Security Policy

for FIPS 140-2 Validation

Microsoft Windows 8

Microsoft Windows Server 2012

Microsoft Windows RT

Microsoft Surface Windows RT

Microsoft Surface Windows 8 Pro

Microsoft Windows Phone 8

Microsoft Windows Storage Server 2012

BitLocker® Windows OS Loader (WINLOAD)

DOCUMENT INFORMATION

Version Number	1.2
Updated On	December 17, 2014

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.

TABLE OF CONTENTS

1 Introduction

The BitLocker ® Windows OS Loader, WINLOAD.EXE, is an operating system loader which loads the Windows 8, Windows RT, Windows Server 2012, Windows Storage Server 2012, and Windows Phone 8 operating system kernel (ntoskrnl.exe) and other boot stage binary image files. Throughout this document, the BitLocker Windows OS Loader may be called the Windows OS Loader for short.

1.1 List of Cryptographic Module Binary Executables

WINLOAD.EXE – Version 6.2.9200 for Windows 8, Windows RT, Windows Server 2012, Windows Storage Server 2012, and Windows Phone 8 on systems using conventional BIOS
WINLOAD.EFI – Version 6.2.9200 for Windows 8, Windows RT, Windows Server 2012, Windows Storage Server 2012, and Windows Phone 8 on systems using UEFI firmware

Note: both versions of winload exist on all platforms. The firmware determines which is used.

1.2 Brief Module Description

BitLocker Windows OS Loader is the binary executable for loading the Windows operating system.

1.3 Validated Platforms

The BitLocker Windows OS Loader components listed in Section 1.1 were validated using the following machine configurations:

x86 Microsoft Windows 8 Enterprise – Dell Dimension C521 (AMD Athlon 64 X2 Dual Core)
x64 Microsoft Windows 8 Enterprise – Dell PowerEdge SC430 (Intel Pentium D without AES-NI)
x64-AES-NI Microsoft Windows 8 Enterprise – Intel Client Desktop (Intel Core i7 with AES-NI)
x64 Microsoft Windows Server 2012 – Dell PowerEdge SC430 (Intel Pentium D without AES-NI)
x64-AES-NI Microsoft Windows Server 2012 – Intel Client Desktop (Intel Core i7 with AES-NI)
ARMv7 Thumb-2 Microsoft Windows RT – NVIDIA Tegra 3 Tablet (NVIDIA Tegra 3 Quad-Core)
ARMv7 Thumb-2 Microsoft Windows RT – Qualcomm Tablet (Qualcomm Snapdragon S4)
ARMv7 Thumb-2 Microsoft Windows RT – Microsoft Surface Windows RT (NVIDIA Tegra 3 Quad-Core)
x64-AES-NI Microsoft Windows 8 Pro – Microsoft Surface Windows 8 Pro (Intel x64 Processor with AES-NI)
ARMv7 Thumb-2 Microsoft Windows Phone 8 – Windows Phone 8 (Qualcomm Snapdragon S4)
x64 Microsoft Windows Storage Server 2012 – Intel Maho Bay (Intel Core i7 without AES-NI)
x64-AES-NI Microsoft Windows Storage Server 2012 – Intel Maho Bay (Intel Core i7 with AES-NI)

BitLocker Windows OS Loader maintains FIPS 140-2 validation compliance (according to FIPS 140-2 PUB Implementation Guidance G.5) on the following platforms:

x86 Microsoft Windows 8
x86 Microsoft Windows 8 Pro

x64 Microsoft Windows 8
x64 Microsoft Windows 8 Pro
x64 Microsoft Windows Server 2012 Datacenter

x64-AES-NI Microsoft Windows 8
x64-AES-NI Microsoft Windows 8 Pro
x64-AES-NI Microsoft Windows Server 2012 Datacenter

1.4 Cryptographic Boundary

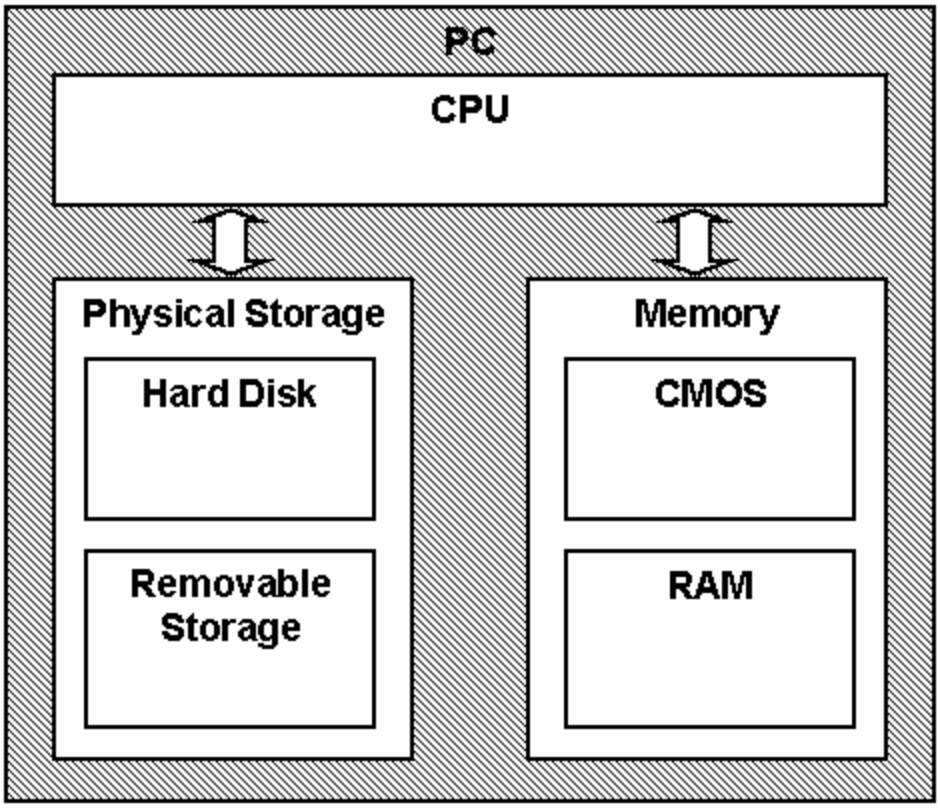
The software binary that comprises the cryptographic boundary for Windows OS Loader is Winload or Winload.efi depending on the CPU architecture. The Crypto boundary is also defined by the enclosure of the computer system, on which Windows OS Loader is to be executed. The physical configuration of Windows OS Loader, as defined in FIPS-140-2, is multi-chip standalone.

2 Security Policy

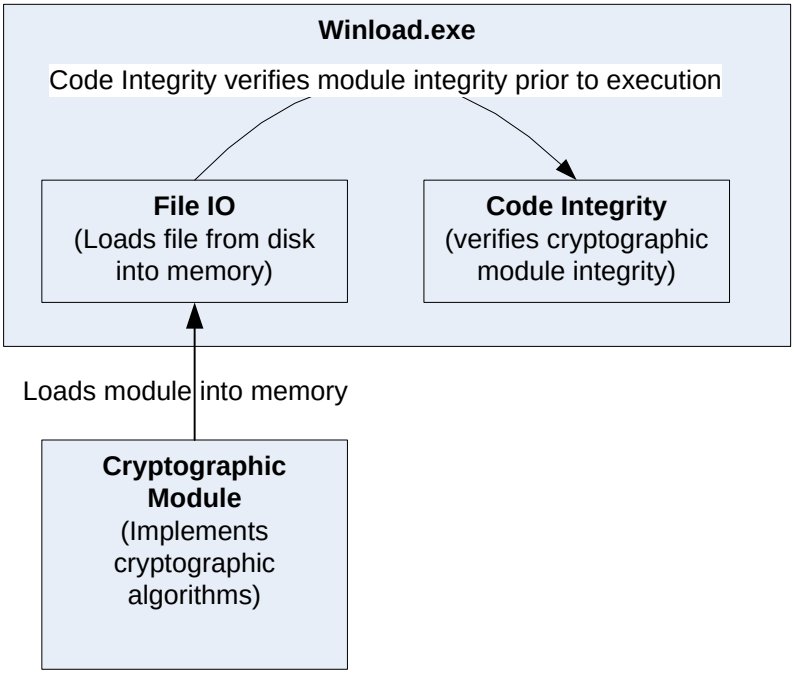
Windows OS Loader operates under several rules that encapsulate its security policy.

- Windows OS Loader is validated on the platforms listed in Section 1.3.
- Windows OS Loader operates in FIPS mode of operation only when used with the FIPS validated version of Windows 8, Windows RT, Windows Server 2012, Windows Storage Server 2012, and Windows Phone 8 Boot Manager (bootmgr) validated to FIPS 140-2 under Cert. #1895, respectively, operating in FIPS mode.
- Windows 8, Windows RT, Windows Server 2012, Windows Storage Server 2012, and Windows Phone 8 are operating systems supporting a “single user” mode where there is only one interactive user during a logon session.
- Windows OS Loader is only in its Approved mode of operation when Windows is booted normally, meaning Debug mode is disabled and Driver Signing enforcement is enabled.
- The Debug mode status and Driver Signing enforcement status can be viewed by using the bcdedit tool.

The following diagram illustrates the master components of the Windows OS Loader module:



The following diagram illustrates Windows OS Loader module interaction with other cryptographic modules:



- Windows OS Loader's main service is to load the Windows 8, Windows RT, Windows Server 2012, Windows Storage Server 2012, and Windows Phone 8 operating system kernel (ntoskrnl.exe) and other boot stage binary image files, including Code Integrity cryptographic module (ci.dll), after it validates their integrity using its cryptographic algorithm implementations using the FIPS 140-2 approved algorithms mentioned below. After the verified kernel and boot stage binary image files, including Code Integrity, are loaded, Windows OS Loader passes the execution control to the kernel and it terminates its own execution. In addition to this service, Windows OS Loader also provides status and self-test services. The Crypto officer and User have access to all services WINLOAD supports.
- If the integrity of the kernel or Code Integrity is not verified, Windows OS Loader does not transfer the execution to the kernel.
- The module provides a power-up self-tests service that is automatically executed when the module is loaded into memory. The module also 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 GPC monitor or to log files.
- Windows OS Loader verifies the integrity of multiple kernel mode crypto modules. This verification relies on RSA 2048-bit signature verification using SHA-256. If the verification fails, the modules are not loaded into memory, and this will prevent Windows from booting. The following binaries are verified in this manner:
    - CI.DLL
    - CNG.SYS

2.1 FIPS 140-2 Approved Algorithms

Windows OS Loader implements the following FIPS-140-2 Approved algorithms:

- RSA PKCS#1 (v1.5) digital signature verification (Cert. # 1132)
    - RSA signature verification with 1024-bit keys and SHA-1 message digest
    - RSA signature verification with 2048-bit keys and SHA-256 message digest
- SHS (SHA-1) (Cert. # 1903)
- SHS (SHA-256) (Cert. # 1903)
- SHS (SHA-384) (Cert. # 1903)
- SHS (SHA-512) (Cert. # 1903)
- AES (Certs. # 2196 and # 2198)
- AES (Cert. #2197; non-compliant)

Note: AES (Cert. #2197) is only used in the module's entropy source. This particular instance of AES is labeled as non-compliant because it does not perform a power-up self-test.

2.2 Non-Approved Algorithms

Windows OS Loader implements the following non-Approved algorithms:

- MD5

Windows OS Loader has a legacy implementation of MD5 for backwards compatibility with the verification of the certificate chain of old certificates that might have been used by certificate authorities (CAs) to sign certificates on kernel mode drivers outside of Windows. This legacy implementation of

MD5 is not used for checking the integrity of this cryptographic module nor any other Windows cryptographic module. Windows OS Loader has an NDRNG that is a non-Approved, but allowed algorithm.

2.3 Cryptographic Bypass

Cryptographic bypass is not supported by Windows OS Loader.

2.4 Machine Configurations

Windows OS Loader was tested using the machine configurations listed in Section 1.3 - Validated Platforms.

3 Operational Environment

The operational environment for Windows OS Loader is Windows 8, Windows RT, Windows Server 2012, Windows Storage Server 2012, and Windows Phone 8 running on the hardware listed in Section 1.3 - Validated Platforms.

Windows OS Loader services are only available before the startup of the operating system. This is done inside the Trusted Computing Base (TCB).

4 Integrity Chain of Trust

4.1 Conventional BIOS and UEFI without Secure Boot Enabled

Boot Manager is the start of the chain of trust. It cryptographically checks its own integrity during its startup. It then cryptographically checks the integrity of the Windows OS Loader before starting it. The Windows OS Loader then checks the integrity of the Code Integrity crypto module, the operating system kernel, and other boot stage binary images. An RSA signature with a 2048-bit key and SHA-256 message digest are used.

4.2 UEFI with Secure Boot Enabled

On UEFI systems with Secure Boot enabled, Boot Manager is still the OS binary from which the integrity of all other OS binaries is rooted, and it does cryptographically check its own integrity. However, Boot Manager's integrity is also checked and verified by the UEFI firmware, which is the root of trust on Secure Boot enabled systems. An RSA signature with a 2048-bit key and SHA-256 message digest are used.

5 Ports and Interfaces

5.1 Control Input Interface

The Windows OS Loader Control Input Interface is the set of internal functions responsible for intercepting control input. These functions are:

- BlBdInitialize – Reads the system status to determine if a boot debugger is attached.
- OslMain – This function receives and parses the Boot Application parameters, which are passed to the module when execution is passed from Boot Manager.
- BlInitializeLibrary – Performs the parsing Boot Application parameters.
- BlXmiRead – Reads the operator selection from the Winload user interface.

5.2 Status Output Interface

The Status Output Interface is the BlXmiWrite function that is responsible for displaying the integrity verification errors to the screen. The Status Output Interface is also defined as the BlLogData responsible for writing the name of the corrupt driver to the bootlog.

5.3 Data Output Interface

The Data Output Interface is represented by the OslArchTransferToKernel function and the AhCreateLoadOptionsString function. OslArchTransferToKernel is responsible for transferring the execution from Winload to the initial execution point of the Windows 8, Windows RT, Windows Server 2012, Windows Storage Server 2012, and Windows Phone 8 kernel. Data exits the module in the form of the initial instruction address of the Windows 8, Windows RT, Windows Server 2012, Windows Storage Server 2012, and Windows Phone 8 kernel.

Data exits the module from the AhCreateLoadOptionsString function in the form of boot application parameters passed to the Windows 8, Windows RT, Windows Server 2012, Windows Storage Server 2012, and Windows Phone 8 kernel.

5.4 Data Input Interface

The Data Input Interface is represented by the BlFileReadEx function and the BlDeviceRead function. BlFileReadEx is responsible for reading the binary data of unverified components from the computer hard drive. In addition the BitLocker Full Volume Encryption Key (FVEK) can also be entered into the module over the module's data input interface. BlDeviceRead is responsible for reading data directly from devices.

6 Specification of Roles

Windows OS Loader supports both User and Cryptographic Officer roles (as defined in FIPS-140-2). Both roles have access to all services implemented in Windows OS Loader. The module does not implement any authentication services. Therefore, roles are assumed implicitly by booting the Windows 8, Windows RT, Windows Server 2012, Windows Storage Server 2012, and Windows Phone 8 operating systems.

6.1 Maintenance Roles

Maintenance roles are not supported.

6.2 Multiple Concurrent Interactive Operators

There is only one interactive operator in Single User Mode. When run in this configuration, multiple concurrent interactive operators are not supported.

6.3 Show Status Services

The User and Cryptographic Officer roles have the same Show Status functionality, which is, for each function, the status information is returned to the caller as the return value from the function.

6.4 Self-Test Services

The User and Cryptographic Officer roles have the same Self-Test functionality, which is described in Section 10 Self-Tests.

6.5 Service Inputs / Outputs

The User and Cryptographic Officer roles have service inputs and outputs as specified in Section 5 Ports and Interfaces.

7 Services

Windows OS Loader services are described in Section 2 above. Windows OS Loader does not export any cryptographic functions.

In addition to the services described in Section 2, Windows OS Loader also implements an entropy source which is used by subsequently-loaded Windows components. This source gathers entropy from the following sources:

1. The contents of the registry value HKLM\System\RNG\Seed, which is written by the Kernel Mode Cryptographic Primitives Library (cng.sys) during its normal operation.
2. The contents of the registry value HKL\System\RNG\ExternalEntropy, which can be populated by system administrators. This value is overwritten after reading, to ensure that it does not get reused.
3. If a Trusted Platform Module (TPM) is available, the output of a TPM_GetRandom call to the TPM.
4. The current system time.
5. The contents of the OEM0 ACPI table in the machine firmware.
6. If the CPU supports the RDRAND CPU instruction, the output of such an operation.
7. If booted from UEFI firmware which supports the UEFI entropy protocol, the output of the UEFI random number generator.
8. The CPU timings.

These inputs are then combined using SHA-512, and the entropy source is conditioned using a non-Approved RNG. From this NDRNG, a block of output bytes is passed to the Windows kernel at boot time.

The module generates cryptographic keys whose strengths are modified by available entropy.

8 Authentication

The Windows OS Loader does not implement any authentication services. The User and Cryptographic Officer roles are assumed implicitly by booting the Windows operating system.

9 Cryptographic Key Management

Windows OS Loader does not store any secret or private cryptographic keys across power-cycles. However, it does use two AES keys in support of the BitLocker feature. These keys are:

- Full Volume Encryption Key (FVEK) - 128 or 256-bit AES key that is used to decrypt data on disk sectors of the hard drive.

This key is stored in memory and is zeroized by power-cycling the OS.

The key enters the module via machine memory; a pointer to this memory is provided to Winload by Boot Manager after it verifies the integrity of Winload.

Windows OS Loader also uses public keys stored on the computer hard disk to verify digital signatures using its implementation of RSA PKCS#1 (v1.5) verify. These public keys are available to both roles. Zeroization is performed by deleting the Winload module.

9.1 Access Control Policy

All the keys (mentioned above) are accessed only by the Windows OS Loader service that loads the operating system kernel (ntoskrnl.exe) and other boot stage binary image files, including Code Integrity. This service only has execute access to the keys mentioned above. For this reason, an access control policy table is not included in this document.

10 Self-Tests

10.1 Power-On Self-Tests

Windows OS Loader performs the following power-on (startup) self-tests:

- SHS (SHA-1) Known Answer Test
- SHS (SHA-256) Known Answer Test
- SHS (SHA-512) Known Answer Test
- RSA PKCS#1 (v1.5) verify with public key
    - RSA signature with 1024-bit key and SHA-1 message digest
    - RSA signature with 2048-bit key and SHA-256 message digest
- AES Known Answer Tests

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

10.2 Conditional Self-Tests

Windows OS Loader performs the following conditional self-test:

Non-Approved RNG CRNGT (entropy pool)

11 Design Assurance

The secure installation, generation, and startup procedures of this cryptographic module are part of the overall Windows 8, Windows RT, Windows Server 2012, and Windows Storage Server 2012 operating system secure installation, configuration, and startup procedures. After the operating system has been installed, it must be configured by enabling the "System cryptography: Use FIPS compliant algorithms for encryption, hashing, and signing" policy setting followed by restarting the system. This procedure is all the crypto officer and user behavior necessary for the secure operation of this cryptographic module.

Windows Phone 8 does not use the same installation, configuration, and startup procedures as the Windows operating system on a computer, but rather, is securely installed and configured by the cellular telephone carrier.

The procedures required for maintaining security while distributing and delivering versions of a cryptographic module to authorized operators are:

1. The secure distribution method is via the physical medium for product installation delivered by Microsoft Corporation, which is a DVD in the case of Windows 8 and Windows Server 2012. In the case of Windows RT, Surface Windows RT, Surface Windows 8 Pro, Windows Phone 8, and Windows Storage Server 2012, the cryptographic module is already installed at the factory and is only distributed with the hardware.
2. An inspection of authenticity of the physical medium can be made by following the guidance at this Microsoft web site: http://www.microsoft.com/en-us/howtotell/default.aspx
3. The installed version of Windows 8, Windows RT, Windows Server 2012, and Windows Storage Server 2012 must be verified 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 by Microsoft certificates. The Windows Update client only accepts content whose 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.

12 Mitigation of Other Attacks

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

Algorithm	Protected Against	Mitigation	Comments
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 used with AES NI

13 Additional Details

For the latest information on Microsoft Windows, check out the Microsoft web site at:

http://windows.microsoft.com

For more information about FIPS 140 evaluations of Microsoft products, please see:

http://technet.microsoft.com/en-us/library/cc750357.aspx

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

14.1 How to Verify Windows Versions

The installed version of Windows 8, Windows RT, Windows Server 2012, and Windows Storage Server 2012 must be verified to match the version that was validated using one of the following methods:

1. The ver command
   a. From Start, open the Search charm.
   b. In the search field type "cmd" and press the Enter key.
   c. The command window will open with a "C:\>" prompt.
   d. At the prompt, type "ver" and press the Enter key.
   e. You should see the answer "`Microsoft Windows [Version 6.2.9200]`".
2. The systeminfo command
   a. From Start, open the Search charm.
   b. In the search field type "cmd" and press the Enter key.
   c. The command window will open with a "C:\>" prompt.
   d. At the prompt, type "systeminfo" and press the Enter key.
   e. Wait for the information to be loaded by the tool.
   f. Near the top of the output, you should see:

```
OS Name:                   Microsoft Windows 8 Enterprise
OS Version:                6.2.9200 N/A Build 9200
OS Manufacturer:           Microsoft Corporation
```

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

14.2 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: x.x.xxxx.xxxxx.
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.