Lokales Storage Teil 2

Linux-Kurs der Unix-AG

Zinching Dang

9. September 2014

CC BY SA

UNIX AG
TU Kaiserslautern

RH Regionales Hochschul-
Rechenzentrum Kaiserslautern RK

LVM (1)

- **L**ogical **V**olume **M**anager
- erfüllt gleichen Zweck wie Partitionierung
- erlaubt jedoch das Partitionieren über mehrere Datenträger
- flexible Verwaltung der Datenträger
- Aufbau über verschiedene Ebenen:
  - Physical Volume
  - Volume Group
  - Logical Volume
- dynamisches Ändern im Betrieb möglich

LVM (2)

- **Physical Volume** (**PV**): physikalischer Datenträger oder Partition
  - z. B. `/dev/sda`, `/dev/sdb1`, `/dev/md0`
- **Volume Group** (**VG**): Datenträger-Pool
  - PVs können einer VG zugeordnet werden
- **Logical Volume** (**LV**): logischer Datenträger
  - wird in einer VG angelegt
  - entspricht einer Partition auf einem Datenträger

Physical Volume

- eigentlicher Datenspeicher (Datenträger oder Partition)
- unterste Abstraktionsebene

- `pvcreate`: neues PV aus einem Datenträger anlegen
- `pvs`: PVs anzeigen
- `pvdisplay`: Informationen über PVs anzeigen

- Aufruf: `pvcreate /dev/mdX`
  - Datenträger `/dev/mdX` als PV anlegen
- Aufruf: `pvs`
- Aufruf: `pvdisplay`

Volume Group

- Container, in dem die PVs verwaltet werden
- neue PVs können dynamisch hinzugefügt werden

- `vgcreate`: neue VG aus einem oder mehreren PV(s) erstellen
- `vgs`: VGs anzeigen
- `vgdisplay`: Informationen über VGs anzeigen

- Aufruf: `vgcreate <VG-Name> /dev/mdX`
  - VG „`VG-Name`“ aus Datenträger `/dev/mdX`, der bereits ein PV ist, erstellen
- Aufruf: `vgs`
- Aufruf: `vgdisplay`

Logical Volume

- entspricht Partitionen auf Datenträgern
- können dynamisch vergrößert/verkleinert werden
- normale Partitionen können nicht auf einfache Weise vergrößert/verkleinert werden

- `lvcreate`: neues LV in einer vorhanden VG erstellen
- `lvs`: LVs anzeigen
- `lvdisplay`: Informationen über LVs anzeigen

- Aufruf: `lvcreate -L 1G -n <Name> <VG-Name>`
  - LV mit der Bezeichnung `<Name>` der Größe 1 GiB in der VG „`VG-Name`“ erstellen
- Aufruf: `lvs`
- Aufruf: `lvdisplay`

Dateisysteme

- Verwaltung von Dateien auf Datenträgern
- Baumstruktur aus Verzeichnissen und Dateien

- verschiedene Dateisysteme:
  - ext2, ext3, ext4, (btrfs, zfs)
  - NTFS, FAT

- werden mit `mkfs` angelegt, bzw. mit
  - `mkfs.ext[234]`
  - `mkfs.ntfs`, `mkfs.vfat`

fstab

- enthält eine Liste von Dateisystemen, die beim Booten automatisch gemountet (eingebunden) werden
- wird von dem Befehl `mount` eingelesen
- in sechs Spalten aufgebaut:
  - Datenträger (z. B. `/dev/sda2` oder UUID)
  - Mount-Point (z. B. `/media/hdd1`)
  - Dateisystem (z. B. ext4)
  - Mount-Optionen (z. B. `defaults`)
  - Backup-Häufigkeit mit `dump`, normalerweise 0
  - Reihenfolge beim `fsck`, 0 für kein Überprüfen

Lab: LVM einrichten

- RAID 1 Device (`md0`) als PV einrichten
- VG mit diesem PV anlegen
- LV innerhalb der VG erstellen
- Dateisystem auf dem LV erstellen
- Dateisystem mounten und in die `fstab` eintragen

Dateisysteme vergrößern mit LVM

- sicherstellen, dass die VG noch freien Speicherplatz hat (`vgs` oder `vgdisplay`)
- ggf. PVs zur VG hinzufügen (`vgextend <VG-Name> <PV>`)
- LV vergrößern (`lvresize`), danach das Dateisystem (`resize2fs`)
- manche Operationen können während dem Betrieb durchgeführt werden (online resizing)
  - LVs können beliebig vergrößert/verkleinert werden
  - beim Verkleinern ist darauf zu achten, dass das LV nicht kleiner als das Dateisystem wird
  - gemountete Dateisysteme können nur vergrößert werden
  - Verkleinern von Dateisystemen ist nur offline möglich

lvresize

- `lvresize -l [+|-]SIZE[\%{VG|LV|FREE}] <LV-Name>`
  - Größe auf `SIZE` festlegen (kein + oder - vor `SIZE`)
  - um `SIZE` vergrößern (+) oder verkleinern (-)
  - Größe auf/um `SIZE` % der VG-Größe (`VG`), LV-Größe (`LV`) oder des freien Speicherbereichs (`FREE`) setzen/ändern
- `lvresize -L [+|-]SIZE[MGTPE] <LV-Name>`
  - Größe auf `SIZE` [MGTPE]B festlegen (kein + oder - vor `SIZE`)
  - um `SIZE` [MGTPE]B vergrößern (+) oder verkleinern (-)
  - Größe auf/um `SIZE` [MGTPE]B setzen/ändern

resize2fs

- `resize2fs <Dateisystem> [SIZE]`
  - vergrößert das Dateisystem auf die maximale Größe, wenn `SIZE` nicht angegeben wird
  - andernfalls wird das Dateisystem auf `SIZE` vergrößert/verkleinert
  - Vergrößern von gemounteten Dateisystemen online möglich
  - Verkleinern von gemounteten Dateisystemen nicht möglich
  - `SIZE` darf nicht größer als die darunterliegende Partition sein

Lab: Dateisysteme vergrößern mit LVM

- LV vergrößern
- Dateisystem online vergrößern

LUKS

- **L**inux **U**nified **K**ey **S**etup
- standardisiertes Verschlüsselungs-Format unter Linux
- Erweiterung von `dm-crypt`
- erlaubt mehrere Passphrasen für eine verschlüsselte Partition
- wird mit dem Paket `cryptsetup` bereitgestellt

cryptsetup

- zentrales Tool, um mit `dm-crypt` und LUKS zu arbeiten
- hier: nur LUKS-Erweiterungen

- Aufruf: `cryptsetup <Action>`
- wichtige `Action`:
  - `luksFormat <Device>`: erstellt einen neuen LUKS-Container auf dem angegebenen Device
  - `luksOpen <Device> <Name>`: öffnet das LUKS-Device und stellt es unter `<Name>` bereit
  - `luksClose <Name>`: schließt ein geöffnetes LUKS-Device

crypttab

- wie `fstab`, enthält Liste mit LUKS-Devices
- während dem Booten werden Devices geöffnet
- Passphrase muss beim Bootvorgang eingegeben werden

- in vier Spalten aufgebaut:
  - Name, unter dem das LUKS-Device bereitgestellt werden soll
  - Pfad des Devices
  - Pfad einer Schlüsseldatei, „none“ um nach Passphrase beim Booten zu fragen
  - `dm-crypt`/LUKS-Optionen, für LUKS nur „luks“

Lab: LUKS

- neues LV anlegen
- LUKS-Container erstellen
- Dateisystem darauf anlegen
- `crypttab` einrichten und testen