McAfee, Inc.

McAfee Core Cryptographic Module (user) 1.0

FIPS 140-2 Non-Proprietary Security Policy

Level 1 Validation

Document revision 015, August 2014

McAfee, Inc.
2821 Mission College Blvd.
Santa Clara, CA 95054
888.847.8766
www.mcafee.com

Prepared for McAfee, Inc. by

Rycombe Consulting Limited
http://www.rycombe.com
+44 1273 476366

Contents

Figures

1 Introduction

This section identifies the cryptographic module; describes the purpose of this document; provides external references for more information; and explains how the document is organized.

1.1 Identification

Module Name	McAfee, Inc. McAfee Core Cryptographic Module (user)
Module Version	1.0
Software Version	1.0

1.2 Purpose

This is the non-proprietary FIPS 140-2 Security Policy for the McAfee Core Cryptographic Module (user), also referred to as "the module" within this document. This Security Policy details the secure operation of McAfee Core Cryptographic Module (user) as required in Federal Information Processing Standards Publication 140-2 (FIPS 140-2) as published by the National Institute of Standards and Technology (NIST) of the United States Department of Commerce.

1.3 References

For more information on McAfee products please visit: http://www.mcafee.com. For more information on NIST and the Cryptographic Module Validation Program (CMVP), please visit http://csrc.nist.gov/groups/STM/cmvp/index.html.

1.4 Document Organization

This Security Policy document is one part of the FIPS 140-2 Submission Package. This document outlines the functionality provided by the module and gives high-level details on the means by which the module satisfies FIPS 140-2 requirements. With the exception of this Non-Proprietary Security Policy, the FIPS 140-2 Submission documentation may be McAfee, Inc. proprietary or otherwise controlled and releasable only under appropriate non-disclosure agreements. For access to these documents, please contact McAfee, Inc.

The various sections of this document map directly onto the sections of the FIPS 140-2 standard and describe how the module satisfies the requirements of that standard.

1.5 Document Terminology

TERM	DESCRIPTION
AES	Advanced Encryption Standard
AES-NI	Advanced Encryption Standard New Instructions. Seven instructions for accelerating different sub-steps of the AES algorithm included in some Intel and AMD microprocessors.

API	Application Programming Interface
CAVP	Cryptographic Algorithm Validation Program
CMVP	Cryptographic Module Validation Program
CSP	Critical Security Parameters
DLL	Dynamic Link Library
DLM	Dynamic Link Module (a type of DLL used in the Pre-boot environment)
DRBG	Deterministic Random Bit Generator
FIPS	Federal Information Processing Standard
GPC	General Purpose Computer
GUI	Graphical User Interface
IPC	Inter-process communication
MBR	Master Boot Record
McAfee ePO	McAfee ePolicy Orchestrator: A McAfee software installation to allow configuration and management of a McAfee product deployment
OS	Operating System
Pre-boot environment	The operating environment of a GPC before the operating system is loaded
RSA	An algorithm for public-key cryptography. Named after Rivest, Shamir and Adleman who first publicly described it.
SHA	Secure Hash Algorithm
SP	Security Policy
Storage Media	Any media for which Cryptographic Module protection in the form of data encryption is required. Storage Media include internal and external hard drives, memory sticks and floppy disks.
TCP/IP	Transmission Control Protocol/Internet Protocol
TLS	Transport Layer Security
XML	Extensible Markup Language

Figure 1 Document terminology

2 McAfee Core Cryptographic Module (user)

This section provides the details of how the module meets the FIPS 140-2 requirements.

2.1 Overview

The module provides cryptographic services to McAfee, Inc. products.

The module is packaged differently depending on its operational environment.

2.2 Module Specification

2.2.1 Hardware, Software and Firmware components

There are no specific hardware or firmware requirements for the module. The module is a software-only module which resides on a General Purpose Computer (see Figure 4).

It is packaged as five distinct binary images, one for each of the following operating environments:

File Name	Operating Environment
MFECCF[32|64]*aa*.dll	Microsoft Windows
MFECCF[32|64]*aa*.dlm	PC BIOS Pre-boot environment
MFECCF[32|64]*aa*.dylink	Apple MacOS
MFECCF[32|64]*aa*.efi	UEFI (PC Pre-boot environment)
MFECCF[32|64]*aa*.efi	EFI (Apple Mac Pre-boot environment)

Figure 2 Module binary image

Note: *aa* are alphanumeric product identifiers, such as MFECCF64DE.dll for Drive Encryption and MFECCF32FF.dll for Files and Folders.

Although there is a common core of functionality between these five images, within this group there are distinct variants. The differences in implementation relate to:

1. The collection of the entropy used to seed the SP800-90 compliant DRBG
2. An AES implementation that is not present in the pre-boot builds.
3. The Macintosh variants do not support the Symmetric Encryption service MCCi_crypto_disk_algs API

All variants use the Intel RdRand instruction to seed the SP800-90 compliant DRBG if this functionality is available on the CPU. If it is not available, then the pre-boot variant uses entropy derived from user-generated inputs such as mouse movements and keyboard key presses, whereas the Windows/MacOS variant uses entropy derived from network activity.

The low-level disk handler AES implementation is only applicable to the non-preboot environment.

Except for in the distinct areas clearly described above, the various modules are identical and in those cases, the term “the module” applies equally to each variant.

2.2.2 Cryptographic Boundary

The cryptographic boundary of the module is the case of the General Purpose Computer (GPC) on which it is installed. See Figure 4. The module is a software module running in a pre-boot, Windows, or Mac OS operating environment on a general-purpose computer. The processor of this platform executes all software. All software components of the module are persistently stored within the device and, while executing, are stored in the device local RAM.

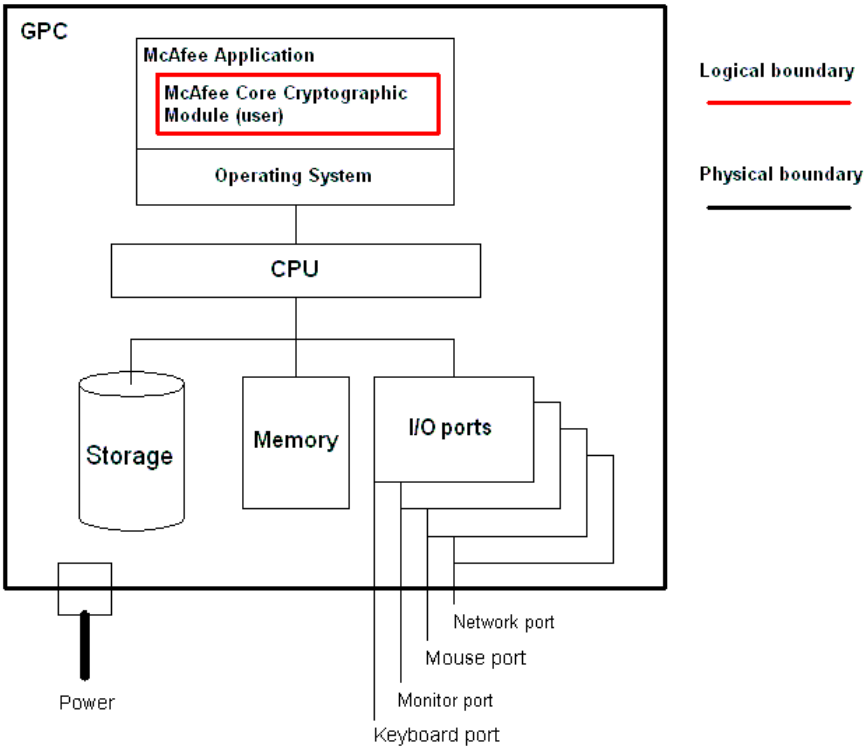

Figure 3 Block Diagram of the Cryptographic Boundary (Pre-boot environment)

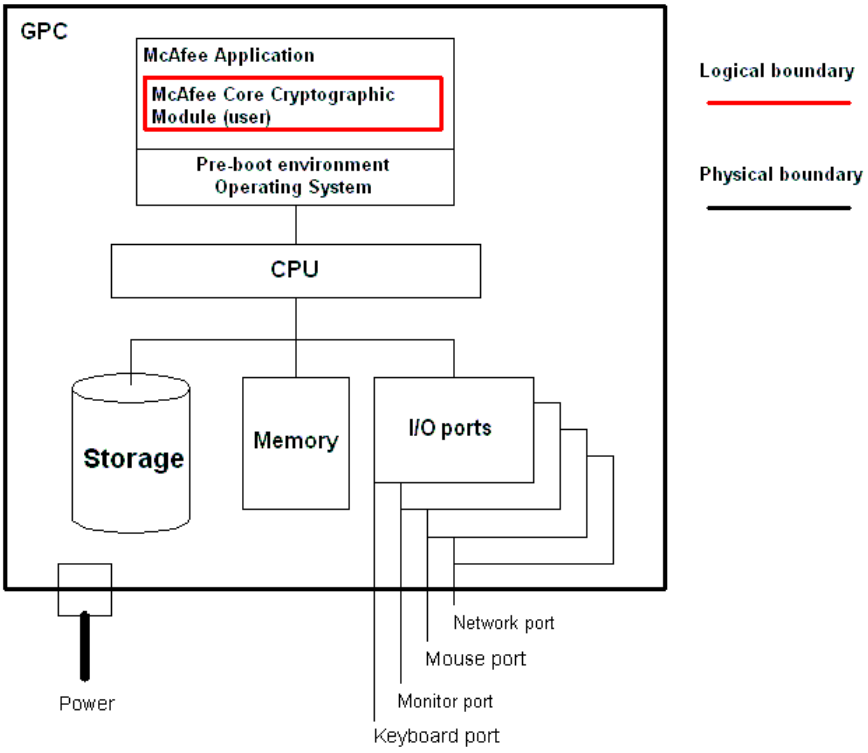

Figure 4 Block Diagram of the Cryptographic Boundary (Windows / MacOS environments)

2.2.3 Scope of Evaluation

The cryptographic module meets the overall requirements applicable to Level 1 security of FIPS 140-2, with Design Assurance at Level 3.

Security Requirements Section	Level
Cryptographic Module Specification	1
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	3
Mitigation of Other Attacks	N/A

Figure 5 Security Level specification per individual areas of FIPS 140-2

2.2.4 Cryptographic Algorithms

2.2.4.1 Approved algorithms

The following table provides details of the approved algorithms that are included within the module:

Algorithm Type	Algorithm	CAVP Certificate	Use
Hashing	SHA-1	#2181 (SHA-1 and SHA-256)	SHA-1 is a constituent of the non-approved PKCS#5 algorithm.
	SHA-256	#2287 (SHA-256)	SHA-256 is part of the DRBG and also used in the module integrity test.
Message authentication code	HMAC	#1604 #1605	Module integrity testing and in the random number generator.
Random number generation	HMAC SHA256 DRBG	#394	Symmetric and Asymmetric key generation
Symmetric key	AES-256 – ECB, CBC, CFB8 and CFB128. Encrypt and decrypt	#2591 #2592 #2593 #2755	Service provided to encrypt and decrypt blocks of data. As it is a user service and so may also be used to encrypt private keys to provide key wrapping (cert #2591 only).

Figure 6 Approved Algorithms

Notes:

There are four implementations of AES-256 in the module, one to support each of the following:

- MCCi_crypto_sym_encryptor service
- MCCi_crypto_disk_algs service interface “get_interface”, a copy of the MFECCF*aa* module algorithms – this can run on processors with or without AES-NI capability. However, it will use AES-NI instructions if run on AES-NI enabled processors.
- MCCi_crypto_disk_algs service interface “get_hook_code”, a copy of the algorithm code to support the cryptographic algorithm requirements of the low-level disk handler (INT 13 disk hook, Windows only).
- MCCi_crypto_disk_algs service interface “secure_crypt_blocks”, an implementation where the data key is stored in model-specific registers (MSRs) rather than in RAM. This interface is only applicable when these registers are present on the processor.

2.2.4.2 Non-approved algorithms allowed in approved mode

RSA encrypt/decrypt is used for key wrapping giving 112 bits of security strength. This service is non-approved but allowed.

The entropy used to seed the random number generator is an NDRNG that is a non-Approved but allowed algorithm.

2.2.4.3 Non-approved algorithms

The following non-approved algorithms are included within the module:

- PKCS#5 (a password hashing algorithm using SHA-1), offered as a non-approved service to users of the module.

During operation, the module can switch service by service between an Approved mode of operation and a non-Approved mode of operation. The module will transition to the non-Approved mode of operation when the above non-Approved security functions is utilized in lieu of an Approved one. The module can transition back to the Approved mode of operation by utilizing an Approved security function.

PKCS#5 is not allowed for use in the FIPS Approved mode of operation. When this algorithm is used, the module is not operating in the FIPS Approved mode of operation. CSPs shall not be shared between the Approved and non-Approved modes of operation.

2.2.5 Components excluded from the security requirements of the standard

There are no components excluded from the security requirements of the standard.

2.3 Physical ports and logical interfaces

The module is classified as a multi-chip standalone module for FIPS 140-2 purposes. The module’s physical boundary is that of the device on which it is installed. The device shall be running a supported operating system (OS) and supporting all standard interfaces, including keys, buttons and switches, and data ports.

The module provides its logical interfaces via Application Programming Interface (API) calls. This logical interface exposes services (described in section 2.4.2) that the User and operating system utilize directly.

The logical interfaces provided by the module are mapped onto the FIPS 140-2 logical interfaces: data input, data output, control input, and status output as follows:

FIPS 140-2 Logical Interface	Module Mapping
Data Input	Parameters passed to the module via API calls
Data Output	Data returned from the module via API calls
Control Input	API Calls and/or parameters passed to API calls
Status Output	Information received in response to API calls
Power Interface	There is no separate power or maintenance access interface beyond the power interface provided by the GPC itself

Figure 7 Module Interfaces

2.4 Roles, Services and Authentication

2.4.1 Roles

The Cryptographic Module implements both a Crypto Officer role and a User role. Roles are assumed implicitly upon accessing the associated services. Section 2.4.2 summarizes the services available to each role.

Role	Description
Crypto Officer	The administrator of the module having full configuration and key management privileges.
User	General User of the module

Figure 8 Roles

2.4.2 Services

Most of the services provided by the module are provided via access to API calls using interfaces exposed by the module.

However, some of the services, such as power-up module integrity testing are performed automatically and so have no function API, but do provide status output.

Service	API	Description	Service Input	Service Output
Asymmetric encryption/ decryption	MCCi_crypto_asym_ encryptor	Provides RSA encryption and decryption. Note: The asymmetric encryption service shall not be used for	Encryption: Public key and plaintext data passed in. Decryption:	Encryption: Encrypted data passed out. Decryption:

SERVICE	API	DESCRIPTION	SERVICE INPUT	SERVICE OUTPUT
		encrypting/decrypting bulk data. It shall only be used specifically to wrap cryptographic keys using RSA.	Private key and encrypted data passed in.	Plaintext data passed out. Service can also output algorithm status information.
	MCCi_crypto_asym_encryptor_var_pad	Provides RSA encryption but allows the padding to be specified.	Encryption: Public key and plaintext data passed in. Decryption: Private key and encrypted data passed in.	Encryption: Encrypted data passed out. Decryption: Plaintext data passed out. Service can also output algorithm status information.
Key generation	MCCi_crypto_asym_key_gen	Provides RSA key generation. The RSA keys are calculated from large numbers generated by the FIPS approved DRBG seeded by a non-approved NDRNG.	None	Service methods exist to generate a key pair and to separately output the public key and the private key.
	MCCi_crypto_asym_key_gen2	Provides RSA key generation but allows the keys to be encoded in various formats. The RSA keys are calculated from large numbers generated by the FIPS approved DRBG seeded by a non-approved NDRNG.	Format required for keys	Service methods exist to generate a key pair and to separately output the public key and the private key.
	MCCi_rand_source	Generates symmetric keys and accepts seed data to add to the random pool.	For seeding: Entropy data and a count of the	For Seeding: None.

SERVICE	API	DESCRIPTION	SERVICE INPUT	SERVICE OUTPUT
		Entropy data is used to seed the DRBG which is used to generate the symmetric keys.	number of entropy bytes: For generation: A count of the number of bytes required.	For generation: A block of random data.
Symmetric encryption/ decryption	MCCi_crypto_enc_data	Utility class to hold encrypted data returned by the symmetric encryptor.	Data buffer, size of data and algorithm type ID	Data buffer, size of data, and algorithm type ID
	MCCi_crypto_sym_encryptor	Provides symmetric algorithm encryption and decryption. Supports AES256 (CBC and CFB128 modes).	Encryption: Algorithm ID, key, plaintext data, Initial Value (IV) Decryption: Key, encrypted data, Initial Value (IV)	Encryption: Encrypted data Decryption: Decrypted data Status: Algorithm information and key size
	MCCi_crypto_disk_algs (Not supported by Macintosh variants)	Provides access to a copy of the MFECCF*aa* symmetric disk encryption algorithms (statically linked into this, the MFECCF[32|64]*aa* module). Supports only FIPS AES 256.	None	Status: Results of Disk Driver encryption algorithm self-tests. Interface pointer providing access to Disk Driver encryption algorithms.
Hashing	MCCi_crypto_digest_gen	Allows hashing of arbitrary data. Supports SHA1 and SHA256.	Data and algorithm	Digest of the data.

SERVICE	API	DESCRIPTION	SERVICE INPUT	SERVICE OUTPUT
Self-tests	MCCi_crypto_self_tests	Runs all the KATs on the algorithms exported by MfeCryptoLib library.	None	Service outputs a Boolean result code, TRUE if all of the self-tests passed, otherwise FALSE.
	N/A	The power-up software integrity test is run automatically when the module is loaded and started. The continuous random number generator test is also run automatically.	None	If passes, writes the string "Passed" to the Integrity Status registry key, otherwise writes the string "Failed" to this registry key and initiates a stop error.
Show Status	N/A	Status is returned in response to individual service API calls and at the completion of the self-tests.	None	Module status.

Figure 9 User Services

SERVICE	API	DESCRIPTION	SERVICE INPUT	SERVICE OUTPUT
Installation	N/A	The module is deployed as part of a McAfee, Inc. product installation.	None	Installed module
Uninstallation	N/A	The module is uninstalled during the uninstallation of the product that deployed the module.	None	Uninstalled module
Key Zeroization	N/A	Keys are zeroized using the zeroization procedure. This is described in section 2.7.4.	None	All keys zeroized.

Figure 10 Crypto Officer Services

SERVICE	API	DESCRIPTION	SERVICE INPUT	SERVICE OUTPUT
PKCS#5	MCCi_crypto_pkcs5_hash_gen	Allows hashing of a password	Password	Digest of the password

Figure 11 Non-Approved Services

2.4.3 Authentication

The module has been evaluated at FIPS 140-2 level 1 and no claims are made for authentication.

2.5 Physical Security

The Cryptographic Module is a software-only cryptographic module and therefore the physical security requirements of FIPS 140-2 do not apply.

2.6 Operational Environment

The Cryptographic Module has been tested on and found to be conformant with the requirements of FIPS 140-2 overall Level 1 on the following GPC platforms:

Platform	CPU	RdRand[1]	AES-NI[2]	Operating System (all tested in single-user mode)	AES Register Implementation[3]
Dell E5510	Intel Core i3	N/A		McAfee Endpoint Encryption Preboot OS	
Dell E6320	Intel Core i5	N/A	X	McAfee Endpoint Encryption Preboot OS	
Dell E6410	Intel Core i7	N/A	X	McAfee Endpoint Encryption Preboot OS	
Dell Inspiron 3520	Intel Core i3	X		Windows 8 running in 64-bit UEFI mode	
Lenovo W530	Intel Core i5	X	X	Windows 8 running in 64-bit UEFI mode	
Lenovo Yoga	Intel Core i7	X	X	Windows 8 running in 64-bit UEFI mode	
Samsung 700T	Intel Core i5			Windows 8 running in 32-bit UEFI mode	
Dell Latitude 10	Intel Atom			Windows 8 running in 32-bit UEFI mode	
MacBook	Intel Core 2 Duo			Macintosh running EFI Preboot	
MacPro	Intel Xeon			Macintosh running EFI Preboot	
MacBook Air	Intel Core i3	X	X	Macintosh running EFI Preboot	
Mac Mini	Intel Core i5	X	X	Macintosh running EFI Preboot	
MacBook Pro	Intel Core i7	X	X	Macintosh running EFI Preboot	
Dell E5510	Intel Core i3			Windows XP 32-bit	
Dell E5510	Intel Core i3			Windows 7 64-bit	
Lenovo Yoga	Intel Core i7	X	X	Windows 7 64-bit	
Lenovo Yoga	Intel Core i7	X	X	Windows 8 64-bit	
Dell Latitude 10	Intel Atom			Windows 8 32-bit	
MacBook	Intel Core 2 Duo			MacOS X Lion v10.7	
MacPro	Intel Xeon			MacOS X Mountain Lion v10.8	
MacBook Air	Intel Core i3	X	X	MacOS X Mountain Lion v10.8	

Platform	CPU	RdRand[1]	AES-NI[2]	Operating System (all tested in single-user mode)	AES Register Implementation[3]
Mac Mini	Intel Core i5	X	X	MacOS X Lion v10.7	
MacBook Pro	Intel Core i7	X	X	MacOS X Mountain Lion v10.8	
Dell E6320	Intel Core i5		X	Windows Vista 32-bit	
Dell E6410	Intel Core i7		X	Windows Vista 64-bit	
Dell E6320	Intel Core i5		X	Windows 7 32-bit	
Lenovo W530	Intel Core i5	X	X	Windows 8 32-bit	
Lenovo W530	Intel Core i5	X	X	Windows 8 64-bit	
Intel UBHB2SISQ	Intel Core i5	X	X	Windows 8 64-bit	X
Lenovo Thinkpad 2	Intel Atom			Windows 8 32-bit	X
Intel UBHB2SISQ	Intel Core i5	X	X	Windows 8 running in 64-bit UEFI mode	X
Lenovo Thinkpad 2	Intel Atom			Windows 8 running in 32-bit UEFI mode	X

Figure 12 Operating Platforms

The module is also capable of running on the following platforms but has not been tested during this evaluation and no compliance is being claimed on these platforms:

- Windows Server 2008 (32-bit and 64-bit) with SP1
- Windows Server 2008 R2 (64-bit only)
- Windows 2003 Server (32-bit only) with SP2
- Windows 2003 Server R2 (32-bit only) with SP2

Notes:

1. RdRand is an instruction for returning random numbers from a random number generator built into the microprocessor. See section 2.2.1 for details on how this is used in the module.
2. AES-NI (the Intel Advanced Encryption Standard (AES) New Instructions (AES-NI)) is an extension to the x86 instruction set architecture for microprocessors from Intel and AMD. The purpose of the instruction set is to improve the speed of applications performing encryption and decryption using AES.
3. AES Register Implementation: The module can use MSRs (model-specific registers) to store the AES System key so that it is stored within the processor and not in system RAM. The decision to use the MSRs in this way is made by the application using the module and it is this application that is responsible for loading the key into the MSRs.

The cryptographic module runs in the thread context of the calling application. This provides it with protection from all other processes, preventing access to all keys, intermediate key generation values, and other CSPs.

The task scheduler and architecture of the operating system maintain the integrity of the cryptographic module.

The module supports only one single user and only one operator can have access to the device that contains the module at a time.

2.7 Cryptographic Key Management

2.7.1 Random Number Generators

The module contains an approved HMAC SHA256 SP800-90 approved DRBG.

2.7.2 Key Generation

Keys generated internally are generated by the HMAC SHA256 DRBG seeded by system entropy. Checks are made to ensure that the quality of the entropy remains high enough to be used to seed the DRBG.

The entropy comes from a promiscuous socket. This is used to provide sampling points for discrete high speed counters in the Mac OS, Windows and Preboot environments. Only the least significant bits of these counters are sampled and statistical tests produce results showing that this provides random data of sufficient entropy.

2.7.3 Key Table

The following tables list all of the keys and CSPs within the module, describe their purpose, and describe how each key is generated, entered and output, stored and destroyed.

KEY	PURPOSE
Data Key	To encrypt and decrypt data using the symmetric encryption services.
Public Key	To encrypt data or keys using one of the asymmetric encryption services.
Private Key	To decrypt data or keys previously encrypted by the Public Key.
HMAC DRBG CSPs: Key, V, seed and entropy	These are variables used internally by the HMAC DRBG that are required by Implementation Guidance 14.5 to be listed in the Cryptographic Module Security Policy document. There is no initial seed, but the algorithm is reseeded from a non-approved NDRNG.

Figure 13 Module Cryptographic Keys and CSPs

KEY	KEY LENGTH/STRENGTH	GENERATION/ESTABLISHMENT	STORAGE LOCATION
Data Key	AES 256 bit	Generated using Key generation service	Not stored within the module.
Public Key	RSA 2048 bit	Generated using Key generation service	Not stored within the module.
Private Key	RSA 2048 bit	As per Public Key	Not stored within the module.
HMAC DRBG "Key" CSP	512 bits	Initial value of 64 bytes all set to "0x00"	Not stored within the module.
HMAC DRBG "V" CSP	512 bits	Initial value of 64 bytes all set to "0x01"	Not stored within the module.
HMAC DRBG seed and entropy CSPs	2048 bits	non-approved NDRNG	Not stored within the module.

Figure 14 Key Table part 1

KEY	ARE KEYS SUPPLIED ENCRYPTED OR PLAINTEXT?	ENTRY/OUTPUT	DESTRUCTION
Data Key	Plaintext	Used as key in symmetric encryption service.	Zeroized using the key zeroization service.
Public Key	Plaintext	Passed as a parameter to asymmetric encryption service.	Zeroized using the key zeroization service.
Private Key	Plaintext	Passed as a parameter to asymmetric encryption service.	Zeroized using the key zeroization service.
HMAC DRBG "Key" CSP	N/A	N/A	Zeroized using the key zeroization service.
HMAC DRBG "V" CSP	N/A	N/A	Zeroized using the key zeroization service.
HMAC DRBG seed and entropy CSPs	N/A	N/A	Zeroized using the key zeroization service.

Figure 15 Key Table part 2

2.7.4 Key Destruction

All key material managed by the module can be zeroized using the key zeroization procedure. This requires uninstallation of the cryptographic module and reformatting the hard drive on which it was installed.

The CO should uninstall the module and then reformat the hard drive on which it was installed and overwrite it at least once. The operator should remain present during this process. This process meets the requirements of IG 7.9 for key zeroization.

Reformatting the hard drive will remove any encrypted or public keys from the hard disk.

In this way all key material and CSPs are zeroized. There are no user-accessible plaintext keys or CSPs in the module.

2.7.5 Access to Key Material

The following table shows the access that an operator has to specific keys or other critical security parameters when performing each of the services relevant to his/her role.

Access Rights

Blank N/A
R Read
W Write
U Use

KEY	DATA KEY	PUBLIC KEY	PRIVATE KEY	HMAC DRBG KEY	HMAC DRBG V
User Services					
Asymmetric encryption		WU	WU		
Key generation	R	R	R	U	U
Symmetric encryption	RWU				
Hashing					
Self-tests					
Show Status					
Crypto Office Services					
Installation	U				
Uninstallation	U				
Key Zeroization	W	W	W	W	W

Figure 16 Access to keys by services

Note: Key zeroization zeroes all keys and CSPs, this is a “write” operation in that all keys are overwritten with zeroes.

2.8 Self-Tests

The module implements both power-up and conditional self-tests as required by FIPS 140-2. The following two sections outline the tests that are performed.

2.8.1 Power-up self-tests

Object	Test
SHA-1	Known answer tests
SHA-256	Known answer tests
HMAC-SHA-256	Known answer tests
AES-256	A separate encryption and decryption known answer test for each of the AES implementations within the module
HMAC DRBG	Known answer test
Module software	HMAC SHA-256 Integrity Check
RSA	Known answer test

Figure 17 Power-up self-tests

Note: The module performs a separate known answer test for each of its SHA implementations.

2.8.2 Conditional self-tests

Event	Test	Consequence of Failure
Module requests a random number from the FIPS Approved SP800-90 DRBG	A continuous random number generator test	Random number is not generated and module enters an error state.
Entropy is supplied to the FIPS approved SP800-90 DRBG	A continuous random number generator test on the entropy NDRNG	Entropy is not added and module enters an error state
RSA key pair generation	Pairwise consistency test	Key is not generated and module enters an error state.

Figure 18 Conditional self-tests

The following error message is returned in the event of a self-test error: MCC_exception("Cryptographic module is in an error state and cannot be accessed", E_MCC_GEN_SELF_TEST_FAILED).

2.9 Design Assurance

McAfee, Inc. employ industry standard best practices in the design, development, production and maintenance of all of its products, including the FIPS 140-2 module.

This includes the use of an industry standard configuration management system that is operated in accordance with the requirements of FIPS 140-2, such that each configuration item that forms part of the module is stored with a label corresponding to the version of the module and that the module and all of its associated documentation can be regenerated from the configuration management system with reference to the relevant version number.

Design documentation for the module is maintained to provide clear and consistent information within the document hierarchy to enable transparent traceability between corresponding areas throughout the document hierarchy, for instance, between elements of this Cryptographic Module Security Policy (CMSP) and the design documentation.

Guidance appropriate to an operator's Role is provided with the module and provides all of the necessary assistance to enable the secure operation of the module by an operator, including the Approved security functions of the module.

Delivery of the Cryptographic Module to customers from the vendor is via the internet. When a customer purchases a license to use the Cryptographic Module software, they are issued with a grant number as part of the sales process. This is then used as a password to allow them to download the software that they have purchased. The delivery channel is protected using secured sockets. Once the Cryptographic Officer has downloaded the cryptographic module, it is his responsibility to ensure its secure delivery to the users that he is responsible for.

2.10 Mitigation of Other Attacks

The module does not mitigate any other attacks.