Apple Inc.

Apple corecrypto Module v13.0 [Intel, Kernel, Software, SL1]

FIPS 140-3 Non-Proprietary Security Policy

Document Version 2.0
December 2024

Prepared by:

Lightship Security Inc.

1101-150 Isabella Street,

Ottawa, ON, K1S 1V7

www.lightshipsec.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.

Contents

Tables

1. General

This document is the non-proprietary FIPS 140-3 Security Policy for Apple corecrypto Module v13.0 [Intel, Kernel, Software, SL1] 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 a Security Level 1 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.

Section	FIPS 140-3 Section Title	Security Level
1	General	1
2	Cryptographic module specification	1
3	Cryptographic module interfaces	1
4	Roles, services, and authentication	1
5	Software/Firmware security	1
6	Operational environment	1
7	Physical security	N/A
8	Non-invasive security	N/A
9	Sensitive security parameter management	1
10	Self-tests	1
11	Life-cycle assurance	1
12	Mitigation of other attacks	N/A

Table 1 – Security levels

The Module has an overall security level of 1.

2. Cryptographic Module Specification

The Apple corecrypto Module v13.0 [Intel, Kernel, Software, SL1] cryptographic module (hereafter referred to as "the Module") is a software module running on a multi-chip standalone general-purpose computing platform. The version of module is 13 written as v13.0. The module provides implementations of low-level cryptographic primitives to the Host OS's (macOS Ventura v13) Security Framework and CommonCrypto. The module has been tested by Lightship Security, Inc. CST lab on the following platforms with and without PAA:

#	Operating System	Hardware Platform	Processor	PAA/Acceleration
1	macOS Ventura v13	MacBook Air (2022)	Intel i5 (Amber Lake)	PAA
2	macOS Ventura v13	MacBook Air (2022)	Intel i7 (Ice Lake)	PAA
3	macOS Ventura v13	MacBook Pro (2022)	Intel i7 (Coffee Lake)	PAA
4	macOS Ventura v13	iMac (2022)	Intel i7 (Comet Lake)	PAA
5	macOS Ventura v13	MacBook Pro (2022)	Intel i9 (Coffee Lake)	PAA
6	macOS Ventura v13	iMac Pro (2022)	Xeon W SkyLake	PAA
7	macOS Ventura v13	Mac Pro (2022)	Xeon W Cascade Lake	PAA
8	macOS Ventura v13	Mac Pro (2022)	Intel i5 (Coffee Lake)	PAA
9	macOS Ventura v13	MacBook Air (2022)	Intel i5 (Amber Lake)	No
10	macOS Ventura v13	MacBook Air (2022)	Intel i7 (Ice Lake)	No
11	macOS Ventura v13	MacBook Pro (2022)	Intel i7 (Coffee Lake)	No
12	macOS Ventura v13	iMac (2022)	Intel i7 (Comet Lake)	No
13	macOS Ventura v13	MacBook Pro (2022)	Intel i9 (Coffee Lake)	No
14	macOS Ventura v13	iMac Pro (2022)	Xeon W SkyLake	No
15	macOS Ventura v13	Mac Pro (2022)	Xeon W Cascade Lake	No
16	macOS Ventura v13	Mac Pro (2022)	Intel i5 (Coffee Lake)	No

Table 2 – Tested operational environments.

In addition to the platforms listed above, Apple Inc. has also tested the module on the following platforms and claims vendor affirmation on them:

#	Operating System	Hardware Platform
1	macOS Ventura v13	MacBook Pro - i5 (Ice Lake), 2021, 2020
2	macOS Ventura v13	MacBook Pro - i5 (Coffee Lake), 2021, 2020, 2019, 2018
3	macOS Ventura v13	MacBook Pro - i7 (Amber Lake), 2021, 2019, 2018
4	macOS Ventura v13	MacBook Pro - i7 (Coffee Lake), 2021, 2020, 2019, 2018
5	macOS Ventura v13	MacBook Pro - i7 (Ice Lake), 2021, 2020
6	macOS Ventura v13	MacBook Pro - i9 (Coffee Lake), 2021, 2019, 2018
7	macOS Ventura v13	MacBook Air - i5 (Ice Lake), 2021, 2020
8	macOS Ventura v13	MacBook Air - i7 (Ice Lake), 2021, 2020
9	macOS Ventura v13	MacBook Air - i5 (Amber Lake), 2021, 2019, 2018
10	macOS Ventura v13	MacBook Air - i7 (Amber Lake), 2021, 2018
11	macOS Ventura v13	Mac mini - i5 (Coffee Lake), 2021, 2018
12	macOS Ventura v13	Mac mini - i7 (Coffee Lake), 2021, 2018

13	macOS Ventura v13	iMac - i5 (Comet Lake), 2021, 2020
14	macOS Ventura v13	iMac - i7 (Comet Lake), 2021, 2020
15	macOS Ventura v13	iMac - i9 (Comet Lake), 2021, 2020
16	macOS Ventura v13	iMac - i5 (Coffee Lake), 2021, 2019
17	macOS Ventura v13	iMac - i7 (Coffee Lake), 2021, 2019
18	macOS Ventura v13	iMac - i9 (Coffee Lake), 2021, 2019
19	macOS Ventura v13	iMac - i9 (Comet Lake), 2022

Table 3 – Affirmed operational environments.

The CMVP makes no statement as to the correct operation of the module or the security strengths of the generated keys when so ported if the specific operational environment is not listed on the validation certificate.

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. 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. 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 9 – Non-approved services. If the device starts up successfully, then the module has passed all self-tests and is operating in the Approved mode.

CAVP Cert	Algorithm and Standard	Mode / Method	Description / Key Size / Key Strength	Use / Function
A3618 (asm_aesni) A3619 (asm_x86) A3620 (c_aesni) A3621 (c_asm)	AES [FIPS 197] [SP 800-38A]	CBC	128, 192, 256	Symmetric encryption and decryption
A3626 (vng_asm) A3627 (vng_aesni)	AES [FIPS 197] [SP 800-38C]	CCM	128, 192, 256	Authenticated encryption and decryption
A3620 (c_aesni) A3621 (c_asm)	AES [FIPS 197] [SP 800-38A]	CFB128	128, 192, 256	Symmetric encryption and decryption
A3620 (c_aesni) A3621 (c_asm)	AES [FIPS 197] [SP 800-38A]	CFB8	128, 192, 256	Symmetric encryption and decryption
A3620 (c_aesni) A3621 (c_asm) A3626 (vng_asm) A3627 (vng_aesni)	AES [FIPS 197] [SP 800-38A]	CTR	128, 192, 256	Symmetric encryption and decryption
A3618 (asm_aesni) A3619 (asm_x86) A3620 (c_aesni) A3621 (c_asm) A3626 (vng_asm) A3627 (vng_aesni)	AES [FIPS 197] [SP 800-38A]	ECB	128, 192, 256	Symmetric encryption and decryption
A3626 (vng_asm) A3627 (vng_aesni)	AES [FIPS 197] [SP 800-38D]	GCM	128, 192, 256	Authenticated encryption and decryption
A3620 (c_aesni)	AES	KW	128, 192, 256	Key wrapping

A3621 (c_asm)	[FIPS 197] [SP 800-38F]			
A3620 (c_aesni) A3621 (c_asm)	AES [FIPS 197] [SP 800-38A]	OFB	128, 192, 256	Symmetric encryption and decryption
A3618 (asm_aesni) A3619 (asm_x86)	XTS-AES [FIPS 197] [SP 800-38E]	XTS	128, 256	Symmetric encryption and decryption on storage devices
A3620 (c_aesni) A3621 (c_asm) A3626 (vng_asm) A3627 (vng_aesni)	CTR_DRBG [SP 800-90Ar1]	AES-CTR	Key Length/ Key Strength: 128, 256 Derivation Function Enabled: Yes	Random Number Generation
A3622 (c_avx) A3623 (c_avx2) A3624 (c_sse3)	ECDSA [FIPS 186-4]	KeyGen, KeyVer, SigGen, SigVer	Curves: P-224, P-256, P-384, P-521 Key Strength: from 112 to 256	Digital signatures and asymmetric key generation and verification
Vendor Affirmed	CKG	Key Pair Generation (CKG) using method in Sections 4 and 5.1 in [SP 800-133r2]	-	Cryptographic key generation
A3622 (c_avx) A3623 (c_avx2) A3624 (c_sse3) A3628 (vng_intel)	HMAC [FIPS 198-1]	HMAC-SHA-1 HMAC-SHA2-224 HMAC-SHA2-256 HMAC-SHA2-384 HMAC-SHA2-512	112 bits or greater	Message authentication
A3622 (c_avx) A3623 (c_avx2) A3624 (c_sse3)	HMAC [FIPS 198-1]	HMAC-SHA2-512/256	112 bits or greater	Message authentication
A3622 (c_avx) A3623 (c_avx2) A3624 (c_sse3)	HMAC_DRBG [SP 800-90Ar1]	SHA-1 SHA2-224 SHA2-256 SHA2-384 SHA2-512	112 bits or greater	Random Number Generation
A3624 (c_sse3)	KBKDF[1] [SP 800-108r1]	Counter (CTR) Feedback	HMAC-SHA-1 HMAC-SHA2-224 HMAC-SHA2-256 HMAC-SHA2-384 HMAC-SHA2-512	Key-based key derivation
A3622 (c_avx) A3623 (c_avx2) A3624 (c_sse3)	RSA [FIPS 186-4]	KeyGen (ANSI X9.31), SigGen (PKCS#1 v1.5) and (PKCS PSS) SigVer (PKCS#1 v1.5) and (PKCS PSS)	KeyGen: 2048, 3072, 4096 SigGen: 2048, 3072, 4096 SigVer: 1024 (legacy use), 2048, 3072, 4096	Digital signatures and asymmetric key generation and verification
A3622 (c_avx) A3623 (c_avx2) A3624 (c_sse3) A3628 (vng_intel)	SHS [FIPS 180-4]	SHA-1 SHA2-224 SHA2-256 SHA2-384 SHA2-512	112 bits or greater	Message digest
A3622 (c_avx) A3623 (c_avx2) A3624 (c_sse3)	SHS [FIPS 180-4]	SHA2-512/256	112 bits or greater	Message digest
A3625 (c_ltc)	Triple-DES	ECB	Keying Option: 1	Symmetric decryption

[1] KBKDF: Supported Lengths: 8 to 4096 in increments of 8. Fixed Data Order: Before Fixed Data. When using HMAC as the PRF.

Table 4 – Approved algorithms

The module does not have non-Approved algorithms used in the Approved mode of operation (with or without security claimed).

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

Algorithm	Use / Function
RSA	ANSI X9.31 Key Pair Generation Key Size < 2048 PKCS#1 v1.5 and PSS Signature Generation Key Size < 2048 PKCS#1 v1.5 and PSS Signature Verification Key Size< 1024
RSA	Key Encapsulation: OAEP, PKCS#1 v1.5 and PSS schemes
X25519	Key Agreement Key Generation
Ed25519	Key Generation Signature Generation Signature Verification
ANSI X9.63 KDF	Hash based Key Derivation Function
RFC6637	Key Derivation Function
HKDF [SP 800-56C]	Key Derivation Function
DES	Encryption / Decryption, Key Size: 56-bits
CAST5	Encryption / Decryption, Key Sizes: 40 to 128-bits in 8-bit increments
RC4	Encryption / Decryption, Key Sizes: 8 to 4096-bits
RC2	Encryption / Decryption Key Sizes 8 to 1024-bits
MD2	Message Digest, Digest size 128-bit
MD4	Message Digest, Digest size 128-bit
MD5	Message Digest, Digest size 128-bit
RIPEMD	Message Digest, Digest size 160-bits
ECDSA	Key-pair generation: Curve P-192 Public key validation: Curve P-192 Signature Generation: Curve P-192 Signature Verification: Curve P-192 Key Pair Generation for compact point representation of points
Integrated Encryption Scheme on elliptic curves (ECIES)	Encryption / Decryption
Blowfish	Encryption / Decryption
OMAC (One-Key CBC MAC)	MAC generation
Triple-DES [SP 800-67r2][2]	CBC, ECB: Encryption/Decryption Note: The module does not enforce the limit of 2^{16} encryptions with the same Triple-DES key, as required by FIPS 140-3 IG C.G.

Table 5 – Non-approved algorithms Not Allowed in the Approved Mode of Operation

[2] Triple-DES encryption/decryption was tested as part of CAVP algorithm testing, but is not utilized for any services implemented/supported by the module in Approved mode of operation.

The Apple corecrypto Module v13.0 [Intel, Kernel, Software, SL1] executes within the kernel space of the computing platforms and operating systems listed in Table 2 – Tested operational environments. Figure 1 below shows the logical block diagram[3] representing the following information:

- The location of the logical object of the 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; and
- The interactions of the logical object of the module with the operating system and other supporting applications resident within the cryptographic boundary.

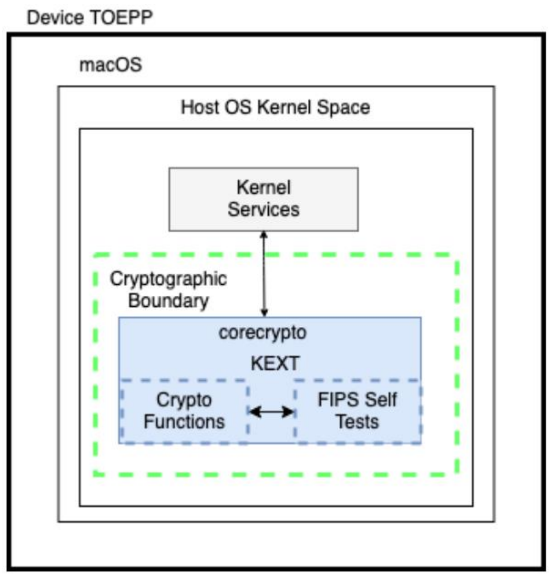

Figure 1 – Logical block diagram

[3] Kernel extension (KEXT) is a bundle that performs low-level tasks. KEXTs run in kernel space, which gives them elevated privileges and the ability to perform tasks that user-space apps cannot.

3. Cryptographic Module Interfaces

As a software-only module, the module does not have physical ports. For the purpose of the FIPS 140-3 validation, the physical ports are interpreted to be the physical ports of the hardware platform on which it runs.

The underlying logical interfaces of the module are the C language Kernel Interfaces (KPIs). In detail these interfaces are described in the table below.

Physical port	Logical interface	Data that passes over port / interface
N/A	Data Input	Data inputs are provided in the variables passed in the KPI and callable service invocations, generally through caller-supplied buffers
N/A	Data Output	Data outputs are provided in the variables passed in the KPI and callable service invocations, generally through caller-supplied buffers
N/A	Control Input	Control inputs which control the mode of the module are provided through dedicated parameters, namely the kernel module plist whose information is supplied to the module by the kernel module loader.
N/A	Control Output	Not Applicable[4]
N/A	Status Output	Status output is provided in return codes and through messages. Documentation for each KPI lists possible return codes. A complete list of all return codes returned by the C language KPIs within the module is provided in the header files and the KPI documentation. Messages are also documented in the KPI documentation.

Table 6 – Ports and interfaces

The module is optimized for library use within the macOS kernel space and does not contain any terminating assertions or exceptions. It is implemented as a macOS dynamically loadable library. The dynamically loadable library is loaded into the macOS kernel and its cryptographic functions are made available to macOS kernel services only. Any internal error detected by the module is reflected back to the caller with an appropriate return code. The calling macOS kernel service must examine the return code and act accordingly.

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 140-3 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] The Module does not output control information, and thus has no specified control output interface.

4. Roles, Services and Authentication

The Module supports a single instance of one authorized role, designated as the Crypto-Officer. No support is provided for multiple concurrent operators or a Maintenance Operator.

The table below lists the services available to the Crypto-Officer:

Role	Service	Input	Output
Crypto-Officer (CO)	Symmetric encryption	AES Key Plain text data	Cipher text
Crypto-Officer (CO)	Symmetric decryption	AES Key Cipher text data	Plain text
Crypto-Officer (CO)	Key wrapping	Key-encryption-key Key to be wrapped	Wrapped key
Crypto-Officer (CO)	Key unwrapping	Key-encryption-key Wrapped key	Unwrapped key
Crypto-Officer (CO)	Secure Hashing	Message	Message digest
Crypto-Officer (CO)	Message Authentication Code (MAC) Generation	message, MAC key, MAC algorithm	Message Authentication Code
Crypto-Officer (CO)	Message Authentication Code (MAC) Verification	MAC, message, HMAC key, MAC algorithm	pass/fail result
Crypto-Officer (CO)	Generate asymmetric key pair	Random numbers, domain parameters	Public/private key pair
Crypto-Officer (CO)	Generate digital signature	private key, message, hash function	Digital signature
Crypto-Officer (CO)	Verify digital signature	public key	True or False
Crypto-Officer (CO)	Generate random number	entropy, seed, V and key values	random bit-string
Crypto-Officer (CO)	Derive key via KBKDF	KBKDF Key Derivation Key	Derived key
Crypto-Officer (CO)	Zeroise symmetric keys	Handler of symmetric crypto function context	Released memory space
Crypto-Officer (CO)	Zeroise asymmetric keys	Handler of asymmetric crypto function context	Released memory space
Crypto-Officer (CO)	Zeroise hash	Handler of hash context	Released memory space
Crypto-Officer (CO)	Self-test	Instantiation	Status
Crypto-Officer (CO)	Show status	KPI invocation	Operational / error status
Crypto-Officer (CO)	Show module info	KPI invocation	Module base name Module version

Table 7 – Roles, Services, Input and Output

4.1 Authentication

FIPS 140-3 does not require an authentication mechanism for level 1 modules. Therefore, the module does not implement an authentication mechanism for Crypto Officer. The Crypto Officer role is authorized to access all services provided by the module (see Table 8 – Approved services and Table 9 – Non-approved services below).

4.2 Services

The module implements a dedicated KPI function to indicate if a requested service utilizes an approved security function. For services listed in Table 8 – Approved services, the indicator function returns 1. For services listed in Table 9 – Non-approved services, the indicator function returns 0.

The table below lists all approved services that can be used in the approved mode of operation. 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= The service does not access any SSP during its operation

Service	Description	Approved Security Functions	Keys and/or SSPs	Roles	Access rights to Keys and/or SSPs	Indicator
ECDSA key pair generation	Generate a public/private key pair	ECDSA[5] CKG	ECDSA key pair	CO	GR	1
ECDSA signature generation	Generate a digital signature	ECDSA	ECDSA private key	CO	RE	1
ECDSA signature verification	Verify a digital signature	ECDSA	ECDSA public key	CO	RWE	1
Derive key via KBKDF	Derive keys	KBKDF	KBKDF Key derivation key KBKDF Derived key	CO	WE GRE	1
Key wrapping	Perform key wrapping	AES-KW	AES Key-encrypting key	CO	WE	1
Key unwrapping	Perform key unwrapping	AES-KW	AES Key-encrypting key	CO	WE	1
Hashing	Compute a message digest	SHA-1 SHA2-224 SHA2-256 SHA2-384 SHA2-512 SHA2-512/256	N/A	CO	N/A	1
MAC Generation	Compute a message authentication code	HMAC	HMAC key	CO	WE	1
MAC Verification	Verify a message authentication code	HMAC	HMAC key	CO	WE	1
RSA key pair generation	Generate a public/private key pair	RSA[6] CKG	RSA key pair	CO	GR	1
RSA signature generation	Generate a digital signature	RSA	RSA private key	CO	RE	1

[5] In accordance with Section 4 and 5.1 of NIST [SP 800-133r2] (CKG), the module uses its approved DRBG to generate random bits and seeds used to generate asymmetric keys. Each generated seed is an unmodified output from the DRBG.

[6] In accordance with Section 4 and 5.1 of NIST [SP 800-133r2] (CKG), the module uses its approved DRBG to generate random bits and seeds used to generate asymmetric keys. Each generated seed is an unmodified output from the DRBG.

RSA signature verification	Verify a digital signature	RSA	RSA public key	CO	RWE	1
Random number generation	Generate a random number	CTR_DRBG	DRBG entropy input DRBG seed DRBG 'V' value DRBG 'Key' value	CO	Input: WE Seed: GE V: GE Key: GE	1
Self-test	Perform pre-operational and algorithm self-test	N/A	N/A	CO	N/A	1
Show status	Return module status	N/A	N/A	CO	N/A	N/A
Show module info	Return module name and versioning information	N/A	N/A	CO	N/A	N/A
Symmetric encryption	Encrypt plaintext data	AES-CBC AES-CCM AES-CFB128 AES-CFB8 AES-CTR AES-ECB AES-OFB AES-GCM XTS-AES	AES Key AES GCM key AES XTS key	CO	WE	1
Symmetric decryption	Decrypt ciphertext data	AES-CBC AES-CCM AES-CFB128 AES-CFB8 AES-CTR AES-ECB AES-OFB AES-GCM XTS-AES	AES Key AES GCM key AES XTS key	CO	WE	1
Zeroise symmetric keys	Release all resources of symmetric crypto function context	N/A	AES Key AES GCM key AES XTS key AES Key-encrypting key KBKDF Key derivation key KBKDF Derived key	CO	Z	1
Zeroise asymmetric keys	Release of all resources of asymmetric crypto function context	N/A	RSA key pair ECDSA key pair	CO	Z	1
Zeroise hash	Release all resources of hash context	N/A	HMAC key	CO	Z	1

Table 8 – Approved services

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

Service	Description	Algorithms Accessed	Role	Indicator
Key Encapsulation	Perform key wrapping	RSA encrypt	CO	0
Key Decapsulation	Perform key unwrapping	RSA decrypt	CO	0
EdDSA/X25519 key pair generation	Generate a public/private key pair Curve: Ed25519 and Curve25519	EdDSA KeyGen X25519 KeyGen	CO	0

EdDSA signature generation	Generate a digital signature Curve: Ed25519	EdDSA SigGen	CO	0
EdDSA signature verification	Verify a digital signature Curve: Ed25519	EdDSA SigVer	CO	0
X25519 key agreement	Perform key agreement	X25519 Key agreement	CO	0
ECDSA key pair generation	Generate a public/private key pair Curve: P-192	ECDSA	CO	0
ECDSA signature generation	Generate a digital signature Curve: P-192	ECDSA	CO	0
ECDSA Signature verification	Verify a digital signature Curve: P-192	ECDSA	CO	0
RSA key pair generation	Generate a public/private key pair Key size < 2048	ANSI X9.31	CO	0
RSA signature generation	Generate a digital signature Key size < 2048	PKCS#1 v1.5 PSS	CO	0
RSA Signature verification	Verify a digital signature Key size < 1024	PKCS#1 v1.5 PSS	CO	0
Key Derivation	Hash-based key derivation Per: ANSI X9.63	SHA-1	CO	0
Key Derivation	Hash-based key derivation Per: SP 800-56C HKDF	SHA-256	CO	0
Key Derivation	Hash-based key derivation Per: RFC6637	SHA-256, SHA-512, AES-128, AES-256	CO	0
Generate MAC	Compute a message authentication code	OMAC	CO	0
Hashing	Compute a message digest	MD2, MD4, MD5, RIPEMD	CO	0
Symmetric encryption	Perform symmetric data encryption	Triple-DES, Blowfish, CAST5, DES, ECIES, RC2, RC4	CO	0
Symmetric decryption	Perform symmetric data decryption	Triple-DES, Blowfish, CAST5, DES, ECIES, RC2, RC4	CO	0

Table 9 – Non-approved services

5. Software/Firmware Security

5.1 Integrity Techniques

The Apple corecrypto Module v13.0 [Intel, Kernel, Software, SL1], which is made up of a single component, is provided in the form of binary executable code. A software integrity test is performed on the runtime image of the module. The HMAC-SHA2-256 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. In this state the module is not operational.

5.2 On Demand Integrity Test

Integrity tests are performed as part of the Pre-Operational Self-Tests. The software integrity test is automatically executed at power-on. It can also be invoked by self-test service or powering-off and reloading the module.

6. Operational Environment

6.1 Applicability

The Apple corecrypto Module v13.0 [Intel, Kernel, Software, SL1] operates in a modifiable operational environment per FIPS 140-3 level 1 specifications. The module is supplied as part of macOS, a commercially available general-purpose operating system executing on the computing platforms specified in section 2.

7. Physical Security

The FIPS 140-3 physical security requirements do not apply to the Apple corecrypto Module v13.0 [Intel, Kernel, Software, SL1] since it is a software module.

8. Non-invasive Security

Currently, the ISO/IEC 19790:2012 non-invasive security area 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	Genera-tion	Import /Export	Establishm e-nt	Storage	Zeroisation	Use & related keys
AES key	128 to 256 bits	AES (CBC, CCM, CFB, CTR, ECB, KW, OFB modes) A3618 (asm_aesni) A3619 (asm_x86) A3620 (c_aesni) A3621 (c_asm) A3626 (vng_asm) A3627 (vng_aesni)	N/A	Imported from kernel No export	N/A	N/A. The module does not provide persistent keys/SSPs storage.	Automatic zeroisation when structure is deallocated or when the system is powered down	Symmetric Encryption and Decryption
AES Key-encrypting key	128 to 256 bits	AES-KW A3620 (c_aesni) A3621 (c_asm)	N/A	Imported from kernel No export	N/A			Key Wrapping and Unwrapping (KTS)
AES GCM key	128 to 256 bits	AES (GCM mode) A3626 (vng_asm) A3627 (vng_aesni)	N/A	Imported from kernel No export	N/A			Symmetric Encryption and Decryption
AES XTS key	128 and 256 bits	XTS-AES A3618 (asm_aesni) A3619 (asm_x86)	N/A	Imported from kernel No export	N/A			Symmetric Encryption and Decryption
HMAC key	>=112 bits	HMAC-SHA-1 HMAC-SHA-2 A3622 (c_avx) A3623 (c_avx2) A3624 (c_sse3) A3628 (vng_intel)	N/A	Imported from kernel No export	N/A			Generate and Verify MAC
ECDSA public key	112 to 256 bits	ECDSA A3622 (c_avx) A3623 (c_avx2) A3624 (c_sse3)	The key pairs are generated conformant to [SP 800-133r2] Section 4 (CKG) using FIPS186-4 Key Generation method, and the random value used in the key generation is generated using [SP 800-90Ar1] DRBG	Imported from or exported to kernel	N/A			Signature verification
ECDSA private key				Exported to kernel Intermediate keygen values are not output	N/A			Signature generation
RSA public key	112 to 256 bits	RSA A3622 (c_avx) A3623 (c_avx2) A3624 (c_sse3)	The key pairs are generated conformant to [SP 800-133r2] Section 4 (CKG) using FIPS186-4 Key Generation	Imported from or exported to kernel	N/A			Signature verification
RSA private key				Exported to kernel. Intermediate keygen values are not output	N/A			Signature generation

			method, and the random value used in the key generation is generated using [SP 800-90Ar1] DRBG					
Entropy Input String	256 bits	Random Number Generation	N/A	Imported from entropy source	N/A			Random Number Generation
DRBG Seed, internal state: V value and Key	256 bits	CTR_DRBG A3620 (c_aesni) A3621 (c_asm) A3627 (vng_aesni) HMAC_DRBG A3622 (c_avx) A3623 (c_avx2) A3624 (c_sse3)	Internally generated as defined by [SP 800-90Ar1]	N/A	N/A			Random Number Generation
KBKDF Key derivation key	Min: 112 bits	KBKDF A3624 (c_sse3)	N/A	Imported from calling application No export	N/A			Key Derivation
KBKDF Derived key	Min: 112 bits	KBKDF A3624 (c_sse3)	Generated via KBKDF	No import Exported to calling application	N/A			Key Derivation

Table 10 – SSPs

9.1 Random Number Generation

A NIST approved deterministic random bit generator based on a block cipher as specified in NIST [SP 800-90Ar1] is used. The default Approved DRBG used for random number generation is a CTR_DRBG using AES-256 with derivation function and without 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 800-90Ar1], the internal state of CTR_DRBG is the value V and Key.

The module also employs a HMAC_DRBG for random number generation. The HMAC_DRBG is only used at the early boot time of macOS kernel for memory randomization. The output of HMAC_DRBG is not used for key generation. Per section 10.1.2.1 of [SP 800-90Ar1], the internal state of HMAC_DRBG is the value V, Key. The deterministic random bit generators are seeded by “read_random”. The read_random is the Kernel Space interface. Two entropy sources (one non-physical entropy source and one physical entropy source) residing within the TOEPP provide the random bits. The output of entropy pool provides 256-bits of entropy to seed and reseed [SP 800-90Ar1] DRBG during initialization (seed) and reseeding (reseed).

For both Apple Entropy sources tested in the OEs listed in Table 2, the customer does not have the ability to modify the ES configuration settings (see details in Public Use Document referenced in 3 and 4).

The module also performs DRBG health tests according to section 11.3 of [SP 800-90Ar1].

Entropy source	Minimum number of bits of entropy	Details
ESV Cert #E14 (physical source) ESV Cert #E110 (non-physical source)	256	The seed is provided by post-processed entropy data from two entropy sources. The entropy sources are located within the physical perimeter of the module but outside the

		cryptographic boundary of the module.

Table 11 – Non-Deterministic Random Number Generation Specification

9.2 Key / SSP Generation

The module generates Keys and SSPs in accordance with FIPS 140-3 IG D.H. The cryptographic module performs Cryptographic Key Generation (CKG) for asymmetric keys as per [SP 800-133r2] Section 4 (vendor affirmed), compliant with [FIPS186-4], and using DRBG compliant with [SP 800-90Ar1]. A seed (the random value) used in asymmetric key generation is obtained from [SP 800-90Ar1] DRBG. The key generation service for RSA, ECDSA, as well as the [SP 800-90Ar1] DRBG have been ACVTS tested with algorithm certificates found in Table 4 – Approved algorithms.

The module also implements KBKDF key derivation according to [SP 800-108r1] to derive symmetric keys. The module supports both Counter and Feedback modes with HMAC-SHA-1, HMAC-SHA2-224, HMAC-SHA2-256, HMAC-SHA2-384, or HMAC-SHA2-512 as the pseudo-random function (PRF).

9.3 Key / SSP Establishment

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

- AES Key Wrapping

The module implements a Key Transport Scheme (KTS) using AES-KW compliant to [SP 800-38F], IG D.G. The SSP establishment methodology provides between 128 and 256 bits of encryption strength.

9.4 Key / SSP Import/Export

All keys and SSPs that are entered from, or output to module, are entered from or output to the invoking application running on the same device. Keys/SSPs entered into the module are electronically entered in plain text form. Keys/SSPs are output from the module in plain text form if required by the calling application.

The module allows the output of plaintext CSPs (for example: ECDSA or RSA Key Pairs). To prevent inadvertent output of sensitive information, the module performs the following two independent internal actions:

1. The module will internally request the random number generation service to obtain the random numbers and verify that the service completed without errors.
2. Once the keys are generated the module will perform the pairwise consistency test and verify that the test is completed without errors.

Only after successful completion of both actions, are the generated CSPs output via the KPI output parameter in plaintext.

9.5 Key / SSP Storage

The Apple corecrypto Module v13.0 [Intel, Kernel, Software, SL1] stores ephemeral keys/SSPs in memory only. They are received for use or generated by the module only at the command of the calling application. The module does not provide persistent keys/SSPs storage.

The module protects all keys/SSPs through the memory separation and protection mechanisms provided by the operating system. No process other than the module itself can access the keys/SSPs in its process' memory.

9.6 Key / SSP Zeroisation

Keys and SSPs are zeroised when the appropriate context object is destroyed or when the system is powered down. Input and output interfaces are inhibited while zeroisation is performed.

10. Self-tests

This section specifies the pre-operational and conditional self-tests performed by the module. The pre-operational and conditional self-tests ensure that the module is not corrupted and that the cryptographic algorithms work as expected.

The module does not implement a bypass mode nor security functions critical to the secure operation of the cryptographic module and thus, does not implement either a pre-operational bypass test or pre-operational critical functions test.

While the module is executing the self-tests, services are not available and input and output are inhibited. If the module fails pre-operational or conditional self-tests fail, it will report an error message indicating the cause of the failure and enters the Error State (See section 10.3). The module permits operators to initiate the pre-operational or conditional self-tests on demand for periodic testing of the module by rebooting the system (i.e., power-cycling).

10.1 Pre-operational Software Integrity Test

The module performs a pre-operational software integrity test automatically when the module is loaded into memory (i.e., at power on) before the module transitions to the operational state. A software integrity test is performed on the runtime image of the Apple corecrypto Module v13.0 [Intel, Kernel, Software, SL1] with HMAC-SHA2-256 used to perform the approved integrity technique. Prior to using HMAC-SHA-256, Conditional Cryptographic Algorithm Self-Test (CAST) is performed. If the CAST on the HMAC-SHA-256 is successful, the HMAC value of the runtime image is recalculated and compared with the stored HMAC value pre-computed at compilation time.

10.2 Conditional Self-Tests

Conditional self-tests are to be performed by a cryptographic module when the conditions specified for the following tests occur: Cryptographic Algorithm Self-Test, Pair-Wise Consistency Test.

The module does not implement any functions requiring a Software/Firmware Load Test, Manual Entry Test, Conditional Bypass Test nor Conditional Critical Functions Test; therefore, these tests are not performed by the module.

The following sub-sections describe the conditional tests supported by the Apple corecrypto Module v13.0 [Intel, Kernel, Software, SL1].

10.2.1. Conditional Cryptographic Algorithm Self-Tests

In addition to the pre-operational software integrity test described in Section 10.1, the Apple corecrypto Module v13.0 [Intel, Kernel, Software, SL1] also runs the Conditional Cryptographic Algorithm Self-Tests (CAST) for all cryptographic functions of each approved cryptographic algorithm implemented by the module during power-up as well. All CASTs are performed prior to the first operational use of the cryptographic algorithm. These tests are detailed in Table 12 – Conditional Cryptographic Algorithm Self-tests below.

Algorithm(s)	Notes
HMAC-SHA256	CAST (KAT) performed prior to module's integrity test during POSTs
AES implementations selected by the module for the	Separate encryption / decryption operations are performed

corresponding environment AES-CCM, AES-GCM, AES-XTS, AES-CBC, AES-ECB, AES-KW using 128-bit key	
CTR_DRBG and HMAC_DRBG	Each DRBG mode tested separately KAT and Health test per NIST [SP 800-90Ar1] Section 11.3
HMAC-SHA-1, HMAC-SHA2-256, HMAC-SHA2-512	KAT
SHA-1, SHA2-256, SHA2-512	Covered by high level HMAC self-test
RSA, 2048-bit modulus with SHA-256	Separate Signature generation/ verification KAT are performed
ECDSA, P-256 curve with SHA-256	Signature generation/verification are performed
KBKDF (counter and feedback modes)	KAT

Table 12 – Conditional Cryptographic Algorithm Self-tests

10.2.2. Conditional Pairwise Consistency Test

The Apple corecrypto Module v13.0 [Intel, Kernel, Software, SL1] does generate RSA and ECDSA asymmetric keys and performs the required RSA and ECDSA pair-wise consistency tests on the newly generated key pairs.

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 being reloaded into memory and reperforming the pre-operational software integrity test and the Conditional CASTs. The module will only enter into the operational state after successfully passing the pre-operational software 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	print statement "FAILED: fipspost_post_integrity" to stdout
Failed Conditional CAST	print statement "FAILED: <event>" to stdout (<event> refers to any of the cryptographic functions listed in Table 12 – Conditional Cryptographic Algorithm Self-tests.)
Failed Conditional PCT	Error code "CCEC_GENERATE_KEY_CONSISTENCY" returned for ECDSA Error code "CCRSA_GENERATE_KEY_CONSISTENCY" returned for RSA

Table 13 – Error Indicators

11. Life-cycle Assurance

11.1 Delivery and Operation

The module is built into macOS Ventura v13 and delivered with device. There is no standalone delivery of the module as a software library.

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

This digital signature-based integrity protection during the delivery/integration process is not to be confused with the HMAC-SHA2-256 based integrity check performed by the module itself as part of its pre-operational self-tests.

11.2 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 9 – Non-approved services. If the device starts up successfully, then the module has passed all self-tests and is operating in the Approved mode.

The ESV Public Use Document (PUD) reference for physical entropy source is:
https://csrc.nist.gov/CSRC/media/projects/cryptographic-module-validation-program/documents/entropy/E14_PublicUse.pdf

The ESV Public Use Document (PUD) reference for non-physical entropy source is:
https://csrc.nist.gov/CSRC/media/projects/cryptographic-module-validation-program/documents/entropy/E110_PublicUse.pdf

Apple Platform Certifications guide [platform certifications] and Apple Platform Security guide [SEC] are 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.

The Crypto Officer shall consider the following requirements and restrictions when using the module:

- AES-GCM IV is constructed in compliance with IG C.H scenario 1. The GCM IV generation follows RFC 4106 and shall only be used for the IPsec protocol version 3. When the IV in RFC 4106 exhausts the maximum number of possible values for a given security association, either party to the security association that encounters this condition triggers a rekeying with IKEv2 to establish a new encryption key for the security association. The module uses RFC 7296 compliant IKEv2 to establish the shared secret SKEYSEED from which the AES-GCM encryption keys are derived. In case the module's power is lost and then restored, the key used for the AES GCM encryption/decryption shall be re-distributed. This condition is not enforced by the module. This protocol has not been reviewed or tested by the CAVP and CMVP.
- AES-XTS mode is only approved for hardware storage applications. The length of the AES-XTS data unit does not exceed 2^{20} blocks. The module checks explicitly that Key_1 ≠ Key_2 before using the keys in the XTS-Algorithm to process data with them compliant with IG C.I.

12. Mitigation of Attacks

The module does not claim mitigation of other attacks.

Appendix A. Glossary and Abbreviations

AES	Advanced Encryption Standard
AES-NI	Advanced Encryption Standard New Instructions
CAVP	Cryptographic Algorithm Validation Program
CAST	Cryptographic Algorithm Self-Test
CAST5	A symmetric-key 64-bit block cipher with 128-bit key
CBC	Cipher Block Chaining
CCM	Counter with Cipher Block Chaining-Message Authentication Code
CFB	Cipher Feedback
CMVP	Cryptographic Module Validation Program
CSP	Critical Security Parameter
CTR	Counter Mode
DRBG	Deterministic Random Bit Generator
ECB	Electronic Code Book
ECDSA	Elliptic Curve Digital Signature Algorithm
ESVP	Entropy Source Validation Program
FIPS	Federal Information Processing Standards
GCM	Galois Counter Mode
HMAC	Hash Message Authentication Code
KAT	Known Answer Test
KBKDF	Key Based Key Derivation Function
KDF	Key Derivation Function
KEXT	Kernel Extension
KW	AES Key Wrap
MAC	Message Authentication Code
KPI	Kernel Programming Interface
NIST	National Institute of Standards and Technology
OAEP	Optimal Asymmetric Encryption Padding
OFB	Output Feedback
PAA	Processor Algorithm Acceleration
PCT	Pairwise Consistency Test
PKG	Key-Pair Generation
PKV	Public Key Validation
PRF	Pseudo-Random Function
PSS	Probabilistic Signature Scheme
PUD	Public Use Document (ESVP)
RSA	Rivest, Shamir, Adleman
SHA	Secure Hash Algorithm
SHS	Secure Hash Standard
TOEPP	Tested Operational Environment Physical Perimeter

XTS	XEX Tweakable Block Ciphertext Stealing

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**
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

PKCS#1 **Public Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1**
February 2003
http://www.ietf.org/rfc/rfc3447.txt

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

SP 800-38A **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

SP 800-38C **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

SP 800-38D **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

SP 800-38E **NIST Special Publication 800-38E - Recommendation for Block Cipher Modes of Operation: The XTS AES Mode for Confidentiality on Storage Devices**
January 2010
http://csrc.nist.gov/publications/nistpubs/800-38E/nist-sp-800-38E.pdf

SP 800-38F **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

SP 800-56Cr2 **Recommendation for Key-Derivation Methods in Key-Establishment Schemes**
August 2020
https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Cr2.pdf

SP 800-57r5 **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

SP 800-67r1 **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

SP 800-90Ar1 **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

SP 800-90B **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

SP 800-108 **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

SP 800-131Ar2 **NIST Special Publication 800-131A - Transitioning the Use of Cryptographic Algorithms and Key Lengths**
March 2019
https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-131Ar2.pdf

SP 800-132 **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

SP 800-133r2 **Recommendation for Cryptographic Key Generation**
June 2020
https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-133r2.pdf

SP 800-135r1 **NIST Special Publication 800-135 Revision 1 - Recommendation for Existing Application-Specific Key Derivation Functions**
December 2011
http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-135r1.pdf

MACOS **macOS Technical Overview**
https://developer.apple.com/macos/

SEC **Apple Platform Security Guide**
https://support.apple.com/guide/security/welcome/web

macOS **Product security certifications for macOS**
https://support.apple.com/HT201159