Microsoft Windows

FIPS 140 Validation

Microsoft Windows Server 2019

Microsoft Azure Stack Edge

Microsoft Azure Stack Hub

Microsoft Azure Stack Edge Rugged

Non-Proprietary

Security Policy Document

Document Information	
Version Number	1.2
Updated On	March 20, 2023

Version History

Version	Date	Summary of Changes
1.0	November 2, 2020	Draft sent to NIST CMVP
1.1	November 3, 2022	Updates in response to NIST feedback
1.2	March 20, 2023	Updates in response to NIST feedback

TABLE OF CONTENTS

Contents

1 Introduction

The Windows Boot Manager module is the first Windows component to load when the computer powers up. When Secure Boot is enabled, the integrity of Boot Manager is validated before loading by the computer's UEFI firmware.

Along with other startup and initialization tasks, Boot Manager loads and cryptographically validates the integrity of Winload.efi, the next module in the startup sequence. For the purpose of this validation, Boot Manager is classified as a Software-Hybrid cryptographic module because the validated platforms all implement the AES-NI instruction set.

1.1 List of Cryptographic Module Binary Executables

Boot Manager cryptographic module contains the following binaries:

- bootmgfw.efi
- bootmgr.efi

The builds covered by this validation are:

- Windows Server 2019 build 10.0.17763.10021 and 10.0.17763.10127

1.2 Validated Platforms

The editions covered by this validation are:

- Windows Server 2019 Datacenter Core

The Boot Manager components listed in Section 1.1 were validated using the combination of computers and Windows operating system editions specified in the table below.

All the computers for Windows Server listed in the table below are all 64-bit Intel architecture and implement the AES-NI instruction set but not the SHA Extensions.

Table 1 Validated Platforms

Computer	Windows Server 2019 Datacenter Core	Processor Image
Microsoft Azure Stack Edge - Dell XR2 - Intel Xeon Silver 4114	√	wikichip.org

Microsoft Azure Stack Hub - Dell PowerEdge R640 - Intel Xeon Gold 6230	√	wikichip.org
Microsoft Azure Stack Hub - Dell PowerEdge R840 - Intel Xeon Platinum 8260	√	wikichip.org
Microsoft Azure Stack Edge Rugged - Rugged Mobile Appliance – Intel Xeon D-1559	√	wikichip.org

1.3 BitLocker

BitLocker is a data protection feature that encrypts entire disk volumes. Boot Manager collects authorization factors by reading data or interacting with the user. In Windows Server, the following keys types may be derived from the authorization factors and used to unlock BitLocker encrypted OS volumes:

Authorization Factor	FIPS Approved?
A password entered by the user.	Yes, when the user chooses a strong password.
An external key which may be used during boot if it is stored on a USB drive, or during recovery if it is stored on a USB drive or typed in manually.	Yes
A key stored in the TPM	Yes, when the TPM has been FIPS 140 validated.
A TPM key combined with a user-provided PIN	Yes, when the TPM has been FIPS 140 validated.
A TPM key combined with the external key	Yes, when the TPM has been FIPS 140 validated.

A TPM key combined with a PIN and the external key	Yes, when the TPM has been FIPS 140 validated.
A TPM key combined with a network key	Yes, when the TPM has been FIPS 140 validated.
A key stored on disk and only used when BitLocker is disabled but the drive is encrypted	Yes

The second diagram in the Finite State Model describes how Boot Manager collects and uses these protection factors.

2 Cryptographic Module Specification

Boot Manager is a multi-chip standalone module that operates in FIPS-approved mode during normal operation of the computer and Windows operating system boot sequence.

The following configurations and modes of operation results in Boot Manager operating in a non-approved mode of operation:

- Boot Windows in Debug mode
- Boot Windows with Driver Signing disabled

For BitLocker and the authorization factors in section 1.3 to fully operate in a FIPS Approved mode, the other Windows cryptographic modules and the TPM must also operate in an Approved mode.

2.1 Cryptographic Boundary

The software-hybrid cryptographic boundary for Boot Manager consists of disjoint software and hardware components within the same physical boundary of the host platform. The software components are defined as the binaries bootmgfw.efi and bootmgr.efi, and the hardware components are the CPUs running on each host platform.

2.2 FIPS 140-2 Approved Algorithms

Boot Manager implements the following FIPS-140-2 Approved algorithms:[1]

Table 2 FIPS 140-2 Approved Algorithms

Algorithm	Windows Server 2019 build 10.0.17763.10021	Windows Server 2019 build 10.0.17763.10127
FIPS 186-4 RSA PKCS#1 (v1.5) digital signature verification with 1024, 2048, and 3072 moduli; supporting SHA-1, SHA-256, SHA-384, and SHA-512	#C1586	#C2052
FIPS 180-4 SHS SHA-1, SHA-256, SHA-384, and SHA-512	#C1577	#C2044

[1] This module may not use some of the capabilities described in each CAVP certificate.

FIPS 197 AES CBC 128, and 256 (Decrypt)	#C1577	#C2044
FIPS PUB 198-1 HMAC-SHA-1[2] and HMAC-SHA-256	#C1577	#C2044
NIST SP 800-38E AES XTS 128 and 256 (Decrypt)	#C1577	#C2044
NIST SP 800-38C AES CCM 256 (Decrypt)	#C1583	#C2049
NIST SP 800-132 PBKDF supporting HMAC-SHA-256	Vendor Affirmed	Vendor Affirmed
NIST SP 800-133 symmetric key generation by combining multiple keys and other data in an Exclusive-Or operation[3]	Vendor Affirmed	Vendor Affirmed

2.3 Non-Approved Algorithms

Boot Manager implements only Approved algorithms.

2.4 Cryptographic Bypass

Cryptographic bypass is not supported by Boot Manager.

2.5 NIST SP 800-132 Password Based Key Derivation Function (PBKDF) Usage

When BitLocker is configured to use a password or PIN protector to protect the system volume, a Password Based Key Derivation Function (PBKDF) is used to derive a suitable key for storage applications such as BitLocker. The PBKDF implemented in this module is NIST SP 800-132 compliant.

The PBKDF implementation has the following characteristics that align with SP 800-132:

- 128-bit Salt
- Iteration count is 2^{20} (1,048,576)

Note that the password length is enforced by the caller of the PBKDF interfaces at the time the password/passphrase is created and not by this cryptographic module because Boot Manager is not involved in the creation of any password.

The following TechNet topics describe the security characteristics of passwords, instructions for setting the enforcement mechanism, and a discussion of strong passwords and recommended minimum settings:

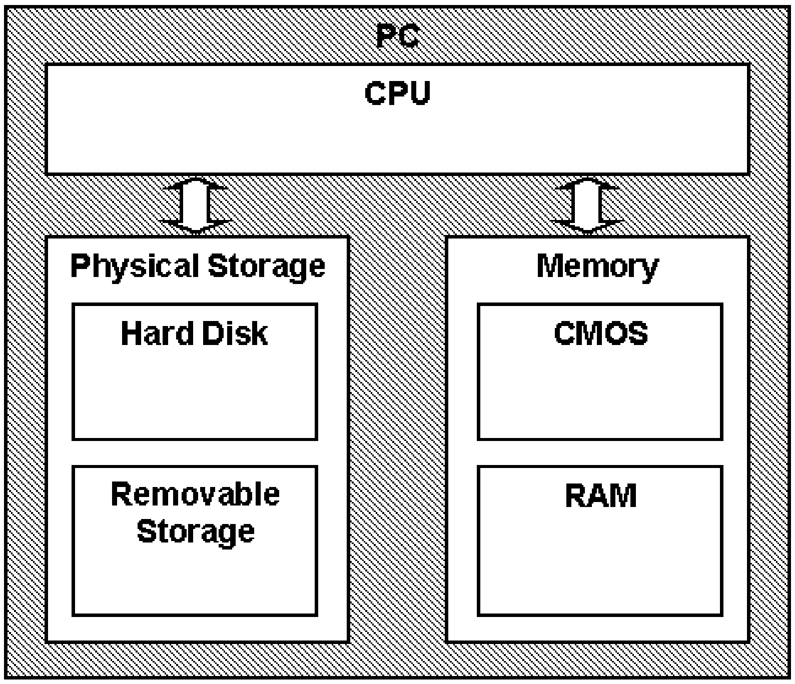
[2] For HMAC, only key sizes that are >= 112 bits in length are used by the module in FIPS mode.
[3] See NIST SP 800-133, section 6.3.

[Passwords Technical Overview]
[Best Practices for Enforcing Password Policies]

SP 800-132 Section 5.4 describes two options for protecting data using a Master Key (MK). Boot Manager uses Option 2 in which the MK produced by the PBKDF is used to decrypt a Data Protection Key (DPK). In Windows, the DPK corresponds to the Volume Master Key (VMK), which is used to decrypt the Full Volume Encryption Key (FVEK) which is used for actual data encryption/decryption. The VMK and FVEK are described later in this document.

2.6 Hardware Components of the Cryptographic Module

The physical boundary of the module is the physical boundary of the computer that contains the module. The following diagram illustrates the hardware components used by the Boot Manager module:



Note: The CPU provides Processor Algorithm Accelerator (PAA)

3 Cryptographic Module Ports and Interfaces

3.1 Control Input Interface

The Boot Manager Control Input Interface is the set of internal functions responsible for reading control input. These input signals are read from various system locations and are not directly provided by the operator. Examples of the internal function calls include:

- BlBdDebuggerEnabled – Reads the system flag to determine if the boot debugger is enabled.
- BlXmiRead – Reads the operator selection from the Boot Selection menu.

- BlGetBootOptionBoolean – Reads control input from a protected area of the Boot Configuration Data registry.

The computer's keyboard can also be used as control input when it is necessary for an operator to provide a response to a prompt for input or in response to an error indicator.

3.2 Status Output Interface

The Status Output Interface is the BlStatusPrint function that is responsible for displaying any integrity verification errors to the display. The Status Output Interface is also defined as the BsdpWriteAtLogOffset responsible for writing the name of the corrupt driver to the boot log.

3.3 Data Output Interface

The Data Output Interface includes two different kinds of functions: initialization and transfer.

The initialization function ImgpInitializeBootApplicationParameters output are the input parameters for the boot application which Boot Manager launches. This function is called before transferring execution to the boot application.

The following functions are transfer functions: Archx86TransferTo32BitApplicationAsm, Archx86TransferTo64BitApplicationAsm, and Archpx64TransferTo64BitApplicationAsm. These functions are responsible for transferring the execution from Boot Manager to the initial execution point of the Windows OS Loader. Data exits the module in the form of the initial instruction address of Winload.efi.

3.4 Data Input Interface

The Data Input Interface includes the BlFileReadEx function. BlFileReadEx is responsible for reading the binary data of unverified components from the computer hard drive.

Additionally, the computer's USB port also forms a part of the Data Input interface. This interface is used to enter the BitLocker Startup key or Recovery key. The keyboard can also serve as a Data Input Interface for password-based protection factors, and the network interface can be used to enter the Network Key Intermediate Key.

4 Roles, Services and Authentication

4.1 Roles

In Windows Server, authentication and assignment of user roles happens after the OS boots. When BitLocker is used to encrypt the system volume, the user booting the system may interact with BitLocker by providing an authorization factor (CSP) as input to the module. Otherwise, since Boot Manager executes between power-on and the start of OS initialization, its functions are fully automatic and not configurable. FIPS 140 validations define formal "User" and "Cryptographic Officer" roles. Both roles can use any Boot Manager service.

4.2 Services

Boot Manager services are:

1. The **Secure Boot** service of the Windows Server Boot Manager will read the Secure Boot policy.
2. **Unlocking** the operating system volume (decrypt the FVEK)
3. **Decrypting** the operating system volume
4. **Loading and verifying** the integrity of the Windows Server operating system loader (winload.efi)
5. **Booting** the next boot application, the Windows OS Loader component, in the overall boot sequence for the Windows operating system. In some BitLocker scenarios Boot Manager must display UI. In this case, Boot Manager first validates and then loads the Multilingual User Interface (MUI) resource file.
6. **Show Status** – The module provides a show status service that is automatically executed by the module to provide the status response of the module either via output to the computer monitor or to log files.
7. **Self-Tests** – The module provides a power-up self-tests service that is automatically executed when the module is loaded into memory.
8. **Zeroizing** cryptographic material (see Section 7 Cryptographic Key Management)

Boot Manager does not export any cryptographic functions that can be called or externally invoked.

The following table maps the services to their corresponding algorithms and critical security parameters (CSPs) as described in Critical Security Parameters.

Table 3 Services

Service	Algorithms	CSPs	Invocation
Secure Boot	RSA PKCS#1 (v1.5) verify with public key	RSA public key (to verify the integrity of the Secure Boot policy)	This service is fully automatic after Secure Boot has been enabled.
Unlocking the operating system volume (decrypt the FVEK)	HMAC-SHA-256 NIST SP 800-132 PBKDF AES in CCM mode (128 and 256 bit) NIST SP 800-133 cryptographic key generation	PIN, Password, DK, ExK, CC, IK, SK, NIK, NK, VMK described in Critical Security Parameters	See the diagrams in Finite State Model for the user actions during “System Volume Unlock” stage. This service is executed automatically after this stage has been reached.

Decrypting data from the BitLocker-encrypted operating system volume to bootstrap the Windows Server operating system	AES CBC (128 and 256 bit) NIST SP 800-38E AES XTS (128 and 256 bit)[4]	FVEK	This service is fully automatic.
Loading and **verifying the integrity** of the Windows Server operating system loader (winload.efi)	RSA PKCS#1 (v1.5) verify with public key SHA-1 hash SHA-256 hash SHA-384 hash SHA-512 hash	RSA public key (to verify the integrity of Windows OS Loader)	This service is fully automatic.
Booting the Windows operating system	None	None	This service is fully automatic.
Show Status	None	None	This service is fully automatic.
Self-Tests	See the Self-Tests section for the list of algorithms	None	This service is fully automatic.
Zeroizing	None	All CSPs	This service is fully automatic.

4.3 Authentication

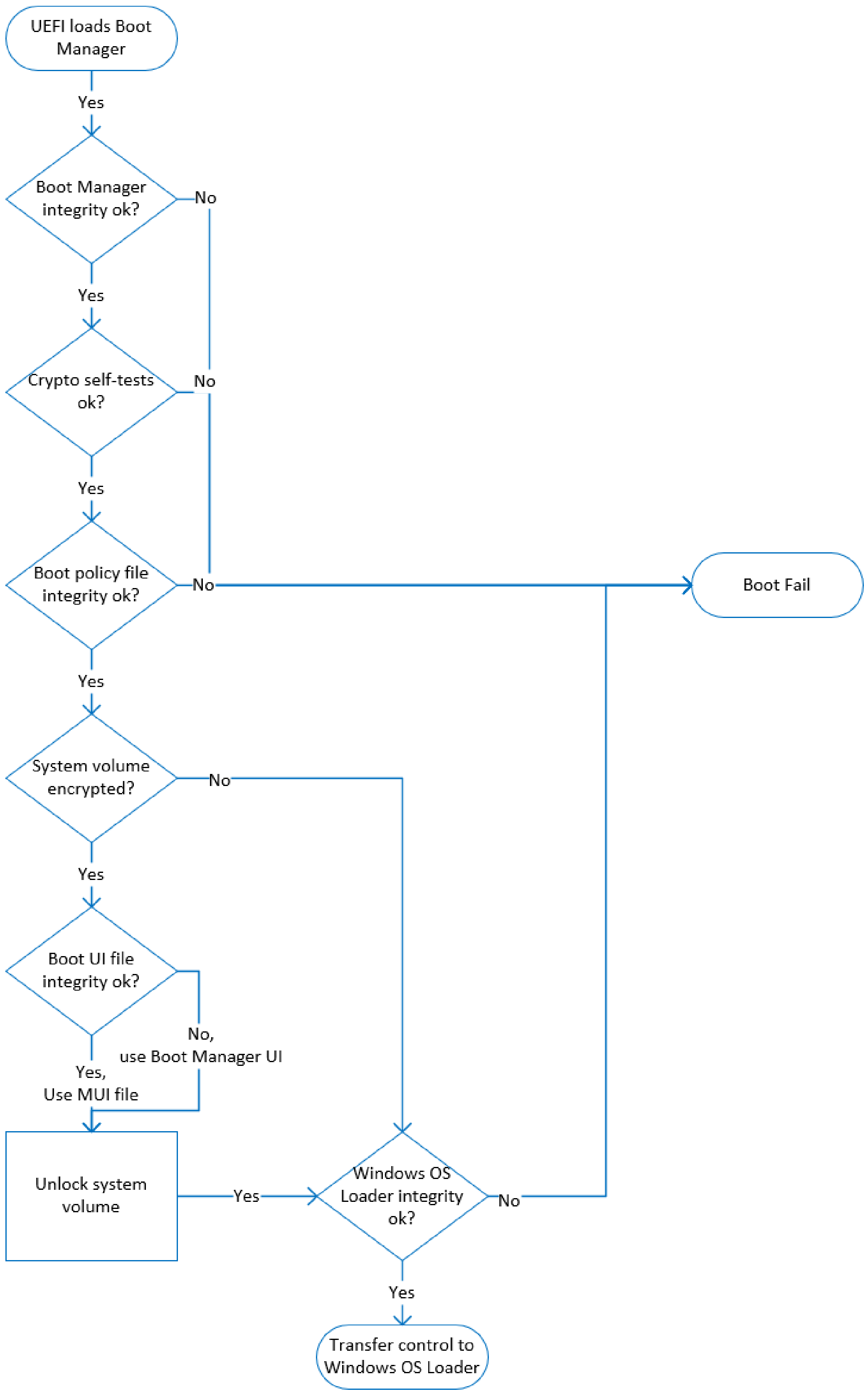
Boot Manager does not implement any authentication services as defined by the FIPS 140-2 standard, which is concerned exclusively with controlling access to cryptographic module ports and services.

5 Finite State Model

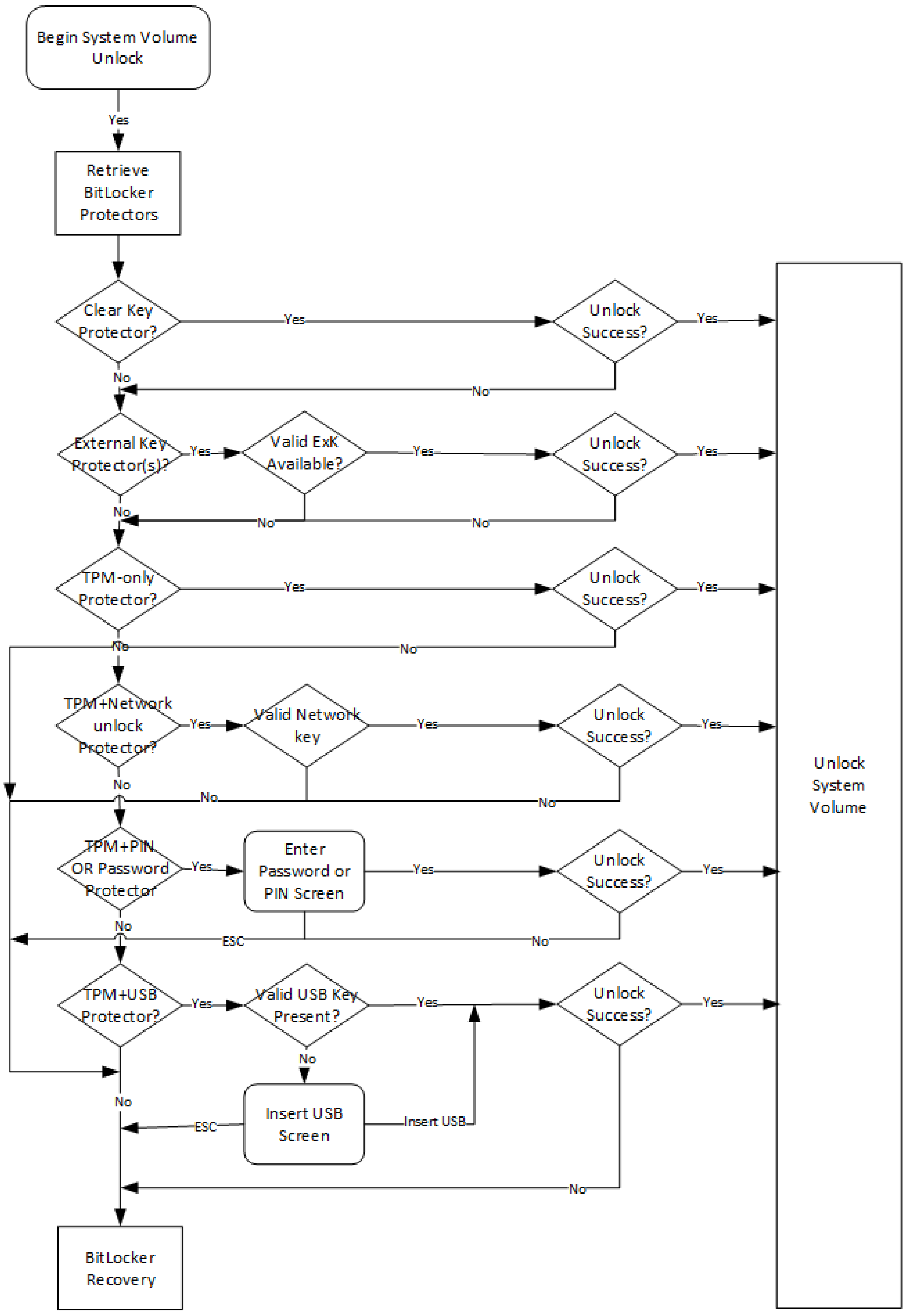
5.1 Specification

The following diagram shows the finite state model for Boot Manager:

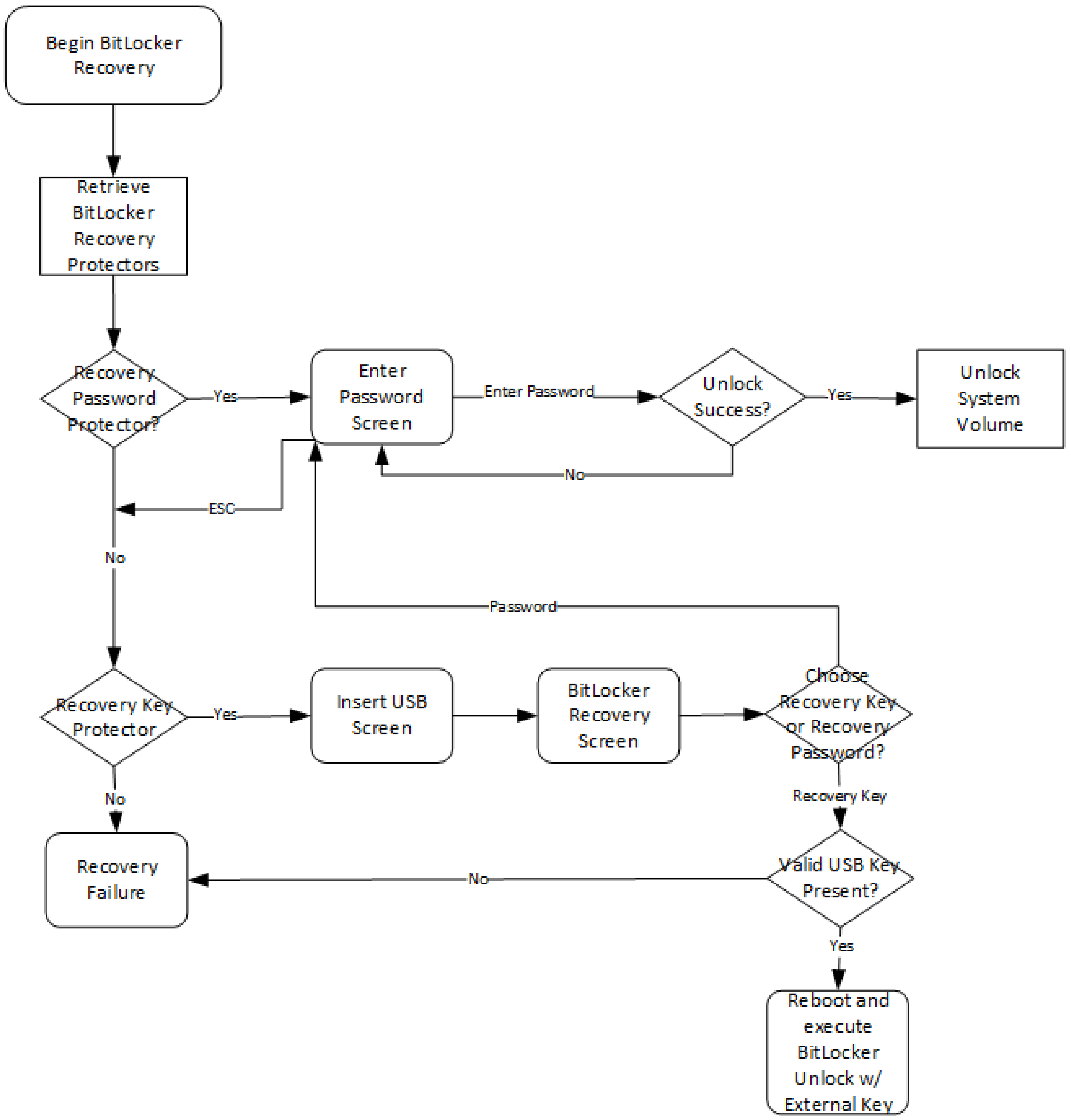
[4] The length of the data unit does not exceed 2^{20} AES blocks for storage applications such as BitLocker.


UEFI loads Boot Manager
Yes
Boot Manager integrity ok?
No
Yes
Crypto self-tests ok?
No
Yes
Boot policy file integrity ok?
No
Boot Fail
Yes
System volume encrypted?
No
Yes
Boot UI file integrity ok?
No, use Boot Manager UI
Yes, Use MUI file
Unlock system volume
Yes
Windows OS Loader integrity ok?
No
Yes
Transfer control to Windows OS Loader

The following diagram shows states and user inputs for the optional Unlock System Volume sequence:



The following state diagram shows the optional Bitlocker Recovery sequence:



6 Operational Environment

The operational environment for Boot Manager is the Windows Server operating system running on a supported hardware platform.

6.1 Single Operator

During the operating system boot process there is no logged on user, so the single operator requirement is met.

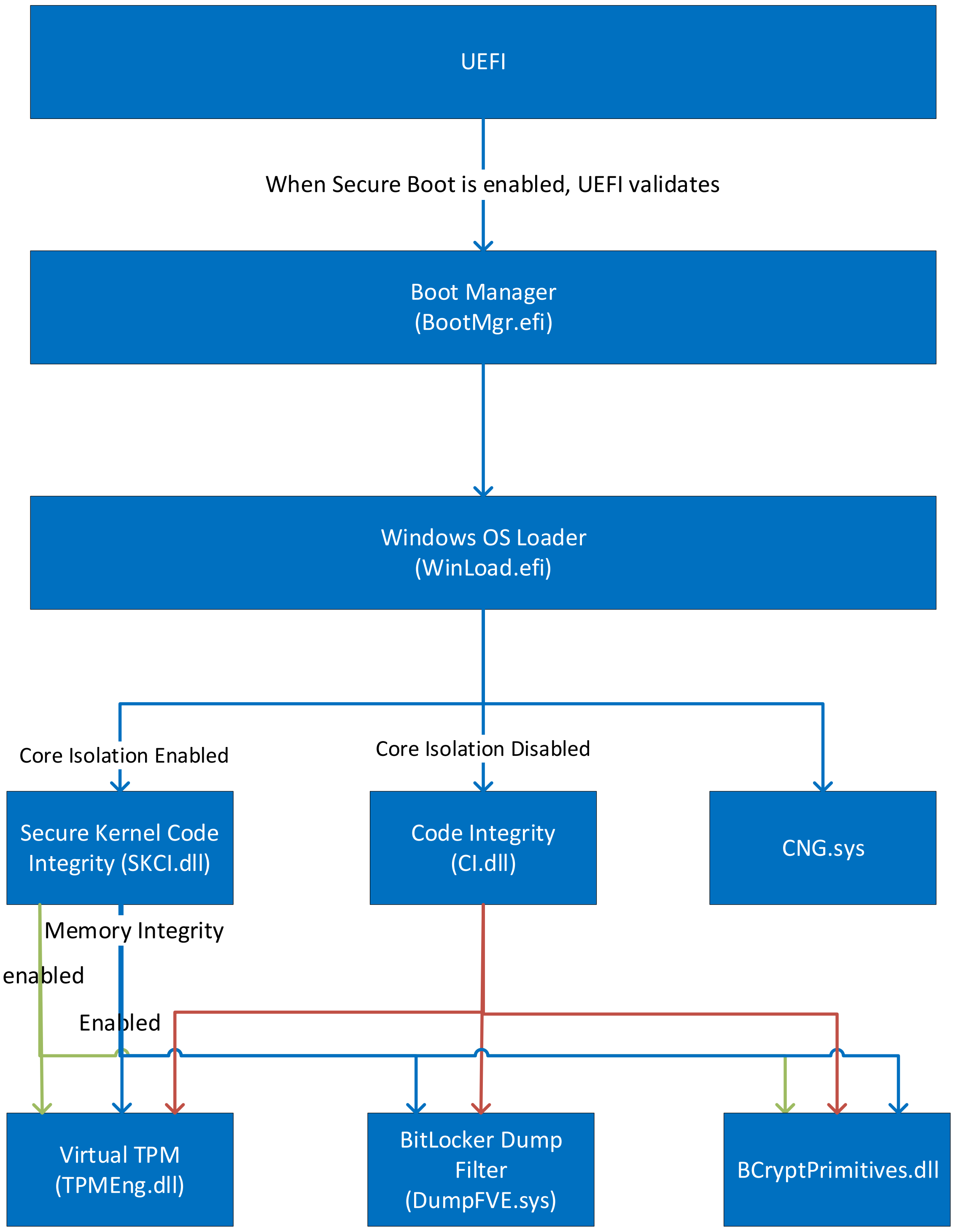
6.2 Cryptographic Isolation

While it is running, Boot Manager is the only process running on the computer.

6.3 Integrity Chain of Trust

Windows uses several mechanisms to provide integrity verification depending on the stage in the boot sequence and the hardware and configuration. The following diagram describes the Integrity Chain of trust for each supported configuration for the following versions:

- Windows Server 2019 build 10.0.17763.10021 and 10.0.17763.10127


UEFI
When Secure Boot is enabled, UEFI validates
Boot Manager (BootMgr.efi)
Windows OS Loader (WinLoad.efi)
Core Isolation Enabled
Core Isolation Disabled
Secure Kernel Code Integrity (SKCI.dll)
Code Integrity (CI.dll)
CNG.sys
Memory Integrity
Not enabled
Enabled
Virtual TPM (TPMEng.dll)
BitLocker Dump Filter (DumpFVE.sys)
BCryptPrimitives.dll

The integrity of Boot Manager is verified by UEFI when Secure Boot is enabled and by the Boot Manager itself.

Boot Manager verifies Windows OS Loader before transferring control to those components.

Windows binaries include a SHA-256 hash of the binary signed with the 2048-bit Microsoft RSA code-signing key (i.e., the key associated with the Microsoft code-signing certificate). The integrity check uses the public key component of the Microsoft code signing certificate to verify the signed hash of the binary.

7 Cryptographic Key Management

7.1 Critical Security Parameters

When BitLocker encrypts the computer's system volume, Boot Manager uses the following critical security parameters (CSPs):

Table 4 Critical Security Parameters

Critical Security Parameters	CSP / Key Description	Import	Export	Generation	Storage
PIN	An alpha-numeric PIN for Trusted Platform Module (TPM) + PIN or TPM + PIN + USB scenarios.	Input by the user. Windows generates the CSP following NIST SP 800-132, Recommendation for Password-Based Key Derivation	No	No	Volatile CSP, not persisted.
Password	A password.	Input by the user who should follow the guidelines in this Security Policy. Windows generates the CSP following NIST SP 800-132, Recommendation for Password-Based Key Derivation	No	No	Volatile CSP, not persisted.

Critical Security Parameters	CSP / Key Description	Import	Export	Generation	Storage
External Key (ExK)	256-bit AES key stored outside the cryptographic boundary, for example, on a USB device. The external key represents either a startup key or a recovery key and is used for AES decryption of the VMK.	Yes, imported into the module as a plaintext bitstring.	No	No	Volatile CSP, not persisted.
Clear Key (CK)	256-bit AES key generated by BitLocker runtime components when BitLocker enters Suspend mode	Yes	No	No	Plaintext CSP read from the same storage volume as the VMK and FVEK
Intermediate Key (IK)	256-bit AES key value that forms the basis of another intermediate AES key, such as the encrypted VMK, by combining with another 256-bit AES key using XOR.	No	No	Combined from the two keys following NIST SP 800-133 Recommendation for Cryptographic Key Generation, specifically, section 6.3, option #2	Volatile CSP, not persisted.
Network Intermediate Key (NIK)	256-bit AES key value that is used for the Network Unlock authenticator[5]. Boot Manager does the actual decryption of the NIK using AES-CCM with the Session Key.	Yes, imported into the module as an encrypted bitstring that was transported over a trusted network.	No	No	Volatile CSP, not persisted.

[5] Network Unlock is for BitLocker. It is not referring to FIPS 140-2 standard authentication used to control access to the ports and services of the cryptographic module.

Critical Security Parameters	CSP / Key Description	Import	Export	Generation	Storage
Network Key (NK)	256-bit key used for AES decryption of the VMK in Network Unlock authentication. Composed by XOR of an IK protected by the TPM and another IK delivered over a trusted network.	Intermediate key imported from a trusted local area network.	No	Combined from the two keys following NIST SP 800-133 Recommendation for Cryptographic Key Generation, specifically, section 6.3, option #2	Volatile CSP, not persisted.
Session Key (SK)	256-bit AES key value used to decrypt an NIK	Yes	No	No	Plaintext CSP read from the same storage volume as the VMK and FVEK
Derived Key (DK)	256-bit AES key value used for AES-CCM decryption of the VMK. The value is not persisted and is derived using a method defined by the system configuration. Derived Keys are used in Password TPM combination mechanisms.	No	No	No	Volatile CSP, not persisted.
Volume Master Key (VMK)	256-bit AES key used for AES-CCM decryption of the FVEK	Yes	No	No	Stored encrypted on the storage volume encrypted by BitLocker.
Full Volume Encryption Key (FVEK)	128 or 256-bit AES key used for AES-XTS encryption/decryption of data on disk sectors This key is stored encrypted, it is decrypted by the VMK using AES-CCM.	Yes	No	No	Stored encrypted on the storage volume encrypted by BitLocker.

Critical Security Parameters	CSP / Key Description	Import	Export	Generation	Storage
Microsoft Root Certificate Authority (CA) Public Key	2048-bit key used for RSA PKCS#1 (v1.5) verification of digital signatures.	Yes	No	No	Embedded within the Boot Manager binary executable file.

The combination of independently established keys using XOR to create Derived Keys such as the combination of the keys in TPM + external key (USB), TPM + Network unlock, TPM + PIN, and TPM + PIN + external key (USB) is approved in NIST SP 800-133 (see section 7.6).

Details about the keys and network protocol used for Network Unlock authentication are in the Network Key Protector Unlock Protocol Specification [MS-NKPU], which is available at https://msdn.microsoft.com/en-us/library/hh537327.aspx.

Appendix B provides a justification for why each of the BitLocker authorization factors in section 1.3 are FIPS Approved.

7.2 Zeroization Procedures

7.2.1 Volatile Keys

All keys and key materials are zeroized after they are used, except for the FVEK. The FVEK is zeroized when the module is unloaded from memory after control has been transferred to WinLoad.efi.

7.2.2 Persistent Keys

Procedural zeroization of persistent keys for this software-hybrid cryptographic module consists of reformatting and overwriting, at least once, the hard drive or other permanent storage media for the operating system.

7.3 Access Control Policy

The Boot Manager cryptographic module does not allow access to the cryptographic keys contained within it, so, an access control table is not included in this document. Boot Manager receives keys from outside and then manages them appropriately once received. Boot Manager prevents access to its keys by zeroizing them.

8 Self-Tests

8.1 Power-On Self-Tests

Boot Manager performs the following power-on (startup) self-tests.

- RSA PKCS#1 (v1.5) signature verification Known Answer Test (RSA 2048 with SHA-256)
- RSA PKCS#1 (v1.5) Software Integrity Test verify with public key (RSA 2048 with SHA-256)
- SHA-1 Known Answer Test
- SHA-256 Known Answer Test
- SHA-512 Known Answer Test
- AES-CBC 128 and 256 Encrypt/Decrypt Known Answer Tests
- AES-CCM 256 Encrypt/Decrypt Known Answer Tests
- XTS-AES 128 and 256 Encrypt/Decrypt Known Answer Tests
- HMAC-SHA-1 Known Answer Test[6]
- HMAC-SHA-256 Known Answer Test
- NIST SP 800-132 PBKDF Known Answer Test

If the self-test fails, the module will not load, the system will not boot, and status will be returned. If the status is not STATUS_SUCCESS, then that is the indicator a self-test failed.

9 Design Assurance

The secure installation, generation, and startup procedures of this cryptographic module are part of the overall operating system secure installation, configuration, and startup procedures for the Windows Server operating system.

The Windows Server operating system must be pre-installed on a computer by an OEM, installed by the end-user, by an organization's IT administrator, or updated from a previous Windows Server version downloaded from Windows Update.

The installed version of Windows must be checked to match the version that was validated. See Appendix A for details on how to do this. The computer operator must ensure that all Windows binaries (cryptographic modules) listed in section 6.3 are from the same operating system build and version (operational environment) to provide assurance for correct operation of Windows and security. This ensures that Windows will use an approved DRBG when generating cryptographic keys, as described in NIST SP 800-133 revision 2, Recommendation for Cryptographic Key Generation.

An inspection of authenticity of the physical installation media can be made by following the guidance at this Microsoft web site: https://www.microsoft.com/en-us/howtotell/default.aspx

For Windows Updates, the client only accepts binaries signed with Microsoft certificates. The Windows Update client only accepts content whose signed SHA-2 hash matches the SHA-2 hash specified in the metadata. All metadata communication is done over a Secure Sockets Layer (SSL) port. Using SSL ensures that the client is communicating with the real server and so prevents a spoof server from sending the client harmful requests. The version and digital signature of new cryptographic module releases must be verified to match the version that was validated. See Appendix A for details on how to do this.

[6] This algorithm is used only for self-tests.

10 Mitigation of Other Attacks

The following table lists the mitigations of other attacks for this cryptographic module:

Table 5 Mitigation of Other Attacks

Algorithm	Protected Against	Mitigation
SHA1	Timing Analysis Attack	Constant time implementation
	Cache Attack	Memory access pattern is independent of any confidential data
SHA2	Timing Analysis Attack	Constant time implementation
	Cache Attack	Memory access pattern is independent of any confidential data
AES	Timing Analysis Attack	Constant time implementation
	Cache Attack	Memory access pattern is independent of any confidential data
		Protected against cache attacks only when running on a processor that implements AES-NI

11 Security Levels

The security level for each FIPS 140-2 security requirement is given in the following table:

Table 6 Security Levels

Security Requirement	Security Level
Cryptographic Module Specification	1
Cryptographic Module Ports and Interfaces	1
Roles, Services, and Authentication	1
Finite State Model	1
Physical Security	1
Operational Environment	1
Cryptographic Key Management	1
EMI/EMC	1
Self-Tests	1
Design Assurance	2
Mitigation of Other Attacks	1

Boot Manager is a multi-chip standalone software-hybrid module whose host platforms meet the level 1 physical security requirements. The module consists of production-grade components that include standard passivation techniques and is entirely contained within a metal or hard plastic production-grade enclosure that may include doors or removable covers.

12 Additional Details

For the latest information on Microsoft Windows, check out the Microsoft web site at:
https://www.microsoft.com/en-us/windows

For more information about FIPS 140 validations of Microsoft products, please see:
https://docs.microsoft.com/en-us/windows/security/threat-protection/fips-140-validation

13 Appendix A – How to Verify Windows Versions and Digital Signatures

13.1 How to Check Windows Versions

The installed version of Windows must be verified to match the version that was validated using the following method:

1. In the Search box type "cmd" and open the Command Prompt desktop app.
2. The command window will open.
3. At the prompt, enter "ver".
4. The version information will be displayed in a format like this:
   ```
   Microsoft Windows [Version 10.0.xxxxx]
   ```

If the version number reported by the utility matches the expected output, then the installed version has been validated to be correct.

13.2 How to Verify Windows Digital Signatures

After performing a Windows Update that includes changes to a cryptographic module, the digital signature and file version of the binary executable file must be verified. This is done like so:

1. Open a new window in Windows Explorer.
2. Type “C:\Windows\” in the file path field at the top of the window.
3. Type the cryptographic module binary executable file name (for example, “CNG.SYS”) in the search field at the top right of the window, then press the Enter key.
4. The file will appear in the window.
5. Right click on the file’s icon.
6. Select Properties from the menu and the Properties window opens.
7. Select the Details tab.
8. Note the File version Property and its value, which has a number in this format: xx.x.xxxxx.xxxx.
9. If the file version number matches one of the version numbers that appear at the start of this security policy document, then the version number has been verified.
10. Select the Digital Signatures tab.
11. In the Signature list, select the Microsoft Windows signer.
12. Click the Details button.
13. Under the Digital Signature Information, you should see: “This digital signature is OK.” If that condition is true, then the digital signature has been verified.

14 Appendix B – Rationale for BitLocker Authorization Factors

Authorization Factor	FIPS Approved?	Justification
A password entered by the user	Yes, when the user chooses a strong password.	• The VMK meets the requirements in NIST SP 800-133, Recommendation for Cryptographic Key Generation; specifically, section 7.1. • The password meets the requirements in NIST SP 800-132, Recommendation for Password-Based Key Derivation. • The key derived from the password is used to decrypt the VMK.
An External Key which may be used during boot if it is stored on a USB drive, or during recovery if it is stored on a USB drive or typed in manually	Yes	• The VMK meets the requirements in NIST SP 800-133, Recommendation for Cryptographic Key Generation; specifically, section 7.1. • The External Key meets the requirements in NIST SP 800-133, Recommendation for Cryptographic Key Generation; specifically, section 7.1 “direct generation”. • The external key is used to decrypt the VMK.
A key stored in the TPM	Yes, when the TPM has been FIPS 140 validated.	• The VMK meets the requirements in NIST SP 800-133, Recommendation for Cryptographic Key Generation; specifically, section 7.1. • The unsealed intermediate key is used to decrypt the VMK.
A TPM key combined with a user-provided PIN	Yes, when the TPM has been FIPS 140 validated.	• The VMK meets the requirements in NIST SP 800-133, Recommendation for Cryptographic Key Generation; specifically, section 7.1. • The user-provided PIN meets the requirements in NIST SP 800-132, Recommendation for Password-Based Key Derivation. • Combining the TPM-protected Intermediate Key and User-provided PIN meets the requirements in NIST SP 800-133 Recommendation for Cryptographic Key Generation; specifically, section 7.6, option #3.

A TPM key combined with the External Key	Yes, when the TPM has been FIPS 140 validated.	• The VMK meets the requirements in NIST SP 800-133, Recommendation for Cryptographic Key Generation; specifically, section 7.1. • The External Key meets the requirements in NIST SP 800-133, Recommendation for Cryptographic Key Generation; specifically, section 7.1 “direct generation”. • Combining the TPM-protected Intermediate Key and External Key meets the requirements in NIST SP 800-133, Recommendation for Cryptographic Key Generation; specifically, section 7.6, option #3.
A TPM key combined with a PIN and the External Key	Yes, when the TPM has been FIPS 140 validated.	• The VMK meets the requirements in NIST SP 800-133, Recommendation for Cryptographic Key Generation; specifically, section 7.1. • The External Key meets the requirements in NIST SP 800-133, Recommendation for Cryptographic Key Generation; specifically, section 7.1 “direct generation”. • The user-provided PIN meets the requirements in NIST SP 800-132, Recommendation for Password-Based Key Derivation. • Combining the TPM-protected Intermediate Key, External Key, and User-provided PIN meets the requirements in NIST SP 800-133, Recommendation for Cryptographic Key Generation; specifically, section 7.6, option #3.
A TPM key combined with a Network Key	Yes, when the TPM has been FIPS 140 validated.	• The VMK meets the requirements in NIST SP 800-133, Recommendation for Cryptographic Key Generation; specifically, section 7.1. • The Network Key meets the requirements in NIST SP 800-133, Recommendation for Cryptographic Key Generation; specifically, section 7.1 “direct generation”. • Combining the TPM-protected Intermediate Key, and Network Key meets the requirements in NIST SP 800-133, Recommendation for Cryptographic Key Generation; specifically, section 7.6, option #3.
A key stored on disk and only used when BitLocker is disabled but the drive is encrypted	Yes	• Meets the requirements in NIST SP 800-133, Recommendation for Cryptographic Key Generation; specifically section 7.1 “direct generation”.