Crypto Officer Role Guide for FIPS 140-2 Compliance

OS X Mavericks v10.9

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 OS X Mavericks v10.9 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 OS X CoreCrypto Module v4.0"** and b) "**Apple OS X CoreCrypto Kernel Module v4.0**." The **CoreCrypto Module** is available to developers for Applications and Services running in User Space. The **CoreCrypto Kernel Module** is used only by the OS X Kernel.

Within this and other Apple documents, those modules are also referred to with the name of **"Apple FIPS Cryptographic Module v4.0."**

Apple OS X CoreCrypto Module v4.0 Validation Certificate **#2015**

http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/1401val2013.htm#2015

Apple OS X CoreCrypto Kernel Module v4.0 Validation Certificate **#2016**

http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/1401val2013.htm#2016

All Apple Validated Crypto Modules can be found under CMVP's FIPS 140-2 Vendor List here - http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/1401vend.htm

This Crypto Officer Role Guide provides IT System Administrators with the necessary technical information to ensure FIPS 140-2 Compliance of OS X Mavericks v10.9 systems. This guide walks the reader through the steps to perform the initial enablement, the system's assertion of cryptographic module integrity and finally the steps necessary if module integrity requires remediation.

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 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 Non-FIPS Compliant condition. A FIPS 140-2 Level 1 Conformance Validation does not require the cryptographic module ensures applications and services only use FIPS-Approved algorithms.

Apple

A high-level, non-exhaustive list of Apple applications and services that are FIPS 140-2 Compliant on OS X Mavericks v10.9 includes the following:

Services

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 several resources available to developers providing guidance on cryptographic services and API documentation for OS X Mavericks v10.9. Developers should refer to these resources to ensure their products and services are FIPS 140-2 Compliant on OS X Mavericks.

Apple OS X CoreCrypto Module, v4.0 FIPS 140-2 Non-Proprietary Security Policy

http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140sp/140sp2015.pdf

Apple OS X CoreCrypto Kernel Module, v4.0 FIPS 140-2 Non-Proprietary Security Policy

http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140sp/140sp2016.pdf

OS X: Security certifications and validations

http://support.apple.com/kb/HT5588

Security Overview

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

Cryptographic Services Guide

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

Security Transforms Programming Guide

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

Certificate, Key, and Trust Services Programming Guide

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

Self-assertion when using deprecated CDSA

In rare cases, developers may still be transitioning from CDSA, a security framework deprecated in OS X Lion v10.7 which was submitted and achieved FIPS 140-2 Level 1 Conformance Validation under Mac OS X Snow Leopard v10.6 [Cert: #1514]. The unchanged CDSA framework was then submitted for re-validation and achieved FIPS 140-2 Level 1 Conformance Validation under OS X Lion v10.7 [Cert: #1701] for the sole benefit of third-party developers. The same, deprecated and unchanged CDSA framework exists in OS X Mountain Lion v10.8 and in OS X Mavericks v10.9, but was not submitted again to the CMVP. CDSA use under OS X Mavericks v10.9 comes with Apple's self-assertion of its FIPS 140-2 Compliance based on the prior two successful validations and the CDSA-based module's unmodified existence in OS X Mountain Lion v10.8 and OS X Mavericks v10.9. Developers are always highly encouraged to deploy solutions based on the current cryptographic module(s) provided by Apple.

Open Source

There are additional services included with OS X that are drawn from externally maintained open source projects. Many of these 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. A high-level, non-exhaustive list includes services such as: Apache, OpenLDAP, OpenSSH, and OpenSSL.

Compliant Platforms

Compliant platforms are all supported Apple systems running OS X Mavericks v10.9. 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 OS X CoreCrypto Module v4.0**

Platforms: Core i5 with OS X Mavericks v10.9 (User Space)
Core i7 with OS X Mavericks v10.9 (User Space)

Module: **Apple OS X CoreCrypto Kernel Module v4.0**

Platforms: Core i5 with OS X Mavericks v10.9 (Kernel Space)
Core i7 with OS X Mavericks v10.9 (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 systems with either the Core i5 or Core i7 processors running OS X Mavericks v10.9 also take advantage of the Intel Advanced Encryption Standard Instructions (AES-NI). Compliant hardware are all Apple systems meeting the technical specifications to run OS X Mavericks v10.9. The Technical Specifications are noted in the Apple support article "**OS X Mavericks: System Requirements**" - http://support.apple.com/kb/HT5842

"FIPS Mode" automatic

"FIPS Mode" is now enforced all the time automatically without the need for installation, administration or configuration. All instances of OS X Mavericks, since the initial release of version 10.9.0, have been using the two validated cryptographic modules and performing the required kernel module and algorithm tests.

Upon installer completion, the OS X Mavericks v10.9 system will be fully in "FIPS Mode" and 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, and pairwise consistency tests. Finally, the conditional self-tests on the random number generator will be performed according to the **FIPS 140-2 Level 1 Conformance Validation**.

FIPS related files installed by the OS X Mavericks installer

During the installation of OS X Mavericks v10.9, FIPS 140-2 related files are installed. Their location and purpose is explained here with sample data.

```
drwxr-xr-x    3 root wheel       /private/var/db/FIPS
```

Directory for FIPS Integrity Data

```
-rw-r--r--    1 root wheel       /private/var/db/FIPS/fips_data
```

HMAC_SHA256 values for 64/32-bit kernel modules

Sample Data:

```
x86_64:0573480d1e07a00b4cabbafcb53be00d470e28bd2168cec1235d256f7e1bb7b7
i386:ec5ab8aa18b219c5d06d2fb6339d3c40837a0a16327f752ebbe1a76d84773a22
```

```
-rwxr-xr-x    1 root wheel       /usr/libexec/cc_fips_test
```

CoreCrypto (User Space) FIPS verification tool

```
-rw-r--r--    1 root wheel       /usr/share/man/man1/cc_fips_test.1
```

Man page for the `cc_fips_test` command.

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

1. Apple Mac system is physically Powered on

2. Operating System (OS X Mavericks v10.9) begins bootstrap process

3. Operating System ensures integrity of **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 cipher 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 test application `/usr/libexec/cc_fips_test`
        4.1.2. An HMAC_SHA256 of the user space corecrypto.dylib will be generated and compared to the HMAC_SHA256 value stored at `/var/db/FIPS/fips_data`
        4.1.3. If the calculated HMAC_SHA256 does not match the stored HMAC_SHA256, the system will panic and halt
    4.2. The cipher Power-On-Self-Test (POST) validates the algorithms and modes
        4.2.1. The `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 are sent to the `/var/log/system.log` file.

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 `cc_fips_test` tool is automatically called during the POST to verify the integrity of the User Space CoreCrypto Module and the FIPS-Approved algorithms. 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:

```
About to call the FIPS_POST function in the corecrypto.dylib
FIPS USER Space POST: Integrity test success!
FIPS USER Space POST: AES GCM Test success!
FIPS USER Space POST: AES CBC Test success!
FIPS USER Space POST: AES AESNI ECB Test success!
FIPS USER Space POST: AES AESNI XTS Test success!
FIPS USER Space POST: TDES CBC Test success!
FIPS USER Space POST: SHA Test success!
FIPS USER Space POST: HMAC Test success!
FIPS USER Space POST: RSA Test success!
FIPS USER Space POST: ECDSA Test success!
FIPS USER Space POST: DRBG Test success!
FIPS USER Space POST Success!
Returned  from calling the FIPS_POST function in the corecrypto.dylib: result = true
```

Executing the above command without first obtaining administrative privileges will result in the system reporting the Integrity test failed.

The result would be:

```
FIPS USER Space POST: Integrity test failed!
```

The man page for this command can be read by running the following command:

```
man cc_fips_test
```

The result would be:

```
NAME

     cc_fips_test

SYNOPSIS

     Called when the system is booted to test the FIPS user space boundary.

DESCRIPTION

     The FIPS user space Power On Self Test (POST) tester.  May be called after the
      system has booted to re-run the POST tests.
```

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:

```
Executable=/usr/libexec/cc_fips_test
Identifier=com.apple.cc_fips_test
Format=Mach-O universal (i386 x86_64)
CodeDirectory v=20100 size=171 flags=0x0(none) hashes=3+2 location=embedded
Hash type=sha1 size=20
CDHash=8c4030c92275806756dca48de0156da85d781d32
Signature size=4064
Authority=Software Signing
Authority=Apple Code Signing Certification Authority
Authority=Apple Root CA
Info.plist=not bound
Sealed Resources=none
Internal requirements count=2 size=120
```

In the event the `cc_fips_test` tool has been modified in any form, the codesign verification will 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 in the /var/log/system.log by the Crypto Officer, then there is only one step that can be taken for remediation.

Reinstall OS X Mavericks

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

If the Crypto Officer needs assistance, Apple Knowledge Base Articles should prove to be quite helpful.

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 in OS X Mavericks v10.9 achieved **Cryptographic Algorithm Validation** under the Cryptographic Algorithm Validation Program (CAVP)

Modes of Operation

The CoreCrypto and CoreCrypto Kernel Modules have 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 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 DH- and ECDH-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 http://csrc.nist.gov/groups/STM/cavp/index.html for the current standards, test requirements, and special abbreviations used.

The Approved Security Functions Table has been recreated below for quick and easy access, but is not 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.

Suite B Cryptographic Algorithms

The CoreCrypto Module (User Space) does provide for the use of Suite B Cryptographic Algorithms as are called out on the NSA Suite B Cryptography web page. Those algorithms include AES, ECDH, ECDSA and SHA-256/-384. For further information from NSA about Suite B Algorithms, refer to http://www.nsa.gov/ia/programs/suiteb_cryptography/.

Cryptographic Module:	CoreCrypto Module v4.0 (User Space)			
Alg.	**Platform Certificate**		**Standards**	**Description**
	i5	**i7**		
	2540 (32-bit) 2524 (64-bit)	2541 (32-bit) 2531 (64-bit)		**User space and generic, non-optimized software.** **ECB** (128 , 192 , 256) **CBC** (128 , 192 , 256) **CFB8** (128 , 192 , 256) **CFB128** (128 , 192 , 256) **OFB** (128 , 192 , 256) **CTR** (128 , 192 , 256) - ***int only*** **GCM** **KS:** **AES_128** Tag Length(s): 128 120 112 104 96 64 32 **AES_192** Tag Length(s): 128 120 112 104 96 64 32 **AES_256** Tag Length(s): 128 120 112 104 96 64 32 **IV Generated:** Internally (using Section 8.2.1) **PT Lengths Tested:** (1024, 120, 960) **AAD Lengths tested:** (1024, 120, 960) **IV Lengths Tested:** (8 , 1024) **96BitIV_Supported** **GMAC_Not_Supported** **DRBG:** **i5:** **i7:** **32-bit** Val# 374 Val# 375 **64-bit** Val# 366 Val# 369 **XTS** **KS: XTS_128** **KS: XTS_256**
	2536 (32-bit) 2523 (64-bit)	2537 (32-bit) 2530 (64-bit)		**User space and the AES-NI Intel instruction set using the generic block chaining modes of CBC and XTS.** **CBC** (128 , 192 , 256) **XTS** **KS: XTS_128** **KS: XTS_256**

<table>
<tr>
<td rowspan="3">AES</td>
<td>2538
(32-bit)
2521
(64-bit)</td>
<td>2539
(32-bit)
2529
(64-bit)</td>
<td rowspan="3">FIPS-197
SP 800-38A
SP 800-38D
SP 800-38E</td>
<td>User space and the AES-NI Intel instruction set with an accelerated implementation for CBC and XTS.

ECB (128 , 192 , 256)
CBC (128 , 192 , 256)
CFB8 (128 , 192 , 256)
CFB128 (128 , 192 , 256)
OFB (128 , 192 , 256)
CTR (128 , 192 , 256) - int only
GCM
KS:
AES_128 Tag Length(s): 128 120 112 104 96 64 32
AES_192 Tag Length(s): 128 120 112 104 96 64 32
AES_256 Tag Length(s): 128 120 112 104 96 64 32
IV Generated: Internally (using Section 8.2.2)
PT Lengths Tested: (1024, 120, 960)
AAD Lengths tested: (1024, 120, 960)
IV Lengths Tested: (8 , 1024)
96BitIV_Supported
GMAC_Not_Supported

DRBG: i5: i7:
32-bit Val# 372 Val# 373
64-bit Val# 365 Val# 368

XTS
KS: XTS_128
KS: XTS_256</td>
</tr>
<tr>
<td>2534
(32-bit)
2520
(64-bit)</td>
<td>2535
(32-bit)
2528
(64-bit)</td>
<td>User space and the Gladman AES CBC implementations.

CBC (128 , 192 , 256)</td>
</tr>
<tr>
<td>2532
(32-bit)
2519
(64-bit)</td>
<td>2533
(32-bit)
2527
(64-bit)</td>
<td>User space and assembler optimized AES.

ECB (128 , 192 , 256)
CBC (128 , 192 , 256)
CFB8 (128 , 192 , 256)
CFB128 (128 , 192 , 256)
OFB (128 , 192 , 256)
CTR (128 , 192 , 256) - int only
GCM
KS:
AES_128 Tag Length(s): 128 120 112 104 96 64 32
AES_192 Tag Length(s): 128 120 112 104 96 64 32
AES_256 Tag Length(s): 128 120 112 104 96 64 32
IV Generated: Internally (using Section 8.2.2)
PT Lengths Tested: (1024, 120, 960)
AAD Lengths tested: (1024, 120, 960)
IV Lengths Tested: (8 , 1024)
96BitIV_Supported
GMAC_Not_Supported
DRBG: i5: i7:
32-bit Val# 370 Val# 371
64-bit Val# 364 Val# 367
XTS
KS: XTS_128
KS: XTS_256</td>
</tr>
</table>

DRBG	374 (32-bit) 366 (64-bit)	375 (32-bit) 369 (64-bit)	SP 800-90A	**User space and generic, non-optimized software.** **CTR_DRBG**: Prediction Resistance Tested: Enabled BlockCipher_Use_df: (**AES-128**) **AES:** **i5:** **i7:** **32-bit** Val# 2540 Val# 2541 **64-bit** Val# 2524 Val# 2531
	372 (32-bit) 365 (64-bit)	373 (32-bit) 368 (64-bit)		**User space and the AES-NI Intel instruction set with an accelerated implementation for CBC and XTS.** **CTR_DRBG**: Prediction Resistance Tested: Enabled BlockCipher_Use_df: (**AES-128**) **AES:** **i5:** **i7:** **32-bit** Val# 2538 Val# 2539 **64-bit** Val# 2521 Val# 2529
	370 (32-bit) 364 (64-bit)	371 (32-bit) 367 (64-bit)		**User space and assembler optimized AES**. **CTR_DRBG**: Prediction Resistance Tested: Enabled BlockCipher_Use_df: (**AES-128**) **AES:** **i5:** **i7:** **32-bit** Val# 2532 Val# 2533 **64-bit** Val# 2519 Vall# 2527
ECDSA	434 (32-bit) 432 (64-bit)	435 (32-bit) 433 (64-bit)	FIPS 186-3 ANSI X9.62	**User space and generic, non-optimized software.** **FIPS186-2:** **PKG:** **CURVES** (P-256 P-384) **PKV:** **CURVES** (P-256 P-384) **SIG(gen):** **CURVES** (P-256 P-384) **SIG(ver):** **CURVES** (P-256 P-384) **SHS:** **i5:** **i7:** **32-bit** Val# 2136 Val# 2137 **64-bit** Val# 2130 Val# 2133 **DRBG:** **i5:** **i7:** **32-bit** Val# 374 Val# 375 **64-bit** Val# 366 Val# 369
HMAC	1558 (32-bit) 1552 (64-bit)	1559 (32-bit) 1555 (64-bit)	FIPS 198	**User space and generic, non-optimized software.** **(Key Sizes:Block Sizes tested: KS<BS KS=BS KS>BS)** **HMAC-SHA1, HMAC-SHA224, HMAC-SHA256, HMAC-SHA384, HMAC-SHA512** **SHS:** **i5:** **i7:** **32-bit** Val# 2136 Val# 2137 **64-bit** Val# 2130 Val# 2133
	1560 (32-bit) 1553 (64-bit)	1561 (32-bit) 1556 (64-bit)		**User space and optimized SHA-1, SHA-224, and SHA-256.** **(Key Sizes:Block Sizes tested: KS<BS KS=BS KS>BS)** **HMAC-SHA1, HMAC-SHA224, HMAC-SHA256** **SHS:** **i5:** **i7:** **32-bit** Val# 2138 Val# 2139 **64-bit** Val# 2131 Val# 2134

	1562 (32-bit) 1554 (64-bit)	1563 (32-bit) 1557 (64-bit)		**User space and optimized SHA-1, SHA-224, and SHA-256 using the SSE3 processor instruction set.** **(Key Sizes:Block Sizes tested: KS<BS KS=BS KS>BS)** **HMAC-SHA1, HMAC-SHA224, HMAC-SHA256** **SHS:** **i5:** **i7:** **32-bit** Val# 2140 Val# 2141 **64-bit** Val# 2132 Val# 2135
PBKDF2	N/A	N/A	SP 800-132	Password based key derivation according to PKCS#5 using HMAC with SHA-1 or SHA-2 as pseudorandom function.
RSA	1295 (32-bit) 1293 (64-bit)	1296 (32-bit) 1294 (64-bit)	FIPS 186-3 ANSI X9.31 PKCS#1v1.5	**User space and generic, non-optimized software.** **FIPS186-2:** **ALG[ANSIX9.31]:** Key(gen)(MOD: 1024 , 1536 , 2048 , 3072 , 4096 PubKey Values: 3 , 17 , 65537 **DRBG:** **i5:** **i7:** **32-bit** Val# 374 Val# 375 **64-bit** Val# 366 Val# 369 **ALG[RSASSA-PKCS1_V1_5]:** SIG(gen), SIG(ver): 1024 , 1536 , 2048 , 3072 , 4096 **SHS**: SHA-1, SHA-224, SHA-256, SHA-384,SHA-512 **i5:** **i7:** **32-bit** Val# 2136 Val# 2137 **64-bit** Val# 2130 Val# 2133
SHS	2136 (32-bit) 2130 (64-bit)	2137 (32-bit) 2133 (64-bit)	FIPS 180-3	**User space and generic, non-optimized software.** **SHA-1** (BYTE-only) **SHA-224** (BYTE-only) **SHA-256** (BYTE-only) **SHA-384** (BYTE-only) **SHA-512** (BYTE-only)
	2138 (32-bit) 2131 (64-bit)	2139 (32-bit) 2134 (64-bit)		**User space and optimized SHA-1, SHA-224, and SHA-256.** **SHA-1** (BYTE-only) **SHA-224** (BYTE-only) **SHA-256** (BYTE-only)
	2140 (32-bit) 2132 (64-bit)	2141 (32-bit) 2135 (64-bit)		**User space and optimized SHA-1, SHA-224, SHA-256 using the SSE3 processor instruction set.** **SHA-1** (BYTE-only) **SHA-224** (BYTE-only) **SHA-256** (BYTE-only)
TDES	1536 (32-bit) 1534 (64-bit)	1537 (32-bit) 1535 (64-bit)	ANSIX9.52-1998 FIPS 46-3 SP 800-67 SP 800-38A Appendix E	**User space and generic, non-optimized software.** **TECB** (KO 1,2) **TCBC** (KO 1,2) **TCFB8** (KO 1,2) **TCFB64** (KO 1,2) **TOFB** (KO 1,2) **CTR** (int only)

Cryptographic Module:	CoreCrypto Kernel Module v4.0 (Kernel Space)			
Alg.	**Platform Certificate**		**Standards**	**Description**
	i5	**i7**		
AES	2514 (64-bit)	2518 (64-bit)	FIPS 197 SP 800-38A SP 800-38E	**Kernel space and generic, non-optimized software.** **ECB** (128 , 192 , 256) **CBC** (128 , 192 , 256) **XTS** **KS: XTS_128** **KS: XTS_256**
	2513 (64-bit)	2517 (64-bit)		**Kernel space and the AES-NI instruction set using the generic block chaining modes of CBC and XTS.** **ECB** (128 , 192 , 256) **CBC** (128 , 192 , 256) **XTS** **KS: XTS_128** **KS: XTS_256**
	2512 (64-bit)	2516 (64-bit)		**Kernel space and the AES-NI intel instruction set with an accelerated implementation for CBC and XTS.** **ECB** (128 , 192 , 256) **CBC** (128 , 192 , 256) **XTS** **KS: XTS_128** **KS: XTS_256**
	2511 (64-bit)	2515 (64-bit)		**Kernel space and assembler optimized AES.** **ECB** (128 , 192 , 256) **CBC** (128 , 192 , 256) **XTS** **KS: XTS_128** **KS: XTS_256**
DRBG	360 (64-bit)	363 (64-bit)	SP 800-90A	**Kernel space and generic, non-optimized software.** **CTR_DRBG**: Prediction Resistance Tested: Enabled BlockCipher_Use_df: (**AES-128**) **AES** **i5:** Val# 2514 **i7:** Val# 2518
	359 (64-bit)	362 (64-bit)		**Kernel space and the AES-NI Intel instruction set with an accelerated implementation for CBC and XTS.** **CTR_DRBG**: Prediction Resistance Tested: Enabled BlockCipher_Use_df: (**AES-128**) **AES** **i5:** Val# 2512 **i7:** Val# 2516

	358 (64-bit)	361 (64-bit)		**Kernel space and assembler optimized AES.** **CTR_DRBG**: Prediction Resistance Tested: Enabled BlockCipher_Use_df: (**AES-128**) **AES** **i5:** Val# 2511 **i7:** Val# 2515
ECDSA	430 (64-bit)	431 (64-bit)	FIPS 186-3 ANSI X9.62	**Kernel space and generic, non-optimized software.** **FIPS186-2:** **PKG:** **CURVES** (P-256 P-384) **PKV:** **CURVES** (P-256 P-384) **SIG(gen):** **CURVES** (P-256 P-384) **SIG(ver):** **CURVES** (P-256 P-384) **i5** **i7** **SHS:** Val# 2124 Val# 2127 **DRBG:** Val# 360 Val# 363
HMAC	1547 (64-bit)	1550 (64-bit)	FIPS 198	**Kernel space and optimized SHA-1, SHA-224, and SHA-256.** **(Key Sizes:Block Sizes tested: KS<BS KS=BS KS>BS)** **HMAC-SHA1, HMAC-SHA224, HMAC-SHA256** **SHS:** **i5:** **i7:** **64-bit** Val# 2125 Val# 2128
	1548 (64-bit)	1551 (64-bit)		**Kernel space and optimized SHA-1, SHA-224, and SHA-256 using the SSE3 processor instructions set.** **(Key Sizes:Block Sizes tested: KS<BS KS=BS KS>BS)** **HMAC-SHA1, HMAC-SHA224, HMAC-SHA256** **SHS:** **i5:** **i7:** **64-bit** Val# 2129 Val# 2126
	1546 (64-bit)	1549 (64-bit)		**Kernel space and generic, non-optimized software.** **(Key Sizes:Block Sizes tested: KS<BS KS=BS KS>BS)** **HMAC-SHA1, HMAC-SHA224, HMAC-SHA256, HMAC-SHA384, HMAC-SHA512** **SHS:** **i5:** **i7:** **64-bit** Val# 2124 Val# 2127
PBKDF2	N/A	N/A	SP 800-132	
RSA	1291 (64-bit)	1292 (64-bit)	PKCS#1v1.5	**Kernel space and generic, non-optimized software.** **FIPS186-2:** **ALG[RSASSA-PKCS1_V1_5]:** SIG(gen), SIG(ver): 1024 , 1536 , 2048 , 3072 , 4096 **SHS**: SHA-1, SHA-224, SHA-256, SHA-384,SHA-512 **i5:** **i7:** **64-bit** Val# 2124 Val# 2127

SHS	2125 (64-bit)	2128 (64-bit)	FIPS 180-3	**Kernel space and optimized SHA-1, SHA-224, and SHA-256.** **SHA-1** (BYTE-only) **SHA-224** (BYTE-only) **SHA-256** (BYTE-only)
	2126 (64-bit)	2129 (64-bit)		**Kernel space and optimized SHA-1, SHA-224, and SHA-256 using the SSE3 processor instruction set.** **SHA-1** (BYTE-only) **SHA-224** (BYTE-only) **SHA-256** (BYTE-only)
	2124 (64-bit)	2127 (64-bit)		**Kernel space and generic, non-optimized software.** **SHA-1** (BYTE-only) **SHA-224** (BYTE-only) **SHA-256** (BYTE-only) **SHA-384** (BYTE-only) **SHA-512** (BYTE-only)
TDES	1532 (64-bit)	1533 (64-bit)	ANSIX9.52-1998 FIPS 46-3 SP 800-67 SP 800-38A Appendix E	**Kernel space and generic, non-optimized software.** **TECB** (KO 1,2) **TCBC** (KO 1,2)