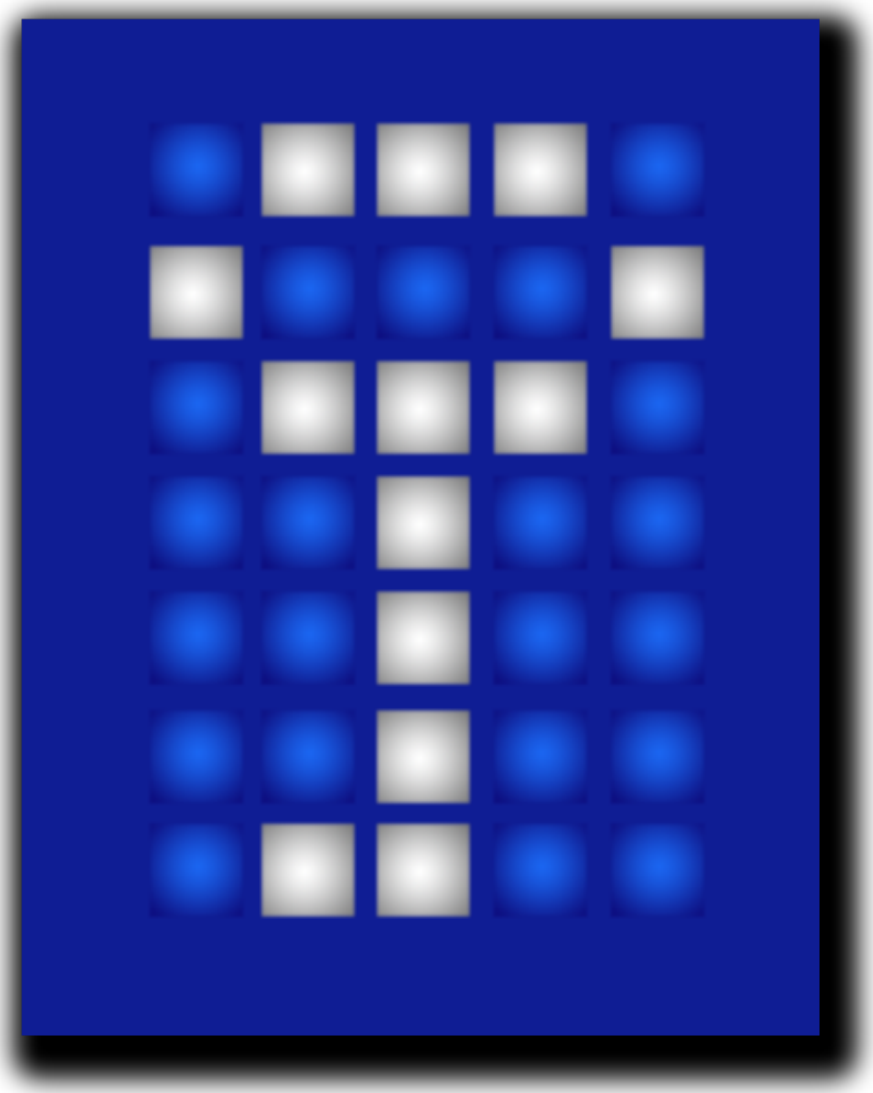
TRUECRYPT

User's Manual



Introduction

TrueCrypt is software 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 TrueCrypt 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 TrueCrypt volume. Similarly, files that are being written or copied to the TrueCrypt 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 TrueCrypt. 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 TrueCrypt volume (therefore, the video file is entirely encrypted). The user provides the correct password (and/or keyfile) and mounts (opens) the TrueCrypt 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 TrueCrypt-encrypted volume to RAM (memory) in order to play it. While the portion is being loaded, TrueCrypt 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 TrueCrypt-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 TrueCrypt 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).

System Encryption

TrueCrypt 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 too.

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 TrueCrypt Boot Loader, which resides in the first track of the boot drive and on the TrueCrypt Rescue Disk.

Note that TrueCrypt 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 TrueCrypt-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.

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*.

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.

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). TrueCrypt 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).

Operating Systems Supported for System Encryption

TrueCrypt 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.

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). TrueCrypt 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.

TrueCrypt Rescue Disk

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

- If the TrueCrypt Boot Loader screen does not appear after you start your computer (or if Windows does not boot), the **TrueCrypt Boot Loader may be damaged**. The TrueCrypt 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 TrueCrypt Boot Loader*. Then press 'Y' to confirm the action, remove the Rescue Disk from your CD/DVD drive and restart your computer.

- If the **TrueCrypt Boot Loader is frequently damaged** (for example, by inappropriately designed activation software) or if **you do not want the TrueCrypt 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 TrueCrypt Rescue Disk (as it contains the TrueCrypt 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 TrueCrypt says that the password is incorrect, it is possible that the **master key or other critical data are damaged**. The TrueCrypt 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. 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 TrueCrypt Rescue Disk, you also restore the password that was valid when the TrueCrypt Rescue Disk was created. Therefore, whenever you change the password, you should destroy your TrueCrypt 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 TrueCrypt 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 TrueCrypt 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.

  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: Boot another operating system, run TrueCrypt, click *Select Device*, select the affected system partition, select*System* > *Mount Without Pre-Boot Authentication*, enter your pre-boot-authentication password and click *OK*. The partition will be mounted as a regular TrueCrypt volume (data will be on-the-fly decrypted/encrypted in RAM on access, as usual).

- Your TrueCrypt Rescue Disk contains a **backup of the original content of the first drive track**(made before the TrueCrypt Boot Loader was written to it) and allows you to restore it if necessary. The first track of a boot drive 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 TrueCrypt 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 TrueCrypt Rescue Disk, insert it into your CD/DVD drive and restart your computer. If the TrueCrypt 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 TrueCrypt Rescue Disk screen should appear now. Note: In the TrueCrypt Rescue Disk screen, you can select 'Repair Options' by pressing F8 on your keyboard.

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

Parallelization

When your computer has a multi-core processor (or multiple processors), TrueCrypt uses all of the cores (or processors) in parallel for encryption and decryption. For example, when TrueCrypt 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, TrueCrypt creates one thread for each logical core/processor. For example, on a 6-core processor that provides two logical cores per one physical core, TrueCrypt uses 12 threads.

When your computer has a multi-core processor/CPU (or multiple processors/CPUs), header key derivationis 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, TrueCrypt uses so-called pipelining (asynchronous processing). While an application is loading a portion of a file from a TrueCrypt-encrypted volume/drive, TrueCrypt is automatically decrypting it (in RAM). Thanks to pipelining, the application does not have 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 TrueCrypt volumes).*

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

---

* 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,* which is typically 4-8 times faster than encryption performed by the purely software implementation on the same processors.

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

Note: By default, TrueCrypt 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 TrueCrypt 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 TrueCrypt uses for hardware-accelerated AES, please check the documentation for the processor or contact the vendor/manufacturer. Alternatively, click here to view an official list of Intel processors that support the AES-NI instructions. 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 (for example, i7-2630/2635QM, i7-2670/2675QM, i5-2430/2435M, i5-2410/2415M). 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 TrueCrypt 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 TrueCrypt components internally perform the requested change of mode. Also note

that when you create a TrueCrypt 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 TrueCrypt Rescue Disk, select *System* > *Create Rescue Disk*.

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

---

* In this chapter, the word 'encryption' also refers to decryption.
** 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]).

Encryption Algorithms

TrueCrypt volumes can be encrypted using the following algorithms:

Algorithm	Designer(s)	Key Size (Bits)	Block (Bi
AES	J. Daemen, V. Rijmen	256	12
Serpent	R. Anderson, E. Biham, L. Knudsen	256	12
Twofish	B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Ferguson	256	12
AES-Twofish		256; 256	12
AES-Twofish-Serpent		256; 256; 256	12
Serpent-AES		256; 256	12
Serpent-Twofish-AES		256; 256; 256	12
Twofish-Serpent		256; 256	12

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]. TrueCrypt 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 31 got votes, RC6 got 23 votes, and MARS got 13 votes [18, 19].*

---

* 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, 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 2^{128} 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.

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 TrueCrypt 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.

TrueCrypt currently supports the following hash algorithms:

- **RIPEMD-160**
- **SHA-512**
- **Whirlpool**

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). TrueCrypt 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].

Technical Details

Notation

C	Cipher text block
$D_K()$	Decryption algorithm using encryption/decryption key K
$E_K()$	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
^	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 o 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; $0 \oplus 3 = 1$)
$\otimes$	Modular multiplication of two polynomials over the binary field GF(2), modulo $x^{128}+x^7+x^2+x+1$ (GF stands for G
||	Concatenation

Encryption Scheme

When mounting a TrueCrypt 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 TrueCrypt 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 TrueCrypt Boot Loader is stored in the first track of the system drive and/or on the TrueCrypt Rescue Disk).

2. Bytes 65536–66047 of the volume are read into RAM (see the section TrueCrypt Volume Format Specification). For system encryption, bytes 65536–66047 of the first partition located behind the active partition* are read into RAM (see the section Hidden Operating System). If there is a hidden volume within this volume (or within the partition behind the active 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 TrueCrypt 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 (TrueCrypt never saves them 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 by the header key derivation function (as specified in PKCS #5 v2.0; see the sectionHeader 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 Header Key Derivation, Salt, and Iteration Count). 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 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, 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).

6. The encryption routine is reinitialized with the primary master key*** and the secondary key (XTS mode), which are retrieved from the decrypted volume header (see the section TrueCrypt 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 the section Modes of Operation and the section Header Key Derivation, Salt, and Iteration Countand also the chapter Security Model.

---

* 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).
** 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 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 of Operation

The mode of operation used by TrueCrypt 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:

$C_i = E_{K1}(P_i \text{ ^ } (E_{K2}(n) \otimes a^i)) \text{ ^ } (E_{K2}(n) \otimes a^i)$

Where:

$\otimes$	denotes multiplication of two polynomials over the binary field GF(2) modulo $x^{128}+x^7+x^2+x+1$
$K1$	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, $n = 0$
a	is a primitive element of Galois Field (2^{128}) that corresponds to polynomial x (i.e., 2)

Note: The remaining symbols are defined in the section Notation.

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].

Header Key Derivation, Salt, and Iteration Count

Header key is used to encrypt and decrypt the encrypted area of the TrueCrypt volume header (for system encryption, of the keydata area), which contains the master key and other data (see the sections Encryption Scheme and TrueCrypt Volume Format Specification). In volumes created by TrueCrypt 5.0 or later (and forsystem encryption), the area is encrypted in XTS mode (see the section Modes of Operation). The method that TrueCrypt uses to generate the header key and the secondary header key (XTS mode) is PBKDF2, specified in PKCS #5 v2.0; see [7].

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 theTrueCrypt 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 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*: Windows CryptoAPI (collected regularly at 500-ms interval)
- *MS Windows*: Network interface statistics (NETAPI32)
- *MS Windows*: 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 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 automatically 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 $B_0...B_q$.
   For $0 \leq i \leq q$ (i.e., for each block B) the following steps are performed:
   a. $M = H(B_0 || B_1 || ... || B_q)$ [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. $R = B_0 || B_1 || ... || B_q$

For example, if $q = 1$, the randomness pool would be mixed as follows:

1. $(B_0 || B_1) = R$

2. $B_0 = B_0 \wedge H(B_0 || B_1)$

3. $B_1 = B_1 \wedge H(B_0 || B_1)$

4. $R = B_0 || B_1$

Generated Values

The content of the RNG pool is never directly exported (even when TrueCrypt 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 TrueCrypt instructs it to generate and export a value:

1. Data obtained from some of 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

TrueCrypt keyfile is a file whose content is combined with a password. The user can use any kind of file as a TrueCrypt keyfile. The user can also generate a keyfile using the built-in keyfile generator, which utilizes the TrueCrypt 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 TrueCrypt GUI).

Keyfiles are processed and applied to a password using the following method:

1. Let *P* be a TrueCrypt 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, $(T_0 \,||\, T_1 \,||\, T_2 \,||\, T_3) = M$
        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 $B_0 ... B_{pl-1}$.
    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 $G_0 ... G_{kpl-1}$
c. For $0 \leq i < kpl$ perform: $B_i = B_i \oplus G_i$
d. $P = B_0 \,||\, B_1 \,||\, ... \,||\, B_{pl-2} \,||\, B_{pl-1}$

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 sectionHeader 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).

TrueCrypt Volume Format Specification

Note that this specification applies to volumes created by 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). TrueCrypt volumes have no "signature" or ID strings. Until decrypted, they appear to consist solely of random data.

Free space on 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 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 initialised 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 Hidden Volume). 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.

TrueCrypt Volume Format Specification:

Offset (bytes)	Size (bytes)	Encryption Status†	Description
0	64	Unencrypted§	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/decrypted volume; bits 2–31 are reserved)
128	4	Encrypted	Sector size (in bytes)
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**
512	65024	Encrypted	Reserved (for system encryption, this item is omitted‡‡)
65536	65536	Encrypted / Unencrypted§	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.‡‡ 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‡	65536	Encrypted / Unencrypted§	Backup header (encrypted with a different header key derived using a different salt). For system encryption, this item is omitted.‡‡ See bytes 0–65535.
S-65536‡	65536	Encrypted / Unencrypted§	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.†† For system encryption, this item is omitted.‡‡ See bytes 0–65535.

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 TrueCrypt 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 TrueCrypt 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 TrueCrypt 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 TrueCrypt volume created by 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 TrueCrypt 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.

---

* 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).
† 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.
‡ *S* denotes the size of the volume host (in bytes).
§ 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).
** 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).
†† See above in this section for information on the method used to fill free volume space with random data when the volume is created.
‡‡ Here, the meaning of "system encryption" does not include a hidden volume containing a hidden operating system.

Compliance with Standards and Specifications

To our best knowledge, TrueCrypt complies with the following standards, specifications, and recommendations:

- ISO/IEC 10118-3:2004 [21]
- FIPS 197 [3]
- 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 TrueCrypt.

Source Code

TrueCrypt is open-source and free software. The complete source code of TrueCrypt (written in C, C++, and assembly) is freely available for peer review at:

http://www.truecrypt.org/

TrueCrypt Volume

There are two types of TrueCrypt volumes:

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

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

A TrueCrypt 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 TrueCrypt partition is a hard disk partition encrypted using TrueCrypt. You can also encrypt entire hard 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 volume 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, 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 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 (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 TrueCrypt 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 TrueCrypt 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 ("greyed out").

Note that the size of a dynamic (sparse-file-hosted) TrueCrypt volume reported by Windows and by TrueCrypt 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 TrueCrypt), then select *Properties* and see the *Size on disk* value.

WARNING: Performance of dynamic (sparse-file-hosted) TrueCrypt volumes is significantly worse than performance of regular volumes. Dynamic (sparse-file-hosted) TrueCrypt 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.

TrueCrypt Volumes on CDs and DVDs

If you want a TrueCrypt volume to be stored on a CD or a DVD, 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 or later, using the CD burning tool provided with the operating system).

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. 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

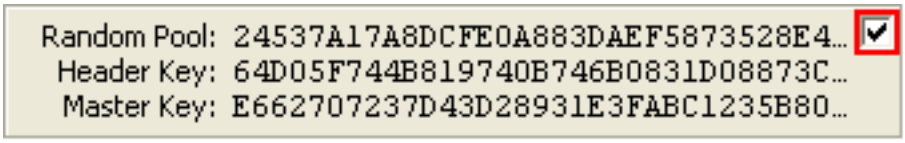
Windows Vista or later: Dynamic volumes are displayed in the 'Select Device' dialog window as`\Device\HarddiskVolume`*N*.

Windows XP/2000/2003: TrueCrypt supports hardware/software RAID as well as Windows dynamic volumes. If you intend to format a Windows dynamic volume as a TrueCrypt 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 TrueCrypt 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 TrueCrypt 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 TrueCrypt volume (for example in the '*Computer*' or '*My Computer*' list) and select 'Format'.

For more information about creating TrueCrypt 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 TrueCrypt 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 TrueCrypt 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 TrueCrypt 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: TrueCrypt 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 TrueCrypt 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 TrueCrypt 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: TrueCrypt 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 TrueCrypt 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 encryptionmounted 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 will be able to 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 form 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 TrueCrypt 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 TrueCrypt only to users with administrator privileges** (select *Settings* > '*System Favorite Volumes*' > '*Allow only administrators to view and dismount system favorite volumes in TrueCrypt*'). 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 TrueCrypt volume actions (such as '*Dismount All*', auto-dismount, etc.) from affecting system favorite volumes. In addition, when TrueCrypt 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 TrueCrypt application window.

To configure a TrueCrypt 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 TrueCrypt 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 TrueCrypt 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 TrueCrypt).

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

TrueCrypt 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 too.

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 TrueCrypt Boot Loader, which resides in the first track of the boot drive and on the TrueCrypt Rescue Disk.

Note that TrueCrypt 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 TrueCrypt-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.

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*.

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.

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). TrueCrypt 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).

Operating Systems Supported for System Encryption

TrueCrypt 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: Supported Operating Systems

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). TrueCrypt 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.

TrueCrypt Rescue Disk

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

- If the TrueCrypt Boot Loader screen does not appear after you start your computer (or if Windows does not boot), the **TrueCrypt Boot Loader may be damaged**. The TrueCrypt 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 TrueCrypt Boot Loader*. Then press 'Y' to confirm the action, remove the Rescue Disk from your CD/DVD drive and restart your computer.

- If the **TrueCrypt Boot Loader is frequently damaged** (for example, by inappropriately designed activation software) or if **you do not want the TrueCrypt 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 TrueCrypt Rescue Disk (as it contains the TrueCrypt 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 TrueCrypt says that the password is incorrect, it is possible that the **master key or other critical data are damaged**. The TrueCrypt 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. 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 TrueCrypt Rescue Disk, you also restore the

password that was valid when the TrueCrypt Rescue Disk was created. Therefore, whenever you change the password, you should destroy your TrueCrypt 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 TrueCrypt 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 TrueCrypt 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.

  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: Boot another operating system, run TrueCrypt, click *Select Device*, select the affected system partition, select*System* > *Mount Without Pre-Boot Authentication*, enter your pre-boot-authentication password and click *OK*. The partition will be mounted as a regular TrueCrypt volume (data will be on-the-fly decrypted/encrypted in RAM on access, as usual).

- Your TrueCrypt Rescue Disk contains a **backup of the original content of the first drive track**(made before the TrueCrypt Boot Loader was written to it) and allows you to restore it if necessary. The first track of a boot drive 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 TrueCrypt 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 TrueCrypt Rescue Disk, insert it into your CD/DVD drive and restart your computer. If the TrueCrypt 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 TrueCrypt Rescue Disk screen should appear now. Note: In the TrueCrypt Rescue Disk screen, you can select 'Repair Options' by pressing F8 on your keyboard.

If your TrueCrypt Rescue Disk is damaged, you can create a new one by selecting *System* > *Create Rescue Disk*. To find out whether your TrueCrypt 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, TrueCrypt provides and supports two kinds of plausible deniability:

1. Hidden volumes (see the section **Hidden Volume**) and hidden operating systems (see the section**Hidden Operating System**).

2. Until decrypted, a TrueCrypt 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 TrueCrypt 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, TrueCrypt 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 the section Data Leaks) and also note that, for system encryption, the first drive track contains the (unencrypted) TrueCrypt 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 TrueCrypt 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 TrueCrypt volume (container) by creating a hidden volume within it (see above).

Notes

- When formatting a hard disk partition as a TrueCrypt volume (or encrypting a partition in place), the partition table (including the partition type) is *never* modified (no TrueCrypt "signature" or "ID" is written to the partition table).

- There are methods to find files or devices containing random data (such as TrueCrypt 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 TrueCrypt volume or that the file, partition, or device, contains a hidden TrueCrypt volume (provided that you follow the security requirements and precautions listed in the chapter Security Requirements and Precautions and in the subsectionSecurity Requirements and Precautions Pertaining to Hidden Volumes).

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 should be impossible to prove 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 (dismounted) hidden volume can be distinguished from random data. Note that TrueCrypt 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.

A hidden volume can be mounted the same way as a standard TrueCrypt 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).

TrueCrypt 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 TrueCrypt 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 TrueCrypt volume, i.e., within a file-hosted volume or partition/device-hosted volume (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 will provide help and all information necessary to successfully create a hidden TrueCrypt 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.***

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).

---

* 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 Volumesare followed.

** 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.

*** 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.

Security Requirements and Precautions Pertaining to Hidden Volumes

If you use a hidden TrueCrypt 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 TrueCrypt to secure data stored in a hidden TrueCrypt volume and the ability to provide plausible deniability.

- If an adversary has access to a (dismounted) TrueCrypt 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 ahidden 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 TrueCrypt container has been defragmented and a copy of the TrueCrypt 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 TrueCrypt 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 TrueCrypt volumes.
- A file-hosted TrueCrypt container is stored in a journaling file system (such as NTFS). A copy of the TrueCrypt container (or of its fragment) may remain on the host volume. To prevent this, do one the following:
  - Use a partition/device-hosted TrueCrypt volume instead of file-hosted.
  - Store the container in a non-journaling file system (for example, FAT32).
- A TrueCrypt 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 TrueCrypt 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 TrueCrypt 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 TrueCrypt 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 TrueCrypt volume, make sure that the volume is not sparse-file-hosted (the Windows version of TrueCrypt 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 sectionHidden Operating System) and mount hidden volumes only when the hidden operating system is running. Note: When a hidden operating system is running, TrueCrypt ensures that all local unencrypted filesystems and non-hidden TrueCrypt volumes are read-only (i.e. no files can be written to such filesystems or TrueCrypt volumes).* Data is allowed to be written to filesystems within hidden TrueCrypt 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 TrueCrypt 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 TrueCrypt 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 TrueCrypt 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 TrueCrypt volumes under Mac OS X.

- When an outer volume is mounted with hidden volume protection enabled, 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).

  - 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 an alternative plausible 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:

Hidden Operating System

If your system partition or system drive is encrypted using TrueCrypt, you need to enter your pre-boot authentication password in the TrueCrypt 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). TrueCrypt 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 TrueCrypt volume is.

A **hidden operating system** is a system (for example, Windows 7 or Windows XP) that is installed in ahidden TrueCrypt volume. It should be impossible to prove that a hidden TrueCrypt 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 TrueCrypt, an unencrypted copy of the TrueCrypt Boot Loader has to be stored on the system drive or on a TrueCrypt Rescue Disk. Hence, the mere presence of the TrueCrypt Boot Loader can indicate that there is a system encrypted by TrueCrypt on the computer. Therefore, to provide a plausible explanation for the presence of the TrueCrypt Boot Loader, the TrueCrypt 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.*

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).

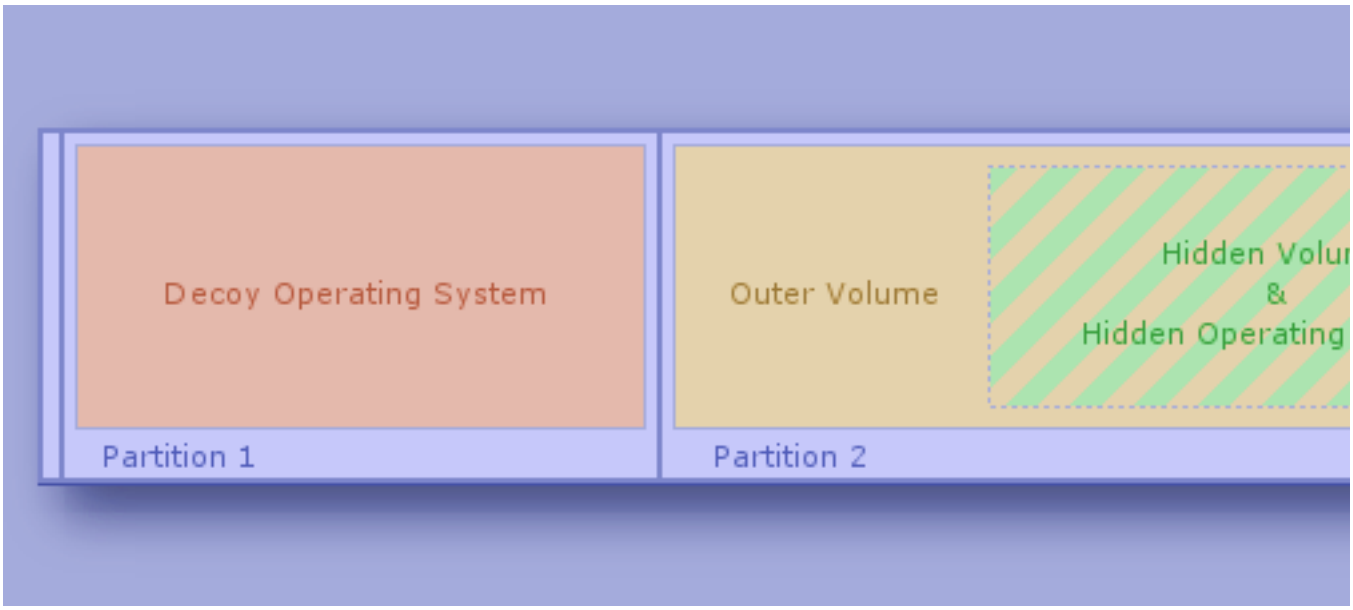
There will be two pre-boot authentication passwords — one for the hidden system and the other for the decoy system. If you want to start the hidden system, you simply enter the password for the hidden system in the TrueCrypt Boot Loader screen (which appears after you turn on or restart your computer). 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 TrueCrypt Boot Loader screen.

Note: When you enter a pre-boot authentication password, the TrueCrypt 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 TrueCrypt 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 where the encrypted header of a possible hidden volume might be stored (however, 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, because Windows 7 and later, by default, do not boot from the partition on which they are installed). Note that TrueCrypt 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 TrueCrypt 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 TrueCrypt 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

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 TrueCrypt 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, TrueCrypt 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 TrueCrypt wizard in the hidden-system-creation mode (which might indicate the existence of a hidden operating system on your computer). To prevent such issues, TrueCrypt 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, TrueCrypt will prompt you to install a new system on the partition and encrypt it using TrueCrypt. Thus, you will create the decoy system and the whole process of creation of the hidden operating system will be completed.

Note: TrueCrypt will erase the content of the partition where the original system resides by filling it with random data entirely. If you revealed the password for the decoy system to an adversary and he asked you why the free space of the (decoy) system partition contains random data, you could answer, for example: "The partition previously contained a system encrypted by TrueCrypt, but I forgot the pre-boot authentication password (or the system was damaged and stopped booting), so I had to reinstall Windows and encrypt the partition again."

Plausible Deniability and Data Leak Protection

For security reasons, when a hidden operating system is running, TrueCrypt ensures that all local unencrypted filesystems and non-hidden TrueCrypt volumes are read-only (i.e. no files can be written to such filesystems or TrueCrypt volumes).† Data is allowed to be written to any filesystem that resides within a hidden TrueCrypt 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 TrueCrypt 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. TrueCrypt 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 TrueCrypt volume.
3. Start the hidden system
4. If you saved the files to a TrueCrypt 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.

Possible Explanations for Existence of Two TrueCrypt Partitions on Single Drive

An adversary might ask why you created two TrueCrypt-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, TrueCrypt 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 TrueCrypt 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 TrueCrypt 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, TrueCrypt 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 TrueCrypt Boot Loader is larger than normal and, therefore, there is not enough space in the first drive track for a backup of the TrueCrypt 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 TrueCrypt Rescue Disk to repair the TrueCrypt Boot Loader or to boot.
  - On some computers, resuming from hibernation takes longer.
- In contrast to a password for a non-system TrueCrypt 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 TrueCrypt 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 TrueCrypt 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 TrueCrypt volume, a user of a hidden operating system must follow all of the security requirements and precautions that apply to normal hidden TrueCrypt 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 sectionProtection 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.

---

* 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 protectionfeature, it would inherently cause system crashes, i.e. 'Blue Screen' errors).
† This does not apply to filesystems on CD/DVD-like media and on custom, atypical, or non-standard devices/media.