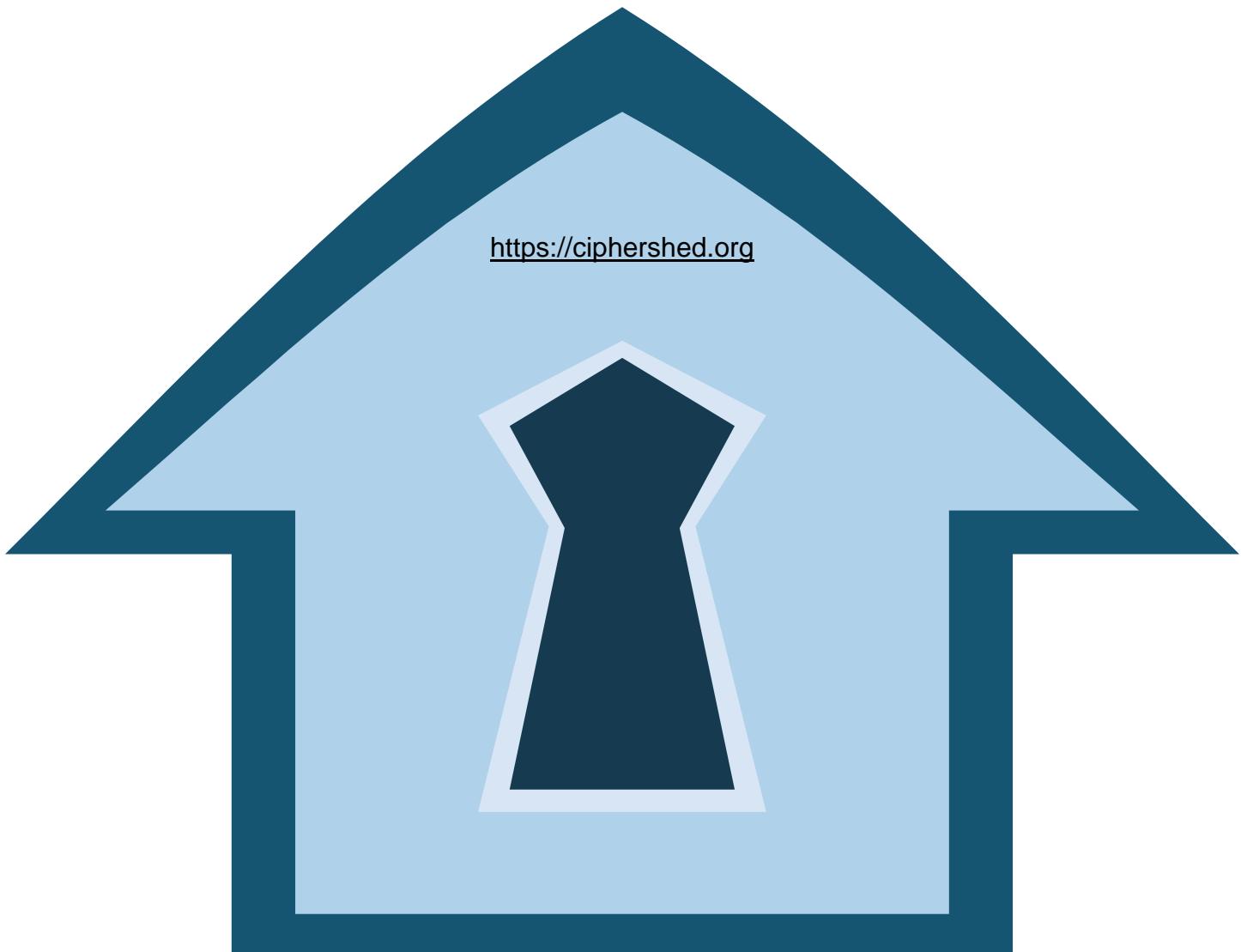


# CIPHERSHED

## SECURE ENCRYPTION SOFTWARE

### USER'S GUIDE



CipherShed User's Guide, version 0.7.3.0  
Released by CipherShed Project on 19-Dec-2014

# CipherShed Preface

This document is the user guide for the CipherShed fork of the abandoned TrueCrypt 7.1a.

The formatting of this document may differ from the last release of the “TrueCrypt User Guide.pdf<sup>1</sup>” since this document has been re-published from a re-built source by the CipherShed Project. The content has not yet been edited and is presented for historical purposes.

For additional information, please visit the CipherShed website (<https://ciphershed.org>) or contact the CipherShed Project at [devs@lists.ciphershed.org](mailto:devs@lists.ciphershed.org).

| Version | Description  |
|---------|--|
| 0.7.3.0 | Rebranded fork in compliance with the TrueCrypt License  |
| 7.1a    | Rebuild of abandoned TrueCrypt 7.1a Open Source Software, unreleased due to TrueCrypt License. |

---

<sup>1</sup> SHA1: 17249d979b3bc52d0a33821cc7f810337dde2b6

# CIPHERSHED

FREE OPEN - SOURCE ON - THE - FLY ENCRYPTION  
USER'S GUIDE

<https://ciphershed.org>

## Version Information

CipherShed User's Guide, version 0.7.3.0  
Released by CipherShed Project

## Legal Notices

THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, WHETHER EXPRESS, IMPLIED, OR STATUTORY. THE ENTIRE RISK AS TO THE QUALITY, CORRECTNESS, ACCURACY, OR COMPLETENESS OF THE CONTENT OF THIS DOCUMENT IS WITH YOU. THE CONTENT OF THIS DOCUMENT MAY BE INACCURATE, INCORRECT, INVALID, INCOMPLETE AND/OR MISLEADING. IN NO EVENT WILL ANY AUTHOR OF THE SOFTWARE OR DOCUMENTATION, OR ANY APPLICABLE COPYRIGHT OWNER, OR ANY OTHER PARTY WHO MAY COPY AND/OR (RE)DISTRIBUTE THIS SOFTWARE OR DOCUMENTATION, BE LIABLE TO YOU OR TO ANY OTHER PARTY FOR ANY DAMAGES, INCLUDING, BUT NOT LIMITED TO, ANY DIRECT, INDIRECT, GENERAL, SPECIAL, INCIDENTAL, PUNITIVE, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, CORRUPTION OR LOSS OF DATA, ANY LOSSES SUSTAINED BY YOU OR THIRD PARTIES, A FAILURE OF THIS SOFTWARE TO OPERATE WITH ANY OTHER PRODUCT, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES, OR BUSINESS INTERRUPTION), WHETHER IN CONTRACT, STRICT LIABILITY, TORT (INCLUDING, BUT NOT LIMITED TO, NEGLIGENCE) OR OTHERWISE, ARISING OUT OF THE USE, COPYING, MODIFICATION, OR (RE)DISTRIBUTION OF THIS SOFTWARE OR DOCUMENTATION (OR A PORTION THEREOF), OR INABILITY TO USE THIS SOFTWARE OR DOCUMENTATION, EVEN IF SUCH DAMAGES (OR THE POSSIBILITY OF SUCH DAMAGES) ARE/WERE PREDICTABLE OR KNOWN TO ANY (CO)AUTHOR, INTELLECTUAL-PROPERTY OWNER, OR ANY OTHER PARTY.

BY INSTALLING, RUNNING, USING, COPYING, (RE)DISTRIBUTING, AND/OR MODIFYING THIS SOFTWARE, INCLUDING, BUT NOT LIMITED TO, ITS DOCUMENTATION, OR A PORTION THEREOF, YOU ACCEPT AND AGREE TO BE BOUND BY ALL TERMS AND CONDITIONS OF THE LICENSE THE FULL TEXT OF WHICH IS CONTAINED IN THE FILE *License.txt* INCLUDED IN BINARY AND SOURCE CODE DISTRIBUTION PACKAGES.

# **Contents**

|   |           |
|---|-----------|
| <b>INTRODUCTION.....</b>  | <b>11</b> |
| <b>BEGINNER'S TUTORIAL .....</b>  | <b>12</b> |
| How to Create and Use a TrueCrypt Container.....                        | 12        |
| How to Create and Use a TrueCrypt-Encrypted Partition/Device.....       | 31        |
| <b>TRUECRYPT VOLUME.....</b>  | <b>32</b> |
| CREATING A NEW TRUECRYPT VOLUME .....                                   | 32        |
| Hash Algorithm.....   | 32        |
| Encryption Algorithm .....  | 32        |
| Quick Format .....  | 32        |
| Dynamic .....   | 33        |
| Cluster Size .....  | 33        |
| TrueCrypt Volumes on CDs and DVDs .....                                 | 33        |
| Hardware/Software RAID, Windows Dynamic Volumes .....                   | 33        |
| Additional Notes on Volume Creation .....                               | 34        |
| FAVORITE VOLUMES.....   | 34        |
| SYSTEM FAVORITE VOLUMES .....   | 36        |
| <b>SYSTEM ENCRYPTION.....</b>   | <b>38</b> |
| Hidden Operating System.....  | 38        |
| Operating Systems Supported for System Encryption.....                  | 39        |
| TrueCrypt Rescue Disk.....  | 39        |
| <b>PLAUSIBLE DENIABILITY .....</b>                                      | <b>42</b> |
| HIDDEN VOLUME .....   | 42        |
| Protection of Hidden Volumes Against Damage.....                        | 44        |
| Security Requirements and Precautions Pertaining to Hidden Volumes..... | 48        |
| HIDDEN OPERATING SYSTEM.....  | 52        |
| Process of Creation of Hidden Operating System .....                    | 54        |

|  |           |
|--|-----------|
| Plausible Deniability and Data Leak Protection .....                                     | 55        |
| Possible Explanations for Existence of Two TrueCrypt Partitions on Single Drive .....    | 56        |
| Safety/Security Precautions and Requirements Pertaining to Hidden Operating Systems..... | 57        |
| <b>MAIN PROGRAM WINDOW .....</b>   | <b>59</b> |
| Select File.....   | 59        |
| Select Device .....  | 59        |
| Mount.....   | 59        |
| Auto-Mount Devices.....  | 59        |
| Dismount.....  | 59        |
| Dismount All.....  | 59        |
| Wipe Cache.....  | 60        |
| Never Save History .....   | 60        |
| Exit.....  | 60        |
| Volume Tools.....  | 61        |
| <b>PROGRAM MENU.....</b>   | <b>61</b> |
| Volumes -> Auto-Mount All Device-Hosted Volumes .....                                    | 61        |
| Volumes -> Dismount All Mounted Volumes.....   | 61        |
| Volumes -> Change Volume Password .....  | 61        |
| Volumes -> Set Header Key Derivation Algorithm .....                                     | 62        |
| Volumes -> Add/Remove Keyfiles to/from Volume.....                                       | 62        |
| Volumes -> Remove All Keyfiles from Volume.....  | 62        |
| Favorites -> Add Mounted Volume to Favorites.....  | 62        |
| Favorites -> Organize Favorite Volumes.....  | 62        |
| Favorites -> Mount Favorites Volumes .....   | 62        |
| Favorites -> Add Mounted Volume to System Favorites .....                                | 62        |
| Favorites -> Organize System Favorite Volumes.....                                       | 62        |
| System -> Change Password.....   | 62        |

|   |           |
|---|-----------|
| System -> Mount Without Pre-Boot Authentication ..... | 62        |
| Tools -> Clear Volume History .....                   | 62        |
| Tools -> Traveler Disk Setup.....                     | 63        |
| Tools -> Keyfile Generator.....                       | 63        |
| Tools -> Backup Volume Header .....                   | 63        |
| Tools -> Restore Volume Header .....                  | 63        |
| Settings -> Preferences .....                         | 64        |
| <b>MOUNTING TRUECRYPT VOLUMES .....</b>               | <b>65</b> |
| Cache Password in Driver Memory.....                  | 65        |
| Mount Options .....                                   | 65        |
| <b>PARALLELIZATION .....</b>                          | <b>67</b> |
| <b>PIPELINING.....</b>                                | <b>68</b> |
| <b>HARDWARE ACCELERATION.....</b>                     | <b>69</b> |
| <b>HOT KEYS.....</b>                                  | <b>70</b> |
| <b>KEYFILES .....</b>                                 | <b>71</b> |
| Keyfiles Dialog Window .....                          | 71        |
| Security Tokens and Smart Cards.....                  | 72        |
| Keyfile Search Path.....                              | 73        |
| Empty Password & Keyfile .....                        | 73        |
| Quick Selection.....                                  | 73        |
| Volumes -> Add/Remove Keyfiles to/from Volume.....    | 73        |
| Volumes -> Remove All Keyfiles from Volume.....       | 74        |
| Tools -> Keyfile Generator.....                       | 74        |
| Settings -> Default Keyfiles .....                    | 74        |
| <b>SECURITY TOKENS &amp; SMART CARDS.....</b>         | <b>75</b> |
| <b>PORTABLE MODE .....</b>                            | <b>76</b> |
| Tools -> Traveler Disk Setup.....                     | 76        |

|  |           |
|--|-----------|
| <b>LANGUAGE PACKS.....</b>                         | <b>78</b> |
| Installation.....                                  | 78        |
| <b>ENCRYPTION ALGORITHMS.....</b>                  | <b>79</b> |
| AES .....  | 79        |
| Serpent .....                                      | 79        |
| Twofish .....                                      | 80        |
| AES-Twofish .....                                  | 80        |
| AES-Twofish-Serpent.....                           | 80        |
| Serpent-AES .....                                  | 80        |
| Serpent-Twofish-AES.....                           | 81        |
| Twofish-Serpent.....                               | 81        |
| <b>HASH ALGORITHMS .....</b>                       | <b>82</b> |
| RIPEMD-160 .....                                   | 82        |
| SHA-512 .....                                      | 82        |
| Whirlpool .....                                    | 82        |
| <b>SUPPORTED OPERATING SYSTEMS .....</b>           | <b>83</b> |
| <b>COMMAND LINE USAGE.....</b>                     | <b>84</b> |
| Syntax .....                                       | 87        |
| Examples.....                                      | 87        |
| <b>SECURITY MODEL.....</b>                         | <b>88</b> |
| <b>SECURITY REQUIREMENTS AND PRECAUTIONS .....</b> | <b>92</b> |
| Data Leaks .....                                   | 92        |
| Paging File .....                                  | 93        |
| Memory Dump Files.....                             | 93        |
| Hibernation File .....                             | 94        |
| Unencrypted Data in RAM .....                      | 95        |
| Physical Security.....                             | 96        |

|  |            |
|--|------------|
| Malware .....  | 97         |
| Multi-User Environment.....                            | 97         |
| Authenticity and Integrity .....                       | 98         |
| Choosing Passwords and Keyfiles .....                  | 98         |
| Changing Passwords and Keyfiles.....                   | 98         |
| Trim Operation.....                                    | 99         |
| Wear-Leveling .....                                    | 100        |
| Reallocated Sectors.....                               | 100        |
| Defragmenting .....                                    | 101        |
| Journaling File Systems .....                          | 101        |
| Volume Clones.....                                     | 101        |
| Additional Security Requirements and Precautions ..... | 101        |
| <b>HOW TO BACK UP SECURELY .....</b>                   | <b>103</b> |
| Non-System Volumes .....                               | 103        |
| System Partitions .....                                | 103        |
| General Notes.....                                     | 105        |
| <b>MISCELLANEOUS .....</b>                             | <b>106</b> |
| Using TrueCrypt Without Administrator Privileges.....  | 106        |
| Sharing over Network .....                             | 106        |
| TrueCrypt Background Task.....                         | 107        |
| Volume Mounted as Removable Medium .....               | 107        |
| TrueCrypt System Files & Application Data.....         | 108        |
| How to Remove Encryption.....                          | 109        |
| Uninstalling TrueCrypt .....                           | 110        |
| Digital Signatures.....                                | 110        |
| <b>TROUBLESHOOTING .....</b>                           | <b>113</b> |
| <b>INCOMPATIBILITIES .....</b>                         | <b>121</b> |

|  |            |
|--|------------|
| <b>KNOWN ISSUES &amp; LIMITATIONS.....</b>             | <b>122</b> |
| Known Issues.....                                      | 122        |
| Limitations .....                                      | 122        |
| <b>FREQUENTLY ASKED QUESTIONS.....</b>                 | <b>125</b> |
| <b>TECHNICAL DETAILS.....</b>                          | <b>135</b> |
| NOTATION.....  | 135        |
| ENCRYPTION SCHEME.....                                 | 135        |
| MODES OF OPERATION.....                                | 137        |
| HEADER KEY DERIVATION, SALT, AND ITERATION COUNT ..... | 138        |
| RANDOM NUMBER GENERATOR .....                          | 138        |
| KEYFILES .....   | 140        |
| TRUECRYPT VOLUME FORMAT SPECIFICATION .....            | 142        |
| COMPLIANCE WITH STANDARDS AND SPECIFICATIONS .....     | 144        |
| SOURCE CODE.....                                       | 145        |
| <b>FUTURE DEVELOPMENT.....</b>                         | <b>146</b> |
| <b>CONTACT.....</b>                                    | <b>147</b> |
| <b>LEGAL INFORMATION.....</b>                          | <b>148</b> |
| <b>VERSION HISTORY .....</b>                           | <b>149</b> |
| <b>ACKNOWLEDGEMENTS .....</b>                          | <b>150</b> |
| <b>REFERENCES.....</b>                                 | <b>151</b> |

## Preface

Please note that although most chapters of this documentation apply generally to all versions of CipherShed, some sections are primarily aimed at users of the Windows versions of CipherShed. Hence, such sections may contain information that is inappropriate in regards to the Mac OS X and Linux versions of CipherShed.

## Introduction

CipherShed is a software system for establishing and maintaining an on-the-fly-encrypted volume (data storage device). On-the-fly encryption means that data is automatically encrypted right before it is saved and decrypted right after it is loaded, without any user intervention. No data stored on an encrypted volume can be read (decrypted) without using the correct password/keyfile(s) or correct encryption keys. Entire file system is encrypted (e.g., file names, folder names, contents of every file, free space, meta data, etc.).

Files can be copied to and from a mounted CipherShed volume just like they are copied to/from any normal disk (for example, by simple drag-and-drop operations). Files are automatically being decrypted on the fly (in memory/RAM) while they are being read or copied from an encrypted CipherShed volume. Similarly, files that are being written or copied to the CipherShed volume are automatically being encrypted on the fly (right before they are written to the disk) in RAM. Note that this does not mean that the whole file that is to be encrypted/decrypted must be stored in RAM before it can be encrypted/decrypted. There are no extra memory (RAM) requirements for CipherShed. For an illustration of how this is accomplished, see the following paragraph.

Let's suppose that there is an .avi video file stored on a CipherShed volume (therefore, the video file is entirely encrypted). The user provides the correct password (and/or keyfile) and mounts (opens) the CipherShed volume. When the user double clicks the icon of the video file, the operating system launches the application associated with the file type – typically a media player. The media player then begins loading a small initial portion of the video file from the CipherShed-encrypted volume to RAM (memory) in order to play it. While the portion is being loaded, CipherShed is automatically decrypting it (in RAM). The decrypted portion of the video (stored in RAM) is then played by the media player. While this portion is being played, the media player begins loading another small portion of the video file from the CipherShed-encrypted volume to RAM (memory) and the process repeats. This process is called on-the-fly encryption/decryption and it works for all file types (not only for video files).

Note that CipherShed never saves any decrypted data to a disk – it only stores them temporarily in RAM (memory). Even when the volume is mounted, data stored in the volume is still encrypted. When you restart Windows or turn off your computer, the volume will be dismounted and files stored in it will be inaccessible (and encrypted). Even when power supply is suddenly interrupted (without proper system shut down), files stored in the volume are inaccessible (and encrypted). To make them accessible again, you have to mount the volume (and provide the correct password and/or keyfile).

# Beginner's Tutorial

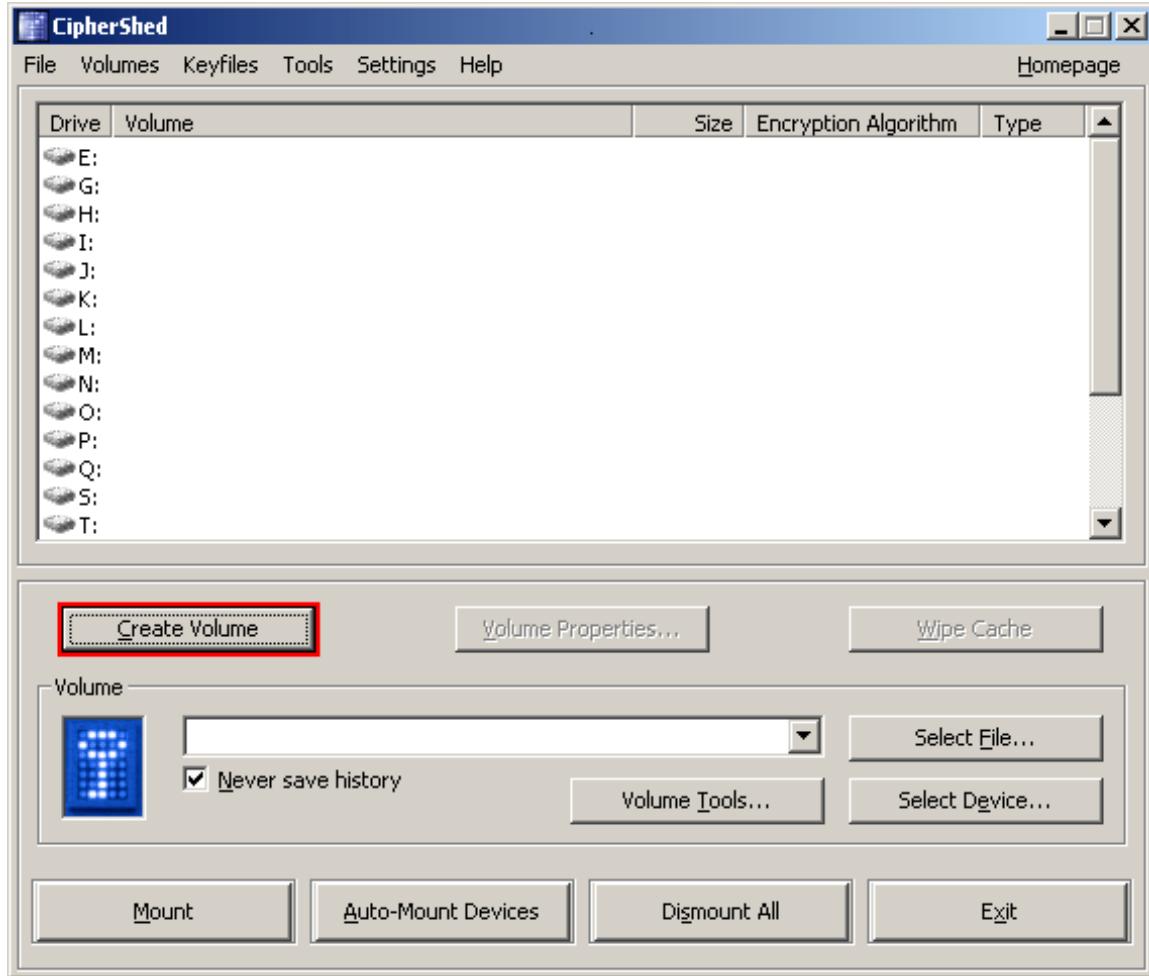
## How to Create and Use a CipherShed Container

This chapter contains step-by-step instructions on how to create, mount, and use a CipherShed volume. We strongly recommend that you also read the other sections of this manual, as they contain important information.

### Step 1:

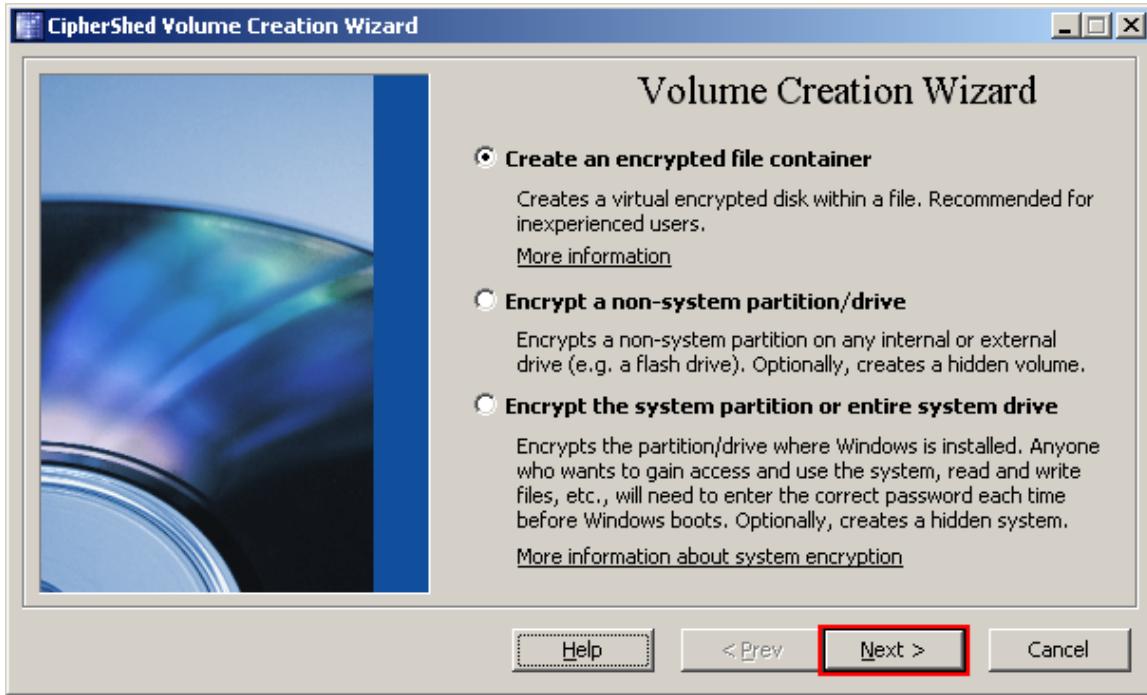
If you have not done so, download and install CipherShed. Then launch CipherShed by double-clicking the file CipherShed.exe or by clicking the CipherShed shortcut in your Windows Start menu.

### Step 2:



The main CipherShed window should appear. Click **Create Volume** (marked with a red rectangle for clarity).

### Step 3:



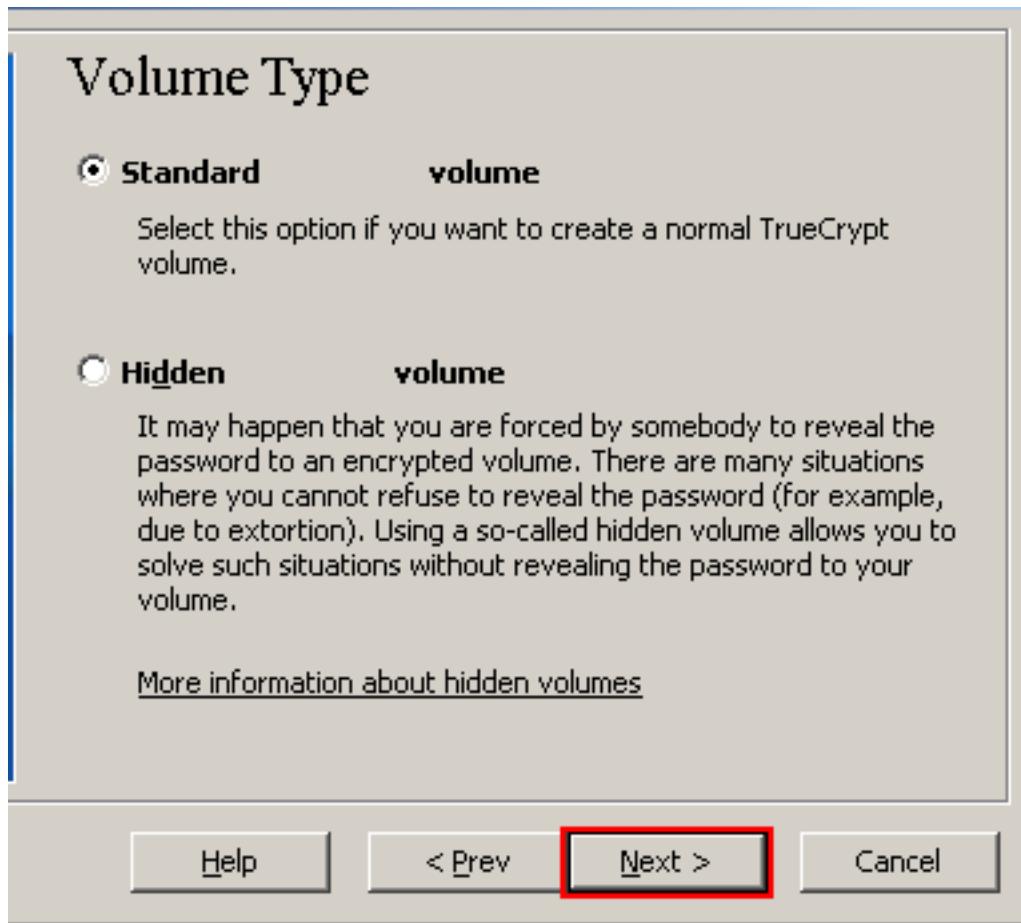
The CipherShed Volume Creation Wizard window should appear.

In this step you need to choose where you wish the CipherShed volume to be created. A CipherShed volume can reside in a file, which is also called container, in a partition or drive. In this tutorial, we will choose the first option and create a CipherShed volume within a file.

As the option is selected by default, you can just click **Next**.

Note: In the following steps, the screenshots will show only the right-hand part of the Wizard window.

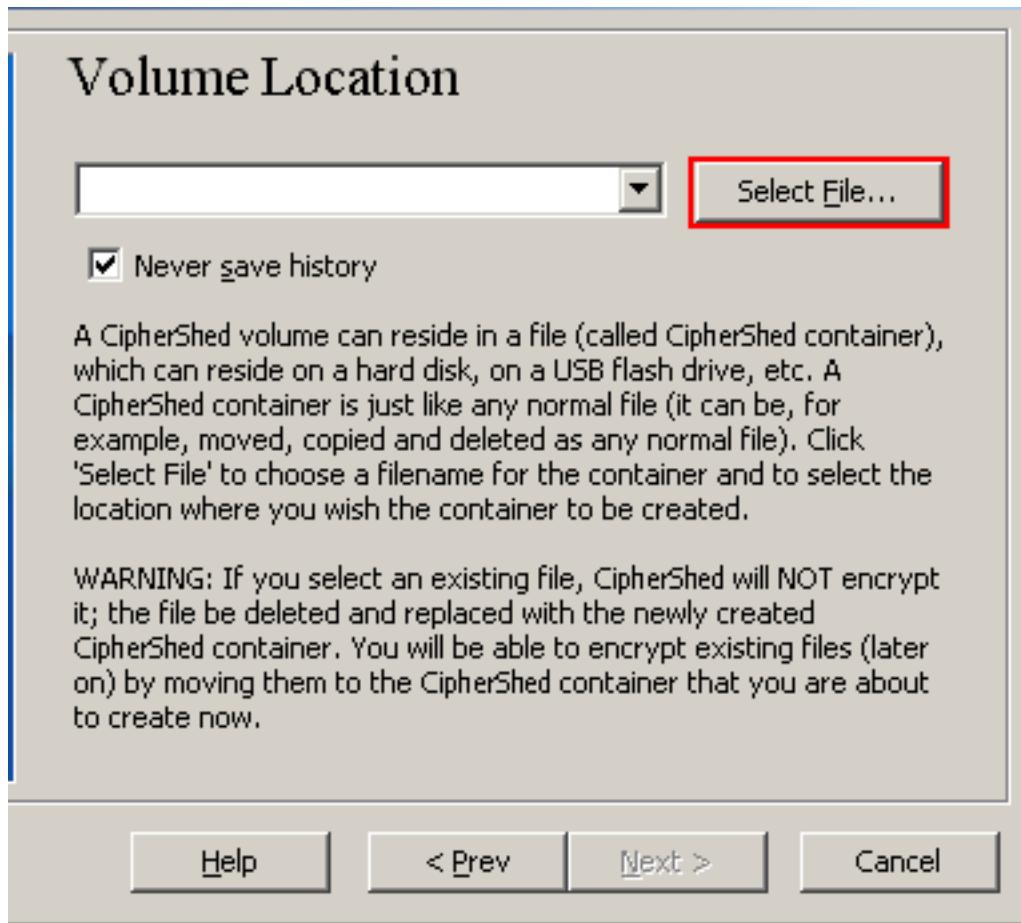
#### Step 4:



In this step you need to choose whether to create a standard or hidden CipherShed volume. In this tutorial, we will choose the former option and create a standard CipherShed volume.

As the option is selected by default, you can just click **Next**.

## Step 5:

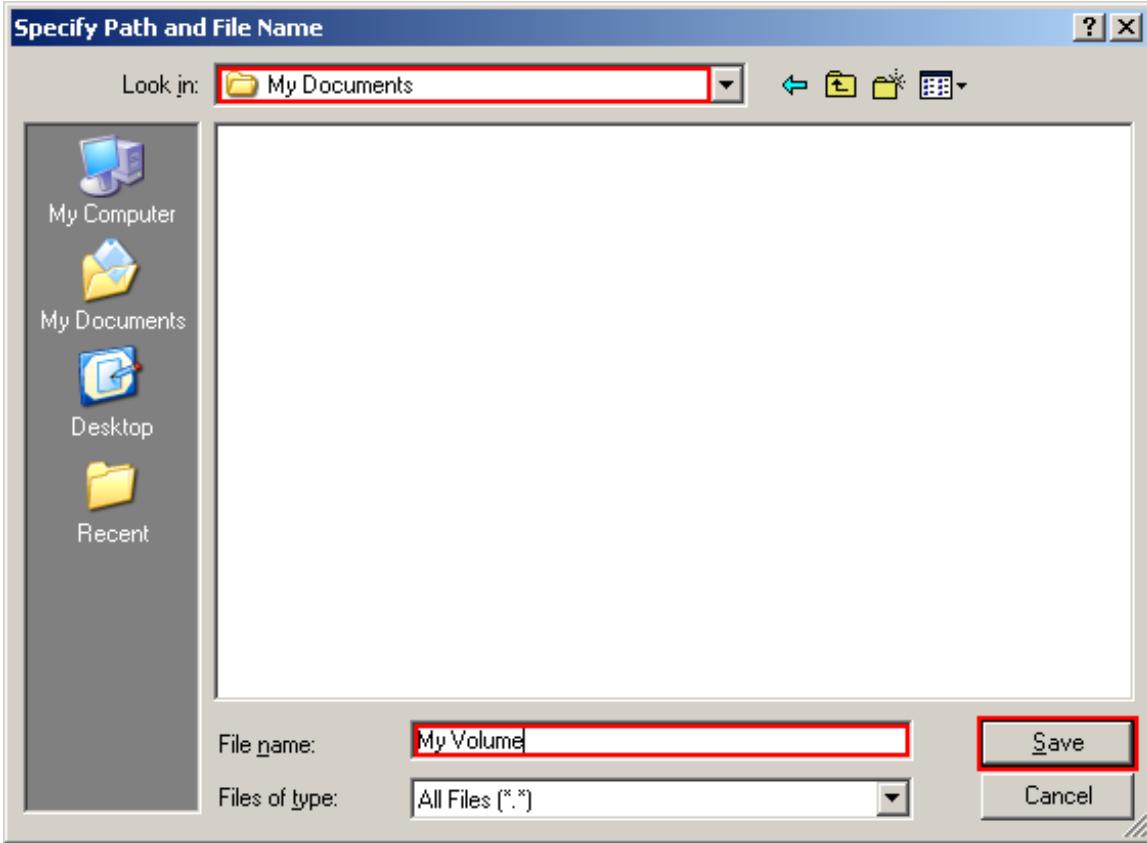


In this step you have to specify where you wish the CipherShed volume (file container) to be created. Note that a CipherShed container is just like any normal file. It can be, for example, moved or deleted as any normal file. It also needs a filename, which you will choose in the next step.

**Click Select File.**

The standard Windows file selector should appear (while the window of the CipherShed Volume Creation Wizard remains open in the background).

## Step 6:



In this tutorial, we will create our CipherShed volume in the folder D:\My Documents\ and the filename of the volume (container) will be My Volume (as can be seen in the screenshot above). You may, of course, choose any other filename and location you like (for example, on a USB memory stick). Note that the file My Volume does not exist yet – CipherShed will create it.

**IMPORTANT:** Note that *CipherShed will not encrypt any existing files (when creating a CipherShed file container). If you select an existing file in this step, it will be overwritten and replaced by the newly created volume (so the overwritten file will be lost, not encrypted). You will be able to encrypt existing files (later on) by moving them to the CipherShed volume that we are creating now.*<sup>2</sup>

Select the desired path (where you wish the container to be created) in the file selector.

Type the desired container filename in the **File name** box.

Click **Save**.

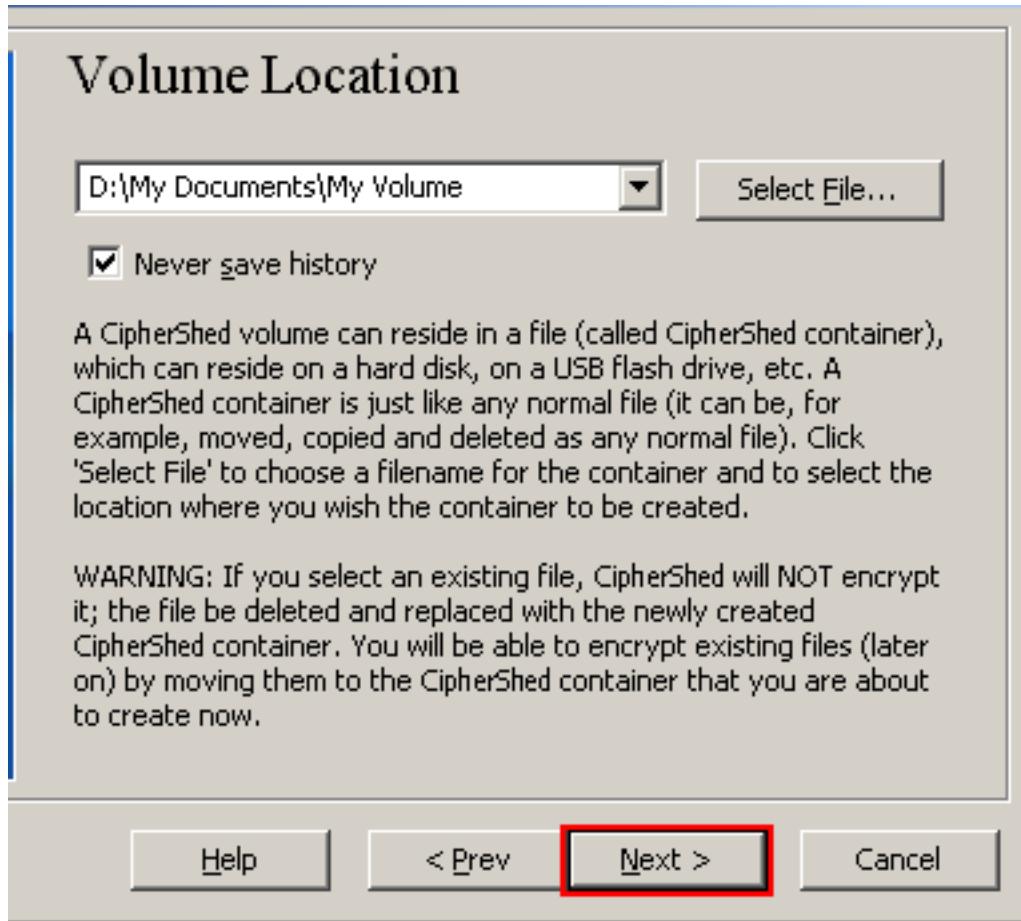
The file selector window should disappear.

In the following steps, we will return to the CipherShed Volume Creation Wizard.

---

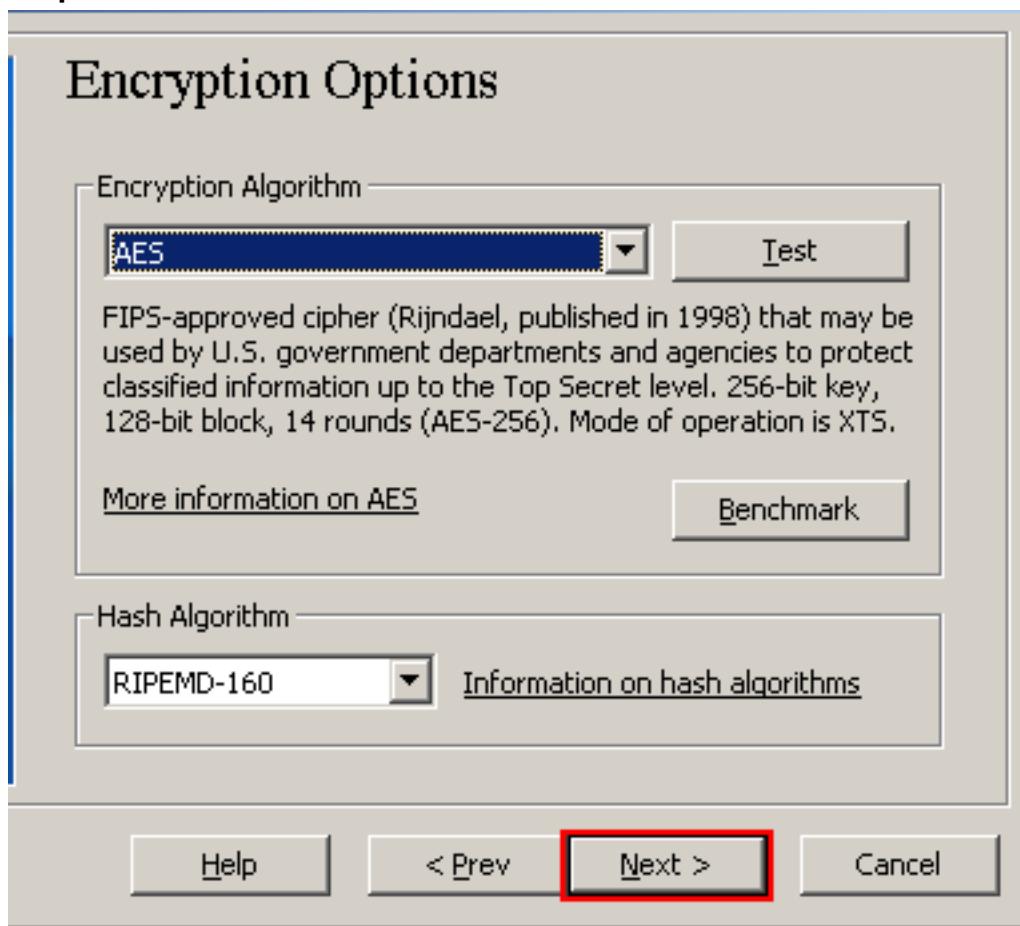
<sup>2</sup> Note that after you copy existing unencrypted files to a TrueCrypt volume, you should securely erase (wipe) the original unencrypted files. There are software tools that can be used for the purpose of secure erasure (many of them are free).

**Step 7:**



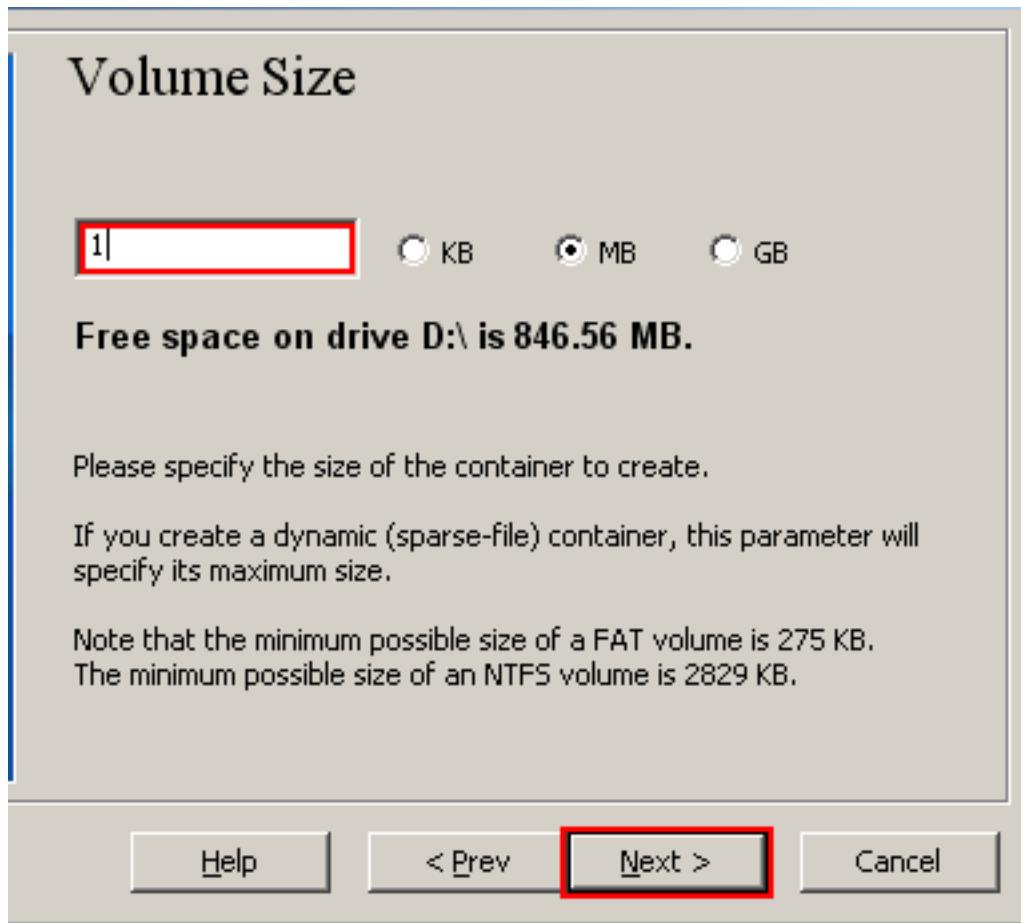
In the Volume Creation Wizard window, click **Next**.

## Step 8:



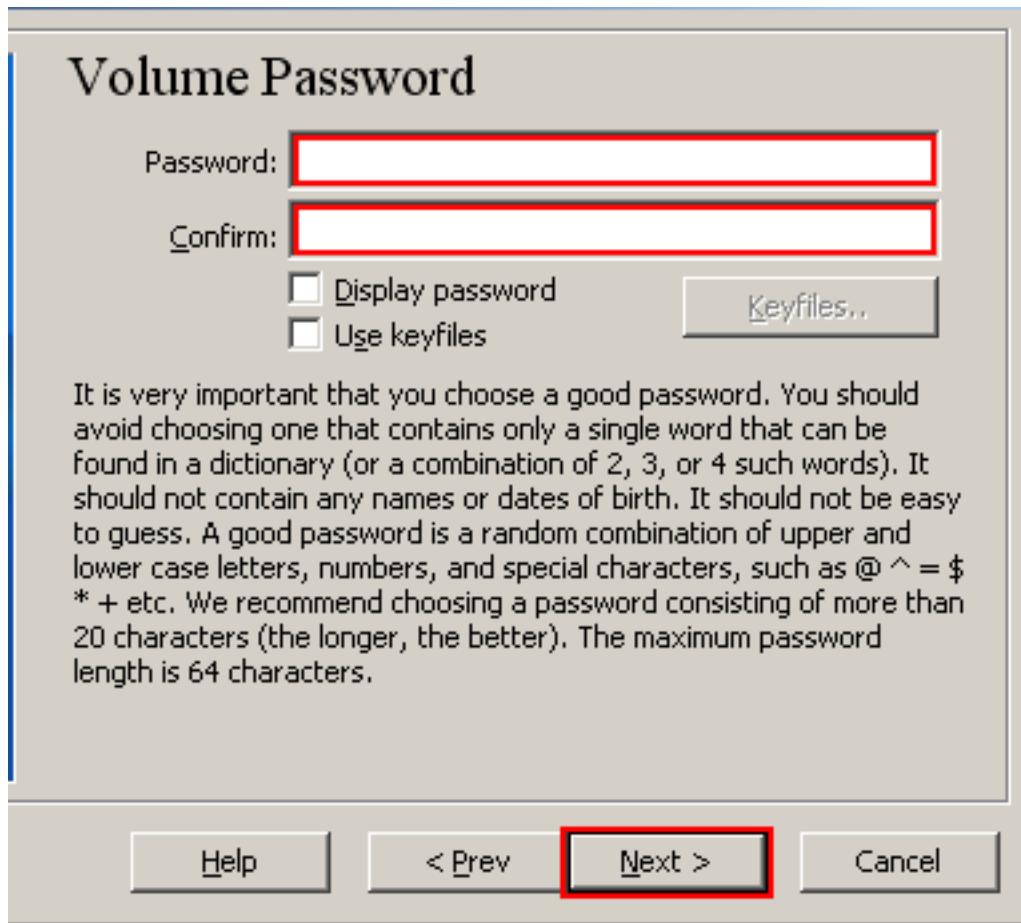
Here you can choose an encryption algorithm and a hash algorithm for the volume. If you are not sure what to select here, you can use the default settings and click **Next** (for more information, see chapters Encryption Algorithms and Hash Algorithms).

**Step 9:**



Here we specify that we wish the size of our CipherShed container to be 1 megabyte. You may, of course, specify a different size. After you type the desired size in the input field (marked with a red rectangle), click **Next**.

## Step 10:

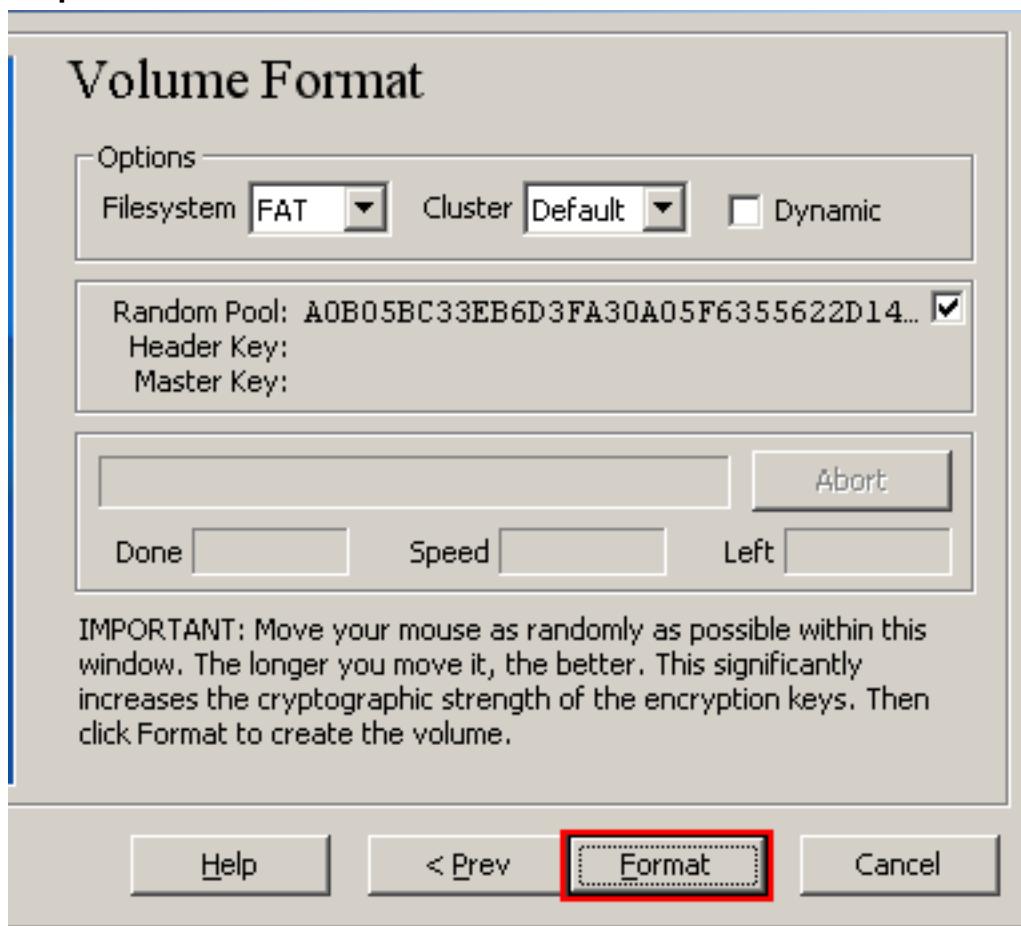


This is one of the most important steps. Here you have to choose a good volume password.

Read carefully the information displayed in the Wizard window about what is considered a good password. After you choose a good password, type it in the first input field. Then re-type it in the input field below the first one and click **Next**.

Note: The button **Next** will be disabled until passwords in both input fields are the same.

## Step 11:



Move your mouse as randomly as possible within the Volume Creation Wizard window at least for 30 seconds. The longer you move the mouse, the better. This significantly increases the cryptographic strength of the encryption keys (which increases security).

**Click Format.**

Volume creation should begin. CipherShed will now create a file called *My Volume* in the folder *D:\My Documents\* (as we specified in Step 6). This file will be a CipherShed container (it will contain the encrypted CipherShed volume). Depending on the size of the volume, the volume creation may take a long time. After it finishes, the following dialog box will appear:



Click **OK** to close the dialog box.

**Step 12:**



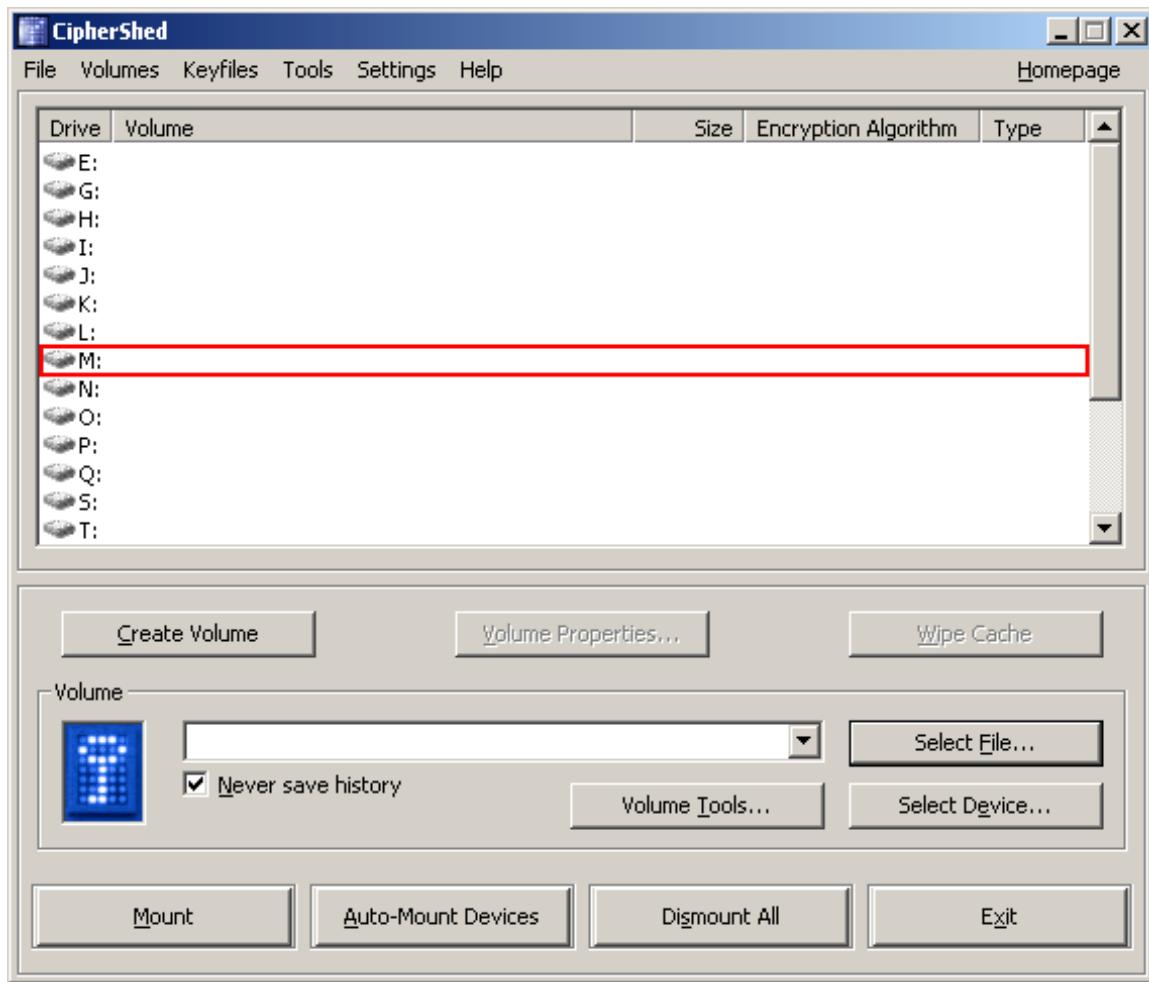
We have just successfully created a CipherShed volume (file container).

In the CipherShed Volume Creation Wizard window, click **Exit**.

The Wizard window should disappear.

In the remaining steps, we will mount the volume we just created. We will return to the main CipherShed window (which should still be open, but if it is not, repeat Step 1: to launch CipherShed and then continue from Step 13:.)

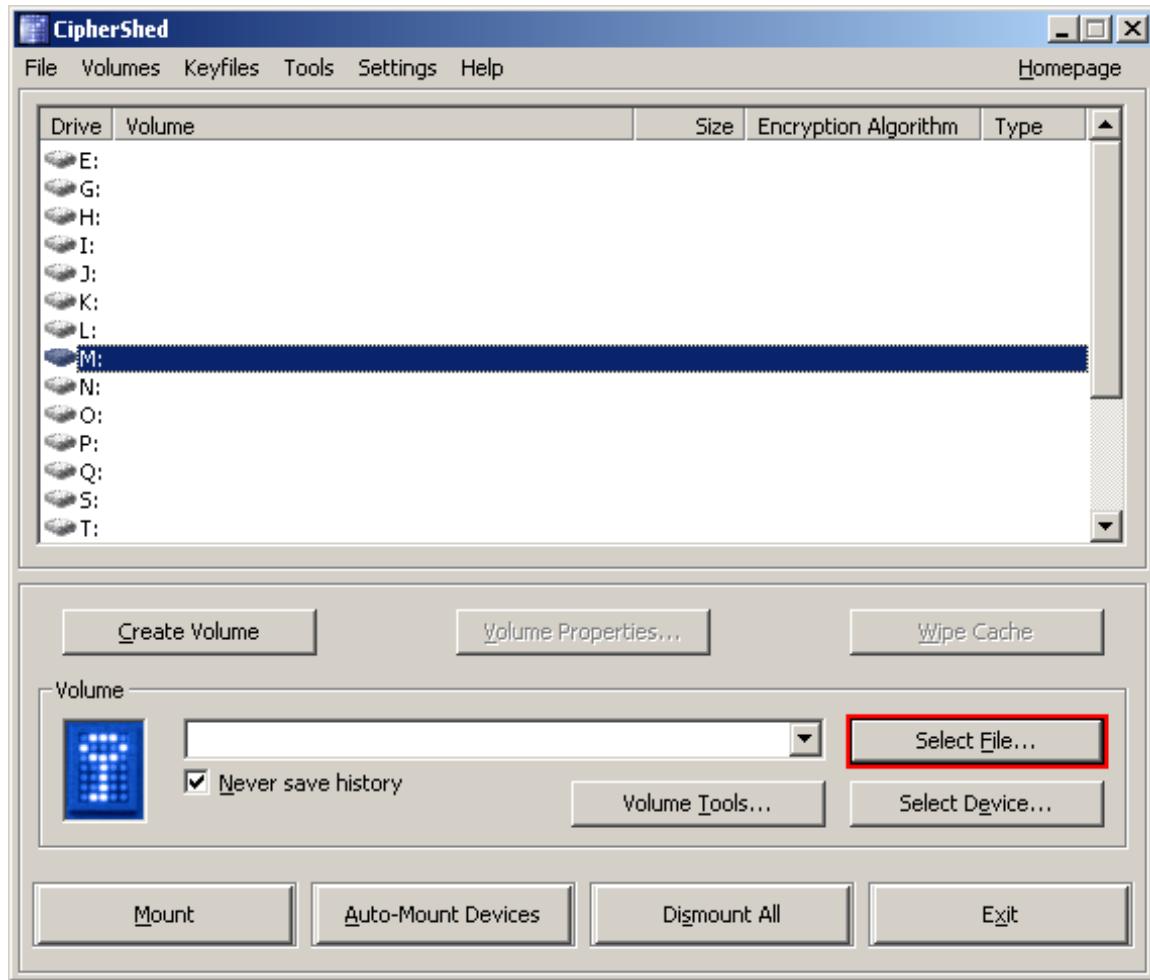
### Step 13:



Select a drive letter from the list (marked with a red rectangle). This will be the drive letter to which the CipherShed container will be mounted.

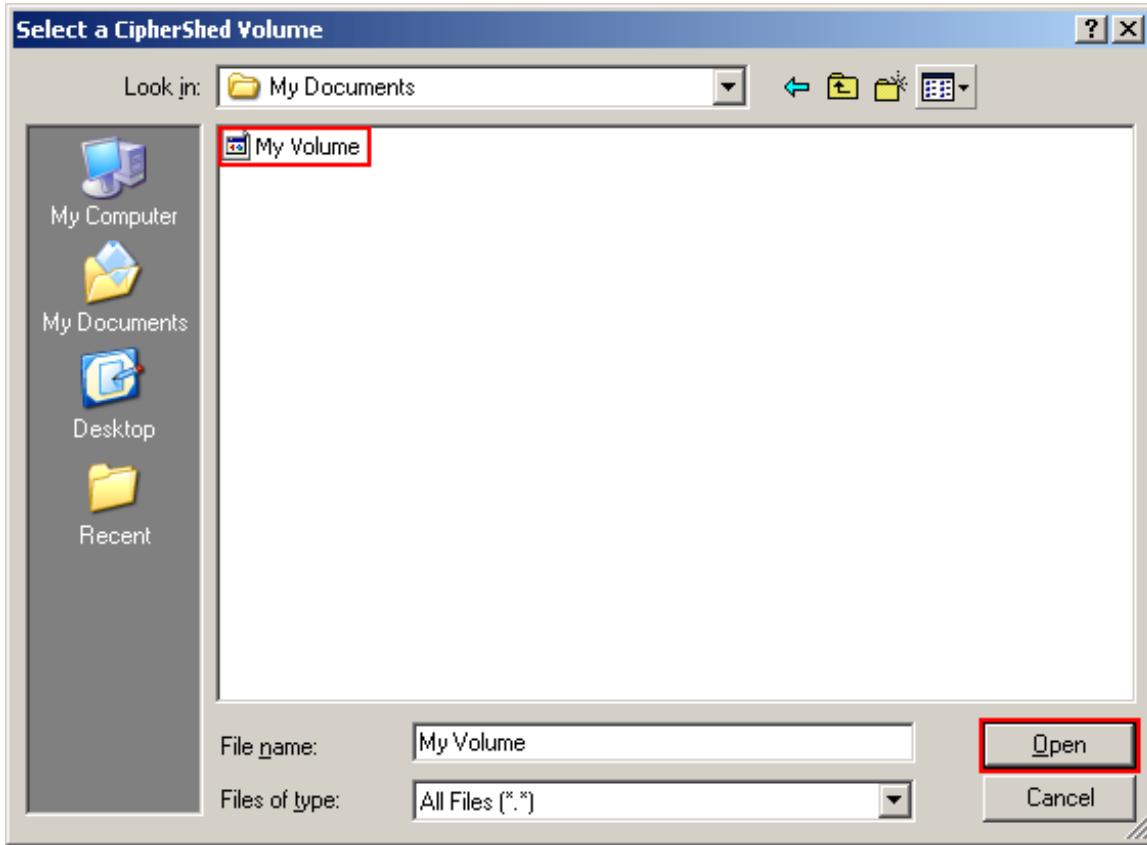
Note: In this tutorial, we chose the drive letter M, but you may of course choose any other available drive letter.

## Step 14:



Click **Select File**. The standard file selector window should appear.

## Step 15:



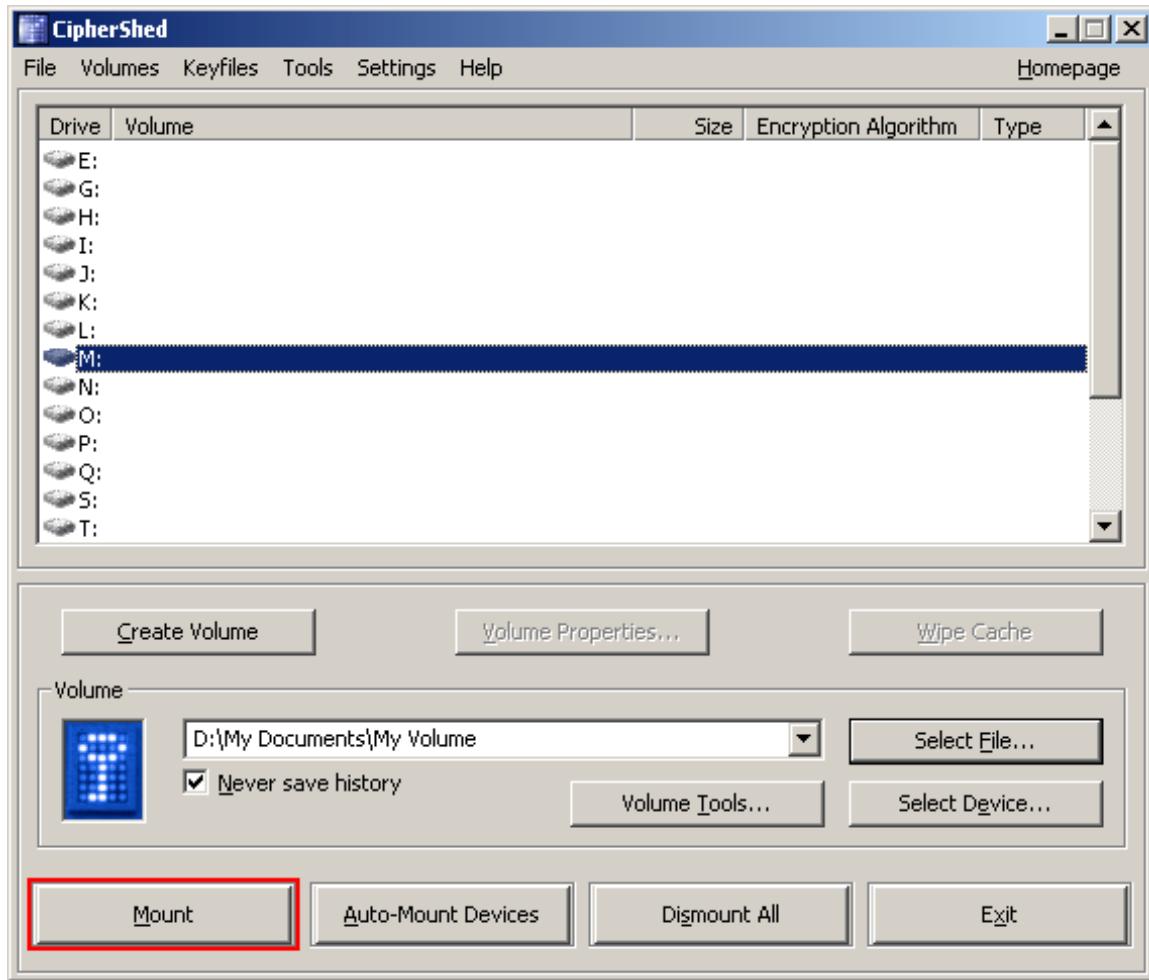
In the file selector, browse to the container file (which we created in Steps 6-11) and select it.

Click **Open** (in the file selector window).

The file selector window should disappear.

In the following steps, we will return to the main CipherShed window.

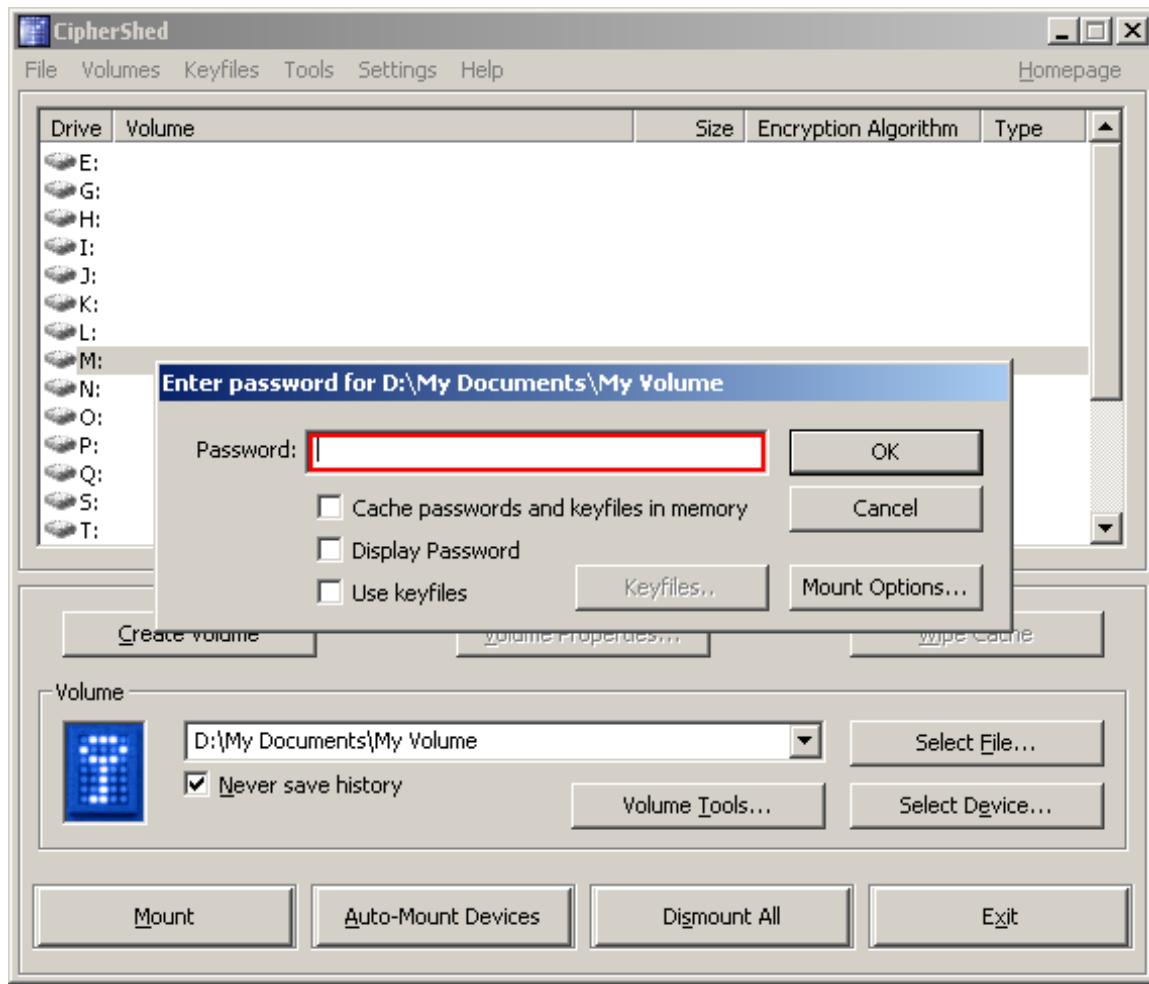
## Step 16:



In the main CipherShed window, click **Mount**.

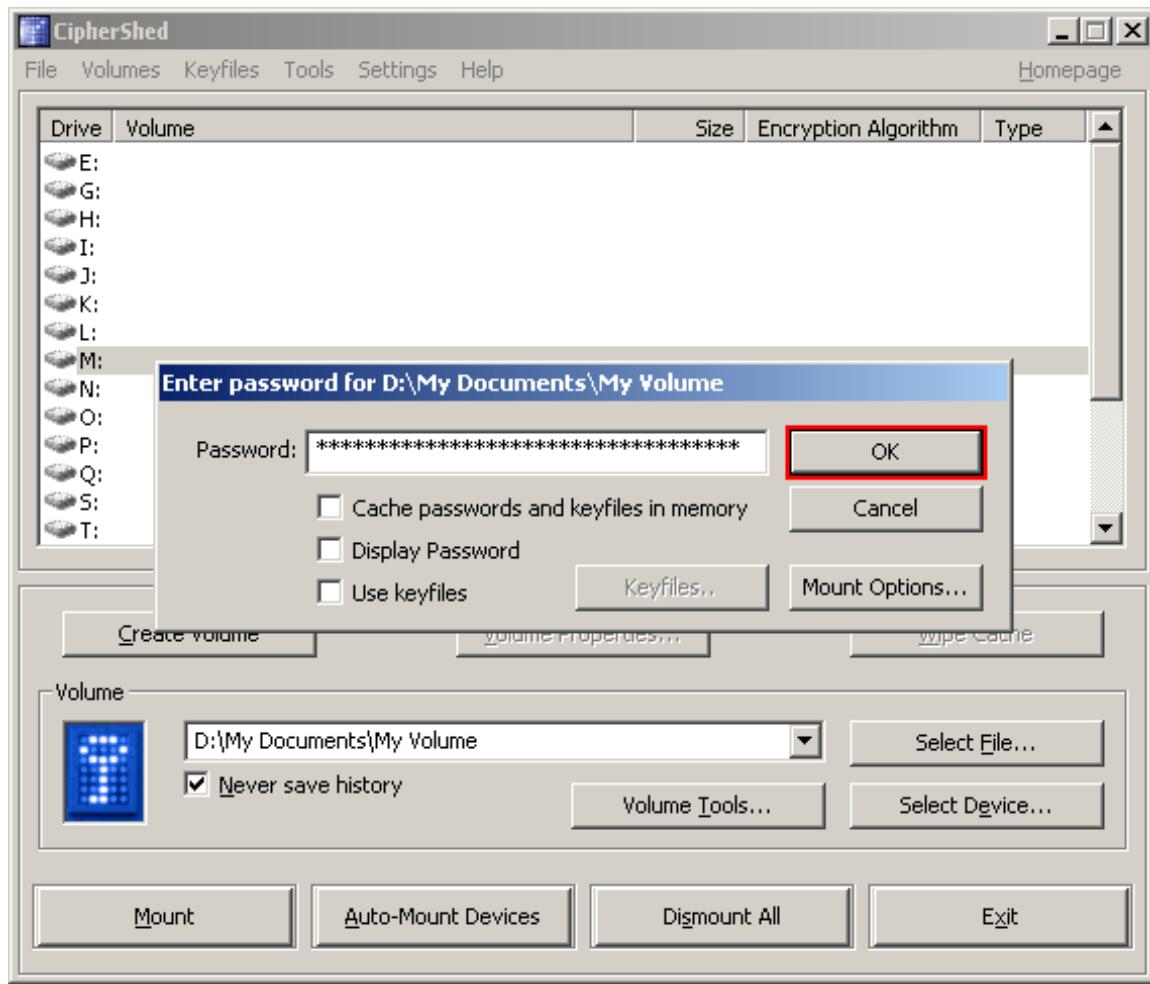
Password prompt dialog window should appear.

## Step 17:



Type the password (which you specified in Step 10:) in the password input field (marked with a red rectangle).

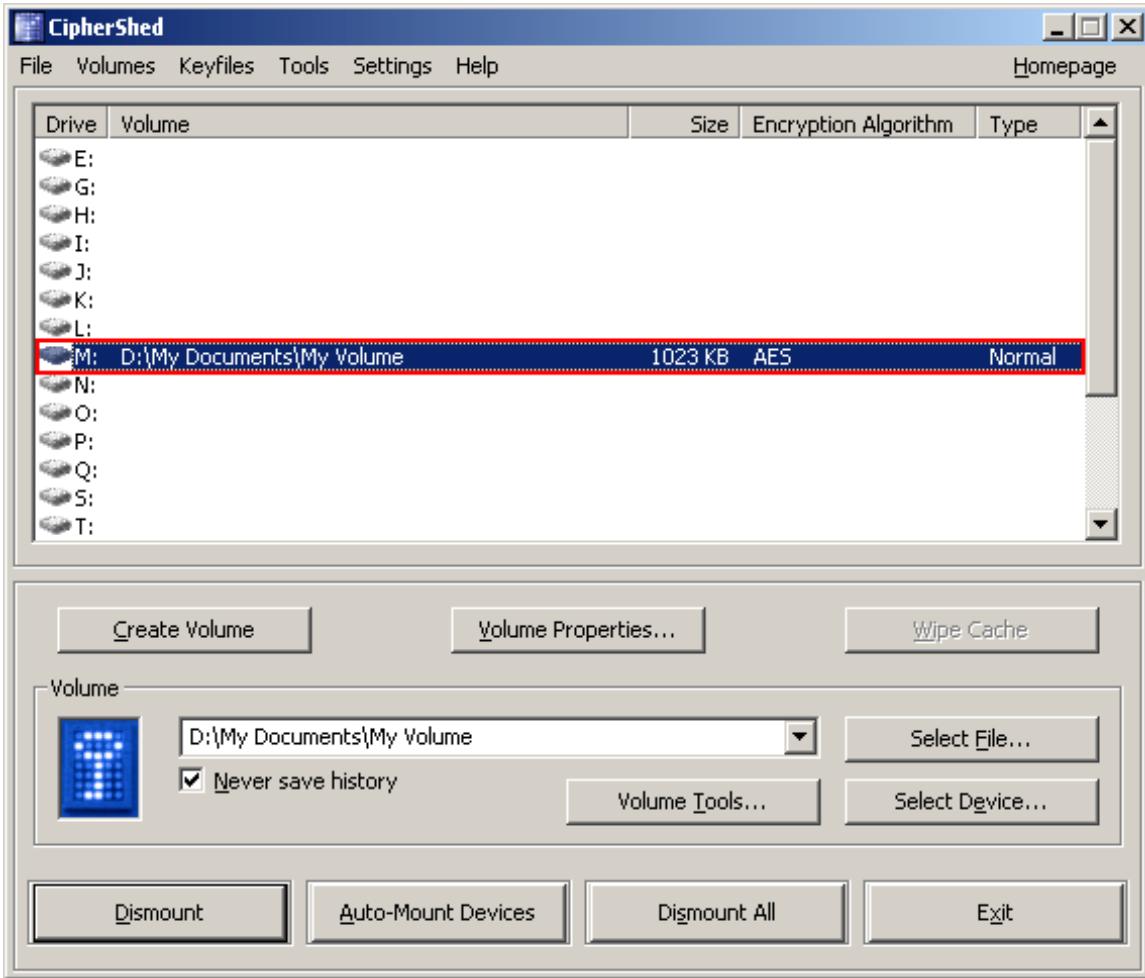
## Step 18:



Click **OK** in the password prompt window.

CipherShed will now attempt to mount the volume. If the password is incorrect (for example, if you typed it incorrectly), CipherShed will notify you and you will need to repeat the previous step (type the password again and click **OK**). If the password is correct, the volume will be mounted.

## Final Step:



We have just successfully mounted the container as a virtual disk M:

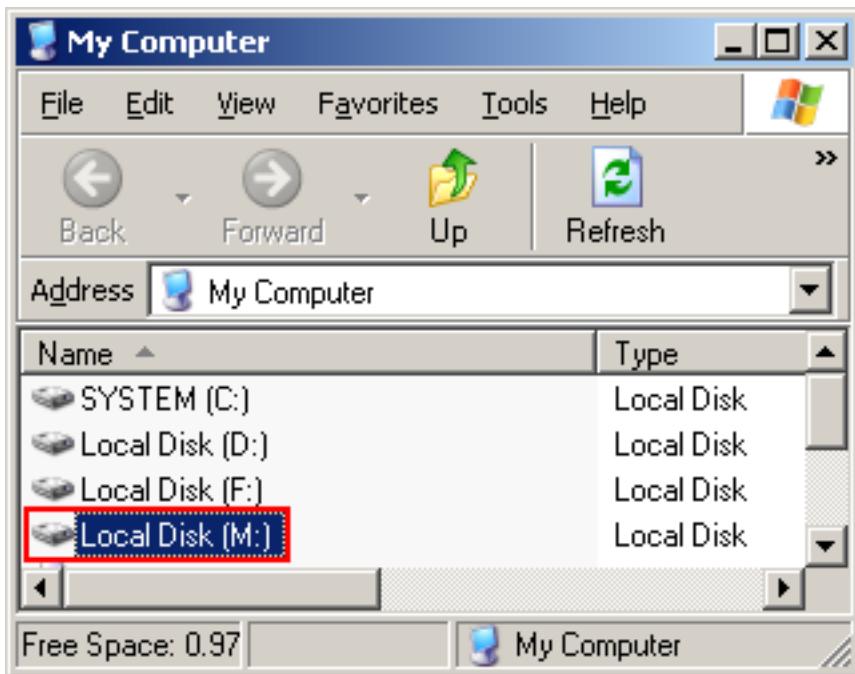
The virtual disk is entirely encrypted (including file names, allocation tables, free space, etc.) and behaves like a real disk. You can save (or copy, move, etc.) files to this virtual disk and they will be encrypted on the fly as they are being written.

If you open a file stored on a CipherShed volume, for example, in media player, the file will be automatically decrypted to RAM (memory) on the fly while it is being read.

***Important: Note that when you open a file stored on a CipherShed volume (or when you write/copy a file to/from the CipherShed volume) you will not be asked to enter the password again. You need to enter the correct password only when mounting the volume.***

You can open the mounted volume, for example, by double-clicking the item marked with a red rectangle in the screenshot above.

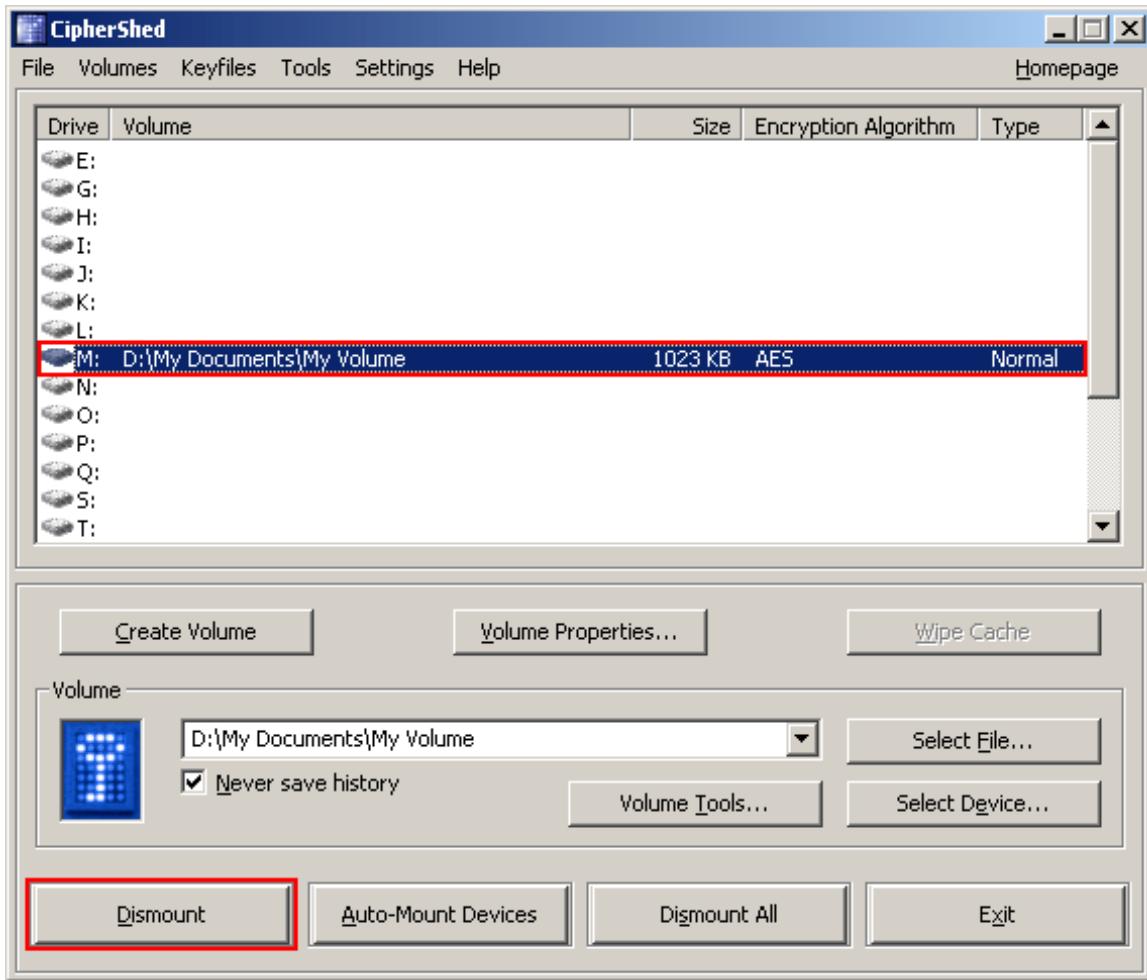
You can also browse to the mounted volume the way you normally browse to any other types of volumes. For example, by opening the 'Computer' (or 'My Computer') list and double clicking the corresponding drive letter (in this case, it is the letter M).



You can copy files (or folders) to and from the CipherShed volume just as you would copy them to any normal disk (for example, by simple drag-and-drop operations). Files that are being read or copied from the encrypted CipherShed volume are automatically decrypted on the fly in RAM (memory). Similarly, files that are being written or copied to the CipherShed volume are automatically encrypted on the fly in RAM (right before they are written to the disk).

Note that CipherShed never saves any decrypted data to a disk – it only stores them temporarily in RAM (memory). Even when the volume is mounted, data stored in the volume is still encrypted. When you restart Windows or turn off your computer, the volume will be dismounted and all files stored on it will be inaccessible (and encrypted). Even when power supply is suddenly interrupted (without proper system shut down), all files stored on the volume will be inaccessible (and encrypted). To make them accessible again, you have to mount the volume. To do so, repeat Step 13:-Step 18:-

If you want to close the volume and make files stored on it inaccessible, either restart your operating system or dismount the volume. To do so, follow these steps:



Select the volume from the list of mounted volumes in the main CipherShed window (marked with a red rectangle in the screenshot above) and then click **Dismount** (also marked with a red rectangle in the screenshot above). To make files stored on the volume accessible again, you will have to mount the volume. To do so, repeat Step 13:-Step 18:-

## How to Create and Use a CipherShed-Encrypted Partition/Device

Instead of creating file containers, you can also encrypt physical partitions or drives (i.e., create CipherShed device-hosted volumes). To do so, repeat the Step 1:-3 but in the Step 3: select the second or third option. Then follow the remaining instructions in the wizard. When you create a device-hosted CipherShed volume within a *non-system* partition/drive, you can mount it by clicking Auto-Mount Devices in the main CipherShed window. For information pertaining to encrypted system partition/drives, see the chapter System Encryption.

**Important:** We strongly recommend that you also read the other chapters of this manual, as they contain important information that has been omitted in this tutorial for simplicity.

## **CipherShed Volume**

There are two types of CipherShed volumes:

- File-hosted (container)
- Partition/device-hosted (non-system)

Note: In addition to creating the above types of virtual volumes, CipherShed can encrypt a physical partition/drive where Windows is installed (for more information, see the chapter System Encryption).

A CipherShed file-hosted volume is a normal file, which can reside on any type of storage device. It contains (hosts) a completely independent encrypted virtual disk device.

A CipherShed partition is a hard disk partition encrypted using CipherShed. You can also encrypt entire hard disks, USB hard disks, USB memory sticks, and other types of storage devices.

### ***Creating a New CipherShed Volume***

To create a new CipherShed file-hosted volume or to encrypt a partition/device (requires administrator privileges), click on ‘Create Volume’ in the main program window. CipherShed Volume Creation Wizard should appear. As soon as the Wizard appears, it starts collecting data that will be used in generating the master key, secondary key (XTS mode), and salt, for the new volume. The collected data, which should be as random as possible, include your mouse movements, key presses, and other values obtained from the system (for more information, please see the section Random Number Generator). The Wizard provides help and information necessary to successfully create a new CipherShed volume. However, several items deserve further explanation:

### **Hash Algorithm**

Allows you to select which hash algorithm CipherShed will use. The selected hash algorithm is used by the random number generator (as a pseudorandom mixing function), which generates the master key, secondary key (XTS mode), and salt (for more information, please see the section Random Number Generator). It is also used in deriving the new volume header key and secondary header key (see the section Header Key Derivation, Salt, and Iteration Count).

For information about the implemented hash algorithms, see the chapter Hash Algorithms.

Note that the output of a hash function is never used directly as an encryption key. For more information, please refer to the chapter Technical Details.

### **Encryption Algorithm**

This allows you to select the encryption algorithm with which your new volume will be encrypted. Note that the encryption algorithm cannot be changed after the volume is created. For more information, please see the chapter Encryption Algorithms.

### **Quick Format**

If unchecked, each sector of the new volume will be formatted. This means that the new volume will be entirely filled with random data. Quick format is much faster but may be less secure because until the whole volume has been filled with files, it may be possible to tell how much data it contains (if the space was not filled with random data beforehand). If you are not sure whether to enable or disable Quick Format, we recommend that you leave this option unchecked. Note that Quick Format can only be enabled when encrypting partitions/devices.

**Important:** When encrypting a partition/device within which you intend to create a hidden volume afterwards, leave this option unchecked.

## Dynamic

Dynamic CipherShed container is a pre-allocated NTFS sparse file whose physical size (actual disk space used) grows as new data is added to it. Note that the physical size of the container (actual disk space that the container uses) will not decrease when files are deleted on the CipherShed volume. The physical size of the container can only increase up to the maximum value that is specified by the user during the volume creation process. After the maximum specified size is reached, the physical size of the container will remain constant.

Note that sparse files can only be created in the NTFS file system. If you are creating a container in the FAT file system, the option Dynamic will be disabled (“grayed out”).

Note that the size of a dynamic (sparse-file-hosted) CipherShed volume reported by Windows and by CipherShed will always be equal to its maximum size (which you specify when creating the volume). To find out current physical size of the container (actual disk space it uses), right-click the container file (in a Windows Explorer window, not in CipherShed), then select Properties and see the Size on disk value.

***WARNING: Performance of dynamic (sparse-file-hosted) CipherShed volumes is significantly worse than performance of regular volumes. Dynamic (sparse-file-hosted) CipherShed volumes are also less secure, because it is possible to tell which volume sectors are unused. Furthermore, if data is written to a dynamic volume when there is not enough free space in its host file system, the encrypted file system may get corrupted.***

## Cluster Size

Cluster is an allocation unit. For example, one cluster is allocated on a FAT file system for a one-byte file. When the file grows beyond the cluster boundary, another cluster is allocated. Theoretically, this means that the bigger the cluster size, the more disk space is wasted; however, the better the performance. If you do not know which value to use, use the default.

## CipherShed Volumes on CDs and DVDs

If you want a CipherShed volume to be stored on a CD or a DVD, first create a file-hosted CipherShed container on a hard drive and then burn it onto a CD/DVD using any CD/DVD burning software (or, under Windows XP or later, using the CD burning tool provided with the operating system). Remember that if you need to mount a CipherShed volume that is stored on a read-only medium (such as a CD/DVD) under Windows 2000, you must format the CipherShed volume as FAT. The reason is that Windows 2000 cannot mount NTFS file system on read-only media (Windows XP and later versions of Windows can).

## Hardware/Software RAID, Windows Dynamic Volumes

CipherShed supports hardware/software RAID as well as Windows dynamic volumes.

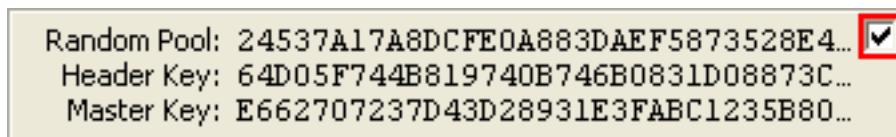
Windows Vista or later: Dynamic volumes are displayed in the ‘Select Device’ dialog window as `\Device\HarddiskVolumeN`.

Windows XP/2000/2003: If you intend to format a Windows dynamic volume as a CipherShed volume, keep in mind that after you create the Windows dynamic volume (using the Windows Disk Management tool), you must restart the operating system in order for the volume to be available/displayed in the ‘Select Device’ dialog window of the CipherShed Volume Creation Wizard. Also note that, in the ‘Select Device’ dialog window, a Windows dynamic volume is not displayed as a single device (item). Instead, all volumes that the Windows dynamic volume consists of are displayed and you can select any of them in order to format the entire Windows dynamic volume.

## **Additional Notes on Volume Creation**

After you click the ‘Format’ button in the Volume Creation Wizard window (the last step), there will be a short delay while your system is being polled for additional random data. Afterwards, the master key, header key, secondary key (XTS mode), and salt, for the new volume will be generated, and the master key and header key contents will be displayed.

For extra security, the portions of the randomness pool, master key, and header key can be prevented from being displayed by unchecking the checkbox in the upper right corner of the corresponding field:



Note that only the first 128 bits of the pool/keys are displayed (not the entire contents).

You can create FAT (whether it will be FAT12, FAT16, or FAT32, is automatically determined from the number of clusters) or NTFS volumes (however, NTFS volumes can only be created by users with administrator privileges). Mounted CipherShed volumes can be reformatted as FAT12, FAT16, FAT32, or NTFS anytime. They behave as standard disk devices so you can right-click the drive letter of the mounted CipherShed volume (for example in the ‘Computer’ or ‘My Computer’ list) and select ‘Format’.

For more information about creating CipherShed volumes, see also the section [Hidden Volume](#).

## **Favorite Volumes**

Favorite volumes are useful, for example, in any the following cases:

- You have a volume that always needs to be mounted to a particular drive letter.
- You have a volume that needs to be automatically mounted when its host device gets connected to the computer (for example, a container located on a USB flash drive or external USB hard drive).
- You have a volume that needs to be automatically mounted when you log on to the operating system.
- You have a volume that always needs to be mounted as read-only or removable medium.

To configure a CipherShed volume as a favorite volume, follow these steps:

1. Mount the volume (to the drive letter to which you want it to be mounted every time).
2. Right-click the mounted volume in the drive list in the main CipherShed window and select ‘Add to Favorites’.
3. The Favorite Volumes Organizer window should appear now. In this window, you can set various options for the volume (see below).
4. Click OK.

**Favorite volumes can be mounted in several ways:** To mount all favorite volumes, select Favorites > Mount Favorite Volumes or press the ‘Mount Favorite Volumes’ hot key (Settings > Hot Keys). To mount only one of the favorite volumes, select it from the list contained in the Favorites menu. When you do so, you are asked for its password (and/or keyfiles) (unless it is cached) and if it is correct, the volume is mounted. If it is already mounted, an Explorer window is opened for it.

**Selected or all favorite volumes can be mounted automatically whenever you log on to Windows.** To set this up, follow these steps:

1. Mount the volume you want to have mounted automatically when you log on (mount it to the drive letter to which you want it to be mounted every time).
2. Right-click the mounted volume in the drive list in the main CipherShed window and select ‘Add to Favorites’.
3. The Favorites Organizer window should appear now. In this window, enable the option ‘Mount selected volume upon logon’ and click **OK**.

Then, when you log on to Windows, you will be asked for the volume password (and/or keyfiles) and if it is correct, the volume will be mounted.

Note: CipherShed will not prompt you for a password if you have enabled caching of the pre-boot authentication password (Settings > ‘System Encryption’) and the volumes use the same password as the system partition/drive.

**Selected or all favorite volumes can be mounted automatically whenever its host device gets connected to the computer.** To set this up, follow these steps:

1. Mount the volume (to the drive letter to which you want it to be mounted every time).
2. Right-click the mounted volume in the drive list in the main CipherShed window and select ‘Add to Favorites’.
3. The Favorites Organizer window should appear now. In this window, enable the option ‘Mount selected volume when its host device gets connected’ and click **OK**.

Then, when you insert e.g. a USB flash drive on which a CipherShed volume is located into the USB port, you will be asked for the volume password (and/or keyfiles) (unless it is cached) and if it is correct, the volume will be mounted.

Note: CipherShed will not prompt you for a password if you have enabled caching of the pre-boot authentication password (Settings > ‘System Encryption’) and the volume uses the same password as the system partition/drive.

**A special label can be assigned to each favorite volume.** This label is not the same as the filesystem label and it is shown within the CipherShed user interface instead of the volume path. To assign such a label, follow these steps:

1. Select Favorites > ‘Organize Favorite Volumes’.
2. The Favorite Volumes Organizer window should appear now. In this window, select the volume whose label you want to edit.
3. Enter the label in the ‘Label of selected favorite volume’ input field and click **OK**.

Note that the Favorite Volumes Organizer window (Favorites > ‘Organize Favorite Volumes’) allows you to set various other options for each favorite volume. For example, any of them can be mounted as read-only or as removable medium. To set any of these options, follow these steps:

- 1 Select Favorites > ‘Organize Favorite Volumes’.
- 2 The Favorite Volumes Organizer window should appear now. In this window, select the volume whose options you want to set.
- 3 Set the options and click OK.

The order in which system favorite volumes are displayed in the Favorites Organizer window (Favorites > ‘Organize Favorite Volumes’) is the order in which the volumes are mounted when you select Favorites > Mount Favorite Volumes or when you press the ‘Mount Favorite Volumes’ hotkey (Settings > Hot Keys). You can use the Move Up and Move Down buttons to change the order of the volumes.

Note that a favorite volume can also be a partition that is within the key scope of system encryption mounted without pre-boot authentication (for example, a partition located on the encrypted system drive of another operating system that is not running). When you mount such a volume and add it to favorites, you will no longer have to select System > Mount Without Pre-Boot Authentication or to enable the mount option ‘Mount partition using system encryption without pre-boot authentication’. You can simply mount the favorite volume (as explained above) without setting any options, as the mode in which the volume is mounted is saved in the configuration file containing the list of your favorite volumes.

Warning: When the drive letter assigned to a favorite volume (saved in the configuration file) is not free, the volume is not mounted and no error message is displayed.

To remove a volume from the list of favorite volumes, select Favorites > Organize Favorite Volumes, select the volume, click Remove, and click OK.

## **System Favorite Volumes**

System favorites are useful, for example, in the following cases:

- You have volumes that need to be mounted before system and application services start and before users start logging on.
- There are network-shared folders located on CipherShed volumes. If you configure these volumes as system favorites, you will ensure that the network shares will be automatically restored by the operating system each time it is restarted.
- You need each such volume to be mounted as the same drive letter each time the operating system starts.

Note that, unlike the regular (non-system) favorites, system favorite volumes use the pre-boot authentication password and, therefore, require your system partition/drive to be encrypted (also note it is not required to enable caching of the pre-boot authentication password).

System favorite volumes can be configured to be available within CipherShed only to users with administrator privileges (select Settings > ‘System Favorite Volumes’ > ‘Allow only administrators to view and dismount system favorite volumes in CipherShed’). This option should be enabled on servers to ensure that system favorite volumes cannot be dismounted by users without administrator privileges. On non-server systems, this option can be used to prevent normal CipherShed volume actions (such as ‘Dismount All’, auto-dismount, etc.) from affecting system favorite volumes. In addition, when CipherShed is run without administrator privileges (the default on Windows Vista and later), system favorite volumes will not be displayed in the drive letter list in the main CipherShed application window.

To configure a CipherShed volume as a system favorite volume, follow these steps:

- 1 Mount the volume (to the drive letter to which you want it to be mounted every time).
- 2 Right-click the mounted volume in the drive list in the main CipherShed window and select ‘Add to System Favorites’.
- 3 The System Favorites Organizer window should appear now. In this window, enable the option ‘Mount system favorite volumes when Windows starts’ and click OK.

The order in which system favorite volumes are displayed in the System Favorites Organizer window (Favorites > ‘Organize System Favorite Volumes’) is the order in which the volumes are mounted. You can use the Move Up and Move Down buttons to change the order of the volumes.

A special label can be assigned to each system favorite volume. This label is not the same as the filesystem label and it is shown within the CipherShed user interface instead of the volume path. To assign such a label, follow these steps:

- 1 Select Favorites > ‘Organize System Favorite Volumes’.
- 2 The System Favorites Organizer window should appear now. In this window, select the volume whose label you want to edit.
- 3 Enter the label in the ‘Label of selected favorite volume’ input field and click OK.

Note that the System Favorites Organizer window (Favorites > ‘Organize System Favorite Volumes’) allows you to set various options for each system favorite volume. For example, any of them can be mounted as read-only or as removable medium.

Warning: When the drive letter assigned to a system favorite volume (saved in the configuration file) is not free, the volume is not mounted and no error message is displayed.

Note that Windows needs to use some files (e.g. paging files, Active Directory files, etc.) before system favorite volumes are mounted. Therefore, such files cannot be stored on system favorite volumes. Note, however, that they can be stored on any partition that is within the key scope of system encryption (e.g. on the system partition or on any partition of a system drive that is entirely encrypted by CipherShed).

To remove a volume from the list of system favorite volumes, select Favorites > Organize System Favorite Volumes, select the volume, click Remove, and click OK.

## **System Encryption**

CipherShed can on-the-fly encrypt a system partition or entire system drive, i.e. a partition or drive where Windows is installed and from which it boots.

System encryption provides the highest level of security and privacy, because all files, including any temporary files that Windows and applications create on the system partition (typically, without your knowledge or consent), hibernation files, swap files, etc., are always permanently encrypted (even when power supply is suddenly interrupted). Windows also records large amounts of potentially sensitive data, such as the names and locations of files you open, applications you run, etc. All such log files and registry entries are always permanently encrypted as well.

System encryption involves pre-boot authentication, which means that anyone who wants to gain access and use the encrypted system, read and write files stored on the system drive, etc., will need to enter the correct password each time before Windows boots (starts). Pre-boot authentication is handled by the CipherShed Boot Loader, which resides in the first track of the boot drive and on the CipherShed Rescue Disk (see below).

Note that CipherShed can encrypt an existing unencrypted system partition/drive in-place while the operating system is running (while the system is being encrypted, you can use your computer as usual without any restrictions). Likewise, a CipherShed-encrypted system partition/drive can be decrypted in-place while the operating system is running. You can interrupt the process of encryption or decryption anytime, leave the partition/drive partially unencrypted, restart or shut down the computer, and then resume the process, which will continue from the point it was stopped.

The mode of operation used for system encryption is XTS (see the section Modes of Operation). For further technical details of system encryption, see the section Encryption Scheme in the chapter Technical Details.

To encrypt a system partition or entire system drive, select System > Encrypt System Partition/Drive and then follow the instructions in the wizard. To decrypt a system partition/drive, select System > Permanently Decrypt System Partition/Drive.

Note: By default, Windows 7 and later boot from a special small partition. The partition contains files that are required to boot the system. Windows allows only applications that have administrator privileges to write to the partition (when the system is running). CipherShed encrypts the partition only if you choose to encrypt the whole system drive (as opposed to choosing to encrypt only the partition where Windows is installed).

## **Hidden Operating System**

It may happen that you are forced by somebody to decrypt the operating system. There are many situations where you cannot refuse to do so (for example, due to extortion). CipherShed allows you to create a hidden operating system whose existence should be impossible to prove (provided that certain guidelines are followed). Thus, you will not have to decrypt or reveal the password for the hidden operating system. For more information, see the section Hidden Operating System in the chapter Plausible Deniability.

## **Operating Systems Supported for System Encryption**

Note: After this version of CipherShed was released, a new version of an operating system may have been released and verified to be fully compatible with CipherShed. Therefore, if this is the latest stable version of CipherShed, you should check the online version of this chapter at:

<https://ciphershed.org>

CipherShed can currently encrypt the following operating systems:

- Windows 7 (32-bit and 64-bit)
- Windows Vista (SP1 or later)
- Windows Vista x64 (64-bit) Edition (SP1 or later)
- Windows XP
- Windows XP x64 (64-bit) Edition
- Windows Server 2008 R2 (64-bit)
- Windows Server 2008
- Windows Server 2008 x64 (64-bit)
- Windows Server 2003
- Windows Server 2003 x64 (64-bit)

Note: The following operating systems (among others) are not supported: Windows 2003 IA-64, Windows 2008 IA-64, Windows XP IA-64, and the Embedded/Tablet versions of Windows.

See also the section Supported Operating Systems.

## **CipherShed Rescue Disk**

During the process of preparing the encryption of a system partition/drive, CipherShed requires that you create a so-called CipherShed Rescue Disk (CD/DVD), which serves the following purposes:

- If the CipherShed Boot Loader screen does not appear after you start your computer (or if Windows does not boot), the CipherShed Boot Loader may be damaged. The CipherShed Rescue Disk allows you restore it and thus to regain access to your encrypted system and data (however, note that you will still have to enter the correct password then). In the Rescue Disk screen, select Repair Options > Restore CipherShed Boot Loader. Then press 'Y' to confirm the action, remove the Rescue Disk from your CD/DVD drive and restart your computer.
- If the CipherShed Boot Loader is frequently damaged (for example, by inappropriately designed activation software) or if you do not want the CipherShed boot loader to reside on the hard drive (for example, if you want to use an alternative boot loader/manager for other operating systems), you can boot directly from the CipherShed Rescue Disk (as it contains the CipherShed boot loader too) without restoring the boot loader to the hard drive. Just insert your Rescue Disk into your CD/DVD drive and then enter your password in the Rescue Disk screen.

- If you repeatedly enter the correct password but CipherShed says that the password is incorrect, it is possible that the master key or other critical data are damaged. The CipherShed Rescue Disk allows you to restore them and thus to regain access to your encrypted system and data (however, note that you will still have to enter the correct password then). In the Rescue Disk screen, select Repair Options > Restore key data. Then enter your password, press ‘Y’ to confirm the action, remove the Rescue Disk from your CD/DVD drive, and restart your computer.

Note: This feature cannot be used to restore the header of a hidden volume within which a hidden operating system resides (see the section Hidden Operating System). To restore such a volume header, click Select Device, select the partition behind the decoy system partition, click OK, select Tools -> Restore Volume Header and then follow the instructions.

***WARNING: By restoring key data using a CipherShed Rescue Disk, you also restore the password that was valid when the CipherShed Rescue Disk was created. Therefore, whenever you change the password, you should destroy your CipherShed Rescue Disk and create a new one (select System -> Create Rescue Disk). Otherwise, if an attacker knows your old password (for example, captured by a keystroke logger) and if he then finds your old CipherShed Rescue Disk, he could use it to restore the key data (the master key encrypted with the old password) and thus decrypt your system partition/drive.***

- If Windows is damaged and cannot start, the CipherShed Rescue Disk allows you to permanently decrypt the partition/drive before Windows starts. In the Rescue Disk screen, select Repair Options > Permanently decrypt system partition/drive. Enter the correct password and wait until decryption is complete. Then you can e.g. boot your MS Windows setup CD/DVD to repair your Windows installation. Note that this feature cannot be used to decrypt a hidden volume within which a hidden operating system resides (see the section Hidden Operating System).

Note: Alternatively, if Windows is damaged (cannot start) and you need to repair it (or access files on it), you can avoid decrypting the system partition/drive by following these steps: If you have multiple operating systems installed on your computer, boot the one that does not require pre-boot authentication. If you do not have multiple operating systems installed on your computer, you can boot a WinPE or BartPE CD/DVD or you can connect your system drive as a secondary or external drive to another computer and then boot the operating system installed on the computer. After you boot a system, run CipherShed, click Select Device, select the affected system partition, click OK , select System > Mount Without Pre-Boot Authentication, enter your pre-boot-authentication password and click OK. The partition will be mounted as a regular CipherShed volume (data will be on-the-fly decrypted/encrypted in RAM on access, as usual).

- Your CipherShed Rescue Disk contains a backup of the original content of the first drive track (made before the CipherShed Boot Loader was written to it) and allows you to restore it if necessary. The first track typically contains a system loader or boot manager. In the Rescue Disk screen, select Repair Options > Restore original system loader.

Note that even if you lose your CipherShed Rescue Disk and an attacker finds it, he or she will not be able to decrypt the system partition or drive without the correct password.

To boot a CipherShed Rescue Disk, insert it into your CD/DVD drive and restart your computer. If the CipherShed Rescue Disk screen does not appear (or if you do not see the ‘Repair Options’ item in the ‘Keyboard Controls’ section of the screen), it is possible that your BIOS is configured to attempt to boot from hard drives before CD/DVD drives. If that is the case, restart your computer, press F2 or Delete (as soon as you see a BIOS start-up screen), and wait until a BIOS configuration screen appears. If no BIOS configuration screen appears, restart (reset) the computer again and start pressing F2 or Delete repeatedly as soon as you restart (reset) the computer. When a BIOS configuration screen appears, configure your BIOS to boot from the CD/DVD drive first (for information on how to do so, please refer to the documentation for your BIOS/motherboard or contact your computer vendor’s technical support team for assistance). Then restart your computer. The CipherShed Rescue Disk screen should appear now. Note: In the CipherShed Rescue Disk screen, you can select ‘Repair Options’ by pressing F8 on your keyboard.

If your Rescue Disk is damaged, you can create a new one by selecting System > Create Rescue Disk. To find out whether your CipherShed Rescue Disk is damaged, insert it into your CD/DVD drive and select System > Verify Rescue Disk.

## **Plausible Deniability**

In case an adversary forces you to reveal your password, CipherShed provides and supports two kinds of plausible deniability:

- 1 Hidden volumes (for more information, see the section [Hidden Volume](#) below) and hidden operating systems (see the section [Hidden Operating System](#)).
- 2 Until decrypted, a CipherShed partition/device appears to consist of nothing more than random data (it does not contain any kind of "signature"). Therefore, it should be impossible to prove that a partition or a device is a CipherShed volume or that it has been encrypted (provided that the security requirements and precautions listed in the chapter [Security Requirements and Precautions](#) are followed). A possible plausible explanation for the existence of a partition/device containing solely random data is that you have wiped (securely erased) the content of the partition/device using one of the tools that erase data by overwriting it with random data (in fact, CipherShed can be used to securely erase a partition/device too, by creating an empty encrypted partition/device-hosted volume within it). However, you need to prevent data leaks (see section [Data Leaks](#)) and also note that, for system encryption, the first drive track contains the (unencrypted) CipherShed Boot Loader, which can be easily identified as such (for more information, see the chapter [System Encryption](#)). When using system encryption, plausible deniability can be achieved by creating a hidden operating system (see the section [Hidden Operating System](#)).

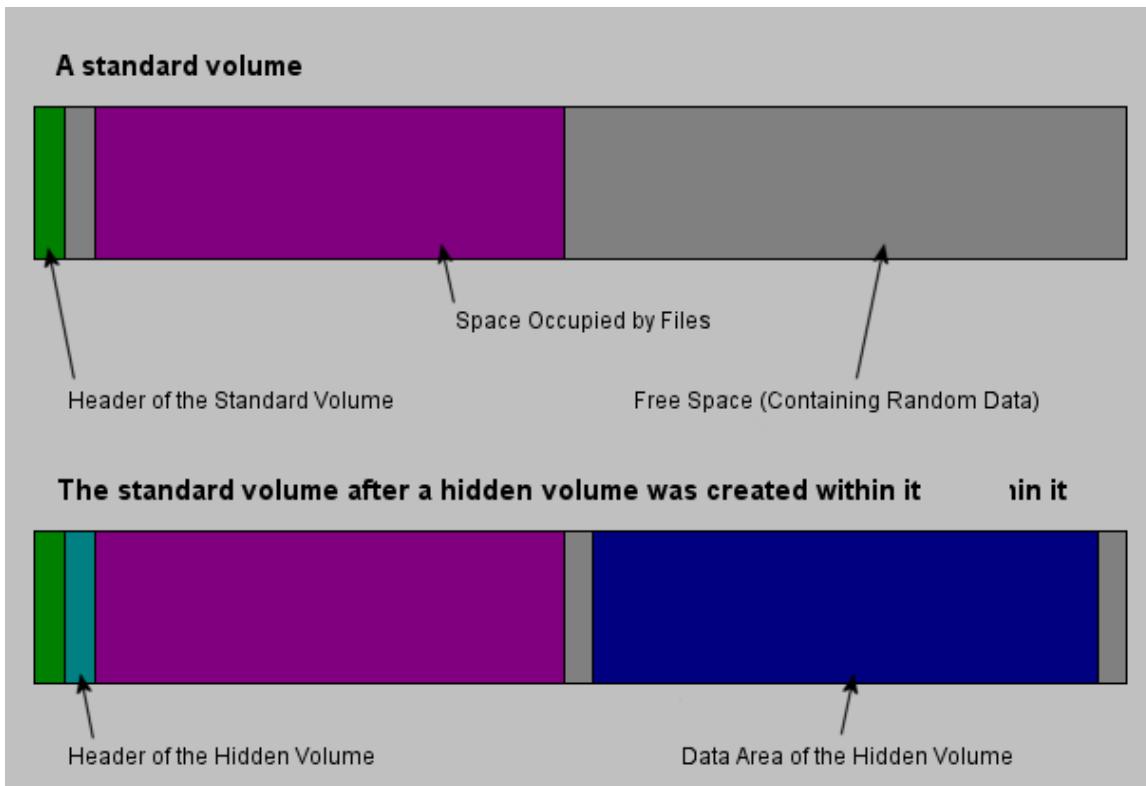
Although file-hosted CipherShed volumes (containers) do not contain any kind of "signature" either (until decrypted, they appear to consist solely of random data), they cannot provide this kind of plausible deniability, because there is practically no plausible explanation for the existence of a file containing solely random data. However, plausible deniability can still be achieved with a file-hosted CipherShed volume (container) by creating a hidden volume within it (see above).

### **Notes**

- When formatting a hard disk partition as a CipherShed volume (or encrypting a partition in place), the partition table (including the partition type) is never modified (no CipherShed "signature" or "ID" is written to the partition table).
- There are methods to find files or devices containing random data (such as CipherShed volumes). Note, however, that this should not affect plausible deniability in any way. The adversary still should not be able to prove that the partition/device is a CipherShed volume or that the file, partition, or device, contains a hidden CipherShed volume (provided that you follow the security requirements and precautions listed in the chapter [Security Requirements and Precautions](#) and in the subsection [Security Requirements and Precautions Pertaining to Hidden Volumes](#)).

## **Hidden Volume**

It may happen that you are forced by somebody to reveal the password to an encrypted volume. There are many situations where you cannot refuse to reveal the password (for example, due to extortion). Using a so-called hidden volume allows you to solve such situations without revealing the password to your volume.



The layout of a standard CipherShed volume before and after a hidden volume was created within it.

The principle is that a CipherShed volume is created within another CipherShed volume (within the free space on the volume). Even when the outer volume is mounted, it should be impossible to prove whether there is a hidden volume within it or not<sup>3</sup>, because free space on any CipherShed volume is always filled with random data when the volume is created<sup>4</sup> and no part of the (dismounted) hidden volume can be distinguished from random data. Note that CipherShed does not modify the file system (information about free space, etc.) within the outer volume in any way.

The password for the hidden volume must be substantially different from the password for the outer volume. To the outer volume, (before creating the hidden volume within it) you should copy some sensitive-looking files that you actually do NOT want to hide. These files will be there for anyone who would force you to hand over the password. You will reveal only the password for the outer volume, not for the hidden one. Files that really are sensitive will be stored on the hidden volume.

---

<sup>3</sup> Provided that all the instructions in the TrueCrypt Volume Creation Wizard have been followed and provided that the requirements and precautions listed in the subsection 'Security Requirements and Precautions Pertaining to Hidden Volumes' are followed.

<sup>4</sup> Provided that the options Quick Format and Dynamic are disabled and provided that the volume does not contain a filesystem that has been encrypted in place (TrueCrypt does not allow the user to create a hidden volume within such a volume). For information on the method used to fill free volume space with random data, see chapter Technical Details, section TrueCrypt Volume Format Specification.

A hidden volume can be mounted the same way as a standard CipherShed volume: Click Select File or Select Device to select the outer/host volume (important: make sure the volume is not mounted). Then click Mount, and enter the password for the hidden volume. Whether the hidden or the outer volume will be mounted is determined by the entered password (i.e., when you enter the password for the outer volume, then the outer volume will be mounted; when you enter the password for the hidden volume, the hidden volume will be mounted).

CipherShed first attempts to decrypt the standard volume header using the entered password. If it fails, it loads the area of the volume where a hidden volume header can be stored (i.e. bytes 65536–131071, which contain solely random data when there is no hidden volume within the volume) to RAM and attempts to decrypt it using the entered password. Note that hidden volume headers cannot be identified, as they appear to consist entirely of random data. If the header is successfully decrypted (for information on how CipherShed determines that it was successfully decrypted, see the section Encryption Scheme), the information about the size of the hidden volume is retrieved from the decrypted header (which is still stored in RAM), and the hidden volume is mounted (its size also determines its offset).

A hidden volume can be created within any type of CipherShed volume, i.e., within a file-hosted volume or partition/device-hosted volume (requires administrator privileges). To create a hidden CipherShed volume, click on Create Volume in the main program window and select Create a hidden CipherShed volume. The Wizard will provide help and all information necessary to successfully create a hidden CipherShed volume. When creating a hidden volume, it may be very difficult or even impossible for an inexperienced user to set the size of the hidden volume such that the hidden volume does not overwrite data on the outer volume. Therefore, the Volume Creation Wizard automatically scans the cluster bitmap of the outer volume (before the hidden volume is created within it) and determines the maximum possible size of the hidden volume<sup>5</sup>.

If there are any problems when creating a hidden volume, refer to the chapter Troubleshooting for possible solutions.

Note that it is also possible to create and boot an operating system residing in a hidden volume (see the section Hidden Operating System in the chapter Plausible Deniability).

## **Protection of Hidden Volumes Against Damage**

If you mount a CipherShed volume within which there is a hidden volume, you may read data stored on the (outer) volume without any risk. However, if you (or the operating system) need to save data to the outer volume, there is a risk that the hidden volume will get damaged (overwritten). To prevent this, you should protect the hidden volume in a way described in this section.

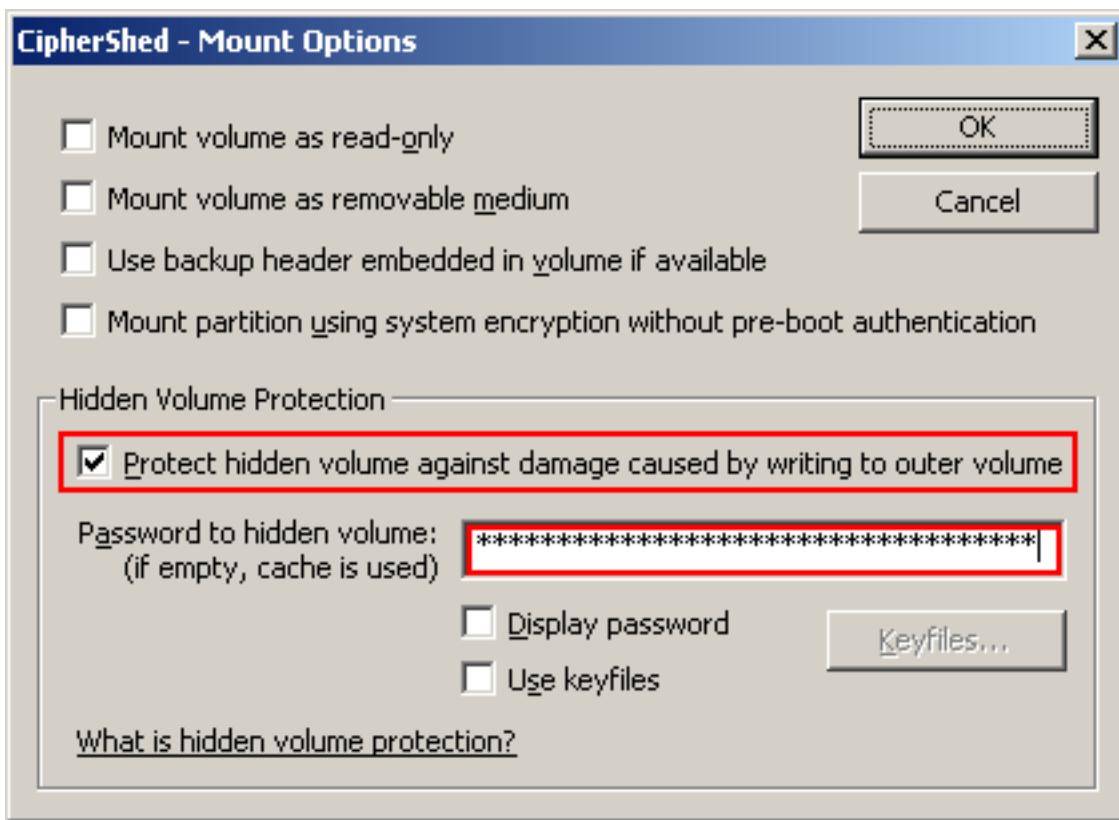
When mounting an outer volume, type in its password and before clicking OK, click Mount Options:

---

<sup>5</sup> The wizard scans the cluster bitmap to determine the size of the uninterrupted area of free space (if there is any) whose end is aligned with the end of the outer volume. This area accommodates the hidden volume and therefore the size of this area limits the maximum possible size of the hidden volume. On Linux and Mac OS X, the wizard actually does not scan the cluster bitmap, but the driver detects any data written to the outer volume and uses their position as previously described.



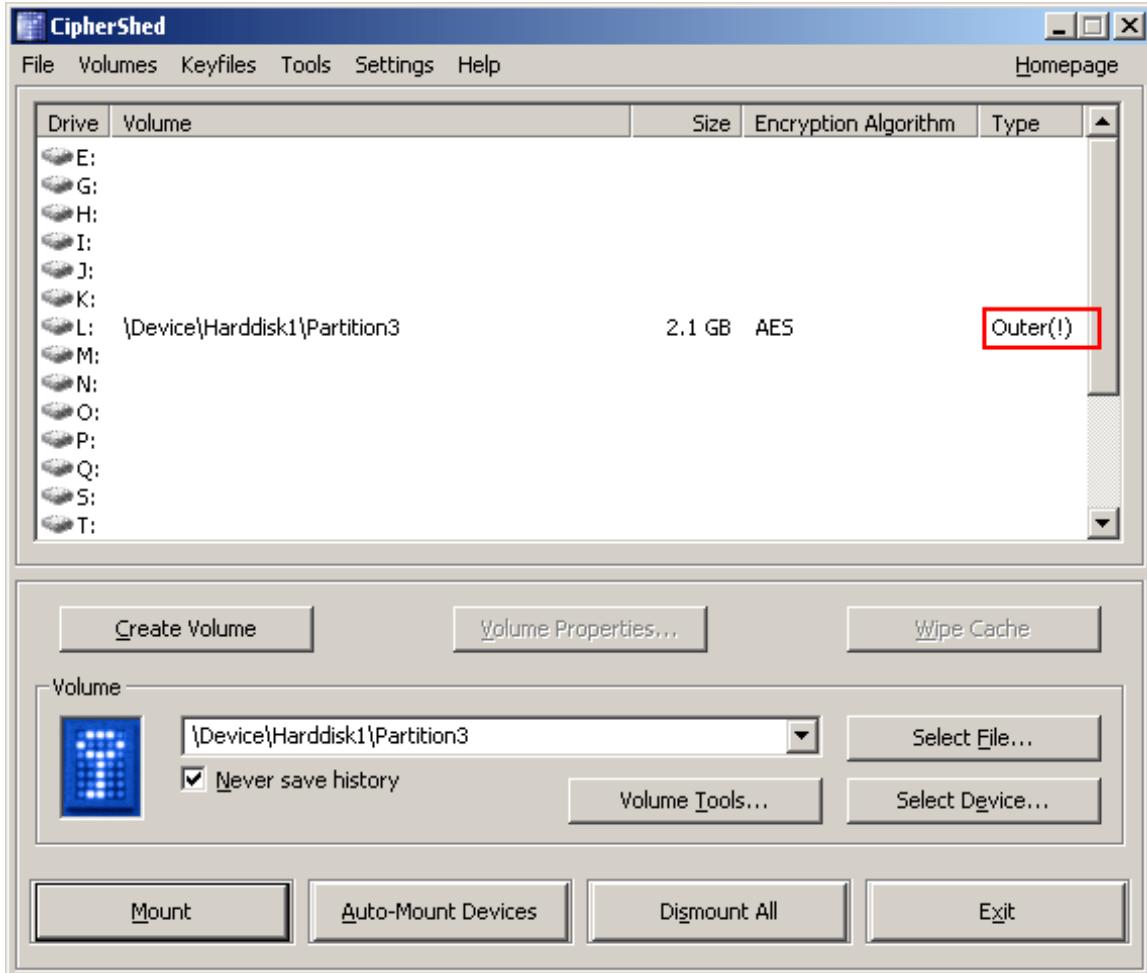
In the Mount Options dialog window, enable the option ‘Protect hidden volume against damage caused by writing to outer volume’. In the ‘Password to hidden volume’ input field, type the password for the hidden volume. Click OK and, in the main password entry dialog, click OK.



Both passwords must be correct; otherwise, the outer volume will not be mounted. When hidden volume protection is enabled, CipherShed does not actually mount the hidden volume. It only decrypts its header (in RAM) and retrieves information about the size of the hidden volume (from the decrypted header). Then, the outer volume is mounted and any attempt to save data to the area of the hidden volume will be rejected (until the outer volume is dismounted).

Note that CipherShed never modifies the filesystem (e.g., information about allocated clusters, amount of free space, etc.) within the outer volume in any way. As soon as the volume is dismounted, the protection is lost. When the volume is mounted again, it is not possible to determine whether the volume has used hidden volume protection or not. The hidden volume protection can be activated only by users who supply the correct password (and/or keyfiles) for the hidden volume (each time they mount the outer volume).

As soon as a write operation to the hidden volume area is denied/prevented (to protect the hidden volume), the entire host volume (both the outer and the hidden volume) becomes write-protected until dismounted (the CipherShed driver reports the ‘invalid parameter’ error to the system upon each attempt to write data to the volume). This preserves plausible deniability (otherwise certain kinds of inconsistency within the file system could indicate that this volume has used hidden volume protection). When damage to hidden volume is prevented, a warning is displayed (provided that the CipherShed Background Task is enabled – see the chapter CipherShed Background Task). Furthermore, the type of the mounted outer volume displayed in the main window changes to ‘Outer(!)’:



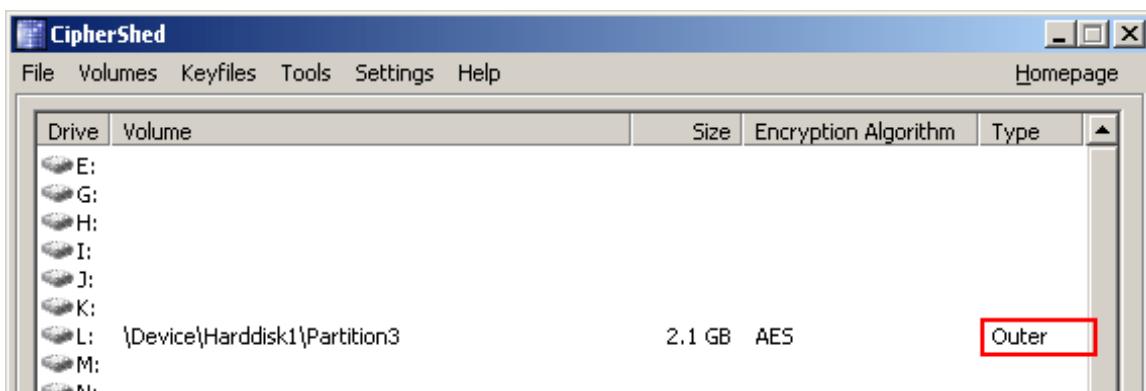
Moreover, the field Hidden Volume Protected in the Volume Properties dialog window says: ‘Yes (damage prevented!)’.

Note that when damage to hidden volume is prevented, no information about the event is written to the volume. When the outer volume is dismounted and mounted again, the volume properties will not display the string “damage prevented”.

There are several ways to check that a hidden volume is being protected against damage:

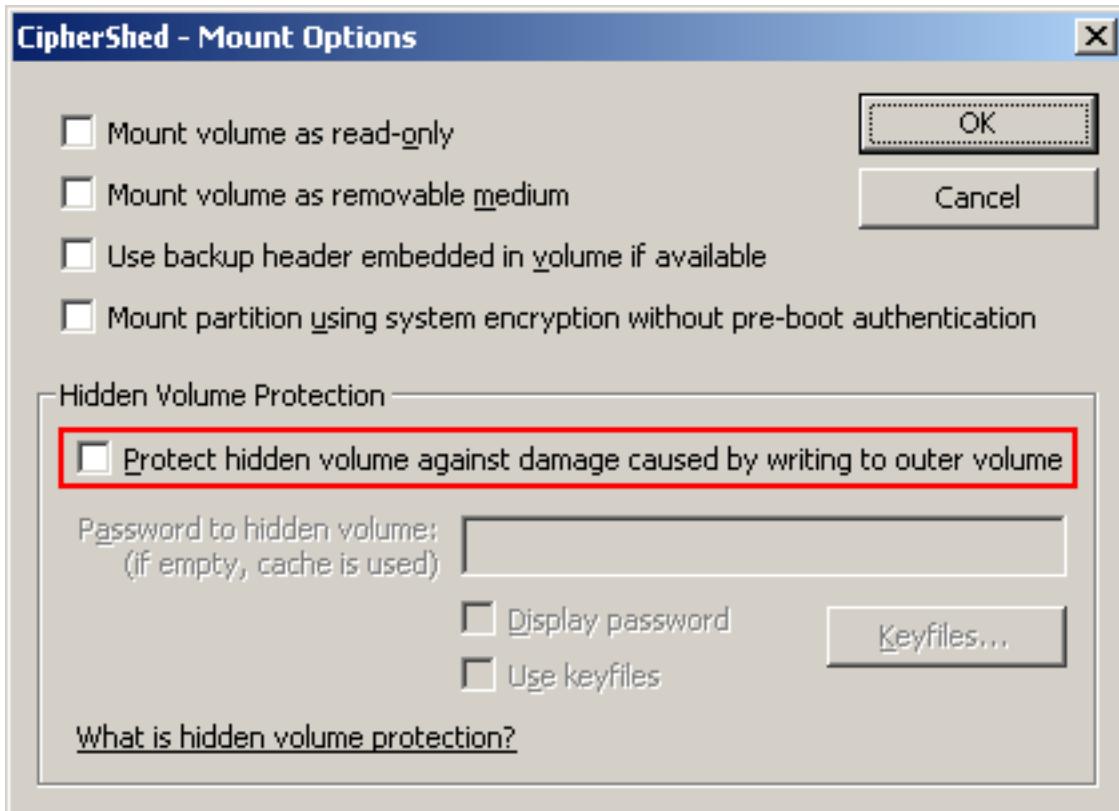
- 1 A confirmation message box saying that hidden volume is being protected is displayed after the outer volume is mounted (if it is not displayed, the hidden volume is not protected!).

- 2 In the Volume Properties dialog, the field Hidden Volume Protected says 'Yes':
- 3 The type of the mounted outer volume is Outer:



**Important:** When an adversary asks you to mount an outer volume, you, of course, must not mount the outer volume with the hidden volume protection enabled. You must mount it as a normal volume (and then CipherShed will not show the volume type "Outer" but "Normal"). Note that during the time when an outer volume is mounted with the hidden volume protection enabled, the adversary can find out that a hidden volume exists within the outer volume (he/she will be able to find it out until the volume is dismounted).

Warning: Note that the option 'Protect hidden volume against damage caused by writing to outer volume' in the Mount Options dialog window is automatically disabled after a mount attempt is completed, no matter whether it is successful or not (all hidden volumes that are already being protected will, of course, continue to be protected). Therefore, you need to check that option each time you attempt to mount the outer volume (if you wish the hidden volume to be protected):



If you want to mount an outer volume and protect a hidden volume within using cached passwords, then follow these steps: Hold down the Control (Ctrl) key when clicking Mount (or select Mount with Options from the Volumes menu). This will open the Mount Options dialog. Enable the option ‘Protect hidden volume against damage caused by writing to outer volume’ and leave the password box empty. Then click OK.

If you need to mount an outer volume and you know that you will not need to save any data to it, then the most comfortable way of protecting the hidden volume against damage is mounting the outer volume as read-only (see the section Mount Options).

## **Security Requirements and Precautions Pertaining to Hidden Volumes**

If you use a hidden CipherShed volume, you must follow the security requirements and precautions listed below in this section. Disclaimer: This section is not guaranteed to contain a list of all security issues and attacks that might adversely affect or limit the ability of CipherShed to secure data stored in a hidden CipherShed volume and the ability to provide plausible deniability.

- If an adversary has access to a (dismounted) CipherShed volume at several points over time, he may be able to determine which sectors of the volume are changing. If you change the contents of a hidden volume (e.g., create/copy new files to the hidden volume or modify/delete/rename/move files stored on the hidden volume, etc.), the contents of sectors (ciphertext) in the hidden volume area will change. After being given the password to the outer volume, the adversary might demand an explanation why these sectors changed. Your failure to provide a plausible explanation might indicate the existence of a hidden volume within the outer volume. Note that issues similar to the one described above may also arise, for example, in the following cases:
  - The file system in which you store a file-hosted CipherShed container has been defragmented and a copy of the CipherShed container (or of its fragment) remains in the free space on the host volume (in the defragmented file system). To prevent this, do one of the following:
    - Use a partition/device-hosted CipherShed volume instead of file-hosted.
    - Securely erase free space on the host volume (in the defragmented file system) after defragmenting.
    - Do not defragment file systems in which you store CipherShed volumes.
  - A file-hosted CipherShed container is stored in a journaling file system (such as NTFS). A copy of the CipherShed container (or of its fragment) may remain on the host volume. To prevent this, do one of the following:
    - Use a partition/device-hosted CipherShed volume instead of file-hosted.
    - Store the container in a non-journaling file system (for example, FAT32).
  - A CipherShed volume resides on a device/filesystem that utilizes a wear-leveling mechanism (e.g. a flash-memory SSD or USB flash drive). A copy of (a fragment of) the CipherShed volume may remain on the device. Therefore, do not store hidden volumes on such devices/filesystems. For more information on wear-leveling, see the section Wear-Leveling in the chapter Security Requirements and Precautions.

- A CipherShed volume resides on a device/filesystem that saves data (or on a device/filesystem that is controlled or monitored by a system/device that saves data) (e.g. the value of a timer or counter) that can be used to determine that a block had been written earlier than another block and/or to determine how many times a block has been written/read. Therefore, do not store hidden volumes on such devices/filesystems. To find out whether a device/system saves such data, please refer to documentation supplied with the device/system or contact the vendor/manufacturer.
  - A CipherShed volume resides on a device that is prone to wear (it is possible to determine that a block has been written/read more times than another block). Therefore, do not store hidden volumes on such devices/filesystems. To find out whether a device is prone to such wear, please refer to documentation supplied with the device or contact the vendor/manufacturer.
  - You back up content of a hidden volume by cloning its host volume or create a new hidden volume by cloning its host volume. Therefore, you must not do so. Follow the instructions in the chapter How to Back Up Securely and in the section Volume Clones.
- Make sure that Quick Format is disabled when encrypting a partition/device within which you intend to create a hidden volume.
  - On Windows, make sure you have not deleted any files within a volume within which you intend to create a hidden volume (the cluster bitmap scanner does not detect deleted files).
  - On Linux or Mac OS X, if you intend to create a hidden volume within a file-hosted CipherShed volume, make sure that the volume is not sparse-file-hosted (the Windows version of CipherShed verifies this and disallows creation of hidden volumes within sparse files).
  - When a hidden volume is mounted, the operating system and third-party applications may write to non-hidden volumes (typically, to the unencrypted system volume) unencrypted information about the data stored in the hidden volume (e.g. filenames and locations of recently accessed files, databases created by file indexing tools, etc.), the data itself in an unencrypted form (temporary files, etc.), unencrypted information about the filesystem residing in the hidden volume (which might be used e.g. to identify the filesystem and to determine whether it is the filesystem residing in the outer volume), the password/key for the hidden volume, or other types of sensitive data. Therefore, the following security requirements and precautions must be followed:

- Windows: Create a hidden operating system (for information on how to do so, see the section Hidden Operating System) and mount hidden volumes only when the hidden operating system is running. Note: When a hidden operating system is running, CipherShed ensures that all local unencrypted filesystems and non-hidden CipherShed volumes are read-only (i.e. no files can be written to such filesystems or CipherShed volumes).<sup>6</sup> Data is allowed to be written to filesystems within hidden CipherShed volumes. Alternatively, if a hidden operating system cannot be used, use a "live-CD" Windows PE system (entirely stored on and booted from a CD/DVD) that ensures that any data written to the system volume is written to a RAM disk. Mount hidden volumes only when such a "live-CD" system is running (if a hidden operating system cannot be used). In addition, during such a "live-CD" session, only filesystems that reside in hidden CipherShed volumes may be mounted in read-write mode (outer or unencrypted volumes/filesystems must be mounted as read-only or must not be mounted/accessible at all); otherwise, you must ensure that applications and the operating system do not write any sensitive data (see above) to non-hidden volumes/filesystems during the "live-CD" session.
- Linux: Download or create a "live-CD" version of your operating system (i.e. a "live" Linux system entirely stored on and booted from a CD/DVD) that ensures that any data written to the system volume is written to a RAM disk. Mount hidden volumes only when such a "live-CD" system is running. During the session, only filesystems that reside in hidden CipherShed volumes may be mounted in read-write mode (outer or unencrypted volumes/filesystems must be mounted as read-only or must not be mounted/accessible at all). If you cannot comply with this requirement and you are not able to ensure that applications and the operating system do not write any sensitive data (see above) to non-hidden volumes/filesystems, you must not mount or create hidden CipherShed volumes under Linux.
- Mac OS X: If you are not able to ensure that applications and the operating system do not write any sensitive data (see above) to non-hidden volumes/filesystems, you must not mount or create hidden CipherShed volumes under Mac OS X.
- When an outer volume is mounted with hidden volume protection enabled (see section Protection of Hidden Volumes Against Damage), you must follow the same security requirements and precautions that you are required to follow when a hidden volume is mounted (see above). The reason is that the operating system might leak the password/key for the hidden volume to a non-hidden or unencrypted volume.
- If you use an operating system residing within a hidden volume (see the section Hidden Operating System), then, in addition to the above, you must follow these security requirements and precautions:
  - You should use the decoy operating system as frequently as you use your computer. Ideally, you should use it for all activities that do not involve sensitive data. Otherwise, plausible deniability of the hidden operating system might be adversely affected (if you revealed the password for the decoy operating system to an adversary, he could find out that the system is not used very often, which might indicate the existence of a hidden operating system on your computer). Note that you can save data to the decoy system partition anytime without any risk that the hidden volume will get damaged (because the decoy system is not installed in the outer volume).

---

<sup>6</sup> This does not apply to filesystems on CD/DVD-like media and on custom, untypical, or non-standard devices/media.

- If the operating system requires activation, it must be activated before it is cloned (cloning is part of the process of creation of a hidden operating system — see the section Hidden Operating System) and the hidden operating system (i.e. the clone) must never be reactivated. The reason is that the hidden operating system is created by copying the content of the system partition to a hidden volume (so if the operating system is not activated, the hidden operating system will not be activated either). If you activated or reactivated a hidden operating system, the date and time of the activation (and other data) might be logged on a Microsoft server (and on the hidden operating system) but not on the decoy operating system. Therefore, if an adversary had access to the data stored on the server or intercepted your request to the server (and if you revealed the password for the decoy operating system to him), he might find out that the decoy operating system was activated (or reactivated) at a different time, which might indicate the existence of a hidden operating system on your computer. For similar reasons, any software that requires activation must be installed and activated before you start creating the hidden operating system.
- When you need to shut down the hidden system and start the decoy system, do not restart the computer. Instead, shut it down or hibernate it and then leave it powered off for at least several minutes (the longer, the better) before turning the computer on and booting the decoy system. This is required to clear the memory, which may contain sensitive data. For more information, see the section Unencrypted Data in RAM in the chapter Security Requirements and Precautions.
- The computer may be connected to a network (including the internet) only when the decoy operating system is running. When the hidden operating system is running, the computer should not be connected to any network, including the internet (one of the most reliable ways to ensure it is to unplug the network cable, if there is one). Note that if data is downloaded from or uploaded to a remote server, the date and time of the connection, and other data, are typically logged on the server. Various kinds of data are also logged on the operating system (e.g. Windows auto-update data, application logs, error logs, etc.) Therefore, if an adversary had access to the data stored on the server or intercepted your request to the server (and if you revealed the password for the decoy operating system to him), he might find out that the connection was not made from within the decoy operating system, which might indicate the existence of a hidden operating system on your computer. Also note that similar issues would affect you if there were any filesystem shared over a network under the hidden operating system (regardless of whether the filesystem is remote or local). Therefore, when the hidden operating system is running, there must be no filesystem shared over a network (in any direction).
- Any actions that can be detected by an adversary (or any actions that modify any data outside mounted hidden volumes) must be performed only when the decoy operating system is running (unless you have a plausible alternative explanation, such as using a "live-CD" system to perform such actions). For example, the option 'Auto-adjust for daylight saving time' option may be enabled only on the decoy system.
- If the BIOS, EFI, or any other component logs power-down events or any other events that could indicate a hidden volume/system is used (e.g. by comparing such events with the events in the Windows event log), you must either disable such logging or ensure that the log is securely erased after each session (or otherwise avoid such an issue in an appropriate way).

In addition to the above, you must follow the security requirements and precautions listed in the following chapters:

- Security Requirements and Precautions

- How to Back Up Securely

## ***Hidden Operating System***

If your system partition or system drive is encrypted using CipherShed, you need to enter your pre-boot authentication password in the CipherShed Boot Loader screen after you turn on or restart your computer. It may happen that you are forced by somebody to decrypt the operating system or to reveal the pre-boot authentication password. There are many situations where you cannot refuse to do so (for example, due to extortion). CipherShed allows you to create a hidden operating system whose existence should be impossible to prove (provided that certain guidelines are followed — see below). Thus, you will not have to decrypt or reveal the password for the hidden operating system.

Before you continue reading this section, make sure you have read the section **Hidden Volume** and that you understand what a hidden CipherShed volume is.

A hidden operating system is a system (for example, Windows 7 or Windows XP) that is installed in a hidden CipherShed volume. It should be impossible to prove that a hidden CipherShed volume exists (provided that certain guidelines are followed; for more information, see the section **Hidden Volume**) and, therefore, it should be impossible to prove that a hidden operating system exists.

However, in order to boot a system encrypted by CipherShed, an unencrypted copy of the CipherShed Boot Loader has to be stored on the system drive or on a CipherShed Rescue Disk. Hence, the mere presence of the CipherShed Boot Loader can indicate that there is a system encrypted by CipherShed on the computer. Therefore, to provide a plausible explanation for the presence of the CipherShed Boot Loader, the CipherShed wizard helps you create a second encrypted operating system, so-called decoy operating system, during the process of creation of a hidden operating system. A decoy operating system must not contain any sensitive files. Its existence is not secret (it is not installed in a hidden volume). The password for the decoy operating system can be safely revealed to anyone forcing you to disclose your pre-boot authentication password.<sup>7</sup>

You should use the decoy operating system as frequently as you use your computer. Ideally, you should use it for all activities that do not involve sensitive data. Otherwise, plausible deniability of the hidden operating system might be adversely affected (if you revealed the password for the decoy operating system to an adversary, he could find out that the system is not used very often, which might indicate the existence of a hidden operating system on your computer). Note that you can save data to the decoy system partition anytime without any risk that the hidden volume will get damaged (because the decoy system is not installed in the outer volume — see below).

There will be two pre-boot authentication passwords — one for the hidden system and the other for the decoy system. If you want to start the hidden system, you simply enter the password for the hidden system in the CipherShed Boot Loader screen (which appears after you turn on or restart your computer). Likewise, if you want to start the decoy system (for example, when asked to do so by an adversary), you just enter the password for the decoy system in the CipherShed Boot Loader screen.

---

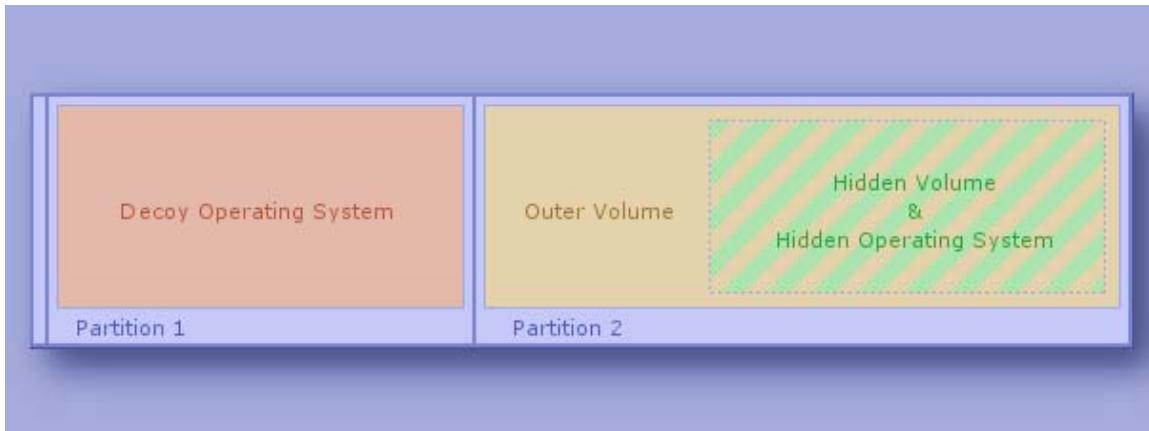
<sup>7</sup> It is not practical (and therefore is not supported) to install operating systems in two TrueCrypt volumes that are embedded within a single partition, because using the outer operating system would often require data to be written to the area of the hidden operating system (and if such write operations were prevented using the hidden volume protection feature, it would inherently cause system crashes, i.e. 'Blue Screen' errors).

Note: When you enter a pre-boot authentication password, the CipherShed Boot Loader first attempts to decrypt (using the entered password) the last 512 bytes of the first logical track of the system drive (where encrypted master key data for non-hidden encrypted system partitions/drives are normally stored). If it fails and if there is a partition behind the active partition, the CipherShed Boot Loader (even if there is actually no hidden volume on the drive) automatically tries to decrypt (using the same entered password again) the area of the first partition behind the active partition<sup>8</sup> where the encrypted header of a possible hidden volume might be stored. Note that CipherShed never knows if there is a hidden volume in advance (the hidden volume header cannot be identified, as it appears to consist entirely of random data). If the header is successfully decrypted (for information on how CipherShed determines that it was successfully decrypted, see the section Encryption Scheme), the information about the size of the hidden volume is retrieved from the decrypted header (which is still stored in RAM), and the hidden volume is mounted (its size also determines its offset). For further technical details, see the section Encryption Scheme in the chapter Technical Details.

When running, the hidden operating system appears to be installed on the same partition as the original operating system (the decoy system). However, in reality, it is installed within the partition behind it (in a hidden volume). All read/write operations are transparently redirected from the system partition to the hidden volume. Neither the operating system nor applications will know that data written to and read from the system partition is actually written to and read from the partition behind it (from/to a hidden volume). Any such data is encrypted and decrypted on the fly as usual (with an encryption key different from the one that is used for the decoy operating system).

Note that there will also be a third password — the one for the outer volume. It is not a pre-boot authentication password, but a regular CipherShed volume password. It can be safely disclosed to anyone forcing you to reveal the password for the encrypted partition where the hidden volume (containing the hidden operating system) resides. Thus, the existence of the hidden volume (and of the hidden operating system) will remain secret. If you are not sure you understand how this is possible, or what an outer volume is, please read the section Hidden Volume. The outer volume should contain some sensitive-looking files that you actually do not want to hide.

To summarize, there will be three passwords in total. Two of them can be revealed to an attacker (for the decoy system and for the outer volume). The third password, for the hidden system, must remain secret.



Example Layout of System Drive Containing Hidden Operating System

<sup>8</sup> If the size of the active partition is less than 256 MB, then the data is read from the second partition behind the active one (Windows 7 and later, by default, do not boot from the partition on which they are installed).

## **Process of Creation of Hidden Operating System**

To start the process of creation of a hidden operating system, select System > Create Hidden Operating System and then follow the instructions in the wizard.

Initially, the wizard verifies that there is a suitable partition for a hidden operating system on the system drive. Note that before you can create a hidden operating system, you need to create a partition for it on the system drive. It must be the first partition behind the system partition and it must be at least 5% larger than the system partition (the system partition is the one where the currently running operating system is installed). However, if the outer volume (not to be confused with the system partition) is formatted as NTFS, the partition for the hidden operating system must be at least 110% (2.1 times) larger than the system partition (the reason is that the NTFS file system always stores internal data exactly in the middle of the volume and, therefore, the hidden volume, which is to contain a clone of the system partition, can reside only in the second half of the partition).

In the next steps, the wizard will create two CipherShed volumes (outer and hidden) within the first partition behind the system partition. The hidden volume will contain the hidden operating system. The size of the hidden volume is always the same as the size of the system partition. The reason is that the hidden volume will need to contain a clone of the content of the system partition (see below). Note that the clone will be encrypted using a different encryption key than the original. Before you start copying some sensitive-looking files to the outer volume, the wizard tells you the maximum recommended size of space that the files should occupy, so that there is enough free space on the outer volume for the hidden volume.

Remark: After you copy some sensitive-looking files to the outer volume, the cluster bitmap of the volume will be scanned in order to determine the size of uninterrupted area of free space whose end is aligned with the end of the outer volume. This area will accommodate the hidden volume, so it limits its maximum possible size. The maximum possible size of the hidden volume will be determined and it will be verified that it is greater than the size of the system partition (which is required, because the entire content of the system partition will need to be copied to the hidden volume — see below). This ensures that no data stored on the outer volume will be overwritten by data written to the area of the hidden volume (e.g., when the system is being copied to it). The size of the hidden volume is always the same as the size of the system partition.

Then, CipherShed will create the hidden operating system by copying the content of the system partition to the hidden volume. Data being copied will be encrypted on the fly with an encryption key different from the one that will be used for the decoy operating system. The process of copying the system is performed in the pre-boot environment (before Windows starts) and it may take a long time to complete; several hours or even several days (depending on the size of the system partition and on the performance of the computer). You will be able to interrupt the process, shut down your computer, start the operating system and then resume the process. However, if you interrupt it, the entire process of copying the system will have to start from the beginning (because the content of the system partition must not change during cloning). The hidden operating system will initially be a clone of the operating system under which you started the wizard.

Windows creates (typically, without your knowledge or consent) various log files, temporary files, etc., on the system partition. It also saves the content of RAM to hibernation and paging files located on the system partition. Therefore, if an adversary analyzed files stored on the partition where the original system (of which the hidden system is a clone) resides, he might find out, for example, that you used the CipherShed wizard in the hidden-system-creation mode (which might indicate the existence of a hidden operating system on your computer). To prevent such issues, CipherShed will securely erase the entire content of the partition where the original system resides

after the hidden system has been created. Afterwards, in order to achieve plausible deniability, CipherShed will prompt you to install a new system on the partition and encrypt it using CipherShed. Thus, you will create the decoy system and the whole process of creation of the hidden operating system will be completed.

## Plausible Deniability and Data Leak Protection

For security reasons, when a hidden operating system is running, CipherShed ensures that all local unencrypted filesystems and non-hidden CipherShed volumes are read-only (i.e. no files can be written to such filesystems or CipherShed volumes)<sup>9</sup>. Data is allowed to be written to any filesystem that resides within a hidden CipherShed volume (provided that the hidden volume is not located in a container stored on an unencrypted filesystem or on any other read-only filesystem).

There are three main reasons why such countermeasures have been implemented:

- 1 It enables the creation of a secure platform for mounting of hidden CipherShed volumes. Note that we officially recommend that hidden volumes are mounted only when a hidden operating system is running. For more information, see the subsection Security Requirements and Precautions Pertaining to Hidden Volumes.
- 2 In some cases, it is possible to determine that, at a certain time, a particular filesystem was not mounted under (or that a particular file on the filesystem was not saved or accessed from within) a particular instance of an operating system (e.g. by analyzing and comparing filesystem journals, file timestamps, application logs, error logs, etc). This might indicate that a hidden operating system is installed on the computer. The countermeasures prevent these issues.
- 3 It prevents data corruption and allows safe hibernation. When Windows resumes from hibernation, it assumes that all mounted filesystems are in the same state as when the system entered hibernation. CipherShed ensures this by write-protecting any filesystem accessible both from within the decoy and hidden systems. Without such protection, the filesystem could become corrupted when mounted by one system while the other system is hibernated.

If you need to securely transfer files from the decoy system to the hidden system, follow these steps:

- 1 Start the decoy system.
- 2 Save the files to an unencrypted volume or to an outer/normal CipherShed volume.
- 3 Start the hidden system
- 4 If you saved the files to a CipherShed volume, mount it (it will be automatically mounted as read-only).
- 5 Copy the files to the hidden system partition or to another hidden volume.

---

<sup>9</sup> This does not apply to filesystems on CD/DVD-like media and on custom, atypical, or non-standard devices/media.

## Possible Explanations for Existence of Two CipherShed Partitions on Single Drive

An adversary might ask why you created two CipherShed-encrypted partitions on a single drive (a system partition and a non-system partition) rather than encrypting the entire disk with a single encryption key. There are many possible reasons to do that. However, if you do not know any (other than creating a hidden operating system), you can provide, for example, one of the following explanations:

- If there are more than two partitions on a system drive and you want to encrypt only two of them (the system partition and the one behind it) and to leave the other partitions unencrypted (for example, to achieve the best possible performance when reading and writing data, which is not sensitive, to such unencrypted partitions), the only way to do that is to encrypt both partitions separately (note that, with a single encryption key, CipherShed could encrypt the entire system drive and all partitions on it, but it cannot encrypt only two of them — only one or all of the partitions can be encrypted with a single key). As a result, there will be two adjacent CipherShed partitions on the system drive (the first will be a system partition, the second will be a non-system one), each encrypted with a different key (which is also the case when you create a hidden operating system, and therefore it can be explained this way). If you do not know any good reason why there should be more than one partition on a system drive at all: It is generally recommended to separate non-system files (documents) from system files. One of the easiest and most reliable ways to do that is to create two partitions on the system drive; one for the operating system and the other for documents (non-system files). The reasons why this practice is recommended include:
  - If the filesystem on one of the partitions is damaged, files on the partition may get corrupted or lost, whereas files on the other partition are not affected.
  - It is easier to reinstall the system without losing your documents (reinstallation of an operating system involves formatting the system partition, after which all files stored on it are lost). If the system is damaged, full reinstallation is often the only option.
- A cascade encryption algorithm (e.g. AES-Twofish-Serpent) can be many times slower than a non-cascade one (e.g. AES). However, a cascade encryption algorithm may be more secure than a non-cascade one (for example, the probability that three distinct encryption algorithms will be broken, e.g. due to advances in cryptanalysis, is significantly lower than the probability that only one of them will be broken). Therefore, if you encrypt the outer volume with a cascade encryption algorithm and the decoy system with a non-cascade encryption algorithm, you can answer that you wanted the best performance (and adequate security) for the system partition, and the highest possible security (but worse performance) for the non-system partition (i.e. the outer volume), where you store the most sensitive data, which you do not need to access very often (unlike the operating system, which you use very often, and therefore you need it to have the best possible performance). On the system partition, you store data that is less sensitive (but which you need to access very often) than data you store on the non-system partition (i.e. on the outer volume).
- Provided that you encrypt the outer volume with a cascade encryption algorithm (e.g. AES-Twofish-Serpent) and the decoy system with a non-cascade encryption algorithm (e.g. AES), you can also answer that you wanted to prevent the problems about which CipherShed warns when the user attempts to choose a cascade encryption algorithm for system encryption (see below for a list of the problems). Therefore, to prevent those problems, you decided to encrypt the system partition with a non-cascade encryption algorithm. However, you still wanted to use a cascade encryption algorithm (because it is more secure than a non-cascade encryption algorithm) for the most sensitive data, so you decided to create a second partition, which those problems do not affect (because it is non-system) and to encrypt it with a cascade encryption algorithm. On the system partition, you store data that is less sensitive than data you store on the non-system partition (i.e. on the outer volume).

- Note: When the user attempts to encrypt the system partition with a cascade encryption algorithm, CipherShed warns him or her that it can cause the following problems (and implicitly recommends to choose a non-cascade encryption algorithm instead):
  - For cascade encryption algorithms, the CipherShed Boot Loader is larger than normal and, therefore, there is not enough space in the first drive track for a backup of the CipherShed Boot Loader. Hence, whenever it gets damaged (which often happens, for example, during inappropriately designed anti-piracy activation procedures of certain programs), the user must use the CipherShed Rescue Disk to repair the CipherShed Boot Loader or to boot.
  - On some computers, resuming from hibernation takes longer.
- In contrast to a password for a non-system CipherShed volume, a pre-boot authentication password needs to be typed each time the computer is turned on or restarted. Therefore, if the pre-boot authentication password is long (which is required for security purposes), it may be very tiresome to type it so frequently. Hence, you can answer that it was more convenient for you to use a short (and therefore weaker) password for the system partition (i.e. the decoy system) and that it is more convenient for you to store the most sensitive data (which you do not need to access as often) in the non-system CipherShed partition (i.e. in the outer volume) for which you chose a very long password. As the password for the system partition is not very strong (because it is short), you do not intentionally store sensitive data on the system partition. However, you still prefer the system partition to be encrypted, because potentially sensitive or mildly sensitive data is stored on it as a result of your everyday use of the computer (for example, passwords to online forums you visit, which can be automatically remembered by your browser, browsing history, applications you run, etc.)
- When an attacker gets hold of your computer when a CipherShed volume is mounted (for example, when you use a laptop outside), he can, in most cases, read any data stored on the volume (data is decrypted on the fly as he reads it). Therefore, it may be wise to limit the time the volume is mounted to a minimum. Obviously, this may be impossible or difficult if the sensitive data is stored on an encrypted system partition or on an entirely encrypted system drive (because you would also have to limit the time you work with the computer to a minimum). Hence, you can answer that you created a separate partition (encrypted with a different key than your system partition) for your most sensitive data and that you mount it only when necessary and dismount it as soon as possible (so as to limit the time the volume is mounted to a minimum). On the system partition, you store data that is less sensitive (but which you need to access often) than data you store on the non-system partition (i.e. on the outer volume).

## **Safety/Security Precautions and Requirements Pertaining to Hidden Operating Systems**

As a hidden operating system resides in a hidden CipherShed volume, a user of a hidden operating system must follow all of the security requirements and precautions that apply to normal hidden CipherShed volumes. These requirements and precautions, as well as additional requirements and precautions pertaining specifically to hidden operating systems, are listed in the subsection Security Requirements and Precautions Pertaining to Hidden Volumes.

**WARNING:** If you do not protect the hidden volume (for information on how to do so, refer to the section Protection of Hidden Volumes Against Damage), do not write to the outer volume (note that the decoy operating system is not installed in the outer volume). Otherwise, you may overwrite and damage the hidden volume (and the hidden operating system within it)!

If all the instructions in the wizard have been followed and if the security requirements and precautions listed in the subsection Security Requirements and Precautions Pertaining to Hidden Volumes are followed, it should be impossible to prove that the hidden volume and hidden operating system exist, even when the outer volume is mounted or when the decoy operating system is decrypted or started.

# Main Program Window

## Select File

Allows you to select a file-hosted CipherShed volume. After you select it, you can perform various operations on it (e.g., mount it by clicking ‘Mount’). It is also possible to select a volume by dragging its icon to the ‘CipherShed.exe’ icon (CipherShed will be automatically launched then) or to the main program window.

## Select Device

Allows you to select a CipherShed partition or a storage device (such as a USB memory stick). After it is selected, you can perform various operations with it (e.g., mount it by clicking ‘Mount’).

Note: There is a more comfortable way of mounting CipherShed partitions/devices – see the section Auto-Mount Devices for more information.

## Mount

After you click ‘Mount’, CipherShed will try to mount the selected volume using cached passwords (if there are any) and if none of them works, it prompts you for a password. If you enter the correct password (and/or provide correct keyfiles), the volume will be mounted.

Important: Note that when you exit the CipherShed application, the CipherShed driver continues working and no CipherShed volume is dismounted.

## Auto-Mount Devices

This function allows you to mount CipherShed partitions/devices without having to select them manually (by clicking ‘Select Device’). CipherShed scans headers of all available partitions/devices on your system (except DVD drives and similar devices) one by one and tries to mount each of them as a CipherShed volume. Note that a CipherShed partition/device cannot be identified, nor the cipher it has been encrypted with. Therefore, the program cannot directly “find” CipherShed partitions. Instead, it has to try mounting each (even unencrypted) partition/device using all encryption algorithms and all cached passwords (if there are any). Therefore, be prepared that this process may take a long time on slow computers.

If the password you enter is wrong, mounting is attempted using cached passwords (if there are any). If you enter an empty password and if Use keyfiles is unchecked, only the cached passwords will be used when attempting to auto-mount partitions/devices. If you do not need to set mount options, you can bypass the password prompt by holding down the Shift key when clicking Auto-Mount Devices (only cached passwords will be used, if there are any).

Drive letters will be assigned starting from the one that is selected in the drive list in the main window.

## Dismount

This function allows you to dismount the CipherShed volume selected in the drive list in the main window. To dismount a CipherShed volume means to close it and make it impossible to read/write from/to the volume.

## Dismount All

Note: The information in this section applies to all menu items and buttons with the same or similar caption (for example, it also applies to the system tray menu item Dismount All).

This function allows you to dismount multiple CipherShed volumes. To dismount a CipherShed volume means to close it and make it impossible to read/write from/to the volume. This function dismounts all mounted CipherShed volumes except the following:

- Partitions/drives within the key scope of active system encryption (e.g., a system partition encrypted by CipherShed, or a non-system partition located on a system drive encrypted by CipherShed, mounted when the encrypted operating system is running).
- CipherShed volumes that are not fully accessible to the user account (e.g. a volume mounted from within another user account).
- CipherShed volumes that are not displayed in the CipherShed application window. For example, system favorite volumes attempted to be dismounted by an instance of CipherShed without administrator privileges when the option 'Allow only administrators to view and dismount system favorite volumes in CipherShed' is enabled.

## **Wipe Cache**

Clears all passwords (which may also contain processed keyfile contents) cached in driver memory. When there are no passwords in the cache, this button is disabled. For information on password cache, see the section Cache Password in Driver Memory.

## **Never Save History**

If this option disabled, the file names and/or paths of the last twenty files/devices that were attempted to be mounted as CipherShed volumes will be saved in the History file (whose content can be displayed by clicking on the Volume combo-box in the main window).

When this option is enabled, CipherShed clears the registry entries created by the Windows file selector for CipherShed, and sets the “current directory” to the user’s home directory (in portable mode, to the directory from which CipherShed was launched) whenever a container or keyfile is selected via the Windows file selector. Therefore, the Windows file selector will not remember the path of the last mounted container (or the last selected keyfile). However, note that the operations described in this paragraph are not guaranteed to be performed reliably and securely (see e.g. Security Requirements and Precautions) so we strongly recommend that you encrypt the system partition/drive instead of relying on them (see System Encryption).

Furthermore, if this option is enabled, the volume path input field in the main CipherShed window is cleared whenever you hide CipherShed.

Note: You can clear the volume history by selecting Tools -> Clear Volume History.

## **Exit**

Terminates the CipherShed application. The driver continues working and no CipherShed volumes are dismounted. When running in ‘portable’ mode, the CipherShed driver is unloaded when it is no longer needed (e.g., when all instances of the main application and/or of the Volume Creation Wizard are closed and no CipherShed volumes are mounted). However, if you force dismount on a

CipherShed volume when CipherShed runs in portable mode, or mount a writable NTFS-formatted volume on Windows Vista or later, the CipherShed driver may not be unloaded when you exit CipherShed (it will be unloaded only when you shut down or restart the system). This prevents various problems caused by a bug in Windows (for instance, it would be impossible to start CipherShed again as long as there are applications using the dismounted volume).

## **Volume Tools**

### **Change Volume Password**

See the section Volumes -> Change Volume Password.

### **Set Header Key Derivation Algorithm**

See the section Volumes -> Set Header Key Derivation Algorithm.

### **Backup Volume Header**

See the section Tools -> Backup Volume Header.

### **Restore Volume Header**

See the section Tools -> Restore Volume Header.

## **Program Menu**

Note: To save space, only the menu items that are not self-explanatory are described in this documentation.

### **Volumes -> Auto-Mount All Device-Hosted Volumes**

See the section Auto-Mount Devices.

### **Volumes -> Dismount All Mounted Volumes**

See the section Dismount All.

### **Volumes -> Change Volume Password**

Allows changing the password of the currently selected CipherShed volume (no matter whether the volume is hidden or standard). Only the header key and the secondary header key (XTS mode) are changed – the master key remains unchanged. This function re-encrypts the volume header using a header encryption key derived from a new password. Note that the volume header contains the master encryption key with which the volume is encrypted. Therefore, the data stored on the volume will not be lost after you use this function (password change will only take a few seconds).

To change a CipherShed volume password, click on Select File or Select Device, then select the volume, and from the Volumes menu select Change Volume Password.

Note: For information on how to change a password used for pre-boot authentication, please see the section System -> Change Password.

See also the chapter Security Requirements and Precautions.

#### **PKCS-5 PRF**

In this field you can select the algorithm that will be used in deriving new volume header keys (for more information, see the section Header Key Derivation, Salt, and Iteration Count) and in generating the new salt (for more information, see the section Random Number Generator).

Note: When CipherShed re-encrypts a volume header, the original volume header is first overwritten 256 times with random data to prevent adversaries from using techniques such as magnetic force microscopy or magnetic force scanning tunneling microscopy [17] to recover the overwritten header (however, see also the chapter Security Requirements and Precautions).

## **Volumes -> Set Header Key Derivation Algorithm**

This function allows you to re-encrypt a volume header with a header key derived using a different PRF function (for example, instead of HMAC-RIPemd-160 you could use HMAC-Whirlpool). Note that the volume header contains the master encryption key with which the volume is encrypted. Therefore, the data stored on the volume will not be lost after you use this function. For more

information, see the section Header Key Derivation, Salt, and Iteration Count.

Note: When CipherShed re-encrypts a volume header, the original volume header is first overwritten 256 times with random data to prevent adversaries from using techniques such as magnetic force microscopy or magnetic force scanning tunneling microscopy [17] to recover the overwritten header (however, see also the chapter Security Requirements and Precautions).

## **Volumes -> Add/Remove Keyfiles to/from Volume**

### **Volumes -> Remove All Keyfiles from Volume**

See the chapter Keyfiles.

## **Favorites -> Add Mounted Volume to Favorites**

### **Favorites -> Organize Favorite Volumes**

### **Favorites -> Mount Favorites Volumes**

See the chapter Favorite Volumes.

## **Favorites -> Add Mounted Volume to System Favorites**

### **Favorites -> Organize System Favorite Volumes**

See the chapter System Favorite Volumes.

## **System -> Change Password**

Changes the password used for pre-boot authentication (see the chapter System Encryption).

**WARNING:** Your CipherShed Rescue Disk allows you to restore key data if it is damaged. By doing so, you also restore the password that was valid when the CipherShed Rescue Disk was created. Therefore, whenever you change the password, you should destroy your CipherShed Rescue Disk and create a new one (select System -> Create Rescue Disk). Otherwise, an attacker could decrypt your system partition/drive using the old password (if he finds the old CipherShed Rescue Disk and uses it to restore the key data). See also the chapter Security Requirements and Precautions.

For more information on changing a password, please see the section Volumes -> Change Volume Password above.

## **System -> Mount Without Pre-Boot Authentication**

Check this option, if you need to mount a partition that is within the key scope of system encryption without pre-boot authentication. For example, if you need to mount a partition located on the encrypted system drive of another operating system that is not running. This can be useful e.g. when you need to back up or repair an operating system encrypted by CipherShed (from within another operating system).

Note 1: If you need to mount multiple partitions at once, click ‘Auto-Mount Devices’, then click ‘Mount Options’ and enable the option ‘Mount partition using system encryption without pre-boot authentication’.

Please note you cannot use this function to mount extended (logical) partitions that are located on an entirely encrypted system drive.

## **Tools -> Clear Volume History**

Clears the list containing the file names (if file-hosted) and paths of the last twenty successfully mounted volumes.

## **Tools -> Traveler Disk Setup**

See the chapter Portable Mode.

## **Tools -> Keyfile Generator**

See section Tools -> Keyfile Generator in the chapter Keyfiles.

## **Tools -> Backup Volume Header**

## **Tools -> Restore Volume Header**

If the header of a CipherShed volume is damaged, the volume is, in most cases, impossible to mount. Therefore, each volume created by CipherShed (TrueCrypt 6.0 or later) contains an embedded backup header, located at the end of the volume. For extra safety, you can also create external volume header backup files. To do so, click Select Device or Select File, select the volume, select Tools -> Backup Volume Header, and then follow the instructions.

Note: A backup header (embedded or external) is not a copy of the original volume header because it is encrypted with a different header key derived using a different salt (see the section Header Key Derivation, Salt, and Iteration Count). When the volume password and/or keyfiles are changed, or when the header is restored from the embedded (or an external) header backup, both the volume header and the backup header (embedded in the volume) are re-encrypted with header keys derived using newly generated salts (the salt for the volume header is different from the salt for the backup header). Each salt is generated by the CipherShed random number generator (see the section Random Number Generator).

Both types of header backups (embedded and external) can be used to repair a damaged volume header. To do so, click Select Device or Select File, select the volume, select Tools -> Restore Volume Header, and then follow the instructions.

**WARNING:** Restoring a volume header also restores the volume password that was valid when the backup was created. Moreover, if keyfile(s) are/is necessary to mount a volume when the backup is created, the same keyfile(s) will be necessary to mount the volume again after the volume header is restored. For more information, see the section Encryption Scheme in the chapter Technical Details.

After you create a volume header backup, you might need to create a new one only when you change the volume password and/or keyfiles. Otherwise, the volume header remains unmodified so the volume header backup remains up-to-date.

Note: Apart from salt (which is a sequence of random numbers), external header backup files do not contain any unencrypted information and they cannot be decrypted without knowing the correct password and/or supplying the correct keyfile(s). For more information, see the chapter Technical Details.

When you create an external header backup, both the standard volume header and the area where a hidden volume header can be stored is backed up, even if there is no hidden volume within the volume (to preserve plausible deniability of hidden volumes). If there is no hidden volume within the volume, the area reserved for the hidden volume header in the backup file will be filled with random data (to preserve plausible deniability).

When restoring a volume header, you need to choose the type of volume whose header you wish to restore (a standard or hidden volume). Only one volume header can be restored at a time. To restore both headers, you need to use the function twice (Tools -> Restore Volume Header). You will need to enter the correct password (and/or to supply the correct keyfiles) that was/were valid when the volume header backup was created. The password (and/or keyfiles) will also automatically determine the type of the volume header to restore, i.e. standard or hidden (note that CipherShed determines the type through the process of trial and error).

Note: If the user fails to supply the correct password (and/or keyfiles) twice in a row when trying to mount a volume, CipherShed will automatically try to mount the volume using the embedded backup header (in addition to trying to mount it using the primary header) each subsequent time that the user attempts to mount the volume (until he or she clicks Cancel). If CipherShed fails to decrypt the primary header but it successfully decrypts the embedded backup header at the same time, the volume is mounted and the user is warned that the volume header is damaged (and informed as to how to repair it).

## **Settings -> Preferences**

Invokes the Preferences dialog window, where you can change, among others, the following options:

### Wipe cached passwords on exit

If enabled, passwords (which may also contain processed keyfile contents) cached in driver memory will be cleared when CipherShed exits.

### Cache passwords in driver memory

When checked, passwords and/or processed keyfile contents for up to last four successfully mounted CipherShed volumes are cached. This allows mounting volumes without having to type their passwords (and selecting keyfiles) repeatedly. CipherShed never saves any password to a disk (however, see the chapter Security Requirements and Precautions). Password caching can be enabled/disabled in the Preferences (Settings -> Preferences) and in the password prompt window. If the system partition/drive is encrypted, caching of the pre-boot authentication password can be enabled or disabled in the system encryption settings (Settings > ‘System Encryption’).

### Open Explorer window for successfully mounted volume

If this option is checked, then after a CipherShed volume has been successfully mounted, an Explorer window showing the root directory of the volume (e.g., T:\) will be automatically opened.

### Use a different taskbar icon when there are mounted volumes

If enabled, the appearance of the CipherShed taskbar icon (shown within the system tray notification area) is different while a CipherShed volume is mounted, except the following:

- Partitions/drives within the key scope of active system encryption (e.g., a system partition encrypted by CipherShed, or a non-system partition located on a system drive encrypted by CipherShed, mounted when the encrypted operating system is running).

- CipherShed volumes that are not fully accessible to the user account (e.g. a volume mounted from within another user account).
- CipherShed volumes that are not displayed in the CipherShed application window. For example, system favorite volumes attempted to be dismounted by an instance of CipherShed without administrator privileges when the option 'Allow only administrators to view and dismount system favorite volumes in CipherShed' is enabled.

#### CipherShed Background Task – Enabled

See the chapter CipherShed Background Task.

#### CipherShed Background Task – Exit when there are no mounted volumes

If this option is checked, the CipherShed background task automatically and silently exits as soon as there are no mounted CipherShed volumes. For more information, see the chapter CipherShed Background Task. Note that this option cannot be disabled when CipherShed runs in portable mode.

#### Auto-dismount volume after no data has been read/written to it for

After no data has been written/read to/from a CipherShed volume for n minutes, the volume is automatically dismounted.

#### Force auto-dismount even if volume contains open files or directories

This option applies only to auto-dismount (not to regular dismount). It forces dismount (without prompting) on the volume being auto-dismounted in case it contains open files or directories (i.e., file/directories that are in use by the system or applications).

### ***Mounting CipherShed Volumes***

If you have not done so yet, please read the sections ‘Mount’ and ‘Auto-Mount Devices’ in the chapter Main Program Window.

### **Cache Password in Driver Memory**

This option can be set in the password entry dialog so that it will apply only to that particular mount attempt. It can also be set as default in the Preferences. For more information, please see the section Settings -> Preferences, subsection Cache passwords in driver memory.

### **Mount Options**

Mount options affect the parameters of the volume being mounted. The Mount Options dialog can be opened by clicking on the Mount Options button in the password entry dialog. When a correct password is cached, volumes are automatically mounted after you click Mount. If you need to change mount options for a volume being mounted using a cached password, hold down the Control (Ctrl) key while clicking Mount or a favorite volume in the Favorites menu, or select Mount with Options from the Volumes menu.

Default mount options can be configured in the main program preferences (Settings -> Preferences).

### **Mount volume as read-only**

When checked, it will not be possible to write any data to the mounted volume.

### **Mount volume as removable medium**

See section Volume Mounted as Removable Medium.

### **Use backup header embedded in volume if available**

All volumes created by CipherShed (TrueCrypt 6.0 or later) contain an embedded backup header (located at the end of the volume). If you check this option, CipherShed will attempt to mount the volume using the embedded backup header. Note that if the volume header is damaged, you do not have to use this option. Instead, you can repair the header by selecting Tools > Restore Volume Header.

### **Mount partition using system encryption without pre-boot authentication**

Check this option, if you need to mount a partition that is within the key scope of system encryption without pre-boot authentication. For example, if you need to mount a partition located on the encrypted system drive of another operating system that is not running. This can be useful e.g. when you need to back up or repair an operating system encrypted by CipherShed (from within another operating system). Note that this option can be enabled also when using the ‘Auto-Mount Devices’ or ‘Auto-Mount All Device-Hosted Volumes’ functions.

### **Hidden Volume Protection**

Please see the section Protection of Hidden Volumes Against Damage.

## Parallelization

When your computer has a multi-core processor (or multiple processors), CipherShed uses all of the cores (or processors) in parallel for encryption and decryption. For example, when CipherShed is to decrypt a chunk of data, it first splits the chunk into several smaller pieces. The number of the pieces is equal to the number of the cores (or processors). Then, all of the pieces are decrypted in parallel (piece 1 is decrypted by thread 1, piece 2 is decrypted by thread 2, etc). The same method is used for encryption.

So if your computer has, for example, a quad-core processor, then encryption and decryption are four times faster than on a single-core processor with equivalent specifications (likewise, they are twice faster on dual-core processors, etc).

Increase in encryption/decryption speed is directly proportional to the number of cores and/or processors.

Note: Processors with the Hyper-Threading technology provide multiple logical cores per one physical core (or multiple logical processors per one physical processor). When Hyper Threading is enabled in the computer firmware (e.g. BIOS) settings, CipherShed creates one thread for each logical core/processor. For example, on a 6-core processor that provides two logical cores per one physical core, CipherShed uses 12 threads.

When your computer has a multi-core processor/CPU (or multiple processors/CPUs), header key derivation is parallelized too. As a result, mounting of a volume is several times faster on a multi-core processor (or multi-processor computer) than on a single-core processor (or a single-processor computer) with equivalent specifications.

Note: Parallelization was introduced in TrueCrypt 6.0.

## Pipelining

When encrypting or decrypting data, CipherShed uses so-called pipelining (asynchronous processing). While an application is loading a portion of a file from a CipherShed-encrypted volume/drive, CipherShed is automatically decrypting it (in RAM). Thanks to pipelining, the application does not have to wait for any portion of the file to be decrypted and it can start loading other portions of the file right away. The same applies to encryption when writing data to an encrypted volume/drive.

Pipelining allows data to be read from and written to an encrypted drive as fast as if the drive was not encrypted (the same applies to file-hosted and partition-hosted CipherShed volumes)<sup>10</sup>.

Note: Pipelining was introduced in TrueCrypt 5.0 and it is implemented only in the Windows versions of CipherShed.

---

<sup>10</sup> Some solid-state drives compress data internally, which appears to increase the actual read/write speed when the data is compressible (for example, text files). However, encrypted data cannot be compressed (as it appears to consist solely of random "noise" without any compressible patterns). This may have various implications. For example, benchmarking software that reads or writes compressible data (such as sequences of zeroes) will report lower speeds on encrypted volumes than on unencrypted volumes (to avoid this, use benchmarking software that reads/writes random or other kinds of uncompressible data).

## Hardware Acceleration

Some processors (CPUs) support hardware-accelerated AES encryption<sup>11</sup>, which is typically 4-8 times faster than encryption performed by the purely software implementation on the same processors.

By default, CipherShed uses hardware-accelerated AES on computers that have a processor where the Intel AES-NI instructions are available. Specifically, CipherShed uses the AES-NI instructions that perform so-called AES rounds (i.e. the main portions of the AES algorithm)<sup>12</sup>. CipherShed does not use any of the AES-NI instructions that perform key generation.

Note: By default, CipherShed uses hardware-accelerated AES also when an encrypted Windows system is booting or resuming from hibernation (provided that the processor supports the Intel AES-NI instructions).

To find out whether CipherShed can use hardware-accelerated AES on your computer, select Settings > Performance and check the field labeled 'Processor (CPU) in this computer supports hardware acceleration for AES'.

To find out whether a processor you want to purchase supports the Intel AES-NI instructions (also called "AES New Instructions"), which CipherShed uses for hardware-accelerated AES, please check the documentation for the processor or contact the vendor/manufacturer. However, note that some Intel processors, which the Intel website lists as AES-NI-supporting, actually support the AES-NI instructions only with a Processor Configuration update. In such cases, you should contact the manufacturer of the motherboard/computer for a BIOS update that includes the latest Processor Configuration update for the processor.

If you want to disable hardware acceleration of AES (e.g. because you want CipherShed to use only a fully open-source implementation of AES), you can do so by selecting Settings > Performance and disabling the option 'Accelerate AES encryption/decryption by using the AES instructions of the processor'. Note that when this setting is changed, the operating system needs to be restarted to ensure that all CipherShed components internally perform the requested change of mode. Also note that when you create a CipherShed Rescue Disk, the state of this option is written to the Rescue Disk and used whenever you boot from it (affecting the pre-boot and initial boot phase). To create a new CipherShed Rescue Disk, select System > Create Rescue Disk.

Note: Support for hardware acceleration was introduced in TrueCrypt 7.0.

---

<sup>11</sup> In this chapter, the word 'encryption' also refers to decryption.

<sup>12</sup> Those instructions are AESENC, AESENCLAST, AESDEC, and AESDECLAST and they perform the following AES transformations: ShiftRows, SubBytes, MixColumns, InvShiftRows, InvSubBytes, InvMixColumns, and AddRoundKey (for more details about these transformations, see [3]).

## **Hot Keys**

To set system-wide CipherShed hot keys, click Settings -> Hot Keys. Note that hot keys work only when CipherShed or the CipherShed Background Task is running.

## Keyfiles

Keyfile is a file whose content is combined with a password (for information on the method used to combine a keyfile with password, see the chapter Technical Details, section Keyfiles). Until the correct keyfile is provided, no volume that uses the keyfile can be mounted.

You do not have to use keyfiles. However, using keyfiles has some advantages:

- May improve protection against brute force attacks (significant particularly if the volume password is not very strong).
- Allows the use of security tokens and smart cards (see below).
- Allows multiple users to mount a single volume using different user passwords or PINs. Just give each user a security token or smart card containing the same CipherShed keyfile and let them choose their personal password or PIN that will protect their security token or smart card.
- Allows managing multi-user shared access (all keyfile holders must present their keyfiles before a volume can be mounted).

Any kind of file (for example, .txt, .exe, mp3<sup>13</sup>, .avi) can be used as a CipherShed keyfile (however, we recommend that you prefer compressed files, such as .mp3, .jpg, .zip, etc). Note that CipherShed never modifies the keyfile contents.

You can select more than one keyfile; the order does not matter. You can also let CipherShed generate a file with random content and use it as a keyfile. To do so, select Tools -> Keyfile Generator.

Note: Keyfiles are currently not supported for system encryption.

**WARNING:** If you lose a keyfile or if any bit of its first 1024 kilobytes changes, it will be impossible to mount volumes that use the keyfile!

**WARNING:** *If password caching is enabled, the password cache also contains the processed contents of keyfiles used to successfully mount a volume. Then it is possible to remount the volume even if the keyfile is not available/accessible.* To prevent this, click ‘Wipe Cache’ or disable password caching (for more information, please see the section Settings -> Preferences, subsection Cache passwords in driver memory).

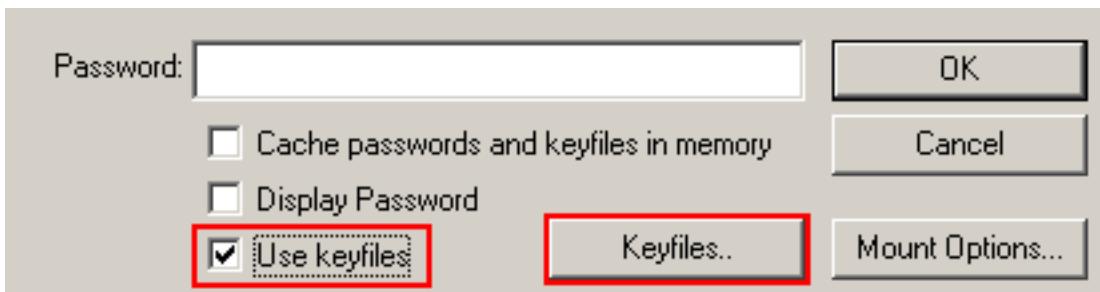
See also the section Choosing Passwords and Keyfiles in the chapter Security Requirements and Precautions.

## Keyfiles Dialog Window

If you want to use keyfiles (i.e. “apply” them) when creating or mounting volumes, or changing passwords, look for the ‘Use keyfiles’ option and the Keyfiles button below a password input field.

---

<sup>13</sup> However, if you use an MP3 file as a keyfile, you must ensure that no program modifies the ID3 tags (e.g. song title, name of artist, etc.) within the MP3 file. Otherwise, it will be impossible to mount volumes that use the keyfile.



These control elements appear in various dialog windows and always have the same functions. Check the Use keyfiles option and click Keyfiles. The keyfile dialog window should appear where you can specify keyfiles (to do so, click Add Files or Add Token Files) or keyfile search paths (click Add Path).

## Security Tokens and Smart Cards

CipherShed can directly use keyfiles stored on a security token or smart card that complies with the PKCS #11 (2.0 or later) standard [23] and that allows the user to store a file (data object) on the token/card. To use such files as CipherShed keyfiles, click Add Token Files (in the keyfile dialog window).

Access to a keyfile stored on a security token or smart card is typically protected by PIN codes, which can be entered either using a hardware PIN pad or via the CipherShed GUI. It can also be protected by other means, such as fingerprint readers.

In order to allow CipherShed to access a security token or smart card, you need to install a PKCS #11 (2.0 or later) software library for the token or smart card first. Such a library may be supplied with the device or it may be available for download from the website of the vendor or other third parties.

If your security token or smart card does not contain any file (data object) that you could use as a CipherShed keyfile, you can use CipherShed to import any file to the token or smart card (if it is supported by the device). To do so, follow these steps:

- 1 In the keyfile dialog window, click Add Token Files.
- 2 If the token or smart card is protected by a PIN, password, or other means (such as a fingerprint reader), authenticate yourself (for example, by entering the PIN using a hardware PIN pad).
- 3 The ‘Security Token Keyfile’ dialog window should appear. In it, click Import Keyfile to Token and then select the file you want to import to the token or smart card.

Note that you can import for example 512-bit keyfiles with random content generated by CipherShed (see Tools -> Keyfile Generator below).

To close all opened security token sessions, either select Tools > Close All Security Token Sessions or define and use a hotkey combination (Settings > Hot Keys > Close All Security Token Sessions).

## Keyfile Search Path

By adding a folder in the keyfile dialog window (click Add Path), you specify a keyfile search path. All files found in the keyfile search path<sup>14</sup> will be used as keyfiles.

Important: Note that folders (and files they contain) found in keyfile search paths are ignored.

Keyfile search paths are especially useful if you, for example, store keyfiles on a USB memory stick that you carry with you. You can set the drive letter of the USB memory stick as a default keyfile search path. To do so, select Settings -> Default Keyfiles. Then click Add Path, browse to the drive letter assigned to the USB memory stick, and click OK. Now each time you mount a volume (and if the option Use keyfiles is checked in the password dialog window), CipherShed will scan the path and use all files that it finds on the USB memory stick as keyfiles.

**WARNING:** When you add a folder (as opposed to a file) to the list of keyfiles, only the path is remembered, not the filenames! This means e.g. that if you create a new file in the folder or if you copy an additional file to the folder, then all volumes that used keyfiles from the folder will be impossible to mount (until you remove the newly added file from the folder).

## Empty Password & Keyfile

When a keyfile is used, the password may be empty, so the keyfile may become the only item necessary to mount the volume (which we do not recommend). If default keyfiles are set and enabled when mounting a volume, then before prompting for a password, CipherShed first automatically attempts to mount using an empty password plus default keyfiles (however, this does not apply to the ‘Auto-Mount Devices’ function). If you need to set Mount Options (e.g., mount as read-only, protect hidden volume etc.) for a volume being mounted this way, hold down the Control (Ctrl) key while clicking Mount (or select Mount with Options from the Volumes menu). This will open the Mount Options dialog.

## Quick Selection

Keyfiles and keyfile search paths can be quickly selected in the following ways:

- Right-click the Keyfiles button in the password entry dialog window and select one of the menu items.
- Drag the corresponding file/folder icons to the keyfile dialog window or to the password entry dialog.

## Volumes -> Add/Remove Keyfiles to/from Volume

This function allows you to re-encrypt a volume header with a header encryption key derived from any number of keyfiles (with or without a password), or no keyfiles at all. Thus, a volume which is possible to mount using only a password can be converted to a volume that require keyfiles (in addition to the password) in order to be possible to mount. Note that the volume header contains the master encryption key with which the volume is encrypted. Therefore, the data stored on the volume will not be lost after you use this function.

This function can also be used to change/set volume keyfiles (i.e., to remove some or all keyfiles, and to apply new ones).

---

<sup>14</sup> Found at the time when you are mounting the volume, changing its password, or performing any other operation that involves re-encryption of the volume header.

Remark: This function is internally equal to the Password Change function.

When CipherShed re-encrypts a volume header, the original volume header is first overwritten 256 times with random data to prevent adversaries from using techniques such as magnetic force microscopy or magnetic force scanning tunneling microscopy [17] to recover the overwritten header (however, see also the chapter Security Requirements and Precautions).

## **Volumes -> Remove All Keyfiles from Volume**

This function allows you to re-encrypt a volume header with a header encryption key derived from a password and no keyfiles (so that it can be mounted using only a password, without any keyfiles). Note that the volume header contains the master encryption key with which the volume is encrypted. Therefore, the data stored on the volume will not be lost after you use this function.

Remark: This function is internally equal to the Password Change function.

When CipherShed re-encrypts a volume header, the original volume header is first overwritten 256 times with random data to prevent adversaries from using techniques such as magnetic force microscopy or magnetic force scanning tunneling microscopy [17] to recover the overwritten header (however, see also the chapter Security Requirements and Precautions).

## **Tools -> Keyfile Generator**

You can use this function to generate a file with random content, which you can use as a keyfile (recommended). This function uses the CipherShed Random Number Generator. Note that the resulting file size is always 64 bytes (i.e., 512 bits), which is also the maximum possible CipherShed password length.

## **Settings -> Default Keyfiles**

Use this function to set default keyfiles and/or default keyfile search paths. This function is particularly useful if you, for example, store keyfiles on a USB memory stick that you carry with you. You can add its drive letter to the default keyfile configuration. To do so, click Add Path, browse to the drive letter assigned to the USB memory stick, and click OK. Now each time you mount a volume (and if Use keyfiles is checked in the password dialog), CipherShed will scan the path and use all files that it finds there as keyfiles.

***WARNING: When you add a folder (as opposed to a file) to your default keyfile list, only the path is remembered, not the filenames! This means e.g. that if you create a new file in the folder or if you copy an additional file to the folder, then all volumes that used keyfiles from the folder will be impossible to mount (until you remove the newly added file from the folder).***

**IMPORTANT:** Note that when you set default keyfiles and/or default keyfile search paths, the filenames and paths are saved unencrypted in the file Default Keyfiles.xml. For more information, please see the chapter CipherShed System Files & Application Data.

## **Security Tokens & Smart Cards**

CipherShed supports security (or cryptographic) tokens and smart cards (smart card readers) that can be accessed using the PKCS #11 (2.0 or later) protocol [23]. For more information, please see the section Security Tokens and Smart Cards in the chapter Keyfiles.

## Portable Mode

CipherShed can run in so-called portable mode, which means that it does not have to be installed on the operating system under which it is run. However, there are two things to keep in mind:

- 1) You need administrator privileges in order to be able to run CipherShed in portable mode (for the reasons, see the chapter Using CipherShed Without Administrator Privileges).

**Note:** *No matter what kind of software you use, as regards personal privacy in most cases, it is not safe to work with sensitive data under systems where you do not have administrator privileges, as the administrator can easily capture and copy your sensitive data, including passwords and keys.*

- 2) After examining the registry file, it may be possible to tell that CipherShed was run (and that a CipherShed volume was mounted) on a Windows system even if it had been run in portable mode.

Note: If that is a problem, see this question in the FAQ for a possible solution.

There are two ways to run CipherShed in portable mode:

- 1) After you extract files from the CipherShed self-extracting package, you can directly run CipherShed.exe.

Note: To extract files from the CipherShed self-extracting package, run it, and then select Extract (instead of Install) on the second page of the CipherShed Setup wizard.

- 2) You can use the Traveler Disk Setup facility to prepare a special traveler disk and launch CipherShed from there.

The second option has several advantages, which are described in the following sections in this chapter.

Note: When running in ‘portable’ mode, the CipherShed driver is unloaded when it is no longer needed (e.g., when all instances of the main application and/or of the Volume Creation Wizard are closed and no CipherShed volumes are mounted). However, if you force dismount on a CipherShed volume when CipherShed runs in portable mode, or mount a writable NTFS-formatted volume on Windows Vista or later, the CipherShed driver may not be unloaded when you exit CipherShed (it will be unloaded only when you shut down or restart the system). This prevents various problems caused by a bug in Windows (for instance, it would be impossible to start CipherShed again as long as there are applications using the dismounted volume).

## Tools -> Traveler Disk Setup

You can use this facility to prepare a special traveler disk and launch CipherShed from there. Note that CipherShed ‘traveler disk’ is not a CipherShed volume but an unencrypted volume. A ‘traveler disk’ contains CipherShed executable files and optionally the ‘autorun.inf’ script (see the section AutoRun Configuration below). After you select Tools -> Traveler Disk Setup, the Traveler Disk Setup dialog box should appear. Some of the parameters that can be set within the dialog deserve further explanation:

### Include CipherShed Volume Creation Wizard

Check this option, if you need to create new CipherShed volumes using CipherShed run from the traveler disk you will create. Unchecking this option saves space on the traveler disk.

### AutoRun Configuration (autorun.inf)

In this section, you can configure the ‘traveler disk’ to automatically start CipherShed or mount a specified CipherShed volume when the ‘traveler disk’ is inserted. This is accomplished by creating a special script file called ‘autorun.inf’ on the traveler disk. This file is automatically executed by the operating system each time the ‘traveler disk’ is inserted.

Note, however, that this feature only works for removable storage devices such as CD/DVD (Windows XP SP2, Windows Vista, or a later version of Windows is required for this feature to work on USB memory sticks) and only when it is enabled in the operating system. Depending on the operating system configuration, these auto-run and auto-mount features may work only when the traveler disk files are created on a non-writable CD/DVD-like medium (which is not a bug in CipherShed but a limitation of Windows).

Also note that the ‘autorun.inf’ file must be in the root directory (i.e., for example G:\, X:\, or Y:\ etc.) of an unencrypted disk in order for this feature to work.

## Language Packs

Language packs contain third-party translations of the CipherShed user interface texts. Some language packs also contain translated CipherShed User Guide. Note that language packs are currently supported only by the Windows version of CipherShed.

### Installation

To install a language pack, follow these steps:

- 1 Download a language pack from: <https://ciphershed.org>
- 2 Extract the language pack to the folder to which you installed CipherShed, i.e. the folder in which the file ‘CipherShed.exe’ resides; for example, ‘C:\Program Files\CipherShed’.
- 3 Run CipherShed.
- 4 Select Settings -> Language, then select your language and click OK.

To revert to English, select Settings -> Language. Then select English and click OK.

# Encryption Algorithms

CipherShed volumes can be encrypted using the following algorithms:

| Algorithm           | Designer(s)   | Key Size (Bits) | Block Size (Bits) | Mode of Operation |
|---------------------|---|-----------------|-------------------|-------------------|
| AES                 | J. Daemen, V. Rijmen  | 256             | 128               | XTS               |
| Serpent             | R. Anderson, E. Biham, L. Knudsen                                   | 256             | 128               | XTS               |
| Twofish             | B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Ferguson | 256             | 128               | XTS               |
| AES-Twofish         |   | 256; 256        | 128               | XTS               |
| AES-Twofish-Serpent |   | 256; 256; 256   | 128               | XTS               |
| Serpent-AES         |   | 256; 256        | 128               | XTS               |
| Serpent-Twofish-AES |   | 256; 256; 256   | 128               | XTS               |
| Twofish-Serpent     |   | 256; 256        | 128               | XTS               |

For information about XTS mode, please see the section Modes of Operation.

## AES

The Advanced Encryption Standard (AES) specifies a FIPS-approved cryptographic algorithm (Rijndael, designed by Joan Daemen and Vincent Rijmen, published in 1998) that may be used by US federal departments and agencies to cryptographically protect sensitive information [3]. CipherShed uses AES with 14 rounds and a 256-bit key (i.e., AES-256, published in 2001) operating in XTS mode (see the section Modes of Operation).

In June 2003, after the NSA (US National Security Agency) conducted a review and analysis of AES, the U.S. CNSS (Committee on National Security Systems) announced in [1] that the design and strength of AES-256 (and AES-192) are sufficient to protect classified information up to the Top Secret level. This is applicable to all U.S. Government Departments or Agencies that are considering the acquisition or use of products incorporating the Advanced Encryption Standard (AES) to satisfy Information Assurance requirements associated with the protection of national security systems and/or national security information [1].

## Serpent

Designed by Ross Anderson, Eli Biham, and Lars Knudsen; published in 1998. It uses a 256-bit key, 128-bit block, and operates in XTS mode (see the section Modes of Operation). Serpent was one of the AES finalists. It was not selected as the proposed AES algorithm even though it appeared to have a higher security margin than the winning Rijndael [4]. More concretely, Serpent appeared to have a high security margin, while Rijndael appeared to have only an adequate security margin [4]. Rijndael has also received some criticism suggesting that its mathematical structure might lead to attacks in the future [4].

In [5], the Twofish team presents a table of safety factors for the AES finalists. Safety factor is defined as: number of rounds of the full cipher divided by the largest number of rounds that has been broken. Hence, a broken cipher has the lowest safety factor 1. Serpent had the highest safety factor of the AES finalists: 3.56 (for all supported key sizes). Rijndael-256 had a safety factor of 1.56.

In spite of these facts, Rijndael was considered an appropriate selection for the AES for its combination of security, performance, efficiency, implementability, and flexibility [4]. At the last AES Candidate Conference, Rijndael got 86 votes, Serpent got 59 votes, Twofish got 31 votes, RC6 got 23 votes, and MARS got 13 votes [18][19]<sup>15</sup>.

## Twofish

Designed by Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall, and Niels Ferguson; published in 1998. It uses a 256-bit key and 128-bit block and operates in XTS mode (see the section Modes of Operation). Twofish was one of the AES finalists. This cipher uses key-dependent S-boxes. Twofish may be viewed as a collection of 2128 different cryptosystems, where 128 bits derived from a 256-bit key control the selection of the cryptosystem [4]. In [13], the Twofish team asserts that key-dependent S-boxes constitute a form of security margin against unknown attacks [4].

## AES-Twofish

Two ciphers in a cascade [15][16] operating in XTS mode (see the section Modes of Operation). Each 128-bit block is first encrypted with Twofish (256-bit key) in XTS mode and then with AES (256-bit key) in XTS mode. Each of the cascaded ciphers uses its own key. All encryption keys are mutually independent (note that header keys are independent too, even though they are derived from a single password – see Header Key Derivation, Salt, and Iteration Count). See above for information on the individual cascaded ciphers.

## AES-Twofish-Serpent

Three ciphers in a cascade [15][16] operating in XTS mode (see the section Modes of Operation). Each 128-bit block is first encrypted with Serpent (256-bit key) in XTS mode, then with Twofish (256-bit key) in XTS mode, and finally with AES (256-bit key) in XTS mode. Each of the cascaded ciphers uses its own key. All encryption keys are mutually independent (note that header keys are independent too, even though they are derived from a single password – see the section Header Key Derivation, Salt, and Iteration Count). See above for information on the individual cascaded ciphers.

## Serpent-AES

Two ciphers in a cascade [15][16] operating in XTS mode (see the section Modes of Operation). Each 128-bit block is first encrypted with AES (256-bit key) in XTS mode and then with Serpent (256-bit key) in XTS mode. Each of the cascaded ciphers uses its own key. All encryption keys are mutually independent (note that header keys are independent too, even though they are derived from a single password – see the section Header Key Derivation, Salt, and Iteration Count). See above for information on the individual cascaded ciphers.

---

<sup>15</sup> These are positive votes. If negative votes are subtracted from the positive votes, the following results are obtained: Rijndael: 76 votes, Serpent: 52 votes, Twofish: 10 votes, RC6: -14 votes, MARS: -70 votes [19].

## **Serpent-Twofish-AES**

Three ciphers in a cascade [15][16] operating in XTS mode (see the section Modes of Operation). Each 128-bit block is first encrypted with AES (256-bit key) in XTS mode, then with Twofish (256-bit key) in XTS mode, and finally with Serpent (256-bit key) in XTS mode. Each of the cascaded ciphers uses its own key. All encryption keys are mutually independent (note that header keys are independent too, even though they are derived from a single password – see the section Header Key Derivation, Salt, and Iteration Count). See above for information on the individual cascaded ciphers.

## **Twofish-Serpent**

Two ciphers in a cascade [15][16] operating in XTS mode (see the section Modes of Operation). Each 128-bit block is first encrypted with Serpent (256-bit key) in XTS mode and then with Twofish (256-bit key) in XTS mode. Each of the cascaded ciphers uses its own key. All encryption keys are mutually independent (note that header keys are independent too, even though they are derived from a single password – see the section Header Key Derivation, Salt, and Iteration Count). See above for information on the individual cascaded ciphers.

## Hash Algorithms

In the Volume Creation Wizard, in the password change dialog window, and in the Keyfile Generator dialog window, you can select a hash algorithm. A user-selected hash algorithm is used by the CipherShed Random Number Generator as a pseudorandom “mixing” function, and by the header key derivation function (HMAC based on a hash function, as specified in PKCS #5 v2.0) as a pseudorandom function. When creating a new volume, the Random Number Generator generates the master key, secondary key (XTS mode), and salt. For more information, please see the section Random Number Generator and section Header Key Derivation, Salt, and Iteration Count.

### RIPEMD-160

RIPEMD-160, published in 1996, is a hash algorithm designed by Hans Dobbertin, Antoon Bosselaers, and Bart Preneel in an open academic community. The size of the output of RIPEMD-160 is 160 bits. RIPEMD-160 is a strengthened version of the RIPEMD hash algorithm that was developed in the framework of the European Union’s project RIPE (RACE Integrity Primitives Evaluation), 1988-1992. RIPEMD-160 was adopted by the International Organization for Standardization (ISO) and the IEC in the ISO/IEC 10118-3:2004 international standard [21].

### SHA-512

SHA-512 is a hash algorithm designed by the NSA and published by NIST in FIPS PUB 180-2 [14] in 2002 (the first draft was published in 2001). The size of the output of this algorithm is 512 bits.

### Whirlpool

The Whirlpool hash algorithm was designed by Vincent Rijmen (co-designer of the AES encryption algorithm) and Paulo S. L. M. Barreto. The size of the output of this algorithm is 512 bits. The first version of Whirlpool, now called Whirlpool-0, was published in November 2000. The second version, now called Whirlpool-T, was selected for the NESSIE (New European Schemes for Signatures, Integrity and Encryption) portfolio of cryptographic primitives (a project organized by the European Union, similar to the AES competition). CipherShed uses the third (final) version of Whirlpool, which was adopted by the International Organization for Standardization (ISO) and the IEC in the ISO/IEC 10118-3:2004 international standard [21].

## **Supported Operating Systems**

Note: After this version of CipherShed was released, a new version of an operating system may have been released, verified to be fully compatible with CipherShed and added to the list of supported systems. Therefore, if this is the latest stable version of CipherShed, you should check the online version of this chapter at <https://ciphershed.org>.

This version of CipherShed supports the following operating systems:

- Windows 7 (32-bit and 64-bit)
- Windows Vista
- Windows Vista x64 (64-bit) Edition
- Windows XP
- Windows XP x64 (64-bit) Edition
- Windows Server 2008 R2 (64-bit)
- Windows Server 2008
- Windows Server 2008 x64 (64-bit)
- Windows Server 2003
- Windows Server 2003 x64 (64-bit)
- Windows 2000 SP4
- Mac OS X 10.7 Lion (64-bit and 32-bit)
- Mac OS X 10.6 Snow Leopard (32-bit)
- Mac OS X 10.5 Leopard
- Mac OS X 10.4 Tiger
- Linux (32-bit and 64-bit versions, kernel 2.6 or compatible)

Note: The following operating systems (among others) are not supported: Windows 2003 IA-64, Windows 2008 IA-64, Windows XP IA-64, and the Embedded/Tablet versions of Windows.

See also the section Operating Systems Supported for System Encryption.

## Command Line Usage

Note that this section applies to the Windows version of CipherShed. For information on command line usage applying to the Linux and Mac OS X versions, please run: ciphershed -h

| Option          | Description  |
|-----------------|--|
| /help or /?     | Display command line help.   |
| /volume or /v   | File and path name of a CipherShed volume to mount (do not use when dismounting). To mount a partition/device-hosted volume, use, for example, /v \\Device\Harddisk1\Partition3 (to determine the path to a partition/device, run CipherShed and click Select Device). You can also mount a partition or dynamic volume using its volume name (for example, /v \\?\Volume{5cceb196-48bf-46ab-ad00-70965512253a}\). To determine the volume name use e.g. mountvol.exe. Also note that device paths are case-sensitive. |
| /letter or /l   | Driver letter to mount the volume as. When /l is omitted and when /a is used, the first free drive letter is used.   |
| /explore or /e  | Open an Explorer window after a volume has been mounted.   |
| /beep or /b     | Beep after a volume has been successfully mounted or dismounted.   |
| /auto or /a     | If no parameter is specified, automatically mount the volume. If devices is specified as the parameter (e.g., /a devices), auto-mount all currently accessible device/partition-hosted CipherShed volumes. If favorites is specified as the parameter, auto-mount favorite volumes. Note that /auto is implicit if /quit and /volume are specified. If you need to prevent the application window from appearing, use /quit.   |
| /dismount or /d | Dismount volume specified by drive letter (e.g., /d x). When no drive letter is specified, dismounts all currently mounted CipherShed volumes.   |
| /force or /f    | Forces dismount (if the volume to be dismounted contains files being used by the system or an application) and forces mounting in shared mode (i.e., without exclusive access).  |
| /keyfile or /k  | Specifies a keyfile or a keyfile search path. For multiple keyfiles, specify e.g.: /k c:\keyfile1.dat /k d:\KeyfileFolder /k c:\kf2To specify a keyfile stored on a security token or smart card, use the following syntax: token://slot/SLOT_NUMBER/file/FILE_NAME  |
| /tokenlib       | Use the specified PKCS #11 library for security tokens and smart cards.  |
| /cache or /c    | y or no parameter: enable password cache; n: disable password cache (e.g., /c n). Note that turning the password cache off will not clear it (use /w to clear the password cache).   |
| /history or /h  | y or no parameter: enables saving history of mounted volumes; n: disables saving history of mounted volumes (e.g., /h n).  |

| Option             | Description   |
|--------------------|---|
| /wipecache or /w   | Wipes any passwords cached in the driver memory.  |
| /password or /p    | <p>The volume password. If the password contains spaces, it must be enclosed in quotation marks (e.g.,<br/> <code>/p "My Password"</code>)</p> <p>). Use<br/> <code>/p ""</code><br/> to specify an empty password.</p> <p>Warning: This method of entering a volume password may be insecure, for example, when an unencrypted command prompt history log is being saved to unencrypted disk.</p>  |
| /quit or /q        | <p>Automatically perform requested actions and exit (main CipherShed window will not be displayed). If <code>preferences</code> is specified as the parameter (e.g.,<br/> <code>/q preferences</code>), then program settings are loaded/saved and they overrule settings specified on the command line.</p> <p><code>/q background</code> launches the CipherShed Background Task (tray icon) unless it is disabled in the <code>P references</code>.</p> <p>.</p> |
| /silent or /s      | <p>If <code>/q</code> is specified, suppresses interaction with the user (prompts, error messages, warnings, etc.). If <code>/q</code> is not specified, this option has no effect.</p>   |
| /mountoption or /m | <p><code>ro</code><br/> or<br/> <code>readonly</code><br/> : Mount volume as read<br/> -<br/> only.</p>   |

| Option | Description  |
|--------|--|
|        | <p>rm<br/> or<br/> removable<br/> : Mount volume as removable medium<br/> (s<br/> ee section<br/> Volume Mounted as Remov<br/> able Medium<br/> )<br/> .<br/> ts<br/> or<br/> timestamp<br/> :<br/> Do not preserve container modification timestamp<br/> .<br/> sm<br/> or<br/> system<br/> : Without pre<br/> -<br/> boot authentication, mount a partition that is<br/> within the key scope of system encryption (for example, a partition located<br/> on the encrypted system<br/> drive of another operating system that is not<br/> running). Useful e.g. for backup or repair operations.<br/> Note: If you supply a password as a parameter of<br/> /p<br/> , make sure that the<br/> password has been typed using the standard US keyboard layout<br/> (in<br/> contrast, the G<br/> UI ensures this automatically).<br/> This is required due to the<br/> fact that the password needs to be typed in the pre<br/> -<br/> boot environment<br/> (before Windows starts) where non<br/> -<br/> US Windows keyboard layouts are not<br/> available.<br/> bk<br/> or<br/> headerbak<br/> : Mount volume using embedded<br/> backup header.<br/> Note: All volumes created by TrueCrypt 6.0 or later contain an embedded<br/> backup header (located at the end of the volume).<br/> recovery<br/> : Do not verify any checksums stored in the volume header. This </p> |

| Option | Description  |
|--------|--|
|        | <p>option should be used only when the volume header is damaged and the volume cannot be mounted even with the mount option <code>headerbak</code></p> <p>.</p> <p><b>Example:</b></p> <pre>/m ro</pre> <p>To specify multiple mount options, use e.g.:</p> <pre>/m rm /m ts</pre> |

### **CipherShed Format.exe (CipherShed Volume Creation Wizard):**

|  |   |
|--|---|
| <code>/noisocheck</code> or<br><code>/n</code> | Do not verify that CipherShed Rescue Disks are correctly burned. <b>WARNING:</b> Never attempt to use this option to facilitate the reuse of a previously created CipherShed Rescue Disk. Note that every time you encrypt a system partition/drive, you must create a new CipherShed Rescue Disk even if you use the same password. A previously created CipherShed Rescue Disk cannot be reused as it was created for a different master key. |
|--|---|

## **Syntax**

```
CipherShed.exe [/a [devices/favorites]] [/b] [/c [y/n]] [/d [drive letter]]
[/-e] [/-f][/-h [y/n]] [/-k keyfile or search path] [/-l drive letter] [/-m
{bk/rm/recovery/ro/sm/ts}] [/-p password] [/-q [background/preferences]] [/-s]
[/-tokenlib path] [/-v volume] [/-w]
```

"*CipherShed Format.exe*" [/-n]

Note that the order in which options are specified does not matter.

## **Examples**

Mount the volume d:\myvolume as the first free drive letter, using the password prompt (the main program window will not be displayed):

```
ciphersched /q /v d:\myvolume
```

Dismount a volume mounted as the drive letter X (the main program window will not be displayed):

```
ciphersched /q /dx
```

Mount a volume called myvolume.tc using the password MyPassword, as the drive letter X. CipherShed will open an explorer window and beep; mounting will be automatic:

```
ciphersched /v myvolume.tc /lx /a /p MyPassword /e /b
```

## Security Model

Note to security researchers: If you intend to report a security issue or publish an attack on CipherShed, please make sure it does not disregard the security model of CipherShed described below. If it does, the attack (or security issue report) will be considered invalid/bogus.

CipherShed is a computer software program whose primary purposes are to:

- Secure data by encrypting it before it is written to a disk.
- Decrypt encrypted data after it is read from the disk.

CipherShed does not:

- Encrypt or secure any portion of RAM (the main memory of a computer).
- Secure any data on a computer<sup>16</sup> if an attacker has administrator privileges<sup>17</sup> under an operating system installed on the computer.
- Secure any data on a computer if the computer contains any malware (e.g. a virus, Trojan horse, spyware) or any other piece of software (including CipherShed or an operating system component) that has been altered, created, or can be controlled, by an attacker.
- Secure any data on a computer if an attacker has physical access to the computer before or while CipherShed is running on it.
- Secure any data on a computer if an attacker has physical access to the computer between the time when CipherShed is shut down and the time when the entire contents of all volatile memory modules connected to the computer (including memory modules in peripheral devices) have been permanently and irreversibly erased/lost.
- Secure any data on a computer if an attacker can remotely intercept emanations from the computer hardware (e.g. the monitor or cables) while CipherShed is running on it (or otherwise remotely monitor the hardware and its use, directly or indirectly, while CipherShed is running on it).
- Secure any data stored in a CipherShed volume<sup>18</sup> if an attacker without administrator privileges can access the contents of the mounted volume (e.g. if file/folder/volume permissions do not prevent such an attacker from accessing it).

---

<sup>16</sup> In this section (Security Model), the phrase “data on a computer” means data on internal and external storage devices/media (including removable devices and network drives) connected to the computer.

<sup>17</sup> In this section (Security Model), the phrase “administrator privileges” does not necessarily refer to a valid administrator account. It may also refer to an attacker who does not have a valid administrator account but who is able (for example, due to improper configuration of the system or by exploiting a vulnerability in the operating system or a third-party application) to perform any action that only a user with a valid administrator account is normally allowed to perform (for example, to read or modify an arbitrary part of a drive or the RAM, etc.)

<sup>18</sup> “TrueCrypt volume” also means a TrueCrypt-encrypted system partition/drive (see the chapter System Encryption).

- Preserve/verify the integrity or authenticity of encrypted or decrypted data.
- Prevent traffic analysis when encrypted data is transmitted over a network.
- Prevent an attacker from determining in which sectors of the volume the content changed (and when and how many times) if he or she can observe the volume (dismounted or mounted) before and after data is written to it, or if the storage medium/device allows the attacker to determine such information (for example, the volume resides on a device that saves metadata that can be used to determine when data was written to a particular sector).
- Encrypt any existing unencrypted data in place (or re-encrypt or erase data) on devices/filesystems that use wear-leveling or otherwise relocate data internally.
- Ensure that users choose cryptographically strong passwords or keyfiles.
- Secure any computer hardware component or a whole computer.
- Secure any data on a computer where the security requirements or precautions listed in the chapter Security Requirements and Precautions are not followed.
- Do anything listed in the section Limitations (chapter Known Issues & Limitations).

Under Windows, a user without administrator privileges can (assuming the default CipherShed and operating system configurations):

- Mount any file-hosted CipherShed volume provided that the file permissions of the container allow it.
- Mount any partition/device-hosted CipherShed volume.
- Complete the pre-boot authentication process and, thus, gain access to data on an encrypted system partition/drive (and start the encrypted operating system).
- Skip the pre-boot authentication process (this can be prevented by disabling the option Settings > ‘System Encryption’> ‘Allow pre-boot authentication to be bypassed by pressing the Esc key’; note that this option can be enabled or disabled only by an administrator).

- Dismount, using CipherShed, (and, in the CipherShed application window, see the path to and properties of) any CipherShed volume mounted by him or her. However, this does not apply to ‘system favorite volumes’, which he or she can dismount (etc.) regardless of who mounted them (this can be prevented by enabling the option Settings > ‘System Favorite Volumes’> ‘Allow only administrators to view and dismount system favorite volumes in CipherShed’; note that this option can be enabled or disabled only by an administrator).
- Create a file-hosted CipherShed volume containing a FAT or no file system (provided that the relevant folder permissions allow it).
- Change the password, keyfiles, and header key derivation algorithm for, and restore or back up the header of, a file-hosted CipherShed volume (provided that the file permissions allow it).

- Access the filesystem residing within a CipherShed volume mounted by another user on the system (however, file/folder/volume permissions can be set to prevent this).
- Use passwords (and processed keyfiles) stored in the password cache (note that caching can be disabled; for more information see the section **Settings -> Preferences**, subsection **Cache passwords in driver memory**).
- View the basic properties (e.g. the size of the encrypted area, encryption and hash algorithms used, etc.) of the encrypted system partition/drive when the encrypted system is running.
- Run and use the CipherShed application (including the CipherShed Volume Creation Wizard) provided that the CipherShed device driver is running and that the file permissions allow it.

Under Linux, a user without administrator privileges can (assuming the default CipherShed and operating system configurations):

- Create a file-hosted or partition/device-hosted CipherShed volume containing a FAT or no file system provided that the relevant folder/device permissions allow it.
- Change the password, keyfiles, and header key derivation algorithm for, and restore or back up the header of, a file-hosted or partition/device-hosted CipherShed volume provided that the file/device permissions allow it.
- Access the filesystem residing within a CipherShed volume mounted by another user on the system (however, file/folder/volume permissions can be set to prevent this).
- Run and use the CipherShed application (including the CipherShed Volume Creation Wizard) provided that file permissions allow it.
- In the CipherShed application window, see the path to and properties of any CipherShed volume mounted by him or her.

Under Mac OS X, a user without administrator privileges can (assuming the default CipherShed and operating system configurations):

- Mount any file-hosted or partition/device-hosted CipherShed volume provided that the file/device permissions allow it.
- Dismount, using CipherShed, (and, in the CipherShed application window, see the path to and properties of) any CipherShed volume mounted by him or her.
- Create a file-hosted or partition/device-hosted CipherShed volume provided that the relevant folder/device permissions allow it.
- Change the password, keyfiles, and header key derivation algorithm for, and restore or back up the header of, a file-hosted or partition/device-hosted CipherShed volume (provided that the file/device permissions allow it).
- Access the filesystem residing within a CipherShed volume mounted by another user on the system (however, file/folder/volume permissions can be set to prevent this).

- Run and use the CipherShed application (including the CipherShed Volume Creation Wizard) provided that the file permissions allow it.

CipherShed does not support the set-euid root mode of execution.

Additional information and details regarding the security model are contained in the chapter Security Requirements and Precautions.

# Security Requirements and Precautions

IMPORTANT: If you want to use CipherShed, you must follow the security requirements and security precautions listed in this chapter.

The sections in this chapter specify security requirements for using CipherShed and give information about things that adversely affect or limit the ability of CipherShed to secure data and to provide plausible deniability. Disclaimer: This chapter is not guaranteed to contain a list of all security issues and attacks that might adversely affect or limit the ability of CipherShed to secure data and to provide plausible deniability.

## Data Leaks

When a CipherShed volume is mounted, the operating system and third-party applications may write to unencrypted volumes (typically, to the unencrypted system volume) unencrypted information about the data stored in the CipherShed volume (e.g. filenames and locations of recently accessed files, databases created by file indexing tools, etc.), or the data itself in an unencrypted form (temporary files, etc.), or unencrypted information about the filesystem residing in the CipherShed volume. Note that Windows automatically records large amounts of potentially sensitive data, such as the names and locations of files you open, applications you run, etc.

In order to prevent data leaks, you must follow these steps (alternative steps may exist):

- If you do not need plausible deniability:
  - Encrypt the system partition/drive (for information on how to do so, see the chapter System Encryption) and ensure that only encrypted or read-only filesystems are mounted during each session in which you work with sensitive data.

or,

  - If you cannot do the above, download or create a "live CD" version of your operating system (i.e. a "live" system entirely stored on and booted from a CD/DVD) that ensures that any data written to the system volume is written to a RAM disk. When you need to work with sensitive data, boot such a live CD/DVD and ensure that only encrypted and/or read-only filesystems are mounted during the session.
- If you need plausible deniability:
  - Create a hidden operating system. CipherShed will provide automatic data leak protection. For more information, see the section Hidden Operating System.

or,

- If you cannot do the above, download or create a "live CD" version of your operating system (i.e. a "live" system entirely stored on and booted from a CD/DVD) that ensures that any data written to the system volume is written to a RAM disk. When you need to work with sensitive data, boot such a live CD/DVD. If you use hidden volumes, follow the security requirements and precautions listed in the subsection Security Requirements and Precautions Pertaining to Hidden Volumes. If you do not use hidden volumes, ensure that only non-system partition-hosted CipherShed volumes and/or read-only filesystems are mounted during the session.

## Paging File

Note: The issue described below does not affect you if the system partition or system drive is encrypted (for more information, see the chapter System Encryption) and if all paging files are located on one or more of the partitions within the key scope of system, for example, on the partition where Windows is installed (for more information, see the fourth paragraph in this subsection).

Paging files, also called swap files, are used by Windows to hold parts of programs and data files that do not fit in memory. This means that sensitive data, which you believe are only stored in RAM, can actually be written unencrypted to a hard drive by Windows without you knowing.

Note that CipherShed cannot prevent the contents of sensitive files that are opened in RAM from being saved unencrypted to a paging file (note that when you open a file stored on a CipherShed volume, for example, in a text editor, then the content of the file is stored unencrypted in RAM).

To prevent the issues described above, encrypt the system partition/drive (for information on how to do so, see the chapter System Encryption) and make sure that all paging files are located on one or more of the partitions within the key scope of system encryption (for example, on the partition where Windows is installed). Note that the last condition is typically met on Windows XP by default. However, Windows Vista and later versions of Windows are configured by default to create paging files on any suitable volume. Therefore, before you start using CipherShed, you must follow these steps: Right-click the 'Computer' (or 'My Computer') icon on the desktop or in the Start Menu, and then select Properties -> (on Windows Vista or later: -> Advanced System Settings ->) Advanced tab -> section Performance -> Settings -> Advanced tab -> section Virtual memory -> Change. On Windows Vista or later, disable 'Automatically manage paging file size for all drives'. Then make sure that the list of volumes available for paging file creation contains only volumes within the intended key scope of system encryption (for example, the volume where Windows is installed). To disable paging file creation on a particular volume, select it, then select 'No paging file' and click Set. When done, click OK and restart the computer.

Note: You may also want to consider creating a hidden operating system (for more information, see the section Hidden Operating System).

## Memory Dump Files

Note: The issue described below does not affect you if the system partition or system drive is encrypted (for more information, see the chapter System Encryption) and if the system is configured to write memory dump files to the system drive (which it typically is, by default).

Most operating systems, including Windows, can be configured to write debugging information and contents of the system memory to so-called memory dump files (also called crash dump files) when an error occurs (system crash, "blue screen," bug check). Therefore, memory dump files may contain sensitive data. CipherShed cannot prevent cached passwords, encryption keys, and the contents of sensitive files opened in RAM from being saved unencrypted to memory dump files. Note that when you open a file stored on a CipherShed volume, for example, in a text editor, then the content of the file is stored unencrypted in RAM (and it may remain unencrypted in RAM until the computer is turned off). Also note that when a CipherShed volume is mounted, its master key is stored unencrypted in RAM. Therefore, you must disable memory dump file generation on your computer at least for each session during which you work with any sensitive data and during which

you mount a CipherShed volume. To do so in Windows XP or later, right-click the 'Computer' (or 'My Computer') icon on the desktop or in the Start Menu, and then select Properties -> (on Windows Vista or later: -> Advanced System Settings ->) Advanced tab -> section Startup and Recovery -> Settings -> section Write debugging information -> select (none) -> OK.

Note for users of Windows XP/2003: As Windows XP and Windows 2003 do not provide any API for encryption of memory dump files, if the system partition/drive is encrypted by CipherShed and your Windows XP system is configured to write memory dump files to the system drive, the CipherShed driver automatically prevents Windows from writing any data to memory dump files.

## Hibernation File

Note: The issue described below does not affect you if the system partition or system drive is encrypted<sup>19\*</sup> (for more information, see the chapter System Encryption) and if the hibernation file is located on one the partitions within the key scope of system encryption (which it typically is, by default), for example, on the partition where Windows is installed. When the computer hibernates, data are encrypted on the fly before they are written to the hibernation file.

When a computer hibernates (or enters a power-saving mode), the content of its system memory is written to a so-called hibernation file on the hard drive. You can configure CipherShed (Settings > Preferences > Dismount all when: Entering power saving mode) to automatically dismount all mounted CipherShed volumes, erase their master keys stored in RAM, and cached passwords (stored in RAM), if there are any, before a computer hibernates (or enters a power-saving mode). However, keep in mind, that if you do not use system encryption (see the chapter System Encryption), CipherShed still cannot reliably prevent the contents of sensitive files opened in RAM from being saved unencrypted to a hibernation file. Note that when you open a file stored on a CipherShed volume, for example, in a text editor, then the content of the file is stored unencrypted in RAM (and it may remain unencrypted in RAM until the computer is turned off).

---

<sup>19</sup> Disclaimer: As Windows XP and Windows 2003 do not provide any API for encryption of hibernation files, TrueCrypt has to modify undocumented components of Windows XP/2003 in order to allow users to encrypt hibernation files. Therefore, TrueCrypt cannot guarantee that Windows XP/2003 hibernation files will always be encrypted. In response to our public complaint regarding the missing API, Microsoft began providing a public API for encryption of hibernation files on Windows Vista and later versions of Windows (for more information, see the Version History, section TrueCrypt 5.1a). Since version 7.0, TrueCrypt has used this API and therefore has been able to safely encrypt hibernation files under Windows Vista and later versions of Windows. Therefore, if you use Windows XP/2003 and want the hibernation file to be safely encrypted, we strongly recommend that you upgrade to Windows Vista or later and to TrueCrypt 7.0 or later.

Note that when Windows enters Sleep mode, it may be actually configured to enter so-called Hybrid Sleep mode, which involves hibernation. Also note that the operating system may be configured to hibernate or enter the Hybrid Sleep mode when you click or select "Shut down" (for more information, please see the documentation for your operating system).

To prevent the issues described above, encrypt the system partition/drive (for information on how to do so, see the chapter System Encryption) and make sure that the hibernation file is located on one of the partitions within the key scope of system encryption (which it typically is, by default), for example, on the partition where Windows is installed. When the computer hibernates, data will be encrypted on the fly before they are written to the hibernation file.

Note: You may also want to consider creating a hidden operating system (for more information, see the section Hidden Operating System).

Alternatively, if you cannot use system encryption, disable or prevent hibernation on your computer at least for each session during which you work with any sensitive data and during which you mount a CipherShed volume.

## **Unencrypted Data in RAM**

It is important to note that CipherShed is a disk encryption software, which encrypts only disks, not RAM (memory).

Keep in mind that most programs do not clear the memory area (buffers) in which they store unencrypted (portions of) files they load from a CipherShed volume. This means that after you exit such a program, unencrypted data it worked with may remain in memory (RAM) until the computer is turned off (and, according to some researchers, even for some time after the power is turned off<sup>20</sup>). Also note that if you open a file stored on a CipherShed volume, for example, in a text editor and then force dismount on the CipherShed volume, then the file will remain unencrypted in the area of memory (RAM) used by (allocated to) the text editor. This also applies to forced auto-dismount.

---

<sup>20</sup> Allegedly, for 1.5–35 seconds under normal operating temperatures (26–44 °C) and up to several hours when the memory modules are cooled (when the computer is running) to very low temperatures (e.g. –50 °C). New types of memory modules allegedly exhibit a much shorter decay time (e.g. 1.5–2.5 seconds) than older types (as of 2008).

Inherently, unencrypted master keys have to be stored in RAM too. When a non-system CipherShed volume is dismounted, CipherShed erases its master keys (stored in RAM). When the computer is cleanly restarted (or cleanly shut down), all non-system CipherShed volumes are automatically dismounted and, thus, all master keys stored in RAM are erased by the CipherShed driver (except master keys for system partitions/drives — see below). However, when power supply is abruptly interrupted, when the computer is reset (not cleanly restarted), or when the system crashes, CipherShed naturally stops running and therefore cannot erase any keys or any other sensitive data. Furthermore, as Microsoft does not provide any appropriate API for handling hibernation and shutdown, master keys used for system encryption cannot be reliably (and are not) erased from RAM when a computer hibernates, is shut down or restarted<sup>21</sup>.

To summarize, CipherShed cannot and does not ensure that RAM contains no sensitive data (e.g. passwords, master keys, or decrypted data). Therefore, after each session in which you work with a CipherShed volume or in which an encrypted operating system is running, you must shut down (or, if the hibernation file is encrypted, hibernate) the computer and then leave it powered off for at least several minutes (the longer, the better) before turning it on again. This is required to clear the RAM (see also the section Hibernation File).

## Physical Security

If an attacker can physically access the computer hardware and you use it after the attacker has physically accessed it, then CipherShed may become unable to secure data on the computer<sup>22</sup>. This is because the attacker may modify the hardware or attach a malicious hardware component to it (such as a hardware keystroke logger) that will capture the password or encryption key (e.g., when you mount a CipherShed volume) or otherwise compromise the security of the computer. Therefore, you must not use CipherShed on a computer that an attacker has physically accessed. Furthermore, you must ensure that CipherShed (including its device driver) is not running when the attacker physically accesses the computer. Additional information pertaining to hardware attacks where the attacker has direct physical access is contained in the section Unencrypted Data in RAM.

Furthermore, even if the attacker cannot physically access the computer hardware directly, he or she may be able to breach the physical security of the computer by remotely intercepting and analyzing emanations from the computer hardware (including the monitor and cables). For example, intercepted emanations from the cable connecting the keyboard with the computer can reveal passwords you type. It is beyond the scope of this document to list all of the kinds of such attacks (sometimes called TEMPEST attacks) and all known ways to prevent them (such as shielding or radio jamming). You must prevent such attacks. It is solely your responsibility to do so. If you do not, CipherShed may become unable to secure data on the computer.

---

<sup>21</sup> Before a key can be erased from RAM, the corresponding TrueCrypt volume must be dismounted. For non-system volumes, this does not cause any problems. However, as Microsoft currently does not provide any appropriate API for handling the final phase of the system shutdown process, paging files located on encrypted system volumes that are dismounted during the system shutdown process may still contain valid swapped-out memory pages (including portions of Windows system files). This could cause 'blue screen' errors. Therefore, to prevent 'blue screen' errors, TrueCrypt does not dismount encrypted system volumes and consequently cannot clear the master keys of the system volumes when the system is shut down or restarted.

<sup>22</sup> In this section (Physical Security), the phrase "data on the computer" means data on internal and external storage devices/media (including removable devices and network drives) connected to the computer.

## Malware

The term ‘malware’ refers collectively to all types of malicious software, such as computer viruses, Trojan horses, spyware, or generally any piece of software (including CipherShed or an operating system component) that has been altered, prepared, or can be controlled, by an attacker. Some kinds of malware are designed e.g. to log keystrokes, including typed passwords (such captured passwords are then either sent to the attacker over the Internet or saved to an unencrypted local drive from which the attacker might be able to read it later, when he or she gains physical access to the computer). If you use CipherShed on a computer infected with any kind of malware, CipherShed may become unable to secure data on the computer<sup>23</sup>. Therefore, you must not use CipherShed on such a computer.

It is important to note that CipherShed is encryption software, not anti-malware software. It is your responsibility to prevent malware from running on the computer. If you do not, CipherShed may become unable to secure data on the computer.

There are many rules that you should follow to help prevent malware from running on your computer. Among the most important rules are the following: Keep your operating system, Internet browser, and other critical software, up-to-date. In Windows XP or later, turn on DEP for all programs<sup>24</sup>. Do not open suspicious email attachments, especially executable files, even if they appear to have been sent by your relatives or friends (their computers might be infected with malware sending malicious emails from their computers/accounts without their knowledge). Do not follow suspicious links contained in emails or on websites (even if the email/website appears to be harmless or trustworthy). Do not visit any suspicious websites. Do not download or install any suspicious software. Consider using good, trustworthy, anti-malware software.

## Multi-User Environment

Keep in mind, that the content of a mounted CipherShed volume is visible (accessible) to all logged on users. NTFS file/folder permissions can be set to prevent this, unless the volume is mounted as removable medium (see section Volume Mounted as Removable Medium) under a desktop edition of Windows Vista or later (sectors of a volume mounted as removable medium may be accessible at the volume level to users without administrator privileges, regardless of whether it is accessible to them at the file-system level).

Moreover, on Windows, the password cache is shared by all logged on users (for more information, please see the section Settings -> Preferences, subsection Cache passwords in driver memory).

Also note that switching users in Windows XP or later (Fast User Switching functionality) does not dismount a successfully mounted CipherShed volume (unlike system restart, which dismounts all mounted CipherShed volumes).

---

<sup>23</sup> In this section (Malware), the phrase “data on the computer” means data on internal and external storage devices/media (including removable devices and network drives) connected to the computer.

<sup>24</sup>DEP stands for Data Execution Prevention. For more information about DEP, please visit <http://support.microsoft.com/kb/875352>, <http://technet.microsoft.com/en-us/library/cc700810.aspx>, and <http://windows.microsoft.com/en-US/windows-vista/What-is-Data-Execution-Prevention>.

On Windows 2000, the container file permissions are ignored when a file-hosted CipherShed volume is to be mounted. On all supported versions of Windows, users without administrator privileges can mount any partition/device-hosted CipherShed volume (provided that they supply the correct password and/or keyfiles). A user without administrator privileges can dismount only volumes that he or she mounted. However, this does not apply to system favorite volumes unless you enable the option (disabled by default) Settings > ‘System Favorite Volumes’>‘Allow only administrators to view and dismount system favorite volumes in CipherShed’.

## **Authenticity and Integrity**

CipherShed uses encryption to preserve the confidentiality of data it encrypts. CipherShed neither preserves nor verifies the integrity or authenticity of data it encrypts or decrypts. Hence, if you allow an adversary to modify data encrypted by CipherShed, he can set the value of any 16-byte block of the data to a random value or to a previous value, which he was able to obtain in the past. Note that the adversary cannot choose the value that you will obtain when CipherShed decrypts the modified block — the value will be random — unless the attacker restores an older version of the encrypted block, which he was able to obtain in the past. It is your responsibility to verify the integrity and authenticity of data encrypted or decrypted by CipherShed (for example, by using appropriate third-party software).

See also: Physical Security, Security Model

## **Choosing Passwords and Keyfiles**

It is very important that you choose a good password. You must avoid choosing one that contains only a single word that can be found in a dictionary (or a combination of such words). It must not contain any names, dates of birth, account numbers, or any other items that could be easy to guess. A good password is a random combination of upper and lower case letters, numbers, and special characters, such as @ ^ = \$ \* + etc. We strongly recommend choosing a password consisting of more than 20 characters (the longer, the better). Short passwords are easy to crack using brute-force techniques.

To make brute-force attacks on a keyfile infeasible, the size of the keyfile must be at least 30 bytes. If a volume uses multiple keyfiles, then at least one of the keyfiles must be 30 bytes in size or larger. Note that the 30-byte limit assumes a large amount of entropy in the keyfile. If the first 1024 kilobytes of a file contain only a small amount of entropy, it must not be used as a keyfile (regardless of the file size). If you are not sure what entropy means, we recommend that you let CipherShed generate a file with random content and that you use it as a keyfile (select Tools -> Keyfile Generator).

When creating a volume, encrypting a system partition/drive, or changing passwords/keyfiles, you must not allow any third party to choose or modify the password/keyfile(s) before/while the volume is created or the password/keyfiles(s) changed. For example, you must not use any password generators (whether website applications or locally run programs) where you are not sure that they are high-quality and uncontrolled by an attacker, and keyfiles must not be files that you download from the internet or that are accessible to other users of the computer (whether they are administrators or not).

## **Changing Passwords and Keyfiles**

Note that the volume header (which is encrypted with a header key derived from a password/keyfile) contains the master key (not to be confused with the password) with which the volume is encrypted. If an adversary is allowed to make a copy of your volume before you change the volume password and/or keyfile(s), he may be able to use his copy or fragment (the old header) of the CipherShed volume to mount your volume using a compromised password and/or compromised keyfiles that were necessary to mount the volume before you changed the volume password and/or keyfile(s).

If you are not sure whether an adversary knows your password (or has your keyfiles) and whether he has a copy of your volume when you need to change its password and/or keyfiles, it is strongly recommended that you create a new CipherShed volume and move files from the old volume to the new volume (the new volume will have a different master key).

Also note that if an adversary knows your password (or has your keyfiles) and has access to your volume, he may be able to retrieve and keep its master key. If he does, he may be able to decrypt your volume even after you change its password and/or keyfile(s) (because the master key does not change when you change the volume password and/or keyfiles). In such a case, create a new CipherShed volume and move all files from the old volume to this new one.

The following sections of this chapter contain additional information pertaining to possible security issues connected with changing passwords and/or keyfiles:

- Security Requirements and Precautions
- Journaling File Systems
- Defragmenting
- Reallocated Sectors

## Trim Operation

Some storage devices (e.g., some solid-state drives, including USB flash drives) use so-called ‘trim’ operation to mark drive sectors as free e.g. when a file is deleted. Consequently, such sectors may contain unencrypted zeroes or other undefined data (unencrypted) even if they are located within a part of the drive that is encrypted by CipherShed. CipherShed does not block the trim operation on partitions that are within the key scope of system encryption (see chapter System Encryption) (unless a hidden operating system is running – see section Hidden Operating System) and under Linux on all volumes that use the Linux native kernel cryptographic services. In those cases, the adversary will be able to tell which sectors contain free space (and may be able to use this information for further analysis and attacks) and plausible deniability (see chapter Plausible Deniability) may be negatively affected. If you want to avoid those issues, do not use system encryption on drives that use the trim operation and, under Linux, either configure CipherShed not to use the Linux native kernel cryptographic services or make sure CipherShed volumes are not located on drives that use the trim operation.

To find out whether a device uses the trim operation, please refer to documentation supplied with the device or contact the vendor/manufacturer.

## **Wear-Leveling**

Some storage devices (e.g., some solid-state drives, including USB flash drives) and some file systems utilize so-called wear-leveling mechanisms to extend the lifetime of the storage device or medium. These mechanisms ensure that even if an application repeatedly writes data to the same logical sector, the data is distributed evenly across the medium (logical sectors are remapped to different physical sectors). Therefore, multiple "versions" of a single sector may be available to an attacker. This may have various security implications. For instance, when you change a volume password/keyfile(s), the volume header is, under normal conditions, overwritten with a re-encrypted version of the header. However, when the volume resides on a device that utilizes a wear-leveling mechanism, CipherShed cannot ensure that the older header is really overwritten. If an adversary found the old volume header (which was to be overwritten) on the device, he could use it to mount the volume using an old compromised password (and/or using compromised keyfiles that were necessary to mount the volume before the volume header was re-encrypted). Due to security reasons, we recommend that CipherShed volumes are not created/stored on devices (or in file systems) that utilize a wear-leveling mechanism (and that CipherShed is not used to encrypt any portions of such devices or filesystems).

If you decide not to follow this recommendation and you intend to use in-place encryption on a drive that utilizes wear-leveling mechanisms, make sure the partition/drive does not contain any sensitive data before you fully encrypt it (CipherShed cannot reliably perform secure in-place encryption of existing data on such a drive; however, after the partition/drive has been fully encrypted, any new data that will be saved to it will be reliably encrypted on the fly). That includes the following precautions: Before you run CipherShed to set up pre-boot authentication, disable the paging files and restart the operating system (you can enable the paging files after the system partition/drive has been fully encrypted). Hibernation must be prevented during the period between the moment when you start CipherShed to set up pre-boot authentication and the moment when the system partition/drive has been fully encrypted. However, note that even if you follow those steps, it is not guaranteed that you will prevent data leaks and that sensitive data on the device will be securely encrypted. For more information, see the sections Data Leaks, Paging File, Hibernation File, and Hibernation File.

If you need plausible deniability, you must not use CipherShed to encrypt any part of (or create encrypted containers on) a device (or file system) that utilizes a wear-leveling mechanism.

To find out whether a device utilizes a wear-leveling mechanism, please refer to documentation supplied with the device or contact the vendor/manufacturer.

## **Reallocated Sectors**

Some storage devices, such as hard drives, internally reallocate/remap bad sectors. Whenever the device detects a sector to which data cannot be written, it marks the sector as bad and remaps it to a sector in a hidden reserved area on the drive. Any subsequent read/write operations from/to the bad sector are redirected to the sector in the reserved area. This means that any existing data in the bad sector remains on the drive and it cannot be erased (overwritten with other data). This may have various security implications. For instance, data that is to be encrypted in place may remain unencrypted in the bad sector. Likewise, data to be erased (for example, during the process of creation of a hidden operating system) may remain in the bad sector. Plausible deniability (see section Plausible Deniability) may be adversely affected whenever a sector is reallocated. Additional examples of possible security implications are listed in the section Security Requirements and Precautions. Please note that this list is not exhaustive (these are just examples). Also note that CipherShed cannot prevent any security issues related to or caused by reallocated sectors. To find out the number of reallocated sectors on a hard drive, you can use e.g. a third-party software tool for reading so-called S.M.A.R.T. data.

## Defragmenting

When you (or the operating system) defragment the file system in which a file-hosted CipherShed container is stored, a copy of the CipherShed container (or of its fragment) may remain in the free space on the host volume (in the defragmented file system). This may have various security implications. For example, if you change the volume password/keyfile(s) afterwards, and an adversary finds the old copy or fragment (the old header) of the CipherShed volume, he might use it to mount the volume using an old compromised password (and/or using compromised keyfiles that were necessary to mount the volume before the volume header was re-encrypted). To prevent this and other possible security issues (such as those mentioned in the section Volume Clones), do one of the following:

- Use a partition/device-hosted CipherShed volume instead of file-hosted.
- Securely erase free space on the host volume (in the defragmented file system) after defragmenting.
- Do not defragment file systems in which you store CipherShed volumes.

## Journaling File Systems

When a file-hosted CipherShed container is stored in a journaling file system (such as NTFS), a copy of the CipherShed container (or of its fragment) may remain in the free space on the host volume. This may have various security implications. For example, if you change the volume password/keyfile(s) and an adversary finds the old copy or fragment (the old header) of the CipherShed volume, he might use it to mount the volume using an old compromised password (and/or using compromised keyfiles using an old compromised password (and/or using compromised keyfiles that were necessary to mount the volume before the volume header was re-encrypted). Some journaling file systems also internally record file access times and other potentially sensitive information. If you need plausible deniability (see section Plausible Deniability), you must not store file-hosted CipherShed containers in journaling file systems. To prevent possible security issues related to journaling file systems, do one of the following:

- Use a partition/device-hosted CipherShed volume instead of file-hosted.
- Store the container in a non-journaling file system (for example, FAT32).

## Volume Clones

Never create a new CipherShed volume by cloning an existing CipherShed volume. Always use the CipherShed Volume Creation Wizard to create a new CipherShed volume. If you clone a volume and then start using both this volume and its clone in a way that both eventually contain different data, then you might aid cryptanalysis (both volumes will share a single key set). This is especially critical when the volume contains a hidden volume. Also note that plausible deniability (see section Plausible Deniability) is impossible in such cases. See also the chapter How to Back Up Securely.

## Additional Security Requirements and Precautions

In addition to the requirements and precautions described in this chapter (Security Requirements and Precautions), you must follow and keep in mind the security requirements, precautions, and limitations listed in the following chapters and sections:

- How to Back Up Securely
- Limitations

- Security Model
- Security Requirements and Precautions Pertaining to Hidden Volumes
- Plausible Deniability

See also: Digital Signatures

# How to Back Up Securely

Due to hardware or software errors/malfunctions, files stored on a CipherShed volume may become corrupted. Therefore, we strongly recommend that you backup all your important files regularly (this, of course, applies to any important data, not just to encrypted data stored on CipherShed volumes).

## Non-System Volumes

To back up a non-system CipherShed volume securely, it is recommended to follow these steps:

1. Create a new CipherShed volume using the CipherShed Volume Creation Wizard (do not enable the Quick Format option or the Dynamic option). It will be your backup volume so its size should match (or be greater than) the size of your main volume.

If the main volume is a hidden CipherShed volume (see the section Hidden Volume), the backup volume must be a hidden CipherShed volume too. Before you create the hidden backup volume, you must create a new host (outer) volume for it without enabling the Quick Format option. In addition, especially if the backup volume is file-hosted, the hidden backup volume should occupy only a very small portion of the container and the outer volume should be almost completely filled with files (otherwise, the plausible deniability of the hidden volume might be adversely affected).

2. Mount the newly created backup volume.
3. Mount the main volume.
4. Copy all files from the mounted main volume directly to the mounted backup volume.

**IMPORTANT:** If you store the backup volume in any location that an adversary can repeatedly access (for example, on a device kept in a bank's safe deposit box), you should repeat all of the above steps (including the step 1) each time you want to back up the volume (see below).

If you follow the above steps, you will help prevent adversaries from finding out:

- Which sectors of the volumes are changing (because you always follow step 1). This is particularly important, for example, if you store the backup volume on a device kept in a bank's safe deposit box (or in any other location that an adversary can repeatedly access) and the volume contains a hidden volume (for more information, see the subsection Security Requirements and Precautions Pertaining to Hidden Volumes in the chapter Plausible Deniability).
- That one of the volumes is a backup of the other.

## System Partitions

Note: In addition to backing up files, we recommend that you also back up your CipherShed Rescue Disk (select System > Create Rescue Disk). For more information, see the section CipherShed Rescue Disk.

To back up an encrypted system partition securely and safely, it is recommended to follow these steps:

1. If you have multiple operating systems installed on your computer, boot the one that does not require pre-boot authentication.

If you do not have multiple operating systems installed on your computer, you can boot a WinPE or BartPE CD/DVD ('live' Windows entirely stored on and booted from a CD/DVD; for more information, search the section Frequently Asked Questions for the keyword 'BartPE').

If none of the above is possible, connect your system drive as a secondary drive to another computer and then boot the operating system installed on the computer.

Note: For security reasons, if the operating system that you want to back up resides in a hidden CipherShed volume (see the section Hidden Operating System), then the operating system that you boot in this step must be either another hidden operating system or a "live-CD" operating system (see above). For more information, see the subsection Security Requirements and Precautions Pertaining to Hidden Volumes in the chapter Plausible Deniability.

2. Create a new non-system CipherShed volume using the CipherShed Volume Creation Wizard (do not enable the Quick Format option or the Dynamic option). It will be your backup volume so its size should match (or be greater than) the size of the system partition that you want to back up.

If the operating system that you want to back up is installed in a hidden CipherShed volume (see the section Hidden Operating System), the backup volume must be a hidden CipherShed volume too. Before you create the hidden backup volume, you must create a new host (outer) volume for it without enabling the Quick Format option. In addition, especially if the backup volume is file-hosted, the hidden backup volume should occupy only a very small portion of the container and the outer volume should be almost completely filled with files (otherwise, the plausible deniability of the hidden volume might be adversely affected).

3. Mount the newly created backup volume.

4. Mount the system partition that you want to back up by following these steps:

- a. Click Select Device and then select the system partition that you want to back up (in case of a hidden operating system, select the partition containing the hidden volume in which the operating system is installed).
- b. Click OK.
- c. Select System > Mount Without Pre-Boot Authentication.
- d. Enter your pre-boot authentication password and click OK.

5. Mount the backup volume and then use a third-party program or a Windows tool to create an image of the filesystem that resides on the system partition (which was mounted as a regular CipherShed volume in the previous step) and store the image directly on the mounted backup volume.

**IMPORTANT:** If you store the backup volume in any location that an adversary can repeatedly access (for example, on a device kept in a bank's safe deposit box), you should repeat all of the above steps (including the step 2) each time you want to back up the volume (see below).

If you follow the above steps, you will help prevent adversaries from finding out:

- Which sectors of the volumes are changing (because you always follow step 2). This is particularly important, for example, if you store the backup volume on a device kept in a bank's safe deposit box (or in any other location that an adversary can repeatedly access) and the volume contains a hidden volume (for more information, see the subsection Security Requirements and Precautions Pertaining to Hidden Volumes in the chapter Plausible Deniability).
- That one of the volumes is a backup of the other.

## General Notes

If you store the backup volume in any location where an adversary can make a copy of the volume, consider encrypting the volume with a cascade of ciphers (for example, with AES-Twofish-Serpent). Otherwise, if the volume is encrypted only with a single encryption algorithm and the algorithm is later broken (for example, due to advances in cryptanalysis), the attacker might be able to decrypt his copies of the volume. The probability that three distinct encryption algorithms will be broken is significantly lower than the probability that only one of them will be broken.

## Miscellaneous

### Using CipherShed Without Administrator Privileges

In Windows, a user who does not have administrator privileges can use CipherShed, but only after a system administrator installs CipherShed on the system. The reason for that is that CipherShed needs a device driver to provide transparent on-the-fly encryption/decryption, and users without administrator privileges cannot install/start device drivers in Windows.

After a system administrator installs CipherShed on the system, users without administrator privileges will be able to run CipherShed, mount/dismount any type of CipherShed volume, load/save data from/to it, and create file-hosted CipherShed volumes on the system. However, users without administrator privileges cannot encrypt/format partitions, cannot create NTFS volumes, cannot install/uninstall CipherShed, cannot change passwords/keyfiles for CipherShed partitions/devices, cannot backup/restore headers of CipherShed partitions/devices, and they cannot run CipherShed in ‘portable’ mode.

Warning: No matter what kind of software you use, as regards personal privacy in most cases, it is not safe to work with sensitive data under systems where you do not have administrator privileges, as the administrator can easily capture and copy your sensitive data, including passwords and keys.

### Sharing over Network

If there is a need to access a single CipherShed volume simultaneously from multiple operating systems, there are two options:

1. A CipherShed volume is mounted only on a single computer (for example, on a server) and only the content of the mounted CipherShed volume (i.e., the file system within the CipherShed volume) is shared over a network. Users on other computers or systems will not mount the volume (it is already mounted on the server).

Advantages: All users can write data to the CipherShed volume. The shared volume may be both file-hosted and partition/device-hosted.

Disadvantage: Data sent over the network will not be encrypted. However, it is still

possible to encrypt them using e.g. SSL, TLS, VPN, or other technologies.

Remarks: Note that, when you restart the system, the network share will be automatically restored only if the volume is a system favorite volume or an encrypted system partition/drive (for information on how to configure a volume as a system favorite volume, see the chapter System Favorite Volumes).

2. A dismounted CipherShed file container is stored on a single computer (for example, on a server). This encrypted file is shared over a network. Users on other computers or systems will locally mount the shared file. Thus, the volume will be mounted simultaneously under multiple operating systems.

Advantage: Data sent over the network will be encrypted (however, it is still recommended to encrypt them using e.g. SSL, TLS, VPN, or other appropriate technologies to make traffic analysis more difficult and to preserve the integrity of the data).

**Disadvantages:** The shared volume may be only file-hosted (not partition/device-hosted). The volume must be mounted in read-only mode under each of the systems (see the section Mount Options for information on how to mount a volume in read-only mode). Note that this requirement applies to unencrypted volumes too. One of the reasons is, for example, the fact that data read from a conventional file system under one OS while the file system is being modified by another OS might be inconsistent (which could result in data corruption).

## CipherShed Background Task

When the main CipherShed window is closed, the CipherShed Background Task takes care of the following tasks/functions:

1. Hot keys
2. Auto-dismount (e.g., upon logoff, inadvertent host device removal, time-out, etc.)
3. Auto-mount of favorite volumes
4. Notifications (e.g., when damage to hidden volume is prevented)
5. Tray icon

**WARNING:** If neither the CipherShed Background Task nor CipherShed is running, the above-mentioned tasks/functions are disabled.

The CipherShed Background Task is actually the CipherShed.exe application, which continues running in the background after you close the main CipherShed window. Whether it is running or not can be determined by looking at the system tray area. If you can see the CipherShed icon there, then the CipherShed Background Task is running. You can click the icon to open the main CipherShed window. Right-click on the icon opens a popup menu with various CipherShed-related functions.

You can shut down the Background Task at any time by right-clicking the CipherShed tray icon and selecting Exit. If you need to disable the CipherShed Background Task completely and permanently, select Settings -> Preferences and uncheck the option Enabled in the CipherShed Background Task area of the Preferences dialog window.

## Volume Mounted as Removable Medium

This section applies to CipherShed volumes mounted when one of the following options is enabled (as applicable):

- Tools > Preferences > Mount volumes as removable media
- Mount Options > Mount volume as removable medium
- Favorites > Organize Favorite Volumes > Mount selected volume as removable medium
- Favorites > Organize System Favorite Volumes > Mount selected volume as removable medium

CipherShed Volumes that are mounted as removable media have the following advantages and disadvantages:

- Windows is prevented from automatically creating the ‘Recycled’ and/or the ‘System Volume Information’ folders on CipherShed volumes (in Windows, these folders are used by the Recycle Bin and System Restore features).
- Windows may use caching methods and write delays that are normally used for removable media (for example, USB flash drives). This might slightly decrease the performance but at the same increase the likelihood that it will be possible to dismount the volume quickly without having to force the dismount.
- The operating system may tend to keep the number of handles it opens to such a volume to a minimum. Hence, volumes mounted as removable media might require fewer forced dismounts than other volumes.
- Under Windows Vista and earlier, the ‘Computer’ (or ‘My Computer’) list does not show the amount of free space on volumes mounted as removable (note that this is a Windows limitation, not a bug in CipherShed).
- Under desktop editions of Windows Vista or later, sectors of a volume mounted as removable medium may be accessible to all users (including users without administrator privileges; see section Multi-User Environment).

## **CipherShed System Files & Application Data**

Note: %windir% is the main Windows installation path (e.g., C:\WINDOWS)

### **CipherShed Driver**

%windir%\SYSTEM32\DRIVERS\truecrypt.sys

Note: This file is not present when CipherShed is run in portable mode.

### **CipherShed Settings, Application Data, and Other System Files**

**WARNING:** Note that CipherShed does not encrypt any of the files listed in this section (unless it encrypts the system partition/drive).

The following files are saved in the folder %APPDATA%\CipherShed\. In portable mode, these files are saved to the folder from which you run the file CipherShed.exe (i.e., the folder in which CipherShed.exe resides):

Configuration.xml (the main configuration file).

System Encryption.xml (temporary configuration file used during the initial process of in-place encryption/decryption of the system partition/drive).

Default Keyfiles.xml

Note: This file may be absent if the corresponding CipherShed feature is not used.

Favorite Volumes.xml

Note: This file may be absent if the corresponding CipherShed feature is not used.

History.xml (the list of last twenty files/devices attempted to be mounted as CipherShed volumes or attempted to be used as hosts for CipherShed volumes; this feature can be disabled – for more information, see the section Never Save History)

Note: This file may be absent if the corresponding CipherShed feature is not used.

#### In-Place EncryptionIn-Place Encryption Wipe Algo

(temporary configuration files used during the initial process of in-place encryption/decryption of a non-system volume).

#### Post-Install Task - Tutorial Post-Install Task -Release Notes

(temporary configuration files used during the process of installation or upgrade of CipherShed).

The following files are saved in the folder %ALLUSERSPROFILE%\CipherShed\:

Original System Loader (a backup of the original content of the first drive track made before the CipherShed Boot Loader was written to it). Note: This file is absent if the system partition/drive has not been encrypted.

The following files are saved in the folder %windir%\system32 (32-bit systems) or %windir%\SysWOW64 (64-bit systems):

#### CipherShed System Favorite Volumes.xml

Note: This file may be absent if the corresponding CipherShed feature is not used.

#### CipherShed.exe

Note: A copy of this file is located in this folder only when mounting of system favorite volumes is enabled.

## How to Remove Encryption

Please note that CipherShed can in-place decrypt only system partitions and system drives (select System > Permanently Decrypt System Partition/Drive). If you need to remove encryption (e.g., if you no longer need encryption) from a non-system volume, please follow these steps:

- 1 Mount the CipherShed volume.
- 2 Move all files from the CipherShed volume to any location outside the CipherShed volume (note that the files will be decrypted on the fly).
- 3 Dismount the CipherShed volume.
- 4 If the CipherShed volume is file-hosted, delete it (the container) just like you delete any other file.

If the volume is partition-hosted (applies also to USB flash drives), in addition to the steps 1-3, do the following:

- a. Right-click the ‘Computer’ (or ‘My Computer’) icon on your desktop, or in the Start Menu, and select Manage. The ‘Computer Management’ window should appear.
- b. In the Computer Management window, from the list on the left, select ‘Disk Management’ (within the Storage sub-tree).
- c. Right-click the partition you want to decrypt and select ‘Change Drive Letter and Paths’.

- d. The ‘Change Drive Letter and Paths’ window should appear. If no drive letter is displayed in the window, click Add. Otherwise, click Cancel. If you clicked Add, then in the ‘Add Drive Letter or Path’ (which should have appeared), select a drive letter you want to assign to the partition and click OK.
- e. In the Computer Management window, right-click the partition you want to decrypt again and select Format. The Format window should appear.
- f. In the Format window, click OK. After the partition is formatted, it will no longer be required to mount it with CipherShed to be able to save or load files to/from the partition.

If the volume is device-hosted (i.e., there are no partitions on the device, and the device is entirely encrypted), in addition to the steps 1-3, do the following:

- a. Right-click the ‘Computer’ (or ‘My Computer’) icon on your desktop, or in the Start Menu, and select Manage. The ‘Computer Management’ window should appear.
- b. In the Computer Management window, from the list on the left, select ‘Disk Management’ (within the Storage sub-tree).
- c. The ‘Initialize Disk’ window should appear. Use it to initialize the disk.
- d. In the ‘Computer Management’ window, right-click the area representing the storage space of the encrypted device and select ‘New Partition’ or ‘New Simple Volume’.
- e. WARNING: Before you continue, make sure you have selected the correct device, as all files stored on it will be lost. The ‘New Partition Wizard’ or ‘New Simple Volume Wizard’ window should appear now; follow its instructions to create a new partition on the device. After the partition is created, it will no longer be required to mount the device with CipherShed to be able to save or load files to/from the device.

## **Uninstalling CipherShed**

To uninstall CipherShed on Windows XP, select Start menu > Settings > Control Panel > Add or Remove Programs > CipherShed > Change/Remove.

To uninstall CipherShed on Windows Vista or later, select Start menu > Computer > Uninstall or change a program > CipherShed > Uninstall.

No CipherShed volume will be removed when you uninstall CipherShed. You will be able to mount your CipherShed volume(s) again after you install CipherShed or when you run it in portable mode.

## **Digital Signatures**

### **Why Verify Digital Signatures**

It might happen that a CipherShed installation package you download from our server was created or modified by an attacker. For example, the attacker could exploit a vulnerability in the server software we use and alter the installation packages stored on the server, or he/she could alter any of the files en route to you.

Therefore, you should always verify the integrity and authenticity of each CipherShed distribution package you download or otherwise obtain from any source. In other words, you should always make sure that the file was created by us and it was not altered by an attacker. One way to do so is to verify so-called digital signature(s) of the file.

## **Types of Digital Signatures We Use**

We currently use two types of digital signatures:

- PGP signatures (available for all binary and source code packages for all supported systems).
- X.509 signatures (available for binary packages for Windows).

## **Advantages of X.509 Signatures**

X.509 signatures have the following advantages, in comparison to PGP signatures:

- It is much easier to verify that the key that signed the file is really ours (not attacker's).
- You do not have to download or install any extra software to verify an X.509 signature (see below).
- You do not have to download and import our public key (it is embedded in the signed file).
- You do not have to download any separate signature file (the signature is embedded in the signed file).

## **Advantages of PGP Signatures**

PGP signatures have the following advantages, in comparison to X.509 signatures:

- They do not depend on any certificate authority (which might be e.g. infiltrated or controlled by an adversary, or be untrustworthy for other reasons).

## **How to Verify X.509 Signatures**

Please note that X.509 signatures are currently available only for the CipherShed self-extracting installation packages for Windows. An X.509 digital signature is embedded in each of those files along with the digital certificate of the CipherShed Foundation issued by a public certification authority. To verify the integrity and authenticity of a self-extracting installation package for Windows, follow these steps:

1. Download the CipherShed self-extracting installation package.
2. In the Windows Explorer, click the downloaded file ('CipherShed Setup.exe') with the right mouse button and select 'Properties' from the context menu.
3. In the Properties dialog window, select the 'Digital Signatures' tab.
4. On the 'Digital Signatures' tab, in the 'Signature list', double click the line saying "CipherShed Foundation".
5. The 'Digital Signature Details' dialog window should appear now. If you see the following sentence at the top of the dialog window, then the integrity and authenticity of the package have been successfully verified:

"This digital signature is OK."

If you do not see the above sentence, the file is very likely corrupted. Note: On some obsolete versions of Windows, some of the necessary certificates are missing, which causes the signature verification to fail.

## **How to Verify PGP Signatures**

To verify a PGP signature, follow these steps:

1. Install any public-key encryption software that supports PGP signatures. Links to such software may be found at <https://ciphershed.org>.
2. Create a private key (for information on how to do so, please see the documentation for the public-key encryption software).
3. Download our PGP public key from our server or from a trusted public key repository, and import the downloaded key to your keyring (for information on how to do so, please see the documentation for the public-key encryption software).
4. Sign the imported key with your private key to mark it as trusted (for information on how to do so, please see the documentation for the public-key encryption software). Note: If you skip this step and attempt to verify any of our PGP signatures, you will receive an error message stating that the signing key is invalid.
5. Download the digital signature by clicking the PGP Signature button next to the file you want to verify (on one of the download pages).
6. Verify the downloaded signature (for information on how to do so, please see the documentation for the public-key encryption software).

# Troubleshooting

It is recommended that you read also the latest online version of this chapter at:

<https://ciphershed.org>

This section presents possible solutions to common problems that you may run into when using CipherShed.

Note: If your problem is not listed here, it might be listed in one of the following sections:

- Incompatibilities
- Known Issues & Limitations
- Frequently Asked Questions

Make sure you use the latest stable version of CipherShed. If the problem is caused by a bug in an old version of CipherShed, it may have already been fixed. Note: Select Help > About to find out which version you use.

## **PROBLEM:**

Writing/reading to/from volume is very slow even though, according to the benchmark, the speed of the cipher that I'm using is higher than the speed of the hard drive.

## **PROBABLE CAUSE:**

This is probably caused by an interfering application.

## **POSSIBLE SOLUTION:**

First, make sure that your CipherShed container does not have a file extension that is reserved for executable files (for example, .exe, .sys, or .dll). If it does, Windows and antivirus software may interfere with the container and adversely affect the performance of the volume.

Second, disable or uninstall any application that might be interfering, which usually is antivirus software or automatic disk defragmentation tool, etc. In case of antivirus software, it often helps to turn off real-time (on-access) scanning in the preferences of the antivirus software. If it does not help, try temporarily disabling the virus protection software. If this does not help either, try uninstalling it completely and restarting your computer subsequently.

## **PROBLEM:**

A CipherShed volume cannot be mounted; CipherShed reports “Incorrect password or not a CipherShed volume”.

## **POSSIBLE CAUSE:**

The volume header may have been damaged by a third-party application or malfunctioning hardware component.

## **POSSIBLE SOLUTIONS:**

- If you created your volume using TrueCrypt 6.0 or later, you can try to restore the volume header from the backup embedded in the volume by following these steps:
  1. Run CipherShed (TrueCrypt 6.0 or later).
  2. Click Select Device or Select File to select your volume. 3) Select Tools > Restore Volume Header.

3. If you created your volume using TrueCrypt 5.1a or earlier, you can try to mount your volume with the command line option /m recovery as follows:
  - Install CipherShed (TrueCrypt 6.1 or later).
    1. On your keyboard, press and hold the Windows key and then press R. The Windows Run dialog should appear.
    2. Type in the following command (replace the last argument, \Device\Harddisk1\Partition0, with the path to your volume and if CipherShed is not installed in %ProgramFiles%, replace %ProgramFiles% with the path to CipherShed):
 

```
"%ProgramFiles%\CipherShed\CipherShed.exe" /q /m recovery /v \Device\Harddisk1\Partition0
```
    3. Press Enter to try to mount your volume.

### **PROBLEM:**

After successfully mounting a volume, Windows reports "This device does not contain a valid file system" or a similar error.

### **PROBABLE CAUSE:**

The file system on the CipherShed volume may be corrupted (or the volume is unformatted).

### **POSSIBLE SOLUTION:**

You can use filesystem repair tools supplied with your operating system to attempt to repair the filesystem on the CipherShed volume. In Windows, it is the 'chkdsk' tool. CipherShed provides an easy way to use this tool on a CipherShed volume: First, make a backup copy of the CipherShed volume (because the 'chkdsk' tool might damage the filesystem even more) and then mount it. Right-click the mounted volume in the main CipherShed window (in the drive list) and from the context menu select 'Repair Filesystem'.

### **PROBLEM:**

When trying to create a hidden volume, its maximum possible size is unexpectedly small (there is much more free space than this on the outer volume).

### **PROBABLE CAUSES:**

4. The outer volume has been formatted as NTFS
5. Fragmentation
6. Too small cluster size + too many files/folders in the root directory of the outer volume.

### **POSSIBLE SOLUTIONS:**

Solutions Related to Cause 1:

Unlike the FAT filesystem, the NTFS filesystem always stores internal data exactly in the middle of the volume. Therefore, the hidden volume can reside only in the second half of the outer volume. If this constraint is unacceptable, do one of the following:

- Reformat the outer volume as FAT and then create a hidden volume within it.

- If the outer volume is too large to be formatted as FAT, split the volume to several 2-terabyte volumes (or 16-terabyte volumes if the device uses 4-kilobyte sectors) and format each of them as FAT.

Solution Related to Cause 2:

Create a new outer volume (defragmentation is not a solution, because it would adversely affect plausible deniability – see section Defragmenting).

Solution Related to Cause 3:

Note: The following solution applies only to hidden volumes created within FAT volumes.

Defragment the outer volume (mount it, right-click its drive letter in the ‘Computer’ or ‘My Computer’ window, click Properties, select the Tools tab, and click ‘Defragment Now’). After the volume is defragmented, exit Disk Defragmenter and try to create the hidden volume again.

If this does not help, delete all files and folders on the outer volume by pressing Shift+Delete, not by formatting, (do not forget to disable the Recycle Bin and System Restore for this drive beforehand) and try creating the hidden volume on this completely empty outer volume again (for testing purposes only). If the maximum possible size of the hidden volume does not change even now, the cause of the problem is very likely an extended root directory. If you did not use the ‘Default’ cluster size (the last step in the Wizard), reformat the outer volume and this time leave the cluster size at ‘Default’.

If it does not help, reformat the outer volume again and copy less files/folders to its root folder than you did last time. If it does not help, keep reformatting and decreasing the number of files/folders in the root folder. If this is unacceptable or if it does not help, reformat the outer volume and select a larger cluster size. If it does not help, keep reformatting and increasing the cluster size, until the problem is solved. Alternatively, try creating a hidden volume within an NTFS volume.

## **PROBLEM:**

One of the following problems occurs:

- A CipherShed volume cannot be mounted.
- NTFS CipherShed volumes cannot be created.

In addition, the following error may be reported: "The process cannot access the file because it is being used by another process."

## **PROBABLE CAUSE:**

This is probably caused by an interfering application. Note that this is not a bug in CipherShed. The operating system reports to CipherShed that the device is locked for an exclusive access by an application (so CipherShed is not allowed to access it).

## **POSSIBLE SOLUTION:**

It usually helps to disable or uninstall the interfering application, which is usually an anti-virus utility, a disk management application, etc.

**PROBLEM:**

In the CipherShed Boot Loader screen, I'm trying to type my password and/or pressing other keys but the CipherShed boot loader is not responding.

**PROBABLE CAUSE:**

You have a USB keyboard (not a PS/2 keyboard) and pre-boot support for USB keyboards is disabled in your BIOS settings.

**POSSIBLE SOLUTION:**

You need to enable pre-boot support for USB keyboards in your BIOS settings. To do so, follow the below steps:

Restart your computer, press F2 or Delete (as soon as you see a BIOS start-up screen), and wait until a BIOS configuration screen appears. If no BIOS configuration screen appears, restart (reset) the computer again and start pressing F2 or Delete repeatedly as soon as you restart (reset) the computer. When a BIOS configuration screen appears, enable pre-boot support for USB keyboards. This can typically be done by selecting: Advanced > ‘USB Configuration’> ‘Legacy USB Support’ (or ‘USB Legacy’) > Enabled. (Note that the word ‘legacy’ is in fact misleading, because pre-boot components of modern versions of MS Windows require this option to be enabled to allow user interaction/control.) Then save the BIOS settings (typically by pressing F10) and restart your computer. For more information, please refer to the documentation for your BIOS/motherboard or contact your computer vendor’s technical support team for assistance.

**PROBLEM:**

After the system partition/drive is encrypted, the computer cannot boot after it is restarted (it is also impossible to enter the BIOS configuration screen).

**PROBABLE CAUSE:**

A bug in the BIOS of your computer.

**POSSIBLE SOLUTIONS:**

Follow these steps:

1. Disconnect the encrypted drive.
2. Connect an unencrypted drive with an installed operating system (or install it on the drive).
3. Upgrade the BIOS.
4. If it does not help, please report this bug to the manufacturer or vendor of the computer.

OR

- If the BIOS/motherboard/computer manufacturer/vendor does not provide any updates that resolve the issue and you use Windows 7 or later and there is an extra boot partition (whose size is less than 1 GB) on the drive, you can try reinstalling Windows without this extra boot partition (to work around a bug in the BIOS). For information how to do so, see:

<https://ciphershed.org>

**PROBLEM:**

One of the following problems occurs:

- After the pre-boot authentication password is entered during the system encryption pretest, the computer hangs (after the message ‘Booting...’ is displayed).
- When the system partition/drive is encrypted (partially or fully) and the system is restarted for the first time since the process of encryption of the system partition/drive started, the computer hangs after you enter the pre-boot authentication password (after the message ‘Booting...’ is displayed).
- After the hidden operating system is cloned and the password for it entered, the computer hangs (after the message ‘Booting...’ is displayed).

#### **PROBABLE CAUSE:**

A bug in the BIOS of your computer.

#### **POSSIBLE SOLUTIONS:**

- Upgrade your BIOS (for information on how to do so, please refer to the documentation for your BIOS/motherboard or contact your computer vendor’s technical support team for assistance).
- Use a different motherboard model/brand.
- If the BIOS/motherboard/computer manufacturer/vendor does not provide any updates that resolve the issue and you use Windows 7 or later and there is an extra boot partition (whose size is less than 1 GB) on the drive, you can try reinstalling Windows without this extra boot partition (to work around a bug in the BIOS). For information how to do so, see:

<https://ciphershed.org>

#### **PROBLEM:**

When mounting or dismounting a CipherShed volume, the system crashes (a ‘blue screen’ error screen appears or the computer abruptly restarts).

OR

Since I installed CipherShed, the operating system has been crashing frequently.

#### **POSSIBLE CAUSES:**

- A bug in a third-party application (e.g. antivirus, system "tweaker", etc.)
- A bug in CipherShed
- A bug in Windows or a malfunctioning hardware component

#### **POSSIBLE SOLUTIONS:**

- Try disabling any antivirus tools, system "tweakers", and any other similar applications. If it does not help, try uninstalling them and restarting Windows.

If the problem persists, run CipherShed and select Help > ‘Analyze a System Crash’. CipherShed will then analyze crash dump files that Windows automatically created when it crashed (if any). If CipherShed determines that a bug in a third party driver is likely to have caused the crash, it will show the name and provider of the driver (note that updating or uninstalling the driver might resolve the issue). Whatever the results, you will be able to choose to send us essential information about the system crash to help us determine whether it was caused by a bug in CipherShed.

### **PROBLEM:**

When trying to encrypt the system partition/drive, during the pretest, the CipherShed Boot Loader always reports that the pre-boot authentication password I entered is incorrect (even though I'm sure it is correct).

### **POSSIBLE CAUSES:**

- Different state of the Num Lock and/or Caps Lock key
- Data corruption

### **POSSIBLE SOLUTION:**

1. When you set a pre-boot authentication password, remember whether the Num Lock and Caps Lock keys are on or off (depending on the manufacturer, the keys may have different labels, such as Num LK). Note: You can change the state of each of the keys as desired before you set the password, but you need to remember the states.
2. When you enter the password in the CipherShed Boot Loader screen, make sure the state of each of the keys is the same as when you set the password.

Note: For other possible solutions to this problem, see the other sections of this chapter.

### **PROBLEM:**

When the system partition/drive is encrypted, the operating system ‘freezes’ for approx. 10-60 seconds every 5-60 minutes (100% CPU usage may co-occur).

### **PROBABLE CAUSE:**

A CPU and/or motherboard issue.

### **POSSIBLE SOLUTIONS:**

- Try upgrading the BIOS.
- Try disabling all power-saving-related features (including any special CPU enhanced halt functions) in the BIOS settings and in the ‘Power Options’ Windows control panel.
- Replace the processor with a different one (different type and/or brand).
- Replace the motherboard with a different one (different type and/or brand).

### **PROBLEM:**

On Windows 7/Vista (and possibly later versions), the Microsoft Windows Backup tool cannot be used to backup data to a non-system CipherShed Volume.

### **CAUSE:**

A bug in the Windows Backup tool.

### **POSSIBLE SOLUTION:**

1. Mount the CipherShed volume to which you want to back up data.
2. Right-click a folder located on the volume (or right-click its drive letter in the ‘Computer’ list) and select an item from the ‘Share with’ submenu (on Windows Vista, select ‘Share’).
3. Follow the instructions to share the folder with your user account.
4. In the Windows Backup tool, select the shared folder (the network location/path) as the destination.
5. Start the backup process.

Note: The above solution does not apply to the Starter and Home editions of Windows 7 (and possibly later versions).

### **PROBLEM:**

The label of a filesystem in a CipherShed volume cannot be changed from within the ‘Computer’ window under Windows Vista or a later version of Windows.

### **CAUSE:**

A Windows issue causes the label to be written only to the Windows registry file, instead of being written to the filesystem.

### **POSSIBLE SOLUTIONS:**

- Right-click the mounted volume in the ‘Computer’ window, select Properties, and enter a new label for the volume.

### **PROBLEM:**

I cannot encrypt a partition/device because CipherShed Volume Creation Wizard says it is in use.

### **POSSIBLE SOLUTION:**

Close, disable, or uninstall all programs that might be using the partition/device in any way (for example an anti-virus utility). If it does not help, right-click the ‘Computer’ (or ‘My Computer’) icon on your desktop and select Manage -> Storage -> Disk Management. Then right-click the partition that you want to encrypt, and click Change Drive Letter and Paths. Then click Remove and OK. Restart the operating system.

### **PROBLEM:**

When creating a hidden volume, the Wizard reports that the outer volume cannot be locked.

### **PROBABLE CAUSE:**

The outer volume contains files being used by one or more applications.

### **POSSIBLE SOLUTION:**

Close all applications that are using files on the outer volume. If it does not help, try disabling or uninstalling any anti-virus utility you use and restarting the system subsequently.

**PROBLEM:**

When accessing a file-hosted container shared over a network, "insufficient memory" or "not enough server storage is available" error is reported.

**PROBABLE CAUSE:**

IRPStackSize in the Windows registry may have been set to a too small value.

**POSSIBLE SOLUTION:**

Locate the IRPStackSize key in the Windows registry and set it to a higher value. Then restart the system. If the key does not exist in the Windows registry, create it at

HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters

and set its value to 16 or higher. Then restart the system. For more information, see:  
<http://support.microsoft.com/kb/285089/> and <http://support.microsoft.com/kb/177078/>

## Incompatibilities

It is recommended that you read also the latest online version of this chapter at:

<https://ciphershed.org>

### Activation of Adobe Photoshop® and Other Products Using FLEXnet Publisher® / SafeCast

Note: The issue described below does not affect you if you use CipherShed (TrueCrypt 5.1 or later) and a non-cascade encryption algorithm (i.e., AES, Serpent, or Twofish).<sup>\*</sup> The issue also does not affect you if you do not use pre-boot authentication (see the chapter System Encryption).

Acrecco FLEXnet Publisher activation software, formerly Macrovision SafeCast, (used for activation of third-party software, such as Adobe Photoshop) writes data to the first drive track. If this happens when your system partition/drive is encrypted by CipherShed, a portion of the CipherShed Boot Loader will be damaged and you will not be able to start Windows. In that case, please use your CipherShed Rescue Disk to regain access to your system. There are two ways to do so:

1. You may keep the third-party software activated but you will need to boot your system from the CipherShed Rescue Disk CD/DVD every time. Just insert your Rescue Disk into your CD/DVD drive and then enter your password in the Rescue Disk screen.
2. If you do not want to boot your system from the CipherShed Rescue Disk CD/DVD every time, you can restore the CipherShed Boot Loader on the system drive. To do so, in the Rescue Disk screen, select Repair Options > Restore CipherShed Boot Loader. However, note that this will deactivate the third-party software.

For information on how to use your CipherShed Rescue Disk, please see the chapter CipherShed Rescue Disk.

**Possible permanent solution:** Upgrade to CipherShed (TrueCrypt 5.1 or later), decrypt the system partition/drive, and then re-encrypt it using a non-cascade encryption algorithm (i.e., AES, Serpent, or Twofish)<sup>25</sup>.

Please note that this not a bug in CipherShed (the issue is caused by inappropriate design of the third-party activation software).

---

<sup>25</sup> The reason is that the TrueCrypt Boot Loader is smaller than the one used for cascades of ciphers and, therefore, there is enough space in the first drive track for a backup of the TrueCrypt Boot Loader. Hence, whenever the TrueCrypt Boot Loader is damaged, its backup copy is run automatically instead.

# Known Issues & Limitations

It is strongly recommended that you read also the latest online version of this chapter at:

<https://ciphershed.org>

## Known Issues

- (There were no confirmed issues when this document was created.)

## Limitations

- [Note: This limitation does not apply to users of Windows Vista and later versions of Windows.] On Windows XP/2003, CipherShed does not support encrypting an entire system drive that contains extended (logical) partitions. You can encrypt an entire system drive provided that it contains only primary partitions. Extended (logical) partitions must not be created on any system drive that is partially or fully encrypted (only primary partitions may be created on it). Note: If you need to encrypt an entire drive containing extended partitions, you can encrypt the system partition and, in addition, create partition-hosted CipherShed volumes within any non-system partitions on the drive. Alternatively, you may want to consider upgrading to Windows Vista or a later version of Windows.
- CipherShed currently does not support encrypting a system drive that has been converted to a dynamic disk.
- CipherShed volume passwords must consist only of printable ASCII characters. Other characters in passwords are not supported and may cause various problems (e.g., inability to mount a volume).
- To work around a Windows XP issue, the CipherShed boot loader is always automatically configured for the version of the operating system under which it is installed. When the version of the system changes (for example, the CipherShed boot loader is installed when Windows Vista is running but it is later used to boot Windows XP) you may encounter various known and unknown issues (for example, on some notebooks, Windows XP may fail to display the log-on screen). Note that this affects multi-boot configurations, CipherShed Rescue Disks, and decoy/hidden operating systems (therefore, if the hidden system is e.g. Windows XP, the decoy system should be Windows XP too).
- The ability to mount a partition that is within the key scope of system encryption without pre-boot authentication (for example, a partition located on the encrypted system drive of another operating system that is not running), which can be done e.g. by selecting System > Mount Without Pre-Boot Authentication, is limited to primary partitions (extended/logical partitions cannot be mounted this way).
- Due to a Windows 2000 issue, CipherShed does not support the Windows Mount Manager under Windows 2000. Therefore, some Windows 2000 built-in tools, such as Disk Defragmenter, do not work on CipherShed volumes. Furthermore, it is not possible to use the Mount Manager services under Windows 2000, e.g., assign a mount point to a CipherShed volume (i.e., attach a CipherShed volume to a folder). CipherShed does not support pre-boot authentication for operating systems installed within VHD files, except when booted using appropriate virtual-machine software such as Microsoft Virtual PC.
- The Windows Volume Shadow Copy Service is currently supported only for partitions within the key scope of system encryption (e.g. a system partition encrypted by CipherShed, or a non-system partition located on a system drive encrypted by CipherShed, mounted when the encrypted operating system is running). Note: For other types of volumes, the Volume Shadow Copy Service is not supported because the documentation for the necessary API is not available.

- Windows boot settings cannot be changed from within a hidden operating system if the system does not boot from the partition on which it is installed. This is due to the fact that, for security reasons, the boot partition is mounted as read-only when the hidden system is running. To be able to change the boot settings, please start the decoy operating system.
- Encrypted partitions cannot be resized except partitions on an entirely encrypted system drive that are resized while the encrypted operating system is running.
- When the system partition/drive is encrypted, the system cannot be upgraded (for example, from Windows XP to Windows Vista) or repaired from within the pre-boot environment (using a Windows setup CD/DVD or the Windows pre-boot component). In such cases, the system partition/drive must be decrypted first. Note: A running operating system can be updated (security patches, service packs, etc.) without any problems even when the system partition/drive is encrypted.
- System encryption is supported only on drives that are connected locally via an ATA/SCSI interface (note that the term ATA also refers to SATA and eSATA).
- When system encryption is used (this also applies to hidden operating systems), CipherShed does not support multi-boot configuration changes (for example, changes to the number of operating systems and their locations). Specifically, the configuration must remain the same as it was when the CipherShed Volume Creation Wizard started to prepare the process of encryption of the system partition/drive (or creation of a hidden operating system).

Note: The only exception is the multi-boot configuration where a running CipherShed-encrypted operating system is always located on drive #0, and it is the only operating system located on the drive (or there is one CipherShed-encrypted decoy and one CipherShed-encrypted hidden operating system and no other operating system on the drive), and the drive is connected or disconnected before the computer is turned on (for example, using the power switch on an external eSATA drive enclosure). There may be any additional operating systems (encrypted or unencrypted) installed on other drives connected to the computer (when drive #0 is disconnected, drive #1 becomes drive #0, etc.)

- When the notebook battery power is low, Windows may omit sending the appropriate messages to running applications when the computer is entering power saving mode. Therefore, CipherShed may fail to auto-dismount volumes in such cases.
- Preserving of any timestamp of any file (e.g. a container or keyfile) is not guaranteed to be reliably and securely performed (for example, due to filesystem journals, timestamps of file attributes, or the operating system failing to perform it for various documented and undocumented reasons). Note: When you write to a file-hosted hidden volume, the timestamp of the container may change. This can be plausibly explained as having been caused by changing the (outer) volume password. Also note that CipherShed never preserves timestamps of system favorite volumes (regardless of the settings). Special software (e.g., a low-level disk editor) that writes data to a disk drive in a way that circumvents drivers in the driver stack of the class ‘DiskDrive’ (GUID of the class is 4D36E967-E325-11CE-BFC1-08002BE10318) can write unencrypted data to a non-system drive hosting a mounted CipherShed volume (‘Partition0’) and to encrypted partitions/drives that are within the key scope of active system encryption (CipherShed does not encrypt such data written that way). Similarly, software that writes data to a disk drive circumventing drivers in the driver stack of the class ‘Storage Volume’ (GUID of the class is 71A27CDD-812A-11D0-BEC7-08002BE2092F) can write unencrypted data to CipherShed partition-hosted volumes (even if they are mounted).

- For security reasons, when a hidden operating system is running, CipherShed ensures that all local unencrypted filesystems and non-hidden CipherShed volumes are read-only. However, this does not apply to filesystems on CD/DVD-like media and on custom, atypical, or non-standard devices/media (for example, any devices/media whose class is other than the Windows device class ‘Storage Volume’ or that do not meet the requirements of this class (GUID of the class is 71A27CDD-812A-11D0-BEC7-08002BE2092F)).
- Device-hosted CipherShed volumes located on floppy disks are not supported. Note: You can still create file-hosted CipherShed volumes on floppy disks.
- Further limitations are listed in the section Security Model.

# Frequently Asked Questions

Note: The latest version of the CipherShed FAQ is available at <https://ciphershed.org>.

## I forgot my password – is there any way ('backdoor') to recover the files from my CipherShed volume?

CipherShed does not allow recovery of any encrypted data without knowing the correct password or key. We cannot recover your data because we do not know and cannot determine the password you chose or the key you generated using CipherShed. The only way to recover your files is to try to "crack" the password or the key, but it could take thousands or millions of years (depending on the length and quality of the password or keyfiles, on the software/hardware performance, algorithms, and other factors). If you find this hard to believe, consider the fact that even the FBI was not able to decrypt a CipherShed volume after a year of trying.

## Is there a "Quick Start Guide" or some tutorial for beginners?

Yes. The first chapter, Beginner's Tutorial, contains screenshots and step-by-step instructions on how to create, mount, and use a CipherShed volume.

## Can I encrypt a partition/drive where Windows is installed?

Yes (see the chapter System Encryption).

## Can I directly play a video (.avi, .mpg, etc.) stored on a CipherShed volume?

Yes, CipherShed-encrypted volumes are like normal disks. You provide the correct password (and/or keyfile) and mount (open) the CipherShed volume. When you double click the icon of the video file, the operating system launches the application associated with the file type – typically a media player. The media player then begins loading a small initial portion of the video file from the CipherShed-encrypted volume to RAM (memory) in order to play it. While the portion is being loaded, CipherShed is automatically decrypting it (in RAM). The decrypted portion of the video (stored in RAM) is then played by the media player. While this portion is being played, the media player begins loading another small portion of the video file from the CipherShed-encrypted volume to RAM (memory) and the process repeats.

The same goes for video recording: Before a chunk of a video file is written to a CipherShed volume, CipherShed encrypts it in RAM and then writes it to the disk. This process is called on-the-fly encryption/decryption and it works for all file types (not only for video files).

## Will CipherShed be open-source and free forever?

Yes, it will. We will never create a commercial version of CipherShed, as we believe in open-source and free security software.

## Is it possible to donate to the CipherShed project?

Not at this time.

## Why is CipherShed open-source? What are the advantages?

As the source code for CipherShed is publicly available, independent researchers can verify that the source code does not contain any security flaw or secret 'backdoor'. If the source code were not available, reviewers would need to reverse-engineer the executable files. However, analyzing and understanding such reverse-engineered code is so difficult that it is practically impossible to do (especially when the code is as large as the CipherShed code).

Remark: A similar problem also affects cryptographic hardware. It is very difficult to reverse-engineer it to verify that it does not contain any security flaw or secret 'backdoor'.

## **CipherShed is open-source, but has anybody actually reviewed the source code?**

Yes. In fact, the source code is constantly being reviewed by many independent researchers and users. We know this because many bugs and several security issues have been discovered by independent researchers (including some well-known ones) while reviewing the source code.

As CipherShed is open-source software, independent researchers can verify that the source code does not contain any security flaw or secret ‘backdoor’. Can they also verify that the official executable files were built from the published source code and contain no additional code?

Yes, they can. In addition to reviewing the source code, independent researchers can compile the source code and compare the resulting executable files with the official ones. They may find some differences (for example, timestamps or embedded digital signatures) but they can analyze the differences and verify that they do not form malicious code.

## **How can I use CipherShed on a USB flash drive?**

You have two options:

1. Encrypt the entire USB flash drive. However, you will not be able run CipherShed from the USB flash drive. Note: Windows does not support multiple partitions on USB flash drives.
2. Create a CipherShed file container on the USB flash drive (for information on how to do so, see the chapter Beginner’s Tutorial). If you leave enough space on the USB flash drive (choose an appropriate size for the CipherShed container), you will also be able to store CipherShed on the USB flash drive (along with the container – not in the container) and you will be able to run CipherShed from the USB flash drive (see also the chapter Portable Mode).

## **Does CipherShed also encrypt file names and folder names?**

Yes. The entire file system within a CipherShed volume is encrypted (including file names, folder names, and contents of every file). This applies to both types of CipherShed volumes – i.e., to file containers (virtual CipherShed disks) and to CipherShed-encrypted partitions/devices.

## **Does CipherShed use parallelization?**

Yes. Increase in encryption/decryption speed is directly proportional to the number of cores/processors your computer has. For more information, please see the chapter Parallelization in the documentation.

## **Can data be read from and written to an encrypted volume/drive as fast as if the drive was not encrypted?**

Yes, since CipherShed uses pipelining and parallelization. For more information, please see the chapters Pipelining and Parallelization.

## **Does CipherShed support hardware-accelerated encryption?**

Yes. For more information, please see the chapter Hardware Acceleration.

## **Is it possible to boot Windows installed in a hidden CipherShed volume?**

Yes, it is (as of TrueCrypt 6.0). For more information, please see the section Hidden Operating System.

## **Will I be able to mount my CipherShed volume on any computer?**

Yes, CipherShed volumes are independent of the operating system. You will be able to mount your CipherShed volume on any computer on which you can run CipherShed (see also the question “Can I use CipherShed in Windows if I do not have administrator privileges?”).

## **Can I unplug or turn off a hot-plug device (for example, a USB flash drive or USB hard drive) when there is a mounted CipherShed volume on it?**

Before you unplug or turn off the device, you should always dismount the CipherShed volume in CipherShed first, and then perform the ‘Eject’ operation if available (right-click the device in the ‘Computer’ or ‘My Computer’ list), or use the ‘Safely Remove Hardware’ function (built in Windows, accessible via the taskbar notification area). Otherwise, data loss may occur.

## **What is a hidden operating system?**

See the section Hidden Operating System.

## **What is plausible deniability?**

See the chapter Plausible Deniability.

## **Will I be able to mount my CipherShed partition/container after I reinstall or upgrade the operating system?**

Yes, CipherShed volumes are independent of the operating system. However, you need to make sure your operating system installer does not format the partition where your CipherShed volume resides.

Note: If the system partition/drive is encrypted and you want to reinstall or upgrade Windows, you need to decrypt it first (select System > Permanently Decrypt System Partition/Drive). However, a running operating system can be updated (security patches, service packs, etc.) without any problems even when the system partition/drive is encrypted.

## **Can I upgrade from an older version of CipherShed to the latest version without any problems?**

Generally, yes. However, before upgrading, please read the release notes for all versions of CipherShed that have been released since your version was released. If there are any known issues or incompatibilities related to upgrading from your version to a newer one, they will be listed in the release notes.

## **Can I upgrade CipherShed if the system partition/drive is encrypted or do I have to decrypt it first?**

Generally, you can upgrade to the latest version without decrypting the system partition/drive (just run the CipherShed installer and it will automatically upgrade CipherShed on the system). However, before upgrading, please read the release notes for all versions of CipherShed that have been released since your version was released. If there are any known issues or incompatibilities related to upgrading from your version to a newer one, they will be listed in the release notes. Note that this FAQ answer is also valid for users of a hidden operating system. Also note that you cannot downgrade CipherShed if the system partition/drive is encrypted.

## **I use pre-boot authentication. Can I prevent a person (adversary) that is watching me start my computer from knowing that I use CipherShed?**

Yes (as of TrueCrypt 6.1). To do so, boot the encrypted system, start CipherShed, select Settings > System Encryption, enable the option ‘Do not show any texts in the pre-boot authentication screen’ and click OK. Then, when you start the computer, no texts will be displayed by the CipherShed boot loader (not even when you enter the wrong password). The computer will appear to be “frozen” while you can type your password. It is, however, important to note that if the adversary can analyze the content of the hard drive, he can still find out that it contains the CipherShed boot loader.

## **I use pre-boot authentication. Can I configure the CipherShed Boot Loader to display only a fake error message?**

Yes (as of TrueCrypt 6.1). To do so, boot the encrypted system, start CipherShed, select Settings > System Encryption, enable the option ‘Do not show any texts in the pre-boot authentication screen’ and enter the fake error message in the corresponding field (for example, the “Missing operating system” message, which is normally displayed by the Windows boot loader if it finds no Windows boot partition). It is, however, important to note that if the adversary can analyze the content of the hard drive, he can still find out that it contains the CipherShed boot loader.

## **Can I configure CipherShed to mount automatically whenever Windows starts a non-system CipherShed volume that uses the same password as my system partition/drive (i.e. my pre-boot authentication password)?**

Yes. To do so, follow these steps:

1. Mount the volume (to the drive letter to which you want it to be mounted every time).
2. Right-click the mounted volume in the drive list in the main CipherShed window and select ‘Add to System Favorites’.
3. The System Favorites Organizer window should appear now. In this window, enable the option ‘Mount system favorite volumes when Windows starts’ and click OK.

For more information, see the chapter ‘System Favorite Volumes’.

## **Can a volume be automatically mounted whenever I log on to Windows?**

Yes. To do so, follow these steps:

1. Mount the volume (to the drive letter to which you want it to be mounted every time).
2. Right-click the mounted volume in the drive list in the main CipherShed window and select ‘Add to Favorites’.
3. The Favorites Organizer window should appear now. In this window, enable the option ‘Mount selected volume upon logon’ and click OK.

Then, when you log on to Windows, you will be asked for the volume password (and/or keyfiles) and if it is correct, the volume will be mounted.

Alternatively, if the volumes are partition/device-hosted and if you do not need to mount them to particular drive letters every time, you can follow these steps:

1. Select Settings > Preferences. The Preferences window should appear now.
2. In the section ‘Actions to perform upon logon to Windows’, enable the option ‘Mount all devices-hosted CipherShed volumes’ and click OK.

Note: CipherShed will not prompt you for a password if you have enabled caching of the pre-boot authentication password (Settings > ‘System Encryption’) and the volumes use the same password as the system partition/drive.

## **Can a volume be automatically mounted whenever its host device gets connected to the computer?**

Yes. For example, if you have a CipherShed container on a USB flash drive and you want CipherShed to mount it automatically when you insert the USB flash drive into the USB port, follow these steps:

1. Mount the volume (to the drive letter to which you want it to be mounted every time).
2. Right-click the mounted volume in the drive list in the main CipherShed window and select ‘Add to Favorites’.
3. The Favorites Organizer window should appear now. In this window, enable the option ‘Mount selected volume when its host device gets connected’ and click OK.

Then, when you insert the USB flash drive into the USB port, you will be asked for the volume password (and/or keyfiles) (unless it is cached) and if it is correct, the volume will be mounted.

Note: CipherShed will not prompt you for a password if you have enabled caching of the pre-boot authentication password (Settings > ‘System Encryption’) and the volume uses the same password as the system partition/drive.

## **Can my pre-boot authentication password be cached so that I can use it mount non-system volumes during the session?**

Yes. Select ‘Settings’> ‘System Encryption’ and enable the following option: ‘Cache pre-boot authentication password in driver memory’.

## **I live in a country that violates basic human rights of its people. Is it possible to use CipherShed without leaving any ‘traces’ on unencrypted Windows?**

Yes. This can be achieved by running CipherShed in portable mode under BartPE or in a similar environment. BartPE stands for "Bart's Preinstalled Environment", which is essentially the Windows operating system prepared in a way that it can be entirely stored on and booted from a CD/DVD (registry, temporary files, etc., are stored in RAM – hard drive is not used at all and does not even have to be present). The freeware Bart's PE Builder can transform a Windows XP installation CD into a BartPE CD. Note that you do not even need any special CipherShed plug-in for BartPE. Follow these steps:

1. Create a BartPE CD and boot it. (Note: You must perform each of the following steps from within BartPE.)
2. Download the CipherShed self-extracting package to the RAM disk (which BartPE automatically creates).

Note: If the adversary can intercept data you send or receive over the Internet and you need to prevent the adversary from knowing you downloaded CipherShed, consider downloading it via I2P, Tor, or a similar anonymizing network.

3. Verify the digital signatures of the downloaded file (see the section Digital Signatures for more information).
4. Run the downloaded file, and select Extract (instead of Install) on the second page of the CipherShed Setup wizard. Extract the contents to the RAM disk.
5. Run the file CipherShed.exe from the RAM disk.

Note: You may also want to consider creating a hidden operating system (see the section Hidden Operating System). See also the chapter Plausible Deniability.

### **Can I encrypt my system partition/drive if I don't have a US keyboard?**

Yes, CipherShed supports all keyboard layouts.

### **Can I save data to the decoy system partition without risking damage to the hidden system partition?**

Yes. You can write data to the decoy system partition anytime without any risk that the hidden volume will get damaged (because the decoy system is not installed within the same partition as the hidden system). For more information, see the section Hidden Operating System.

### **Can I use CipherShed in Windows if I do not have administrator privileges?**

See the chapter 'Using CipherShed Without Administrator Privileges'.

### **Does CipherShed save my password to a disk?**

No.

### **How does CipherShed verify that the correct password was entered?**

See the chapter Technical Details, section Encryption Scheme.

### **Can I encrypt a partition/drive without losing the data currently stored on it?**

Yes, but the following conditions must be met:

- If you want to encrypt an entire system drive (which may contain multiple partitions) or a system partition (in other words, if you want to encrypt a drive or partition where Windows is installed), you can do so provided that you use CipherShed (TrueCrypt 5.0 or later) and that you use Windows XP or a later version of Windows (such as Windows 7) (select 'System' > 'Encrypt System Partition/Drive' and then follow the instructions in the wizard).
- If you want to encrypt a non-system partition in place, you can do so provided that it contains an NTFS filesystem, that you use CipherShed (TrueCrypt 6.1 or later), and that you use Windows Vista or a later version of Windows (for example, Windows 7) (click 'Create Volume' > 'Encrypt a non-system partition' > 'Standard volume' > 'Select Device' > 'Encrypt partition in place' and then follow the instructions in the wizard).

### **Can I run CipherShed if I don't install it?**

Yes, see the chapter Portable Mode.

### **Some encryption programs use TPM to prevent attacks. Will CipherShed use it too?**

No. Those programs use TPM to protect against attacks that require the attacker to have administrator privileges, or physical access to the computer, and the attacker needs you to use the computer after such an access. However, if any of these conditions is met, it is actually impossible to secure the computer (see below) and, therefore, you must stop using it (instead of relying on TPM).

If the attacker has administrator privileges, he can, for example, reset the TPM, capture the content of RAM (containing master keys) or content of files stored on mounted CipherShed volumes (decrypted on the fly), which can then be sent to the attacker over the Internet or saved to an unencrypted local drive (from which the attacker might be able to read it later, when he gains physical access to the computer).

If the attacker can physically access the computer hardware (and you use it after such an access), he can, for example, attach a malicious component to it (such as a hardware keystroke logger) that will capture the password, the content of RAM (containing master keys) or content of files stored on mounted CipherShed volumes (decrypted on the fly), which can then be sent to the attacker over the Internet or saved to an unencrypted local drive (from which the attacker might be able to read it later, when he gains physical access to the computer again).

The only thing that TPM is almost guaranteed to provide is a false sense of security (even the name itself, “Trusted Platform Module”, is misleading and creates a false sense of security). As for real security, TPM is actually redundant (and implementing redundant features is usually a way to create so-called bloatware). Features like this are sometimes referred to as ‘security theater’ [6].

For more information, please see the sections Physical Security and Malware.

### **Why does Windows Vista (and later versions of Windows) ask me for permission to run CipherShed every time I run it in ‘portable’ mode?**

When you run CipherShed in portable mode, CipherShed needs to load and start the CipherShed device driver. CipherShed needs a device driver to provide transparent on-the-fly encryption/decryption, and users without administrator privileges cannot start device drivers in Windows. Therefore, Windows Vista and later versions of Windows ask you for permission to run CipherShed with administrator privileges.

Note that if you install CipherShed on the system (as opposed to running CipherShed in portable mode), you will not be asked for permission every time you run it.

### **Do I have to dismount CipherShed volumes before shutting down or restarting Windows?**

No. CipherShed automatically dismounts all mounted CipherShed volumes on system shutdown/restart.

### **Which type of CipherShed volume is better – partition or file container?**

File containers are normal files so you can work with them as with any normal files (file containers can be, for example, moved, renamed, and deleted the same way as normal files). Partitions/drives may be better as regards performance. Note that reading and writing to/from a file container may take significantly longer when the container is heavily fragmented. To solve this problem, defragment the file system in which the container is stored (when the CipherShed volume is dismounted).

What’s the recommended way to back up a CipherShed volume?

See the chapter How to Back Up Securely.

### **What will happen if I format a CipherShed partition?**

See the question “Is it possible to change the file system of an encrypted volume?” in this FAQ.

### **Is it possible to change the file system of an encrypted volume?**

Yes, when mounted, CipherShed volumes can be formatted as FAT12, FAT16, FAT32, NTFS, or any other file system. CipherShed volumes behave as standard disk devices so you can right-click the device icon (for example in the ‘Computer’ or ‘My Computer’ list) and select ‘Format’. The actual volume contents will be lost. However, the whole volume will remain encrypted. If you format a CipherShed-encrypted partition when the CipherShed volume that the partition hosts is not mounted, then the volume will be destroyed, and the partition will not be encrypted anymore (it will be empty).

## **Is it possible to mount a CipherShed container that is stored on a CD or DVD?**

Yes. However, if you need to mount a CipherShed volume that is stored on a read-only medium (such as a CD or DVD) under Windows 2000, the file system within the CipherShed volume must be FAT (Windows 2000 cannot mount an NTFS file system on read-only media).

## **Is it possible to change the password for a hidden volume?**

Yes, the password change dialog works both for standard and hidden volumes. Just type the password for the hidden volume in the ‘Current Password’ field of the ‘Volume Password Change’ dialog.

Remark: CipherShed first attempts to decrypt the standard volume header and if it fails, it attempts to decrypt the area within the volume where the hidden volume header may be stored (if there is a hidden volume within). In case it is successful, the password change applies to the hidden volume. (Both attempts use the password typed in the ‘Current Password’ field.)

## **When I use HMAC-RIPemd-160, is the size of the header encryption key only 160 bits?**

No, CipherShed never uses an output of a hash function (nor of a HMAC algorithm) directly as an encryption key. See the section ‘Header Key Derivation, Salt, and Iteration Count’ for more information.

## **How do I burn a CipherShed container larger than 2 GB onto a DVD?**

The DVD burning software you use should allow you to select the format of the DVD. If it does, select the UDF format (ISO format does not support files larger than 2 GB).

## **Can I use tools like chkdsk, Disk Defragmenter, etc. on the contents of a mounted CipherShed volume?**

Yes, CipherShed volumes behave like real physical disk devices, so it is possible to use any filesystem checking/repairing/defragmenting tools on the contents of a mounted CipherShed volume.

## **Does CipherShed support 64-bit versions of Windows?**

Yes.

## **Can I mount my CipherShed volume under Windows, Mac OS X, and Linux?**

Yes, CipherShed volumes are fully cross-platform.

## **Is it possible to install an application to a CipherShed volume and run it from there?**

Yes.

## **What will happen when a part of a CipherShed volume becomes corrupted?**

In encrypted data, one corrupted bit usually corrupts the whole ciphertext block in which it occurred. The ciphertext block size used by CipherShed is 16 bytes (i.e., 128 bits). The mode of operation used by CipherShed ensures that if data corruption occurs within a block, the remaining blocks are not affected (for more information, see the section Modes of Operation). See also the question ‘What do I do when the encrypted filesystem on my CipherShed volume is corrupted?’

## **What do I do when the encrypted filesystem on my CipherShed volume is corrupted?**

File system within a CipherShed volume may become corrupted in the same way as any normal unencrypted file system. When that happens, you can use filesystem repair tools supplied with your operating system to fix it. In Windows, it is the ‘chkdsk’ tool. CipherShed provides an easy way to use this tool on a CipherShed volume: Right-click the mounted volume in the main CipherShed window (in the drive list) and from the context menu select ‘Repair Filesystem’.

We use CipherShed in a corporate/enterprise environment. Is there a way for an administrator to reset a volume password or pre-boot authentication password when a user forgets it (or loses a keyfile)?

Yes. Note that there is no “backdoor” implemented in CipherShed. However, there is a way to “reset” volume passwords/keyfiles and pre-boot authentication passwords. After you create a volume, back up its header to a file (select Tools -> Backup Volume Header) before you allow a non-admin user to use the volume. Note that the volume header (which is encrypted with a header key derived from a password/keyfile) contains the master key with which the volume is encrypted. Then ask the user to choose a password, and set it for him/her (Volumes -> Change Volume Password); or generate a user keyfile for him/her. Then you can allow the user to use the volume and to change the password/keyfiles without your assistance/permission. In case he/she forgets his/her password or loses his/her keyfile, you can “reset” the volume password/keyfiles to your original admin password/keyfiles by restoring the volume header from the backup file (Tools -> Restore Volume Header).

Similarly, you can reset a pre-boot authentication password. To create a backup of the master key data (that will be stored on a CipherShed Rescue Disk and encrypted with your administrator password), select ‘System’> ‘Create Rescue Disk’. To set a user pre-boot authentication password, select ‘System’> ‘Change Password’. To restore your administrator password, boot the CipherShed Rescue Disk, select ‘Repair Options’> ‘Restore key data’, and enter your administrator password. Note: It is not required to burn each CipherShed Rescue Disk ISO image to a CD/DVD. You can maintain a central repository of ISO images for all workstations (rather than a repository of CDs/DVDs). For more information, see the section Command Line Usage (option /noisocheck).

### **Can our commercial company use CipherShed free of charge?**

Provided that you comply with the terms and conditions of the CipherShed License, you can install and run CipherShed free of charge on an arbitrary number of your computers.

We share a volume over a network. Is there a way to have the network share automatically restored when the system is restarted?

Please see the chapter Sharing over Network.

### **It is possible to access a single CipherShed volume simultaneously from multiple operating systems (for example, a volume shared over a network)?**

Please see the chapter Sharing over Network.

### **Can a user access his or her CipherShed volume via a network?**

Please see the chapter Sharing over Network.

**I encrypted a non-system partition, but its original drive letter is still visible in the ‘My Computer’ list. When I double click this drive letter, Windows asks if I want to format the drive. Is there a way to hide or free this drive letter?**

Yes, to free the drive letter follow these steps:

1. Right-click the ‘Computer’ (or ‘My Computer’) icon on your desktop or in the Start Menu and select Manage. The ‘Computer Management’ window should appear.
2. From the list on the left, select ‘Disk Management’ (within the Storage sub-tree).
3. Right-click the encrypted partition/device and select ‘Change Drive Letter and Paths’.
4. Click Remove.

5. If Windows prompts you to confirm the action, click Yes.

**When I plug in my encrypted USB flash drive, Windows asks me if I want to format it. Is there a way to prevent that?**

Yes, but you will need to remove the drive letter assigned to the device. For information on how to do so, see the question ‘I encrypted a non-system partition, but its original drive letter is still visible in the ‘My Computer’ list.’

**How do I remove or undo encryption if I do not need it anymore? How do I permanently decrypt a volume?**

Please see the section How to Remove Encryption.

**What will change when I enable the option ‘Mount volumes as removable media’?**

Please see the section Volume Mounted as Removable Medium.

**Do I have to “wipe” free space and/or files on a CipherShed volume?**

Remark: to "wipe" = to securely erase; to overwrite sensitive data in order to render them unrecoverable.

If you believe that an adversary will be able to decrypt the volume (for example that he will make you reveal the password), then the answer is yes. Otherwise, it is not necessary, because the volume is entirely encrypted.

**How does CipherShed know which encryption algorithm my CipherShed volume has been encrypted with?**

Please see the section Encryption Scheme (chapter Technical Details).

# Technical Details

## Notation

| Symbol      | Description   |
|-------------|---|
| C           | Ciphertext block  |
| DK()        | Decryption algorithm using encryption/decryption key K  |
| EK()        | Encryption algorithm using encryption/decryption key K  |
| H()         | Hash function   |
| I           | Block index for n-bit blocks; n is context-dependent  |
| K           | Cryptographic key   |
| P           | Plaintext block   |
| $\wedge$    | Bitwise exclusive-OR operation (XOR)  |
| $\oplus$    | Modulo $2^n$ addition, where n is the bit size of the left-most operand and of the resultant value (e.g., if the left operand is a 1-bit value, and the right operand is a 2-bit value, then: $1 \oplus 0=1$ ; $1 \oplus 1=0$ ; $1 \oplus 2=1$ ; $1 \oplus 3=0$ ; $0 \oplus 0=0$ ; $0 \oplus 1=1$ ; $0 \oplus 2=0$ ; $0 \oplus 3=1$ ) |
| $\otimes$   | Modular multiplication of two polynomials over the binary field GF(2) modulo $x^{128}+x^7+x^2+x+1$ (GF stands for Galois Field)   |
| $\parallel$ | Concatenation   |

## Encryption Scheme

When mounting a CipherShed volume (assume there are no cached passwords/keyfiles) or when performing pre-boot authentication, the following steps are performed:

1. The first 512 bytes of the volume (i.e., the standard volume header) are read into RAM, out of which the first 64 bytes are the salt (see CipherShed Volume Format Specification). For system encryption (see the chapter System Encryption), the last 512 bytes of the first logical drive track are read into RAM (the CipherShed Boot Loader is stored in the first track of the system drive and/or on the CipherShed Rescue Disk).

2. Bytes 65536–66047 of the volume are read into RAM (see the section CipherShed Volume Format Specification). For system encryption, bytes 65536–66047 of the first partition located behind the active partition<sup>26</sup> are read (see the section Hidden Operating System). If there is a hidden volume within this volume (or within the partition behind the boot partition), we have read its header at this point; otherwise, we have just read random data (whether or not there is a hidden volume within it has to be determined by attempting to decrypt this data; for more information see the section Hidden Volume).
3. Now CipherShed attempts to decrypt the standard volume header read in (1). All data used and generated in the course of the process of decryption are kept in RAM (CipherShed never saves them to disk). The following parameters are unknown<sup>27</sup> and have to be determined through the process of trial and error (i.e., by testing all possible combinations of the following):
  - a. PRF used by the header key derivation function (as specified in PKCS #5 v2.0; see the section Header Key Derivation, Salt, and Iteration Count), which can be one of the following: HMAC-SHA-512, HMAC-RIPEMD-160, HMAC-Whirlpool. A password entered by the user (to which one or more keyfiles may have been applied – see the section Keyfiles) and the salt read in (1) are passed to the header key derivation function, which produces a sequence of values (see the section Header Key Derivation, Salt, and Iteration Count) from which the header encryption key and secondary header key (XTS mode) are formed. (These keys are used to decrypt the volume header.)
  - b. Encryption algorithm: AES-256, Serpent, Twofish, AES-Serpent, AES-Twofish-Serpent, etc.
  - c. Mode of operation: XTS, LRW (deprecated/legacy), CBC (deprecated/legacy)
  - d. Key size(s)
4. Decryption is considered successful if the first 4 bytes of the decrypted data contain the ASCII string “TRUE”, and if the CRC-32 checksum of the last 256 bytes of the decrypted data (volume header) matches the value located at byte #8 of the decrypted data (this value is unknown to an adversary because it is encrypted – see the section CipherShed Volume Format Specification). If these conditions are not met, the process continues from (3) again, but this time, instead of the data read in (1), the data read in (2) are used (i.e., possible hidden volume header). If the conditions are not met again, mounting is terminated (wrong password, corrupted volume, or not a CipherShed volume).
5. Now we know (or assume with very high probability) that we have the correct password, the correct encryption algorithm, mode, key size, and the correct header key derivation algorithm. If we successfully decrypted the data read in (2), we also know that we are mounting a hidden volume and its size is retrieved from data read in (2) decrypted in (3).

<sup>26</sup> If the size of the active partition is less than 256 MB, then the data is read from the second partition behind the active one (Windows 7 and later, by default, do not boot from the partition on which they are installed).

<sup>27</sup> These parameters are kept secret not in order to increase the complexity of an attack, but primarily to make TrueCrypt volumes unidentifiable (indistinguishable from random data), which would be difficult to achieve if these parameters were stored unencrypted within the volume header. Also note that if a non-cascaded encryption algorithm is used for system encryption, the algorithm is known (it can be determined by analyzing the contents of the unencrypted TrueCrypt Boot Loader stored in the first logical drive track or on the TrueCrypt Rescue Disk).

- The encryption routine is reinitialized with the primary master key<sup>28</sup> and the secondary master key (XTS mode – see the section Modes of Operation), which are retrieved from the decrypted volume header (see the section CipherShed Volume Format Specification). These keys can be used to decrypt any sector of the volume, except the volume header area (or the key data area, for system encryption), which has been encrypted using the header keys. The volume is mounted.

See also section Modes of Operation and section Header Key Derivation, Salt, and Iteration Count and also the chapter Security Model.

## **Modes of Operation**

The mode of operation used by CipherShed for encrypted partitions, drives, and virtual volumes is XTS.

XTS mode is in fact XEX mode [12], which was designed by Phillip Rogaway in 2003, with a minor modification (XEX mode uses a single key for two different purposes, whereas XTS mode uses two independent keys).

In 2010, XTS mode was approved by NIST for protecting the confidentiality of data on storage devices [24]. In 2007, it was also approved by the IEEE for cryptographic protection of data on block-oriented storage devices (IEEE 1619).

### **Description of XTS mode:**

$$C_i = E_{K1} (P_i \wedge (E_{K2}(n) \otimes \alpha^i)) \wedge (E_{K2}(n) \otimes \alpha^i)$$

Where:

$\otimes$  denotes multiplication of two polynomials over the binary field GF(2) modulo  $x^7+x^2+x+1$

K1 is the encryption key (256-bit for each supported cipher; i.e., AES, Serpent, and Twofish)

K2 is the secondary key (256-bit for each supported cipher; i.e., AES, Serpent, and Twofish)

i is the cipher block index within a data unit; for the first cipher block within a data unit,  $i = 0$

n is the data unit index within the scope of K1; for the first data unit,  $n = 0$

$\alpha$  is a primitive element of Galois Field (2128) that corresponds to polynomial x (i.e., 2)

The size of each data unit is always 512 bytes (regardless of the sector size).

For further information pertaining to XTS mode, see e.g. [12] and [24].

<sup>28</sup> The master keys were generated during the volume creation and cannot be changed later. Volume password change is accomplished by re-encrypting the volume header using a new header key (derived from a new password). Modes

## **Header Key Derivation, Salt, and Iteration Count**

Header key is used to encrypt and decrypt the encrypted area of the CipherShed volume header (for system encryption, or the key data area), which contains the master key and other data (see the sections Encryption Scheme and CipherShed Volume Format Specification). In volumes created by CipherShed (TrueCrypt 5.0 or later) (and for system encryption), the area is encrypted in XTS mode (see the section Modes of Operation). The method that CipherShed uses to generate the header key and the secondary header key (XTS mode) is PBKDF2, specified in PKCS #5 v2.0; see References.

512-bit salt is used, which means there are  $2^{512}$  keys for each password. This significantly decreases vulnerability to ‘off-line’ dictionary/‘rainbow table’ attacks (pre-computing all the keys for a dictionary of passwords is very difficult when a salt is used) [7]. The salt consists of random values generated by the CipherShed random number generator during the volume creation process. The header key derivation function is based on HMAC-SHA-512, HMAC-RIPEMD-160, or HMAC-Whirlpool (see [8][9][20][22]) – the user selects which. The length of the derived key does not depend on the size of the output of the underlying hash function. For example, a header key for the AES-256 cipher is always 256 bits long even if HMAC-RIPEMD-160 is used (in XTS mode, an additional 256-bit secondary header key is used; hence, two 256-bit keys are used for AES-256 in total). For more information, refer to [7]. 1000 iterations (or 2000 iterations when HMAC-RIPEMD-160 is used as the underlying hash function) of the key derivation function have to be performed to derive a header key, which increases the time necessary to perform an exhaustive search for passwords (i.e., brute force attack) [7].

Header keys used by ciphers in a cascade are mutually independent, even though they are derived from a single password (to which keyfiles may have been applied). For example, for the AES-Twofish-Serpent cascade, the header key derivation function is instructed to derive a 768-bit encryption key from a given password (and, for XTS mode, in addition, a 768-bit secondary header key from the given password). The generated 768-bit header key is then split into three 256-bit keys (for XTS mode, the secondary header key is split into three 256-bit keys too, so the cascade actually uses six 256-bit keys in total), out of which the first key is used by Serpent, the second key is used by Twofish, and the third by AES (in addition, for XTS mode, the first secondary key is used by Serpent, the second secondary key is used by Twofish, and the third secondary key by AES). Hence, even when an adversary has one of the keys, he cannot use it to derive the other keys, as there is no feasible method to determine the password from which the key was derived (except for brute force attack mounted on a weak password).

## **Random Number Generator**

The CipherShed random number generator (RNG) is used to generate the master encryption key, the secondary key (XTS mode), salt, and keyfiles. It creates a pool of random values in RAM (memory). The pool, which is 320 bytes long, is filled with data from the following sources:

- Mouse movements
- Keystrokes
- Mac OS X and Linux: Values generated by the built-in RNG (both /dev/random and /dev/urandom)
- MS Windows only: MS Windows CryptoAPI (collected regularly at 500-ms interval)
- MS Windows only: Network interface statistics (NETAPI32)
- MS Windows only: Various Win32 handles, time variables, and counters (collected regularly at 500-ms interval)

Before a value obtained from any of the above-mentioned sources is written to the pool, it is divided into individual bytes (e.g., a 32-bit number is divided into four bytes). These bytes are then individually written to the pool with the modulo  $2^8$  addition operation (not by replacing the old values in the pool) at the position of the pool cursor. After a byte is written, the pool cursor position is advanced by one byte. When the cursor reaches the end of the pool, its position is set to the beginning of the pool. After every 16th byte written to the pool, the pool mixing function is automatically applied to the entire pool (see below).

## Pool Mixing Function

The purpose of this function is to perform diffusion [2]. Diffusion spreads the influence of individual “raw” input bits over as much of the pool state as possible, which also hides statistical relationships. After every 16th byte written to the pool, this function is applied to the entire pool.

Description of the pool mixing function:

1. Let R be the randomness pool.
2. Let H be the hash function selected by the user (SHA-512, RIPEMD-160, or Whirlpool).
3. l = byte size of the output of the hash function H (i.e., if H is RIPEMD-160, then l = 20; if H is SHA-512, l = 64)
4. z = byte size of the randomness pool R (320 bytes)
5. q =  $z / l - 1$  (e.g., if H is Whirlpool, then q = 4)
6. R is divided into l-byte blocks  $B_0 \dots B_q$ .

For  $0 \leq i \leq q$  (i.e., for each block B) the following steps are performed:

- a.  $M = H(B_0 \parallel B_1 \parallel \dots \parallel B_q)$  [i.e., the randomness pool is hashed using the hash function H, which produces a hash M]
- b.  $B_i = B_i \wedge M$
7.  $R = B_0 \parallel B_1 \parallel \dots \parallel B_q$

For example, if  $q = 1$ , the randomness pool would be mixed as follows:

- 1)  $(B_0 \parallel B_1) = R$
- 2)  $B_0 = B_0 \wedge H(B_0 \parallel B_1)$
- 3)  $B_1 = B_1 \wedge H(B_0 \parallel B_1)$
- 4)  $R = B_0 \parallel B_1$

## Generated Values

The content of the RNG pool is never directly exported (even when CipherShed instructs the RNG to generate and export a value). Thus, even if the attacker obtains a value generated by the RNG, it is infeasible for him to determine or predict (using the obtained value) any other values generated by the RNG during the session (it is infeasible to determine the content of the pool from a value generated by the RNG).

The RNG ensures this by performing the following steps whenever CipherShed instructs it to generate and export a value:

1. Data obtained from the sources listed above is added to the pool as described above.
2. The requested number of bytes is copied from the pool to the output buffer (the copying starts from the position of the pool cursor; when the end of the pool is reached, the copying continues from the beginning of the pool; if the requested number of bytes is greater than the size of the pool, no value is generated and an error is returned).
3. The state of each bit in the pool is inverted (i.e., 0 is changed to 1, and 1 is changed to 0).
4. Data obtained from some of the sources listed above is added to the pool as described above.
5. The content of the pool is transformed using the pool mixing function. Note: The function uses a cryptographically secure one-way hash function selected by the user (for more information, see the section Pool Mixing Function above).
6. The transformed content of the pool is XORed into the output buffer as follows:
  - a. The output buffer write cursor is set to 0 (the first byte of the buffer).
  - b. The byte at the position of the pool cursor is read from the pool and XORed into the byte in the output buffer at the position of the output buffer write cursor.
  - c. The pool cursor position is advanced by one byte. If the end of the pool is reached, the cursor position is set to 0 (the first byte of the pool).
  - d. The position of the output buffer write cursor is advanced by one byte.
  - e. Steps b–d are repeated for each remaining byte of the output buffer (whose length is equal to the requested number of bytes).
7. The content of the output buffer, which is the final value generated by the RNG, is exported.

## Design Origins

The design and implementation of the random number generator are based on the following works:

- Software Generation of Practically Strong Random Numbers by Peter Gutmann [10]
- Cryptographic Random Numbers by Carl Ellison [11]

## Keyfiles

CipherShed keyfile is a file whose content is combined with a password. The user can use any kind of file as a CipherShed keyfile. The user can also generate a keyfile using the built-in keyfile generator, which utilizes the CipherShed RNG to generate a file with random content (for more information, see the section Random Number Generator).

The maximum size of a keyfile is not limited; however, only its first 1,048,576 bytes (1 MB) are processed (all remaining bytes are ignored due to performance issues connected with processing extremely large files). The user can supply one or more keyfiles (the number of keyfiles is not limited).

Keyfiles can be stored on PKCS-11-compliant [23] security tokens and smart cards protected by multiple PIN codes (which can be entered either using a hardware PIN pad or via the CipherShed GUI).

Keyfiles are processed and applied to a password using the following method:

1. Let P be a CipherShed volume password supplied by user (may be empty)
2. Let KP be the keyfile pool
3. Let kpl be the size of the keyfile pool KP, in bytes (64, i.e., 512 bits); kpl must be a multiple of the output size of a hash function H
4. Let pl be the length of the password P, in bytes (in the current version:  $0 \leq pl \leq 64$ )
5. if  $kpl > pl$ , append  $(kpl - pl)$  zero bytes to the password P (thus  $pl = kpl$ )
6. Fill the keyfile pool KP with kpl zero bytes.
7. For each keyfile perform the following steps:
  - a. Set the position of the keyfile pool cursor to the beginning of the pool
  - b. Initialize the hash function H
  - c. Load all bytes of the keyfile one by one, and for each loaded byte perform the following steps:
    - i. Hash the loaded byte using the hash function H without initializing the hash, to obtain an intermediate hash (state) M. Do not finalize the hash (the state is retained for next round).
    - ii. Divide the state M into individual bytes. For example, if the hash output size is 4 bytes,  $(T_0 \parallel T_1 \parallel T_2 \parallel T_3) = M$
    - iii. Write these bytes (obtained in step 7.c.ii) individually to the keyfile pool with the modulo  $2^8$  addition operation (not by replacing the old values in the pool) at the position of the pool cursor. After a byte is written, the pool cursor position is advanced by one byte. When the cursor reaches the end of the pool, its position is set to the beginning of the pool.
8. Apply the content of the keyfile pool to the password P using the following method:
  - a. Divide the password P into individual bytes  $B_0 \dots B_{pl-1}$ .

Note that if the password was shorter than the keyfile pool, then the password was padded with zero bytes to the length of the pool in Step 5 (hence, at this point the length of the password is always greater than or equal to the length of the keyfile pool).

- b. Divide the keyfile pool KP into individual bytes  $G_0 \dots G_{kpl-1}$
- c. For  $0 \leq i \leq kpl$  perform:  $B_i = B_i \oplus G_i$

d.  $P = B_0 \parallel B_1 \parallel \dots \parallel B_{pl-2} \parallel B_{pl-1}$

9. The password P (after the keyfile pool content has been applied to it) is now passed to the header key derivation function PBKDF2 (PKCS #5 v2), which processes it (along with salt and other data) using a cryptographically secure hash algorithm selected by the user (e.g., SHA-512). See the section Header Key Derivation, Salt, and Iteration Count for more information.

The role of the hash function H is merely to perform diffusion [2]. CRC-32 is used as the hash function H. Note that the output of CRC-32 is subsequently processed using a cryptographically secure hash algorithm: The keyfile pool content (in addition to being hashed using CRC-32) is applied to the password, which is then passed to the header key derivation function PBKDF2 (PKCS #5 v2), which processes it (along with salt and other data) using a cryptographically secure hash algorithm selected by the user (e.g., SHA-512). The resultant values are used to form the header key and the secondary header key (XTS mode).

## **CipherShed Volume Format Specification**

| Offset<br>(bytes) | Size<br>(bytes) | Encryption Status <sup>29</sup> | Description   |
|-------------------|-----------------|---------------------------------|---|
| 0                 | 64              | Unencrypted <sup>30</sup>       | Salt  |
| 64                | 4               | Encrypted                       | ASCII string “TRUE”   |
| 68                | 2               | Encrypted                       | Volume header format version (5)  |
| 70                | 2               | Encrypted                       | Minimum program version required to open the volume   |
| 72                | 4               | Encrypted                       | CRC - 32 checksum of the (decrypted) bytes 256 – 511  |
| 76                | 16              | Encrypted                       | Reserved (must contain zeroes)  |
| 92                | 8               | Encrypted                       | Size of hidden volume (set to zero in non-hidden volumes)   |
| 100               | 8               | Encrypted                       | Size of volume  |
| 108               | 8               | Encrypted                       | Byte offset of the start of the master key scope  |
| 116               | 8               | Encrypted                       | Size of the encrypted area within the master key scope  |
| 124               | 4               | Encrypted                       | Flag bits (bit 0 set: system encryption; bit 1 set: non – system in – place - encrypted volume; bits 2 - 31 are reserved) |
| 128               | 4               | Encrypted                       | Sector size (in bytes)  |

---

<sup>29</sup> The encrypted areas of the volume header are encrypted in XTS mode using the primary and secondary header keys. For more information, see the section Encryption Scheme and the section Header Key Derivation, Salt, and Iteration Count.

<sup>30</sup> Note that the salt does not need to be encrypted, as it does not have to be kept secret [7] (salt is a sequence of random values).

| Offset<br>(bytes)      | Size<br>(bytes) | Encryption Status <sup>29</sup>     | Description   |
|------------------------|-----------------|-------------------------------------|---|
| 132                    | 120             | Encrypted                           | Reserved (must contain zeroes)  |
| 252                    | 4               | Encrypted                           | CRC - 32 checksum of the (decrypted) bytes 64 – 251   |
| 256                    | Var.            | Encrypted                           | Concatenated primary and secondary master keys <sup>31</sup>  |
| 512                    | 65024           | Encrypted                           | Reserved (for system encryption, this item is omitted <sup>32</sup> )   |
| 65536                  | 65536           | Encrypted/Unencrypted <sup>30</sup> | Area for hidden volume header (if there is no hidden volume within the volume, this area contains random data**). For system encryption, this item is omitted <sup>32</sup> . See bytes 0–65535.  |
| 131072                 | Var.            | Encrypted                           | Data area (master key scope). For system encryption, offset may be different (depending on offset of system partition).   |
| S-131072 <sup>33</sup> | 65536           | Encrypted/Unencrypted <sup>30</sup> | Backup header (encrypted with a different header key derived using a different salt). For system encryption, this item is omitted <sup>32</sup> . See bytes 0–65535.  |
| S-65536                | 65536           | Encrypted/Unencrypted <sup>30</sup> | Backup header for hidden volume (encrypted with a different header key derived using a different salt). If there is no hidden volume within the volume, this area contains random data <sup>34</sup> . For system encryption, this item is omitted <sup>32</sup> . See bytes 0–65535. |

Note that this specification applies to volumes created by CipherShed (TrueCrypt 7.0 or later). The format of file-hosted volumes is identical to the format of partition/device-hosted volumes (however, the "volume header", or key data, for a system partition/drive is stored in the last 512 bytes of the first logical drive track). CipherShed volumes have no "signature" or ID strings. Until decrypted, they appear to consist solely of random data.

<sup>31</sup> § Multiple concatenated master keys are stored here when the volume is encrypted using a cascade of ciphers (secondary master keys are used for XTS mode).

<sup>32</sup> Here, the meaning of "system encryption" does not include a hidden volume containing a hidden operating system.

<sup>33</sup> denotes the size of the volume host (in bytes).

<sup>34</sup> See below in this section for information on the method used to fill free volume space with random data when the volume is created.

Free space on each CipherShed volume is filled with random data when the volume is created.<sup>35</sup> The random data is generated as follows: Right before CipherShed volume formatting begins, a temporary encryption key and a temporary secondary key (XTS mode) are generated by the random number generator (see the section Random Number Generator). The encryption algorithm that the user selected is initialized with the temporary keys. The encryption algorithm is then used to encrypt plaintext blocks consisting of zeroes. The encryption algorithm operates in XTS mode (see the section Modes of Operation). The resulting ciphertext blocks are used to fill (overwrite) the free space on the volume. The temporary keys are stored in RAM and are erased after formatting finishes.

The fields located at byte #0 (salt) and #256 (master keys) contain random values generated by the random number generator (see the section Random Number Generator) during the volume creation process. If a CipherShed volume hosts a hidden volume (within its free space), the header of the hidden volume is located at byte #65536 of the host volume (the header of the host/outer volume is located at byte #0 of the host volume – see the section Hidden Volume). If there is no hidden volume within a CipherShed volume, bytes 65536–131071 of the volume (i.e., the area where the header of a hidden volume can reside) contain random data (see above for information on the method used to fill free volume space with random data when the volume is created). The layout of the header of a hidden volume is the same as the one of a standard volume (bytes 0–65535).

The maximum possible CipherShed volume size is  $2^{63}$  bytes (8,589,934,592 GB). However, due to security reasons (with respect to the 128-bit block size used by the encryption algorithms), the maximum allowed volume size is 1 PB (1,048,576 GB).

## Embedded Backup Headers

Each CipherShed volume created by CipherShed (TrueCrypt 6.0) or later contains an embedded backup header, located at the end of the volume (see above). The header backup is not a copy of the volume header because it is encrypted with a different header key derived using a different salt (see the section Header Key Derivation, Salt, and Iteration Count).

When the volume password and/or keyfiles are changed, or when the header is restored from the embedded (or an external) header backup, both the volume header and the backup header (embedded in the volume) are re-encrypted with different header keys (derived using newly generated salts – the salt for the volume header is different from the salt for the backup header). Each salt is generated by the CipherShed random number generator (see the section Random Number Generator).

For more information about header backups, see the subsection Tools -> Restore Volume Header in the chapter Main Program Window.

## Compliance with Standards and Specifications

To our best knowledge, CipherShed complies with the following standards, specifications, and recommendations:

- ISO/IEC 10118-3:2004 [21]
- FIPS 197 [3]

---

<sup>35</sup> Provided that the options Quick Format and Dynamic are disabled and provided that the volume does not contain a filesystem that has been encrypted in place (note that TrueCrypt does not allow the user to create a hidden volume within such a volume).

- FIPS 198 [22]
- FIPS 180-2 [14]
- NIST SP 800-3E [24]
- PKCS #5 v2.0 [7]
- PKCS #11 v2.20 [23]

The correctness of the implementations of the encryption algorithms can be verified using test vectors (select Tools > Test Vectors) or by examining the source code of CipherShed.

## **Source Code**

CipherShed is open-source and free software. The complete source code of CipherShed (written in C, C++, and assembly) is freely available for peer review at:

<https://ciphershed.org> and <https://github.com/CipherShed/CipherShed.git>

## **Future Development**

For the list of features that are planned for a future release, please refer to:

<https://ciphershed.org>

## Contact

Information on how to contact us can be found at:

<https://ciphershed.org>

# **Legal Information**

## **License**

The text of the license under which CipherShed is distributed is contained in the file License.txt that is included in the CipherShed binary and source code distribution packages, and is also available at:

<https://ciphershed.org>

## **Copyright Information**

This software as a whole: Copyright © 2014 CipherShed Project. All rights reserved.

Portions of this software: Copyright © 2003-2012 TrueCrypt Developers Association. All rights reserved.  
Copyright © 1998-2000 Paul Le Roux. All rights reserved. Copyright © 1998-2008 Brian Gladman, Worcester,  
UK. All rights reserved. Copyright © 2002-2004 Mark Adler. All rights reserved.

For more information, please see the legal notices attached to parts of the source code.

## **Trademark Information**

Any trademarks mentioned in this document are the sole property of their respective owners.

# **Version History**

7.1a

February 7, 2012

## **Improvements and bug fixes:**

- Minor improvements and bug fixes (Windows, Mac OS X, and Linux)

For a list of changes in older versions, see: <https://ciphershed.org>

## Acknowledgements

We would like to thank the following people:

Paul Le Roux for making his E4M source code available. TrueCrypt 1.0 was derived from E4M and some parts of the E4M source code are still incorporated in the latest version of the CipherShed source code. Brian Gladman, who wrote the excellent AES, Twofish, and SHA-512 routines. Peter Gutmann for his paper on random numbers, and for creating his cryptlib, which was the source of parts

of the random number generator source code. Wei Dai, who wrote the Serpent and RIPEMD-160 routines. Mark Adler et al., who wrote the Inflate routine. The designers of the encryption algorithms, hash algorithms, and the mode of operation:

Horst Feistel, Don Coppersmith, Walt Tuchmann, Lars Knudsen, Ross Anderson, Eli Biham, Bruce Schneier, David Wagner, John Kelsey, Niels Ferguson, Doug Whiting, Chris Hall, Joan Daemen, Vincent Rijmen, Carlisle Adams, Stafford Tavares, Phillip Rogaway, Hans Dobbertin, Antoon Bosselaers, Bart Preneel, Paulo S. L. M. Barreto.

All the others who have made this project possible, all who have morally supported us, and all who sent us bug reports or suggestions for improvements.

Thank you very much.

## References

- [1] U.S. Committee on National Security Systems (CNSS), National Policy on the Use of the Advanced Encryption Standard (AES) to Protect National Security Systems and National Security Information, CNSS Policy No. 15, Fact Sheet No. 1, June 2003, available at <http://csrc.nist.gov/groups/STM/cmvp/documents/CNSS15FS.pdf>.
- [2] C. E. Shannon, Communication Theory of Secrecy Systems, Bell System Technical Journal, v. 28, n. 4, 1949
- [3] NIST, Advanced Encryption Standard (AES), Federal Information Processing Standards Publication 197, November 26, 2001, available at <http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>.
- [4] J. Nechvatal, E. Barker, L. Bassham, W. Burr, M. Dworkin, J. Foti, E. Roback, NIST, Report on the Development of the Advanced Encryption Standard (AES), October 2, 2000, Journal of Research of the National Institute of Standards and Technology, Vol. 106, No. 3, May-June 2001, available at <http://nvl.nist.gov/pub/nistpubs/jres/106/3/j63nec.pdf>.
- [5] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Ferguson, T. Kohno, M. Stay, The Twofish Team's Final Comments on AES Selection, May 15, 2000, available at <http://csrc.nist.gov/archive/aes/round2/comments/20000515-bschneier.pdf>.
- [6] Bruce Schneier, Beyond Fear: Thinking Sensibly About Security in an Uncertain World, Springer, 2003
- [7] RSA Laboratories, PKCS #5 v2.0: Password-Based Cryptography Standard, RSA Data Security, Inc. Public-Key Cryptography Standards (PKCS), March 25, 1999, available at <ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-5v2/pkcs5v2-0.pdf>.
- [8] H. Krawczyk, M. Bellare, R. Canetti, HMAC: Keyed-Hashing for Message Authentication, RFC 2104, February 1997, available at <http://www.ietf.org/rfc/rfc2104.txt>.
- [9] M. Nystrom, RSA Security, Identifiers and Test Vectors for HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512, RFC 4231, December 2005, available at <http://www.ietf.org/rfc/rfc4231.txt>.
- [10] Peter Gutmann, Software Generation of Practically Strong Random Numbers, presented at the 1998 Usenix Security Symposium, available at <http://www.cs.auckland.ac.nz/~pgut001/pubs/usenix98.pdf>.
- [11] Carl Ellison, Cryptographic Random Numbers, originally an appendix to the P1363 standard, available at <http://world.std.com/~cme/P1363/ranno.html>.
- [12] P. Rogaway, Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC, *Asiacrypt 2004*. LNCS vol. 3329. Springer, 2004. Also available at: <http://www.cs.ucdavis.edu/~rogaway/papers/offsets.pdf>.
- [13] J. Kelsey, Twofish Technical Report #7: Key Separation in Twofish, AES Round 2 public comment, April 7, 2000
- [14] NIST, Secure Hash Standard, FIPS 180-2, August 1, 2002, available at <http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf>.

- [15] U. Maurer, J. Massey, Cascade Ciphers: The Importance of Being First, *Journal of Cryptology*, v. 6, n. 1, 1993
- [16] Bruce Schneier, *Applied Cryptography*, Second Edition, John Wiley & Sons, 1996
- [17] Peter Gutmann, Secure Deletion of Data from Magnetic and Solid-State Memory, first published in the Sixth USENIX Security Symposium Proceedings, San Jose, California, July 22-25, 1996, available at [http://www.cs.auckland.ac.nz/~pgut001/pubs/secure\\_del.html](http://www.cs.auckland.ac.nz/~pgut001/pubs/secure_del.html)
- [18] Serpent home page: <http://www.cl.cam.ac.uk/~rja14/serpent.html>.
- [19] M. E. Smid, AES Issues, AES Round 2 Comments, May 22, 2000, available at <http://csrc.nist.gov/archive/aes/round2/comments/20000523-msmid-2.pdf>.
- [20] A. Menezes, P. van Oorschot, S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, October 1996
- [21] International Organization for Standardization (ISO), Information technology – Security techniques – Hash-functions – Part 3: Dedicated hash-functions, ISO/IEC 10118-3:2004, February 24, 2004
- [22] NIST, The Keyed-Hash Message Authentication Code (HMAC), Federal Information Processing Standards Publication 198, March 6, 2002, available at <http://csrc.nist.gov/publications/fips/fips198/fips-198a.pdf>.
- [23] RSA Laboratories, PKCS #11 v2.20: Cryptographic Token Interface Standard, RSA Security, Inc. Public-Key Cryptography Standards (PKCS), June 28, 2004, available at <ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-11/v2-20/pkcs-11v2-20.pdf>.
- [24] Morris Dworkin, Recommendation for Block Cipher Modes of Operation: The XTS-AES Mode for Confidentiality on Storage Devices, NIST Special Publication 800-3E, January 2010, available at <http://csrc.nist.gov/publications/nistpubs/800-38E/nist-sp-800-38E.pdf>.