FIPS 140-2 Non-Proprietary Security Policy

McAfee SIEM Cryptographic Module, Version 1.0

Document Version 1.4

December 12, 2013

Prepared For:

McAfee, Inc.

2821 Mission College Blvd.

Santa Clara, CA 95054

www.mcafee.com

Prepared By:

Apex Assurance Group, LLC

530 Lytton Avenue, Ste. 200

Palo Alto, CA 94301

www.apexassurance.com

Abstract

This document provides a non-proprietary FIPS 140-2 Security Policy for the McAfee SIEM Cryptographic Module, Version 1.0.

Table of Contents

List of Tables

List of Figures

1 Introduction

1.1 About FIPS 140

Federal Information Processing Standards Publication 140-2 — Security Requirements for Cryptographic Modules specifies requirements for cryptographic modules to be deployed in a Sensitive but Unclassified environment. The National Institute of Standards and Technology (NIST) and Communications Security Establishment of Canada (CSEC) Cryptographic Module Validation Program (CMVP) runs the FIPS 140 program. The CMVP accredits independent testing labs to perform FIPS 140 testing; the CMVP also validates test reports for modules meeting FIPS 140 validation. *Validated* is the term given to a module that meets the FIPS 140 criteria.

More information is available on the CMVP website at http://csrc.nist.gov/groups/STM/cmvp/index.html.

1.2 About this Document

This non-proprietary Cryptographic Module Security Policy for the McAfee SIEM Cryptographic Module, Version 1.0 from McAfee provides an overview of the product and a high-level description of how it meets the security requirements of FIPS 140-2. This document contains details on the module's cryptographic keys and critical security parameters. This Security Policy concludes with instructions and guidance on running the module in a FIPS 140-2 mode of operation.

The McAfee SIEM Cryptographic Module, Version 1.0 may also be referred to as the "module" in this document.

1.3 External Resources

The McAfee website (http://www.mcafee.com) contains information on McAfee products. The Cryptographic Module Validation Program website (http://csrc.nist.gov/groups/STM/cavp/validation.html) contains links to the FIPS 140-2 certificate and McAfee contact information.

1.4 Notices

This document may be freely reproduced and distributed in its entirety without modification.

1.5 Acronyms

The following table defines acronyms found in this document:

Acronym	Term
AES	Advanced Encryption Standard
ANSI	American National Standards Institute
API	Application Programming Interface
CMVP	Cryptographic Module Validation Program
CO	Crypto Officer
CSEC	Communications Security Establishment Canada
CSP	Critical Security Parameter
DES	Data Encryption Standard
DH	Diffie-Hellman
DSA	Digital Signature Algorithm
EMC	Electromagnetic Compatibility
EMI	Electromagnetic Interference
FCC	Federal Communications Commission
FIPS	Federal Information Processing Standard
GPC	General Purpose Computer
GUI	Graphical User Interface
HMAC	(Keyed-) Hash Message Authentication Code
KAT	Known Answer Test
MAC	Message Authentication Code
MD	Message Digest
NIST	National Institute of Standards and Technology
OS	Operating System
PKCS	Public-Key Cryptography Standards
PRNG	Pseudo Random Number Generator
PSS	Probabilistic Signature Scheme
RNG	Random Number Generator
RSA	Rivest, Shamir, and Adleman
SIEM	Security Information and Event Management
SHA	Secure Hash Algorithm
SSL	Secure Sockets Layer
TDES	Triple Data Encryption Algorithm
TLS	Transport Layer Security
USB	Universal Serial Bus

Table 1 – Acronyms and Terms

2 McAfee SIEM Cryptographic Module, Version 1.0

2.1 Cryptographic Module Specification

The module, the McAfee SIEM Cryptographic Module, Version 1.0, is a software module that provides cryptographic services required by the McAfee SIEM appliances. The module is a multi-chip standalone embodiment installed on McAfee SIEM hardware.

All operations of the module occur via calls from the McAfee SIEM applications and their respective internal daemons/processes. As such there are no untrusted services calling the services of the module.

2.1.1 Validation Level Detail

The following table lists the level of validation for each area in FIPS 140-2:

FIPS 140-2 Section Title	Validation 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
Electromagnetic Interference / Electromagnetic Compatibility	1
Self-Tests	1
Design Assurance	1
Mitigation of Other Attacks	N/A

Table 2 – Validation Level by FIPS 140-2 Section

2.1.2 Approved Cryptographic Algorithms

The module's cryptographic algorithm implementations have received the following certificate numbers from the Cryptographic Algorithm Validation Program:

Algorithm	CAVP Certificate	
	32-bit	**64-bit**
AES	2229	2230
Triple-DES	1395	1396
DSA	690	691
ECDSA	343	344
RNG	1115	1116
RSA	1141	1142
SHS	1917	1918

HMAC	1357	1358
CVL	33	34

Table 3 – FIPS-Approved Algorithm Certificates

Keys generated by this module using these algorithms are limited to a maximum of 160 bits of encryption strength.

2.1.3 Non-Approved Cryptographic Algorithms

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

- Diffie-Hellman with key sizes of 1024-10000 bits (key agreement and key establishment methodology provides between 80 and 160 bits of encryption strength)
- MD5 (for use in TLS only)
- RSA with a key size of 2048 bits (key wrapping; key establishment provides 112 bits of encryption strength)

2.2 Module Interfaces

The figure below shows the module's physical and logical block diagram:

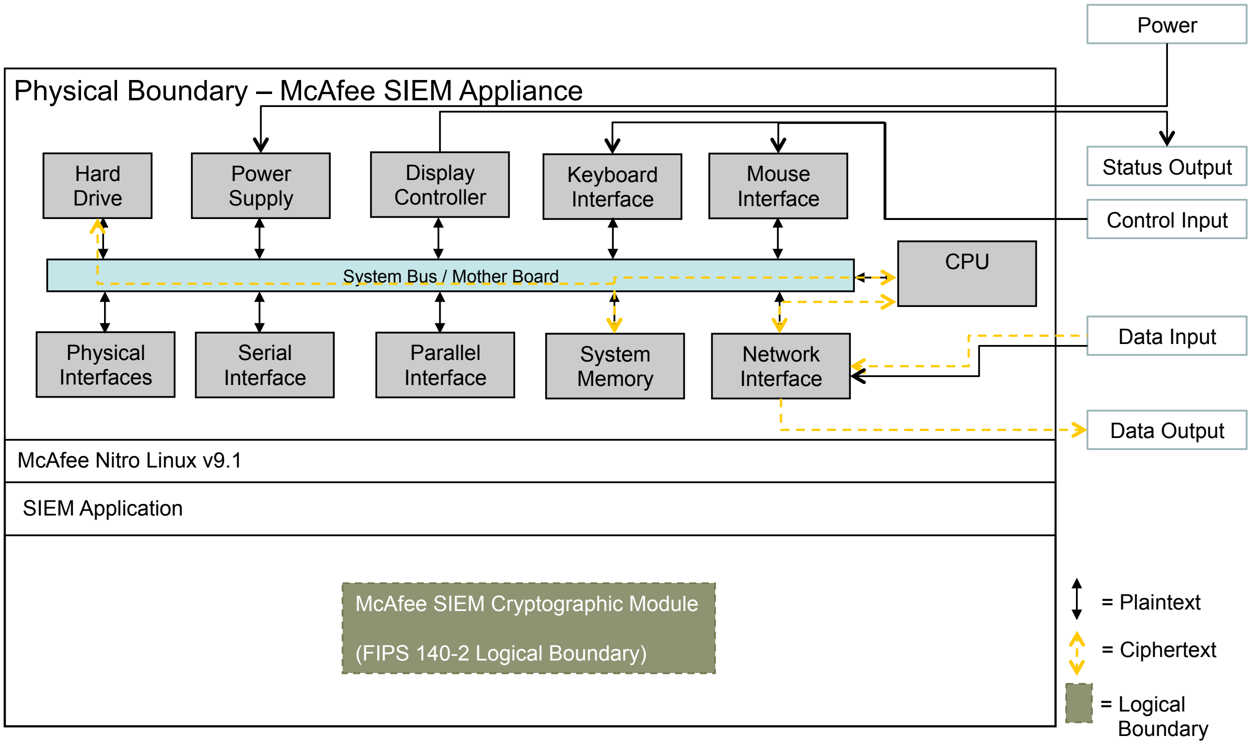

Figure 1 – Module Boundary and Interfaces Diagram

The interfaces (ports) for the physical boundary could include the computer keyboard port, CDROM drive, floppy disk, mouse, network port, parallel port, USB ports, monitor port and power plug. When operational, the module does not transmit any information across these physical ports because it is a software cryptographic module. Therefore, the module's interfaces are purely logical and are provided through the Application Programming Interface (API) that a calling daemon can operate. The logical interfaces expose services that applications directly call, and the API provides functions that may be called by a referencing application (see Section 2.3 – Roles, Services, and Authentication for the list of available functions). The module distinguishes between logical interfaces by logically separating the information according to the defined API.

The API provided by the module is mapped onto the FIPS 140- 2 logical interfaces: data input, data output, control input, and status output. Each of the FIPS 140- 2 logical interfaces relates to the module's callable interface, as follows:

FIPS 140-2 Interface	Logical Interface	Module Physical Interface
Data Input	Input parameters of API function calls	Network Interface
Data Output	Output parameters of API function calls	Network Interface
Control Input	API function calls	Keyboard Interface, Mouse Interface
Status Output	For FIPS mode, function calls returning status information and return codes provided by API function calls.	Display Controller
Power	None	Power Supply

Table 4 – Logical Interface / Physical Interface Mapping

As shown in Figure 1 – Module Boundary and Interfaces Diagram and Table 6 – Module Services and Descriptions, the output data path is provided by the data interfaces and is logically disconnected from processes performing key generation or zeroization. No key information will be output through the data output interface when the module zeroizes keys.

2.3 Roles, Services, and Authentication

The module supports a Crypto Officer and a User role. The module does not support a Maintenance role. The supported role definitions are as follows:

Role	Services

Role	Services
User	Encryption, Decryption (symmetric and public/private), Key Transport, Digital Signature, Symmetric Key Generation, TLS, Key agreement (TLS and Asymmetric), Keyed Hash, Message Digest, Random Numbers, Show Status, Self-test, and Zeroize
Crypto Officer	All User services plus Module Initialization.

Table 5 – Role Descriptions

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

2.3.1 Operator Services and Descriptions

The module supports services that are available to users in the various roles. The following table shows the services available to the various roles and the access to cryptographic keys and CSPs resulting from services:

Service	Roles	CSP / Algorithm	Permission
Symmetric encryption/decryption	User, Crypto Officer	AES Key, TDES Key	User and CO: read/write/execute
Key transport	User, Crypto Officer	RSA Private Key	User and CO: read/write/execute
Digital signature	User, Crypto Officer	RSA Private Key, DSA Private Key	User and CO: read/write/execute
Symmetric key generation	User, Crypto Officer	AES Key, TDES Key	User and CO: read/write/execute
TLS	User, Crypto Officer	AES Key, TDES Key, RSA Public Key, RSA Private Key, HMAC Key	User and CO: read/write/execute
TLS Key Agreement	User, Crypto Officer	AES Key, TDES Key, RSA Public Key, RSA Private Key, HMAC Key, DH Private Key, DH Public Key	User and CO: read/write/execute
Asymmetric key generation	User, Crypto Officer	RSA Private Key, DSA Private Key	User and CO: read/write/execute
Keyed Hash (HMAC)	User, Crypto Officer	HMAC Key HMAC SHA-1	User and CO: read/write/execute
Message digest (SHS)	User, Crypto Officer	SHA-1	User and CO: read/write/execute
Random number generation	User, Crypto Officer	PRNG Seed and Seed Key	User and CO: read/write/execute
Show status	User, Crypto Officer	none	User and CO: execute
Module initialization	Crypto Officer	none	CO: execute

Service	Roles	CSP / Algorithm	Permission
Integrity test	User, Crypto Officer	Integrity Key (HMAC SHA-1)	User and CO: read/execute
On-Demand Self Test	User, Crypto Officer	All CSPs	User and CO: read/write/execute
Zeroize	User, Crypto Officer	All CSPs	User and CO: read/write/execute

Table 6 – Module Services and Descriptions

2.3.2 Operator Authentication

As required by FIPS 140-2, there are two roles (a Crypto Officer role and User role) in the module that operators may assume. As allowed by Level 1, the module does not support authentication to access services.

2.4 Physical Security

This section of requirements does not apply to this module. The module is a software-only module and does not implement any physical security mechanisms.

2.5 Operational Environment

The module was tested on a McAfee SIEM Appliance (single-user mode) with Intel Xeon processor running McAfee Nitro OS 9.1 (32-bit and 64-bit).

Compliance is maintained for other versions of the respective operating system's family where the binary is unchanged.

The platform(s) used during testing met Federal Communications Commission (FCC) FCC Electromagnetic Interference (EMI) and Electromagnetic Compatibility (EMC) requirements for business use as defined by 47 Code of Federal Regulations, Part15, Subpart B. FIPS 140-2 validation compliance is maintained when the module is operated on other platforms where the binary is unchanged.

2.6 Cryptographic Key Management

The table below provides a complete list of Critical Security Parameters used within the module:

Keys and CSPs	Storage Locations	Storage Method	Input Method	Output Method	Zeroization	Access
AES Key	RAM	Plaintext	API call parameter	None	`OPENSSL_cleanse()` power cycle	CO: RWD U: RWD

Keys and CSPs	Storage Locations	Storage Method	Input Method	Output Method	Zeroization	Access
TDES Key	RAM	Plaintext	API call parameter	None	OPENSSL_cleanse() power cycle	CO: RWD U: RWD
RSA Public Key	RAM	Plaintext	API call parameter	None	OPENSSL_cleanse() power cycle	CO: RWD U: RWD
RSA Private Key	RAM	Plaintext	API call parameter	None	OPENSSL_cleanse() power cycle	CO: RWD U: RWD
DSA Public Key	RAM	Plaintext	API call parameter	None	OPENSSL_cleanse() power cycle	CO: RWD U: RWD
DSA Private Key	RAM	Plaintext	API call parameter	None	OPENSSL_cleanse() power cycle	CO: RWD U: RWD
ECDSA Public Key	RAM	Plaintext	None	None	OPENSSL_cleanse() power cycle	CO: RWD U: RWD
ECDSA Private Key	RAM	Plaintext	None	None	OPENSSL_cleanse() power cycle	CO: RWD U: RWD
HMAC Key	RAM	Plaintext	API call parameter	None	OPENSSL_cleanse() power cycle	CO: RWD U: RWD
PRNG Seed	RAM	Plaintext	API call parameter	None	OPENSSL_cleanse() power cycle	CO: RWD U: RWD
PRNG Seed Key	RAM	Plaintext	API call parameter	None	OPENSSL_cleanse() power cycle	CO: RWD U: RWD
Integrity Key	RAM	Plaintext	None	None	OPENSSL_cleanse() power cycle	CO: RWD U: RWD
DH Private Key	RAM	Plaintext	API call parameter	None	OPENSSL_cleanse() power cycle	CO: RWD U: RWD
DH Public Key	RAM	Plaintext	API call parameter	None	OPENSSL_cleanse() power cycle	CO: RWD U: RWD

R = Read W = Write D = Delete

Table 7 – Module Keys/CSPs

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.

2.6.1 Random Number Generation

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

The module accepts results from /dev/urandom (which is outside the cryptographic boundary) as an entropy source of random numbers for PRNG seeds.

The module performs continual tests on the random numbers it uses to ensure that the seed and seed key input to the Approved PRNG do not have the same value. The module also performs continual tests on the output of the approved PRNG to ensure that consecutive random numbers do not repeat.

The module generates cryptographic keys whose strengths are modified by available entropy. No assurance of the minimum strength of generated keys.

2.6.2 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.

2.6.3 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.
Key/CSP Zeroization

The memory occupied by keys is allocated by `openssl mem.c` and `OPENSSL_malloc()/cleanse()`. The application is responsible for calling the appropriate destruction functions from the API. The destruction functions then overwrite the memory occupied by keys with zeros and deallocates the memory with the `OPENSSL_cleanse()` call.

2.7 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. In the event of a self-test error, the module will log the error and will halt. The module must be initialized into memory to resume function.

The following sections discuss the module’s self-tests in more detail.

2.7.1 Power-On Self-Tests

Power-on self-tests are executed automatically when the module is loaded into memory. The `FIPS_mode_set()` function verifies the integrity of the runtime executable using a HMAC SHA-1 digest computed at build time. If the digest match, the power-up self-tests are 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.

TYPE	DETAIL
Known Answer Tests	• AES encrypt/decrypt • PRNG • RSA • TDES encrypt/decrypt • ECC CDH
Pair-wise Consistency Tests	• DSA • ECDSA
Module integrity	• HMAC-SHA-1

Table 8 – Power-On Self-Tests

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.

2.7.2 Conditional Self-Tests

The module implements the following conditional self-tests upon key generation, or random number generation (respectively):

TYPE	DETAIL
Pair-wise Consistency Tests	• DSA • ECDSA • RSA
Continuous RNG Test	Performed on approved PRNG

Table 9 – Conditional Self-Tests

2.7.3 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 module is being initialized, it will automatically call `FIPS_mode_set(1)` during this initialization. Prior to this invocation the Module is uninitialized.

The `FIPS_mode_set()` function verifies the integrity of the runtime executable using a HMAC SHA-1 digest which is computed at build time. If this computed HMAC SHA-1 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.

2.8 Mitigation of Other Attacks

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

[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

3 Guidance and Secure Operation

This section describes how to configure and initialize the module for FIPS-Approved mode of operation. When configured and initialized per this Security Policy, the module will only operate in the FIPS Approved mode of operation.

3.1 Crypto Officer Guidance

3.1.1 Software Installation

The module is included with the McAfee NitroSecurity solution and is not available for direct download.

The NitroSecurity software is configured to use the module only in FIPS mode as follows:

- The first time you log on to the system you will be prompted to select whether or not you want the system to operate in FIPS mode. Keep in mind that once this selection is made it cannot be changed. Detail on selecting FIPS mode can be located in the McAfee Enterprise Security Manager Interface 9.1.3 Users Guide in section 3.3 (p.38) Selecting FIPS Mode.

- The McAfee NitroSecurity engineering team is responsible for ensuring the source files that comprise the McAfee SIEM Cryptographic Module, Version 1.0 are built into the NitroSecurity solution.

3.2 User Guidance

3.2.1 General Guidance

Please see the FIPS relevant sections of the McAfee Enterprise Security Manager Interface 9.1.3 Users Guide. Throughout the Guide, sections specify where FIPS mode removes or modifies the functionality of a feature. Appendix C of the McAfee Enterprise Security Manager Interface 9.1.3 Users Guide includes a list of features that should not be enabled when operating in FIPS mode.