# Experimental Computing Laboratory
August 24, 2023

Advanced Computing Systems Research Section

Jeffrey Vetter, Section Head

Steve Moulton, Systems Engineer

## Systems Status

- No major changes
- All systems updated during August 15 maintenance window (except Leconte)
- File system changed from RAID to ZFS (replacement hardware)

## ZFS

- ZFS file services in production use
  - Old file server will lie fallow for a couple of months, then become DR backup
  - Will deprecate tape services, which aren't working well anyway.

# Old file server architecture

- Files served by NFS, which ran on top of
- Ext4 for file services, which ran on top of
- LVM, for volume services
  - And could have provided software raid
- Block file services provided by hardware RAID
  - Broadcom (was Avago, was LSI, was probably something else)
    - MegaRAID SAS-3 3108

## Old File server (FS00) architecture (2)

- 21 8 TB disks
  - 2 10 disk RAID6 volumes
  - 2 Hot spare (configured to be dynamically added to raid set on disk failure)
    - But didn’t, had to force rebuild by hand
  - One disk failure in 6 service years (not bad)
    - Disk has been replaced.

## New File Server (FS01)

- 2 CPU * 12 Cores * Two Threads
- 512 GB Memory (why so much? … later)
- 16 14.6 TB disks
  - Expandable to 24 if needed, need not be same disk model/size
- 2 1.7 TB SSDs for secondary read cache
- 2 894 GB SSDs for ZFS Intention Log (fast commit)

## ZFS Architecture Basics

- All services (pool management, volume management, file service, NFS) in one integrated stack
- Will use (nearly) all available memory for read cache
  - Outstanding read performance for in-cache data
    - Cache optimized for ZFS, better performance than Linux disk caching
  - Similar to RAID performance for writes
  - Write speeds limited only by saturation of ZIL
- Not a high performance/parallel file system.
  - Not a replacement for Lustre, GPFS, Ceph, etc.

## ZFS Intention Log

- ZFS guarantees all data is consistent
  - Writes happen in parallel to ZIL devices (generally fast SSDs) and physical (rotating) disks.
  - Writes complete when ZIL write is complete
  - ZIL automatically replayed after system (i.e. power) failure.
    - Ensures written data always consistent
- In our instance, ZIL SSDs are mirrored for reliability.

## Data Integrity

- All data written with 256 bit hash on each file system block (uses Fletcher4 checksum, more secure checksums available if needed)
- All hashes check on block read
- All blocks in all file systems read (in our case once every other week) and hashes compared to ensure no bit rot.
- Disks that have bit rot automatically replaced and rebuilt.

## The future for ZFS

- Sadly, ZFS is likely to decrease in importance as rotating disk devices disappear
- Competing approaches (like WEKA) promise superior performance on SSDs.
- But for rationally priced storage, ZFS will be around for quite some time.

## Read Cache

- 50% of available memory use for ZFS read cache by default
  - We use 80% to maximize performance – based on monitoring memory use.
- Main memory is primary read cache
  - Data written to SSD secondary cache when evicted from main memory
  - MRU *and* MFU algorithm used for evictions
  - Read cache automatically invalidated on write
  - We see 99.2% hit rate on average on primary read cache.
    - This high due to metadata caching

## Compression

- Compression is enabled on the fly per file system or file subsystem
  - Whether a file is compressed is stored in its metadata
  - Files are compressed only when written if compression is turned on for that file system
  - LZ4 default, can be altered on a per file subsystem basis
- Overall /home compress ration is 1.9
- Overall /noback compress ration is 2.0
- Some users much higher – data dependent

# Pool Layout

Raid6 Set

Raid6 Set

```
  NAME                                       STATE     READ WRITE CKSUM
  pool                                      ONLINE       0     0     0
   raidz2-0                                 ONLINE       0     0     0
     scsi-SWDC_WUH721816AL5204_2BJ569ND     ONLINE       0     0     0
     scsi-SWDC_WUH721816AL5204_2BJ56KPD     ONLINE       0     0     0
     scsi-SWDC_WUH721816AL5204_2BJ4VA6D     ONLINE       0     0     0
     scsi-SWDC_WUH721816AL5204_2BJ5DJHD     ONLINE       0     0     0
     scsi-SWDC_WUH721816AL5204_2BJ5DG9D     ONLINE       0     0     0
     scsi-SWDC_WUH721816AL5204_2BJ4U3LD     ONLINE       0     0     0
     scsi-SWDC_WUH721816AL5204_2BJ55N4D     ONLINE       0     0     0
   raidz2-1                                 ONLINE       0     0     0
     scsi-SWDC_WUH721816AL5204_2BJ541YD     ONLINE       0     0     0
     scsi-SWDC_WUH721816AL5204_2BJ5D75D     ONLINE       0     0     0
     scsi-SWDC_WUH721816AL5204_2BJ540LD     ONLINE       0     0     0
     scsi-SWDC_WUH721816AL5204_2BJ5D1WD     ONLINE       0     0     0
     scsi-SWDC_WUH721816AL5204_2BJ55X5D     ONLINE       0     0     0
     scsi-SWDC_WUH721816AL5204_2BJ547DD     ONLINE       0     0     0
     scsi-SWDC_WUH721816AL5204_2BJ54KLD     ONLINE       0     0     0
 logs
   mirror-2                                 ONLINE       0     0     0
     scsi-SATA_Micron_5300_MTFD_222839DAD973  ONLINE     0     0     0
     scsi-SATA_Micron_5300_MTFD_222839DADBB8  ONLINE     0     0     0
 cache
   scsi-SATA_Micron_5300_MTFD_22073524555B    ONLINE     0     0     0
   scsi-SATA_Micron_5300_MTFD_22073524556A    ONLINE     0     0     0
 spares
   scsi-SWDC_WUH721816AL5204_2BJ5DNSD         AVAIL
   scsi-SWDC_WUH721816AL5204_2BJ5D98D         AVAIL
```

## Layout Details

- RAIDZ2 sets (in ZFS nomenclature) have same functionality as RAID6 sets.
- 7 disks to get 5 effective (effectively 2 parity).
  - Two sets so effectively 10 * 14.6 TB
- The log SSDs are the Zfs Intention Logs
  - Raid 1 (mirror) duplication for reliability
- The cache SSDs are the secondary read cache
  - Raid 0 (stripes) for performance.

## Subfilesystems

- Each pool (only one in our case) can have an arbitrary number of file systems
- Each file system can have an arbitrary number of subfilesystems, and of arbitrary depth
- Properties are assigned to each (sub)filesystem, and propagate from the parent file system.
  - I.e., compression is on for pool, and is therefore on for pool/home/hsm
- Snapshots are done per (sub)filesystem

## Snapshots

- Snapshots take a picture of the file system as it exists at that time.
- Metadata are copied, actual files not touched until …
- When a file is written, the original copy is moved to the snapshot, and the new data written to the primary file system (Copy On Write).
- Older snaps maintain state as you would expect (modified file not reflected here).

## Snapshot Management

- /home and /noback have full periodic snapshot implemented (as do /auto/software, u250, sysadmin and devdocs subfilesystes).
- Durations
  - Hourly for (at least) 48 hours
  - Daily for 14 days
  - Weekly for 8 weeks
- Snapsnot purges done nightly at midnight
  - Takes about 20 minutes
  - Low priority, does not affect file system performance

## Ok, the magic!

- cd into `~/.zfs/snapshot`
- There they are!
- Snapshots are read only, but you can copy files back into your primary file system.
- The .zfs directory is only visible when you are in it, but it is always there in the top level of your filesystem hierarchy.

## Snapshot layout

```
hsm@milan2:~/.zfs/snapshot$ ls
daily-2023-08-11_00.00.01--14d          hourly-2023-08-23_03.00.01--48h
daily-2023-08-12_00.00.01--14d          hourly-2023-08-23_04.00.01--48h
daily-2023-08-13_00.00.02--14d          hourly-2023-08-23_05.00.01--48h
daily-2023-08-14_00.00.01--14d          hourly-2023-08-23_06.00.01--48h
daily-2023-08-15_00.00.02--14d          hourly-2023-08-23_07.00.01--48h
daily-2023-08-16_00.00.01--14d          hourly-2023-08-23_08.00.01--48h
daily-2023-08-17_00.00.02--14d          hourly-2023-08-23_09.00.02--48h
daily-2023-08-18_00.00.02--14d          hourly-2023-08-23_10.00.01--48h
daily-2023-08-19_00.00.01--14d          hourly-2023-08-23_11.00.01--48h
daily-2023-08-20_00.00.01--14d          hourly-2023-08-23_12.00.01--48h
daily-2023-08-21_00.00.01--14d          hourly-2023-08-23_13.00.01--48h
daily-2023-08-22_00.00.01--14d          hourly-2023-08-23_14.00.01--48h
```

And so on

# How to access snapshots

```
hsm@milan2:~$ cd .zfs/snapshot/weekly-2023-07-30_00.00.01--4w/
hsm@milan2:~/.zfs/snapshot/weekly-2023-07-30_00.00.01--4w$ ls -latr | tail -
3
-rw-r--r--  1 hsm  users    7645811 Jul 27 15:01 SeaChest_Lite_x86_64-
redhat-linux_R_RAID.zip
-rw-------  1 hsm  users      18549 Jul 27 20:20 .bash_history
drwxrwxrwx  2 root root           2 Aug 24 12:00 ..
hsm@milan2:~/.zfs/snapshot/weekly-2023-07-30_00.00.01--4w$ cp .bash_history
~/.bash_history.old
hsm@milan2:~/.zfs/snapshot/weekly-2023-07-30_00.00.01--4w$
```