OpenSSH Client Cryptographic Module for Perimeta SBC

Software Module Version 1.0

FIPS 140-2 Non-Proprietary Security Policy

Document Version 1.0

Date: 2020-Jul-10

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

Metaswitch Networks Ltd. reserves the right to, without notice, modify or revise all or part of this document and/or change product features or specifications under the guidance of the Cryptography Module Validation Program (CMVP), and shall not be responsible for any loss, cost, or damage, including consequential damage, caused by reliance on these materials.

1 Introduction

This document is the non-proprietary FIPS 140-2 Security Policy for version 1.0 of the OpenSSH Client Cryptographic Module for Perimeta SBC. 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 Overview

This section is informative to the reader to reference to cryptographic services of OpenSSH Client Cryptographic Module for Perimeta SBC. Only the software listed in Section 2.1 is subject to the FIPS 140-2 validation. The CMVP makes no statement as to the correct operation of the module or the security strengths of the generated keys when supported if the specific operational environment is not listed on the validation certificate.

1.2 Document Overview

This Security Policy describes the features and design of the OpenSSH Client Cryptographic Module for Perimeta SBC using the terminology contained in the FIPS 140-2 specification. FIPS 140-2, Security Requirements for Cryptographic Module specifies the security requirements that will be satisfied by a cryptographic module utilized within a security system protecting sensitive but unclassified information. The NIST/CCCS Cryptographic Module Validation Program (CMVP) validates cryptographic module to FIPS 140-2. Validated products are accepted by the Federal agencies of both the USA and Canada for the protection of sensitive or designated information.

With the exception of this Non-Proprietary Security Policy, the FIPS 140-2 Validation Documentation is proprietary to Metaswitch Networks Ltd. and is releasable only under appropriate non-disclosure agreements. For access to these documents, please contact the vendor.

1.3 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 Cryptographic Module Specification

2.1 Module Overview

The OpenSSH Client Cryptographic Module for Perimeta SBC (hereafter referred to as the “module”) is a software module implementing the Secure Shell (SSH) protocol and acts as SSH client interacting with other entities acting as an SSH server.

The module uses the OpenSSL Cryptographic Module for Perimeta SBC as a bound module (FIPS 140-2 Certificate #3657; also referred to as “the bound OpenSSL module”), which provides the underlying cryptographic algorithms necessary for establishing and maintaining SSH sessions.

2.2 Definition of the Cryptographic Module

The OpenSSH Client Cryptographic Module for Perimeta SBC 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/.ssh.hmac
- /usr/bin/fipscheck
- /usr/bin/.fipscheck.hmac
- /lib64/libfipscheck.so.1.1.0
- /lib64/.libfipscheck.so.1.1.0.hmac

The delivery, installation and configuration methods are described in Section 10. The module is provided in the following packages:

- The OpenSSH client RPM package with version: openssh-client-7.4p1-11.el6.x86_64
- The fipscheck RPM package with version: fipscheck-lib-1.2.0-7.el6.x86_64
- The fipscheck-lib RPM package with version: fipscheck-1.2.0-7.el6.x86_64

2.2.1 Definition of the Logical Cryptographic Boundary

The block diagram in Figure 1 shows the module, its interfaces with the operational environment and the delimitation of its logical boundary.

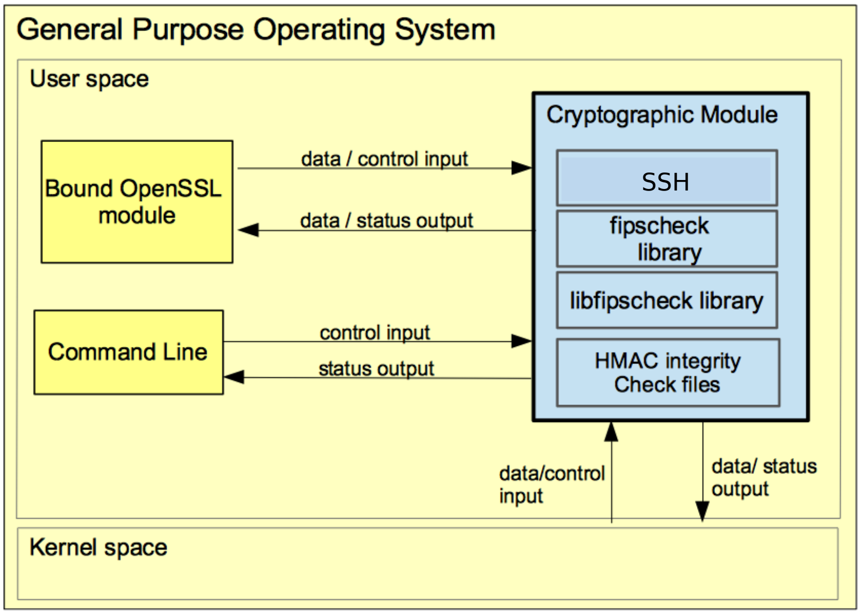

Figure 1: Software Block Diagram.

2.2.2 Definition of the Physical Cryptographic Boundary

The module is aimed to run in a general-purpose computer; the physical boundary is the surface of the case of the target platform, as shown with dotted lines in Figure 2.

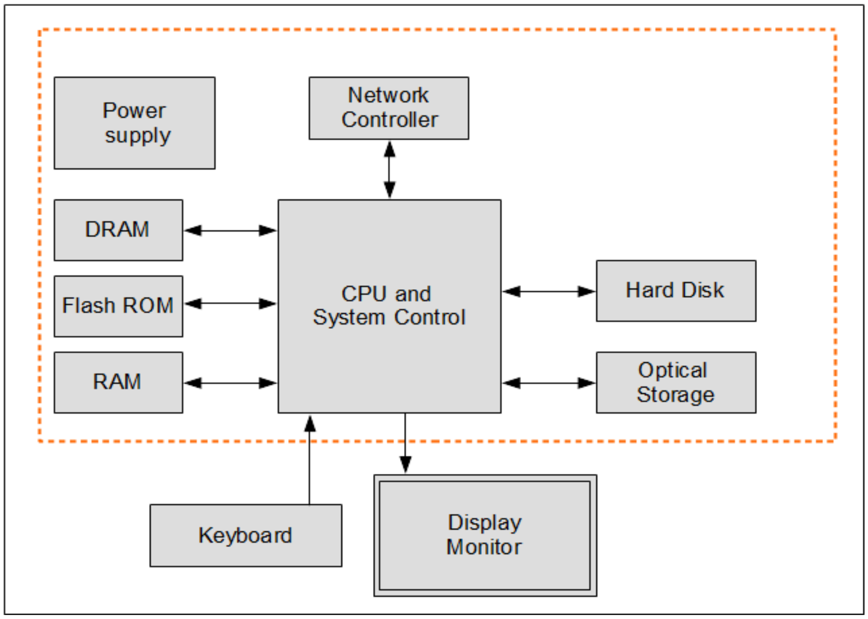

Figure 2: Hardware block diagram.

2.3 Tested Environments

Table 2 lists the platform on which the module was tested by the laboratory, and the vendor-affirmed platforms that were tested by the vendor, indicated by the corresponding sub-titles. The table brings the corresponding module variants, configuration options and whether the platform includes Processor Algorithm Acceleration (PAA), or not (i.e., with or without PAA).

Table 2: Tested Platforms.

Hardware	Processor	Operating System	Processor Algorithm Acceleration (PAA)
Tested by the Laboratory			
Dell PowerEdge R630	Intel® Xeon® E5	Metaswitch Linux 6 on VMWare Hypervisor ESXi 6.7	AES-NI
Dell PowerEdge R630	Intel® Xeon® E5	Metaswitch Linux 6 on VMWare Hypervisor ESXi 6.7	None
Vendor-Affirmed Platforms			
Dell PowerEdge R620	Intel® Xeon® E5	Metaswitch Linux 6	AES-NI
Dell PowerEdge R620	Intel® Xeon® E5	Metaswitch Linux 6	None
Dell PowerEdge R630	Intel® Xeon® E5	Metaswitch Linux 6	AES-NI
Dell PowerEdge R630	Intel® Xeon® E5	Metaswitch Linux 6	None
Dell PowerEdge R640	Intel® Xeon® Gold	Metaswitch Linux 6	AES-NI
Dell PowerEdge R640	Intel® Xeon® Gold	Metaswitch Linux 6	None
Dell PowerEdge R720	Intel® Xeon® E5	Metaswitch Linux 6	AES-NI
Dell PowerEdge R720	Intel® Xeon® E5	Metaswitch Linux 6	None
Dell PowerEdge R730	Intel® Xeon® E5	Metaswitch Linux 6	AES-NI
Dell PowerEdge R730	Intel® Xeon® E5	Metaswitch Linux 6	None
Dell PowerEdge R740	Intel® Xeon® Gold	Metaswitch Linux 6	AES-NI
Dell PowerEdge R740	Intel® Xeon® Gold	Metaswitch Linux 6	None
Dell PowerEdge R630	Intel® Xeon® E5	Metaswitch Linux 6 on OpenStack Mitaka Hypervisor	AES-NI

Hardware	Processor	Operating System	Processor Algorithm Acceleration (PAA)
Dell PowerEdge R630	Intel® Xeon® E5	Metaswitch Linux 6 on OpenStack Mitaka Hypervisor	None
Dell PowerEdge R630	Intel® Xeon® E5	Metaswitch Linux 6 on OpenStack Queens Hypervisor	AES-NI
Dell PowerEdge R630	Intel® Xeon® E5	Metaswitch Linux 6 on OpenStack Queens Hypervisor	None
Dell PowerEdge R720	Intel® Xeon® E5	Metaswitch Linux 6 on VMWare Hypervisor ESXi 6.7	AES-NI
Dell PowerEdge R720	Intel® Xeon® E5	Metaswitch Linux 6 on VMWare Hypervisor ESXi 6.7	None
AWS EC2 c4.2xlarge	Intel® Xeon® E5	Metaswitch Linux 6	AES-NI

Note: Per IG G.5, the CMVP makes no statement as to the correct operation of the module or the security strengths of the generated keys when the module is ported to the vendor affirmed platforms that are not listed on the validation certificate.

2.4 Modes of Operation

The module supports two modes of operation:

- in "FIPS mode" (the FIPS Approved mode of operation) only approved or allowed security functions with sufficient security strength can be used.
- in "non-FIPS mode" (the non-Approved mode of operation) non-approved security functions can also be used.

The module enters FIPS mode after power-up tests succeed. Once the module is operational, the mode of operation is implicitly assumed depending on the security function invoked and the security strength of the cryptographic keys.

Critical security parameters used or stored in FIPS mode are not used in non-FIPS mode, and vice versa.

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_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 configuration file /etc/ssh/ssh_config and ~/. ssh/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

4 Roles, Services and Authentication

4.1 Roles

The module supports the following roles:

- **User role**: performs services establish, maintain 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. The tables use the following convention when specifying the access permissions that the module has for each CSP or key.

- **Create (C)**: create a new CSP.
- **Read (R)**: read the CSP.
- **Update (U)**: write a new value to the CSP.
- **Zeroize (Z)**: zeroize the CSP.
- **N/A (not applicable)**: does not access any CSP or key during its 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 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 (AES, Triple-DES), and derived data authentication (HMAC) keys	
Maintain SSH Session	Provide data encryption and data authentication over SSH protocol	X		Derived session encryption keys (AES, Triple-DES), and derived data authentication (HMAC) keys	R, U
Close SSH session	Zeroize SSH derived session encryption and data	X		Derived session encryption key (AES, Triple-DES) and	Z

Service	Service Description and Algorithms	Role		Keys and CSPs	Access Types
		U	CO		
	authentication keys by closing the SSH session			data authentication keys, Shared secret	
Terminate ssh application	Zeroize SSH derived session encryption and data authentication keys by terminating the ssh application	X		Derived session encryption key (AES, Triple-DES) 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/curves not listed in Table 6 or with SHA-1, (e.g., Ed25519)	C, R, U
	Negotiate SSH key agreement	X		Diffie-Hellman or EC Diffie-Hellman with keys/curves not listed in Table 6, (e.g., Ed25519)	C, R, U
Maintain SSH Session	Provide data encryption and data authentication over SSH protocol	X		session encryption keys (RC4, Blowfish, CAST), and data authentication keys (HMAC-MD5, HMAC-MD5-96, HMAC- RIPEMD160, HMAC-SHA1-96, umac-44@opensh.com)	C, R, U

4.3 Algorithms

The module implements the SSH KDF algorithm and 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, but for the key derivation function (KDF).

Table 6, Table 7 and Table 8 present the cryptographic algorithms in specific modes of operation. 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.

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#[1]
SSH KDF	[SP800-135]	SHA-1, SHA-256, SHA-384, SHA-512	N/A	Key Derivation	#C1016
Algorithms from the bound OpenSSL Module					
AES	[FIPS197] [SP800-38A]	CBC, CTR	128, 192 and 256 bits	Data Encryption and Decryption	#C1008 #C1009 #C1010
Triple-DES	[SP800-67] [SP800-38A]	CBC	192 bits	Data Encryption and Decryption	#C1009
HMAC	[FIPS198-1]	SHA-1, SHA-256	112 bits or greater	Message Authentication Code	#C1007
		SHA-1, SHA-256, SHA-512	112 bits or greater	Message Authentication Code	#C1009 #C1010
SHS	[FIPS180-4]	SHA-1, SHA-256	N/A	Message Digest	#C1007
		SHA-1, SHA-256, SHA-384, SHA-512	N/A	Message Digest	#C1009 #C1010
DRBG	[SP800-90A]	CTR_DRBG AES256	n/a	Random Number Generation	#C1008 #C1009 #C1010
ECDSA	[FIPS186-4]	SHA-256, SHA-384, SHA-512	P-256, P-384, P-521	Signature Generation	#C1009

[1] NOTE: Only a subset of what is tested by the CAVS is used by the module.

Algorithm	Standard	Mode	Key size	Use	CAVP Cert#[1]
		SHA-256, SHA-384, SHA-512	P-256, P-384, P-521	Signature Verification	
KAS FFC Component	[SP800-56A]	FFC dhEphem scheme	2048 bits	Shared secret computation	#C1009
KAS ECC Component	[SP800-56A]	ECC Ephemeral Unified scheme	P-256, P-384, P-521	Shared secret computation	#C1009
RSA	[FIPS186-2]	PKCS#1v1.5 with SHA-256, SHA-512	4096 bits	Digital Signature Generation	#C1009
	[FIPS186-4]	PKCS#1v1.5 with SHA-256, SHA-512	2048, 3072 bits	Digital Signature Generation	
		PKCS#1v1.5 with SHA-1, SHA-256, SHA-512	1024, 2048, and 3072 bits	Signature Verification	

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 in the FIPS mode of operation.

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

Algorithm	Caveat	Use
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 Curve Ed25519	Key agreement using 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
RSA	Signature generation with keys not listed in Table 6 or SHA-1
ECDSA	Signature generation and verification with curves or message digest algorithms not listed in Table 6 (e.g., signature generation with SHA-1)
RC4	Session encryption and decryption
Blowfish	Session encryption and decryption
Cast	Session encryption and decryption
HMAC-MD5	Message authentication code
HMAC-MD5-96	Message authentication code
HMAC-RIPEMD160	Message authentication code
HMAC-SHA1-96	Message authentication code
umac-44 @opensh.com	Message authentication code

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 level 1 specifications. The module runs on a general-purpose operating system executing on the hardware specified in Table 2.

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 module does not support a maintenance role.

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 and 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 for SSH authentication	Keys are read from the key file	Entry: read from key files. Output: via API parameter to bound OpenSSL module.
Client ECDSA private key	Used for SSH authentication		
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 does not implement any random number generator, nor does it provide key generation.

7.2 Key Derivation

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

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 key files in ~/.ssh directory, which are outside its logical boundary.

7.5 Key/CSP Zeroization

The module will perform the zeroization when the SSH session is closed or the application is terminated as denoted by respective services. In case of abnormal termination, the keys 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 OpenSSL 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 OpenSSL 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 module is distributed alongside the remainder of the Perimeta software and operating system that are installed onto Metaswitch's Perimeta SBC. The software is distributed in different ways depending on the customer requirements. Customers will receive detailed instructions for acquiring and installing the secure software.

For the proper installation and configuration of the module, this guidance details two cases: (1) the full installation of a Perimeta SBC image that includes the FIPS 140-2 validated module; and (2) upgrading a previous version of the Perimeta SBC to the version that includes the FIPS 140-2 validated module. These cases are described next.

10.1.1 Full Installation of Perimeta SBC Image

This case comprises the installation of a version of Perimeta SBC that includes the validated module following the normal processes. Here, the module is already properly configured from the vendor, and there is no additional configuration required from the Crypto Officer for the install process. The Crypto Officer simply installs the appropriate image and then the installation and configuration are considered complete.

10.1.2 Upgrade Installation from another Version of Perimeta SBC

A previous version of the Perimeta SBC can be upgraded to the version that includes the FIPS 140-2 validated module. The specific upgrade instructions are provided with the Perimeta SBC package and involves first terminating the Perimeta software and decommissioning the system, and then proceeding with the upgrade.

After the upgrade is performed, the validated module is present in a packaged form within the Perimeta SBC. To use the module as the validated module with its modes of operation per the rules in Section 2.4, the FIPS capabilities of the module must be enabled as such:

1. The Crypto Officer is required to log in to the Perimeta SBC environment (using a physical console, an emulated physical console, or through SSH).
2. In the menu that is presented, the Crypto Officer selects the option to enable FIPS capabilities. Once the FIPS capabilities are enabled, the system automatically performs the following actions:
    a. Records that the FIPS capabilities are enabled and ready to use.
    b. Removes the unpacked versions of OpenSSL and OpenSSH from disk.
    c. Adds the appropriate kernel boot parameters to ensure that the FIPS capabilities are enabled upon boot.
    d. Reboots the operational environment to have the changes take effect.

After the reboot, the installation and configuration process of the module is complete.

10.1.3 OpenSSH Client Configuration

For the module, the mode of operation is implicitly assumed depending on the services/security functions invoked as stated in Section 2.4 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.

10.2 User Guidance

This module is used indirectly by users interacting with the Perimeta session border controller (SBC) through the Terminal and CLI.

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. 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.

Appendix A. Acronyms, Terms and Abbreviations

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

Appendix B. References

FIPS140-2 **FIPS PUB 140-2 - Security Requirements For Cryptographic Modules**
May 2001
http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf

FIPS140-2_IG **Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program**
August 16, 2019
http://csrc.nist.gov/groups/STM/cmvp/documents/fips140-2/FIPS1402IG.pdf

FIPS180-4 **Secure Hash Standard (SHS)**
March 2012
http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf

FIPS186-4 **Digital Signature Standard (DSS)**
July 2013
http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf

FIPS197 **Advanced Encryption Standard**
November 2001
http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

FIPS198-1 **The Keyed Hash Message Authentication Code (HMAC)**
July 2008
http://csrc.nist.gov/publications/fips/fips198-1/FIPS-198-1_final.pdf

PKCS#1 **Public Key Cryptography Standards (PKCS) #1: RSA Cryptography**
Specifications Version 2.1
February 2003
http://www.ietf.org/rfc/rfc3447.txt

SP800-38A **NIST Special Publication 800-38A - Recommendation for Block Cipher Modes of Operation Methods and Techniques**
December 2001
http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf

SP800-38B **NIST Special Publication 800-38B - Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication**
May 2005
http://csrc.nist.gov/publications/nistpubs/800-38B/SP_800-38B.pdf

SP800-38C **NIST Special Publication 800-38C - Recommendation for Block Cipher Modes of Operation: the CCM Mode for Authentication and Confidentiality**
May 2004
http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38c.pdf

SP800-38D **NIST Special Publication 800-38D - Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC**
November 2007
http://csrc.nist.gov/publications/nistpubs/800-38D/SP-800-38D.pdf

SP800-38F **NIST Special Publication 800-38F - Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping**
December 2012
http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38F.pdf

SP800-56A **NIST Special Publication 800-56A - Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography (Revised)**
March, 2007
http://csrc.nist.gov/publications/nistpubs/800-56A/SP800-56A_Revision1_Mar08-2007.pdf

SP800-57 **NIST Special Publication 800-57 Part 1 Revision 4 - Recommendation for Key Management Part 1: General**
January 2016
http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r4.pdf

SP800-67 **NIST Special Publication 800-67 Revision 1 - Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher**
January 2012
http://csrc.nist.gov/publications/nistpubs/800-67-Rev1/SP-800-67-Rev1.pdf

SP800-90A **NIST Special Publication 800-90A - Revision 1 - Recommendation for Random Number Generation Using Deterministic Random Bit Generators**
June 2015
http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf

SP800-131A **NIST Special Publication 800-131A Revision 2- Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths**
November 2015
https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-131Ar2.pdf

SP800-135 **NIST Special Publication 800-135 Revision 1 - Recommendation for Existing Application-Specific Key Derivation Functions**
December 2011
http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-135r1.pdf