Challenges in Using Persistent Memory In Distributed Storage Systems

Dan Lambright
Storage System Software Developer
Adjunct Professor University of Massachusetts Lowell
Aug. 23, 2016
Overview

- Storage class memory (SCM)
- Distributed storage
- GlusterFS, Ceph
- Network latency
- Accelerating parts of the system with SCM
- CPU latency
Storage Class Memory

What do we know / expect?

- Near DRAM speeds
- Better wearability than SSDs
- Byte or block addressable (via driver)
- Likely to be expensive
- Fast random access
- Accessible via API (crash-proof transactions)
- Bottlenecks move elsewhere within system
- Support in Linux
The problem

Must lower latencies throughout system: storage, network, CPU

<table>
<thead>
<tr>
<th>Media</th>
<th>Latency</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDD</td>
<td>10ms</td>
<td>Slow</td>
</tr>
<tr>
<td>SSD</td>
<td>1ms</td>
<td>Wears out</td>
</tr>
<tr>
<td>SCM</td>
<td>&lt; 1us</td>
<td>Cost</td>
</tr>
</tbody>
</table>
Distributed Storage

Why use it?

- Single server (NFS) scales poorly
- Benefits of distributed storage
  - “scale out” to 1000s of nodes
  - Single namespace
  - Minimal impact on node failure
  - Good fit for commodity hardware
Case Studies
GlusterFS

- Primarily used as a file store
- Combines multiple file systems into a single namespace
Case Studies

Ceph

- Popular in OpenStack
- Block, object, file
- RADOS as intermediate representation
Framing The Problem

What to analyze

- Plethora of workloads and configurations
  - HPC, sequential, random, mixed read/write/transfer size, etc
  - # OSDs, nodes, replica/EC sets, ...

- Benchmark one
  - (e.g OSD/core)
  - Storage is memory /dev/pmem
  - Single workload 4K RW - larger transfers see better benefit with RDMA
NETWORK LATENCY
Replication

Latency to copy across nodes

- “Primary copy”, update replicas in parallel, processes reads and writes
  - Gluster’s forthcoming “JBR”
- “Chain”, forward writes sequentially, updates reads at tail
  - (tail sends ACK to client, so fewer messages, more latency)
- Ceph uses “splay” replication, combining parallel updates with reads at tail
Client vs Server-side Replication

Client fan-out uses more client side bandwidth; it’s likely client has slower network than server

Server side requires extra hop - adds to latency
Consistency in Ceph

- Reads following writes
  - Only return most recently committed data
  - May see bottleneck at tail
- Writes to different objects (but same PG) are serialized
  - PG size configurable online
  - But, each PG uses resources
Improving Network Latency

Techniques

- 2X replication
  - If MTBF for SCM is better than rotational storage
- Coalescing operations
  - Observed 10% improvement in small file creates on gluster
- Pipelining
- Better hardware
  - RDMA (helps larger transfers)
  - Increase MTU
ACCELERATION
Improve Parts of System With SCM

Heterogeneous storage

- Tiering
- Ceph Filestore journal
- Ceph Bluestore Write ahead log
- DM cache
- XFS journal
In Depth: Gluster Tiering

Illustration of network problem

- Heterogeneous storage in a single volume
- Fast/expensive storage caches slower storage
- Introduced in Gluster 3.7
- Fast “Hot tier” (e.g. SSD, SCM)
- Slow “Cold tier” (e.g. erasure coded)
- Cache policies:
  - All data placed on hot tier, until “full”
  - Once “full”, data “promoted/demoted” based on access frequency
Gluster’s “Small File” Problem

Analysis

- Tier helped large I/Os, not small
- Pattern seen elsewhere ..
  - RDMA performance tests
  - Customer Feedback, overall GlusterFS reputation ...
- Profiles show many LOOKUP round trips
- Conclusion: LOOKUP RTT dominates faster data transfers
  - the problem is exacerbated with SCM
Understanding LOOKUPs in Gluster

Problem: Path Traversal

- Each directory in path is tested on open
  - Existence tests
  - Permission tests
Understanding LOOKUPs in Gluster

Problem: Coalescing Distributed Hash Ranges

- Client side replication and distributed hash computation
- “Layout” definition
- Layouts are split across nodes
- Each node checked on every Lookup to get the full picture
- Must confirm each file up to date
  - File moved
  - Node membership changes
LOOKUP Amplification

d1/d2/d3
Three LOOKUPs
Four servers
12 LOOKUPs total in worse case
For a single I/O
Client Metadata Cache

Gluster’s md-cache translator

- Cache Gluster’s per-file metadata at client
- Enhancements under development to cache longer
- Invalidate cache entry on another client’s change
  - Change to layout
- Invalidate intelligently, not spuriously
  - Some attributes may change a lot (crime, atime, ..)
CPU LATENCY
CPU problem

Services needed to distribute storage add to CPU overhead

- Data distribution over nodes
- Replication + ec over nodes
- Single namespace management
- Conversion between external and internal representation
In Depth : Ceph Datapath

Analysis

- Upper (fast) and lower (slow) halves of I/O path
- Context switch between halves
- Memory allocation matters (Jmalloc)
Community Contributions
SanDisk, CohortFS, others

- **SanDisk**
  - Sharded work queues
  - Bluestore optimizations (shrink metadata, tuning RocksDB)
  - Identified TCMalloc problems, introduced JEMalloc
  - .. more.. ongoing

- **CohortFS (now Red Hat)**
  - Accelio RDMA module
  - Divide and Conquer performance analysis using memstore
  - Lockless algorithms / RCU (coming soon)
Bluestore
Key-value database as store

- Motivation
  - Transactions difficult to implement with posix
  - Ceph journal necessitated double writes
  - Object enumeration inefficient

- Why a database?
  - ACID semantics for transactions
  - Efficient storage allocation (formally managed by fs)
Bluestore

- Shorter code path helps latency
- No longer traverse XFS file system
- RocksDB used
Some results

Code in flux - YMMV!
Some results

Code in flux - YMMV!
Bluestore
Hardening performance

- Near term improvements
  - Sharded extents, not all in one omap (so 4K random reads won't incur large metadata writes)
  - Tune RocksDB compaction options
- Seek alternative to RocksDB?
  - LSM style optimizes for sequential access
  - Incurs periodic background compaction, write amplification, ...
  - Instead, try SanDisk’s ZetaScale?
SUMMARY
Summary

- Distributed storage poses unique problems with latency.
- Network
  - Reduce round trips by streamlining, coalescing protocol, etc
  - Cache at client
- CPU
  - Keep shrinking the stack
  - Run to completion
- Consider
  - SCM as a tier/cache
  - 2x replication
THANK YOU

plus.google.com/+RedHat
linkedin.com/company/red-hat
youtube.com/user/RedHatVideos
facebook.com/redhatinc
twitter.com/RedHatNews