CephFS

A Filesystem for the Future

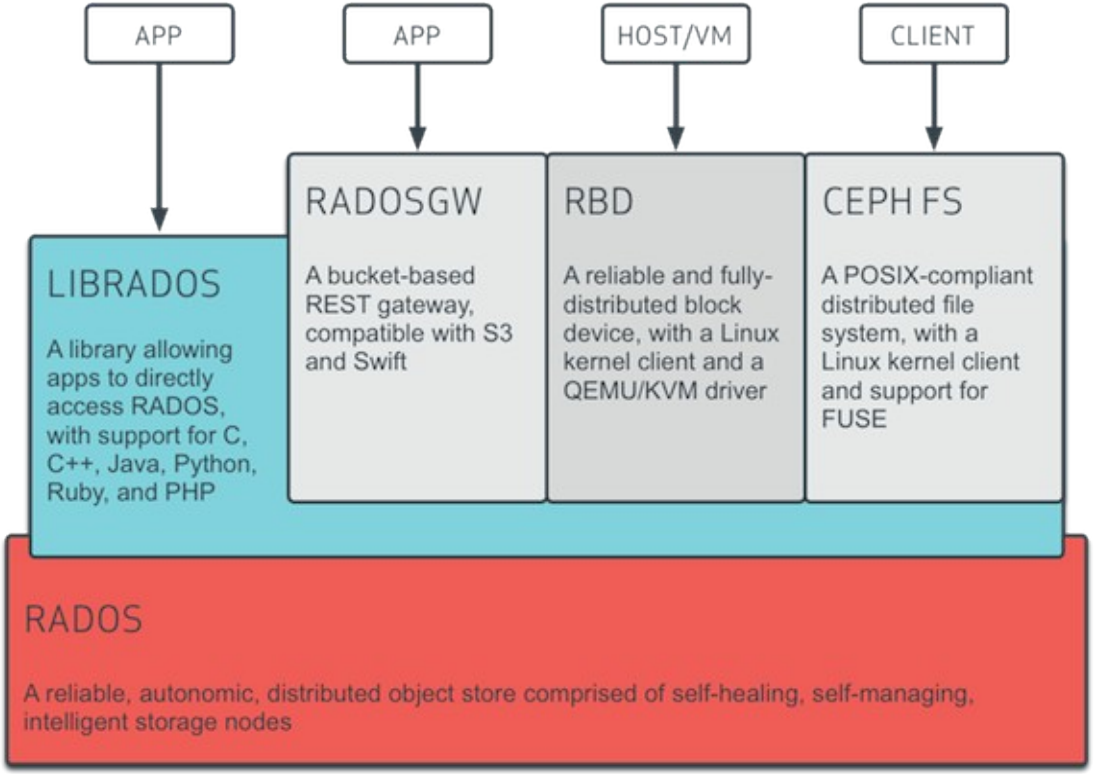
David Disseldorp
Software Engineer
ddiss@suse.com

Jan Fajerski
Software Engineer
jfajerski@suse.com

Introduction to Ceph

- **Distributed storage system based on RADOS**
  - Scalable
  - Fault tolerant
  - Relatively performant
  - Self-healing and self-managing
  - Runs on commodity hardware
- **Various client access mechanisms**
  - Object storage
  - Block device
  - POSIX compatible filesystem
  - Your application

Introduction to Ceph



Introduction to CephFS

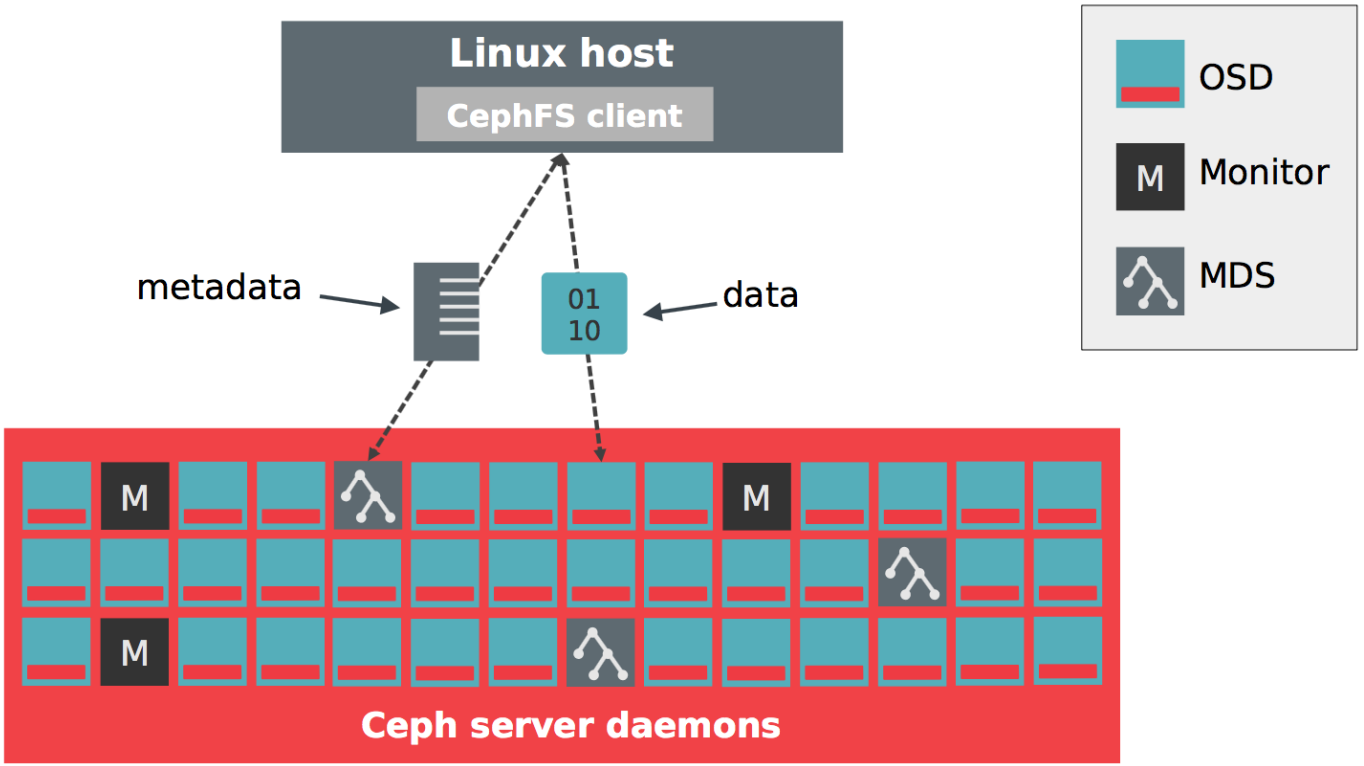
- **POSIX like clustered filesystem atop Ceph**

- **File access remains important in storage**
  - Allow existing applications to utilize Ceph storage
  - Interoperability with existing infrastructure
  - Directories and permissions
  - Elastic capacity

Architecture

- **Object Storage Daemon (OSD)**
- **Monitors (MON)**
- **Metadata Server (MDS)**
  - Manages filesystem namespace
  - State stored within RADOS cluster
  - Active/Standby
    - Standby MDS steps in on failure of primary
  - Active/Active
    - Not currently supported
    - Sharding of directory tree
- **Client**
  - Communicates directly with OSD and MDS daemons

Architecture



Deployment

Using ceph-deploy...

```
ceph-deploy mds create mymds
ceph osd pool create fs_data
ceph osd pool create fs_metadata
ceph fs new myfs fs_metadata fs_data


mount -t ceph <mon_ip>:6789/ /mnt/ceph
```

...or automated with our salt-based deployment tool DeepSea!

Configuration

- **Nothing to do for basic use, even with multiple MDSes**
- **Handle failover parameters:**
  - *mds_beacon_grace*
  - *mds_standby_for_X*
  - *mds_standby_replay*
- **Basic caching parameters:**
  - *mds_cache_size*
  - *client_cache_size*

Configuration

- **Settings applied at run-time via extended attributes**
- **File layout**
  - Specify which RADOS pool should be used for data storage
  - Define how files are striped across RADOS objects
- **Quotas**
  - File size and count limits
  - Only enforced by the fuse client

Clients

- **In-kernel CephFS client**
  - *mount.ceph*
- **FUSE**
- **Libcephfs**
  - NFS Ganesha
  - Samba
  - Your application

NFS Ganesha

- **NFS server in user-space**
  - Comprehensive protocol support: V2, V3, V4, v4.1, v4.2
  - Pluggable back-end for filesystem specific functionality
- **CephFS back-end (FSAL)**
- **Technical Preview with SUSE Enterprise Storage 4**

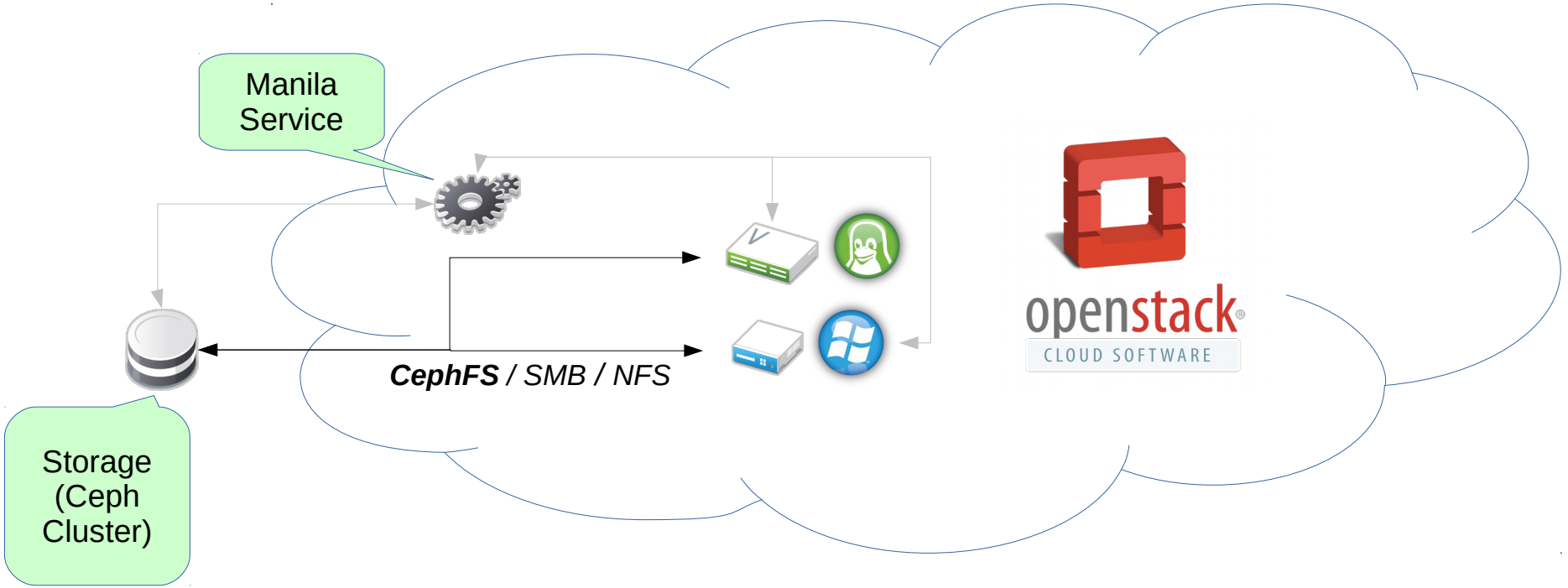
Samba

- **Windows interoperability suite**
  - File sharing, authentication and identity mapping
- **Ceph module for Samba**
- Access CephFS from any SMB client
  - Windows, macOS, etc.
  - Enabled via smb.conf parameter: *vfs objects = ceph*
- **Coming soon!**

Openstack Manila

- **FSaaS for cloud deployed virtual machines**
- **Management and provisioning of file shares**
  - Independent of underlying file server and data-path
  - Back end file server specific drivers
- **Clients restricted by path, pool and namespace**
- **Quotas**
  - Defined and used to advertise available space

Openstack Manila



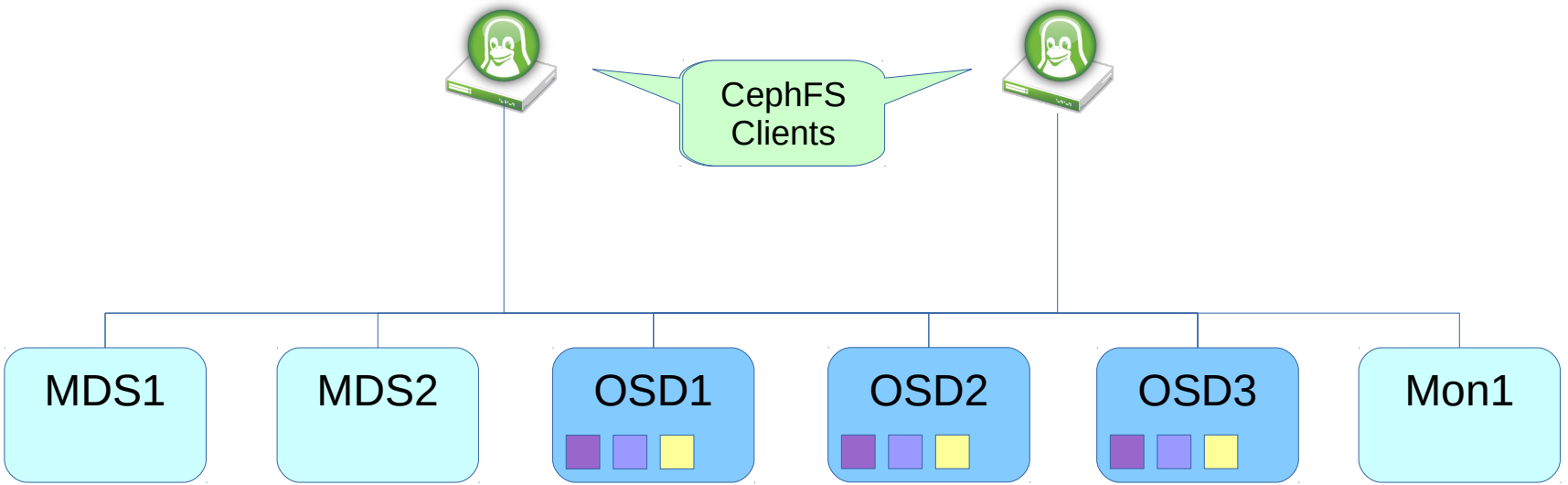
Client Caching

- **CephFS protocol allows for aggressive client caching**

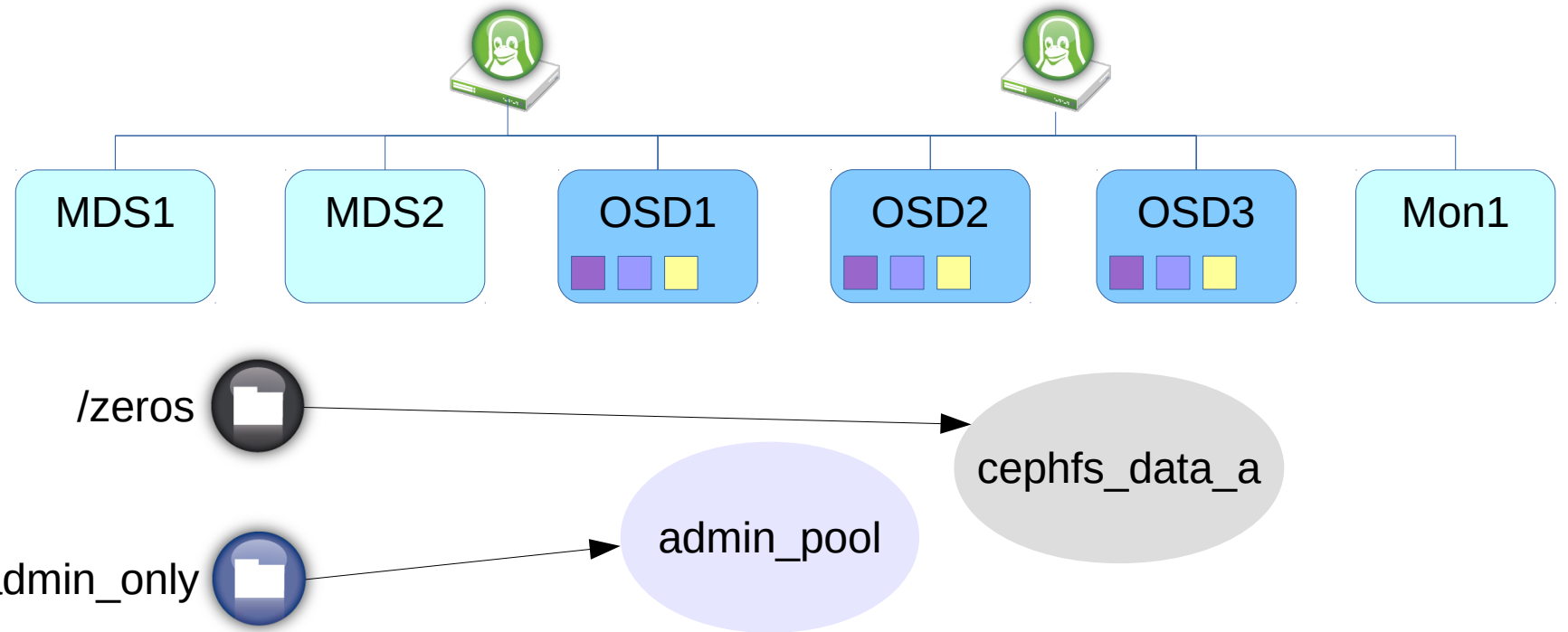
- **Fine grained**
  - Data and/or metadata
- **Shared**
  - Cache existing state
- **Exclusive**
  - Change state, and buffer new state
- **MDS can grant and revoke client cache *capabilities***

Demonstration

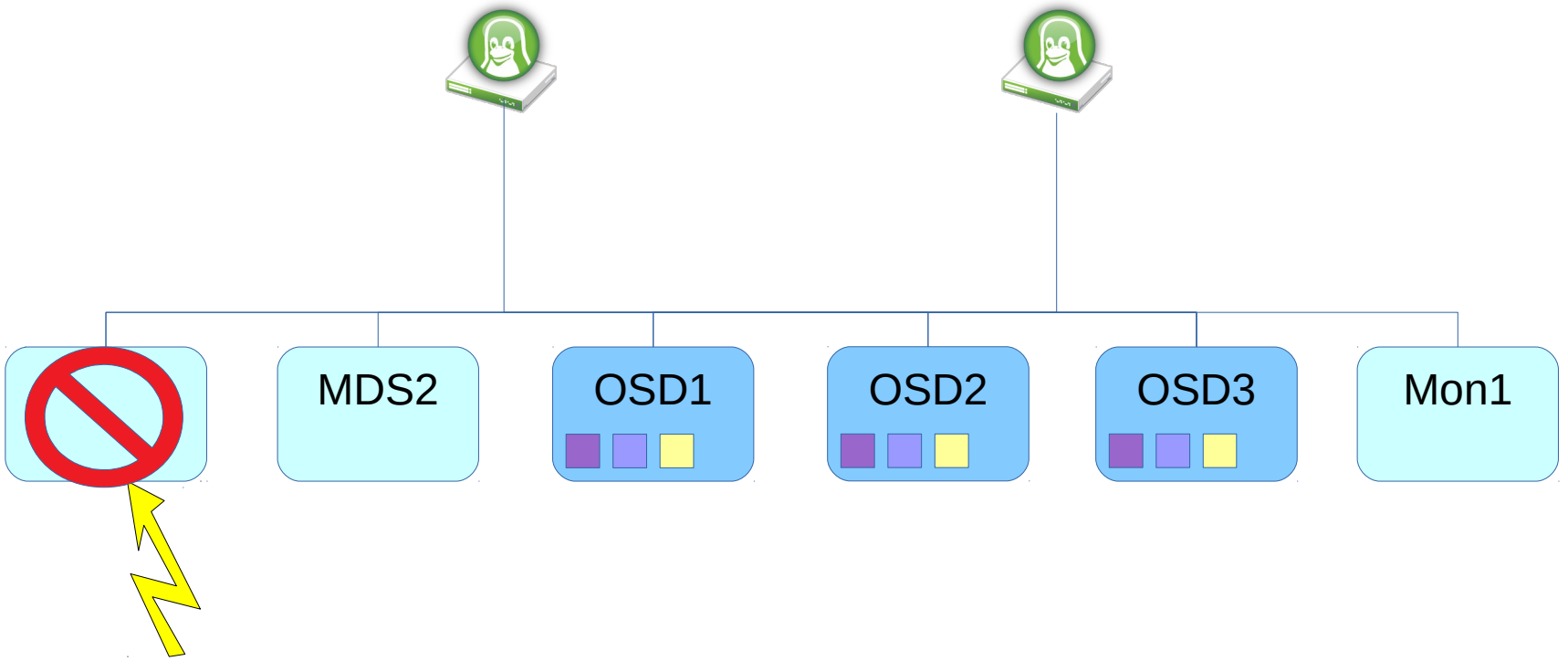
Demonstration



Demonstration: Layout



Demonstration: MDS Failover



Operational considerations

Sizing

- **Hardware recommendations**
  - Highly dependent on workload
  - Multi core CPU with high single core performance
  - Two NICs – at least 1 Gbps
  - RAM - the more, the better as a general rule (but at least 1GB per daemon)

- **Multi-role nodes**
  - MDS on Monitor / OSD nodes
  - Workload dependend – if in doubt, don’t do it
  - Monitor nodes for resource usage

Benchmarks

Thanks David Byte!

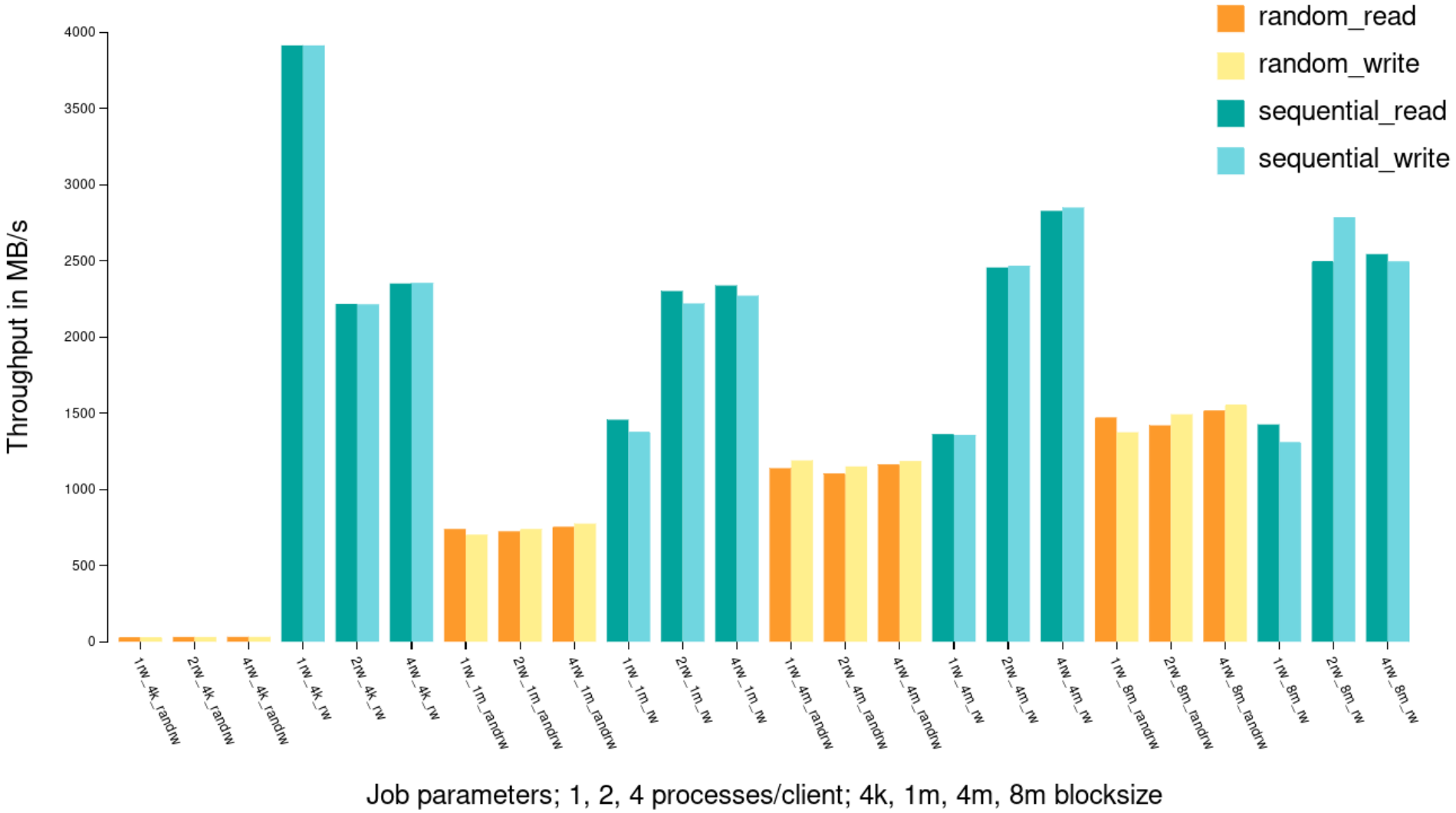
Setup

- **Ceph on 6 nodes**
  - 3 OSD/MON nodes – 24 cores – 320 GB RAM
  - 3 OSD/MDS nodes – 48 cores – 320 GB RAM
  - 24 OSD daemons per node – SSD journals
- **3 client nodes**
  - 40 cores – 64 GB RAM
- **Network interconnect via 2 bonded 40 Gbit/s interfaces per node**
- **Out-of-the-box deployment**

Experiment

- **Read/write one large file**

Accumulated Random and Sequential Throughput - 3 clients - 1 large file



Troubleshooting

- **Collection of tools developed by John Spray**

- ***cephfs-journal-tool***
  - Debug and repair CephFS journals
- ***cephfs-data-scan***
  - Reconstruct metadata based on data pool input
- ***cephfs-table-tool***
  - Dump or reset session, snapshot or inode tables

Future Challenges

- **Active / Active MDS deployments**
  - Serious known bugs
  - Test coverage improvements needed
- **Metadata scalability**
  - Directory metadata occupies one RADOS object
  - Better support for large directories through sharding to multiple objects
- **Multiple filesystems**
  - Can be mitigated with cephx and path/pool restrictions
- **Snapshots**
  - Insufficient test coverage
  - Not working with multiple file systems
  - Not working with active / active MDSes

Questions?

References

- https://jtlayton.wordpress.com/2016/09/01/cephfs-and-the-nfsv4-change-attribute/
- http://docs.ceph.com/docs/master/architecture/
- https://github.com/SUSE/DeepSea
- https://events.linuxfoundation.org/sites/events/files/slides/CephFS-Manila-0.2_0.pdf
- https://events.linuxfoundation.org/sites/events/files/slides/nfs-ganesha-glusterfs.pdf