VMware, Inc.
3401 Hillview Ave
Palo Alto, CA 94304, USA
Tel: 877-486-9273
Email: info@vmware.com
http://www.vmware.com

VMware OpenSSL FIPS Object Module

Software Version: 2.0.9

FIPS 140-2 Non-Proprietary Security Policy

FIPS Security Level: 1
Document Version: 1.0

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 Introduction

1.1 Purpose

This is a non-proprietary Cryptographic Module Security Policy for the VMware OpenSSL FIPS Object Module from VMware, Inc. This Security Policy describes how the VMware OpenSSL FIPS Object Module meets the security requirements of Federal Information Processing Standards (FIPS) Publication 140-2, which details the U.S. and Canadian Government requirements for cryptographic modules. More information about the FIPS 140-2 standard and validation program is available on the National Institute of Standards and Technology (NIST) and the Communications Security Establishment (CSE) Cryptographic Module Validation Program (CMVP) website at https://csrc.nist.gov/projects/cryptographic-module-validation-program.

This document also describes how to run the module in a secure FIPS-Approved mode of operation. The VMware OpenSSL FIPS Object Module is also referred to in this document as "the module".

1.2 Reference

This document deals only with operations and capabilities of the composite module in the technical terms of a FIPS 140-2 cryptographic module security policy. More information is available on the module from the following sources:

- The VMware website (http://www.vmware.com) contains information on the full line of products from VMware.
- The CMVP website (https://csrc.nist.gov/Projects/Cryptographic-Module-Validation-Program/Validated-Modules/Search) contains options to get contact information for individuals to answer technical or sales-related questions for the module.

2 VMware OpenSSL FIPS Object Module

2.1 Introduction

VMware, Inc., a global leader in virtualization, cloud infrastructure, and business mobility, delivers customer-proven solutions that accelerate Information Technology (IT) by reducing complexity and enabling more flexible, agile service delivery. With VMware solutions, organizations are creating exceptional experiences by mobilizing everything, responding faster to opportunities with modern data and apps hosted across hybrid clouds, and safeguarding customer trust with a defense-in-depth approach to cybersecurity. VMware enables enterprises to adopt an IT model that addresses their unique business challenges. VMware's approach accelerates the transition to solutional-computing while preserving existing investments and improving security and control.

2.1.1 VMware OpenSSL FIPS Object Module

The VMware OpenSSL FIPS Object Module is a software cryptographic module that is built from the OpenSSL FIPS Object Module source code according to the instructions prescribed in Appendix A. The module is a software library that provides cryptographic functions to various VMware applications via a well-defined C-language application program interface (API). The module only performs communications with the calling application (the process that invokes the module services).

The VMware OpenSSL FIPS Object Module is validated at the FIPS 140-2 Section levels shown in Table 1:

Table 1 – Security Level Per FIPS 140-2 Section

Section	Section Title	Level
1	Cryptographic Module Specification	1
2	Cryptographic Module Ports and Interfaces	1
3	Roles, Services, and Authentication	2
4	Finite State Model	1
5	Physical Security	N/A[1]
6	Operational Environment	1
7	Cryptographic Key Management	1
8	EMI/EMC[2]	1
9	Self-tests	1
10	Design Assurance	3
11	Mitigation of Other Attacks	N/A

2.2 Module Specification

The VMware OpenSSL FIPS Object Module is a software cryptographic module with a multiple-chip standalone embodiment. The overall security level of the module is 1. The software version of the module is 2.0.9, and it is built from the 2.0.9 version of the OpenSSL FIPS Object Module source code.

[1] N/A – Not Applicable

[2] EMI/EMC – Electromagnetic Interference/Electromagnetic Compatibility

The module was tested and found to be FIPS 140-2 compliant on the platforms listed in Table 2 below:

Table 2 – Tested Configuration

#	Operational Environment (on ESXi 6.0 U2)	Processor Family	Optimizations (Target)	EC	B
1	VMware PhotonOS[3] 1.0	Intel Xeon E5	AES-NI[4]	PKB	U2
2	VMware PhotonOS 1.0	Intel Xeon E5	None	PKB	U1
3	NSX Edge OS 3.14 (aka, NSX Edge 6.3.0 OS)	Intel Xeon E5	AES-NI	PKB	U2
4	NSX Edge OS 3.14 (aka, NSX Edge 6.3.0 OS)	Intel Xeon E5	None	PKB	U1
5	NSX Controller OS 12.04 (aka, NSX Controller 6.3.0 OS)	Intel Xeon E5	AES-NI	PKB	U2
6	NSX Controller OS 12.04 (aka, NSX Controller 6.3.0 OS)	Intel Xeon E5	None	PKB	U1
7	NSX Manager OS 3.17 (aka, NSX Manager 6.3.0 OS)	Intel Xeon E5	AES-NI	PKB	U2
8	NSX Manager OS 3.17 (aka, NSX Manager 6.3.0 OS)	Intel Xeon E5	None	PKB	U1
9	SLES[5] 11 SP3	Intel Xeon E5	AES-NI	PKB	U2
10	SLES 11 SP3	Intel Xeon E5	None	PKB	U1
11	Windows 2012	Intel Xeon E5	AES-NI	PKB	W2
12	Windows 2012	Intel Xeon E5	None	PKB	W1
13	Windows 2012 R2	Intel Xeon E5	AES-NI	PKB	W2
14	Windows 2012 R2	Intel Xeon E5	None	PKB	W1
15	Windows 10	Core i	AES-NI	PKB	W2
16	Windows 10	Core i	None	PKB	W1
17	Windows 8.1	Core i	AES-NI	PKB	W2
18	Windows 8.1	Core i	None	PKB	W1
19	Windows 7 SP1	Core i	AES-NI	PKB	W2
20	Windows 7 SP1	Core i	None	PKB	W1
21	Windows Server 2016	Intel Xeon E5	AES-NI	PKB	W2
22	Windows Server 2016	Intel Xeon E5	None	PKB	W1

[3] OS – Operating System
[4] AES-NI – Advanced Encryption Standard – New Instructions
[5] SLES – SUSE Linux Enterprise Server

23	Ubuntu 16.04	Intel Xeon E5	AES-NI	PKB	U2
24	Ubuntu 16.04	Intel Xeon E5	None	PKB	U1
25	Ubuntu 14.04	Intel Xeon E5	AES-NI	PKB	U2
26	Ubuntu 14.04	Intel Xeon E5	None	PKB	U1
27	PhotonOS 2.0	Intel Xeon E5	AES-NI	PKB	U2
28	PhotonOS 2.0	Intel Xeon E5	None	PKB	U1
On ESXI 6.5					
29	Windows 10	Intel Xeon E5	AES-NI	PKB	W2
30	Windows 10	Intel Xeon E5	None	PKB	W1
31	Windows Server 2008	Intel Xeon E5	AES-NI	PKB	W2
32	Windows Server 2008	Intel Xeon E5	None	PKB	W1
33	Windows Server 2012	Intel Xeon E5	AES-NI	PKB	W2
34	Windows Server 2012	Intel Xeon E5	None	PKB	W1
35	Windows Server 2016	Intel Xeon E5	AES-NI	PKB	W2
36	Windows Server 2016	Intel Xeon E5	None	PKB	W1
37	Ubuntu 16.04 (aka, VMware NSX Controller OS 16.04)	Intel Xeon E5	AES-NI	PKB	U2
38	Ubuntu 16.04 (aka, VMware NSX Controller OS 16.04)	Intel Xeon E5	None	PKB	U1
39	Ubuntu 14.04	Intel Xeon E5	AES-NI	PKB	U2
40	Ubuntu 14.04	Intel Xeon E5	None	PKB	U1
41	BLUX 4.4 (aka, VMware NSX Edge OS 4.4)	Intel Xeon E5	AES-NI	PKB	U2
42	BLUX 4.4 (aka, VMware NSX Edge OS 4.4)	Intel Xeon E5	None	PKB	U1
43	BLUX 4.9	Intel Xeon E5	AES-NI	PKB	U2
44	BLUX 4.9	Intel Xeon E5	None	PKB	U1
45	PhotonOS 2.0	Intel Xeon E5	AES-NI	PKB	U2
46	PhotonOS 2.0	Intel Xeon E5	None	PKB	U1
47	PhotonOS 1.0	Intel Xeon E5	AES-NI	PKB	U2
48	PhotonOS 1.0	Intel Xeon E5	None	PKB	U1
49	SLES 12	Intel Xeon E5	AES-NI	PKB	U2
50	SLES 12	Intel Xeon E5	None	PKB	U1
Bare Metal					

51	Windows 10	Intel Core i	AES-NI	PKB	W2
52	Windows 10	Intel Core i	None	PKB	W1
On ESXi 6.7					
53	PhotonOS 2.0	Intel Xeon E5	AES-NI	PKB	U2
54	PhotonOS 2.0	Intel Xeon E5	None	PKB	U1
55	PhotonOS 1.0	Intel Xeon E5	AES-NI	PKB	U2
56	PhotonOS 1.0	Intel Xeon E5	None	PKB	U1
57	SLES 11	Intel Xeon E5	AES-NI	PKB	U2
58	SLES 11	Intel Xeon E5	None	PKB	U1
59	Windows Server 2016	Intel Xeon E5	AES-NI	PKB	W2
60	Windows Server 2016	Intel Xeon E5	None	PKB	W1
61	In ESXi 6.7 (as a host)	Intel Xeon E5	AES-NI	PKB	U2
62	In ESXi 6.7 (as a host)	Intel Xeon E5	None	PKB	U1
63	Ubuntu 16.04	Intel Xeon E5	AES-NI	PKB	U2
64	Ubuntu 16.04	Intel Xeon E5	None	PKB	U1
65	Ubuntu 16.04	Intel Xeon 6126	AES-NI	PKB	U2
66	Ubuntu 16.04	Intel Xeon 6126	None	PKB	U1
67	PhotonOS 2.0	Intel Xeon 6126	AES-NI	PKB	U2
68	PhotonOS 2.0	Intel Xeon 6126	None	PKB	U1
69	In ESXi 6.7 (as a host)	Intel Xeon 6126	AES-NI	PKB	U2
70	In ESXi 6.7 (as a host)	Intel Xeon 6126	None	PKB	U1

Tested Configurations (B = Build Method; EC = Elliptic Curve Support). The EC column indicates support for prime curve only (P), or all NIST defined P, K, and B curves (PKB).

See Appendix A for additional information on build method and optimizations. See Appendix C for a list of the specific compilers used to generate the Module for the respective operational environments.

2.2.1 Physical Cryptographic Boundary

As a software module, there are no physical protection mechanisms implemented. Therefore, the module must rely on the physical characteristics of the host system. The module runs on a General-Purpose Computer (GPC) and the physical boundary of the cryptographic module is defined by the hard enclosure around the host system on which it runs. The module supports the physical interfaces of the GPC. See Figure 1 below for a block diagram of the typical GPC and its physical cryptographic boundary marked with red dotted line.

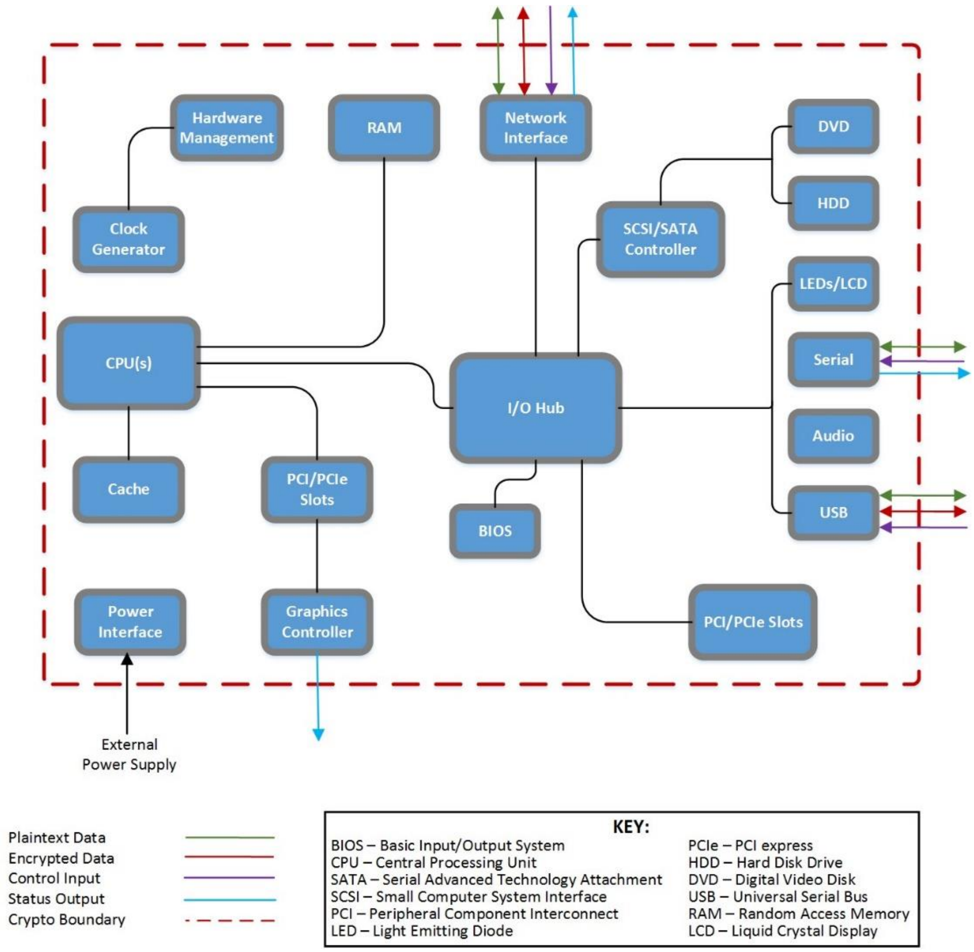

Figure 1 – Hardware Block Diagram

2.2.2 Logical Cryptographic Boundary

The logical cryptographic boundary of the module is the fipscanister object module, a single object module file named *fipscanister.o* (Linux®[6]) or *fipscanister.lib* (Microsoft Windows®[7]). Figure 2 depicts the logical cryptographic boundary for the module which surrounds the VMware OpenSSL FIPS Object Module. The module's logical boundary is a contiguous perimeter that surrounds all memory-mapped functionality provided by the module when loaded and stored in the host platform's memory.

[6] Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.
[7] Windows is a registered trademark of Microsoft Corporation in the United States and other countries.

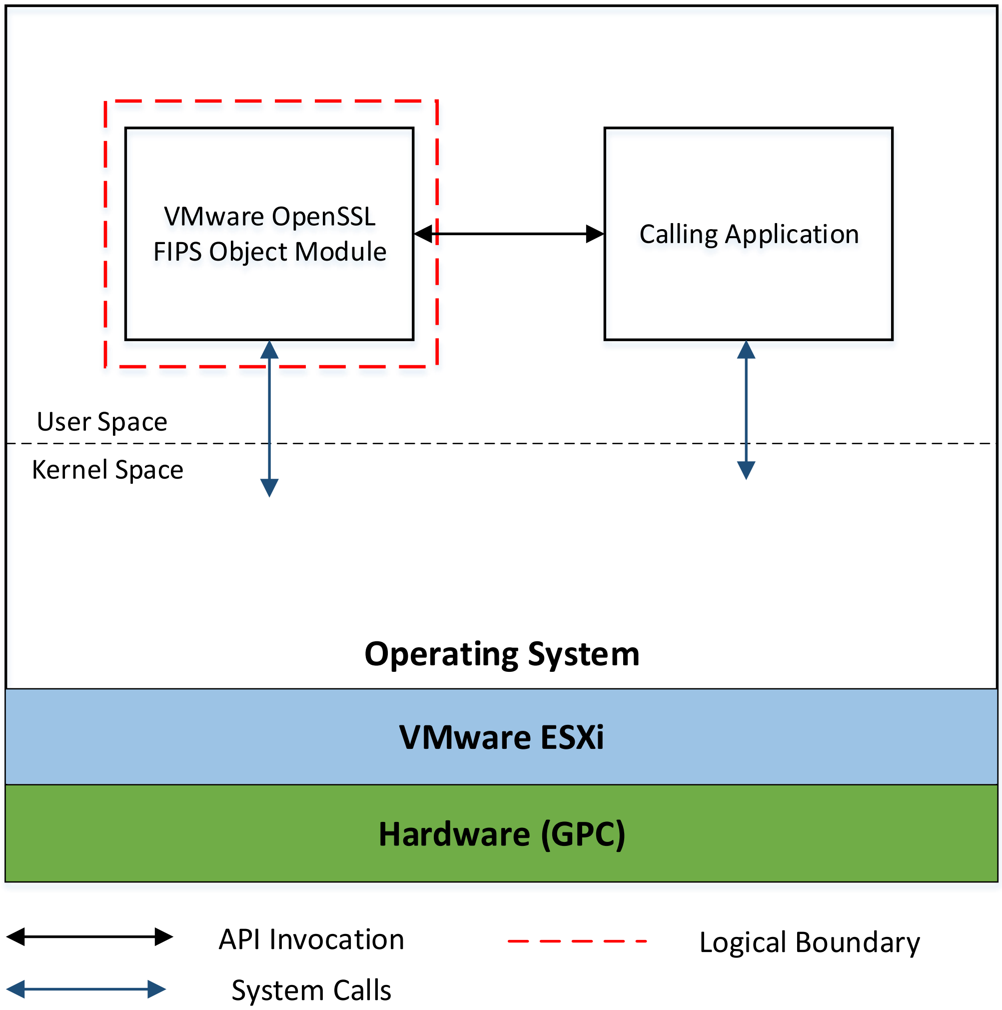

Figure 2 – Module's Logical Cryptographic Boundary

2.2.3 Cryptographic Implementation and modes of operation

The module implements the FIPS-Approved algorithms listed in Table 3 below.

Table 3 – FIPS-Approved Algorithm Implementations

Function	Algorithm	Options	Cert #
Random Number Generation; Symmetric Key Generation	[SP 800-90] DRBG[8] Prediction resistance supported for all variations	Hash DRBG HMAC DRBG, no reseed CTR DRBG (AES), no derivation function	1254

[8] For all DRBGs the "supported security strength" is just the highest supported security strength per [SP 800-90] and [SP 800-57].

Encryption, Decryption and CMAC	[SP 800-67]	3-Key TDES TECB, TCBC, TCFB 1, TCFB 8, TCFB 64, TOFB; CMAC generate and verify	2261
	[FIPS 197] AES [SP 800-38B] CMAC [SP 800-38C] CCM [SP 800-38D] GCM [SP 800-38E] XTS	128/ 192/256 ECB, CBC, OFB, CFB 1, CFB 8, CFB 128, CTR, XTS; CCM; GCM; CMAC generate and verify	4137
Message Digests	[FIPS 180-3]	SHA-1, SHA-2 (224, 256, 384, 512)	3407
Keyed Hash	[FIPS 198] HMAC	HMAC with SHA-1, SHA-2 (224, 256, 384, 512)	2710
Digital Signature and Asymmetric Key Generation	[FIPS 186-2] RSA	GenKey9.31 (2048/3072/4096)	2251
		SigGen9.31, SigGenPSS (4096 with SHA-256, 384, 512)	
		SigGenPKCS1.5 (4096 with SHA-224, 256, 384, 512)	
		SigVer9.31 (1024/1536/2048/3072/4096 with SHA-1, 256, 384, 512)	
		SigVerPKCS1.5 (1024/1536/2048/3072/4096 with SHA-1, 224, 256, 384, 512)	
		SigVerPSS (1024/1536/2048/3072/4096 with SHA-1, 224, 256, 384, 512)	
	[FIPS 186-4] RSA	SigGen9.31 (2048/3072 with SHA-224, 256, 384, 512)	
		SigGenPSS (2048/3072 with SHA-224, 256, 384, 512)	
		SigGenPKCS1.5 (2048/3072 with SHA-224, 256, 384, 512)	
		SigVer9.31 (2048/3072 with SHA-1, 224, 256, 384, 512)	
		SigVerPSS (2048/3072 with SHA-1, 224, 256, 384, 512)	
		SigVerPKCS1.5 (2048/3072 with SHA-1, 224, 256, 384, 512)	
	[FIPS 186-4] DSA	PQG Gen (2048, 224 with SHA-224, 256, 384, 512; 2048, 256 with SHA-256, 384, 512; 3072, 256 with SHA-256, 384, 512)	1123
		PQG Ver (1024, 160 with SHA-1, 224, 256, 384, 512; 2048, 224 with SHA-224, 256, 384, 512; 2048, 256 with SHA-256, 384, 512; 3072,256 with SHA-256, 384, 512)	
		KeyPairGen (2048, 224; 2048, 256; 3072, 256)	
		SigGen (2048, 224 with SHA-224, 256, 384, 512; 2048, 256 with SHA-224, 256, 384, 512; 3072, 256 with SHA-224, 256, 384, 512)	
		SigVer (1024/2048/3072 with SHA-1, 224, 256, 384, 512)	

	[FIPS 186-4] ECDSA	PKG: CURVES (P-224 P-256 P-384 P-521 K-233 K- 283 K-409 K-571 B-233 B-283 B-409 B-571 ExtraRandomBits TestingCandidates)	949
		PKV: CURVES (ALL-P ALL-K ALL-B)	
		SigGen: CURVES(P-224: (SHA-224, 256, 384, 512) P-256: (SHA-224, 256, 384, 512) P-384: (SHA-224, 256, 384, 512) P-521: (SHA-224, 256, 384, 512) K-233: (SHA-224, 256, 384, 512) K-283: (SHA-224, 256, 384, 512) K-409: (SHA-224, 256, 384, 512) K-571: (SHA-224, 256, 384, 512) B-233: (SHA-224, 256, 384, 512) B-283: (SHA-224, 256, 384, 512) B-409: (SHA-224, 256, 384, 512) B-571: (SHA-224, 256, 384, 512))	
		SigVer: CURVES(P-192: (SHA-1, 224, 256, 384, 512) P-224: (SHA-1, 224, 256, 384, 512) P-256: (SHA-1, 224, 256, 384, 512) P-384: (SHA-1, 224, 256, 384, 512) P-521: (SHA-1, 224, 256, 384, 512) K-163: (SHA-1, 224, 256, 384, 512) K-233: (SHA-1, 224, 256, 384, 512) K-283: (SHA-1, 224, 256, 384, 512) K-409: (SHA-1, 224, 256, 384, 512) K-571: (SHA-1, 224, 256, 384, 512) B-163: (SHA-1, 224, 256, 384, 512) B-233: (SHA-1, 224, 256, 384, 512) B-283: (SHA-1, 224, 256, 384, 512) B-409: (SHA-1, 224, 256, 384, 512) B-571: (SHA-1, 224, 256, 384, 512))	
ECC CDH (KAS)	[SP 800-56A] (§5.7.1.2)	All NIST defined B, K and P curves except sizes 163 and 192	943

The module supports only NIST defined curves for use with ECDSA and ECC CDH. The module supports two operational environments configurations for elliptical curves; NIST prime curve only and all NIST defined PKB curves.

The module also employs the following key establishment methodologies, which are allowed to be used in FIPS-Approved mode of operation:

- RSA (key wrapping[9]; key establishment methodology provides between 112 and 256 bits of encryption strength)
- EC DH (key agreement[10]; key establishment methodology provides between 112 and 256 bits of encryption strength)

The module employs non-compliant algorithms and associated services, which are not allowed for use in a FIPS-Approved mode of operation. Their use will result in the module operating in a non- Approved mode. Please refer to Table 4 below for the list of non-Approved algorithms and associated services.

[9] No claim is made for SP 800-56B compliance, and no CSPs are established into or exported out of the module using this service.

[10] Non-compliant (untested) DH scheme using all NIST defined P, K, and B elliptical curves. Key agreement is a service provided for calling process use, but is not used to establish keys into the module.

Table 4 – Non FIPS-Approved Algorithm Implementations and services

Algorithm	Options	Description
ANSI X9.31 PRNG	AES 128/192/256	Random Number Generation; Symmetric Key Generation
SP 800-90A Dual_EC_DRBG	Dual EC DRBG	Random Number Generation; Symmetric Key Generation
RSA (FIPS 186-2)	KeyGen9.31, SigGen9.31, SigGenPKCS1.5, SigGenPSS (1024/1536 with all SHAs, 2048/3072/4096 with SHA-1)	Digital Signature Generation and Asymmetric Key Generation
DSA (FIPS 186-2)	PQG Gen, Key Pair Gen, SigGen (1024 with all SHAs, 2048/3072 with SHA-1)	Digital Signature Generation and Asymmetric Key Generation
DSA (FIPS 186-4)	PQG Gen, Key Pair Gen, SigGen (1024 with all SHAs, 2048/3072 with SHA-1)	Digital Signature Generation and Asymmetric Key Generation
ECDSA (FIPS 186-2)	PKG: Curve (P-192 K-163 B-163) SIG(gen): Curve (P-192 P-224 P-256 P-384 P-521 K-163 K-233 K-283 K-409 K-571 B-163 B-233 B-283 B-409 B-571)	Digital Signature Generation and Asymmetric Key Generation
ECDSA (FIPS 186-4)	PKG: Curve (P-192 K-163 B-163) SigGen: Curve (P-192: (SHA-1, 224, 256, 384, 512) P-224:(SHA-1) P-256:(SHA-1) P-384: (SHA-1) P-521:(SHA-1) K-163: (SHA-1, 224, 256, 384, 512) K-233:(SHA-1) K-283:(SHA-1) K-409:(SHA-1) K-571:(SHA-1) B-163: (SHA-1, 224, 256, 384, 512) B-233: (SHA-1) B-283: (SHA-1) B-409:(SHA-1) B-571:(SHA-1))	Digital Signature Generation and Asymmetric Key Generation
ECC CDH (KAS, SP800-56A – 5.7.1.2)	All NIST recommended P, K, and B with Curves 163 and 192	Key Agreement Scheme

The module requires an initialization sequence (see IG 9.5): the calling application invokes *FIPS_mode_set()*[11], which returns a "1" for success and "0" for failure. If *FIPS_mode_set()* fails, then all cryptographic services fail from then on. The application can test to see if FIPS mode has been successfully performed.

The Module is a cryptographic engine library, which can be used only in conjunction with additional software. Aside from the use of the NIST defined elliptic curves as trusted third-party domain parameters, all other FIPS 186-3 assurances are outside the scope of the module, and are the responsibility of the calling process.

2.3 Module Interfaces

The module's logical interfaces exist at a low level in the software as an API. Both the API and physical interfaces can be categorized into the following interfaces defined by FIPS 140-2:

- Data input

[11] The function call in the module is *FIPS_module_mode_set()* which is typically used by an application via the *FIPS_mode_set()* wrapper function.

- Data output
- Control input
- Status output
- Power input

As a software module, the module's manual controls, physical indicators, and physical and electrical characteristics are those of the host platform. A mapping of the FIPS 140-2 logical interfaces, the physical interfaces, and the module interfaces can be found in Table 5 below.

Table 5 – FIPS 140-2 Logical Interface Mapping

FIPS Interface	Physical Interface	Module Interface (API)
Data Input	Network port, Serial port, USB port, SCSI/SATA Controller	The function calls that accept input data for processing through their arguments.
Data Output	Network port, Serial port, USB port, SCSI/SATA Controller	The function calls that return by means of their return codes or argument generated or processed data back to the caller.
Control Input	Network port, Serial port, USB port, Power button	The function calls that are used to initialize and control the operation of the module.
Status Output	Network port, Serial port, USB port, Graphics controller	Return values for function calls; Module generated error messages.
Power Input	AC Power socket	Not applicable.

As a software module, control of the physical ports is outside module scope. However, when the module is performing self-tests, or is in error state, all output on the logical data output interface is inhibited. The module is single-threaded and in error states returns only an error value, and no data output is returned.

2.4 Roles and Services

There are two roles in the module (as required by FIPS 140-2) that operators may assume: a Cryptographic Officer (CO) role and a User role. The module implements authentication of the operators. Roles are assumed implicitly by passing the appropriate password to the *FIPS_module_mode_set()* function. The password values may be specified at build time and must have a minimum length of 16 characters. Any attempt to authenticate with an invalid password will result in an immediate and permanent failure condition rendering the module unable to enter the FIPS mode of operation, even with subsequent use of a correct password. Authentication data is loaded into the module during the module build process, performed by the Crypto Officer, and otherwise cannot be accessed.

Since the minimum password length is 16 characters, the probability of a random successful authentication attempt in one try is a maximum of $1/256^{16}$, or less than $1/10^{38}$. The module permanently disables further authentication attempts after a single failure, so this probability is independent of time.

Only one role may be active at a time and the module does not allow concurrent operators. Each role and their corresponding services are detailed in the sections below. Please note that the keys and Critical Security Parameters (CSPs) listed in Table 6 below indicates the types of access required using the following notation:

- R – Read: The CSP is read.
- W – Write: The CSP is established, generated, modified, or zeroized.
- X – Execute: The CSP is used within an FIPS-Approved or Allowed security function or authentication mechanism.

2.4.1 Crypto Officer and User Roles

The CO and User roles share many services. Both roles have access to all of the services provided by the module.

- Crypto Officer Role: Installation of the Module on the host computer system and calling of any API functions.
- User Role: Loading of the module and calling any of the API functions.

Below, Table 6 describes the CO and User services and CSP access, while Table 4 in Section 2.2.3 above describes the Non-Approved algorithms and services.

Table 6 – Crypto Officer and Users Services

Role	Service	Description	CSP and Type of Access
CO, User	Initialization of the module	Initialization of the module following the Secure Operation section of the Security Policy	None
CO, User	Run self-test	Runs Self-tests on demand during module operation	None
CO, User	Show status	Returns the current mode (Boolean) of operation of the module, and version (as unsigned long or const char*)	None
CO, User	Zeroize	Zeroizes all CSPs	All CSPs - W
CO, User	Random number generation	Generate random number and symmetric key by using the DRBGs	DRBGs CSPs – RXW
CO, User	Asymmetric key generation	Generate RSA, DSA, and ECDSA key pairs	RSA SGK – W RSA SVK – W DSA SGK – W DSA SVK – W ECDSA SGK – W ECDSA SVK – W
CO, User	Symmetric Encryption/Decryption	Encrypt or decrypt data using supplied key and algorithm specification (key passed in by the calling process)	AES EDK – RX Triple-DES EDK – RX
CO, User	Symmetric digest (CMAC)	Generate or verify data integrity using CMAC with AES or TDES (key passed in by the calling process)	AES CMAC – RX Triple-DES CMAC – RX
CO, User	Hash generation	Compute and return a message digest using SHA algorithm	None
CO, User	Message Authentication Code generation (HMAC)	Compute and return a hashed message authentication code	HMAC Key – RX

CO, User	Transport[12] key	Wrap/unwrap a key on behalf of the calling application but does not establish keys into the module (key passed in by the calling process)	RSA KEK – RX RSA KDK – RX
CO, User	Key agreement	Perform key agreement primitives on behalf of the calling process but does not establish keys into the module (keys passed in by the calling process)	EC DH Private/Public Key – RX
CO, User	Digital signature	Generate and verify RSA, DSA, and ECDSA digital signatures (keys passed in by the calling process)	RSA SGK – RX RSA SVK – RX DSA SGK – RX DSA SVK – RX ECDSA SGK – RX ECDSA SVK – RX
CO, User	Utility	Miscellaneous helper functions	None

2.5 Physical Security

The VMware OpenSSL FIPS Object Module is a software module, which FIPS defines as a multi-chip standalone cryptographic module. As such, it does not include physical security mechanisms. Thus, the FIPS 140-2 requirements for physical security are not applicable.

2.6 Operational Environment

The module was tested and found to be compliant with FIPS 140-2 requirements on the following platforms:

- A Dell PowerEdge T620 with an Intel Xeon E5-2440 processor running VMware ESXi 6.0 U2 and VMware PhotonOS 1.0, PhotonOS 2.0, NSX Edge OS 3.14 (aka NSX Edge 6.3.0 OS), NSX Controller OS 12.04 (aka NSX Controller 6.3.0 OS), NSX Manager OS 3.17 (aka NSX Manager 6.3.0 OS), SLES 11 SP3, Ubuntu 16.04, Ubuntu 14.04, Windows 2012, Windows 2012 R2, or Windows Server 2016.

- A Dell PowerEdge T620 with an Intel Xeon E5-2440 processor running VMware ESXi 6.5 and VMware BLUX 4.4 (aka, VMware NSX Edge OS 4.4), Windows Server 2016, Ubuntu 16.04 (aka, VMware NSX Controller OS 16.04), Ubuntu 14.04, PhotonOS 2.0, or PhotonOS 1.0.

- A Dell PowerEdge T620 with an Intel Xeon E5 processor running VMware ESXi 6.5 and BLUX 4.9, SLES 12, Windows Server 2012, Windows Server 2008, or Windows 10 operating system.

- A Dell computer with Intel Core i processor running VMware ESXi 6.0 U2 and Windows 10, Windows 8.1, or Windows 7 SP1.

[12] "Key transport" can refer to a) moving keys in and out of the module or b) the use of keys by an external application. The latter definition is the one that applies to the VMware OpenSSL FIPS Object Module.

- A Dell Computer with Intel Core i processor running Windows 10 operating system.

- A Dell PowerEdge T620 with an Intel Xeon E5-2440 processor running VMware ESXi 6.7 and PhotonOS 1.0, PhotonOS 2.0, SLES 11, Ubuntu 16.04, or Windows Server 2016.

- A Dell PowerEdge T620 with an Intel Xeon E5-2440 processor running VMware ESXi 6.7.

- A Dell PowerEdge R740 with an Intel Xeon 6126 processor running Ubuntu 16.04 or PhotonOS 2.0 on VMware ESXi 6.7.

- A Dell PowerEdge R740 with an Intel Xeon 6126 processor running VMware ESXi 6.7

Further, VMware, Inc. affirms that the VMware OpenSSL FIPS Object Module runs in its configured, Approved mode of operation on the following binary compatible platforms executing VMware ESXi 6.0, ESXi 6.5, or ESXi 6.7 with any of the above listed OS:

- Dell PowerEdge T320 with Intel Xeon Processor
- Dell PowerEdge R530 with Intel Xeon Processor
- Dell PowerEdge R730 with Intel Xeon Processor
- Dell PowerEdge R830 with Intel Xeon Processor
- Dell PowerEdge R740 Gen14 with Intel Xeon Processor
- HPE ProLiant DL380 Gen9 with Intel Xeon Processor
- HPE ProLiant DL38P Gen8 with AMD Opteron Processor
- Cisco UCS – B22 M Series Blade Servers with Intel Processor
- Cisco UCS – C24 M3 Series Rackmount with Intel Xeon Processor
- A general-purpose computer platform with an Intel Core I, Intel Xeon, or AMD Opteron Processor executing VMware ESXi and any OS (including Apple OS (OS X, macOS, iOS), Android OS, and others) with single user mode.
- A cloud computing environment composed of a general-purpose computing platform executing VMware ESXi or a VMware cloud solution that is executing VMware ESXi.

CMVP makes no claims to the correct operation of the module or the minimum strength of generated keys when ported to an OE not on the validation certificate.

In addition to its full AES software implementations, the VMware OpenSSL FIPS Object Module is capable of leveraging the AES-NI instruction set of supported Intel and AMD processors in order to accelerate AES calculations.

All cryptographic keys and CSPs are under the control of the OS, which protects its CSPs against unauthorized disclosure, modification, and substitution. The module only allows access to CSPs through its well-defined API.

The tested operating systems segregate user processes into separate process spaces. Each process space is logically separated from all other processes by the operating system software and hardware. The Module functions entirely within the process space of the calling application, and implicitly satisfies the FIPS 140-2 requirement for a single user mode of operation.

2.7 Cryptographic Key Management

The module supports the CSPs listed below in Table 7 and Table 8.

Table 7 – List of Cryptographic Keys, Key Components, and CSPs

CSP Name	Description
RSA SGK	RSA (1024 to 16384 bits) signature generation key
RSA KDK	RSA (1024 to 16384 bits) key decryption (private key transport) key
DSA SGK	[FIPS 186-4] DSA (1024/2048/3072) signature generation key or [FIPS 186-2] DSA (1024) signature generation key
ECDSA SGK	ECDSA (All NIST defined B, K, and P curves) signature generation key
EC DH Private	EC DH (All NIST defined B, K, and P curves) private key agreement key
AES EDK	AES (128/192/256) encrypt / decrypt key
AES CMAC	AES (128/192/256) CMAC generate / verify key
AES GCM	AES (128/192/256) encrypt / decrypt / generate / verify key
AES XTS	AES (256/512) XTS encrypt / decrypt key
TDES EDK	TDES (3-Key) encrypt / decrypt key
TDES CMAC	TDES (3-Key) CMAC generate / verify key
HMAC Key	Keyed hash key (160/224/256/384/512)
Hash_DRBG CSPs	V (440/888 bits) and C (440/888 bits), entropy input (length dependent on security strength)
HMAC_DRBG CSPs	V (160/224/256/384/512 bits) and Key (160/224/256/384/512 bits), entropy input (length dependent on security strength)
CTR_DRBG CSPs	V (128 bits) and Key (AES 128/192/256), entropy input (length dependent on security strength)
CO-AD-Digest	Pre-calculated HMAC SHA-1 digest used for Crypto Officer role authentication
User-AD-Digest	Pre-calculated HMAC SHA-1 digest used for User role authentication

Authentication data is loaded into the module during the module build process, performed by an authorized operator (Crypto Officer), and otherwise cannot be accessed.

The module does not output intermediate key generation values.

Table 8 – List of Public Keys, Key Components, and CSPs

CSP Name	Description
RSA SVK	RSA (1024 to 16384 bits) signature generation key
RSA KEK	RSA (1024 to 16384 bits) key decryption (private key transport) key
DSA SVK	[FIPS 186-4] DSA (1024/2048/3072) signature generation key or [FIPS 186-2] DSA (1024) signature generation key
ECDSA SVK	ECDSA (All NIST defined B, K, and P curves) signature generation key
EC DH Public	EC DH (All NIST defined B, K, and P curves) private key agreement key

For all CSPs and Public Keys:

Storage: RAM, associated to entities by memory location. The module stores DRBG state values for the lifetime of the DRBG instance. The module uses CSPs passed in by the calling application on the stack. The module does not store any CSP persistently (beyond the lifetime of an API call), with the exception of DRBG state values used for the modules' default key generation service.

Generation: The module implements 800-90A compliant DRBG services for creation of symmetric keys, and for generation of DSA, elliptic curve, and RSA keys as shown in Table 3. The calling application is responsible for storage of generated keys returned by the module.

Entry: All CSPs enter the module's logical boundary in plaintext as API parameters, associated by memory location. However, none cross the physical boundary.

Output: The module does not output CSPs, other than as explicit results of key generation services. However, none cross the physical boundary.

Destruction: Zeroization of sensitive data is performed automatically by API function calls for temporarily stored CSPs. In addition, the module provides functions to explicitly destroy CSPs related to random number generation services. The calling application is responsible for parameters passed in and out of the module.

Private and secret keys as well as seeds and entropy input are provided to the Module by the calling application, and are destroyed when released by the appropriate API function calls. Keys residing in internally allocated data structures (during the lifetime of an API call) can only be accessed using the module defined API. The operating system protects memory and process space from unauthorized access. Only the calling application that creates or imports keys can use or export such keys. All API functions are executed by the invoking calling application in a non-overlapping sequence such that no two API functions will execute concurrently. An authorized application as user (Crypto Officer and User) has access to all key data generated during the operation of the module.

In the event module power is lost and restored the calling application must ensure that any AES-GCM keys used for encryption or decryption are redistributed.

Module users (the calling applications) shall use entropy sources that meet the security strength required for the random number generation mechanism as shown in [SP 800-90A] Table 2 (Hash_DRBG, HMAC_DRBG) and Table 3 (CTR_DRBG). This entropy is supplied by means of callback functions. Those functions must return an error if the minimum entropy strength cannot be met.

2.8 Self-Tests

Cryptographic self-tests are performed by the module on invocation of Initialize or Self-test, as well as when the module is operating in the FIPS-Approved mode and when a random number is generated, or asymmetric keys are generated. The following sections list the self-tests performed by the module, their expected error status, and any error resolutions.

2.8.1 Power-Up Self-Tests

The Module performs the self-tests listed below on invocation of Initialize or Self-test.

The VMware OpenSSL FIPS Object Module performs the following Power-up Self-tests:

- Software integrity check (HMAC SHA-1 Integrity Test)
- Known Answer Tests (KATs)
  - AES Encryption KAT in ECB mode with 128-bit key
  - AES Decryption KAT in ECB mode with 128-bit key
  - AES CCM Encryption KAT with 192-bit key
  - AES CCM Decryption KAT with 192-bit key
  - AES GCM Encryption KAT with 256-bit key
  - AES GCM Decryption KAT with 256-bit key
  - XTS-AES KAT with 128, 256-bit key sizes to support either 256-bit key size (for XTS-AES-128) or the 512-bit key size (for XTS-AES-256)
  - AES CMAC Sign KAT with 128, 192, 256-bit keys
  - AES CMAC Verify KAT with 128, 192, 256-bit keys
  - Triple-DES Encryption KAT in ECB mode with 3-Key
  - Triple-DES Decryption KAT in ECB mode with 3-Key
  - Triple-DES CMAC Generate KAT in CBC mode with 3-Key
  - Triple-DES CMAC Verify KAT in CBC mode with 3-Key
  - HMAC SHA-1, SHA-224, SHA-256, SHA-384, SHA-512 KATs (Per IG 9.3, this testing covers SHA POST requirements)
  - RSA (PKCS#1) Signature Generation KAT using 2048-bit key and SHA-256
  - RSA (PKCS#1) Signature Verification KAT using 2048-bit key and SHA-256
  - DSA Signature Generation KAT using 2048-bit key and SHA-384
  - DSA Signature Verification KAT using 2048-bit key and SHA-384
  - CTR_DRBG KAT with AES 256-bit key and with and without derivation function
  - HASH_DRBG KAT with SHA-256
  - HMAC_DRBG KAT with SHA-256
  - ECDSA Pairwise Consistency Test (KeyGen, Sign, Verify using P-224, K-233 and SHA-512)
  - ECC CDH KAT (Shared secret calculation per SP 800-56A, 5.7.1.2, IG 9.6)

The module is installed using one of the set of instructions in Appendix A, as appropriate for the target system. The HMAC SHA-1 of the module distribution file as tested by the CMT Laboratory and listed in Appendix A is verified during installation of the module file as described in Appendix A.

The FIPS_mode_set()[13] function performs all power-up self-tests listed above with no operator intervention required, returning a “1” if all power-up self-tests succeed, and a “0” otherwise. If any component of the power-up self-test fails an internal flag is set to prevent subsequent invocation of any cryptographic function calls. The module will only enter the FIPS Approved mode if the module is reloaded and the call to FIPS_mode_set() succeeds.

[13] FIPS_mode_set() calls module function FIPS_module_mode_set()

The power-up self-tests may also be performed on-demand by calling FIPS_selftest() , which returns a “1” for success and “0” for failure. Interpretation of this return code is the responsibility of the calling application.

2.8.2 Conditional Self-Tests

The module also implements the following conditional self-tests:

- DRBG Continuous RNG[25] Test for stuck fault.
- DRBG Health Tests as required by Section 11 of SP 800-90A
- DSA Pairwise Consistency Test on each key pair generation
- ECDSA Pairwise Consistency Test on each key pair generation
- RSA Pairwise Consistency Test on each key pair generation

In the event of a DRBG self-test failure the calling application must uninstantiate and reinstantiate the DRBG per the requirements of [SP 80090A]; this is not something the module can do itself.

Pairwise consistency tests are performed for both possible modes of use, e.g. Sign/Verify and Encrypt/Decrypt.

The Module supports two operational environment configurations for elliptic curve: NIST prime curves only (listed in Table 2 with the EC column marked "P") and all NIST defined curves (listed in Table 2 with the EC column marked "PKB").

2.9 Mitigation of Other Attacks

This section is not applicable. The module was not designed to mitigate any attacks beyond the FIPS 140-2 Level 1 requirements for this validation.

3 Secure Operation

The VMware OpenSSL FIPS Object Module meets Level 1 requirements for FIPS 140-2. The sections below describe how to install, use, and keep the module in FIPS-Approved mode of operation.

3.1 Appendix A: Installation and Usage Guidance

The test platforms represent different combinations of installation instructions. For each platform there is a build system, the host providing the build environment in which the installation instructions are executed, and a target system on which the generated object code is executed. The build and target systems may be the same type of system or even the same device, or may be different systems – the module supports cross-compilation environments.

Each of these command sets are relative to the top of the directory containing the uncompressed and expanded contents of the distribution files openssl-fips-2.0.9.tar.gz (all NIST defined curves as listed in Table 2 with the EC column marked "PKB") or openssl-fips-ecp-2.0.9.tar.gz (NIST prime curves only as listed in Table 2 with the EC column marked "P"). The command sets are:

```
U1:
      ./config no-asm
      make
      make install
U2:
      ./config
      make
      make install
W1:
      ms\do_fips no-asm
W2:
     ms\do_fips
```

Installation instructions

1. Download and copy the distribution file to the build system.
   These files can be downloaded from http://www.openssl.org/source/.

2. Verify the HMAC SHA-1 digest of the distribution file; see Appendix B. An independently acquired FIPS 140-2 validated implementation of SHA1 HMAC must be used for this digest verification. Note that this verification can be performed on any convenient system and not necessarily on the specific build or target system. Alternatively, a copy of the distribution on physical media can be obtained from OSF[14].

[14] For some prospective users the acquisition, installation, and configuration of a suitable FIPS 140-2 validated product may not be convenient. OSF will on request mail a CD containing the source code distribution, via USPS or international post. A distribution file received by that means need not be verified by a FIPS 140-2 validated implementation of HMAC SHA-1. For instructions on requesting this CD see http://openssl.com/fips/verify.html.

3. Unpack the distribution

   gunzip -c openssl-fips-2.0.9.tar.gz | tar xf -
   cd openssl-fips-2.0.9

   or

   gunzip -c openssl-fips-ecp-2.0.9.tar.gz | tar xf -
   cd openssl-fips-ecp-2.0.9

4. Execute one of the installation command sets U1, W1, U2, W2 as shown above. No other command sets shall be used.

5. The resulting *fipscanister.o* or *fipscanister.lib* file is now available for use.

6. The calling application enables FIPS mode by calling the *FIPS_mode_set()*[15] function.

Note that failure to use one of the specified commands sets exactly as shown will result in a module that cannot be considered compliant with FIPS 140-2.

Linking the Runtime Executable Application

Note that applications interfacing with the FIPS Object Module are outside of the cryptographic boundary. When linking the application with the FIPS Object Module two steps are necessary:

1. The HMAC SHA-1 digest of the FIPS Object Module file must be calculated and verified against the installed digest to ensure the integrity of the FIPS object module.

2. A HMAC SHA-1 digest of the FIPS Object Module must be generated and embedded in the FIPS Object Module for use by the FIPS_mode_set()[16] function at runtime initialization.

The *fips_standalone_sha1* command can be used to perform the verification of the FIPS Object Module and to generate the new HMAC SHA-1 digest for the runtime executable application. Failure to embed the digest in the executable object will prevent initialization of FIPS mode.

At runtime the *FIPS_mode_set()* function compares the embedded HMAC SHA-1 digest with a digest generated from the FIPS Object Module object code. This digest is the final link in the chain of validation from the original source to the runtime executable application file.

Optimization

The “asm” designation means that assembler language optimizations were enabled when the binary code was built, “no-asm” means that only C language code was compiled.

For OpenSSL with x86 there are three possible optimization levels:

1. No optimization (plain C)
2. SSE2 optimization
3. AES-NI+PCLMULQDQ+SSSE3 optimization

Other theoretically possible combinations (e.g. AES-NI only, or SSE3 only) are not addressed individually, so that a processor which does not support all three of AES-NI, PCLMULQDQ, and SSSE3 will fall back to SSE2 optimization.

[15] *FIPS_mode_set()* calls the module function *FIPS_module_mode_set()*

For more information, see:

- http://www.intel.com/support/processors/sb/CS-030123.htm?wapkw=sse2
- http://software.intel.com/en-us/articles/intel-advanced-encryption-standard-instructions-aes-ni/?wapkw=aes-ni

For OpenSSL with ARM there are two possible optimization levels:

1. Without NEON
2. With NEON (ARM7 only)

For more information, see http://www.arm.com/products/processors/technologies/neon.php

3.2 Appendix B: Controlled Distribution File Fingerprint

The VMware OpenSSL FIPS Object Module v2.0.9 consists of the FIPS Object Module (the *fipscanister.o* or *fipscanister.lib* contiguous unit of binary object code) generated from the specific source files.

For all NIST defined curves (listed in Table 2 with the EC column marked "PKB") the source files are in the specific special OpenSSL distribution *openssl-fips-2.0.9.tar.gz* with HMAC SHA-1 digest of

    54552e9a3ed8d1561341e8945fcdec55af961322

located at http://www.openssl.org/source/openssl-fips-2.0.9.tar.gz.

The *openssl* command from a version of OpenSSL that incorporates a previously validated version of the module may be used:

openssl sha1 -hmac etaonrishdlcupfm openssl-fips-2.0.9.tar.gz

For NIST prime curves only (listed in Table 2 with the EC column marked "P") the source files are in the specific special OpenSSL distribution *openssl-fips-ecp-2.0.9.tar.gz* with HMAC SHA-1 digest of

    91d267688713c920f85bc5e69c8b5d34e1112672

located at http://www.openssl.org/source/openssl-fips-ecp-2.0.9.tar.gz. Note this is from the previous revision of the FIPS Object Module as no modifications relevant to NIST prime curves only were introduced in revision 2.0.9.

The set of files specified in this tar file constitutes the complete set of source files of this module. There shall be no additions, deletions, or alterations of this set as used during module build. The OpenSSL distribution tar file (and patch file if used) shall be verified using the above HMAC SHA-1 digest(s).

The arbitrary 16-byte key of:

65 74 61 6f 6e 72 69 73 68 64 6c 63 75 70 66 6d

(equivalent to the ASCII string "*etaonrishdlcupfm*") is used to generate the HMAC SHA-1 value for the FIPS Object Module integrity check.

The functionality of all earlier revisions of the FIPS Object Module are subsumed by this latest revision, so there is no reason to use older revisions for any new deployments. However, older revisions remain valid. The source distribution files and corresponding HMAC SHA-1 digests are listed below:

openssl-fips-2.0.8.tar.gz
URL: http://www.openssl.org/source/old/fips/openssl-fips-2.0.8.tar.gz

Digest: 7f486fbb598f3247ab9db10c1308f1c19f384671

Openssl-fips-ecp-2.0.8.tar.gz
URL: http://www.openssl.org/source/old/fips/openssl-fips-ecp-2.0.8.tar.gz
Digest: 7a5f40ef8cebe959372d16e26391fcf23689209b

Openssl-fips-2.0.7.tar.gz
URL: http://www.openssl.org/source/old/fips/openssl-fips-2.0.7.tar.gz
Digest: 295064925a6d95271e2fa2920181ec060f95c7ab

Openssl-fips-ecp-2.0.7.tar.gz
URL: http://www.openssl.org/source/old/fips/openssl-fips-ecp-2.0.7.tar.gz
Digest: dddfdc78c7e827c61fe92bd4817a7f2c3e67153

openssl-fips-2.0.6.tar.gz
URL: http://www.openssl.org/source/old/fips/openssl-fips-2.0.6.tar.gz
Digest: 2b8d831df22d4dfe6169aa2a8e74c35484c26c21

openssl-fips-ecp-2.0.6.tar.gz
URL: http://www.openssl.org/source/old/fips/openssl-fips-ecp-2.0.6.tar.gz
Digest: 852f43cd9ae1bd2eba60e4f9f1f266d3c16c0319

openssl-fips-2.0.5.tar.gz
URL: http://www.openssl.org/source/old/fips/openssl-fips-2.0.5.tar.gz
Digest: 8b44f2a43d098f6858eb1ebe77b73f8f027a9c29

openssl-fips-ecp-2.0.5.tar.gz
URL: http://www.openssl.org/source/old/fips/openssl-fips-ecp-2.0.5.tar.gz
Digest: 148e4e127ffef1df80c0ed61bae35b07ec7b7b36

openssl-fips-2.0.4.tar.gz
URL: http://www.openssl.org/source/old/fips/openssl-fips-2.0.4.tar.gz
Digest: eaa5f86dab2c5da7086aec4786bce27d3b3c1b8a

openssl-fips-ecp-2.0.4.tar.gz
URL: http://www.openssl.org/source/old/fips/openssl-fips-ecp-2.0.4.tar.gz
Digest: 13302f75c82c8b482c9ac96828984a270a45c284

openssl-fips-2.0.3.tar.gz
URL: http://www.openssl.org/source/old/fips/openssl-fips-2.0.3.tar.gz
Digest: 5dfe03bc3f57c2862ea97823ea3111d7faf711b2

openssl-fips-ecp-2.0.3.tar.gz
URL: http://www.openssl.org/source/old/fips/openssl-fips-ecp-2.0.3.tar.gz
Digest: 9d6b21218d7d5480aa0add68e682d321e3ffbfa7

openssl-fips-2.0.2.tar.gz
URL: http://www.openssl.org/source/old/fips/openssl-fips-2.0.2.tar.gz
Digest: e099d5096eb69c2dd8591379f38b985801188663

openssl-fips-ecp-2.0.2.tar.gz
URL: http://www.openssl.org/source/old/fips/openssl-fips-ecp-2.0.2.tar.gz
Digest: 887fa6802c253c32e6c4c83b7a091118fa8c6217

openssl-fips-2.0.1.tar.gz
URL: http://www.openssl.org/source/old/fips/openssl-fips-2.0.1.tar.gz

Digest: 1e05b021fdcd6e77c6155512bbce2d0cbc725aec

openssl-fips-ecp-2.0.1.tar.gz
URL: http://www.openssl.org/source/old/fips/openssl-fips-ecp-2.0.1.tar.gz
Digest: af82c8ebb9d3276be11feffd35e6b55bd0d1839f

openssl-fips-2.0.tar.gz
URL: http://www.openssl.org/source/old/fips/openssl-fips-2.0.tar.gz
Digest: 2cdd29913c6523df8ad38da11c342b80ed3f1dae

openssl-fips-ecp-2.0.tar.gz
URL: http://www.openssl.org/source/openssl-fips-ecp-2.0.tar.gz
Digest: e8d5ee306425b278bf6c8b077dae8e4a542e8215

3.3 Appendix C: Compilers

This appendix lists the specific compilers (see Table 9) used to generate the Module for the respective Operational Environments. Note this list does not imply that use of the Module is restricted to only the listed compiler versions, only that the use of other versions has not been confirmed to produce a correct result. VMware has confirmed all of the compilers specified may be used to re-compile the unmodified source code including gcc 4.4, gcc 4.8, gcc 6, and others for Linux platforms and 19.12.25835 and all earlier versions for Windows platforms.

Table 9 – Compilers

#	Operational Environment	Compiler
1	VMware PhotonOS 1.0	gcc 5.3.0
2	NSX Edge OS 3.14	gcc 4.6.3
3	NSX Controller OS 12.04	gcc 4.6.3
4	NSX Manager OS 3.17	gcc 4.6.3
5	Windows 2012	Microsoft C/C++ Optimizing Compiler Version 18.00.21005.1
6	Windows 2012 R2	Microsoft C/C++ Optimizing Compiler Version 18.00.21005.1
7	Windows 10	Microsoft C/C++ Optimizing Compiler Version 18.00.21005.1
8	Windows 8.1	Microsoft C/C++ Optimizing Compiler Version 18.00.21005.1
9	Windows 7 SP1	Microsoft C/C++ Optimizing Compiler Version 18.00.21005.1
10	SLES 11 SP3	gcc 5.3.0
	SLES 12	gcc 5.3.0
11	Windows Server 2016	Microsoft C/C++ Optimizing Compiler Version 18.00.21005.1
12	Windows Server 2008	Microsoft C/C++ Optimizing Compiler Version 18.00.21005.1

#	Operational Environment	Compiler
13	Windows Server 2012	Microsoft C/C++ Optimizing Compiler Version 18.00.21005.1
14	Ubuntu 16.04	gcc 4.6.3
15	Ubuntu 14.04	gcc 4.6.3
16	BLUX 4.4	gcc 5.3.0
17	BLUX 4.9	gcc 5.3.0
18	PhotonOS 2.0	gcc 5.3.0

4 Acronyms

Table 10 provides definitions for the acronyms used in this document.

Table 10 – Acronyms

Acronym	Definition
AES	Advanced Encryption Standard
AES-NI	Advanced Encryption Standard – New Instructions
AKA	Also Known As
AMD	Advanced Micro Devices
ANSI	American National Standards Institute
API	Application Programming Interface
BIOS	Basic Input/Output System
CBC	Cipher Block Chaining
CCM	Counter with CBC-MAC
CD	Compact Disc
CFB	Cipher Feedback
CMAC	Cipher-based Message Authentication Code
CMVP	Cryptographic Module Validation Program
CO	Crypto Officer
CPU	Central Processing Unit
CSE	Communication Security Establishment
CSP	Critical Security Parameter
CTR	Counter
DES	Data Encryption Standard
DRBG	Deterministic Random Bit Generator
DSA	Digital Signature Algorithm
DVD	Digital Video Disc
EC	Elliptical Curve
ECB	Electronic Code Book
ECC CDH	Elliptical Curve Cryptography Cofactor Diffie-Hellman
EC DH	Elliptical Curve Diffie-Hellman
ECDSA	Elliptical Curve Digital Signature Algorithm
EMC	Electromagnetic Compatibility
EMI	Electromagnetic Interference

FIPS	Federal Information Processing Standard
GCM	Galois/Counter Mode
GPC	General Purpose Computer
HDD	Hard Disk Drive
HMAC	(Keyed) Hash Message Authenticating Code
IG	Implementation Guidance
IT	Information Technology
KAS	Key Agreement Scheme
KAT	Known Answer Test
LCD	Liquid Crystal Display
LED	Light Emitting Diode
N/A	Not Applicable
NIST	National Institute of Standards and Technology
OFB	Output Feedback
OS	Operating System
PCI	Peripheral Component Interconnect
PCIe	Peripheral Component Interconnect Express
PRNG	Pseudo Random Number Generator
RAM	Random Access Memory
RNG	Random Number Generator
RSA	Rivest, Shamir and Adleman
SATA	Serial Advanced Technology Attachment
SCSI	Small Computer System Interface
SHA	Secure Hash Algorithm
SLES	SUSE Linux Enterprise Server
SP	Special Publication
TCBC	Triple-DES Cipher Block Chaining
TCFB	Triple-DES Cipher Feedback
TDES	Triple-Data Encryption Standard
TECB	Triple-DES Electronic Code Book
TOFB	Triple-DES Output Feedback
USB	Universal Serial Bus
XTS	XEX-based Tweaked-Codebook mode with Ciphertext Stealing