Amazon Linux 2 OpenSSH Client Cryptographic Module

Software Version 1.0

FIPS 140-2 Non-Proprietary Security Policy

Document Version 1.1

Last update: 2019-10-28

Prepared by:

atsec information security corporation

9130 Jollyville Road, Suite 260

Austin, TX 78759

www.atsec.com

Table of Contents

List of Tables

List of Figures

Copyrights and Trademarks

1 Introduction

This document is the non-proprietary FIPS 140-2 Security Policy for version 1.0 of the Amazon Linux 2 OpenSSH Client Cryptographic Module. It contains the security rules under which the module must be operated and describes how this module meets the requirements as specified in FIPS 140-2 (Federal Information Processing Standards Publication 140-2) for a Security Level 1 module.

1.1 Purpose of the Security Policy

There are three major reasons that a security policy is needed:

- It is required for FIPS 140 2 validation,
- It allows individuals and organizations to determine whether a cryptographic module, as implemented, satisfies the stated security policy, and
- It describes the capabilities, protection and access rights provided by the cryptographic module, allowing individuals and organizations to determine whether it will meet their security requirements.

1.2 Target Audience

This document is part of the package of documents that are submitted for FIPS 140 2 conformance validation of the module. It is intended for the following audience:

- Developers.
- FIPS 140-2 testing lab.
- The Cryptographic Module Validation Program (CMVP).
- Customers using or considering integration of Amazon Linux 2 OpenSSH Client Cryptographic Module.

2 Cryptographic Module Specification

2.1 Module Overview

The Amazon Linux 2 OpenSSH Client Cryptographic Module (hereafter referred to as the "module") is a software module implementing the Secure Shell (SSH) protocol and acts as a client daemon interacting with other entities acting as SSH server.

The module uses the Amazon Linux 2 OpenSSL Cryptographic Module as a bound module (also referred to as "the bound OpenSSL module"), which provides the underlying cryptographic algorithms necessary for establishing and maintaining SSH sessions.

2.2 FIPS 140-2 Validation Scope

Table 1 shows the security level claimed for each of the eleven sections that comprise the FIPS 140-2 standard.

Table 1: FIPS 140-2 Security Requirements.

Security Requirements Section		Level
1	Cryptographic Module Specification	1
2	Cryptographic Module Ports and Interfaces	1
3	Roles and Services and Authentication	1
4	Finite State Machine 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

2.3 Definition of the Cryptographic Module

The Amazon Linux 2 OpenSSH Client Cryptographic Module is defined as a Multi-chip Standalone software module per the requirements within FIPS 140-2. The logical cryptographic boundary of the module consists of the application, library files and their integrity test HMAC files as listed here:

- /usr/bin/ssh
- /usr/bin/fipscheck
- /usr/lib64/libfipscheck.so.1.2.1
- /usr/lib64/fipscheck/ssh.hmac
- /usr/lib64/fipscheck/fipscheck.hmac
- /usr/lib64/fipscheck/libfipscheck.so.1.2.1.hmac

The module is delivered through the Amazon Linux 2 yum core repository (ID amz2-core/2/x86_64) from the following RPMs which are need to be installed on the system together with the bound OpenSSL module:

- The OpenSSH client RPM package with version openssh-client-7.4p1-16.amzn2.0.5.x86_64
- The fipscheck RPM package with version fipscheck-1.4.1-6.amzn2.0.2.x86_64
- The fipscheck-lib RPM package with version fipscheck-lib-1.4.1-6.amzn2.0.2.x86_64
- The bound Amazon Linux 2 OpenSSL Cryptographic Module with certificate #3553

The OpenSSH client RPM package includes the binary files, integrity check HMAC files and Man Pages. Any application other than the OpenSSH client application delivered with the aforementioned OpenSSH RPM packet is not part of the Module. The FIPS certificate for this module will not be valid if any other application than the OpenSSH client application is used.

Figure 1 shows the logical block diagram of the module executing in memory on the host system. The logical cryptographic boundary is indicated with a dashed colored box.

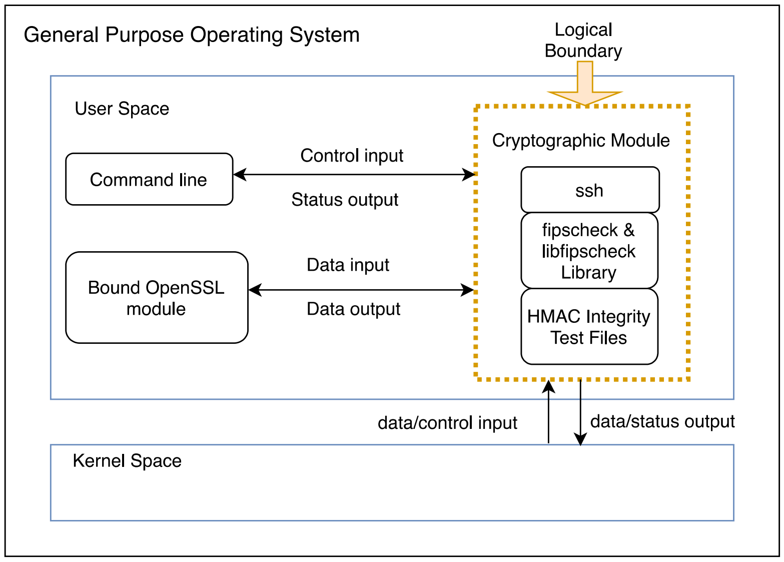

Figure 1: Logical cryptographic boundary.

2.4 Definition of the Physical Cryptographic Boundary

The physical cryptographic boundary of the module is defined as the hard enclosure of the host system on which the module runs. Figure 2 depicts the hardware block diagram. The physical hard enclosure is indicated by the dashed colored line. No components are excluded from the requirements of FIPS 140-2.

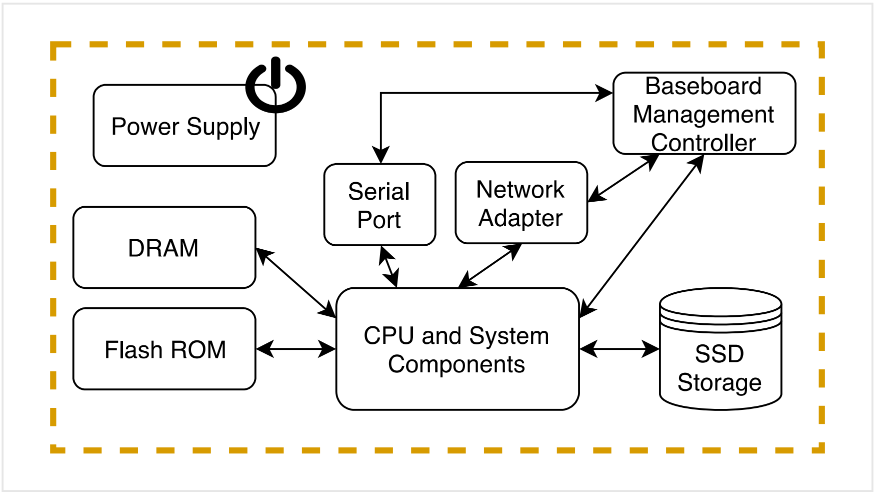

Figure 2: Hardware block diagram.

2.5 Tested Environments

The module was tested on the environments/platforms listed in Table 2. The tested operational environment is not a virtualized cloud service, and was controlled such that the laboratory had full and exclusive access to the environment and module during the testing procedures.

Table 2: Tested operational environments.

Operating System	Processor	Hardware
Amazon Linux 2	Intel Xeon E5-2686 (Broadwell) x86_64bit with PAA (i.e., AES-NI)	Amazon EC2 i3.metal 512 GiB system memory 13.6 TiB SSD storage + 8 GiB SSD boot disk 25 Gbps Elastic Network Adapter
Amazon Linux 2	Intel Xeon E5-2686 (Broadwell) x86_64bit without PAA (i.e., AES-NI)	Amazon EC2 i3.metal 512 GiB system memory 13.6 TiB SSD storage + 8 GiB SSD boot disk 25 Gbps Elastic Network Adapter

2.6 Modes of Operation

The module supports two modes of operation.

- In "**FIPS mode**" (the Approved mode of operation), only approved or allowed security functions with sufficient security strength are offered by the module.
- In "**non-FIPS mode**" (the non-Approved mode of operation), non-approved security functions are offered by the module.

The module enters the operational mode after Power-On Self-Tests (POST) succeed. Once the module is operational, the mode of operation is implicitly assumed depending on the security function invoked and the security strength[1] of the cryptographic keys chosen for the service.

If the POST fails (Section 9), the module goes into the error state. The status of the module can be determined by the availability of the module. If the module is available, then it had passed all self-tests. If the module is unavailable, it is because any self-test failed, and the module has transitioned to the error state.

Keys and Critical Security Parameters (CSPs) used or stored in FIPS mode shall not be used in non-FIPS mode, and vice versa.

[1] See Section 5.6.1 in [SP800-57] for a definition of “security strength”.

3 Module Ports and Interfaces

As a Software module, the module does not have physical ports. Thus, the physical ports within the physical boundary are interpreted to be the physical ports of the hardware platform on which the module runs and are directed through the interfaces provided by the module. Table 3 summarizes the module's interfaces.

Table 3: Ports and interfaces.

FIPS 140-2 Interface	Physical Port	Module Interfaces
Data Input	Keyboard, Ethernet port	Input parameters of the ssh command on the command line with configuration file~/.ssh/known_hosts, /etc/ssh/ssh_known_hosts, key files ~/.ssh/id_dsa, ~/.ssh/id_ecdsa*, ~/.ssh/id_rsa*, input data via SSHv2 channel, input data via local or remote port-forwarding port, input data sent to the bound OpenSSL module via its API parameters.
Data Output	Display, Ethernet Port	Output data returned by the ssh command, output data sent via the SSHv2 channel, output data sent via local or remote port-forwarding port, output data sent to the bound OpenSSL module via its API parameters.
Control Input	Keyboard, Ethernet port	Invocation of the ssh command on the command line or via the etc/ssh/ssh_config and ~./.ssh/config file, SSHv2 protocol message requests received from SSH server
Status Output	Display, Ethernet Port	Status messages returned after the command execution, status of processing SSHv2 protocol message requests
Power Input	PC power supply	N/A

4 Roles, Services and Authentication

4.1 Roles

The module supports the following roles:

- **User role**: performs services to establish, maintain, terminate and close SSH session, show status and self-tests. This role is assumed by the entity using the module.
- **Crypto Officer role**: performs module installation and configuration. This role is assumed by the entity installing the module.

The User and Crypto Officer roles are implicitly assumed depending on the service requested.

4.2 Services

Table 4 and Table 5 depict all services, which are described with more detail in the user documentation.

The tables use the following convention when specifying the access permissions that the module has for each CSP or key.

- **Create (C)**: creating a new CSP.
- **Read (R)**: reading the CSP.
- **Update (U)**: writing a new value to the CSP.
- **Zeroize (Z)**: zeroizing the CSP.
- **N/A**: No access to any CSP during its operation or there are no CSPs used in this operation.

For the “Role” column, U indicates the User role, and CO indicates the Crypto Officer role. An X marks which role has access to that service.

4.2.1 Services in the FIPS-Approved Mode of Operation

Table 4 provides a full description of FIPS Approved services and the non-Approved but Allowed services provided by the module in the FIPS-approved mode of operation and lists the roles allowed to invoke each service.

Table 4: Services in the FIPS-approved mode of operation.

Service	Service Description and Algorithms	Role		Keys and CSPs	Access Types
		U	CO		
Establish SSH Session	SSH authentication	X		RSA, DSA or ECDSA key pair	C, R, U
	Negotiate SSH key agreement	X		Diffie-Hellman or EC Diffie-Hellman key pair	
	Key derivation using SP800-135 SSH KDF	X		Shared secret, derived session encryption keys (Triple-DES or AES), and derived data authentication (HMAC) keys	
Maintain SSH Session	Provide data encryption and data authentication over SSH protocol	X		Derived session encryption keys (Triple-DES or AES), and derived data	R

Service	Service Description and Algorithms	Role		Keys and CSPs	Access Types
		U	CO		
				authentication (HMAC) keys	
Close SSH session	Zeroize SSH derived session encryption and data authentication keys by closing the SSH session	X		Derived session encryption key (Triple-DES or AES) and data authentication keys, shared secret	Z
Terminate ssh application	Zeroize SSH derived session encryption and data authentication keys by terminating the ssh application	X		Derived session encryption key (Triple-DES or AES) and data authentication keys, shared secret	Z
Self-Test	Perform on-demand self-tests	X		None	N/A
Show Status	Show status of the module	X		None	N/A
Module Installation	Install the SSH Client		X	None	N/A
Configure SSH Client	Configure the SSH Client		X	None	N/A

4.2.2 Services in the Non-FIPS-Approved Mode of Operation

Table 5 presents the services only available in non-FIPS-approved mode of operation.

Table 5: Services in the non-FIPS approved mode of operation.

Service	Service Description and Algorithms	Role		Keys and CSPs	Access Types
		U	CO		
Establish SSH Session	SSH authentication	X		RSA, DSA, ECDSA with keys sizes/curves and message digest algorithms not listed in Table 6	C, R, U
	Negotiate SSH key agreement	X		Diffie-Hellman or EC Diffie-Hellman with keys sizes/curves not listed in Table 6	C, R, U

4.3 Algorithms

The module implements the SSH KDF algorithm. The rest of the cryptographic algorithms are from the bound OpenSSL module. The cryptographic algorithms that are approved to be used in the FIPS mode of operation are tested and validated by the CAVP. No parts of the SSH protocol have been tested by the CAVP or CMVP, but for the key derivation function (KDF).

Table 6, Table 7 and Table 8 present the cryptographic algorithms in specific modes of operation. Where applicable, these tables include the CAVP certificates for different implementations, the algorithm name, respective standards, the available modes and key sizes wherein applicable, and usage. Information from certain columns may be applicable to more than one row. Please note that this module uses a subset of the cryptographic algorithms available in the bound module. As such, the tables present this subset of cryptographic algorithms.

4.3.1 FIPS-Approved

Table 6 lists the cryptographic algorithms that are approved to be used in the FIPS mode of operation.

Table 6: FIPS-approved cryptographic algorithms.

Algorithm	Standard	Mode	Key size	Use	CAVP Cert#
KDF SSH Component (CVL)	[SP800-135]	SHA-1, SHA-256, SHA-384, SHA-512	N/A	Key Derivation	#C562
Algorithms from the bound OpenSSL Module					
AES	[FIPS197] [SP800-38A]	CBC, CTR	128, 192 and 256 bits	Data Encryption and Decryption	#C523, #C524 #C525
Triple-DES	[SP800-67] [SP800-38A]	CBC	192 bits	Data Encryption and Decryption	#C523
HMAC	[FIPS198-1]	SHA-1, SHA-256, SHA-512	112 bits or greater	Message Authentication Code	#C523, #C524 #C525, #C526
SHS	[FIPS180-4]	SHA-1, SHA-256, SHA-384, SHA-512	N/A	Message Digest	#C523, #C524 #C525, #C526
DRBG	[SP800-90A]	CTR_DRBG AES-256	n/a	Random Number Generation	#C523, #C524 #C525
DSA	[FIPS186-4]	SHA-1	L=2048, N=224; L=2048, N=256; L=3072, N=256	Signature Verification	#C523
ECDSA	[FIPS186-4]	SHA-256, SHA-384, SHA-512	P-256, P-384, P-521	Signature Generation	#C523
		SHA-256, SHA-384, SHA-512	P-256, P-384, P-521	Signature Verification	#C523

Algorithm	Standard	Mode	Key size	Use	CAVP Cert#
KAS FFC Component	[SP800-56A]	FFC dhEphem scheme	2048 bits	Shared secret computation	#C523
KAS ECC Component	[SP800-56A]	ECC Ephemeral Unified scheme	P-256, P-384, P-521	Shared secret computation	#C523
RSA	[FIPS186-2]	PKCS#1v1.5 with SHA-256, SHA-512	4096 bits	Digital Signature Generation	#C523
	[FIPS186-4]	PKCS#1v1.5 with SHA-256, SHA-512	2048 and 3072 bits	Digital Signature Generation	#C523
		PKCS#1v1.5 with SHA-1, SHA-256, SHA-512	1024, 2048, and 3072 bits	Signature Verification	#C523

4.3.2 Non-Approved-but-Allowed

Table 7 lists the non-Approved-but-Allowed cryptographic algorithms provided by the bound module that are allowed to be used in the FIPS mode of operation.

Table 7: Non-Approved-but-allowed cryptographic algorithms from bound module.

Algorithm	Caveat	Usage
Diffie-Hellman with key size between 2048 bits and 8192 bits	Provides between 112 and 202 bits of encryption strength.	Key Establishment (in combination with this module; see below)
EC Diffie-Hellman with P-256, P-384, P-521 curves	Provides between 128 and 256 bits of encryption strength.	Key Establishment (in combination with this module; see below)
NDRNG	N/A	Used for seeding NIST SP 800-90A DRBG.

The OpenSSH and the bound OpenSSL module together provide the Diffie Hellman and EC Diffie Hellman key agreement. The OpenSSH module only implements the KDF portion of the key agreement and the bound OpenSSL module provides the shared secret computation.

- Diffie-Hellman with key sizes between 2048 and 8192 bits provides between 112 and 202 bits of encryption strength.
- EC Diffie-Hellman with P-256, P-384, P-521 curves provides between 128 and 256 bits of encryption strength.

4.3.3 Non-Approved

Table 8 lists the cryptographic algorithms that are not allowed to be used in the FIPS mode of operation. Use of any of these algorithms (and corresponding services in Table 5) will implicitly switch the module to the non-Approved mode.

Table 8: Non-FIPS approved cryptographic algorithms.

Algorithm	Usage
Ed25519 from EdDSA	Signature generation and verification based on Curve25519
ECDH with Curve25519	Key agreement using based on Curve25519
Algorithms from the bound OpenSSL Module	
Diffie-Hellman	Shared secret computation using 1024-bit key
EC Diffie-Hellman	Shared secret computation using curves not listed in Table 6.
DSA	Signature generation and verification with key sizes not listed in Table 6 (the module supports SHA-1 only).
RSA	Signature generation with keys sizes less than 2048 bits or using SHA-1.
ECDSA	Signature generation and verification with curves or message digest algorithms not listed in Table 6.

4.4 Operator Authentication

The module does not support operator authentication mechanisms. The role of the operator is implicitly assumed based on the service requested.

5 Physical Security

The module is comprised of software only and thus this Security Policy does not claim any physical security.

6 Operational Environment

6.1 Applicability

The module operates in a modifiable operational environment per FIPS 140-2 Security Level 1 specifications. The module runs on the Amazon Linux 2 operating system executing on the hardware specified in Section 2.5.

6.2 Policy

The operating system is restricted to a single operator mode of operation (i.e., concurrent operators are explicitly excluded by the operating system).

The entity using the SSH application is the single user of the modules, even when the application is serving multiple clients.

7 Cryptographic Key Management

Table 9 summarizes the keys and other CSPs that are used by the cryptographic services implemented in the module.

Table 9: Lifecycle of keys and other Critical Security Parameters (CSPs).

Name	Use	Generation	Entry and Output
Session Encryption key (AES, Triple-DES)	SSH session keys used for encryption and decryption.	Derived from shared secret using SP800-135 SSH KDF.	Entry: N/A. Output: via API parameter to bound OpenSSL module.
Data authentication key (HMAC)	SSH session key used for data authentication.	Derived from shared secret using SP800-135 SSH KDF.	Entry: N/A. Output: via API parameter to bound OpenSSL module.
Shared secret	Used to derive session keys.	N/A	Entered via API input parameter from bound OpenSSL module. No output.
Client RSA private key	Used to authenticate SSH client	Keys are read from the host key file.	Entry: read from Host key files. Output: via API parameter to bound OpenSSL module.
Client DSA private key	Used to authenticate SSH client		
Client ECDSA private key	Used to authenticate SSH client		
Client Diffie-Hellman private key	Used in Key agreement.	N/A (generated by the bound OpenSSL module).	Entry via API input parameter from bound OpenSSL module. Output via API input parameter to the bound OpenSSL module.
Client EC Diffie-Hellman private key	Used in Key agreement.		

7.1 Random Number Generation and Key generation

The module itself does not implement any random number generator nor does it provide key generation services. These services are provided exclusively by the bound module.

Section 7.1 of the Security Policy of the bound module is transcribed below.

"The module provides a DRBG compliant with [SP800-90A] for the creation of key components of asymmetric keys, and random number generation. The DRBG implements a Hash_DRBG, CTR_DRBG, and HMAC_DRBG mechanisms. The DRBG is initialized during module initialization and seeded from the NDRNG from /dev/urandom. The NDRNG is provided by the operational environment (i.e., Linux RNG), which is within the module's physical boundary but outside of the module's logical boundary. The NDRNG provides at least 256 bits of entropy to the DRBG.

The module performs continuous random number generator tests (CRNGT) on the output of SP800-90A DRBG to ensure that consecutive random numbers do not repeat. The operational environment, the Linux RNG, performs the continuous test on the NDRNG."

7.2 Key Establishment

The module provides key derivation through the implementation of the SP 800-135 SSH KDF.

When establishing the SSH session, the module calls the bound OpenSSL module that generates the shared secret. The module derives keys from this shared secret by applying the SP 800-135 KDF. When the module requests encryption/decryption services provided by the bound OpenSSL module, the resulting derived symmetric key will be passed to the bound OpenSSL module via API parameters.

The module provides approved key transport methods according to IG D.9 exclusively within the context of the SSH protocol. The methods are available once the SSH connection is established using the approved services of this module. The approved methods are provided with the assistance of the bound module by using a combination method, consisting of using an approved symmetric encryption mode from the bound module (e.g., AES-CTR, Triple-DES-CBC) together with an approved message authentication method from the bound module (e.g., HMAC).

Table 6 specifies the key sizes allowed in the FIPS mode of operation. According to "Table 2: Comparable strengths" in [SP800-57], the key sizes of key transport provide the following security strengths:

- Combination of approved AES encryption and HMAC message authentication key establishment methodology provides between 128 and 256 bits of encryption strength.
- Combination of approved Triple-DES encryption and HMAC message authentication key establishment methodology provides 112 bits of encryption strength.

7.3 Key Entry/Output

The module does not support manual key entry. The keys can be entered into or output from the module electronically.

7.4 Key/CSP Storage

The module does not perform persistent storage of keys. The keys and CSPs are temporarily stored as plaintext in the RAM. The client's public and private keys are stored in the host key files in ~/.ssh directory, which are outside its logical boundary.

7.5 Key/CSP Zeroization

The module performs zeroization of keys and CSPs when the module is terminated or when the SSH session is closed by invocation of the respective termination and close services. The module calls zeroization services from the bound module upon termination and closing. The memory occupied by the keys and CSPs is overwritten with zeros and the memory deallocated with the free() call. In case of abnormal termination, the keys and CSPs in physical memory are overwritten by the Linux kernel before the physical memory is allocated to another process.

8 Electromagnetic Interference/Electromagnetic Compatibility (EMI/EMC)

The test platforms listed in Table 2 have been tested and found to conform to the EMI/EMC requirements specified by 47 Code of Federal Regulations, FCC PART 15, Subpart B, Unintentional Radiators, Digital Devices, Class A (i.e., Business use). These devices are designed to provide reasonable protection against harmful interference when the devices are operated in a commercial environment.

9 Self-Tests

9.1 Power-Up Self-Tests

The module performs power-up self-tests (POSTs) automatically during initialization of the module. These POSTs ensure that the module is not corrupted. No operator intervention is necessary to run the POSTs. While the module is executing the POSTs, services are not available, and input and output are inhibited. The module is not available for use until successful completion of the POSTs.

The integrity check of the module is performed by the fipscheck application using the HMAC-SHA-256 algorithm implemented by the bound Amazon Linux 2 OpenSSL Cryptographic Module. The HMAC value is computed at build time and stored in the .hmac file. The value is recalculated at runtime and compared against the stored value.

The integrity verification is performed as follows: the OpenSSH Client application links with the library libfipscheck.so which is intended to execute fipscheck to verify the integrity of the OpenSSH client application file using the HMAC-SHA-256 algorithm. Upon calling the FIPSCHECK_verify() function provided with libfipscheck.so, fipscheck is loaded and executed, and the following steps are performed:

1. OpenSSL, loaded by fipscheck, performs the integrity check of the OpenSSL library files using the HMAC-SHA-256 algorithm.
2. fipscheck performs the integrity check of its application file using the HMAC-SHA-256 algorithm provided by the OpenSSL Module.
3. fipscheck automatically verifies the integrity of libfipscheck.so before processing requests of calling applications.
4. The fipscheck application performs the integrity check of the OpenSSH client application file. The fipscheck computes the HMAC-SHA-256 checksum of that and compares the computed value with the value stored inside the */usr/lib64/fipscheck/<application filename>.hmac* checksum file. The fipscheck application returns the appropriate exit value based on the comparison result: zero if the checksum is OK, an error code otherwise (which brings the OpenSSH Module into the error state). The libfipscheck.so library reports the result to the OpenSSH client application.

If any of the above steps fail, an error code is returned and the OpenSSH Module enters the error state with the message ‘FIPS integrity verification test failed’. In Error state, all data output is inhibited and no cryptographic operation is allowed. The module needs to be reloaded to recover from the Error state. On successful completion of the tests, the module becomes operational and crypto services are then available.

The OpenSSH module uses the bound Amazon Linux 2 OpenSSL Cryptographic Module which provides the underlying cryptographic algorithms. All the known answer tests are implemented by the bound OpenSSL Module.

9.2 On-Demand self-tests

The module provides the Self-Test service to perform self-tests on demand. On demand self-tests can be invoked by powering-off and reloading the module. This service performs the same cryptographic algorithm tests executed during power-up. During the execution of the on-demand self-tests, cryptographic services are not available and no data output or input is possible.

10 Guidance

This section provides guidance for the Crypto Officer and the User to maintain proper use of the module per FIPS 140-2 requirements.

10.1 Crypto-Officer Guidance

The RPM files containing the FIPS validated module referenced in Section 2.3 must be installed according to this guidance.

As stated in Guidance section of Amazon Linux 2 OpenSSL Cryptographic Module security policy, after configuring the operating environment to support FIPS, the file /proc/sys/crypto/fips_enabled will contain 1. If the file does not exist or does not contain “1”, the operating environment is not configured to support FIPS and the module will not operate as a FIPS validated module.

After performing the configuration described above, the Crypto Officer shall proceed for module installation with the version of the RPM package listed in Section 2.3. The integrity of the RPM is automatically verified during the installation of the modules and the Crypto Officer shall not install the RPM file if the RPM tool indicates an integrity error.

10.1.1 OpenSSH Client Configuration

For the module, the mode of operation is implicitly assumed depending on the services/security functions invoked as stated in section 4.2 and the successive sections lists the available ciphers from the module. Any use of non-approved cipher or non-Approved key size will result in the module entering the non-FIPS mode of operation. With operating environment setup as stated in the above section, the following restrictions are applicable. No more cipher addition is possible by configuration or command line options.

- SSH protocol version 1 is not allowed
- GSSAPI is not allowed
- Only the following ciphers are allowed:
    - aes128-ctr
    - aes192-ctr
    - aes256-ctr
    - aes128-cbc
    - aes192-cbc
    - aes256-cbc/rijndael-cbc@lysator.liu.se
    - 3des-cbc

Only the following message authentication codes are allowed:

- hmac-sha1/ hmac-sha1 etm@openssh.com
- hmac-sha2-256/ hmac-sha2-256 etm@openssh.com
- hmac-sha2-512/ hmac-sha2-512 etm@openssh.com

10.2 User Guidance

This module is used by connecting with a ssh server. See the ssh man page for more information. Use the ‘ssh username@hostname' command to connect to the peer ssh server. When connecting to a previously unknown server, the user will be prompted to verify a fingerprint of the server's public key. This must be done by consulting a trusted source.

10.2.1 Triple-DES Data Encryption

Data encryption using the same three-key Triple-DES key shall not exceed 2^{16} Triple-DES (64-bit) blocks, in accordance to [SP800-67] and IG A.13 in [FIPS140-2-IG]. The user of the module is responsible for ensuring the module's compliance with this requirement.

10.3 Handling Self-Test Errors

The OpenSSH self-test consists of the software integrity test. If the integrity test fails, OpenSSH enters an error state. To recover from the error state, the module must be restarted. If the failure persists, the module must be reinstalled. The bound OpenSSL module's self-tests failures will prevent OpenSSH from operating. Upon the bound module entering its error state, the OpenSSH module cannot complete the requested operation; the OpenSSH module then exits and the "error in libcrypto" message is logged . See the Guidance section in the OpenSSL Security Policy for instructions on handling OpenSSL self-test failures.

11 Mitigation of Other Attacks

The module does not mitigate against attacks.

12 Acronyms, Terms and Abbreviations

Term	Definition
AES	Advanced Encryption Standard
CAVP	Cryptographic Algorithm Validation Program
CMVP	Cryptographic Module Validation Program
CSE	Communications Security Establishment
CSP	Critical Security Parameter
DH	Diffie-Hellman
DHE	Diffie-Hellman Ephemeral
DRBG	Deterministic Random Bit Generator
ECDH	Elliptic Curve Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
HMAC	(Keyed) Hash Message Authentication Code
KAT	Known Answer Test
KDF	Key Derivation Function
NDRNG	Non-Deterministic Random Number generator
NIST	National Institute of Standards and Technology
PAA	Processor Algorithm Acceleration
POST	Power On Self Test
PR	Prediction Resistance
PSS	Probabilistic Signature Scheme
PUB	Publication
SHA	Secure Hash Algorithm
SSH	Secure Shell

13 References

The FIPS 140-2 standard, and information on the CMVP, can be found at http://csrc.nist.gov/groups/STM/cmvp/index.html. More information describing the module can be found on the vendor web site at aws.amazon.com .

This Security Policy contains non-proprietary information. All other documentation submitted for FIPS 140-2 conformance testing and validation is proprietary and is releasable only under appropriate non-disclosure agreements.

Document	Author	Title
FIPS 140-2	NIST	FIPS 140-2: Security Requirements for Cryptographic Modules
FIPS IG	NIST	Implementation Guidance for FIPS 140-2 and the Cryptographic Module Validation Program
FIPS 140-2 Annex A	NIST	FIPS 140-2 Annex A: Approved Security Functions
FIPS 140-2 Annex B	NIST	FIPS 140-2 Annex B: Approved Protection Profiles
FIPS 140-2 Annex C	NIST	FIPS 140-2 Annex C: Approved Random Number Generators
FIPS 140-2 Annex D	NIST	FIPS 140-2 Annex D: Approved Key Establishment Techniques
DTR for FIPS 140-2	NIST	Derived Test Requirements (DTR) for FIPS 140-2, Security Requirements for Cryptographic Modules
NIST SP 800-67	NIST	Recommendation for the Triple Data Encryption Algorithm TDEA Block Cipher
FIPS PUB 197	NIST	Advanced Encryption Standard
FIPS PUB 198-1	NIST	The Keyed Hash Message Authentication Code (HMAC)
FIPS PUB 186-4	NIST	Digital Signature Standard (DSS)
FIPS PUB 180-4	NIST	Secure Hash Standard (SHS)
NIST SP 800-131A	NIST	Recommendation for the Transitioning of Cryptographic Algorithms and Key Sizes
PKCS#1	RSA Laboratories	PKCS#1 v2.1: RSA Cryptographic Standard