TrueCrypt: Linux

Fix Me!

From ArchWiki

TrueCrypt is a free open source on-the-fly encryption (OTFE) program. Some of its features are:

- Virtual encrypted disks within files that can be mounted as real disks.
- Encryption of an entire hard disk partition or a storage device/medium.
- All encryption algorithms use the LRW mode of operation, which is more secure than CBC mode with predictable initialization vectors for storage encryption.
- «Hidden volumes» within a normal «outer» encrypted volume. A hidden volume can not be distinguished from random data without access to a passphrase and/or keyfile.

For more details on how TrueCrypt compares to other disk encryption solution, see Disk Encryption#Comparison table.

Contents

Installation

Note: For opening and accessing an existing TrueCrypt container cryptsetup is the preferred way, since it is well integrated with the rest of the system. Creating a new TrueCrypt container can be done using `truecrypt`, after which it can be opened using `cryptsetup`.

Install truecrypt from the official repositories. If you use any kernel other than linux install the corresponding kernel module.

If you are using truecrypt to encrypt a virtual filesystem (e.g. a file), the module will be automatically loaded whenever you run the *truecrypt* command.

If you are using truecrypt to encrypt a physical device (e.g. a hard disk or usb drive), you will likely want to load the module during the boot sequence:

Add the module to `/etc/modules-load.d/`:

```
# tee /etc/modules-load.d/truecrypt.conf <<< "truecrypt"
```

Note:

- This didn't work for me (module truecrypt seems to be non-existent now), but adding «loop» module worked

```
# tee /etc/modules-load.d/truecrypt.conf <<< "loop"
# modprobe loop
```

- It does not appear that loading a module applies with TrueCrypt 7.1a, the current version in Arch as of 4/19/2013. The above advice may be outdated with respect to the module, however it is still important to enable **FUSE**, **loop** and your encryption algorithm (e.g. **AES**, **XTS**, **SHA512**) in custom kernels.

If you only want to open and access an existing truecrypt container, this can also be done with `cryptsetup` i.e. without installing Truecrypt.

Accessing a TrueCrypt container using cryptsetup

Since version 1.6, cryptsetup supports opening TrueCrypt containers natively, without the need of the truecrypt package. To do so, execute the following command:

```
$ cryptsetup --type tcrypt open container-to-mount container-name
```

Replace `container-to-mount` with the device file under `/dev` or the path to the file you wish to open. Upon successful opening, the plaintext device will appear as `/dev/mapper/container-name`, which you can `mount` like any normal device.

If you are using key files, supply them using the `–key-file` option, to open a hidden volume, supply the `–tcrypt-hidden` option and for a partition or whole drive that is encrypted in system mode use the `–tcrypt-system` option.

See `man cryptsetup` for more details and all supported options.

Automounting using /etc/crypttab

Since version 206, systemd supports (auto)mounting TrueCrypt containers at boot or runtime using `/etc/crypttab`.

The following example setup will mount `/dev/sda2` in system encryption mode as soon as `/mnt/truecrypt-volume` is accessed using systemd's automounting logic. The passphrase to open the volume is given in `/etc/volume.password`. Note that the device file given in `/etc/fstab` needs to be the one from `/dev/mapper/` and not, for example, from `/dev/disk/by-uuid/` for automounting logic to kick in. Other than that you can still reliably identify the encrypted volume itself inside of `/etc/crypttab` using device file names from `/dev/disk/`.

```
/etc/crypttab
truecrypt-volume    /dev/sda2    /etc/volume.password    tcrypt-system,noauto
/etc/fstab
/dev/mapper/truecrypt-volume    /mnt/truecrypt-volume    auto    noauto,x-systemd.automount    0
0
```

See `man crypttab` for more details and options supported.

Encrypting a file as a virtual volume

The following instructions will create a file that will act as a virtual filesystem, allowing you to mount it and store files within the encrypted file. This is a convenient way to store sensitive information, such as financial data or passwords, in a single file that can be accessed from Linux, Windows, or Macs.

To create a new truecrypt file interactively, type the following in a terminal:

```
$ truecrypt -t -c
```

Follow the instructions, choosing the default values unless you know what you are doing:

```
Volume type:
 1) Normal
 2) Hidden
```

```
Select [1]: 1
Enter file or device path for new volume: /home/user/EncryptedFile.tc
Enter volume size (bytes - size/sizeK/sizeM/sizeG): 32M
Encryption algorithm:
 1) AES
 2) Blowfish
 3) CAST5
 4) Serpent
 5) Triple DES
 6) Twofish
 7) AES-Twofish
 8) AES-Twofish-Serpent
 9) Serpent-AES
10) Serpent-Twofish-AES
11) Twofish-Serpent
Select [1]: 1
Hash algorithm:
 1) RIPEMD-160
 2) SHA-1
 3) Whirlpool
Select [1]: 1
```

Filesystem:

```
1) None
2) FAT
3) Linux Ext2
4) Linux Ext3
5) Linux Ext4
```

Select [2]:

```
Enter password for new volume '/home/user/EncryptedFile.tc': ****************************
Re-enter password: ****************************
Enter keyfile path [none]:
Please type at least 320 randomly chosen characters and then press Enter:
Done: 32.00 MB  Speed: 10.76 MB/s  Left: 0:00:00
Volume created.
```

You can now mount the new encrypted file to a previously-created directory:

```
$ truecrypt -t /home/user/EncryptedFile.tc /home/user/EncryptedFileFolder
```

Note: Truecrypt requires root privileges and as such, running the above command as a user will attempt to use **sudo** for authentication. To work with files as a regular user, please seeMount volumes as a normal user.

Once mounted, you can copy or create new files within the encrypted directory as if it was any normal directory. When you are you ready to re-encrypt the contents and unmount the directory, run:

```
$ truecrypt -t -d
```

Again, this will require administrator privileges through the use of **sudo**. After running it check if the files that are to be encrypted are indeed no longer in the directory. (might want to try unimportant data first) If they are still there, note that *rm* doesn't make the data unrecoverable.

For more information about truecrypt in general, run:

```
$ man truecrypt
```

Note: As of 1:7.1a-1 dont see a man or info page.

Several options can be passed at the command line, making automated access and creation a simple task. The man page is highly recommended reading.

Encrypting a physical volume

Note: If you are having problems with the graphical interface, you can run in CLI mode with the -t flag.

If you want to use a keyfile, create one with this command:

```
truecrypt --create-keyfile /etc/disk.key
```

By default both passphrase and key will be needed to unlock the volume.

Create a new volume in the device `/dev/sda1`:

```
# truecrypt --volume-type=normal -c /dev/sda1
```

Map the volume to `/dev/mapper/truecrypt1`:

```
# truecrypt -N 1 /dev/sda1
```

If this command does not for you try this to map the volume:

```
# truecrypt --filesystem=none --slot=1 /dev/sda1
```

Simply format the disk like you normally would choosing your favourite file system, except use the path `/dev/mapper/truecrypt1`. E.g. for ext4 use:

```
# mkfs.ext4 /dev/mapper/truecrypt1
```

Mount the volume:

```
# mount /dev/mapper/truecrypt1 /media/disk
```

Map and mount a volume:

```
# truecrypt /dev/sda1 /media/disk
```

Unmount and unmap a volume:

```
# truecrypt -d /dev/sda1
```

Creating a hidden volume

First, create a normal outer volume as described in #Encrypting a physical volume.

Map the outer volume to `/dev/mapper/truecrypt1`:

```
# truecrypt -N 1 /dev/sda1
```

Create a hidden truecrypt volume in the free space of the outer volume:

```
 # truecrypt --type hidden -c /dev/sda1
```

You need to use another passphrase and/or keyfile here than the one you used for the outer volume.

Unmap the outer truecrypt volume and map the hidden one:

```
# truecrypt -d /dev/sda1
# truecrypt -N 1 /dev/sda1
```

Just use the passphrase you chose for the hidden volume and TrueCrypt will automatically choose it before the outer.

Create a file system on it (if you have not already) and mount it:

```
# mkfs.ext4 /dev/mapper/truecrypt1
```

```
# mount /dev/mapper/truecrypt1 /media/disk
```

Map and mount the outer volume with the hidden write-protected:

```
truecrypt -P /dev/sda1 /media/disk
```

Mount a special filesystem

Note: Current Versions of truecrypt seem to support NTFS write support by default so the `–filesystem` flag no longer seems to be necessary.

In the following example I want to mount a ntfs-volume, but TrueCrypt does not use *ntfs-3g by* default (so there is no write access; checked in version 6.1). The following command works for me:

```
truecrypt --filesystem=ntfs-3g --mount /file/you/want/to/mount
```

You may also want to mount ntfs volume without execute flag on all files

```
truecrypt   --filesystem=ntfs-3g   --fs-options=users,uid=$(id   -u),gid=$(id   -
g),fmask=0113,dmask=0002
```

Mount volumes via fstab

First of all, we need to write a script which will handle the way mounting via fstab is done. Place the following in `/usr/bin/mount.truecrypt`:

```
#!/usr/bin/env sh
DEV="$1"
MNTPT="$2"
OPTIONS=""
TCOPTIONS=""
shift 3
IFS=','
for arg in $*; do
        if [ "${arg}" == "system" ]; then
                TCOPTIONS="${TCOPTIONS}-m=system "
        elif [[ "${arg}" == fs=* ]]; then
                FS=${arg#*=}
                TCOPTIONS="${TCOPTIONS}--filesystem=${FS} "
        else
                OPTIONS="${OPTIONS}${arg},"
        fi
done
truecrypt ${DEV} ${MNTPT} ${TCOPTIONS% *} --fs-options="${OPTIONS%,*}"
```

Also do not forget to make the file executable:

```
# chmod +x /usr/bin/mount.truecrypt
```

Finally, add the device to fstab somewhat like this:

```
/dev/sdb3 /mnt truecrypt fs=vfat,defaults 0 0
```

Tip: This script is also provided by the truecrypt-mount package.

Mount volumes as a normal user

TrueCrypt needs root privileges to work: this procedure will allow normal users to use it, also giving writing permissions to mounted volumes.

Both methods below require Sudo. Make sure it is configured before proceeding.

Method 1: add a truecrypt group

Create a new group called truecrypt and give it the necessary permissions. Any users that belongs to that group, will be able to use TrueCrypt.

```
# groupadd truecrypt
```

Edit the sudo configuration:

```
# visudo
```

Append the following lines at the bottom of the sudo configuration file:

```
# Users in the truecrypt group are allowed to run TrueCrypt as root.
%truecrypt ALL=(root) NOPASSWD:/usr/bin/truecrypt
```

You can now add your users to the truecrypt group:

```
# gpasswd -M first_user,second_user,etc truecrypt
```

Note: In order to make these changes active, any user that has been added to the truecrypt group have to logout.

After that, you can mount your device by

```
# truecrypt --mount /path/to/device /path/to/mountpoint
```

Default mountpoint is `/media/truecrypt1`. Normally, it is not necessary to explicitly specify the filesystem of your device using the `–filesystem` flag.

It is definitely reasonable to give truecrypt some permission masks. Otherwise, every file on your mounted device will be executable. So instead of the above, you can use

```
# truecrypt --fs-options=users,uid=$(id -u),gid=$(id -g),fmask=0113,dmask=0002 --mount
/PATH/TO/DEVICE /PATH/TO/MOUNTPOINT
```

and add this line to your bash configuration file, `~/.bashrc` as an alias:

```
alias tc1='truecrypt --fs-options=users,uid=$(id -u),gid=$(id -g),fmask=0113,dmask=0002 --mount
/path/to/device"" /path/to/mountpoint
```

To mount this specific device, use

```
# tc1
```

as a normal user.

Method 2: sudo simplified

Simply enable desired user to run truecrypt without a password:

```
# visudo
```

Append the following:

```
USERNAME ALL = (root) NOPASSWD:/usr/bin/truecrypt
```

alternatively, if you make use of the wheel group:

```
%wheel ALL = (root) NOPASSWD:/usr/bin/truecrypt
```

If you have any difficulties with permissions as a normal user, just add the `-u` flag to the truecrypt mount command, for example:

```
$ truecrypt -u /home/user/EncryptedFile.tc /home/user/EncryptedFileFolder
```

Automatic mount on login

Simply add:

```
$ truecrypt /home/user/Encrypted File.tc /home/user/Encrypted File Folder <<EOF
password
EOF
```

to your startup procedure. Do not use the `-p` switch, this method is more secure. Otherwise everyone can just look up the password via *ps* and similar tools, as it is in the process name! source

The most recent truecrypt has a couple of followup questions. If you have expect installed, this will work (assuming no keyfile and no desire to protect hidden volume), saved to a file with root-only perms called from /etc/rc.local:

```
#! /bin/bash
expect << EOF
spawn /usr/bin/truecrypt ''/path/to/EncryptedFile'' ''/mount/point''
expect "Enter password"
send "volume password\n"
expect "Enter keyfile"
send "\n"
expect "Protect hidden volume"
send "\n"
expect eof;
EOF
```

Of course, this isn't as secure as entering your password manually. But for some use cases, such as when your TrueCrypt filesystem is in a file on shared storage, it's better than being unencrypted.

Safely unmount and unmap volumes (on shutdown)

You can unmount a specific device by

```
# truecrypt -d /path/to/mountpoint
```

or leave away the path to unmount all truecrypt volumes.

If you want your truecrypt device to be unmounted automatically at shutdown, add the following to the file `/etc/rc.local.shutdown`:

```
if (/usr/bin/truecrypt --text --list)
then {
/usr/bin/truecrypt -d
sleep 3
}
fi
```

You can also leave away the *sleep* command, it is just to give the unmounting some time to complete before the actual shutdown.

If you're using systemd, there is a service trying to unmount truecrypt-encrypted filesystems at shutdown automatically on the systemd/Services page.

Errors

TrueCrypt is already running

If a messagebox *TrueCrypt is already running* appears when starting TrueCrypt, check for a hidden file in the home directory of the concerned user called *.TrueCrypt-lock-username*. Substitute *username* with the individual username. Delete the file and start TrueCrypt again.

Deleted stale lockfile

If you always get a message «Delete stale lockfile [....]» after starting Truecrypt, the Truecrypt process with the lowest ID has

to be killed during Gnome log out. Edit `/etc/gdm/PostSession/Default` and add the following line before exit 0:

```
kill $(ps -ef | grep truecrypt | tr -s ' ' | cut -d ' ' -f 2)
```

Issues with Unicode file/folder names

NTFS

Should files resp. folders containing Unicode characters in their names be incorrectly or not at all displayed on TrueCrypt NTFS volumes (while e. g. being correctly handled on non-encrypted NTFS partitions), first verify that you have the NTFS-3G driver installed and then create the following symlink as root:

```
ln -s /sbin/mount.ntfs-3g /sbin/mount.ntfs
```

That will cause TrueCrypt to automatically use this driver for NTFS volumes, having the same effect as the explicit use of

```
truecrypt --filesystem=ntfs-3g /path/to/volume
```

via the console.

One may also consider setting e.g.:

```
rw,noatime
```

amongst other options in the TrueCrypt GUI (*Settings* > *Preferences* > *Mount Options*).

FAT

Similarly, FAT32 volumes created using Windows may use Unicode rather than ISO 8859-1. In order to use UTF-8, set the mount option:

```
iocharset=utf8
```

when mounting such volumes, or globally as described above.

Unmount error (device mapper)

If you always get a message «device-mapper: remove ioctl failed: Device or resource busy» when attempting to dismount your truecrypt volume, the solution is to goto: Setting > Preferences > System Integration > Kernel Service and check the box

```
Do not use kernel cryptographic services
```

Mount error (device mapper, truecrypt partition)

When attempting to mount your truecrypt volume, a message like this one may appear:

```
Error: device-mapper: create ioctl failed: Device or resource busy
Command failed
```

If so, run:

```
# cryptsetup remove /dev/mapper/truecrypt1
```

Failed to set up a loop device

If you get a message «Failed to set up a loop device» when trying to create/mount a TrueCrypt volume, it may be because you updated your kernel recently without rebooting. Rebooting should fix this error.

Otherwise, check if *loop* has been loaded as kernel module:

```
$ lsmod | grep loop
```

If not listed, retry the TrueCrypt command after `modprobe loop`. Should it work, consider to add *loop* to the modules in `/etc/modules-load.d`:

```
# tee /etc/modules-load.d/truecrypt.conf <<< "loop"
```

Note: As of udev 181-5, the loop device module is no longer auto-loaded, and the procedure described here is necessary.

System partition passwords need en_US keymap

If you are using Xorg (which you most likely are, should you not know what that is), use the following command to use US keymap until restart:

```
# setxkbmap us
```

See also

- TrueCrypt homepage
- HOWTO: Truecrypt Gentoo wiki
- Truecrypt tutorial on HowToForge
- There is a good chance the CIA has a backdoor? (via wp)

Retrieved from «https://wiki.archlinux.org/index.php?title=TrueCrypt&oldid=296527»

Categories:

- Security
- File systems

Navigation menu

Views

- https://wiki.archlinux.org/index.php/TrueCryptPage
- https://wiki.archlinux.org/index.php/Talk:TrueCrypthttps://wiki.archlinux.org/index.php/Talk:TrueCryptDiscussion
- https://wiki.archlinux.org/index.php?title=TrueCrypt&action=editView source
- https://wiki.archlinux.org/index.php?title=TrueCrypt&action=historyHistory

Personal tools

- https://wiki.archlinux.org/index.php?title=Special:UserLogin&returnto=TrueCrypt&type=signupCreate account
- https://wiki.archlinux.org/index.php?title=Special:UserLogin&returnto=TrueCryptLog in

Navigation

- https://wiki.archlinux.org/index.php/Main_PageMain page
- https://wiki.archlinux.org/index.php/Table_of_ContentsCategories
- https://wiki.archlinux.org/index.php/Getting_InvolvedGetting Involved
- https://wiki.archlinux.org/index.php/ArchWiki:NewsWiki News
- https://wiki.archlinux.org/index.php/Special:RecentChangesRecent changes
- https://wiki.archlinux.org/index.php/Special:RandomRandom page
- https://wiki.archlinux.org/index.php/Help:ContentsHelp

Search

Tools

- https://wiki.archlinux.org/index.php/Special:WhatLinksHere/TrueCryptWhat links here
- https://wiki.archlinux.org/index.php/Special:RecentChangesLinked/TrueCryptRelated changes
- https://wiki.archlinux.org/index.php/Special:SpecialPagesSpecial pages
- https://wiki.archlinux.org/index.php?title=TrueCrypt&printable=yesPrintable version
- https://wiki.archlinux.org/index.php?title=TrueCrypt&oldid=296527Permanent link
- https://wiki.archlinux.org/index.php?title=TrueCrypt&action=infoPage information

In other languages

- Deutsch
- This page was last modified on 8 February 2014, at 02:21.
- Content is available under GNU Free Documentation License 1.3 or later unless otherwise noted.
- https://wiki.archlinux.org/index.php/ArchWiki:Privacy_policyPrivacy policy
- https://wiki.archlinux.org/index.php/ArchWiki:AboutAbout ArchWiki
- https://wiki.archlinux.org/index.php/ArchWiki:General_disclaimerDisclaimers

Ссылки

Install TrueCrypt on Debian 7 Wheezy

http://sysadminmosaic.ru/truecrypt/linux

2016-08-23 14:57