TRUECRYPT

FREE OPEN-SOURCE ON-THE-FLY ENCRYPTION

USER'S GUIDE

Version Information

TrueCrypt User's Guide, version 3.0a. Released December 11, 2004.

Trademark Information

All registered and unregistered trademarks in this document are the sole property of their respective owners.

Licensing and Patent Information

Before installing this product, you must agree to the license displayed in the TrueCrypt Setup window (the text of the license is also contained in the file *License.txt*).

The CAST5 encryption algorithm is described in U.S. patent number 5,511,123 [1]. However, CAST5 is available worldwide on a royalty-free basis for commercial and non-commercial uses [6].

Copyright Information

A TrueCrypt Foundation Release

For more information, please see the legal notices attached to parts of the source code.

Limitations

The TrueCrypt Foundation does not warrant that the information contained in this document meets your requirements or that the information is free of errors. The information may include technical inaccuracies or typographical errors.

CONTENTS

PREFACE

This document assumes that the reader is generally familiar with using computer hardware and software. Describing a feature that is usually easily understood has been avoided wherever possible.

Introduction

TrueCrypt is a software system for establishing and maintaining an on-the-fly-encrypted volume (data storage device). On-the-fly encryption means that data are automatically encrypted or decrypted right before they are loaded or saved, without any user intervention. *No* data stored on an encrypted volume can be read without using the correct password or encryption key. Until decrypted, encrypted volume appears to be nothing more than a series of random numbers. Entire file system is encrypted (i.e., file names, folder names, contents of every file, and free space). *No* unencrypted data are ever stored on any storage device (they are only temporarily kept in RAM during the encryption/decryption process).

TrueCrypt Volume

There are two basic types of TrueCrypt volumes:

- Container
- Partition/device

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

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

Creating a New TrueCrypt Volume

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

Hash Algorithm

Allows you to select which hash algorithm TrueCrypt will use. The selected hash algorithm is used by the random number generator (which generates the master key, salt, and the values used to create IV and whitening values). It is also used in deriving the new volume header key.

TrueCrypt currently supports two hash algorithms: RIPEMD-160, which was designed by an open academic community, and SHA-1 designed by the NSA and NIST.

Note that the output of a hash function is *never* used directly as an encryption key. Please refer to the section *Technical Details* for more information.

Encryption Algorithm

This allows you to select the encryption algorithm with which your new volume will be encrypted. For more information, please see the section *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.

Cluster Size

Cluster is an allocation unit. For example, for a one-byte file, at least one cluster should be allocated on FAT file system. 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 performance is better. If you do not know which value to use, leave the setting at default.

Auto-Test All Algorithms

The built-in automatic self-test facility, accessible from the Encryption Options page of the Volume Creation Wizard, automatically tests all the encryption algorithms and all the hash algorithms (HMAC's) implemented in TrueCrypt and reports the results. These tests are also run each time right before you start the Volume Creation Wizard. If there is any error, it is reported and the wizard will not start (this will prevent you from creating new volumes when the program is corrupted).

TrueCrypt Volume on CD, DVD, and Other Read-Only Media

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

Hardware/Software RAID, Windows Dynamic Volumes

TrueCrypt supports hardware/software RAID as well as Windows dynamic volumes. If you intend to format a dynamic volume as a TrueCrypt volume, keep in mind that after you create the 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' window of the TrueCrypt Volume Creation Wizard.

Also note that, in the 'Select Device' window, a dynamic volume is not displayed as a single device (one item). Instead, all the volumes that the dynamic volume consists of are displayed and you can select any of them in order to format the entire dynamic disk.

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, salt, and the values used to create the IV and whitening values for the new volume will be generated, and the master key and header key contents will be displayed.

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

Fig. 1. Randomness pool, master key, and header key contents displayed by the Volume Creation Wizard.

Note that only the first 112 bits of the pool/keys are displayed (not the entire contents).

Warning: When encrypting entire hard drive partition/device, i.e., formatting it as a TrueCrypt volume, all data stored on the partition/device will be lost!

Important: Several users reported that data on their TrueCrypt volumes that are located on USB hard drives were becoming corrupted. Later, these users found out that it was not a problem with TrueCrypt but with their hardware (chipset, USB PCI card, etc.) Therefore, we recommend that, before using a USB hard drive, you make sure data written to it is not becoming corrupted. For example, by copying a large set of files (at least 5 GB in total) and then comparing the original files with the copies.

You can create FAT (FAT12, FAT16, FAT32) or NTFS volumes (however, NTFS volumes can only be created by users with administrator privileges). TrueCrypt volumes can be reformatted as FAT12, FAT16, FAT32, or NTFS anytime. They behave as standard disk devices so you can right-click the device icon and select 'Format'.

For more information about creating TrueCrypt volumes, see also the section *Hidden Volume*.

Plausible Deniability

In case an adversary forces you to reveal the password, TrueCrypt provides two levels of plausible deniability. It is impossible to identify a TrueCrypt container or partition. Until decrypted, a TrueCrypt volume appears to consist of nothing more than random data (it does not contain any "signature"). Therefore, it is impossible to prove that a file, a partition or a device is a TrueCrypt volume and/or that it has been encrypted. The second level of plausible deniability is provided by the hidden volume feature (for more information, see the section *Hidden Volume*).

TrueCrypt container files do not have to have a standard file extension. They can have any file extension you like (for example, .raw, .dat, .iso, .img, .rnd, .tc) or they can have no file extension at all. TrueCrypt ignores file extensions. If you need plausible deniability, make sure your TrueCrypt volumes do not have the .tc file extension (this file extension is 'officially' associated with TrueCrypt).

When formatting a hard disk partition as a TrueCrypt volume, the partition table (including the partition type) is *never* modified. If you intend to use a TrueCrypt partition and you need plausible deniability, follow these steps (Windows XP):

1) Make sure you have administrator privileges.
2) Right-click the *My Computer* icon on your desktop or in the Start Menu and select *Manage.*
3) In the list (on the left) click *Disk Management* (within the *Storage* sub-tree).
4) If the partition that you want to format as a TrueCrypt volume has already been created, right-click it and select *Delete Partition…* If the partition has not yet been created, continue with step 4)
5) Right-click the free space (should be labeled as *Unallocated*) and select *New Partition…*
6) *New Partition Wizard* should appear now. Follow its instructions. On the Wizard page called '*Assign Drive Letter or Path'* select '*Do not assign a drive letter or drive path'*. Click *Next*.
7) Select *Do not format this partition* and click *Next*.
8) Click *Finish*.
9) The partition now appears to be "reserved" for future use (and future reformatting). As it is unformatted, it can contain any random data, which might, for example, have resided on the hard drive since the last time you repartitioned the hard disk. Therefore, there is no difference between such an *unformatted* partition and a TrueCrypt volume. Now you can format the partition as TrueCrypt (to do that, click *Create Volume* in the main program window and follow the Volume Creation Wizard's instructions).

   Note: If, instead of an *unformatted* partition, you format an NTFS or FAT partition as TrueCrypt, such partition will then appear to be a corrupted NTFS or FAT partition. Such partition is more likely to be suspected to have been encrypted than an unformatted partition (described above).

The timestamp of a file-hosted container (date and time that the container was last accessed, and last modified) is never updated when TrueCrypt accesses the container (i.e., after dismounting, attempting to mount, changing or attempting to change the password, or creating a hidden volume within it).

Remark: If you use the Windows File Properties tool to view a container timestamp, you will alter the date and time that the container was last accessed.

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, when the adversary uses violence). Using a so-called hidden volume allows you to solve such situations in a diplomatic manner without revealing the password to your volume.

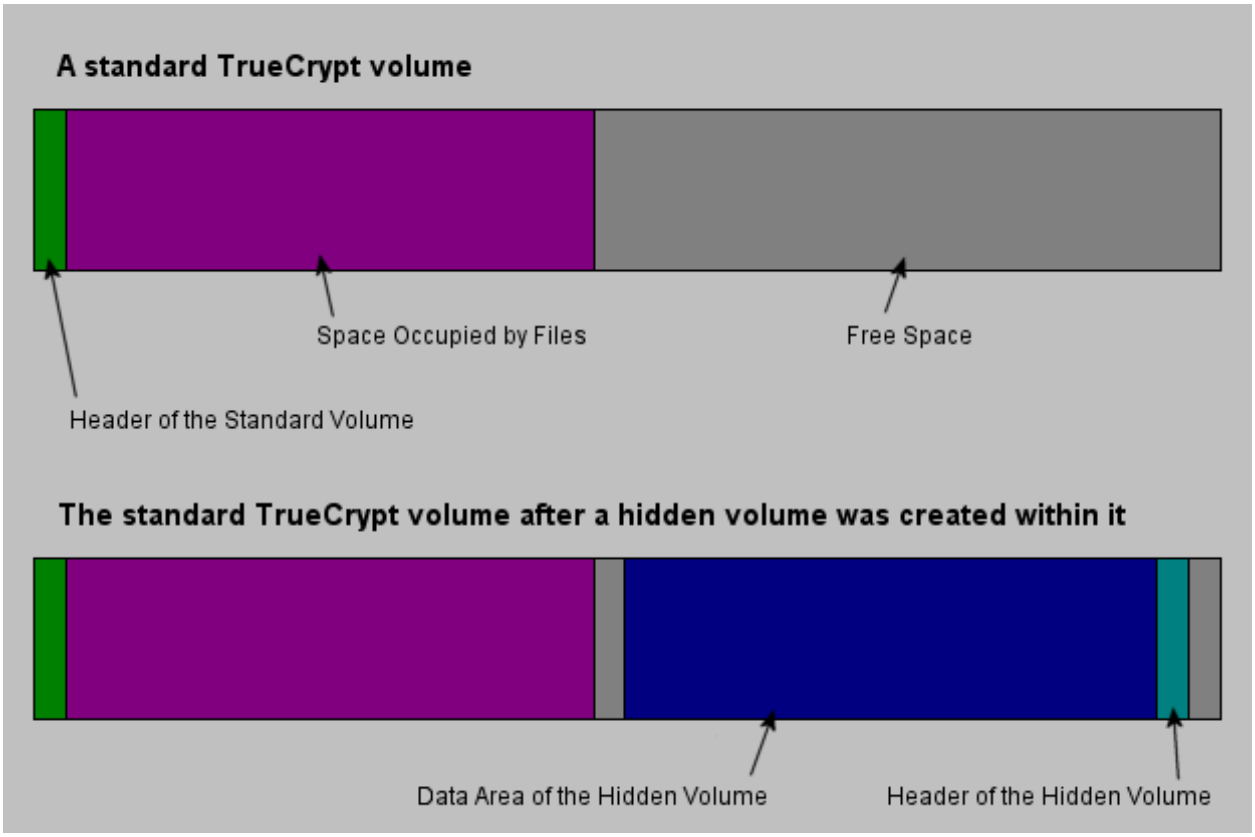

Fig. 2. The layout of a standard TrueCrypt volume before and after a hidden volume was created within it.

The principle is that a TrueCrypt volume is created within another TrueCrypt volume (within the free space on the volume). Even when the outer volume is mounted, it is impossible to tell whether there is a hidden volume within it or not, because free space on *any* TrueCrypt volume is always filled with random data when the volume is created[*] and no part of the hidden volume can be distinguished from random data.

The password for the hidden volume must be 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 do NOT really 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

[*] Right before TrueCrypt volume formatting begins, a temporary encryption key, plaintext block, IV and whitening seeds, are generated by the built-in random number generator (all these items are stored in RAM and are discarded after formatting finishes). The encryption algorithm that the user selected is initialised with the temporary key and the ciphertext blocks it produces are used to fill (overwrite) the free space on the volume. IVs are generated as usual (see *Modes of Operation*) except that the IV seed is not retrieved from the volume header but is generated by the random number generator. Whitening is also applied as usual (see *Whitening*) but the whitening values are derived from the value generated by the random number generator.

for the hidden one. Files that are really sensitive will be stored on the hidden volume.

The user can mount the hidden volume the same way as a standard TrueCrypt volume. TrueCrypt first attempts to decrypt the standard volume header using the entered password. If it fails, it attempts to decrypt the location within the volume where hidden volume headers are normally stored (the third sector from the end of the volume) using the entered password again. If successful, the information about the size and the position of the hidden volume within the outer volume is retrieved from the header, and the hidden volume is mounted.

A hidden volume can be created within any type of TrueCrypt volume, i.e., within a file-hosted container or within a partition/device (requires administrator privileges). To create a hidden TrueCrypt volume, click on 'Create Volume' in the main program window and select "Create a hidden TrueCrypt volume". The Wizard provides help and all information necessary to successfully create a hidden TrueCrypt volume.

As it is 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 any data on the outer volume (especially when the files are fragmented), 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.
Remark: The wizard scans the cluster bitmap to determine the size of the uninterrupted block of free space (if there is any) whose end is aligned with the end of the outer volume. This block accommodates the hidden volume and therefore the size of this block limits the maximum possible size of the hidden volume.

Warning: We recommend that you do not create or copy any more files to outer volume once you create a hidden volume within it. If you do, you may overwrite and damage the hidden volume!
It is, however, possible to open, delete, and rename files stored on the outer volume (after a hidden volume is created within), and it is possible to move files from one folder on the outer volume to another folder on the outer volume.

A hidden volume can only be created within a FAT TrueCrypt volume (the file system of the outer volume must be either FAT12, FAT16, or FAT32). NTFS file system stores various data throughout the entire volume (as opposed to FAT) leaving no room for the hidden volume. Therefore, the Volume Creation Wizard prevents the user from selecting NTFS as the file system for the outer volume. The hidden volume can contain any file system you like and the outer volume (when file-hosted) can be stored on any file system as well.
Note: Should you be asked why the file system of the outer volume is FAT, you can answer that you left all settings at default (FAT is the default file system for any TrueCrypt volume).

Warning: If an adversary can make a copy of a (dismounted) TrueCrypt volume at several points over time, he will 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 update/delete/rename/move files stored on the hidden volume) and the adversary compares the whole host volume with an older copy of it that does not contain these changes, then (after being given the password to the outer volume) he might demand an explanation why these sectors changed. Your failure to provide a plausible explanation might cause the adversary to suspect that the volume contains a hidden volume.

Remark: The timestamp of a file-hosted container (date and time that the container was last accessed, and last modified) is never updated when TrueCrypt accesses the container (i.e., after dismounting, attempting to mount, changing or attempting to change the password, etc.), which applies both to hidden and normal volumes.

Main Program Window

Select File

Allows you to select a file-hosted TrueCrypt volume. After you select it, you can mount it by clicking ‘Mount’ (see below). It is also possible to select a volume by dragging its icon to the ‘TrueCrypt.exe’ icon (TrueCrypt will be automatically launched then).

Select Device

Allows you to select a TrueCrypt partition or a storage device (such as floppy disk or ZIP disk). After it is selected, it can be mounted by clicking ‘Mount’ (see below). Instead of clicking ‘Select Device*’,* you can select a container to mount by dragging its icon over the icon/alias of *TrueCrypt.exe*.

Note: There is a more comfortable way of mounting TrueCrypt partitions – see ‘Auto-Mount Partitions’ for more information.

Mount

To mount a TrueCrypt volume, select a free drive letter from the list in the main window. Then select a file or device that hosts the TrueCrypt volume and click ‘Mount’. TrueCrypt will try to mount the volume using cached passwords (if there are any) and if none of them works, it asks you to enter a password. If you enter the correct password, the volume will be mounted.

Important: Note that switching users or logging off under Windows XP/2000/2003 does not dismount a successfully mounted TrueCrypt volume. Also note that when you exit the TrueCrypt application, the TrueCrypt driver continues working and no TrueCrypt volume is dismounted.

Auto-Mount Partitions

This function allows you to mount TrueCrypt partitions without having to select them manually (by clicking ‘Select Device’). TrueCrypt goes through all available partitions (on all hard drives) one by one and tries to mount each of them as a TrueCrypt volume. Note that TrueCrypt partition cannot be identified, nor the cipher it has been encrypted with. Therefore, the program cannot directly “find” TrueCrypt partitions. Instead, it has to try mounting each (even unencrypted) partition 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. Drive letters will be assigned starting from the one that is selected in the drive list in the main window. If the password you enter is not correct, mounting is tried using cached passwords (if there are any). If you enter empty password, only the cached passwords will be used when attempting to mount partitions.

Dismount

To dismount a TrueCrypt volume basically means to make any access to the data it contains impossible. To do so, select a TrueCrypt volume and click on ‘Dismount’.

Dismount All

Dismounts all currently mounted TrueCrypt volumes.

Wipe Cache

Clears any passwords cached in driver memory. When there are no passwords in the cache, this button is disabled. Up to last four successfully mounted TrueCrypt volume passwords can be cached. This allows mounting volumes without having to type their passwords repeatedly. Passwords are *never* saved on any disk – they are only temporarily stored in RAM. Driver memory is never swapped to disk. Password caching can be enabled/disabled in the Preferences (Tools menu).

Change Password

Allows changing the password of the currently selected TrueCrypt volume (no matter whether the volume is hidden or standard). The main encryption key remains unchanged. Therefore, reformatting is not necessary and is *not* performed (i.e., *no* data will be lost after changing the password and the password change will only take a few seconds). To change a TrueCrypt volume password, click on 'Select File' or 'Select Device', then select the volume, and click on 'Change Password'.

PKCS-5 PRF Algorithm: When changing a volume password, you can also select the HMAC hash algorithm that will be used in deriving the new volume header key (for more information, see *Header Key Derivation, Salt, and Iteration Count*) and in generating the new salt (for more information, see *Random Number Generator*).

Never Save History

If checked, the file names and paths of the last eight mounted volumes will not be saved in the History (which can be displayed by clicking on the Volume combo-box). Note that checking this option does not prevent Windows from saving file selector history of last used items (file containers). To avoid using the file selector, do not click 'Select File' but select the container by dragging its icon to the 'TrueCrypt.exe' icon (TrueCrypt will be automatically launched then).

Exit

Terminates the TrueCrypt application. The driver continues working; no TrueCrypt volumes are dismounted.

Cache Password in Driver Memory

When checked, the volume password you enter will be cached in driver memory (if the password is correct). Then, later, volumes can be mounted using the cached password without having to type it again. Up to four passwords can be cached. The passwords are *never* saved on any disk. They are only temporarily stored in RAM. Driver memory is never swapped to disk. Note that turning the password cache off will not clear it (click *Wipe Cache* to do so).

Program Menu

Note: Only the menu items that are not self-explanatory are described in this documentation.

File -> Exit

Terminates the TrueCrypt application. The driver continues working and no TrueCrypt volumes are dismounted.

Tools -> Clear Volume History

Clears the list containing file names and paths of the last eight successfully mounted TrueCrypt volumes.

Tools -> Preferences

Wipe cached passwords on exit

If enabled, passwords cached in driver memory will be cleared when exiting TrueCrypt.

Cache passwords in driver memory

When checked, up to last four successfully mounted TrueCrypt volume passwords will be cached in driver memory. Then, later, volumes can be mounted using a cached password without having to type it again. A cached password is *never* saved to a disk. It is only temporarily stored in RAM. TrueCrypt driver memory is never swapped to disk.

Open Explorer window for successfully mounted volume

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

Close all Explorer windows of volume being dismounted

Sometimes, dismounting a TrueCrypt volume is not possible due to the fact that some files or folders located on the volume are in use or “locked”. This also applies to Explorer windows displaying directories located on TrueCrypt volumes. When this option is checked, all such windows will be automatically closed before dismounting, so that the user does not have to close them manually.

Command Line Usage

/help or */?*	Displays command line help.
/volume or */v*	File and path name of a TrueCrypt volume. To mount a hard disk partition, use, for example, `/v \Device\Harddisk1\Partition3` (to determine the path to a partition, run TrueCrypt and click *Select Device*).
/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*	Opens an Explorer window after a volume has been mounted.
/beep or */b*	Beeps after a volume has been mounted.
/auto or */a*	Automatically mounts the volume.
/dismount or */d*	Dismounts the given volume (specified by its drive letter) or when no volume is specified, dismounts all currently mounted TrueCrypt 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).
/cache or */c*	Enables (Y) or disables (N) the password cache. Note that turning the password cache off will not clear it.
/history or */h*	Enables (Y) or disables (N) the history.
/wipecache or */w*	Wipes any passwords cached in the driver memory.
/password or */p*	The volume password. If the password contains spaces, it must be enclosed in quotation marks (e.g., `/p "My Password"`). *Warning: This method of entering a volume password is not secure, particularly when you save the password in an unecrypted batch file, or when an unencrypted command prompt history log is being saved to disk. Consider using `/q /a` instead.*
/quiet or */q*	Quiet mode. When used along with */auto* and if no cached password is correct, the password prompt appears (the main TrueCrypt window is not displayed). This could increase the level of privacy in multi-user environments. Program settings are not saved when in quiet mode.

Syntax

```
truecrypt [/v volume] [/d [letter]] [/l letter] [/e] [/b] [/p password]
[/h {Y|N}] [/q] [/c {Y|N}] [/w] [/a] [/f]
```

The order of the parameters is not important. Whitespaces between parameters and parameter values do not matter.

Examples

Mounting a volume called 'myvolume.tc' using the password ‘MyPassword’, as the drive letter X, TrueCrypt will open an explorer window and beep, mounting will be automatic:

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

Mounting a volume called ' myvolume.tc' as the first free drive letter, using the password prompt (the main program window will not be displayed):

```
truecrypt /v myvolume.tc /a /q
```

Encryption Algorithms

TrueCrypt volumes can be encrypted using one of the following algorithms:

Algorithm	Designer(s)	Key Size (Bits)	Block Size (Bits)	Mode
AES	J. Daemen, V. Rijmen	256	128	CBC
Blowfish	B. Schneier	448	64	CBC
CAST5	C. Adams, S. Tavares	128	64	CBC
Serpent	R. Anderson, E. Biham, L. Knudsen	256	128	CBC
Triple DES	IBM, NSA	3*56	64	Outer-CBC
Twofish	B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Ferguson	256	128	CBC
AES-Blowfish		256; 448	128; 64	Inner-CBC
AES-Blowfish-Serpent		256; 448; 256	128; 64; 128	Inner-CBC
AES-Twofish		256; 256	128	Outer-CBC
AES-Twofish-Serpent		256; 256; 256	128	Outer-CBC
Serpent-AES		256; 256	128	Outer-CBC
Serpent-Twofish-AES		256; 256; 256	128	Outer-CBC
Twofish-Serpent		256; 256	128	Outer-CBC

A random value, unique to each sector and volume, is used as the IV (for more information, please see *Modes of Operation*).

AES

The Advanced Encryption Standard 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 (unclassified) information [3]. 256-bit key, 14 rounds (AES-256, published in 2001).

In June 2003, after the NSA (US National Security Agency) has conducted a review and analysis of AES, the U.S. CNSS (Committee on National Security Systems) announces in [2] that the

design and strength of AES-256 (and AES-192) is 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 (IA) requirements associated with the protection of national security systems and/or national security information [2].

Blowfish

Designed by Bruce Schneier in 1993. Unpatented, license-free, available free for all uses. TrueCrypt uses Blowfish with 16 rounds and a 448-bit key. Blowfish is the fastest of the implemented ciphers.

CAST5

CAST5, alias CAST-128, was designed by Carlisle Adams and Stafford Tavares, and published in 1997. 128-bit key, 64-bit block. This encryption algorithm is described in U.S. patent number 5,511,123 [1]. However, CAST5 is royalty-free both for commercial and non-commercial uses [6]. It is also one of the encryption algorithms that are officially used by the Canadian government to cryptographically protect sensitive (unclassified) information [17].

Serpent

Designed by Ross Anderson, Eli Biham, and Lars Knudsen. Published in 1998. 256-bit key, 128-bit block. 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]. 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 may 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 and Rijndael-128 had the lowest safety factor of the finalists: 1.11.

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 conference, Rijndael got 86 votes, Serpent got 59 votes, Twofish 31 votes, RC6 23 votes and MARS 13 votes [18, 19].[*]

Triple DES

Triple DES, published in 1978, is three iterations (encrypt-decrypt-encrypt) of the DES cipher designed by IBM and NSA (in 1976). Outer-CBC mode [16] and three independent 56-bit keys are used (1 per iteration) [13]. Note that this cipher is very slow.

[*] 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].

Twofish

Designed by Bruce Schneier, David Wagner, John Kelsey, Niels Ferguson, Doug Whiting, and Chris Hall. Published in 1998. 256-bit key, 128-bit block. Twofish was one of the AES finalists.

AES-Blowfish

Two ciphers in a cascade [15, 16] operating in inner-CBC mode. Each sector is first encrypted with Blowfish (448-bit key, 64-bit block) and then with AES (256-bit key, 128-bit block). Each of the cascaded ciphers uses its own key. All keys are mutually independent (note that header keys are independent as well, even though they are derived from one password – see *Header Key Derivation, Salt, and Iteration Count*). See above for information on the individual cascaded ciphers.

AES-Blowfish-Serpent

Three ciphers in a cascade operating in inner-CBC mode. Each sector is first encrypted with Serpent (256-bit key, 128-bit block), then with Blowfish (448-bit key, 64-bit block), and finally with AES (256-bit key, 128-bit block). Each of the cascaded ciphers uses its own key. All keys are mutually independent (note that header keys are independent as well, even though they are derived from one password – see *Header Key Derivation, Salt, and Iteration Count*). See above for information on the individual cascaded ciphers.

AES-Twofish

Two ciphers in a cascade operating in outer-CBC mode. Each 128-bit block is first encrypted with Twofish (256-bit key) and then with AES (256-bit key). Each of the cascaded ciphers uses its own key. All keys are mutually independent (note that header keys are independent as well, even though they are derived from one 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 operating in outer-CBC mode. Each 128-bit block is first encrypted with Serpent (256-bit key), then with Twofish (256-bit key), and finally with AES (256-bit key). Each of the cascaded ciphers uses its own key. All keys are mutually independent (note that header keys are independent as well, even though they are derived from one password – see *Header Key Derivation, Salt, and Iteration Count*). See above for information on the individual cascaded ciphers.

Serpent-AES

Two ciphers in a cascade operating in outer-CBC mode. Each 128-bit block is first encrypted with AES (256-bit key) and then with Serpent (256-bit key). Each of the cascaded ciphers uses its own key. All keys are mutually independent (note that header keys are independent as well, even though they are derived from one password – see *Header Key Derivation, Salt, and Iteration Count*). See above for information on the individual cascaded ciphers.

Serpent-Twofish-AES

Three ciphers in a cascade operating in outer-CBC mode. Each 128-bit block is first encrypted with AES (256-bit key), then with Twofish (256-bit key), and finally with Serpent (256-bit key). Each of the cascaded ciphers uses its own key. All keys are mutually independent (note that header keys are independent as well, even though they are derived from one password – see *Header Key Derivation, Salt, and Iteration Count*). See above for information on the individual cascaded ciphers.

Twofish-Serpent

Two ciphers in a cascade operating in outer-CBC mode. Each 128-bit block is first encrypted with Serpent (256-bit key) and then with Twofish (256-bit key). Each of the cascaded ciphers uses its own key. All keys are mutually independent (note that header keys are independent as well, even though they are derived from one password – see *Header Key Derivation, Salt, and Iteration Count*). See above for information on the individual cascaded ciphers.

Supported Operating Systems

TrueCrypt runs on the following operating systems:

- Windows 2003
- Windows XP
- Windows 2000

We recommend Windows XP (at least Service Pack 1) or later as the optimum environment for running TrueCrypt.

Windows 64-Bit Edition is currently not supported.

Remark: TrueCrypt also runs on the Windows "Longhorn" operating system (beta version of the future successor to Windows XP).

TrueCrypt System Files

WindowsPath`\TrueCryptSetup.exe` (uninstaller)

WindowsPath`\SYSTEM32\DRIVERS\truecrypt.sys` (driver)

Note: Replace *WindowsPath* with your Windows installation path (e.g., `C:\WINDOWS`)

Uninstalling TrueCrypt

To uninstall TrueCrypt, open the Windows Control Panel and select 'Add/Remove Programs', locate TrueCrypt and click the 'Add/Remove' button.

Normally, all TrueCrypt files except the uninstaller (*`WindowsPath`*`\TrueCryptSetup.exe`) should be removed, and most of the changes made to the registry should be undone.
No TrueCrypt volume will be removed – you will be able to mount your TrueCrypt volume(s) again after you install TrueCrypt.

Incompatibilities

We are currently not aware of any incompatibilities.

Known Issues & Limitations

- Network volumes are not supported (the contents of a mounted TrueCrypt volume can be shared over a network but it is not possible to mount a TrueCrypt volume stored on a network volume).

- TrueCrypt-encrypted floppy disks: When a floppy disk is ejected and another one is inserted, garbage will be read/written to the disk, which could lead to data corruption. This affects *only raw* floppy disk volumes (not file-hosted TrueCrypt containers stored on floppy disks).

Troubleshooting

This section presents possible solutions to common problems that you may run into when using TrueCrypt. If your problem is not listed here, it might be listed in one of the following sections:

Incompatibilities
Known Issues & Limitations
Frequently Asked Questions

PROBLEM:

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

POSSIBLE SOLUTION:

First make sure that you are not trying to encrypt the operating system boot partition (TrueCrypt does not support this). Then 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 *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.

PROBLEM:

One of the following problems occurs:

1. *A TrueCrypt volume cannot be mounted*
2. *NTFS TrueCrypt 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 TrueCrypt. The operating system reports to TrueCrypt that the device is locked for an exclusive access by an application (so TrueCrypt 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 (such as PartitionMagic), etc.

Frequently Asked Questions

The most recent version of the TrueCrypt FAQ is available at:
http://truecrypt.sourceforge.net/faq.php

Q: *I forgot the password – is there any way to recover the files from my TrueCrypt volume?*

A: *TrueCrypt does not contain any mechanism or facility that would allow partial or complete recovery of your encrypted data without knowing the correct password or the key used to encrypt the data. 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 on the key size, on the software/hardware efficiency, and on other factors.*

Q: *Is my password stored somewhere?*

A: *No.*

Q: *Is some hash of my password stored somewhere?*

A: *No.*

Q: *Is the key size limited to 160 bits when I use HMAC-SHA-1 or HMAC-RIPEMD-160?*

A: No, TrueCrypt 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.*

Q: *What is the maximum possible size of a TrueCrypt volume?*

A: *TrueCrypt volumes can be up to 9223372036 GB. However, you need to take into account the following: the limitations of the file system that the container will be stored on, the limitations of the file system of the container itself, the hardware connection standard, and your operating system limitations. Remember that file-hosted containers stored on the FAT32 file system cannot be larger than 4 GB (if you need a larger volume, store it on the NTFS file system or, instead of creating a file-hosted volume, encrypt a partition). Also note that any FAT32 volume, encrypted or not, cannot be larger than 2048 GB (if you need larger volumes, format them as NTFS).*

Q: *Which cipher is the most secure?*

A: *Unfortunately, it is impossible to answer this question. However, all ciphers implemented in TrueCrypt are well known and trusted. No weak cipher has been implemented in TrueCrypt.*

Q: *Is it possible to install an application to a TrueCrypt volume and run it from there?*

A: *Yes.*

Q: *How does TrueCrypt verify that the correct password was entered?*

See the section *Technical Details – Encryption Scheme*.

Q: *Does TrueCrypt support hardware/software RAID and dynamic volumes?*

A: *Yes, it does. If you intend to format a dynamic volume as a TrueCrypt volume, keep in mind that after you create the 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' window of the TrueCrypt Volume Creation Wizard. Also note that, in the 'Select Device' window, a dynamic volume is not displayed as a single device. Instead, all the volumes that the dynamic volume consists of are displayed and you can select any of them in order to format the entire dynamic disk.*

Q: *Is it possible to mount a TrueCrypt container that is stored on a CD or DVD?*

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

Q: *What will happen if I format a TrueCrypt partition?*

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

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

A: *Yes, when mounted, TrueCrypt volumes can be formatted as FAT12, FAT16, FAT32, or NTFS. The volumes behave as standard disk devices so you can right-click the device icon (for example in 'My Computer' list) and select 'Format'. The actual volume contents will be lost, the whole volume will remain encrypted though. If you format a TrueCrypt-encrypted partition when the TrueCrypt 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).*

Q: *Is it possible to change the password for a 'hidden' volume?*

A: *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: TrueCrypt first attempts to decrypt the standard volume header and if it fails, it attempts to decrypt the location 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.)

Q: *How do I decrypt a TrueCrypt partition permanently?*

A: *If you format a TrueCrypt-encrypted partition when the TrueCrypt volume hosted by the partition is not mounted, then the volume will be destroyed and the partition will not be encrypted anymore (it will be empty). Note that the contents of the TrueCrypt volume will be lost.*

Q: *How do I burn a TrueCrypt container larger than 2 GB onto a DVD?*

A: *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 over 2 GB).*

Q: *How do I encrypt a partition without losing the data currently stored on it?*

A: *Unfortunately, TrueCrypt does not allow this, and we do not plan to implement such feature either. The reason being that we consider this technique potentially insecure (and methods to make it secure would decrease the encryption speed unacceptably). For more information, refer to [21].*

Q: *Can I use tools like chkdsk, Defrag, etc. on the contents of a TrueCrypt volume?*

A: *Yes, TrueCrypt 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 TrueCrypt volume.*

Q: *Is it possible to encrypt my operating system boot partition?*

A: *No, TrueCrypt does not allow this. However, there are ways to ensure that the volume where operating system resides is read-only, which should prevent information leakage (registry, temporary files, etc., are stored in RAM) and make it impossible for an adversary to install a Trojan horse on the system. One of the ways is using BartPE. 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 disk is not used at all and does not even have to be present). The freeware Bart's PE Builder can transform any Windows installation CD into BartPE. In order to include certain applications on a BartPE disk (to make it available to the BartPE system), you need a plug-in for the application. There are several free TrueCrypt plug-ins for BartPE available on the Internet. The type of the CD or DVD on which you store BartPE should be "write once, read many" (for example CD-R), because rewritable disk types (such as CD-RW) might allow an adversary to alter the contents of the disk.*

Q: *Can I mount a TrueCrypt volume stored on another TrueCrypt volume?*

A: *Yes, TrueCrypt volumes can be nested without any limitation.*

Q: *Can I run TrueCrypt with another on-the-fly disk encryption tool on one system?*

A: *We are not aware of any on-the-fly encryption tool that would cause problems when run with TrueCrypt, or vice versa.*

Q: *Can I resize a TrueCrypt partition?*

A: *Unfortunately, TrueCrypt does not support this. Resizing a TrueCrypt partition using a program such as PartitionMagic will, in most cases, corrupt its contents.*

Q: *Can I use TrueCrypt if I do not have administrator privileges?*

A: *Users without administrator privileges can mount, dismount and create TrueCrypt volumes. They cannot encrypt/format partitions (they can only create file containers), cannot create NTFS volumes, and cannot install/uninstall TrueCrypt.*

Q: *Why is it not possible to create arbitrary cascades?*

A: *The reason is that the encryption algorithm (and the mode of operation) that a TrueCrypt volume has been encrypted with is unknown. The correct encryption algorithm has to be determined through the process of trial and error. If we added the support for creating arbitrary cascades, the number of encryption algorithms to attempt mounting with would increase tremendously. The time needed to mount a volume would no longer be acceptable especially on slow PCs.*

Q: *What will happen when a part of a TrueCrypt volume becomes corrupted?*

A: *Data within each sector (sector is 512 bytes) are chained (see '*Modes of Operation*') so when a block becomes corrupted, each successive block within the sector will also become corrupted (block size is either 8 or 16 bytes, depending on the encryption algorithm). Corrupting the first sector (volume header) will, in most cases, make the volume impossible to mount.*

Q: *Will I always be able to mount a TrueCrypt container no matter how fragmented it is?*

A: Yes.

Q: *Is it necessary to restart the computer before copying a TrueCrypt container?*

A: *No, it is not necessary.*

Q: *Is it secure to create a new container by cloning an existing container?*

A: *You should always use the Volume Creation Wizard to create a new TrueCrypt volume. If you copy a container and then start using both this container and its clone in a way that both eventually contain different data, then you could aid cryptanalysis. The reason is that both volumes would share the same key, IVs, whitening values, etc.*

Q: *Do I have to "wipe" free space and/or files on a TrueCrypt volume?*

["Wiping" – secure deletion; overwriting sensitive data in order to render them unrecoverable.]

A: *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.*

Q: *Does TrueCrypt support something like 'traveller' mode?*

A: *Currently not (however, it is planned to be implemented in a future version). If you need to use TrueCrypt on a computer on which you cannot (or do not want to) install TrueCrypt, we recommend using BartPE for this purpose. For further information regarding BartPE, see the question "Is it possible to encrypt my operating system boot partition?" in this FAQ.*

Q: *How is TrueCrypt related to E4M?*

A: *TrueCrypt 1.0 was derived from E4M 2.02a. For information on differences between E4M and TrueCrypt, please see Version History*.

Q: *Will TrueCrypt be open-source and free forever?*

A: *Yes, it will. No commercial version is planned and never will be. We believe in open-source and free security software.*

Remark: We know that there are certain individuals trying to distribute TrueCrypt as paid and closed-source software. We are not affiliated with these individuals.

Technical Details

Notation

C	Ciphertext block
$D_K()$	Decryption algorithm using decryption key K
$E_K()$	Encryption algorithm using encryption key K
$H()$	Hash function (e.g., RIPEMD-160)
i	Block index for n-bit blocks; n is context-dependent
K	Encryption/decryption key
m	Block index for 64-bit blocks
n	Block index for 128-bit blocks
P	Plaintext block
R	Randomness pool
W	Whitening value
^	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: $1 \oplus 0 = 1$; $1 \oplus 1 = 0$; $1 \oplus 2 = 1$; $1 \oplus 3 = 0$; $0 \oplus 0 = 0$; $0 \oplus 1 = 1$; $0 \oplus 2 = 0$; $0 \oplus 3 = 1$)
||	Concatenation

Encryption Scheme

When mounting a TrueCrypt volume (assume there are no cached passwords), the following steps are performed:

1. The first 512 bytes (i.e., the standard volume header) of the volume are read into RAM, out of which the first 64 bytes are the salt (see *TrueCrypt Volume Format Specification*).

2. 512 bytes at byte 1536 (offset) from the end of the volume are read into RAM (see *TrueCrypt Volume Format Specification*). If there is a hidden volume within this volume, at this point we have read its header (whether or not there is a hidden volume has to be

determined by attempting to decrypt this data).

3. Now, TrueCrypt attempts to decrypt the standard volume header and the candidate for the hidden volume header, read in (1) and (2). All data used and generated in the course of the process of decryption are kept in RAM (never saved to disk). The following parameters are unknown[*] 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 in deriving the header key (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-1, HMAC-RIPEMD-160.
    A password entered by the user and the salt read in (1) or (2) 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 IV (used to decrypt the volume header) are derived.
    b. Encryption algorithm(s):
    AES-256, Blowfish, CAST5, Serpent, Triple DES, Twofish
    c. Number of ciphers to use (whether a cascade or single encryption).
    d. Mode of operation: CBC, inner-CBC, outer-CBC
    e. Block size(s)
    f. 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 the 8th byte of the decrypted data (this value is unknown to an adversary because it is encrypted – see *TrueCrypt Volume Format Specification*). If these conditions are not met, mounting is terminated (either a wrong password, a corrupted volume, or not a TrueCrypt volume).

5. Now we know (or assume, with very high probability) that we have the correct password, the correct encryption algorithm, mode, key size, block size, and the correct header key derivation algorithm. We also know whether we are mounting a hidden volume or not. The minimum program version required to open the volume, stored in the decrypted volume header is checked. If it is not equal or less than the version of the program that we are using to mount the volume, mounting is terminated.

6. The encryption routine is reinitialised with the master key[†] and IV retrieved from the decrypted volume header. This key can be used to decrypt any sector of the volume, except the volume header area (which has been encrypted using the header key). The volume is mounted.

See also *Modes of Operation, Whitening,* and *Header Key Derivation, Salt, and Iteration Count*.

[*] These parameters are kept secret not in order to increase the complexity of an attack, but primarily to make TrueCrypt volumes unidentifiable (undistinguishable from random data), which would be difficult to achieve if these parameters were stored within the volume header.

[†] The master key was 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 of Operation

The following table lists all encryption algorithms implemented in TrueCrypt and the modes in which they operate:

Encryption Algorithm	Mode of Operation	Details of the Mode of Operation
AES[*]	CBC	$C_i = E_K(P_i \oplus C_{i-1})$; C_0 is IV.
AES-Blowfish ($E2$) ($E1$)	Inner-CBC	$C_n = E2_{K2}((S_m \parallel S_{m+1}) \oplus C_{n-1})$; $S_m = E1_{K1}(P_m \oplus S_{m-1})$; C_0 and S_0 are IVs.
AES-Blowfish-Serpent ($E3$) ($E2$) ($E1$)	Inner-CBC	$C_n = E3_{K3}((S_m \parallel S_{m+1}) \oplus C_{n-1})$; $S_m = E2_{K2}(T_m \oplus S_{m-1})$; $T_m \parallel T_{m+1} = Q_n = E1_{K1}(P_n \oplus Q_{n-1})$; C_0, S_0, and Q_0 are IVs.[†]
AES-Twofish ($E2$) ($E1$)	Outer-CBC	$C_i = E2_{K2}(E1_{K1}(P_i \oplus C_{i-1}))$; C_0 is IV.
AES-Twofish-Serpent ($E3$) ($E2$) ($E1$)	Outer-CBC	$C_i = E3_{K3}(E2_{K2}(E1_{K1}(P_i \oplus C_{i-1})))$; C_0 is IV.
Blowfish	CBC	$C_i = E_K(P_i \oplus C_{i-1})$; C_0 is IV.
CAST5	CBC	$C_i = E_K(P_i \oplus C_{i-1})$; C_0 is IV.
Serpent	CBC	$C_i = E_K(P_i \oplus C_{i-1})$; C_0 is IV.
Serpent-AES ($E2$) ($E1$)	Outer-CBC	$C_i = E2_{K2}(E1_{K1}(P_i \oplus C_{i-1}))$; C_0 is IV.
Serpent-Twofish-AES ($E3$) ($E2$) ($E1$)	Outer-CBC	$C_i = E3_{K3}(E2_{K2}(E1_{K1}(P_i \oplus C_{i-1})))$; C_0 is IV.
Triple DES	Outer-CBC	$C_i = E_{K3}(D_{K2}(E_{K1}(P_i \oplus C_{i-1})))$; C_0 is IV.
Twofish	CBC	$C_i = E_K(P_i \oplus C_{i-1})$; C_0 is IV.
Twofish-Serpent ($E2$) ($E1$)	Outer-CBC	$C_i = E2_{K2}(E1_{K1}(P_i \oplus C_{i-1}))$; C_0 is IV.

[*] In this table, we use the term "AES" to refer to "AES-256".
[†] $T_m \parallel T_{m+1} = Q_n$ denotes a 128-bit block Q_n being split into two 64-bit blocks T_m and T_{m+1}.

IV (initialisation vector) is always a random value (unknown to an adversary), which is unique to each sector* and volume. This value is generated as follows:

1. Bytes 256-263 (for a 128-bit block cipher, bytes 256-271) of the decrypted volume header are retrieved (see the sections *TrueCrypt Volume Format Specification* and *Encryption Scheme*). If ciphers in a cascade use more than one IV, the successive bytes are retrieved for the additional IVs (for example, for three 128-bit block ciphers in a cascade in inner-CBC mode, bytes 256-271 are retrieved for the first IV, bytes 272-287 for the second, and 288-303 for the third IV).

2. Data retrieved in (1) are XORed with the 64-bit sector number (each sector is 512 bytes long; sectors are numbered starting at 0). In case of a 128-bit block cipher, the upper and lower 64-bit words of the 128-bit value retrieved in (1) are XORed with an identical value. The resultant 64-bit value (or 128-bit for 128-bit block ciphers) is the IV.

   i.e.,
   For a 128-bit block cipher:
   T_1 = upper 64-bit word of the value retrieved in (1)
   T_2 = lower 64-bit word of the value retrieved in (1)
   S = sector number *(64-bit integer)*
   $IV = (T_1 \wedge S) \,||\, (T_2 \wedge S)$

   For a 64-bit block cipher:
   T = 64-bit value retrieved in (1)
   S = sector number *(64-bit unsigned integer)*
   $IV = T \wedge S$

   Note: Step (1) is only performed once, right after the volume is mounted. The retrieved data remain in RAM then.

Ciphers in a cascade use mutually independent keys (note that the header keys they use are independent as well, even though they are derived from one password – see *Header Key Derivation, Salt, and Iteration Count*).

Whitening

To make obtaining a plaintext/ciphertext pair more difficult [16], the following technique, similar to one that is sometimes called whitening, is applied: Every eight bytes of each sector (after the sector is encrypted) are XORed with a 64-bit value, which is unique to each sector and volume (and is unknown to an adversary). The value is generated as follows:

1. Bytes 264-271 (for a 128-bit block cipher, bytes 272-279) of the decrypted volume header are retrieved† (see *TrueCrypt Volume Format Specification* and *Encryption Scheme*).

* Each sector is 512 bytes long; sectors are numbered starting at 0.
† All necessary data are retrieved from the volume header right after the volume is mounted, and they remain in RAM until the volume is dismounted.

2. Bytes 272-279 (for a 128-bit block cipher, bytes 280-287) of the decrypted volume header are retrieved.

3. Data retrieved in (1) are XORed with the 64-bit sector number (each sector is 512 bytes long; sectors are numbered starting at 0).

4. Data retrieved in (2) are XORed with the 64-bit sector number.

5. A 32-bit CRC-32 value of the first 8 bytes of the resultant value in (3) is calculated.

6. A 32-bit CRC-32 value of the second 8 bytes of the resultant value in (3) is calculated.

7. A 32-bit CRC-32 value of the first 8 bytes of the resultant value in (4) is calculated.

8. A 32-bit CRC-32 value of the second 8 bytes of the resultant value in (4) is calculated.

9. The value calculated in (5) is XORed with the value calculated in (8).

10. The value calculated in (6) is XORed with the value calculated in (7).

11. The 32-bit value calculated in (9) is written to the upper 32-bit word and the value calculated in (10) is written to the lower 32-bit word of the 64-bit whitening value.

Remark: The steps 5-8 are performed to increase the Hamming distance between individual whitening values.

Summary:

T_1 = 64-bit value retrieved in (1)
T_2 = 64-bit value retrieved in (2)
S = sector number *(64-bit unsigned integer)*
$T_1 = T_1 \wedge S$
$T_2 = T_2 \wedge S$
Q_1 = upper 32-bit word of T_1
Q_2 = lower 32-bit word of T_1
Q_3 = upper 32-bit word of T_2
Q_4 = lower 32-bit word of T_2
$W = (\text{CRC32}(Q_1) \wedge \text{CRC32}(Q_4)) \,||\, (\text{CRC32}(Q_2) \wedge \text{CRC32}(Q_3))$

Actual whitening is applied as follows:

For a 128-bit block cipher:
T_1 = upper 64-bit word of C
T_2 = lower 64-bit word of C
$C' = (T_1 \wedge W) \,||\, (T_2 \wedge W)$

For a 64-bit block cipher:
$C' = C \wedge W$

Header Key Derivation, Salt, and Iteration Count

Header key is used to decrypt the encrypted area of the TrueCrypt volume header (see the sections *Encryption Scheme* and *TrueCrypt Volume Format Specification*). The technique that TrueCrypt uses to generate header keys conforms to PKCS #5 v2.0 (refer to [7] for details).

A 512-bit salt (random values generated using the built-in random number generator during the volume creation process) is used, which means there are 2^{512} keys for each password. This significantly decreases vulnerability to 'off-line' dictionary attacks (pre-computing all the keys for a dictionary of passwords is very difficult when a salt is used) [7]. Two thousand iterations of the key derivation function have to be performed to derive a header key, which significantly increases the time necessary to perform an exhaustive search for passwords (i.e., brute force attack) [7]. The header key derivation function is based on HMAC-SHA-1 or HMAC-RIPEMD-160 (see [8, 9, 20, 22]) – the user selects which. Neither the quality, nor the length of the derived key is limited by the size of the output of the underlying hash function (the derived key has always the required size, i.e., *not* only 160 bits). For more information, refer to [7].

Random Number Generator

The random number generator implemented in TrueCrypt is used to generate the master encryption key, salt, and the values used to create IV and whitening values (see the section *Modes of Operation*, and *Whitening*).

The random number generator creates a pool of random values in RAM (memory). The pool, which is 256 bytes long, is filled with data from the following sources:

- Mouse movement (CRC32-hashed mouse coordinates, and event delta/absolute times):
  CRC32(*MouseCoordinates*) || CRC32(*EventDeltaTime* || *EventTime*)
- Mouse clicks (CRC32-hashed button IDs, and event delta/absolute times):
  CRC32(*MouseButtonID*) || CRC32(*EventDeltaTime* || *EventTime*)
- Keystrokes (CRC32-hashed key codes and event delta/absolute times):
  CRC32(*KeyID*) || CRC32(*EventDeltaTime* || *AbsoluteEventTime*)
- Performance statistics of disk drives
- Network interface statistics (NETAPI32)
- MS Windows CryptoAPI
- Various Win32 handles, time variables, and counters (collected at 250-ms interval)

A mouse or keystroke event is accepted only if it is different from the last and penultimate events from the respective source. *EventDeltaTime* denotes the time difference between the current and last accepted event from the source. *AbsoluteEventTime* denotes the value of the system timer when the event is accepted.

Before a value obtained from any of the above-mentioned sources is written to the pool, it is split into bytes (e.g., a 32-bit output of CRC-32 is split 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 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. In addition, after a value (byte) is added to the pool, the pool is entirely hashed using a hash function (SHA-1 or RIPEMD-160 – the user selects which).

The design and implementation of the random number generator are based on the following:

- *Software Generation of Practically Strong Random Numbers* by Peter Gutmann [10]
- *Cryptographic Random Numbers* by Carl Ellison [11]

TrueCrypt Volume Format Specification

TrueCrypt volume has no "signature". Until decrypted, it appears to consist of random data entirely. Therefore, it is impossible to identify a TrueCrypt container or partition.

TrueCrypt volume format version 1 specification:

Offset (bytes)	Size (bytes)	Encryption Status	Description
0	64	Not Encrypted	Salt*
64	4	Encrypted	ASCII string "TRUE"
68	2	Encrypted	Volume format version
70	2	Encrypted	Minimum program version required to open the volume
72	4	Encrypted	CRC-32 checksum of the (decrypted) bytes 256-511
76	8	Encrypted	Volume creation time
84	8	Encrypted	Header creation/modification time
92	8	Encrypted	Reserved (set to zero)
100	156	Encrypted	Unused
256	32	Encrypted	Data used to generate IV and whitening values
288	224	Encrypted	Master encryption key
512	*N/A*	Encrypted	Data area (actual volume contents)

If a TrueCrypt volume hosts a hidden volume (within its free space), the header of the hidden volume is located at the byte 1536 (offset) from the end of the host volume (the header of the host/outer volume is located at the beginning of the volume – see the section *Hidden Volume*). The format of the hidden volume header is specified in the following table:

* Note that 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 (bytes)	Size (bytes)	Encryption Status	Description
0	64	Not Encrypted	Salt
64	4	Encrypted	ASCII string “TRUE”
68	2	Encrypted	Volume format version
70	2	Encrypted	Minimum program version required to open the volume
72	4	Encrypted	CRC-32 checksum of the (decrypted) bytes 256-511
76	8	Encrypted	Volume creation time
84	8	Encrypted	Header creation/modification time
92	8	Encrypted	Size of the hidden volume
100	156	Encrypted	Unused
256	32	Encrypted	Data used to generate IV and whitening values
288	224	Encrypted	Master encryption key

The bytes 0-63 (salt), bytes 256-287 (data used to generate IV and whitening values), and bytes 288-511 (master encryption key), contain random values that have been generated using the built-in random number generator (see the section *Random Number Generator)* during the volume creation process.

Free space of each TrueCrypt volume is filled with random data when the volume is created. The random data is generated as follows: Right before TrueCrypt volume formatting begins, a temporary encryption key, plaintext block, IV and whitening seeds, are generated by the built-in random number generator (all these items are stored in RAM and are discarded after formatting finishes). The encryption algorithm that the user selected is initialised with the temporary key and the ciphertext blocks it produces are used to fill (overwrite) the free space on the volume. IVs are generated as usual (see *Modes of Operation*) except that the IV seed is not retrieved from the volume header but is generated by the random number generator. Whitening is also applied as usual (see *Whitening*) but the whitening values are derived from the value generated by the random number generator.

Compliance with Standards and Specifications

TrueCrypt complies with the following standards, specifications, and recommendations:

- PKCS #5 v2.0 [7]
- FIPS 46-3 [13]
- FIPS 197 [3]
- FIPS 198 [22]
- FIPS 180-2 [14]
- NIST S. P. 800-38A [12]

Future Development

The following features are planned for a future release:

- Linux version
- Hash algorithms conceptually distinct from SHA-1 and RIPEMD-160
- Header key derivation PRF algorithms conceptually distinct from HMAC-SHA-1 and HMAC-RIPEMD-160
- Keyfiles
- Auto-dismount
- Hotkeys
- ‘Traveller’ mode
- TrueCrypt API

and more.

For the most recent version of this list, refer to: http://truecrypt.sourceforge.net/future.php

Contact

Information on how to contact us can be found at:
http://truecrypt.sourceforge.net/contact.php

Version History

3.0a

December 11, 2004

Bug fixes:

- Data corruption will not occur when data is written to a volume encrypted with Twofish or Serpent while another TrueCrypt volume is mounted (applies also to volumes encrypted using a cascade of ciphers, out of which one is Twofish or Serpent).
- Other minor bug fixes

3.0

December 10, 2004

New features:

- Ability to create and mount a hidden TrueCrypt volume (file container or partition/device). This allows solving situations where the user is forced by an adversary to reveal the password and cannot refuse to do so (for example, when the adversary uses violence).

  The principle is that a TrueCrypt volume is created within another TrueCrypt volume (within the free space on the volume). Even when the outer volume is mounted, it is impossible to tell whether there is a hidden volume within it or not, because free space on any TrueCrypt volume is always filled with random data when the volume is created and no part of the hidden volume can be distinguished from random data.

  The password for the hidden volume must be 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 do NOT really 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 are really sensitive will be stored on the hidden volume.

  As it is 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 any data on the outer volume, 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.
- Serpent encryption algorithm (256-bit key)
- Twofish encryption algorithm (256-bit key)
- Forced/“brutal” dismount (allows dismounting a volume containing files being used by the system or an application).
- Cascades of ciphers added (e.g., AES-Twofish-Serpent, AES-Blowfish, etc.) Each of the ciphers in a cascade uses its own encryption key (the keys are mutually independent).
- Ability to mount a TrueCrypt volume that is being used by the system or an application (shared access mode).
- Ability to encrypt devices/partitions that are being used by the system or an application.
- The ‘Select Device’ dialog and the ‘Auto-Mount Partitions’ facility now support devices that do not contain any partitions.

- Encryption Algorithm Benchmark facility added to the Tools menu and to the Volume Creation Wizard.
- A warning is displayed if Caps Lock is on when creating a new volume or changing a password.
- When /l is omitted and /a is used, the first free drive letter is used *(command line usage)*
- New command line option: /force or /f enables forced ("brutal") dismount or mounting in shared mode (i.e., without exclusive access).
- Drive letters are now displayed in the 'Select Device' window.

Bug fixes:

- 'Blue screen' errors (system crashes) will not occur when dismounting a volume (remark: this bug was inherited from E4M).
- The 'Select Device' dialog will display also partitions being used by the system or an application.
- If the size of a partition/device was not a multiple of 1024 bytes, its last sector (512 bytes) was not used for TrueCrypt volume (the volume was 512 bytes shorter than the partition/device).
  Remark: This bug was inherited from E4M, so it applies also to encrypted partitions/devices created by E4M.
- FAT volumes that are exactly 129 MB in size will not have zero size of free space (129-MB FAT volumes created by the previous versions had no free space available).
- Users without administrator privileges can now create file containers under Windows Server 2003.
- Other minor bug fixes

Improvements:

- The timestamp of a container (date and time that the container was last accessed, and last modified) will not be updated when TrueCrypt accesses the container (i.e., after dismounting, attempting to mount, changing or attempting to change the password, or creating a hidden volume within it).
- The TrueCrypt Service is no longer necessary and has been removed because its functions are now handled by the TrueCrypt driver.
- When 'Never save history' is checked, Windows is prevented from saving the file names of the last accessed file containers to the 'Recent Documents' and File Selector history.
- Other minor improvements

Miscellaneous:

- TrueCrypt has been successfully tested on the Windows "Longhorn" operating system (beta version of the future successor to Windows XP).
- The user can now override the minimum password length (a warning is displayed and a confirmation is required).

2.1a

October 1, 2004

Removed Features:

- IDEA encryption algorithm removed. This allows non-profit and profit organizations to use TrueCrypt without having to obtain a separate license for IDEA (according to the IDEA license, *any* use of software containing the IDEA algorithm by a non-profit or profit organization is considered as use for commercial purposes, and is subject to a license from MediaCrypt AG).

  Important: TrueCrypt volumes encrypted using the IDEA encryption algorithm cannot be mounted using TrueCrypt 2.1a. If you have such a volume, before upgrading to TrueCrypt 2.1a, please create a new TrueCrypt volume using a cipher other than IDEA and move your files to this new volume.

2.1

June 21, 2004

New features:

- RIPEMD-160 hash algorithm added. The user can now select which hash algorithm TrueCrypt will use (SHA-1 or RIPEMD-160).

  Note: RIPEMD-160, which was designed by an open academic community, represents a valuable alternative to SHA-1 designed by the NSA and NIST. In the previous versions there was a risk that the whole program would be practically useless, should a major weakness be found in SHA-1. The user-selected hash algorithm is used by the random number generator when creating new volumes, and by the header key derivation function (HMAC based on a hash function, as specified in PKCS #5 v2.0). The random number generator generates the master encryption key, salt, and the values used to create IV and 'whitening' values.

- When changing a volume password, the user can now select the HMAC hash algorithm that will be used in deriving the new volume header key.

- It is now possible to create NTFS TrueCrypt volumes and unformatted TrueCrypt volumes. This enhancement also removes the 2048 GB volume size limit. (Only FAT volumes can be created using the previous versions of TrueCrypt. Any FAT volume, encrypted or not, cannot be over 2048 GB.)

- Header key content is now displayed in the Volume Creation Wizard window (instead of salt).

- Random pool, master key, and header key contents can be prevented from being displayed in the Volume Creation Wizard window.

Bug fixes:

- When there is a mounted TrueCrypt container that is stored in another TrueCrypt container, it will be possible to dismount both of them using the 'Dismount All' function, and 'blue screen' errors will not occur upon system shutdown.

- Minor bug fixes to command line handling.

Improvements:

- Several minor improvements to the driver.

Miscellaneous:

- Released under the original E4M license to avoid potential problems relating to the GPL license (added the IDEA patent information and specific legal notices).

2.0

June 7, 2004

Bug fixes:

- Data corruption will no longer occur when a TrueCrypt partition is subjected to heavy parallel usage (usually when copying files to or from a TrueCrypt partition). This also fixes the problem with temporarily inaccessible files stored in TrueCrypt partitions.

  Note: File-hosted volumes were not affected by this bug.

- After dismounting and remounting a volume, its file system will be correctly recognized by the operating system and it will be possible to reuse the same drive letter *(Windows 2000 issue)*.

- The main program window will not be displayed when run in quiet mode *(command line usage)*.

- Two password entry attempts are no longer necessary to be able to mount a volume *(command line usage)*.

- All partitions will be visible to TrueCrypt even if one of them is inaccessible to the operating system (an inaccessible partition made all successive partitions on the hard disk unavailable to TrueCrypt).

- Relative path can be specified when mounting a file-hosted volume *(command line usage).*

- Incorrect passwords are reported when auto-mounting *(command line usage).*

New features:

- AES-256 (Rijndael) encryption algorithm.

- The command line option */dismountall* was renamed to */dismount* which can now be also used to dismount a single volume by specifying its drive letter.

Improvements:

- Memory pages containing sensitive data are now locked to prevent them from being swapped to the Windows page file.

- The state of the random pool will never be exported directly so the pool contents will not be leaked.

Miscellaneous:

- Released under GNU General Public License (GPL)

1.0a *(by TrueCrypt Team)*

February 3, 2004

Removed features:

- TrueCrypt no longer supports Windows 98/ME.

1.0 *(by TrueCrypt Team)*

February 2, 2004

Note: TrueCrypt is based on E4M (Encryption for the Masses). Therefore, the following list contains differences between E4M 2.02a and TrueCrypt 1.0 (minor differences have been omitted).

Improvements:

- Windows XP/2000 support
- The maximum volume size is 18,446,744,073 GB (E4M only allows 2 GB).
  Note: File system, hardware connection standard, and operating system limitations have to be taken into account when determining maximum volume size.
- Plausible deniability. It is impossible to identify a TrueCrypt container or partition. Until decrypted, a TrueCrypt volume appears to consist of nothing more than random data (it does not contain any "signature"). Therefore, it is impossible to prove that a file, a partition or a device is a TrueCrypt volume and/or that it has been encrypted. To achieve plausible deniability, the format of the volume and the encryption process had to be significantly changed.
- The salt is 64 bytes long now (E4M uses 20 bytes).
- The iteration count of the key derivation function increased to 2,000 (E4M uses 1,000).
- Free space is filled with random data during volume creation, instead of filling it with zeroes. This reduces the amount of predictable plaintext and, in future, will increase the level of plausible deniability of hidden volumes.
- Up to 32 partitions per physical disk drive can be encrypted now (Windows XP/2000).
- The minimum volume password length has been increased to 12 characters.
- The maximum volume password length has been decreased from 100 to 64 characters. This was necessary to avoid the following: When a password longer than 64 characters was passed to HMAC-SHA-1, the whole password was first hashed using SHA-1 and the resultant 160-bit value was then used instead of the original password (which complies with HMAC-SHA-1 specification), thus the password length was in fact reduced.
- The Blowfish key length size has been increased to 448 bits.
  Remark: Even though our increasing the key size to 448 bits might appear to be a significant overkill, there was no reason for us not to do so (note that there is no decrease in speed of encryption/decryption).

Bug fixes:

- Sector scrambling algorithm flaw fixed. Two or more disk sectors to be encrypted consisting of the same values (e.g. filled with zeroes), after being encrypted by E4M, start with the same 8-byte

sequence of values (i.e. the first eight bytes of any of these encrypted sectors contain the same values as the first eight bytes of any other of these encrypted sectors). If this had not been fixed, the plausible deniability would not have been possible.

- TrueCrypt volumes can be dismounted (Windows XP issue).
- "Blue screen" errors no longer occur during Windows shutdown when there is one or more mounted TrueCrypt volumes.
- Drive geometry is calculated correctly now (*chkdsk.exe* and *format.exe* do not fail anymore).
- A TrueCrypt volume can be reformatted as FAT32 or NTFS using the Windows built-in format tool (Windows XP/2000 issue).
- Windows Check Disk can now be used on TrueCrypt volumes (Windows XP/2000 issue).
- Windows Disk Defragmenter can now be used on encrypted volumes (Windows XP/2000 issue).

New features:

- New IV (initialization vector) generation algorithm (see the documentation for more information)
- Every 8 bytes of each sector (after the sector is encrypted) are XORed with a random 64-bit value, which is unique to each sector and volume (sector is 512 bytes long). This makes obtaining a plaintext/ciphertext pair a bit more difficult.
- New function to clear the volume history.
- When selecting a partition/device, the sizes and file system types of available partitions/devices are displayed (Windows XP/2000).
- List of mounted TrueCrypt volumes now contains their sizes and encryption algorithms used (Windows XP/2000).
- Free volume space is reported (in 'My Computer' list etc.)
- Windows XP format facilities do not support formatting volumes larger than 32 GB as FAT32. However, with TrueCrypt Volume Creation Wizard it is now possible to create FAT32 volumes larger than 32 GB.
- New function that allows multiple TrueCrypt partitions to be mounted provided that their correct password(s) has/have been entered (this includes the cached passwords, if there are any).
- Quick format (partitions/devices only)
- Cluster size selection (when creating new volumes)
- Volume properties can now be examined (encryption algorithm, volume creation time, last password change time etc.)
- New function to dismount all mounted TrueCrypt volumes.
- New command line options to dismount all mounted TrueCrypt volumes: /d and /dismountall
- HMAC-SHA1 and CRC-32 algorithm tests are now included in the self-test facility.
- Program menu and Preferences window added.

- Custom user interface fonts supported.
- Optionally, the TrueCrypt installer can now create System Restore points (Windows XP/ME).
- Password input field is wiped after a correct volume password has been entered.
- New graphics, icons, user interface
- New documentation

Removed features:

- E4M and SFS volumes are no longer supported.
- DES cipher removed.
- HMAC-MD5 removed (to be replaced by HMAC-RIPEMD-160).

Acknowledgements

We would like to thank the following people:

Paul Le Roux for making his E4M source code available; TrueCrypt is based on E4M.

Eric Young for writing his excellent libdes, libcast, etc., which were the sources of some of the cryptography code used in TrueCrypt.

Dr. Brian Gladman, who wrote the excellent AES, Serpent, and Twofish 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 used in TrueCrypt.

Andy Neville for providing some of the code and inspiration, useful in the implementation of the file-hosted volumes (E4M).

David Kelvin, who added the privacy password command line argument, and the quiet mode (E4M).

The designers of the encryption and hash algorithms:
Horst Feistel, Don Coppersmith, Whitfield Diffie, Martin Hellman, 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, *Hans Dobbertin, Antoon Bosselaers,* and *Bart Preneel.*

All the others who have made this project possible and all who have morally supported us.

Thank you very much.

References

[1] C. Adams, *Symmetric cryptographic system for data encryption*, U.S. Patent 5,511,123, filed August 4, 1994, issued April 23, 1996, available at http://patft.uspto.gov/.

[2] 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://www.nstissc.gov/Assets/pdf/fact%20sheet.pdf.

[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, available at http://csrc.nist.gov/CryptoToolkit/aes/round2/r2report.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/CryptoToolkit/aes/round2/comments/20000515-bschneier.pdf.

[6] C. Adams, *The CAST-128 Encryption Algorithm*, Request for Comments 2144, May 1997, available at http://www.rfc-editor.org/rfc/rfc2144.txt.

[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, IBM, M. Bellare, UCSD, R. Canetti, IBM, *HMAC: Keyed-Hashing for Message Authentication*, Request for Comments 2104, February 1997, available at http://www.rfc-editor.org/rfc/rfc2104.txt.

[9] P. Cheng, IBM, R. Glenn, NIST, *Test Cases for HMAC-MD5 and HMAC-SHA-1*, Request for Comments 2202, February 1997, available at http://www.rfc-editor.org/rfc/rfc2202.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] Morris Dworkin, *Recommendation for Block Cipher Modes of Operation*, NIST Special Publication 800-38A, 2001 Edition, available at http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf.

[13] NIST, *Data Encryption Standard*, Federal Information Processing Standards Publication 46-3, October 25, 1999, available at http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf.

[14] NIST, *Secure Hash Standard*, 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] List of the approved cryptographic algorithms for the protection of Protected Information within the Government of Canada: http://www.cse-cst.gc.ca/en/services/crypto_services/crypto_algorithms.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/CryptoToolkit/aes/round2/comments/20000523-msmid-2.pdf.

[20] A. Menezes, P. van Oorschot, S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, October 1996

[21] 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

[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.