Lustre HSM & Object storage

Developing an open source copytool

Frédérick Lefebvre - fred@calculquebec.ca
Simon Guilbault - simon.guilbault@calculquebec.ca
Who Are We?

- Advanced Computing Center @Université Laval in Quebec City
- Part of Compute Canada & Calcul Quebec
- Operate 2 parallel cluster and 4 Lustre FS
  - Users of Lustre since 2009
  - We use a mix of community releases and Seagate
Why

- User are demanding more storage capacity
- Researchers have asked us for cheaper tiers of storage (but still want to run parallel jobs)
- Users are coming to us today with creative poor solutions from non-traditional vendors
  - We need to move if we want to retain our sanity
Evolving Landscape

- Compute Canada is an acquisition process for a large pool of object storage
  - 40+ PB in phases
- We like the idea of being able to extend our local parallel storage unto this new central storage at a low cost for the users
Object API - Things to consider

- We use CEPH internally
- The Compute Canada object storage is not chosen yet
- Rados, while an obvious choice for us, might be too restrictive
- Scalability/stability (especially over WAN) of available POSIX gateways is unclear
Object API

- Ceph/rados eliminated as too restrictive
- 2 generic/common APIs:
  - S3
    - Well supported by most Object stores
  - Swift
    - C library unmaintained: [https://github.com/ukyg9e5r6k7gubiekd6/swift-client](https://github.com/ukyg9e5r6k7gubiekd6/swift-client)
    - CEPH/RadosGW implementation behaves differently than others
    - Keystone auth is more complicated
How

● Started from lhsmtool_posix.c contributed by CEA and included in the Lustre source tree
● Initially modified it to do S3 puts and gets
● Added a Rados version for validation
● ~30% of common code
  ○ modularized it in a library for reuse
  ○ ‘Libct’ also included
LibS3 Reliability

- libS3 is not available in official Centos repo
  - only in epel
- Numbering of packaged version has not changed in 5 years. Changelog not updated in 8 years!
- Initial tests of our copytool segfaulted in libcurl with large files
LibS3 Reliability (cont.)

- We use the latest ‘master’ from git
  - https://github.com/bji/libs3
- Need to patch libS3 to prevent the segfault
  - comment out 2 lines
- The patch and instructions are included with our copytool
- We only tested with RadosGW
HSM Overview
Ceph APIs

LIBRADOS
A library allowing apps to directly access RADOS, with support for C, C++, Java, Python, Ruby, and PHP

RADOSGW
A bucket-based REST gateway, compatible with S3 and Swift

RBD
A reliable and fully-distributed block device, with a Linux kernel client and a QEMU/KVM driver

CEPH FS
A POSIX-compliant distributed file system, with a Linux kernel client and support for FUSE

RADOS
A reliable, autonomic, distributed object store comprised of self-healing, self-managing, intelligent storage nodes
Ceph/radosGW Overview
Technical Overview

- Mapping Lustre FID to S3 Objects

```
[0x200000bd8:0x29a:0x0]  s3://lustre_hsm_141/00000000200000bd8_0000029a_00000000
```

- Lustre FID
- bucket_prefix
- Bucket ID
- Lustre FID
- Chunk ID

- Bucket_prefix and bucket_count are from the configuration file
- Bucket ID
  - Sharding objects across multiple buckets to improve PUT speed
- Chunk ID
  - Used to store file larger than the chunk_size
Technical Overview (cont.)

● Not using multipart upload
  ○ 5TB limit
  ○ More complicated to handle

● Compression with LZ4
  ○ Native on ZFS
  ○ Reduce the problem caused by sparse file

● Checksum with the MD5 hash in the object’s metadata

● Bucket sharding
  ○ Reduce contention for the index of each bucket
    ■ PUT will get slower with a large amount of object in the same bucket
    ■ GET should be unaffected
Metadata on S3 Objects

# s3cmd info s3://lustre_hsm_141/0000000200000bd8_0000029a_00000000.0
File size: 105268808
MIME type: application/x-lz4
MD5 sum: 7c053eb2358c1420ce93ceaa3710f262
x-amz-meta-chunksize: 104857600
x-amz-meta-totalength: 19209912320

- Incompressible file, small overhead
- To support multiple compression algo
- Checked when restoring
- Size of each chunk (100MiB)
- Total size (~19GB)

- Also storing UID/GID and a few others metadata for a disaster recovery
  - Everything should already be in Robinhood
Test Hardware

- We used hardware on loan from HPE
- 2x SL4540 for CEPH OSDs
  - Centos 7.2 + CEPH 0.94 (Hammer)
  - Journals on SSD
- 2x Apollo 4520 for Lustre
  - Centos 7.2 + Lustre 2.8 + ZFS 0.6.5.4
Test Hardware (cont.)

Only for tests purposes

Lustre MDS/MGS + Ceph Mon

DTN + Ceph Mon

HP DL360 G9

Ceph OSDs

HP SL4540
2 nodes per chassis
20 x 4TB HDD
5 x 400GB SSD

Lustre OSSs

HPE Apollo 4520
2 nodes per chassis
23 x 4TB HDD
Failover capability
Benchmark (Ceph Setup)

- **Erasure encoding**
  - Jerasure 8+2
    - Not the fastest implementation
    - Not host redundant with this amount of servers

- **Replication with 3 copies**
  - Performance limited by the network
    - Only one QDR (IPoIB) connection per server

- **Journals on SSD**
  - Could use the SSD’s leftover for a fast Ceph pool or cache
Benchmark (Lustre Setup)

- **ZFS**
  - Compression with LZ4
  - Large blocks enabled
  - Stripping across 2x raidz2 (9+2)

- **Performance for 4 nodes**
  - 8.2 GB/s write
  - 4.5 GB/s read
  - IOR Tests on the older generation
  - Did not tune the Apollo 4520 for the HSM’s tests
    - [http://slideshare.net/Lefebvre2/lustrezfs-on-the-apollo-4000-platform-55112048](http://slideshare.net/Lefebvre2/lustrezfs-on-the-apollo-4000-platform-55112048)
Benchmark Datasets

- Compressible data
  - SAM files (Genomic, huge ASCII files)
  - 200 files of 0-20GB each (2TB)

- Incompressible data
  - BAM files (Genomic, compressed format of SAM)
  - 200 files of 0-20GB each (2TB)

- Large-ish amount of files
  - Kernel sources (~50k files, median ~5kb)
Benchmarks (Archival)

Archive throughput

- S3 replication: Compressible 1123, Uncompressible 650
- CephFS replication: Compressible 1122, Uncompressible 1173
- Rados replication: Compressible 2322, Uncompressible 754
- S3 erasure: Compressible 754, Uncompressible 235
- Rados erasure: Compressible 1079, Uncompressible 704
Benchmarks (Restore)

Restore throughput

- CephFS replication: 616 (compressible), 625 (uncompressible)
- Rados replication: 627 (compressible), 690 (uncompressible)
- S3 erasure: 422 (compressible), 796 (uncompressible)
- Rados erasure: 571 (compressible), 1750 (uncompressible)
Limitations

- HSM Restore is 30% to 50% slower than HSM Archive
- Lustre client hangs at intervals in the restore process (soft lockup, CPU stuck)
- Valid for hsmtool_s3 and hsmtool_posix
- Others have seen Similar results
Limitations (cont.)

- Partial archives and restore not tested
  - HSMv2 ?
    - Use multiple worker for one file
- Cancel not supported
- Priority
Sometimes it can return a negative number of transfers in progress

[root@r2-u10 ~]# cat /proc/fs/lustre/MDT0000/hsm/agents
uuid=21b44f0a-49eb-de43-99ff-99894552a6b3 archive_id=ANY requests=[ current:-2 ok:207 errors:11]
Weird bug in HSM (cont.)

Not a good idea to change `max_requests` if HSM is activated:

```
# cat /proc/fs/lustre/mdt/lustreHP-MDT0000/hsm/agents
uuid=21b44f0a-49eb-de43-99ff-99894552a6b3 archive_id=ANY requests=[current: 20 ok:195 errors:0]

● Increasing to 40 requests

# lctl set_param mdt.lustreHP-MDT0000.hsm.max_requests= 40
# cat /proc/fs/lustre/mdt/lustreHP-MDT0000/hsm/agents
uuid=21b44f0a-49eb-de43-99ff-99894552a6b3 archive_id=ANY requests=[current: 40 ok:200 errors:0]

● Reducing it to 20 requests

# lctl set_param mdt.lustreHP-MDT0000.hsm.max_requests= 20
# cat /proc/fs/lustre/mdt/lustreHP-MDT0000/hsm/agents
uuid=21b44f0a-49eb-de43-99ff-99894552a6b3 archive_id=ANY requests=[current: 60 ok:200 errors:0]

● After a minute, it blew up

# cat /proc/fs/lustre/mdt/lustreHP-MDT0000/hsm/agents
uuid=21b44f0a-49eb-de43-99ff-99894552a6b3 archive_id=ANY requests=[current: 173 ok:200 errors:0]
```
Future work...

- Data indexing?
- Out of band remote/public access to S3 objects?
- Local mirror of S3 public dataset?
Thank You Note

Our work was supported by HPE by the loaning of hardware to develop and test our solution on.

We relied on work contributed to Lustre by CEA
Source repo

github.com/ComputeCanada/lustre-obj-copytool