The ZFS filesystem

COSCUP 2019

Philip Paeps — Trouble 麻煩

18 August 2019

Taipei, Taiwan

FreeBSD

What is FreeBSD?

FreeBSD is an open source Unix-like **operating system** descended from patches developed at the University of California, Berkeley in the 1970s.

The FreeBSD Project is an active **open source community** since 1993 with hundreds of committers and thousands of contributors around the world.

The FreeBSD Foundation is a **non-profit organisation** registered in Colorado, USA in 2001 dedicated to supporting the FreeBSD Project, its development and its community.

Who uses FreeBSD?

Where FreeBSD excels

Community

- Friendly and professional
- Many active contributors and committers for 10+ and even 20+ years (and longer)

Mentoring

- Built into the Project's culture and processes

Documentation

- FreeBSD Handbook, FAQ, Developers' Handbook, Porters' Handbook, Unix manual pages

Licence

- 2-clause BSD licence
- Does not restrict what you can do with your own code!

History of ZFS

- 2001: Development started at Sun (now Oracle)
- 2005: ZFS source code released
- 2008: ZFS released in FreeBSD 7.0
- (2019: ZFS still doesn’t work reliably on Linux)

ZFS in a nutshell

End-to-end data integrity

- Detects and corrects silent data corruption

Transactional design

- Data always consistent
- Huge performance wins

Pooled storage

- The first 128 bit filesystem
- Eliminates the antique notion of volumes

Simple administration

- Two commands to manage entire storage configuration

End-to-end data integrity

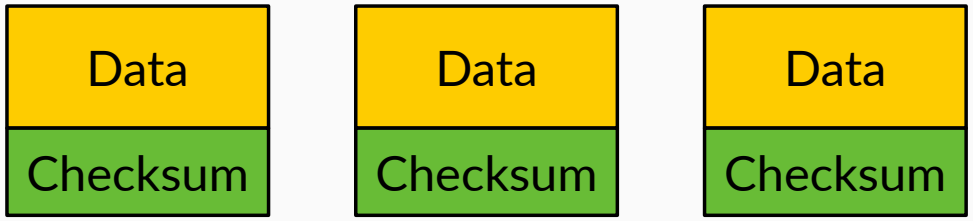
- Disks
- Controllers
- Cables
- Firmware
- Device drivers
- Non-ECC memory

FreeBSD

Disk block checksums

- Checksums are stored with the data blocks
- Any self-consistent block will have a correct checksum
- Can't even detect stray writes
- Inherently limited to single filesystems or volumes

Disk block checksums only validate media



✓Bit rot

✗Phantom writes

✗Misdirected reads and writes

✗DMA parity errors

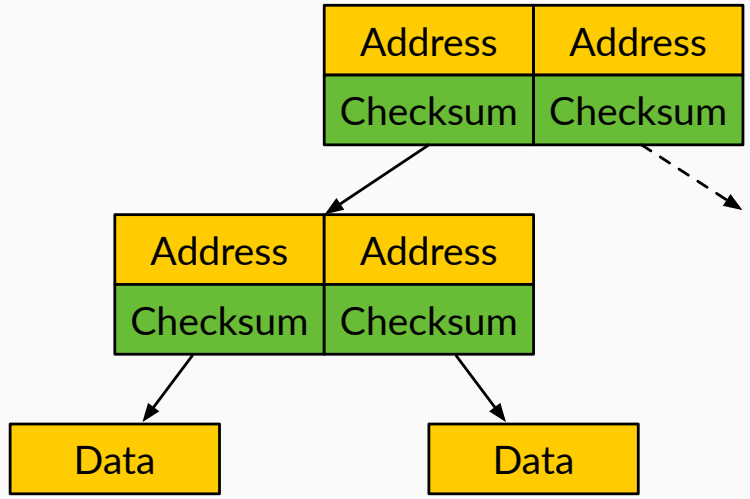
✗Driver bugs

✗Accidental overwrite

FreeBSD

ZFS data authentication

- Checksums are stored in parent block pointers
- Fault isolation between data and checksum
- Entire storage pool is a self-validating Merkle tree



ZFS data authentication validates entire I/O path

✓Bit rot

✓Phantom writes

✓Misdirected reads and writes

✓DMA parity errors

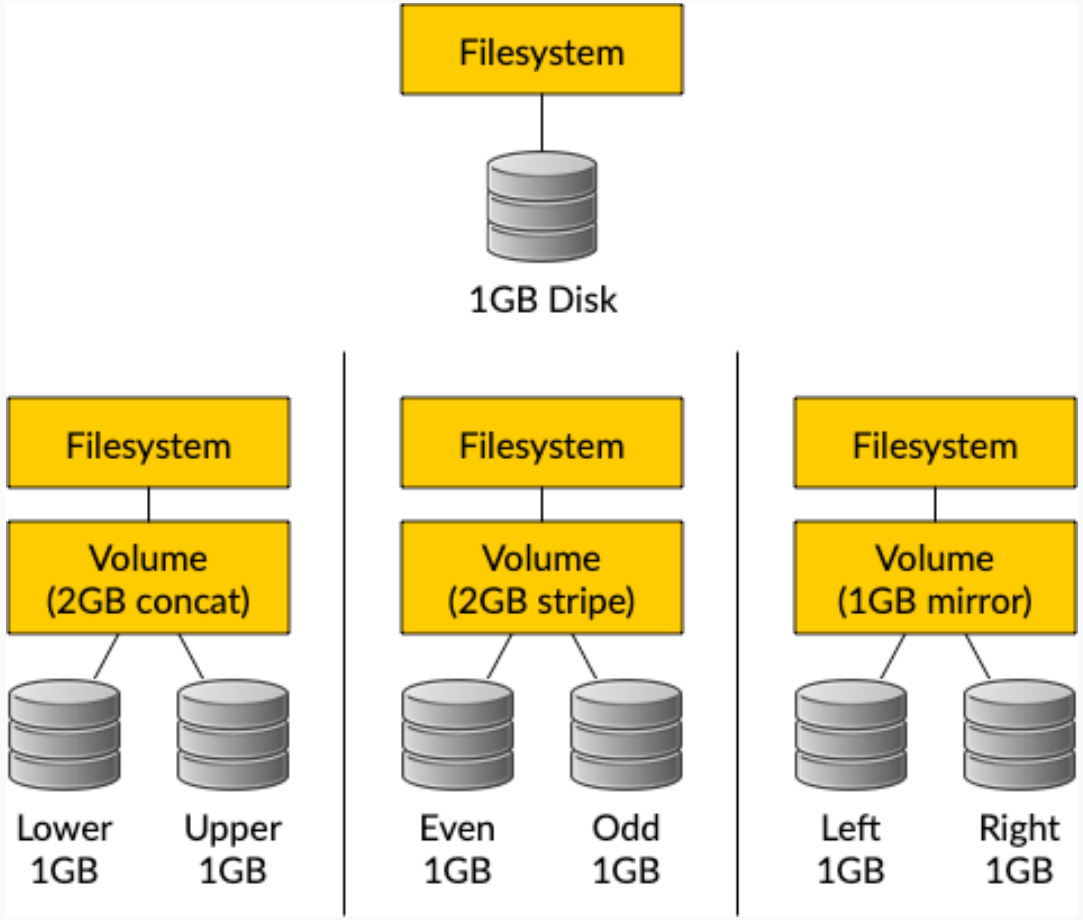
✓Driver bugs

✓Accidental overwrite

FreeBSD

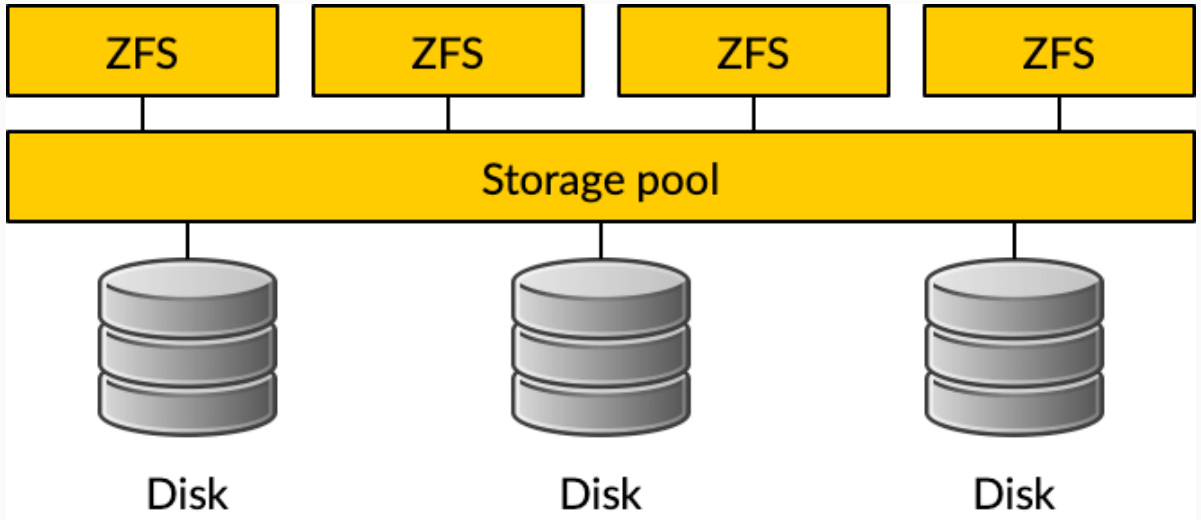
Traditional storage architecture

- Single partition or volume per filesystem
- Each filesystem has limited I/O bandwidth
- Filesystems must be manually resized
- Storage is fragmented



ZFS pooled storage

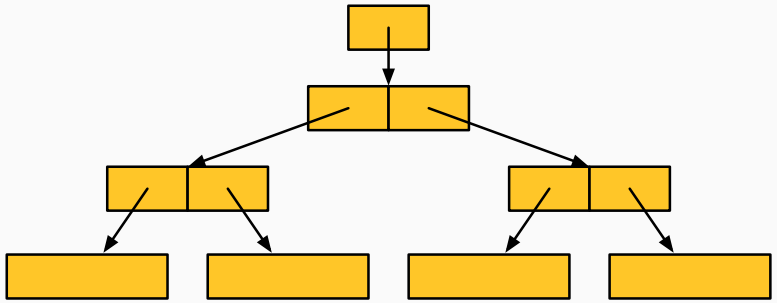
- No partitions required
- Storage pool grows automatically
- All I/O bandwidth is always available
- All storage in the pool is shared



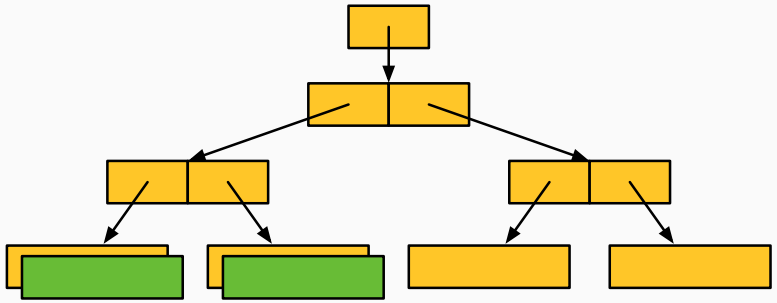
FreeBSD

Copy-on-write transactions

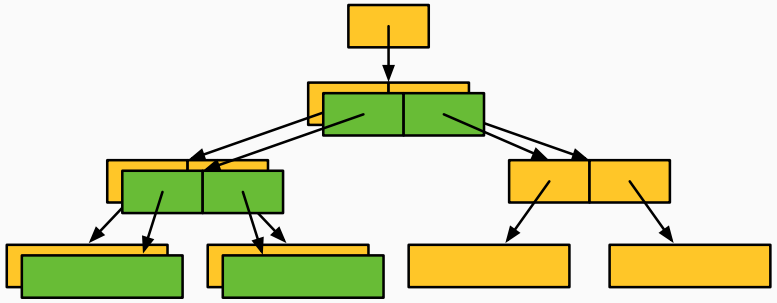
1. Initial consistent state



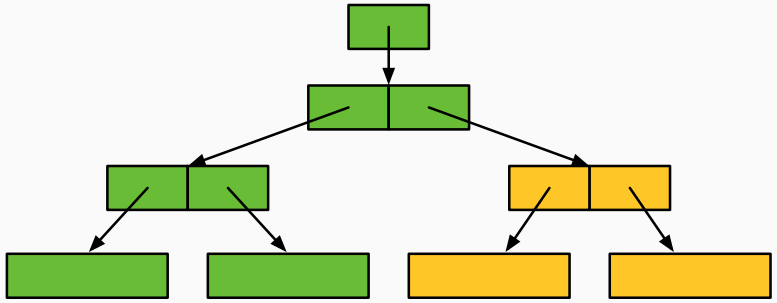
2. COW some blocks



3. COW indirect blocks



4. Rewrite uberblock (atomic)



Simple administration

Only two commands:

1. Storage pools: `zpool`
   - Add and replace disks
   - Resize pools

2. Filesystems: `zfs`
   - Quotas, reservations, etc.
   - Compression and deduplication
   - Snapshots and clones
   - atime, readonly, etc.

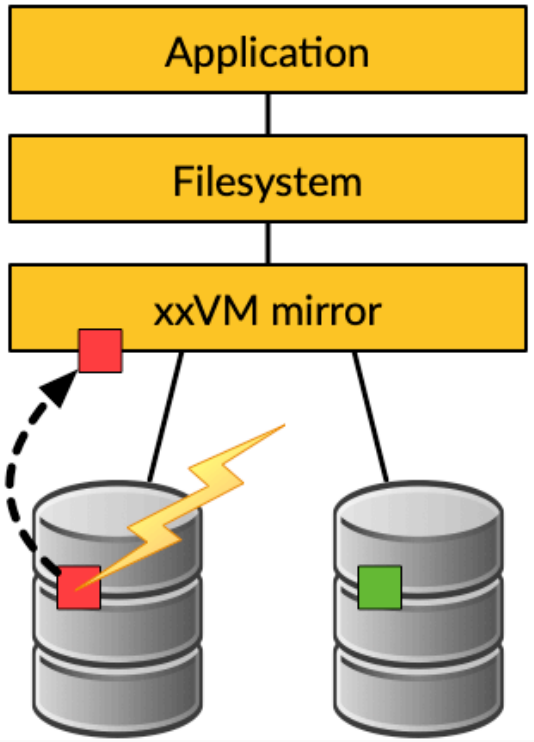
FreeBSD

Self-healing data

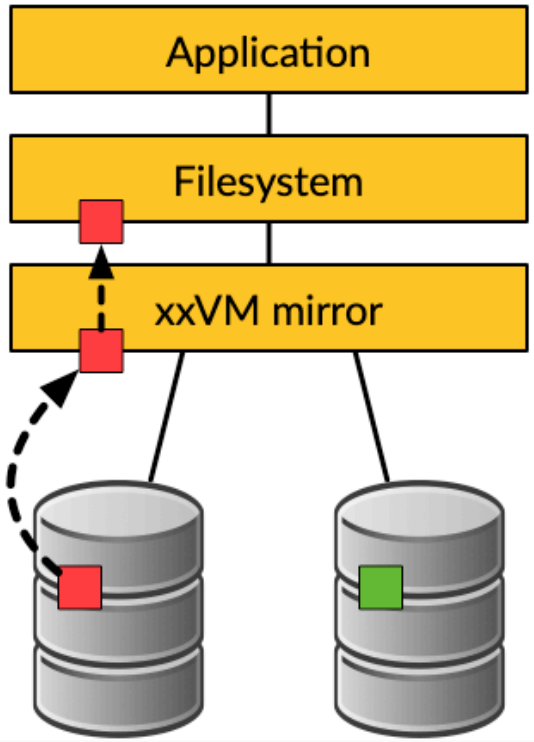
Demo

Traditional mirroring

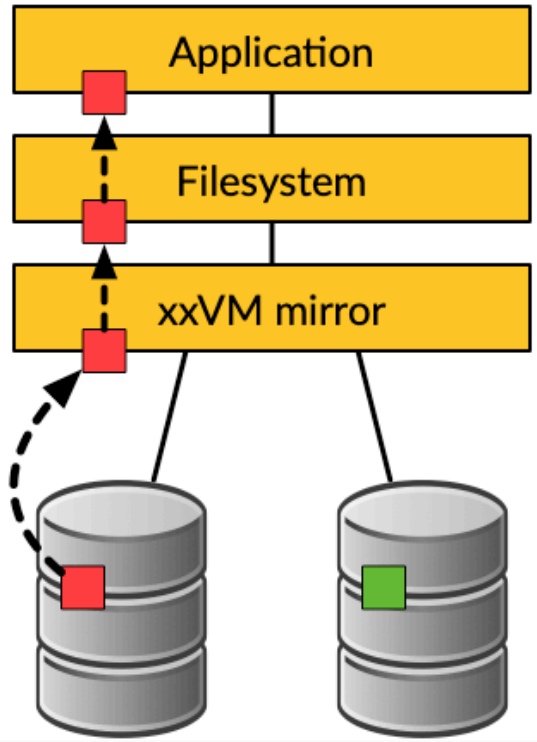
1. Application issues a read. Mirror reads the first disk, which has a corrupt block. It can't tell.



2. Volume manager passes bad block up to filesystem. If it's a metadata block, the filesystem panics. If not...



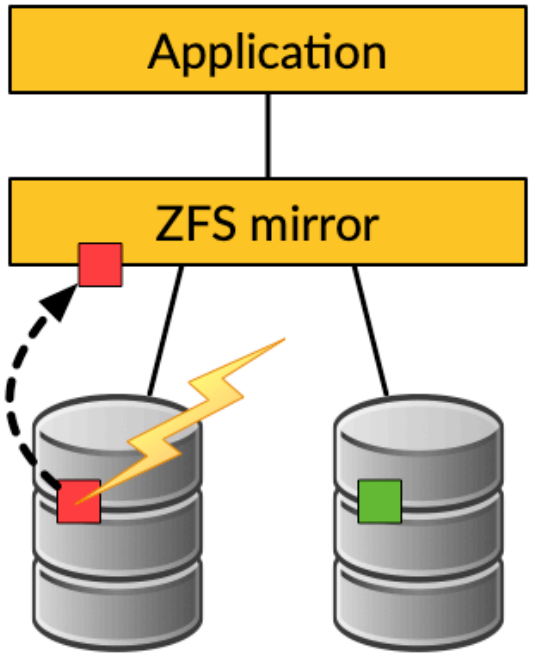
3. Filesystem returns bad data to the application.



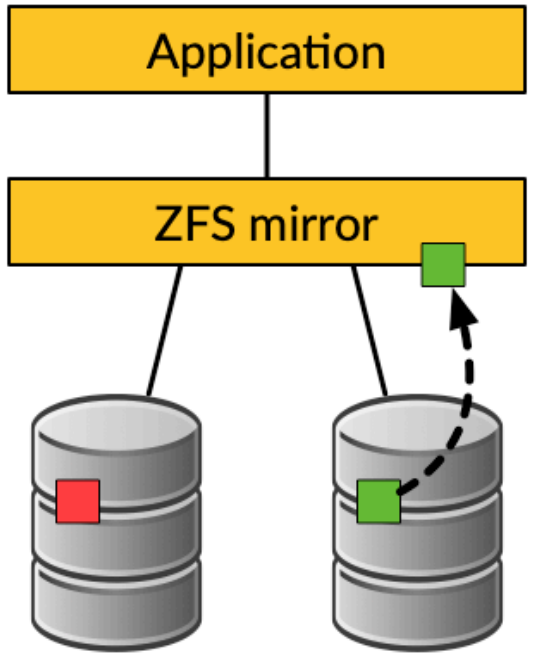
FreeBSD

Self-healing data in ZFS

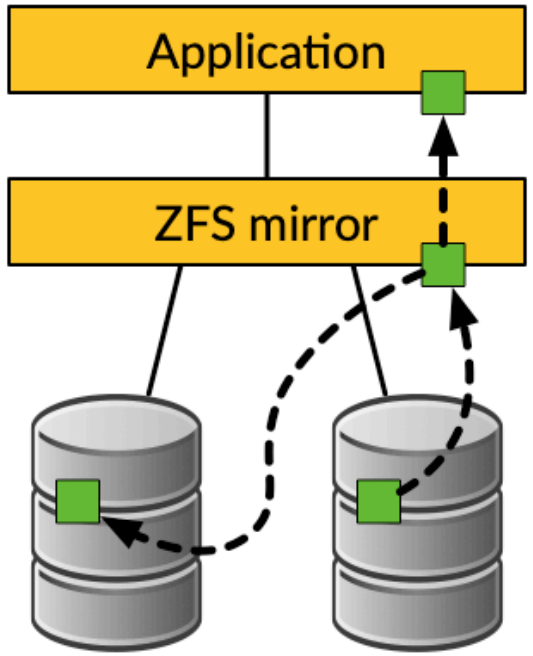
1. Application issues a read. ZFS mirror tries the first disk. Checksum reveals that the block is corrupt on disk.



2. ZFS tries the second disk. Checksum indicates that the block is good.



3. ZFS returns good data to the application **and repairs the damaged block** on the first disk.



FreeBSD

Self-healing data demo

Store some important data (1/2)

- We have created a redundant pool with two mirrored disks and stored some important data on it
- We will be very sad if the data gets lost! :-(

```
# zfs list tank
NAME   USED  AVAIL  REFER  MOUNTPOINT
tank    74K   984G    23K  /tank

# cp -a /some/important/data/ /tank/

# zfs list tank
NAME   USED  AVAIL  REFER  MOUNTPOINT
tank  3.23G   981G  3.23G  /tank
```

Self-healing data demo

Store some important data (2/2)

```
# zpool status tank
  pool: tank
 state: ONLINE
  scan: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            md0     ONLINE       0     0     0
            md1     ONLINE       0     0     0

errors: No known data errors

# zpool list tank
NAME   SIZE  ALLOC   FREE  CKPOINT  EXPANDSZ   FRAG    CAP  DEDUP  HEALTH  ALTROOT
tank  1016G  3.51G  1012G        -         -     0%     0%  1.00x  ONLINE  -
```

FreeBSD

Self-healing data demo

Destroy one of the disks (1/2)

Caution!

This example can destroy data when used on the wrong device or a non-ZFS filesystem!

Always check your backups!

```
# zpool export tank

# dd if=/dev/random of=/dev/md1 bs=1m count=200

# zpool import tank
```

Self-healing data demo

Destroy one of the disks (2/2)

```
# zpool status tank
  pool: tank
 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://illumos.org/msg/ZFS-8000-9P
  scan: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            md0     ONLINE       0     0     5
            md1     ONLINE       0     0     0

errors: No known data errors
```

Self-healing data demo

Make sure everything is okay (1/3)

```
# zpool scrub tank
# zpool status tank
  pool: tank
 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://illumos.org/msg/ZFS-8000-9P
  scan: scrub in progress since Fri Oct 12 22:57:36 2018
        191M scanned out of 3.51G at 23.9M/s, 0h2m to go
        186M repaired, 5.32% done
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            md0     ONLINE       0     0 1.49K  (repairing)
            md1     ONLINE       0     0     0

errors: No known data errors
```

Self-healing data demo

Make sure everything is okay (2/3)

```
# zpool status tank
  pool: tank
 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://illumos.org/msg/ZFS-8000-9P
  scan: scrub repaired 196M in 0h0m with 0 errors on Fri Oct 12 22:58:14 2018
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            md0     ONLINE       0     0 1.54K
            md1     ONLINE       0     0     0

errors: No known data errors
```

Self-healing data demo

Make sure everything is okay (3/3)

```
# zpool clear tank

# zpool status tank
  pool: tank
 state: ONLINE
  scan: scrub repaired 196M in 0h0m with 0 errors on Fri Oct 12 22:58:14 2018
config:

        NAME        STATE     READ WRITE CKSUM
        tank        ONLINE       0     0     0
          mirror-0  ONLINE       0     0     0
            md0     ONLINE       0     0     0
            md1     ONLINE       0     0     0

errors: No known data errors
```

Self-healing data demo

But what if it goes very wrong? (1/2)

```
# zpool status
  pool: tank
 state: ONLINE
status: One or more devices has experienced an error resulting in data
        corruption.  Applications may be affected.
action: Restore the file in question if possible.  Otherwise restore the
        entire pool from backup.
   see: http://illumos.org/msg/ZFS-8000-8A
  scan: scrub in progress since Fri Oct 12 22:46:01 2018
        498M scanned out of 3.51G at 99.6M/s, 0h0m to go
        19K repaired, 13.87% done
config:

        NAME         STATE     READ WRITE CKSUM
        tank         ONLINE       0     0 1.48K
          mirror-0   ONLINE       0     0 2.97K
            md0      ONLINE       0     0 2.97K
            md1      ONLINE       0     0 2.97K

errors: 1515 data errors, use '-v' for a list
```

Self-healing data demo

But what if it goes very wrong? (2/2)

```
# zpool status -v
  pool: tank
 state: ONLINE
status: One or more devices has experienced an error resulting in data
        corruption.  Applications may be affected.
action: Restore the file in question if possible.  Otherwise restore the
        entire pool from backup.
   see: http://illumos.org/msg/ZFS-8000-8A
  scan: scrub repaired 19K in 0h0m with 1568 errors on Fri Oct 12 22:46:25 2018
config:

        NAME         STATE     READ WRITE CKSUM
        tank         ONLINE       0     0 1.53K
          mirror-0   ONLINE       0     0 3.07K
            md0      ONLINE       0     0 3.07K
            md1      ONLINE       0     0 3.07K

errors: Permanent errors have been detected in the following files:

        /tank/FreeBSD-11.2-RELEASE-amd64.vhd.xz
        /tank/base-amd64.txz
        /tank/FreeBSD-11.2-RELEASE-amd64-disc1.iso.xz
        /tank/intro_slides.pdf
```

The Evolution of ZFS

- ZFS was originally developed at Sun Microsystems starting in 2001, and open sourced under the CDDL license in 2005
- Oracle bought Sun in 2010, and close sourced further work
- illumos, a fork of the last open source version of Solaris became the new upstream for work on ZFS
- ZFS was ported to many platforms, including FreeBSD in 2007 and Linux in 2008. The OpenZFS project was founded to coordinate development across platforms.

OpenZFS

- The original plan for OpenZFS was a single common repository where the OS independent code would live and be tested
- Each OS would sync with this repo and add their own glue
- However, the effort required to maintain a repo that would not be directly used by any of the consumers was not viable
- The “repo of record” became a fork of illumos
- FreeBSD tracked very closely
- Linux spent a great deal of effort getting caught up

Platforms

- OpenZFS is now available on almost every platform
  - illumos (OmniOS, OpenIndiana, SmartOS, DilOS, Tribblix)
  - FreeBSD (FreeNAS, XigmaNAS, TrueOS, pfSense, etc)
  - Linux (ZFS-on-Linux, Ubuntu, Gentoo, OviOS)
  - Mac OS X (ZFS-on-OSX, GreenBytes/ZEVO, Akitio, MacZFS)
  - Windows (https://openzfsonwindows.org/)
  - NetBSD

Platforms

- OpenZFS is now available on almost every platform
  - Illumos (OmniOS, OpenIndiana, SmartOS, DilOS, Tribblix)
  - FreeBSD (FreeNAS, XigmaOS, TrueOS, pfSense, and more)
  - macOS (ZFS-on-OSX, GreenBytes/ZEVO, Akitio, MacZFS)
  - Windows (ZFS on Windows)
  - NetBSD

  - And even Linux

Divergence

- Each different platform's version of ZFS started to diverge
- OpenZFS replaced the old "pool version number" with "Feature Flags", since features would land in different orders
- Bugs were fixed in one repo and not necessarily upstreamed or communicated to other platform's could apply the same fix
- Each camp did their development within their own community, and other communities might not be aware of duplicate efforts, etc.

And Linux?

- Greg Kroah-Hartman followed up on the mailing list with:
  - "Sorry, no, we do not keep symbols exported for no in-kernel users."
  - "my tolerance for ZFS is pretty non-existant."
- Longtime Linux kernel developer Christoph Hellwig also suggested users switch to FreeBSD instead if they care about ZFS.

OpenZFS developer summit

- The new OpenZFS project organized a conference in November 2013 to have developers from the various platforms share their work and future ideas and find solutions
- Included a platform panel (FreeBSD, Illumos, MacOS, Linux) and vendor lightning talks
- Attended by over 30 developers, since grown to over 100
- Now includes a hackathon to work on prototypes while experts are in the room for advice / design discussions

Leadership meeting

- At the OpenZFS Developer Summit 2018 a discussion between the various platform leaders lead to the formation of a monthly video conference to discuss ongoing issues
- Meeting once a month instead of once a year provides more information exchange and faster response times
- Goal is to keep the platforms better in-sync and compatible
- Open to anyone. Live streamed and recorded to YouTube

Outcomes

- The leadership meetings have been very successful
- OpenZFS is working to standardize the command line interface where it has diverged across platforms
- New features are discussed during the design phase and platform specific issues are resolved early, with better results
- More effort into effective naming of tunables (ashift is an internal implementation detail, the user tunable should be called sectorsize and be expressed in bytes)

Get involved!

- The OpenZFS community is very active and very welcoming
- Watch some of the past “OpenZFS Leadership Meeting” conference calls on youtube to see for yourself
- The “repo of record” is transitioning to the OpenZFS (formerly “ZFS on Linux”) repo as it has the most active development and the most code that still needs to be pulled into other platforms
- Github Issues and Pull requests
- Mailing Lists (Topic Box) for discussions

Credits

- ZFS: The last word in filesystems
  Jeff Bonwick and Bill Moore
  URL:
  https://wiki.illumos.org/download/attachments/1146951/zfs_last.pdf

- Introduction to the ZFS filesystem
  Benedict Reuschling
  URL: [offline]