Apple Inc.

Apple corecrypto Module v11.1 [Apple silicon, Secure Key Store, Hardware, SL2/PHY3] FIPS 140-3 Non-Proprietary Security Policy

document version 1.1

July 2024

Prepared for: Apple

One Apple Park Way

Cupertino, CA 95014

Prepared by: atsec information security corporation

4516 Seton Center Parkway, Suite 250

Austin, TX 78759

www.atsec.com

Trademarks

Apple's trademarks applicable to this document are listed in https://www.apple.com/legal/intellectual-property/trademark/appletmlist.html.

Other company, product, and service names may be trademarks or service marks of others.

Table of Contents

List of Tables

1 General

This document is the non-proprietary FIPS 140-3 Security Policy for Apple corecrypto Module v11.1 [Apple silicon, Secure Key Store, Hardware, SL2/PHY3] cryptographic module. It contains the security rules under which the module must operate and describes how this module meets the requirements as specified in FIPS PUB 140-3 (Federal Information Processing Standards Publication 140-3) for an overall Security Level 2 module.

This document provides all tables and diagrams (when applicable) required by NIST SP 800-140B. The column names of the tables follow the template tables provided in NIST SP 800-140B.

Table 1 describes the individual security areas of FIPS 140-3, as well as the Security Levels of those individual areas.

ISO/IEC 24759 Section 6. [Number Below]	FIPS 140-3 Section Title	Security Level
1	General	2
2	Cryptographic Module Specification	2
3	Cryptographic Module Interfaces	2
4	Roles, Services, and Authentication	2
5	Software/Firmware Security	2
6	Operational Environment	Not Applicable
7	Physical Security	3
8	Non-invasive Security	Not Applicable
9	Sensitive Security Parameter Management	2
10	Self-tests	2
11	Life-cycle Assurance	2
12	Mitigation of Other Attacks	Not Applicable

Table 1 - Security Levels

2 Cryptographic Module Specification

The Apple corecrypto Module v11.1 [Apple silicon, Secure Key Store, Hardware, SL2/PHY3] cryptographic module (hereafter referred to as "the module") is a Hardware module implemented as a sub-chip running on a single-chip processor. The version of module's firmware is 11.1 and the Hardware version is 2.0. The sub-chip module is embedded in the hardware listed in Table 2. The sub-chip module's firmware is bundled together with the underlying Device OS.

2.1 Module components

The module consists of both firmware and hardware components. The Secure Key Store (SKS) application is the module's firmware which operates within the sepOS execution environment which is separate from the Device OS's (iOS 14.2, iPadOS 14.2, and macOS 11.0.1) execution environment. The firmware boundary is defined as the API offered by the mailbox interface to callers from the Device OS execution environment. SKS has an API layer that provides consistent interfaces to the supported services and therefore the supported cryptographic algorithms. In addition, the module provides Inter-Process Communication (IPC) interfaces to other applications executing within the sepOS execution environment.

The sepOS execution environment is driven by its own SoC and operates from a dedicated region of the device's memory. Both the Device's and sepOS' execution environments are physically separated on the SoC and thus execute independently of each other.

The cryptographic module boundary includes the following hardware components:

- Hardware Random Number Generator composed of an SP800-90ARev1 Approved CTR_DRBG and a compliant SP800-90B physical entropy source.
- Hardware AES implementations using 128-bit to 256-bit keys.
- Hardware Public Key Accelerator (PKA) used for generating non-approved P-224, P-256, P-384 or P-521 asymmetric key pairs.
- A shared memory segment (called Mailbox) that can be accessed by both SKS and the Device OS's XNU kernel, supported with an interrupt system and used by XNU to request services of the SKS module.
- A volatile RAM for storing runtime SSPs.
- A non-volatile Flash for storing Class D key and encrypted user keybag.

2.1.1 Photograph and Block Diagram

The module physical boundary is defined by the SoC perimeter.

Figure 1 - Picture of the SoC tested (Apple A13 Bionic, Apple A14 Bionic and Apple M1)

The block diagram below depicts the following information:

- The location of the logical object of the firmware components of the hardware module with respect to the operating system, other supporting applications, and the cryptographic boundary so that all the logical and physical layers between the logical object and the cryptographic boundary are clearly defined.
- The interactions of the logical object of the module with the operating system and other supporting applications resident within the cryptographic boundary.

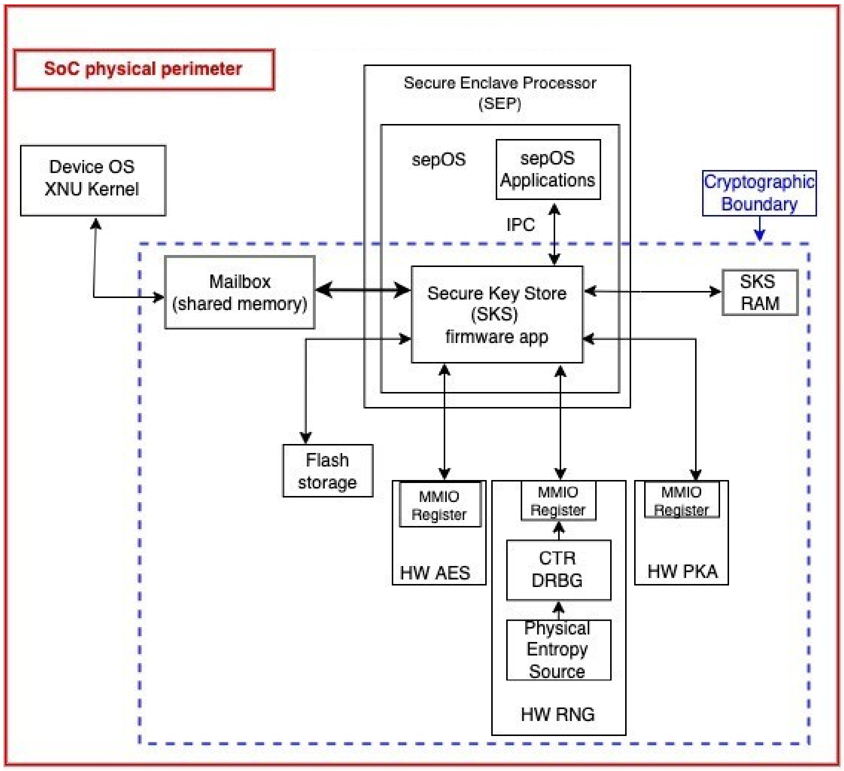

Figure 2 - Block diagram

2.2 Tested Platforms

The hardware module has been tested by atsec CST lab on the following platforms:

Model	Hardware version(s)	Firmware version(s)	Processor(s)	Distinguishing Features
iPad Air (4th generation) running sepOS distributed with iPadOS 14.2	2.0	11.1	Apple A14 Bionic	N/A
iPhone 11 Pro running sepOS distributed with iOS 14.2	2.0	11.1	Apple A13 Bionic	N/A
iPhone 12 running sepOS distributed with iOS 14.2	2.0	11.1	Apple A14 Bionic	N/A
MacBook Air running sepOS distributed with macOS Big Sur 11.0.1	2.0	11.1	Apple M1	N/A

Table 2 - Tested Operational Environments

2.3 Cryptographic Algorithms

The table below lists all Approved or Vendor-affirmed security functions of the module, including specific key size(s) employed for approved services, and implemented modes of operation. Some of the CAVP certificates, show testing for AES CTR, CCM or OFB modes but they are not used by the module. The module is in the Approved mode of operation when the module utilizes the services that use the security functions listed in the table below.

2.3.1 Approved Security Functions

CAVP Cert.	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A1342 (asm_arm)	AES [FIPS 197] [SP 800-38 A]	CBC	Key Length/ Key Strength: 256	Symmetric Encryption and Decryption
A1343 (c_asm)	AES [FIPS 197] [SP 800-38 A]	CBC	Key Length/ Key Strength: 256	Symmetric Encryption and Decryption
A1344 (c_glad)	AES [FIPS 197] [SP 800-38 A]	CBC	Key Length/ Key Strength: 256	Symmetric Encryption and Decryption
A1345 (c_ltc)	AES [FIPS 197] [SP 800-38 A]	CBC	Key Length/ Key Strength: 256	Symmetric Encryption and Decryption
A510 (skg)	AES [FIPS 197] [SP 800-38 A]	CBC	Key Length/ Key Strength: 256	Symmetric Encryption and Decryption
A1469 (skg)	AES [FIPS 197] [SP 800-38 A]	CBC	Key Length/ Key Strength: 256	Symmetric Encryption and Decryption
A501 (trng)	AES [FIPS 197] [SP 800-38 A]	ECB	Key Length/ Key Strength: 256	Symmetric Encryption and Decryption
A1362 (trng)	AES [FIPS 197] [SP 800-38 A]	ECB	Key Length/ Key Strength: 256	Symmetric Encryption and Decryption
A510 (skg)	AES [FIPS 197] [SP 800-38 A]	ECB	Key Length/ Key Strength: 256	Symmetric Encryption and Decryption
A1469 (skg)	AES [FIPS 197] [SP 800-38 A]	ECB	Key Length/ Key Strength: 256	Symmetric Encryption and Decryption
A501 (trng)	CTR_DRBG [SP800-90ARev1]	AES-256; No Derivation Function; Prediction Resistance Enabled	Key Length/ Key Strength: 256	Random Number Generation
A1362 (trng)	CTR_DRBG [SP800-90ARev1]	AES-256; No Derivation Function; Prediction Resistance Enabled	Key Length/ Key Strength: 256	Random Number Generation
vendor affirmed	CKG [SP800-133Rev2 section 4]	AES key	Key Length/ Key Strength: 256	Key Generation
A1340 (vng_ltc)	HMAC [FIPS 198]	SHA-1	Key Length/ Key Strength: 112 bits or greater	Keyed Hash
A1345 (c_ltc)	HMAC [FIPS 198]	SHA-1	Key Length/ Key Strength: 112 bits or greater	Keyed Hash
A1340 (vng_ltc)	HMAC [FIPS 198]	SHA2-224	Key Length/ Key Strength: 112 bits or greater	Keyed Hash
A1345 (c_ltc)	HMAC [FIPS 198]	SHA2-224	Key Length/ Key Strength: 112 bits or greater	Keyed Hash
A1340 (vng_ltc)	HMAC [FIPS 198]	SHA2-256	Key Length/ Key Strength: 112 bits or greater	Keyed Hash
A1345 (c_ltc)	HMAC [FIPS 198]	SHA2-256	Key Length/ Key Strength: 112 bits or greater	Keyed Hash
A1341 (vng_neon)	HMAC [FIPS 198]	SHA2-256	Key Length/ Key Strength: 112 bits or greater	Keyed Hash
A1340 (vng_ltc)	HMAC [FIPS 198]	SHA2-384	Key Length/ Key Strength: 112 bits or greater	Keyed Hash
A1345 (c_ltc)	HMAC [FIPS 198]	SHA2-384	Key Length/ Key Strength: 112 bits or greater	Keyed Hash
A1340 (vng_ltc)	HMAC [FIPS 198]	SHA2-512	Key Length/ Key Strength: 112 bits or greater	Keyed Hash
A1345 (c_ltc)	HMAC [FIPS 198]	SHA2-512	Key Length/ Key Strength: 112 bits or greater	Keyed Hash
A1340 (vng_ltc)	HMAC [FIPS 198]	SHA2-512/256	Key Length/ Key Strength: 112 bits or greater	Keyed Hash
A1343 (c_asm)	KTS [SP 800-38 F]	AES-KW	Key Length/ Key Strength: 256	Key Wrapping
A1345 (c_ltc)	KTS [SP 800-38 F]	AES-KW	Key Length/ Key Strength: 256	Key Wrapping

CAVP Cert.	Algorithm and Standard	Mode / Method	Description / Key Size(s) / Key Strength(s)	Use / Function
A1340 (vng_ltc)	SHS [FIPS 180-4]	SHA-1	N/A	Message Digest
A1345 (c_ltc)	SHS [FIPS 180-4]	SHA-1	N/A	Message Digest
A1340 (vng_ltc)	SHS [FIPS 180-4]	SHA2-224	N/A	Message Digest
A1345 (c_ltc)	SHS [FIPS 180-4]	SHA2-224	N/A	Message Digest
A1340 (vng_ltc)	SHS [FIPS 180-4]	SHA2-256	N/A	Message Digest
A1345 (c_ltc)	SHS [FIPS 180-4]	SHA2-256	N/A	Message Digest
A1341 (vng_neon)	SHS [FIPS 180-4]	SHA2-256	N/A	Message Digest
A1340 (vng_ltc)	SHS [FIPS 180-4]	SHA2-384	N/A	Message Digest
A1345 (c_ltc)	SHS [FIPS 180-4]	SHA2-384	N/A	Message Digest
A1340 (vng_ltc)	SHS [FIPS 180-4]	SHA2-512	N/A	Message Digest
A1345 (c_ltc)	SHS [FIPS 180-4]	SHA2-512	N/A	Message Digest
A1340 (vng_ltc)	SHS [FIPS 180-4]	SHA2-512/256	N/A	Message Digest

Table 3 - Approved Algorithms

This module does not implement non-Approved algorithms Allowed in the Approved Mode of operation nor non-Approved algorithms used in the Approved mode of operation with no security claimed.

2.3.2 Non-Approved Security Functions

The table below lists Non-Approved security functions that are not Allowed in the Approved Mode of Operation:

Algorithm/Functions	Use / Function
Ed25519 Key Generation	EdDSA signature scheme
Ed25519 shared secret generation	EdDSA shared secret generation
Curve 25519 key generation	Key generation
Curve 25519 shared secret generation	shared secret generation
ECDH Key Pair Generation	Elliptic Curve Integrated Encryption Scheme (ECIES) key generation
ECDH Shared Secret Computation	Elliptic Curve Integrated Encryption Scheme (ECIES) Encryption
ANSI X9.63 KDF	
AES-GCM	
ECDH Shared Secret Computation	Elliptic Curve Integrated Encryption Scheme (ECIES) Decryption
ANSI X9.63 KDF	
AES-GCM	
HKDF RFC5869	HMAC based Key Derivation Function
PBKDF	Key Derivation
ECDSA implemented in FW	Key generation as part of Ref key generation service and validation, Signature generation and verification as part of Device keybag service
ECDSA implemented in HW PKA	Key generation as part of Ref key generation service Signature generation primitive
ECDH implemented in FW	Shared secret computation
ECDH implemented in HW PKA	Shared secret computation
AES KW using class D key, keys from Device keybag, keys	Key wrapping and unwrapping

from iCloud keybag, keys from Escrow keybag, keys from any keybag used with Class B Curve 25519 encrypt/decrypt, keys from Backup keybag used for wrapping Ed25519 keys, or NVM storage controller key	

Table 4 - Non-Approved Algorithms Not Allowed in the Approved Mode of Operation

3 Cryptographic Module Interfaces

The cryptographic interfaces of the module are provided through the mailbox interface that is used between the module and the Device OS kernel, and the IPC channel used between the module and other sepOS applications. In detail these interfaces are described in (Table 5):

Physical Port[1]	Logical Interface	Data that passes over port/interface
Mailbox Memory, IPC channel	Data Input	Data inputs are provided through the memory used for mailbox and IPC.
Mailbox Memory, IPC channel	Data Output	Data outputs are provided through the memory used for mailbox and IPC.
Mailbox Memory, IPC channel	Control Input	Control input which controls the module’s operation is provided through the mailbox by the Device OS’ kernel and to applications located within the sepOS execution environment through IPC.
Mailbox Memory, IPC channel	Status Output	Status output is provided in return codes and through messages returned via the mailbox or the IPC. Documentation for each service invocation lists possible return codes. A complete list of all return codes returned by the C language APIs within the module is provided in the header files and the API documentation. Messages are also documented in the API documentation.
single chip's Power port	Power interface	Power

Table 5 - Ports and Interfaces

The module’s logical interfaces used for input data and control information are logically disconnected from the logical paths used for the output of data and status information by virtue of the module's API. The module’s API distinguishes all output data from SSP information.

The module communicates any error status synchronously through the use of its documented return codes, thus indicating the module’s status.

Caller-induced or internal errors do not reveal any sensitive material to callers. Cryptographic bypass capability is not supported by the module.

The module does not implement or support the use of a trusted channel.

[1] The module does not implement a Control Output Logical Interface

4 Roles, services, and authentication

The module supports two authorized roles: the User and the Crypto Officer. No support is provided for multiple concurrent operators or a maintenance operator.

The module authentication mechanism is defined by IG 4.4.A case 2 as follows. The User role is authenticated with the mechanism described in section 4.1. The User role can access the module via mailbox interface using the Device OS's XNU kernel. The User role can perform subset of services from Table 8. The Crypto Officer performs services from Table 8 and Table 9 that do not affect the module's security, per IG 4.1.A. The services are performed via mailbox interface using the Device OS's XNU kernel or via IPC channel using the software applications running on sepOS.

The Crypto Officer and User are assumed implicitly.

Role	Service	Input	Output
User	User keybag Services via Mailbox	User credential, reference to class C/A key from the user keybag	status (success/error)
	General Authentication service	User credential, reference to class C/A key from the user keybag	status (success/error)
	Generation of DEK	reference to class C/A key from the user keybag	wrapped DEK
	Backup keybag generation	N/A	status (success/error)
	Backup keybag service	wrapped DEK, reference to class C or A key from the user keybag	wrapped DEK
	Keychain DEK service using AK/ AKU/ AKPU/ CK/ CKU class key	pointer to AK/AKU/ AKPU/ CK/ CKU class key, wrapped DEK	unwrapped DEK
	Escrow keybag creation	N/A	status (success/error)
	Export keybag	reference to a keybag to be exported	keybag with HMAC tag
Crypto Officer (CO)	Device Wipe	N/A	N/A
	Show Status	N/A	status (success/error)
	Show Module Information	N/A	Module name and version
	Class D File System Services to wrap or unwrap DEK (Non-approved)	Pointer to Class D key from Backup keybag or Flash in SEP, wrapped or unwrapped DEK	wrapped or unwrapped file DEK
	Class D key service to encrypt or decrypt data (Non-approved)	Pointer to Class D key from Device or iCloud Keybag, plaintext or ciphertext data	ciphertext or plaintext data
	Class DK/DKU File System Services to wrap or unwrap keychain (Non-approved)	Pointer to Class DK/DKU key from Backup or User Keybag, wrapped or unwrapped keychain	wrapped or unwrapped file keychain
	Class DK/DKU key used for encrypting or decrypting of data (Non-approved)	Pointer to Class DK/DKU key from Device or iCloud Keybag, plaintext or ciphertext data	ciphertext or plaintext data
	Generate Ref-Keys (Non-approved)	N/A	status success/error, ref-key
	Signature generation using Ref-key (Non-approved)	pointer to ref-key, data	signed data
	Signature verification using Ref-key (Non-approved)	pointer to ref-key, signed data	verification result pass/error
	Encryption using Ref-key (Non-approved)	Pointer to ref key, data	ciphertext
	Decryption using Ref-key (Non-approved)	Ciphertext, Pointer to ref key	plaintext
	Generate Shared Secret using Ref-key (Non-approved)	pointer to ref-key, remote public key	shared secret

	Device Keybag Services for data encrypt or decrypt (Non-approved)	pointer to class key from device keybag, plaintext or ciphertext data	ciphertext or plaintext data
	iCloud Keybag services for data encrypt or decrypt (Non-approved)	pointer to class key from device keybag, plaintext during encryption or ciphertext data during decryption	ciphertext during encryption; plaintext data during decryption
	Escrow keybag service for key wrapping and unwrapping (Non-approved)	pointer to any key from Escrow keybag, plaintext key wrapping or wrapped key during unwrapping operation	wrapped key during wrapping; plaintext key during unwrapping
	Encrypt or Decrypt service using Class B Curve 25519 key from any keybag (Non-approved)	Pointer to class B key from any keybag, plaintext or ciphertext data	ciphertext and ephemeral public key during encryption; plaintext data during decryption
	Wrap or unwrap service for DEK or keychain using D/C/A Curve 25519 key from asymmetric keybag (non-approved)	Pointer to D/C/A key from asymmetric keybag, plaintext DEK or keychain during wrapping operation or wrapped DEK or keychain during unwrapping operation	wrapped DEK or keychain during wrapping; plaintext DEK or keychain during unwrapping
	Wrap and unwrap service for keychain using DK/DKU/CK/ CKU/AK/AKU/AKPU Ed25519 key from asymmetric keybag (Non-approved)	Pointer to DK/ DKU/ CK/ CKU/AK/ AKU/ AKPU key from asymmetric keybag, plaintext keychain during wrapping operation or wrapped keychain during unwrapping operation	wrapped keychain during wrapping; plaintext keychain during unwrapping
	Asymmetric (Ed25519) backup keybag wrap and unwrap (Non-approved)	Pointer to Ed 25519 key from backup keybag, plaintext or ciphertext data	ciphertext or plaintext data
	NVM Storage Controller Key Service (Non-approved)	pointer to NVM storage controller key, DEK	Wrapped DEK
	Elliptic Curve Integrated Encryption Scheme (ECIES) Encryption (Non-approved)	data, public key	encrypted data
	Elliptic Curve Integrated Encryption Scheme (ECIES) Decryption (Non-approved)	data, private key	decrypted data
	PBKDF Key Derivation (Non-approved)	password	derived key
	Filesystem DEK services (Non-approved)	wrapped DEK, class key reference from User keybag.	Wrapped DEK or Error
	Generation of DEK via IPC using class D key (Non-approved)	N/A	DEK wrapped with class D key
	Requesting backup keybag service vi IPC using class D key (Non-approved)	DEK wrapped with class D key	DEK wrapped with back up keybag key

Table 6 – Roles, Service Commands, Input and Output

4.1 Operator Authentication

Within the constraints of FIPS 140-3 level 2, the module implements a role-based authentication mechanism for authentication of the user role.

The module implements authenticated encryption-based mechanism in the following way: to request an authenticated service from the module the user must provide the credential and a reference to the class C or A keys of the user keybag[2] that is stored encrypted under SP800-38F AES Key Wrapping (AES-KW) within the module. The module performs obfuscation on the Operator provided credential and the resulting value -called REK (Root Encryption Key)- is used as the 256-bit AES key. Using this key, the module decrypts all the class C or A keys in the referenced user keybag with SP800-38F AES Key Unwrapping function (i.e. AES-KW-AD[3]). As AES-KW is an authentication cipher, the decryption operation will only succeed if there is no authentication error. If the user keybag can be successfully decrypted, the user is authenticated

[2] A keybag is a data structure used to store a collection of class keys. Each type (User, device, escrow, backup, or iCloud) has the same format.

[3] Section 6.2 SP800-38F, Algorithm 4: KW-AD(C)

to the module and the requested crypto service will then be proceeded with the decrypted user key. The failure of decrypting the user keybag is also a user authentication failure and the Operator will be denied access to the module.

The User keybags are configured in the module during factory install. Each User keybag consists of set of class C, A and D keys. Specifically, class C keys include C key, CK key, CKU keys and the class A keys include A key, AK key, AKU key and AKPU key. Only the class A or C keys are considered as approved. Any use of class D keys is considered as non-approved. The module maintains authenticated session from the time the User keybags are unwrapped until the power off. Upon power off, the unwrapped User keybags are zeroised and at the next power on the User credential needs to be provided again in order to unwrap the User keybag. All authentication data is provided electronically from the calling application/service and hence is not in visible form.

4.2 Strength of Authentication:

The AES-KW 256 bit key unwrapping function provides 256 bits of strength. Therefore, the strength of the authentication mechanism in use is 1/ 2^256. Even using a rate of 1µs per failed authentication, which would allow 60,000,000 consecutive attempts per minute (60s / 0.000001s), only provides a probability of successfully authenticating that is less than or equal to 60,000,000 * 1 / 2^256.

The SP 800-63B requirements are not applicable here based on the type of authentication mechanism deployed by the module because the authenticated decryption is not one of the methods listed in SP 800-63B.

Role	Authentication Method	Authentication Strength
User	AES-KW unwrapping function	256 bits
Crypto Officer (CO)	No authentication	N/A

Table 7– Roles and Authentication

4.3 Services

The Module has an Approved and non-Approved mode of operation. The Approved mode of Operation is assumed automatically without any specific configuration. If the device starts up successfully then the module has passed all self-tests and is operating in the Approved mode. Any calls to the non-Approved security functions listed in Table 9 will cause the module to assume the non-Approved mode of operation.

The module implements a dedicated API function to indicate if a requested service utilizes an approved security function. The approved service indicator utilizes one of two functions (fips_allowed and fips_allowed_mode) depending on the service in question. Calling fips_allowed_mode with AES-ECB, AES-CBC or AES-KW will return a zero to indicate it is an approved algorithm. Similarly, calling fips_allowed with any other approved algorithm will return zero. Calling either of these with an algorithm not listed in the Approved Algorithms Table will return a non-zero value, and as such indicates a non-approved service.

The table below lists all approved services that can be used in the approved mode of operation to authorized operators of either the User or Crypto Officer Roles. The abbreviations of the access rights to keys and SSPs have the following interpretation:

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 = **Zeroise**: The module zeroises the SSP.

N/A= Not Applicable: The service does not access any SSP during its operation

4.3.1 Approved Services

The table below includes the Approved Security Functions utilized by the service and Roles and access writes provided to the Keys and/or SSPs affected by the services. The last column provides a description of the service indicator reported by the service to show that the service utilizes an approved cryptographic algorithm, security function or process in an approved manner.

#	Service	Description	Approved Security Functions	Keys and/or SSPs	Role	Access rights to Keys and/ or SSPs	Indicator
1	User Keybag Services via Mailbox	Step 1. The module receives User credential and the reference to the class C or A key from the User keybag Step 2. Obfuscation operation is performed on the User provided credential resulting into a value called REK. Step 3. REK is used as a key for the AES KW operation to unwrap the referenced class A or C keys in the user keybag stored in the module. Step 4. Status of unwrapping operation of class keys is returned via mailbox interface and the REK is zeroised	Key Unwrapping: AES-KW	User credential, REK, User keybag (Class A key, Class AK key, Class AKU key, Class AKPU key, Class C key, Class CK key, Class CKU key)	User	W, E	0
2	General Authentication service	The module invokes the User keybag Services via Mailbox (i.e. #1 above)	Key Unwrapping: AES-KW	User credential, REK, User keybag (Class A key, Class AK key, Class AKU key, Class AKPU key, Class C key, Class CK key, Class CKU key)	User	W, E	0
3	Generation of Data Encryption Key (DEK)	Step 1: The module receives the reference to the class C or A key from the user keybag Step 2: The module generates a new DEK using the DRBG Step 3: Referenced class C or A key is used to wrap the DEK using AES-KW Step 4: Wrapped DEK is sent out of the module	Symmetric Key Generation (CKG using method in Section 4 example 1 [SP 800-133Rev2], AES-ECB, AES-CBC) Key Wrapping: AES-KW	Entropy input string, DRBG internal state	User	E	0
				User keybag (Class A key, Class AK key, Class AKU key, Class AKPU key, Class C key, Class CK key, Class CKU key)		W, E	
				DEK		G, E	
				Wrapped DEK		R	
4	Keychain DEK service using AK/ AKU/ AKPU/ CK/ CKU class key	Step 1. The module receives wrapped DEK (that was sent as part of service 3 above) and the pointer to class key AK/ AKU/AKPU/CK/ CKU from the user keybag. Step 2. Using the referenced class key, the module unwraps the DEK using AES-KW. If the class key is not available, an error is returned. Step 3. plaintext DEK is sent out to the User.	Key Wrapping: AES-KW	User keybag (Class A key, Class AK key, Class AKU key, Class AKPU key, Class C key, Class CK key, Class CKU key)	User	E	0
				DEK		R, E	
				Wrapped DEK		W, E	
5	Backup keybag generation	The module generates new set of back up keybags using the DRBG	Symmetric Key Generation (CKG using method in Section 4 example 1 [SP 800-133Rev2], AES-ECB, AES-CBC)	Entropy input string, DRBG internal state	User	E	0
				Backup keybag (Class A key, Class AK key, Class C key, Class CK key)		G, E	
6	Backup keybag service	Step 1. The module receives wrapped DEK and the class key reference for C and A from the user keybag. Step 2. Using the referenced class key, the module unwraps the DEK using AES-KW. If the class key is not available, an error is returned. Step 3. The module generates a set of backup key bag using DRBG Step 4. Unwrapped DEK is re-wrapped with	Key Wrapping and Unwrapping: AES-KW Symmetric Key Generation (CKG using method in Section 4 example 1 [SP 800-133Rev2], AES-ECB, AES-CBC)	DEK, User keybag (Class A key, Class AK key, Class AKU key, Class AKPU key, Class C key, Class CK key, Class CKU key)	User	W, E	0
				Entropy input string, DRBG internal state		E	

#	Service	Description	Approved Security Functions	Keys and/or SSPs	Role	Access rights to Keys and/or SSPs	Indicator
		backup key bag key using AES-KW Step 5. Wrapped DEK is sent out.		Wrapped DEK		R	
				Backup keybag (Class A key, Class AK key, Class C key, Class CK key) HMAC key		G, E	
7	Escrow keybag creation	The module generates new set of escrow key bag using the DRBG	Symmetric Key Generation (CKG using method in Section 4 example 1 [SP 800-133Rev2], AES-ECB, AES-CBC)	Entropy input string, DRBG internal state	User	E	0
				Escrow keybag (Class A key, Class AK key, Class AKU key, Class AKPU key, Class C key)		G,E	
8	Export Keybag	Step 1. The module receives reference to a keybag. Step 2: A HMAC key is taken as input based on the hardware specific data for the SKS Step 3: HMAC value is calculated on the entire referenced keybag that includes encrypted[4] keys. Step 4: HMAC is appended at the end of the keybag Step 5: Keybag with the appended HMAC is output to the User	Message Authentication HMAC	HMAC key	User	W, E	0
				Keybag to be exported (User or Backup or Escrow keybag)		R, E	
9	Device Wipe	Erase all content (Factory Reset)	N/A	All SSPs	CO	Z	N/A
10	Show Status	N/A	N/A	N/A	CO	N/A	N/A
11	Show Module Information	N/A	N/A	N/A	CO	N/A	N/A
12	Perform Self-Test	Perform all pre-operational self-tests and cryptographic algorithm self-tests (CASTs)	All	N/A	CO	N/A	N/A

Table 8 - Approved Services

4.3.2 Non-Approved Services and non-authenticated services

The table below lists all non-Approved services that can only be used in the non-Approved mode of operation and the services are non-authenticated.

Service	Description	Algorithms Accessed	Role	Indicator
Class D File System Services to wrap or unwrap DEK	Wrapping of provided plaintext DEK or unwrapping of provided wrapped DEK using class D key from Backup keybag or secure storage in SEP	AES-KW	CO	non-zero value
Class D key service to encrypt or decrypt data	Encryption of provided plaintext or decryption of provided ciphertext using class D key from Device or iCloud Keybag	AES-KW	CO	non-zero value
Class DK/DKU File System Services to wrap or unwrap keychain	Wrapping of provided plaintext keychain or unwrapping of provided wrapped keychain using class DK/DKU key from Backup keybag or User keybag	AES-KW	CO	non-zero value
Class DK/DKU key service for data encrypt or decrypt	Encryption of provided plaintext or decryption of provided ciphertext using DK/DKU key from Device or iCloud keybag	AES-KW	CO	non-zero value

[4] Note: only class A and C keys in the keybag are encrypted with REK whereas class D keys are in plaintext as they are non-approved and not considered as CSP

Service	Description	Algorithms Accessed	Role	Indicator
Generate Ref-Keys	Key Generation	ECDSA KeyGen	CO	non-zero value
Sign and verify using Ref-key	Signature Generation and Verification	ECDSA SigGen, ECDSA SigVer	CO	non-zero value
Encryption and decryption using Ref-key	shared secret is generated using user provided key and existing ref key followed by HKDF is applied to derive a key which is used to encrypt the provided plaintext or decrypt the provided ciphertext	ECDSA HKDF AES-GCM AES-KW	CO	non-zero value
Generate Shared Secret using Ref-key	Shared secret generation	ECDH	CO	non-zero value
Device keybag service for data encrypt or decrypt	Encryption of provided plaintext or decryption of provided ciphertext using any key from Device keybag	AES-KW	CO	non-zero value
iCloud keybag service for data encrypt or decrypt	Encryption of provided plaintext or decryption of provided ciphertext using any key from iCloud keybag	AES-KW	CO	non-zero value
Escrow keybag service for key wrapping and unwrapping	Wrapping of provided plaintext key or unwrapping of provided wrapped key using any key from Escrow keybag	AES-KW	CO	non-zero value
Encrypt or Decrypt service using Class B Curve 22519 key from any key bag	shared secret is computed by generating new ephemeral keypair and existing Curve25519 key followed by HKDF is applied to derive a key which is used for data encryption or decryption. During encryption operations, the wrapped key and the ephemeral public key are sent to the user	AES-KW HKDF Curve 25519	CO	non-zero value
Wrap or unwrap service for DEK or keychain using any Curve 22519 key from asymmetric key bag	shared secret is computed by generating new ephemeral keypair and existing Curve25519 key followed by HKDF is applied to derive a key which is used to wrap and unwrap DEK or keychain. During wrapping operation, the wrapped key and the ephemeral public key are sent to the user	AES-KW HKDF Curve 25519	CO	non-zero value
Asymmetric (Ed25519) backup keybag wrap and unwrap	shared secret is computed by generating new ephemeral keypair and existing Curve25519 key followed by HKDF is applied to derive a key which is used to wrap and unwrap. The wrapped key and the ephemeral public key are sent to the user	AES-KW HKDF Ed25519	CO	non-zero value
Wrap or unwrap service for keychain using DK/DKU/CK/CKU/AK/AKU/AKPU Ed25519 key from asymmetric key bag (Non-approved)	Pointer to DK/DKU/CK/CKU/AK/AKU/AKPU key from asymmetric keybag, plaintext keychain during wrapping operation or wrapped keychain during unwrapping operation	AES-KW HKDF Ed25519	CO	non-zero value
NVM Storage Controller Key	wrapping DEK using NVM storage controller key	AES KW	CO	non-zero value
Elliptic Curve Integrated Encryption Scheme (ECIES) Encryption	Encryption	ECDH AES-GCM ANSI X9.63 Key Derivation	CO	non-zero value
Elliptic Curve Integrated Encryption Scheme (ECIES) Decryption	Decryption	ECDH AES-GCM ANSI X9.63 Key Derivation	CO	non-zero value
PBKDF Key Derivation	Hash-based Key Derivation	PBKDF	CO	non-zero value
File system DEK service	Unwrap the DEK using referenced class key and re-wrap using NVM storage controller key	AES KW	CO	non-zero value
Generation of DEK via IPC using class D key	Requesting generate DEK service via IPC Channel using class D keys	AES KW DRBG	CO	non-zero value
Requesting backup keybag service via IPC using class D key	Requesting backup keybag service via IPC Channel using class D keys	AES KW DRBG	CO	non-zero value

Table 9 - Non-Approved and non-authenticated Services

5 Software/Firmware security

5.1 Integrity Techniques

The Apple corecrypto Module v11.1 [Apple silicon, Secure Key Store, Hardware, SL2/PHY3] is in the form of binary executable code. A firmware integrity test is performed on the runtime image of the module. The HMAC-SHA256 implemented in the module is used as an approved algorithm for the integrity test. If the test fails, the module enters an error state where no cryptographic services are provided, and data output is prohibited i.e., the module is not operational.

5.2 On-Demand Integrity Test

The Integrity tests are performed as part of the Pre-Operational Self-Tests. It is automatically executed at power-on.

6 Operational Environment

The Apple corecrypto Module v11.1 [Apple silicon, Secure Key Store, Hardware, SL2/PHY3] operates in a limited operational environment per FIPS 140-3 security level 2 specifications. The module operates within the sepOS execution environment which is separate from the Device OS execution environment. The SEP operating system provides memory isolation between all applications executing on it. The Device OS is unable to access the module's memory or observe the module's operation.

7 Physical Security

The defined physical boundary of the Apple corecrypto Module v11.1 [Apple silicon, Secure Key Store, Hardware, SL2/PHY3] is the entire System-on-Chip (SoC) listed in Table 2. Consequently, the physical embodiment of each SoC is a single-chip cryptographic module.

The hardware module conforms to the Level 3 requirements for physical security as detailed in Table 10.

Physical Security Mechanism	Recommended Frequency of Inspection/Text	Inspection/Test Guidance Details
Production Grade Components that include standard passivation	No operator-performed testing is recommended	N/A
- Tamper-Evident coating or black hard coated material or metal coating, - The Ball Grid Array (BGA back side of the SoC soldered on the logic board.) The components listed above are opaque within the visible spectrum.	No operator-performed testing is recommended	N/A
Hardness of the coating tested in the module's intended temperature range of operation	No operator-performed testing is recommended	N/A
Environmental Failure Protection (EFP) forces the module to shut down	No operator-performed testing is recommended	N/A

Table 10 – Physical Security Inspection Guidelines

The module correctly implements the Environmental Failure Protection (EFP) features as detailed in Table 11.

	Temperature or voltage measurement	Specify EFP or EFT	Specify if this condition results in a shutdown or zeroisation
Low Temperature, High Temperature	Values found in Apple proprietary document	EFP	shutdown
Low Voltage, High Voltage	Values found in Apple proprietary document	EFP	shutdown

Table 11 – EFP/EFT

	Hardness tested temperature measurement
Low Temperature (˚C)	-25˚C
High Temperature (˚C)	+ 51˚C

Table 12 – Hardness testing temperature ranges

8 Non-invasive Security

Currently, the non-invasive security is not required by FIPS 140-3 (see NIST SP 800-140F). The requirements of this area are not applicable to the module.

9 Sensitive Security Parameter Management

The following table summarizes the keys and Sensitive Security Parameters (SSPs) that are used by the cryptographic services implemented in the module:

Key / SSP Name / Type	Strength	Security Function and Cert. Number	Generation	Import / Export	Establishment (see section 9.3	Storage	Zeroisation	Use & related keys (Service # in section 4.3.1)
Class A, Class C, Class AK, Class AKU, Class CK, Class CKU in User Keybag (AES keys)	128, 192, 256-bits	AES-KW with CAVP Certs. # A1343, A1345 (for services 1,2,3,4, 6)	N/A: Preloaded at factory	Entry: N/A Output: encrypted using AES-KW for service #8 only	N/A	Flash	Device Wipe;	1,2,3,4,6,8
Class A, Class C, Class AK, Class AKU, Class CK, Class CKU keys in backup keybag (AES keys)	128, 192, 256-bits	CTR_DRBG with CAVP Certs. # A501, A1362 (for services 3, 5, 6, 7) AES-KW with CAVP Certs. # A1343, A1345 (for services 1,2,3,4, 6)	Generated using direct output of CTR DRBG compliant to section 4 example 1 of SP800-133r2. CKG (vendor affirmed)	Entry: N/A Output: encrypted using AES-KW for service #8 only	N/A	RAM	Context object destruction; Device Wipe	5,6,8
Class A, Class C, Class AK, Class AKU, Class CK, Class CKU keys in escrow keybag (AES keys)	128, 192, 256-bits		Generated using direct output of CTR DRBG compliant to section 4 example 1 of SP800-133r2. CKG (vendor affirmed)	Entry: N/A Output: encrypted using AES-KW for service #8 only	N/A	RAM	Context object destruction; Device Wipe	7,8
Data Encryption Key (DEK) (AES key)	128, 192, 256-bits		Symmetric key generation services of the module using DRBG compliant to section 4 example 1 of SP800-133r2. CKG (vendor affirmed)	Entry: In encrypted form Output in encrypted form via service 3/6, or plaintext via service 4	N/A	RAM	Context object destruction; Device Wipe	3,4, 6
Entropy input string	256-bits	Random Number Generation ESV E#113	Obtained from the physical entropy source	No import No export	N/A	RAM	Device Wipe	3,5,6,7
DRBG internal state: V value, key, and seed material	256-bits	Random Number Generation CTR_DRBG with CAVP Certs. # A501, A1362	Updated during DRBG initialization	No import No export	N/A	RAM	Device Wipe	3,5,6,7
HMAC Key (Message Authentication Key)	112 bits or greater	HMAC-SHA-256 CAVP Certs. # A1340, A1341, A1345	N/A	Entry: taken as input based on the hardware specific data Output: N/A	N/A	RAM	Context object destruction; Device Wipe	8
User Credential	N/A	N/A	N/A	Entry: input by User Output: N/A	N/A	RAM	Device Wipe	1,2
REK	256-bits	N/A	N/A	Entry: N/A Output: N/A	N/A	RAM	Device Wipe	1,2

Table 13 - SSPs

9.1 Random Number Generation

A [SP800-90ARev1] approved deterministic random bit generator based on block cipher is used: CTR_DRBG using AES-256 without derivation function and with prediction resistance. The random numbers used for key generation are all generated by CTR_DRBG in this module. Per section 10.2.1.1 of [SP 800-90ARev1], the internal state of CTR_DRBG consists of the V, Key and a seed.

The module also performs DRBG health tests according to section 11.3 of [SP800-90ARev1].

In accordance with FIPS 140-3 IG D.L, the 'Entropy input string', 'seed', 'DRBG internal state (V and key values)' are considered CSPs by the module.

No non-DRBG functions or instances are able to access the DRBG internal state.

The deterministic random bit generators are seeded by an internal physical noise source. The hardware based entropy source provides 256-bits of security strength in instantiating and reseeding the module approved DRBGs.

Entropy Source	Minimum number of bits of entropy	Details
ESV #E113 (physical entropy source)	256	The entropy source consists of twenty-four Free Ring Oscillator (FROs). The entropy source has been shown to provide full 256-bits of entropy at the output of the vetted conditioning function, SHA2-256 (#C1223).

Table 14 - Non-Deterministic Random Number Generation Specification

9.2 Key / SSP Generation

The module provides a key generation service for symmetric cipher i.e. AES in accordance with FIPS 140-3 IG D.H. The cryptographic module performs Cryptographic Key Generation (CKG) for symmetric keys as per section 4 [SP800-133r2]. The implementation follows example 1 from Section 4 whereby V is a string of binary zeroes, such that B = U (i.e., the output of an approved RBG). The symmetric keys are generated directly output from an approved DRBG compliant with [SP800-90ARev1].

9.3 Keys/SSPs Establishment

The module provides the following key/SSP establishment service in the Approved mode:

- AES-Key Wrapping: The module implements a Key Transport Scheme (KTS) using AES-KW compliant to [SP800-38F], per IG D.G. The SSP establishment methodology provides between 128 and 256 bits of encryption strength.

9.4 Keys/SSPs Import/Export

Per the definition in IG 2.3.B, "*Transferring SSPs including the entropy input between a sub-chip cryptographic subsystem and an intervening functional subsystem for Security Levels 1 and 2 on the same single chip is considered as not having Sensitive Security Parameter Establishment crossing the HMI*". As such, the import or export Keys/SSP as defined in Table 1 of IG 9.5.A do not apply.

Within the TOEPP, keys and SSPs can either be entered into, or output from the Apple Secure Key Store Cryptographic Module to/from intervening functional subsystems in plaintext.

9.5 Keys/SSPs Storage

During runtime operation, the Apple corecrypto Module v11.1 [Apple silicon, Secure Key Store, Hardware, SL2/PHY3] module stores keys/SSPs in volatile memory, except for the user keybag that is stored in Flash. The module protects all keys/SSPs through the memory separation and protection mechanisms provided by the operating system while the Flash component only provides exclusive access to the module. No process other than the module itself can access the keys/SSPs in its process memory or Flash component.

9.6 Keys/SSPs Zeroization

Keys and SSPs (including temporary SSPs) are zeroised when the appropriate context object is destroyed by overwriting the entire context object with all zeros. Zeroization occurs at the end of an API function that uses the CSPs.

Zeroization is also performed by calling the "Device Wipe" service. The "Device Wipe" service performs end of life of the device.

Input and output interfaces are inhibited while zeroisation is performed. Zeroisation is immediate and uninterruptible, preventing the retrieval and reuse of the zeroised values. The module provides an implicit indication that the zeroisation has successfully completed by returning access to the User, ready to service the next request.

10 Self-tests

The module performs pre-operational self-tests automatically when the module is loaded into memory; the pre-operational self-tests triggered at power-on ensure that the module is not corrupted and that the cryptographic algorithms work as expected. The module transitions to approved Mode upon successful completion of the pre-operational self-tests and CASTs.

FIPS 140-3 only requires that software/firmware integrity test(s) and the requisite cryptographic algorithm(s) be tested during power-up, but the Apple corecrypto Module v11.1 [Apple silicon, Secure Key Store, Hardware, SL2/PHY3] runs all Cryptographic Algorithm Self-Tests (CASTs) during power-up as well.

The following tests (Table 15) are performed each time the Apple corecrypto Module v11.1 [Apple silicon, Secure Key Store, Hardware, SL2/PHY3] starts. If any of the following tests fail the device fails to startup.

While the module is executing the self-tests, services are not available, and input and output are inhibited.

Cryptographic Algorithm	Notes
HMAC-SHA256	CAST (KAT) performed prior to module's firmware integrity test at power-up
Pre-operational Integrity Test using HMAC-SHA-256	Integrity Test of module's firmware
AES Implementations AES-KW, AES-CBC, AES-ECB using 128-bit key	Separate encryption / decryption CAST (KAT) performed for each mode
CTR_DRBG with 256-bit key	CAST (KAT) performed CAST: Health test per SP800-90ARev1 section 11.3
HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-512	CAST (KAT) performed
SHA-1, SHA-256, SHA-512	Covered by HMAC CAST
Physical entropy source	SP800-90B health test (APT and RCT) classified as CAST: - at start-up: performed on 1,024 consecutive samples. - during runtime.

Table 15 - Self-Tests

10.1 Pre-Operational Integrity Test

A pre-operational integrity test is performed on the firmware component of the Apple corecrypto Module v11.1 [Apple silicon, Secure Key Store, Hardware, SL2/PHY3]. The module's HMAC-SHA2-256 is used as an approved algorithm for the integrity test. The module performs a cryptographic algorithm self-test using a KAT on the HMAC algorithm prior to performing the pre-operational integrity test. If the test fails, then the module enters an Error State. The HMAC value is pre-computed at build time and stored in the module. The HMAC value is recalculated during runtime and compared with the stored value.

10.2 Conditional Self-Tests

The following sub-sections describe the conditional self-tests supported by the Apple corecrypto Module v11.1 [Apple silicon, Secure Key Store, Hardware, SL2/PHY3].

10.2.1 Cryptographic algorithm self-tests

The Apple corecrypto Module v11.1 [Apple silicon, Secure Key Store, Hardware, SL2/PHY3] runs all Cryptographic Algorithm Self-Tests during power-up. These tests are detailed in Table 15.

10.2.2 Pairwise Consistency Test

The Apple corecrypto Module v11.1 [Apple silicon, Secure Key Store, Hardware, SL2/PHY3] does not provide asymmetric key generation service in the approved mode. Therefore, this section is not applicable.

10.2.3 On-Demand Self-Test

On demand and periodic self-tests are performed by powering off the module and powering it on again. This service performs the same cryptographic algorithm tests executed during pre-operational self-tests and CASTs. During the execution of the periodic and on-demand self-tests, crypto services are not available and no data output or input is possible.

10.3 Error Handling

If any of the self-tests described in Sections 10.1, 10.2.1 or 10.2.2 fail, the module reports the cause of the error and enters an error state. In the Error State, no cryptographic services are provided, and data output is prohibited. The only method to recover from the error state is to power cycle the device which results in the module restarting and re-performing the pre-operational firmware integrity test and the Conditional Self-Test CASTs. The module will only enter into the operational state after successfully passing the pre-operational firmware integrity test and the CASTs. The table below shows the different causes that lead to the Error State and the status indicators reported.

Cause of Error	Error Indicator
Failed Pre-operational Software Integrity Test	Error message “FAILED: fipspost_post_integrity” sent to caller
Failed CAST	Error message “FAILED:<event>” sent to caller (<event> refers to any of the cryptographic functions listed in Table 15)

Table 16 – Error States

11 Life-cycle assurance

11.1 Delivery and Operation

The module's firmware with the sepOS is delivered as part of the Device OS image.

The vendor's internal development process guarantees that the correct version of module goes with its intended Device OS version. For additional assurance, the module is digitally signed by the vendor, and it is verified during the integration into Device OS.

This digital signature-based integrity protection during the delivery/ integration process is not to be confused with the HMAC-SHA-256 based integrity check performed by the module itself as part of its pre-operational self-tests.

The biometric authentication option provided by the underlying test platform shall be disabled in order to run the module in the FIPS validated manner.

11.2 Crypto Officer Guidance

The ESV Public Use Document (PUD) reference for physical entropy source is published at https://csrc.nist.gov/projects/cryptographic-module-validation-program/entropy-validations/certificate/113

The Approved mode of operation is configured in the system by default and can only be transitioned into the non-Approved mode by calling one of the non-Approved services listed in Table - Non-Approved and non-authenticated Services. If the device starts up successfully, then the module has passed all self-tests and is operating in the Approved mode.

A Crypto Officer Role Guide is provided by Apple which offers IT System Administrators with the necessary technical information to ensure FIPS 140-3 Compliance of the deployed systems. This guide walks the reader through the system's assertion of cryptographic module integrity and the steps necessary if module integrity requires remediation. A link to the Guide can be found on the Product security, validations, and guidance page found in [Device OS].

11.3 User Guidance

The User role is authenticated with the mechanism described in section 4.1. The User role can access the module via mailbox interface using the Device OS's XNU kernel. The User role can perform subset of services from Table 8.

As stated in the Crypto Officer Guidance, the Approved mode of operation is configured in the system by default and can only be transitioned into the non-Approved mode by calling one of the non-Approved services listed in Table - Non-Approved and non-authenticated Services. This transition cannot be made by the User directly, as all non-approved services require an implicit transition into the Crypto-Officer role. Any calling of such services is therefore implicitly performed by the Crypto Officer. If the device starts up successfully, then the module has passed all self-tests and is operating in the Approved mode.

When performing a Device Wipe service to erase all content of the module, the procedure must be performed under the control of the Operator.

12 Mitigation of other attacks

The module does not claim mitigation of other attacks.

Appendix A. Glossary and Abbreviations

AES	Advanced Encryption Standard
API	Application Programming Interfaces
APT	Adaptive Proportion Test (SP800-90B health test)
BGA	Ball Grid Array (Physical Security)
CAVP	Cryptographic Algorithm Validation Program
CBC	Cipher Block Chaining
CCM	Counter with Cipher Block Chaining-Message Authentication Code
CMVP	Cryptographic Module Validation Program
CST	Cryptographic and Security Testing
CTR	Counter Mode
DEK	Data Encryption Key
DRBG	Deterministic Random Bit Generator
ECB	Electronic Code Book
ECDSA	DSA (Digital Signature Algorithm) based on Elliptic Curve Cryptography (ECC)
EMI	Electromagnetic Interference (Physical Security)
ESV	Entropy Source Validation
FIPS	Federal Information Processing Standards Publication
GCM	Galois Counter Mode
HMAC	Hash Message Authentication Code
IPC	Inter-Process Communication
IHS	Integrated Heat Spreader (Physical Security)
KAT	Known Answer Test
KDF	Key Derivation Function
KEK	Key Encryption Key
KW	AES Key Wrap
MAC	Message Authentication Code
NIST	National Institute of Science and Technology
NVM	Non-Volatile Memory
OFB	Output Feedback
OS	Operating System
PBKDF	Password Based Key Derivation Function
RCT	Repetition Count Test (SP800-90B health test)
SEP	Secure Enclave Processor
SHA	Secure Hash Algorithm
SHS	Secure Hash Standard
SKS	Secure Key Store
SoC	System on Chip
SSP	Sensitive Security Parameters

Appendix B. References

FIPS140-3	FIPS PUB 140-3 - Security Requirements for Cryptographic Modules March 2019 https://doi.org/10.6028/NIST.FIPS.140-3
SP 800-140x	CMVP FIPS 140-3 Related Reference https://csrc.nist.gov/Projects/cryptographic-module-validation-program/fips-140-3-standards
FIPS140-3_IG	Implementation Guidance for FIPS PUB 140-3 and the Cryptographic Module Validation Program September 2020 https://csrc.nist.gov/Projects/cryptographic-module-validation-program/fips-140-3-ig-announcements
FIPS140-3_MM	CMVP FIPS 140-3 Draft Management Manual https://csrc.nist.gov/CSRC/media/Projects/cryptographic-module-validation-program/documents/fips%20140-3/Draft%20FIPS-140-3-CMVP%20Management%20Manual%2009-18-2020.pdf
SP 800-140	FIPS 140-3 Derived Test Requirements (DTR) https://csrc.nist.gov/publications/detail/sp/800-140/final
SP 800-140A	CMVP Documentation Requirements https://csrc.nist.gov/publications/detail/sp/800-140a/final
SP 800-140B	CMVP Security Policy Requirements https://csrc.nist.gov/publications/detail/sp/800-140b/final
SP 800-140C	CMVP Approved Security Functions https://csrc.nist.gov/publications/detail/sp/800-140c/final
SP 800-140D	CMVP Approved Sensitive Security Parameter Generation and Establishment Methods https://csrc.nist.gov/publications/detail/sp/800-140d/final
SP 800-140E	CMVP Approved Authentication Mechanisms https://csrc.nist.gov/publications/detail/sp/800-140e/final
SP 800-140F	CMVP Approved Non-Invasive Attack Mitigation Test Metrics https://csrc.nist.gov/publications/detail/sp/800-140f/final
FIPS180-4	Secure Hash Standard (SHS) March 2012 http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf
FIPS186-4	Digital Signature Standard (DSS) July 2013 http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf
FIPS197	Advanced Encryption Standard November 2001 http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf
FIPS198-1	The Keyed Hash Message Authentication Code (HMAC) July 2008 http://csrc.nist.gov/publications/fips/fips198-1/FIPS-198-1_final.pdf
RFC3394	Advanced Encryption Standard (AES) Key Wrap Algorithm September 2002 http://www.ietf.org/rfc/rfc3394.txt

RFC5649	Advanced Encryption Standard (AES) Key Wrap with Padding Algorithm September 2009 http://www.ietf.org/rfc/rfc5649.txt
SP800-38A	NIST Special Publication 800-38A - Recommendation for Block Cipher Modes of Operation Methods and Techniques December 2001 http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf
SP800-38C	NIST Special Publication 800-38C - Recommendation for Block Cipher Modes of Operation: the CCM Mode for Authentication and Confidentiality May 2004 http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38c.pdf
SP800-38D	NIST Special Publication 800-38D - Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC November 2007 http://csrc.nist.gov/publications/nistpubs/800-38D/SP-800-38D.pdf
SP800-38F	NIST Special Publication 800-38F - Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping December 2012 http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38F.pdf
SP800-56Cr2	Recommendation for Key-Derivation Methods in Key-Establishment Schemes August 2020 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Cr2.pdf
SP800-57	NIST Special Publication 800-57 Part 1 Revision 5 - Recommendation for Key Management Part 1: General May 2020 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r5.pdf
SP800-67	NIST Special Publication 800-67 Revision 1 - Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher January 2012 http://csrc.nist.gov/publications/nistpubs/800-67-Rev1/SP-800-67-Rev1.pdf
SP800-90ARev1	NIST Special Publication 800-90A - Revision 1 - Recommendation for Random Number Generation Using Deterministic Random Bit Generators June 2015 http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf
SP800-90B	NIST Special Publication 800-90B - Recommendation for the Entropy Sources Used for Random Bit Generation January 2018 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90B.pdf
SP800-108	NIST Special Publication 800-108 - Recommendation for Key Derivation Using Pseudorandom Functions (Revised) October 2009 http://csrc.nist.gov/publications/nistpubs/800-108/sp800-108.pdf
SP800-131Ar2	Transitioning the Use of Cryptographic Algorithms and Key Lengths March 2019 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-131Ar2.pdf

SP800-132	NIST Special Publication 800-132 - Recommendation for Password-Based Key Derivation - Part 1: Storage Applications December 2010 http://csrc.nist.gov/publications/nistpubs/800-132/nist-sp800-132.pdf
SP800-133r2	Recommendation for Cryptographic Key Generation June 2020 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-133r2.pdf
Developer	Device OS Technical Overview https://developer.apple.com
SEC	Apple Platform Security https://support.apple.com/guide/security/welcome/web https://manuals.info.apple.com/MANUALS/1000/MA1902/en_US/apple-platform-security-guide.pdf
Device OS	Product security certifications for Device OS https://support.apple.com/en-gw/guide/certifications/welcome/web