Burning CDs and DVDS with an encrypted filesystem

BATTENING DOWN THE DISKS

An encrypted hard disk on your server is no help if valuable data on CDs or DVDs falls into the hands of spies.

We'll show you some convenient solutions for encrypting data on removable media.

BY MATTHIAS JANSEN

You may already store sensitive data in an encrypted area of your hard disk – and if you haven't so far, you might start after reading this issue of Linux Magazine. If you use your data on the road, you might store a snapshot on a CD or copy your files to a USB stick. But you should be concerned about the risk of carrying around copies of your data. USB sticks are used just like external hard disks, so protecting a USB stick with encryption is easy. Encrypting CDs and DVDs is more difficult, but you do have some options. This article explores a pair of useful techniques for putting encrypted data on a CD or DVD.

Options

The simplest way to put encrypted data on a CD is to use GPG or a similar tool to individually encrypt the files and then store the results on the CD. This approach is fine for many applications, and it provides adequate security, but the risks become apparent on closer inspection. A user might manually decrypt the file and temporarily store the cleartext on a writable medium. Because the encryption and decryption are handled haphazardly, without the benefit of an encrypted filesystem to enforce security, this option could result in many separate versions of the file – some encrypted and some decrypted. Besides the drawback

Listing 1a: Encryption Patch

```
01 wget 'ftp://ftp.berlios.de/pub/cdrecord/cdrtools-2.01.tar.bz2'
02 wget 'http://burbon04.gmxhome.de/linux/files/
   cdrtools-2.01-encrypt-1.0rc2.diff.gz'
03 tar xjvf cdrtools-2.01.tar.bz2
04 cd cdrtools-2.01
05 zcat ../cdrtools-2.01-encrypt-1.0rc2.diff.gz | patch -p1
```

Terms

AES: The Advanced Encryption Standard (or the Rijndael algorithm) is a symmetric block cipher that supports block and key sizes of 128, 192, and 256 bits.

CBC: Cipher Block Chaining links each encrypted block with the previously encrypted block. This makes it impossible to decipher one block without knowledge of the previous block. The advantage is that two cleartext blocks with the same content will produce different ciphertext. This helps to conceal patterns that occur in the cleartext.

IV: The initialization vector provides a salt value for the CBC. The first round does not have a previous value to work with and uses the IV instead.

of multiple rounds of encryption and decryption, this approach also wastes storage capacity.

A simpler approach of dumping the encrypted block device onto a disk seems workable. Unfortunately, DVD-capable data partitions of less than 8 GBytes are a rarity today. You therefore need to distribute the partition over multiple disks. To read the image, the user has to put the pieces back together again. Again, this approach wastes disk space, although this solution at least avoids the problem of multiple encryption and decryption cycles.

This article examines two more elegant approaches to the problem of CD/DVD encryption. The first technique makes use of an encryption extension to the cdrecord tool. The other method uses AES Pipe, a tool that provides AES encryption for an arbitrary data stream. These techniques provide transparent encryption for read access and are almost as fast as a crypto filesystem, although there is obviously no way to modify the CD later.

Fast as Lightning

The first approach encrypts the data directly while burning a medium using the cdrecord tool. Maximilian Decker wrote a patch [1] for Jörg Schilling's CD recording software to support this form of encryption. The excerpt in Listing 1a shows how to apply the patch. The resulting executable lets users supply a key for the current burning session. The tool uses 256-bit AES encryption in CBC mode (see "Terms").

The GPL'd version of cdrecord will burn CDs but not DVDs. Two patches are required to handle DVDs, both of which have been modified for the new cdrecord 2.01.01a05 version (from the tool's alpha branch).

OSS DVD support [2] adds a feature for burning DVDs, and the encryption patch from the same server adds encryption capabilities (Listing 1b).

The need to patch cdrecord is the only drawback with this approach. Good news for Gentoo users – you just need to set the *on-the-fly-crypt* USE flag when emerging *app-cdr/cdrtools*.

Cryptic Burning

In both cases, the encryption patch extends the *write_track_data(* function

Table 1: Required Packages

Distribution	CD-Record -encstyle=old	CD-Record -encstyle=new	AES-Pipe
Debian		cryptsetup	loop-aes-utils,loop-aes-source
Ubuntu		cryptsetup	loop-aes-utils, loop-aes-source
Gentoo-USE-Flag	old-crypt: utils-linux, sys-fs/ loop-aes	sys-fs/ cryptsetup	crypt: utils-linux, sys-fs/ loop-aes

Listing 1b: OSS DVD and Encryption

```
01 wget 'ftp://ftp.berlios.de/pub/cdrecord/alpha/cdrtools-2.01.01a05.
   tar.bz2'
02 wget 'http://www.crashrecovery.org/oss-dvd/
   cdrtools-2.01.01a05-ossdvd.patch.bz2'
03 wget 'http://crashrecovery.org/oss-dvd/
   cdrtools-2.01.01a01-encrypt-1.0rc1.diff.gz'
04 tar xjvf cdrtools-2.01.01a05.tar.bz2
05 cd cdrtools-2.01.01
06 bzcat ../cdrtools-2.01.01a05-ossdvd.patch.bz2 | patch -p1
07 zcat ../cdrtools-2.01.01a01-encrypt-1.0rc1.diff.gz | patch -p1
```

Listing 2a: Old Variant

```
01 # mkisofs -J -R ~/daten/ | cdrecord dev=/dev/hda -encrypt
   -encstyle=old -encpass=Passwort -
02 [...]
03 NOTE: this version of cdrecord is an inofficial (modified) release of
04       cdrecord and thus may have bugs that are not present in the
05       original version. Please send bug reports and support requests
   to
06       <burbon04 at gmx.de>. For more information please see
07       http://burbon04.gmxhome.de/linux/CDREncryption.html. The
   original
08       author should not be bothered with problems of this version.
09 [...]
10 Starting to write CD/DVD at speed 48 in real TAO mode for single
   session. Last chance to quit;
   starting real write    0 seconds. Operation starts.
11 Turning BURN-Free off
12 Using aes-256-cbc encryption with plain password, plain IV. (e.g.
   cryptoloop >2.4.22, 2.6).
13 [...]
14 Track 01: Total bytes read/written: 42958848/42958848 (20976
   sectors).
15 # mount -t iso9660 /dev/cdrom /media/crypt -o encryption=aes256
16 # ls /media/crypt
17 cat.mpeg             funny_cats.wmv
18 newsreportfromIraq.wmv
19 Karate_Beetle.avi  German_Engineering_Arab_Technology.wmv
20 sexy_nutcracker.mpg
21 felina_in_the_snow.mpg  noteastgermany.mpg
22 Languageproblems.mpg
```

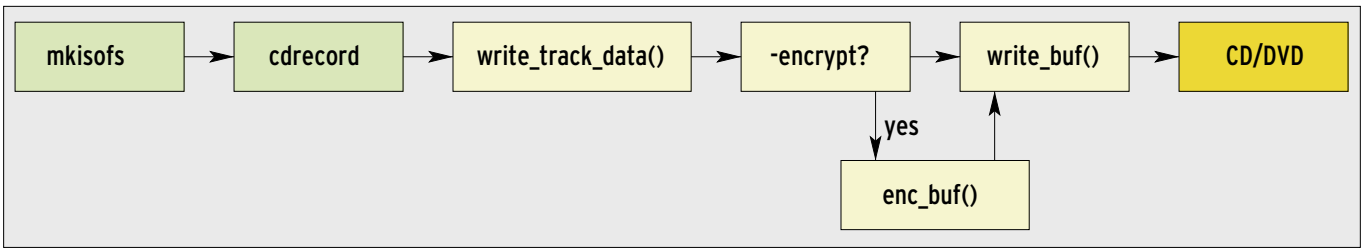

Figure 1: Depending on the -encrypt parameter, the patched version of cdrecord will pass the data through an additional encryption step before burning it onto CD.

(Figure 1). The function burns the internal buffer onto CD/DVD. The patch uses the GPL'd AES implementation by Dr. B. R. Gladman [3] to encrypt data, splitting the payload data into packages of 256 bits (32 bytes) apiece, and applying a 256-bit key. It uses CBC to combine 16 of these packages to create a 512 byte block, and then to combine four blocks to create a 2048-byte block that cdrecord then burns onto the CD.

The restriction to 512 byte units restricts the approach to CD mode 1, which stores blocks of exactly 2048 bytes. Mode 2 uses blocks of 2352 bytes, which are not divisible by 512. The patches cdrecord version supports a number of new parameters. To enable encryption, just set the *-encrypt* flag. There are three methods of providing the required password:

- Cleartext: *-encpass = password*
- Hex value: *-encpasshex = 70617373776 F7264*
- File: *-encpassfile = /home/Name/secret.key* (only the first 32 bytes are significant)

The third method is preferable. The first two mean storing the key in your shell history and process list where a third party might discover it.

If the input key is too short, the patch will pad it out to 32 bytes. The code will interpret any line breaks before the 32-byte boundary as part of the key. This could be tricky if a user attempts to enter the key manually on encryption.

A Question of Keys

The approach distinguishes between two key styles. The old variant uses the key (and any padding) directly, whereas the new style first applies an SHA 256 algorithm (secure hash). The difference becomes apparent on encryption.

Data burnt onto the CD using the old variant can be mounted using the crypto-loop device. The new variant creates a CD for DM-Crypt, the device mapper crypt target.

The older style is only recommended if the target system is running an older kernel. The new DM-Crypt variant requires kernel 2.5.6 or newer, although it does not require any patches. For old-style mounting of the crypto-loop device, created by 256-bit AES, a number of Losetup patches are required. Not every distribution gives you the patches by default (see Table 1). Newer versions of the packages include patches that may be incompatible with the old variant.

Go for New

If your kernel is DM-Crypt capable, you should go for the new variant. Listing 2a

Listing 2b: New Variant

```
01 # mkisofs -J -R /daten/small/Witzig/Videos/ | cdrecord dev=/dev/hda
   -encrypt -encstyle=new -encpass=password -
02 [...]
03 NOTE: this version is an inofficial (modified) release of
04  cdrecord and thus may have bugs that are not present in the
05  original version. Please send bug reports and support requests to
06  <burbon04 at gmx.de>. For more information please see
07  http://burbon04.gmxhome.de/linux/CDREncryption.html. The
08  author should not be bothered with problems of this version.
09 [...]
10 Starting to write CD/DVD at speed 48 in real TAO mode for single
   session. Last chance to quit;
   starting real write    0 seconds. Operation starts.
11 Turning BURN-Free off
12 Using aes-256-cbc encryption with sha-256 hashed key, plain IV.
   (e.g. dm-crypt)
13 [...]
14 Track 01: Total bytes read/written: 42958848/42958848 (20976
   sectors).
15 # cryptsetup -r -c aes -s 256 -h sha256 create ecdrom /dev/cdrom
16 # mount -t iso9660 /dev/mapper/ecdrom /media/crypt
17 # ls /media/crypt
18 cat.mpeg             funny_cats.wmv
19 newsreportfromIraq.wmv
20 Karate_Beetle.avi  German_Engineering_Arab_Technology.wmv
21 sexy_nutcracker.mpg
22 felina_in_the_snow.mpg  noteastgermany.mpg
23 Languageproblems.mpg
24 #
```

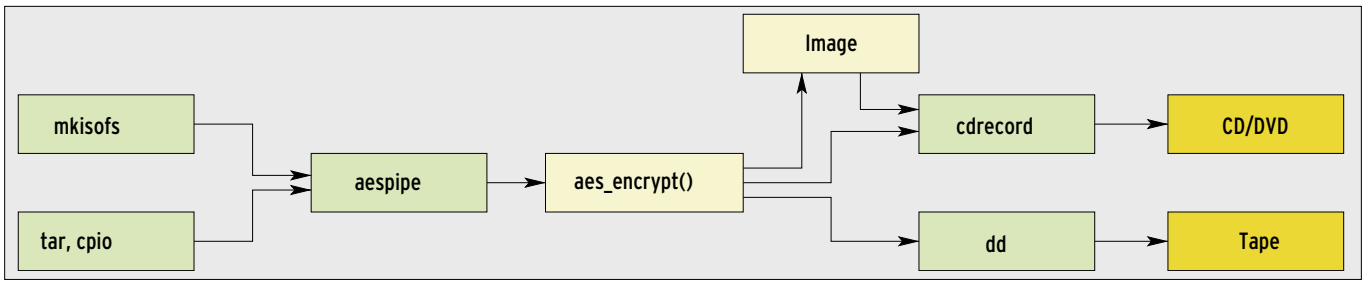

Figure 2: AES Pipe encrypts a data stream from standard input and sends it to standard output, no matter what program creates the stream. You can therefore use AES Pipe for scripting with other CD burning commands.

gives you the whole procedure, from burning to mounting for the old approach, and the procedure for the new approach is summarized in Listing 2b. The first lines in both listings create an ISO filesystem and pass it to cdrecord. The parameter for the old variant is *-encstyle = old* (Listing 2a); this is *-encstyle = new* for the new variant. In both cases, *-encrypt* enables encryption.

The differences become apparent on mounting an encrypted CD. Listing 2a uses an extended mount command (Line 15), whereas the variant in Listing 2b calls Cryptsetup (Line 15) to prepare for mounting via the device mapper (Line 16). Note that the *-r* (Read-Only) parameter is not available in the current version 0.1 of Cryptsetup. Without this option, Cryptsetup will refuse to create the mapper for write-protected media.

To avoid building the CVS version, you can map the CD to a loop device, using *losetup* as a workaround, and use Cryptsetup. (See Listing 2c, Line 9.)

Pitfalls

A possible pitfall of the DM-Crypt method becomes apparent if a file is used to pass in the key to Cryptsetup. Cryptsetup will then ignore the *-h* (hash algorithm) parameter and apply the key as a cleartext key without any hashing. In contrast to this, cdrecord interprets the value as key material, which it runs through a hash function. Decryption is impossible in this case. To avoid this issue, you might prefer to store the hash value in the file in both cases and use *-encstyle = old* or *encstyle = aes256-cbc-plain* with cdrecord. Cryptsetup will then use the key from the file as intended.

As this method encrypts on the fly during the burning process, it is easily

Listing 2c: New with losetup

```
01 # mkisofs -J -R /daten/small/Witzig/Videos/ | cdrecord dev=/dev/hda
   -encrypt -encstyle=new -encpass=password -
02 [...]
03 Starting to write CD/DVD at speed 48 in real TAO mode for single
   session.
04 Last chance to quit, starting real write    0 seconds. Operation
   starts.
05 Turning BURN-Free off
06 Using aes-256-cbc encryption with sha-256 hashed key, plain IV.
   (e.g. dm-crypt)
07 [...]
08 Track 01: Total bytes read/written: 42958848/42958848 (20976
   sectors).
09 # losetup /dev/loop0 /dev/cdrom
10 # cryptsetup -c aes -s 256 -h sha256 create ecdrom /dev/loop0
11 # mount -t iso9660 /dev/mapper/ecdrom /media/crypt
12 # ls /media/crypt
13 cat.mpeg              funny_cats.wmv
14 newsreportfromIraq.wmv
15 Karate_Beetle.avi  German_Engineering_Arab_Technology.wmv
16 sexy_nutcracker.mpg
17 felina_in_the_snow.mpg  noteastgermany.mpg
18 Languageproblems.mpg
```

integrated with programs that use cdrecord as a burning tool. You just need to add the command line arguments for cdrecord as user-definable parameters.

If you can't modify cdrecord, you will prefer the second method, which uses the AES Pipe program [4]. Use any burning software to drop the encrypted AES Pipe image onto a CD or DVD.

Encrypted Pipes

AES Pipe encrypts an arbitrary data stream from Stdin and outputs the results on Stdout (Figure 2). Output from *mkisofs* is suitable. The program uses CBC mode and links 16 32-byte packets to form 512 byte blocks. Just like the patched cdrecord version, this method is only useful for CD mode 1.

Key lengths of 128, 192, and 256 bits are supported, as are several hashing functions. If you do not tell it to do otherwise, AES Pipe defaults to the AES algorithm with a 128-bit key and SHA 256 as the key hashing algorithm. Line 1 of Listing 3 shows the most simple version with simple password input (Line 2). The password must contain at least 20 characters. Line 8 shows that the results can be mounted.

AES Pipe also has a *-K* option for passing in a GPG encrypted file as the key (Listing 4a). This method gives users the ability to transport keys safely and securely – on USB sticks, for example. A thief would need the encrypted CD, the password, the USB stick, and the passphrase (mantra) for the key.

Listing 3: AES Pipe with Password

```
01 # mkisofs -J -R . | aespipe | cdrecord dev=/dev/hdd -
02 Password:
03 cdrecord: Asuming -tao mode.
04 cdrecord: Future versions of cdrecord may have different drive
   dependent defaults.
05 Cdrecord-Clone 2.01.01a06 (x86_64-unknown-linux-gnu) Copyright (C)
   1995-2006 Jörg Schilling
06 [...]
07 Track 01: Total bytes read/written: 475136/614400 (300 sectors).
08 # mount -t iso9660 /dev/hdd /mnt/crypted -o loop,encryption=AES128
09 Password:
10 # ll /mnt/crypted/
11 total 22
12 drwxr-xr-- 2 maz network  2048  1. Apr 20:21 Images
13 -rw-r--r-- 1 maz network 18064 25. Jul 12:35 Crypto-CD.txt
14 -rw-r--r-- 1 maz network   349  8. Jan 2006  distris.txt
15 -rw-r--r-- 1 maz network   649 30. Mar 20:23 patch.txt
```

The GPG variant adds the ability to specify multiple passwords for one CD. To do this, you would need to store the main key in multiple GPG archives and protect the archives with different passwords. You could even issue a new key, without needing to burn a new CD, if a user forgot their password. This method also supports safe mailing of storage media if you use Public Key encryption to protect the GPG file.

Building on this idea, admins could write the GPG file at the start of the CD (Listing 4b). This would mean putting the encryption key on the CD to be encrypted, but protecting the key with a robust password. To do this, admins would need to create a fixed-length file (Line 1) with the GPG file at its beginning (Line 2), and then append the encrypted image (Line 3), before burning the results onto a CD/DVD (Line 4).

On mounting the CD, the user then specifies the CD drive both as the source and as the device (Line 6). An offset of 8192 (16 times 512) lets mount find the start of the encrypted data. Check out [5] for a Bash script that combines the major steps. This approach adds a dependency on top of the ones in Table 1 (GPG), and you will need to patch the *losetup* and *mount* tools. Most distributions have the required support built in.

Not Just Disks

AES Pipe will encrypt any data stream, so you can create encrypted tar archives:

```
tar cj /data | aespipe > ➥
data.tar.bz2.enc
aespipe -d ➥
< data.tar.bz2.enc | tar xj
```

Encrypted tar backups onto a medium are another good idea:

```
tar cj /data| aespipe| dd ➥
of=/dev/st0 bs=56
dd if=/dev/st0 bs=➥
56 | aespipe -d | tar tj
```

AES Pipe also supports Bzip 2 compression. Compression only makes sense prior to encryption as there is no way of compressing encrypted data. Additionally, this variant prevents tar warnings about it ignoring gargabe data. Messages such as *bzip2: (stdin): trailing garbage after EOF ignored* occur because AES Pipe uses a fixed block size of 16 bytes and pads missing bytes with null values. This confuses tar and the Bzip 2 function.

Compression will not work with ISO images since CDs and DVDs lose the ability to be mounted via *losetup*. If you need to compress data, do it at the filesystem level.

Paranoia

AES Pipe has a number of settings to make life hard for potential attackers. For example, you can use 64 encryption keys. In Multi-Key mode, the program encrypts the first sector with the first key, the second sector with the second key, and so on. Instead of the normal initialization vector (IV, see "Terms") it uses an MD5 Hash IV.

However, this mode only works in combination with a GPG-encrypted file. Multi-Key mode is automatically used if the GPG file contains at least 64 keys, with at least 20 characters apiece, separated by newlines.

Another way of making brute force attacks more difficult is to pass keys though several thousand rounds of AES, where the *-C factor* parameter specifies the number of thousands. This approach increases CPU load prior to encryption, but it also makes life far more difficult for an attacker trying out different keys. The program will also use a seed, specified by the *-S* parameter, if needed. However neither of these modes will work in Multi-Key mode.

Listing 4b: AES Pipe, GPG, and Burning

```
01 # yes "" | dd of=image.iso bs=512 count=16
02 # head -c 2925 /dev/urandom | uuencode -m - | head -n 66 | tail -n 1
   | gpg --symmetric -a | dd of=image.iso conv=notrunc
03 # mkisofs -iso-level 3 -l -r /daten/ | aespipe -e aes256 -w 5 -K
   image.iso -O 16 >> image.iso
04 # growisofs -dvd-compat -Z /dev/dvdrw=image.iso
05 [...]
06 # mount -t iso9660 /dev/hdd /mnt/crypted -o loop,gpgkey=/dev/hdd,enc
   ryption=AES256,offset=8192
07 Password:
08 # ll /mnt/crypted/
09 total 22
10 drwxr-xr-- 2 maz network  2048  1. Apr 20:21 Images
11 -rw-r--r-- 1 maz network 18064 25. Jul 12:35 Crypto-CD.txt
12 -rw-r--r-- 1 maz network   349  8. Jan 2006  distris.txt
13 -rw-r--r-- 1 maz network   649 30. Mar 20:23 patch.txt
```

Listing 4a: AES Pipe with GPG Key

```
01 # mkisofs -J -R . | aespipe -K ~/test.gpg | cdrecord dev=/dev/hdd -
02 Password:
03 cdrecord: Asuming -tao mode.
04 cdrecord: Future versions of cdrecord may have different drive
   dependent defaults.
05 Cdrecord-Clone 2.01.01a06 (x86_64-unknown-linux-gnu) Copyright (C)
   1995-2006 Jörg Schilling
06 [...]
07 Track 01: Total bytes read/written: 475136/614400 (300 sectors).
08 # mount -t iso9660 /dev/hdd /mnt/crypted -o loop,encryption=AES128
09 Password:
10 # ll /mnt/crypted/
11 total 22
12 drwxr-xr-- 2 maz network  2048  1. Apr 20:21 Images
13 -rw-r--r-- 1 maz network 18064 25. Jul 12:35 Crypto-CD.txt
14 -rw-r--r-- 1 maz network   349  8. Jan 2006  distris.txt
15 -rw-r--r-- 1 maz network   649 30. Mar 20:23 patch.txt
```

Conclusions

Both the modified version of cdrecord and the combination of an unmodified burning with AES Pipe are useful for creating encrypted disks and mounting them transparently on the filesystem. Thanks to its generic nature, AES Pipe gives you more options. On the other hand, the cdrecord patch is hard to beat for user-friendliness, especially considering the ease with which it jells with front-end burning tools such as K3B. But make sure you go for the *-encstyle = new* variant to avoid incompatibility. ■

INFO

[1] Maximilian Decker, "On-the-fly encryption for cdrecord/cdrtools": *http://burbon04.gmxhome.de/linux/CDREncryption.html*

[2] DVD patch for the OSS version of cdrecord/CDR Tools: *http://www.crashrecovery.org/oss-dvd.html*

[3] AES implementation by Dr. Gladman: *http://fp.gladman.plus.com/AES/*

[4] AES Pipe by the Loop AES project: *http://loop-aes.sourceforge.net*

[5] Bash script for creating an encrypted image: *http://matthiasjansen.de/~maz/create_enc_image*