

Oracle Database on Solaris ZFS Done Right

Marcel Hofstetter

hofstetter@jomasoft.ch
<https://jomasoftmarcel.blogspot.ch>

CEO / Enterprise Consultant
JomaSoft GmbH



Agenda

- About JomaSoft
- Solaris ZFS
- ZFS Fragmentation
- ZFS Cache
- Oracle DB Performance

About JomaSoft

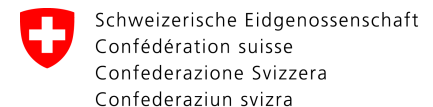
- Engineering company founded in July 2000
- **Solaris Full Service:** SPARC server, software development, consulting, ops, troubleshooting
- Product **VDCF** (Virtual Datacenter Cloud Framework)
Installation, Management, Monitoring, Security/Compliance, Hardening and DR on Solaris 10/11, Virtualize using LDoms and Solaris Zones
- VDCF is used in production since 2006



Specialized
Oracle Solaris 11



Specialized
SPARC T-Series Servers



Eidgenössisches Finanzdepartement EFD
**Bundesamt für Informatik
und Telekommunikation BIT**

Marcel Hofstetter

Working in IT since 30+ years
Solaris since 26 years
CEO at JomaSoft GmbH since 23 years



International Speaker:
Oracle OpenWorld, DOAG, UKOUG, SOUG, AOUG, FRAOSUG



SOUG (Swiss Oracle User Group) – Speaker of the Year 2016

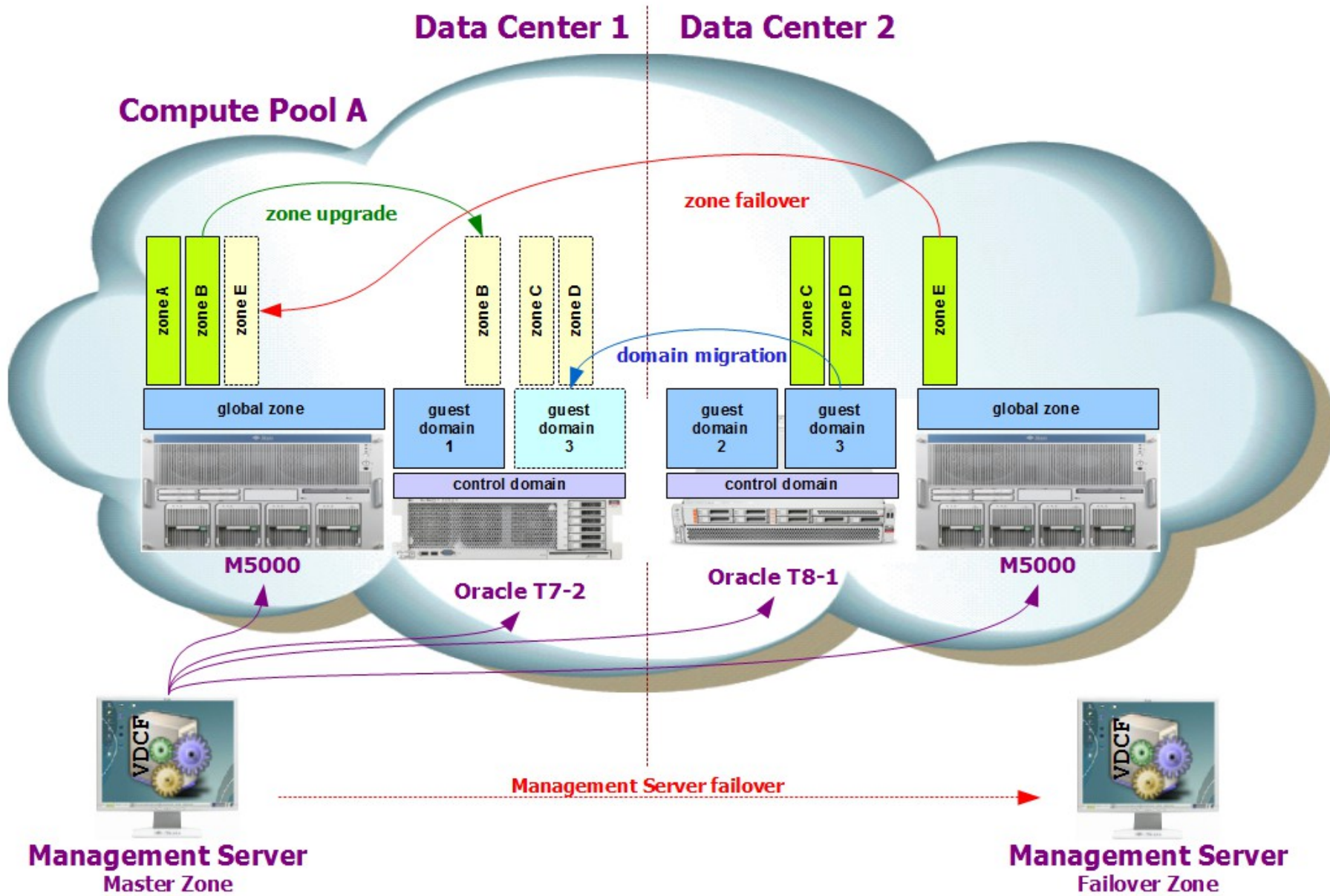
Hobbies: Family, Travel, Wine & Dine, Movies

 <https://www.linkedin.com/in/marcelhofstetter>

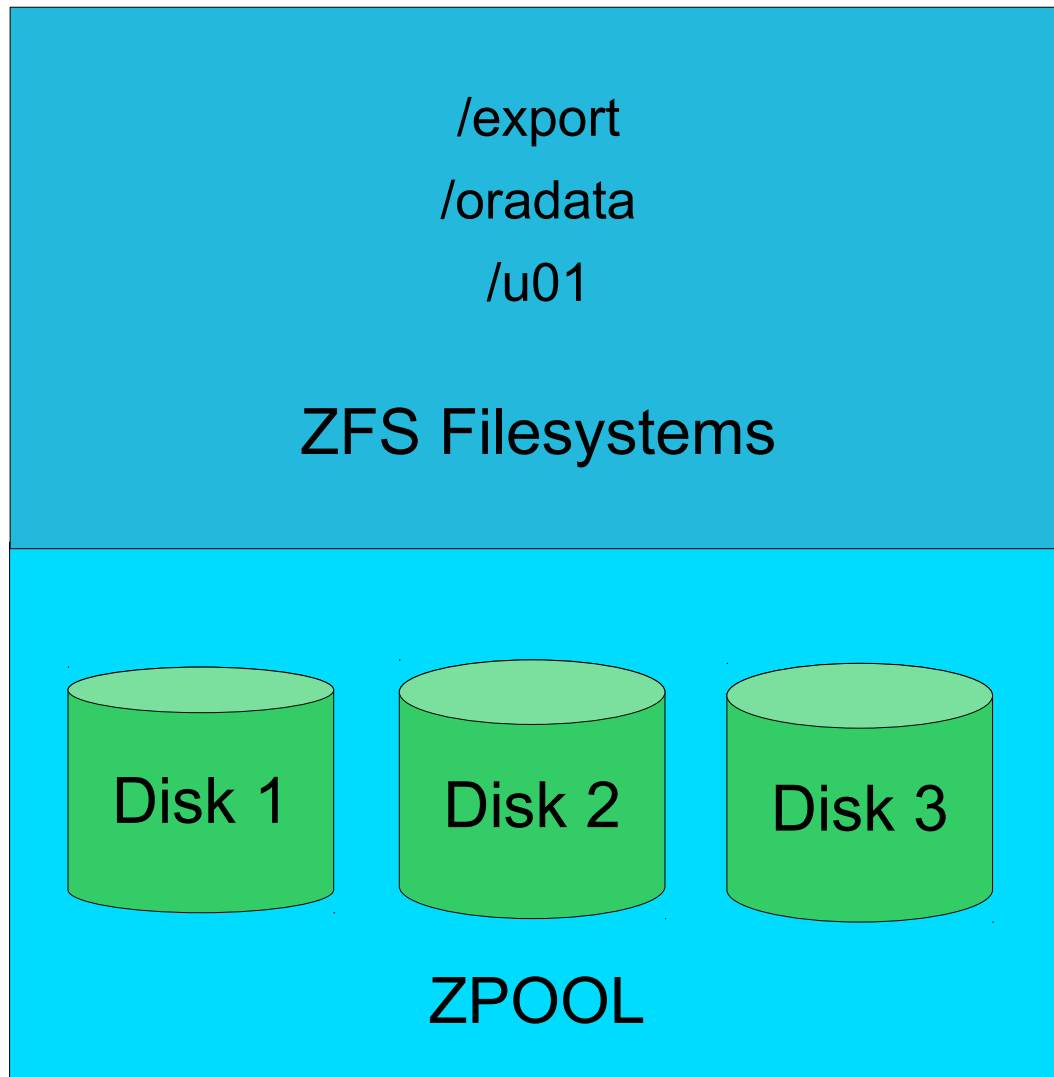
 https://twitter.com/marcel_jomasoft

 <https://jomasoftmarcel.blogspot.ch>

VDCF – Automation for Solaris



Solaris ZFS



- Easy to use
- Flexible
- Filesystem size is optional
- Snapshots & Clones
- COW / No fscheck

- Stripe, Mirror, RAID
- Add disks to grow
- Export & Import

- Remove disk since Solaris 11.4



Solaris ZFS by example

```
# zpool create mypool c2d1 c2d2 c2d3

# zfs create -o mountpoint=/u01 mypool/u01

# zfs get used,available mypool

NAME          PROPERTY      VALUE      SOURCE
mypool        used          183G      -
mypool        available    111G      -

# zfs set compress=lz4 mypool/u01
# zfs set recordsize=32k mypool/u01

# zfs snapshot mypool/u01@mysave_before_test

# zpool history mypool

History for 'mypool':
2012-05-15.20:21:13 zpool create mypool c2d1 c2d2 c2d3
2012-05-15.20:21:16 zfs create -o mountpoint=/u01 mypool/u01
2016-03-31.18:11:59 zpool add mypool c2d20
```

Solaris ZFS – LZ4 Compression

```
$ zfs get compression,compressratio,used uncompressed/fs
```

| NAME | PROPERTY | VALUE | SOURCE |
|-----------------|---------------|------------|-----------------------------|
| uncompressed/fs | compression | off | inherited from uncompressed |
| uncompressed/fs | compressratio | 1.00x | - |
| uncompressed/fs | used | 390M | - |

```
$ zfs get compression,compressratio,used compressed/fs
```

| NAME | PROPERTY | VALUE | SOURCE |
|---------------|---------------|--------------|---------------------------|
| compressed/fs | compression | lz4 | inherited from compressed |
| compressed/fs | compressratio | 6.37x | - |
| compressed/fs | used | 61.4M | - |

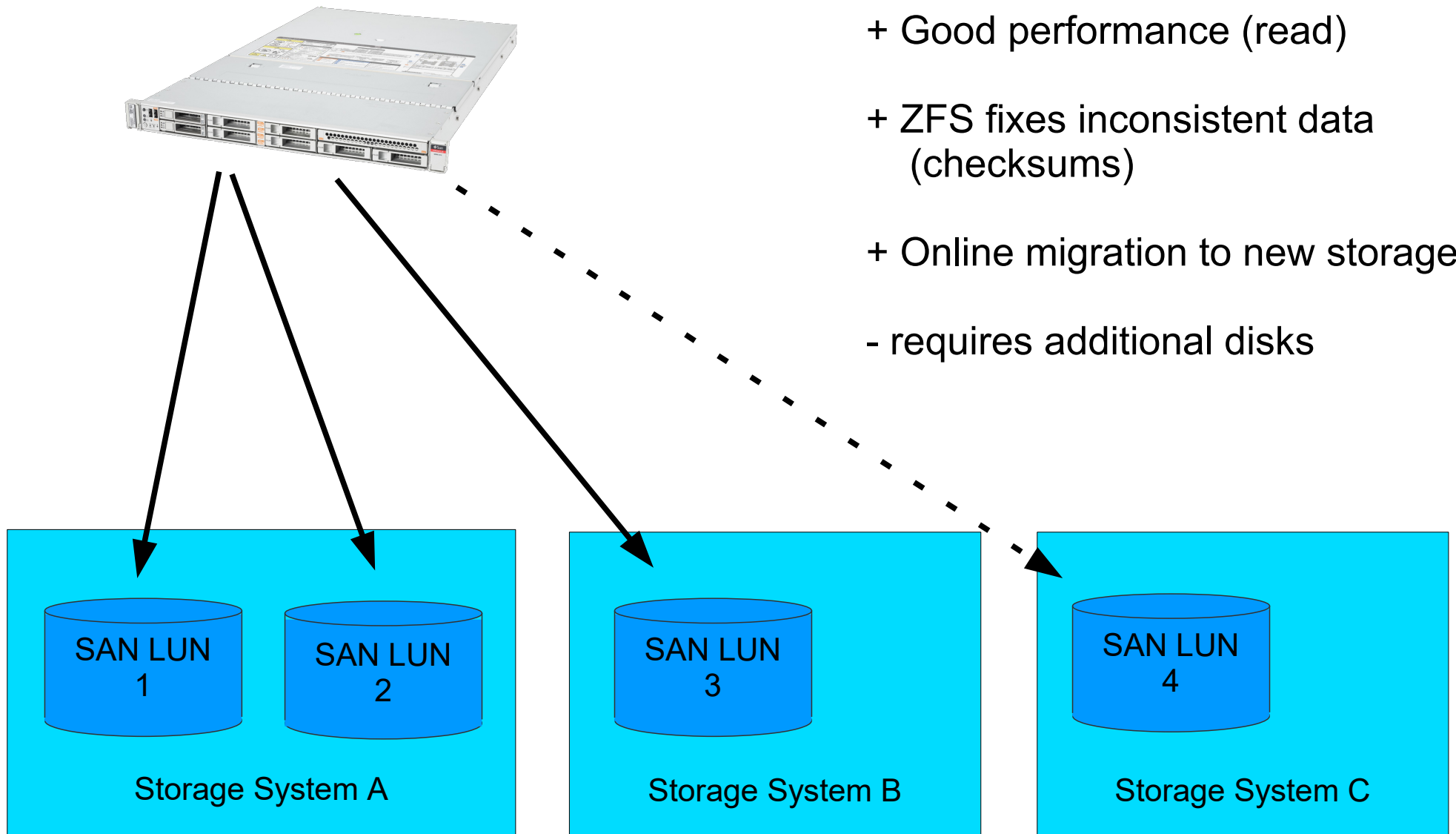
Read Performance

```
# time cp /compressed/framework.log /tmp; time cp /uncompressed/framework.log /tmp
real    0m17.415s
real    0m24.479s
```

Better results from compressed filesystem. CPU decompression is faster than doing I/O. Need to read 6x the data from uncompressed zfs filesystem.

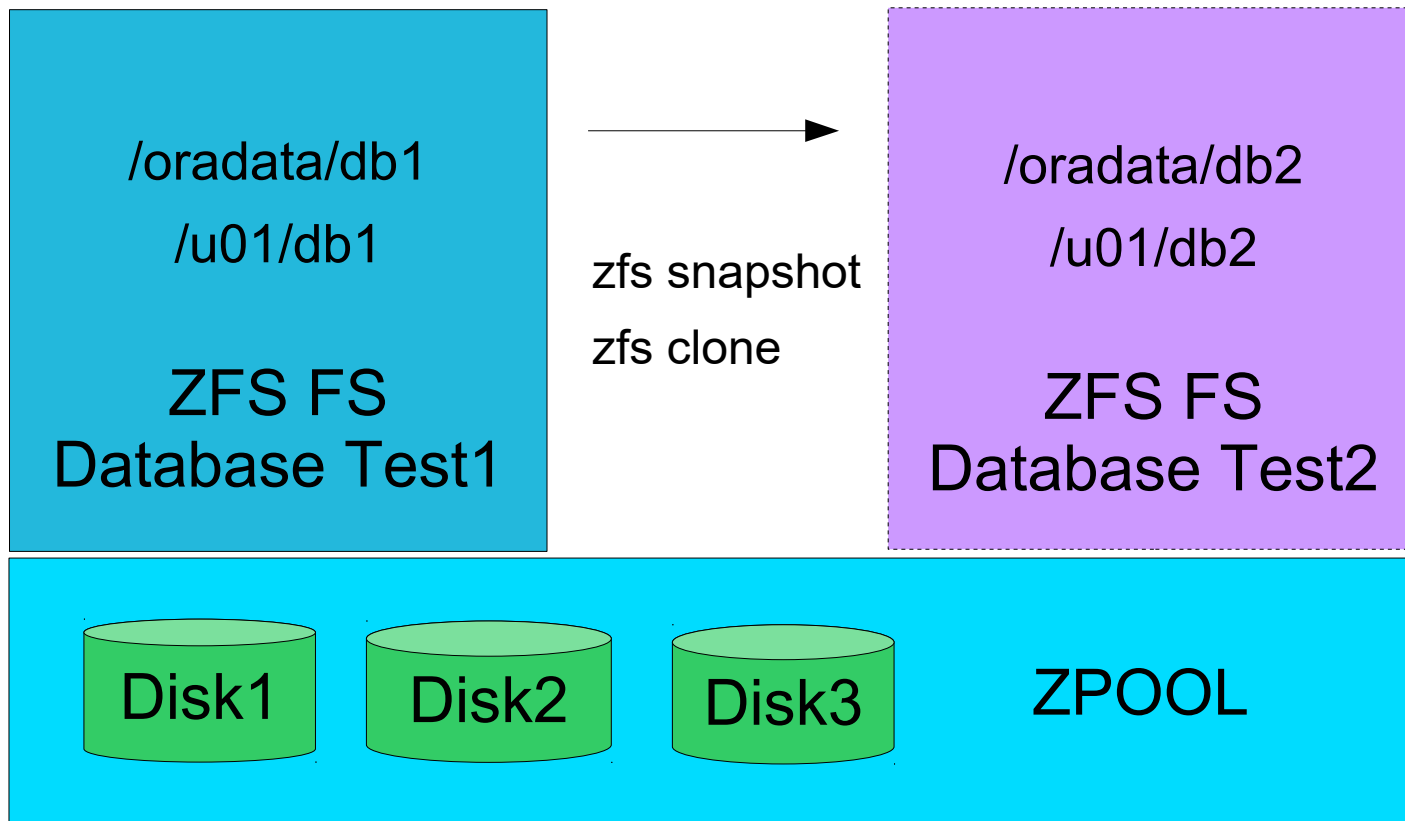
At a customer with 7 TB DWH / compressratio 2.8x

ZFS Host-Based Mirroring



ZFS - Cloning

- Cloning is done in 2 minutes
- Save disk space / only changes need to be stored in zpool



Solaris 11.4 / ZFS destroy

```
# zfs list destroytest/fs1
```

| NAME | USED | AVAIL | REFER | MOUNTPOINT |
|-----------------|-------|-------|-------|------------|
| destroytest/fs1 | 22.1G | 17.1G | 22.1G | /fs1 |

```
# time zfs destroy destroytest/fs1; zfs create -o mountpoint=/fs1 destroytest/fs1
```

```
real 0m0.654s
```

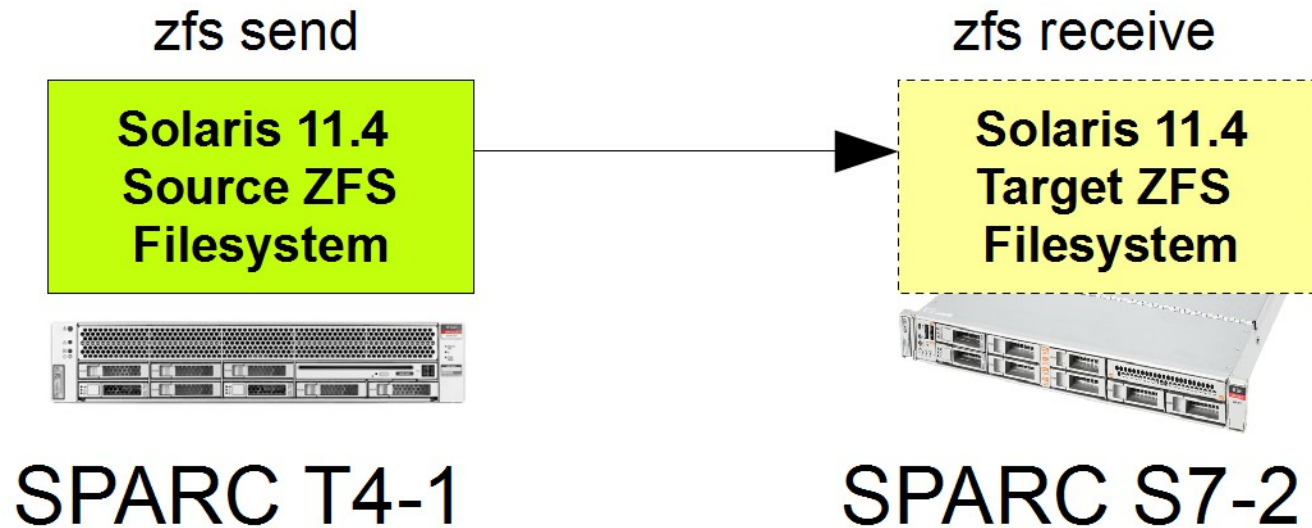
```
user 0m0.005s
```

```
sys 0m0.621s
```

```
# zpool monitor -t destroy destroytest 5
```

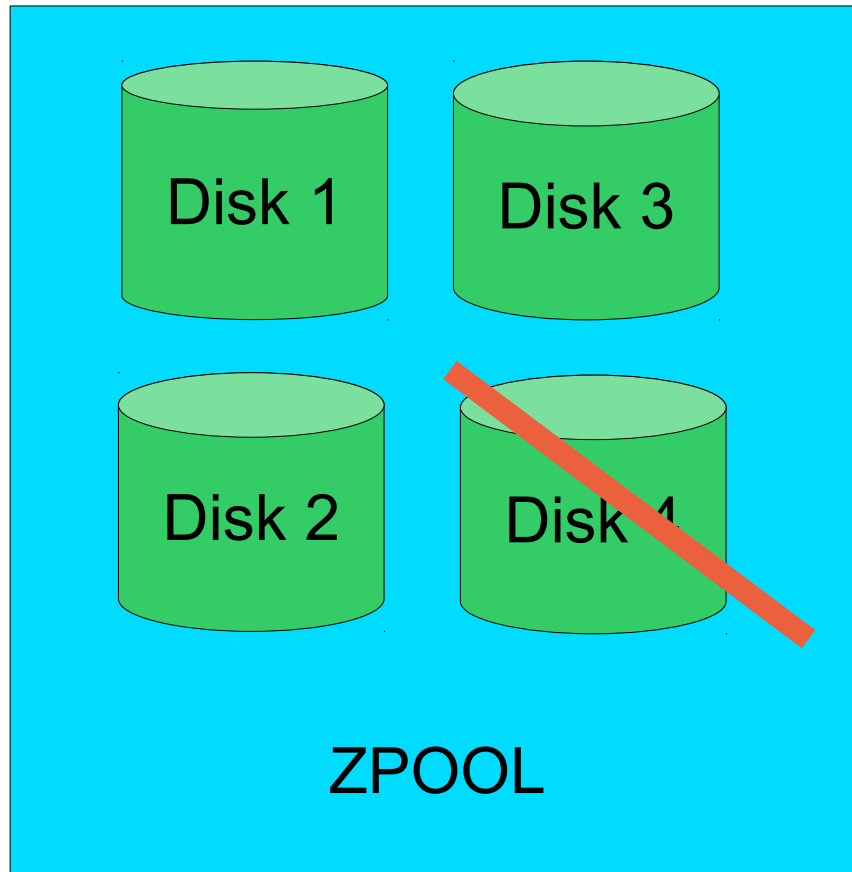
| POOL | PROVIDER | TOTAL | SPEED | TIMELEFT |
|-------------|----------|-------|-------|----------|
| destroytest | destroy | 22.1G | 0 | unknown |
| destroytest | destroy | 20.1G | 401M | 51s |
| destroytest | destroy | 13.5G | 872M | 15s |
| destroytest | destroy | 10.8G | 767M | 14s |
| destroytest | destroy | 4.92G | 878M | 5s |

Solaris 11.4 / ZFS Replication



- Filesystem replication over the network
- Restart is supported now
- Compressed data is now transferred compressed

Solaris 11.4 / ZFS Disk Remove



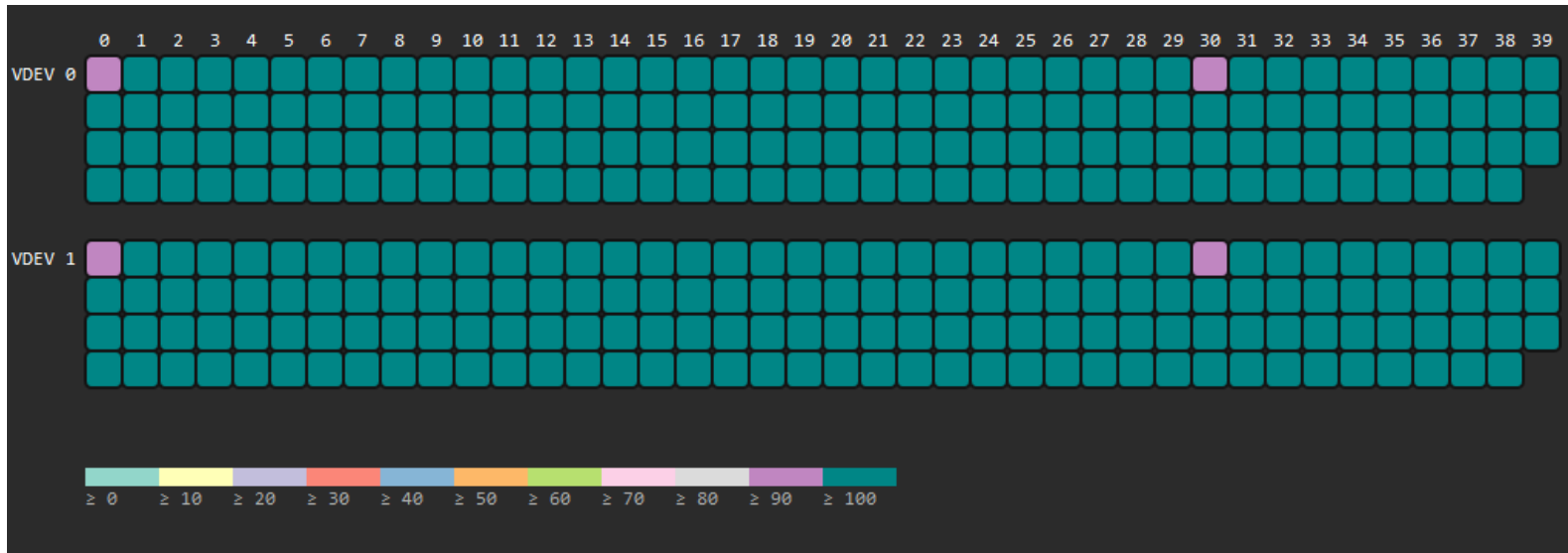
zpool must be upgraded to
version 44
zpool remove mypool c1d4

**Use only after accidental add of
disk to wrong zpool**
**Major read performance impact with
lots of data on the removed disk**

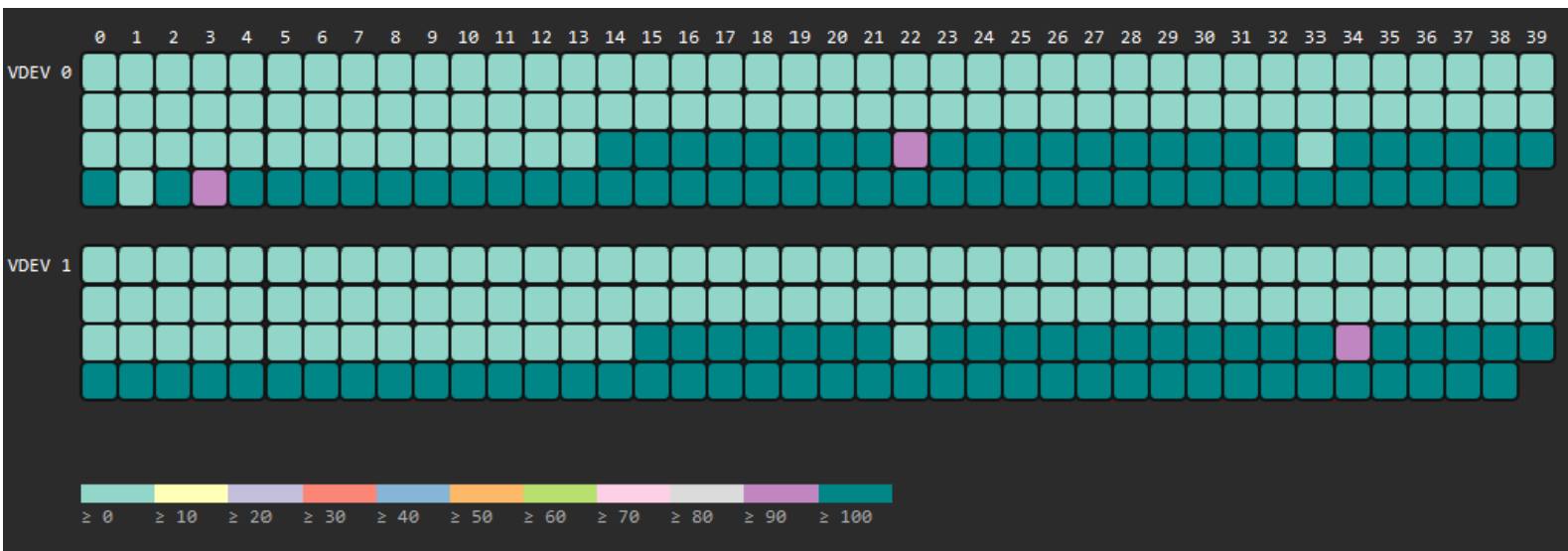


- ZPOOLS can shrink now (finally)
- Data is distributed to the other disks

ZFS - Fragmentation

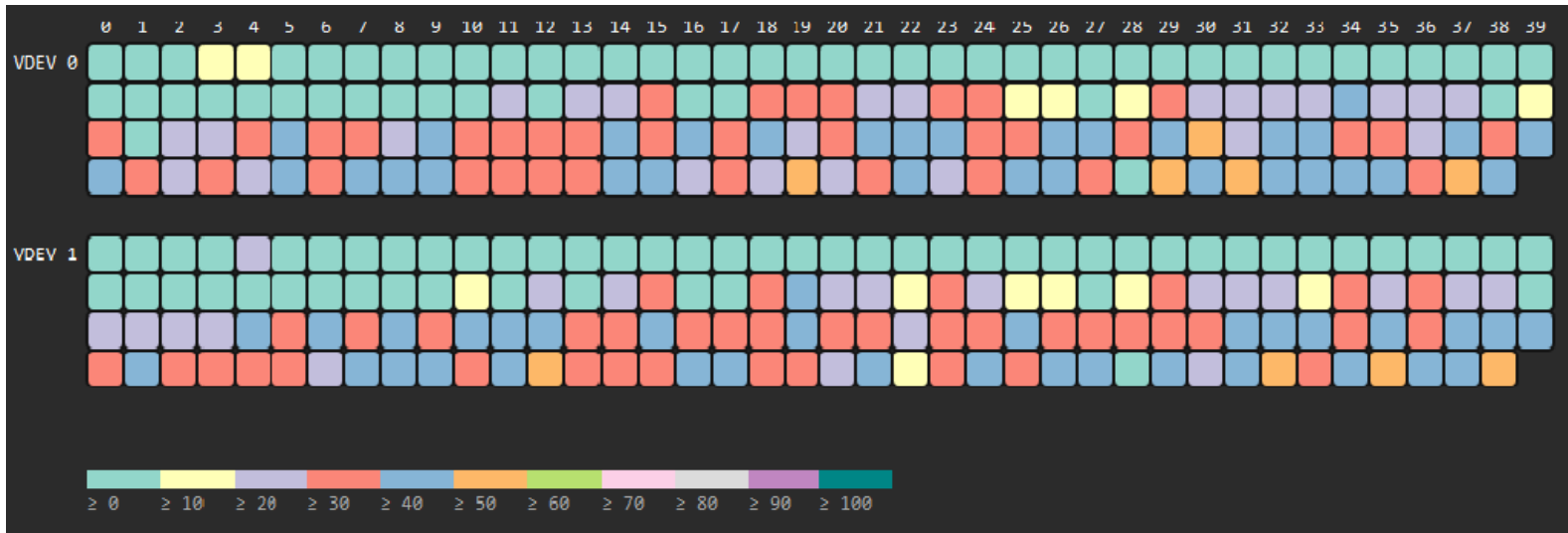


Empty ZPOOL



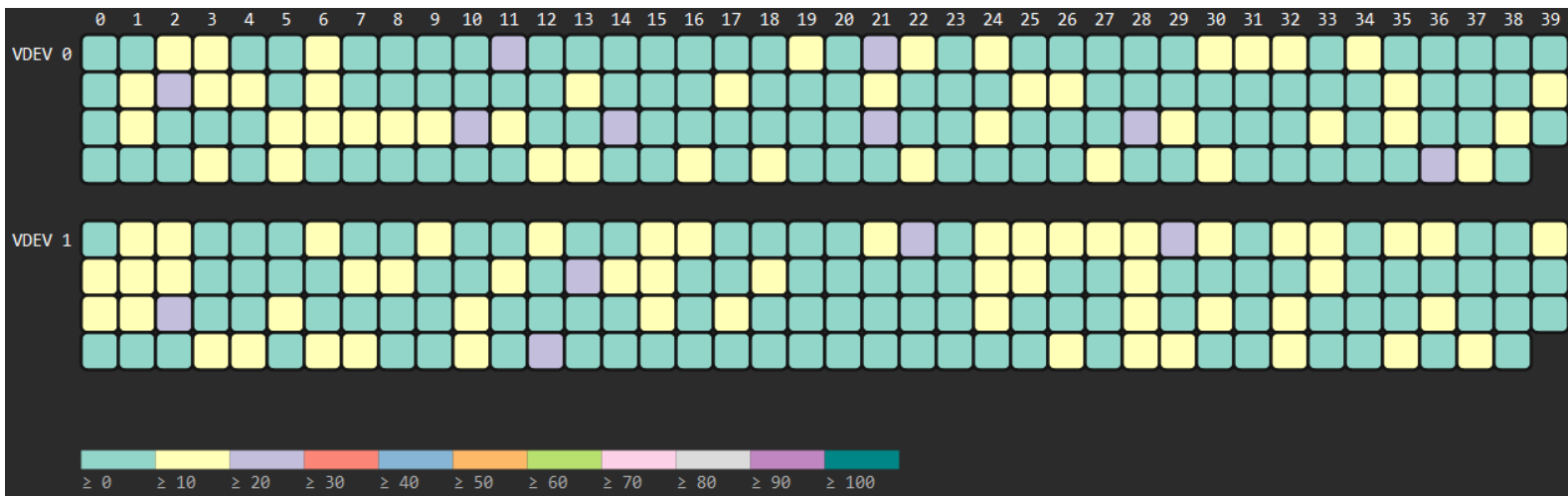
Now 60% full

ZFS - Fragmentation



After a few
deletes

76% full



90% full

fragmented

ZFS - Fragmentation

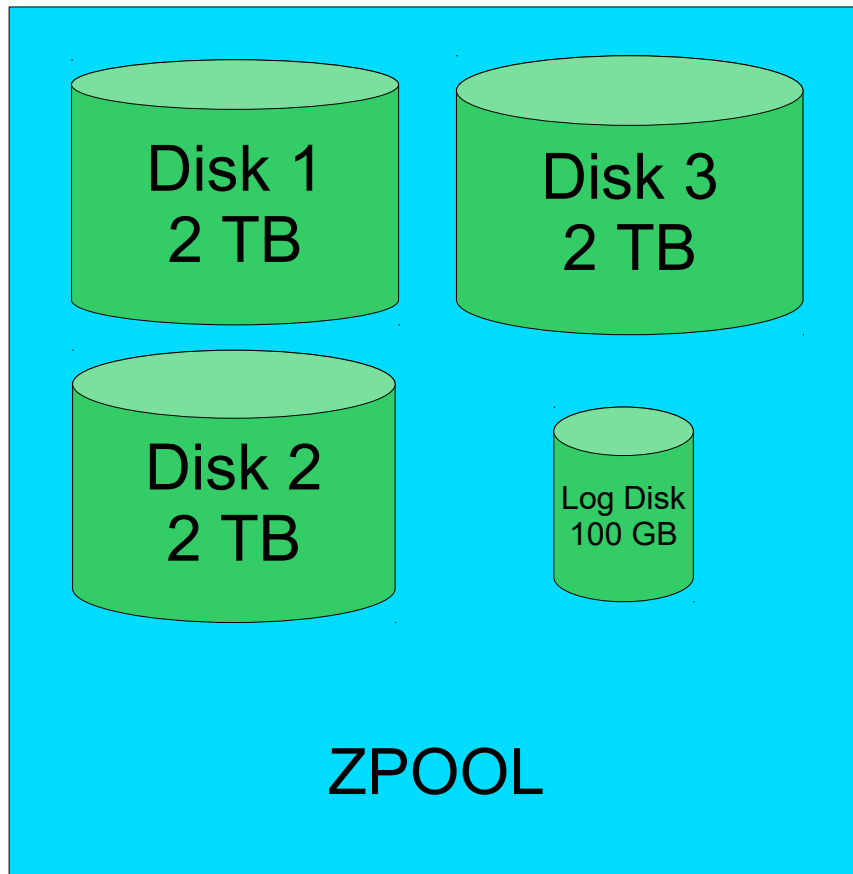
With very large ZPOOLS which are nearly full.
Write performance goes down because empty blocks must be searched for.

What to do?

- Update to current Solaris versions (improved algorithms)
- Add fast ZPOOL log disk and set logbias=latency
- If possible, export your database (new clean setup)
- Replicate your data to a new zpool (not mirroring)

- ZPOOL should be below 80% used

ZFS – Log Disk



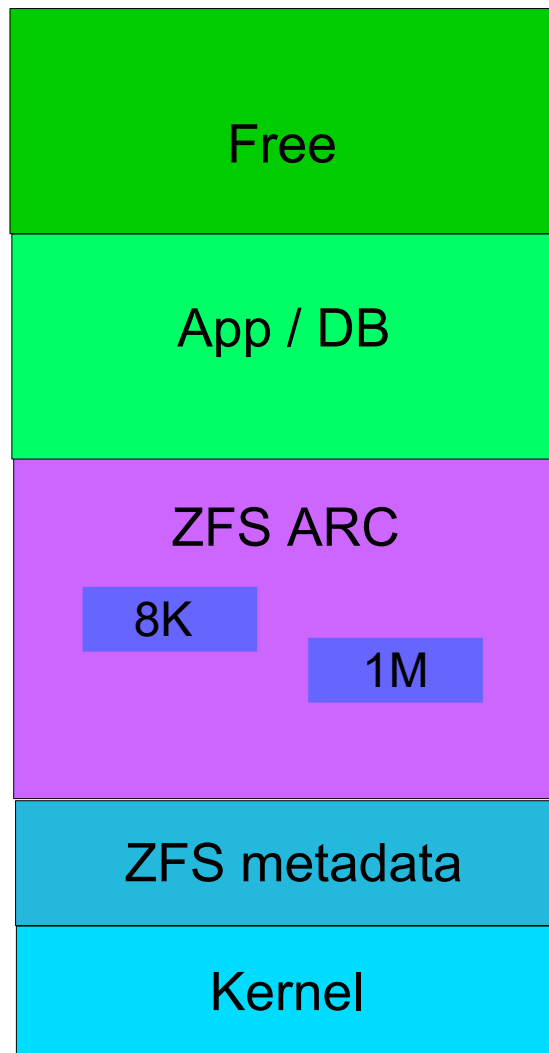
Data is written to the small empty Log disk

Database can continue to work

ZFS writes data to the 3 data disk (background)

Low latency but more IOPS

ZFS – Cache (ARC / Adaptive Replacement Cache)



Limit the ZFS ARC, but not too much

- Apps/Databases need memory
- Make sure there is free memory available

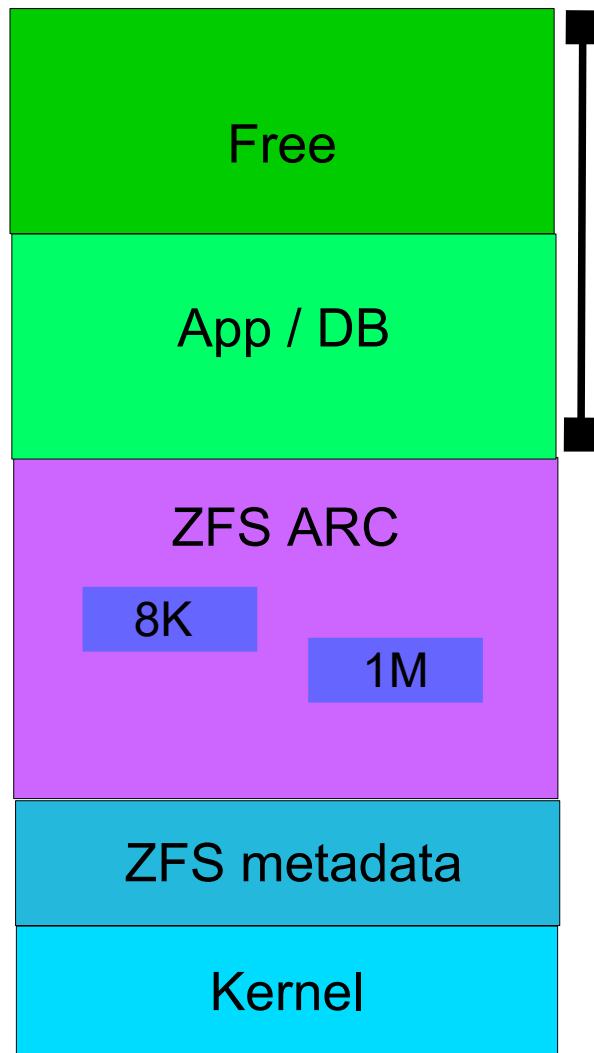
`/etc/system`

`set zfs:zfs_arc_max=`

Monitoring with:

```
echo "::memstat -v" | mdb -k
```

ZFS – Cache (ARC / Adaptive Replacement Cache)



/etc/system

set user_reserve_hint_pct= 30-75

New setting in Solaris 11.2
 Can be changed online
 Don't set too low or high

ZFS – Oracle Whitepaper for DB

New version dated 09/2020

recordsize=32K (for OLTP)
recordsize=128K (for DWH)

Use Log Devices
logbias=latency

Old version dated 01/2014

recordsize=8K (DB Block Size)

No Log Devices
logbias=throughput

Unchanged

Limit ZFS Cache (arc_max)

Create separate redo and archive pools

Oracle DB on ZFS – Customer Case

18 TB Database with lot of data changes

Very slow Average Write > 40ms

System „Freezes“ from time to time / Usually at Log Switch

Findings

ZPOOLS with Log Disk, but unused because
logbias=throughput → latency

Very large ZFS ARC - set user_reserve_hint_pct=50

→ Not enough free memory to allocate new buffers

→ Memory pressure

→ zfs_arc_max=256 GB

Oracle DB on ZFS – Customer Case

Additional changes

Separated redo zpool with log disks

Separated archive zpool

I/O was limited with

set zfs:zfs_vdev_max_pending=8 → 20

Update to Solaris 11.4

Result

Average write < 1ms

ZFS – Performance Recommendations

Use multiple zpools with multiple disks (Striping)

On SAN: increase `zfs_vdev_max_pending` (Default 10)

Add fast log disks for write performance

Adjust ZFS ARC settings

- Enough free memory (around the size of ARC MAX)
- Depends on App/DB requirement

Make sure the system has enough CPUs to handle I/O

If possible, export or reorganize your database from time to time

Links

Solaris 11.4 Download (GA or CBE 11.4.42)

<https://www.oracle.com/solaris/solaris11/downloads/solaris-downloads.html>

zfs_msviz: A Tool to Visualise ZFS Metaslab allocations

MOS Doc ID 1624945.1

Oracle DB on ZFS Whitepaper

<https://www.oracle.com/technetwork/server-storage/solaris10/config-solaris-zfs-wp-167894.pdf>

ZFS Admin Guide – Remove Disk



<https://docs.oracle.com/en/operating-systems/solaris/oracle-solaris/11.4/manage-zfs/removing-devices-storage-pool.html>

JomaSoft VDCF - Virtual Datacenter Cloud Framework

<https://www.jomasoft.ch/vdcf/>

Oracle Database on Solaris ZFS Done Right

Questions?

Marcel Hofstetter

hofstetter@jomasoft.ch

<https://jomasoftmarcel.blogspot.ch>

CEO / Enterprise Consultant
JomaSoft GmbH

 <https://www.linkedin.com/in/marcelhofstetter>

 https://twitter.com/marcel_jomasoft

 <https://jomasoftmarcel.blogspot.ch>

