LuksCrypt Grundlagen

Festplatte und USB-Stick verschlüsseln unter GNU/Linux

2015 eTc

Motivation

- Daten auf mobilen Geräten (Laptop) vor fremden Zugriff sichern
- Probleme mit einer „verschlüsselten“ Festplatte lösen
- LUKS verstehen – Was passiert hinter den Kulissen?
- Alternative zu ~~Truecrypt~~ Veracrypt

Vorraussetzungen

- GNU / Linux basiertes Betriebssystem
- Leerer Datenträger (USB-Stick, SD-Card, Hdisk, Datei, …)
- Cryptsetup Paket
  - Debian / Ubuntu: **`apt-get install cryptsetup`**
  - RHEL / CentOS / Fedora: **`yum install cryptsetup-luks`**
  - ArchLinux / Manjaro: **`pacman -S cryptsetup`**

Tasks

- Erzeugen des verschlüsselten Mediums
- Ein- und Aushängen des Mediums
- Ändern der Passphrase(s)
- Status prüfen
- Zerstören des Mediums

Task: Erzeugen des verschlüsselten Mediums

Step 1: LUKS Partition erzeugen

```
$ lsblk
NAME                        MAJ:MIN RM   SIZE RO TYPE  MOUNTPOINT
...
sdc                           8:32   1   3,7G  0 disk
└─sdc1                        8:33   1   3,7G  0 part
$ sudo cryptsetup -v luksFormat /dev/sdc1
WARNING!
========
Hiermit überschreiben Sie Daten auf »/dev/sdc1« unwiderruflich.
```

```
Are you sure? (Type uppercase yes): YES
Passphrase eingeben:
Passphrase wiederholen:
Befehl erfolgreich.
```

Task: Erzeugen des verschlüsselten Mediums

Step 2: LUKS Partition aktivieren

```
$ sudo cryptsetup luksOpen /dev/sdc1 cruzer-crypted
Geben Sie die Passphrase für »/dev/sdc1« ein:
```

```
$ lsblk
NAME                    MAJ:MIN RM   SIZE RO TYPE  MOUNTPOINT
...
sdc                       8:32   1   3,7G  0 disk
└─sdc1                    8:33   1   3,7G  0 part
  └─cruzer-crypted      254:5    0   3,7G  0 crypt
```

Task: Erzeugen des verschlüsselten Mediums

Step 3: Blöcke 'nullen'

VORSICHT: auf den korrekten Pfad hinter `of=` achten!

```
$ sudo dd if=/dev/urandom of=/dev/mapper/cruzer-crypted bs=128M
```

```
30+0 Datensätze ein
30+0 Datensätze aus
4001366016 Bytes (4,0 GB) kopiert, 631,543 s, 6,3 MB/
```

- **Das dauert …**
- **… erzeugt Rauschen in allen Blöcken**

- **Warum (nicht) mit `wipefs` oder `shred`? Geht auch!**

Task: Erzeugen des verschlüsselten Mediums

Step 4: Dateisystem erzeugen und einhängen

```
$ sudo mkfs.ext4 /dev/mapper/cruzer-crypted
mke2fs 1.42.5 (29-Jul-2012)
Dateisystem-Label=
OS-Typ: Linux
…
node-Tabellen werden geschrieben: erledigt
Erstelle Journal (16384 Blöcke): erledigt
Schreibe Superblöcke und Dateisystem-Accountinginformationen:
erledigt
$ sudo mount /dev/mapper/cruzer-crypted /mnt
```

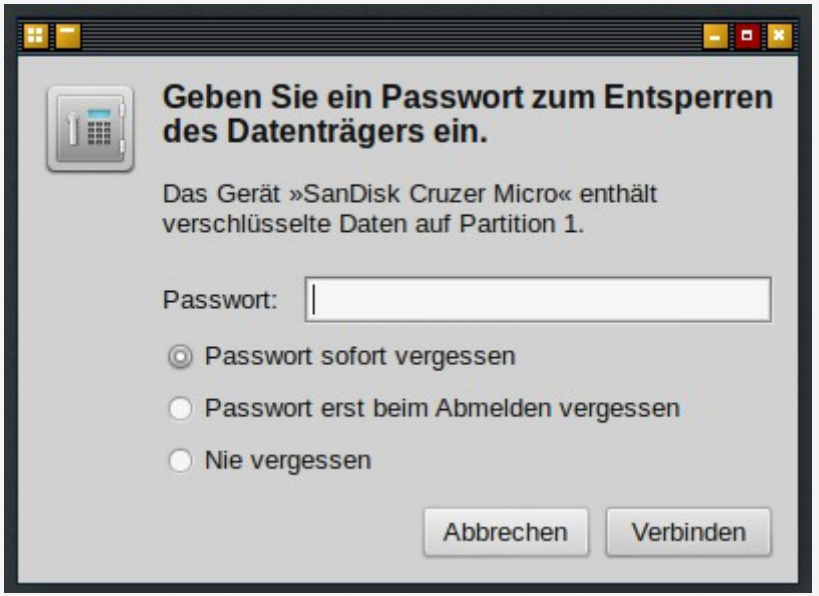
Für die nächsten Schritte wieder aushängen:

```
$ sudo umount /mnt; sudo cryptsetup close cruzer-crypted
```

Task: Ein- und Aushängen des Mediums

Einhängen:

- Wie beim Task davor in Step2 (luksOpen) und Step4 (mount)
- Automatisch durch den automounter beim Einstecken des Sticks



Aushängen:

- **`sudo umount`** `<Device>` dann
  `cryptsetup close` `<Name>`
- Dateimanager (Rechtsklick): Medium auswerfen/aushängen/umount

Task: Status prüfen

- **Ist /dev/sdc1 ein luks-Device?**

```
$ sudo cryptsetup -v isLuks /dev/sdc1
Befehl erfolgreich.
```

- **Allgemeiner Status:**

```
$ sudo cryptsetup -v status cruzer-crypted
/dev/mapper/cruzer-crypted is active and is in use.
  type:    LUKS1
  cipher:  aes-xts-plain64
  keysize: 256 bits
  device:  /dev/sdc1
  offset:  4096 sectors
  size:    7815168 sectors
  mode:    read/write
Befehl erfolgreich.
```

Task: Status prüfen

```
$ sudo cryptsetup luksDump /dev/sdc1
LUKS header information for /dev/sdc1

Version:        1
Cipher name:    aes
Cipher mode:    xts-plain64
Hash spec:      sha1
Payload offset: 4096
MK bits:        256
MK digest:      3a 0b 69 81 34 47 04 fb 2c 59 62 71 8a 5c c6 44 4c 76 26 5e
MK salt:        c2 bd 1a 79 bf 81 73 ae 88 38 41 f4 c1 e4 66 f5
                96 ff 8f c4 93 86 8f a8 a3 2a a4 6d 51 93 62 61
MK iterations:  100625
UUID:           2cfdbeae-9214-41af-a954-f00762d1d68e

Key Slot 0: ENABLED
Iterations:             412902
Salt:                   aa b8 ca bb 16 c3 24 04 2d fe 71 c6 1e c2 5e 41
                        82 f2 39 50 a0 e7 4c 65 8d 92 e1 83 85 6e 84 85
Key material offset:    8
AF stripes:             4000
...
Key Slot 7: DISABLED
```

Task: Ändern der Passphrase(s)

`luksAddKey` `/dev/sdc1`

- Fügt Passphrase zu LUKS-Gerät hinzu

`luksRemoveKey` `/dev/sdc1`

- Entfernt bereitgestellte Passphrase vom LUKS-Gerät

`luksChangeKey` `/dev/sdc1`

- Ändert die angegebene Passphrase des LUKS-Geräts

`luksKillSlot` `/dev/sdc1 <KeySlot>`

- Löscht Passphrase im <KeySlot> vom LUKS-Gerät

- Sicheres Ändern (statt ChangeKey): AddKey dann RemoveKey
- Zu entfernende Passphrases müssen bekannt sein!

Task: Zerstören des Mediums

erase /dev/sdc1

- Alle Schlüsselfächer löschen (Verschlüsselungsschlüssel entfernen)

luksFormat /dev/sdc1 [<neue Schlüsseldatei>]

- Formatiert ein LUKS-Gerät, löscht aber keine Daten
  - Nicht geeignet!

dd if=/dev/zero of=/dev/sdc1

- Sowas geht immer :)

Auf korrekte Pfade achten!

Datenträger

- USB-Stick, Festplatten, SD-Cards
  - → wie in den Tasks beschrieben
- Datei (Container File)
- Festpatten mit Betriebssystem

Datenträger: Datei (Container File)

```
$ dd if=/dev/urandom of=luksTest bs=1M count=128
$ cryptsetup luksFormat luksTest
$ sudo cryptsetup luksOpen luksTest luksDev
$ lsblk
loop0                   7:0    0   128M  0 loop
└─luksDev             254:5    0   126M  0 crypt
$ sudo mkfs.ext4 /dev/mapper/luksDev
$ sudo mount /dev/mapper/luksDev mnt
loop0                                          7:0    0   128M  0 loop
└─luksDev                                    254:5    0   126M  0
crypt /home/eric/tmp/luks/mnt
$ df /home/eric/tmp/luks/mnt
Dateisystem         Größe Benutzt Verf. Verw% Eingehängt auf
/dev/mapper/luksDev  119M    1,6M  108M    2% /home/eric/tmp/luks/mnt
```

Auf korrekte Pfade achten!

Datenträger: Festpatten mit Betriebssystem

- Beispiel Layout

```
$ lsblk
NAME                                MAJ:MIN RM   SIZE RO TYPE  MOUNTPOINT
sda                                   8:0    0 238,5G  0 disk
├─sda1                                8:1    0   255M  0 part  /boot
└─sda2                                8:2    0 238,2G  0 part
  └─cryptManjaro                    254:0    0 238,2G  0 crypt
    ├─ManjaroVG-ManjaroRoot         254:1    0  29,3G  0 lvm   /
    ├─ManjaroVG-ManjaroHome         254:2    0 205,3G  0 lvm   /home
    └─ManjaroVG-ManjaroSwap         254:3    0   3,7G  0 lvm   [SWAP]
```

- Initramfs muss angepasst werden

```
HOOKS="... block encrypt filesystems …"
```

- Bootloader / Kernelparameter müssen angepasst werden

```
GRUB_CMDLINE_LINUX="cryptdevice=/dev/sda2:ManjaoVG
                                root=/dev/mapper/ManjaroVG-ManjaroRoot"
```

Verschlüsselungsparameter

- Standard-Verschlüsselungsparameter:
  - LUKS1: aes-xts-plain64, Schlüssel: 256 Bits, LUKS-Header-Hashen: sha1, Zufallszahlengenerator: /dev/urandom
- Vorgabewerte für Schlüssel und Passphrasen:
  - Maximale Größe der Schlüsseldatei: 8192kB
  - Maximale Länge der interaktiven Passphrase: 512 Zeichen
  - Vorgabe für die Durchlaufzeit für PBKDF2 mit LUKS: 1000 Millisekunden

Schlußbemerkung

- Geöffnete / eingehängte Datenträger sind nicht wirklich geschützt
- Auf die Passphrase(s) aufpassen (Snowden-Decke)
- „Absolute Sicherheit gibt es nicht.“

- Veracrypt (Nachfolger von Truecrypt)
- Ext4 crypt