Security Policy for FIPS 140-2 Validation

BitLocker® Dump Filter (dumpfve.sys) in
Microsoft Windows 8.1 Enterprise
Windows Server 2012 R2
Windows Storage Server 2012 R2
Surface Pro 3
Surface Pro 2
Surface Pro
Surface 2
Surface
Windows RT 8.1
Windows Phone 8.1
Windows Embedded 8.1 Industry Enterprise
StorSimple 8000 Series

DOCUMENT INFORMATION

Version Number	2.0
Updated On	April 22, 2015

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.

CHANGE HISTORY

Date	Version	Updated By	Change
29 OCT 2013	0.1	Tim Myers	First Draft
7 MAY 2014	0.2	Tim Myers	Second Draft; processor name updates
1 JUL 2014	0.3	Tim Myers	Third Draft; added Security Levels Table and cryptographic algorithm certificate numbers
10 JUL 2014	1.0	Tim Myers	First Release to Validators; includes all algorithm and module certificate numbers
18 JUL 2014	1.1	Tim Myers	Update platforms; consolidate services
12 DEC 2014	1.2	Tim Myers	Address CMVP comments; update platform names
17 DEC 2014	1.3	Tim Myers	Update IG G.5 platforms; update binary versions; add StorSimple
22 JAN 2015	1.4	Tim Myers	Update copyright
29 JAN 2015	1.5	Tim Myers	Update Design Assurance
20 FEB 2015	1.6	Tim Myers	Add StorSimple to Validated Platforms
9 MAR 2015	1.7	Tim Myers	Prepare for publication
17 MAR 2015	1.8	Tim Myers	Reorder StorSimple 8100 OE description; fix typos in Section 10
14 APR 2015	1.9	Tim Myers	Specify AES modes in approved algorithms
22 APR 2015	2.0	Tim Myers	Add Microsoft Surface Pro 3 to Validated Platforms

TABLE OF CONTENTS

1 Introduction

BitLocker® Drive Encryption is a data protection feature available in Microsoft Windows 8.1 Enterprise, Windows Server 2012 R2, Windows Storage Server 2012 R2, Surface Pro 3, Surface Pro 2, Surface Pro, Surface 2, Surface, Windows RT 8.1, Windows Phone 8.1, Windows Embedded 8.1 Industry Enterprise, and StorSimple 8000 Series (herein referred to as Windows 8.1 OEs). BitLocker is Microsoft's response to one of our top customer requests: address the very real threats of data theft or exposure from lost, stolen or inappropriately decommissioned computer hardware with a tightly integrated solution in the Windows Operating System.

BitLocker prevents an attacker who boots another operating system or runs a software hacking tool from breaking Windows file and system protections or performing offline viewing of the files stored on the protected drive. This protection is achieved by encrypting the entire Windows volume. With BitLocker all user and system files are encrypted including the swap and hibernation files.

BitLocker ideally uses a Trusted Platform Module (TPM 1.2 or 2.0) to protect user data and to ensure that a computer running Windows 8.1 OEs has not been tampered with while the system was offline. BitLocker provides its users enhanced data protection should their systems be lost or stolen, and more secure data deletion when it comes time to decommission those assets. BitLocker enhances data protection by bringing together two major sub-functions: full drive encryption and the integrity checking of early boot components.

Integrity checking the early boot components helps to ensure that data decryption is performed only if those components appear unmolested and that the encrypted drive is located in the original computer.

BitLocker offers the option to lock the normal boot process until the user supplies a PIN, much like an ATM card PIN, or inserts a USB flash drive that contains keying material. These additional security measures provide multi-factor authentication and assurance that the computer will not boot or resume from hibernation until the correct PIN or USB flash drive is presented.

This security policy document describes the BitLocker Dump Filter cryptographic module which protects hibernation files and crash dump files on BitLocker encrypted volumes. For BitLocker security policy details related to boot components, see the security policy documents for Boot Manager, Windows OS Loader, and Windows OS Resume.

1.1 List of Cryptographic Module Binary Executables

DUMPFVE.SYS – Versions 6.3.9600 and 6.3.9600.17031 for Windows 8.1 OEs

1.2 Brief Module Description

The BitLocker Dump Filter is the full volume encryption filter that sits in the system dump stack. Whenever the dump stack is called (in the event of a crash or for hibernation), this filter ensures that all data is encrypted before it gets written to the disk as a dump file or hibernation file.

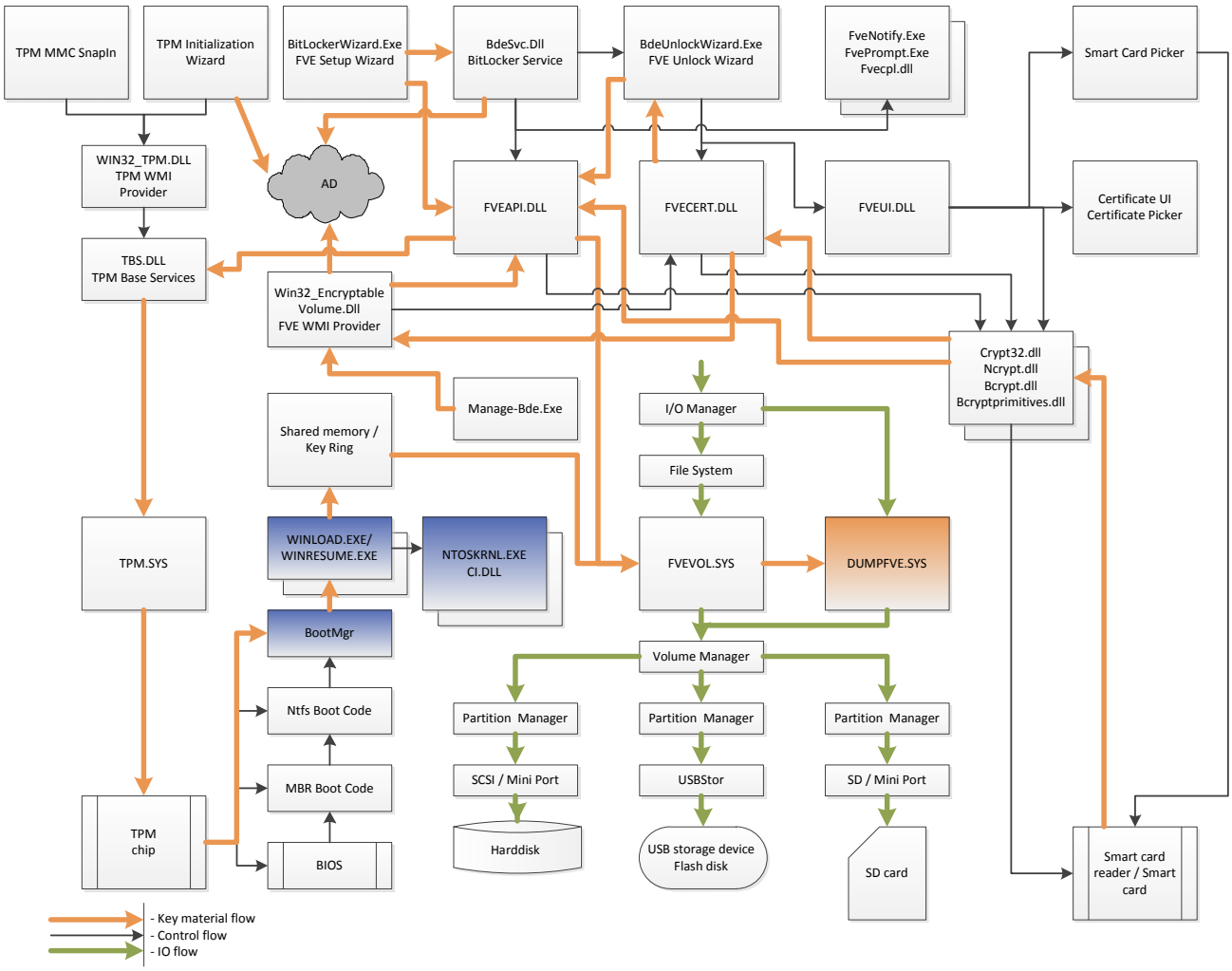

Figure 1 - Logical Operation of Module (this cryptographic module is in orange, other BitLocker-related cryptographic modules are in blue)

1.3 Validated Platforms

The BitLocker Dump Filter component listed in Section 1.1 was validated using the following machine configurations:

1. Microsoft Windows 8.1 Enterprise (x86) running on a Dell PowerEdge SC440 without AES-NI;
2. Microsoft Windows Embedded 8.1 Industry Enterprise (x86) running on a Dell PowerEdge SC440 without AES-NI;
3. Microsoft Windows 8.1 Enterprise (x86) running on a Dell Dimension E521 without AES-NI;
4. Microsoft Windows Embedded 8.1 Industry Enterprise (x86) running on a Dell Dimension E521 without AES-NI;
5. Microsoft Windows 8.1 Enterprise (x86) running on an Intel Core i7 with AES-NI running on an Intel Maho Bay;
6. Microsoft Windows Embedded 8.1 Industry Enterprise (x86) running on an Intel Core i7 with AES-NI running on an Intel Maho Bay;
7. Microsoft Windows 8.1 Enterprise (x86) running on an HP Compaq Pro 6305 with AES-NI;
8. Microsoft Windows Embedded 8.1 Industry Enterprise (x86) running on an HP Compaq Pro 6305 with AES-NI;
9. Microsoft Windows 8.1 Enterprise (x64) running on a Dell PowerEdge SC440 without AES-NI;
10. Microsoft Windows Embedded 8.1 Industry Enterprise (x64) running on a Dell PowerEdge SC440 without AES-NI;
11. Microsoft Windows Server 2012 R2 (x64) running on a Dell PowerEdge SC440 without AES-NI;
12. Microsoft Windows Storage Server 2012 R2 (x64) running on a Dell PowerEdge SC440 without AES-NI;
13. Microsoft Windows 8.1 Enterprise (x64) running on a Dell Dimension E521 without AES-NI;
14. Microsoft Windows Embedded 8.1 Industry Enterprise (x64) running on a Dell Dimension E521 without AES-NI;
15. Microsoft Windows Server 2012 R2 (x64) running on a Dell Dimension E521 without AES-NI;
16. Microsoft Windows Storage Server 2012 R2 (x64) running on a Dell Dimension E521 without AES-NI;
17. Microsoft Windows 8.1 Enterprise (x64) running on an Intel Core i7 with AES-NI running on an Intel Maho Bay;
18. Microsoft Windows Embedded 8.1 Industry Enterprise (x64) running on an Intel Core i7 with AES-NI running on an Intel Maho Bay;
19. Microsoft Windows Server 2012 R2 (x64) running on an Intel Core i7 with AES-NI running on an Intel Maho Bay;
20. Microsoft Windows Storage Server 2012 R2 (x64) running on an Intel Core i7 with AES-NI running on an Intel Maho Bay;
21. Microsoft Windows 8.1 Pro (x64) running on an Intel x64 Processor with AES-NI running on a Microsoft Surface Pro;
22. Microsoft Windows 8.1 Pro (x64) running on an Intel i5 with AES-NI running on a Microsoft Surface Pro 2;
23. Microsoft Windows 8.1 Enterprise (x64) running on an HP Compaq Pro 6305 with AES-NI;
24. Microsoft Windows Embedded 8.1 Industry Enterprise (x64) running on an HP Compaq Pro 6305 with AES-NI;
25. Microsoft Windows Server 2012 R2 (x64) running on an HP Compaq Pro 6305 with AES-NI;
26. Microsoft Windows Storage Server 2012 R2 (x64) running on an HP Compaq Pro 6305 with AES-NI;

27. Microsoft Windows RT 8.1 (ARMv7 Thumb-2) running on an NVIDIA Tegra 3 Tablet;
28. Microsoft Windows RT 8.1 (ARMv7 Thumb-2) running on a Microsoft Surface RT;
29. Microsoft Windows RT 8.1 (ARMv7 Thumb-2) running on a Microsoft Surface 2;
30. Microsoft Windows RT 8.1 (ARMv7 Thumb-2) running on a Qualcomm Tablet;
31. Microsoft Windows Phone 8.1 (ARMv7 Thumb-2) running on a Qualcomm Snapdragon S4 running on a Windows Phone 8.1;
32. Microsoft Windows Phone 8.1 (ARMv7 Thumb-2) running on a Qualcomm Snapdragon 400 running on a Windows Phone 8.1;
33. Microsoft Windows Phone 8.1 (ARMv7 Thumb-2) running on a Qualcomm Snapdragon 800 running on a Windows Phone 8.1;
34. Microsoft Windows 8.1 Enterprise (x64) running on a Dell Inspiron 660s without AES-NI and with PCLMULQDQ and SSSE 3;
35. Microsoft Windows Embedded 8.1 Industry Enterprise (x64) running on a Dell Inspiron 660s without AES-NI and with PCLMULQDQ and SSSE 3;
36. Microsoft Windows Server 2012 R2 (x64) running on a Dell Inspiron 660s without AES-NI and with PCLMULQDQ and SSSE 3;
37. Microsoft Windows Storage Server 2012 R2 (x64) running on a Dell Inspiron 660s without AES-NI and with PCLMULQDQ and SSSE 3;
38. Microsoft Windows 8.1 Enterprise (x64) running on an Intel Core i5 with AES-NI and with PCLMULQDQ and SSSE 3 running on a Microsoft Surface Pro 2;
39. Microsoft Windows Embedded 8.1 Industry Enterprise (x64) running on an Intel Core i7 with AES-NI and with PCLMULQDQ and SSSE 3 running on an Intel Maho Bay;
40. Microsoft Windows Server 2012 R2 (x64) running on an Intel Core i7 with AES-NI and with PCLMULQDQ and SSSE 3 running on an Intel Maho Bay;
41. Microsoft Windows Storage Server 2012 R2 (x64) running on an Intel Core i7 with AES-NI and with PCLMULQDQ and SSSE 3 running on an Intel Maho Bay;
42. Microsoft Windows 8.1 Enterprise (x64) running on an HP Compaq Pro 6305 with AES-NI and with PCLMULQDQ and SSSE 3;
43. Microsoft Windows Embedded 8.1 Industry Enterprise (x64) running on an HP Compaq Pro 6305 with AES-NI and with PCLMULQDQ and SSSE 3;
44. Microsoft Windows Server 2012 R2 (x64) running on an HP Compaq Pro 6305 with AES-NI and with PCLMULQDQ and SSSE 3;
45. Microsoft Windows Storage Server 2012 R2 (x64) running on an HP Compaq Pro 6305 with AES-NI and with PCLMULQDQ and SSSE 3;
46. Windows Server 2012 R2 (x64) running on a Microsoft StorSimple 8100 with an Intel Xeon E5-2648L without AES-NI;
47. Windows Server 2012 R2 (x64) running on a Microsoft StorSimple 8100 with an Intel Xeon E5-2648L with AES-NI;
48. Microsoft Windows 8.1 Pro (x64) running on an Intel Core i7 with AES-NI and with PCLMULQDQ and SSSE 3 running on a Microsoft Surface Pro 3

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

x86 Microsoft Windows 8.1
x86 Microsoft Windows 8.1 Pro

x64 Microsoft Windows 8.1
x64 Microsoft Windows Server 2012 R2 Datacenter

x64-AES-NI Microsoft Windows 8.1
x64-AES-NI Microsoft Windows Server 2012 R2 Datacenter

1.4 Cryptographic Boundary

The Windows 8.1 OEs BitLocker® Dump Filter cryptographic boundary consists solely of the BitLocker Dump Filter component, DUMPFVE.SYS. The physical configuration of the BitLocker Dump Filter, as defined in FIPS-140-2, is multi-chip standalone.

2 Security Policy

BitLocker Dump Filter operates under several rules that encapsulate its security policy.

- BitLocker Dump Filter is validated on Microsoft Windows 8.1 Enterprise and Windows Embedded 8.1 Industry Enterprise running on x86 and x64.
- BitLocker Dump Filter is validated on Microsoft Windows RT 8.1 and Windows Phone 8.1 running on ARMv7 Thumb.
- BitLocker Dump Filter is validated on Microsoft Windows Server 2012 R2 and Windows Storage Server 2012 R2 running on x64.
- Windows 8.1 OEs are operating systems supporting a “single user” mode where there is only one interactive user during a logon session.
- BitLocker Dump Filter 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.
- BitLocker Dump Filter is only in its Approved mode of operation when the “System cryptography: Use FIPS compliant algorithms for encryption, hashing, and signing” policy setting is enabled.
- BitLocker Dump Filter will only operate in compliance once BitLocker volume conversion (encryption) has completed and the volume is fully encrypted.
- BitLocker Dump Filter operates in FIPS mode of operation only when used with the FIPS approved version of Windows 8.1 OEs Code Integrity (ci.dll) validated to FIPS 140-2 under Cert. #2355, respectively, operating in FIPS mode.

2.1 FIPS 140-2 Approved Algorithms

BitLocker Dump Filter implements the following FIPS 140-2 Approved algorithm.

- AES-CBC (Cert. #2832)
- AES-GCM decryption (Cert. #2832)

2.2 Cryptographic Bypass

Cryptographic bypass is not supported by BitLocker® Dump Filter.

2.3 Machine Configurations

BitLocker Dump Filter was tested using the machine configurations listed in Section 1.3 - Validated Platforms.

3 Operational Environment

The operational environment for BitLocker Dump Filter (DUMPFVE.SYS) is the Windows 8.1 OEs running on the software and hardware configurations listed in Section 1.3 - Validated Platforms.

4 Integrity Chain of Trust

Boot Manager is the start of the chain of trust for the collection of cryptographic modules that cooperate to provide the Windows feature called BitLocker®. Boot Manager cryptographically checks its own integrity during its startup. It then cryptographically checks the integrity of the Windows OS Loader or Windows OS Resume (if resuming from hibernation) before starting it. The Windows OS Loader or Windows OS Resume module then checks the integrity of the Code Integrity crypto module, the operating system kernel, and other boot stage binary images. Finally, the Code Integrity crypto module checks the integrity of the BitLocker Dump Filter.

Code Integrity verifies the integrity of the BitLocker Dump Filter using the following FIPS-140-2 Approved algorithms.

- RSA PKCS#1 (v1.5) verify with public key
- SHA-1 hash
- SHA-256 hash

BitLocker Dump Filter ensures that Windows 8.1 OEs crash dump files and/or hibernation files are encrypted on shutdown, thus ensuring the contents can only be accessed through the integrity chain of trust above.

5 Ports and Interfaces

The following block diagram show the interfaces and internal functions of the BitLocker Dump Filter (DUMPFVE.SYS) component.

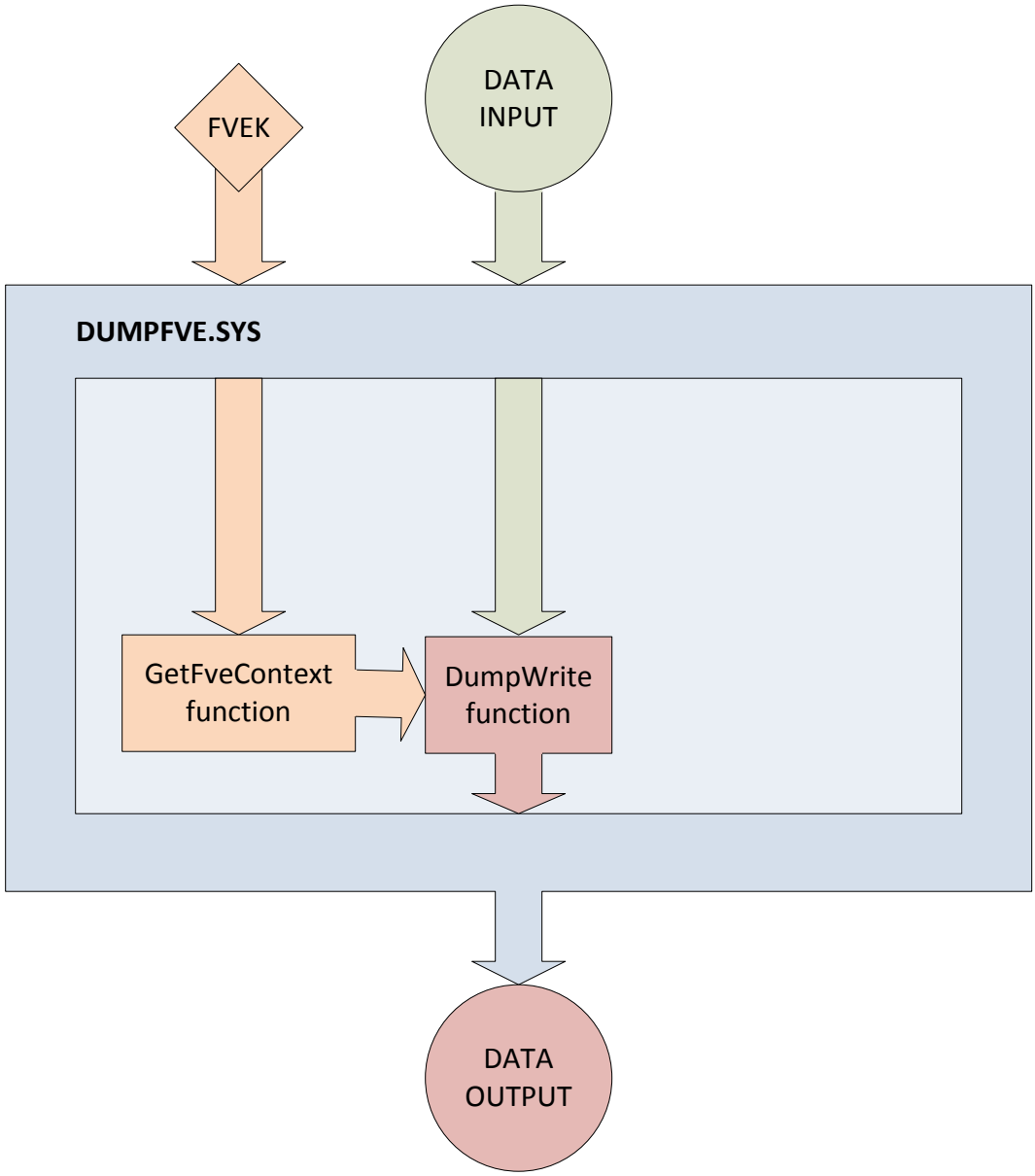

Figure 2 – BitLocker Dump Filter Block Diagram

5.1 Control Input Interface

The BitLocker Dump Filter module's control input interface consists of parameter interfaces for the GetFveContext and DumpWrite functions. These interfaces are not exported, but rather, are internal to the cryptographic module.

5.1.1 GetFveContext

```
NTSTATUS  GetFveContext(
            __in PFILTER_EXTENSION Context,
            __in ULONG MaxPagesPerWrite,
            __inout_xcount(FveContext->StructureSize) PFVE_CONTEXT FveContext
            )
```

This function gets the Full Volume Encryption Key (FVEK) for the volume. The Context parameter supplies the dump stack filter context. The FveContext parameter supplies the internal FVE context, which includes the FVE status and FVEK in this context so it can be used later when writing data to the volume.

5.1.2 DumpWrite

```
NTSTATUS  DumpWrite(
        PFILTER_EXTENSION Context,
        PLARGE_INTEGER DiskByteOffset,
        PMDL Mdl
        )
```

This function uses the FVEK from the Context parameter that is provided by the GetFveContext interface. The DiskByteOffset parameter is used to specify the location on the volume to receive the encrypted output data. The Mdl parameter points to the input data to be encrypted.

5.2 Status Output Interface

The BitLocker Dump Filter status output is a return value of type NTSTATUS that indicates whether the function completed successfully or not.

The BitLocker Dump Filter has no status output interface for self-test errors. If the self-tests pass, the module is loaded. If not, the dump filter securely zeroes out memory for any keys handed to it and unloads itself.

5.3 Data Output Interface

The Data Output Interface is the data returned from the DumpWrite function.

This function is responsible for providing the encrypted content for the crash dump file or hibernate file. Data exits the module in the form of encrypted blocks that may be written to a crash dump file or a hibernation file on an encrypted volume.

5.4 Data Input Interface

The Data Input Interface includes the GetFveContext function and DumpWrite function. GetFveContext is responsible for reading the FVEK. DumpWrite accepts the memory blocks to encrypt with the FVEK and the target disk locations for the blocks as input.

6 Specification of Roles

BitLocker Dump Filter provides two different, implicitly assumed roles and a set of services particular to each of the roles. As a FIPS 140-2 level 1 validated product, BitLocker Dump Filter itself does not provide any authentication.

Services available to the Cryptographic Officer role:

- Configure BitLocker into FIPS mode
- Write encrypted crash dump file to disk
- Write encrypted hibernation file to disk

Services available to the User role:

- Write encrypted crash dump file to disk
- Write encrypted hibernation file to disk

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.

7 Services

7.1 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.

7.2 Self-Test Services

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

7.3 Service Inputs / Outputs

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

8 Authentication

The module does not provide authentication. Roles are implicitly assumed based on the services that are executed.

9 Cryptographic Key Management

BitLocker encrypts disk sectors with a Full Volume Encryption Key (FVEK). This module receives the FVEK from Windows 8.1 OEs and uses it to encrypt crash dump files and hibernation files.

9.1 Cryptographic Keys

The BitLocker Dump Filter uses only the Full Volume Encryption Key it receives, and does not generate any cryptographic keys. It receives the necessary full volume encryption key for encrypting dump files and hibernation files from the Cryptographic Operator by way of the running system booted through the Integrity Chain of Trust. On shutdown, the FVEK is zeroized in memory (by overwriting once with 0s).

9.2 Critical Security Parameters

The BitLocker Dump Filter cryptographic module has the following Critical Security Parameter (CSP):

Critical Security Parameter	Description
Full Volume Encryption Key (FVEK)	A 128/256 bit AES key that is input into the crypto module as plaintext and is used for encryption of crash dump files and hibernation files

9.3 Access Control Policy

The BitLocker Dump Filter crypto module does not allow read or write access to the cryptographic keys contained within it. Neither role (Crypto Officer or User) sees the key within the module. Nevertheless, both roles have execute access to the FVEK. Due to the simplicity of this policy, an access control policy table is not included in this document. BitLocker Dump Filter simply automatically uses the FVEK to write crash dump and hibernation files, for the role of the User (or Crypto Officer if that applies). Since the module has to operate under an assumed role, the operator must have the FVEK in order to encrypt data to the drive.

10 Self-Tests

10.1 Power-On Self-Tests

The BitLocker Dump Filter implements Known Answer Test (KAT) functions each time the module is loaded. The module performs the following KATs:

- AES-CBC - Encrypt/Decrypt KATs
- AES-CCM - Encrypt/Decrypt KATs
- Software Integrity Test

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

11 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 8.1 OEs. The various methods of delivery and installation for each product are listed in the following table.

Product	Delivery and Installation Method
Windows 8.1	• DVD • Pre-installed on the computer by OEM • Download that updates Windows 8
Windows Server 2012 R2	• DVD • Pre-installed on the computer by OEM • Download that updates Windows Server 2012
Windows Storage Server 2012 R2	• Pre-installed by the OEM (Third party)
Surface Pro 3, Surface Pro 2, Surface Pro, Surface 2, Surface	• Pre-installed by the OEM (Microsoft)
Windows RT 8.1	• Pre-installed on the device by OEM • Download that updates Windows RT
Windows Phone 8.1	• Pre-installed on the device by OEM or mobile network operator • Download that updates Windows 8
Windows Embedded 8.1 Industry Enterprise	• Pre-installed by the OEM (Third party)
StorSimple 8000 Series	• Pre-installed by the OEM (Third party)

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.

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

The installed version of Windows 8.1 OEs 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
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 Security Levels

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

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	NA
Operational Environment	1
Cryptographic Key Management	1
EMI/EMC	1
Self-Tests	1
Design Assurance	2
Mitigation of Other Attacks	1

14 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

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

15.1 How to Verify Windows Versions

The installed version of Windows 8.1 OEs 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 6.3.9600]`".
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 8.1 Enterprise
OS Version:                6.3.9600 N/A Build 9600
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.

15.2 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.