Crypto Officer Role Guide for FIPS 140-2 Compliance

for Intel

(macOS High Sierra 10.13)

Contents

Overview

In highly regulated industries, IT System Administrators and Crypto Officers are frequently required to ensure deployed systems are correctly using FIPS 140-2 Validated Cryptographic Modules. The two Apple Cryptographic Modules in macOS High Sierra v10.13 achieved **FIPS 140-2 Level 1 Conformance Validation** under the Cryptographic Module Validation Program (CMVP) – a joint American and Canadian security accreditation program for cryptographic modules.

These two modules are identified under the CMVP with the module names of: a) "**Apple CoreCrypto Module v8 for Intel"** and b) "**Apple CoreCrypto Kernel Module v8 for Intel.**" The **CoreCrypto Module** is available to developers for Applications and Services running in User Space. The **CoreCrypto Kernel Module** is used only by the macOS Kernel running on **Intel** processors.

Apple CoreCrypto Module v8 for Intel Validation Certificate #3155
csrc.nist.gov/projects/cryptographic-module-validation-program/Certificate/3155

Apple CoreCrypto Kernel Module v8 for Intel Validation Certificate #3156
csrc.nist.gov/projects/cryptographic-module-validation-program/Certificate/3156

Systems with a T2 also contain the following FIPS 140-2 validated hardware module

Apple Secure Enclave Processor Secure Key Store Cryptographic Module, v1.0

Validation Certificate #3223

csrc.nist.gov/projects/cryptographic-module-validation-program/Certificate/3223

Validation References and Resources

CMVP

All Apple Validated Crypto Modules can be found under CMVP's FIPS 140-2 Vendor List here - csrc.nist.gov/projects/cryptographic-module-validation-program/validated-modules/search

Apple

Apple Validated Crypto Modules, related Crypto Officer Role Guides, and links to the Security Policy document and CMVP issued certificates can be found in the Knowledge Base Article - **Product security, validations, and guidance for macOS** - located here - https://support.apple.com/en-us/HT201159

Compliant Applications and Services

Compliancy Requirements on Crypto Officers are not limited to the use of products containing a validated cryptographic module, but extend to their attestation that applications and services in use are FIPS 140-2 Compliant. Compliance is defined by CMVP to include both the use of a FIPS 140-2 validated module and the proper use of FIPS-Approved Algorithms. A cryptographic module may contain additional algorithms that are not FIPS-Approved and if used, would indicate a temporary Non-FIPS Compliant condition. A FIPS 140-2 Level 1 Conformance Validation does not require the cryptographic module restricts applications and services only to use FIPS-Approved algorithms.

Apple

A high-level, non-exhaustive list of Apple applications and services that are FIPS 140-2 Compliant would include the following:

Services

Bluetooth, FileVault 2, Kerberos, Keychain Services, Software Update Services, Time Machine, VPN, and 802.1X

Applications

App Store, Apple Remote Desktop, Calendar, Contacts, Disk Utility, Messages, iTunes, Keychain Access, Mail, Notes, Preview, and Safari

Developer and Crypto Officer Resources

There are many resources available to developers providing guidance on cryptographic services and API documentation. Developers should refer to these resources to ensure their products and services are FIPS 140-2 Compliant on macOS High Sierra v10.13.

Apple macOS CoreCrypto Module, v8.0 FIPS 140-2 Non-Proprietary Security Policy
csrc.nist.gov/CSRC/media/projects/cryptographic-module-validation-program/documents/security-policies/140sp3155.pdf

Apple macOS CoreCrypto Kernel Module, v8.0 FIPS 140-2 Non-Proprietary Security Policy
csrc.nist.gov/CSRC/media/projects/cryptographic-module-validation-program/documents/security-policies/140sp3156.pdf

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

csrc.nist.gov/CSRC/media/projects/cryptographic-module-validation-program/documents/security-policies/140sp3223.pdf

Crypto Officer Role Guide

Provides IT System Administrators with the necessary technical information to ensure FIPS 140-2 Compliance of the 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 Knowledge Base Article listed on the previous page.

Security Overview

https://developer.apple.com/library/ios/documentation/Security/Conceptual/Security_Overview/Introduction/Introduction.html

Cryptographic Services Guide

https://developer.apple.com/library/mac/documentation/Security/Conceptual/Security_Overview/CryptographicServices/CryptographicServices.html

Security Development Checklist

https://developer.apple.com/library/mac/documentation/Security/Conceptual/SecureCodingGuide/SecurityDevelopmentChecklists/SecurityDevelopmentChecklists.html

Certificate, Key, and Trust Services Programming Guide

https://developer.apple.com/library/mac/documentation/Security/Conceptual/CertKeyTrustProgGuide/

Generating New Cryptographic Keys

https://developer.apple.com/documentation/security/certificate_key_and_trust_services/keys/generating_new_cryptographic_keys

Storing Keys in the Keychains

https://developer.apple.com/documentation/security/certificate_key_and_trust_services/keys/storing_keys_in_the_keychain

Open Source

Many of open source projects use their own cryptographic libraries typically for the purposes of maintaining platform independence. These services are not covered by the Apple FIPS 140-2 Level 1 Conformance Validation of the CoreCrypto and CoreCrypto Kernel modules.

Compliant Platforms

Compliant platforms are all supported Apple systems running macOS High Sierra v10.13. During the validation process for FIPS 140-2 Conformance, the cryptographic modules are put through operational testing environments on supported platforms and noted on the issued certificate. The **CoreCrypto** and **CoreCrypto Kernel** modules were validated under the following operational testing environments:

Module: **Apple CoreCrypto Module v8 for Intel**

Platforms: **Core M** with macOS High Sierra v10.13 (User Space)
Core i5 with macOS High Sierra v10.13 (User Space)
Core i7 with macOS High Sierra v10.13 (User Space)
Xeon W with macOS High Sierra v10.13 (User Space)

Module: **Apple CoreCrypto Kernel Module v8 for intel**

Platforms: **Core M** with macOS High Sierra v10.13 (Kernel Space)
Core i5 with macOS High Sierra v10.13 (Kernel Space)
Core i7 with macOS High Sierra v10.13 (Kernel Space)
Xeon W with macOS High Sierra v10.13 (Kernel Space)

Compliant hardware

For FIPS 140-2 Compliance, the platforms noted above articulate Apple systems which were used for operational testing of the cryptographic modules. The CoreCrypto and CoreCrypto Kernel modules on Apple Intel-based systems with either the Intel Core M, Core i5, Core i7, or Xeon processor running macOS High Sierra v10.13 also take advantage of the additional processor embedded cryptographic engine. Compliant hardware are all Apple systems meeting the technical specifications to run macOS High Sierra v10.13. The Technical Specifications are noted in the Apple support article "**How to download macOS High Sierra**" - https://support.apple.com/HT201475.

"FIPS Mode" automatic

"FIPS Mode" is enabled all the time automatically without the need for installation, administration or configuration. All instances of macOS, since the initial release of version 10.8.0, have been using the two validated cryptographic modules and performing the required kernel module and algorithm tests.

These systems will perform all required tests such as the Power-On-Self-Tests (POST) for both the kernel and user space modules, integrity tests on the algorithms and module components, pairwise consistency tests, and finally the conditional self-tests on the random number generator according to the **FIPS 140-2 Level 1 Conformance Validation**.

The Power-On-Self-Test (POST) process flow

1. Apple Mac system is physically Powered on

2. Operating System begins the bootstrap process

3. Operating System ensures integrity of the **CoreCrypto Kernel Module**

    3.1. Validation of the `corecrypto.kext`
        3.1.1. The kernel determines operating environment (i.e i7)
        3.1.2. The kernel reads a validated HMAC_SHA256 from the `corecrypto.kext`
        3.1.3. The `corecrypto.kext` is launched and given the correct validated HMAC from 3.1.2
        3.1.4. The `corecrypto.kext` will generate an HMAC_SHA256 of the corecrypto.kext code and compare the result against the validated HMAC_SHA256 from 3.1.2
        3.1.5. If the calculated HMAC_SHA256 does not match the validated HMAC_SHA256, the system will panic and halt

    3.2. The Power-On-Self-Test (POST) validates the algorithms and modes
        3.2.1. The `corecrypto.kext` performs POST on algorithms and modes
        3.2.2. If any part of the POST fails, the system will panic and halt

4. Operating System ensures Integrity of **CoreCrypto Module**

    4.1. Validation of the `corecrypto.dylib`
        4.1.1. Upon user space environment setup by the kernel, **launchCtl** will launch the integrity test application `/usr/libexec/cc_fips_test`
        4.1.2. The system reads a validated HMAC_SHA256 from the `corecrypto.dylib`
        4.1.3. An HMAC_SHA256 of the user space corecrypto.dylib will be generated and compared to the HMAC_SHA256 value `from 4.1.2`
        4.1.4. If the calculated HMAC_SHA256 does not match the stored HMAC_SHA256, the system will panic and halt
    4.2. The Power-On-Self-Test (POST) validates the algorithms and modes
        4.2.1. The validated tool `cc_fips_test` performs POST on algorithms and modes
        4.2.2. If any part of the POST fails, the system will panic and halt

5. Halt upon failure of any tests

    5.1. If any phase or step of testing components fails, the system will log the failure and Halt/Shutdown the Operating System immediately.

5.2. The logging messages would be viewable using the log commands which provides access to system wide log messages created by os_log, os_trace and other logging systems.

How to verify integrity of the CoreCrypto Kernel module

As noted in the POST process flow, the kernel will ensure the integrity of the CoreCrypto Kernel module and its FIPS-Approved Algorithms. If any integrity issue is found during the POST, the device will automatically log the failure and halt/shutdown the system. There is no need or capability for a Crypto Officer to independently verify the integrity of the CoreCrypto Kernel module other than rebooting the system which automatically forces the complete POST.

How to verify integrity of the CoreCrypto Module

The tool is automatically executed during the POST to verify the integrity of the User Space CoreCrypto Module and the FIPS-Approved algorithms is called: `cc_fips_test`. There is no technical requirement for a Crypto Officer to ever need to run this tool, but it can be executed at any time by a user with administrative privileges.

Launch Terminal in the `/Applications/Utilities` folder and run the following command :

```
sudo /usr/libexec/cc_fips_test -v
```

The result should be:

```
Tracing: disabled
FIPSPOST_USER [143710727580520] fipspost_post:123: PASSED: (7 ms) - fipspost_post_integrity
FIPSPOST_USER [143710727636289] fipspost_post:129: PASSED: (0 ms) - fipspost_post_hmac
FIPSPOST_USER [143710727969561] fipspost_post:130: PASSED: (0 ms) - fipspost_post_aes_ecb
FIPSPOST_USER [143710727985848] fipspost_post:131: PASSED: (0 ms) - fipspost_post_aes_cbc
FIPSPOST_USER [143710731762706] fipspost_post:132: PASSED: (3 ms) - fipspost_post_rsa_sig
FIPSPOST_USER [143710732900924] fipspost_post:133: PASSED: (1 ms) - fipspost_post_ecdsa
FIPSPOST_USER [143710733217254] fipspost_post:134: PASSED: (0 ms) - fipspost_post_ecdh
FIPSPOST_USER [143710733230560] fipspost_post:135: PASSED: (0 ms) - fipspost_post_drbg_ctr
FIPSPOST_USER [143710733234766] fipspost_post:137: PASSED: (0 ms) - fipspost_post_aes_gcm
FIPSPOST_USER [143710733629585] fipspost_post:138: PASSED: (0 ms) - fipspost_post_aes_xts
FIPSPOST_USER [143710733685596] fipspost_post:139: PASSED: (0 ms) - fipspost_post_tdes_cbc
FIPSPOST_USER [143710733728620] fipspost_post:140: PASSED: (0 ms) - fipspost_post_drbg_hmac
FIPSPOST_USER [143710736071073] fipspost_post:142: PASSED: (2 ms) - fipspost_post_ffdh
FIPSPOST_USER [143710739264711] fipspost_post:143: PASSED: (3 ms) - fipspost_post_rsa_enc_dec
FIPSPOST_USER [143710739268434] fipspost_post:163: all tests PASSED (18 ms)
Returned from calling the FIPS_POST function in the corecrypto.dylib: result = true
```

How to verify integrity of the cc_fips_test tool

The `cc_fips_test` tool has been digitally signed by Apple. You can verify the tool and the tool's integrity by verifying its signature with the following command.

```
codesign -dvvv /usr/libexec/cc_fips_test
```

The result should be similar to the following sample data, but will not match exactly to the following due to the dependency on hardware and the OS version running on that hardware.

```
Executable=/usr/libexec/cc_fips_test
Identifier=com.apple.cc_fips_test
Format=Mach-O thin (x86_64)
CodeDirectory v=20100 size=231 flags=0x0(none) hashes=3+2 location=embedded
Platform identifier=4
Hash type=sha256 size=32
CandidateCDHash sha256=98fd7d729035e58cb1128d73110a0616b1c94f5c
Hash choices=sha256
CDHash=98fd7d729035e58cb1128d73110a0616b1c94f5c
Signature size=4485
Authority=Software Signing
Authority=Apple Code Signing Certification Authority
Authority=Apple Root CA
Info.plist=not bound
TeamIdentifier=not set
Sealed Resources=none
Internal requirements count=1 size=72
```

In the event the `cc_fips_test` tool had been modified in any form, the codesign verification would fail.

The result would be similar to:

```
/usr/libexec/cc_fips_test: object file format unrecognized, invalid, or
unsuitable
```

How to mitigate a crypto module integrity issue

If a crypto module integrity issue has been identified by the Crypto Officer, there is only one step that can be taken for remediation.

Reinstall macOS High Sierra v10.13

If there is an integrity issue found with the CoreCrypto module or the CoreCrypto Kernel module, it may not be appropriate to reinstall macOS High Sierra v10.13 until the Crypto Officer has effectively researched the integrity issues and any ramifications. After completing appropriate research, reinstalling macOS High Sierra v10.13 is the only corrective step that can be taken.

Information can also be found from the following Apple Support Knowledge Base Article.

Product security, validations, and guidance for macOS.

https://support.apple.com/en-us/HT201159

If choosing to perform an Apple Support-wide search for all articles pertaining to "FIPS iOS", use the following URL:

https://support.apple.com/kb/index?page=search&q=FIPS%20macOS&product=&doctype=&currentPage=1&includeArchived=false&src=support_site.kbase.search.searchresults

FIPS 140-2 Validated Algorithms

The CoreCrypto and CoreCrypto Kernel Modules are cryptographic libraries offering various cryptographic mechanisms to Apple frameworks. Algorithms from the two Apple cryptographic modules achieved **Cryptographic Algorithm Validation** under the Cryptographic Algorithm Validation Program (CAVP).

Modes of Operation

The CoreCrypto and CoreCrypto Kernel Modules have an Approved and Non-Approved modes of operation. The Approved mode of operation is configured in the system by default and cannot be changed. If the device boots up successfully then CoreCrypto framework and CoreCrypto KEXT have passed all self-tests and are operating in the Approved mode.

The Approved security functions are listed in **Table 3: Approved or Vendor Affirmed Security Functions** of the Non-Proprietary Security Policy documents posted along with the module validation certificate under CMVP. The Security Policy document links can be found above in the *Developer Resources* section. Column four (Val. No.) lists the validation numbers obtained from NIST for successful validation testing of the implementation of the cryptographic algorithms on the platforms as shown in Table 2 under CAVP.

Any calls to the non-Approved security functions listed in **Table 4: Non-Approved Security Functions** of the **Non-Proprietary Security Policy** documents will cause the module to assume the non-Approved mode of operation. Operators of the modules are strongly advised to avoid calling the functions in Table 4. If the module is operating in the non-Approved mode, operators are strongly cautioned to not use any CSP's previously utilized in the Approved mode of operation.

Note in the Security Policy documents under Key / CSP Establishment that the module provides AES key wrapping, RSA key wrapping, Diffie-Hellman- and EC Diffie-Hellman-based key establishment services in the Approved mode. The module provides key establishment services in the Approved mode through the PBKDFv2 algorithm. The PBKDFv2 function is provided as a service and returns the key derived from the provided password to the caller. The caller shall observe all requirements and should consider all recommendations specified in SP800-132 with respect to the strength of the generated key, including the quality of the password, the quality of the salt as well as the number of iterations. The implementation of the PBKDFv2 function requires the user to provide this information.

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

To see the exhaustive list of all algorithms supported by the cryptographic modules, Crypto Officers are highly encouraged to obtain and read the Security Policy document for complete technical explanations on the CoreCrypto and CoreCrypto Kernel modules. Links are provided in the Developer and Crypto Officer Resources section above.

Suite B Cryptographic Algorithms

The CoreCrypto Module (User Space) does provide for the use of Suite B Cryptographic Algorithms as are called out by NSA. Those algorithms include AES (FIPS 197), ECDH (SP 800-56A), ECDSA (FIPS 186-4) and SHA-256/-384 (FIPS 180-4).

Public Review of Cryptographic Libraries

The same libraries that secure all Apple Operating Systems are available to third-party developers to help them build advanced security features.

Cryptographic Libraries **https://developer.apple.com/cryptography/**

— Security Framework

Security Framework provides interfaces for managing certificates, public and private keys, and trust policies. It supports the generation of cryptographically secure pseudorandom numbers. It also supports the storage of certificates and cryptographic keys in the keychain, which is a secure repository for sensitive user data.

— Common Crypto

The Common Crypto library provides additional support for operations like symmetric encryption, hash-based message authentication codes, and digests.

— corecrypto

Although the CoreCrypto Modules do not directly provide programming interfaces for developers and should not be used by iOS, tvOS, watchOS or macOS apps, the source code has been posted and is available to allow for verification of its security characteristics and correct functioning.