Warum sollten wir Btrfs verwenden?

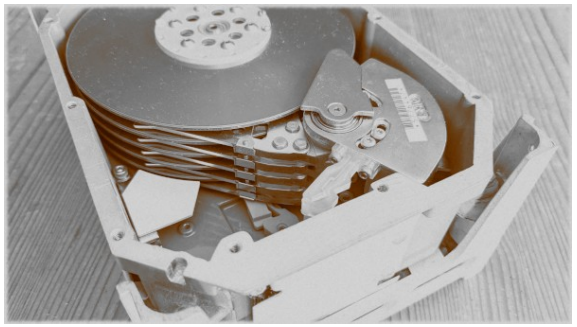
CLT 2012
- Linux/Ubuntu im sicheren und virtuellen Netz
- Warum GNU/Linux/Ubuntu?

CLT 2016
- rsync, SSH, LUKS im Team
- Backup mit Linux Bordmitteln

.
.
.

CLT heute
- der Unterbau von allem
- Btrfs

Physiker in Halle
Fernstudium Theologie
seit 1988 in München
Physiker am Max-Planck-Institut
IT Berater und Entwickler
Linuxtage in
Augsburg, Graz, Dornbirn,
Wien, Kiel, Chemnitz, ...



Btrfs funktioniert nicht wie die gewohnten Dateisysteme.
Um BTRFS gut zu nutzen, muss man wissen, was man tut.
Die Dokumentation zu lesen ist der beste Weg das zu erreichen:
https://btrfs.readthedocs.io/en/latest/

Themen

- Btrfs ist Unterbau für Desktops und Server
- Links im Filesystem
- Copy-on-Write
- Subvolumes und Snapshots
- RAID und LUKS
- Tipps für neue Partitionen

einige Links

https://wiki.archlinux.org/title/Btrfs
https://help.ubuntu.com/community/btrfs
https://www.usenix.org/system/files/login/articles/bacik_0.pdf

- „The Swiss Army Knife of Storage“

Demos:

- Ext4 nach Btrfs umwandeln
- Datenintegrität online testen
- Softlink, Hardlink, Reflink
- Arbeiten mit Snapshots
- RAID mit LUKS

Vielfalt

Wie beginnen?

- `apt install btrfs-progs`
- formatieren und genau wie ein Ext4 verwenden
- Btrfs kann aber viel mehr

Eigenschaften

- Filesystem ist immer konsistent
- Checksummen pro File, Erkennen von Bitfehlern
- Copy-on-Write
- Subvolumes mit Snapshots
  - neue Arbeitsweise mit Mountpoints über Subvolumes
- Backup von Datenstrukturen, die gerade verwendet werden
- optimiert für SSD
- optimiert für SMR Disks
- RAID über Prüfsummen

Nachteile?

- RAID 5/6 nicht stabil

Online Neuformatierung eines Ext4 Filesystems mit Btrfs

```
# convert from ext3/4 → Btrfs
btrfs-convert /dev/sdc1
# oder
btrfs-convert /dev/disk/by-partlabel/mydisk

# mount the resulting Btrfs filesystem
mount -t btrfs /dev/disk/by-partlabel/mydisk /mnt/mydisk

# mount the ext3/4 snapshot
mount -t btrfs -o subvol=ext2_saved /dev/disk/by-partlabel/e1loop /mnt/ext4_saved

# loopback mount the image file
mount -t ext4 -o loop,ro /mnt/ext4_saved/image /mnt/ext4
diff -r ext4/ mydisk/
```

https://btrfs.readthedocs.io/en/latest/Convert.html

Warnung:

- Umwandlung ist instabil bei Stromausfall

stabiler:

- Backup, Format mit Btrfs, Restore

Scrub – Kontrolle aller Prüfsummen

Scrub

- ca. alle 100 TB ein Bitfehler
- `btrfs scrub start /mnt/mydisk`
- `btrfs scrub status /mnt/mydisk`

Ergebnis wird gespeichert

- in `/var/lib/btrfs/scrub.status.UUID`

  `cat /var/lib/btrfs/scrub.status.d5de08ef-45a5-4e5b-af6d-72b70528158b`

lange Laufzeit, wöchentlich empfohlen

Script

http://marc.merlins.org/perso/btrfs/post_2014-03-19_Btrfs-Tips_-Btrfs-Scrub-and-Btrfs-Filesystem-Repair.html

Links in Linux

Softlink = Link zu einem File oder zu einem Verzeichnis

- `files -> /mnt/b1/btrfstestfiles/files`
- über Partitionen hinweg
- kann auf nichts zeigen, dead link

Hardlink = nur für Files, weiterer Inode, der auf die gleichen Daten zeigt

```
ln hardlinkedfile.txt hardlinkedfile.txt1
```

- Referenzzähler zur Verwaltung
- mit *ls* anzeigbar

```
ls -ali,   8732422 -rw-r----- 2 root root 48 Sep 20 20:36 hardlinkedfile.txt
                                ^ hardlinkcount
```

- nur im gleichen Filesystem
- Ändern der Daten ändert alle Files im Hardlink

Reflink = Copy-on-Write = CoW

```
cp --reflink=always reflinkmaster.txt reflinkcopy.txt
```

- wird intern verwaltet, mit *ls* unsichtbar
- aus Sicht des Benutzers sind das 2 verschiedene Files
- „shallow copy“
- erst bei Änderung der Daten werden diese kopiert
- atomarer Vorgang
- nur im gleichen Filesystem

CoW = Copy-on-Write

Vorteile

- schnell, platzsparend, **transaktionssicher**

```
cp --reflink=always quelle ziel
```

Nachteile

- belegt Platz, wenn Daten geändert werden
- nicht mit herkömmlichen Tools darstellbar

  `filefrag -v` kann verwendet werden
- man sollte wissen, was man auf der eigenen HD alles anlegt

Abschalten von Copy-on-Write

```
chattr +C new_empty_folder
chattr +C file
```

Snapshot

- schnelle „Kopie“
- Zusammenfassung von vielen CoW-Kommandos im Hintergrund
- transaktionssicher
- Basis ist Subvolume
- Snapshot ist wieder Subvolume

```
btrfs subvolume snapshot subvol/btrfstestfiles/ abc
```

CoW, Beispiel

1 File, 850.000.000 Zeichen, 800M, reflinkmaster.txt

kopiere File mit *cp*: `cp --reflink=always reflinkmaster.txt reflinkcopy.txt`

ändere letztes Zeichen nach ‚0' mit *sed*

```
master                                                                copy
...
thoquoo4ibiewai0Phiy
iguu3Ieghe4gahcoog2e   ->   sed -i '$ s/.$/0/' reflinkcopy.txt   ->   ...
TaiTh7irah0geidibeev        tail -f reflinkcopy.txt                   thoquoo4ibiewai0Phiy
                                                                      iguu3Ieghe4gahcoog2e
                                                                      TaiTh7irah0geidibee0
```

filefrag -v (nach *cp*)

```
File size of reflinkmaster.txt is 850926120 (207746 blocks of 4096 bytes)
 ext:     logical_offset:        physical_offset: length:   expected: flags:
   0:        0..  207745:  272645376.. 272853121: 207746:             last,eof
reflinkmaster.txt: 1 extent found


File size of reflinkcopy.txt is 850926120 (207746 blocks of 4096 bytes)
 ext:     logical_offset:        physical_offset: length:   expected: flags:
   0:        0..  207745:  272645376.. 272853121: 207746:             last,shared,eof
reflinkcopy.txt: 1 extent found
```

filefrag -v (nach *sed*)

```
File size of reflinkcopy.txt is 850926120 (207746 blocks of 4096 bytes)
 ext:     logical_offset:        physical_offset: length:   expected: flags:
   0:        0..  163839:  174368941.. 174532780: 163840:
   1:   163840..  207745:  174595328.. 174639233:  43906:  174532781: last,eof
reflinkcopy.txt: 2 extents found
```

Subvolumes

Doku

- „A subvolume looks like a normal directory, with some additional operations described below.“
  https://btrfs.readthedocs.io/en/latest/Subvolumes.html

Subvolumes?

- funktionieren nicht wie normale Volumemanager
- „dynamische Partitionen“
- „Filesystem im Filesystem“
- keine Namensregeln, muss selbst gestaltet werden

mountable

- statt nur ROOT des Filesystems in Partition
- Mount ist nicht mehr fixe Zuordnung von Partitionen, sondern Zuordnung von Arbeitsbereichen
- Daten sind in tiefen Strukturen mit Subvolumes
- Mountpoints flach

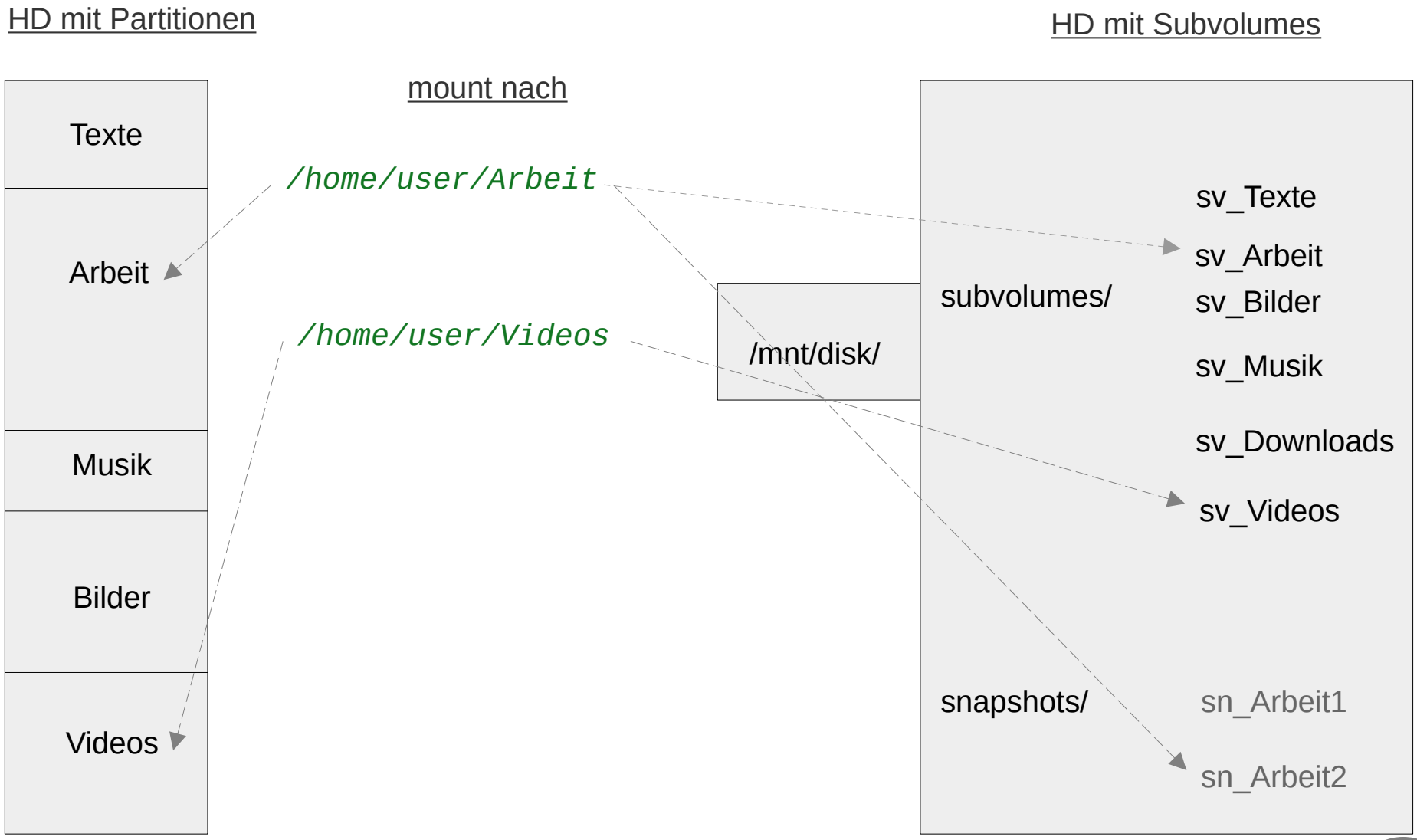
rekursiv

- Mountpoints bilden Zugriffsgrenze
- wie eine Schicht von zusätzlichen Rechten

Subvolumes sind mit Btrfs-Tools sichtbar

```
btrfs subvolume list folder
```

Partitionen vs. Subvolumes



Snapshot (=Subvolume)

Snapshot

- schnelle „Kopie“ eines Subvolume mit Reflinks

```
btrfs subvolume snapshot subvol/btrfstestfiles/ abc
```

nicht rekursiv

- Subvolumes downstream sind nicht enthalten
  - für selektive Backups nutzbar

Snapshots sind keine Backups

- nur durch Reflinks mit dem Subvolume verbundene Files
- wegen CoW wird Plattenplatz benötigt, wenn man einen Snapshot ändert
- Btrfs-Filesysteme nicht voll schreiben (ca. 70%)

Backups

- RO-Snapshots
- Backup von Subvolumes mit offenen Files möglich
- kein Zeitfenster für Backups nötig

Löschen

```
rm -r abc
```
(langsam)

```
btrfs subvolume delete abc
```
(sehr schnell)

Arbeiten mit Subvolumes und Snapshots

Arbeitsfolder: **workingfolder**

- „workingfolder“ bleibt gleich, *mount* zum *subvolume* mit den Daten wird angepasst

1. mount

```
mount -t btrfs -o subvol=daten/subvolume /dev/disk/by-partlabel/pl  workingfolder/
```

2. jeden Tag einen Snapshot

```
btrfs subvolume snapshot subvolume/ snaps/montag
btrfs subvolume snapshot subvolume/ snaps/dienstag
btrfs subvolume snapshot subvolume/ snaps/mittwoch
```

3. Wechsel auf Mittwoch, weil die aktuellen Arbeitsdaten kaputt sind ...

```
umount workingfolder
mount -t btrfs -o subvol=snaps/mittwoch /dev/disk/by-partlabel/p1  workingfolder/
```

4. Mittwoch zum Original machen, weil am Mittwoch noch alles ok war.

```
umount workingfolder
btrfs subvolume delete subvolume/
btrfs subvolume snapshot  snaps/mittwoch subvolume
```

Wir bauen ein RAID, mit LUKS

RAID in Btrfs ist anders

- Redundanz über Prüfsummen an den Files
- nicht über das Blockdevice darunter
- einfach erweiterbar

```
dd if=/dev/zero of=crdisk0.img bs=1 count=0 seek=200G
dd if=/dev/zero of=crdisk1.img bs=1 count=0 seek=200G

parted crdisk0.img mklabel gpt
parted crdisk1.img mklabel gpt
parted -a optimal -- crdisk0.img mkpart "crdisk0" 0% 100%
parted -a optimal -- crdisk1.img mkpart "crdisk1" 0% 100%
losetup /dev/loop600 -P crdisk0.img
losetup /dev/loop601 -P crdisk1.img

cryptsetup -q luksFormat /dev/disk/by-partlabel/crdisk0 keyfile
cryptsetup -q luksFormat /dev/disk/by-partlabel/crdisk1 keyfile
cryptsetup luksOpen /dev/disk/by-partlabel/crdisk0 --key-file keyfile crdisk0
cryptsetup luksOpen /dev/disk/by-partlabel/crdisk1 --key-file keyfile crdisk1

mkfs.btrfs -L crloop -m raid1 -d raid1 /dev/mapper/crdisk0  /dev/mapper/crdisk1
mount -L crloop rr
btrfs filesystem show rr
```

Vorgehensweise bei neuer Festplatte

neue HD
- mit viel Platz
- nur eine Partition mit Btrfs anlegen

Präfix (Namespace) für Subvolumes festlegen

sv .. = Subvolumes
sn .. = Snapshots
@.. = Subvolumes

mounten, nach Bedarf

/mnt/work
/mnt/cpp
- Basissubvolume der Partition selbst wird nicht gemounted
  - nur nötig, wenn man Strukturen ändert

mit Daten füllen, *rsync*, *mv*, *cp –reflink=always*
- keine Daten ohne Subvolume verwenden

Scripte anlegen für
- Snapshots
- Backup
- Organisation

Filesystem kopieren

(K)eine gute Idee?

Subvolumes

- oberstes Subvolume mounten
- alle Subvolumes feststellen
- am Ziel neu anlegen
- mit *rsync* jedes Subvolume kopieren (?)
- oder mit *send* von Btrfs
- downstream Subvolumes berücksichtigen

Snapshots

- wie Subvolumes kopieren
- Reflinks zu Quellsubvolumes gehen verloren
- benötigter Plattenplatz am Ziel u.U. größer

Demos

Demos:

- Ext4 nach Btrfs umwandeln
- Datenintegrität mit Scrub testen
- Softlink, Hardlink, Reflink
- `cp` oder `cp --reflink` oder Snapshot
- Arbeiten mit Snapshots
- RAID und LUKS

Danke

Take Home Message

- ein sicheres Filesystem mit Linux Bordmitteln ist machbar
- mit Btrfs gibt es eine zuverlässige Lösung

Vielen Dank für Eure Aufmerksamkeit.

Richard Albrecht

LUG-Ottobrunn
LUG-Görlitz
LUG-Zittau

https://www.görlinux.de