# FIPS 140-2 Non-Proprietary Security Policy

---

## McAfee Agent Cryptographic Module (Version 1.0)

Document Version 1.4

July 19, 2011

*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 Agent 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 products meeting FIPS 140 validation. *Validated* is the term given to a product that is documented and tested against 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 Agent 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 Agent 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 the full line of products from McAfee. The Cryptographic Module Validation Program website (http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/1401val2011.htm) 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 |
| CAVP | Cryptographic Algorithm Validation Program |
| CBC | Cipher Block Chaining |
| CSEC | Communications Security Establishment of Canada |
| CSP | Critical Security Parameter |
| DTR | Derived Testing Requirement |
| ePO | ePolicy Orchestrator |
| FIPS | Federal Information Processing Standard |
| GPC | General Purpose Computer |
| GPOS | General Purpose Operating System |
| KAT | Known Answer Test |
| NIST | National Institute of Standards and Technology |
| RSA | Rivest Shamir Adelman |
| SHA | Secure Hashing Algorithm |

**Table 1 – Acronyms and Terms**

# 2 McAfee Agent Cryptographic Module (Version 1.0)

## 2.1 Product Overview

The McAfee Agent provides common communication functionality between McAfee ePolicy Orchestrator and all of McAfee's endpoint products that run under the ePO framework. McAfee ePolicy Orchestrator is a scalable management framework for centralized policy management and enforcement of McAfee's security products and the systems on which they reside.

More information can be found at http://www.mcafee.com/us/products/epolicy-orchestrator.aspx.

## 2.2 Cryptographic Module Specification

The module, the McAfee Agent Cryptographic Module (Version 1.0), provides the McAfee Agent application with cryptographic functionality. The module is a software-only module installed on a multi-chip standalone device, such as a General Purpose Computer running a General Purpose Operating System and provides cryptographic services to the McAfee Agent application.

The module is a uniquely identifiable set of libraries built into the McAfee Agent application. All operations of the module occur via calls from the Agent application and its internal daemons, and all calls are authenticated via digital signature. As such there are no untrusted services or daemons calling the services of the module. No security functions outside the cryptographic module provide FIPS-relevant functionality to the module.

Once configured for FIPS mode of operation (see the Guidance and Secure Operation section), the module cannot be placed into a non-FIPS mode.

The boundary is composed of the following files:

- mfecryptc.dll
- ccme_base.dll
- ccme_ecc.dll
- ccme_eccaccel.dll
- cryptocme2.dll
- cryptocme2.sig
- mfecryptc.sig

## 2.3 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 | 2 |
| 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 | 3 |
| Mitigation of Other Attacks | N/A |

**Table 2 – Validation Level by DTR Section**

The "Mitigation of Other Attacks" section is not relevant as the module does not implement any countermeasures towards special attacks.

## 2.4 Cryptographic Algorithms

### 2.4.1 Algorithm Implementation Certificates

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

| Algorithm Type | Algorithm | Standard | CAVP Certificate | Use |
|---|---|---|---|---|
| Asymmetric Key | RSA 2048-bit | X9.31, PKCS#1 V.1.5 | 203 | Sign / verify operations<br><br>Module Integrity |
| | DSA 1024-bit | FIPS 186-3 | 199 | Verify legacy data |
| Hashing | SHA-1, SHA-256 | FIPS 180-3 | 560 | Digital signature generation and verification (SHA-256)<br><br>Verification of legacy data (SHA-1)<br><br>User password hashing |

| Algorithm Type | Algorithm | Standard | CAVP Certificate | Use |
|---|---|---|---|---|
| Random Number Generation | FIPS 186-2 PRNG (Change Notice 1-with and without the mod q step) | FIPS 186-2 | 270 | Random Number Generation |
| Symmetric Key | AES 128-bit and 256-bit in CBC and ECB mode | FIPS 197 | 490 | Data encryption/ decryption |
| | 3DES mode CBC mode | FIPS 46-3 | 501 | Decryption of legacy data |

**Table 3 – FIPS-Approved Algorithm Certificates Crypto C ME**[1]

### 2.4.2 Non-Approved Algorithms

The module implements the following non-FIPS approved algorithms:

- Software-based entropy mechanism
    - This RNG is used only as a seeding mechanism to the FIPS-approved PRNG.

## 2.5 Module Interfaces

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

[1] Note this implementation has received FIPS 140-2 Level 1 validation certificate #828: http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/1401val2007.htm#828

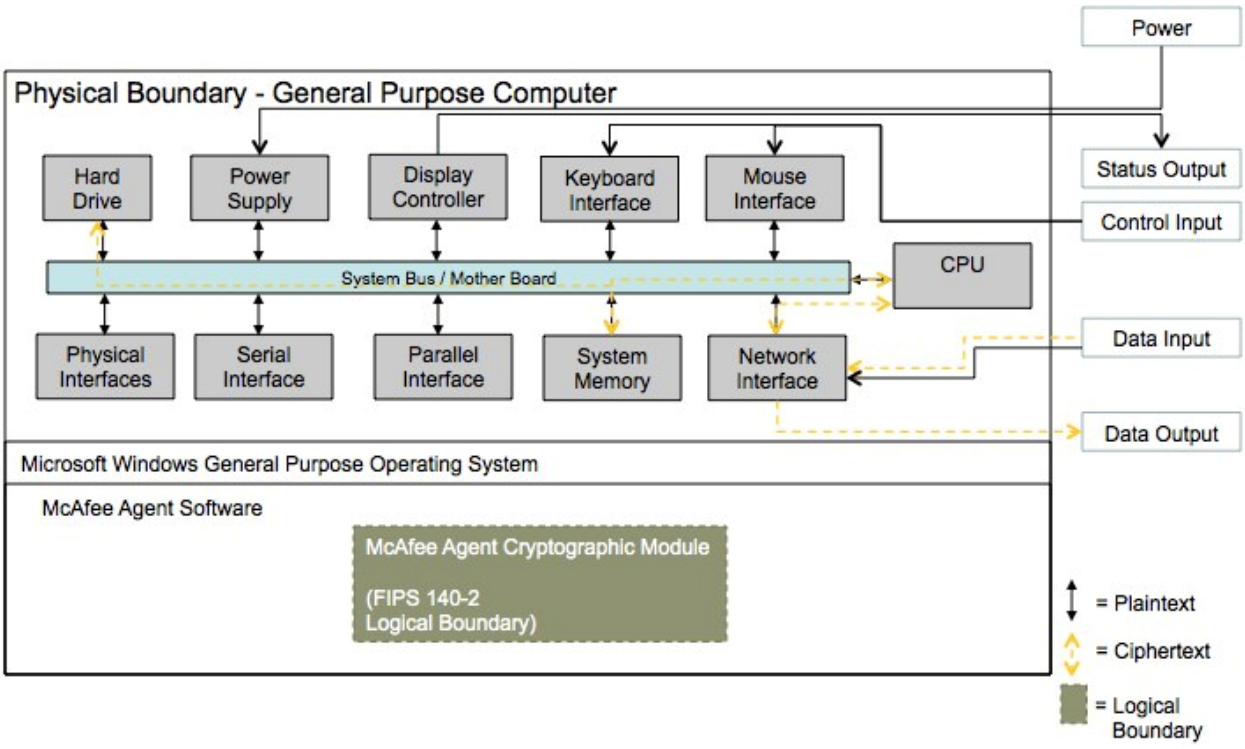


**Figure 1 – Module Interfaces Diagram**

The interfaces (ports) for the physical boundary 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/service 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 2.6 – Roles, Services, and Authentication for the list of available functions).

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 | Ethernet/Network port |
| Data Output | Output parameters of API function calls | Ethernet/Network port |
| Control Input | API function calls | Keyboard and mouse |

| FIPS 140-2 Interface | Logical Interface | Module Physical Interface |
|---|---|---|
| Status Output | For FIPS mode, function calls returning status information and return codes provided by API function calls. | Monitor |
| Power | None | Power supply/connector |

**Table 4 – Logical Interface / Physical Interface Mapping**

The module's logical interfaces are provided only through the Application Programming Interface (API) that a calling daemon can operate. The module distinguishes between logical interfaces by logically separating the information according to the defined API.

As shown in Figure 1 – Module Interfaces Diagram 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.

## 2.6 Roles, Services, and Authentication

The module supports a Crypto Officer and a User role as specified in the following section. The module does not support a Maintenance role.

### 2.6.1 Operator Services and Descriptions

The services available to the User and Crypto Officer roles in the module are as follows:

| Service | Description | Service Input/Output (API) | Key/CSP Access | Roles |
|---|---|---|---|---|
| Configure | Initializes the module for FIPS mode of operation | `R_FIPS140_library_init()`<br>`PRODUCT_FIPS_140_MODE_RESOURCE_LIST()`<br>`R_FIPS140_get_default()`<br>`R_FIPS140_get_mode()`<br>`R_FIPS140_get_info()` | User public key | User |
| Initialization | Configures and initializes the module for FIPS mode of operation | `mc_get_fips140_resource_list`<br>`mc_get_fips140_ssl_resource_list`<br>`mc_LIB_set_officer_role`<br>`mc_LIB_set_keystore_path`<br>`mc_LIB_set_user_sigfile_path`<br>`mc_LIB_load`<br>`mc_LIB_unload` | Integrity public key<br><br>User public key | User |

| Service | Description | Service Input/Output (API) | Key/CSP Access | Roles |
|---|---|---|---|---|
| Key management | Allows import, generation and storage of keys | `mc_RKEY_asym_new`<br>`mc_RKEY_sym_import`<br>`mc_RKEY_asym_import`<br>`mc_RKEY_get_info`<br>`mc_RKEY_free`<br>`mc_RKEY_to_persist`<br>`mc_RKEY_from_persist`<br>`mc_RKEY_destroy`<br>`Mc_RKEY_persist_exists` | User public key | User |
| Encrypt and Decrypt | Allows encryption and decryption of data with keys accessed using the key management services | `mc_CIPHER_new`<br>`mc_CIPHER_free`<br>`mc_CIPHER_get_info`<br>`mc_CIPHER_encrypt_buffer`<br>`mc_CIPHER_encrypt_data_update`<br>`mc_CIPHER_encrypt_data_final`<br>`mc_CIPHER_decrypt_buffer`<br>`mc_CIPHER_decrypt_data_update`<br>`mc_CIPHER_decrypt_data_final` | TDES key<br>AES key<br><br>User public key | User |
| Sign and Verify | Allows generation and verification of digital signatures | `mc_SIGN_new`<br>`mc_SIGN_free`<br>`mc_SIGN_sign_buffer`<br>`mc_SIGN_sign_data_update`<br>`mc_SIGN_sign_data_final`<br>`mc_SIGN_verify_buffer`<br>`mc_SIGN_verify_data_update`<br>`mc_SIGN_verify_data_final` | RSA Private key<br><br>RSA Public key<br><br>DSA Public key<br><br>User public key | User |
| Random number generation | Allows generation of random number | `mc_PRNG_gen_random_number` | FIPS 186-2 PRNG Seed<br><br>FIPS 186-2 PRNG Seed Key<br><br>User public key | User |

| Service | Description | Service Input/Output (API) | Key/CSP Access | Roles |
|---|---|---|---|---|
| Self test | Performs various self tests | `mc_LIB_perform_all_self_tests`<br>`mc_LIB_software_integrity_test`<br>`mc_LIB_algorithm_test`<br>`mc_LIB_critical_functions_test` | Integrity public key<br><br>User public key | User |
| All key zeroization | All keys listed in the keys sheet | `mc_RKEY_zeroize` | All Keys | User |
| Status function | Shows the status of module | `mc_LIB_get_status`<br>`mc_LIB_status_log`<br>`mc_LIB_status_get_info`<br>`mc_lib_status_new` | User public key | User |

**Table 5 – Authenticated Module Services and Descriptions for Crypto C ME Implementation**

| Service | Description | Service Input/Output (API) | Key/CSP Access | Roles |
|---|---|---|---|---|
| Reboot | Restart module or application | Restart module or application | None | Crypto Officer |
| Procedural Zeroization | Zeroize keys stored on disk in keystore | RAM keys are zeroized when the Operating System clears the process memory, static Keys stored on disk are zeroized according to IG 7.9 by uninstalling the module and formatting the disk | All Keys | Crypto Officer |

**Table 6 – Unauthenticated Module Services and Descriptions for Crypto C ME Implementation**

### 2.6.2 Operator Authentication

The module supports Level 2 requirements for authentication, which defines role-based authentication. The module verifies the digital signatures of calling daemons prior to the allowing access to any module services. The signature is RSA 2048-bit key with SHA-256 hash signature. Since this key has 112-bits of security strength the probability of a successful random attempt is $1/2^{112}$, which is less than 1/1,000,000. Assuming a scripted attack of 60 attempts in one minute, the probability of a success with multiple consecutive attempts in a one-minute period is $60/2^{112}$ which is less than 1/100,000.

The module contains User authentication data in the form of the public key but does not contain CO authentication data. The User Services require authentication, which is performed by the module as

described above. The Crypto Officer services do not require authentication as they are not security relevant functions. The Reboot and Procedural Zeroization services do not affect the security of the module; these services do not create, disclose, or substitute cryptographic keys or CSPs, nor do they utilize any Approved security functions.

The module does not permit an operator to change roles.

## 2.7 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.

## 2.8 Operational Environment

The module operates on a general-purpose computer (GPC) running a general-purpose operating system (GPOS). The module was tested on the following:

- Microsoft Windows Server 2003 on Intel Core2 Duo

For FIPS purposes, the module is running on a platform in single user mode and does not require any additional configuration to meet the FIPS requirements.

The GPC(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 versions of the supported GPOS running in single user mode, assuming that the requirements outlined in NIST IG G.5 are met.

## 2.9 Cryptographic Key Management

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

| Key/CSP Name | Description / Use | Generation | Storage | Establishment / Export | Interface | Privileges |
|---|---|---|---|---|---|---|
| Integrity public key | 2048-bit RSA public key for verifying the integrity of crypto module | Generated at build time via FIPS-approved PRNG | **Storage**: RAM, on disk in plaintext<br><br>**Type:** Static<br><br>**Association**: The system is the one and only owner. Relationship is maintained by the operating system via protected memory. | **Agreement**: NA<br><br>**Entry**: NA<br><br>**Output**: NA | Module configuration | CO<br>R W D<br><br>User<br>R W D |
| User public key | 2048-bit RSA public key for authenticating User role | Generated at build time via FIPS-approved PRNG | **Storage**: on disk plaintext<br><br>**Type:** Static<br><br>**Association**: The system is the one and only owner. Relationship is maintained by the operating system via protected memory. | **Agreement**: NA<br><br>**Entry**: NA<br><br>**Output**: NA | Module configuration | CO<br>R W D<br><br>User<br>R W D |

| Key/CSP Name | Description / Use | Generation | Storage | Establishment / Export | Interface | Privileges |
|---|---|---|---|---|---|---|
| RSA Private Key | 2048-bit RSA private key data, for use in specific services | Generated via FIPS-approved PRNG | **Storage**: RAM, on disk in keystore in plaintext<br><br>**Type:** Ephemeral<br><br>**Association**: User specified identifier for disk to memory association, OS maintained association via protected memory in RAM | **Agreement**: NA<br><br>**Entry**: NA<br><br>**Output**: key handle output to application, and persisted to disk | Sign<br><br>Establish Session | CO<br>R W D<br><br>User<br>R |
| RSA Public Key | 2048-bit RSA public key data, for use in specific service) | Generated via FIPS-approved PRNG | **Storage**: RAM, on disk in keystore in plaintext<br><br>**Type:** Ephemeral<br><br>**Association**: User specified identifier for disk to memory association, OS maintained association via protected memory in RAM | **Agreement**: NA<br><br>**Entry**: NA<br><br>**Output**: key handle output to application, and persisted to disk | Verify<br><br>Establish Session | CO<br>R W D<br><br>User<br>R |
| TDES key | General purpose 168-bit TDES key for data decryption of legacy data | Passed by calling process | **Storage**: on disk plaintext<br><br>**Type:** Ephemeral<br><br>**Association**: The system is the one and only owner. Relationship is maintained by the operating system via protected memory. | **Agreement**: NA<br><br>**Entry**: NA<br><br>**Output**: NA | Decrypt | CO<br>R W D<br><br>User<br>R |

| Key/CSP Name | Description / Use | Generation | Storage | Establishment / Export | Interface | Privileges |
|---|---|---|---|---|---|---|
| AES Key | AES CBC 128 or 256-bit key for encryption / decryption of session traffic | Generated via FIPS-approved PRNG | **Storage**: RAM plaintext<br>**Type:** Ephemeral<br>**Association**: The system is the one and only owner. Relationship is maintained by the operating system via protected memory. | **Agreement**: NA<br>**Entry**: NA<br>**Output**: Key handle from API request is output only to the application | Decrypt<br>Encrypt | CO<br>R W D<br>User<br>R |
| DSA Public Key | McAfee public repository DSA 1024-bit key for verifying signatures | Generated at build time | **Storage**: on disk plaintext<br>**Type:** Ephemeral<br>**Association**: The system is the one and only owner. Relationship is maintained by the operating system via protected memory. | **Agreement**: NA<br>**Entry**: NA<br>**Output**: NA | Verify | CO<br>D<br>User<br>R W D |
| FIPS 186-2 PRNG Seed | Seed value for approved PRNG | Internally generated | **Storage**: RAM plaintext<br>**Type:** Ephemeral<br>**Association**: The system is the one and only owner. Relationship is maintained by the operating system via protected memory. | **Agreement**: NA<br>**Entry**: NA<br>**Output**: NA | Random Number Generation | CO<br>D<br>User<br>R W D |

| Key/CSP Name | Description / Use | Generation | Storage | Establishment / Export | Interface | Privileges |
|---|---|---|---|---|---|---|
| FIPS 186-2 PRNG Seed Key | Seed key for approved PRNG | Internally generated | **Storage**: RAM plaintext<br><br>**Type:** Ephemeral<br><br>**Association**: The system is the one and only owner. Relationship is maintained by the operating system via protected memory. | **Agreement**: NA<br><br>**Entry**: NA<br><br>**Output**: NA | Random Number Generation | CO<br>D<br><br>User<br>R W D |
| CO Password | Crypto Officer password | No | **Storage**: on disk plaintext<br><br>**Type:** Static<br><br>**Association**: The system is the one and only owner. Relationship is maintained by the operating system via protected memory. | **Agreement**: NA<br><br>**Entry**: Electronic<br><br>**Output**: NA | Control Input Physical Interface | CO<br>R W D |

R = Read W = Write D = Delete

**Table 7 – Module Keys/CSPs**

### 2.9.1 Key Generation

The module supports the generation of the asymmetric and symmetric keys via Federal Information processing Standard 186-2, Digital Signature Standard (FIPS 186-2) Approved random number generator.

### 2.9.2 Key Entry, Output, and Protection

All keys and CSPs reside on memory internally allocated by the module and can only be output using the exposed APIs. The module does not support key entry or output from the physical boundary. The operating system and runtime environment protect the memory and process space from unauthorized access.

## 2.10 Self-Tests

The module includes an array of self-tests that are run during startup and periodically during operations to prevent any secure data from being released and to ensure all components are functioning correctly. In the event of any self-test failure, the module/ McAfee Agent application will output an error to the audit log and will shutdown. In addition to self-test failures, successful loading of the module is also logged. To access status of self-tests, success or failure, the application provides access to the audit log. Status is viewable via operating environment's audit mechanism and by verifying proper loading and operation of the McAfee Agent application. While the module is running self-tests, the module will not output cryptographic data. The McAfee Agent application makes calls to the Agent Cryptographic Module (Version 1.0), and data will not be returned until the self-tests complete.

No keys or CSPs will be output when the module is in an error state. The module will halt and the process will terminate; as such, no data will be output via the data output interface. Additionally, the module does not support a bypass function, and the module does not allow plaintext cryptographic key components or other unprotected CSPs to be output on physical ports. No external software or firmware is allowed to be loaded into the module in a FIPS mode of operation.

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

### 2.10.1 Power-On Self-Tests

Power-on self-tests are run upon every initialization of the module. If any of the tests fail, the module will not initialize, the module will enter an error state, and no services can be accessed by the users. The module implements the following power-on self-tests:

- RSA pairwise consistency (signing and signature verification)
- DSA pairwise consistency (signing and signature verification)
- SHA-1 and SHA-256 KAT
- AES KAT (encryption and decryption)
- TDES KAT (encryption and decryption)
- KAT for Approved PRNG
- Module integrity check via RSA 2048-bit digital signature verification

The module performs all power-on self-tests automatically when the module is initialized. All power-on self-tests must be passed before a User/Crypto Officer can perform services. The Power-on self-tests can be run on demand by reinitializing the module in FIPS approved Mode of Operation. Upon passing the power-on self-tests, the module will log the success and will continue to boot normally; successful

loading of the McAfee Agent application will indicate that all self-tests have passed. If a self-test fails, the module will not load, the McAfee Agent application will halt, and an error will be logged.

### 2.10.2 Conditional Self-Tests

Conditional self-tests are on-demand tests and tests run continuously during operation of the module. If any of these tests fail, the module will enter an error state and no services can be accessed by the users. The module can be re-initialized to clear the error and resume FIPS mode of operation. The module performs the following conditional self-tests:

- RSA pairwise consistency
- DSA pairwise consistency
- Continuous RNG test run on output of Approved PRNG
- Continuous test on output of Approved PRNG seed mechanism
- Test to ensure Approved PRNG output and seed do not match

The module will inhibit data output via the output interface when conditional tests are performed. Once the tests have passed and the keys have been generated, the module will pass the key to the calling daemon.

## 2.11 Mitigation of Other Attacks

The module does not mitigate other attacks.

# 3 Guidance and Secure Operation

This section describes how to configure the module for FIPS-approved mode of operation.

## 3.1 Crypto Officer and User Guidance

### 3.1.1 Software Packaging and OS Requirements

The module is included with McAfee Agent version 4.6 and is not available for direct download. The McAfee Agent application must be installed on a supported operating system running in single user mode. To configure single-user mode, the following must be disabled:

- Remote registry and remote desktop services
- Remote assistance
- Guest accounts
- Server and terminal services

Specific configuration steps are beyond the scope of this document.

### 3.1.2 Enabling FIPS Mode

To meet the cryptographic security requirements, certain restrictions on the installation and use of McAfee Agent must be followed. The steps below will ensure that the module implements all required self-tests and uses only approved algorithms. Please note that once the module is in FIPS-approved mode, it cannot transition to a non-approved mode.

#### *3.1.2.1 Installation*

1. The installation must be a new install. Upgrading from a previous version of McAfee Agent is not valid.
2. The module is included with McAfee Agent 4.6 and is not separately purchased or installed. McAfee Agent 4.6 (and subsequently the module) can be installed either via deployment from ePO Server or downloading and executing `framepkg.exe` from the ePO server.

### 3.1.3 Additional Rules of Operation

1. All host system components that can contain sensitive cryptographic data (main memory, system bus, disk storage) must be located in a secure environment.

2. The writable memory areas of the module (data and stack segments) are accessible only by the McAfee Agent application so that the module is in "single user" mode, i.e. only the McAfee Agent application has access to that instance of the Module.
3. Only 2048-bit asymmetric keys should be used where available.
4. The operating system is responsible for multitasking operations so that other processes cannot access the address space of the process containing the Module.