(aes-arm64)	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-ECB composed of AES using assembler and ECB mode using assembler.		AES-ECB: (A5828)
ecb(aes-fixed-time)	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-ECB composed of Fixed time AES using generic C implementation and ECB mode using generic C implementation.		AES-ECB: (A5827)
ecb-aes-neonbs	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-ECB composed of Bit sliced AES using Arm Neon instructions and ECB mode using assembler.		AES-ECB: (A5834)
ctr(aes-generic)	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-CTR composed of AES using generic C implementation		AES-CTR: (A5831)

Name	Type	Description	Properties	Algorithms
		and CTR mode using generic C implementation.		
ctr(aes-ce)	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-CTR composed of AES using ARMv8 Crypto Extensions (CE) and CTR mode using generic C implementation.		AES-CTR: (A5830)
ctr-aes-ce	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-CTR composed of AES using ARMv8 Crypto Extensions (CE) and CTR mode using assembler.		AES-CTR: (A5829)
ctr(aes-arm64)	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-CTR composed of AES using assembler and CTR mode using assembler.		AES-CTR: (A5828)
ctr(aes-fixed-time)	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-CTR composed of Fixed time AES using generic C implementation and CTR mode using generic C implementation.		AES-CTR: (A5827)
ctr-aes-neonbs	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-CTR		AES-CTR: (A5834)

Name	Type	Description	Properties	Algorithms
		composed of Bit sliced AES using Arm Neon instructions and CTR mode using assembler.		
cbc(aes-generic)	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-CBC composed of AES using generic C implementation and CBC mode using generic C implementation.		AES-CBC: (A5831)
cbc(aes-ce)	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-CBC composed of AES using ARMv8 Crypto Extensions (CE) and CBC mode using generic C implementation.		AES-CBC: (A5830)
cbc-aes-ce	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-CTR composed of AES using ARMv8 Crypto Extensions (CE) and CTR mode using assembler.		AES-CBC: (A5829)
cbc(aes-arm64)	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-CBC composed of AES using assembler and CBC mode using assembler.		AES-CBC: (A5828)

Name	Type	Description	Properties	Algorithms
cbc(aes-fixed-time)	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-CBC composed of Fixed time AES using generic C implementation and CBC mode using generic C implementation.		AES-CBC: (A5827)
cbc-aes-neonbs	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-CBC composed of Bit sliced AES using Arm Neon instructions and CBC mode using assembler.		AES-CBC: (A5834)
cts(cbc(aes-generic))	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-CBC-CS3 composed of AES using generic C implementation and CBC-CS3 mode using generic C implementation.		AES-CBC-CS3: (A5831)
cts(cbc(aes-ce))	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-CBC-CS3 composed of AES using ARMv8 Crypto Extensions (CE) and CBC-CS3 mode using generic C implementation.		AES-CBC-CS3: (A5830)
cts-cbc-aes-ce	BC-UnAuth	Used to encrypt/decrypt inputted data		AES-CBC-CS3: (A5829)

Name	Type	Description	Properties	Algorithms
		with AES-CBC-CS3 composed of AES using ARMv8 Crypto Extensions (CE) and CBC-CS3 mode using assembler.		
cts(cbc(aes-arm64))	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-CBC-CS3 composed of AES using assembler and CBC-CS3 mode using assembler.		AES-CBC-CS3: (A5828)
cts(cbc(aes-fixed-time))	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-CBC-CS3 composed of Fixed time AES using generic C implementation and CBC-CS3 mode using generic C implementation.		AES-CBC-CS3: (A5827)
cts-cbc-aes-neon	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-CBC-CS3 composed of AES using Arm Neon instructions and CBC-CS3 mode using assembler.		AES-CBC-CS3: (A5833)
cmac(aes-generic)	MAC	Used to generate MAC based on AES encryption using generic C implementation and CMAC		AES-CMAC: (A5831)

Name	Type	Description	Properties	Algorithms
		mode using generic C implementation.		
cmac(aes-ce)	MAC	Used to generate MAC based on AES encryption using ARMv8 Crypto Extensions (CE) and CMAC mode using generic C implementation.		AES-CMAC: (A5830)
cmac-aes-ce	MAC	Used to generate MAC based on AES encryption using ARMv8 Crypto Extensions (CE) and CMAC mode using assembler.		AES-CMAC: (A5829)
cmac(aes-arm64)	MAC	Used to generate MAC based on AES encryption using assembler and CMAC mode using assembler.		AES-CMAC: (A5828)
cmac(aes-fixed-time)	MAC	Used to generate MAC based on Fixed time AES encryption using generic C implementation and CMAC mode using generic C implementation.		AES-CMAC: (A5827)
cmac-aes-neon	MAC	Used to generate MAC based on AES encryption using Arm Neon instructions and CMAC mode		AES-CMAC: (A5833)

Name	Type	Description	Properties	Algorithms
		using assembler.		
xts(aes-generic)	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-XTS composed of AES using generic C implementation and XTS mode using generic C implementation.		AES-XTS Testing Revision 2.0: (A5831)
xts(aes-ce)	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-XTS composed of AES using ARMv8 Crypto Extensions (CE) and XTS mode using generic C implementation.		AES-XTS Testing Revision 2.0: (A5830)
xts-aes-ce	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-XTS composed of AES using ARMv8 Crypto Extensions (CE) and XTS mode using assembler.		AES-XTS Testing Revision 2.0: (A5829)
xts(aes-arm64)	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-XTS composed of AES using assembler and XTS mode using assembler.		AES-XTS Testing Revision 2.0: (A5828)
xts(aes-fixed-time)	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-XTS composed of		AES-XTS Testing Revision 2.0: (A5827)

Name	Type	Description	Properties	Algorithms
		Fixed time AES using generic C implementation and XTS mode using generic C implementation.		
xts-aes-neonbs	BC-UnAuth	Used to encrypt/decrypt inputted data with AES-XTS composed of Bit sliced AES using Arm Neon instructions and XTS mode using assembler.		AES-XTS Testing Revision 2.0: (A5834)
kw(aes-generic)	BC-Auth	Used to wrap/unwrap inputted key with AES-KW composed of AES using generic C implementation and KW mode using generic C implementation.		AES-KW: (A5831)
kw(aes-ce)	BC-Auth	Used to wrap/unwrap inputted key with AES-KW composed of AES using ARMv8 Crypto Extensions (CE) and KW mode using generic C implementation.		AES-KW: (A5830)
kw(aes-arm64)	BC-Auth	Used to wrap/unwrap inputted key with AES-KW composed of AES using assembler and KW mode using assembler.		AES-KW: (A5828)
kw(aes-fixed-time)	BC-Auth	Used to wrap/unwrap		AES-KW: (A5827)

Name	Type	Description	Properties	Algorithms
		inputted key with AES-KW composed of Fixed time AES using generic C implementation and KW mode using generic C implementation.		

Table 7: Security Function Implementations

## 2.7 Algorithm Specific Information

### AES XTS:

As specified in SP800-38E, the AES algorithm in XTS mode (AES-XTS) was designed for the cryptographic protection of data on storage devices that use fixed length data units. Thus, it can only be used for the disk encryption functionality offered by dm-crypt (i.e., the hard disk encryption scheme). For dm-crypt, the length of a single data unit encrypted with the AES XTS is at most 65,536 bytes (64KiB of data), which does not exceed  $2^{20}$  AES blocks (16MiB of data).

To meet the requirement stated in FIPS140-3 IG C.I, the module has a function that checks if two keys for the AES-XTS are different from each other.

### SHA-1:

The use of SHA-1 is only approved for integrity checks and is not approved if used as part of digital signature generation.

## 2.8 RBG and Entropy

N/A for this module.

N/A for this module.

## 2.9 Key Generation

N/A for this module.

## 2.10 Key Establishment

N/A for this module.

## 2.11 Industry Protocols

N/A for this module.

## 3 Cryptographic Module Interfaces

### 3.1 Ports and Interfaces

Physical Port	Logical Interface(s)	Data That Passes
N/A	Data Input	API input parameters from kernel system calls, AF_ALG type socket.
N/A	Data Output	API output parameters from kernel system calls, AF_ALG type socket.
N/A	Control Input	API function calls, API input parameters for control from kernel system calls, AF_ALG type socket, command line input.
N/A	Status Output	API return codes, AF_ALG type socket, kernel logs, user logs, command line output.

Table 8: Ports and Interfaces

## 4 Roles, Services, and Authentication

### 4.1 Authentication Methods

N/A for this module.

The module does not support authentication for roles.

### 4.2 Roles

Name	Type	Operator Type	Authentication Methods
Cryptographic Officer	Role	CO	None

Table 9: Roles

Cryptographic Officer role is implicitly and always assumed.

### 4.3 Approved Services

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
AES encryption and decryption	AES encryption and decryption.	Approved if the following conditions are met. Condition 1: the return value of	AES keys, Data to encrypt or decrypt	Encrypted data or decrypted data	ecb(aes-generic) ecb(aes-ce) ecb-aes-ce ecb(aes-arm64) ecb(aes-fixed-time) ecb-aes-neonbs	Cryptographic Officer - AES Key: W,E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
		indicator function is 1. Condition 2: the return value of key input is 0.			ctr(aes-generic) ctr(aes-ce) ctr-aes-ce ctr(aes-arm64) ctr(aes-fixed-time) ctr-aes-neonbs cbc(aes-generic) cbc(aes-ce) cbc-aes-ce cbc(aes-arm64) cbc(aes-fixed-time) cbc-aes-neonbs cts(cbc(aes-generic)) cts(cbc(aes-ce)) cts-cbc-aes-ce cts(cbc(aes-arm64)) cts(cbc(aes-fixed-time)) cts-cbc-aes-neon xts(aes-generic) xts(aes-ce) xts-aes-ce xts(aes-arm64) xts(aes-fixed-time) xts-aes-neonbs	
Key wrapping and unwrapping	Key wrapping and unwrapping using AES.	Approved if the following conditions are met. Condition 1: the return value of key input is 1. Condition 2: the return value of key input is 0.	AES key, Key as data to key wrap or	Wrapped key or unwrapped key	kw(aes-generic) kw(aes-ce) kw(aes-arm64) kw(aes-fixed-time)	Cryptographic Officer - AES Key: W,E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
		n 1: the return value of indicator function is 1. Condition 2: the return value of key input is 0.	key unwrap			
Message digest	Message digest generation.	Approved if the return value of indicator function is 1.	Data for digest	Digest value	sha1-generic sha1-ce sha256-generic sha256-ce sha256-arm64 sha256-arm64-neon sha384-generic sha384-arm64 sha512-generic sha512-arm64 sha3-256-generic sha3-384-generic sha3-512-generic	Cryptographic Officer
Message authentication code (HMAC)	Message authentication code generation based on a hash function.	Approved if the return value of indicator function is 1.	HMAC Key, message	MAC	hmac(sha1-generic) hmac(sha1-ce) hmac(sha256-generic) hmac(sha256-ce) hmac(sha256-arm64) hmac(sha256-arm64-neon) hmac(sha38)	Cryptographic Officer - HMAC Key: W,E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
					4-generic) hmac(sha384-arm64) hmac(sha512-generic) hmac(sha512-arm64) hmac(sha3-256-generic) hmac(sha3-384-generic) hmac(sha3-512-generic)	
Message authentication code (CMAC)	Message authentication code generation based on a cipher function.	Approved if the return value of indicator function is 1.	AES Key, message	MAC	cmac(aes-generic) cmac(aes-ce) cmac-aes-ce cmac(aes-arm64) cmac(aes-fixed-time) cmac-aes-neon	Cryptographic Officer - AES Key: W,E
Error detection	Compute an EDC (crc32, crc32c, xxhash64)	None	Data	EDC	None	Cryptographic Officer
Memory copy	Memory copy operation	None	Data	Output	None	Cryptographic Officer
Show Module Information (show version)	Show module ID and version.	None	None	Module ID, module version.	None	Cryptographic Officer
Show Status	Show module status.	None	None	Module status	None	Cryptographic Officer
Zeroise	Zeroise SSPs.	None	None	None	None	Cryptographic Officer - AES Key: Z - HMAC Key: Z
On-demand self test	Perform Integrity	None	None	None	Integrity Check Utility	Cryptographic Officer

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	check and CASTs.					

Table 10: Approved Services

All approved services implemented by the module are listed above. Each service description also describes all usage of SSPs by the service. The access rights to keys and/or SSPs modes shown in the table are defined as:

- 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 executes using the SSP in performing a cryptographic operation.
- Z = Zeroise: The module zeroes the SSP.

The module may have multiple implementations for the same algorithms. When a service is invoked using an algorithm name, the module selects one available implementation based on the specified algorithm name and the priority assigned to each implementation. Specific implementations can also be invoked using the driver names.

#### 4.4 Non-Approved Services

Name	Description	Algorithms	Role
RSA signature generation	Signature generation based on the PKCS#1.	RSA-sigGen (pkcs1pad(rsa-generic,sha256/sha384/sha512))	Cryptographic Officer
RSA signature verification	Signature verification based on the PKCS#1.	RSA-sigVer (pkcs1pad(rsa-generic,sha256/sha384/sha512))	Cryptographic Officer
RSA signature primitive operation	Signature primitive operation based on the RSA algorithm.	RSA-signaturePrimitive (rsa-generic)	Cryptographic Officer
RSA decrypt primitive operation	Decrypt primitive operation based on the RSA algorithm.	RSA-decryptionPrimitive (rsa-generic)	Cryptographic Officer

Table 11: Non-Approved Services

#### 4.5 External Software/Firmware Loaded

N/A

### 5 Software/Firmware Security

#### 5.1 Integrity Techniques

The integrity of the static kernel binary, the cryptographic kernel object files, the self-test program, the integrity test program, the Conditional Cryptographic Algorithm Self-Tests (CAST) result check program and the version printout program are tested by comparing the SHA2-256 digest values calculated at startup with the SHA2-256 digest values calculated and stored in the module during module development.

## 5.2 Initiate on Demand

The integrity tests are performed as part of the pre-operational self-tests. Thus, the integrity tests can be initiated on demand by power cycle or reboot of the operational environment of the module.

# 6 Operational Environment

## 6.1 Operational Environment Type and Requirements

**Type of Operational Environment:** Modifiable

## 6.2 Configuration Settings and Restrictions

The ptrace system call, the debugger gdb and strace shall not be used. In addition, other tracing mechanisms offered by the Linux environment, such as ftrace or systemtap shall not be used.

# 7 Physical Security

N/A. Since the module consists only of software, this section is not applicable.

# 8 Non-Invasive Security

N/A. The module does not implement non-invasive security techniques.

# 9 Sensitive Security Parameters Management

## 9.1 Storage Areas

Storage Area Name	Description	Persistence Type
Memory	A volatile memory on the operational environment	Dynamic

Table 12: Storage Areas

The module does not store SSPs in persistent storage. The SSPs are temporarily stored in process memory when the module is being used.

## 9.2 SSP Input-Output Methods

Name	From	To	Format Type	Distribution Type	Entry Type	SFI or Algorithm
API Input	Memory	Memory	Plaintext	Manual	Electronic	
AF_ALG_type sockets (input)	Memory	Memory	Plaintext	Manual	Electronic	

Table 13: SSP Input-Output Methods

### 9.3 SSP Zeroization Methods

Zeroization Method	Description	Rationale	Operator Initiation
Power cycle	Power cycle of the operational environment	All SSPs of the module are zeroised by Power cycle because all SSPs are on a volatile memory.	Yes
Reboot	Reboot of the operational environment	All SSPs of the module are zeroised by Reboot because all SSPs are on a volatile memory.	Yes

Table 14: SSP Zeroization Methods

Administrators of the module can zeroise all SSPs of the module by cycling power or reboot of the Hitachi Embedded Storage Manager.

### 9.4 SSPs

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
AES Key	AES key used for encryption, decryption, and generating MAC.	128, 192, 256 bits; XTS 128, 256 bits - 128, 192, 256 bits; XTS 128, 256 bits	Symmetric Key - CSP			ecb(aes-generic) ecb(aes-ce) ecb-aes-ce ecb(aes-arm64) ecb(aes-fixed-time) ecb-aes-neonbs ctr(aes-generic) ctr(aes-ce) ctr-aes-ce ctr(aes-arm64) ctr(aes-fixed-time) ctr-aes-neonbs cbc(aes-generic) cbc(aes-ce) cbc-aes-ce

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
						cbc(aes-arm64) cbc(aes-fixed-time) cbc-aes-neonbs cts(cbc(aes-generic)) cts(cbc(aes-ce)) cts-cbc-aes-ce cts(cbc(aes-arm64)) cts(cbc(aes-fixed-time)) cts-cbc-aes-neon cmac(aes-generic) cmac(aes-ce) cmac-aes-ce cmac(aes-arm64) cmac(aes-fixed-time) cmac-aes-neon xts(aes-generic) xts(aes-ce) xts-aes-ce xts(aes-arm64) xts(aes-fixed-time) xts-aes-neonbs kw(aes-generic) kw(aes-ce) kw(aes-arm64) kw(aes-fixed-time)
HMAC Key	HMAC key used for generating MAC	112 - 65,535 bits - 112 - 256 bits	Symmetric Key - CSP			hmac(sha1-generic) hmac(sha1-ce) hmac(sha256-

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
						generic) hmac(sha256-ce) hmac(sha256-arm64) hmac(sha256-arm64-neon) hmac(sha384-generic) hmac(sha384-arm64) hmac(sha512-generic) hmac(sha512-arm64) hmac(sha3-256-generic) hmac(sha3-384-generic) hmac(sha3-512-generic)

Table 15: SSP Table 1

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
AES Key	API Input AF_ALG_type sockets (input)	Memory:Plaintext	During the execution of the service.	Power cycle Reboot	
HMAC Key	API Input AF_ALG_type sockets (input)	Memory:Plaintext	During the execution of the service.	Power cycle Reboot	

Table 16: SSP Table 2

## 9.5 Transitions

The SHA-1 algorithm, as implemented by the module, will be non-approved for all purposes after December 31, 2030.

# 10 Self-Tests

## 10.1 Pre-Operational Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details
SHA2-256 (A5832)	SHA2-256	KAT	SW/FW Integrity	user.err log	Message Digest

Table 17: Pre-Operational Self-Tests

If the pre-operational self-test fails, the module enters the error state.

## 10.2 Conditional Self-Tests

<b>Algorithm or Test</b>	<b>Test Properties</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Indicator</b>	<b>Details</b>	<b>Conditions</b>
AES-ECB (A5827)-Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-CBC (A5827)-Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-CTR (A5827)-Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-XTS (A5827)-Encrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-CBC-CS3 (A5827)-Encrypt	Key size: 128 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-KW (A5827)-Encrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-time use of this algorithm after OS boot
AES-CMAC (A5827)-Generation	Key size: 128, 256 bits	KAT	CAST	Kernel log	Generation	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-ECB (A5828)-Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-

<b>Algorithm or Test</b>	<b>Test Properties</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Indicator</b>	<b>Details</b>	<b>Conditions</b>
						time use of this algorithm after OS boot.
AES-CBC (A5828)-Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-CTR (A5828)-Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-XTS (A5828)-Encrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-CBC-CS3 (A5828)-Encrypt	Key size: 128 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-KW (A5828)-Encrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-time use of this algorithm after OS boot
AES-ECB (A5827)-Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-CBC (A5827)-Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-CTR (A5827)-Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-

<b>Algorithm or Test</b>	<b>Test Properties</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Indicator</b>	<b>Details</b>	<b>Conditions</b>
						time use of this algorithm after OS boot.
AES-XTS (A5827)-Decrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-CBC-CS3 (A5827)-Decrypt	Key size: 128 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-KW (A5827)-Decrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-time use of this algorithm after OS boot
AES-CMAC (A5827)-Verification	Key size: 128, 256 bits	KAT	CAST	Kernel log	Verification	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-ECB (A5828)-Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-CBC (A5828)-Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-CTR (A5828)-Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-XTS (A5828)-Decrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-

<b>Algorithm or Test</b>	<b>Test Properties</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Indicator</b>	<b>Details</b>	<b>Conditions</b>
						time use of this algorithm after OS boot.
AES-CBC-CS3 (A5828)-Decrypt	Key size: 128 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-KW (A5828)-Decrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-time use of this algorithm after OS boot
SHA2-256 (A5828)	N/A	KAT	CAST	Kernel log	Hash	From the module startup to the operation of the module.
SHA2-384 (A5828)	N/A	KAT	CAST	Kernel log	Hash	From the module startup to the operation of the module.
SHA2-512 (A5828)	N/A	KAT	CAST	Kernel log	Hash	From the module startup to the operation of the module.
AES-CMAC (A5828)-Generation	Key size: 128, 256 bits	KAT	CAST	Kernel log	Generation	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-CMAC (A5828)-Verification	Key size: 128, 256 bits	KAT	CAST	Kernel log	Verification	From the operation of the module to the first-time use of this algorithm after OS boot.
HMAC-SHA2-256 (A5828)	Key size: 32, 256, 296, 640 bits	KAT	CAST	Kernel log	MAC	From the operation of the module to the first-time use of this algorithm after OS boot.
HMAC-SHA2-384 (A5828)	Key size: 32, 160, 1048 bits	KAT	CAST	Kernel log	MAC	From the operation of the module to the first-

<b>Algorithm or Test</b>	<b>Test Properties</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Indicator</b>	<b>Details</b>	<b>Conditions</b>
						time use of this algorithm after OS boot.
HMAC-SHA2-512 (A5828)	Key size: 32, 160, 1048 bits	KAT	CAST	Kernel log	MAC	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-ECB (A5830)-Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-CBC (A5830)-Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-CTR (A5830)-Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-XTS (A5830)-Encrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-CBC-CS3 (A5830)-Encrypt	Key size: 128 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-KW (A5830)-Encrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-time use of this algorithm after OS boot
AES-ECB (A5830)-Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-

<b>Algorithm or Test</b>	<b>Test Properties</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Indicator</b>	<b>Details</b>	<b>Conditions</b>
						time use of this algorithm after OS boot.
AES-CBC (A5830)-Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-CTR (A5830)-Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-XTS (A5830)-Decrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-CBC-CS3 (A5830)-Decrypt	Key size: 128 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-KW (A5830)-Decrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-time use of this algorithm after OS boot
SHA-1 (A5830)	N/A	KAT	CAST	Kernel log	Hash	From the module startup to the operation of the module.
SHA2-256 (A5830)	N/A	KAT	CAST	Kernel log	Hash	From the module startup to the operation of the module.
AES-CMAC (A5830)-Generation	Key size: 128, 256 bits	KAT	CAST	Kernel log	Generation	From the operation of the module to the first-time use of this algorithm after OS boot.

<b>Algorithm or Test</b>	<b>Test Properties</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Indicator</b>	<b>Details</b>	<b>Conditions</b>
AES-CMAC (A5830)- Verification	Key size: 128, 256 bits	KAT	CAST	Kernel log	Verification	From the operation of the module to the first-time use of this algorithm after OS boot.
HMAC-SHA-1 (A5830)	Key size: 32, 160, 200, 640 bits	KAT	CAST	Kernel log	MAC	From the operation of the module to the first-time use of this algorithm after OS boot.
HMAC-SHA2-256 (A5830)	Key size: 32, 256, 296, 640 bits	KAT	CAST	Kernel log	MAC	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-ECB (A5829)- Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Encrypt	From the module startup to the operation of the module.
AES-CBC (A5829)- Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Encrypt	From the module startup to the operation of the module.
AES-CTR (A5829)- Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Encrypt	From the module startup to the operation of the module.
AES-XTS (A5829)- Encrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Encrypt	From the module startup to the operation of the module.
AES-CBC-CS3 (A5829)- Encrypt	Key size: 128 bits	KAT	CAST	Kernel log	Encrypt	From the module startup to the operation of the module.
AES-ECB (A5829)- Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Decrypt	From the module startup to the operation of the module.
AES-CBC (A5829)- Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Decrypt	From the module startup to the operation of the module.

<b>Algorithm or Test</b>	<b>Test Properties</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Indicator</b>	<b>Details</b>	<b>Conditions</b>
AES-CTR (A5829)-Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Decrypt	From the module startup to the operation of the module.
AES-XTS (A5829)-Decrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Decrypt	From the module startup to the operation of the module.
AES-CBC-CS3 (A5829)-Decrypt	Key size: 128 bits	KAT	CAST	Kernel log	Decrypt	From the module startup to the operation of the module.
AES-CMAC (A5829)-Generation	Key size: 128, 256 bits	KAT	CAST	Kernel log	Generation	From the module startup to the operation of the module.
AES-CMAC (A5829)-Verification	Key size: 128, 256 bits	KAT	CAST	Kernel log	Verification	From the module startup to the operation of the module.
AES-ECB (A5831)-Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Encrypt	From the module startup to the operation of the module.
AES-CBC (A5831)-Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Encrypt	From the module startup to the operation of the module.
AES-CTR (A5831)-Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Encrypt	From the module startup to the operation of the module.
AES-XTS (A5831)-Encrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Encrypt	From the module startup to the operation of the module.
AES-CBC-CS3 (A5831)-Encrypt	Key size: 128 bits	KAT	CAST	Kernel log	Encrypt	From the module startup to the operation of the module.
AES-KW (A5831)-Encrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Encrypt	From the operation of the module to the first-time use of this algorithm after OS boot

<b>Algorithm or Test</b>	<b>Test Properties</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Indicator</b>	<b>Details</b>	<b>Conditions</b>
AES-ECB (A5831)-Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Decrypt	From the module startup to the operation of the module.
AES-CBC (A5831)-Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Decrypt	From the module startup to the operation of the module.
AES-CTR (A5831)-Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Decrypt	From the module startup to the operation of the module.
AES-XTS (A5831)-Decrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Decrypt	From the module startup to the operation of the module.
AES-CBC-CS3 (A5831)-Decrypt	Key size: 128 bits	KAT	CAST	Kernel log	Decrypt	From the module startup to the operation of the module.
AES-KW (A5831)-Decrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Decrypt	From the operation of the module to the first-time use of this algorithm after OS boot
SHA-1 (A5831)	N/A	KAT	CAST	Kernel log	Hash	From the module startup to the operation of the module.
SHA2-256 (A5831)	N/A	KAT	CAST	Kernel log	Hash	From the module startup to the operation of the module.
SHA2-384 (A5831)	N/A	KAT	CAST	Kernel log	Hash	From the module startup to the operation of the module.
SHA2-512 (A5831)	N/A	KAT	CAST	Kernel log	Hash	From the module startup to the operation of the module.
SHA3-256 (A5831)	N/A	KAT	CAST	Kernel log	Hash	From the module startup to the operation of the module.
SHA3-384 (A5831)	N/A	KAT	CAST	Kernel log	Hash	From the module startup to the

<b>Algorithm or Test</b>	<b>Test Properties</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Indicator</b>	<b>Details</b>	<b>Conditions</b>
						operation of the module.
SHA3-512 (A5831)	N/A	KAT	CAST	Kernel log	Hash	From the module startup to the operation of the module.
AES-CMAC (A5831)-Generation	Key size: 128, 256 bits	KAT	CAST	Kernel log	Generation	From the module startup to the operation of the module.
AES-CMAC (A5831)-Verification	Key size: 128, 256 bits	KAT	CAST	Kernel log	Verification	From the module startup to the operation of the module.
HMAC-SHA-1 (A5831)	Key size: 32, 160, 200, 640 bits	KAT	CAST	Kernel log	MAC	From the operation of the module to the first-time use of this algorithm after OS boot.
HMAC-SHA2-256 (A5831)	Key size: 32, 256, 296, 640 bits	KAT	CAST	Kernel log	MAC	From the operation of the module to the first-time use of this algorithm after OS boot.
HMAC-SHA2-384 (A5831)	Key size: 32, 160, 1048 bits	KAT	CAST	Kernel log	MAC	From the operation of the module to the first-time use of this algorithm after OS boot.
HMAC-SHA2-512 (A5831)	Key size: 32, 160, 1048 bits	KAT	CAST	Kernel log	MAC	From the operation of the module to the first-time use of this algorithm after OS boot.
HMAC-SHA3-256 (A5831)	Key size: 32, 160, 1048 bits	KAT	CAST	Kernel log	MAC	From the operation of the module to the first-time use of this algorithm after OS boot.
HMAC-SHA3-384 (A5831)	Key size: 32, 160, 1048 bits	KAT	CAST	Kernel log	MAC	From the operation of the module to the first-time use of this

<b>Algorithm or Test</b>	<b>Test Properties</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Indicator</b>	<b>Details</b>	<b>Conditions</b>
						algorithm after OS boot.
HMAC-SHA3-512 (A5831)	Key size: 32, 160, 1048 bits	KAT	CAST	Kernel log	MAC	From the operation of the module to the first-time use of this algorithm after OS boot.
SHA2-256 (A5832)	N/A	KAT	CAST	user.err log	Hash	From the module startup to the integrity testing.
AES-CBC-CS3 (A5833)-Encrypt	Key size: 128 bits	KAT	CAST	Kernel log	Encrypt	From the module startup to the operation of the module.
AES-CBC-CS3 (A5833)-Decrypt	Key size: 128 bits	KAT	CAST	Kernel log	Decrypt	From the module startup to the operation of the module.
SHA2-256 (A5833)	N/A	KAT	CAST	Kernel log	Hash	From the module startup to the operation of the module.
AES-CMAC (A5833)-Generation	Key size: 128, 256 bits	KAT	CAST	Kernel log	Generation	From the module startup to the operation of the module.
AES-CMAC (A5833))-Verification	Key size: 128, 256 bits	KAT	CAST	Kernel log	Verification	From the module startup to the operation of the module.
HMAC-SHA2-256 (A5833)	Key size: 32, 256, 296, 640 bits	KAT	CAST	Kernel log	MAC	From the operation of the module to the first-time use of this algorithm after OS boot.
AES-ECB (A5834)-Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Encrypt	From the module startup to the operation of the module.
AES-CBC (A5834)-Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Encrypt	From the module startup to the operation of the module.
AES-CTR (A5834)-Encrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Encrypt	From the module startup to the

<b>Algorithm or Test</b>	<b>Test Properties</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Indicator</b>	<b>Details</b>	<b>Conditions</b>
						operation of the module.
AES-XTS (A5834)-Encrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Encrypt	From the module startup to the operation of the module.
AES-ECB (A5834)-Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Decrypt	From the module startup to the operation of the module.
AES-CBC (A5834)-Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Decrypt	From the module startup to the operation of the module.
AES-CTR (A5834)-Decrypt	Key size: 128, 192, 256 bits	KAT	CAST	Kernel log	Decrypt	From the module startup to the operation of the module.
AES-XTS (A5834)-Decrypt	Key size: 128, 256 bits	KAT	CAST	Kernel log	Decrypt	From the module startup to the operation of the module.

Table 18: Conditional Self-Tests

When the operational environment is powered on and the module is loaded, the module starts to perform each cryptographic algorithm self-test for the algorithm which "Conditions" item in the Conditional Self-Tests table is "From the module startup to the operation of the module" or "From the module startup to the integrity testing". If one of the self-tests fails, the module enters the error state.

### 10.3 Periodic Self-Test Information

<b>Algorithm or Test</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Period</b>	<b>Periodic Method</b>
SHA2-256 (A5832)	KAT	SW/FW Integrity	On Demand	Manually

Table 19: Pre-Operational Periodic Information

<b>Algorithm or Test</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Period</b>	<b>Periodic Method</b>
AES-ECB (A5827)-Encrypt	KAT	CAST	On Demand	Manually
AES-CBC (A5827)-Encrypt	KAT	CAST	On Demand	Manually
AES-CTR (A5827)-Encrypt	KAT	CAST	On Demand	Manually
AES-XTS (A5827)-Encrypt	KAT	CAST	On Demand	Manually

<b>Algorithm or Test</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Period</b>	<b>Periodic Method</b>
AES-CBC-CS3 (A5827)-Encrypt	KAT	CAST	On Demand	Manually
AES-KW (A5827)-Encrypt	KAT	CAST	On Demand	Manually
AES-CMAC (A5827)-Generation	KAT	CAST	On Demand	Manually
AES-ECB (A5828)-Encrypt	KAT	CAST	On Demand	Manually
AES-CBC (A5828)-Encrypt	KAT	CAST	On Demand	Manually
AES-CTR (A5828)-Encrypt	KAT	CAST	On Demand	Manually
AES-XTS (A5828)-Encrypt	KAT	CAST	On Demand	Manually
AES-CBC-CS3 (A5828)-Encrypt	KAT	CAST	On Demand	Manually
AES-KW (A5828)-Encrypt	KAT	CAST	On Demand	Manually
AES-ECB (A5827)-Decrypt	KAT	CAST	On Demand	Manually
AES-CBC (A5827)-Decrypt	KAT	CAST	On Demand	Manually
AES-CTR (A5827)-Decrypt	KAT	CAST	On Demand	Manually
AES-XTS (A5827)-Decrypt	KAT	CAST	On Demand	Manually
AES-CBC-CS3 (A5827)-Decrypt	KAT	CAST	On Demand	Manually
AES-KW (A5827)-Decrypt	KAT	CAST	On Demand	Manually
AES-CMAC (A5827)-Verification	KAT	CAST	On Demand	Manually
AES-ECB (A5828)-Decrypt	KAT	CAST	On Demand	Manually
AES-CBC (A5828)-Decrypt	KAT	CAST	On Demand	Manually
AES-CTR (A5828)-Decrypt	KAT	CAST	On Demand	Manually
AES-XTS (A5828)-Decrypt	KAT	CAST	On Demand	Manually
AES-CBC-CS3 (A5828)-Decrypt	KAT	CAST	On Demand	Manually
AES-KW (A5828)-Decrypt	KAT	CAST	On Demand	Manually
SHA2-256 (A5828)	KAT	CAST	On Demand	Manually

<b>Algorithm or Test</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Period</b>	<b>Periodic Method</b>
SHA2-384 (A5828)	KAT	CAST	On Demand	Manually
SHA2-512 (A5828)	KAT	CAST	On Demand	Manually
AES-CMAC (A5828)-Generation	KAT	CAST	On Demand	Manually
AES-CMAC (A5828)-Verification	KAT	CAST	On Demand	Manually
HMAC-SHA2-256 (A5828)	KAT	CAST	On Demand	Manually
HMAC-SHA2-384 (A5828)	KAT	CAST	On Demand	Manually
HMAC-SHA2-512 (A5828)	KAT	CAST	On Demand	Manually
AES-ECB (A5830)-Encrypt	KAT	CAST	On Demand	Manually
AES-CBC (A5830)-Encrypt	KAT	CAST	On Demand	Manually
AES-CTR (A5830)-Encrypt	KAT	CAST	On Demand	Manually
AES-XTS (A5830)-Encrypt	KAT	CAST	On Demand	Manually
AES-CBC-CS3 (A5830)-Encrypt	KAT	CAST	On Demand	Manually
AES-KW (A5830)-Encrypt	KAT	CAST	On Demand	Manually
AES-ECB (A5830)-Decrypt	KAT	CAST	On Demand	Manually
AES-CBC (A5830)-Decrypt	KAT	CAST	On Demand	Manually
AES-CTR (A5830)-Decrypt	KAT	CAST	On Demand	Manually
AES-XTS (A5830)-Decrypt	KAT	CAST	On Demand	Manually
AES-CBC-CS3 (A5830)-Decrypt	KAT	CAST	On Demand	Manually
AES-KW (A5830)-Decrypt	KAT	CAST	On Demand	Manually
SHA-1 (A5830)	KAT	CAST	On Demand	Manually
SHA2-256 (A5830)	KAT	CAST	On Demand	Manually
AES-CMAC (A5830)-Generation	KAT	CAST	On Demand	Manually

<b>Algorithm or Test</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Period</b>	<b>Periodic Method</b>
AES-CMAC (A5830)-Verification	KAT	CAST	On Demand	Manually
HMAC-SHA-1 (A5830)	KAT	CAST	On Demand	Manually
HMAC-SHA2-256 (A5830)	KAT	CAST	On Demand	Manually
AES-ECB (A5829)-Encrypt	KAT	CAST	On Demand	Manually
AES-CBC (A5829)-Encrypt	KAT	CAST	On Demand	Manually
AES-CTR (A5829)-Encrypt	KAT	CAST	On Demand	Manually
AES-XTS (A5829)-Encrypt	KAT	CAST	On Demand	Manually
AES-CBC-CS3 (A5829)-Encrypt	KAT	CAST	On Demand	Manually
AES-ECB (A5829)-Decrypt	KAT	CAST	On Demand	Manually
AES-CBC (A5829)-Decrypt	KAT	CAST	On Demand	Manually
AES-CTR (A5829)-Decrypt	KAT	CAST	On Demand	Manually
AES-XTS (A5829)-Decrypt	KAT	CAST	On Demand	Manually
AES-CBC-CS3 (A5829)-Decrypt	KAT	CAST	On Demand	Manually
AES-CMAC (A5829)-Generation	KAT	CAST	On Demand	Manually
AES-CMAC (A5829)-Verification	KAT	CAST	On Demand	Manually
AES-ECB (A5831)-Encrypt	KAT	CAST	On Demand	Manually
AES-CBC (A5831)-Encrypt	KAT	CAST	On Demand	Manually
AES-CTR (A5831)-Encrypt	KAT	CAST	On Demand	Manually
AES-XTS (A5831)-Encrypt	KAT	CAST	On Demand	Manually
AES-CBC-CS3 (A5831)-Encrypt	KAT	CAST	On Demand	Manually
AES-KW (A5831)-Encrypt	KAT	CAST	On Demand	Manually
AES-ECB (A5831)-Decrypt	KAT	CAST	On Demand	Manually

<b>Algorithm or Test</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Period</b>	<b>Periodic Method</b>
AES-CBC (A5831)-Decrypt	KAT	CAST	On Demand	Manually
AES-CTR (A5831)-Decrypt	KAT	CAST	On Demand	Manually
AES-XTS (A5831)-Decrypt	KAT	CAST	On Demand	Manually
AES-CBC-CS3 (A5831)-Decrypt	KAT	CAST	On Demand	Manually
AES-KW (A5831)-Decrypt	KAT	CAST	On Demand	Manually
SHA-1 (A5831)	KAT	CAST	On Demand	Manually
SHA2-256 (A5831)	KAT	CAST	On Demand	Manually
SHA2-384 (A5831)	KAT	CAST	On Demand	Manually
SHA2-512 (A5831)	KAT	CAST	On Demand	Manually
SHA3-256 (A5831)	KAT	CAST	On Demand	Manually
SHA3-384 (A5831)	KAT	CAST	On Demand	Manually
SHA3-512 (A5831)	KAT	CAST	On Demand	Manually
AES-CMAC (A5831)-Generation	KAT	CAST	On Demand	Manually
AES-CMAC (A5831)-Verification	KAT	CAST	On Demand	Manually
HMAC-SHA-1 (A5831)	KAT	CAST	On Demand	Manually
HMAC-SHA2-256 (A5831)	KAT	CAST	On Demand	Manually
HMAC-SHA2-384 (A5831)	KAT	CAST	On Demand	Manually
HMAC-SHA2-512 (A5831)	KAT	CAST	On Demand	Manually
HMAC-SHA3-256 (A5831)	KAT	CAST	On Demand	Manually
HMAC-SHA3-384 (A5831)	KAT	CAST	On Demand	Manually
HMAC-SHA3-512 (A5831)	KAT	CAST	On Demand	Manually
SHA2-256 (A5832)	KAT	CAST	On Demand	Manually
AES-CBC-CS3 (A5833)-Encrypt	KAT	CAST	On Demand	Manually

<b>Algorithm or Test</b>	<b>Test Method</b>	<b>Test Type</b>	<b>Period</b>	<b>Periodic Method</b>
AES-CBC-CS3 (A5833)-Decrypt	KAT	CAST	On Demand	Manually
SHA2-256 (A5833)	KAT	CAST	On Demand	Manually
AES-CMAC (A5833)-Generation	KAT	CAST	On Demand	Manually
AES-CMAC (A5833))-Verification	KAT	CAST	On Demand	Manually
HMAC-SHA2-256 (A5833)	KAT	CAST	On Demand	Manually
AES-ECB (A5834)-Encrypt	KAT	CAST	On Demand	Manually
AES-CBC (A5834)-Encrypt	KAT	CAST	On Demand	Manually
AES-CTR (A5834)-Encrypt	KAT	CAST	On Demand	Manually
AES-XTS (A5834)-Encrypt	KAT	CAST	On Demand	Manually
AES-ECB (A5834)-Decrypt	KAT	CAST	On Demand	Manually
AES-CBC (A5834)-Decrypt	KAT	CAST	On Demand	Manually
AES-CTR (A5834)-Decrypt	KAT	CAST	On Demand	Manually
AES-XTS (A5834)-Decrypt	KAT	CAST	On Demand	Manually

Table 20: Conditional Periodic Information

The pre-operational self-tests, and all cryptographic algorithm self-tests listed in Conditional Self-Tests Table are available on demand by power cycling or rebooting of the operational environment.

## 10.4 Error States

<b>Name</b>	<b>Description</b>	<b>Conditions</b>	<b>Recovery Method</b>	<b>Indicator</b>
Error	A state when the module has encountered an error condition.	Condition 1: Failed the pre-operational self-tests or the conditional self-test for SHA2-256 (A5832). Condition 2: Failed the conditional self-tests except for that of SHA2-256 (A5832).	Power cycling or rebooting of the operational environment.	Condition 1: user.err log; Condition 2: kernel log

Table 21: Error States

After the OS boots, the user.err log and the kernel log (kern.log) are moved to /psstorage1/pltf/snapshot-log/<id>/var/log/ directory. <id> is a number from 0 to 4. <id> is incremented by the OS boot and wrapped back to 0 when that max value is reached. The latest <id> corresponds to that of the filename of the “latest-number-<id>” file in /psstorage1/pltf/snapshot-log/ directory.

## 11 Life-Cycle Assurance

### 11.1 Installation, Initialization, and Startup Procedures

The module is integrated into Embedded Storage Manager. When Embedded Storage Manager is installed by the vendor of Hitachi storage system, the module is also installed. No other special procedure is required to securely install and initialize the module.

### 11.2 Administrator Guidance

Administrators can verify that an ID and a version of the module is identical to the ID (Embedded Storage Manager Kernel Crypto API) and the version (1.1). To show an ID and a version of the module, run “showversion-fips” command in Command Console of ESM.

All the functions, physical ports, and logical interfaces of the module are available to the Crypto Officer.

### 11.3 Non-Administrator Guidance

There are no requirements for non-administrator.

### 11.4 Design and Rules

The module design corresponds to the module security rules. This subsection documents the security rules enforced by the module to implement the security requirements of this FIPS 140-3 Level 1 module.

1. The module shall provide a Cryptographic Officer role.
2. The operator shall be capable of commanding the module to perform the pre-operational self-tests and the cryptographic algorithm self-tests which are configured to be executed during the module initialization process by cycling power or reboot of the operational environment.
3. Pre-operational self-tests do not require any operator action.
4. Data output shall be inhibited during self-tests, zeroisation, and error states.
5. Status information does not contain CSPs or sensitive data that if misused could lead to a compromise of the module.
6. The module does not support degraded operation.
7. The module does not support concurrent operators.
8. The module does not support a maintenance interface or role.
9. The module does not support manual key entry.

10. The module does not have any external input/output devices used for entry/output of data.

## 12 Mitigation of Other Attacks

N/A. The module does not provide mitigation of other attacks.