How Not To Use a Cluster

Petabytes

- Managed 10's of petabytes of POSIX storage
- The goal was to be completely hands off
- How do we let everyone have it their way?
  - Ceph and Gluster support many different types of deployments. Each with their own unique quirks
    - All flash? Sure. All spinning rust? Sure
    - Mix and match? Why not

Tools of the Software Storage Admins

- Bash, Pdsh, rsync, python, C, gdb, chef/salt
- Ask lots of questions before adding more load to the shared cluster
  - You want to write how many TB's?
  - Only read once in awhile you say?
  - You project it probably won't be more than a few TB?

Backing up petabytes

At what point are backups useless?

Strategies

- Shift the burden to the client. Can you write twice?
- rsync hell. Works but sometimes hangs the cluster
  - Cron job once per minute? What could go wrong
- geo replication multimaster - Still being worked on by Redhat
- How much bandwidth do you have again?
- What's the turnover rate?

Software storage advantages

What does Gluster and Ceph excel at vs traditional?

- Cheap servers
- Near linear throughput scaling
- Big files
- Cost. Invest the savings in engineers to extend it as needed

Software Storage Disadvantages

- Small I/O (stop writing 4KB to the cluster!)
  - Enterprise storage will probably always win here
  - You could add flash but you still pay the network round trip penalty
- Fibre channel support
- Some operations feel slow. ls -l in particular

Distributed Storage Has Unique Challenges

- Who's stomping the cluster?
  - Maybe you should have your own playground
- Small files
- ls -l readdir_plus
- Fan out of NFS vs fuse.
- IO operations pilling up
- One server can take down the cluster?
- Cron job the defragmentation

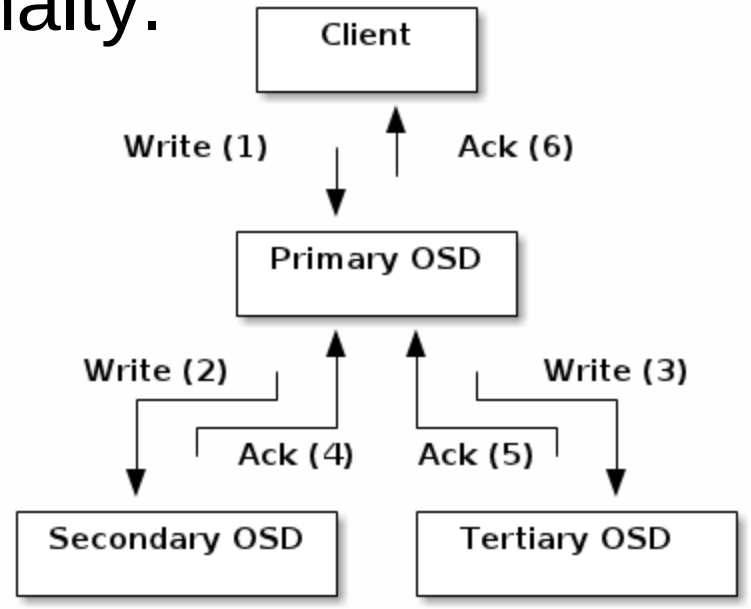
- Needed tools
  - Central logging
  - Charts
  - Real time stats
  - Distributed SSH

Small Files

<128KB is not optimal

Why? Network round trip penalty.

NFS mounts with small read and write sizes



Gluster Use cases

- Database Backups - Throughput requirements can be challenging
- Write once in awhile, read from 100's of servers. AI training data
- Git?
- 1000's of servers sharing data
- Home directory mounts
- Laptop backups for engineers

Git on Gluster

Git assumes it is on a FS that supports atomic operations. Atomic rename
Gluster tries hard but Git can produce a race condition that causes split brain
Even with 3 replicas this fails

Rsync and Gluster

cron job once per minute of an rsync job
Runs on NFS Gluster mount
Gluster experiences a problem that hangs the mounts
rsync piles up on clients and hammers the cluster when it comes back

The Cluster is 100% Full

I have seen Gluster fill up completely

- rm -rf /mnt/gluster/* ?
- Someone’s files need to get dumped overboard

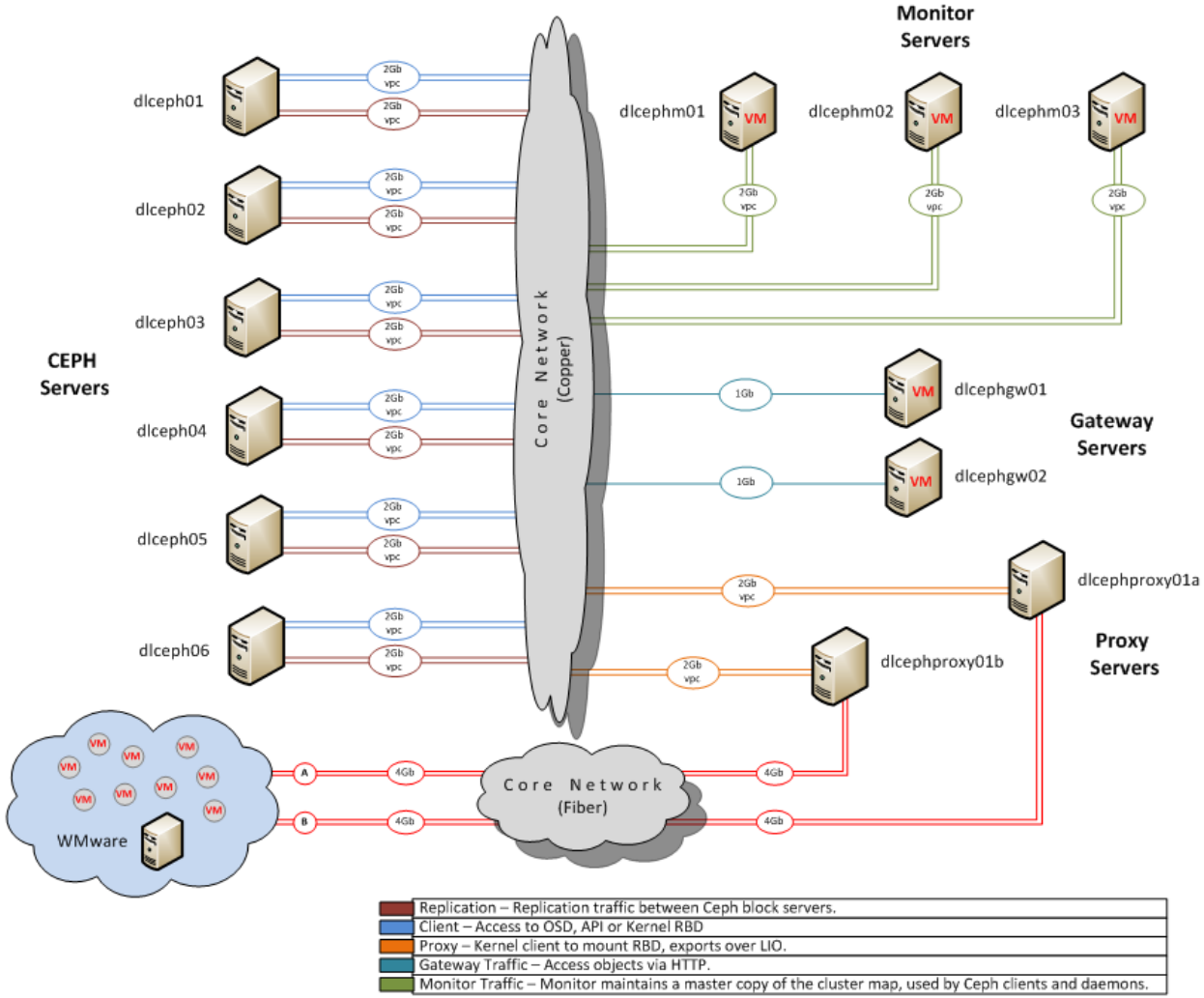
Gluster actually recovers well

Retire the Enterprise Storage

Ceph’s bread and butter is the rados block device
So I can retire my hitachi right?

- Not so fast. Depends on your workload

I decided to do just that at Corporation Service Company
Target CLI is not easy to script and fairly brittle


Monitor Servers
dlcephm01
dlcephm02
dlcephm03
VM
2Gb vpc
CEPH Servers
dlceph01
dlceph02
dlceph03
dlceph04
dlceph05
dlceph06
Core Network (Copper)
1Gb
dlcephgw01
dlcephgw02
Gateway Servers
dlcephproxy01a
dlcephproxy01b
Proxy Servers
WMware
A
B
4Gb
Core Network (Fiber)
Replication – Replication traffic between Ceph block servers.
Client – Access to OSD, API or Kernel RBD
Proxy – Kernel client to mount RBD, exports over LIO.
Gateway Traffic – Access objects via HTTP.
Monitor Traffic – Monitor maintains a master copy of the cluster map, used by Ceph clients and daemons.

Active Active Crashover

Several crashes later we found that LIO (Linux-IO Target) could not handle Ceph's latency. Specifically over Fibre Channel
Mike Christie @ Redhat has taken up the torch to modify LIO's libraries to support Ceph

Questions?