Skyhigh Security OpenSSL Module

FIPS 140-2 Non-Proprietary Security Policy

Software Version: 1.1.1v

Date: January 24th, 2023

Skyhigh Security LLC
6000 Headquarters Drive
Plano, TX 75024
888.847.8766
https://www.skyhighsecurity.com/

Table of Contents

List of Tables

List of Figures

1. Introduction

This document is the non-proprietary security policy for the Skyhigh Security OpenSSL Module, hereafter referred to as the cryptographic module, or Module.

The Module is a software library providing a C-language application program interface (API) for use by other processes that require cryptographic functionality. The Module is classified by FIPS 140-2 as a software module, multi-chip standalone module embodiment. The physical cryptographic boundary is the general purpose computer on which the Module is installed. The logical cryptographic boundary of the Module are the following sets of shared library files and their associated HMAC files for integrity checking.

OS Platform	OE #	Component	Description
Linux	1,4,6	libcrypto.so.1.1.1	Shared library for cryptographic algorithms
		libssl.so.1.1.1	Shared library for TLS/DTLS network protocols
		libcrypto.so.1.1.1.hmac	Integrity check HMAC value for libcrypto
		libssl.so.1.1.1.hmac	Integrity check HMAC value for libssl
Windows 32-bit	3	Libcrypto-1_1.dll	Shared library for cryptographic algorithms
		Libssl-1_1.dll	Shared library for TLS/DTLS network protocols
		Libcrypto-1_1.dll.hmac	Integrity check HMAC value for libcrypto
		Libssl-1_1.dll.hmac	Integrity check HMAC value for libssl
Windows 64-bit	2	libcrypto-1_1-x64.dll	Shared library for cryptographic algorithms
		libssl-1_1-x64.dll	Shared library for TLS/DTLS network protocols
		libcrypto-1_1-x64.dll.hmac	Integrity check HMAC value for libcrypto
		libssl-1_1-x64.dll.hmac	Integrity check HMAC value for libssl
Darwin	5	libcrypto.1.1.1.dylib	Shared library for cryptographic algorithms
		libssl. 1.1.1.dylib	Shared library for TLS/DTLS network protocols
		libcrypto.1.1.1.dylib.hmac	Integrity check HMAC value for libcrypto
		libssl.1.1.1.dylib.hmac	Integrity check HMAC value for libssl

Table 1 - Cryptographic Module Components

The Module performs no communications other than with the calling application (the process that invokes the Module services).

The Module is defined as multiple-chip standalone for the purposes of FIPS 140-2, and the Module executes Software Version 1.1.1v and was tested on the following operational environments and platforms detailed in the table below:

#	Operational Environment	Hardware Platform	Processor	PAA/Acceleration
1	McAfee Linux 3.8.0 on VMware ESXi 6.7.0	Intel® Taylor Pass 2U Xeon® DP Quad Board Server	Intel® Xeon® E5-2699	with PAA and without PAA
2	Windows Server 2019 H2 64-bit on VMware ESXi 6.7.0	Intel® Taylor Pass 2U Xeon® DP Quad Board Server	Intel® Xeon® E5-2699	with PAA and without PAA
3	Windows 10 Enterprise 20H2 32-bit	HP EliteBook 860 G3	Intel® Core™ i5-6300U	with PAA and without PAA
4	Ubuntu 20.04.03 LTS	Dell PowerEdge R720xd	Intel® Xeon® E5-2620	with PAA and without PAA
5	macOS 12.2.1	Apple MacBook Pro	Intel® Core™ i7-7920HQ	with PAA and without PAA
6	SUSE Linux 15 SP3 Enterprise on VMware ESXi 6.7.0	Intel® Taylor Pass 2U Xeon® DP Quad Board Server	Intel® Xeon® E5-2699	with PAA and without PAA

Table 2 - Tested Operational Environments

If applicable: The Module is also supported on the following operating environments for which operational testing and algorithm testing was not performed:

- Ubuntu 18.04 LTS
- Red Hat Enterprise Linux 7 and 8
- CentOS Linux 7 and 8
- Windows 7, 8, and 11
- macOS Big Sur

As per FIPS 140-2 Implementation Guidance G.5, compliance is maintained by vendor or user affirmation for other versions of the respective operational environments where the Module binary is unchanged.

The platforms used during testing met Federal Communications Commission (FCC) Electromagnetic Interference (EMI) and Electromagnetic Compatibility (EMC) requirements for business use as defined by 47 Code of Federal Regulations, Part 15, Subpart B. FIPS 140-2 validation compliance is maintained when the Module is operated on other versions of the operating system running in single user mode, assuming that the requirements outlined in NIST IG G.5 are met.

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

The following table lists the level of validation for each area in FIPS 140-2:

FIPS 140-2 Section Title	**Level**
1. Cryptographic Module Specification	1
2. Cryptographic Module Ports and Interfaces	1
3. Roles, Services, and Authentication	1
4. Finite State Model	1
5. Physical Security	N/A
6. Operational Environment	1
7. Cryptographic Key Management	1
8. EMI/EMC	1
9. Self ‐ Tests	1
10. Design Assurance	1
11. Mitigation of Other Attacks	N/A
Overall Level	1

Table 3 - Security Level Detail

2. Cryptographic Module Specification

2.1 Cryptographic Boundary

Figure 1 shows the logical relationship of the Module to the other software and hardware components of the computer.

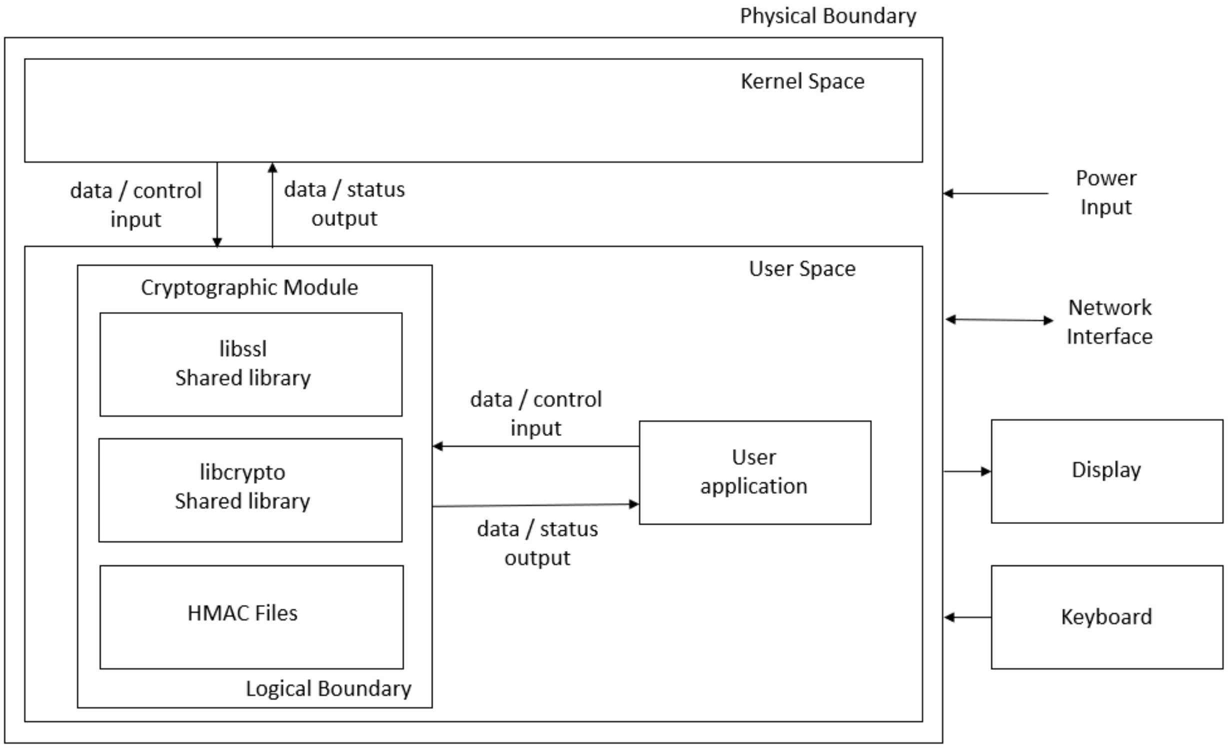

Figure 1 - Logical and Physical Boundaries

2.2 Ports and Interfaces

FIPS Interface	Logical Interface Type
Data Input	API input parameters
Data Output	API output parameters
Control Input	API function calls, API input parameters for control
Status Output	API return values
Power Input	N/A

Table 4 - Ports and Interfaces

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

2.3 Mode of Operation

The Module supports two modes of operation:

- FIPS mode (the Approved mode of operation): only approved or allowed security functions with sufficient security strength can be used.
- Non-Approved mode (the non-Approved mode of operation): when non-approved security functions are used.

The Module will be in FIPS-approved mode when all power up self-tests have completed successfully and only Approved or Allowed algorithms are invoked. See Table 5 below for a list of the supported Approved algorithms and Table 6 and 7 for allowed algorithms. The non-Approved mode is entered when a non-Approved algorithm is invoked. See 8 for a list of non-Approved algorithms.

3. Cryptographic Functionality

The Module implements the FIPS Approved, Non-Approved but Allowed, and Non-Approved cryptographic functions listed in the following tables.

3.1 Approved Cryptographic Algorithms

The approved and vendor affirmed security functions included in the Module and utilized by the Module's callable services or internal functions.

CAVP Cert	Algorithm [Standard]	Mode/Method	Description / Key Size(s) / Keys / CSPs
A2366	AES [FIPS 197] [SP800-38B]	CBC, ECB, OFB, CFB1, CFB8, CFB128 Encryption, Decryption	128/192/256 bits
A3012	AES [FIPS 197] [SP800-38B]	CTR Encryption, Decryption	128/192/256 bits
A2366	AES [FIPS 197] [SP800-38B]	CMAC	MAC length 128 bits
A2366	AES [FIPS 197] [SP800-38C]	CCM Encryption, Decryption	128/192/256 bits
A2366	AES [FIPS 197] [SP800-38D]	GCM Encryption, Decryption	128/192/256 bits
A2366	AES [FIPS 197] [SP800-38E]	XTS Encryption, Decryption	128/256 bits
A2366	AES [FIPS 197] [SP800-38F]	KW, KWP Encryption, Decryption	128/192/256 bits

A2366	DSA
	[FIPS 186-4]
		Domain Parameters Generation and Verification, Key Generation, Signature Generation	DSA keys
			L=2048, N=224
			L=2048, N=256
			L=3072, N=256

CAVP Cert	Algorithm [Standard]	Mode/Method	Description / Key Size(s) / Keys / CSPs

A2366	DSA
	[FIPS 186-4]
		Signature Verification	DSA keys
			L=1024, N=160
			L=2048, N=224
			L=2048, N=256
			L=3072, N=256
			Note: 1024 bits DSA signature verification is legacy use

A2366	RSA [FIPS 186-4] Appendix B.3.6	Key Generation	RSA keys 2048, 3072, 4096 bits
A2366	RSA [FIPS 186-4] PKCS#1 v1.5 and PSS with SHA2-224, SHA2-256, SHA2-384, SHA2-512	Signature Generation	RSA keys 2048, 3072, 4096 bits
A2366	RSA [FIPS 186-4] PKCS#1 v1.5 with SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	Signature Verification	RSA keys 1024, 2048, 3072 bits Note: 1024 bits RSA signature verification is legacy use
A3012	RSA [FIPS 186-4] PKCS#1 v1.5 with SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	Signature Verification	RSA key 4096 bits
A2366	RSA [FIPS 186-4] PKCSPSS with SHA2-224, SHA2-256, SHA2-384, SHA2-512	Signature Generation	RSA keys 2048, 3072, 4096 bits
A2366	RSA [FIPS 186-4] PKCSPSS with SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	Signature Verification	RSA keys 1024, 2048, 3072 bits Note: 1024 bits RSA signature verification is legacy use

CAVP Cert	Algorithm [Standard]	Mode/Method	Description / Key Size(s) / Keys / CSPs
A3012	RSA [FIPS 186-4] PKCSPSS with SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	Signature Verification	RSA key 4096 bits
A2366	RSA [FIPS 186-4] ANSI X9.31 with SHA2-256, SHA2-384, SHA2-512	Signature Generation	RSA keys 2048, 3072, 4096 bits
A2366	RSA [FIPS 186-4] ANSI X9.31 with SHA-1, SHA2-256, SHA2-384, SHA2-512	Signature Verification	RSA keys 1024, 2048, 3072 bits Note: 1024 bits RSA signature verification is legacy use
A3012	RSA [FIPS 186-4] ANSI X9.31 with SHA-1, SHA2-256, SHA2-384, SHA2-512	Signature Verification	RSA key 4096 bits
A2366	ECDSA [FIPS 186-4]	Key Pair Generation and Public Key Verification	ECDSA keys based on B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 curve
A2366	ECDSA with SHA2-224, SHA2-256, SHA2-384, SHA2-512 [FIPS 186-4]	Signature Generation	ECDSA keys based on B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 curve
A2366	ECDSA with SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512 [FIPS 186-4]	Signature Verification	ECDSA keys based on B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 curve

CAVP Cert	Algorithm [Standard]	Mode/Method	Description / Key Size(s) / Keys / CSPs
A2366	DRBG [SP800-90A]	CTR_DRBG (AES-128, AES-192, AES-256) Hash_DRBG, HMAC_DRBG (SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512)	CSPs: Entropy input string, seed, C, V and Key
A2366	SHS [FIPS 180-4]	SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	
A2366	SHA3 [FIPS 202]	SHA3-224, SHA3-256, SHA3-384, SHA3-512, SHAKE-128, SHAKE-256	
A2366	HMAC [FIPS 198-1]	HMAC-SHA-1, HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512	112 bits or larger HMAC Key
A2366	HMAC [FIPS 198-1]	HMAC-SHA3-224, HMAC-SHA3-256, HMAC-SHA3-384, HMAC-SHA3-512)	112 bits or larger HMAC Key
A2366	CVL KDF TLS [SP800-135rev1]	KDF TLS (TLS v1.0, v1.1 and v1.2) with SHA2-256, SHA2-384, SHA2-512	TLS Pre-Master Secret and Master Secret
A2366	TLS 1.3 KDF [RFC-8446]	TLS v1.3 KDF (HMAC Algorithm: SHA2-256, SHA2-384; KDF Running Modes: DHE, PSK, PSK-DHE)	TLS Pre-Master Secret and Master Secret
A2366	KAS-FFC-SSC [SP800-56Arev3]	Scheme: dhEphem KAS Role: initiator, responder MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192 ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192	Public key size 2048 bits or larger, and private key size 224 bits or 256 bits Shared secret: MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192; ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192 KAS-FFC-SSC: Key establishment methodology provides between 112 and 200 bits of encryption strength

CAVP Cert	Algorithm [Standard]	Mode/Method	Description / Key Size(s) / Keys / CSPs
A2366	KAS-ECC-SSC [SP800-56Arev3]	Scheme: ephemeralUnified KAS Role: initiator, responder P-224, P-256, P-384, P-521	NIST curves P-224, P-256, P-384, P-521 Shared secret: P-224, P-256, P-384, P-521 KAS-ECC-SSC: Key establishment methodology provides between 112 and 256 bits of encryption strength
A2366	KAS (ECC/FFC) KAS (KAS-SSC Cert. #A2366, CVL Cert. #A2366); [SP800-56Arev3] [SP800-135rev1]	KAS (ECC): Scheme: ephemeralUnified KAS Role: initiator, responder KAS (FFC): Scheme: dhEphem KAS Role: initiator, responder	Key Agreement Scheme per SP800-56Arev3 with key derivation function (SP800-135rev1) Note: The module's KAS (ECC/FFC) implementation is FIPS140-2 IG D.8 Scenario X1 (path 2) compliant
A2366	PBKDF [SP800-132]	SHA-1, SHA-224, SHA-256, SHA-512	PBKDF Password PBKDF Derived Key Please refer to section 8.2.3 for more details
A2366	Safe Prime Key Generation [SP800-56Arev3]	Safe Prime Groups: ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192	FFC Key-Pair Domain Parameters Generation
A2366	Safe Prime Key Verification [SP800-56Arev3]	Safe Prime Groups: ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192	FFC Key-Pair Domain Parameters Verification
A2366	CVL SSH KDF [SP800-135rev1]	KDF SSH (AES-128, AES-192, AES-256; SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512)	SSH-KDF Derived Key
A2366	KTS [SP800-38F]	AES KW, KWP (128, 192, 256 bits) AES CCM (128, 192, 256 bits) AES GCM (128, 192, 256 bits)	AES Cert. #A2366; Key establishment methodology provides between 128 and 256 bits of encryption strength

CAVP Cert	Algorithm [Standard]	Mode/Method	Description / Key Size(s) / Keys / CSPs
A2366	KTS [SP800-38F]	AES and HMAC (128, 192, 256 bits)	AES Cert. #A2366 and HMAC Cert. #A2366; Key establishment methodology provides between 128 and 256 bits of encryption strength
Vendor Affirmed	CKG [SP800-133rev2]		Key Generation SP800-133rev2, section 4 (Using the Output of a Random Bit Generator), SP800-133rev2, section 5 for Asymmetric Key Generation SP800-133rev2, section 6 for Symmetric Key Generation

Table 5 - Approved Algorithms and CAVP Certificates

Notes:

1. No parts of TLS and SSH protocols, other than the KDF, have been tested by the CAVP and CMVP.

3.2 Non-Approved Algorithms Allowed in the Approved Mode of Operation

The following cryptographic algorithm is not approved but is allowed to be used in a FIPS approved mode of operation.

Algorithm	Use / Function
RSA (encrypt, decrypt) with key size equal or larger than 2048 bits up to 16384 bits	Key wrapping; key establishment methodology provides between 112 and 256 bits of encryption strength

Table 6 - Cryptographic Algorithms Allowed in FIPS Mode

3.3 Non-Approved Algorithm Allowed in the Approved Mode of Operation with No Security Claimed

The following cryptographic algorithm is not approved but is allowed to be used in a FIPS approved mode of operation with no security claimed.

Algorithm	Use / Function
MD5	Message Digest used only in TLS

Table 7 - Cryptographic Algorithms Allowed in FIPS Mode of operation with no security claimed

3.4 Non-Approved Algorithms

The Module supports the following non-FIPS-approved and not allowed algorithms. The use of the non-conformant algorithms listed in Table will place the Module in a non-approved mode of operation.

Algorithm	Use/Function
DSA with key sizes not listed in Table 5	sign, verify, and key generation

Algorithm	Use/Function
Diffie-Hellman with key sizes not listed in Table 6	key agreement
AES-OCB	authenticated encryption/decryption
DES	encryption/decryption
KBKDF	key derivation
Single-step KDF derived key	key derivation
[RFC-3961] KDF	key derivation

Table 8 - Non-FIPS-Approved Algorithms

3.5 Cryptographic Key Management

The table below provides a complete list of Private Keys and CSPs used by the Module:

Key/CSP	Storage
AES Key	Dynamic Memory
HMAC Key	
CMAC Key	Dynamic Memory
RSA Private Key	Dynamic Memory
DSA Private Key	
ECDSA Private Key	
Diffie-Hellman Private Components	Dynamic Memory
EC Diffie-Hellman Private Components	
Shared secret	Dynamic Memory
SP 800-90A DRBG seed and entropy input	Dynamic Memory
SP 800-90A DRBG internal values (C, K, V values)	
TLS Pre-Master Secret and Master Secret	Dynamic Memory
PBKDF Password	Dynamic Memory
PBKDF Derived Key	Dynamic Memory
SSH-KDF Derived Key	Dynamic Memory

Table 9 - Private Keys and CSPs

Notes:

1. Public and private keys are provided to the Module by the calling process and are destroyed when released by the appropriate API function calls. The Module does not perform persistent storage of CSPs.
2. The key sizes under each algorithm, please refer to Table 5.

3.5.1 Key Generation

The Module implements SP 800-90A compliant DRBG services for creation of symmetric keys, and for generation of DSA, elliptic curve, and RSA keys as shown in Table 5. The module directly uses the output of the DRBG. The generated seed used in the asymmetric key generation is an unmodified output from DRBG. The calling application is responsible for storage of generated keys returned by the module. For operation in the Approved mode, Module users (the calling applications) shall use entropy sources that contain at least 112 bits of entropy.

3.5.2 Key Establishment

The Module provides Diffie-Hellman and EC Diffie-Hellman key agreement. The Module also offers AES key wrapping per [SP800-38F] (using GCM, CCM, AES-KW, AES-KWP and a combination of any approved block chaining modes with HMAC for authentication), and RSA key wrapping (encapsulation) using public key encryption and private key decryption primitives as allowed by [IG] D.9.

The module provides Diffie-Hellman shared secret computation (KAS-FFC-SSC) and EC Diffie-Hellman shared secret computation (KAS-ECC-SSC) compliant with SP800-56Arev3, in accordance with scenario X1 (path 2) of IG D.8. The module provides support for Diffie-Hellman and EC Diffie-Hellman key agreement schemes compliant with SP800-56Arev3 by offering separate services for shared secret computation and the key derivation using the SP800- 135 TLS/SSH KDF (for use within the TLS/SSH protocol), so that the user application can implement the key agreement.

AES, RSA, KAS-FFC-SSC (Diffie-Hellman) and KAS-ECC-SSC (EC Diffie-Hellman) provide the following security strengths:

- AES: key wrapping provides between 128 and 256 bits of encryption strength.
- RSA: key wrapping provides between 112 and 256 bits of encryption strength.
- Diffie-Hellman: key agreement provides between 112 and 200 bits of encryption strength.
- EC Diffie-Hellman: key agreement provides between 128 and 256 bits of encryption.

3.5.3 Key Zeroization

The application that uses the Module is responsible for appropriate destruction and zeroization of the key material. The library provides functions for key allocation and destruction, which overwrites the memory that is occupied by the key information with “zeros” before it is deallocated.

The memory occupied by keys is allocated by regular libc malloc/calloc() calls. The application is responsible for calling the appropriate destruction functions from the OpenSSL API. The destruction functions then overwrite the memory occupied by keys with pre-defined values and deallocates the memory with the free() call. In case of abnormal termination, or swap in/out of a physical memory page of a process, the keys in physical memory are overwritten before the physical memory is allocated to another process.

3.6 Public Keys

The table below provides a complete list of the Public Keys used by the Module:

Public Key Name	**Key Description and Usage**
RSA SVK	RSA (1024 to 16384 bits) signature verification public key
RSA KEK	RSA (2048 to 16384 bits) key encryption (public key transport) key
DSA SVK	[FIPS 186-4] DSA (2048/3072) signature verification key
ECDSA SVK	ECDSA (All NIST defined B, K and P curves) signature verification key
DH Public	Diffie-Hellman public key agreement key
EC DH Public	EC DH (All NIST defined B, K and P curves) public key agreement key

Table 10 - Public Keys

4. Roles, Authentication and Services

4.1 Roles and Authentication of Operators to Roles

The Module has two roles: User and Crypto Officer which are implicitly assumed upon invoking services offered by the Module. The application that is requesting services of the Module is the single user of the Module.

4.2 Services

The Approved services supported by the Module and access rights within services accessible over the Module's public interface are listed in the table below.

Service	Approved Security Functions / CSPs	Roles	Access rights to Keys and/or CSPs
Symmetric encryption/decryption	AES key	User	R, E
Asymmetric key generation	RSA, DSA and ECDSA private key	User	R, G, W, E
Symmetric key generation	AES Key, HMAC Key and CMAC Key	User	R, G, W, E
Digital signature generation and verification	RSA, DSA and ECDSA private key	User	R, G, E
TLS network protocol	AES key, HMAC key	User	R, E
TLS key agreement	AES key, RSA, DSA or ECDSA private key, HMAC Key, Premaster Secret, Master Secret, Diffie-Hellman Private Components and EC Diffie-Hellman Private Components	User	R, G, W, E
SSH network protocol	AES key, HMAC key	User	R, E
SSH key agreement	AES key, RSA, DSA or ECDSA private key, HMAC Key, SSH Exchange Hash, SSH Session ID, Diffie-Hellman Private Components and EC Diffie-Hellman Private Components	User	R, G, W, E
Shared secret computation	Diffie-Hellman shared secret and EC Diffie-Hellman shared secret	User	R
RSA key wrapping	RSA, private key	User	R, E
Certificate Management / Handling	RSA, DSA or ECDSA private key parts of certificates	User	R, W, E
Keyed Hash (HMAC)	HMAC key	User	R, E
Keyed Hash (CMAC)	CMAC key	User	R, E
Message digest (SHS)	none	User	N/A
Random number generation [800-90A] DRBG	Entropy input string and seed (C, K and V values)	User	R, G, W, E
Key Derivation (through PBKDF or SSH-KDF or Single-step KDF)	PBKDF password and derived key and SSH-KDF derived key and Single-step KDF derived key	User	R, W, E
Show status	none	User	N/A
Module initialization	none	User	N/A
Self-test	none	User	N/A
Zeroize	All CSPs	User	Z
Module installation	none	Crypto Officer	N/A

Table 11 - Approved Services, Roles and Access Rights

G or Generate: The Module generates the CSP(s)
R or Read: The CSP is read from the Module (e.g. the CSP is output)
W or Write: The CSP is updated or written to the Module
E or Execute: Capability to execute or use the Critical Security Parameter
Z or Zeroize: The Module zeroizes the CSP

The Module also provides the following non-Approved services available in non-FIPS mode:

Service	Role	Access rights to Keys and/or CSPs
Asymmetric encryption/decryption using non-Approved RSA key size	User	R, E
Symmetric encryption/decryption using non-Approved algorithms	User	R, E
Hash operation using non-Approved algorithms	User	R, E
Digital signature generation and verification using non-Approved RSA and DSA private key sizes	User	R, G, E
Digital signature generation using SHA-1	User	R, G, E
TLS connection using keys established by Diffie-Hellman with non-Approved key sizes	User	R, G, W, E
TLS connection using keys established by RSA with key size less than 2048 bits	User	R, G, W, E
Asymmetric key generation using non-Approved RSA and DSA key sizes	User	R, G, W, E
Random number generation using ANSI X9.31 RNG	User	R, G, W, E
KBKDF Key derivation [SP800-108]	User	R, E
Key derivation used in Kerberos 5 [RFC-3961]	User	R, E

Table 12 - Non-Approved Services and Roles

4.3 Authentication Methods

Authentication to a role is implicit upon entry.

5. Self-Tests

5.1 Power-Up Self Tests

The Module performs both power-up self-tests (at Module initialization) and conditional self-tests (during operation). The power-up self-test starts with the integrity test, where the integrity of the runtime executable is verified using a HMAC SHA-256 digest which is compared to a value computed at build time. If this computed HMAC SHA-256 digest matches the stored, known digest, then the remainder of the power-up self-tests (algorithm-specific pairwise consistency tests and known answer tests) are performed. Input, output, and cryptographic functions cannot be performed while the Module is in a self-test or error state because the Module is single-threaded and will not return to the calling application until the power-up self-tests are complete. After successful completion of the power-up tests, the Module is loaded and cryptographic functions are available for use. If the power-up self-tests fail the Module will enter the error state, subsequent calls to the Module will fail and no further cryptographic operations are possible. The Module performs the following power-up self-tests:

Algorithm	Test
AES (ECB, KW, XTS modes)	KAT, encryption and decryption are tested separately

Algorithm	Test
AES (CMAC, CCM, GCM modes)	KAT, authenticated encryption and decryption are tested separately.
DSA	Pairwise consistency test (PWCT), sign and verify
RSA	KAT, signature generation and verification are tested separately
ECDSA	PWCT, sign and verify
KAS-FFC-SSC (Diffie-Hellman)	Primitive "Z" Computation KAT
KAS-ECC-SSC (EC Diffie-Hellman)	Primitive "Z" Computation KAT
SP 800-90A DRBG (HASH_DRBG, HMAC_DRBG and CTR_DRBG)	KAT (Health test per section 11.3 of SP 800-90A DRBG)
HMAC-SHA-1, HMAC-SHA-224 HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512	KAT
SHA-1, SHA-256, SHA-512	KAT
SHA3-256, SHA3-512, SHAKE-128, SHAKE-256	KAT
PBKDF (SP800-132)	KAT
SP800-135rev1 KDF (SSH, TLSv1.2)	KAT
TLSv1.3 KDF	KAT
Module Software integrity	HMAC-SHA-256

Table 13 - Power-On Self Tests

5.2 Conditional Self Tests

The Module also performs the following conditional self-tests:

Algorithm	Test
DSA	Pairwise consistency test (PWCT): signature generation and verification
RSA	Pairwise consistency test (PWCT): signature generation and verification, encryption and decryption
ECDSA	Pairwise consistency test (PWCT): signature generation and verification

Table 14 - Conditional Self Tests

6. Physical Security

The Module is comprised of software only and does not claim any physical security.

7. Operational Environment

The Module operates under the operating environments specified in Table 2.

8. Guidance and Secure Operation

8.1 Crypto Officer Guidance

The Skyhigh Security OpenSSL Module is available from the Product Certifications Team. To ensure compliance with this Security Policy, the Crypto Officer must verify the HMAC values match those listed in the table below.

The Module will be in FIPS-approved mode when all power up self-tests have completed successfully and continues to operate in FIPS mode unless the User calls Algorithms listed in Table . There are no additional installation, configuration, or usage instructions for operators intending to use the Skyhigh Security OpenSSL Module.

Processor Architecture	Component	Hmac value
Linux	libcrypto.so.1.1.1	057e6d933f75b30869cfa34e95fce1e4f0f52e0f9f08dc21c1bb315b5a0bc0d7
	libssl.so.1.1.1	a4707b1dbc1ba5d6eea85e11666285a2c102035d761c511a18a867d04042fed4
Windows 32-bit	Libcrypto-1_1.dll	044d0e83d278fd1a0123816d4b1c3c15f4ed1e592c9dc74656b809eabac259f2
	Libssl-1_1.dll	20925704d707fce9392ce902509b5acb45255b667e0dcee555402c6f3ad8b698
Windows 64-bit	libcrypto-1_1-x64.dll	cf59f68b7483f63bdc64f688629ff89c8264dfb009d8bd3ce3104c34d4aad754
	libssl-1_1-x64.dll	9d351ab01c9575025d297b4b17220bd369c0766831c2d33dcd262210c5ca7bbc
Darwin	libcrypto.1.1.1.dylib	435cc9811ffb581f85d416c4311e72d3080c8ee82edbee44c4ce4d62bcf4e895
	libssl. 1.1.1.dylib	31e824733b1a9d3e473a4ddcbf772cea60d35a15761b9442222d156e7d310ecb

Table 15 – Module hmac values

8.2 User Guidance

The user should use FIPS approved services and their associated FIPS approved and FIPS allowed cryptographic algorithms as identified in this Security Policy to operate the Module in a FIPS approved mode.

8.2.1 Use of AES-XTS

According to SP 800-38E, the AES algorithm in XTS mode can be only used for the cryptographic protection of data on storage devices, as specified in SP800-38E. The length of a single data unit encrypted or decrypted with the XTS-AES shall not exceed 2^{20} AES blocks that is 16MB of data per AES-XTS instance. An XTS instance is defined in section 4 of SP800-38E.

8.2.2 AES-GCM IV

In case the module's power is lost and then restored, the key used for the AES GCM encryption or decryption shall be redistributed. The nonce_explicit part of the IV does not exhaust the maximum number of possible values for a given session key. The design of the TLS protocol in this module implicitly ensures that the nonce_explicit, or counter portion of the IV will not exhaust all of its possible values. The AES GCM IV generation is in compliance with the [RFC5288] and shall only be used for the TLS protocol version 1.2 to be compliant with [FIPS140-2_IG] IG A.5, provision 1 ("TLS protocol IV generation"); thus, the module is compliant with [SP800-52]. When a GCM IV is used for decryption, the responsibility for the IV generation lies with the party that performs the AES GCM encryption and therefore there is no restriction on the IV generation. The module supports the TLS GCM ciphersuites from SP800-52 Rev1, section 3.3.1.

8.2.3 Key derivation using SP800-132 PBKDF

The module provides password-based key derivation (PBKDF), compliant with SP800-132. The module supports option 1a from section 5.4 of [SP800-132], in which the Master Key (MK) or a segment of it is used directly as the Data Protection Key (DPK). In accordance to [SP800-132] and IG D.6, the following requirements shall be met.

- Derived keys shall only be used in storage applications. The Master Key (MK) shall not be used for other purposes. The length of the MK or DPK shall be of 112 bits or more.
- A portion of the salt, with a length of at least 128 bits, shall be generated randomly using the SP800-90A DRBG.
- The iteration count shall be selected as large as possible, as long as the time required to generate the key using the entered password is acceptable for the users. The minimum value shall be 1000.
- Passwords or passphrases, used as an input for the PBKDF, shall not be used as cryptographic keys. The length of the password or passphrase shall be of at least 20 characters, and shall consist of lower-case, upper-case and numeric characters. The probability of guessing the value is estimated to be $1/62^{20} = 10^{-36}$, which is less than 2^{-112}. The calling application shall also observe the rest of the requirements and recommendations specified in [SP800-132].

9. Mitigation of other Attacks

The module is not designed to mitigate against attacks which are outside of the scope of FIPS 140-2.

10. References and Standards

The following Standards are referred to in this Security Policy.

Abbreviation	Full Specification Name
[FIPS140-2]	*National Institute of Standards and Technology, Security Requirements for Cryptographic Modules, May 25, 2001*
[SP800-131A]	*National Institute of Standards and Technology, Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths, Special Publication 800-131Arev2, March 2019*
[IG]	*National Institute of Standards and Technology, Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program*
[SP800-38B]	*National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation: the CMAC Mode for Authentication, Special Publication 800-38B, May 2005 (updated 10/6/2016)*
[SP800-38C]	*National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation: the CCM Mode for Authentication and Confidentiality, Special Publication 800-38C, May 2004 (updated 7/20/2007)*
[SP800-38D]	*National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC, Special Publication 800- 38D, November 2007*
[SP800-38E]	*National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation: the XTS-AES Mode for Confidentiality on Storage Devices, Special Publication 800-38E, January 2010*

Abbreviation	Full Specification Name
[SP800-38F]	*National Institute of Standards and Technology, Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping, Special Publication 800-38F, December 2012*
[SP800-52]	*National Institute of Standards and Technology, Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations, Special Publication 800-52rev2, August 2019*
[SP800-56Ar3]	*National Institute of Standards and Technology, Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography, Special Publication 800-56Arev3, April 2018*
[SP800-56C]	*National Institute of Standards and Technology, Recommendation for Key-Derivation Methods in Key-Establishment Schemes, Special Publication 800-56Crev2, August 2020*
[SP800-90A]	*National Institute of Standards and Technology, Recommendation for Random Number Generation Using Deterministic Random Bit Generators, Special Publication 800-90Arev1, June 2015.*
[SP800-108]	*National Institute of Standards and Technology, Recommendation for Key Derivation Using Pseudorandom Functions (Revised), Special Publication 800-108, October 2009*
[FIPS 198-1]	*National Institute of Standards and Technology, The Keyed-Hash Message Authentication Code (HMAC), Federal Information Processing Standards Publication 198- 1, July, 2008*
[SP800-132]	*National Institute of Standards and Technology, Recommendation for Password-Based Key Derivation: Part 1: Storage Applications, Special Publication 800-132, December 2010*
[SP800-133rev2]	*National Institute of Standards and Technology, Recommendation for Cryptographic Key Generation, Special Publication 800-133rev2, June 2020*
[SP800-135rev1]	*National Institute of Standards and Technology, Recommendation for Existing Application-Specific Key Derivation Functions, Special Publication 800-135rev1, December 2011.*
[FIPS 180-4]	*National Institute of Standards and Technology, Secure Hash Standard, Federal Information Processing Standards Publication 180-4, August, 2015*
[FIPS 186-4]	*National Institute of Standards and Technology, Digital Signature Standard (DSS), Federal Information Processing Standards Publication 186-4, July, 2013.*
[FIPS 197]	*National Institute of Standards and Technology, Advanced Encryption Standard (AES), Federal Information Processing Standards Publication 197, November 26, 2001*
[FIPS 202]	*National Institute of Standards and Technology, SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions, Federal Information Processing Standards Publication 202, August, 2015*
[ANSI X9.31]	*ANSI X9.31, 1998 Edition, September 9, 1998 - DIGITAL SIGNATURES USING REVERSIBLE PUBLIC KEY CRYPTOGRAPHY FOR THE FINANCIAL SERVICES INDUSTRY (RDSA)*
[RFC-3961]	*Internet Engineering Task Force, Encryption and Checksum Specifications for Kerberos 5, RFC-3961, February 2005*
[RFC-5288]	*Internet Engineering Task Force, AES Galois Counter Mode (GCM) Cipher Suites for TLS, RFC-5288, August 2008*
[RFC-8446]	*Internet Engineering Task Force, The Transport Layer Security (TLS) Protocol Version 1.3, August 2018*

Table 16 - References

11. Acronyms and Definitions

The following Acronyms are referred to in this Security Policy.

Acronym	Definition
AES	Advanced Encryption Standard
ANSI	American National Standards Institute
API	Application Programming Interface
CAVP	Cryptographic Algorithm Validation Program
CBC	Cipher-Block Chaining
CCM	Counter with CBC-MAC
CFB	Cipher FeedBack
CMAC	Cipher-based Message Authentication Code
CMVP	Crypto Module Validation Program
CSP	Critical Security Parameter
CTR	Counter-mode
CVL	Component Validation List
DES	Data Encryption Standard
DH	Diffie-Hellman
DLC	Discrete Logarithm Cryptography
DPK	Data Protection Key
DRBG	Deterministic Random Bit Generator
DSA	Digital Signature Standard
EC	Elliptic Curve
ECB	Electronic Code Book
ECDSA	Elliptic Curve Digital Signature Algorithm
EMC	Electromagnetic Compatibility
EMI	Electromagnetic Interference
FCC	Federal Communications Commission
FIPS	Federal Information Processing Standard
GCM	Galois/Counter Mode
HMAC	key-Hashed Message Authentication Code
IG	Implementation Guidance
IV	Initialization Vector
KAS	Key Agreement Scheme
KAT	Known Answer Test
KDA	Key Derivation Algorithm
KDF	Key Derivation Function
KTS	Key Transport Scheme
KW	Key Wrap
KWP	Key Wrap with Padding
L	Length
MAC	Message Authentication Code
MD2	Message Digest algorithm MD2
MD4	Message Digest algorithm MD4
MD5	Message Digest algorithm MD5
MK	Master Key
MLOS	McAfee Linux Operating System
N	Modulus Length

Acronym	Definition
N/A	Not Applicable
NIST	National Institute of Standards and Technology
NDRNG	Non Deterministic Random Number Generator
OFB	Output FeedBack
OS	Operating System
PBKDF	Password-Based Key Derivation Function
PCT	Pairwise Consistency Test
PKCS	Public Key Cryptography Standards
RACE	Research and development in Advanced Communications technologies in Europe
RC2	Rivest Cipher RC2
RC4	Rivest Cipher RC4
RC5	Rivest Cipher RC5
RFC	Request for Comments
RIPE	RACE Integrity Primitives Evaluation
RIPEMD	RIPE Message Digest
RNG	Random Number Generator
ROM	Read-Only Memory
RSA	Public-key encryption technology developed by RSA Data Security, Inc.
SSC	Shared Secret Computation
SHA	Secure Hash Algorithms
SHAKE	Secure Hash Algorithm with KECCAK
SHS	Secure Hash Standard
SPC	Services Processing Card
SSH	Secure Shell
TLS	Transport Layer Security
XEX	XOR Encrypt XOR
XTS	XEX Tweakable block cipher with ciphertext Stealing

Table 17 - Acronyms