ZFS: Love Your Data

Neal H. Walfield

RMLL, 6 July 2015

ZFS Features

- Security
  - End-to-End consistency via checksums
  - Self Healing
  - Copy on Write Transactions
  - Additional copies of important data
- Snapshots and Clones
- Simple, Incremental Remote Replication
- Easier Administration
  - One shared pool rather than many statically-sized volumes
- Performance Improvements
  - Hierarchical Storage Management (HSM)
  - Pooled Architecture $\Longrightarrow$ shared IOPs
  - Developed for many-core systems
- Scalable
  - Pool Address Space: 2^{128} bytes
  - $O(1)$ operations
  - Fine-grained locks

Hard Drive Errors

Silent Data Corruption

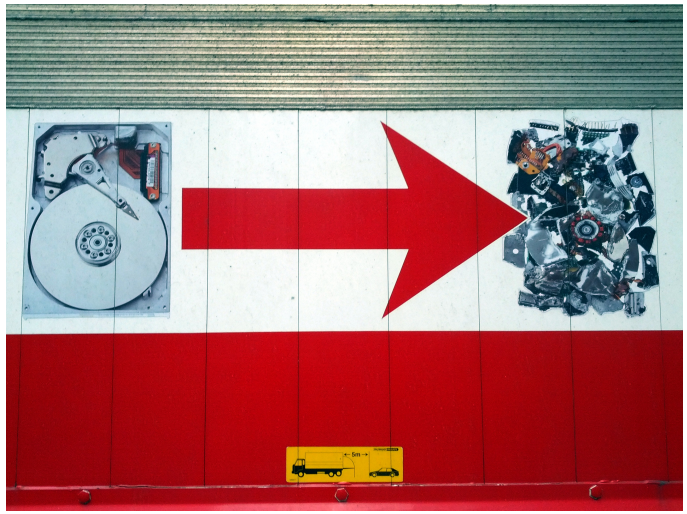
- Data errors that are not caught by hard drive
- = Read returns different data from what was written

Silent Data Corruption

- Data errors that are not caught by hard drive
- = Read returns different data from what was written

- On-disk data is protected by ECC
- But, doesn't correct / catch all errors

Uncorrectable Errors



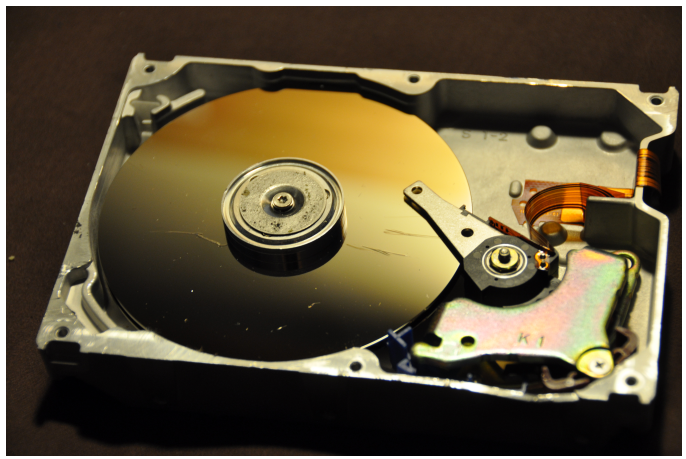
By Cory Doctorow, CC BY-SA 2.0

- Reported as BER (Bit Error Rate)
- According to Data Sheets:
  - Desktop: 1 corrupted bit per 10^{14} (12 TB)
  - Enterprise: 1 corrupted bit per 10^{15} (120 TB)
- Practice: 1 corrupted sector per 8 to 20 TB*

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

Types of Errors

- Bit Rot
- Phantom writes
- Misdirected read / write
  - 1 per 10^8 to 10^9 IOs [†]
  - = 1 error per 50 to 500 GB (assuming 512 byte IOs)
- DMA Parity Errors
- Software / Firmware Bugs
- Administration Errors



By abdallahh, CC BY-SA 2.0

[†] Ulrich Gräf: ZFS Internal Structures, OSDevCon, 2009

Errors are Occuring More Frequently!

By Ilya, CC BY-SA 2.0

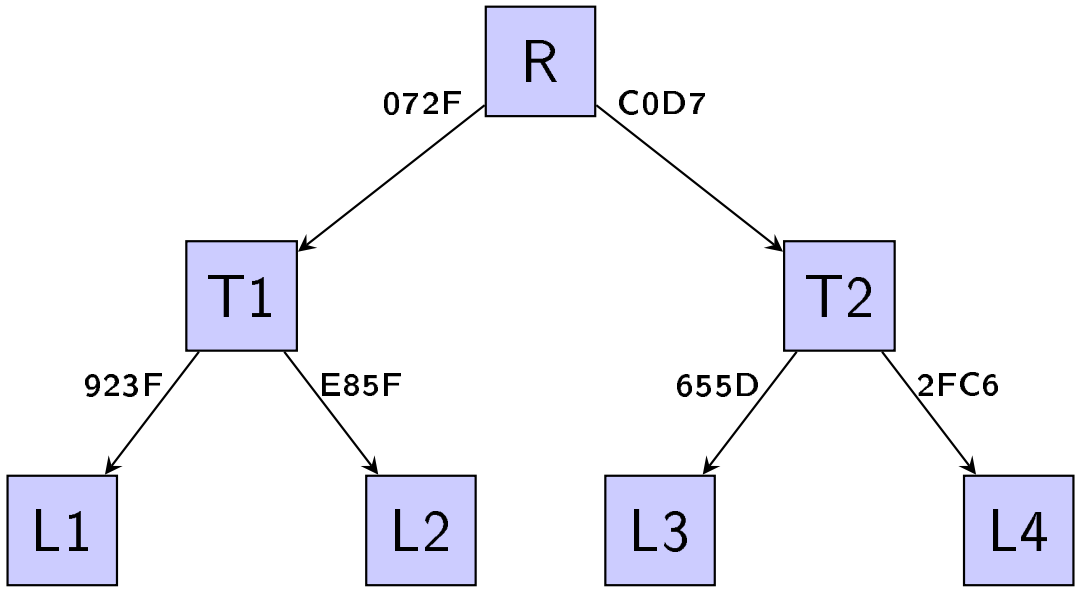
- Error rate has remained constant
- Capacity has increased exponentially
- $\Longrightarrow$ many more errors per unit time!

Summary

- Uncorrectable errors are unavoidable
- More ECC is not going to solve the problem
- Solution:
  - Recognize errors
  - Correct errors

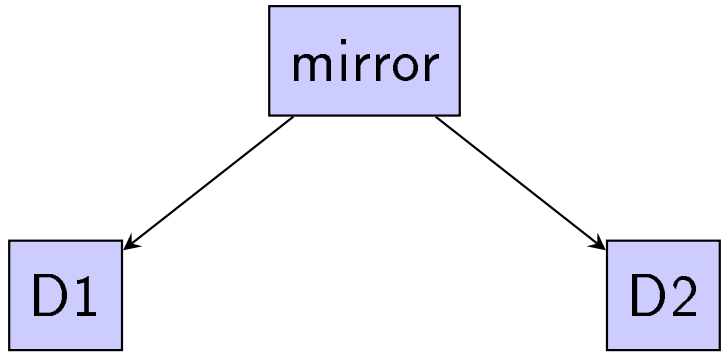
A Look at ZFS' Features

End to End Data Consistency



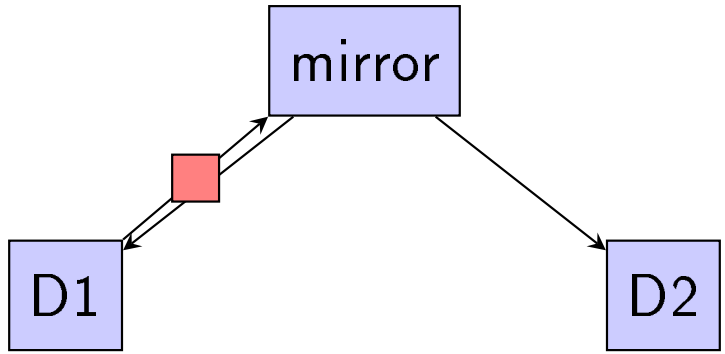
- Every block has a checksum
- The checksum is stored with the pointer
  - Not next to the data!
  - Protects against phantom writes, etc.
- Forms a Merkle Tree
  - (Root is secured using multiple copies.)

Self Healing



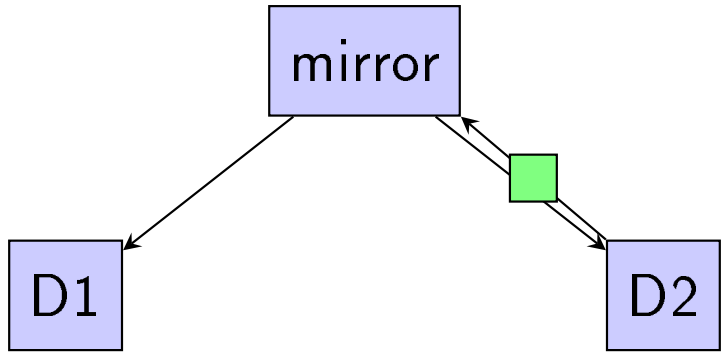
- Checksum verified when data is read
- On mismatch, ZFS reads a different copy
- The bad copy is automatically corrected

Self Healing



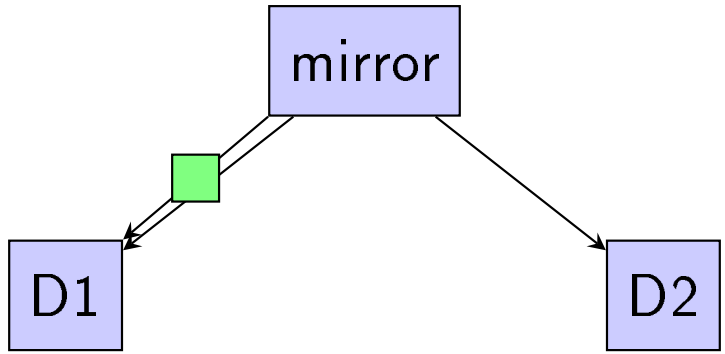
- Checksum verified when data is read
- On mismatch, ZFS reads a different copy
- The bad copy is automatically corrected

Self Healing



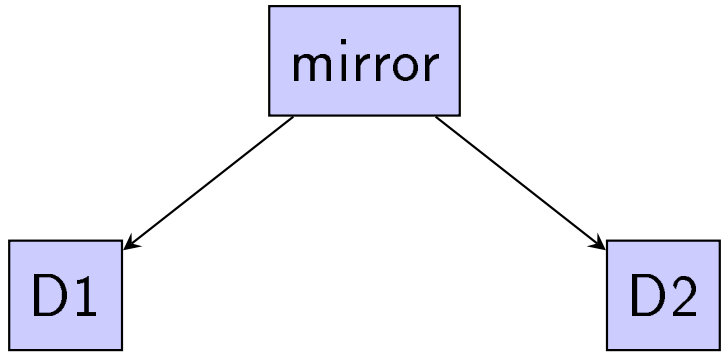
- Checksum verified when data is read
- On mismatch, ZFS reads a different copy
- The bad copy is automatically corrected

Self Healing



- Checksum verified when data is read
- On mismatch, ZFS reads a different copy
- The bad copy is automatically corrected

Self Healing



- Checksum verified when data is read
- On mismatch, ZFS reads a different copy
- The bad copy is automatically corrected

- `zfs scrub` checks all of the data in the pool
- Should be run once or twice a month.

Self Healing: 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
```

Initialize a mirrored pool

Status: healthy

Self Healing 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
```

Copy data

Corrupt it

Errors on reimport.

Errors

Self Healing: No application errors!

```
$ 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
```

Read the data

More errors.

Self Healing: 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
```

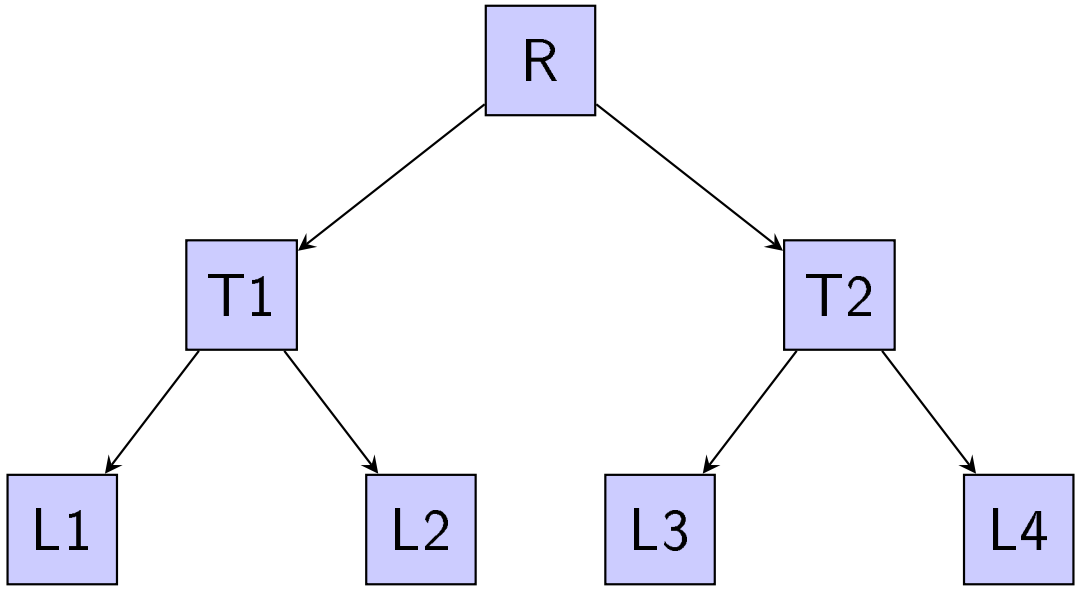
Scrub the disks

All errors repaired!

Self Healing: Summary

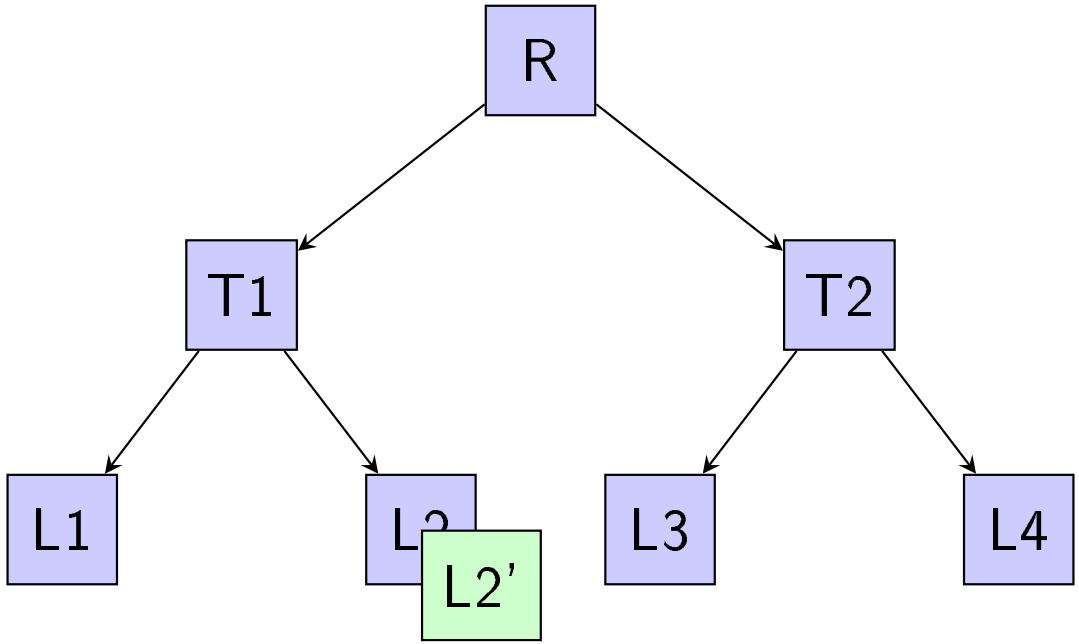
- Self Healing not possible with RAID
  - RAID / Volume Manager / FS layering is wrong!
  - Volume manager was a hack from the 80s!
    - Don't modify the FS to support multiple disks
    - Add a layer of indirection
    - FS talks to volume manager
    - Volume manager has same interface as a block device

Copy on Write Transactions



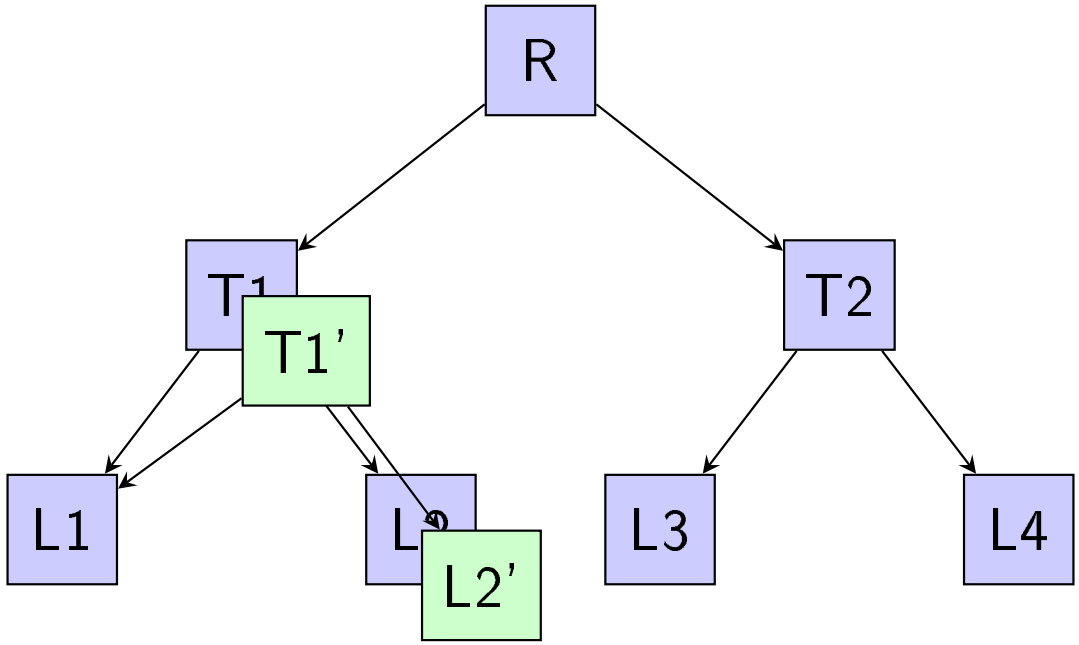
- Live data is never overwritten
- To change a block, a new block is allocated
- The pointer is updated in the same manner
- $\Longrightarrow$ Data on disk is always consistent
- Unreferenced blocks are freed at the end of the transaction

Copy on Write Transactions



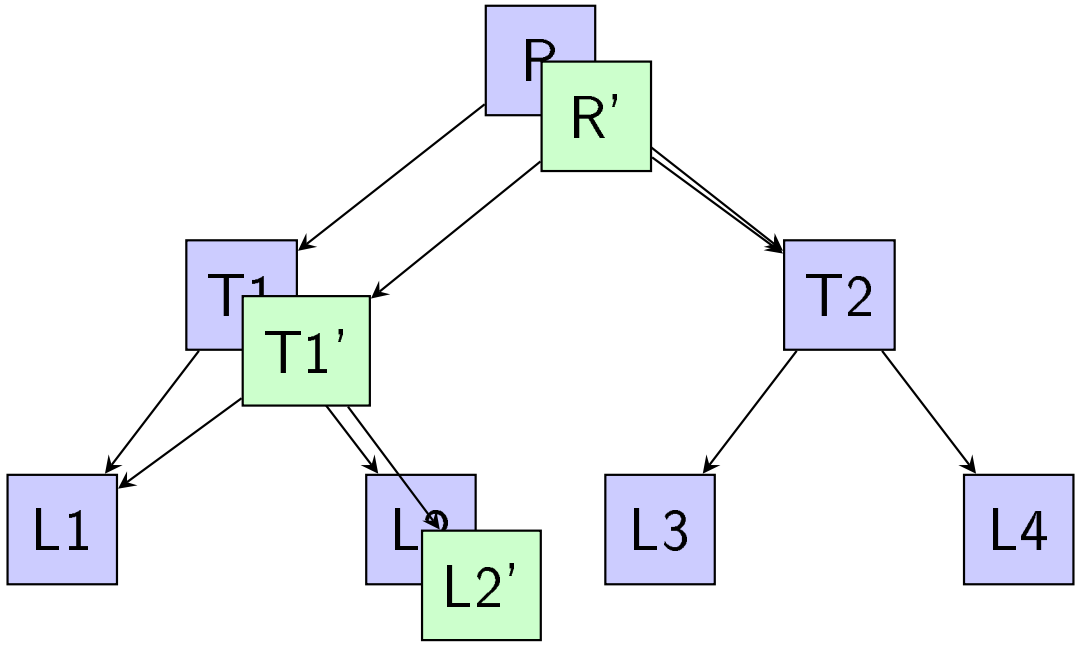
- Live data is never overwritten
- To change a block, a new block is allocated
- The pointer is updated in the same manner
- $\Longrightarrow$ Data on disk is always consistent
- Unreferenced blocks are freed at the end of the transaction

Copy on Write Transactions



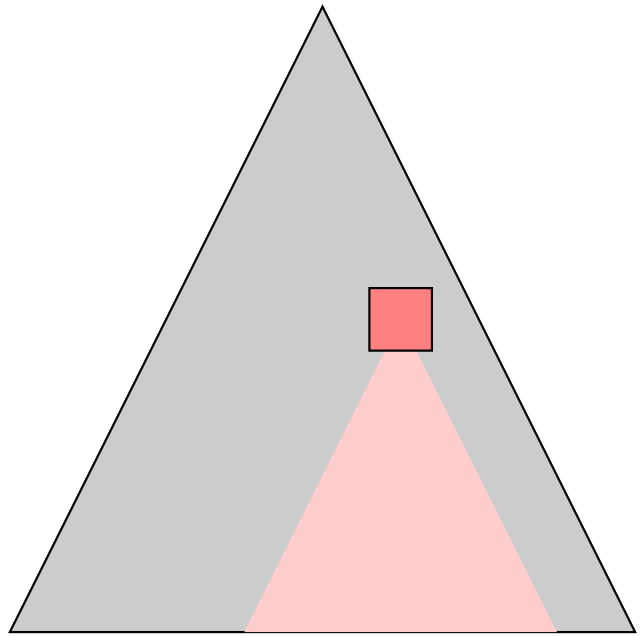
- Live data is never overwritten
- To change a block, a new block is allocated
- The pointer is updated in the same manner
- $\Longrightarrow$ Data on disk is always consistent
- Unreferenced blocks are freed at the end of the transaction

Copy on Write Transactions



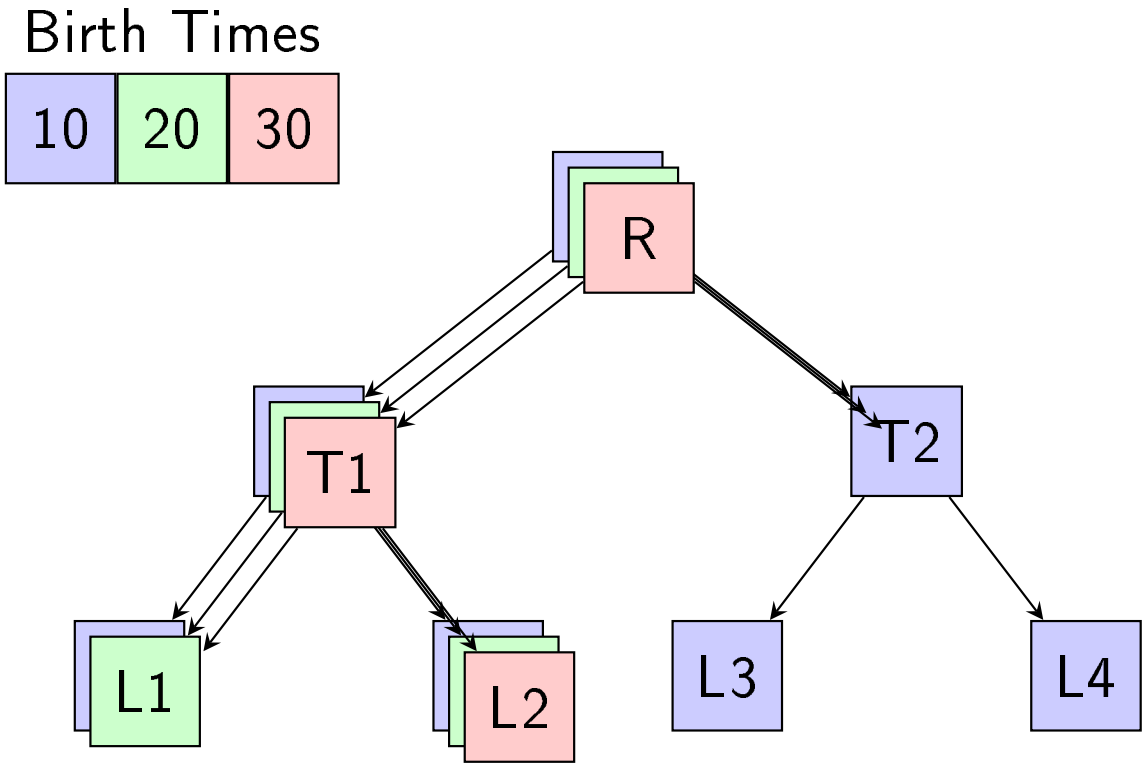
- Live data is never overwritten
- To change a block, a new block is allocated
- The pointer is updated in the same manner
- $\Longrightarrow$ Data on disk is always consistent
- Unreferenced blocks are freed at the end of the transaction

Additional Copies of Important Data



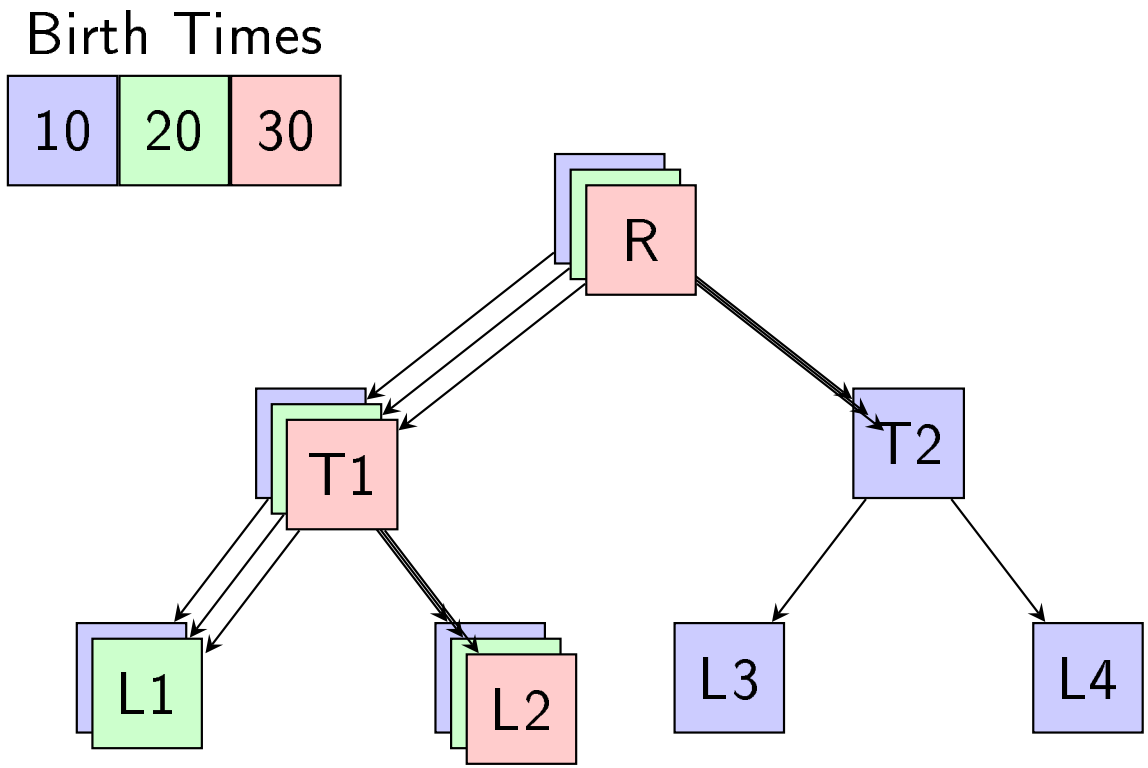
- Metadata is saved multiple times
  - FS Metadata: Twice
  - Pool Metadata: Thrice
  - "Uberblock:" Four times per physical disk
  - Metadata account for just 1% to 2% of the disk space
- Also possible to store multiple copies of normal data
  - (Good for laptops.)

Snapshots and Clones



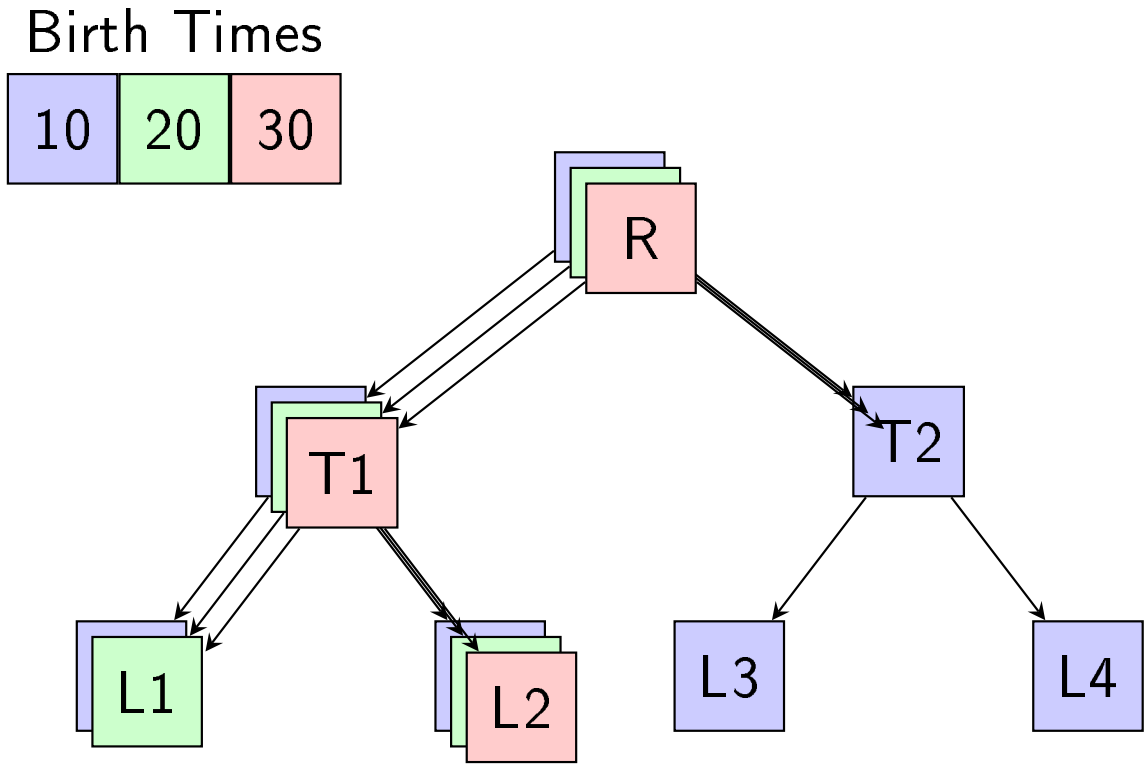
- Birth Time is the transaction number during which the block was allocated
- Blocks don't have reference counts

Snapshots and Clones



- To delete a block:
  - The birth time is compared with the most recent snapshot
  - Larger? Block is unreferenced.
  - Smaller? Block is still referenced.
    - (Added to the snapshot's so-called dead list.)

Snapshots and Clones



- To delete a snapshot:
  - Can again use the birth time
  - See `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
```

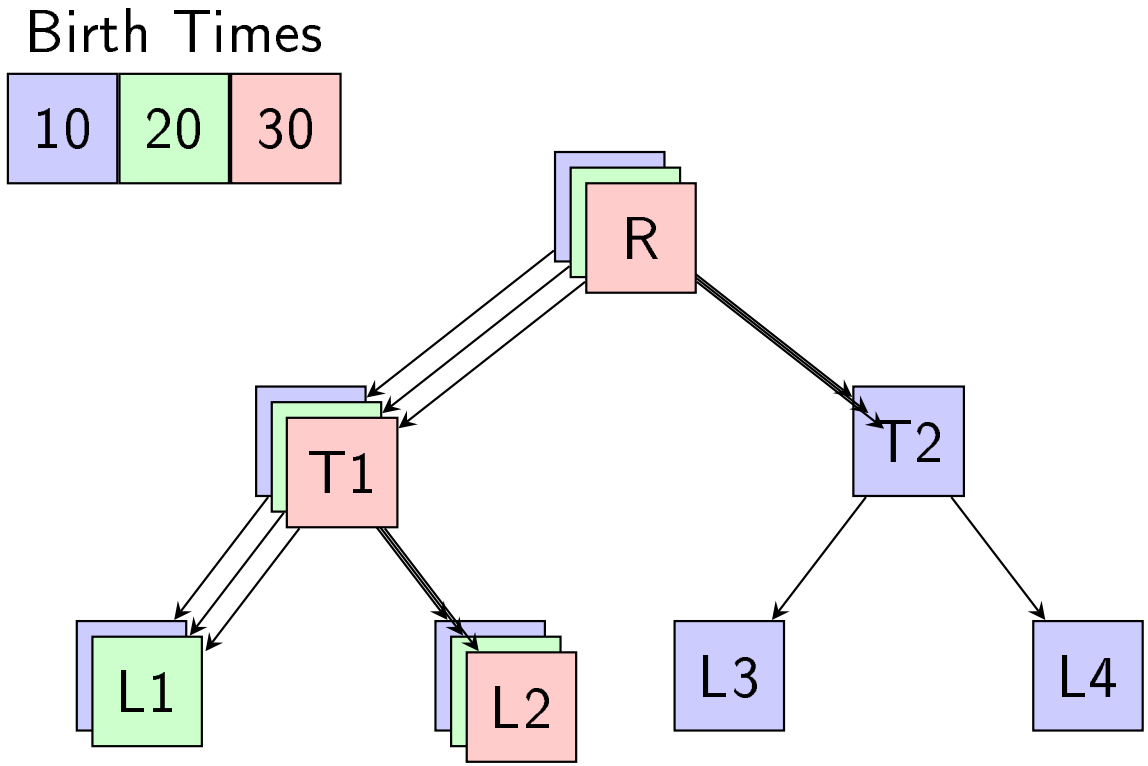
Create a snapshot

Modify data

Browse snapshot

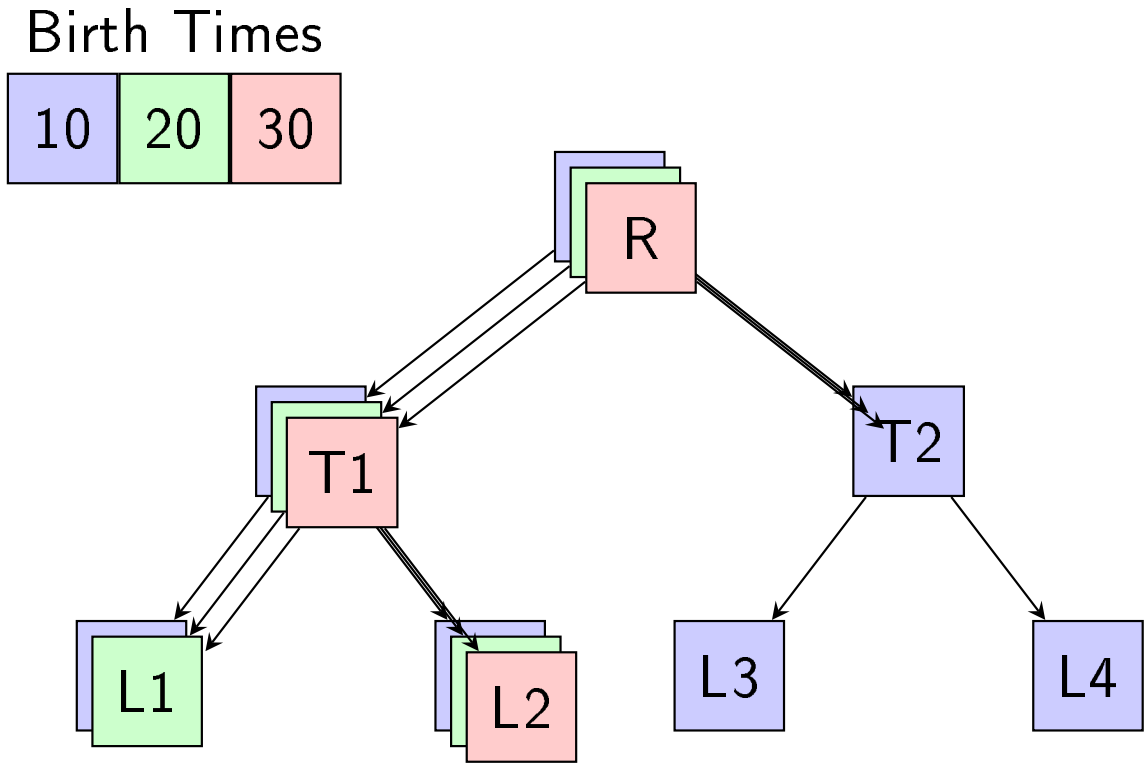
diff

Simple, Incremental Remote Replication



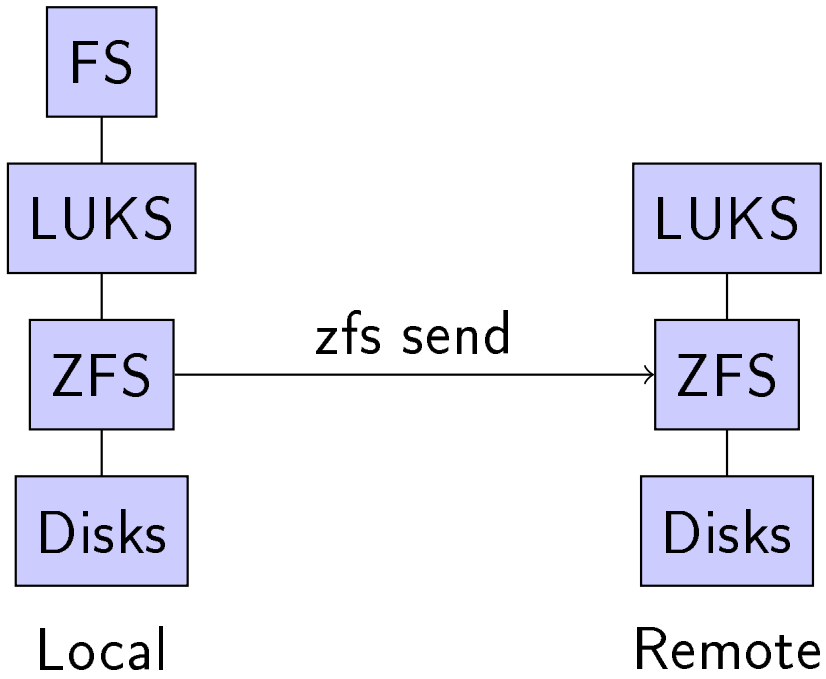
- Incremental Replication
  - `zfs send [base snapshot] snapshot`
  - `| ssh zfs receive fs`

Simple, Incremental Remote Replication



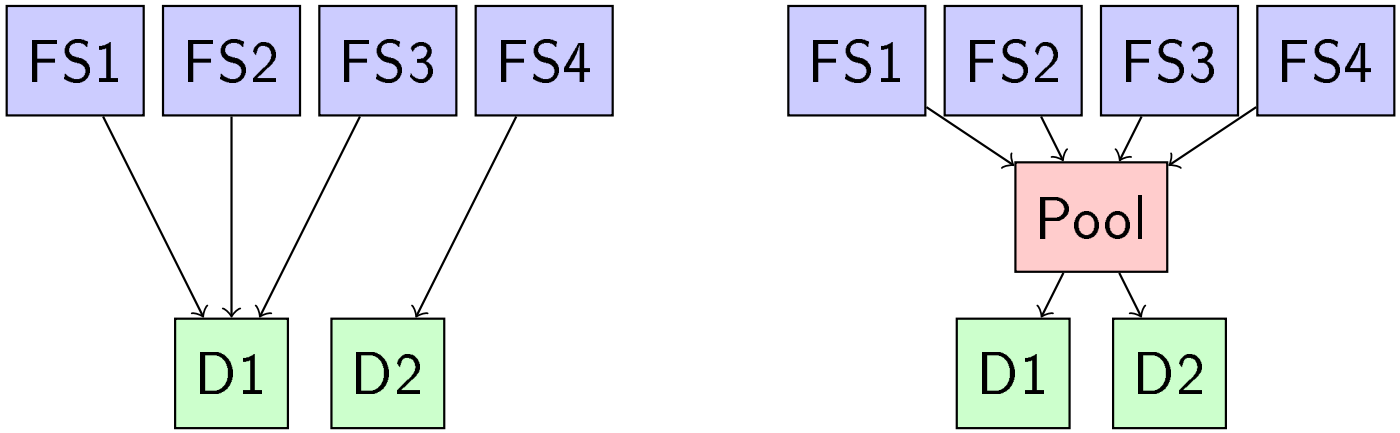
- Much faster than `rsync`
  - `rsync` needs to `stat` all files.
  - When comparing the trees, ZFS can aggressively prune subtrees
  - Replication often takes just a few seconds
    - Even with multi-TB data sets
    - $\Longrightarrow$ possible to replicate every minute

Encrypted Off Site Backups



- Yes
- Use a LUKs container in the ZFS Dataset

Simpler Administration



- Shared pool instead of many statically sized volumes
- Idea: Administer the hard drives like RAM
  - $\Longrightarrow$ mostly nothing to do
  - Occasionally impose some quotas or reservations
- Bonus: Shared IOPS!

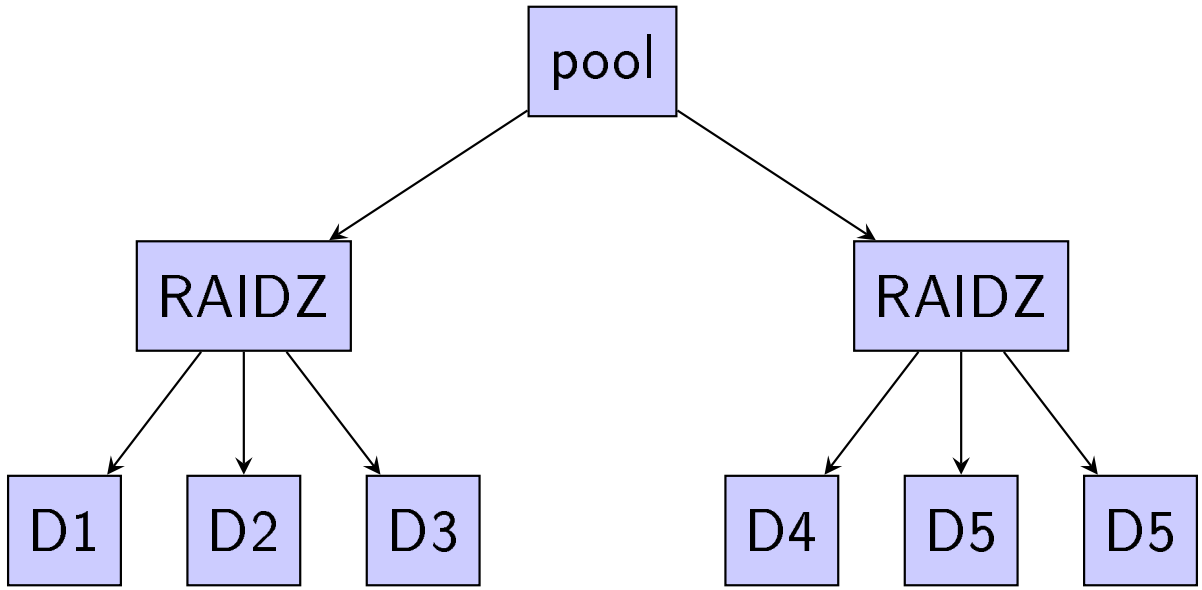
Hierarchical Storage Management (HSM)

- SSD as a Read Cache
  - Write-Through Cache
  - $\Longrightarrow$ Error can influence performance, but not correctness
  - $\Longrightarrow$ MLC SSD is sufficient
  - Automatically managed, like the page cache

Hierarchical Storage Management (HSM)

- SSD as a Write Cache
  - Absorb synchronous writes
  - Latency goes from 4-6 ms to 50 μs (factor 100)
  - 1 GB of space is sufficient
  - Very write intense!
  - $\Longrightarrow$ SLC or eMLC SSD
    - Or, over provision a huge MLC SSD
  - If the SSD dies
    - Data will only be lost if the server crashes
    - The data is still buffered in memory
    - SSD is only read after a system crash to commit transactions

ZFS Pool



- ZFS combines (stripes) hard drive groups (logical vdev)
- Groups consist of multiple drives (physical vdev)
  - Mirrors
  - RAIDs
- Easy to grow a existing pool: just add another group

Hardware Tips: ECC

- Use ECC Memory!
  - ZFS trusts that the RAM is error-free!
  - Google Study: 4k errors per year per DIMM!
    `http://www.zdnet.com/blog/storage/dram-error-rates-nightmare-on-dimm-street/638`

Hardware Tips: Diversity

Diversity inhibits correlated errors!

- Use hard drives from different manufacturers
- Connect drives from same vdev to different controllers
- etc.

Why not BTRFS?

Russel Coker: Monthly 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/

Thanks!

More information about ZFS:

- `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`