Lustre on clown drives

October 5th, 2023

Gaël DELBARY

Agenda

- An old Lustre filesystem
- « Saving private work Lustre filesystem »
- « Live and let die »

An old Lustre filesystem: id

- Main property: users love me
  - **Not purged**
  - But quotas enforced
  - Mounted on all compute nodes
- Main weakness: **I am old**

Name	work
Vendor	HPE ClusterStor L300
Age	6 years old
#Clients	~6000
Used Capacity	5.7 PB (total 8.2 PB)
Inodes	300 millions
Lustre	2.12.4 (CS)
Write seq throughput	60 GB/s
OSS HDDs (GridRaid 8+2+1)	~1300 (Seagate 10TB)

An old Lustre filesystem: life events

- H1 2022:
  - One drive failure, Raid recovery stale
  - OST blocking
    - Another drive is slow (unread sectors)
    - Kill slow drive
  - Raid recovery resuming
  - OST living back ☺

- H2 2022: my end?
  - One drive failure, raid recovery stale, another slow drive killed
  - Raid recovery stale, another slow drive (second) repaired
  - Raid recovery definitively stale, a third slow drive
  - OST offline (~300 TB, ~5 millions of files unavailable)
  - Same severe issue on some OSTs

An old file systems: analysis

- Raid stack recovery stale on unreadable sectors
- Drives related?
  - Glist growing up, how far?
    - 200 HDDs, Glist > 0
    - 120 HDDs > BER (1 unreadable sector on 125TB read)
  - Begin a replacement process
    - Fall on others ost recovery stale
  - Worst:
    - Successfully replaced “bad BER” drive
    - 15 days later, one drive dies
    - Recovery stale on unreadable sector (not seen 15 days ago)
- Data “stays” 15 days on some/many HDD

An old Lustre filesystem: recovery?

- Unreadable sector = data not readable

- 2 plans:
  - Isolate unreadable sectors (-k fsck option)
  - **Remap bad sector on write**

- Tradeoff: high probability to corrupt final data

- How can we be sure?
  - Hypothesis: raid mapping is well known (link between raid volume sector and drives sector)
  - Raid volume sector = used by ldiskfs

An old Lustre filesystem: Lustre internals

- File layout example:

```
[user@client delbar4c]# echo “iamalive” > iamstillalive
[user@client delbar4c]# lfs getstripe iamstillalive
iamstillalive
lmm_stripe_count:  1
lmm_stripe_size:   1048576
lmm_pattern:       raid0
lmm_layout_gen:    0
lmm_stripe_offset: 18
lmm_pool:          t1_tgcc_ssu3
        obdidx       objid          objid            group
          18       267211928      0xfed5498            0
```

- How my object is stored on OST index 18 and where?

An old Lustre filesystem: Lustre internals

- Find my FID

```
[user@client delbar4c]# lfs path2fid iamstillalive
[0x6800c7805:0x16:0x0]
```

- « Data layout » (lfs getstripe)

```
        obdidx          objid              objid                group
         18          267211928          0xfed5498                 0
```

- Which OSS ?

```
[user@client delbar4c]# cat /proc/fs/lustre/osc/work4-OST00$(echo "obase=16; ibase=10; 18" |
bc)*/import | grep current_conn
       current_connection: 192.168.1.11@o2ib1
```

- Which file on the OSS?

```
[root@oss210 ~]# debugfs -c -R "stat /O/0/d$((267211928%32))/267211928" /dev/md0 | grep -A1 -E
'(parent|EXTENTS)' | grep ':'| grep -v EXTENTS
debugfs 1.45.6.cr1 (14-Aug-2020)
/dev/md0: catastrophic mode - not reading inode or group bitmaps
  fid: parent=[0x6800c7805:0x16:0x0] stripe=0 stripe_size=1048576 stripe_count=1
(0):4515846334
```

- File data is on block `4515846334` from block device md0 on OSS210
- Reverse process (ldiskfs block->ost object->fid) makeable
  - Debugfs icheck command

2. « Saving private work Lustre filesystem »

How to recover? (real event)

- Sum-up: one OST fully offline:
  - Disable osc on all clients (to “save” production), time to try to fix:
    - `mgs# lctl set_param -P osc.osc_name.active=0`
  - 2 HDDs killed (raid pool is degraded)
  - 1 slow drive (recovery stale):
    - Get bad sectors list (badblocks util) : ~100 sectors
    - Get back to ldiskfs : many raid spares, no data blocks, no fid impacted
    - Remap ~100 sectors:

      -
        ```
        dd of=/dev/ostblockdevice seek=badsector bs=4096
        obs=4096 count=1 if=/dev/zero oflag=direct
        ```
  - Recovery continues but … 1 other slow drive (recovery stale again):
    - Badblocks unable to read more than 80%
    - About 450 millions of blocks unreadable

Recovery plan

- Main idea: isolate write on this OST (users have to read datas):
  - Old readonly feature (LU-8200):
    - `mount -t lustre /dev/'xxxx' -o rdonly_dev /mnt/'yyyy'`
    - Kernel patches mandatory (luckily, inside Lustre ClusterStor version)
    - Only for tests ☺
    - Removed in Lustre 2.15 (LU-12477)
      - Replaced by dm_flakey (nice kernel feature)
      - Not working with external journal

  - Disable object creation on mdt level (just in case):
    - `mgs# lctl set_param -P osp.osc_name.max_create_count=0`

  - Assemble in frozen mode (recovery blocked until I/O access)

Recovery plan, continue

  - ddrescue, one pass ~80% (like mdraid) during 2 days
  - Evaluate number of files impacted
    - Mapping raid on 450 millions of blocks (20% non readable full device blocks)
    - Intersec with 2 missing drives
    - Takes months, needs compute resources
    - Result (1 week): ~20 millions of files impacted (min raid group 8+1 outside 2 missing)
- Can we save a bit more (add parity)?
  - Rely on ddrescue (20% of data remaining):
    - Complex and amazing tool
    - Multiple “smart” pass
    - 3 weeks later, 98.2% read success
  - 2 days later, mdadm killed the drive
    - Manually revival drive, unable to read more than 30%

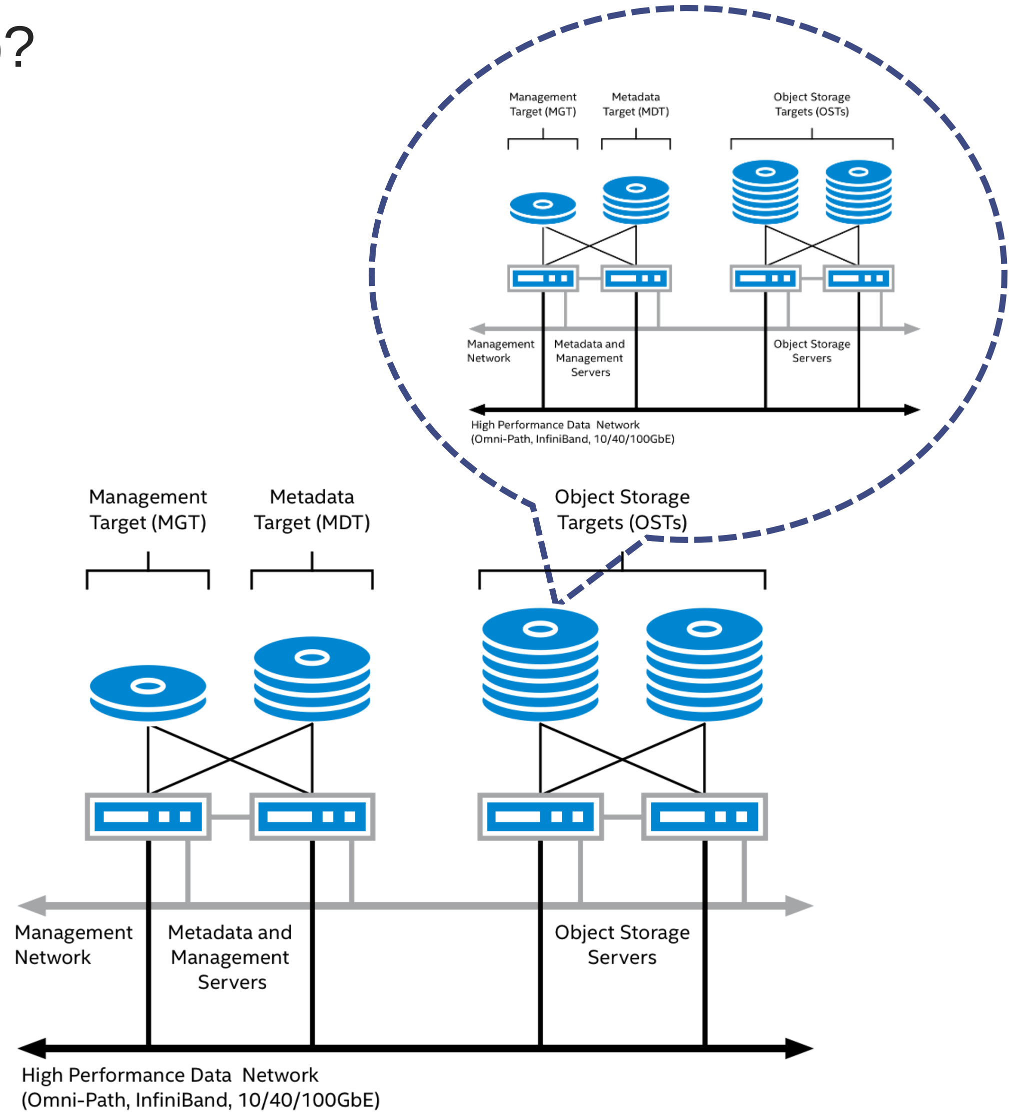
Recovery: final step1

- Raid pool fully degraded (38/41) : read occurred on missing member => panic ldiskfs!
- One cool thing, we have 98.2% of last failed drive
  - Insert loop device in raid pool
  - Fully corrupted ldiskfs device:
    - Unable to mount (block descriptors corrupted)
    - Binary analysis (dumpe2fs helpful):
      - Inode/block bitmap inside group descriptors (<=>)
      - Backups group descriptors invalids (not usable by e2fsck meta_bg related)
      - e2fsck need to rebuild corrupted group descriptors
    - Root cause:
      - old kernel (CentOS 7.6) = loop device 512 bytes
      - Mdraid mixes devices 512 and 4K sector size => not lovely
      - Resync started with bad sector size!

Recovery: final step 2

- Recovery full ldiskfs corruption, which can help?
  - E2fsck (to repair the corruption)
  - Lustre (of course ☺)?
  - Lustre on Lustre?
  - Through loop devices

- Creating 38 loop images
  - We have one more (latest drive)
  - 39 is enough
  - Rely on CEA Lustre FS (store)
  - Flexibility (copy/backup fast)
  - 38 parallel ddrescue on duty
  - Same sector size


Management Target (MGT)
Metadata Target (MDT)
Object Storage Targets (OSTs)
Management Network
Metadata and Management Servers
Object Storage Servers
High Performance Data Network (Omni-Path, InfiniBand, 10/40/100GbE)

Recovery: result part1

- Images creation duration: 24h (ddrescue rocks)
- Performance hurt:
  - Limit read_ahead
  - Adjust max_pages_per_rpc (64) related to chunk_size
  - Read sequential performance drops:
    - 3x slower (1.2 GB/s)
    - Enough for users
- Get time to analyze file corruption (e2fsck returns)
  - Figure out e2fsck corruption, silent corruption?
  - Robinhood:
    - Sha1 policy
    - Using DB Backup done before the incident
    - File only read (due to ost fake readonly mode)

Recovery: result part2

- Files corrupted: 9308/5 000 000
- A bit more, can we do better?
  - Winemaking?
  - Not grapes blend but drive images blend
  - Benefit: the best ldiskfs image with limited corruption
- We can:
  - Revive the 2 first drives
  - Assemble raid (not so easy, mdraid events, timestamps event)
  - Dumpe2fs full analysis
  - E2fsck analysis
  - Can play => loop devices

Recovery: final way

- Found a “perfect” combination:
  - Adding one missing stripe parity on many files
  - Still group descriptor corruption
  - E2fsck handle it
  - Restore/fix lost+found objects on the write place
- Result: **6** instead of 9308
  - 6 files were also on others filesystems
  - One OST saved!
- Unfortunately (January 2023):
  - We have replaced ~100 drives
  - Supplier out of stock on this type of drive
  - Filesystem is going to die
  - Time plays against the production

3. « Live and let die »

« Live and let die »

- No new drive = not able to save filesystem

- Adopt a different strategy: full fs migration
  - “Read useful” strategy
  - Tricky on almost dead drives (let mdraid remap unreadable sectors)
  - Large FS:
    - 300 millions inodes (thanks rbh-du)
    - 5.2 PB
    - 36 millions of hardlinks
  - Limit production impacts

- Workflow migration chain to build

Workflow migration chain: tools

- We have to rely on some tools:
  - Robinhood (hardlinks list)
  - Rsync to sync hardlinks
  - MPIFileutils to sync data

- Issues:
  - Rsync:
    - no way to sync large tree (timelimit)
    - Hardlink mode is “monothread”
  - Robinhood hardlink resolution slow
  - MPIFileutils:
    - No hardlink support
    - Lustre xattr issue…

Workflow migration chain: adaptations

- MPIFileutils:
  - Patch to ignore hardlinks (rely on rsync!)
  - Lustre xattr compliant (`commit 96a9fe7d2f49a9d6b3e28941bd51f33bef8af8ce`), 0.11.1 included
  - Patch to build on RHEL8
- Robinhood:
  - Mariadb hardlink path resolution costly
  - Use fid hardlink (robinhood db) like source
  - Build a tree of files rsync compliant syntax
    - Thanks to O(1) with fid2path
    - Achieved 36 000 000 in 2 hours
- Rsync limited to one fid file
  - Run multiple rsync in parallel (2048)
  - Achieved 36 000 000 of hardlinks in 8 hours

Workflow migration chain: result

- Macro steps to migrate a group or a container (multiple groups):
  - Production running: sync data/hardlinks
  - Day of migration:
    - Block connections to group/container on cluster
    - Save group jobs related to the migration
    - Check open_files on MDTs
    - Set safeguard unix rights on top directories
    - Parallel diff (walking metadatas)
    - Delta parallel sync
    - PAM namespace modification
    - Remove safeguard unix rights
    - Re-enable group/container access
- Performance: 100 000 000 inodes in 8h

Summary

- “successfully” copied the filesystem with clown drives
- Lustre versatility
- Strong parallel opensource tools
- Old HDD : Glist monitoring (IA model running)
- Issues hit:
  - E2fsck pass-1d stuck (1 year to finish)
  - Mballoc (LU-13291)
  - l_tunedisk (LU-12029)
  - RO ost (LU-12477)
  - MPIFileutils (lustre xattr)
  - Fiemap, salb corruption (LU-17110)

Thanks

CEA / DAM Ile-de-France

Bruyères-le-Châtel
91297 Arpajon Cedex
France

gael.delbary@cea.fr

Standard. + 33 1 69 26 40 00

Questions?

CEA / DAM Ile-de-France

Bruyères-le-Châtel
91297 Arpajon Cedex
France

gael.delbary@cea.fr

Standard. + 33 1 69 26 40 00