McAfee Inc.

McAfee Linux Cryptographic Module

Software Version: 1.0.1

FIPS 140-2 Non-Proprietary Security Policy

FIPS Security Level: 1
Document Version: 1.3

Prepared for:

McAfee Inc.
2821 Mission College Blvd
Santa Clara, CA 95054
United States of America

Phone: +1 888 847 8766
Email: info@mcafee.com
http://www.mcafee.com

Prepared by:

Corsec Security, Inc.
13921 Park Center Road, Suite 460
Herndon, VA 20171
United States of America

Phone: +1 703 267-6050
Email: info@corsec.com
http://www.corsec.com

Table of Contents

Table of Figures

List of Tables

1 Introduction

1.1 Purpose

This is a non-proprietary Cryptographic Module Security Policy for the McAfee Linux Cryptographic Module from McAfee Inc. This Security Policy describes how the McAfee Linux Cryptographic Module meets the security requirements of Federal Information Processing Standards (FIPS) Publication 140-2, which details the U.S. and Canadian Government requirements for cryptographic modules. More information about the FIPS 140-2 standard and validation program is available on the National Institute of Standards and Technology (NIST) and the Communications Security Establishment (CSE) Cryptographic Module Validation Program (CMVP) website at http://csrc.nist.gov/groups/STM/cmvp.

This policy was prepared as part of the Level 1 FIPS 140-2 validation of the module. The McAfee Linux Cryptographic Module may also be referred to in this document as the crypto-module or the module.

1.2 References

This document deals only with operations and capabilities of the module in the technical terms of a FIPS 140-2 cryptographic module security policy. More information is available on the module from the following sources:

- The McAfee website (http://www.mcafee.com) contains information on the full line of products from McAfee.
- The CMVP website (http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140val-all.htm) contains contact information for individuals to answer technical or sales-related questions for the module.

1.3 Document Organization

The Security Policy document is one document in a FIPS 140-2 Submission Package. In addition to this document, the Submission Package contains:

- Vendor Evidence document
- Finite State Model document
- Other supporting documentation as additional references

This Security Policy and the other validation submission documentation were produced by Corsec Security, Inc. under contract to McAfee. With the exception of this Non-Proprietary Security Policy, the FIPS 140-2 Submission Package is proprietary to McAfee and is releasable only under appropriate non-disclosure agreements. For access to these documents, please contact McAfee.

2 McAfee Linux Cryptographic Module

2.1 Overview

The McAfee Linux Operating System (MLOS) v2.2.3 is a proprietary Operating System (OS) developed by McAfee in order to provide a single OS distribution that can be used across multiple McAfee products and appliances. Incorporated into MLOS 2.2.3 is the McAfee Linux Cryptographic Module, a software cryptographic library which will provide security services such as encryption, decryption, hashing, signature generation, and signature verification to McAfee software products which incorporate MLOS 2.2.3.

The McAfee Linux Cryptographic Module is validated at the FIPS 140-2 Section levels shown in Table 1:

Table 1 – Security Level Per FIPS 140-2 Section

Section	Section Title	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]	1
9	Self-tests	1
10	Design Assurance	1
11	Mitigation of Other Attacks	N/A

2.2 Module Specification

The McAfee Linux Cryptographic Module is a software cryptographic module (Software Version: 1.0.1) with a multichip standalone embodiment. The overall security level of the module is Level 1. The McAfee Linux Cryptographic Module is a shared cryptographic library providing symmetric and asymmetric encryption and decryption, hashing, message authentication, key generation, and digital signature generation and verification. The McAfee Linux Cryptographic Module consists of OpenSSL 1.0.1j and a "FIPS Controller Daemon" that controls the operational status of the module.

The module was tested and found to be FIPS 140-2 compliant on three separate Intel Server Systems executing McAfee Linux Operating System v2.2.3. Section 2.6 provides the full list of operational environments on which the module was tested.

Because the McAfee Linux Cryptographic Module is defined as a software cryptographic module, it possesses both a logical cryptographic boundary and a physical cryptographic boundary. The physical and logical boundaries are outlined in Section 2.2.1 and 2.2.2 respectively.

[1] EMI/EMC – Electromagnetic Interference / Electromagnetic Compatibility

2.2.1 Physical Cryptographic Boundary

As a software cryptographic module, there are no physical protection mechanisms implemented. Therefore, the module must rely on the physical characteristics of the host system. The physical boundary of the cryptographic module is defined by the hard enclosure around the host system on which it runs. The module supports the physical interfaces an Intel Server System hardware appliance. These interfaces include the integrated circuits of the system board, processor, network adapters, RAM[2], hard disk, device case, power supply, and fans. Figure 1 and Figure 2 provide block diagram representations of each Intel Server System hardware appliance.

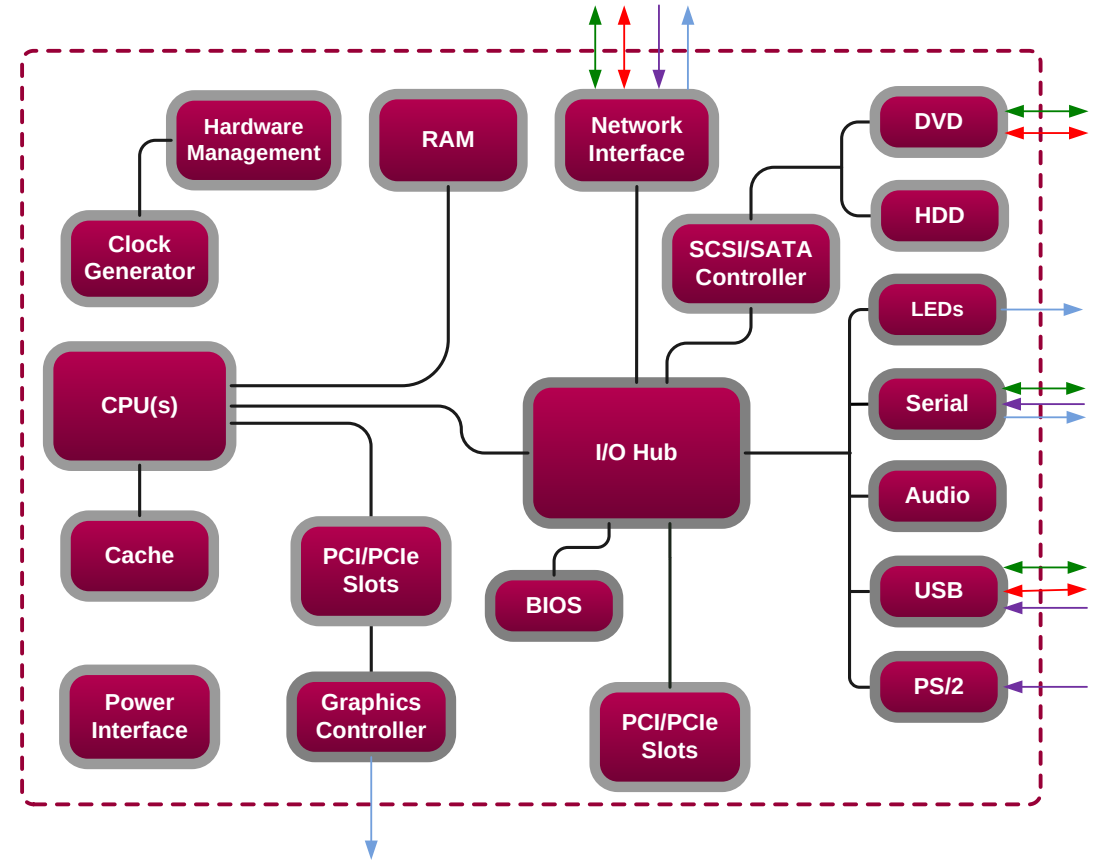

Figure 1 – Intel SR1530SH Server System Block Diagram

[2] RAM – Random Access Memory

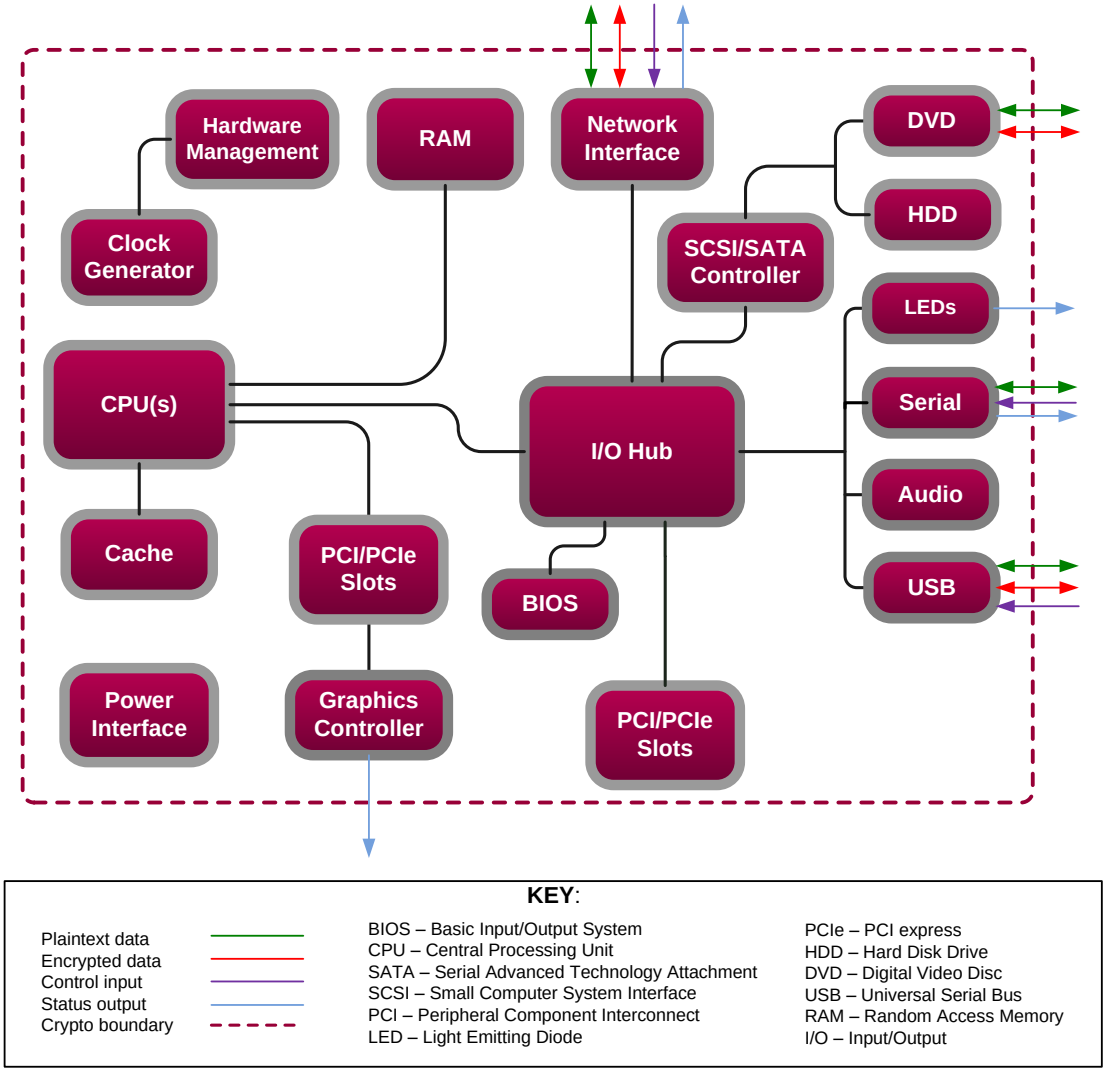

Figure 2 – Intel SR2625URLX Server System Block Diagram

2.2.2 Logical Cryptographic Boundary

Figure 3 shows a logical block diagram of the module executing in memory and its interactions with surrounding software components, as well as the module's logical cryptographic boundary. The cryptographic module consists of OpenSSL 1.0.1j and the FIPS Controller Daemon. These files are collectively shown as "McAfee Linux Cryptographic Module" in the diagram below. The module's services are designed to be called by applications developed by McAfee. The module's logical boundary is a contiguous perimeter that surrounds all memory-mapped functionality provided by the module when loaded and stored in the host platform's memory

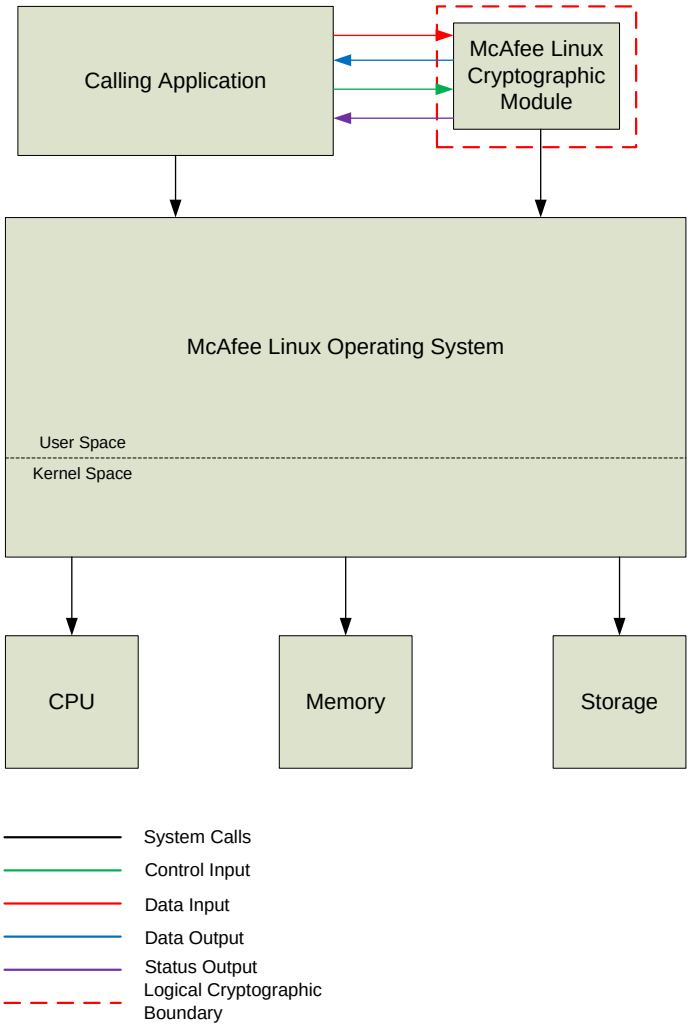

Figure 3 – McAfee Linux Cryptographic Module Logical Block Diagram

2.3 Module Interfaces

The module's logical interfaces exist at a low level in the software as an Application Programming Interface (API). Both the API and physical interfaces can be categorized into the following interfaces defined by FIPS 140-2:

- Data input
- Data output
- Control input
- Status output
- Power input

As a software module, the module has no physical characteristics. Thus, the module's manual controls, physical indicators, and physical and electrical characteristics are those of the Intel Server System. A mapping of the FIPS 140-2 logical interfaces, the physical interfaces, and the module interfaces while the module is executing on the Intel SR1530SH Server System can be found in Table 2 below.

Table 2 – FIPS 140-2 Logical Interface Mappings to the Intel SR1530SH Server System

FIPS Interface	Physical Interface	Module Interface (API)
Data Input	USB ports (3), network interfaces (2), serial ports (2), DVD drive (1)	The API calls that accept input data for processing through their arguments.
Data Output	USB ports (3), network interfaces (2), serial ports (1), DVD drive (1)	The API calls that return by means of their return codes or arguments generated or processed data back to the caller.
Control Input	USB ports (3), network interfaces (2), serial ports(1), PS/2 ports (2)	The API calls that are used to initialize and control the operation of the module.
Status Output	LED (7), network interfaces (2), serial ports (1), display port (1)	Return values for API calls. Values translated to either logical or physical status output.

A mapping of the FIPS 140-2 logical interfaces, the physical interfaces, and the module interfaces while the module is executing on the Intel SR2625URLX Server Systems can be found in Table 3 below.

Table 3 – FIPS 140-2 Logical Interface Mappings to the Intel SR2625URLX Server System

FIPS Interface	Physical Interface	Module Interface (API)
Data Input	USB ports (5), network interfaces (2), serial ports (2), DVD drive (1)	The API calls that accept input data for processing through their arguments.
Data Output	USB ports (5), network interfaces (2), serial ports (2), DVD drive (1)	The API calls that return by means of their return codes or arguments generated or processed data back to the caller.
Control Input	USB ports (5), network interfaces (2), serial ports (2)	The API calls that are used to initialize and control the operation of the module.
Status Output	LEDs (8), network interfaces (2), serial ports (2), display port (1)	Return values for API calls. Values translated to either logical or physical status output.

2.4 Roles and Services

The module supports role-based authentication. There are two roles in the module (as required by FIPS 140-2) that operators may assume: a Crypto Officer (CO) role and a User role. Each role and their corresponding services are detailed in the sections below. Please note that the keys and Critical security Parameters (CSPs) listed in the tables below indicate the type of access required using the following notation:

- R – Read: The CSP is read.
- W – Write: The CSP is established, generated, modified, or zeroized.
- X – Execute: The CSP is used within an Approved or Allowed security function or authentication mechanism.

2.4.1 Crypto Officer Role

The CO has the ability to initialize the module for first use, run on-demand self-tests, manage operator passwords, and zeroize keys. Descriptions of the services available to the CO role are provided in Table 4 below.

Table 4 – Crypto Officer Services

Service	Description	CSP and Type of Access
Initialize FIPS module configuration	Sets the FIPS flag to "1"; Performs integrity check and power-up self-tests.	None
Run self-tests on demand	Performs power-up self-tests	None
Zeroize key	Zeroizes and de-allocates memory containing sensitive data	All Keys and CSPs – W
Show Status	View the status of the module	None

2.4.2 User Role

The User role performs general security services, including cryptographic operations and other Approved security functions such as random number generation, encryption and decryption, message authentication, and signature generation and verification. Descriptions of the services available to the User role are provided in Table 5 below.

Table 5 – User Services

Service	Description	CSP and Type of Access
Generate random number	Returns the specified number of random bits to calling application	DRBG[3] Seed: WRX DRBG Entropy: RX
Generate message digest (SHS[4])	Compute and return a message digest using SHS algorithms	None
Generate keyed hash	Compute and return a message authentication code	HMAC[5] key – RX AES[6] GCM[7] Key – RX AES CMAC[8] Key: WRX Triple-DES[9] CMAC Key: WRX
Generate Symmetric Key	Generate keys and return the specified symmetric key (Triple-DES or AES)	AES key – W AES GCM Key – W AES GCM IV[10] – WRX XTS[11,12,13]-AES Key – W AES CMAC Key – W Triple-DES key – W Triple-DES CMAC Key – W

[3] DRBG – Deterministic Random Bit Generator
[4] SHS – Secure Hash Standard
[5] HMAC – (keyed-) Hash-based Message Authentication Code
[6] AES – Advance Encryption Service
[7] GCM – Galois Counter Mode
[8] CMAC – Cipher-based Message Authentication Code
[9] DES – Data Encryption Standard
[10] IV – Initialization Vector
[11] XTS - XEX-based tweaked-codebook mode with ciphertext stealing
[12] XEX – XOR-Encrypt-XOR
[13] XOR – Exclusive Or

Service	Description	CSP and Type of Access
Symmetric encryption	Encrypt plaintext using supplied key and algorithm specification (Triple-DES or AES)	AES key – RX AES GCM Key – RX AES GCM IV – RX XTS-AES Key – RX Triple-DES key – RX
Symmetric decryption	Decrypt ciphertext using supplied key and algorithm specification (Triple-DES or AES)	AES key – RX AES GCM Key – RX AES GCM IV – RX XTS-AES Key – RX Triple-DES key – RX
Generate asymmetric key pair	Generate and return the specified type of asymmetric key pair (RSA or DSA)	RSA[14] private/public key – W DSA[15] private/public key – W
RSA encapsulation	Encrypt plaintext using RSA public key (used for key transport)	RSA public key – RX

2.5 Physical Security

The McAfee Linux Cryptographic Module is a software module, which FIPS defines as a multi-chip standalone cryptographic module. As such, it does not include physical security mechanisms. Thus, the FIPS 140-2 requirements for physical security are not applicable.

2.6 Operational Environment

The operational environment of the McAfee Linux Cryptographic Module consists of an Intel Server System hardware platform executing MLOS v2.2.3. The module was tested and found to be FIPS 140-2 compliant in the following operational environments:

- MLOS v2.2.3 (64-bits) running on an Intel Celeron Processor operating on an Intel SR1530SH Server System hardware appliance
- MLOS v2.2.3 (64-bits) running on an Intel Xeon Processor operating on an Intel SR2625URLX Server System hardware appliance
- MLOS v2.2.3 (64-bits) on a VMware ESXi 5.0 hypervisor executing on an Intel SR2625URLX Server System hardware appliance using an Intel Xeon Processor

All keys, intermediate values, and other CSPs remain in the process space of a single operator. The OS provides process isolation for all user-mode processes through private virtual address spaces (private per process page tables), execution context (registers, program counters), and security context (handle table and token).

2.7 Cryptographic Key Management

The module implements the FIPS-Approved algorithms listed in Table 6 below.

[14] RSA – Rivest, Shamir, Adleman
[15] DSA – Digital Signature Algorithm

Table 6 – OpenSSL FIPS-Approved Algorithm Implementations

Algorithm	Certificate Number (Hardware Environment)	Certificate Number (Virtual Environment)
Encryption and Decryption		
AES[16] encryption/decryption in ECB[17], CBC[18], CTR[19], CFB1[20], CFB8, CFB128, and OFB[21]modes with 128-,192-, and 256-bit keys	3116	3117
AES GCM[22] encryption/decryption with 128-, 192-, and 256-bit keys	3116	3117
Triple-DES[23] encryption/decryption in ECB, CBC, CFB1, CFB8, CFB64, and OFB modes; 3-key	1787	1788
Key Generation; Signature Generation and Verification		
RSA[24] Key-pair Generation (FIPS 186-4) of 2048- and 3072-bit keys	1587	1588
RSA (FIPS 186-4) ANSI[25] X9.31, PKCS #1.5, PSS signature generation – 2048- and 3072-bit	1587	1588
RSA (FIPS 186-4) ANSI X9.31, PKCS #1.5, PSS signature verification – 1024-, 2048-, and 3072-bit (Legacy Use: 1024)	1587	1588
RSA (FIPS 186-2) ANSI X9.31, PKCS #1.5, PSS signature verification – 1024-, 2048-, 3072-, and 4096-bit (Legacy Use: 1024)	1587	1588
DSA[26] Key-pair Generation of 2048- and 3072-bit keys	900	901
DSA Signature Generation (2048- and 3072-bits) and Verification (1024-, 2048-, 3072-bits) (Legacy Use for Verification: 1024)	900	901
DSA PQG Generation (2048- and 3072-bits) and Verification (1024-, 2048-, 3072-bits) (Legacy Use for Verification: 1024)	900	901
Hashing and Message Authentication		
SHA[27]-1, SHA-224, SHA-256, SHA-384, SHA-512	2572	2573
HMAC[28] with SHA-1, SHA-224, SHA-256, SHA-384, SHA-512	1953	1954
AES CMAC[29] and CCM[30] Message Authentication with 128-, 192-, and 256-bit keys	3116	3117

[16] AES – Advance Encryption Standard
[17] ECB – Electronic Code Book
[18] CBC – Cipher Block Chaining
[19] CTR - Counter
[20] CFB – Cipher Feedback
[21] OFB – Output Feedback
[22] GCM – Galois Counter Mode
[23] DES – Data Encryption Standard
[24] RSA – Rivest, Shamir, Adleman
[25] ANSI – American National Standards Institute
[26] DSA – Digital Signature Algorithm
[27] SHA – Secure Hash Algorithm
[28] HMAC – (keyed-) Hash-based Message Authentication Code
[29] CMAC – Cipher-based Message Authentication Code
[30] CCM – Counter with CBC-MAC

Algorithm	Certificate Number (Hardware Environment)	Certificate Number (Virtual Environment)
AES GCM (GMAC[31]) message authentication with 128-, 192-, and 256-bit keys	3116	3117
Triple-DES CMAC; 3-key	1787	1788
Random Number Generation		
SP[32]800-90A HASH, HMAC, and CTR DRBG	627	628

The cryptographic module implements the TLS and SSH secure networking protocols. Each protocol implements a Key Derivation Function (KDF) listed in NIST SP 800-135rev1 and has been validated by the CMVP. There certificate numbers are provided in Table 7. The complete protocol implementations have not been reviewed or tested by the CAVP[33] and CMVP.

Outside the cryptographic module boundary the McAfee Linux Operating System employs a non-Approved Non-Deterministic Random Number Generator (NDRNG), which is used as an entropy source for seeding the Approved DRBGs listed in Table 6. Its use is allowed per FIPS 140-2 Implementation Guidance 7.11.

Table 7 – Network Protocol Component Validation

Algorithm	Certificate Number (Hardware Environment)	Certificate Number (Virtual Environment)
TLS[34] 1.0/1.1 and TLS 1.2 KDF using SHA 256 and SHA 384	378	379
SSH KDF using SHA-256, -384, and -512	378	379
SNMP KDF using SHA-1	378	379

The module employs the following key establishment methodologies:

- RSA (2048- and 3072-bit keys; key establishment methodology provides 112 or 128 bits of encryption strength).
- Diffie-Hellman (2048- and 3072-bit keys; key agreement; key establishment methodology provides 112 or 128 bits of encryption strength)

[31] GMAC – Gallois Message Authentication Code
[32] SP – Special Publication
[33] CAVP – Cryptographic Algorithm Validation Program
[34] TLS – Transport Layer Security

The module supports the CSPs listed below in Table 8.

Table 8 – Cryptographic Keys, Cryptographic Key Components, and CSPs

CSP	CSP Type	Generation / Input	Output	Storage	Zeroization	Use
AES key	AES128-, 192-, 256-bit key	Internally generated via approved DRBG; or Input via API call parameter	Output in plaintext via GPC[35] INT Path[36]	Keys are not persistently stored by the module	Unload module; API call; Remove Power	Encryption, decryption
AES GCM Key	AES 128- , 192-, or 256-bit key	Internally generated via approved DRBG; or Input via API call parameter	Output in plaintext via GPC INT Path	Keys are not persistently stored by the module	Unload module; API call; Remove Power	Encrypt and decrypt blocks of data; Keyed Message Authentication Code
AES GCM IV	96 bits of Random data	Internally Generated via approved DRBG	Never	Keys are not persistently stored by the module	Unload module; API call; Remove Power	IV input to AES GCM function
AES CMAC Key	AES 128-, 192-, or 256-bit key	Internally generated via approved DRBG; or Input via API call parameter	Output in plaintext via GPC INT Path	Keys are not persistently stored by the module	Unload module; API call; Remove Power	Keyed Message Authentication Code
TDES key	TDES 168 bit key	Internally generated via approved DRBG; or Input via API call parameter	Output in plaintext via GPC INT Path	Keys are not persistently stored by the module	Unload module; API call; Remove Power	Encryption, decryption

[35] GPC – General Purpose Computer
[36] GPC INT Path defined in Implementation Guidance (IG) 7.7

CSP	CSP Type	Generation / Input	Output	Storage	Zeroization	Use
TDES CMAC Key	TDES 168 bit key	Internally generated via approved DRBG; or Input via API call parameter	Output in plaintext via GPC INT Path	Keys are not persistently stored by the module	Unload module; API call; Remove Power	Keyed Message Authentication Code
HMAC key	HMAC SHA-1, SHA-224, SHA-256, SHA-384, or SHA-512 key	Internally generated via approved DRBG; or Input via API call parameter	Output in plaintext via GPC INT Path	Keys are not persistently stored by the module	Unload module; API call; Remove Power	Message Authentication with SHS
RSA private key	RSA 2048- or 3072-bit key	Internally generated via approved DRBG; or Input via API call parameter	Output in plaintext via GPC INT Path	Keys are not persistently stored by the module	Unload module; API call; Remove Power	Signature generation, decryption
RSA public key	RSA 1024-, 1536-, 2048-, 3072-, or 4096- bit key	Internally generated via approved DRBG; or Input via API call parameter	Output in plaintext via GPC INT Path	Keys are not persistently stored by the module	Unload module; API call; Remove Power	Signature verification, encryption
DSA private key	DSA 2048- or 3072-bit key	Internally generated via approved DRBG; or Input via API call parameter	Output in plaintext via GPC INT Path	Keys are not persistently stored by the module	Unload module; API call; Remove Power	Signature generation
DSA public key	DSA 1024, 2048, or 3072 bit key	Internally generated via approved DRBG; or Input via API call parameter	Output in plaintext via GPC INT Path	Keys are not persistently stored by the module	Unload module; API call; Remove Power	Signature verification

CSP	CSP Type	Generation / Input	Output	Storage	Zeroization	Use
DH[37] public components	Public components of DH protocol	Internally generated via approved DRBG; or Input via API call parameter	Output in plaintext via GPC INT Path	Keys are not persistently stored by the module	Unload module; API call; Remove Power	Used by host application
DH private component	Private exponent of DH protocol	Internally generated via approved DRBG; or Input via API call parameter	Output in plaintext via GPC INT Path	Keys are not persistently stored by the module	Unload module; API call; Remove Power	Used by host application
DRBG Seed	Random data[38]	Internally Generated	Never	Keys are not persistently stored by the module	Unload module; API call; Remove Power	Seeding material for SP 800-90A DRBGs
DRBG Entropy	Random data (75203 Bytes)	Externally Generated; Input via API	Never	Plaintext in volatile memory	Unload module; API call; Remove Power	Entropy material for SP 800-90A DRBGs
DRBG 'C' Value	Internal state value	Internally Generated	Never	Plaintext in volatile memory	Unload module; API call; Remove Power	Used for Hash_DRBG
DRBG 'V' Value	Internal state value	Internally Generated	Never	Plaintext in volatile memory	Unload module; API call; Remove Power	Used for Hash_DRBG, HMAC_DRBG, and CTR_DRBG
DRBG 'Key' Value	Internal state value	Internally Generated	Never	Plaintext in volatile memory	Unload module; API call; Remove Power	Used for HMAC_DRBG and CTR_DRBG

[37] DH – Diffie-Hellman
[38] Length of seed value is dependent on the implementation. Please refer to SP 800-90A DRBG specification.

2.8 Self-Tests

Cryptographic self-tests are performed by the module when the module is first powered up and loaded into memory as well as when a random number or asymmetric key pair is created. The following sections list the self-tests performed by the module, expected error status, and error resolution.

2.8.1 Software Integrity Tests

The module performs an HMAC SHA-256 integrity test on itself, and then the entirety of MLOS. If either the module or MLOS integrity tests fail, the FIPS Controller Daemon will log the error and halt all system processes, ensuring that data output from the module is inhibited. In order to resolve a cryptographic self-test error, the FIPS Controller Daemon will reboot the OS. If the error persists, the module must be reinstalled.

2.8.2 Power-Up Self-Tests

Power-up Self-tests are performed by the module when the module is first loaded into memory. The list of power-up self-tests may also be run on-demand when the CO reboots the Operating System. The module will perform the listed power-up self-tests to completion. During the execution of self-tests, data output from the module is inhibited.

If any of the self-tests fail, the FIPS Controller Daemon will log the error and halt all system processes, ensuring that data output from the module is inhibited. In order to resolve a cryptographic self-test error, the FIPS Controller Daemon will reboot the OS. If the error persists, the module must be reinstalled.

The McAfee Linux Cryptographic Module performs the following self-tests at power-up:

- OpenSSL AES Encrypt KAT
- OpenSSL AES Decrypt KAT
- OpenSSL AES CMAC KAT
- OpenSSL AES CCM Encrypt KAT
- OpenSSL AES CCM Decrypt KAT
- OpenSSL AES GCM Encrypt KAT
- OpenSSL AES GCM Decrypt KAT
- OpenSSL AES XTS Encrypt KAT
- OpenSSL AES XTS Decrypt KAT
- OpenSSL Triple-DES Encrypt KAT
- OpenSSL Triple-DES Decrypt KAT
- OpenSSL Triple-DES CMAC KAT
- OpenSSL RSA Sign KAT
- OpenSSL RSA Verify KAT
- OpenSSL DSA PCT[39]
- OpenSSL SHA-1 KAT
- OpenSSL HMAC with SHA-1, SHA-224, SHA-256, SHA-384, SHA-512 KAT[40]
- OpenSSL ANSI X9.31 KAT
- OpenSSL SP 800-90A DRBG KAT

2.8.3 Conditional Self-Tests

Conditional self-tests are performed by the module whenever a new random number is generated or when a new RSA or DSA key pair is generated. If any of the self-tests fail, the FIPS Controller Daemon will log the

[39] PCT – Pairwise Consistency Test
[40] OpenSSL satisfies the SHA-224, SHA-256, SHA-384, SHA-512 KAT requirements by performing these HMAC KATs

error and halt all system processes, ensuring that data output from the module is inhibited[41]. In order to resolve a cryptographic self-test error, the FIPS Controller Daemon will reboot the OS. If the error persists, the module must be reinstalled.

The McAfee Linux Cryptographic Module performs the following conditional self-tests:

- OpenSSL SP 800-90A DRBG Continuous Random Number Generator Test (CRNGT)
- OpenSSL SP 800-90A DRBG Reseed CRNGT
- OpenSSL RSA Key Generation PCT
- OpenSSL DSA Key Generation PCT

2.9 Mitigation of Other Attacks

This section is not applicable. The modules do not claim to mitigate any attacks beyond the FIPS 140-2 Level 1 requirements for this validation.

[41] The Continuous Random Number Generator Test in OpenSSL must fail twice in a row in order to produce this result. The second CRNGT is run immediately after the first CRNGT fails – no cryptographic processes are run in-between the first and second CRNGT and are not allowed to run again until the second CRNGT passes; All other cryptographic self-tests will only need to fail once.

3 Secure Operation

The McAfee Linux Cryptographic Module meets Level 1 requirements for FIPS 140-2. The sections below describe how to place and keep the module in operation.

3.1 Crypto Officer Guidance

This section describes the steps the CO must take to place the module into operation.

3.1.1 Secure Installation

The McAfee Linux Cryptographic Module is installed as part of the McAfee Linux Operating System installation. A CO shall use the following procedures to install the McAfee Linux Cryptographic Module:

1. Contact McAfee to obtain the MLOS version 2.2.3 build 5642 image.
2. Burn the image onto a blank DVD
3. Place the DVD into the disc drive of the device you would like to install MLOS
4. Reboot the device; specifying to boot from the disc drive
5. Follow the prompts provided by the installation wizard to successfully install the operating system

3.1.2 FIPS Module Configuration

In order to place the McAfee Linux Cryptographic Module in operation, the CO must perform the following steps from the MLOS CLI[42]:

1. Install FIPS RPMs
   # yum install mlos-fipserror mlos-fipshook mlos-fipstest mlosfipscheck fipscheck-devel
2. Update OpenSSL and libgcrypt (to contain FIPS hooks)
   # yum update openssl libgcrypt
3. Re-write the grub.conf file
   # /lib/fips/fipsgrub
4. Create integrity hmac (checksums)
   # /lib/fips/fipshmac
5. Reboot
   # reboot

3.2 User Guidance

The McAfee Linux Cryptographic Module does not input, output, or persistently store CSPs within its logical boundary. However, the module may store CSPs within the physical boundary of the host system on which it runs. Operators are responsible for providing persistent storage of the cryptographic keys and CSPs, and to ensure that keys are transmitted outside the physical cryptographic boundary in the appropriate manner.

The McAfee Linux Cryptographic Module includes libgcrypt within its cryptographic boundary. However, the user of the McAfee Linux Cryptographic Module must only use the cryptographic services as documented within this Security Policy.

[42] CLI – Command Line Interface

4 Acronyms

Table 9 provides definitions for the acronyms used in this document.

Table 9 – Acronyms

Acronym	Definition
AES	Advanced Encryption Standard
API	Application Programming Interface
ANSI	American National Standards Institute
BIOS	Basic Input/Output System
CAVP	Cryptographic Algorithm Validation Program
CBC	Cipher Block Chaining
CCM	Counter with CBC-MAC
CFB	Cipher Feedback
CLI	Command Line Interface
CMAC	Cipher-based Message Authentication Code
CMVP	Cryptographic Module Validation Program
CO	Cryptographic Officer
CPU	Central Processing Unit
CRNGT	Continuous Random Number Generator Test
CSE	Communications Security Establishment
CSP	Critical Security Parameter
CTR	Counter
DES	Data Encryption Standard
DH	Diffie-Hellman
DSA	Digital Signature Algorithm
DRBG	Deterministic Random Bit Generator
DVD	Digital Video Disc
ECB	Electronic Code Book
EMC	Electromagnetic Compatibility
EMI	Electromagnetic Interference
FIPS	Federal Information Processing Standard
GCM	Gallois Counter Mode
GPC	General Purpose Computer
HDD	Hard Disk Drive
HMAC	(Keyed-) Hash Message Authentication Code

Acronym	Definition
INT	Internal
IV	Initialization Vector
KAT	Known Answer Test
LCD	Liquid Crystal Display
LED	Light Emitting Diode
MLOS	McAfee Linux Operating System
NDRNG	Non-Deterministic Random Number Generator
NIST	National Institute of Standards and Technology
OFB	Output Feedback
OS	Operating System
PCI	Peripheral Component Interconnect
PCIe	PCI express
PCT	Pairwise Consistency Test
PKCS	Public Key Cryptography Standard
PRNG	Pseudo-Random Number Generator
PSS	Probabilistic Signature Scheme
RAM	Random Access Memory
RC	Rivest Cipher
RSA	Rivest, Shamir, Adleman
SATA	Serial Advanced Technology Attachment
SCSI	Small Computer System Interface
SHA	Secure Hash Standard
SHS	Secure Hash Standard
SP	Special Publication
TLS	Transport Layer Security
USB	Universal Serial Bus
VGA	Video Graphics Array
XEX	XOR-Encrypt-XOR
XOR	Exclusive Or
XTS	XEX-based tweaked-codebook mode with ciphertext stealing

Prepared by:
Corsec Security, Inc.

13921 Park Center Road
Suite 460
Herndon, VA 20171
United States of America

Phone: +1 (703) 267-6050
Email: info@corsec.com
http://www.corsec.com