

Dieser Artikel ist Teil der **HOWTO Sammlung**

Inhaltsverzeichnis

- 1 Rechtliches
- 2 DM-Crypt und Cryptsetup-LUKS
 - ◆ 2.1 Dm-Crypt, cryptsetup und die Schwächen
 - ◆ 2.2 May the force be with you, LUKS
- 3 Einrichtung
 - ◆ 3.1 Komplettes Listing
- 4 Zusätze
 - ◆ 4.1 Weitere Schlüssel
 - ◆ 4.2 Informationen
 - ◆ 4.3 Systemlast senken
 - ◆ 4.4 Beim Systemstart mounten
 - ◆ 4.5 Einhängen von externen Datenträgern
- 5 Siehe auch
- 6 Links

Rechtliches

Wer sich Gedanken um die rechtliche Seite macht, der sollte sich einer Tatsache bewusst sein, dass es ein Recht ist, seine eigenen Daten geheim zu halten. Dies schlägt sich unter anderem im Gesetz zum Postgeheimnis nieder und dies ist ein Grundrecht.

Ein in politischen Diskussionen häufig genannter Einwand gegen die private Nutzung von Datenverschlüsselung ist die Argumentation, dass nur die Menschen etwas verstecken, die auch was zu verstecken haben.

Datenschutzrechtler halten diese Argumentation sowohl datenschutzrechtlich als auch verfassungsrechtlich für sehr gefährlich. Begründet wird diese Haltung damit, dass eine derartige Einstellung im Gegensatz zu der freiheitlich-demokratischen Grundordnung Deutschlands steht, die durch das Grundgesetz garantiert wird. Weiterhin wird darauf verwiesen, dass in Deutschland das Rechtsstaatsprinzip herrscht und somit auch in Hinsicht auf die private Verschlüsselung von Daten die Unschuldsvermutung gilt.

DM-Crypt und Cryptsetup-LUKS

Dm-Crypt, cryptsetup und die Schwächen

DM-Crypt ist ein Verfahren, das ein in den Kernel integriertes Modul zur Verschlüsselung von Geräten nutzt. Dabei ist es völlig unerheblich, ob das "Gerät" eine Partition, eine Festplatte, eine virtuelle Partition oder sonst etwas ist.

Da aber bei DM-Crypt der Schlüssel als Hex-Zeichenkette mit fester Länge erwartet wird, und sich so etwas niemand merken kann, gibt es das Programm *cryptsetup*, das diesen Schlüssel aus einem beliebig (langem) Passwort generiert. Dadurch kann der Anwender auch Passwörter nutzen, die man sich wenigstens etwas einfacher merken kann.

Ein Problem bleibt aber bestehen: *cryptsetup* kann den verschlüsselten Schlüssel aus naheliegenden Gründen nicht auf der verschlüsselten Platte hinterlegen. Man muss also immer darauf achten, dass man den Schlüssel bei der Platte mit transportiert, was z.B. für tragbare Festplatten sehr umständlich ist.

May the force be with you, LUKS

Die Lösung ist **cryptsetup-luks**. Dieses Paket bringt ein um einige Funktionen erweitertes [cryptsetup](#) mit sich. Das *Linux Unified Key Setup* ist dabei ein formaler Standard, der von *cryptsetup-luks* implementiert wird. Er unterscheidet sich vom alten [cryptsetup](#) dadurch, dass die verschlüsselten Partitionen einen *Header* bekommen, der alle notwendigen Informationen wie Verschlüsselungsalgorithmus, LUKS-Version, Verschlüsselungsmodus und andere beinhaltet. Außerdem gibt es 8 so genannte Key-Slots, welche Kopien der Schlüssel beinhalten, die man zum Entschlüsseln der Platte benötigt.

Der Vorteil dabei ist, dass man so nur das Passwort im Kopf haben muss. Außerdem kann man mit dieser Methode gleich mehrere Passwörter für eine Partition angeben, was z.B. bei mehreren Nutzern, in Firmen und Institutionen oder aber für Notfälle sehr nützlich ist.

Es ist zum Beispiel sinnvoll, sich ein normales, sicheres Passwort auszudenken, und dann für den Notfall noch ein weiteres Backup-Passwort, das man nutzt, wenn man das erste vergisst. Eine Idee sind da z.B. als Schlüssel die ISO-Images der drei Lieblings-CDs oder das ISO-Image der ersten Linux-CD oder der komplette zweite Akt von Romeo und Julia, oder oder oder. Dabei sollte das zweite Passwort natürlich nicht leichter zu erraten sein, als das erste...

Da *cryptsetup-luks* aktuell auch in die meisten Distributionen einfließt, ist zu erwarten, dass man demnächst eine *cryptsetup-luks*-bearbeitete tragbare Festplatte an jedem Linux-Rechner ohne Probleme anhängen kann.

Einrichtung

Zuerst muss geprüft werden, ob man alle wichtigen Pakete installiert hat. Ein

```
[root]# yum install cryptsetup-luks
erledigt das im Zweifelsfall.
```

Verschlüsselte_Festplatten

Ich gehe davon aus, dass man eine vorliegende Partition (im Beispiel `/dev/sda2`) hat, die verschlüsselt und dann mit einem Dateisystem belegt werden soll.

```
cryptsetup -c aes-cbc-essiv:sha256 -y -s 256 luksFormat /dev/sda2
```

Dieser Befehl erstellt eine verschlüsselte Partition im Gerät `/dev/sda2`. Der Parameter `-c` bestimmt den Algorithmus, `-y` lässt `cryptsetup` das Passwort zweimal abfragen (was in Verbindung mit Key-Files sinnlos wäre) und `-s` bestimmt die Länge des Schlüssels. `luksFormat` legt fest, dass der Header nach dem LUKS-Standard erstellt wird.

Nach einer Bestätigung und einem zweimal eingegebenen Passwort kann man die Partition entschlüsseln und bereit stellen:

```
cryptsetup luksOpen /dev/sda2 sehrsicher
```

Die Partition steht jetzt unter dem virtuellen Gerät `/dev/mapper/sehrsicher` zur Verfügung, und kann mit

```
[root]# mkfs.ext3 /dev/mapper/sehrsicher  
mit einem Dateisystem beschrieben werden.
```

Nun kann man sie ganz normal einhängen:

```
[root]# mount /dev/mapper/sehrsicher /mnt  
Das Aushängen und Beenden der Verschlüsselung funktioniert dann in folgenden Schritten:
```

```
[root]# umount /mnt  
und
```

```
[root]# cryptsetup luksClose sehrsicher
```

Komplettes Listing

```
[root@laptop ~]# cryptsetup -c aes-cbc-essiv:sha256 -y -s 256 luksFormat /dev/sda2
```

```
WARNING!
```

```
=====
```

```
This will overwrite data on /dev/sda2 irrevocably.
```

```
Are you sure? (Type uppercase yes): YES
```

```
Enter LUKS passphrase:
```

```
Verify passphrase:
```

```
[root@laptop ~]# cryptsetup luksOpen /dev/sda2 sehrsicher
```

```
Enter LUKS passphrase:
```

```
key slot 0 unlocked.
```

```
[root@laptop ~]# mkfs.ext3 /dev/mapper/sehrsicher
```

```
mke2fs 1.37 (21-Mar-2005)
```

```
Dateisystem-Label=
```

```
OS-Typ: Linux
```

```
Blockgröße=4096 (log=2)
```

```
Fragmentgröße=4096 (log=2)
```

```
27918336 Inodes, 55805536 Blöcke
```

```
2790276 Blöcke (5.00%) reserviert für den Superuser
```

Verschlüsselte_Festplatten

```
erster Datenblock=0
Maximum filesystem blocks=58720256
1704 Blockgruppen
32768 Blöcke pro Gruppe, 32768 Fragmente pro Gruppe
16384 Inodes pro Gruppe
Superblock-Sicherungskopien gespeichert in den Blöcken:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
    4096000, 7962624, 11239424, 20480000, 23887872
```

```
Schreibe Inode-Tabellen: erledigt
Erstelle Journal (8192 Blöcke): erledigt
Schreibe Superblöcke und Dateisystem-Accountinginformationen: erledigt
```

Das Dateisystem wird automatisch alle 31 Mounts bzw. alle 180 Tage überprüft, je nachdem, was zuerst eintritt. Veränderbar mit `tune2fs -c` oder `-t`.

```
[root@laptop ~]# mount /dev/mapper/sehrsicher /mnt
```

```
[root@laptop ~]# ls /mnt
lost+found
```

```
[root@laptop ~]# umount /mnt
```

```
[root@laptop ~]# cryptsetup luksClose sehrsicher
```

Zusätze

Weitere Schlüssel

Will man, wie oben schon erwähnt, einen zweiten Schlüssel hinzufügen, muss man folgende Aktionen auf das Gerät anwenden:

```
cryptsetup luksAddKey /dev/sda2
```

Dabei muss zuerst ein schon existierender Schlüssel doppelt eingegeben werden, bevor danach der neue Schlüssel angelegt und verifiziert werden kann.

Ebenso kann man Schlüssel löschen

```
cryptsetup luksDelKey /dev/sda2 <Slot>
```

Für `<Slot>` muss die Nummer des Keyslots eingegeben werden. Diesen findet man am einfachsten raus, wenn mit `cryptsetup luksOpen` einen Container oder eine Partition einhängt. Dort ist dann auch der Slot angegeben, der, basierend auf dem Schlüssel, benutzt wurde.

Informationen

Möchte man Informationen über eine verschlüsselte Partition, kann folgender Befehl verwendet werden.

```
cryptsetup luksDump /dev/sda2
```

Systemlast senken

Die Ver- und Entschlüsselung ist sehr CPU-lastig. Damit der Rechner nicht ins Stocken kommt, hilft es `kcryptd` zu renicen. Damit das nicht immer von Hand geschehen muss, kann man in die [/etc/rc.local](#) eintragen:

```
renice 0 `pgrep kcryptd`
```

So wird gleich nach dem Start `kcrypt` mit 0 statt mit -5 ausgeführt.

Beim Systemstart mounten

Falls nötig die Datei [/etc/crypttab](#) anlegen und das Folgende eintragen:

```
sehrsicher /dev/sda2 none luks, retry=3, cipher=aes-cbc-essiv:sha256
```

Zusätzlich muss noch ein Verzeichnis anlegen, in das gemountet werden soll:

```
mkdir /home/<user>/sehrsicher
```

Zu [/etc/fstab](#) hinzufügen:

```
/dev/mapper/sehrsicher      /home/<user>/sehrsicher    ext3      defaults      0 2
```

Nach diesen Maßnahmen wird nun die verschlüsselte Partition automatisch im Home-Verzeichnis beim Systemstart eingehängt. Beim Boot-Vorgang muss daher die Passphrase eingegeben werden.

Einhängen von externen Datenträgern

Wenn sich die verschlüsselte Partition auf einem externen Datenträger befindet, erledigt Fedora das Einbinden automatisch. Nach der Passworteingabe ist der Datenträger automatisch eingebunden.



Siehe auch

- ◇ Verschlüsselte Swap- und Temp-Partition
- ◇ Verschlüsseltes Homeverzeichnis
- ◇ Verschlüsselungsoptionen

Links

- ◇ LUKS Homepage
- ◇ dm-crypt in der Wikipedia
- ◇ Clemens Fruhwirth, Markus Schuster: *Geheime Niederschrift. Festplattenverschlüsselung mit DM-Crypt und Cryptsetup-LUKS: Technik und Anwendung* In: *Linux-Magazin* 08/2005. Linux New Media AG, S. 28-36, ISSN 1432-640X