Red Hat Enterprise Linux 6.2 OpenSSH Client Cryptographic Module v2.0

FIPS 140-2 Security Policy

Version 1.3

Last Update: 2012-08-14

Contents

1 Cryptographic Module Specification

This document is the non-proprietary security policy for the OpenSSH Client FIPS Object 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 OpenSSH Client 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. The RPM version for the validated module is 5.3p1-70.el6_2.2.

All cryptographic operations and the module integrity check are performed by the Red Hat Enterprise Linux OpenSSL Cryptographic Module for the OpenSSH module. The files that make up the module are:

For x86_64

/usr/sbin/.ssh.hmac
/usr/sbin/ssh
/usr/share/man/man1/ssh.1.gz

/usr/lib64/.libcrypto.so.1.0.0.hmac
/usr/lib64/.libcrypto.so.10.hmac
/usr/lib64/libcrypto.so.1.0.0
/usr/lib64/libcrypto.so.10

/lib64/.libfipscheck.so.1.1.0.hmac
/lib64/.libfipscheck.so.1.hmac
/lib64/libfipscheck.so.1.1.0
/lib64/libfipscheck.so.1

/usr/bin/.fipscheck.hmac
/usr/bin/fipscheck

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

Security Component	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

Security Component	Security Level
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 has been tested on the following multi-chip standalone platforms:

Manufacturer	Model	O/S & Ver.
HP	ProLiant DL585	Red Hat Enterprise Linux 6.2 (Single User Mode)
IBM	HS22	Red Hat Enterprise Linux 6.2 (Single User Mode)

Table 2. Tested Platforms

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'.

In Approved mode, the module will support the following Approved and allowed functions/protocols:

- Triple-DES (Certs. #1226, #1227, #1231 and #1232)
- AES (Certs. #1887, #1888, #1889, #1893, #1894 and #1895), which is implemented in software and also with AES-NI, but only one of these implementations is enabled when the module is loaded
- DSA (Certs. #592, #593, #597 and #598)
- RNG (ANSI X9.31) (Certs. #989, #990, #994 and #995)
- SHA-1, SHA-224, SHA-256, SHA-384, SHA-512: SHS (Certs. #1658, #1659, #1663, #1664)
- HMAC-SHA-1, HMAC-SHA224, HMAC-SHA256, HMAC-SHA384, HMAC-SHA512: HMAC (Certs. #1129, #1130, #1134, #1135)
- RSA (Certs. #964, #965, #969 and #970)

In Approved mode the module will support the following Non-Approved functions:

- RSA (encrypt, decrypt) (*see caveat below*)
- Diffie-Hellman (key agreement; key establishment methodology) (*see caveat below*)

Note: The Red Hat Enterprise Linux 6.2 OpenSSH Client Cryptographic Module will use the Red Hat Enterprise Linux OpenSSL Cryptographic Module (FIPS 140-2 Validation #1758) for standard cryptographic operations and will require that a copy of a FIPS 140-2 level 1 validated version of Red Hat Enterprise Linux OpenSSL Cryptographic Module be installed on the system for the Red Hat Enterprise Linux 6.2 OpenSSH Client Cryptographic Module to operate in a validated mode.

The Red Hat Enterprise Linux 6.2 OpenSSH Client Cryptographic Module itself implements the SSHv2 protocol. The module should be used with SSHv2 protocol only, as SSHv1 protocol is not supported. The SSHv2 protocol allows the user to specify the HMAC protecting the connection where the module uses HMAC-SHA1. However, the protocol allows specifying the full HMAC SHA1 size of 160 bit as well as using the HMAC-SHA1 which is truncated to the first 96 bits.

The integrity check is performed by the Red Hat Enterprise Linux OpenSSL module utility fipscheck using HMAC/SHA256 (Certs. #1129, #1130, #1134 and #1135). The version is 1.2.0-7.el6.

CAVEAT:

The Module will support the following non-approved functions:

1) RSA (key wrapping; key establishment methodology provides between 80 and 160 bits of encryption strength)

2)Diffie-Hellman (key agreement; key establishment methodology provides between 80 and 160bits of encryption strength)

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 and is embodied by the SSH client application found at /usr/bin/ssh and the OpenSSL shared library module.

Note that during load time, libfipscheck and the fipscheck binary are used.

1.3.1 Hardware Block Diagram

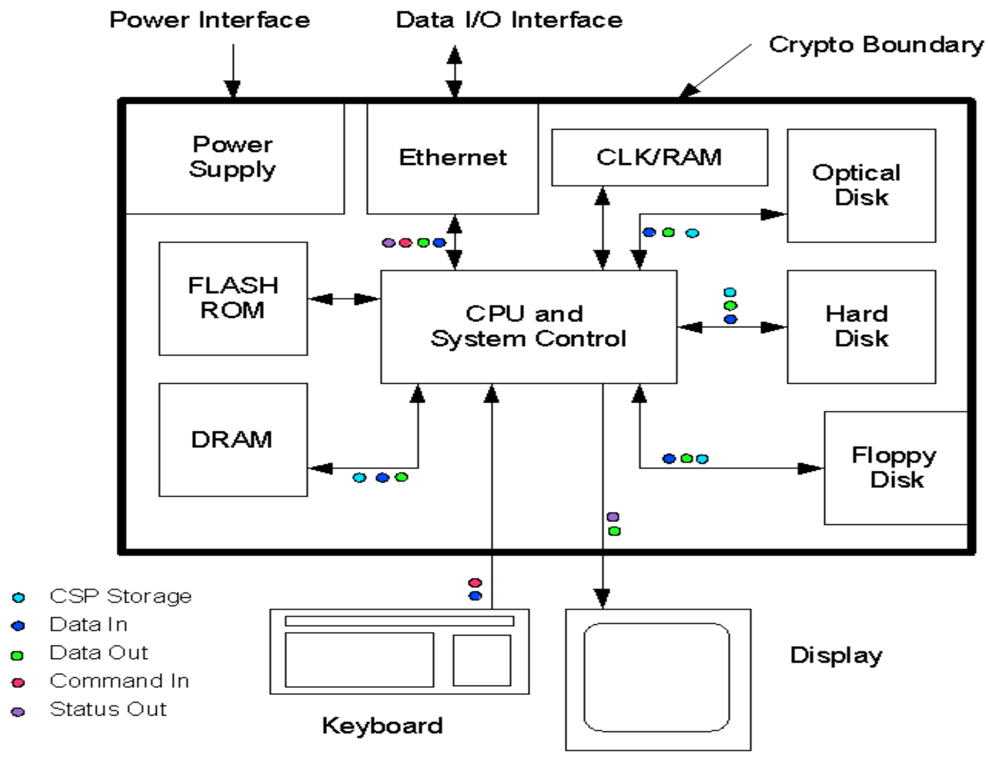

Figure 1. Hardware Block Diagram

1.3.2 Software Block Diagram

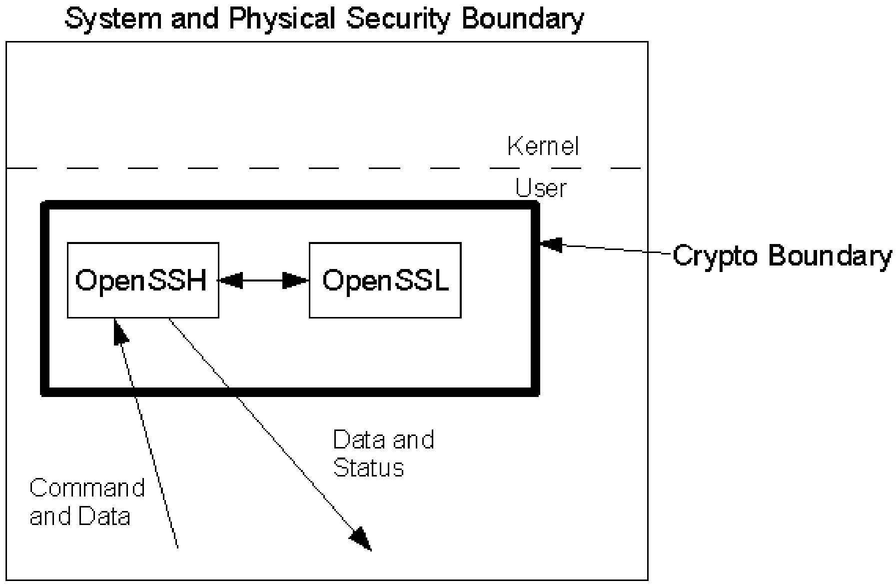

Figure 2. Software Block Diagram

1.4 Red Hat Enterprise Linux 6.2 Cryptographic Modules and FIPS 140-2 Certification

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 certified.

The certification is performed at FIPS level 1, a software-only certification that does not make any claims about the hardware enclosure. This allows vendors to develop their own higher level FIPS-certified modules using cryptographic modules.

The following cryptographic modules are included in the RHEL6.2 certification:

- 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 certification was performed on a 64-bit system, which are capable of executing both 32 and 64-bit code concurrently. Vendors can "transfer" the FIPS 140-2 certificate to the other similar platforms, provided the binary is only recompiled without changing the code. This is called a vendor assertion.

1.4.2 FIPS Approved Mode

Any vendor-provided FIPS certified cryptographic modules and services that use the RHEL6.2 underlying services to provide cryptographic functionality must use the RHEL6.2 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 certified 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 RHEL6 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

Function	Port
Command In	Keyboard, Network, Configuration File ~/.ssh/config, /proc/sys/crypto/fips_enabled, environment variable SSH_USE_STRONG_RNG, Command Line Options
Status Out	Display and Network
Data In	Keyboard, Configuration File ~/.ssh/known_hosts, /etc/ssh/ssh_known_hosts, Network (data via SSHv2 channel, data via local or remote port-forwarding port, data via TUN device), Key Files ~/.ssh/id_dsa* ~/ssh/id_rsa*
Data Out	Display, Network (data via SSHv2 channel, data via local or remote port-forwarding port, data via TUN device)

Table 3. Ports and Interfaces

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	Configure SSH Client Establish & Maintain SSH Session Close SSH Session (Zeroize) Terminate SSH Application Self Tests Show Status Installation

Table 4. 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.

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 OpenSSL Cryptographic Module validated cryptographic module. The Red Hat Enterprise Linux 6.2 OpenSSH Client Cryptographic Module performs SSH v2 functions only, and passes all cryptographic operations to Red Hat Enterprise Linux OpenSSL Cryptographic Module.

Service	Category	Function	Role	Cryptographic Keys and CSPs Accessed	Access Type (RWZ)
Establish & Maintain SSH Session	Symmetric Ciphers	Encrypt/Decrypt, Keyed-Hash, Key Exchange, Sign/Verify, Zeroize of private DSA/RSA client keys	User, crypto officer	RSA or DSA Client private key/public key Server Public Key DH private and public parameters, Session Encryption and Data Authentication Keys	RWZ RW RW

Service	Category	Function	Role	Cryptographic Keys and CSPs Accessed	Access Type (RWZ)
				DRNG Seed and Seed Key	RW
Close SSH Session	None	Zeroize	User, crypto officer	DH private and public parameters, Session Encryption and Data Authentication Keys DRNG Seed and Seed Key	Z
Terminate SSH Application	None	Zeroize	User, crypto officer	DH private and public parameters, Session Encryption and Data Authentication Keys DRNG Seed and Seed Key	Z
Self Tests	Self Test (includes Integrity and known answer tests)	Invoked by restarting the module	User, crypto officer	Software Integrity Key	R
Show Status	Status	Via verbose mode and exit codes	User, crypto officer	None	N/A
Installation	None	None	Crypto officer	None	N/A

Table 5. Services

3.2.1 Non-Approved Allowed Services

In Approved mode, the module will support the following Non-Approved services:

- RSA (encrypt, decrypt)
- Diffie-Hellman (key agreement; key establishment methodology)

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

5.1 Applicability

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

5.2 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.

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	Generation	Establishment	Access by Service	Entry and Output method	Storage	Zeroization
Client Private Keys	DSA or RSA keys	N/A	N/A	Establish & Maintain SSH Session	N/A	Plaintext	Immediately after use
Client Public Keys (not a CSP)	DSA or RSA keys	N/A	N/A	Establish & Maintain SSH Session	Exported	Plaintext	N/A
Server Public Key	DSA 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-SHA256	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 Parameters	DH	ANSI X9.31 RNG	N/A	Establish & Maintain SSH Session	N/A	Ephemeral	Close SSH Session or Terminate SSH Application
DRNG Seed	128-bit value	N/A	N/A	Establish & Maintain SSH Session	N/A (see section 6.3), provided by /dev/urandom	Ephemeral	N/A

Key	Type	Generation	Establishment	Access by Service	Entry and Output method	Storage	Zeroization
DRNG Seed Key	128-bit value	N/A	N/A	Establish & Maintain SSH Session	N/A (see section 6.3), provided /dev/urandom	Ephemeral	N/A

Table 6, Key Life Cycle

Notes:

The module ships without containing any keys and CSPs. When the module is configured, the crypto officer generates a DSA 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 transpires at the beginning of a session and with sessions after each 1 GB of data transfer or 1 hour of operation, 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

A FIPS 140-2, ANSI X9.31 approved pseudo random number generation mechanism will be used in the module, called from OpenSSL, which is seeded by the kernel.

The kernel uses /dev/urandom as a source of random numbers for RNG seeds. The Linux kernel initializes this pseudo device at system startup. Please note that when setting the environment variable of SSH_USE_STRONG_RNG to a value greater or equal to 6, the specified number of bytes will be pulled from /dev/random and mixed into the seed.

The kernel performs continual 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 Server DL585 Series
Regulatory Model Number: HSTNS-1025
Product Options: All
EMC: Class A

Product Name and Model: IBM BladeCenter HS22 Series
Regulatory Model Number: 09-EMCRTP-0008
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

Software Integrity Test: all cryptographic function tests are performed by the Red Hat Enterprise Linux OpenSSL module before it will perform cryptographic operations for the OpenSSH Client module.

8.1.1 Software Integrity Test Details

OpenSSH userspace modules have their integrity verified at startup by the software integrity test.

The integrity check is performed by the Red Hat Enterprise Linux OpenSSL module utility fipscheck using HMAC-SHA256.

When the module starts, it exercises the power-on self test, including the software integrity test. The software integrity test (HMAC-SHA256) constitutes a known answer test for the HMAC-SHA256 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 calling application file using HMAC-SHA256. Upon calling the FIPSCHECK_verify() function provided with libfipscheck.so, the fipscheck application is loaded and executed, and the following steps are performed:

- OpenSSL, as loaded by fipscheck, performs the integrity check of the OpenSSL library files using HMAC-SHA256.
- The application fipscheck performs the integrity check of its application file using HMAC-SHA256 provided by OpenSSL.
- The fipscheck application performs the integrity check of the calling application. The fipscheck computes the HMAC-SHA256 checksum of the file from the command line and compares the computed value to 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 – which is enforced by the libfipscheck.so library). The fipscheck application also automatically verifies the integrity of libfipscheck when loaded.

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.

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.

NOTE: All cryptographic functions for the Red Hat Enterprise Linux 6.2 OpenSSH Client Cryptographic Module will be provided by a copy of a FIPS 140-2 validated version of the Red Hat OpenSSL cryptographic module.

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.

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.

In addition to the configuration of the kernel, the OpenSSH client configuration ~/.ssh/config should contain:

- Either no "Ciphers" option or the option with a subset out of "aes128-ctr, aes192-ctr, aes256-ctr, aes128-cbc, 3des-cbc, aes192-cbc, aes256-cbc"

- Either no "MACs" option or the option with "hmac-sha1" and/or “hmac-sha1-96”
- "Protocol 2" must be specified

Reboot to apply these settings.

To operate the crypto module, the operating system must be restricted to a single operator mode of operation, and the ptrace(2) system call, the debugger (gdb(1)) and the strace(1) utility must not be used. In addition, other tracing mechanisms offered by the Linux environment, such as ftrace or systemtap shall not be used.

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.

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

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 explicitely 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	Cypher Block Chaining
CCM	Counter with Cipher Block Chaining-Message Authentication Code
CFB	Cypher 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
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
TDES	Triple DES
UI	User Interface

Table 7, Abbreviations

12 References

[1] OpenSSH Client user guide (provided with installation RPM, see section 1.1 Description of Module for version)

[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 180-3 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 186-3 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.