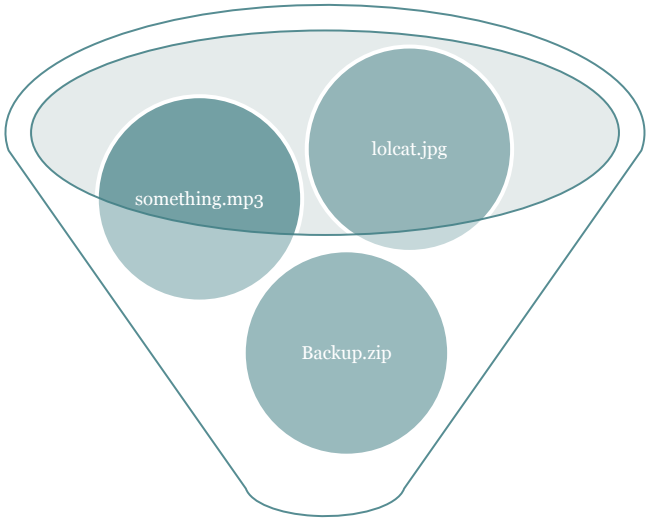
ZFS

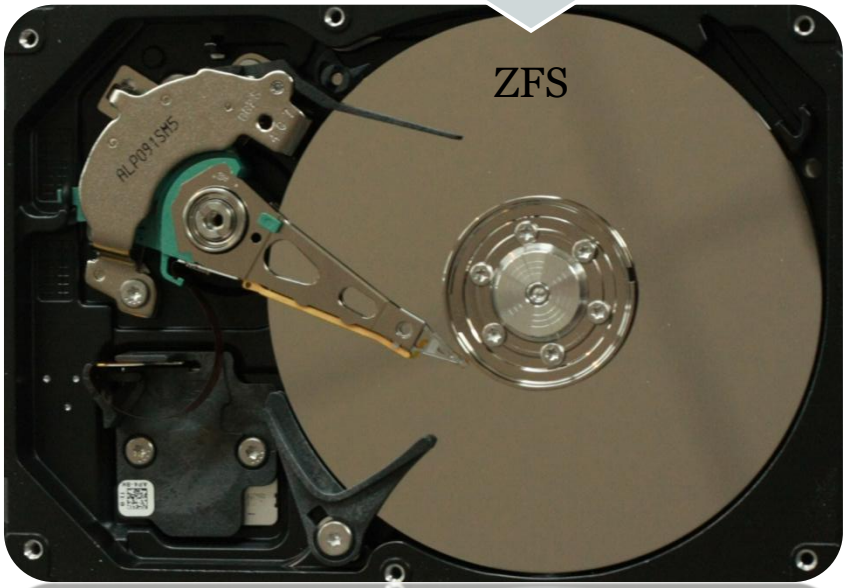
filesystem++

Now with 100% more pictures of kittens!

Marc Seeger (marc-seeger.de)

The basic idea





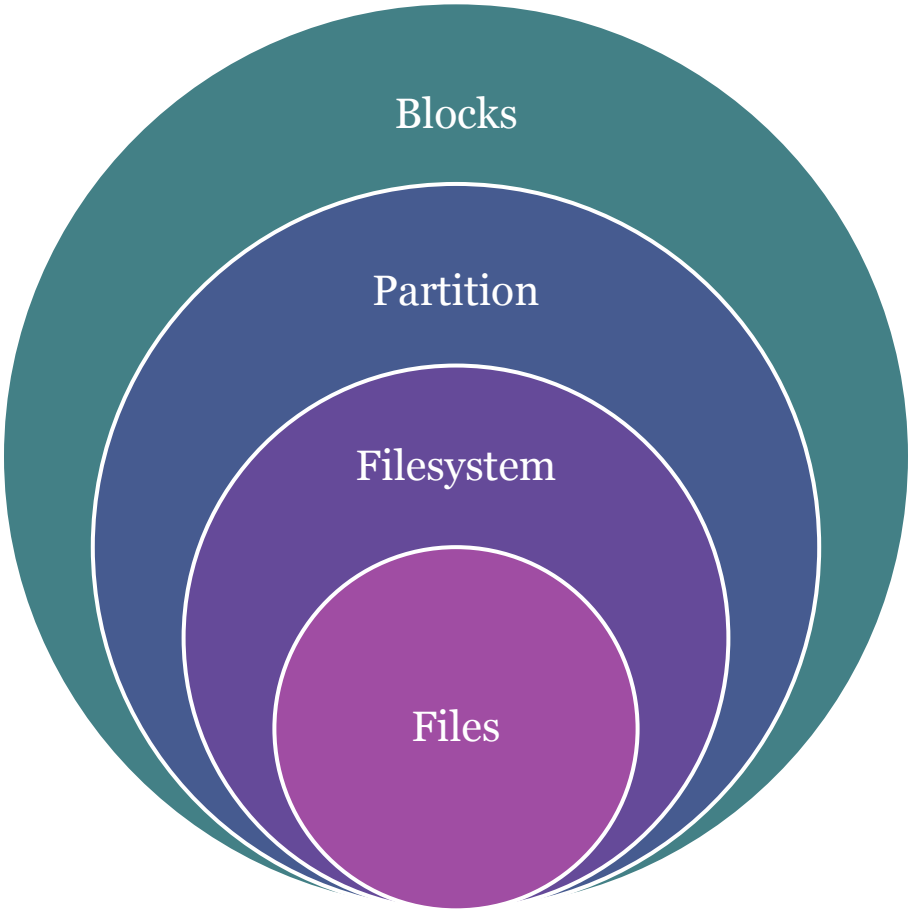
The usual features

- File names/Directories => FAT / Inodes
- Metadata => time/size/…
- CRUD => + truncate, appending, moving, links
- Security => ACL (☹)/capabilites (☺)

The good stuff

- Journaling => metadata only /complete
- Encryption
- Transparent compression
- Checksums
- Snapshots/Versioning

Layout



Logical Volume Manager (LVM)

Operating System

Logical Volume Manager

Volume
(consisting of HDDs)

Problems today

Silent data corruption

- Controller, cable, drive, firmware, ...
- CERN: Large Hadron Collider = >15.000 TB/year
  ➔„Data integrity“ paper*
- **Disk errors.** 2 GB file to > 3,000 nodes every 2 hours for 5 weeks
  => 500 errors on 100 nodes.
  - **Single bit errors.** 10% of disk errors.
  - **Sector (512 bytes) sized errors.** 10% of disk errors.
  - **64 KB regions.** 80% of disk errors. (Bug in WD disk firmware + 3Ware controller cards)
- **RAID errors.** 492 RAID systems each week for 4 weeks.
  Specs: Bit Error Rate of 10^14 read/written.
  Good news: only about 1/3 of the spec'd rate.
  Bad news: 2.4 petabytes of data => 300 errors.
- **Memory errors.**
  Good news: only 3 double-bit errors in 3 months on 1300 nodes.
  Bad news: according to the spec there shouldn't have been any. (double bit errors can't be corrected.)

➔ CERN found an overall byte error rate of 3 * 10^7

* http://indico.cern.ch/getFile.py/access?contribId=3&resId=1&materialId=paper&confId=13797

Management

- Labels, partitions, volumes, provisioning, grow/shrink, /etc files…
- Limits: filesystem/volume size, file size, number of files,
- files per directory, number of snapshots …
- Different tools to manage file, block, iSCSI, NFS, CIFS …

Slow

- Linear-time create
- fat locks
- fixed block size
- naïve prefetch
- dirty region logging
- painful RAID rebuilds
- growing backup time

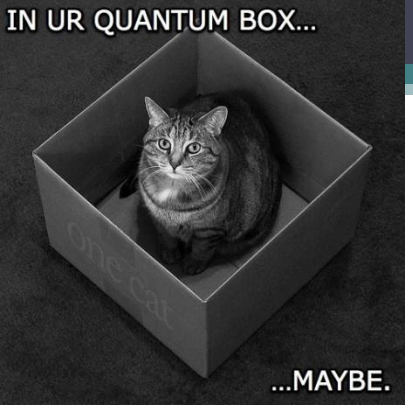
What‘s different about ZFS

“ZFS is a new kind of file system that provides:

- *simple administration*
- *transactional semantics*
- *end-to-end data integrity*
- *immense scalability.*

ZFS is not an incremental improvement to existing technology; it is a fundamentally new approach to data management.

We've blown away 20 years of obsolete assumptions, eliminated complexity at the source, and created a storage system that's actually a pleasure to use.”



pooled storage model

completely eliminates:

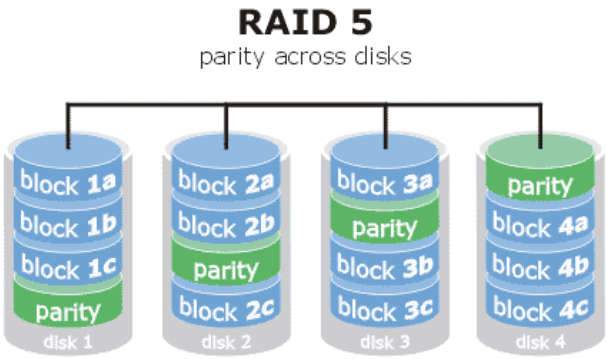
- the concept of volumes
  - and the associated problems of:
    - Partitions
    - Provisioning
    - Wasted bandwidth
    - Stranded storage.

Thousands of file systems can draw from a common storage pool, each one consuming only as much space as it actually needs. The combined I/O bandwidth of all devices in the pool is available to all filesystems at all times.

All operations are copy-on-write transactions

- ➔ the on-disk state is always valid.
- There is no need to fsck(1M) a ZFS file system, ever.
- Every block is checksummed to prevent silent data corruption (user-selectable algorithm)
- the data is self-healing in replicated (mirrored or RAID) configurations.
- If one copy is damaged, ZFS detects it and uses another copy to repair it.

RAID-Z



similar to RAID-5 but:

- uses variable stripe width to eliminate the RAID-5 write hole.

➔All RAID-Z writes are full-stripe writes.

- no read-modify-write tax
- no write hole
- no need for NVRAM in hardware.
  (ZFS loves cheap disks)

But cheap disks can fail!

- No problem: ZFS provides disk scrubbing
  (like ECC memory scrubbing)
  - 256 bit block checksum
  - works while storage pool is live and in use!

Scalability

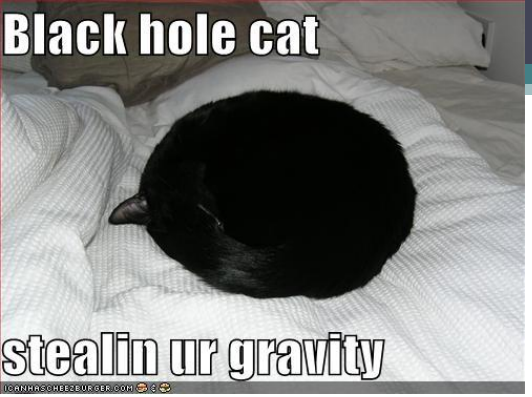
- 128-bit filesystem ➔256 quadrillion zettabytes.
- All metadata is allocated dynamically
  - → no need to pre-allocate inodes or otherwise limit the scalability of the file system when it is first created.
- Directories can have up to 2^{48} (256 trillion) entries
- No limit exists on the number of file systems …
- … or number of files that can be contained within a file system.

Snapshots

- A **snapshot** is a read-only copy of a file system or volume. Snapshots can be created quickly and easily. Initially, snapshots consume no additional space within the pool.
- Snapshots are happening at constant-time
- As data within the active dataset changes, the snapshot consumes space by continuing to reference the old data.
- Incremental backups are so efficient that they can be used for remote replication — e.g. to transmit an incremental update every 10 seconds.

Performance!

- ZFS has a pipelined I/O engine, similar in concept to CPU pipelines.
- The pipeline operates on I/O dependency graphs and provides scoreboarding, priority, deadline scheduling, out-of-order issue and I/O aggregation.
- I/O loads that bring other file systems to their knees are handled with ease by the ZFS I/O pipeline. (quote: sun)



Compression

- ZFS provides built-in compression. In addition to reducing space usage by 2-3x, compression also reduces the amount of I/O by 2-3x. For this reason, enabling compression actually makes some workloads go faster.
- In addition to file systems, ZFS storage pools can provide volumes for applications that need raw-device semantics. ZFS volumes can be used as swap devices, for example. And if you enable compression on a swap volume, you now have compressed virtual memory.