Microsoft Windows

FIPS 140 Validation

Microsoft Windows 10 (Creators Update, Fall Creators Update, April 2018 Update)

Microsoft Windows 10 Mobile (Creators Update, Fall Creators Update)

Microsoft Windows Server (version 1709 and 1803)
The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This document is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS OR IMPLIED, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. This work is licensed under the Creative Commons Attribution-NoDerivs-NonCommercial License (which allows redistribution of the work). To view a copy of this license, visit http://creativecommons.org/licenses/by-nd-nc/1.0/ or send a letter to Creative Commons, 559 Nathan Abbott Way, Stanford, California 94305, USA.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2018 Microsoft Corporation. All rights reserved.

Microsoft, Windows, the Windows logo, Windows Server, and BitLocker are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.
## Version History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Summary of Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>October 3, 2017</td>
<td>Draft sent to NIST CMVP</td>
</tr>
<tr>
<td>1.1</td>
<td>November 18, 2017</td>
<td>Updates for Windows 10 version 1709</td>
</tr>
<tr>
<td>1.2</td>
<td>October 22, 2018</td>
<td>Updates for Windows 10 version 1803</td>
</tr>
<tr>
<td>1.3</td>
<td>March 5, 2019</td>
<td>Updates for build 10.0.15063.728</td>
</tr>
</tbody>
</table>
# Table of Contents

## SECURITY POLICY DOCUMENT

- **Version History** ........................................................................................................... 3

## 1 Introduction .................................................................................................................. 6

### 1.1 List of Cryptographic Module Binary Executables .................................................. 6
### 1.2 Validated Platforms ................................................................................................. 6
### 1.3 BitLocker.................................................................................................................. 9

## 2 Cryptographic Module Specification ............................................................................ 10

### 2.1 Cryptographic Boundary ....................................................................................... 10
### 2.2 FIPS 140-2 Approved Algorithms ......................................................................... 10
### 2.3 Non-Approved Algorithms .................................................................................... 12
### 2.4 Cryptographic Bypass ............................................................................................ 12
### 2.5 NIST SP 800-132 Password Based Key Derivation Function (PBKDF) Usage ............ 12
### 2.6 Hardware Components of the Cryptographic Module ........................................... 13

## 3 Cryptographic Module Ports and Interfaces ................................................................. 13

### 3.1 Control Input Interface ......................................................................................... 13
### 3.2 Status Output Interface ......................................................................................... 14
### 3.3 Data Output Interface ............................................................................................ 14
### 3.4 Data Input Interface ............................................................................................... 14

## 4 Roles, Services and Authentication ............................................................................. 14

### 4.1 Roles ...................................................................................................................... 14
### 4.2 Services ................................................................................................................. 15
### 4.3 Authentication ....................................................................................................... 17

## 5 Finite State Model ....................................................................................................... 17

### 5.1 Specification .......................................................................................................... 17
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>OPERATIONAL ENVIRONMENT</td>
<td>21</td>
</tr>
<tr>
<td>6.1</td>
<td>SINGLE OPERATOR</td>
<td>22</td>
</tr>
<tr>
<td>6.2</td>
<td>CRYPTOGRAPHIC ISOLATION</td>
<td>22</td>
</tr>
<tr>
<td>6.3</td>
<td>INTEGRITY CHAIN OF TRUST</td>
<td>22</td>
</tr>
<tr>
<td>7</td>
<td>CRYPTOGRAPHIC KEY MANAGEMENT</td>
<td>25</td>
</tr>
<tr>
<td>7.1</td>
<td>CRITICAL SECURITY PARAMETERS</td>
<td>25</td>
</tr>
<tr>
<td>7.2</td>
<td>ZEROIZATION PROCEDURES</td>
<td>26</td>
</tr>
<tr>
<td>7.2.1</td>
<td>VOLATILE KEYS</td>
<td>26</td>
</tr>
<tr>
<td>7.2.2</td>
<td>PERSISTENT KEYS</td>
<td>26</td>
</tr>
<tr>
<td>7.3</td>
<td>ACCESS CONTROL POLICY</td>
<td>26</td>
</tr>
<tr>
<td>8</td>
<td>SELF-TESTS</td>
<td>26</td>
</tr>
<tr>
<td>8.1</td>
<td>POWER-ON SELF-TESTS</td>
<td>26</td>
</tr>
<tr>
<td>9</td>
<td>DESIGN ASSURANCE</td>
<td>27</td>
</tr>
<tr>
<td>10</td>
<td>MITIGATION OF OTHER ATTACKS</td>
<td>28</td>
</tr>
<tr>
<td>11</td>
<td>SECURITY LEVELS</td>
<td>29</td>
</tr>
<tr>
<td>12</td>
<td>ADDITIONAL DETAILS</td>
<td>29</td>
</tr>
<tr>
<td>13</td>
<td>APPENDIX A – HOW TO VERIFY WINDOWS VERSIONS AND DIGITAL SIGNATURES</td>
<td>30</td>
</tr>
<tr>
<td>13.1</td>
<td>HOW TO CHECK WINDOWS VERSIONS</td>
<td>30</td>
</tr>
<tr>
<td>13.2</td>
<td>HOW TO VERIFY WINDOWS DIGITAL SIGNATURES</td>
<td>30</td>
</tr>
<tr>
<td>14</td>
<td>APPENDIX B – RATIONALE FOR BITLocker AUTHORIZATION FACTORS</td>
<td>31</td>
</tr>
</tbody>
</table>
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. When Windows resumes from hibernation (ACPI power state S4), the Boot Manager loads and cryptographically validates the integrity of Winresume.efi instead of Winload.efi.¹

1.1 List of Cryptographic Module Binary Executables
Boot Manager cryptographic module contains the following binaries:

- bootmgfw.efi
- bootmgr.efi

The Windows builds covered by this validation are:

- Windows 10 version 1703, build 10.0.15063
- Windows 10 version 1703, build 10.0.15063.728
- Windows 10 Mobile version 1703, build 10.0.15063
- Windows 10 Mobile version 1703, build 10.0.15063.728
- Windows 10 version 1709 and Windows Server version 1709 build 10.0.16299
- Windows 10 Mobile version 1709 build 10.0.15254
- Microsoft Surface Hub build 10.0.15063
- Microsoft Surface Hub build 10.0.15063.674
- Windows 10 version 1803 and Windows Server version 1803 build 10.0.17134

1.2 Validated Platforms
The Windows editions covered by this validation are:

- Microsoft Windows 10 Home Edition (32-bit version)
- Microsoft Windows 10 Pro Edition (64-bit version)
- Microsoft Windows 10 Enterprise Edition (64-bit version)
- Microsoft Windows 10 Education Edition (64-bit version)
- Microsoft Windows 10 S Edition (64-bit version)
- Microsoft Windows 10 Mobile
- Microsoft Surface Hub
- Windows Server Standard Core

¹ Windows Resume is not a FIPS 140 cryptomodule in Windows 10 version 1803.
• Windows Server 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 10 and 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. The exceptions are:

• Dell Inspiron 660s - Intel Core i3 without AES-NI and SHA Extensions
• HP Slimline Desktop - Intel Pentium with AES-NI and SHA Extensions

Windows 10 Mobile runs on the ARM architecture, which does not implement AES-Ni instructions or SHA extensions:

• Microsoft Lumia 950 - Qualcomm Snapdragon 808 (A57, A53)
• Microsoft Lumia 950 XL - Qualcomm Snapdragon 810 (A57, A53)
• Microsoft Lumia 650 - Qualcomm Snapdragon 212 (A7)
• HP Elite x3 - Qualcomm Snapdragon 820 (Kryo)

Table 1 Validated Platforms for Windows 10 Creators Update, Fall Creators Update and Windows Server

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft Surface Book 2</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft Surface Laptop</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft Surface Pro</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft Surface Book</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft Surface Pro 4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft Surface Pro 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft Surface 3 with LTE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft Surface Studio</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Windows Server Standard Core Hyper-V²</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2 Tested on Windows 10 version 1709
3 Tested on Windows 10 versions 1703 and 1709
4 Tested on Dell 5810MT hardware platform
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft Surface Go</td>
<td>✨</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft Surface Book 2</td>
<td>✨</td>
<td>✨</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft Surface Pro LTE</td>
<td>✨</td>
<td>✨</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft Surface Laptop</td>
<td>✨</td>
<td>✨</td>
<td></td>
<td>✨</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microsoft Surface Studio</td>
<td>✨</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5 Tested on Surface Pro 4 hardware platform
6 Tested on Windows 10 Mobile versions 1703 and 1709
7 Tested on Windows 10 version 1703
8 Tested on Surface Hub 10.0.15063 (1703), 10.0.15063.674, and 10.0.15063.728
## 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 10, the following key types may be derived from the authorization factors and used to unlock BitLocker encrypted OS volumes:

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

---

9 Hardware platform: Dell Precision Tower 5810MT

10 Hardware platform: Dell PowerEdge R740

---

© 2019 Microsoft. All Rights Reserved

This Security Policy is non-proprietary and may be reproduced only in its original entirety (without revision).
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
- Windows enters the ACPI S4 power state (for Windows 10 version 1803 only)

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 cryptographic boundary for Boot Manager is defined as the binaries bootmgfw.efi, and bootmgr.efi.

### 2.2 FIPS 140-2 Approved Algorithms

Boot Manager implements the following FIPS-140-2 Approved algorithms:¹¹

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>FIPS 186-4 RSA PKCS#1 (v1.5) digital</td>
<td>#2523</td>
<td>#2674</td>
<td>#2673</td>
<td>#2675</td>
<td>#3081</td>
<td>#2846</td>
</tr>
</tbody>
</table>

¹¹ This module may not use some of the capabilities described in each CAVP certificate.
<table>
<thead>
<tr>
<th>Signature Verification with 1024, 2048, and 3072 moduli; supporting SHA-1, SHA-256, SHA-384, and SHA-512</th>
<th>FIPS 180-4 SHS SHA-1, SHA-256, SHA-384, and SHA-512</th>
<th>FIPS 197 AES CBC 128, and 256</th>
<th>FIPS PUB 198-1 HMAC-SHA-1(^2) and HMAC-SHA-256</th>
<th>NIST SP 800-38E AES XTS 128 and 256</th>
<th>NIST SP 800-38C AES CCM 256</th>
<th>NIST SP 800-132 PBKDF supporting HMAC-SHA-256</th>
<th>NIST SP 800-133 symmetric</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>#3790</td>
<td>#4009</td>
<td>#4010</td>
<td>#4011</td>
<td>#4633</td>
<td>#4253</td>
<td></td>
</tr>
<tr>
<td></td>
<td>#4624</td>
<td>#4901</td>
<td>#4902</td>
<td></td>
<td>#5847</td>
<td>#5300</td>
<td></td>
</tr>
<tr>
<td></td>
<td>#3061</td>
<td>#3267</td>
<td>#3268</td>
<td>#3269</td>
<td>#3858</td>
<td>#3499</td>
<td></td>
</tr>
<tr>
<td></td>
<td>#4624</td>
<td>#4901</td>
<td>#4902</td>
<td>#5847</td>
<td>#5300</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>#4625</td>
<td>#4894</td>
<td>#4895</td>
<td>#4896</td>
<td>#5859</td>
<td>#5316</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>vendor affirmed</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\(^2\) For HMAC, only key sizes that are >= 112 bits in length are used by the module in FIPS mode.
2.3 Non-Approved Algorithms
Boot Manager implements the following non-Approved algorithms:

- IEEE 1619-2007 XTS-AES

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:

- Passwords Technical Overview
- Best Practices for Enforcing Password Policies

---

13 See NIST SP 800-133, section 7.6.
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:

![Diagram of hardware components]

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:

- `BIBdDebuggerEnabled` – Reads the system flag to determine if the boot debugger is enabled.
- `BLXmiRead` – Reads the operator selection from the Boot Selection menu.
- `BILGetBootOptionBoolean` – 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 BIStatusPrint 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 or Windows OS Resume. Data exits the module in the form of the initial instruction address of Winload.efi or Winresume.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.

### 4 Roles, Services and Authentication

#### 4.1 Roles

In Windows 10, 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 10 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 10 operating system loader (winload.efi or winresume.efi)
5. **Booting** the next boot application, either the Windows OS Loader or the Windows Resume 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>
<thead>
<tr>
<th>Service</th>
<th>Algorithms</th>
<th>CSPs</th>
<th>Invocation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Secure Boot</strong></td>
<td>RSA PKCS#1 (v1.5) verify with public key</td>
<td>RSA public key (to verify the integrity of the Secure Boot policy)</td>
<td>This service is fully automatic after Secure Boot has been enabled.</td>
</tr>
<tr>
<td><strong>Unlocking the operating system volume</strong></td>
<td>HMAC-SHA-256</td>
<td>PIN, Password, DK, ExK, CC, IK, SK, NK, VMK described in <strong>Critical Security Parameters</strong></td>
<td>See the diagrams in <strong>Finite State Model</strong> for the user actions during “System Volume Unlock” stage. This service is executed automatically after this stage has been reached.</td>
</tr>
<tr>
<td><strong>Unlocking</strong> the operating system volume (decrypt the FVEK)</td>
<td>NIST SP 800-132 PBKDF</td>
<td>PSK, Password, DK, ExK, CC, IK, SK, NK, VMK described in <strong>Critical Security Parameters</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Unlocking</strong> the operating system volume (decrypt the FVEK)</td>
<td>AES in CCM mode (128 and 256 bit)</td>
<td>PSK, Password, DK, ExK, CC, IK, SK, NK, VMK described in <strong>Critical Security Parameters</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Unlocking</strong> the operating system volume (decrypt the FVEK)</td>
<td>NIST SP 800-133 cryptographic key generation</td>
<td>PSK, Password, DK, ExK, CC, IK, SK, NK, VMK described in <strong>Critical Security Parameters</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Decryption</strong> data from the BitLocker-encrypted operating system volume to bootstrap the Windows 10 operating system</td>
<td>AES CBC (128 and 256 bit)</td>
<td>FVEK</td>
<td>This service is fully automatic.</td>
</tr>
<tr>
<td><strong>Loading and verifying the integrity</strong> of the Windows 10 operating system loader (winload.elfi or winresume.elfi)</td>
<td>RSA PKCS#1 (v1.5) verify with public key</td>
<td>RSA public key (to verify the integrity of Windows OS Loader)</td>
<td>This service is fully automatic.</td>
</tr>
<tr>
<td><strong>Loading and verifying the integrity</strong> of the Windows 10 operating system loader (winload.elfi or winresume.elfi)</td>
<td>SHA-1 hash</td>
<td>PSK, Password, DK, ExK, CC, IK, SK, NK, VMK described in <strong>Critical Security Parameters</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Loading and verifying the integrity</strong> of the Windows 10 operating system loader (winload.elfi or winresume.elfi)</td>
<td>SHA-256 hash</td>
<td>PSK, Password, DK, ExK, CC, IK, SK, NK, VMK described in <strong>Critical Security Parameters</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Loading and verifying the integrity</strong> of the Windows 10 operating system loader (winload.elfi or winresume.elfi)</td>
<td>SHA-384 hash</td>
<td>PSK, Password, DK, ExK, CC, IK, SK, NK, VMK described in <strong>Critical Security Parameters</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Loading and verifying the integrity</strong> of the Windows 10 operating system loader (winload.elfi or winresume.elfi)</td>
<td>SHA-512 hash</td>
<td>PSK, Password, DK, ExK, CC, IK, SK, NK, VMK described in <strong>Critical Security Parameters</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Booting the Windows operating system</strong></td>
<td>None</td>
<td>None</td>
<td>This service is fully automatic.</td>
</tr>
<tr>
<td><strong>Show Status</strong></td>
<td>None</td>
<td>None</td>
<td>This service is fully automatic.</td>
</tr>
<tr>
<td><strong>Self-Tests</strong></td>
<td>See the <strong>Self-Tests</strong> section for the list of algorithms</td>
<td>None</td>
<td>This service is fully automatic.</td>
</tr>
<tr>
<td><strong>Zeroizing</strong></td>
<td>None</td>
<td>All CSPs</td>
<td>This service is fully automatic.</td>
</tr>
</tbody>
</table>

14 The length of the data unit does not exceed $2^{20}$ AES blocks for storage applications such as BitLocker.
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:
UEFI loads Boot Manager

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

- No
  - Boot Fail
The following state diagram shows states and user inputs for the optional Unlock System Volume sequence:
This Security Policy is non-proprietary and may be reproduced only in its original entirety (without revision).
The following state diagram shows the optional Bitlocker Recovery sequence:

6 Operational Environment
The operational environment for Boot Manager is the Windows 10 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 10 version 1703, build 10.0.15063
- Windows 10 version 1703, build 10.0.15063.728
- Windows 10 Mobile version 1703, build 10.0.15063
- Windows 10 Mobile version 1703, build 10.0.15063.728
- Windows 10 Mobile version 1709 build 10.0.15254
- Microsoft Surface Hub build 10.0.15063
- Microsoft Surface Hub build 15063.674
The following diagram describes the Integrity Chain of trust for each supported configuration for the following versions:

- Windows 10 version 1709 and Windows Server build 10.0.16299
- Windows 10 version 1803 and Windows Server version 1803 build 10.0.17134
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 or Windows Resume 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>
<thead>
<tr>
<th>Critical Security Parameters</th>
<th>CSP / Key Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>External Key (ExK), Clear Key (CC)</td>
<td>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.</td>
</tr>
<tr>
<td>Intermediate Key (IK)</td>
<td>256-bit AES key value that is stored encrypted and forms the basis of another AES key, such as the NK or VMK, by combining with another 256-bit AES key using XOR.</td>
</tr>
<tr>
<td>Session Key (SK)</td>
<td>256-bit AES key value that is stored in plaintext on disk and used to decrypt an IK transported over a trusted network to Boot Manager during Network Unlock authentication. Boot Manager does the actual decryption of the IK using AES-CCM.</td>
</tr>
<tr>
<td>Network Key (NK)</td>
<td>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.</td>
</tr>
<tr>
<td>Volume Master Key (VMK)</td>
<td>256-bit AES key used for AES-CCM decryption of the FVEK.</td>
</tr>
<tr>
<td>Full Volume Encryption Key (FVEK)</td>
<td>128 or 256-bit AES key used for AES encryption/decryption of data on disk sectors. This key is stored encrypted on the system volume. It is encrypted and decrypted by the VMK using AES-CCM.</td>
</tr>
<tr>
<td>Derived Key (DK)</td>
<td>256-bit AES key value used for AES 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.</td>
</tr>
<tr>
<td>PIN</td>
<td>An alpha-numeric PIN for Trusted Platform Module (TPM) + PIN or TPM + PIN + USB scenarios.</td>
</tr>
<tr>
<td>Password</td>
<td>A password.</td>
</tr>
</tbody>
</table>

\[15\] 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.
The approved password-based key derivation method in NIST SP 800-132 is used when a password or PIN is used to derive a key.

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 or WinResume.efi.

7.2.2 Persistent Keys
Procedural zeroization of persistent keys for this software 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 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 Encrypt/Decrypt Known Answer Tests
• AES-CCM Encrypt/Decrypt Known Answer Tests
• XTS-AES Encrypt/Decrypt Known Answer Tests
• HMAC-SHA-1 Known Answer Test
• 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 10 operating system.

The Windows 10 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 10 version downloaded from Windows Update.

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

The installed version of Windows 10 must be checked to match the version that was validated. See Appendix A for details on how to do this.

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.

---

16 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>
<thead>
<tr>
<th>Algorithm</th>
<th>Protected Against</th>
<th>Mitigation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHA1</td>
<td>Timing Analysis Attack</td>
<td>Constant time implementation</td>
</tr>
<tr>
<td></td>
<td>Cache Attack</td>
<td>Memory access pattern is independent of any confidential data</td>
</tr>
<tr>
<td>SHA2</td>
<td>Timing Analysis Attack</td>
<td>Constant time implementation</td>
</tr>
<tr>
<td></td>
<td>Cache Attack</td>
<td>Memory access pattern is independent of any confidential data</td>
</tr>
<tr>
<td>AES</td>
<td>Timing Analysis Attack</td>
<td>Constant time implementation</td>
</tr>
<tr>
<td></td>
<td>Cache Attack</td>
<td>Memory access pattern is independent of any confidential data</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Protected against cache attacks only when running on a processor that implements AES-NI</td>
</tr>
</tbody>
</table>
11 Security Levels
The security level for each FIPS 140-2 security requirement is given in the following table:

<table>
<thead>
<tr>
<th>Security Requirement</th>
<th>Security Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cryptographic Module Specification</td>
<td>1</td>
</tr>
<tr>
<td>Cryptographic Module Ports and Interfaces</td>
<td>1</td>
</tr>
<tr>
<td>Roles, Services, and Authentication</td>
<td>1</td>
</tr>
<tr>
<td>Finite State Model</td>
<td>1</td>
</tr>
<tr>
<td>Physical Security</td>
<td>NA</td>
</tr>
<tr>
<td>Operational Environment</td>
<td>1</td>
</tr>
<tr>
<td>Cryptographic Key Management</td>
<td>1</td>
</tr>
<tr>
<td>EMI/EMC</td>
<td>1</td>
</tr>
<tr>
<td>Self-Tests</td>
<td>1</td>
</tr>
<tr>
<td>Design Assurance</td>
<td>2</td>
</tr>
<tr>
<td>Mitigation of Other Attacks</td>
<td>1</td>
</tr>
</tbody>
</table>

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

13.1 How to Check Windows Versions
The installed version of Windows 10 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.
## Appendix B – Rationale for BitLocker Authorization Factors

<table>
<thead>
<tr>
<th>Authorization Factor</th>
<th>FIPS Approved?</th>
<th>Justification</th>
</tr>
</thead>
</table>
| 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”.

This Security Policy is non-proprietary and may be reproduced only in its original entirety (without revision).