Gateway LSVD Title TBD

Isaac Khor
Northeastern University

Sumatra K Dhimoyee
Boston University

Timothy Borunov
Boston University

Peter Desnoyers
Northeastern University

Orran Krieger
Boston University

Abstract

Problem: 1. disaggregated storage is heavy on bandwidth, and 2. writes are very expensive due to requirements around consistency and replication

Our solution: log-structure to batch writes, gateway for replication and backwards compat and other benefits

1 Introduction

- Distributed networked storage has high demands on bandwidth and on the storage medium

- Traditionally solved with a write-back cache, but that makes state management hard

- Small writes are incredibly costly, you need consistency/replication semantics on every request

- We propose a log-structured virtual disk on a central storage gateway, on top of immutable objects

Benefits:

- We get statistical multiplexing between different clients
- Built on top of immutable objects = trivial caching
- Batched writes for good write performance
- Snapshots and clones are trivial
- Elastic backend to prevent low-space GC thrashing

2 Background and Motivation

Explain the architecture of:

- How much of LSVD do we explain here?

2.1 Semantics

- Don't lose data - remote journal in a different failure domain

- Sources of overhead: - IO amplification per write request

- Replication consistency invariants to maintain - Data consistency between writes – maintain ordering

- Solved with batched writes (with write journal) - Doesn't this just move the problem to our gateway instead?

- Backend representation is always self-consistent - we never present a non-consistent view of the disk at any point in time

2.2 Ceph RADOS Block Device

- Fairly simple implementation

- Image is divided up into xMB chunks, each of them mapped to a RADOS object

- Writes just write to that object, reads read from the object

- Clones and snapshots are COW for that object

- Since pools are triple-replicated, incredible amonuts of amplification

- Consistency for the three copies, cost to maintain consistency is high

2.3 OpenStack Cinder

- Metadata stored in a DB

- Data is backend agnostic, can be openstack swift or ceph rados?

- Architecture unclear TODO

- ??? does anybody know how this thing works

2.4 Others

- What we know about other proprietary elastic block stores

3 Architecture

- Log structured on immutable backend objects

- Present as librbd, on top of spdk

- Remote write log, can be in-memory, on nvme, or on multiple nvmes

- Greedy GC

- Clones are forks in history

- Centralised cache on the gateway, clones share base objects

4 Evaluation

- 6 machine cluster
  - 4 Ceph osd hosts, 1 gateway, 1 client
  - 48 HDDs 7200rpm, 16 SSDs - find out aggregate backend bandwidth
  - Plain qemu vms

5 Results

- microbenchmarks: fio randread/write, vary bs, qd, - filebench
  - compare lsvd/rbd - theoretical upper limit with local nvme
  - latency histograms? bimodal of hit/miss should show up somewhere - vm boot results - image optimization

  Notes on figures themselves:
  - Separate by workload instead of config – ex one fileserver graph with all the configs instead of multiple with fileserver

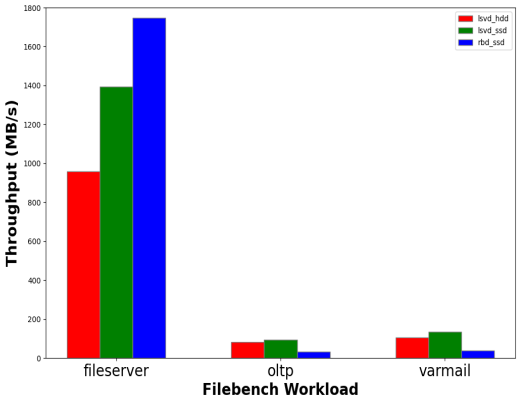

Fig1. File benchmark result with 20GB Cache and 80GB Volume.

- The hdd with lsvd is comparable to rbd and lsvd with ssd

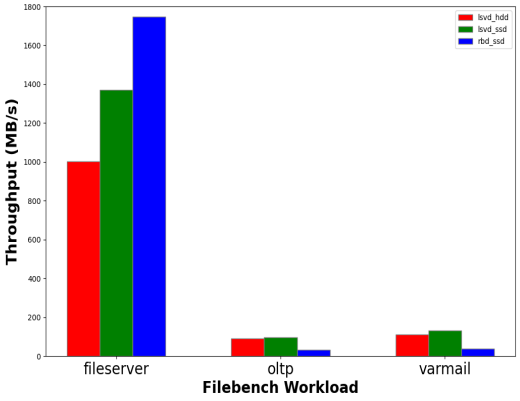

Fig2. File benchmark result with 240GB Cache and 80GB Volume.

- The result is similar to small cache (should we keep both?)

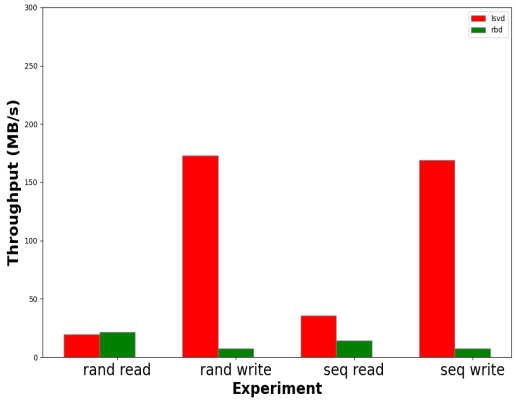

Fig3. Micro benchmark result with 20GB Cache, 80GB Volume and Hard Disk Backend.

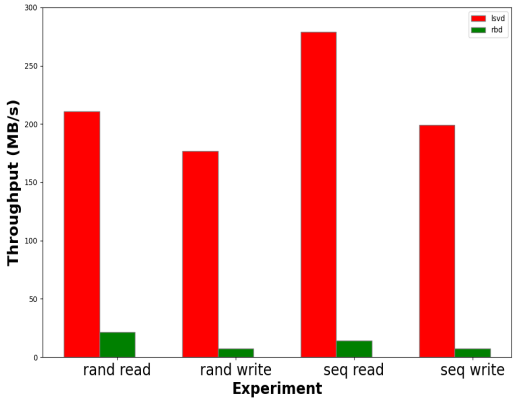

Fig4. Micro benchmark result with 240GB Cache, 80GB Volume and Hard Disk Backend.

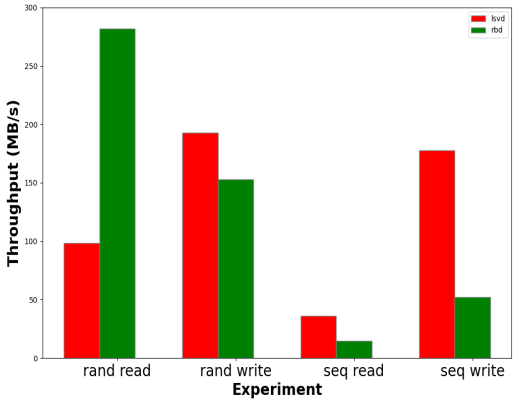

Fig5. Micro benchmark result with 240GB Cache, 80GB Volume and SSD Backend.

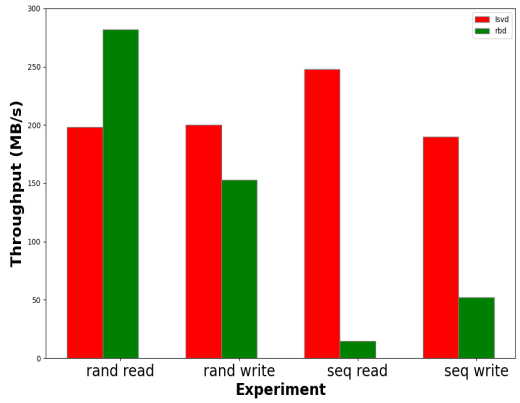

Fig6. Micro benchmark result with 20GB Cache, 80GB Volume and SSD Backend.

– more results to be added–

6 Conclusion

- ?

7 Related Works

- LSVD
  - RBD

Acknowledgments

- Funding source? - IBM/Ceph?

Availability

USENIX program committees give extra points to submissions that are backed by artifacts that are publicly available. If you made your code or data available, it's worth mentioning this fact in a dedicated

The artefacts of this project are publicly available at the GitHub repository.

[[Isaac: redact this for the review]]

References