Red Hat Enterprise Linux 6.2 Openswan Cryptographic Module v2.0

FIPS 140-2 Security Policy

version 1.1
Last Update: 2012-11-13

Contents

1 Cryptographic Module Specification

This document is the non-proprietary security policy for the Red Hat Enterprise Linux 6.2 Openswan 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 6.2 Openswan Cryptographic Module is a software only cryptographic module that provides the IKE protocol version 1 and 2 key agreement services required for IPSec. The Openswan module is a software only, security level 1 cryptographic module, running on a multi-chip standalone platform.

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
Cryptographic Key Management	1
EMI/EMC	1
Self Tests	1
Design Assurance	1
Mitigation of Other Attacks	1

Table 1: Security Levels

The module has been tested on the following platforms:

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

Table 2: Platforms Tested

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 6.2 Openswan Cryptographic Module with the version of the RPM file of 2.6.32-18.el6_3.
- Network Security Service (NSS), a separately validated cryptographic module (FIPS 140-2 validation certificate #1837) provides cryptographic algorithms used by the IKE daemon. The IKE daemon uses

the NSS module in accordance with the Security Rules stated in the NSS Cryptographic Module Version 3.12.9.1 Security Policy.

- OpenSSL, a separately validated cryptographic module (FIPS 140-2 validation certificate #1758) provides cryptographic algorithms and security functions for the fipscheck application and library, which performs the integrity validation of the IKE daemon binary and supporting applications. The fipscheck application and library uses the OpenSSL module in accordance with the Security Rules stated in the OpenSSL Security Policy. The integrity check is performed by using HMAC SHA-256 (Certs. #1129, #1130, #1134 and #1135). The version of fipscheck and its library is 1.2.0-7.el6.
- The configuration of the FIPS mode is provided by the dracut-fips package with the version of the RPM file of 004-284.el6_3.1. This RPM version is provided with RHBA:2012-1318 accessible on the Red Hat Network.

This cryptographic boundary limits the external interface of the module to the interface provided by the Pluto daemon for the purposes of IKE. Therefore, it eliminates the need to protect cryptographic keys and other CSPs when crossing inter-component boundaries internal to the module. IKEv1 and IKEv2 provides the core security functionality.

The module is a FIPS security level 1 software module. The Linux platform is constrained to be used by a single user.

1.2 Description of Approved Mode

The Red Hat Enterprise Linux 6.2 Openswan Cryptographic Module cryptographic boundary is defined by the following ciphers:

Approved algorithms provided by the NSS library:

- Triple-DES (Certs #1289, 1290)
- AES (Certs #1985, 1986)
- SHA-1, SHA -256, SHA-384, SHA-512 (Certs #1741, 1742)
- DRBG (Certs #183, 184)
- DSA (Certs #634, 635)
- RSA (Cert #979, vendor affirmed)
- HMAC SHA-1, HMAC SHA-256, HMAC SHA-384, HMAC SHA-512 (Certs #1199, 1200)

Approved algorithms provided by the OpenSSL library:

- HMAC SHA-256 (Certs #1129, 1130, 1134, and 1135)

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

- RSA (encrypt, decrypt) (see caveat below)

The NSS library implements the following non-Approved algorithms, which shall not be used in the FIPS approved mode of operation:

- RC2; RC4; DES, Seed, or CAMELLIA

- MD2 and MD5 for hashing

CAVEAT:

The Module will support the following non-approved functions:

1) RSA (key wrapping; key establishment methodology provides between 80 and 150 bits of encryption strength)

2) Diffie-Hellman (key agreement; key establishment methodology provides between 80 and 112 bits of encryption strength)

1.3 Cryptographic Module Boundary

The Red Hat Enterprise Linux 6.2 Openswan Cryptographic Module physical boundary is defined by the surface of the case of the platform tested on. The logical module boundary is depicted in the software block diagram and is embodied by the Pluto IKE Daemon and its supportive applications found at /usr/libexec/ipsec/ which links with the NSS library. The integrity check mechanism provided by the fipscheck application and the OpenSSL library are part of the cryptographic module but not depicted as they are only used during load time of the module.

1.3.1 Hardware Block Diagram

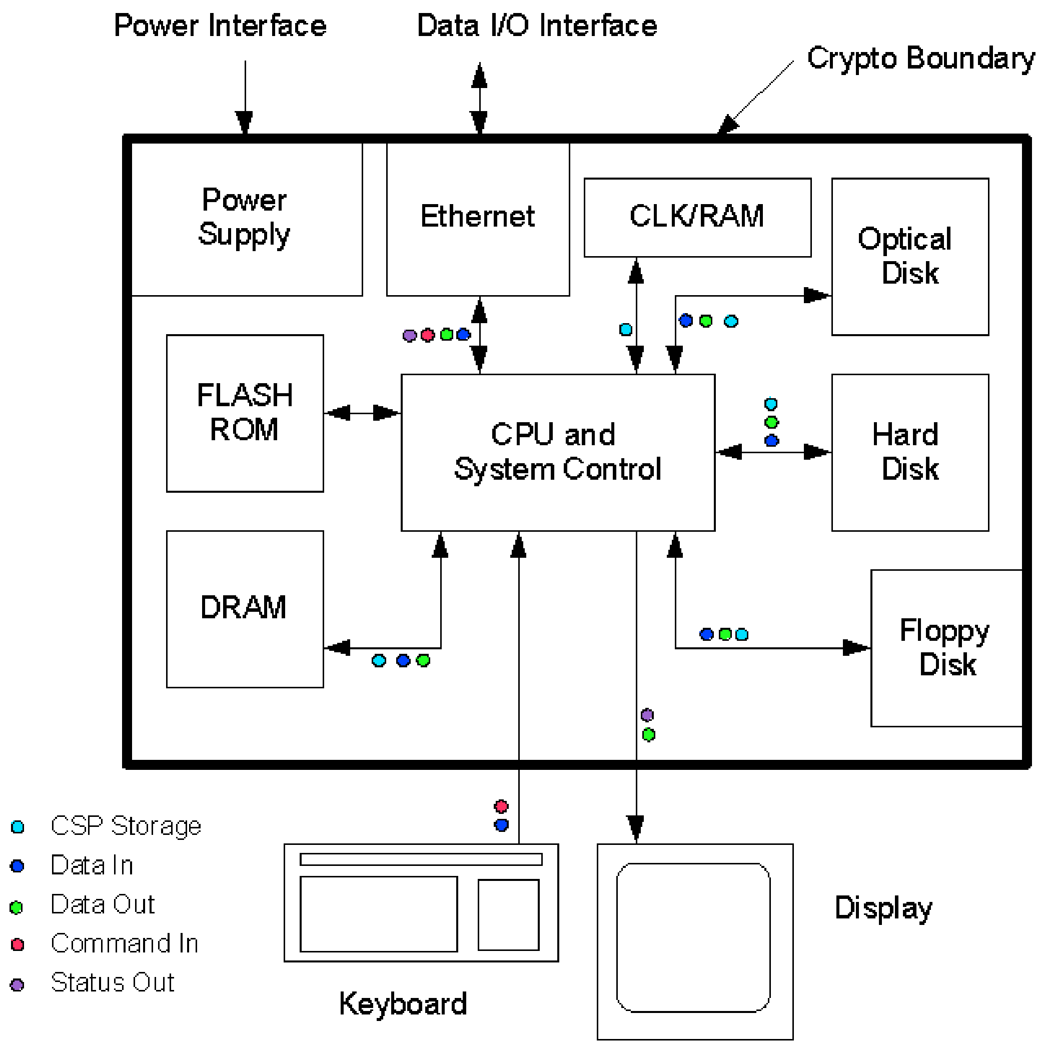

Figure 1: Hardware Block Diagram

1.3.2 Software Block Diagram

Note: This security policy only covers the user space module which includes the parts above the User/Kernel line in the block diagram below.

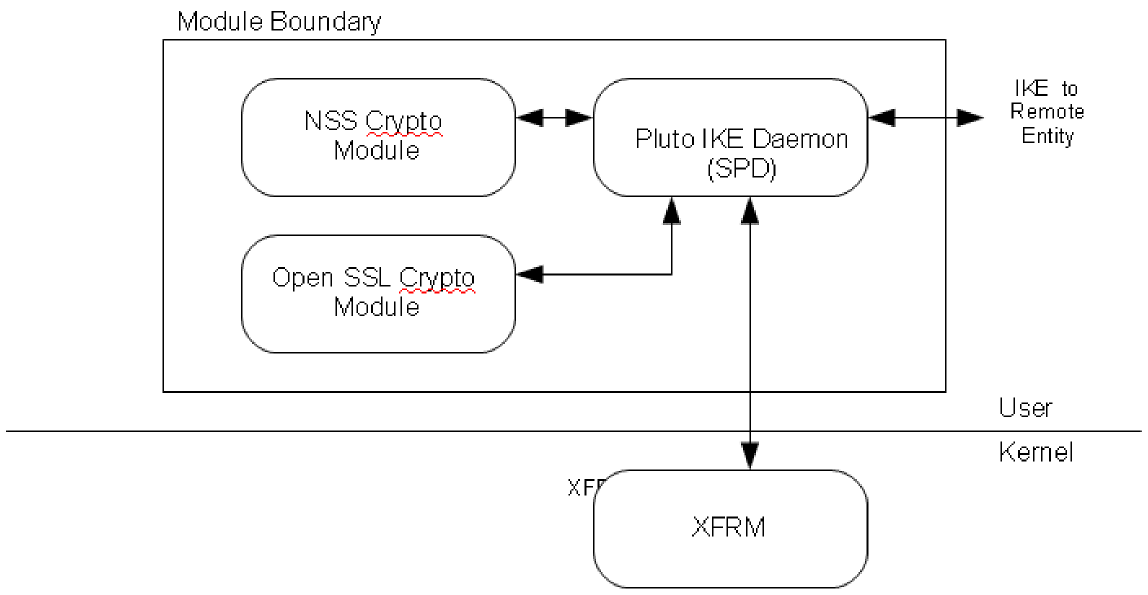

Figure 2: Software Block Diagram

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 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 and 2 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 FIPS services (libgcrypt, OpenSSL, Openswan) 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 RHEL6.2 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
Control In	Network Port/Protocol, Configuration Files (/etc/ipsec.conf, /etc/ipsec.d/, /etc/ipsec.secrets), Linux Kernel (XFRM Interface), command line
Status Out	Log File, Network Port/Protocol
Data In	Network Port/Protocol, NSS Key Database file stored in /etc/ipsec.d/
Data Out	Network Port/Protocol, Linux Kernel (XFRM Interface)

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	Key wrapping, key establishment, and key generation
Crypto Officer	Configuration, key wrapping, key establishment, and key generation

Table 4: Roles

The user role is assumed by the underlying server application that makes calls to the module on behalf of one or more external clients [Reference: Implementation Guidance for FIPS PUB 140-2, dated 3-3-11, Section 6.1].

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.

Service	Cryptographic Keys and CSPs Accessed	Crypto Officer	User
Install and Configure the module.	RSA public/private keys are added for SPD	●	
Manage Pluto IKE Daemon start, stop, etc.	DRNG Seed and Seed Key Zeroize of CSPs, Keys	●	
Negotiate IKE to establish security associations	DH private and public parameters RSA public/private used for authentication ISAKMP SA encryption key IKEv2 SA encryption key IPSEC SA encryption key	●	●
Run the FIPS self test (initiation by restarting the module)	N/A	●	
Read FIPS Status	N/A	●	

Table 5: Operational Services

3.3 Operator Authentication

There is no operator authentication, assumption of role is implicit by action.

3.4 Mechanism and Strength of Authentication

No authentication is required at security level 1; authentication is implicit by assumption of the role.

4 Physical Security

This is a security level 1, software only module and does not claim any physical security.

5 Operational Environment

5.1 Applicability

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

5.2 Policy

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

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

In the FIPS 140-2 approved mode, the ptrace(2) system call, 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

This section describes how keys and critical security parameters (CSP) are handled by 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 NSS.

6.1 Key Life Cycle Table

Key	Type	Generation	Establishment	Access by Service	Entry and output method	Storage	Zeroization
RSA Private and Public Keys	RSA key	N/A	N/A	Authentication in the ISAKMP SA, IKEv2 SA, and negotiation	N/A	Plaintext	Immediately after use by NSS
ISAKMP Security Association Tunnel Encryption Keys	AES or Triple-DES	N/A	Established during the ISAKMP SA handshake using DH	Establish & Maintain ISAKMP SA	N/A	Ephemeral	Close of ISAKMP SA or termination of Pluto IKE Daemon
IKEv2 Security Assocation Tunnel Encryption Keys	AES or Triple-DES	N/A	Established during the IKEv2 SA handshake using DH	Establish & Maintain IKEv2 SA	N/A	Ephemeral	Close of IKEv2 SA or termination of Pluto IKE Daemon
IPSEC Security Association Tunnel Encryption Keys	AES or Triple-DES	N/A	Established during the IPSEC SA handshake using DH	Establish & Maintain IPSEC SA	Transfer to the Linux Kernel via XFRM Interface	Ephemeral	Close of ISAKMP SA and IKEv2 SA or overwritten by re-negotiated IPSEC SA or termination of Pluto IKE Daemon
Diffie-Hellman Private and Public Parameters	DH	HASH_DRBG RNG	N/A	Establish & Maintain ISAKMP SA, IKEv2 SA, and IPSEC SA	N/A	Ephemeral	Close of ISAKMP SA and IKEv2 SA or termination of Pluto IKE Daemon

Key	Type	Generation	Establishment	Access by Service	Entry and output method	Storage	Zeroization
RNG Seed	Random value	N/A	N/A	Establish & Maintain ISAKMP SA, IKEv2 SA, and IPSEC SA	N/A, provided by /dev/urandom	Ephemeral	N/A (Termination of Pluto IKE Daemon where NSS zeroizes seed)
RNG Seed Key	Random value	N/A	N/A	Establish & Maintain ISAKMP SA, IKEv2 SA, and IPSEC SA	N/A, provided by /dev/urandom	Ephemeral	N/A (Termination of Pluto IKE Daemon where NSS zeroizes key)
Software Integrity Key for OpenSSL, fipscheck and all Pluto IKE applications	HMAC SHA-256	N/A	N/A	Self Tests	N/A	Plaintext within the OpenSSL and fipscheck libraries	Termination of the fipscheck application
Software Integrity Key for NSS library	DSA	N/A	N/A	Self Tests	N/A	Plaintext within the NSS library	Termination of Pluto IKE Daemon

Table 6: Key Life Cycle

Notes:

Private keys are always encrypted by the NSS library. When an operation requires a private key, the first pointer or handle to the private key is obtained using the public key and CKAID. Only during the operation, private keys are decrypted and the operation is performed. After the operation is over, the memory pointing to the private key is zeroized by NSS. Until the private keys from the NSS database need to be deleted, there is special zeroization required – for details about the maintenance of keys by the NSS library, see CMVP certificate #815.

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 employs a HASH_DRBG RNG provided by the NSS library, which is seeded by the kernel.

The Linux kernel provides /dev/urandom as a source of random numbers for RNG seeds. The Linux kernel initializes this pseudo device at system startup.

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.

The module performs both power-on self test (POST) and conditional self tests to verify the integrity and correct operational functioning of the cryptographic module. If the system fails a self test, it reports status indicating that a failure has occurred and transitions to an error state, which blocks all data input, data output, and control input via their respective interfaces.

While the module is performing any power on self test or conditional test, software rules within the executable image prevent the module from entering a state where data output via the data output interface is possible.

The crypto officer with physical or logical access to the module can run the POST on demand by power cycling the module or by rebooting the operating system.

The following table summarizes the system self tests and conditional tests.

Self Test	**Description**
Mandatory power-up tests performed at power-up and on demand:	
Cryptographic Algorithm Known Answer Tests	Each cryptographic algorithm (see section 1.1 for algorithm list) performed by the module, is tested using a “known answer” test to verify the correct operation of the algorithm. These tests are performed by the NSS library before it makes itself available to the Pluto IKE Daemon. In addition, the HMAC SHA-256 KAT is performed by the Red Hat Enterprise Linux OpenSSL library. HMAC SHA-256 is only used for the integrity check.
Integrity Test	The module computes an HMAC SHA-256 value for the the OpenSSL library, the fipscheck utility, and all applications forming the Openswan (this) module and compares it to a pre-calculated value stored within the system. The integrity check is performed by the Red Hat Enterprise Linux OpenSSL Cryptographic Module utility fipscheck using OpenSSL for the HMAC SHA-256 implementation. The integrity verification of the NSS library is performed by the NSS library using DSA.
Critical Functions tests performed at power-up:	
None	No security-relevant critical functions tests are performed.
Conditional tests performed, as needed, during operation:	
Continuous RNG	16 bits continuous testing is performed during each use of the approved RNG. This test is a “stuck at” test to check the RNG output data for failure to a constant value.

Table 7: Self Tests

Any self test success or failure messages are output to the Pluto IKE Daemon error log file.

Known answer tests for encryption/decryption or hashing, function by encrypting or hashing a string for which the calculated output is known and stored within the cryptographic module. An encryption or hashing test passes when the freshly calculated output matches the expected (stored) value. A test fails when the calculated output does not match the expected value. For decryption, the test then decrypts the ciphertext encrypted string. A decryption test passes when the freshly calculated output matches the plaintext value. A decryption test fails when the calculated output does not match the plaintext value.

Known answer tests for Random Number Generators function by seeding the RNG with known values and checking that the output matches the pre-calculated value stored within the cryptographic module. The test passes when the freshly generated output matches the pre-calculated value. A test fails when the generated output does not match the pre-calculated value.

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

The following section provides guidance for the Crypto Office for using the module in a way that maintains compliance with FIPS 140-2.

No specific guidance for the user is to be provided, as the user actions do not have an impact to the maintenance of a secure operational state of the module.

9.1 Cryptographic Officer Guidance

NOTE: All cryptographic functions for the Red Hat Enterprise Linux 6.2 Openswan Cryptographic Module will be provided by a copy of a FIPS 140-2 validated version of the Red Hat NSS library. The OpenSSL library is used to perform integrity verification.

- Configure pluto as specified in ipsec.conf(5) and ipsec.secrets(5) man pages, as well as the file README.nss provided by the Openswan RPM package.
- To start and stop the module, use the (service ipsec) command.
- Stopping the module will zeroize the ephemeral CSPs and keys.
- To check FIPS 140-2 module status, read the pluto debug data using the ipsec_barf(8) tool.
- The version of the RPM containing the validated module is stated in section 1.1 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.
- Pre-shared Keys are not supported and shall not be used in FIPS approved mode.
- Only the FIPS 140-2 approved and allowed ciphers listed in section 1.1 shall be used in configuring the pluto daemon.
- When zeroizing the module, the crypto officer is responsible for using a FIPS140-2 approved mechanism to clear the keys written on disk.
- The database for the cryptographic keys used by the pluto daemon must be initialized after it has been created as documented in the README.nss documentation with the following command, assuming that the database is stored in the directory /etc/ipsec.d/
  - `modutil -fips true -dbdir /etc/ipsec.d`

NOTE: Encryption and decryption of data is done implicitly when the kernel triggers pluto to set up a new Security Association.

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.

To bring the module into FIPS 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 must check and append, if necessary, 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"
```

This operation ensures proper operation in the Approved mode. If the flag in /proc/sys/crypto/fips_enabled is different than 1, the operation of the machine must be halted, the flag must be set to 1, and the machine must be rebooted. The module must not operate if the flag is set incorrectly, i.e., different than 1.

Reboot to apply these settings.

Configuration Changes and FIPS Approved Mode

Use care whenever making configuration changes that could potentially prevent access to the 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

There is no User Guidance as the user role is assumed by the underlying server application that makes calls to the module on behalf of one or more external clients.

9.3 Handling Self Test Errors

OpenSSL and NSS self test failures may prevent Openswan from operating. See the Guidance section in the OpenSSL and NSS Security Policies for instructions on handling OpenSSL or NSS self test failures.

The Openswan self tests consist of the software integrity test, cryptographic algorithm known answer tests, and a continuous RNG test. Integrity tests for Openswan are performed using the fipscheck API that is based on OpenSSL. Cryptographic algorithm known answer tests, and continuous RNG tests are performed by NSS. See the NSS security policy for guidance for NSS power-on self tests.

Power-on self test errors are non-fatal errors that transition the module into an error state. The application must be restarted or reinstalled to recover from these errors. Openswan outputs NSS error codes that can be used to determine the cause of the errors. In the case of integrity test failure, Openswan enters an error state and outputs the following error:

FIPS integrity verification test failed.

The only recovery from this type of failure is to reinstall the Openswan module. If you downloaded the software, verify the package hash to confirm a proper download.

10 Mitigation of Other Attacks

The following is taken from the FIPS 140-2 Security Policy documents of the OpenSSL module:

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.

The following is taken from the FIPS 140-2 Security Policy documents of the NSS module:

Other Attacks	Mitigation Mechanism	Specific Limitations
Timing attacks on RSA	RSA blinding Timing attack on RSA was first demonstrated by Paul Kocher in 1996 [2], who contributed the mitigation code to our module. Most recently Boneh and Brumley [3] showed that RSA blinding is an effective defense against timing attacks on RSA.	None
Cache-timing attacks on the modular exponentiation operation used in RSA and DSA	Cache invariant modular exponentiation This is a variant of a modular exponentiation implementation that Colin Percival [4] showed to defend against cache-timing attacks.	This mechanism requires intimate knowledge of the cache line sizes of the processor. The mechanism may be ineffective when the module is running on a processor whose cache line sizes are unknown.
Arithmetic errors in RSA signatures	Double-checking RSA signatures Arithmetic errors in RSA signatures might leak the private key. Ferguson and Schneier [5] recommend that every RSA signature generation should verify the signature just generated.	None

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
CC	Common Criteria
CMT	Cryptographic Module Testing
CMVP	Cryptographic Module Validation Program
CSP	Critical Security Parameter
CVT	Component Verification Testing
DES	Data Encryption Standard
DSA	Digital Signature Algorithm
EAL	Evaluation Assurance Level
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
PP	Protection Profile
RNG	Randome Number Generator
RSA	Rivest, Shamir, Addleman
SAP	Service Access Points
SDK	Software Development Kit
SHA	Secure Hash Algorithm
SHS	Secure Hash Standard
SLA	Service Level Agreement
SNMP	Simple Network Management Protocol
SOF	Strength of Function

SSH	Secure Shell
SVT	Scenario Verification Testing
TDES	Triple DES
TOE	Target of Evaluation
UI	User Interface

Table 8: Abbreviations

12 References

[1] Openswan 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.