

## WIKIPEDIA ON LINUX ENCRYPTION

[http://en.wikipedia.org/wiki/Encryption\\_on\\_Linux](http://en.wikipedia.org/wiki/Encryption_on_Linux)

=====  
Tools to wipe out unwanted data:

Darik's Boot and Nuke:

<http://dban.sourceforge.net/>

Thomas Greene's Linux Wipe Tools:

<http://basicsec.org/tools.html>

## DM-CRYPT

Must be enabled in the kernel. Install utilities:

```
#apt-get install cryptsetup dmsetup
```

## ENCRYPTED SWAP

This process is like the other we done for /home, but the only difference is that the password will be different for every boot, since it will be read from /dev/random.

```
#swapoff -a
```

```
#cryptsetup create swap /dev/hda2
```

Type any crap as password, since it will not be used by the user.

Edit /etc/crypttab and add the line for swap:

```
swap /dev/hda2 /dev/random swap
```

Edit /etc/fstab and add the file for swap:

comment out this line:

```
# /dev/hda2 none swap sw 0 0
```

add this line:

```
# /dev/mapper/swap none swap sw 0 0
```

To enable swap immediately:

```
# cryptsetup remove swap
# /etc/init.d/cryptdisks start
# swapon -a
```

The following sources were checked for writing this article: cryptsetup(8), crypttab(5), <http://www.saout.de/tikiwiki/tiki-index.php?page=HOWTO>

---

## ENCRYPTED PARTITION

```
# cryptsetup -y create secret /dev/hda7
```

Now let's test to see if it worked:

```
# dmsetup ls
```

Next we have to create a filesystem (or what Windows refugees call "formatting a partition." I suggest using the ext2 filesystem. The command for doing this is as follows:

```
# mke2fs /dev/mapper/secret
```

All that remains to be done is to mount our device. We can mount it on any mount point we choose - create a new directory if you like:

```
# mkdir /data
# mount /dev/mapper/secret /data
```

The encrypted /data directory is ready for use. It should contain only one file, lost+found. You can copy all the data you want to it. When you've finished your work, you should unmount the /data mount point and remove the secret device so that unauthorized individuals cannot access the data (that is, mount the partition):

```
# umount /data
# cryptsetup remove secret
```

If we like, we can go ahead and shut down the computer now. Next time we boot up and we want to access the encrypted data, the procedure is very similar to our original effort except that we won't make a new filesystem:

```
# cryptsetup -y create secret /dev/hda7
```

Be sure you type the correct passphrase. Then mount:

```
# mount /dev/mapper/secret /data
```

If you type the wrong passphrase, you'll get an error message saying mount: you must specify the filesystem type. If you do create a new filesystem, you'll wipe out all of your previously saved data! Instead, type cryptsetup remove secret and start over again, this time using the correct passphrase.

Using a script:

```
#!/bin/sh

if [ -b /dev/mapper/secret ]; then
  /sbin/cryptsetup remove secret
fi
/sbin/cryptsetup create secret /dev/hda7
mount /dev/mapper/secret /data
```

Plus another script to remove:

```
#!/bin/sh

umount /data
/sbin/cryptsetup remove secret
```

To make the encrypted partition the home partition for user "robert", edit /etc/passwd:

```
robert:x:1003:1003::/data:/bin/bash
```

=====

## ENCRYPTED LOOPBACK FILE

You must create a loopback file, this is just going to be a regular file on your filesystem made up of random data. Make the loopback file as large as you are going to want your encrypted loopback to be.

The following creates a 100 meg file of random data at the location /home/secret:

```
# dd if=/dev/urandom of=/home/secret bs=1M count=100
```

Set this loopback file as a loop device:

```
# losetup /dev/loop/0 /home/secret
```

Encrypt the loopback...

It is a three step process to create our encrypted partition:

1. run cryptsetup on the loopback file, which creates a device mapper device with target 'crypt'.
2. create the file system on our new device.
3. mount our new device.

In our example, we will be turning the file /home/secret into the encrypted device at /dev/mapper/mycrypt (ie. label is 'mycrypt') and then mounting it at /sekret:

create logical volume (with cryptsetup binary):

```
# cryptsetup -y create mycrypt /dev/loop0
```

confirm it worked:

```
# dmsetup ls
mycrypt (254, 0)
```

create filesystem:

```
# mkfs.ext3 /dev/mapper/mycrypt
```

mount filesystem:

```
# mount /dev/mapper/mycrypt /sekret
```

You can add something like the following to your /etc/fstab:

```
# /dev/mapper/mycrypt /sekret ext3 noauto,noatime 0 0
```

When you are finished with using your encrypted loopback filesystem you need to unmount it and remove the device you created in the devicemapper, if you don't do this, anyone can remount it without typing the passphrase!

```
# umount /sekret
# cryptsetup remove mycrypt
```

Create a simple script to set this up and mount it so you can easily do this:

```
#!/bin/sh
```

```
if [ -b /dev/mapper/secret ]; then
  /sbin/cryptsetup remove secret
fi
/sbin/cryptsetup create secret /dev/hda7
mount /dev/mapper/secret /data
```

And create another script to remove:

```
#!/bin/sh
```

```
umount /sekret
/sbin/cryptsetup remove mycrypt
```

=====

Additional Resources:

A pretty good little summary about how to use DM-Crypt and loop\_AES for Debian (and other Debian-based systems including Ubuntu) is found here:

<http://deb.riseup.net/storage/encryption/>

3) Gentoo users might want to look here:

[http://gentoo-wiki.com/SECURITY\\_dmccrypt](http://gentoo-wiki.com/SECURITY_dmccrypt)

=====

Article I wrote for DistroWatch:

Bcrypt & DM-Crypt

<http://distrowatch.com/weekly.php?issue=20060227#3>

Other relevant articles:

Loop-AES

<http://distrowatch.com/weekly.php?issue=20040816#feedback>

Steganography

<http://distrowatch.com/weekly.php?issue=20040809#feedback>

GFTP and SFTP

<http://distrowatch.com/weekly.php?issue=20050418#tips>

Darik's Boot and Nuke

<http://distrowatch.com/weekly.php?issue=20050523#osa>

## RAW DATA CD: Security through obscurity

Create script burn.sh:

```
#!/bin/sh

# burn is a simple wrapper around cdrecord to facilitate burning arbitrary
# files to CDR, which might be pre-mastered ISO filesystem images or whatever.
#
# Copyright (c) 2005 by Reid Priedhorsky, <reid@reidster.net>.
#
# This script is distributed under the terms of the GNU General Public
# License; see http://www.gnu.org/licenses/gpl.txt for more information.

device=/dev/scd0
driveropts=burnfree
speed=99
mode=-dao
dummy=
overburn=
fs=32m

if [ "$1" = '--help' ]; then
    echo "options:"
    echo " -o DRIVEROPTS  set driver options"
    echo " -s SPEED      set speed (default $speed)"
    echo " -d           set dummy mode"
    echo " -r           set overburn mode"
    exit 1
fi

while getopts do:s:r f; do
    case $f in
        d)
            dummy='-dummy'
            ;;
        o)
            driveropts="{driveropts},$OPTARG"
            ;;
        s)
            speed=$OPTARG
            ;;
        r)
            overburn=-overburn
            ;;
    esac
done
shift `expr $OPTIND - 1`
```

```
echo "*** device:   $device"
echo "*** driveropts: $driveropts"
echo "*** speed:    $speed"
echo "*** modes:   $mode $dummy $overburn"
```

```
cdrecord -v $dummy $mode dev=$device speed=$speed fs=$fs \
  driveropts=$driveropts $overburn "$@"
```

Usage (example):

```
./burn.sh LinuxWipeTools.tar.gz
```

```
tar -zxvf /dev/scd0
```

=====

linux-crypto mailing list:

<http://mail.nl.linux.org/linux-crypto/>

=====

aespipe.README example 3.3 shows how to encrypt CDs. It puts a key file at beginning of the CD and specifies offset for encrypted data. If you want to be able to change passphrase afterwards, then encrypt the key file using gpg public-key crypto. Changing gpg private-key passphrase changes your CD/DVD mount passphrase.

<http://loop-aes.sourceforge.net/aespipe.README>

During the presentation, Robert had some difficulty in getting the data back off the "secure" CD as he writes "I found that the problem I was having reading those raw-data CDs seems to be that after I burn one, I have to eject it first. During the meeting, I was burning the CDs and trying to read them immediately. I know that sounds weird, but if I burn the CD and try to read it straight away, it won't read. I had to eject it once, then close the tray again and read it, then no problem."

This problem is not unique and it not only happens on Linux but this behavior has been observed on Windows and other environments. It has to do with the "cache" the operating system uses to store the information on a device. If the cache is not flushed somehow or synced (refreshed) to what is actually stored on the device, errors will occur. Ejecting the CD will usually force a flush and re-reading of the contents of the device but not always guaranteed. *-Editor*