Apple Inc.

Apple Secure Key Store Cryptographic Module, v1.0 FIPS 140-2 Non-Proprietary Security Policy

Hardware Versions:
1.2, 2.0
Firmware version:
SEPOS

June, 2018

Prepared for:
Apple Inc.
1 Infinite Loop
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

1 INTRODUCTION 5

1.1 PURPOSE 5
1.2 DOCUMENT ORGANIZATION / COPYRIGHT 5
1.3 EXTERNAL RESOURCES / REFERENCES 5
1.3.1 Additional References 5
1.4 ACRONYMS 7

2 CRYPTOGRAPHIC MODULE SPECIFICATION 8

2.1 MODULE DESCRIPTION 8
2.1.1 Module Validation Level 8
2.1.2 Module Components 8
2.1.3 Tested Platforms 9
2.2 MODE OF OPERATION 9
2.2.1 Approved or Allowed Security Functions 10
2.2.2 Non-Approved Security Functions: 13
2.3 CRYPTOGRAPHIC MODULE BOUNDARY 14

3 CRYPTOGRAPHIC MODULE PORTS AND INTERFACES 15

4 ROLES, SERVICES AND AUTHENTICATION 16

4.1 ROLES 16
4.2 SERVICES 16
4.3 OPERATOR AUTHENTICATION 19

5 PHYSICAL SECURITY 20

6 OPERATIONAL ENVIRONMENT 21

6.1 APPLICABILITY 21
6.2 POLICY 21

7 CRYPTOGRAPHIC KEY MANAGEMENT 22

7.1 RANDOM NUMBER GENERATION 24
7.2 KEY / CSP GENERATION 24
7.3 KEY / CSP ESTABLISHMENT 24
7.4 KEY / CSP ENTRY AND OUTPUT 24
7.5 KEY / CSP STORAGE 24
7.6 KEY / CSP ZEROIZATION 25

8 ELECTROMAGNETIC INTERFERENCE/ELECTROMAGNETIC COMPATIBILITY (EMI/EMC) 26

9 SELF-TESTS 27

9.1 POWER-UP TESTS 27
9.1.1 Cryptographic Algorithm Tests 27
9.1.2 Software / Firmware Integrity Tests 27
9.1.3 Critical Function Tests 27
9.2 CONDITIONAL TESTS 28
9.2.1 Repetition Count Test 28
9.2.2 Pair-wise Consistency Test 28
9.2.3 SP 800-90A Assurance Tests 28
9.2.4 Critical Function Test 28

10 DESIGN ASSURANCE 29
10.1 CONFIGURATION MANAGEMENT 29
10.2 DELIVERY AND OPERATION 29
10.3 DEVELOPMENT 29
10.4 GUIDANCE 29
10.4.1 Cryptographic Officer Guidance 29
10.4.2 User Guidance 29
11 MITIGATION OF OTHER ATTACKS 31

List of Tables

Table 1: Module Validation Level 8
Table 2a: Tested Platforms with hardware DRBG v1.2 9
Table 2b: Tested Platforms with hardware DRBG v2.0 9
Table 3: Approved, Allowed or Vendor Affirmed Security Functions 12
Table 4: Non-Approved or Non-Compliant Functions 13
Table 5: Roles 16
Table 6a: Approved Services in Approved Mode 18
Table 6b: Non-Approved Services in Non-Approved Mode 18
Table 7: Life Cycle of Critical Security Parameters (CSP) 23
Table 8: Cryptographic Algorithm Tests 27

List of Figures

Figure 1: Logical Block Diagram 14

1 Introduction

1.1 Purpose

This document is a non-proprietary Security Policy for the Apple Secure Key Store Cryptographic Module, v1.0.0. 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, v1.0.0." "cryptographic module", "SKS" or "the module" are used interchangeably to refer to the Apple Secure Key Store Cryptographic Module, v1.0.0.

1.2 Document Organization / Copyright

This non-proprietary Security Policy document may be reproduced and distributed only in its original entirety without any revision, © 2018 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 (http://csrc.nist.gov/groups/STM/cmvp/index.html) 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: http://csrc.nist.gov/groups/STM/cmvp/standards.html
FIPS 140-2 IG	NIST, "Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program," November 15, 2016 Location: http://csrc.nist.gov/groups/STM/cmvp/standards.html
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, v1.0 is a hardware cryptographic module implemented as a sub-chip running on a single-chip standalone 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 1 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	1
Cryptographic Module Ports and Interfaces	1
Roles, Services and Authentication	1
Finite State Model	1
Physical Security	1
Operational Environment	N/A
Cryptographic Key Management	1
EMI/EMC	1
Self-Tests	1
Design Assurance	1
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, v1.0 are listed in detail. The referenced devices listed in Table 2a and Table 2b embed an SoC which has multiple execution environments, including different CPUs for the main operating system (iOS, watchOS, tvOS, iBridgeOS -- the remainder of this document only uses the reference of iOS which equally applies to the other operating systems as they are functionally equivalent with respect to the FIPS 140-2 mechanisms to iOS unless explicitly noted) as well as the operating system driving the Secure Enclave Processor (SEP) execution environment (SEPOS). Both environments are operating systems executing on different CPUs embedded within the SoC. Both execution environments are separated by the SoC and thus execute independently of each other. The execution environment of the cryptographic module is SEPOS. The module consists of firmware and a hardware component. The firmware part operates on the SoC.

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 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 on the following platforms:

Model	Operating System
iPhone 5S with Apple A7 CPU	SEPOS for A7 under iOS 11
iPhone 6 with Apple A8 CPU (iPhone 6 and iPhone 6 Plus)	SEPOS for A8 under iOS 11
iPad Air 2 with Apple A8X CPU	SEPOS for A8X under iOS 11

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 11
iPhone 7 with Apple A10[1] Fusion CPU (iPhone 7 and iPhone 7 Plus)	SEPOS for A10 under iOS 11
iPhone 8 and iPhone X with Apple A11[2] Bionic CPU (iPhone 8, iPhone 8 Plus, iPhone X)	SEPOS for A11 under iOS 11
iPad Pro with Apple A9X CPU	SEPOS for A9X under iOS 11
iPad Pro with Apple A10X[1] Fusion CPU	SEPOS for A10X under iOS 11
Apple TV 4K with Apple A10X[1] Fusion CPU	SEPOS for A10X under tvOS 11
Apple Watch Series 1 with Apple S1P CPU	SEPOS for S1P under watchOS 4
Apple Watch Series 3 with Apple S3 CPU	SEPOS for S3 under watchOS 4
Apple iMac Pro 2017 with Apple iBridge2,1	SEPOS for iBridge2,1 under iBridgeOS 15YP2064

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

2.2 Mode of Operation

The Apple Secure Key Store Cryptographic Module, v1.0 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-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.

[1] Apple A10 and A10X are also known as Apple A10 Fusion and Apple A10X Fusion.

[2] Apple A11 is also known as Apple A11 Bionic.

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 automatically which cipher is used 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 http://csrc.nist.gov/groups/STM/cavp/index.html 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: AES-256 No Derivation Function Prediction Resistance enabled	2013, 2014, 2020, 2021, 2022, 2023, 2024, 2025, 2026, 2027, 2028, 2029
Symmetric Encryption and Decryption	[FIPS 197] AES SP 800-38 A SP 800-38 C SP 800-38 D SP 800-38 E	Generic Software Implementation (Based on LibTomCrypt): Modes: CBC CFB8 GCM CCM CTR OFB CFB128 ECB XTS[3] Key Lengths: 128, 192, 256 (bits)	5145, 5150, 5151, 5152, 5153, 5154, 5155, 5156, 5157, 5199, 5217, 5219
		Generic Software Implementation (Based on Gladman): Modes: CBC Key Lengths: 128, 192, 256 (bits)	5104, 5105, 5107, 5108, 5109, 5111, 5112, 5134, 5144, 5196, 5213, 5214
		Generic Software Implementation using Assembler Implementation of ECB: Modes: CBC CFB8 GCM CCM CTR OFB CFB128 ECB XTS[3] Key Lengths: 128, 192, 256 (bits)	5102, 5103, 5106, 5110, 5114, 5115, 5118, 5131, 5143, 5198, 5215, 5216

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

Cryptographic Function	Algorithm	Modes/Options	Algorithm Certificate Number
		Generic Software Implementation with ARM PAA: CCM ECB CTR Key Lengths: 128, 192, 256 (bits)	5148, 5159, 5160, 5161, 5162, 5163, 5164, 5165, 5182, 5202, 5224, 5225
		Assembler Implementation with ARM PAA: Modes: CBC OFB CFB128 XTS ECB Key Lengths: 128, 192, 256 (bits)	5113, 5116, 5117, 5119, 5120, 5121, 5132, 5133, 5146, 5200, 5222, 5223
		Optimized Assembler with ARM PAA: Modes: CCM GCM CTR ECB Key Lengths: 128, 192, 256 (bits)	5149, 5201, 5188, 5189, 5190, 5191, 5192, 5193, 5194, 5195, 5221, 5226
		SKG AES Hardware Implementation Modes: ECB CBC Key Lengths: 128, 256 (bits)	5135, 5136, 5137, 5138, 5139, 5140, 5141, 5142, 5147, 5197, 5218, 5220
		Hardware AES Implementation serving DRBG Modes: ECB Key Lengths: 256 (bits)	5260, 5261, 5270, 5271, 5272, 5273, 5274, 5275, 5276, 5277, 5278, 5279
Key Transport	[FIPS 197] AES SP 800-38 F	Generic Software Implementation (Based on LibTomCrypt): Modes: AES-KW Key Lengths: 128, 192, 256 (bits)	5145, 5150, 5151, 5152, 5153, 5154, 5155, 5156, 5157, 5199, 5217, 5219
		Generic Software Implementation using Assembler Implementation of ECB: Modes: AES-KW Key Lengths: 128, 192, 256 (bits)	5102, 5103, 5106, 5110, 5114, 5115, 5118, 5131, 5143, 5198, 5215, 5216
		Generic Software Implementation with ARM PAA: Modes: AES-KW Key Lengths: 128, 192, 256 (bits)	5148, 5159, 5160, 5161, 5162, 5163, 5164, 5165, 5182, 5202, 5224, 5225
		Assembler Implementation with ARM PAA: Modes: AES-KW Key Lengths: 128, 192, 256 (bits)	5113, 5116, 5117, 5119, 5120, 5121, 5132, 5133, 5146, 5200, 5222, 5223

Cryptographic Function	Algorithm	Modes/Options	Algorithm Certificate Number
		Optimized Assembler with ARM PAA: Modes: AES-KW Key Lengths: 128, 192, 256 (bits)	5149, 5201, 5188, 5189, 5190, 5191, 5192, 5193, 5194, 5195, 5221, 5226
Digital Signature and Asymmetric Key Generation	[FIPS 186-4] ECDSA ANSI X9.62	PKG: curves P-224, P-256, P-384, P-521 PKV: curves P-224, P-256, P-384, P-521 Signature Generation: curves P-224, P-256, P-384, P-521 using (SHA-224, SHA-256, SHA384, SHA512) Signature Verification: curves P-224, P-256, P-384, P-521 using (SHA-1, SHA-224, SHA-256, SHA384, SHA512)	1327, 1328, 1329, 1330, 1331, 1332, 1333, 1334, 1335, 1346, 1351, 1352
Message Digest	[FIPS 180-4] SHS	Generic Software Implementation: SHA-1 SHA-384 SHA-224 SHA-512 SHA-256	4155, 4157, 4158, 4159, 4160, 4161, 4162, 4163, 4164, 4191, 4203, 4204
		Generic Software with ARM PAA implementation: SHA-1 SHA-384 SHA-224 SHA-512 SHA-256	4156, 4166, 4167, 4168, 4169, 4170, 4171, 4172, 4186, 4192 4205, 4206
Shared Secret Computation for KAS	[SP800-56A] EC Diffie-Hellman Implementation follows SP800-56A for primitive only	One-Pass Diffie-Hellman Section 6.2.2.2 (1e, 1s, ECC CDH) Curves: P-256, P384	CVL: 1654, 1656, 1658, 1660, 1662, 1664, 1666, 1668, 1670, 1686, 1696, 1698
Keyed Hash	[FIPS 198] HMAC	Generic Software Implementation: HMAC-SHA-1 HMAC-SHA-384 HMAC-SHA-224 HMAC-SHA-512 HMAC-SHA-256	3409, 3411, 3412, 3413, 3414, 3415, 3416, 3417, 3418, 3443, 3455, 3456
		Generic Software with ARM PAA implementation: HMAC-SHA-1 HMAC-SHA-384 HMAC-SHA-224 HMAC-SHA-512 HMAC-SHA-256	3410, 3420, 3421, 3422, 3423, 3424, 3425, 3426, 3438, 3444, 3457, 3458
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
RSA Key Wrapping	RSA Key Wrapping using OAEP	Non-Approved
Curve25519-based ECDH	EC Diffie-Hellman Key Agreement using Curve25519	Non-Approved
Ed25519	Key Agreement	Non-Approved
RFC5869 KDF	HMAC based Key Derivation Function	Non-Approved
ANSI X9.63 KDF	Hash based KDF based on ANSI X9.63	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 and is listed in Figure 1 (red outline). Consequently, the embodiment of the module is a single-chip standalone cryptographic module. The logical module boundary is a sub-chip boundary including the firmware application (i.e. secure key store – SKS; green 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.

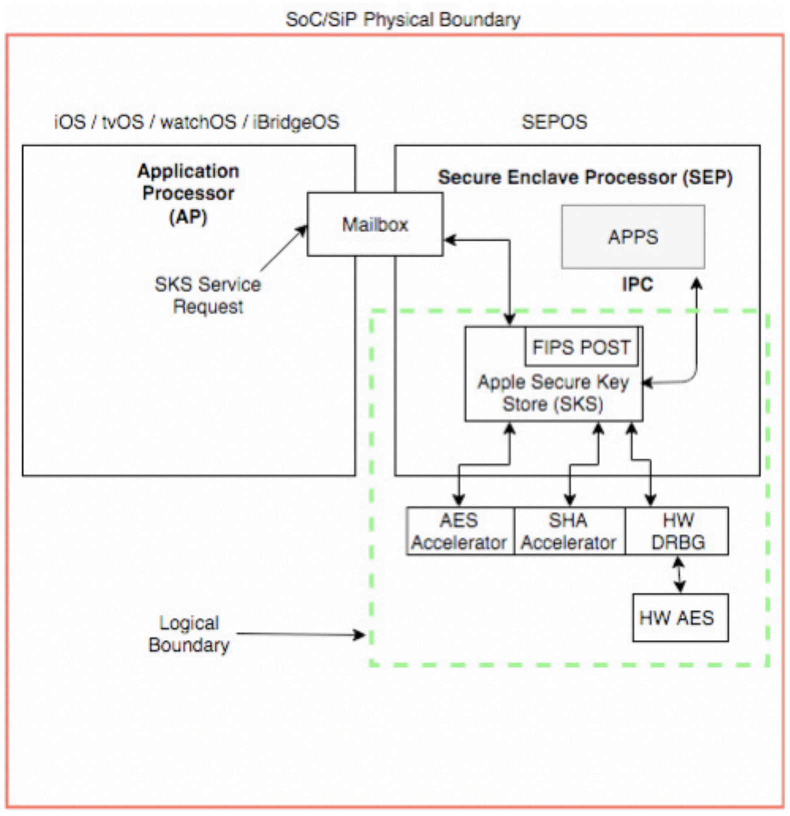
SoC/SiP Physical Boundary
iOS / tvOS / watchOS / iBridgeOS
SEPOS
Application Processor (AP)
Mailbox
Secure Enclave Processor (SEP)
APPS
IPC
SKS Service Request
FIPS POST
Apple Secure Key Store (SKS)
AES Accelerator
SHA Accelerator
HW DRBG
HW AES
Logical Boundary

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 L4 IPC communication channel to other SEP applications:

- Data input and data output are provided through the memory used for mailbox and L4 IPC.
- Control inputs which control the mode of the module are provided through the mailbox and the L4 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 L4 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 single instance of the two authorized roles: the Crypto Officer and the User. No support is provided for multiple concurrent operators or a Maintenance operator.

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

Table 5: Roles

4.2 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 functions employed by the module in the Approved and 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 wrapping key - Wrapping and unwrapping of class D key - Wrapping and unwrapping of 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

Service Number	Service	Roles		CSPs & crypto keys	Access Type
		USER	CO		
2.	User Keybag holding keys protecting iOS file system object storage keys - Wrapping and unwrapping of User Keybag - Processing of file system keys	X		Keybag wrapping: AES Key shared with NVM Storage Controller Keybag content: KEK	R W
3.	Device Keybag holding keys protecting trusted device communication - Wrapping and unwrapping of Keybag - Processing device-specific keys - Import of keychain keys - Encryption/Decryption using keychain keys -sign/verify using ECDSA key	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
4.	Backup Keybag holding keys protecting the backups - Wrapping and unwrapping of Keybag - Encrypting or decrypting 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 keys protecting MDM related data - Wrapping and unwrapping of Keybag - Wrapping and unwrapping of file system keys - Authenticating the system update	X		Keybag wrapping: AES Key used to wrap Escrow Keybag Keybag content: KEK from User Keybag, DEK	R W
6.	iCloud Keybag keys protecting iCloud data - Wrapping and unwrapping of Keybag - Encrypting or decrypting of user data	X		Keybag wrapping: REK derived from UID Keybag content: DEK	

Service Number	Service	Roles		CSPs & crypto keys	Access Type
		USER	CO		
7.	- Create new 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.	- Erase all content	X		All Keys[4] and CSPs	Z
10.	- Reboot that implies Self-test	X	X	HMAC Key	N/A
11.	- Show Status		X	None	N/A

Table 6a: Approved Services in Approved Mode

Service	Roles	
	USER	CO
RSA Key Wrapping using OAEP	X	
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	

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

[4] Except UID; UID stored in hardware cannot be zeroized

4.3 Operator authentication

Within the constraints of FIPS 140-2 level 1, the module does not implement an authentication mechanism for operator authentication. The assumption of a role is implicit in the action taken. The module relies upon the operating system for any operator authentication.

5 Physical Security

The Apple Secure Key Store Cryptographic Module, v1.0 is a hardware module implemented as a sub-chip and is identified as a single-chip standalone module. The physical boundary is considered to be SoC listed in Table 1. The module conforms to the Level 1 requirements for physical security. The physical components that comprise the module are of production grade components with industry standard passivation applied.

6 Operational Environment

The following sections describe the operational environment of the Apple Secure Key Store Cryptographic Module, v1.0.

6.1 Applicability

The Apple Secure Key Store Cryptographic Module, v1.0 operates in a limited modifiable operational environment per FIPS 140-2 level 1 specifications. It is part of SEPOS associated with the respective Apple System-on-a-Chip (SoC) driving the hardware, a commercially available special-purpose operating system executing on the hardware specified in section 2.1.3.

6.2 Policy

The operating system is restricted to a single operator (single-user mode; 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	1,3,8,7
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.	6,7,8,9
AES Key used to wrap Backup Keybag (stored locally)	Derived from passcode using PBKDF2	N/A – Generated inside the module and is never output.	Zeroized when freeing the secure memory.	4,9
PBKDF Password for the REK	N/A	Entered by calling application.	Zeroized when freeing the secure memory.	7,8,9
PBKDF Password for the Backup Keybag	N/A	Entered by calling application.	Zeroized when freeing the secure memory.	4,9
AES Key used to wrap the Escrow Keybag and the Class D key	Derived from UID	N/A – Generated inside the module and is never output	Zeroized when freeing the secure memory.	1,5,9
AES Keys as part of module-managed keybags	Symmetric key generation services of the module	Entered by iOS in wrapped form. Output to iOS in wrapped form Generation by the module	Non-volatile store: cryptographically zeroized when overwriting the KEK Volatile store: zeroized when freeing the secure memory.	3,9
AES Keys used to wrap file system object (DEK)	Symmetric key generation services of the module	Entered by iOS in wrapped form. Output to iOS in wrapped form Generation by the module	Non-volatile store: cryptographically zeroized when overwriting the KEK Volatile store: zeroized when freeing the secure memory.	1,3,4, 5,6,9

Name	Generation	Entry and Output	Zeroization	Used in service
AES Keys shared with NVM Storage controller 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	2,9
ECDSA Private Keys held as part of keychain items	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	3,9
Entropy input string	Obtained from NDRNG	N/A	Zeroized when freeing the secure memory	7,8
DRBG internal state: V value, C value, key (if applicable) and seed material	Updated during DRBG initialization	N/A	Zeroized when freeing the secure memory	7,8,9
PBKDF Salt for REK	Symmetric key generation services of the module	Entered by iOS in wrapped form. Output to iOS in wrapped form	Zeroized when freeing the secure memory	7,8,9
PBKDF Salt for Backup Keybag	Symmetric key generation services of the module	Entered by iOS in wrapped form Output to iOS in wrapped form	Zeroized when freeing the secure memory	4,9
AES Key held in effaceable storage (Class D Key)	Symmetric key generation services of the module	Entered and Output in wrapped form	Non-volatile store: cryptographically zeroized when caller requests new key Volatile store: zeroized when freeing the secure memory.	1,9
KEK as AES key used to wrap REK	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.	7,9
HMAC Key	Symmetric key generation services of the module	Entered by calling application	Zeroized when freeing the secure memory	9,10

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, v1.0.

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 and iBridge2,1). 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 secret asymmetric keys for the ECDSA / ECDH algorithm.

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 MK is used directly as the DPK. Keys derived from passwords may only be used for data at rest. The length of the passcode used in the key derivation is 6 digits with the probability of guessing this password is (1/10)^6. 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.

7.4 Key / CSP Entry and Output

The module does not support entry or output of cryptographic keys beyond its 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 key derived from the user's password by iOS which is entered into the module in clear. All keys and CSPs entered into the module are electronically entered. Keys and CSP are output from the module if required by the calling iOS users.

7.5 Key / CSP Storage

The Apple Secure Key Store Cryptographic Module, v1.0 considers all keys in memory to be ephemeral. They are received for use or generated by the module only at the command of the calling iOS service. The same holds for CSPs.

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.

7.6 Key / CSP Zeroization

Clear 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 EMI/EMC properties of the SEP are not meaningful for the software library. The devices containing the software components of the module have their own overall EMI/EMC rating. The validation test environments have FCC, part 15, Class B rating.

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 required module self-tests pertaining to the firmware. This self-test is invoked automatically when starting the module. In addition, during startup of the hardware, the hardware DRBG invokes its independent self-test.

The occurrence of a self-test error in either the firmware or the hardware DRBG triggers an immediate shutdown of the device 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, v1.0 starts and must be completed successfully for the module to operate in the FIPS approved mode. If any of the following tests fails the device powers itself off. To rerun the self-tests on demand, the user must reboot the device.

9.1.1 Cryptographic Algorithm Tests

Algorithm	Modes	Test
AES Implementation selected by the module for the corresponding environment AES-128	ECB, CBC	KAT[5] 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[6]
ECDSA	SIG(ver), SIG(gen)	PCT
EC Diffie-Hellman “Z” computation	N/A	KAT

Table 8: Cryptographic Algorithm Tests

9.1.2 Software / Firmware Integrity Tests

A software integrity test is performed on the runtime image of the Apple Secure Key Store Cryptographic Module, v1.0. 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.

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

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

9.2 Conditional Tests

The following sections describe the conditional tests supported by the Apple Secure Key Store Cryptographic Module, v1.0.

9.2.1 Repetition Count Test

The Apple Secure Key Store Cryptographic Module, v1.0 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, v1.0 performs a pair-wise consistency tests on asymmetric keys generated for ECDSA cipher.

9.2.3 SP 800-90A Assurance Tests

The Apple Secure Key Store Cryptographic Module, v1.0 performs a subset of the assurance tests as specified in section 11 of SP 800-90A, in particular it complies with the mandatory documentation requirements and performs known-answer tests and prediction resistance.

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_1.0

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 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:

- AES-GCM IV is constructed in accordance with SP800-38D section 8.2.1. Users should consult SP 800-38D, especially section 8, for all of the details and requirements of using AES-GCM mode. The GCM IV may only be use used in the context of the AES GCM within the TLS v1.2/IPSec protocol.

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