



**Red Hat Enterprise Linux Libreswan
Cryptographic Module v4.0**

FIPS 140-2 Non-Proprietary Security Policy

Version 1.3

Last update: 2016-12-19

Prepared by:

atsec information security corporation

9130 Jollyville Road, Suite 260

Austin, TX 78759

www.atsec.com

Table of contents

1	Introduction.....	3
2	Cryptographic Module Specification.....	4
2.1	Module Overview.....	4
2.2	FIPS 140-2 Validation.....	6
2.3	Modes of Operation.....	7
3	Cryptographic Module Ports and Interfaces.....	8
4	Roles, Services and Authentication.....	9
4.1	Roles.....	9
4.2	Services.....	9
4.3	Authentication.....	9
5	Physical Security.....	10
6	Operational Environment.....	11
6.1	Applicability.....	11
6.2	Policy.....	11
7	Cryptographic Key Management.....	12
7.1	Random Number Generation.....	12
7.2	Key / CSP Storage.....	13
7.3	Key / CSP Zeroization.....	13
8	EMI/EMC.....	14
9	Self Tests.....	15
9.1	Power-Up Tests.....	15
9.1.1	Integrity Tests.....	15
9.1.2	Cryptographic Algorithm Tests.....	15
10	Guidance.....	16
10.1	Crypto Officer Guidance.....	16
10.1.1	Configuration Changes and FIPS Approved Mode.....	17
10.2	User Guidance.....	17
10.3	Handling Self-Test Errors.....	17
	Appendix A Glossary and Abbreviations.....	18
	Appendix B References.....	19

1 Introduction

This document is the non-proprietary Security Policy for the Red Hat Enterprise Linux Libreswan Cryptographic Module v4.0. It contains the security rules under which the module must operate and describes how this module meets the requirements as specified in FIPS PUB 140-2 (Federal Information Processing Standards Publication 140-2) for a Security Level 1 module.

2 Cryptographic Module Specification

2.1 Module Overview

The Red Hat Enterprise Linux Libreswan Cryptographic Module v4.0 (hereafter referred to as “the module”) is a software library implementing the cryptographic algorithms. The module provides cryptographic services to other network entities implementing the IKEv1 and IKEv2 protocols.

Note: This security policy only covers the IKE protocol, which is a part from the IPSEC protocol family.

The logical module boundary is depicted in the software block diagram below.

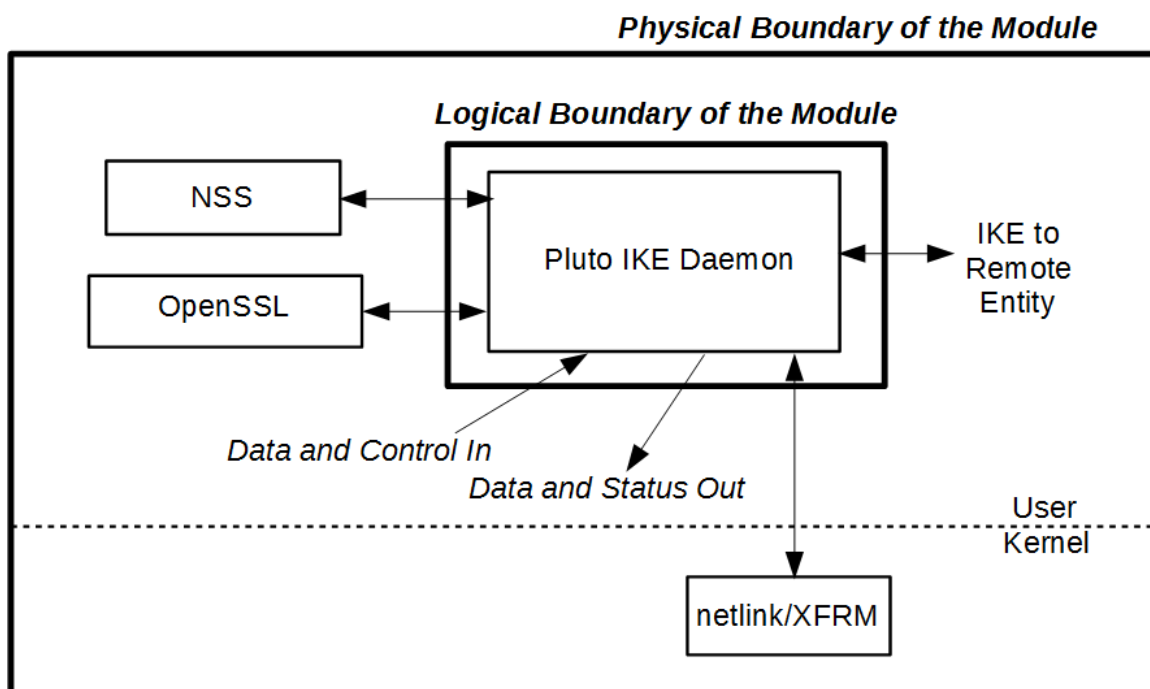


Figure 1: Software Block Diagram

The module is aimed to run on a general purpose computer; the physical boundary is the surface of the case of the target platform, as shown in the diagram below:

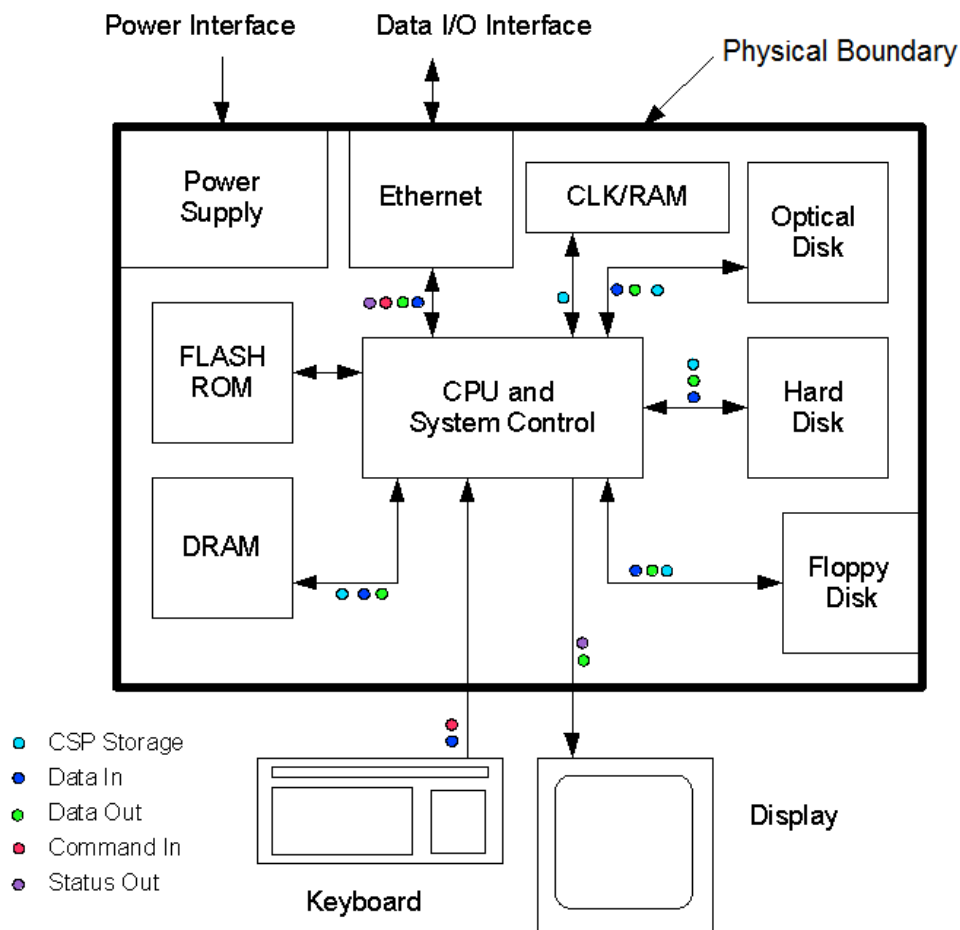


Figure 2: Hardware Block Diagram

This cryptographic module combines a vertical stack of Linux components, and the module intends to limit implementations, which are proved by each separate component, to the external interface. The components within the cryptographic boundary comprising the module are listed as follows:

- Red Hat Enterprise Linux Libreswan Cryptographic Module v4.0 with the version of the Libreswan RPM file 3.15-5.el7_1. This consists of Pluto IKE Daemon and its supportive applications found at `/usr/libexec/ipsec/`.
- Fipscheck RPM package (version 1.4.1-5.el7), that includes fipscheck library and application. Fipscheck performs the integrity validation of the IKE Daemon binary and supporting applications.

The following components which act as bound modules need to be installed for the Red Hat Enterprise Linux Libreswan Cryptographic Module v4.0 to operate:

- The bound module Red Hat Enterprise Linux NSS Cryptographic Module v4.0 with FIPS 140-2 Certificate #2711 (hereafter referred to as the "NSS module") 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 Security Policy.
- The bound module Red Hat Enterprise Linux 7.1 OpenSSL Module with FIPS 140-2 Certificate #2441 (hereafter referred to as the "OpenSSL module") provides HMAC SHA-256 algorithm required by fipscheck application and library for integrity check.

2.2 FIPS 140-2 Validation

For the purpose of the FIPS 140-2 validation, the module is a software-only, multi-chip standalone cryptographic module validated at security level 1. The table below shows the security level claimed for each of the eleven sections that comprise the FIPS 140-2 standard:

FIPS 140-2 Section		Security Level
1	Cryptographic Module Specification	1
2	Cryptographic Module Ports and Interfaces	1
3	Roles, Services and Authentication	1
4	Finite State Model	1
5	Physical Security	N/A
6	Operational Environment	1
7	Cryptographic Key Management	1
8	EMI/EMC	1
9	Self Tests	1
10	Design Assurance	1
11	Mitigation of Other Attacks	N/A

Table 1: Security Levels

The module has been tested on the following platforms:

Hardware Platform	Processor	Operating System	Tested	
			With AES-NI	Without AES-NI
HP Proliant DL380p Gen8	Intel® Xeon® E5-2600 v2 product family (x86)	Red Hat Enterprise Linux 7.1	yes	yes
IBM POWER8 Little Endian 8286-41A	POWER8E	Red Hat Enterprise Linux 7.1	n/a	n/a
IBM z13	IBM/S390	Red Hat Enterprise Linux 7.1	n/a	n/a

Table 2: Tested Platforms

The following operational environments are vendor affirmed:

Manufacturer	Model	O/S & Ver.
HP (Hewlett-Packard)	ProLiant DL380p Gen8	Red Hat Enterprise Linux 6.8 Red Hat Enterprise Linux 6.9
IBM (International Business Machines)	System x3500 M4	Red Hat Enterprise Linux 6.8 Red Hat Enterprise Linux 6.9

Table 3: Vendor Affirmed Platforms

The physical boundary is the surface of the case of the target platform. The logical boundary is depicted in Figure 1: software block diagram.

The module also includes algorithm implementations using Processor Algorithm Acceleration (PAA) functions provided by the different processors supported, as shown in the following table:

Processor	Processor Algorithm Acceleration (PAA) function	Cryptographic Module implementation
Intel x86	AES-NI	AES

Table 4: PAA function implementations

2.3 Modes of Operation

The module only supports the FIPS approved mode, and it turns to FIPS approved mode after initialization and power-on self-tests succeed.

The module verifies its integrity using a HMAC-SHA-256 digest operation and compares the value with the build time pre-computed value. If the digests match, the power-up self-tests are successful.

The services available in FIPS mode can be found in section 4.2, Table 6.

3 Cryptographic Module Ports and Interfaces

As a software-only module, the module does not have physical ports. For the purpose of the FIPS 140-2 validation, the physical ports are interpreted to be the physical ports of the hardware platform on which it runs.

The logical interfaces through which applications request services are summarized in following table:

Logical interface	Description
Data input	IKE Network Port/Protocol, NSS Key Database file stored in /etc/ipsec.d/
Data output	IKE Network Port/Protocol, Linux Kernel (netlink/XFRM Interface)
Control input	IKE Network Port/Protocol, Configuration Files (/etc/ipsec.conf, /etc/ipsec.d/, /etc/ipsec.secrets), Linux Kernel (netlink/XFRM Interface), command line
Status output	Log File, IKE Network Port/Protocol

Table 5: Logical Interfaces

4 Roles, Services and Authentication

4.1 Roles

The module supports the following roles:

- **User role:** performs key derivation and negotiates IKE to establish security association.
- **Crypto Officer role:** performs module installation and configuration, manages Pluto IKE Daemon, self tests and show status.

The module is a Security Level 1 software-only cryptographic module and does not implement authentication. The User and Crypto Officer roles are implicitly assumed by the entity accessing the module services. The User role is assumed by the underlying server application that makes calls to the module on behalf of one or more external clients.

4.2 Services

The module supports services available to users in the available roles. All services are described in detail in the user documentation.

The following table shows the available services, the roles allowed (“CO” stands for Crypto Officer and “U” stands for User), the Critical Security Parameters (CSPs) involved and how they are accessed in the FIPS mode.

“R” stands for Read permission, “W” stands for Write permission, and “Z” stands for Zeroization of the module:

Service	Algo(s).	CAVS Cert(s).	Role	CSPs	Access
Install and Configure the module	N/A	N/A	CO	RSA private keys	R, W, Z
Manage Pluto IKE Daemon start, stop, etc.	Commands	N/A	CO	Zeroize of CSPs, Keys	R, Z
Negotiate IKE to establish security associations (SAs)	SP 800-135 Key Derivation Function (KDF) in IKEv1 and IKEv2	CVL Certs. #679, #680, and #681	U	RSA private keys, shared secret IKE SA encryption keys and integrity keys, IPSEC SA encryption keys and integrity keys	R W
Self tests	N/A	N/A	CO	HMAC SHA-256 keys for integrity check	R
Show Status	N/A	N/A	CO	None	N/A

Table 6: Services available in FIPS mode

Note: Only the SP 800-135 Key Derivation Function has been validated by CAVP.

4.3 Authentication

The module is a Security Level 1 software-only cryptographic module and does not implement authentication. The role is implicitly assumed based on the service requested.

5 Physical Security

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

6 Operational Environment

6.1 Applicability

The module operates in a modifiable operational environment per FIPS 140-2 level 1 specifications. The module runs on a commercially available general-purpose operating system executing on the hardware specified in section 2.2.

6.2 Policy

The operating system is restricted to a single operator (concurrent operators are explicitly excluded). The application that request cryptographic services is the single user of the module, even when the application is serving multiple clients.

In FIPS Approved mode, the ptrace(2) system call, the debugger (gdb(1)), and strace(1) shall be not used.

7 Cryptographic Key Management

The application that uses the module is responsible for appropriate destruction and zeroization of the key material. The library provides functions for key allocation and destruction, which overwrites the memory that is occupied by the key information with “zeros” before it is deallocated.

7.1 Random Number Generation

The module does not implement any random number generator nor provides key generation. The module only provides key derivation through the implementation of the SP 800-135 KDF.

The table below lists the CSPs/keys used by the module:

Keys/CSPs	Type	Key Generation	Key Storage	Key Entry/Output	Key Zeroization
RSA 2048 and 3072 bits private keys	RSA keys used for authentication	N/A	Ephemeral	N/A	N/A
Shared secret according to the IKE protocol	Shared secret established by Diffie-Hellman key agreement	N/A	Ephemeral	IKE Network Port/Protocol	Close of IKE SA or termination of Pluto IKE Daemon
IKE SA Tunnel Encryption Keys	AES 128, 192, and 256 bits or Triple-DES 168 bits keys	N/A (derived from shared secret by using KDF)	Ephemeral	N/A	Close of IKE SA or termination of Pluto IKE Daemon
IKE SA Tunnel Integrity Keys	HMAC keys with at least 112 bits	N/A (derived from shared secret by using KDF)	Ephemeral	N/A	Close of IKE SA or termination of Pluto IKE Daemon
IPSEC SA Tunnel Encryption Keys	AES 128, 192, and 256 bits or Triple-DES 168 bits keys	N/A (derived from shared secret by using KDF)	Ephemeral	N/A	Close of IKE SA or overwritten by re-negotiated IPSEC SA or termination of Pluto IKE Daemon
IPSEC SA Tunnel Integrity Keys	HMAC keys with at least 112 bits	N/A (derived from shared secret by using KDF)	Ephemeral	N/A	Close of IKE SA or overwritten by re-negotiated IPSEC SA or termination of Pluto IKE Daemon

Table 7: Keys/CSPs

Notes:

The RSA private keys are encrypted by the NSS module. When an operation requires a private key, the first pointer or handle to the private key is obtained using the public key and CKA_ID (key ID). Only during the operation, private keys are decrypted and the operation is performed. After the operation, the memory pointing to the private key is zeroized by the NSS module.

7.2 Key / CSP Storage

Public and private keys are provided to the module by the calling process, and are destroyed when released by the appropriate IKE Network Port/Protocol. The module does not perform persistent storage of keys.

7.3 Key / CSP Zeroization

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

8 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 z13
Product Options: All
EMC: Class A

Product Name and Model: IBM Power8 Little Endian 8286-41A
Product Options: All
EMC: Class A

The HP ProLiant DL380p Gen8, IBM z13 and IBM Power8 Little Endian 8286-41A test platforms have “been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case the user will be required to correct the interference at his own expense.”

9 Self Tests

9.1 Power-Up Tests

The module performs power-up tests at module initialization which includes the software integrity test to ensure that the module is not corrupted. The self-tests are triggered automatically without any user intervention.

While the module is performing the power-up tests, services are not available and input or output is not possible: the module is single-threaded and will not return to the calling application until the self-tests are completed successfully.

9.1.1 Integrity Tests

The integrity check is performed by the fipscheck application using the HMAC-SHA-256 algorithm implemented by the OpenSSL module. The OpenSSL module computes an HMAC SHA-256 value for the fipscheck utility, as well as all applications forming the Libreswan module.

The integrity verification is performed as follows:

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

1. Fipscheck loads the OpenSSL module, which performs its own integrity check using the HMAC SHA-256 algorithm;
2. Fipscheck performs the integrity check of its own application file using the HMAC SHA-256 algorithm provided by the OpenSSL module;
3. Fipscheck automatically verifies the integrity of libfipscheck.so library before processing requests of calling applications;
4. The fipscheck application performs the integrity check of the Libreswan application file as follows:
The fipscheck computes the HMAC SHA-256 checksum of the file from the command line and compares the computed value to the value stored inside the /usr/lib64/fipscheck/<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. Otherwise, an error code will be shown, which puts the module into the error state.

If any of the above steps fails, an error code (a non-zero value) will be returned and the module enters the error state. In Error state, all output is inhibited and no cryptographic operation is allowed. The Module needs to be reinitialized in order to recover from the Error state.

The power-up self tests can be performed on demand by reinitializing the Module.

9.1.2 Cryptographic Algorithm Tests

The power-up self tests for the SP 800-135 KDF are covered by the SHS Known-Answer-Tests (KAT) performed by the NSS module. If any of the power-up self-tests fail, the Module enters the Error state. In the Error state, all outputs are inhibited and no cryptographic operation is allowed.

10 Guidance

The following guidance items are to be used for assistance in maintaining the module's validated status while in use.

10.1 Crypto Officer Guidance

NOTE: All cryptographic functions for the Red Hat Enterprise Linux Libreswan Cryptographic Module v4.0 will be provided by a copy of a FIPS 140-2 validated version of the NSS module. The OpenSSL module 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 RPM package: `libreswan-3.15-5.el7_1`.
- To start and stop the module, use the `(service ipsec)` command.
- `ikelifetime` should not be larger than 1 hour.
- `salifetime` should not be larger than 1 hour.
- Galois Counter Mode (GCM) and Counter with Cipher Block Chaining Message Authentication Code (CCM) should be used with their full tag lengths.
- `SHA2-truncbug` should be set as “no”.
- Aggressive mode should not be used.
- Only RSA authentication is allowed for authenticating the remote peer.
- 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 2.1 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 2.1 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.

10.1.1 Configuration Changes and FIPS Approved Mode

Use caution 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.

10.2 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.

10.3 Handling Self-Test Errors

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

Power-up 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. Libreswan outputs NSS error codes that can be used to determine the cause of the errors. In the case of integrity test failure, Libreswan 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 Libreswan module. If you downloaded the software, verify the package hash to confirm a proper download.

Appendix A Glossary and Abbreviations

AES	Advanced Encryption Standard
AES-NI	Advanced Encryption Standard New Instructions
CAVP	Cryptographic Algorithm Validation Program
CCM	Counter with Cipher Block Chaining Message Authentication Code
CMAC	Cipher-based Message Authentication Code
CMVP	Cryptographic Module Validation Program
CSP	Critical Security Parameter
DES	Data Encryption Standard
DSA	Digital Signature Algorithm
FIPS	Federal Information Processing Standards Publication
GCM	Galois Counter Mode
HMAC	Hash Message Authentication Code
MAC	Message Authentication Code
NIST	National Institute of Science and Technology
PAA	Processor Algorithm Acceleration
RSA	Rivest, Shamir, Adleman
SHA	Secure Hash Algorithm
SHS	Secure Hash Standard
TDES	Triple DES

Appendix B References

- FIPS180-4** **Secure Hash Standard (SHS)**
March 2012
http://csrc.nist.gov/publications/fips/fips180-4/fips_180-4.pdf
- FIPS197** **Advanced Encryption Standard**
November 2001
<http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>
- FIPS198-1** **The Keyed Hash Message Authentication Code (HMAC)**
July 2008
http://csrc.nist.gov/publications/fips/fips198_1/FIPS-198_1_final.pdf
- PKCS#1** **Public Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1**
February 2003
<http://www.ietf.org/rfc/rfc3447.txt>
- RFC3394** **Advanced Encryption Standard (AES) Key Wrap Algorithm**
September 2002
<http://www.ietf.org/rfc/rfc3394.txt>
- RFC5649** **Advanced Encryption Standard (AES) Key Wrap with Padding Algorithm**
September 2009
<http://www.ietf.org/rfc/rfc5649.txt>
- SP800-38F** **NIST Special Publication 800-38F - Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping**
December 2012
<http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38F.pdf>
- SP800-56A** **NIST Special Publication 800-56A Revision 2 - Recommendation for Pair Wise Key Establishment Schemes Using Discrete Logarithm Cryptography**
May 2013
http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800_56Ar2.pdf
- SP800-56C** **Recommendation for Key Derivation through Extraction-then-Expansion**
November 2011
<http://csrc.nist.gov/publications/nistpubs/800-56C/SP-800-56C.pdf>
- SP800-67** **NIST Special Publication 800-67 Revision 1 - Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher**
January 2012
<http://csrc.nist.gov/publications/nistpubs/800-67-Rev1/SP-800-67-Rev1.pdf>
- SP800-90B** **NIST Draft Special Publication 800-90B - Recommendation for the Entropy Sources Used for Random Bit Generation**
August 2012
<http://csrc.nist.gov/publications/drafts/800-90/draft-sp800-90b.pdf>
- SP800-108** **NIST Special Publication 800-108 - Recommendation for Key Derivation Using Pseudorandom Functions**
October 2009
<http://csrc.nist.gov/publications/nistpubs/800-108/sp800-108.pdf>

SP800-131A NIST Special Publication 800-131A - Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths

January 2011

<http://csrc.nist.gov/publications/nistpubs/800-131A/sp800-131A.pdf>