Apple Inc.

Apple Secure Key Store Cryptographic Module, v9.0
FIPS 140-2 Non-Proprietary Security Policy

Hardware Versions:
1.2, 2.0
Firmware version:
SEPOS

July, 2019

Prepared for:
Apple Inc.
One Apple Park Way
Cupertino, CA 95014
www.apple.com

Prepared by:
atsec information security Corp.
9130 Jollyville Road, Suite 260
Austin, TX 78759
www.atsec.com

Table of Contents

List of Tables

List of Figures

1 Introduction

1.1 Purpose

This document is a non-proprietary Security Policy for the Apple Secure Key Store Cryptographic Module, v9.0. It describes the module and the FIPS 140-2 cryptographic services it provides. This document also defines the FIPS 140-2 security rules for operating the module.

This document was prepared in fulfillment of the FIPS 140-2 requirements for cryptographic modules and is intended for security officers, developers, system administrators, and end-users.

FIPS 140-2 details the requirements of the Governments of the U.S. and Canada for cryptographic modules, aimed at the objective of protecting sensitive but unclassified information.

For more information on the FIPS 140-2 standard and validation program please refer to the NIST website at https://csrc.nist.gov/Projects/Cryptographic-Module-Validation-Program.

Throughout the document “Apple Secure Key Store Cryptographic Module, v9.0.” “cryptographic module”, “SKS” or “the module” are used interchangeably to refer to the Apple Secure Key Store Cryptographic Module, v9.0.

1.2 Document Organization / Copyright

This non-proprietary Security Policy document may be reproduced and distributed only in its original entirety without any revision, © 2019 Apple Inc.

1.3 External Resources / References

The Apple website (http://www.apple.com) contains information on the full line of products from Apple Inc. For a detailed overview of the operating system Apple Secure Key Store (SKS) and the Secure Enclave Processor (SEP) and its security properties refer to [iOS] and [SEC]. For details on iOS releases with their corresponding validated modules and Crypto Officer Role Guides refer to the Apple Knowledge Base Article HT202739 – “ Product security certifications, validations, and guidance for SEP” (https://support.apple.com/en-us/HT202739)

The Cryptographic Module Validation Program website (https://csrc.nist.gov/projects/cryptographic-module-validation-program) contains links to the FIPS 140-2 certificate and Apple Inc. contact information.

1.3.1 Additional References

FIPS 140-2	Federal Information Processing Standards Publication, “FIPS PUB 140-2 Security Requirements for Cryptographic Modules,” Issued May-25-2001, Effective 15-Nov-2001, Location: https://csrc.nist.gov/projects/cryptographic-module-validation-program/standards
FIPS 140-2 IG	NIST, “Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program,” February 5, 2019 Location: https://csrc.nist.gov/projects/cryptographic-module-validation-program/standards
FIPS 180-4	Federal Information Processing Standards Publication 180-4, March 2012, Secure Hash Standard (SHS)
FIPS 186-4	Federal Information Processing Standards Publication 186-4, July 2013, Digital Signature Standard (DSS)
FIPS 197	Federal Information Processing Standards Publication 197, November 26, 2001 Announcing the ADVANCED ENCRYPTION STANDARD (AES)

FIPS 198	Federal Information Processing Standards Publication 198, July, 2008 The Keyed-Hash Message Authentication Code (HMAC)
SP800-38 A	NIST Special Publication 800-38A, “Recommendation for Block Cipher Modes of Operation”, December 2001
SP800-38 D	NIST Special Publication 800-38D, “Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC”, November 2007
SP800-38 E	NIST Special Publication 800-38E, “ Recommendation for Block Cipher Modes of Operation: The XTS-AES Mode for Confidentiality on Storage Devices”, January 2010
SP800-38 F	NIST Special Publication 800-38F, “Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping”, December 2012
SP800-56A	NIST Special Publication 800-56A Revision 2, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography” May 2013
SP800-56B	NIST Special Publication 800-56B Revision 1, “Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography” September 2014
SP800-57P1	NIST Special Publication 800-57, “Recommendation for Key Management – Part 1: General (Revised),” July 2012
SP 800-90A	NIST Special Publication 800-90A, “Recommendation for Random Number Generation Using Deterministic Random Bit Generators,” January 2012
SP800-108	NIST Special Publication 800-108, “Recommendation for Key Derivation Using Pseudorandom Functions”, October 2009
SP800-132	NIST Special Publication 800-132, “Recommendation for Password-Based Key Derivation”, December 2010
SEC	Security Overview Location: http://developer.apple.com/library/ios/#documentation/Security/Conceptual/Security_Overview/Introduction/Introduction.html
iOS	iOS Technical Overview Location: http://developer.apple.com/library/ios/#documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/Introduction/Introduction.html#//apple_ref/doc/uid/TP40007898
UG	User Guide Location: https://www.apple.com/business/docs/iOS_Security_Guide.pdf

1.4 Acronyms

Acronyms found in this document are defined as follows:

AES	Advanced Encryption Standard
AKS	Apple Key Store
API	Application Programming Interface
CAVP	Cryptographic Algorithm Validation Program
CBC	Cipher Block Chaining mode of operation
CMVP	Cryptographic Module Validation Program
CSP	Critical Security Parameter
DEK	Data Encryption Key
DRBG	Deterministic Random Bit Generator
ECB	Electronic Codebook mode of operation
ECC	Elliptic Curve Cryptography
EC Diffie-Hellman	DH based on ECC
ECDSA	DSA based on ECC
EMC	Electromagnetic Compatibility
EMI	Electromagnetic Interference
FIPS	Federal Information Processing Standard
GCM	Galois/Counter Mode
HMAC	Hash-based Message Authentication Code
KAT	Known Answer Test
KDF	Key Derivation Function
KEK	Key Encryption Key
KS	Key Size (Length)
NIST	National Institute of Standards and Technology
OS	Operating System
PBKDF	Password-based Key Derivation Function
PCT	Pair-wise Consistency Test
REK	Root Encryption Key
RNG	Random Number Generator
SHS	Secure Hash Standard
SEP	Secure Enclave Coprocessor
SiP	System in Package
SKS	Secure Key Store
SoC	System on Chip
Triple-DES	Triple Data Encryption Standard

2 Cryptographic Module Specification

2.1 Module Description

The Apple Secure Key Store Cryptographic Module, v9.0 is a hardware cryptographic module implemented as a sub-chip running on a single-chip processor.

The cryptographic services provided by the module are:

- data encryption / decryption
- generation of hash values
- key wrapping
- random number generation
- key generation
- key derivation

2.1.1 Module Validation Level

The module is intended to meet requirements of FIPS 140-2 security level 2 overall. The following table shows the security level for each of the eleven requirement areas of the validation.

FIPS 140-2 Security Requirement Area	Security Level
Cryptographic Module Specification	2
Cryptographic Module Ports and Interfaces	2
Roles, Services and Authentication	2
Finite State Model	2
Physical Security	2
Operational Environment	N/A
Cryptographic Key Management	2
EMI/EMC	2
Self-Tests	2
Design Assurance	2
Mitigation of Other Attacks	N/A

Table 1: Module Validation Level

2.1.2 Module Components

In the following sections the components of the Apple Secure Key Store Cryptographic Module, v9.0 are listed in detail. The module consists of both firmware and a hardware component. The module's firmware operates within the SEPOS execution environment which is separate from the iOS execution environment -- the SEPOS execution environment is driven by its own CPU and uses isolated memory. Both execution environments are physically separated on the SoC and thus execute independently of each other. The execution environment of the cryptographic module is SEPOS.

The two modules covered by this security policy are:

- Hardware DRBG version 1.2 with the operating system outlined in table 2a
- Hardware DRBG version 2.0 with the operating system outlined in table 2b

2.1.2.1 Firmware components

The SKS application linking with CoreCrypto is the cryptographic module. The firmware boundary is defined as the API offered by the module's mailbox interface to callers from the iOS execution environment. SKS has an API layer that provides consistent interfaces to the supported services

and therefore the supported cryptographic algorithms. These implementations include proprietary optimizations of algorithms that are fitted into the SEP framework. In addition, the module provides Interprocess Communication (IPC) interfaces to other applications executing within the SEPOS execution environment.

2.1.2.2 Hardware components

The cryptographic module boundary includes a DRBG hardware component with an AES and SHA hardware accelerator as part of the module which is integrated into the SoC and is reachable by the SEP execution environment. The module hardware version is v1.2 for hardware DRBG in Apple A7, Apple A8 and Apple 8X SoCs and v2.0 found in all others.

2.1.3 Tested Platforms

The module has been tested with and without PAA[1] on the following platforms:

Model	Operating System
iPhone 5S with Apple A7 CPU	SEPOS for A7 under iOS 12
iPhone 6 with Apple A8 CPU (iPhone 6 and iPhone 6 Plus)	SEPOS for A8 under iOS 12
iPad Air 2 with Apple A8X CPU	SEPOS for A8X under iOS 12

Table 2a: Tested Platforms with hardware DRBG v1.2

Model	Operating System
iPhone 6S with Apple A9 CPU (iPhone 6S and iPhone6S Plus)	SEPOS for A9 under iOS 12
iPhone 7 with Apple A10[2] Fusion CPU (iPhone 7 and iPhone 7 Plus)	SEPOS for A10 under iOS 12
iPhone 8 and iPhone X with Apple A11[3] Bionic CPU (iPhone 8, iPhone 8 Plus, iPhone X)	SEPOS for A11 under iOS 12
iPhone XS [iPhone XR / iPhone XS / iPhone XS Max] with Apple A12[2] Bionic CPU	SEPOS for A12 under iOS 12
iPad Pro with Apple A9X CPU	SEPOS for A9X under iOS 12
iPad Pro with Apple A10X[1] Fusion CPU	SEPOS for A10X under iOS 12
Apple TV 4K with Apple A10X[1] Fusion CPU	SEPOS for A10X under tvOS 12
iPad Pro with Apple A12X[3] Bionic CPU	SEPOS for A12X under iOS 12
Apple Watch Series 1 with Apple S1P CPU	SEPOS for S1P under watchOS 5
Apple Watch Series 3 with Apple S3 CPU	SEPOS for S3 under watchOS 5
Apple Watch Series 4 with Apple S4 CPU	SEPOS for S4 under watchOS 5
Apple iMac Pro 2017 with Apple T2	SEPOS for T2 under TxFW 16P374
MacBook Pro (13-inch and 15-inch) with Apple T2	SEPOS for T2 under TxFW 16P374

Table 2b: Tested Platforms with hardware DRBG v2.0

2.2 Mode of Operation

The Apple Secure Key Store Cryptographic Module, v9.0 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-

[1] PAA provided here is the ARM NEON present in Apple A series processors
[2] Apple A10 and A10X are also known as Apple A10 Fusion and Apple A10X Fusion.
[3] Apple A11, A12 and A12X are also known as Apple A11 Bionic, Apple A12 Bionic and Apple A12X Bionic.

tests and is operating in the Approved mode. Any calls to the non-Approved security functions listed in Table 4 will cause the module to assume the non-Approved mode of operation.

The module transitions back into FIPS mode immediately when invoking one of the approved ciphers as all keys and Critical Security Parameters (CSP) handled by the module are exclusively assigned to different services which are bound to either approved or non-approved ciphers. There are no keys and CSPs shared between approved or non- approved functions as this is technically impossible due to the fact that the non-approved functions use key types that are cryptographically unusable by approved functions. A re-invocation of the self-tests or integrity tests is not required.

Even when using this FIPS 140-2 non-approved mode, the module ensures that the self-tests are always performed during initialization time of the module.

The module contains multiple implementations of the same cipher as listed below. If multiple implementations of the same cipher are present, the module selects the most appropriate cipher based on internal heuristics.

The Approved security functions are listed in Table 3. Column four (Algorithm Certificate Number) lists the validation numbers obtained from NIST for successful validation testing of the implementation of the cryptographic algorithms on the platforms as shown in Tables 2a and 2b under CAVP.

Refer to https://csrc.nist.gov/projects/cryptographic-algorithm-validation-program for the current standards, test requirements, and special abbreviations used in the following table.

2.2.1 Approved or Allowed Security Functions

Cryptographic Function	Algorithm	Modes/Options	Algorithm Certificate Number
Random Number Generation; Symmetric Key Generation	[SP 800-90A] DRBG	Hardware DRBG (CTR_DRBG) Counter mode: AES-256 No Derivation Function Prediction Resistance enabled	2013, 2014, 2020, 2021, 2022, 2023, 2024, 2025, 2026, 2028, 2029, C323, C324, C331
Symmetric Encryption and Decryption	[FIPS 197] AES SP 800-38A SP 800-38C SP 800-38D SP 800-38E	Generic Software (C) Implementation (Based on LibTomCrypt): Mode(s): CBC CFB8 OFB CCM CTR XTS[4] CFB128 ECB Key Lengths: 128, 192, 256 (bits)	C162, C163, C164, C165, C166, C167, C168, C169, C258, C260, C290, C296, C310, C363, C364
		Generic Software (C) Implementation (Based on Gladman): CBC Key Lengths: 128, 192, 256 (bits)	C151, C152, C153, C154, C155, C156, C157, C158, C159, C160, C286, C294, C305, C359, C361
		Generic Software (C) Implementation using Assembler Implementation of ECB: Mode(s): CBC CFB8 OFB CCM CTR XTS[4] CFB128 ECB Key Lengths: 128, 192, 256 (bits)	C87, C88, C89, C90, C91, C92, C93, C94, C161, C192, C288, C293, C306, C360, C362

[4] AES-XTS only supports 128-bit and 256-bit keys

Cryptographic Function	Algorithm	Modes/Options	Algorithm Certificate Number
		Assembler Implementation with ARM PAA: Mode(s): CBC OFB CFB128 ECB Key Lengths: 128, 192, 256 (bits)	C199, C200, C201, C202, C203, C204, C205, C206, C207, C208, C289, C298, C307, C367, C368
		VNG Implementation using (C) Implementation of ECB: Mode(s): CCM ECB CTR Key Lengths: 128, 192, 256 (bits)	C268, C270, C271, C277, C278, C279, C280, C282, C283, C284, C292, C300, C301, C309, C369
		VNG Implementation using Assembler Implementation of ECB: Mode(s): CCM CTR ECB Key Lengths: 128, 192, 256 (bits)	C261, C262, C263, C264, C265, C266, C267, C273, C274, C275, C291, C299, C308, C365, C366
		SKG AES Hardware Implementation Mode(s): ECB CBC Key Lengths: 128, 256 (bits)	C311, C312, C313, C314, C315, C317, C318, C319, C320, C322, C325, C326, C330, C358
		Hardware AES Implementation serving DRBG ECB Key Lengths: 256 (bits)	5260, 5261, 5270, 5271, 5272, 5273, 5274, 5275, 5276, 5278, 5279, C323, C324, C331
Key Transport	[FIPS 197] AES SP 800-38F	Generic Software (C) Implementation (Based on LibTomCrypt): Mode(s): AES-KW Key Lengths: 128, 192, 256 (bits)	C162, C163, C164, C165, C166, C167, C168, C169, C258, C260, C290, C296, C310, C363, C364
		Generic Software (C) Implementation using Assembler Implementation of ECB: Mode(s): AES-KW Key Lengths: 128, 192, 256 (bits)	C87, C88, C89, C90, C91, C92, C93, C94, C161, C192, C288, C293, C306, C360, C362
Digital Signature and Asymmetric Key Generation	[FIPS 186-4] ECDSA ANSI X9.62	PKG: P-224, P-256, P-384, P-521 PKV: P-224, P-256, P-384, P-521 Signature Generation: P-224, P-256, P-384, P-521 Signature Verification: P-224, P-256, P-384, P-521	C162, C163, C164, C165, C166, C167, C168, C169, C258, C260, C290, C296, C310, C363, C364

Cryptographic Function	Algorithm	Modes/Options	Algorithm Certificate Number
Message Digest	[FIPS 180-4] SHS	Generic Software (C) Implementation: SHA-1 SHA-384 SHA-224 SHA-512 SHA-256	C162, C163, C164, C165, C166, C167, C168, C169, C258, C260, C290, C296, C310, C363, C364
		SHA-VNG Implementation: SHA-1 SHA-384 SHA-224 SHA-512 SHA-256	C268, C270, C271, C277, C278, C279, C280, C282, C283, C284, C292, C300, C301, C309, C369
Shared Secret Computation for KAS	[SP800-56A] EC Diffie-Hellman Implements all of SP800-56A Except the Key Derivation Function	One-Pass Diffie-Hellman Section 6.2.2.2 (1e, 1s, ECC CDH) Curves: P-256, P384	CVL: C162, C163, C164, C165, C166, C167, C168, C169, C258, C260, C290, C296, C310, C363, C364
Keyed Hash	[FIPS 198] HMAC	Generic Software (C) Implementation: HMAC-SHA-1 HMAC-SHA-384 HMAC-SHA-224 HMAC-SHA-512 HMAC-SHA-256	C162, C163, C164, C165, C166, C167, C168, C169, C258, C260, C290, C296, C310, C363, C364
		SHA-VNG Implementation: HMAC-SHA-1 HMAC-SHA-384 HMAC-SHA-224 HMAC-SHA-512 HMAC-SHA-256	C268, C270, C271, C277, C278, C279, C280, C282, C283, C284, C292, C300, C301, C309, C369
Key Derivation	[SP 800-132] PBKDF	Password Based Key Derivation using HMAC with SHA-1 or SHA-256	Vendor Affirmed
NDRNG	Random number generation	N/A	Allowed

Table 3: Approved, Allowed or Vendor Affirmed Security Functions

2.2.2 Non-Approved Security Functions:

Cryptographic Function	Usage / Description	Caveat
Curve25519-based ECDH Key Agreement	EC Diffie-Hellman Key Agreement using Curve25519	Non-Approved
Ed25519 Key Agreement	Key Agreement	Non-Approved
RFC5869 Key Derivation	HMAC based Key Derivation Function	Non-Approved
ANSI X9.63 Key Derivation	Hash based KDF based on ANSI X9.63	Non-Approved
AES-GCM Encryption and Decryption	Encryption and Decryption	Non-Approved

Table 4: Non-Approved or Non-Compliant Functions

2.3 Cryptographic Module Boundary

The physical boundary of the module is the perimeter of the package. The logical module boundary is a sub-chip boundary including the firmware application (i.e. secure key store – SKS; blue outline) together with the Hardware DRBG and AES/SHA accelerator. The module's logical and physical boundaries are depicted in the logical block diagram given in Figure 1.

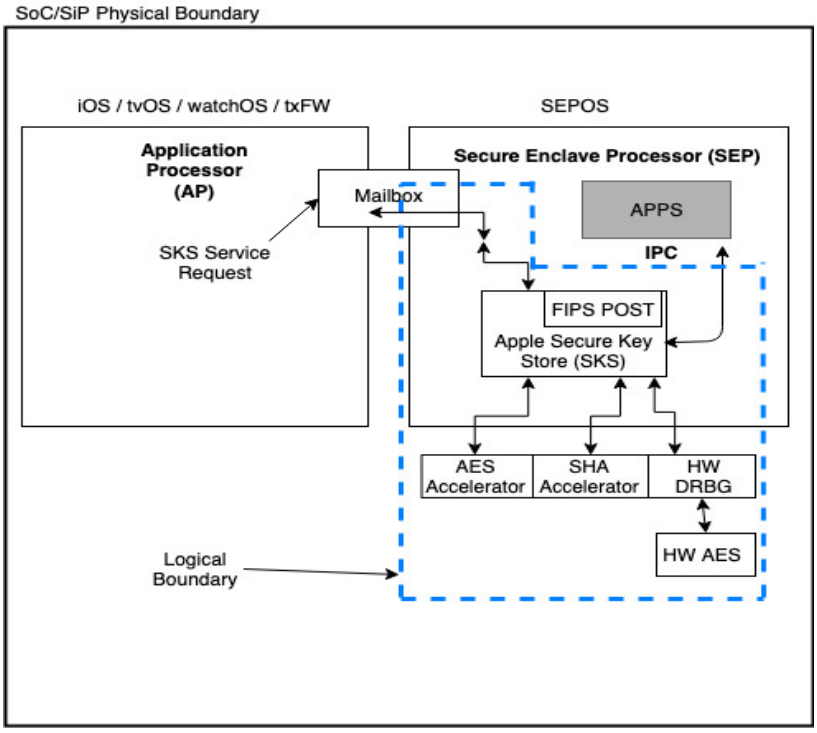

Figure 1: Cryptographic Module Block Diagram

3 Cryptographic Module Ports and Interfaces

The underlying logical interfaces of the module are the mailbox interface used between the module and the iOS kernel, and the IPC communication channel to other SEP applications:

- Data input and data output are provided through the memory used for mailbox and IPC.
- Control inputs which control the mode of the module are provided through the mailbox and the IPC. The HMAC control value is provided as part of the executable image implementing the module.
- 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.

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 key/CSP information. The module is optimized for use with the SEP coprocessor and does not contain any terminating assertions or exceptions. It is implemented as a hardware module with a hardware DRBG and an AES and SHA hardware accelerator implemented as part of the SoC and accessible to the SEP environment. Any internal error detected by the module is reflected back to the caller with an appropriate return code. The calling iOS application must examine the return code and act accordingly.

The function executing FIPS 140-2 module self-tests does not return an error code but causes the system to crash if any self-test fails – see Section 9.

The module communicates any error status synchronously through the use of its documented return codes, thus indicating the module's status. It is the responsibility of the caller to handle exceptional conditions in a FIPS 140-2 appropriate manner.

Caller-induced or internal errors do not reveal any sensitive material to callers.

Cryptographic bypass capability is not supported by the module.

4 Roles, Services and Authentication

This section defines the roles, services and authentication mechanisms and methods with respect to the applicable FIPS 140-2 requirements.

4.1 Roles

The module supports a two authorized roles: A Crypto Officer Role and a User Role. No support is provided for a Maintenance operator. The module does not implement a bypass mode nor concurrent operators.

When a device is delivered, the Crypto Officer is responsible of initializing the module i.e. configure the device by properly setting up key registers for storage of keys/CSPs and the FIFOs that will be later used by the software applications. Consequently, the Crypto Officer is not authenticated, the User can perform services from Table 6a and 6b only after the Crypto Officer takes possession by initializing it, thus creating data to be protected is generated.

The User of the module are software applications that assume the User Role when requesting any cryptographic services provided by the module.

A device can be completely factory reset which implies that all data is cryptographically destroyed by the module. Factory reset returns the module to an uninitialized state as an initially shipped device and thus the aforementioned statement about taking possession applies again.

Role	General Responsibilities and Services (details see below)
User	Utilization of cryptographic services of the module.
Crypto Officer (CO)	Initialization and configuration of the module (e.g. reboot, self-test).

Table 5: Roles

4.2 Operator Authentication

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

The module implements password-based authentication in the following way: When the User requests a service from the module, it must provide the passcode. This passcode is used to derive the key using PBKDF. The derived key is then used to decrypt the key utilized to perform the requested service. If the decryption of the key is successful, authentication is confirmed and the User's request is serviced with the unwrapped key. A failed decryption results in a failed authentication and the service request is rejected by the module.

The module does not maintain authenticated sessions upon power cycling. All authentication data is obscured during data entry.

4.2.1 Strength of Authentication

Once properly configured, the minimum length of the passcode is 7 digits, each with 10 different possibilities for usage. The chance of a random attempt falsely succeeding is $1:10^7$ which is less than 1:1,000,000 as required by FIPS 140-2.

Furthermore, the module implements delays between passcode attempts after four failed attempts. After the fourth failed attempt the module requires 1 min delay. This means that an attacker has the probability of guessing the password in one minute as $4:10^7$ which is far less than the requirement of 1/100,000

4.3 Services

The module provides services to authorized operators of either the User or Crypto Officer Roles according to the applicable FIPS 140-2 security requirements.

Table 6a contains the cryptographic services employed by the module in the Approved and Table 6b contains the cryptographic services employed by the module in the non-Approved mode. For each available service it lists, the associated role, the Critical Security Parameters (CSPs) and cryptographic keys involved, and the type(s) of access to the CSPs and cryptographic keys.

CSPs contain security-related information (for example, secret and private cryptographic keys) whose disclosure or modification can compromise the main security objective of the module, namely the protection of sensitive information.

The access types are denoted as follows:

- 'R': the item is read or referenced by the service
- 'W': the item is written or updated by the service
- 'Z': the persistent item is zeroized by the service

Service Number	Service	Roles		CSPs & crypto keys	Access Type
		USER	CO		
1.	Class D File System Services Encryption and Decryption of: - Class D key - File System keys	X		Key wrapping: UID, AES Key used to wrap Class D Key, Class D Key File system keys: DEK Storage controller key: KEK	R W
2.	User Keybag Services Encryption and Decryption of: - iOS file system object storage keys - User Keybag - User File system keys	X		Keybag wrapping: AES Key shared with NVM Storage Controller Keybag content: KEK	R W
3.	Device Keybag Services ECDSA signature generation and signature verification Encryption and Decryption of: - Trusted device communication keys - Device keybag - Device-specific keys - Keychain keys	X		Keybag wrapping: UID, AES Keys as part of module-managed keybags Keybag content: KEK Keychain keys: DEK Sign/verify key: ECDSA Private Key	R W

Service Number	Service	Roles		CSPs & crypto keys	Access Type
		USER	CO		
4.	Backup Keybag Services Encryption and Decryption of: - Backup keys - Backup keybag - Backup data	X		Keybag wrapping: AES Key used to wrap Backup Keybag, PBKDF password for Backup Keybag, PBKDF Salt for Backup Keybag Keybag content: KEK Backup keys: DEK	R W
5.	Escrow Keybag Services Device system update Authentication Encryption and Decryption of: - MDM keys - Escrow Keybag - Escrow file system keys	X		Keybag wrapping: AES Key used to wrap Escrow Keybag Keybag content: KEK from User Keybag, DEK	R W
6.	iCloud Keybag Services Encryption and Decryption of: - iCloud data keys - iCloud Keybag - iCloud user data	X		Keybag wrapping: REK derived from UID Keybag content: DEK	
7.	Create REK	X		UID PBKDF Password PBKDF Salt for REK DRBG internal state Entropy input string REK KEK as AES key used to wrap REK	R W
8.	Update REK	X		Old / new PBKDF Password PBKDF Salt for REK Old / new REK derived from PBKDF Password DRBG internal state Entropy input string UID	R W
9.	Generate Ref-Keys	X		EC Key Pair Password PBKDF Salt for EC Private Key Encryption Key	R W

Service Number	Service	Roles		CSPs & crypto keys	Access Type
		USER	CO		
10.	Generate Shared Secret	X		EC Key Pair Password PBKDF Salt for EC Private Key Encryption Key	R W
11.	Erase all content (Factory Reset)		X	All Keys[6] and CSPs	Z
12	Reboot that implies Self-test		X	None	N/A
13	Show Status		X	None	N/A

Table 6a: Approved Services in Approved Mode

Service	Roles	
	USER	CO
Ed 25519 Key Agreement	X	
ANSI X9.63 Hash based KDF based on ANSI X9.63	X	
EC Diffie-Hellman Key Agreement using Curve25519	X	
RFC 5869 based HKDF	X	
AES-GCM Encryption and Decryption	X	

Table 6b: Non-Approved Services in Non-Approved Mode

[6] Except UID; UID stored in hardware cannot be zeroized (See Section 7.5)

5 Physical Security

The Apple Secure Key Store Cryptographic Module, v9.0 is a hardware module implemented as a sub-chip and is identified as a single-chip embodiment. The physical boundary is considered to be each SoC listed in Table 1. The module conforms to the Level 2 requirements for physical security. The physical components that comprise the module are of production grade components with industry standard passivation applied. In addition, the module is covered with a tamper-evident coating that deters direct observation, probing, or manipulation of the single-chip.

6 Operational Environment

The following sections describe the operational environment of the Apple Secure Key Store Cryptographic Module, v9.0.

6.1 Applicability

The Apple Secure Key Store Cryptographic Module, v9.0 operates in a non-modifiable operational environment per FIPS 140-2 level 2 specifications. The module operates within the SEPOS execution environment which is separate from the iOS execution environment. The SEP operating system provides memory isolation between all applications executing on it. The iOS operating system is unable to access the module's memory or observe the module's operation.

6.2 Policy

The operating system is restricted to a single operator (i.e. concurrent operators are explicitly excluded).

7 Cryptographic Key Management

Table 7 summarizes the CSPs that are used by the cryptographic services implemented in the module. The rightmost column maps to the service number listed in Table 6a.

Name	Generation	Entry and Output	Zeroization	Used in service
Device-specific hardware key (UID)	A7, A8, A8X: N/A: Entered during manufacturing process Other SoCs: Output of module's DRBG during manufacturing process	A7, A8, A8X: Entry during manufacturing process Other SoCs: N/A – Generated using module's DRBG during manufacturing process and is never output.	N/A	1,3,7,8
Root Encryption Key (REK)	Derived from passcode using PBKDF2 and entanglement with UID	N/A – Generated inside the module and is never output.	Zeroized when freeing the secure memory.	6,7,8,11
PBKDF Password for the REK	N/A	Entered by calling application.	Zeroized when freeing the secure memory.	7,8,11
Backup Keybag Key	Derived from passcode using PBKDF2	N/A – Generated inside the module and is never output.	Zeroized when freeing the secure memory.	4,11
PBKDF Password for the Backup Keybag	N/A	Entered by calling application.	Zeroized when freeing the secure memory.	4,11
Escrow Keybag and the class D key wrapping key	Derived from UID	N/A – Generated inside the module and is never output	Zeroized when freeing the secure memory.	1,5,11
Module-managed keybags key	Symmetric key generation services of the module	Entered in wrapped form using AES-256 KW. Output in wrapped form using AES-256 KW	Non-volatile store: cryptographically zeroized when overwriting the KEK Volatile store: zeroized when freeing the secure memory.	3,11
File system object DEK	Symmetric key generation services of the module	Entered in wrapped form using AES-256 KW. Output in wrapped form using AES-256 KW	Non-volatile store: cryptographically zeroized when overwriting the KEK Volatile store: zeroized when freeing the secure memory.	1,3,4, 5,6,11

Name	Generation	Entry and Output	Zeroization	Used in service
NVM storage controller shared key	Symmetric key generation service of the module	Output to the storage controller of the SoC	Zeroized when volatile memory loses power during power down	2,11
Keychain ECDSA Private Keys	asymmetric key generation services of the module following FIPS 186-4	N/A – Generated inside the module and is never output.	Zeroized when freeing the secure memory	3,11
Entropy input string	Obtained from NDRNG	N/A	Zeroized when freeing the secure memory	7,8
DRBG internal state: V value, key and seed material	Updated during DRBG initialization	N/A	Zeroized when freeing the secure memory	7,8,11
PBKDF Salt for REK	Symmetric key generation services of the module	Entered in wrapped form using AES-256 KW. Output in wrapped form using AES-256 KW	Zeroized when freeing the secure memory	7,8,11
PBKDF Salt for Backup Keybag	Symmetric key generation services of the module	Entered in wrapped form using AES-256 KW. Output in wrapped form using AES-256 KW	Zeroized when freeing the secure memory	4,11
Class D Key	Symmetric key generation services of the module	Entered in wrapped form using AES-256 KW. Output in wrapped form using AES-256 KW	Non-volatile store: cryptographically zeroized when caller requests new key Volatile store: zeroized when freeing the secure memory.	1,11
REK wrapping key	Symmetric key generation services of the module	Entered in plaintext by calling application within physical the boundary Output to in plaintext calling application within the physical boundary	Zeroized when freeing the secure memory.	7,11
Ref Key	asymmetric key generation services of the module following FIPS 186-4	N/A – Generated inside the module and is never output.	Zeroized when freeing the secure memory	9,10
HMAC Key	Symmetric key generation services of the module	Entered by calling application	Zeroized when freeing the secure memory	11,12

Table 7: Life Cycle of Critical Security Parameters (CSP)

The following section defines the key management features available through the Apple Secure Key Store Cryptographic Module, v9.0.

7.1 Random Number Generation

A FIPS 140-2 approved deterministic random bit generator based on a block cipher as specified in NIST SP 800-90A is used. The Approved DRBG used for random number generation is a CTR_DRBG using AES-256 without derivation function and with prediction resistance. The deterministic random bit generator is seeded by an internal noise source consisting of 8 ring oscillators (up to and including A8X) or 24 ring oscillators (A9 and newer, S1P, S3, S4 and T2). The ring oscillators provide 256-bits of entropy.

7.2 Key / CSP Generation

The following approved key generation methods are used by the module:

- The Approved DRBG specified in section 7.1 is used to generate secret symmetric keys for the AES algorithm.
- The Approved DRBG specified in section 7.1 is used to generate the random values used in the key generation of asymmetric keys. Asymmetric keys for the ECDSA / ECDH algorithm are generated using FIPS 186-4.

The module provides a key generation service for symmetric ciphers and HMAC keys. The key generation service is compliant with SP800-133 that requires the symmetric key is an XOR of the DRBG output with a value V. In case of the module, the value V is a string of zeros which implies that the key is unmodified from the output of the DRBG.

It is not possible for the module to output information during the key generating process.

7.3 Key / CSP Establishment

The module provides key transport service through SP 800-38F AES key wrapping. In addition, the module provides key derivation services in the Approved mode through the PBKDF2 algorithm. The module supports option 1a from Section 5.4 of SP 800-132, whereby the Master Key (MK) is used directly as the Data Protection Key (DPK). Keys derived from passwords may only be used for data at rest. The length of the passcode used in the key derivation is 7 digits with the probability of guessing this password is $(1/10)^7$. The caller shall observe all requirements and should consider all recommendations specified in SP800-132 with respect to the strength of the generated key, including the quality of the password, the quality of the salt as well as the number of iterations. The module also provides key agreement scheme based on SP800-56A without KDF. The module implements ECC primitive calculation based on Section 5.7.1.2 ECC CDH Primitive of SP 800-56A. The module itself does not include implementation of key derivation function (KDF). In the approved mode, the module provides EC Diffie-Hellman shared secret computation with curves P-256 or P-384, providing 128- or 192-bit equivalent security strength, respectively.

7.4 Key / CSP Entry and Output

The module does not support entry or output of cryptographic keys beyond the physical boundary of the SoC. Within the physical boundary, all secret keys and CSPs are entered into, or output from the Apple Secure Key Store Cryptographic Module in wrapped form using AES-KW (SP800-38F). The exception is the User's password which is entered into the module in the clear. All keys and CSPs entered into the module are electronically enteredKey / CSP Storage

The Apple Secure Key Store Cryptographic Module, v9.0 considers all keys in memory to be ephemeral.

The keys managed by the module in keybags are stored in non-volatile memory by the iOS operating system. The Keybag is wrapped with AES-256 KW followed by an export to iOS for permanent storage. After a power-up, the module imports the wrapped keybags from iOS and unwraps them.

The module protects all keys at runtime, secret or private, and CSPs through the memory protection mechanisms provided by SEPOS. No process can read the memory of another process.

7.5 Key / CSP Zeroization

Cleartext keys and CSPs are zeroized immediately after their usage is completed or when the device is powered down. Additionally, the user can zeroize the entire device directly (locally) or remotely, returning it to the original factory settings.

The exception is the key called the device UID which is stored in a specially protected hardware component. The UID key is programmed during manufacturing process and cannot be directly read or written by any software/firmware. It can only be used for an AES encryption or decryption operation. The UID is used to wrap the file system Class D key or keys that are intended to be bound to the current device. For wrapping the remaining Class keys, a key is derived using the KDF from the UID and a key derived from the User's password. Therefore, the UID is required for the life-time of the device. The UID stored in hardware cannot be zeroized.

8 Electromagnetic Interference/Electromagnetic Compatibility (EMI/EMC)

The cryptographic module hardware component cannot be certified by the FCC as it is not a standalone device. It is a sub-chip embedded in the devices listed in Section 2.1.3. The devices containing the cryptographic module are conformant to the EMI/EMC requirements specified by 47 Code of Federal Regulations, Part 15, Subpart B, Unintentional Radiators, Digital Appliances, Class A (business use).

9 Self-Tests

FIPS 140-2 requires that the module perform self-tests to ensure the integrity of the module and the correctness of the cryptographic functionality at start up. In addition, the noise source feeding the random bit generator requires continuous verification. The module runs all the required self-tests which are invoked automatically when module is powered on.

The occurrence of any self-test error triggers an immediate shutdown of the module, preventing any operation.

All self-tests performed by the module are listed and described in this section.

9.1 Power-Up Tests

The following tests are performed each time the Apple Secure Key Store Cryptographic Module, v9.0 starts and must be completed successfully for the module to operate. If any of the following tests fail, the module enters into an error state whereby data output interface is inhibited and the device powers itself off. To rerun the self-tests on demand, the user must reboot the module.

9.1.1 Cryptographic Algorithm Tests

Algorithm	Modes	Test
AES Implementation selected by the module for the corresponding environment AES-128	ECB, CBC	KAT[7] Separate encryption / decryption operations are performed
AES SKG Hardware Accelerator Implementation AES-128	ECB, CBC	KAT Separate encryption / decryption operations are performed
Hardware DRBG (CTR_DRBG)	N/A	KAT
HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-512	N/A	KAT[8]
ECDSA	Signature Generation, Signature Verification	PCT
EC Diffie-Hellman “Z” computation	N/A	KAT

Table 8: Cryptographic Algorithm Tests

9.1.2 Firmware Integrity Tests

A firmware integrity test is performed on the runtime image of the Apple Secure Key Store Cryptographic Module, v9.0. The module’s HMAC-SHA-256 is used as an Approved algorithm for the integrity test. If the test fails, then the device powers itself off.

9.1.3 Critical Function Tests

No other critical function test is performed on power up.

9.2 Conditional Tests

The following sections describe the conditional tests supported by the Apple Secure Key Store Cryptographic Module, v9.0.

[7] Self-test is subject to the "selector" approach for the different implementations of AES.
[8] Self-test is subject to the "selector" approach for the different implementations of SHA.

9.2.1 Repetition Count Test

The Apple Secure Key Store Cryptographic Module, v9.0 performs a continuous random number generator test by way of a Repetition Count Test (RCT) on the NDRNG, whenever the DRBG is seeded or reseeded.

9.2.2 Pair-wise Consistency Test

The Apple Secure Key Store Cryptographic Module, v9.0 performs a pair-wise consistency tests on asymmetric keys generated for ECDSA cipher.

9.2.3 SP 800-90A Health Tests

The Apple Secure Key Store Cryptographic Module, v9.0 performs a the health tests as specified in section 11.3 of SP 800-90A.

9.2.4 Critical Function Test

No other critical function test is performed conditionally.

10 Design Assurance

10.1 Configuration Management

Apple manages and records source code and associated documentation files by using the revision control system called “Git”.

The Apple module hardware data, which includes descriptions, parts data, part types, bills of materials, manufacturers, changes, history, and documentation are managed and recorded. Additionally, configuration management is provided for the module’s FIPS documentation.

The following naming/numbering convention for documentation is applied.

<evaluation>_<module>_<os>_<mode>_<doc name>_<doc version (#.#)>

Example: FIPS_SEP_SECPOL_2.0

Document management utilities provide access control, versioning, and logging. Access to the Git repository (source tree) is granted or denied by the server administrator in accordance with company and team policy.

10.2 Delivery and Operation

The module’s firmware with the SEPOS is delivered as part of the iOS image. The Approved mode is configured by default and can only be transitioned into the non-Approved mode by calling one of the non-Approved algorithms listed in Table 4.

10.3 Development

The Apple crypto module (like any other Apple software) undergoes frequent builds utilizing a “train” philosophy. Source code is submitted to the Build and Integration group (B & I). B & I builds, integrates and does basic sanity checking on the operating systems and apps that they produce. Copies of older versions are archived offsite in underground granite vaults.

10.4 Guidance

The following guidance items are to be used for assistance in maintaining the module’s validated status while in use.

10.4.1 Cryptographic 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 algorithms listed in Table 4. If the device starts up successfully then the module has passed all self-tests and is operating in the Approved mode.

10.4.2 User Guidance

As above, 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 algorithms listed in Table 4. If the device starts up successfully then the module has passed all self-tests and is operating in the Approved mode.

10.4.2.1 Module Usage Considerations

A user of the module must consider the following requirements and restrictions when using the module:

- As specified in SP800-38E, the AES algorithm in XTS mode is designed for the cryptographic protection of data on storage devices. It can only be used for encryption of data at rest.

- To meet the requirement stated in IG A.9, the module implements a check to ensure that the two AES keys used in AES XTS mode are not identical.

11 Mitigation of Other Attacks

The module has not been designed to mitigate any specific attacks outside the scope of FIPS 140-2 requirements.