Cryptoloop HOWTO

Ralf Hölzer

<cryptoloop@ralfhoelzer.com>

2004−01−15

Revision History

Revision 1.2	2004−03−12	Revised by: rh
Added information on dm−crypt, updated loop−AES info, added more info on security		
Revision 1.1	2004−01−24	Revised by: rh
Updated information on patching util−linux, Loop−AES, Best Crypt		
Revision 1.0	2004−01−17	Revised by: rh
Initial release, reviewed by TM at LDP.		
Revision v0.9	2004−01−15	Revised by: rh
Updated and converted to DocBook XML.		

This document explains how to create encrypted file systems using the Cryptoloop functionality. Cryptoloop is part of the CryptoAPI in the 2.6 Linux kernel series.

Table of Contents

1. About this document

This HOWTO describes how to use the Cryptoloop loop device encryption in the 2.6 Linux kernel series. Cryptoloop makes it possible to create encrypted file systems within a partition or another file in the file system. These encrypted files can the be moved to a CD, DVD, USB memory stick, etc. Cryptoloop makes use of the loop device. This device is a pseudo−device which serves as a "loop" through which each call to a the file system has to pass. This way, data can be processed in order to encrypt and decrypt it. Since kernel 2.6, the Crypto API has been integrated into the main kernel, and setting up an encrypted file system has become much easier. No additional kernel patches are required. An update of some userspace utilities is necessary. Unfortunately, the use of Cryptoloop is not very well−documented so far. This HOWTO is an attempt to make it easy everyone to create an encrypted file system using the standard Cryptoloop functionality. Cryptoloop is based on the Crypto API in the 2.6 Linux kernel. It should not be confused with Loop−AES, which is a completely separate project. Cryptoloop is similar to the Crypto API that was available as a separate patch for the 2.4 kernel series. The new version is not compatible with the older one.

1.1. Copyright and License

1.2. Disclaimer

No liability for the contents of this document can be accepted. Use the concepts, examples and information at your own risk. There may be errors and inaccuracies, that could be damaging to your system. Proceed with caution, and although this is highly unlikely, the author(s) do not take any responsibility.

All copyrights are held by their by their respective owners, unless specifically noted otherwise. Use of a term in this document should not be regarded as affecting the validity of any trademark or service mark. Naming of particular products or brands should not be seen as endorsements.

1.3. Credits / Contributors

I'd like to thank the following people who helped me improve this HOWTO:

- Dennis Kaledin
- Binh Nguyen
- David Lawyer
- Tabatha Marshall
- Kian Spongsveen

1.4. Feedback

Feedback is most certainly welcome for this document. Send your additions, comments and criticisms to the following email address : `<cryptoloop@ralfhoelzer.com>`.

2. Introduction

There are currently a few alternatives to using Cryptoloop. Loop−AES (http://loop−aes.sourceforge.net) is probably the most well−known. It provides very similar functionality to Cryptoloop. Aes−loop is currently more mature than Cryptoloop and it is also faster (about twice as fast, according to the author of loop−AES), because it uses a highly optimized assembler implementation for AES. This doesn't mean that Cryptoloop is slow. I have not noticed any significant speed differences between a Cryptoloop−encrypted partition and a non−encrypted partition during everyday work with normal amounts of I/O. Unless I/O performance is extremely important to you, Cryptoloop should do just fine. Loop−AES offers some additional features that are not yet present in the kernel implementation of Cryptoloop. Loop−AES requires modified userspace tools (mount, losetup) and these modifications are incompatible with Cryptoloop. You will not be able to use Cryptoloop and Loop−AES at the same time.

In terms of security, Cryptoloop is doing ok. The key is usually generated from a password and its hash is used as the key to AES. This leads to the possibility of a known−plaintext attack. Loop−AES is superior in this regard, because it generates a random key and encrypts this key separately, making a known−plaintext attack more difficult. Loop−AES also supports a multi−key mode, where sectors are encrypted with 64 separate AES keys. In general, a brute−force attack on your password can be very effective, if you choose a weak password. To be on the safe side, your password should be at least 20 characters long. Otherwise a brute−force attack on the password will be much easier than trying to brute−force the AES encryption directly.

The Cryptoloop functionality in the standard kernel provides a stable and clean implementation without the need for extra patches. Since it is still fairly new, it may not have gotten the necessary amount of review in terms of security. You have to decide for yourself what is suitable for you.

IMPORTANT: Cryptoloop has been marked deprecated in the latest 2.6 kernel. This means that it will no longer be maintained actively. The successor to Cryptoloop will be dm−crypt. Dm−crypt is available in the main kernel since 2.6.4. Cryptoloop will still be available in the main kernel for a long time, but dm−crypt will be the method of choice for disk encryption in the future. Dm−crypt is based on the device mapper and offers pretty much the same functionality as Cryptoloop. It is still very new and there are no easy−to−use userspace tools available yet. Dm−crypt is considered to be much cleaner code than Cryptoloop, but there are some important differences. For example, creating an ecrypted filesystem within a file will still require to go through a loop device, but this support is still in development.

There are other tools which allow you to create an encrypted file system. BestCrypt is a commercial product from Jetico. It allows you to create encrypted containers and has a large choice of ciphers. It also offers some nifty features such as hidden containers. It is available for Windows and Linux, which makes it suitable for interchanging encrypted containers between Windows and Linux. BestCrypt now compiles on 2.6 kernels as well. Cryptoloop can also create containers that can be moved around, by creating the encrypted file system within a file as described below. I don't know of a way to access the Cryptoloop−encrypted files from other operating systems such as Windows. In this case, BestCrypt may be your only choice.

There are other commercial disk encryption tools such as PGP disk, but to my knowledge there is no Linux support for them.

3. Configuring the kernel

In order to use Cryptoloop, you need to activate a few kernel options. You have the option to either compile these requirements as modules or compile them directly into the kernel. The following steps enable them as modules. If you are not familiar with building a 2.6 kernel, you should refer to the Linux Kernel HOWTO. The following instructions just give the minimal steps.

1. Go to the directory that holds your kernel source tree (usually `/usr/src/linux/`) and start the configuration:
   ```
   make menuconfig
   ```
2. Enable general loop device support. Active "Loopback device support" under:
   ```
   Device Drivers -> Block Devices -> Loopback device support
   ```
3. Enable Cryptoloop support in the same section. The option should show up as soon as you enable general loopback support.
4. Enable the cryptographic API by going to "Cryptographic options" from the main menu. You can safely enable most algorithms here. I would recommend that you enable the following:
   ```
       -- Cryptographic API
        <*>   HMAC support
        < >   Null algorithms
        <*>   MD4 digest algorithm
        <*>   MD5 digest algorithm
        <*>   SHA1 digest algorithm
        <*>   SHA256 digest algorithm
        <*>   SHA384 and SHA512 digest algorithms
        <*>   DES and Triple DES EDE cipher algorithms
        <*>   Blowfish cipher algorithm
        <*>   Twofish cipher algorithm
        <*>   Serpent cipher algorithm
        <*>   AES cipher algorithms
        <*>   CAST5 (CAST-128) cipher algorithm
        <*>   CAST6 (CAST-256) cipher algorithm
        <*>   Deflate compression algorithm
        < >   Testing module
   ```
   If you decide to make them as modules, make sure you load the appropriate modules (cryptoloop, aes, etc.) at startup before you continue.
5. Make your kernel and modules and install them. For example, if you are using lilo on a x86 machine, this can be done like this:
   ```
        make
        make modules_install
        cp arch/i386/boot/bzImage /boot/kernel-2.6.1
        lilo
   ```
6. Load the required modules at startup. This is handled differently on the various distributions. For example, on Gentoo these modules can be added to `/etc/modules.autoload/kernel-2.6`. If you have compiled Cryptoloop as a module, it will have to be loaded first. It will automatically load the basic loop device module as well. You can manually load the module with:
   ```
   modprobe cryptoloop
   ```

---

4. Getting the user space tools

The Cryptoloop driver requires updated userspace tools to actually create and mount the encrypted file system. An updated util−linux package is needed and can be obtained from http://ftp.cwi.nl/aeb/util−linux/util−linux−2.12.tar.gz. The most current version is 2.12. There will be new versions out soon that will probably introduce major changes, so make sure you check this HOWTO for updates before upgrading to a newer version. Unfortunately there are many patches for util−linux out there. There are differences in the way how encrypted partitions are created and mounted. In order to use util−linux 2.12 with a 2.6 kernel at least the following two patches need to be applied:

1. Combined losetup patch
2. Util−linux 2.6 patch

Download the util−linux package and the two patches above. First extract the util−linux package and then apply the two patches:

```
tar xvfz util-linux-2.12.tar.gz

cd util-linux-2.12

patch -p1 < /path_to_patchfile/losetup-combined.patch

patch -p1 < /path_to_patchfile/util-linux-2.12-kernel-2.6.patch
```

After applying the patches, compile and install util−linux according to the instructions in the INSTALL file.

I recommend to use Gentoo Linux, which automatically applies these patches when emerging the util−linux patches. Other distributions may have versions of util−linux available, that have these patches aleady applied as well.

5. Setting up the loop device

Cryptoloop can be used either on a file or an entire file system. The following describes how to set it up on a particular partition. This partition can be any partition you like; the following example uses `/dev/sda1`. I have chosen to use AES as a cipher, but you can substitute any cipher you like that has been enabled in the kernel. You can get a list of the algorithms supported by your currently running kernel by looking into `/proc/crypto`. An excellent resource, discussing the different cryptographic algorithms, are Bruce Schneier's books, Applied Cryptography and Practical Cryptography. Both AES and Serpent are probably a reasonable choice. AES has been cryptanalyzed a lot and no serious weaknesses have been discovered so far. Serpent has not been analyzed as much, but is considered to be even a little bit stronger than AES. However, Serpent is also slower than AES. Stay away from DES, it is both slow and weak. Triple−DES may be an option, but AES is probably more secure and faster, so there is really no reason to use Triple−DES anymore.

1. It is recommended that you format your partition and fill it with random data before you create the encrypted file system on it. This will make it harder for an attacker to detect patterns in your encrypted partition.

   WARNING!

   Be careful what you type here for your partition. If you do make a mistake, you can easily overwrite the wrong partition with random garbage!

   Filling a partition with random data can be done as follows:

   ```
   dd if=/dev/urandom of=/dev/sda1 bs=1M
   ```

   You may get an error message that the device is full. You can ignore it.
2. Select a cipher and key size. A list of ciphers supported by your kernel can be obtained from `/proc/crypto`. I recommend that you use AES with a 256−bit key.
3. Set up the loop device. This is done using the **losetup** command from the util−linux package. The following command creates an encrypted filesystem using the loop device 0 using the AES cipher with a 256−bit key on the device `/dev/sda1`:
   ```
   losetup -e aes-256 /dev/loop0 /dev/sda1
   ```
   The command prompts for a password. Select a strong password and try to remember it without having to stick a Post−It note to your monitor. There is one big downside to using Cryptoloop. Since the password is hashed to create the encryption key, it is not easy to change the password later on. The most straight−forward way of changing the password is to create a new encrypted partition or file and move all data into it. For this reason, make sure you select a strong password from the start. AES may be a strong algorithm, but if you chose a weak password, that security goes down the drain.

   If **losetup** fails with an INVALID ARGUMENT error message, there is a problem with your util−linux package. Make sure you have followed the instructions above on how to install a patched version of util−linux. Older and unpatched version use a different way of passing the key size, and do not work with the 2.6 Crypto API.
4. Create a file system. You can chose whatever file system you like. The following creates an ext3 file system using the loop device:
   ```
   mkfs.ext3 /dev/loop0
   ```
5. Mount the encrypted file system. First you need to create a mount point, such as `/mnt/crypto`:
   ```
   mkdir /mnt/crypto
   ```

Then you need to mount the file system. At this stage you need to tell mount explicitly which loop device to use:

```
mount -t ext3 /dev/loop0 /mnt/crypto
```

6. You can now play with your encrypted file system until you are bored.
7. Unmount the file system. After you are done playing, unmount the filesystem:

```
umount /mnt/crypto
```

8. Detach the loop device. The loop device is still attached to your partition. Detach it with:

```
losetup -d /dev/loop0
```

6. Mounting the encrypted file system

For all operations on the Cryptoloop device, it is important that the necessary modules are loaded. You need to load at least the Cryptoloop module and the modules for each cipher with **modprobe**. If the features are compiled directly into the kernel, this is not necessary.

In order to mount the encrypted file system created above, you can use the standard mount command from util−linux:

```
mount −t ext3 /dev/sda1 /mnt/crypto/ −oencryption=aes−256
```

You will be prompted for the password and the file system will be mounted just as any other. Since the encryption option implies that this is a Cryptoloop filesystem, it will automatically pick an available loopback device.

When you are done, unmount it with:

```
umount /mnt/crypto
```

You can add the following line to `/etc/fstab`:

```
/dev/sda1               /mnt/crypto     ext3            noauto,encryption=aes−256       0 0
```

Now you can simply mount the device with:

```
mount /mnt/crypto
```

That's it. Have fun.

---

7. Using a file instead of a partition

It is just as easy to create an encrypted file system within a file on another file system. This is especially useful if you want to back up this file by burning it to a DVD, etc. You can then easily move the file around to other machines as well.

To initially create a 100MB file containing random data use the following command:

```
dd if=/dev/urandom of=/mystuff.aes bs=1k count=100000
```

If you want to change the size of the file, change the `count` value accordingly. The above command creates 100000 blocks of 1k in size, but you can change this to whatever you like. Just make sure it is not too small to hold the file system you chose. You can choose any file name and path you want instead of `/mystuff.aes` as long as there's enough space on the partition.

You can then create the encrypted file system within this file, similar to the way it is done above:

```
losetup -e aes-256 /dev/loop0 /mystuff.aes
```

Now you can create the file system:

```
mkfs.ext3 /dev/loop0
```

and mount it:

```
mount -t ext3 /dev/loop0 /mnt/crypto
```

Finally, unmount and detach the loop device:

```
umount /mnt/crypto
losetup -d /dev/loop0
```

You can then mount the file system later on as follows:

```
mount /mystuff.aes /mnt/crypto -oencryption=aes-256
```

If you want to move the file or burn it to a CD or DVD, make sure you **unmount** it first.