Lokale Verschlüsselung unter Linux

dies ist ein kurzes Tutorial zum praktischen Einsatz von AES und wie man es verwendet um unter Linux einen verschlüsselten Ordner/Container einzurichten. Damit das Funktioniert muss der Kernel schon so konfiguriert sein, das er das unterstützt. Weiterhin sollte dies zwar für alle Distributionen gelten, habe es dennoch nur unter Debian und SuSE ausprobiert.

Anmerkung: Hier wird zum Verschlüsseln AES verwendet, doch man kann auch jedes andere Kryptoverfahren wie zum Beispiel RC6 oder Blowfish verwenden.

Bevor man mit das Loopback-Device konfiguriert, sollte man zunächst folgenden Eintrag in die Konfigurationsdatei */etc/fstab* hinzufügen:

vi /etc/fstab

Code:
... */dev/loop0 /mnt/crypt reiserfs user,noauto,rw,loop 0 0* ...

Zuerst legt man einen leeren Container an, in den später die streng vertraulichen Daten gespeichert werden sollen (als Beispiel dient die Datei ".safe", 50 MB groß - count in MegaByte):

Code:
dd if=/dev/urandom of=/etc/.safe bs=1M count=50

Um die Größe zu variieren, muss man die Parameter "bs=" und "count" entsprechend anpassen (die Faustformel lautet dabei bs*count=Filegröße). Des Weiteren wird zur Erstellung des Files das */dev/urandom*-Device benutzt, um das File mit zufällig gewählten Zeichen zu füllen. Dieser Vorgang nimmmt zwar einige Zeit in Anspruch, ist aber weit mehr sicherer als die Verwendung des */dev/zero*-Devices, welches das gesamte File mit Nullen ausfüllt und dementsprechend schneller ist.

Ein potenzieller Angreifer könnte nämlich anhand der genauen Analyse des File feststellen, welche Blöcke bereits belegt sind (welche nicht mehr durch Nullen ausgefühlt sind), und dies könnte ihm wiederum mehr Informationen zukommen lassen als unbedingt notwendig.

Damit die Verschlüsselung funktioniert, müssen die Module "cryptoloop" und "aes" geladen sein:

Code:
modprobe cryptoloop;modprobe aes

(Um die Module immer - also auch nach einem Neustart - zur Verfügung zu haben, müssen diese in der Datei */etc/modules* eingetragen werden.)

Jetzt verbinden wir die Datei /etc/.safe als Loopback-Device. Das Loopback-Device ist für die Online-Verschlüsselung zuständig. Außerdem ermöglicht ein Loopback-Device, eine Datei wie ein Device anzusprechen, also in unserem Fall, mountbar zu machen.

Das muss nicht nur bei der ersten Anlage der Datei, sondern auch vor jedem späteren Zugriff darauf erfolgen:

Code:

```
# losetup -e aes /dev/loop0 /etc/.safe
```

Wichtig: Das Kennwort muss mindestens 8 Zeichen betragen, ansonsten wird es nicht angenommen. Also haltet euch am besten an die Regeln zum erstellen eines sicherren Kennworts.

Da das Loopback-Device (das auch unter /dev/loop/0 liegen könnte) verschlüsselt werden soll (*-e aes*) fragt *losetup* jetzt nach dem Passwort, mit dem */etc/.safe* kodiert werden soll.

Wichtig: Dieses Passwort sollte man unter garkeinen Umständen vergessen, da man ansonsten später nie mehr (und das heisst auch nie mehr) an seine Daten im Container gelangt. Es sei den AES wird irgendwann geknackt ;-)

Um Dateien in /etc/.safe abspeichern zu können, muss darauf ein Filesytem existieren. Ich nehme hier im Beispiel einfach reiserfs - das bietet sich unter Linux ja auch an:

Code:

```
# mkreiserfs /dev/loop0
```

Dieser Befehl darf natürlich nur bei der Neuanlage des Containers ausgeführt werden, da sonst das gleiche wie für Passwortvergesser gilt. Ja gut, man hätte hier nachträglich wenigstens die Chance, wieder an die Daten zu kommen...

Jetzt gilt für den Container, wie für jeden anderen "Datenträger" auch:

Code:

```
# mkdir /mnt/crypt
# mount /dev/loop0 /mnt/crypt
```

Alle Dateien, die nun nach /mnt/crypt kopiert oder wahrscheinlicher verschoben werden, sind nun automatisch verschlüsselt. Natürlich kann man auch die zu verschlüsselnde Datei direkt auf dem Filesystem erstellen.

Code:

```
#vi vertraulich
Diese Datei ist vertraulich!
```

Damit der Einsatz des Containers seinen Sinn erfüllt und nur Root seine Dateien darauf verschlüsselt ablegen darf, werden die Rechte noch entsprechend geändert:

Code:

```
#chmod go= /etc/.safe
#chown root -R /etc/.safe
#chmod go= -R /etc/.safe
#chmod go= /mnt/crypt
```

Um die Daten auf dem Verschlüsselten Filesystem vor einem unerlaubten Zugriff zu schützen, sollte man stets darauf Achten, das Filesystem nur dann anzuhängen, wenn man Zugriff auf die sich darauf befindlichen Daten benötigt. Benutzt man keine Daten auf dem Filesystem, so sollte man dieses stets abhängen, um somit einen unerlaubten Zugriff gänzlich auszuschließen.

Möchte man den Container wieder schließen, reicht ein einfacher Unmount, sowie das Lösen von /etc/.safe vom Loop-Device:

Code:

```
# umount /mnt/crypt
# losetup -d /dev/loop0
```

Bei späterer Benutzung reicht es dann, die Datei mit

Code:

```
losetup -e aes /dev/loop0 /etc/.safe
```

wieder an ein Loopback-Device zu binden, das Passwort einzugeben und danach zu mounten.

Um den Voragang des Ver- und Entschlüsselns zu verkürzen, schreiben wir uns ein kleines Shell-Script.

cd /sbin
vi encrypt

Code:

```
#!/bin/sh
# /sbin/encrypt
#
# Script zum Initialisieren des verschlüsselten Loopback-Filesystems
MOUNT=$(which mount)
LOSETUP=$(which losetup)

$LOSETUP -e aes /dev/loop0 /etc/.safe
$MOUNT  /dev/loop0 /mnt/crypt
echo "Crypto start."
```

cd /sbin
vi decrypt

Code:

```
#!/bin/sh
# /sbin/decrypt
#
# Script zum Deaktivieren des verschlüsselten Loopback-Filesystems
UMOUNT=$(which umount)
LOSETUP=$(which losetup)

$UMOUNT  /mnt/crypt
$LOSETUP -d /dev/loop0
echo "Crypto stop."
```

chmod 700 encrypt
chmod 700 decrypt

Von nun an lässt sich das verschlüsselte Loopback-Filesystem mit dem Kommando encrypt starten und mit decrypt wieder deaktivieren.

mfg duddits