Apple Inc.

Apple corecrypto User Space Module for Intel (ccv10)
FIPS 140-2 Non-Proprietary Security Policy

Module Version 10.0

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 corecrypto User Space Module for Intel (ccv10). 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 the Cryptographic Module Validation Program please refer to the NIST website [CMVP].

Throughout the document "Apple corecrypto User Space Module for Intel (ccv10)", "cryptographic module", "corecrypto" or "the module" are used interchangeably to refer to the Apple corecrypto User Space Module for Intel (ccv10). macOS 10.15 is also referred as macOS Catalina or macOS Catalina 10.15. "ccv10" is used to refer to the module version 10.0.

1.2 Document Organization / Copyright

This non-proprietary Security Policy document may be reproduced and distributed only in its original entirety without any revision, ©2021 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 macOS and the associated security properties refer to [MACOS] and [SEC]. For details on macOS releases with their corresponding validated modules and Crypto Officer Role Guides refer to the "Product security certifications, validations, and guidance for macOS" [UGuide]

1.3.1 Additional References

CMVP Cryptographic Module Validation Program

https://csrc.nist.gov/projects/cryptographic-module-validation-program

CAVP Cryptographic Algorithm Validation Program

https://csrc.nist.gov/projects/cryptographic-algorithm-validation-program

FIPS 140-2 Federal Information Processing Standards Publication, "FIPS PUB 140-2 Security Requirements for Cryptographic Modules," May 2001

FIPS 140-2 IG NIST, "Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program," August, 2020

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 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 C NIST Special Publication 800-38C, “Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality”, May 2004

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

SP800-57P1 NIST Special Publication 800-57, “Recommendation for Key Management – Part 1: General (Revised),” July 2016

SP 800-90A NIST Special Publication 800-90A, “Recommendation for Random Number Generation Using Deterministic Random Bit Generators,” January 2012

SP800-132 NIST Special Publication 800-132, “Recommendation for Password-Based Key Derivation”, December 2010

MACOS macOS Technical Overview

https://developer.apple.com/macos/

SEC Security Overview

https://developer.apple.com/security/

UGuide User Guide

https://support.apple.com/HT201159

1.4 Acronyms

AES	Advanced Encryption Standard
CAVP	Cryptographic Algorithm Validation Program
CBC	Cipher Block Chaining mode of operation
CFB	Cipher Feedback mode of operation
CMVP	Cryptographic Module Validation Program
CSP	Critical Security Parameter
CTR	Counter mode of operation
DES	Data Encryption Standard
DH	Diffie-Hellman
DRBG	Deterministic Random Bit Generator
ECB	Electronic Codebook mode of operation
ECC	Elliptic Curve Cryptography
EC Diffie-Hellman	Diffie-Hellman 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
MAC	Message Authentication Code
NIST	National Institute of Standards and Technology
OFB	Output Feedback (mode of operation)
OS	Operating System
PBKDF	Password-based Key Derivation Function
PCT	Pair-wise Consistency Test
PRF	Pseudorandom Functions
RNG	Random Number Generator
SHS	Secure Hash Standard
Triple-DES	Triple Data Encryption Standard
TLS	Transport Layer Security

2 Cryptographic Module Specification

2.1 Module Description

The Apple corecrypto User Space Module for Intel (ccv10) is a software cryptographic module with version v10.0 running on a multi-chip standalone general-purpose computing platform.

The cryptographic services provided by the module are:

- data encryption / decryption
- generation of hash values
- key wrapping
- message authentication
- random number generation
- key generation
- signature generation / verification
- 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 1 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	N/A
Operational Environment	1
Cryptographic Key Management	1
EMI/EMC	1
Self-Tests	1
Design Assurance	1
Mitigation of Other Attacks	1

Table 1: Module Validation Level

2.1.2 Module components

In the following sections the components of the Apple corecrypto User Space Module for Intel (ccv10) are listed in detail. There are no components excluded from the validation testing.

2.1.2.1 Software components

corecrypto has an API layer that provides consistent interfaces to the supported algorithms. These implementations include proprietary optimizations of algorithms that are fitted into the corecrypto framework.

2.1.2.2 Hardware components

AES-NI hardware acceleration is included within the cryptographic module boundary.

2.1.3 Tested Platforms

The module has been tested on the following platforms with and without AES-NI:

Manufacturer	Model (Hardware, Processor)	Operating System
Apple Inc.	MacBook with Intel Core M	macOS Catalina 10.15
Apple Inc.	Mac mini with Intel Core i5	macOS Catalina 10.15
Apple Inc.	MacBook Pro with Intel Core i7	macOS Catalina 10.15
Apple Inc.	MacBook Pro with Intel Core i9	macOS Catalina 10.15
Apple Inc.	iMac Pro with Intel Xeon W	macOS Catalina 10.15

Table 2: Tested Platforms

In addition to the configurations tested by the laboratory, vendor-affirmed testing was performed on the following platforms for macOS 10.15 Catalina:

- MacBook, MacBook Air, MacBook Pro and iMac with an Intel i5
- Mac mini, MacBook Air, MacBook and iMac with an Intel i7
- iMac with an Intel i9

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 (IG G.5).

2.2 Modes of operation

The Apple corecrypto User Space Module for Intel (ccv10) has an Approved and non-Approved mode of operation. The Approved mode of operation is configured by default and cannot be changed. If the device starts up successfully the corecrypto framework 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.

The module transitions back into FIPS mode immediately when invoking one of the approved ciphers as all keys and Critical Security Parameters (CSPs) handled by the module are ephemeral and there are no keys and CSPs shared between any 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 configuration 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 automatically selects which cipher is used based on internal heuristics. This includes the hardware-assisted AES and SHA implementations (AES-NI).

The Approved security functions are listed in Table 3. The Algorithm Certificate numbers obtained from NIST are based on the successful ACVT validation testing of the cryptographic algorithm implementations of the module that runs on the hardware platforms referenced in Table 2.

Refer to [CAVP] for the current standards, test algorithm requirements, and special abbreviations used in the following Table 3.

2.2.1 Approved Security Functions:

Cryptographic Function	Algorithm	Options	Algorithm Certificate
Random Number Generation	[SP 800-90A] DRBG	CTR_DRBG Modes: AES-128, AES-256 Derivation Function Enabled	A7 (c_asm) A8 (c_ltc) A10 (vng_asm) A21 (c_aesni) A31 (vng_asni)
		HMAC_DRBG Modes: SHA-1, SHA-224, SHA-256, SHA-384, SHA-512 Without Prediction Resistance	A8 (c_ltc) A22 (c_avx) A27 (c_ssse3) A33 (c_avx2)
Symmetric Encryption and Decryption	[FIPS 197] AES SP 800-38 A SP 800-38 C SP 800-38 D SP 800-38 E	Key Length: 128, 192, 256 Modes: CBC ECB CCM GCM CFB128 OFB CFB8 XTS (128 and 256-bits key size only) CTR	A7 (c_asm) A8 (c_ltc) A21 (c_aesni)
		Key Length: 128, 192, 256 Modes CBC	A11 (c_glad)
		Key Length: 128, 192, 256 Modes: CBC XTS (128 and 256-bits key size only) ECB	A19 (asm_aesni) A25 (asm_x86)
		Key Length: 128, 192, 256 Modes: ECB GCM CCM CTR	A10 (vng_asm) A31 (vng_asni)
	[SP 800-67] Triple-DES	(Keying Option: 1; All Keys Independent) Modes: CBC CTR CFB64 ECB CFB8 OFB	A8 (c_ltc)
Key Wrapping	SP 800-38 D	Key Length: 128, 192, 256 Modes: AES-CCM AES-GCM	A7 (c_asm) A8 (c_ltc) A10 (vng_asm) A21 (c_aesni) A31 (vng_asni)
	SP 800-38 F	Key Length: 128, 192, 256 Modes: AES-KW	A7 (c_asm) A8 (c_ltc) A21 (c_aesni)

Cryptographic Function	Algorithm	Options	Algorithm Certificate
Digital Signature and Asymmetric Key Generation	[FIPS186-4] RSA	Key Generation (ANSI X9.31), Modulus: 2048, 3072, 4096 Signature Generation (PKCS#1 v1.5) and (PKCS PSS)Modulus: 2048, 3072, 4096 Signature Verification (PKCS#1 v1.5) and (PKCS PSS)Modulus: 1024, 2048, 3072, 4096	A8 (c_ltc) A22 (c_avx) A27 (c_ssse3) A33 (c_avx2)
	[FIPS 186-4] ECDSA ANSI X9.62	Key Pair Generation (PKG): P-224, P-256, P-384, P-521 Public Key Validation (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	A8 (c_ltc) A22 (c_avx) A27 (c_ssse3) A33 (c_avx2)
Message Digest	[FIPS 180-4] SHS	Modes: SHA-1 SHA-384 SHA-224 SHA-512 SHA-256	A8 (c_ltc) A22 (c_avx) A27 (c_ssse3) A29 (vng_Intel) A33 (c_avx2)
Keyed Hash	[FIPS 198] HMAC	Key size: 112 bits or greater Modes SHA-1 SHA-384 SHA-224 SHA-512 SHA-256	A8 (c_ltc) A22 (c_avx) A27 (c_ssse3) A29 (vng_Intel) A33 (c_avx2)
KAS FFC Component	[SP800-56A] DLC Primitive Diffie-Hellman	Public key size 2048-bits Private key size 256-bits	CVL: A8 (c_ltc)
KAS ECC Component	[SP800-56A] DLC Primitive EC Diffie-Hellman	NIST Curves: P-256, P-384	CVL: A8 (c_ltc)
Key Derivation	[SP 800-132] PBKDF	Password Based Key Derivation using HMAC with SHA-1 or SHA-224, SHA-256, SHA-384, SHA-512 PRFs	Vendor Affirmed
CKG	[SP800-133r2]	RSA Key Generation (ANSI X9.31), Modulus: 2048, 3072, 4096 ECDSA Key Pair Generation (PKG): P-224, P-256, P-384, P-521	Vendor Affirmed
RSA Key Wrapping	[SP 800-56B]	KTS RSAOAEP Modulus size: 2048, 3072 or 4096-bits	Vendor Affirmed

Table 3: Approved, Allowed and Vendor Affirmed Security Functions

Cryptographic Function	Algorithm	Options	Algorithm Certificate
Key Agreement IG D.8	[ANSI X9.63] [SP 800-56A] EC Diffie-Hellman	ECC curves P-256, P-384	Non-Approved, but allowed

Cryptographic Function	Algorithm	Options	Algorithm Certificate
	[ANSI X9.42] [SP 800-56A] Diffie-Hellman	Key sizes: 2048 bits	Non-Approved, but allowed
MD5 (used as part of the TLS key establishment scheme only)	Message Digest [RFC 1321]	Digest Size: 128-bit	Non-Approved, but Allowed
NDRNG (is provided by the underlying operational environment)	Random Number Generation	N/A	Non-Approved, but Allowed
RSA Key Wrapping	Non-[SP 800-56B], IG D.9 [SP800-131A]	PKCS#1 v1.5 and PSS Modulus size: 2048, 3072 or 4096-bits	Non-Approved, but allowed

Table 3a: Non-Approved but Allowed Security Functions

Note: PBKDFv2 is implemented to support all options specified in section 5.4 of [SP800-132]. The password consists of at least 6 alphanumeric characters from the ninety-six (96) printable and human-readable characters. The probability that a random attempt at guessing the password will succeed or a false acceptance will occur is equal to 1/96^6. The derived keys may only be used in storage applications. Additional guidance to appropriate usage is specified in section 7.3.

2.2.2 Non-Approved Security Functions:

Cryptographic Function	Usage / Description	Caveat
RSA Signature Generation / Signature Verification / Asymmetric Key Generation	ANSI X9.31 Signature Generation Key Pair Generation Key Size < 2048 Signature Verification Key Size < 1024	Non-Approved
	PKCS#1 v1.5 and PSS Signature Generation Key Size < 2048 Signature Verification Key Size < 1024	
RSA Key Wrapping	PKCS#1 v1.5 and PSS Key Size < 2048	Non-Approved
DSA used for Diffie-Hellman Key Generation only	L=2048, N=256	Non-Approved
Diffie-Hellman	Key agreement scheme using key sizes different than 2048-bits	Non-Approved

Ed25519	Key Agreement
	Sig(gen)
	Sig(ver)
		Non-Approved

ANSI X9.63 KDF	Hash based Key Derivation Function	Non-Approved

Cryptographic Function	Usage / Description	Caveat
RFC6637	Key Derivation Function	Non-Approved
DES	Encryption / Decryption Key Size 56-bits	Non-Approved
CAST5	Encryption / Decryption Key Sizes 40 to 128 bits in 8-bit increments	Non-Approved
RC4	Encryption / Decryption Key Sizes 8 to 4096-bits	Non-Approved
RC2	Encryption / Decryption Key Sizes 8 to 1024-bits	Non-Approved
MD2	Message Digest Digest size 128-bit	Non-Approved
MD4	Message Digest Digest size 128-bit	Non-Approved
RIPEMD	Message Digest Digest size 128, 160, 256, 320 -bits	Non-Approved
ECDSA	PKG: Curve P-192 PKV: Curve P-192 Signature Generation: Curve P-192 Signature Verification: Curve P-192	Non-Approved
	Key Pair Generation for compact point representation of points	Non-Approved
Integrated Encryption Scheme on elliptic curves	Encryption / Decryption	Non-Approved
Blowfish	Encryption / Decryption	Non-Approved
OMAC (One-Key CBC MAC)	MAC generation	Non-Approved
Triple-DES	Encryption / Decryption Two Key Implementation	Non-Approved
	asm_x86 (Optimized Assembler) ImplementationEncryption / Decryption Mode: CTR	Non-Compliant
AES-CMAC	AES-128/192/256 MAC generation/verification	Non-Approved
[SP800-108] KBKDF	HMAC-SHA1 or HMAC-SHA-224 or HMAC-SHA-256 or HMAC-SHA-384 or HMAC-SHA-512 and AES-CMAC based PRFs Modes: CTR and Feedback	Non-Compliant A8 (c_ltc) A22 (c_avx) A27 (c_ssse3) A33 (c_avx2)
[SP800-56C]	Key Derivation Function	Non-Compliant

Table 4: Non-Approved or Non-Compliant Security Functions

Note: A Non-Approved function in Table 4 is that the function implements a non-Approved algorithm, while a Non-Compliant function is that the function implements an Approved algorithm but the implementation is either not validated by the CAVP or/and the self-tests are not implemented (IG 9.4).

2.3 Cryptographic Module Boundary

The physical boundary of the module is the physical boundary of the macOS device that contains the module. Consequently, the embodiment of the module is a multi-chip standalone cryptographic module.

The logical module boundary is depicted in the logical block diagram given in Figure 1.

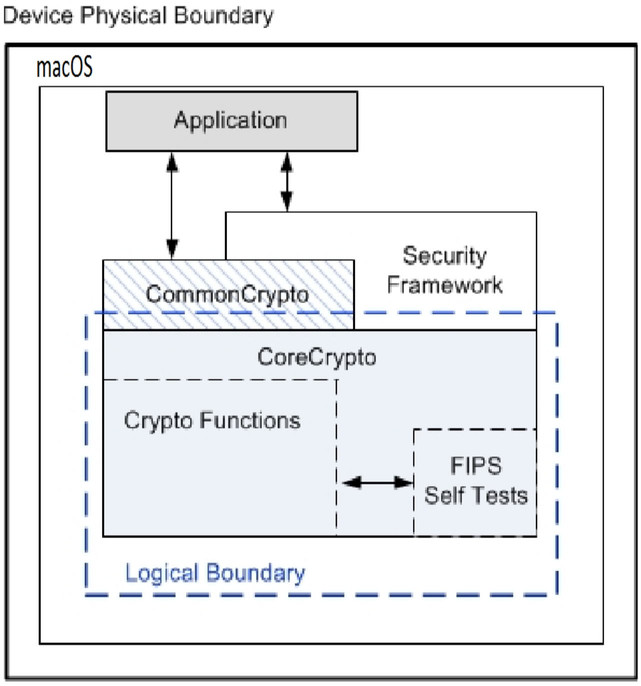

Figure 1: Logical Block Diagram

2.4 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] in compliance with IG A.5 scenario 1. The GCM IV generation follows RFC 5288 and shall only be used for the TLS protocol version 1.2. Users should consult [SP 800-38D], especially section 8, for all of the details and requirements of using AES-GCM mode. In case the module's power is lost and then restored, the key used for the AES GCM encryption/decryption shall be re-distributed.

- AES-XTS mode is only approved for hardware storage applications. The length of the XTS-AES data unit does not exceed 2^{20} blocks

- When using AES, the caller must obtain a reference to the cipher implementation via the functions of ccaes_[cbc|ecb|…]_[encrypt|decrypt]_mode.

- When using SHA, the caller must obtain a reference to the cipher implementation via the functions ccsha[1|224|256|384|512]_di.

- In order to meet the IG A.13 requirement, the same Triple-DES key shall not be used to encrypt more than 2^{16} 64-bit blocks of data.

3 Cryptographic Module Ports and Interfaces

The underlying logical interfaces of the module are the C language Application Programming Interfaces (APIs). In detail these interfaces are the following:

- Data input and data output are provided in the variables passed in the API and callable service invocations, generally through caller-supplied buffers. Hereafter, APIs and callable services will be referred to as "API."
- Control inputs which control the mode of the module are provided through dedicated parameters, as well as mach-o header holding the HMAC check file.
- Status output is provided in return codes and through messages. Documentation for each API lists possible return codes. A complete list of all return codes returned by the C language APIs within the module is provided in the header files and the API documentation. Messages are documented also in the API documentation.

The module is optimized for library use within the macOS user space and does not contain any terminating assertions or exceptions. It is implemented as an macOS dynamically loadable library. The dynamically loadable library is loaded into the macOS application and its cryptographic functions are made available. Any internal error detected by the module is reflected back to the caller with an appropriate return code. The calling macOS application must examine the return code and act accordingly. There is one notable exception: ECDSA and RSA do not return a key if the pair-wise consistency test fails.

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 services of the module listed in section 2.1 and 4.2.
Crypto Officer (CO)	Utilization of services of the module listed in section 2.1 and 4.2.

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 6 contains the cryptographic functions employed by the module in the 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/execute 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	Roles		CSPs & Crypto Keys	Access Type
	USER	CO		
Triple-DES Encryption / Decryption	X	X	Triple-DES key	R
AES Encryption / Decryption	X	X	AES key	R
AES Key Wrapping	X	X	AES	R
RSA Key Wrapping	X	X	RSA Private Key	R
Secure Hash Generation	X	X	none	N/A
HMAC generation	X	X	HMAC key	R
RSA signature generation and verification	X	X	RSA key pair	R
ECDSA signature generation and verification	X	X	ECDSA key pair	R

Service	Roles		CSPs & Crypto Keys	Access Type
	USER	CO		
Random number generation	X	X	Entropy input string, Nonce, V and Key	R W Z
PBKDF Password-based key derivation	X	X	derived key, password	R W Z
ECDSA key pair generation	X	X	ECDSA key pair	W
RSA key pair generation	X	X	RSA key pair	W
Diffie-Hellman Key agreement	X	X	Asymmetric keys (DH private key), shared secret and secret session key (AES/Triple-DES key)	R W
EC Diffie-Hellman Key agreement	X	X	Asymmetric keys (EC private key), shared secret and secret session key (AES/Triple-DES key)	R W
Release all resources of symmetric crypto function context	X	X	AES/Triple-DES key	Z
Release all resources of hash context	X	X	HMAC key	Z
Release of all resources of Diffie-Hellman context for Diffie-Hellman and EC Diffie-Hellman	X	X	Asymmetric keys (EC/DH) and secret session key (AES/Triple-DES)	Z
Release of all resources of asymmetric crypto function context	X	X	RSA/EC/DH keys	Z
Self-test	X	X	Software integrity key	R
Show Status	X	X	None	N/A

Table 6: Approved and Allowed Services in Approved Mode

Service	Roles	
	User	CO
Integrated Encryption Scheme on elliptic curves Encryption / Decryption	X	X
DES Encryption / Decryption	X	X
Triple-DES (Optimized Assembler-asm_x86- Implementation) Encryption / Decryption Mode: CTR	X	X
Triple-DES (Two-Key implementation) Encryption / Decryption	X	X
CAST5 Encryption / Decryption	X	X
Blowfish Encryption / Decryption	X	X
RC4 Encryption / Decryption	X	X
RC2 Encryption / Decryption	X	X
MD2 Hash	X	X
MD4 Hash	X	X
RIPEMD Hash	X	X
RSA Key Wrapping using Key Size < 2048	X	X
RSA PKCS#1 v1.5 and PSS Signature Generation and Verification Key Sizes: 1024-4096-bits in multiple of 32 bits not listed in Table 3	X	X

Service	Roles	
	User	CO
RSA ANSI X9.31 Key Pair Generation, Signature Generation and Signature Verification Key sizes (modulus): 1024-4096 bits in multiple of 32 bits not listed in Table 3 Public key exponent values: 65537 or larger	X	X
ECDSA Key Pair Generation for compact point representation of points	X	X
ECDSA PKG: curves P-192 PKV: curves P-192 SIG(gen): curves P-192SIG(ver): curves P-192	X	X
Diffie-Hellman Key Agreement Key Sizes different than 2048 bits	X	X
DSA used for Diffie-Hellman key pair generation	X	X
Ed25519 Key agreement, Signature Generation, Signature Verification	X	X
[SP800-56C] Key Derivation Function	X	X
Hash based Key Derivation Function using ANSI X9.63	X	X
[SP800-108] Key Derivation Function using HMAC-SHA1 or HMAC-SHA-224 or HMAC-SHA-256 or HMAC-SHA-384 or HMAC-SHA-512 and AES-CMAC Based Pseudo Random Functions Modes: Feedback, Counter	X	X
RFC6637 Key Derivation Function	X	X
AES-CMAC (AES-128/192/256) MAC Generation/Verification	X	X
OMAC MAC Generation	X	X

Table 7: Non-Approved Services in Non-Approved Mode

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 FIPS 140-2 physical security requirements do not apply to the Apple corecrypto User Space Module for Intel (ccv10) since it is a software module.

6 Operational Environment

6.1 Applicability

The Apple corecrypto User Space Module for Intel (ccv10) operates in a modifiable operational environment per FIPS 140-2 level 1 specifications. It is part of macOS 10.15, a commercially available general-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).

When the operating system loads the module into memory, it invokes the FIPS Self-Test functionality, which in turn runs the mandatory FIPS 140-2 tests.

7 Cryptographic Key Management

The Table 8 summarizes the cryptographic keys and CSPs used in the Apple corecrypto User Space Module for Intel (ccv10), with the ley lengths supported, the available methods for key generation, key entry and key output, and zeroization.

Name	Key / CSP Size	Generation	Entry / Output	Zeroization
AES Keys	128, 192, 256 bits	N/A. The key is entered via API parameter	Entry : calling application (see 7.4) Output: calling application (see 7.4)	automatic zeroization when structure is deallocated or when the system is powered down (see 7.6).
HMAC Keys	min 112- bits			
Triple-DES Keys	192 bits			
ECDSA key pair	P-224, P-256, P-384, P-521 curves	The private keys are generated using FIPS186-4 Key Generation method, and the random value used in the key generation is generated using SP800-90A DRBG		
RSA key pair	2048, 3072, 4096			
Entropy Input string		Obtained from the NDRNG.	Entry: OS Output: N/A	
DRBG nonce		Obtained from the NDRNG.		
DRBG V, Key		Derived from entropy input string as defined by SP800-90A	Entry: N/A Output: N/A	
PBKDF Keys	min: 112 bits	Internally generated via SP800-132 PBKDF key derivation algorithm	Entry: N/A Output: calling application (see 7.4)	
PBKDF Password		N/A. The password is provided by calling application	Entry : calling application (see 7.4) Output: N/A	
DH Key Pair	2048	N/A. The key pair is provided by calling application	Entry: calling application (see 7.4) Output: N/A	
DH Shared Secret	2048	Internally generated via SP800-56A DH shared secret computation.	Entry: N/A Output: calling application (see 7.4)	
ECC CDH Key Pair	P-256, P-384	The private keys can be generated using FIPS186-4 Key Generation method, and the random value used in the key generation is generated using SP800-90A DRBG	Entry: calling application (see 7.4) Output: calling application (see 7.4)	
ECC CDH Shared Secret	P-256, P-384	Internally generated via SP800-56A ECC CDH shared secret computation.	Entry: N/A Output: calling application (see 7.4)	

Table 8: Module Cryptographic key and CSPs

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 default Approved DRBG used for random number generation (i.e., random padding, nonce/salt generation, etc.) is a CTR_DRBG using AES-256 with derivation function and without prediction resistance. Additionally, the module provides the caller with additional random number generation functionality through an HMAC-DRBG which can be configure by the caller. The deterministic random bit generators are seeded by /dev/random. The /dev/random is the User Space interface that extracts random bits from the entropy pool. The NDRNG feeds entropy from the pool into the DRBG on demand. The NDRNG provides 256-bits of entropy.

7.2 Key / CSP Generation

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

- The module does not implement symmetric key generation.
- In accordance with FIPS 140-2 IG D.12, the cryptographic module performs Cryptographic Key Generation (CKG) for asymmetric keys as per [SP800-133] (vendor affirmed), compliant with [FIPS186-4], and using DRBG compliant with [SP800-90A]. A seed (i.e. the random value) used in asymmetric key generation is obtained from [SP800-90A] DRBG. The key generation service for RSA, ECDSA as well as the [SP 800-90A] DRBG have been ACVT tested with algorithm certificates found in Table 3.

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

7.3 Key / CSP Establishment

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

- AES key wrapping using KW, CCM and GCM modes,
- RSA key wrapping, RSA key wrapping encompasses:
  - RSA key wrapping using OAEP mode compliant to [SP 800-56B] (vendor affirmed)
  - RSA key wrapping using PKCS#1 v1.5 and PSS modes, non-approved but allowed per IG D.9
- Diffie-Hellman key establishment
- EC Diffie-Hellman key establishment
- [SP800-132] PBKDFv2 algorithm. The PBKDFv2 function is provided as a service and returns the key derived from the provided password to the caller. The keys derived from [SP800-132] map to section 4.1 of [SP800-133] as indirect generation from DRBG. 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 salt as well as the number of iterations. The implementation of the PBKDFv2 function requires the user to provide this information.

The encryption strengths for the key establishment methods are determined in accordance with FIPS 140-2 Implementation Guidance [IG] section 7.5 and NIST Special Publication 800-57 (Part1) [SP800-57P1].

- AES key wrapping is used for key establishment. Methodology provides between 128 and 256 bits of encryption strength.
- RSA key wrapping is used for key establishment. Methodology provides between 112 and 152 bits of encryption strength.
- Diffie-Hellman is used for key establishment; Methodology provides 112 bits of encryption strength.
- EC Diffie-Hellman is used for key establishment. Methodology provides 128 bits or 192 bits of encryption strength.

7.4 Key / CSP Entry and Output

All keys are entered from, or output to, the invoking application running on the same device. All keys entered into the module are electronically entered in plain text form. Keys are output from the module in plain text form if required by the calling application. The same holds for the CSPs.

7.5 Key / CSP Storage

The Apple corecrypto User Space Module for Intel (ccv10) 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 application. The same holds for CSPs.

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

7.6 Key / CSP Zeroization

Keys and CSPs are zeroized when the appropriate context object is destroyed or when the system is powered down.

8 Electromagnetic Interference/Electromagnetic Compatibility (EMI/EMC)

The EMI/EMC properties of the Apple corecrypto User Space Module for Intel (ccv10) 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 DRBG requires continuous verification. The FIPS Self-Tests application runs all required module self-tests. This application is invoked by the macOS startup process upon device initialization.

The execution of an independent application for invoking the self-tests in the corecrypto.dylib makes use of features of the macOS architecture: the module, implemented in corecrypto.dylib, is linked by libcommoncrypto.dylib which is linked by libSystem.dylib. The libSystem.dylib is a library that must be loaded into every application for operation. The operating system ensures that there is a strict CSP separation between the instances used by each application.

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 corecrypto User Space Module for Intel (ccv10) 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 fails to startup. To invoke the self-tests on demand, the user may reboot the system.

9.1.1 Cryptographic Algorithm Tests

Algorithm	Modes	Test
Triple-DES	CBC	KAT (Known Answer Test) Separate encryption / decryption operations are performed
AES implementations selected by the module for the corresponding environment. AES-128	CBC, ECB, GCM, XTS, CCM	KAT Separate encryption / decryption operations are performed
DRBG (CTR_DRBG and HMAC_DRBG; tested separately)	N/A	KAT
HMAC-SHA implementations selected by the module for the corresponding environment. HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-512	N/A	KAT
RSA	Signature Generation / Signature Verification; Encrypt / Decrypt	KAT, Separate encryption /decryption operations are performed
ECDSA	Signature Generation, Signature Verification	pair-wise consistency test
Diffie-Hellman “Z” computation	N/A	KAT
EC Diffie-Hellman “Z” computation	N/A	KAT

Table 9: Cryptographic Algorithm Tests

9.1.2 Software / Firmware Integrity Tests

A software integrity test is performed on the runtime image of the Apple corecrypto User Space Module for Intel (ccv10). The corecrypto’s HMAC-SHA256 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 corecrypto User Space Module for Intel (ccv10).

9.2.1 Continuous Random Number Generator Test

The Apple corecrypto User Space Module for Intel (ccv10) performs a continuous random number generator test on the noise source (i.e. NDRNG), whenever it is invoked to seed the [SP800-90A] DRBG.

9.2.2 Pair-wise Consistency Test

The Apple corecrypto User Space Module for Intel (ccv10) generates RSA and ECDSA asymmetric keys and performs the required pair-wise consistency tests with the newly generated key pair.

9.2.3 SP 800-90A Health Tests

The Apple corecrypto User Space Module for Intel (ccv10) performs the health tests as specified in section 11.3 of [SP800-90A].

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_CORECRYPTO_MACOS_US_SECPOL_5.0

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 corecrypto is built into macOS 10.15. For additional assurance, it is digitally signed.

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). Integration can only take place after the corecrypto project successfully passes internal integration and system tests on all platforms. B & I builds, integrates, system tests on all the platforms and checks 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 corecrypto has passed all self-tests and is operating in the Approved mode.

A Crypto Officer Role Guide is provided by Apple which offers IT System Administrators with the necessary technical information to ensure FIPS 140-2 Compliance of macOS 10.15 systems. This guide walks the reader through the system’s assertion of cryptographic module integrity and the steps necessary if module integrity requires remediation. A link to the Guide can be found on the Product security, validations, and guidance page found in [UGuide].

10.4.2 User Guidance

As explained 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 corecrypto has passed all self-tests and is operating in the Approved mode.

11 Mitigation of Other Attacks

The module protects against the utilization of known Triple-DES weak keys. The following keys are not permitted:

{0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01},

{0xFE,0xFE,0xFE,0xFE,0xFE,0xFE,0xFE,0xFE},

{0x1F,0x1F,0x1F,0x1F,0x0E,0x0E,0x0E,0x0E},

{0xE0,0xE0,0xE0,0xE0,0xF1,0xF1,0xF1,0xF1},

{0x01,0xFE,0x01,0xFE,0x01,0xFE,0x01,0xFE},

{0xFE,0x01,0xFE,0x01,0xFE,0x01,0xFE,0x01},

{0x1F,0xE0,0x1F,0xE0,0x0E,0xF1,0x0E,0xF1},

{0xE0,0x1F,0xE0,0x1F,0xF1,0x0E,0xF1,0x0E},

{0x01,0xE0,0x01,0xE0,0x01,0xF1,0x01,0xF1},

{0xE0,0x01,0xE0,0x01,0xF1,0x01,0xF1,0x01},

{0x1F,0xFE,0x1F,0xFE,0x0E,0xFE,0x0E,0xFE},

{0xFE,0x1F,0xFE,0x1F,0xFE,0x0E,0xFE,0x0E},

{0x01,0x1F,0x01,0x1F,0x01,0x0E,0x01,0x0E},

{0x1F,0x01,0x1F,0x01,0x0E,0x01,0x0E,0x01},

{0xE0,0xFE,0xE0,0xFE,0xF1,0xFE,0xF1,0xFE},

{0xFE,0xE0,0xFE,0xE0,0xFE,0xF1,0xFE,0xF1}