Red Hat Enterprise Linux 6.6 OpenSSH Client Cryptographic Module v3.1

FIPS 140-2 Security Policy

Version 3.3

Last Update: 2016-04-13

Contents

1. Cryptographic Module Specification

This document is the non-proprietary security policy for the Red Hat Enterprise Linux 6.6 OpenSSH Client Cryptographic Module, and was prepared as part of the requirements for conformance to Federal Information Processing Standard (FIPS) 140-2, Level 1.

The following section describes the module and how it complies with the FIPS 140-2 standard in each of the required areas.

1.1. Description of Module

The Red Hat Enterprise Linux 6.6 OpenSSH Client Cryptographic Module is a software only, security level 1 cryptographic module, running on a multi-chip standalone platform. The module supplies cryptographic support the SSH protocol for the Red Hat Enterprise Linux user space. All components of the module will be provided with the software outlined below.

The following table shows the overview of the security level for each of the eleven sections of the validation.

Security Component	FIPS 140-2 Security Level
Cryptographic Module Specification	1
Cryptographic Module Ports and Interfaces	1
Roles, Services and Authentication	1
Finite State Model	1
Physical Security	N/A
Operational Environment	1
Cryptographic Key Management	1
EMI/EMC	1
Self Tests	1
Design Assurance	1
Mitigation of Other Attacks	1

Table 1. Security Level of the Module

The Module's logical cryptographic boundary are the shared library files and their integrity check HMAC files. As the Module is a software, the Module's physical cryptographic boundary is considered as the enclosure of the computer system which the Module runs on.

This cryptographic module combines a vertical stack of Linux components intended to limit the external interface each separate component may provide. The list of components and the cryptographic boundary of the composite module is defined as follows:

- Red Hat Enterprise Linux 6.6 OpenSSH Client Cryptographic Module with the version of the OpenSSH clients RPM file of 5.3p1-104.el6_6.1.
- The configuration of the FIPS mode is provided by the dracut-fips package with the version of the RPM file of 004-356.el6_6.1.
- The bound module of OpenSSL with FIPS Certificate #2441.
- The contents of the fipscheck RPM package (version 1.2.0-7.el6).

- The contents of the fipscheck-lib RPM package (version 1.2.0-7.el6).

The OpenSSH client RPM package of the Module includes the binary files, integrity check HMAC files, Man Pages and the OpenSSL Engines provided by the standard OpenSSL shared library. Any application other than the OpenSSH client application delivered with the aforementioned OpenSSH RPM packet is not part of the Module, and therefore they must not be used when operating the Module in FIPS Approved mode.

The files comprising the module are the following:

- /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
- Files of the bound module of OpenSSL as specified by the OpenSSL Security Policy

The dracut-fips RPM package is only used for configuring the operating environment to be in FIPS mode. The code in this RPM package is not active when the Module is installed and in operational state, and it does not provide any services to users interacting with the Module. Thus, the code in the dracut-fips RPM package is out of FIPS 140-2 validation scope.

The Module has been tested in the following configurations:

- 32-bit x86_64 with AES-NI
- 32-bit x86_64 without AES-NI
- 64-bit x86_64 with AES-NI
- 64-bit x86_64 without AES-NI

The Module has been tested on the following multi-chip standalone platforms:

Manufacturer	Model	O/S & Ver.
HP (Hewlett-Packard)	ProLiant DL380p Gen8	Red Hat Enterprise Linux 6.6 (single operator)
IBM (International Business Machines)	System x3500 M4	Red Hat Enterprise Linux 6.6 (single operator)

Table 2. Tested Platforms

To operate the Module, the operating system must be restricted to a single operator mode of operation. (This should not be confused with single user mode which is runlevel 1 on RHEL. This refers to processes having access to the same cryptographic instance which RHEL ensures this cannot happen by the memory management hardware.)

1.2. Description of Approved Mode

When in FIPS 140-2 approved mode, the contents of the file /proc/sys/crypto/fips_enabled will be '1'.

The Module verifies the integrity of the runtime executable using a HMAC-SHA-256 digest computed at build time. If the digests matched, the power-up self-test is then performed. If the power-up self-test is successful, the FIPS_mode flag is set to TRUE and the Module is in FIPS

Approved mode.

The Module supports the following FIPS 140-2 Approved algorithms in FIPS Approved mode:

Algorithm	**Validation Certificate**	**Usage**	**Keys/CSPs**
SP800-135 Key Derivation in SSH protocol version 2	#526, #527	Key derivation	Session encryption and data authentication keys
Counter mode where the AES cipher primitive is used from the bound OpenSSL module	N/A	Block chaining mode	N/A

Table 3. Approved Algorithms

In Approved mode the module will support the following Approved functions/protocols provided by the bound OpenSSL module – the CAVS certificate numbers for the respective ciphers are provided with the Security Policy document of the OpenSSL FIPS module.

- Triple-DES
- AES, which is implemented in software and also with AES-NI, but only one of these implementations is enabled when the module is loaded
- DRBG (SP800-90A)
- SHA-1, SHA-224, SHA-256, SHA-384, SHA-512: SHS
- HMAC-SHA-1, HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512: HMAC
- RSA
- DSA
- ECDSA
- Diffie-Hellman
- EC Diffie-Hellman

In non-Approved mode the Module supports the following non-FIPS 140-2 Approved algorithms provided by the bound OpenSSL module, which shall not be used in the FIPS Approved mode.

Algorithm	**Usage**	**Keys/CSPs**
DSA signature generation and verification	User and host authentication	DSA key 1024 bits
RSA signature generation and verification	User and host authentication	RSA key with key sizes not equal to 2048 and 3072 bits
Diffie-Hellman	Key agreement	Diffie-Hellman keys with size smaller than 2048 bits

Table 4. Non-Approved Algorithms

The module disables other non-approved functions offered by the bound OpenSSL module as specified by the OpenSSL Security Policy. Neither a user nor a crypto officer can use those non-approved functions offered by OpenSSL.

Note: The Red Hat Enterprise Linux 6.6 OpenSSH Client Cryptographic Module will use the Red Hat Enterprise Linux 6.6 OpenSSL Module (FIPS 140-2 Certificate #2441) for standard cryptographic operations and will require that a copy of a FIPS 140-2 level 1 validated version of Red Hat Enterprise Linux 6.6 OpenSSL Module be installed on the system for the Red Hat Enterprise Linux 6.6 OpenSSH Client Cryptographic Module to operate in a validated mode.

The integrity check is performed by the Red Hat Enterprise Linux OpenSSL module as used by the utility application of fipscheck using HMAC-SHA-256. The CAVS certificate numbers of the cipher implementation are provided with the Security Policy document of the OpenSSL FIPS module.

CAVEAT (obtained from the bound OpenSSL module):

1) The module generates cryptographic keys whose strengths are modified by available entropy.

1.3. Cryptographic Module Boundary

The physical module boundary is the surface of the case of the test platform. The logical module boundary is depicted in the software block diagram.

1.3.1. Hardware Block Diagram

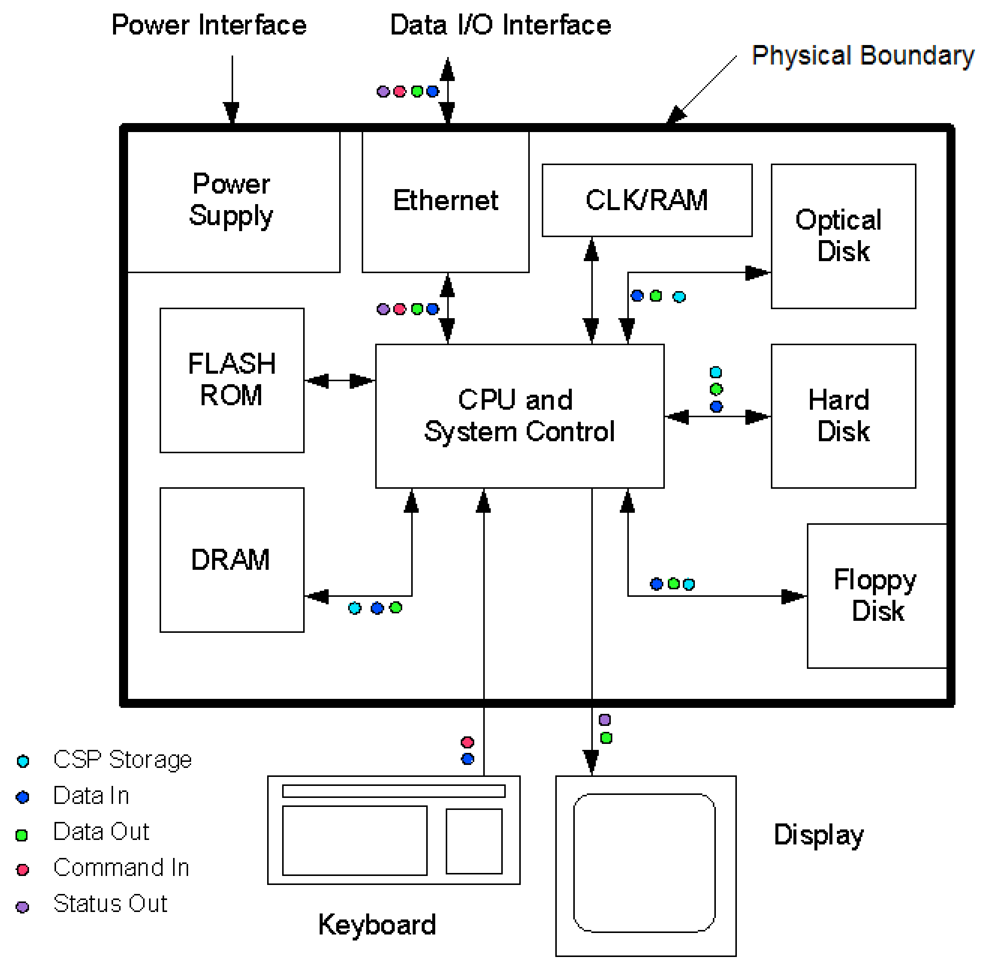

Figure 1. Hardware Block Diagram

1.3.2. Software Block Diagram

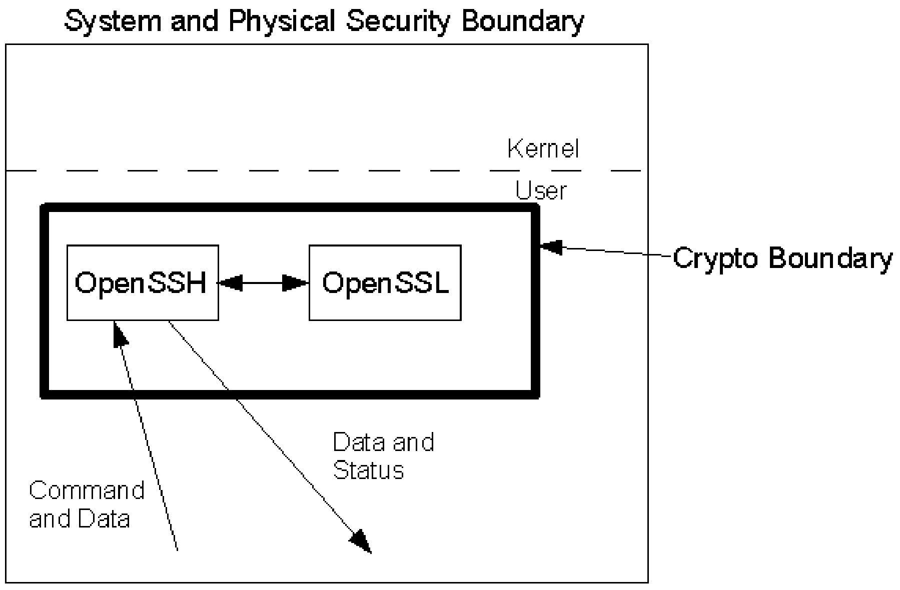

Figure 2. Software Block Diagram

1.4. Red Hat Enterprise Linux Cryptographic Modules

A set of kernel cryptographic libraries, services and user level cryptographic applications are certified at FIPS 140-2 level 1, providing a secure foundation for vendor use in developing dependent services, applications, and even purpose built appliances that may be FIPS 140-2 validated.

This section is informative to the reader to reference other cryptographic services of Red Hat Enterprise Linux. Only the software listed in section 1.2 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.

The FIPS 140-2 validation is performed at Security Level 1, on software only modules that do not make any claims about the hardware enclosure. This allows vendors to develop their own modules using these basic cryptographic modules and validate the new modules at a higher FIPS 140-2 security level.

The following cryptographic modules are included in RHEL:

- Kernel Crypto API - a software only cryptographic module that provides general-purpose cryptographic services to the remainder of the Linux kernel
- Disk Volume Encryption - provides disk management and transparent partial or full disk encryption. Partial disk encryption encrypts only one or more partitions, leaving at least one partition as plaintext.

- Libgcrypt- supplies general cryptographic support for the Red Hat Enterprise Linux user space
- OpenSSL - a software library supporting FIPS 140-2-approved cryptographic algorithms for general use by vendors
- OpenSSH-Server - supplies cryptographic support for the SSH protocol
- OpenSSH-Client - supplies cryptographic support for the SSH protocol
- Openswan - provides the IKE protocol version 1 key agreement services required for IPSec

1.4.1. Platforms

The validation was performed on a 64-bit system, which are capable of executing both 32 and 64-bit code concurrently. According to FIPS 140-2 IG G.5, the CMVP allows vendor porting and recompilation of a validated module to similar platforms without involving code changes. Although vendor may affirm the correct operation of the ported module on other platforms, the CMVP makes no statement as to the correct operation of the module or the security strengths of the generated keys when so ported.

1.4.2. FIPS Approved Mode

Any vendor-provided FIPS validated cryptographic modules and services that use the RHEL underlying services to provide cryptographic functionality must use the RHEL services in their approved mode. The approved mode ensures that FIPS required self tests are executed and that ciphers are restricted to those that have been FIPS validated by independent testing.

The kernel services (Kernel Crypto API) must be in FIPS approved mode as many services rely on these underlying services for their cryptographic capabilities. See the Kernel Crypto API Security Policy for instructions.

If dm-crypt is used to encrypt a disk or partition, particularly when other modules may store cryptographic keys or other CSPs (critical security parameters) there, it must be in FIPS approved mode as described in dm-crypt Security Policy.

If the kernel is in the approved mode, the remaining RHEL6.2 FIPS services (libgcrypt, OpenSSL, Openswan, OpenSSH) start up in the approved mode by default. (Note that Openswan uses NSS for its cryptographic operations and NSS must explicitly be put into the approved mode with the modutil command.)

The approved mode for a module becomes effective as soon as the module power on self tests complete successfully and the module loads into memory. Self tests and integrity tests triggered in RHEL at startup or on the first invocation of the crypto module:

- kernel: during boot, before dm-crypt becomes available via dracut
- user space: before the user space module is available to the caller (i.e., for libraries during the initialization call, for applications: at load time)

See each module security policy for descriptions of self tests performed by each module and descriptions of how self test errors are reported for each module.

2. Cryptographic Module Ports and Interfaces

As a software-only module, the logical interfaces are of the most importance. For the purpose of the FIPS 140-2 validation, the logical interfaces of the software-only module can be mapped to the external physical ports of the hardware platform on which it runs as shown in the table below.

Logical interfaces	Description	Physical ports of the platform on which the module runs
Command In	Invocation of the ssh command on the command line or via the configuration file ~/.ssh/config, /proc/sys/crypto/fips_enabled, environment variable SSH_USE_STRONG_RNG	Keyboard, Ethernet port
Status Out	Status messages returned after the command execution	Display, Ethernet port
Data In	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*, data via SSHv2 channel, data via local or remote port-forwarding port, data via TUN device	Keyboard, Ethernet port
Data Out	Output data returned by the ssh command	Display, Ethernet port

Table 5. Logical Interfaces and Physical Ports

3. Roles, Services, and Authentication

This section defines the roles, services, and authentication mechanisms and methods with respect to the applicable FIPS 140-2 requirements.

3.1. Roles

Role	Services (see list below)
User	Configure SSH Client Establish & Maintain SSH Session Close SSH Session (Zeroize) Terminate SSH Application Self Tests Show Status
Crypto Officer	Installation

Table 6. Roles

3.2. Services

The module supports services that are available to users in the various roles. All of the services are described in detail in the module's user documentation. The following table shows the services available to the various roles and the access to cryptographic keys and CSPs resulting from services.

The list of algorithms supported by the module is given in section 1.2.

R – The item is read or referenced by the service.

W – The item is written or updated by the service.

Z – The persistent item is zeroized by the service.

All of the ciphers are from the Red Hat Enterprise Linux 6.6 OpenSSL Module validated cryptographic module. The Red Hat Enterprise Linux 6.6 OpenSSH Client Cryptographic Module performs SSH v2 functions only, and passes all cryptographic operations to Red Hat Enterprise Linux OpenSSL Cryptographic Module.

Service	Algorithm Category	Function	Role	Keys and CSPs Accessed	Access Type
Establish & Maintain SSH Session	AES, Triple-DES, SHS, RSA, ECDSA, SP800-90A DRBG, SP800-135 Key Derivation in SSH, Diffie-Hellman and EC Diffie-Hellman	Encrypt/Decrypt, Keyed Hash, Sign/Verify, Random Number Generate, Key Derivation, Key Agreement, Zeroize of private ECDSA/RSA client keys	User	RSA or ECDSA client private key/public key Server public key Diffie-Hellman/EC Diffie-Hellman private and public components, session encryption and data authentication keys DRBG seed and nonce	RWZ RW RW RW

Service	Algorithm Category	Function	Role	Keys and CSPs Accessed	Access Type
Close SSH Session	None	Zeroize	User	Diffie-Hellman/EC Diffie-Hellman private and public components, session encryption and data authentication keys DRBG seed and nonce	Z
Terminate SSH Application	None	Zeroize	User	Diffie-Hellman/EC Diffie-Hellman private and public components, session encryption and data authentication keys DRBG seed and nonce	Z
Self-Test	HMAC	Integrity test invoked by restarting the module	User	HMAC-SHA-256 key	R
Show Status	None	Via verbose mode and exit codes	User	None	N/A
Configure SSH Client	None	None	User	None	N/A
Installation	None	None	Crypto officer	None	N/A

Table 7. Approved Services

The following table lists the non-Approved services available in non-Approved mode.

Service	Algorithm Category	Function	Role	Access Type
Establish & Maintain SSH Session	DSA, RSA and Diffie-Hellman with non-Approved key size as listed in Table 4	Sign/Verify, Key Agreement, Zeroize of private DSA/RSA client keys	User	RWZ
Close SSH Session	None	Zeorize keys with non-Approved key size	User	Z
Terminate SSH Application	None	Zeorize keys with non-Approved key size	User	Z

Table 8. Non-Approved Services

3.3. Operator Authentication

There is no operator authentication. The assumption of a role is implicit by the action taken.

3.4. Mechanism and Strength of Authentication

At security level 1, authentication is not required.

4. Physical Security

This Module is a security level 1 software module and offers no physical security.

5. Operational Environment

This module will operate in a modifiable operational environment per the FIPS 140-2 definition.

5.1. Policies

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

The operator that makes use of the cryptographic module is the single user, even when the application is serving multiple clients.

In the FIPS Approved mode, the ptrace(2) system call, the debugger (gdb(1)) and strace(1) shall not be used. In addition, other tracing mechanisms offered by the Linux environment, such as ftrace or systemtap shall not be used.

6. Cryptographic Key Management

The following table identifies the Cryptographic Keys and Critical Security Parameters (CSPs) used within the module. Cryptographic keys and CSPs are never output from the module in plaintext. An Approved key generation method is used to generate keys that are generated by the module via OpenSSL.

6.1. Key life cycle table:

Key	Type	Gen.	Establishment	Access by Service	Entry and Output method	Storage	Zero.
Client Private Keys	ECDSA or RSA keys	N/A	N/A	Establish & Maintain SSH Session	N/A	Plaintext	Immediately after use
Client Public Keys (not a CSP)	ECDSA or RSA keys	N/A	N/A	Establish & Maintain SSH Session	Exported	Plaintext	N/A
Server Public Keys (not a CSP)	ECDSA or RSA key	N/A	N/A	Establish & Maintain SSH Session	Imported	Plaintext	N/A
Session Data Authentication Keys	HMAC-SHA-1	N/A	Established during the SSH handshake through DH	Establish & Maintain SSH Session	N/A	Ephemeral	Close SSH Session or Terminate SSH Application
Session Encryption Keys	AES or Triple-DES	N/A	Established during the SSH handshake through DH	Establish & Maintain SSH Session	N/A	Ephemeral	Close SSH Session or Terminate SSH Application
Software Integrity Key	HMAC-SHA-256	N/A	N/A	Self Tests	N/A	Plaintext within the OpenSSL and fipscheck libraries	Close SSH Session or Terminate SSH Application
Diffie-Hellman Private and Public Components	Diffie-Hellman	DRBG	N/A	Establish & Maintain SSH Session	N/A	Ephemeral	Close SSH Session or Terminate SSH Application
EC Diffie-Hellman Private and Public Components	EC Diffie-Hellman	DRBG	N/A	Establish & Maintain SSH Session	N/A	Ephemeral	Close SSH Session or Terminate SSH Application
DRBG SP800-90A seed	Depending on DRBG type: up to 256-bit	N/A	N/A	Establish & Maintain SSH Session	N/A (see section 6.3), provided by /dev/urandom	Ephemeral	N/A (Termination of the SSH application where OpenSSL zeroizes seed)

Key	Type	Gen.	Establishment	Access by Service	Entry and Output method	Storage	Zero.
DRBG SP800-90A nonce	Depending on DRBG type: up to 128-bit	N/A	N/A	Establish & Maintain SSH Session	N/A (see section 6.3), provided by /dev/urandom	Ephemeral	N/A (Termination of the SSH application where OpenSSL zeroizes nonce)

Table 9. Key Life Cycle

Notes:

The module ships without containing any keys and CSPs. When the module is configured, the crypto officer generates an ECDSA or RSA private/public key pair that is stored in plaintext form in keystore files in the filesystem.

A crypto officer or user adds server public keys to the ~/.ssh/known_hosts file using the ssh-keyscan utility or by manually adding the public keys. If the ssh_keyscan utility is used, the crypto officer or user must verify the keys are correct to prevent a man-in-the-middle attack.

The public key is associated with the correct entity as each key is associated with its relevant hostname, or IPv4 address as a lookup index. Moreover, using an incorrect key results in failure to establish a valid session as the corresponding private key cannot correctly authenticate against an incorrect public key.

The only key management operations during initial configuration include generating the client public-private key pair and storing client public keys in the ~/.ssh/, which are out of scope for this validation. At runtime, public keys may be added, removed, or updated from the ~/.ssh/known_hosts file or a new client public-private key pair may be generated and deployed as needed.

Diffie-Hellman key agreement is invoked at the beginning of a session as well as after each 1 GB of data transfers or after 1 hour of operation through that session, whichever occurs first.

Persistently stored secret and private keys are out of scope, but may be zeroized using the a FIPS140-2 approved mechanism to clear data on hard disks.

6.2. Key Zeroization

For volatile memory, memset is included in deallocation operations. There are no restrictions when zeroizing any cryptographic keys and CSPs.

6.3. Random Number Generation

The module uses an FIPS-Approved, SP800-90A compliant Deterministic Random Bit Generator (known as DRBG 800-90A). It is provided by the OpenSSL cryptographic module, to which this OpenSSH module is bound to. It is seeded by the Linux kernel using either /dev/urandom or a seed that is a combination of /dev/random and /dev/urandom.

When the environment variable of SSH_USE_STRONG_RNG is used, the seed source of /dev/random is used in addition to /dev/urandom. This environment variable must have a positive integer greater than or equal to 6 to be honored. That integer value specifies the number of bytes obtained from /dev/random and mixed into the DRBG state via the OpenSSL RNG RAND_add API call. This variable can be set in /etc/sysconfig/sshd as this file is sourced by the sshd start script.

For the details of DRBG 800-90A implementation in the OpenSSL module and its seed sources, please refer to the Red Hat Enterprise Linux 6.6 OpenSSL Module v3.0 FIPS 140-2 Security Policy, Section 6.1, “Random Number Generation.”

The kernel performs a continuous self tests on the random numbers it uses to ensure that the seed and seed key input to the Approved RNG do not have the same value. The kernel also performs continual tests on the output of the approved RNG to ensure that consecutive random numbers do not repeat.

7. Electromagnetic Interference/Electromagnetic Compatibility (EMI/EMC)

Product Name and Model: HP ProLiant DL380p Gen8

Regulatory Model Number: HSTNS-5163

Product Options: All

EMC: Class A

Product Name and Model: IBM System x3500 M4

Regulatory Model Number: 7383-AC1

Product Options: All

EMC: Class A

8. Self Tests

FIPS 140-2 requires that the module perform self tests to ensure the integrity of the module and the correctness of the cryptographic functionality at startup. In addition, some functions require continuous verification of function, such as the random number generator. All of these tests are listed and described in this section.

8.1. Power-Up Tests

No operator intervention is required during the running of the self tests.

See section 9.3 for descriptions of possible self test errors and recovery procedures.

The power-up self tests of the cryptographic algorithms are entirely performed by the bound OpenSSL FIPS module.

8.1.1. Software Integrity Test Details

The OpenSSH userspace module has its integrity verified at startup by the software integrity test.

The integrity check is performed by the application of fipscheck using the HMAC-SHA-256 cipher implemented by the bound Red Hat Enterprise Linux 6.6 OpenSSL Module.

When the module starts, it exercises the power-on self test, including the software integrity test. The software integrity test (HMAC-SHA-256) constitutes a known answer test for the HMAC-SHA-256 algorithm.

The user space 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 HMAC-SHA-256. Upon calling the FIPSCHECK_verify() function provided with libfipscheck.so, fipscheck is loaded and executed, and the following steps are performed:

1. OpenSSL, as loaded by fipscheck, performs the integrity check of the OpenSSL library files using HMAC-SHA256.
2. fipscheck performs the integrity check of its application file using HMAC-SHA-256 provided by OpenSSL.
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 /path/to/application/.<applicationfilename>.hmac checksum file. The fipscheck application returns the appropriate exit value based on the comparison result: zero if the checksum is OK. The libfipscheck.so library reports the result to the OpenSSH client application.

9. Guidance

Password-based encryption and password-based key generation do not provide sufficient strength to satisfy FIPS 140-2 requirements. As a result, data processed with password-based encryption methods are considered to be unprotected.

9.1. Crypto Officer Guidance

The version of the RPM containing the validated module is stated in section 1 above. The integrity of the RPM is automatically verified during the installation and the crypto officer shall not install the RPM file if the RPM tool indicates an integrity error. In addition, the OpenSSL FIPS 140-2 module referenced in section 1 must be installed according to its Security Policy.

The RPM package of the module can be installed by standard tools recommended for the installation of RPM packages on a Red Hat Enterprise Linux system (for example, yum, rpm, and the RHN remote management tool).

For proper operation of the in-module integrity verification, the prelink has to be disabled. This can be done by setting PRELINKING=no in the /etc/sysconfig/prelink configuration file. Existing prelinking, if any, should be undone on all the system files using the 'prelink -u -a' command.

To bring the module into FIPS approved mode, perform the following:

1. Install the dracut-fips package:

   ```
   # yum install dracut-fips
   ```

2. Recreate the INITRAMFS image:

   ```
   # dracut -f
   ```

After regenerating the initrd, the crypto officer has to append the following string to the kernel command line by changing the setting in the boot loader:

```
fips=1
```

If /boot or /boot/efi resides on a separate partition, the kernel parameter boot=<partition of /boot or /boot/efi> must be supplied. The partition can be identified with the command "df /boot" or "df /boot/efi" respectively. For example:

```
$ df /boot
Filesystem        1K-blocks   Used        Available         Use%  Mounted on
/dev/sda1          233191     30454       190296            14%   /boot
```

The partition of /boot is located on /dev/sda1 in this example. Therefore, the following string needs to be appended to the kernel command line:

```
"boot=/dev/sda1"
```

Reboot to apply these settings.

A client public/private key pair that can be used for non-interactive authentication of a user to a server can be generated using 'ssh-keygen'. See the ssh-keygen(1) man page for documentation and setup instructions.

The OpenSSH client maintains a list of known servers and their public keys, globally in /etc/ssh/ssh_known_hosts and per-user in ~/.ssh/known_hosts, which the crypto officer can install keys of known servers from. If any of these files is pre-populated using ssh-keyscan, the list of known public keys must be verified against a trusted source.

9.1.1. Configuration Changes and FIPS Approved Mode

Use care whenever making configuration changes that could potentially prevent access to the /proc/sys/crypto/fips_enabled flag (fips=1) in the file/proc. If the module does not detect this flag during initialization, it does not enable the FIPS approved mode. For example, executing the module inside a chroot without the proc file system available, would instantiate the module in non-approved mode.

All user space modules depend on this file for transitioning into FIPS approved mode.

9.1.2. OpenSSH and NSS

The OpenSSH client is able to store the RSA and ECDSA host / user keys when using the UseNSS configuration option. When NSS is used to manage RSA / DSA keys, the signature generation and verification of the host or user keys is performed by NSS as well.

The module of OpenSSH is defined to be bound to OpenSSL only as outlined in chapter 1. The crypto officer can only use the UseNSS configuration option when the FIPS 140-2 validated NSS library is installed and used by OpenSSH.

9.1.3. OpenSSH Configuration

The user must not use DSA keys for performing key-based authentication as OpenSSH only allows DSA keys with 1024 bit size which are disallowed as per SP800-131A.

The user must not accept DSA host keys potentially offered during the first contact of an SSH server as OpenSSH only allows DSA keys with 1024 bit size which are disallowed as per SP800-131A.

When re-generating RSA host keys, the crypto officer should generate RSA keys with a size of 2048 bit or 3072 bit according to SP800-131A. The crypto office should inform the user base to not use RSA keys with key sizes other than 2048 bits or 3072 bits.

In FIPS 140-2 mode, the module enforces the following restrictions. When these restrictions are violated by configuration options or command line options, the module will not be operational:

- SSH protocol version 1 is not allowed
- Diffie-Hellman keys of less than 2048 bits are not allowed.
- GSSAPI is not allowed
- Only the following ciphers are allowed:
    - aes128-ctr
    - aes192-ctr
    - aes256-ctr
    - aes128-cbc
    - aes192-cbc
    - aes256-cbc
    - 3des-cbc
    - rijndael-cbc@lysator.liu.se
- Only the following message authentication codes are allowed:
    - hmac-sha1
    - hmac-sha2-256
    - hmac-sha2-512

9.2. User Guidance

See the ssh(1) man page for general usage documentation of the ssh client.

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.

9.3. Handling Self Test Errors

OpenSSL self test failures may prevent OpenSSH from operating. See the Guidance section in the OpenSSL Security Policy for instructions on handling OpenSSL self test failures.

The OpenSSH self test consists of the software integrity test. If the integrity test fails, OpenSSH enters an error state. The only recovery from this type of failure is to reinstall the OpenSSH module. If you downloaded the software, verify the package hash to confirm a proper download.

10. Mitigation of Other Attacks

RSA is vulnerable to timing attacks. In a setup where attackers can measure the time of RSA decryption or signature operations, blinding must be used to protect the RSA operation from that attack.

The API function of RSA_blinding_on turns blinding on for key rsa and generates a random blinding factor. The random number generator must be seeded prior to calling RSA_blinding_on.

Weak Triple-DES keys are detected as follows:

```
/* Weak and semi week keys as taken from
 * %A D.W. Davies
 * %A W.L. Price
 * %T Security for Computer Networks
 * %I John Wiley & Sons
 * %D 1984
 * Many thanks to smb@ulysses.att.com (Steven Bellovin) for the reference
 * (and actual cblock values).
 */
#define NUM_WEAK_KEY    16
static const DES_cblock weak_keys[NUM_WEAK_KEY]={
        /* weak keys */
        {0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01},
        {0xFE,0xFE,0xFE,0xFE,0xFE,0xFE,0xFE,0xFE},
        {0x1F,0x1F,0x1F,0x1F,0x0E,0x0E,0x0E,0x0E},
        {0xE0,0xE0,0xE0,0xE0,0xF1,0xF1,0xF1,0xF1},
        /* semi-weak keys */
        {0x01,0xFE,0x01,0xFE,0x01,0xFE,0x01,0xFE},
        {0xFE,0x01,0xFE,0x01,0xFE,0x01,0xFE,0x01},
        {0x1F,0xE0,0x1F,0xE0,0x0E,0xF1,0x0E,0xF1},
        {0xE0,0x1F,0xE0,0x1F,0xF1,0x0E,0xF1,0x0E},
        {0x01,0xE0,0x01,0xE0,0x01,0xF1,0x01,0xF1},
        {0xE0,0x01,0xE0,0x01,0xF1,0x01,0xF1,0x01},
        {0x1F,0xFE,0x1F,0xFE,0x0E,0xFE,0x0E,0xFE},
        {0xFE,0x1F,0xFE,0x1F,0xFE,0x0E,0xFE,0x0E},
        {0x01,0x1F,0x01,0x1F,0x01,0x0E,0x01,0x0E},
        {0x1F,0x01,0x1F,0x01,0x0E,0x01,0x0E,0x01},
        {0xE0,0xFE,0xE0,0xFE,0xF1,0xFE,0xF1,0xFE},
        {0xFE,0xE0,0xFE,0xE0,0xFE,0xF1,0xFE,0xF1}};
```

Please note that there is no weak key detection by default. The caller can explicitly set the DES_check_key to 1 or call DES_check_key_parity() and/or DES_is_weak_key() functions on its own.

11. Glossary and Abbreviations

AES	Advanced Encryption Specification
CAVP	Cryptographic Algorithm Validation Program
CBC	Cipher Block Chaining
CCM	Counter with Cipher Block Chaining-Message Authentication Code
CFB	Cipher Feedback
CMT	Cryptographic Module Testing
CMVP	Cryptographic Module Validation Program
CSP	Critical Security Parameter
CVT	Component Verification Testing
DES	Data Encryption Standard
DH	Diffie-Hellman
DRBG	Deterministic Random Bit Generator (this document refers to the DRBG types defined in SP800-90A)
DSA	Digital Signature Algorithm
ECB	Electronic Code Book
FSM	Finite State Model
HMAC	Hash Message Authentication Code
LDAP	Lightweight Directory Application Protocol
MAC	Message Authentication Code
NIST	National Institute of Science and Technology
NVLAP	National Voluntary Laboratory Accreditation Program
OFB	Output Feedback
O/S	Operating System
RNG	Random Number Generator
RSA	Rivest, Shamir, Addleman
SAP	Service Access Points
SDK	Software Development Kit
SHA	Secure Hash Algorithm
SHS	Secure Hash Standard
SOF	Strength of Function
SSH	Secure Shell
UI	User Interface

Table 10. Abbreviations

12. References

[1] OpenSSH man pages regarding ssh(8) and ssh_config(5)

[2] FIPS 140-2 Standard, http://csrc.nist.gov/groups/STM/cmvp/standards.html

[3] FIPS 140-2 Implementation Guidance, http://csrc.nist.gov/groups/STM/cmvp/standards.html

[4] FIPS 140-2 Derived Test Requirements, http://csrc.nist.gov/groups/STM/cmvp/standards.html

[5] FIPS 197 Advanced Encryption Standard, http://csrc.nist.gov/publications/PubsFIPS.html

[6] FIPS 180-4 Secure Hash Standard, http://csrc.nist.gov/publications/PubsFIPS.html

[7] FIPS 198-1 The Keyed-Hash Message Authentication Code (HMAC), http://csrc.nist.gov/publications/PubsFIPS.html

[8] FIPS 186-4 Digital Signature Standard (DSS), http://csrc.nist.gov/publications/PubsFIPS.html

[9] ANSI X9.52:1998 Triple Data Encryption Algorithm Modes of Operation, http://webstore.ansi.org/FindStandards.aspx?Action=displaydept&DeptID=80&Acro=X9&DpName=X9,%20Inc.