btrfs

Michael Göhler

Linux User Group Erding

25. September 2013

Inhalt

- Einleitung
- Features (Vorteile)
- Theorie
  - B-Tree
  - Aufbau
  - Copy On Write
  - Selbstheilung
- Nachteile
- Performance
- Praxisbeispiel

Features (1/2)

- Copy On Write
- Journaling implizit durch CoW
- Checksummen
- Fault isolation (Trennung von Daten und Metadaten)
- RAID 0/1/10 (5/6 experimental)
  - bessere Rebuild Zeiten
- SSD optimiert (-o ssd)
- Subvolumes
  - Rollback

Features (2/2)

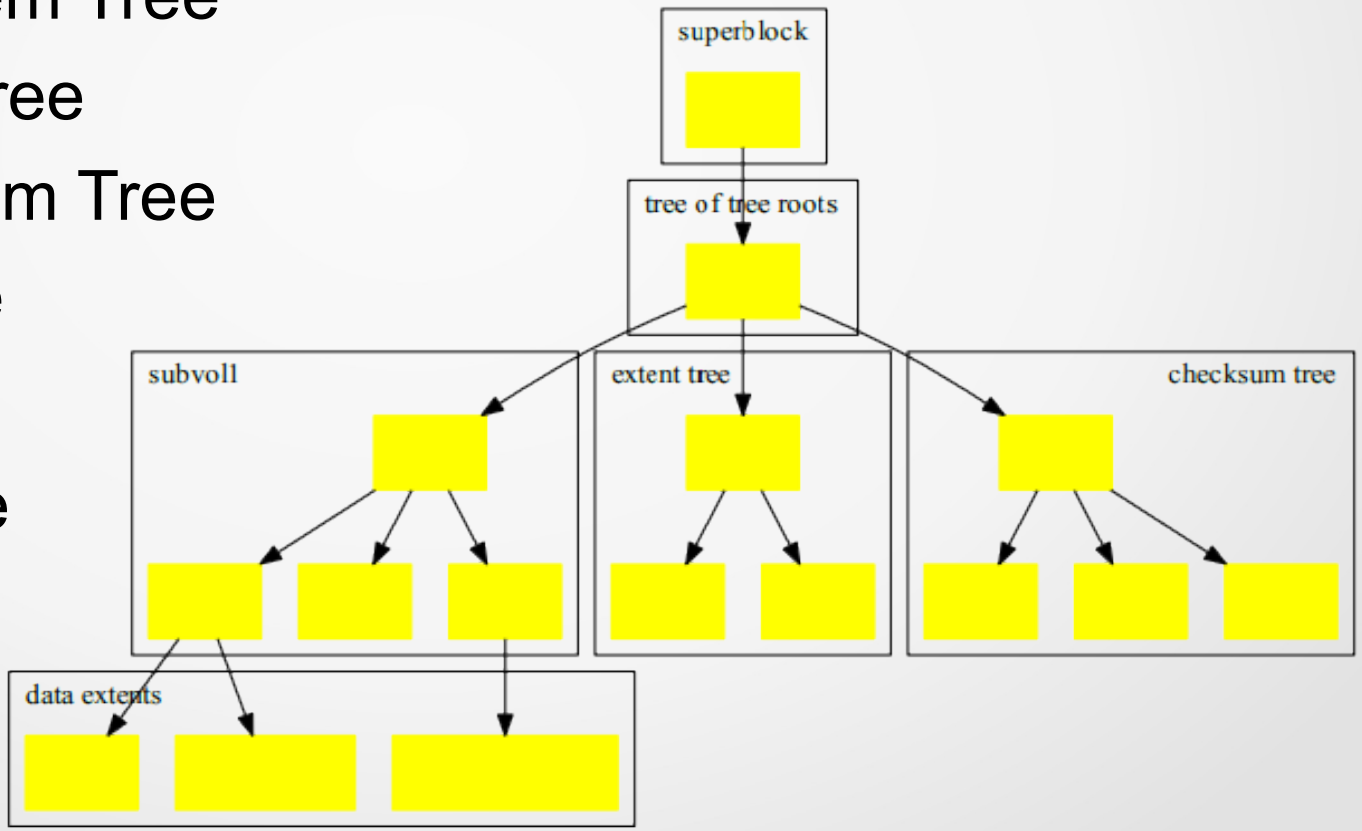
- Online
  - Vergrößern / verkleinern
  - Neue Platten hinzufügen
  - Raidlevel ändern
  - Defragmentieren
  - Komprimierung aktivieren / deaktivieren
  - fsck und Autokorrektur (btrfs scrub)
    - Btrfs hat kein offline fsck Modul

B-Tree

- B-Tree
  - Jede Verzweigung (Node) kann mehreren Verweise auf Unterverzweigungen enthalten
  - Daten sind nur in den Blättern (Leafs) enthalten
- B^+-Tree
  - Verweise von Leaf zu Leaf um sequenzielles Lesen zu beschleunigen
  - wird z.B. auch für Indexe in Relationalen Datenbanken verwendet

Aufbau (1/3)

- Super Block
  - Root Tree
    - Filesystem Tree
    - Extent Tree
    - Checksum Tree
    - Log Tree
  - Chunk Tree
  - Device Tree



Aufbau (2/3)

- Filesystem Tree
  - Metadaten der Dateien die für Anwender sichtbar sind
  - Referenzen auf logische Blöcke (Chunks) über Extent Tree
- Extent Tree
  - Enthält Block Gruppen mit Items die auf Chunks verweisen
- Chunk Tree
  - Enthält das Mapping zwischen logischen Chunks und physikalischen Blöcken auf der Platten

Aufbau (3/3)

- Checksum Tree
  - Enthält Checksummen für alle Chunks jedes Extents
- Log Tree
  - Wird für Protokollierung bei fsync (Flush to Disk) verwendet um COW Operationen asynchron durchführen zu können (herkömmliches Journaling)
    - Ähnlich Redo-Logs bei Datenbanken
- Device Tree
  - Mapping von der Festplatte zum Logischen Chunk
  - Zur Beschleunigung von Massendaten-Operationen z.B. Online Verkeinerung

Copy On Write

- Klonen
  - Metadaten werden bei jeder Änderung geklont **(billig)**
  - Bei Datendateien werden nur geänderte Chunks geklont und Referenzen aktualisert **(teuer)**

- Änderungen am den Trees erfolgen komplett im Hauptspeicher
  - Geänderte Trees werden alle 30 Sekunden auf die Festplatte geschrieben (nicht überschrieben)

Selbstheilung

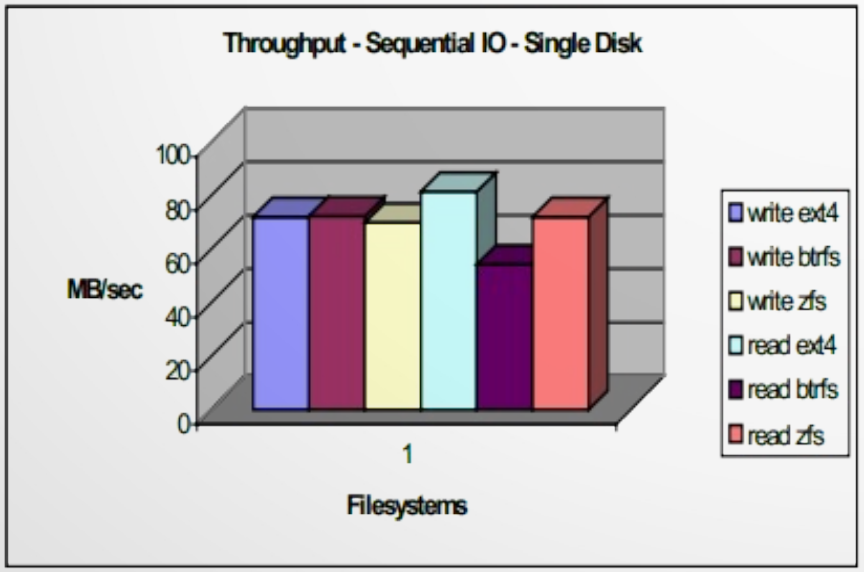
- Implizites Journaling durch Copy On Write vor jedem Schreibvorgang
- online fsck (btrfs scrub)
  - Ist eine Art Batch-Job (pausierbar)
  - Scannt Daten und Metadaten
  - CRC Fehler werden automatisch korrigiert wenn eine heile Kopie auf einer anderen Platte vorliegt
    - Empfehlung: immer Raid>0 einsetzen
- Wenn alles schief geht...
  - btrfs-restore kann Dateien aus einem nicht gemounteten Btrfs wiederherstellen
  - Backup (btrfs subvolume find-new)

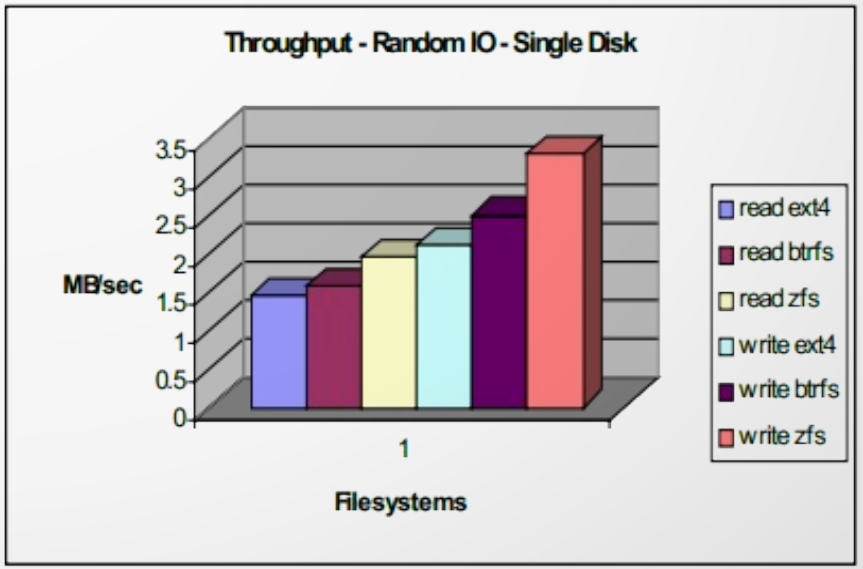
Nachteile

- Datenverlust von 30 Sekunden möglich bei Crash
  - Nur wenn Anwendungen keine fsync Operationen verwendet
    - Log Tree wird bei Auto-Recovery wieder angewendet
- Ungeeignet für Datenbank- oder VM-Hosts
  - Hohe Datenfragmentierung durch zu viele CoW Operationen bei zufälligen Schreibzugriffen in großen Dateien (-o nocow)
- Performance-Einbruch bei voller Festplatte
- Keine build-in Verschlüsselung (mit dm-crypt möglich)

Performance

- Btrfs ↔ Ext4
  - Btrfs ist schneller bei kleinen Dateien
  - Ext4 hat durch seine Extends bei großen Dateien die Nase vorn





Praxisbeispiel (1/2)

```
sudo su
modprobe loop
dd bs=1M count=100 if=/dev/zero of=disk0
dd bs=1M count=100 if=/dev/zero of=disk1
losetup /dev/loop0 disk0
losetup /dev/loop1 disk1

mkfs.btrfs -L loop -m raid1 -d raid1 /dev/loop0 /dev/loop1
btrfs device scan
blkid /dev/loop0 /dev/loop1
mkdir mnt
mount /dev/disk/by-uuid/9c42... mnt
df -h mnt
btrfs filesystem df mnt
btrfs filesystem show /dev/loop0
umount mnt
losetup -d /dev/loop0
btrfs filesystem show /dev/loop1
mount -o degraded /dev/disk/by-uuid/9c42... mnt
dd bs=1M count=100 if=/dev/zero of=disk2
losetup /dev/loop2 disk2
btrfs device add /dev/loop2
btrfs device delete missing mnt

dd bs=1M count=100 if=/dev/zero of=disk1 oflag=append conv=notrunc
losetup -c /dev/loop1
btrfs filesystem show /dev/loop1
btrfs filesystem resize 2:max mnt
btrfs filesystem show /dev/loop1
```

Praxisbeispiel (2/2)

- Btrfs als Root Filesystem
  - Snapshot & Rollback via Subvolumes
    - initcpio
      - Wählen des zu bootenden subvolumes aus initrd
    - kexec
      - Nachladen des Kernels vom richtigen subvolume
- Was noch nicht geht
  - Automatisches Mounten von RAIDs mit defekten Devices

Referenzen

- IBM Research Report RJ10501
  - Brtfs: The linux b-tree filesystem
  - von Ohad Rodeh, Josef Bacik, Chris Mason
- Wikipedia (englisch) – Btrfs
  - http://en.wikipedia.org/wiki/Btrfs
- Btrfs Wiki
  - https://btrfs.wiki.kernel.org/
- Performance Evaluation Btrfs, Ext4, ZFS
  - http://www.dhtusa.com/media/IOPerfCMG09.pdf