Red Hat Enterprise Linux - OpenSSL Module v1.1

FIPS 140-2 Security Policy

version 1.1

Last Update: 2012-08-24

Contents

Document History

Version	Date of Change	Author	Changes to Previous Version
0.1	2008-07-22	SHW - atsec	Initial
0.2	2009-01-14	SHW - atsec	updated
0.3	2009-10-09	SHW - atsec	First draft
1.0	2009-10-27	SHW - atsec	First release
1.1	2010-04-13	SHW - atsec	Single User Mode
1.2	2010-05-06	SHW - atsec	Final cleanup for NIST
1.0	2012-06-06	JF- atsec	Updated to Module version 1.1
1.1	2012-08-24	ATV-atsec	Add RHEL 5.8 as new tested OE

1 Cryptographic Module Specification

This document is the non-proprietary security policy for the OpenSSL FIPS Object Module, and was prepared as part of the requirements for conformance to Federal Information Processing Standard (FIPS) 140-2, Level 1.

1.1 Description of the Module

The OpenSSL FIPS Object Module (hereafter referred to as the "Module") is a software library supporting FIPS 140-2 -approved cryptographic algorithms. For the purposes of the FIPS 140-2 level 1 validation, the OpenSSL FIPS Object Module is Red Hat Enterprise Linux OpenSSL Module 1.1.

This module provides a C language application program interface (API) for use by other processes that require cryptographic functionality.

For FIPS 140-2 purposes, the Module is classified as a multi-chip standalone module. The Module's logical cryptographic boundary is the shared library files and their integrity check HMAC files, they are: .libcrypto.so.0.9.8e.hmac, libcrypto.so.6.hmac, .libssl.so.0.9.8e.hmac, .libssl.so.6.hmac, libcrypto.so.0.9.8e, libcrypto.so.6, libssl.so.0.9.8e and libssl.so.6. For ia64 they are in the /lib directory, for x86_64 they are in the /lib64 directory, for x86_32 (i386) they are in the /lib directory.

The Module's physical cryptographic boundary is the enclosure of the computer system on which it is executing.

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

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	N/A

Table 1: Security Level of the Module

The module has been tested in the following software configurations:

- 32 bit x86_64
- 64 bit x86_64
- 64 bit Itanium

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

Manufacturer	Model	O/S & Ver.
HP	HP Integrity Server RX2660	Red Hat Enterprise Linux 5.4 or Red Hat Enterprise Linux 5.8 (Single User Mode)
HP	HP ProLiant Server DL585 (library in 64 bit word size and 32 bit word size)	Red Hat Enterprise Linux 5.4 or Red Hat Enterprise Linux 5.8 (Single User Mode)

Table 2: Tested platforms

1.2 Description of the Approved Mode

The module supports the following FIPS 140-2 approved algorithms:

Algorithm	Validation Certificate	Usage	Keys/CSPs
AES	Certs. #1160, #1161 and #1162	encrypt/decrypt	AES keys 128, 192, 256 bits
Triple-DES	Certs. #839, #840 and #841	encrypt/decrypt	Triple-DES keys 168 bits
DSA	Certs. #378, #379 and #380	Sign, verify and keygen	DSA keys 1024 bits
ANSI X9.31PRNG	Certs. #642, #643 and #644	random number generation	PRNG seed value and seed key 128 bits
SHA-1	Certs. #1073, #1074 and #1075	hashing	N/A
SHA-224	Certs. #1073, #1074 and #1075	hashing	N/A
SHA-256	Certs. #1073, #1074 and #1075	hashing	N/A
SHA-384	Certs. #1073, #1074 and #1075	hashing	N/A
SHA-512	Certs. #1073, #1074 and #1075	hashing	N/A
HMAC-SHA-1	Certs. #661, #662 and #663	message integrity	HMAC Key
HMAC-SHA224	Certs. #661, #662 and #663	message integrity	HMAC Key
HMAC-SHA256	Certs. #661, #662 and #663	message integrity	HMAC Key
HMAC-SHA384	Certs. #661, #662 and #663	message integrity	HMAC Key
HMAC-SHA512	Certs. #661, #662 and #663	message integrity	HMAC Key
RSA (X9.31, PKCS #1.5, PSS)	Certs. #549, #550 and #551	Sign, verify and keygen	RSA keys 1024 to 16384 bits

Table 3: Approved algorithms

The module supports the following non-FIPS 140-2 approved algorithms:

Algorithm	**Validation Certificate**	**Usage**	**Keys/CSPs**
Diffie-Hellman	N/A, see caveat below	Key agreement and establishment	none
RSA (encrypt, decrypt)	N/A See caveat below	Key agreement and establishment	RSA keys 1024 to 16384 bits
MD5	NA, see caveat below	Message digest	N/A

Table 4Non-Approved algorithms

CAVEATS:

1) Diffie-Hellman (key agreement; key establishment methodology provides between 80 and 219 bits of encryption strength).

2)RSA (key wrapping; key establishment methodology provides between 80 and 256 bits of encryption strength).

3) MD5 for use in TLS only.

1.3 Cryptographic Boundary

The Module's physical boundary is the surface of the case of the platform (depicted in the hardware block diagram). The Module's logical boundary is depicted in the software block diagram.

1.3.1 Hardware Block Diagram

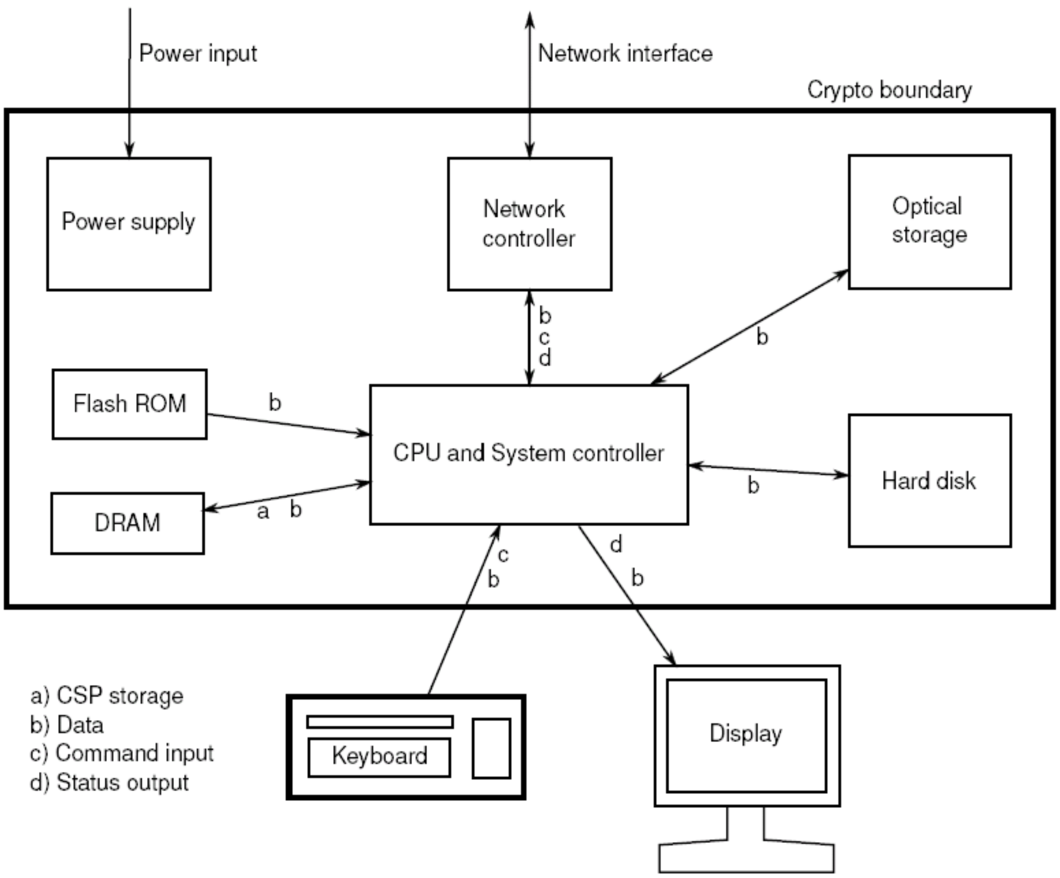

Figure 1, Hardware Block Diagram

1.3.2 Software Block Diagram

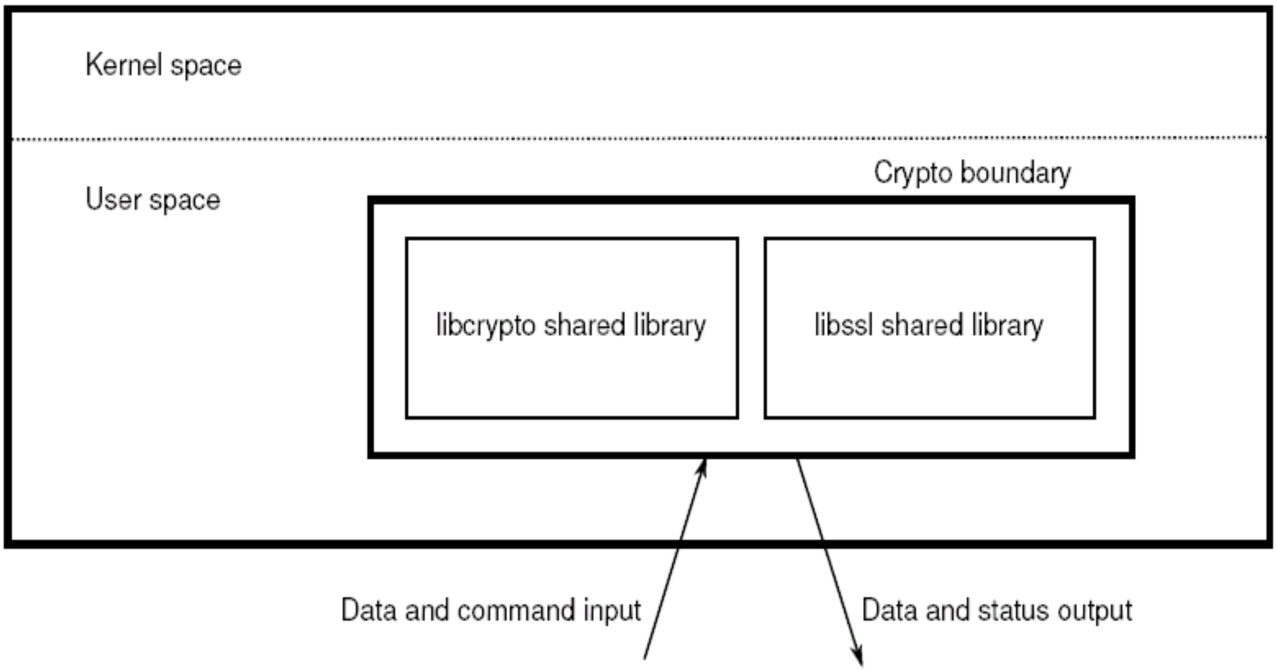

Figure 2, Software Block Diagram

2 Cryptographic Module Ports and Interfaces

The physical ports of the Module are the same as the computer system on which it is executing. The logical interface is a C-language application program interface (API).

The Data Input interface consists of the input parameters of the API functions. The Data Output interface consists of the output parameters of the API functions. The Control Input interface consists of the actual API functions. The Status Output interface includes the return values of the API functions. The ports and interfaces are shown in the following table.

FIPS Interface	Physical Port	Module Interface
Data Input	Ethernet ports	API input parameters, kernel I/O – network or files on filesystem
Data Output	Ethernet ports	API output parameters, kernel I/O – network or files on filesystem
Control Input	Keyboard, Serial port, Ethernet port	API function calls, or configuration files on filesystem
Status Output	Serial port, Ethernet port	API
Power Input	PC Power Supply Port	N/A

Table 5: 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

There are two users of the module, which are identified along with their allowed services in Table 5 (and the services are further detailed in Table 6).

Role	Services (see list below)
User	Encryption, Decryption (symmetric and public/private), Random Numbers
Crypto Officer	Configuration of FIPS 140-2 validated mode, Encryption, Decryption (symmetric and public/private), Random Numbers

Table 6: Roles

The User and Crypto-Officer roles are implicitly assumed by the entity accessing services implemented by the Module.

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.

Service	Role	Algorithms and CSPs	Access
Symmetric encryption/decryption	User, Crypto Officer	symmetric key AES, TDES	read/write/execute
Key transport	User, Crypto Officer	asymmetric private key RSA	read/write/execute
Digital signature	User, Crypto Officer	asymmetric private key RSA, DSA	read/write/execute
Symmetric key generation	User, Crypto Officer	symmetric key AES, TDES	read/write/execute
TLS	User, Crypto Officer	symmetric key AES, TDES asymmetric public/private key RSA, HMAC Key	read/write/execute
TLS Key Agreement	User, Crypto Officer	symmetric key AES, TDES asymmetric public/private key	read/write/execute

Service	Role	Algorithms and CSPs	Access
		RSA, HMAC Key, Premaster Secret, Master Secret and DH Secret.	
Certificate Management/ Handling	User, Crypto Officer	Certificates	read/write/execute
Asymmetric key generation	User, Crypto Officer	asymmetric private key RSA, DSA	read/write/execute
Keyed Hash (HMAC)	User, Crypto Officer	HMAC Key, HMAC SHA-1, HMAC SHA-224, HMAC SHA-256, HMAC SHA-384, HMAC SHA-512	read/write/execute
Message digest (SHS)	User, Crypto Officer	SHA-1, SHA-224, SHA-256, SHA-384, SHA-512	read/write/execute
Random number generation	User, Crypto Officer	PRNG Seed and Seed Key	read/write/execute
Show status	User, Crypto Officer	none	execute
Module initialization	User, Crypto Officer	none	execute
Self test	User, Crypto Officer	HMAC SHA-256 key	read/execute
Zeroize	User, Crypto Officer	symmetric key, asymmetric key, HMAC key, Seed and Seed key	read/write/execute

Table 7: Service details

3.3 Operator Authentication

At security level 1, authentication is neither required nor employed. The role is implicitly assumed on entry.

3.4 Mechanism and Strength of Authentication

At security level 1, authentication is not required.

4 Physical Security

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

5 Operational Environment

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

5.1 Policy

The operating system is 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 FIPS-approved mode, the ptrace(2) system call, the debugger (gdb(1)), and strace(1) shall be not used.

6 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 overwrite the memory that is occupied by the key information with “zeros” before it is deallocated.

6.1 Random Number Generation

The Module employs an ANSI X9.31-compliant random number generator for creation of asymmetric and symmetric keys.

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.

6.2 Key/Critical Security Parameter (CSP) Authorized Access and Use by Role and Service/Function

An authorized application as user (the User role) has access to all key data generated during the operation of the Module.

6.3 Key/CSP Storage

Public and private keys are provided to the Module by the calling process, and are destroyed when released by the appropriate API function calls. The Module does not perform persistent storage of keys.

6.4 Key/CSP Zeroization

The memory occupied by keys is allocated by regular libc malloc/calloc() calls. The application is responsible for calling the appropriate destruction functions from the OpenSSL API. The destruction functions then overwrite the memory occupied by keys with “zeros” and deallocates the memory with the free() call. In case of abnormal termination, or swap in/out of a physical memory page of a process, the keys in physical memory are overwritten by the Linux kernel before the physical memory is allocated to another process.

7 Electromagnetic Interference/Electromagnetic Compatibility (EMI/EMC)

Product Name and Model: HP ProLiant Server DL585 Series
Regulatory Model Number: HSTNS-1025
Product Options: All
conforms to the following Product Specifications and Regulations:

EMC: Class A
CISPR 22:2005
EN 55022:2006
EN 55024:1998 +A1:2001 +A2:2003
EN 61000-3-2:2006
EN 61000-3-3:1995 +A1:2001 +A2:2005

Product Name and Model: HP Integrity Server rx2660
Regulatory Model Number: RSVLA-0503
Product Options: All
conforms to the following Product Specifications and Regulations :
EMC: Class A
CISPR22:1997 / EN 55022:1998
CISPR 24:1997 + A1:2001 + A2: 2002 / EN 55024:1998 + A1:2001 + A2:2003
EN 61000-3-2:2000
EN 61000-3-3:1995 +A1:2001

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 start up. 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

The Module performs both power-up self tests (at module initialization) and continuous condition tests (during operation). Input, output, and cryptographic functions cannot be performed while the Module is in a self-test or error state because the module is single-threaded and will not return to the calling application until the power-up self tests are complete. If the power-up self tests fail, subsequent calls to the module will also fail - thus no further cryptographic operations are possible.

Algorithm	Test
AES	KAT
Triple-DES	KAT
DSA	pairwise consistency test, sign/verify
RSA	KAT
PRNG	KAT
HMAC-SHA-1	KAT
HMAC-SHA-224	KAT
HMAC-SHA-256	KAT
HMAC-SHA-384	KAT
HMAC-SHA-512	KAT
SHA-1	Tested as part of HMAC SHA-1
SHA-224	Tested as part of HMAC SHA-224
SHA-256	Tested as part of HMAC SHA-256
SHA-384	Tested as part of HMAC SHA-384

Algorithm	Test
SHA-512	Tested as part of HMAC SHA-512
module integrity	HMAC-SHA-256

Table 7: Module self tests

8.2 Conditional Tests

Algorithm	Test
DSA	pairwise consistency
RSA	pairwise consistency
PRNG	continuous test

Table 8: Module conditional tests

8.3 Cryptographic Function

A single initialization call, FIPS_mode_set, is required to initialize the Module for operation in the FIPS 140-2 Approved mode. When the Module is in FIPS mode, all security functions and cryptographic algorithms are performed in Approved mode.

The FIPS mode initialization is performed when the application invokes the FIPS_mode_set() call which returns a “1” for success or a “0” for failure. The module will support either explicit FIPS mode initialization through the FIPS_mode_set() function or implicit initialization by querying the /proc/sys/crypto/fips_enabled flag. If the flag is set and the OpenSSL library is being initialized, it will automatically call FIPS_mode_set(1) during this initialization. Interpretation of this return code is the responsibility of the host application. Prior to this invocation the Module is uninitialized in non-FIPS mode by default.

The FIPS_mode_set() function verifies the integrity of the runtime executable using a HMAC SHA-256 digest which is computed at build time. If this computed HMAC SHA-256 digest matches the stored, known digest, then the power-up self-test (consisting of the algorithm-specific Pairwise Consistency and Known Answer tests) is performed. If any component of the power-up self-test fails, an internal global error flag is set to prevent subsequent invocation of any cryptographic function calls. Any such power-up self test failure is a hard error that can only be recovered by reinstalling the Module[1]. If all components of the power-up self-test are successful, then the Module is in FIPS mode. The power-up self-tests may be performed at any time by reloading the module.

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

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 version 0.9.8e-22.el5_8.3. The integrity of the RPM

1 *The FIPS_mode_set() function could be re-invoked but such re-invocation does not provide a means from recovering from an integrity test or known answer test failure.*

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. If the libraries were already prelinked, the prelink should be undone on all the system files using the 'prelink -u -a' command.Operators must first invoke OPENSSL_init(void) before using the module.

ENGINE_register_* and ENGINE_set_default_* function calls are prohibited while in the Approved mode. Furthermore, once the Approved mode is entered, it must not be exited, which prohibits calls to FIPS_mode_set(0).

Only the cipher types listed in section 1.2 are allowed to be used.

To bring the module into FIPS mode, the crypto officer has to regenerate the initrd by using the following command:

For the x86_64 platform, the command is:

mkinitrd --with-fips -f /boot/initrd-$(uname -r).img $(uname -r)

For the IA64, the command is:

mkinitrd --with-fips -f /boot/efi/efi/redhat/initrd-$(uname -r).img $(uname -r)

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

10 Mitigation of Other Attacks

The Module does not contain additional security mechanisms beyond the requirements for FIPS 140-2 level 1 cryptographic modules.

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
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
PRNG	Pseudo Random Number Generator
RNG	Random Number Generator
RSA	Rivest, Shamir, Addleman
SDK	Software Development Kit
SHA	Secure Hash Algorithm
SHS	Secure Hash Standard
SLA	Service Level Agreement
SOF	Strength of Function
SSH	Secure Shell
TDES	Triple DES
UI	User Interface

12 References

[1] OpenSSL user guide (provided with installation and -devel RPMs, see section 1.1 Description of Module for version)

[2] rx2660_EMIEMC_cert.pdf (On file at Red Hat)

[3] DL585_EMIEMC_CEcert.pdf (On file at Red Hat)

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

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

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

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

[8] FIPS 180-3 Secure Hash Standard, http://csrc.nist.gov/publications/PubsFIPS.html

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

[10] FIPS 186-3 Digital Signature Standard (DSS), http://csrc.nist.gov/publications/PubsFIPS.html

[11] 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.