ZFS: Love Your Data

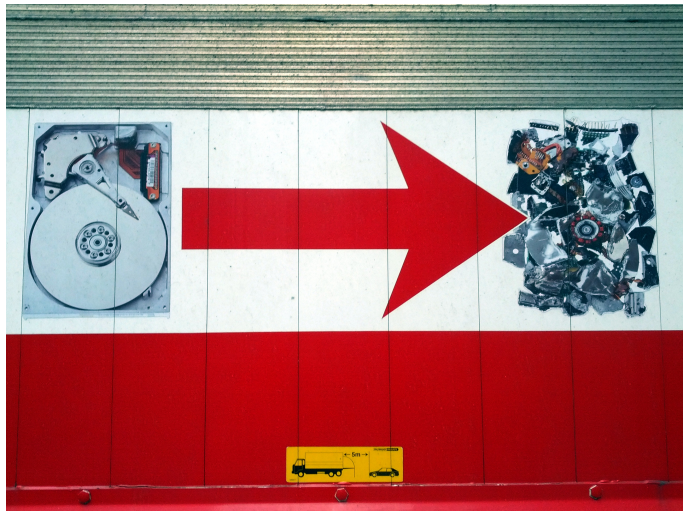
Neal H. Walfield

Düsseldorfer Fellowship Treffen, 30.7.2014

ZFS Besonderheiten

- Datensicherheit
  - End-to-End Daten Konsistenz durch Prüfsummen
  - Selbstheilend
  - Copy on Write Transaktionen
  - Zusätzliche Kopien von wichtigen Daten
- Snapshots und Clones
- Einfache, inkrementelle Replikation
- Vereinfachte Administration
  - Ein geteilter Pool, statt vieler statischer Volumes
- Verbesserte Leistung
  - Hierarchisches Speichermanagement (HSM)
  - Pooled Architektur $\Longrightarrow$ geteilte IOPs
  - Entwickelt für Many-Core Systeme
- Skalierbar
  - Pool Adressraum: 2^{128} bytes
  - $O(1)$ Operationen
  - Feine Granularität der Sperren

Festplatten Fehler: Bitrot



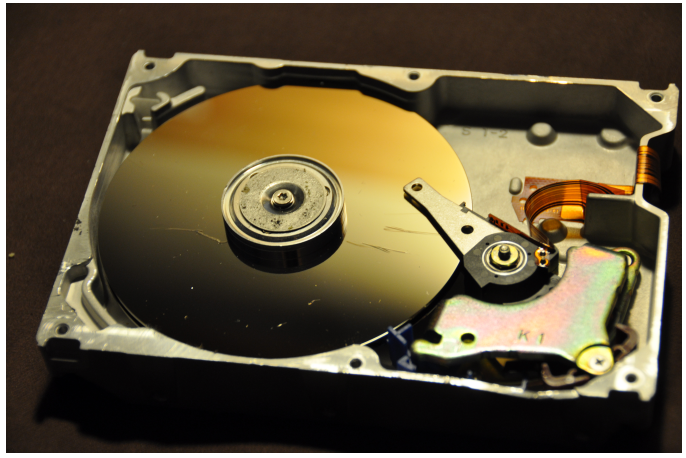
By Cory Doctorow, CC BY-SA 2.0

- BER (Bit Error Rate)
- Laut Datenblätter
  - Desktop: 1 zu 10^{14} (12 TB)
  - Enterprise: 1 zu 10^{15} (120 TB)
- Erfahrungsgemäß: 1 Schlechter Sektor pro 8 bis 20 TB*

*Jeff Bonwick and Bill Moore, ZFS: The Last Word in File Systems, 2008

Mehr Fehler Arten

- Phantom writes
- Misdirected read / write
  - 1 zu 10^8 bis 10^9 †
  - = 1 zu 50 bis 500 GB
- DMA Parity Errors
- Software / Firmware Bugs
- Administration Fehler



By abdallahh, CC BY-SA 2.0

†Ulrich Gräf: ZFS Internal Structures, OSDevCon, 2009

Fehler Kommen Häufiger vor!

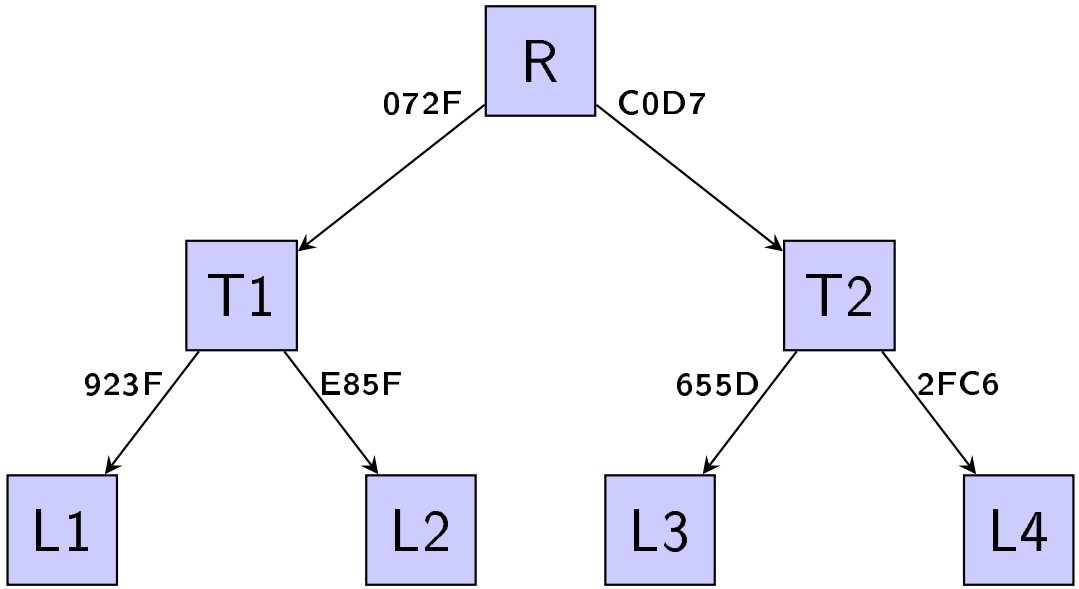
By Ilya, CC BY-SA 2.0

- Fehlerraten sind konstant geblieben
- Kapazität ist exponentiell gestiegen
- $\Longrightarrow$ viel mehr Fehler pro Zeiteinheit als früher!

Silent Data Corruption

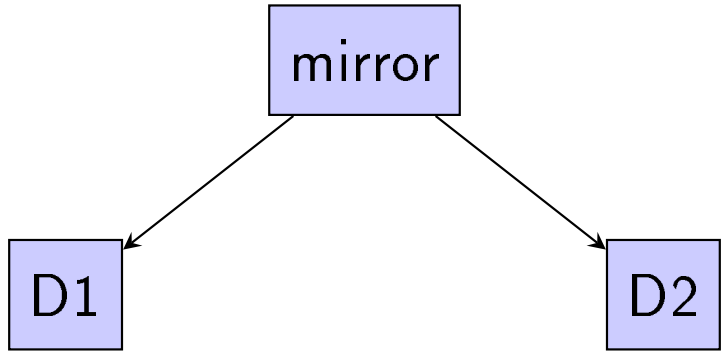
- Fehler passieren!
- Ziel:
  - Fehler erkennen
  - Fehler korrigieren

End to End Daten Konsistenz



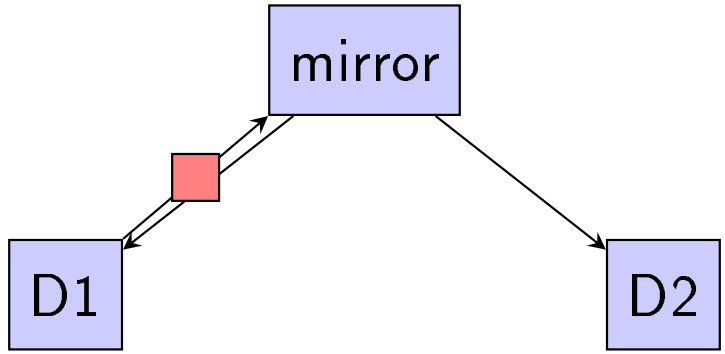
- Jeder Block hat eine Prüfsumme
- Die Prüfsumme wird mit dem Pointer gespeichert
  - Nicht neben den Daten!
  - Schutz gegen Phantom writes, etc.
- Formt einen Merkle Baum
- Wurzel wird durch mehrere Kopien gesichert

Selbstheilend



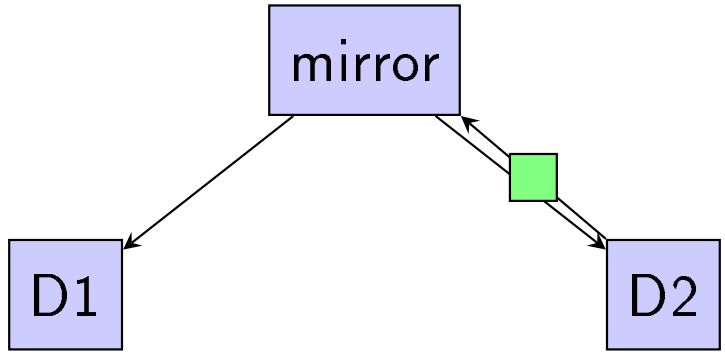
- Die Prüfsumme wird beim Lesen geprüft
- Falls sie nicht übereinstimmen, liest ZFS eine andere Kopie
- Die schlechte Kopie wird auch korrigiert

Selbstheilend



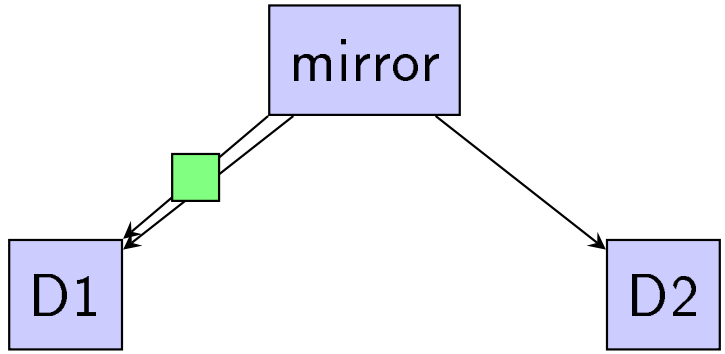
- Die Prüfsumme wird beim Lesen geprüft
- Falls sie nicht übereinstimmen, liest ZFS eine andere Kopie
- Die schlechte Kopie wird auch korrigiert

Selbstheilend



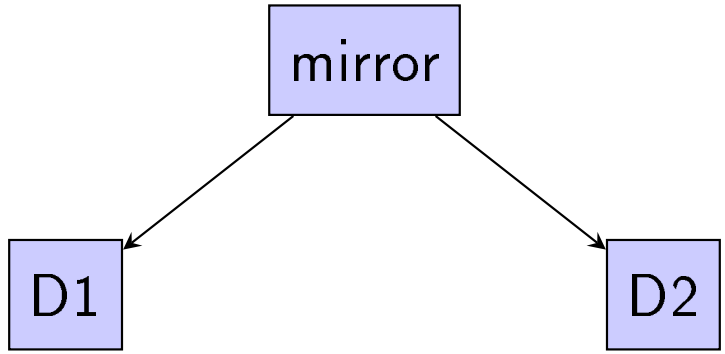
- Die Prüfsumme wird beim Lesen geprüft
- Falls sie nicht übereinstimmen, liest ZFS eine andere Kopie
- Die schlechte Kopie wird auch korrigiert

Selbstheilend



- Die Prüfsumme wird beim Lesen geprüft
- Falls sie nicht übereinstimmen, liest ZFS eine andere Kopie
- Die schlechte Kopie wird auch korrigiert

Selbstheilend



- Die Prüfsumme wird beim Lesen geprüft
- Falls sie nicht übereinstimmen, liest ZFS eine andere Kopie
- Die schlechte Kopie wird auch korrigiert

- `zfs scrub` prüft alle Daten in einem Pool
- Soll 1 bis 2 Mal monatlich ausgeführt werden

Selbstheilend: Demo

```
$ dd if=/dev/zero of=disk1 count=1M bs=1k
$ dd if=/dev/zero of=disk2 count=1M bs=1k
$ sudo zpool create test mirror $(pwd)/disk1 $(pwd)/disk2
$ sudo zpool status test
  pool: test
 state: ONLINE
  scan: none requested
config:

NAME                    STATE     READ WRITE CKSUM
test                    ONLINE       0     0     0
  mirror-0              ONLINE       0     0     0
    /home/us/tmp/disk1  ONLINE       0     0     0
    /home/us/tmp/disk2  ONLINE       0     0     0

errors: No known data errors
```

Selbstheilend Demo: Die! Die! Die!

```
$ sudo cp /usr/bin/s* /test
$ sudo zpool export test
$ dd if=/dev/zero of=disk1 conv=notrunc bs=4k count=100k
$ sudo zpool import test -d .
$ sudo zpool status test
  pool: test
 state: ONLINE
status: One or more devices has experienced an unrecoverable error.
An attempt was made to correct the error.  Applications are
unaffected.
action: Determine if the device needs to be replaced, and clear the
errors using 'zpool clear' or replace the device with 'zpool replace'.
   see: http://zfsonlinux.org/msg/ZFS-8000-9P
  scan: none requested
config:

NAME                    STATE     READ WRITE CKSUM
test                    ONLINE       0     0     0
  mirror-0              ONLINE       0     0     0
    /home/us/tmp/disk1  ONLINE       0     0    18
    /home/us/tmp/disk2  ONLINE       0     0     0

errors: No known data errors
```

Selbstheilend: Alles Ok!

```
$ sudo zpool import test -d .
$ md5sum /test/* >/dev/null
$ sudo zpool status test
  pool: test
 state: ONLINE
status: One or more devices has experienced an unrecoverable error.
        An attempt was made to correct the error.  Applications are
        unaffected.
action: Determine if the device needs to be replaced, and clear the
        errors using 'zpool clear' or replace the device with
        'zpool replace'.
   see: http://zfsonlinux.org/msg/ZFS-8000-9P
  scan: none requested
config:

NAME                    STATE     READ WRITE CKSUM
test                    ONLINE       0     0     0
  mirror-0              ONLINE       0     0     0
    /home/us/tmp/disk1  ONLINE       0     0    47
    /home/us/tmp/disk2  ONLINE       0     0     0

errors: No known data errors
```

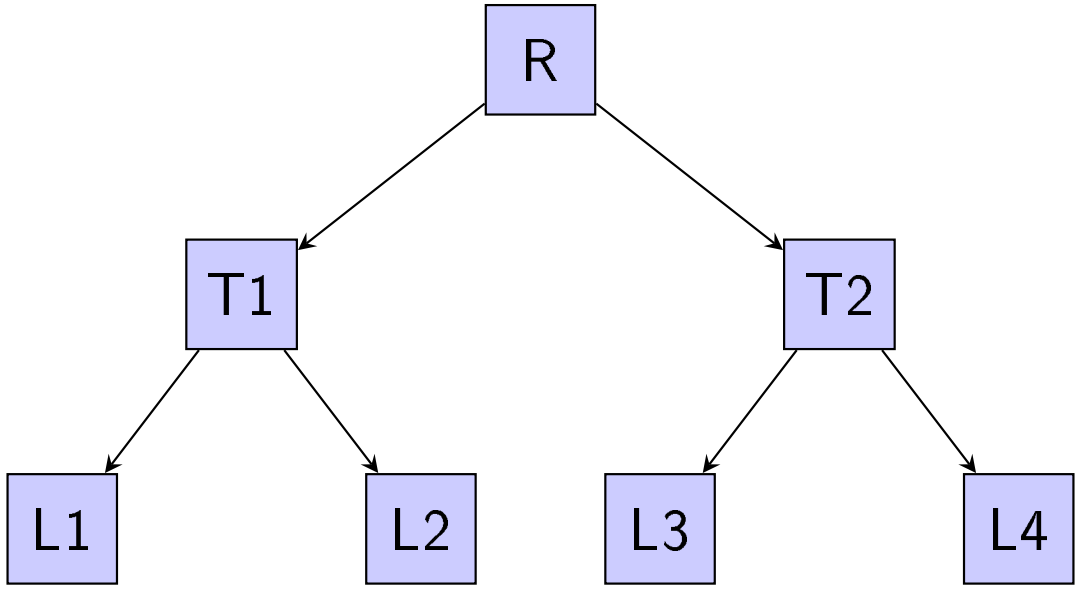
Selbstheilend: Clean Up

```
$ sudo zpool scrub
$ sudo zpool status test
  pool: test
 state: ONLINE
status: One or more devices has experienced an unrecoverable error.
        An attempt was made to correct the error.  Applications are
        unaffected.
action: Determine if the device needs to be replaced, and clear the
        errors using 'zpool clear' or replace the device with
        'zpool replace'.
   see: http://zfsonlinux.org/msg/ZFS-8000-9P
  scan: scrub repaired 4.33M in 0h0m with 0 errors on Wed Jul 30 14:22
config:

NAME                    STATE     READ WRITE CKSUM
test                    ONLINE       0     0     0
  mirror-0              ONLINE       0     0     0
    /home/us/tmp/disk1  ONLINE       0     0   209
    /home/us/tmp/disk2  ONLINE       0     0     0

errors: No known data errors
$ sudo zpool clear test
```

Selbstheilend: Zusammenfassung

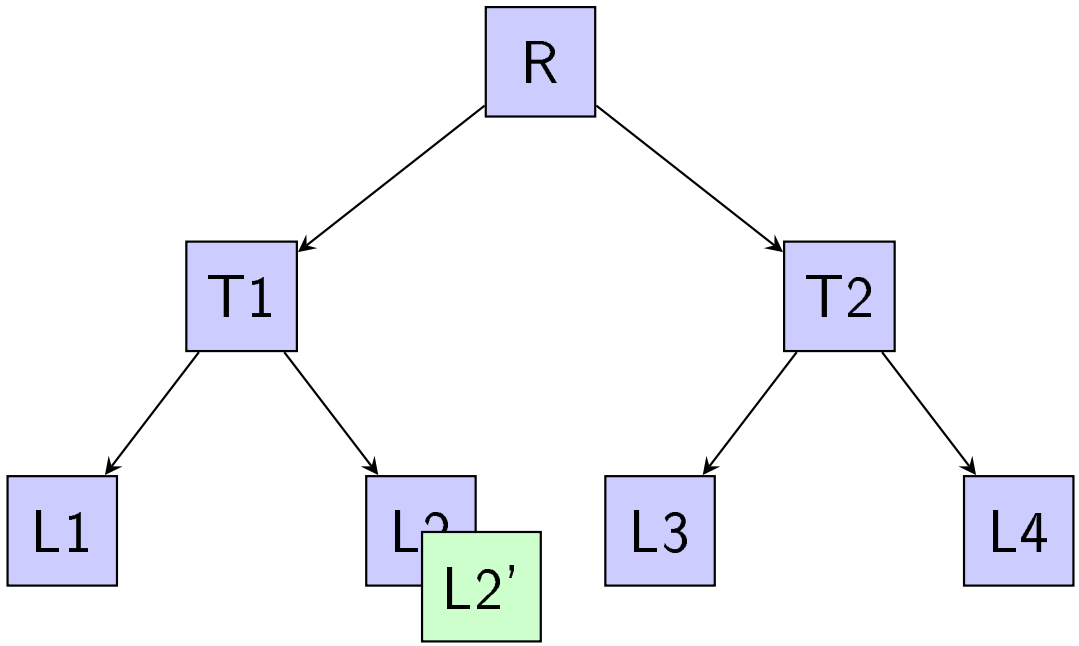
- Funktioniert nicht mit normalem RAID!
  - Die RAID / Volumen Manager / Dateisystem Aufteilung ist falsch!
  - Kam nur zustande, weil man die Dateisysteme nicht neu schreiben wollte!
    - Volume Manage bietet das Block Device Protokol an
    - Super Hack!
    - Leider geblieben

Copy on Write Transaktionen



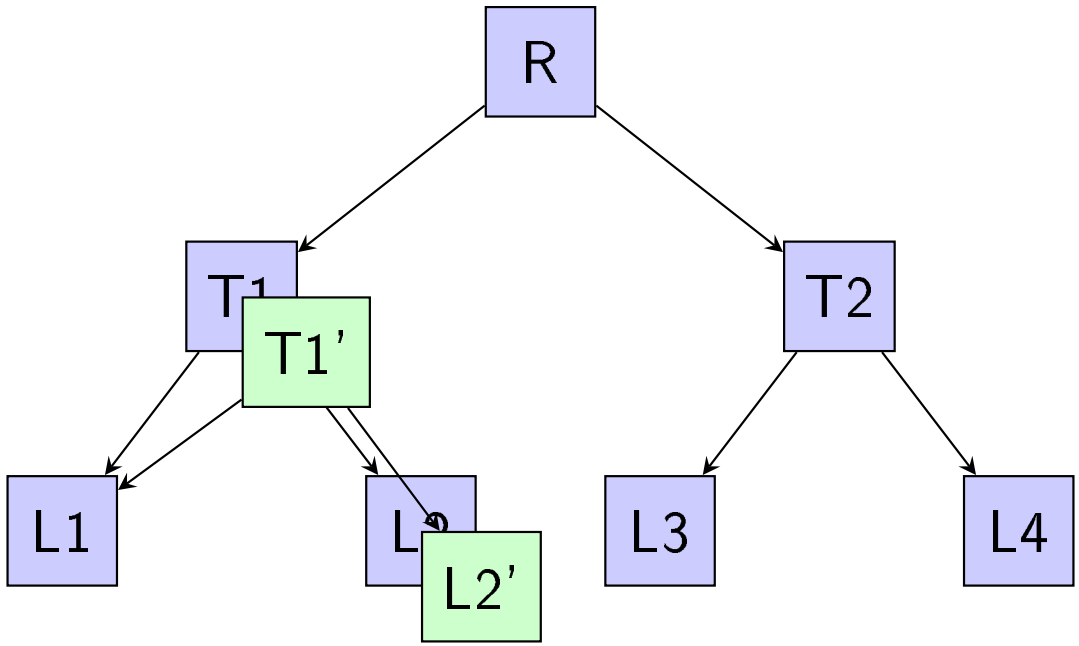
- Lebendige Daten werden nie überschrieben!
- Um einen Block zu ändern wird ein neuen Block allokiert
- Der Pointer wird aktualisiert in der gleichen Art und Weise
- $\Longrightarrow$ Die Daten bleiben (on disk) immer konsistent!
- Nicht referenzierte Blöcke werden am Ende freigegeben

Copy on Write Transaktionen



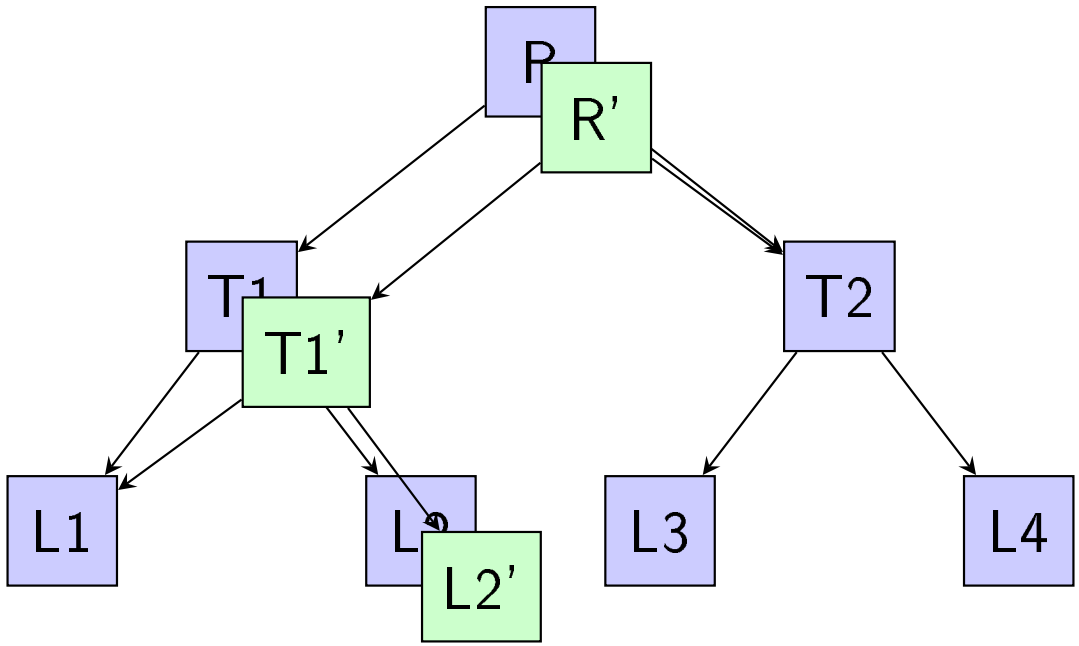
- Lebendige Daten werden nie überschrieben!
- Um einen Block zu ändern wird ein neuen Block allokiert
- Der Pointer wird aktualisiert in der gleichen Art und Weise
- $\Longrightarrow$ Die Daten bleiben (on disk) immer konsistent!
- Nicht referenzierte Blöcke werden am Ende freigegeben

Copy on Write Transaktionen



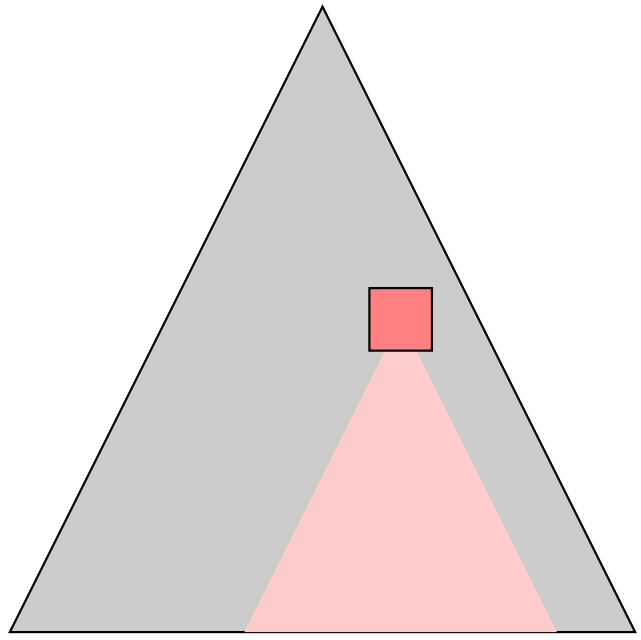
- Lebendige Daten werden nie überschrieben!
- Um einen Block zu ändern wird ein neuen Block allokiert
- Der Pointer wird aktualisiert in der gleichen Art und Weise
- $\Longrightarrow$ Die Daten bleiben (on disk) immer konsistent!
- Nicht referenzierte Blöcke werden am Ende freigegeben

Copy on Write Transaktionen



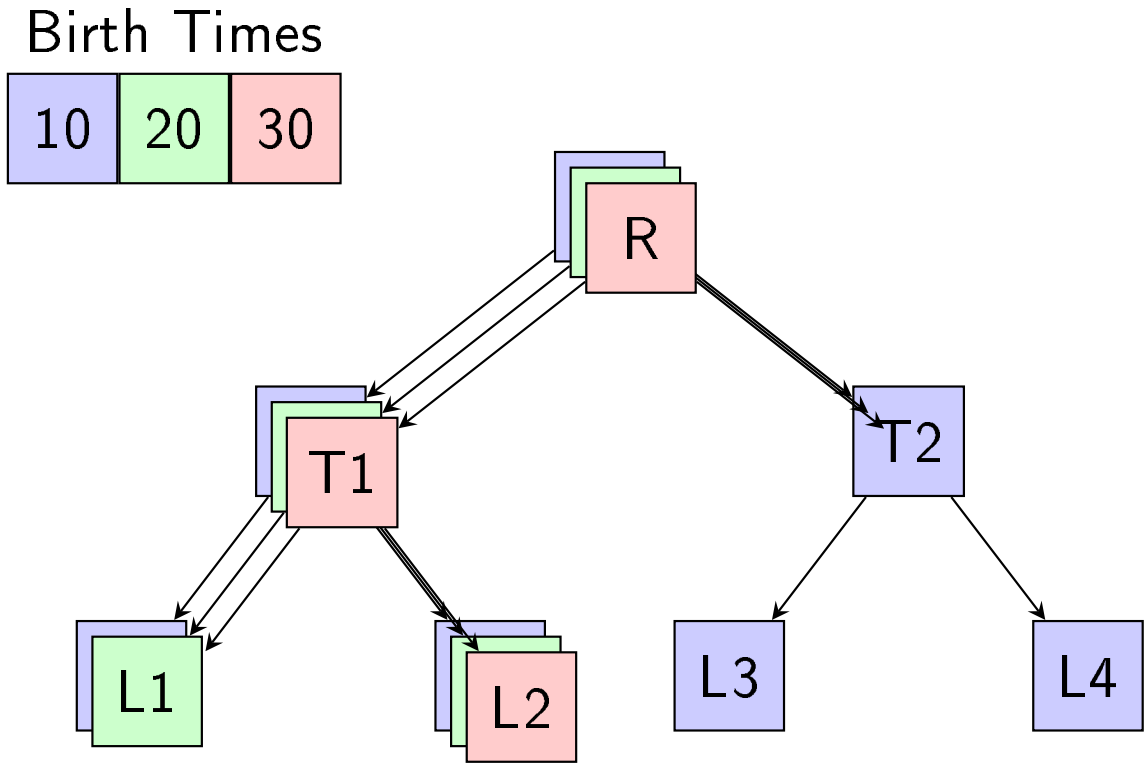
- Lebendige Daten werden nie überschrieben!
- Um einen Block zu ändern wird ein neuen Block allokiert
- Der Pointer wird aktualisiert in der gleichen Art und Weise
- $\Longrightarrow$ Die Daten bleiben (on disk) immer konsistent!
- Nicht referenzierte Blöcke werden am Ende freigegeben

Zusätzliche Kopien von wichtigen Daten



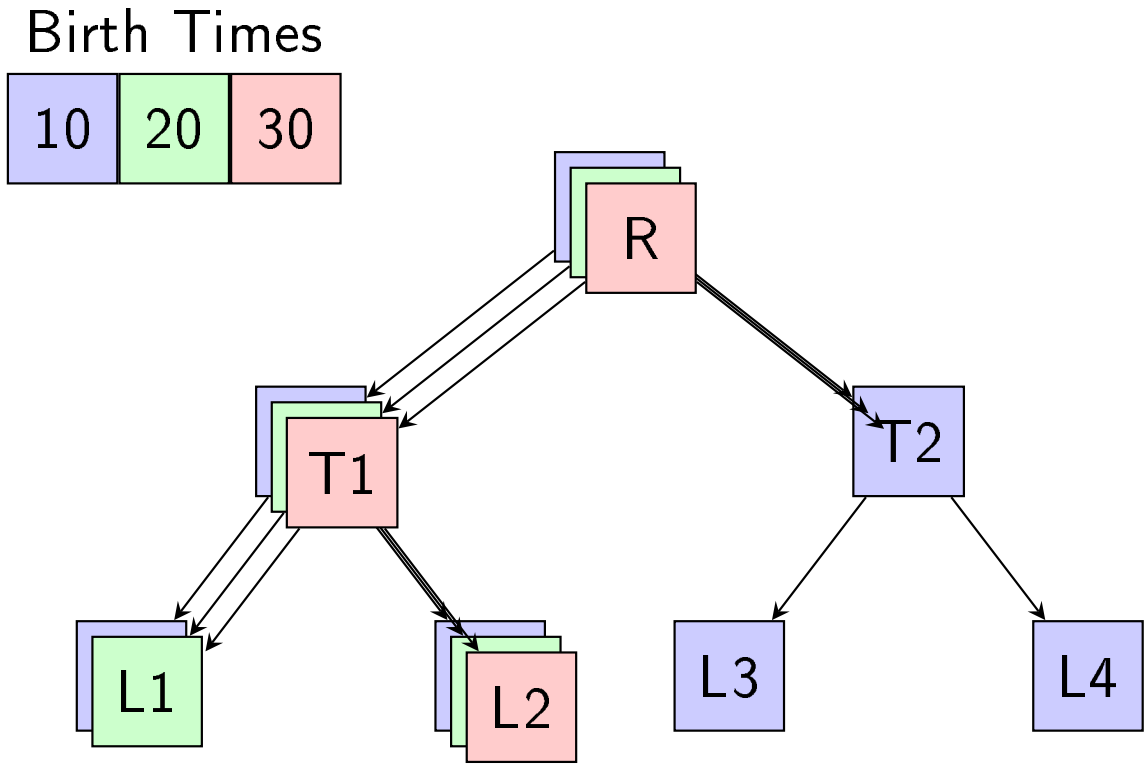
- Metadaten werden mehrmals gespeichert
  - Dateisystem Metadaten: 2 Mal
  - Pool Metadaten: 3 Mal
  - “Uberblock:” 4 Mal
  - Metadaten sind ungefähr 1% bis 2% der gesamten Daten
- Man kann auch mehre Kopien von normalen Daten speichern
  - Gut wenn man nur eine Festplatte hat (Laptop)

Snapshots und Clones



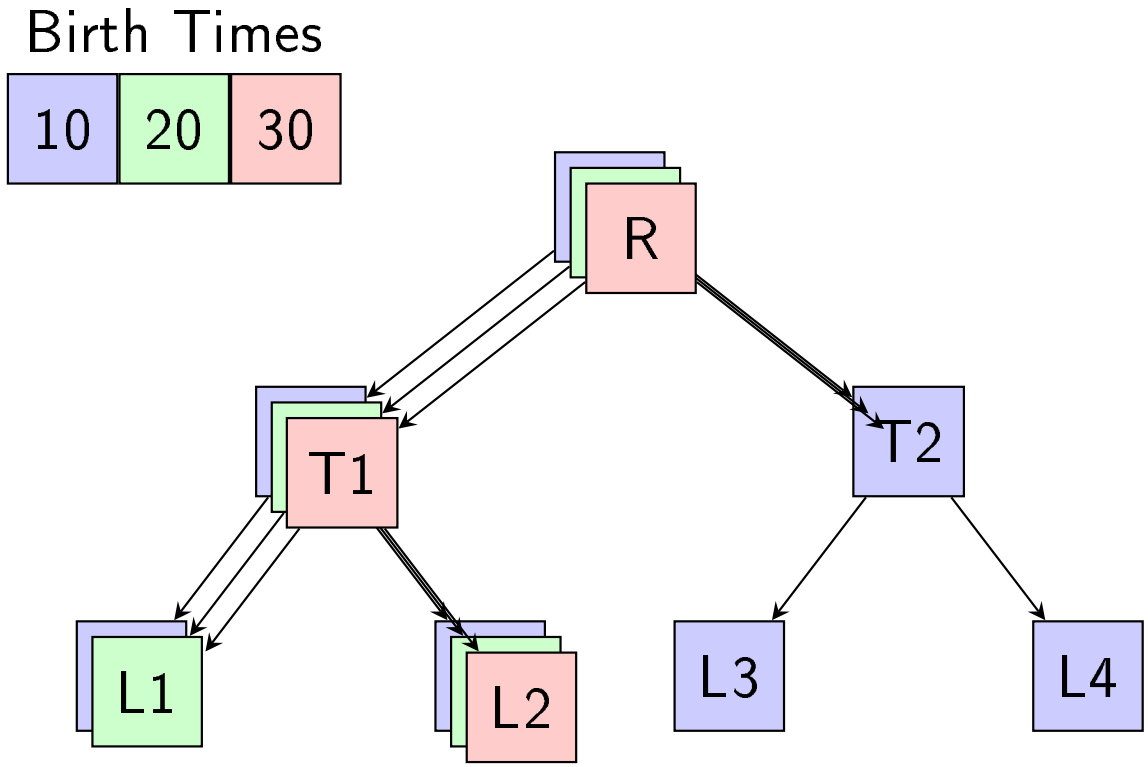
- Birth Time ist die Nummer der Transaktion in der der Block allokiert wird
- Blöcke haben kein reference count

Snapshots und Clones



- Um einen Block zu löschen:
  - Wird die Birth Time mit der des neusten Snapshot verglichen
  - Größer? Block kann freigeben werden.
  - Kleiner? Block ist noch im Gebrauch.
    - Wird zum Dead List des Snapshots hinzugefügt.

Snapshots und Clones

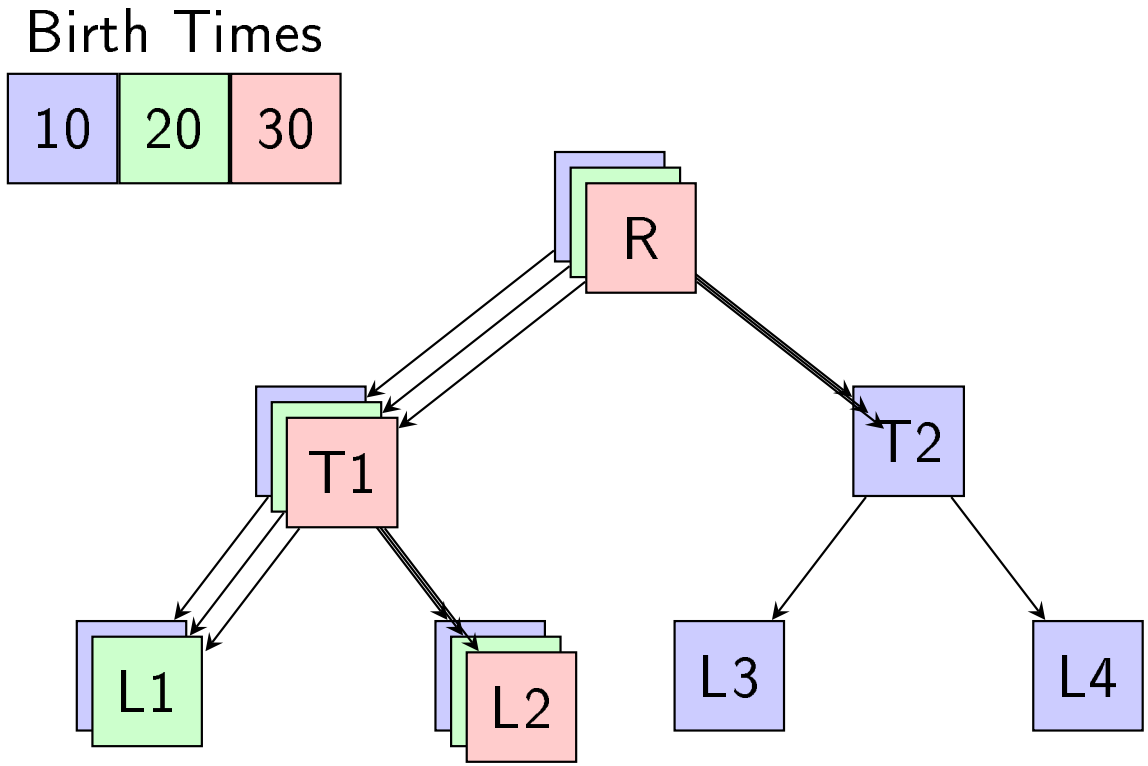


- Um einen Snapshot zu löschen:
  - Kann Birth Time wieder verwendet werden
  - Siehe `https://blogs.oracle.com/ahrens/entry/is_it_magic`

Snapshot Demo

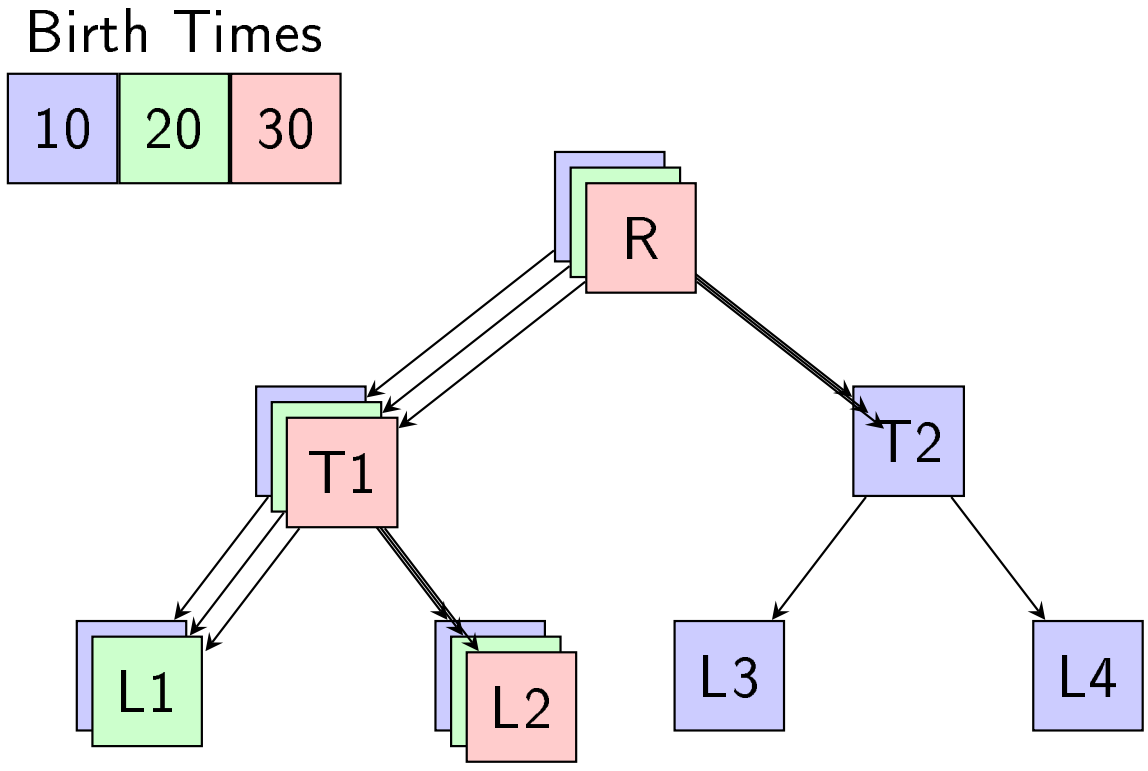
```
$ sudo touch /test/a
$ sudo zfs snapshot test@1
$ sudo rm /test/a
$ ls /test/.zfs/snapshot/1
a
$ ls /test/
$ sudo zfs diff test@1 test
M /test/
- /test/a
```

Einfache, Inkrementelle Replikation



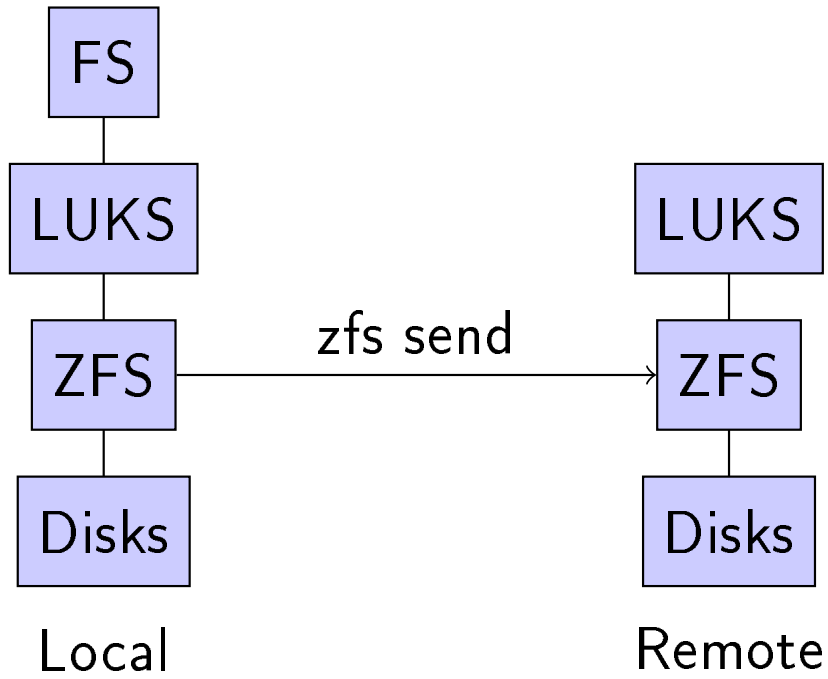
- Inkrementelle Replikation
  - `zfs send [base snapshot] snapshot`
  - `| ssh zfs receive fs`

Einfache, Inkrementelle Replikation



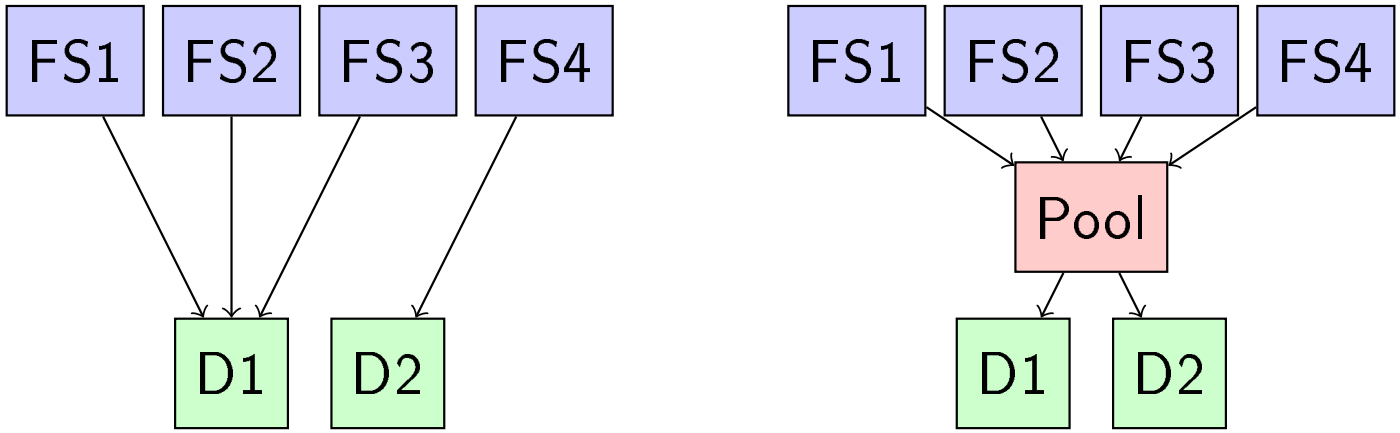
- Viel schneller als `rsync`
  - `rsync` muss alle Dateien `stat`en
  - Beim Vergleich der Bäume, kann ZFS aggressiv zurückschneiden
  - Replikation dauert nur Sekunden
    - Auch bei multi-TB Dateisystemen
    - $\Longrightarrow$ Kann, z.B., jede Minute repliziern

Verschlüsselte Offsite Backups?



- Ja!
- Nutzt ein LUKs container in dem ZFS Dateisystem

Vereinfachte Administration



- Ein geteilter Pool, statt vieler statischer Volumes
- Wir verwalten Festplatten, wie wir Arbeitsspeicher verwalten!
  - $\Longrightarrow$ Meist gar nicht!
  - Können immer noch quotas und Reservierungen benutzen.
- Bonus: Geteilte IOPs!

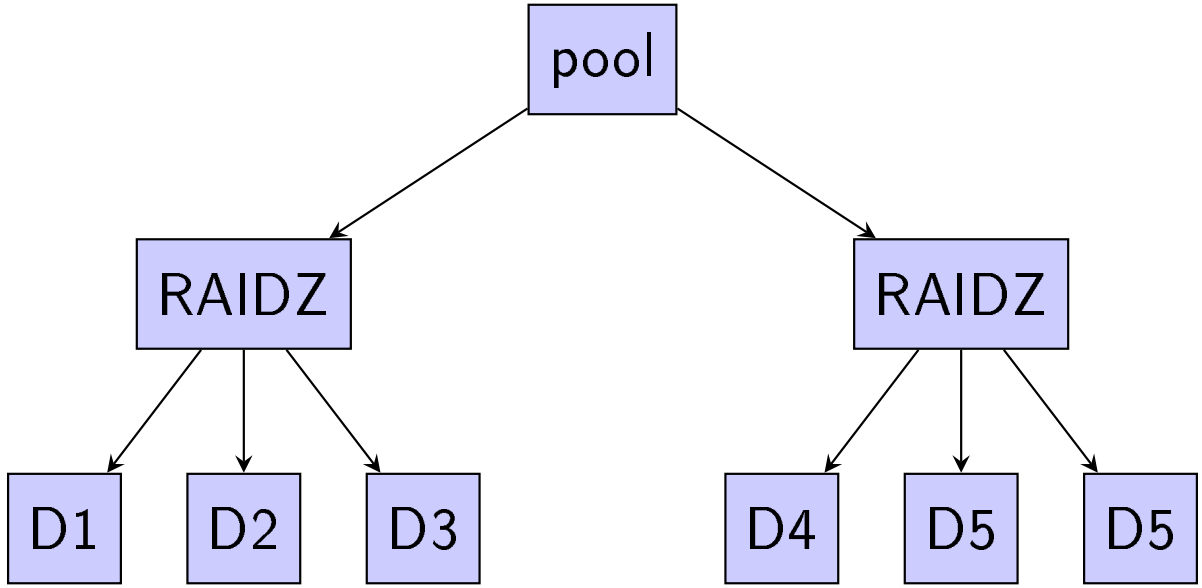
Hierarchisches Speichermanagement (HSM)

- SSD als Read Cache
  - Write-Through Cache
  - Ein Fehler beeinflusst nur die Performance, nicht die Richtigkeit
  - $\Longrightarrow$ MLC SSD reicht völlig
  - Automatisch verwaltet wie der Page Cache

Hierarchisches Speichermanagement (HSM)

- SSD als Write Cache
  - Synchronous Writes werden hier gespeichert, bis sie auf den Platten landen
  - Latenz geht von 4-6 ms auf 50 μs (factor 100)
  - 1 GB platz reicht
  - Aber es wird ständig geschrieben!
  - Am besten ein SLC oder eMLC SSD nutzen
    - Oder ein sehr großes SSD
  - Auch wenn der SSD Kaputt geht, gehen Daten nur verloren wenn der Server abstürzt
  - Die Daten werden nur bei Wiederherstellung gelesen

ZFS Pool



- ZFS bindet Festplatten Gruppen zusammen (logical vdev)
- Gruppen bestehen aus mehre Festplatten (physical vdev)
  - Mirrors
  - RAIDs
- Sehr einfach ein Pool zu vergrößern: neue Gruppe hinzufügen

Hardware Tips: ECC

- ECC Arbeitspeicher!
  - ZFS vertraut dem Arbeitsspeicher
  - Google Studie: 4k Fehler pro Jahr pro DIMM!
    `http://www.zdnet.com/blog/storage/dram-error-rates-nightmare-on-dimm-street/638`

Hardware Tips: Mainboard

- Intel Atom C2750 basiert Mainboards
  - Ideal, aber teuer (220 - 500 Euro für das Mainboard)
  - Mehrere Ethernet Anschlüsse
  - 6-12 SATA Anschlüsse
  - 8 Cores
  - 20 W (Max)
  - AES-NI
- AMD Prozessoren (fast?) immer Unterstützen ECC
  - HP G7 Microserver N54L: 150 Euro
  - 2 Cores
  - 4 SATA Anschlüsse

Hardware Tips: Festplatten

- Lieber “NAS” Festplatten
  - Unterstützt TLER (fast fail)
  - Festplatten kann minuten lang versuchen Fehler zu korrigieren
  - ZFS kann aber eine andere Festplatte benutzen

Hardware Tips: Vielfalt

Vielfalt verhindert korrelierte Fehler!

- Festplatten von unterschiedlichen Hersteller
- Platten der gleichen Gruppe an unterschiedliche SATA Controller anbinden
- etc.

Warum nicht BTRFS?

Russel Coker: Monatliche BTRFS Status Reports[‡]

- Since my blog post about BTRFS in March [1] not much has changed for me. Until yesterday I was using 3.13 kernels on all my systems and dealing with the occasional kmail index file corruption problem.
  Yesterday my main workstation ran out of disk space and went read-only. I started a BTRFS balance which didn't seem to be doing any good because most of the space was actually in use so I deleted a bunch of snapshots. Then my X session aborted (some problem with KDE or the X server – I'll never know as logs couldn't be written to disk). I rebooted the system and had kernel threads go into infinite loops with repeated messages about a lack of response for 22 seconds (I should have photographed the screen).
  ...

[‡] http://etbe.coker.com.au/2014/04/26/btrfs-status-april-2014/

Danke!

Mehr Information zu ZFS on Linux:

- `http://zfsonlinux.org/`
- `zfs-discuss@zfsonlinux.org`
- Install ZFS on Debian GNU/Linux von Aaron Toponce:
  `https://pthree.org/2012/04/17/install-zfs-on-debian-gnulinux/`

Copyright

Images (all are CC BY-SA 2.0):

- Cory Doctorow `https://flic.kr/p/bvNXHu`
- abdallahh `https://flic.kr/p/6VHVze`
- Ilya `https://flic.kr/p/7PUjGu`