Jürgen Schmidt

Doppeltes Spiel

Ubuntus Komplettverschlüsselung mit mehreren Festplatten

Bei Ubuntu ist es keineswegs trivial, mehrere Festplatten zu verschlüsseln. Denn die naheliegenden Lösungen haben einen Pferdefuß.

Bereits im Ubuntu-Installer kann man angeben, dass man sein System komplett verschlüsseln möchte. Das legt dann ein verschlüsseltes LVM-Volume auf der Festplatte an. Kommt eine zweite Festplatte ins Spiel, wird es unschön. Natürlich könnte man das Volume über den neuen Datenträger ausdehnen. Damit hat man aber keine Kontrolle mehr darüber, welche Daten wo landen. Und das ist insbesondere bei einer Kombination von einer SSD mit klassischer Festplatte nicht das, was man will.

Legt man hingegen mehrere verschlüsselte Volumes an, fragt Ubuntu beim Start für jedes einzeln nach dem Passwort. Es kann nämlich nicht wie Fedora die einmal eingegebene Passphrase zwischenspeichern und dann für alle Volumes verwenden. Wenn Sie ebenfalls tippfaul sind, müssen Sie also selber etwas nachhelfen.

Als Erstes richten Sie dazu das System über den normalen Installer mit Komplett-Verschlüsselung auf einem Datenträger ein. Den zweiten Datenträger können Sie dann nachträglich mit dem Werkzeug zur Laufwerks-Verwaltung komfortabel partitionieren (gnome-disks). Allerdings liefert die dort angebotene Option, eine verschlüsselte Partition anzulegen (formatieren mit „LUKS + Ext4“) kein optimales Ergebnis – jedenfalls beim aktuellen Ubuntu 15.10.

Auf den ersten Blick scheinen die Verschlüsselungsparameter durchaus angemessen. Allerdings ist zu beachten, dass der AES-Modus XTS die Schlüssel halbiert. Die von cryptsetup luksDump angezeigten 256 Bit für die Keysize bedeuten also tatsächlich nur AES mit 128-Bit-Schlüsseln. Das lässt sich zwar heute nicht knacken; für langfristige Sicherheit über einen Zeitraum von zehn Jahren hinweg, empfiehlt sich aber der Einsatz von echten 256-Bit-Schlüsseln. Das Hash-Verfahren SHA-1 ist ebenfalls nicht zeitgemäß und sollte durch SHA-256 ersetzt werden. Wem das nichts ausmacht, der kann bei „Automatisch einbinden“ weiterlesen; das Einrichten „zu Fuß“ ist aber auch kein Hexenwerk.

Das Weitere geht davon aus, dass die zweite Festplatte als /dev/sdX im System bekannt ist und eine Partition /dev/sdX1 enthält, die das verschlüsselte Dateisystem aufnehmen soll. Sie finden die für Sie richtigen Bezeichner etwa über das Laufwerks-Tool. Außerdem benötigen die Befehle Root-Rechte, die Sie etwa durch ein vorangestelltes sudo erhalten. Mit

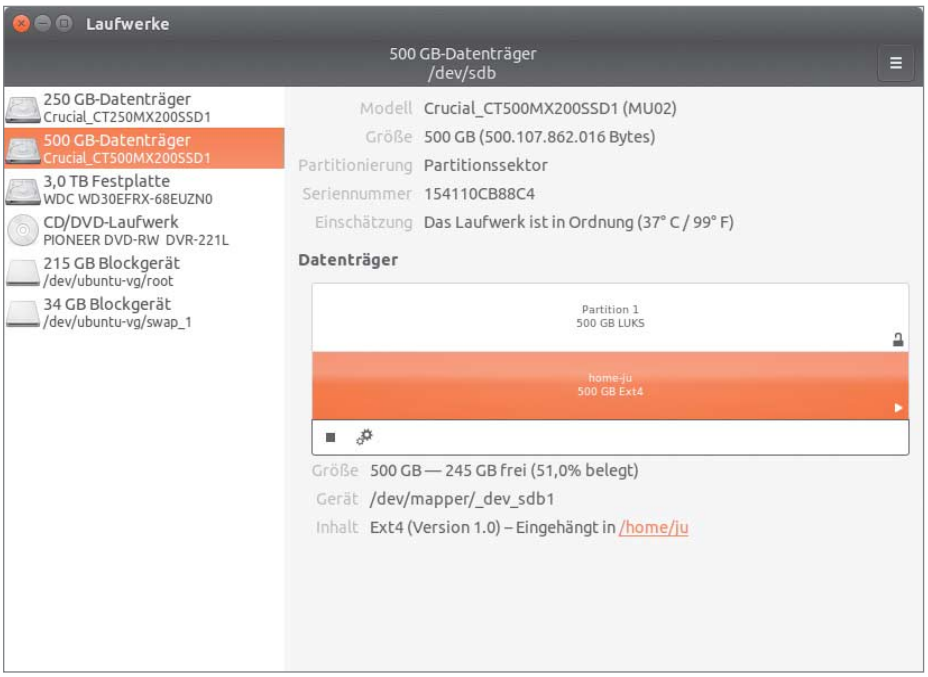
```
wipefs -a /dev/sdX1
```

löschen Sie eventuell bereits vorhandene Partitions-Header, bevor Sie via

```
cryptsetup luksFormat -c aes-xts-plain64 -s 512 ⤸
                      -h sha256 -y/dev/sdX1
```

zeitgemäße Verschlüsselung mit AES-256 und SHA-256 einrichten. Dabei müssen Sie zwei Mal die gewünschte Passphrase eingeben. Schreiben Sie die auf und verwahren Sie den Zettel sorgfältig. Ohne die Passphrase ist kein Zugang zu den Daten mehr möglich.

Messbare Auswirkungen auf die Performance hatten diese Sicherheitsverbesserungen in unseren Kurztests übrigens nicht. Nach dem luksFormat taucht die Partition wieder in Disk-Utility auf; Sie können sie dort entsperren und dann mit Ext4 formatieren. Achten Sie darauf, dass Sie wirklich das entschlüsselte Block-Device /dev/mapper/luks-<UUID> formatieren und nicht etwa /dev/sdX1.



Die Laufwerksverwaltung kann auch verschlüsselte Partitionen erzeugen und einbinden.

Automatisch einbinden

Behalten Sie im Hinterkopf, dass nach dem Mounten etwa nach /home die ursprünglichen Home-Verzeichnisse nicht mehr sichtbar sind. Binden Sie also die neue LUKS-Partition vor dem Neustart einmal von Hand über den Datei-Manager ein und kopieren Sie die Home-Verzeichnisse dorthin. So stehen sie nach dem nächsten Neustart gleich wieder zur Verfügung.

Später soll das System die verschlüsselten Partitionen automatisch einbinden. Dazu fragt es beim Booten nach dem Passwort – bei zwei LUKS-Partitionen zwei Mal. Der gängige Workaround ist es, in der bereits verschlüsselten Partition ein Keyfile abzulegen, mit dem Linux dann die zweite Festplatte entsperren kann.

```
dd if=/dev/urandom bs=4096 count=1 of=/root/keyfile
chmod 0400 /root/keyfile
cryptsetup luksAdd /dev/sdX1 /root/keyfile
```

Anschließend tragen Sie die neue LUKS-Partition in /etc/crypttab und /etc/fstab wie im Kasten gezeigt ein. Die dazu benötigte UUID finden Sie im Verzeichnis /dev/disk/by-uuid; einen Überblick über Festplatten, Partitionen und was

Konfigurationsdateien

Diese Einträge benötigen Sie nur, wenn das System die Festplatten alle beim Start automatisch einbinden soll.

/etc/crypttab

```
sda3_crypt UUID=1d4635c6-fa49-... none luks,discard
sdX1_crypt UUID=3bc923e0-... /root/keyfile luks,discard
```

/etc/fstab

```
/dev/mapper/sdX1_crypt /opt ext4 defaults 0 2
```

gerade wohin gemountet ist, liefert der Befehl lsblk. Abschließend erzeugt

```
update-initramfs -u -k all
```

ein neues Initramfs für den Systemstart. Nach dem nächsten Booten sollte das System die zweite Festplatte automatisch benutzen.

Mount beim Login

Dieses Vorgehen hat jedoch Nachteile. Insbesondere liegt damit im laufenden Betrieb mit dem Keyfile ein wichtiges Passwort im Klartext herum. Das ist in etwa so, als würde Linux Benutzer-Passwörter im Klartext speichern. Deshalb raten wir von der Keyfile-Variante ab. Das an vielen Stellen noch beschriebene Verfahren zur Schlüsselableitung funktioniert mit systemd und damit ab Ubuntu 15.04 nicht mehr.

Eleganter ist es, den zweiten Datenträger beim Anmelden einbinden zu lassen. Die Plugable Authentication Modules (PAM) bilden eine Infrastruktur, um den Login-Vorgang flexibel zu erweitern. Die Erweiterung pam_mount nutzt das Anmelde-Passwort, um beim Login automatisch weitere Partitionen einzubinden und erkennt und öffnet LUKS-Partitionen automatisch richtig. Voraussetzung ist, dass man für das LUKS auf dem zweiten Datenträger und den Account das gleiche, möglichst gute Passwort verwendet. Die Passphrase der Root-Disk kann eine andere sein – muss aber nicht.

Das ist keineswegs ein kruder Hack, sondern eine saubere Lösung, die beispielsweise auch beim Anmelden via SSH funktioniert. Allerdings fragt der SSH-Dienst unter Umständen hartnäckig nach dem zum Entschlüsseln der Platte benötigten Passwort; ein Login mit einem Public Key funktioniert erst, wenn das Laufwerk bereits entsperrt wurde. Das Aushängen nach dem Abmelden funktioniert beim aktuellen Ubuntu ebenfalls wie erwartet.

Sie müssen das PAM-Paket zunächst mit `sudo apt-get install libpam-mount` nachinstallieren. Anschließend können Sie in der Datei /etc/security/pam_mount.conf.xml die zu mountenden Volumes eintragen. Dazu fügen Sie am Ende, direkt vor dem abschließenden </pam_mount> folgende Zeile ein:

```
<volume user="ju" path="/dev/sdX1" ↵
                    mountpoint="/home/ju" />
```

Benutzernamen und Pfad zum Device müssen Sie an Ihre Gegebenheiten anpassen; Sie können natürlich auch den Pfad via disk/by-uuid verwenden. Wer eine SSD auf diesem Weg einbindet, sollte das noch nicht dokumentierte, aber funktionierende options="allow_discard" einfügen, das den seit Ubuntu 14.04 eingeführten, wöchentlich durchgeführten Trim gestattet.

Damit bleiben alle anderen Home-Verzeichnisse auf der System-Partition. Als Mount-Point käme auch das übergeordnete Verzeichnis /home in Frage. Das bedeutet dann aber, dass Sie allen Benutzern, deren Home-Verzeichnis dort liegt, Zugriff auf die verschlüsselte Platte geben müssen. Geben Sie als user dann %(USER) an. Damit die anderen Nutzer ihr eigenes Passwort zum Entsperren der LUKS-Partition nutzen können, fügen sie es mit

```
sudo cryptsetup luksAddKey /dev/sdX1
```

hinzu. Sie müssen dazu ein für die Entschlüsselung bereits freigeschaltetes Passwort angeben. Ein erneutes cryptsetup mit der Option luksDump sollte dann die Key Slots 0 und 1 als belegt anzeigen. LUKS kann insgesamt bis zu acht Passwörter für eine Partition verwalten. Beachten Sie, dass jeder Login damit alle Home-Verzeichnisse entsperrt; ein Test-Account mit einem primitiven Passwort gefährdet somit auch Ihre wichtigen Daten im eigenen Home-Verzeichnis.

Vergessen Sie auch nicht, mit

```
sudo cryptsetup luksHeaderBackup /dev/sdX1 ↵
                    --header-backup-file BACKUP-DATEI
```

ein Backup der LUKS-Header anzulegen und anschließend auf einem externen Medium zu sichern. Das kann bei zwei Platten jeweils wechselseitig erfolgen. Diese Header-Backups erlauben keinen Zugang ohne gültiges Passwort; aber nahezu alle Rettungsoperationen an einer beschädigten Platte benötigen sie. Denn anders als etwa bei einer beschädigten Partitionstabelle finden Datenrettungs-Tools ohne LUKS-Header nur noch verschlüsselten Datenmüll. (ju@ct.de) c't

Schnelles Überschreiben

Bevor man eine Festplatte komplett verschlüsselt oder verkauft, wird meist empfohlen, sie mit Zufallszahlen zu überschreiben. Das geschieht etwa mit

```
dd if=/dev/urandom of=/dev/sdX.
```

Doch bei einer aktuellen 3-TByte-Platte würde das schon fast eine Woche dauern.

Der Flaschenhals ist dabei nicht etwa das Schreiben selber, sondern das Zufallszahlen-Device von Linux /dev/urandom. Das liefert selbst auf schnellen Systemen die Zufallszahlen nur mit etwa 10 bis 20 MByte/s an.

Für das Überschreiben der Festplatte benötigen Sie keine wirklich hochwertigen Zufallszahlen, wie man sie etwa für geheime Schlüssel haben will. Die Daten enthalten ja selbst kein Geheimnis, sondern dienen nur als Tarn-Hintergrund für die echten verschlüsselten Daten. Folglich genügt es vollkommen, Nullen mit einem zufälligen Passwort zu verschlüsseln:

```
openssl enc -aes-256-ctr -pass pass:"$(dd ↵
    if=/dev/urandom bs=128 count=1 2>/dev/null | ↵
                base64)" -nosalt < /dev/zero >/dev/sdX
```

Dabei liefert /dev/urandom nur noch das zufällige Passwort; openssl erzeugt den Zufallszahlenstrom durch AES-Verschlüsselung. Das macht es so schnell, dass es schon auf einem älteren Testsystem rund 270 MByte/s anlieferte, auf einem modernen sogar fast 2 GByte/s – also definitiv mehr als die Schreibgeschwindigkeit einer Festplatte.

Damit hängt die benötigte Zeit nur noch von der Größe der Festplatte und der realen Schreibgeschwindigkeit ab. Bei einer 3-TByte-Platte mit circa 150 MByte/s sind das nur noch etwa fünf Stunden. Das ist immer noch so lang, dass man es wohl kaum vorbeugend, sondern nur zum gezielten Löschen unverschlüsselter Daten macht.

SSDs sind ein Spezialfall. Ihnen melden Betriebssysteme wie Ubuntu standardmäßig via Trim, welche Bereiche derzeit ungenutzt sind. Deren Inhalt verwirft die SSD; das vorherige zufällige Überschreiben nützt also nichts. Trim deswegen abzuschalten ist nicht empfehlenswert, da das Performance und Lebensdauer der SSD reduzieren kann.

```
ju@ju-PC:~$ lsblk
NAME                    MAJ:MIN RM   SIZE RO TYPE  MOUNTPOINT
sda                       8:0    0 232,9G  0 disk
├─sda1                    8:1    0   512M  0 part  /boot/efi
├─sda2                    8:2    0   244M  0 part  /boot
└─sda3                    8:3    0 232,2G  0 part
  └─sda3_crypt          252:0    0 232,1G  0 crypt
    ├─ubuntu--vg-root   252:1    0 200,3G  0 lvm   /
    └─ubuntu--vg-swap_1 252:2    0  31,9G  0 lvm   [SWAP]
sdb                       8:16   0 465,8G  0 disk
└─sdb1                    8:17   0 465,8G  0 part
  └─_dev_sdb1           252:4    0 465,8G  0 crypt /home/ju
sdc                       8:32   0   2,7T  0 disk
└─sdc1                    8:33   0   2,7T  0 part
  └─sdc1_crypt          252:3    0   2,7T  0 crypt /opt
sr0                      11:0    1  1024M  0 rom
ju@ju-PC:~$
```

In diesem System sind gleich drei Festplatten verschlüsselt eingebunden – die Systemplatte /dev/sda, das Datengrab /opt und das via PAM-mount eingebundene Home-Verzeichnis.