

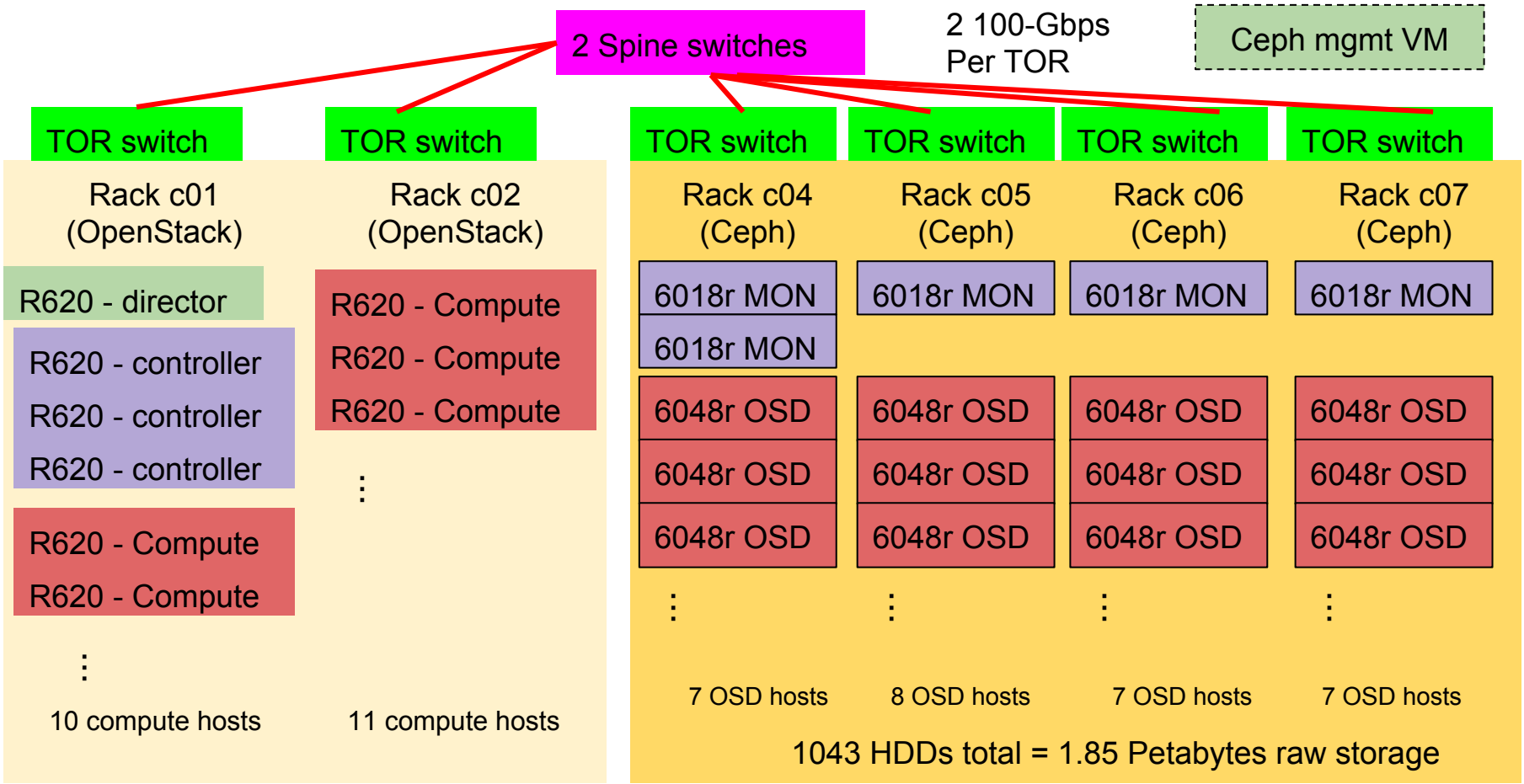
Ceph-OpenStack Deployment at Scale

Perf & Scale Engineering, 1/18/2017

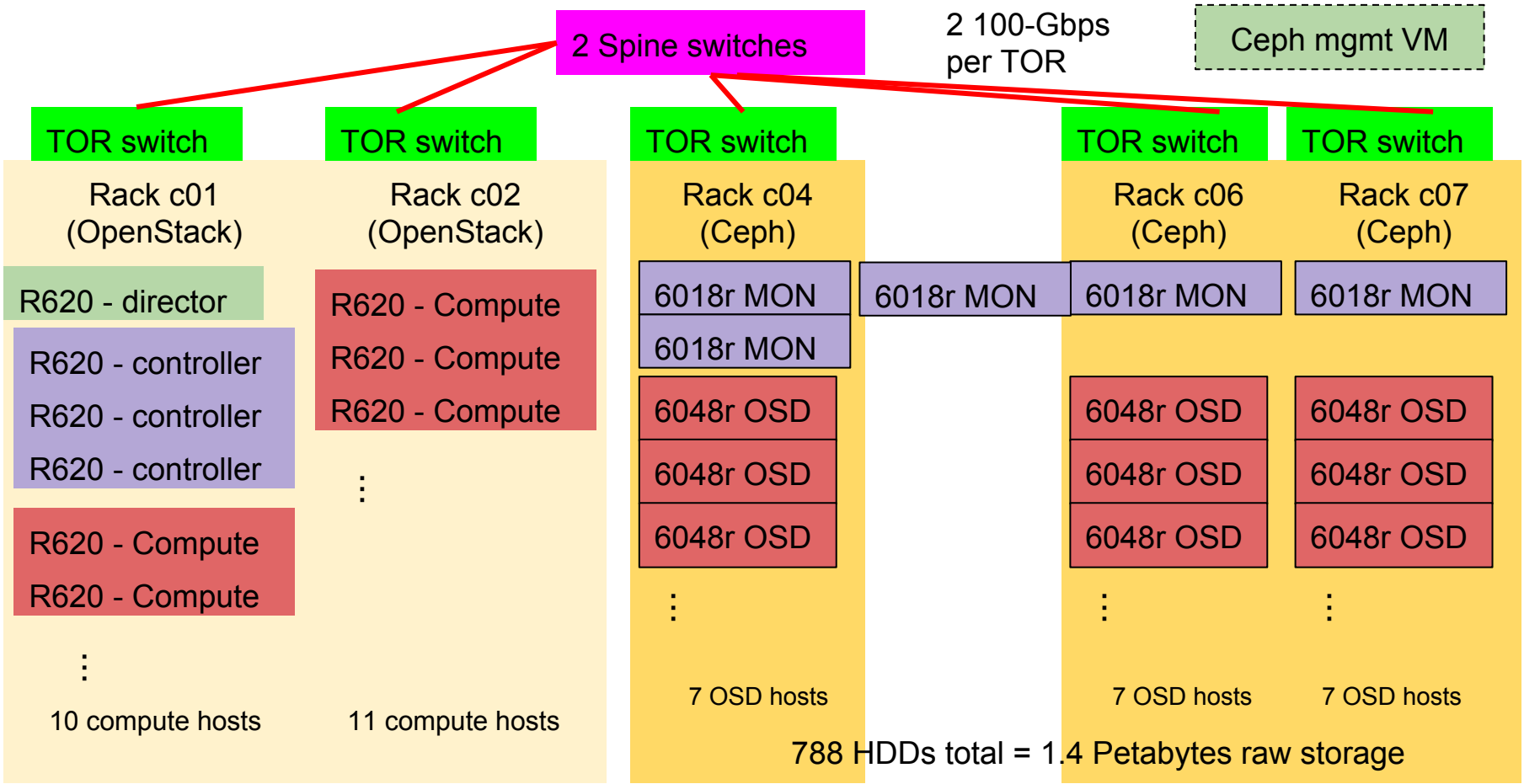
Topics

- What was tested
- Scale related client-side problems
- New methodology - fio histogram logging
  - High max latencies monitored with this
- Monitor problems when network partitioning
- Spread of OSD utilization too high

Physical hardware view - racks and switches



Physical hardware view - racks and switches



Versions

- RHCS 2.0 = Ceph Jewel
- RHEL OSP 10 = OpenStack Newton
- RHEL 7.3
- Fio 2.14

Deployment time for Ceph

- Ignoring OS deployment time (QUADS)
- RHCS “pre-flight” script < 5 min
- Ceph-ansible about 1 hour (using fanout feature of ansible)
- OSP 10 deploy ran into problems (below) with external Ceph cluster

Tests Performed

- Create a storage pool that consumes approximately 50% of physical space
- Measure throughput and latency as a function of guest count
- Throttle back workload to 80% of capacity and
  - Remove & replace: monitor, HDD, and OSD host while workload runs
  - Measure latency percentiles as a function of time while this happens

Problems we've found (and worked around) so far

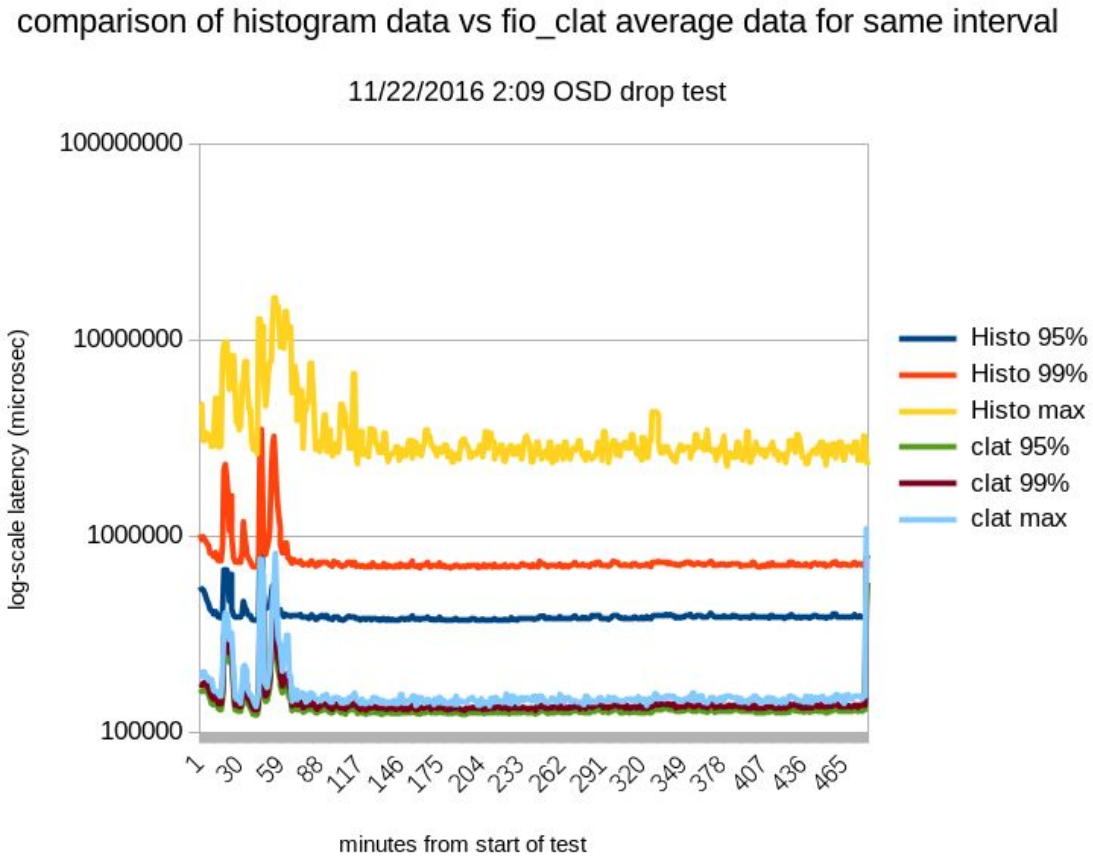
Bug id	status	description
Bz 1388119	resolved	Can't snapshot image from nova instance (on OSP10)
Tr 17573	open	Librbd hangs because of failure to create socket
Bz 1384079	Fixed in Newton	Openstack OOO client disregards ceph keys from YAML
Bz 1375378	On portal for RHCS 2.1	Document ceph-ansible method for host with different block device names
Bz 1389502	Fixed in Newton	Director should increase kernel.pid_max on ceph-backed compute hosts
Bz 1389503	Fixed in Ocata	Director should increase libvirtd FD limits on ceph-backed compute nodes

Bz = bugzilla.redhat.com number
Tr = tracker.ceph.com number

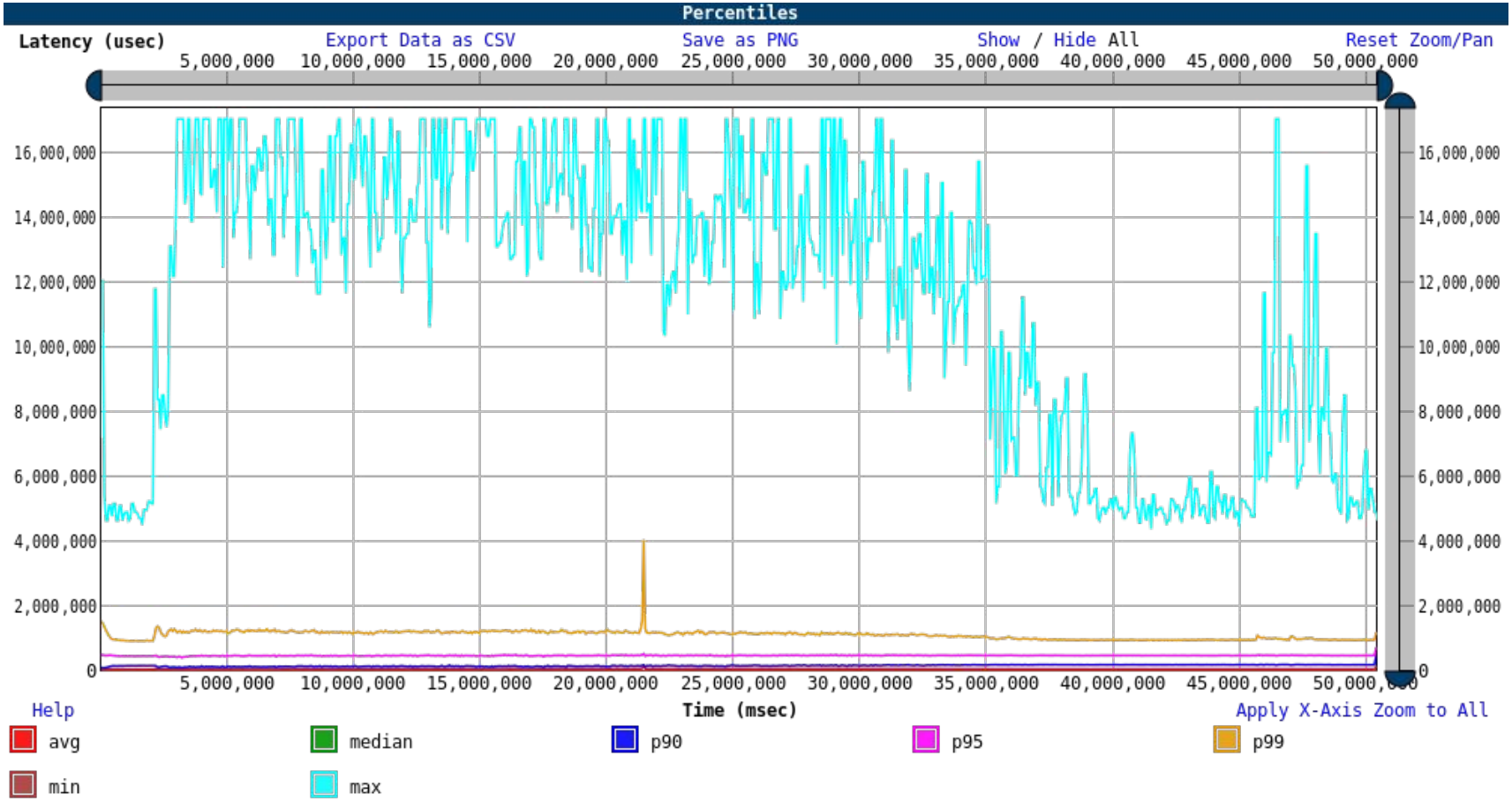
New methodology - fio histogram logging

- Developed by Karl Cronburg with some help from me last summer
- fio --write_hist_log=1 --log_hist_msec=N
- periodically logs latency histogram to fio_clat_hist.*.log
- fiologparser_hist.py merges per-thread histogram logs and computes percentiles for entire cluster (or any subset of interest)
- Much more accurate version of latency percentiles than fiologparser.py for the same time interval, particularly for long tests
- More scalable because we can use longer intervals
- At present, all latencies >= ~17 seconds in the last histogram bucket
    - Can be corrected by adjusting **FIO_IO_U_PLAT*** params in fio/stat.h

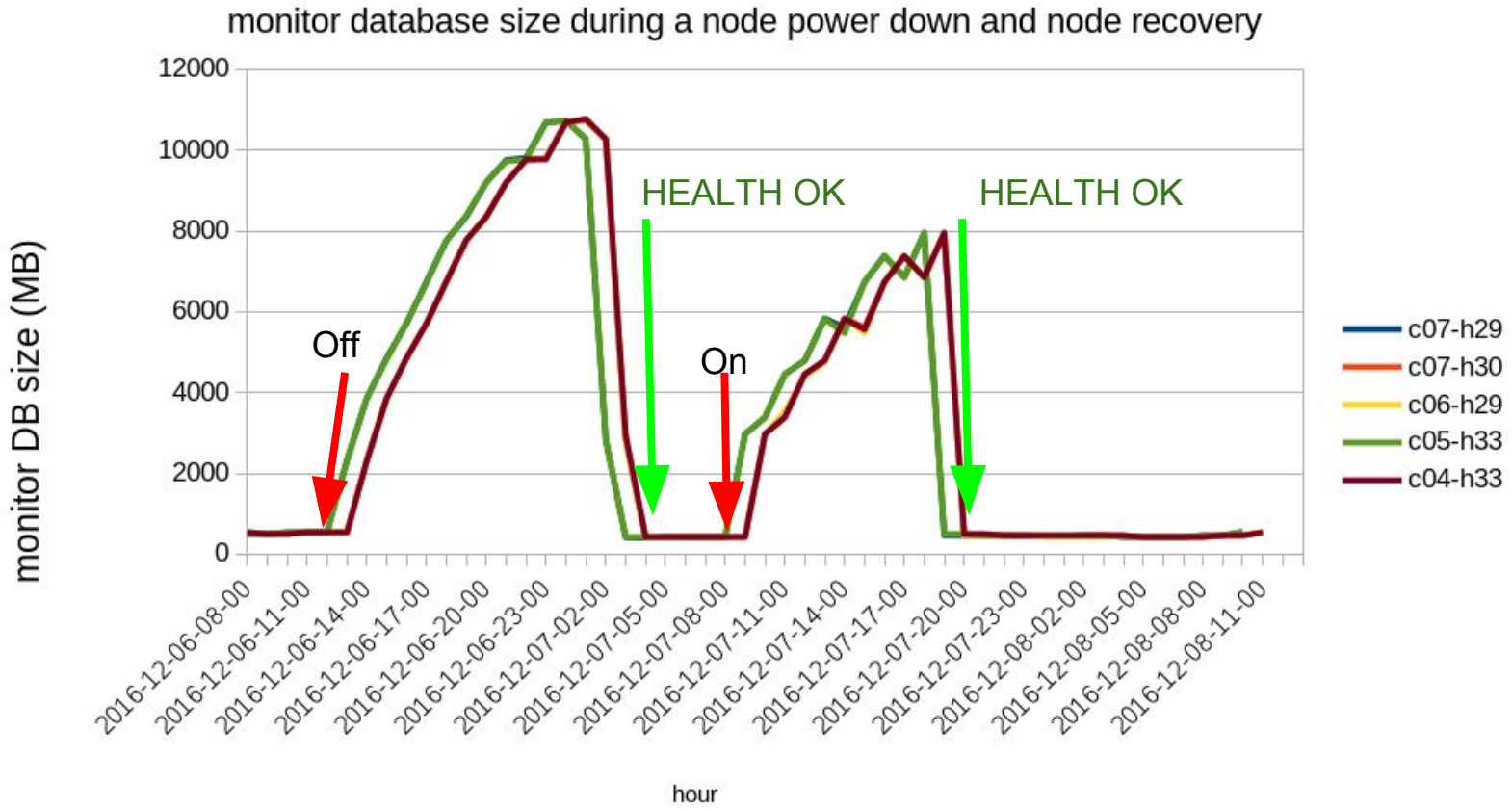
why latency with histograms



OSD node power-down - impact on I/O latency



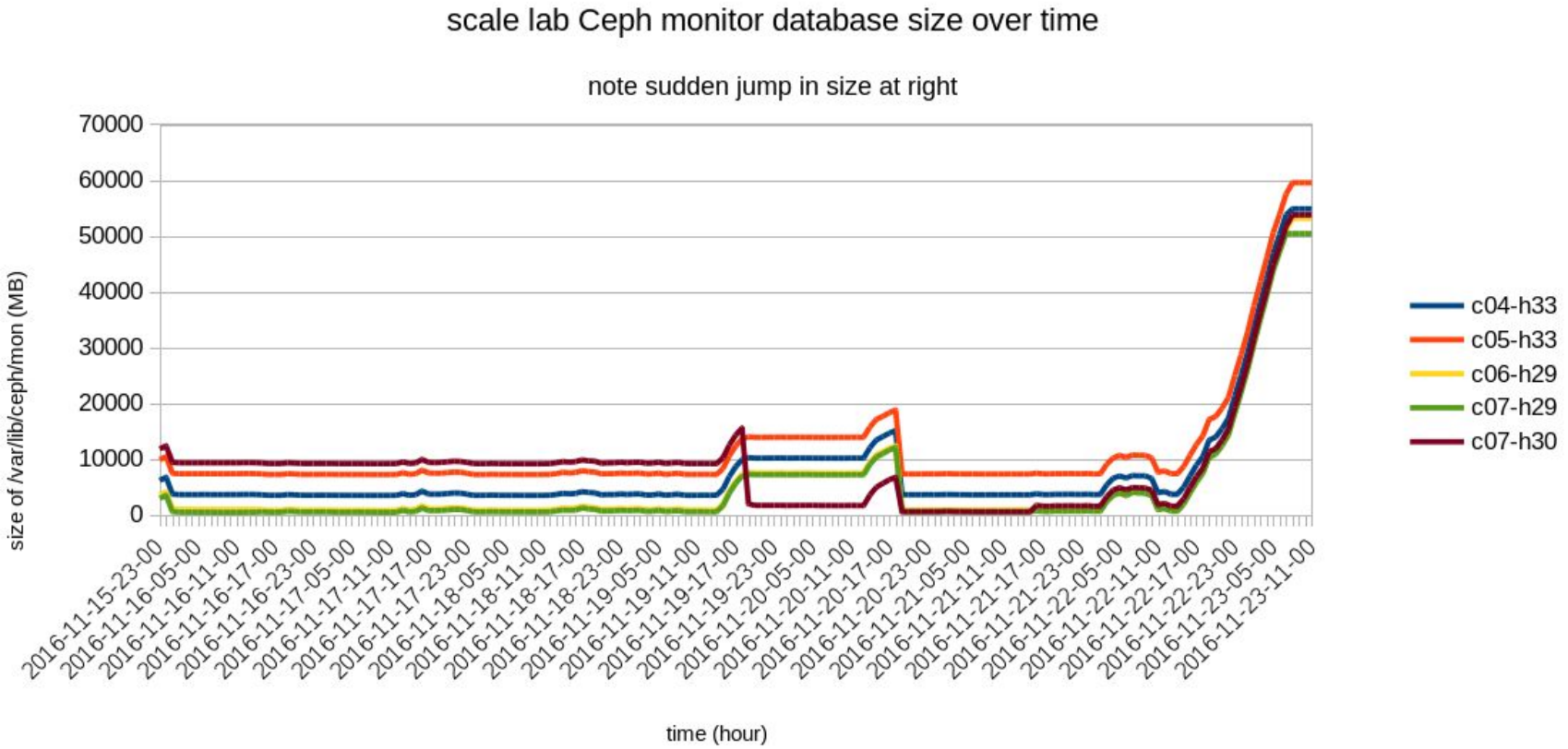
Node power down and recovery - monitor DB size



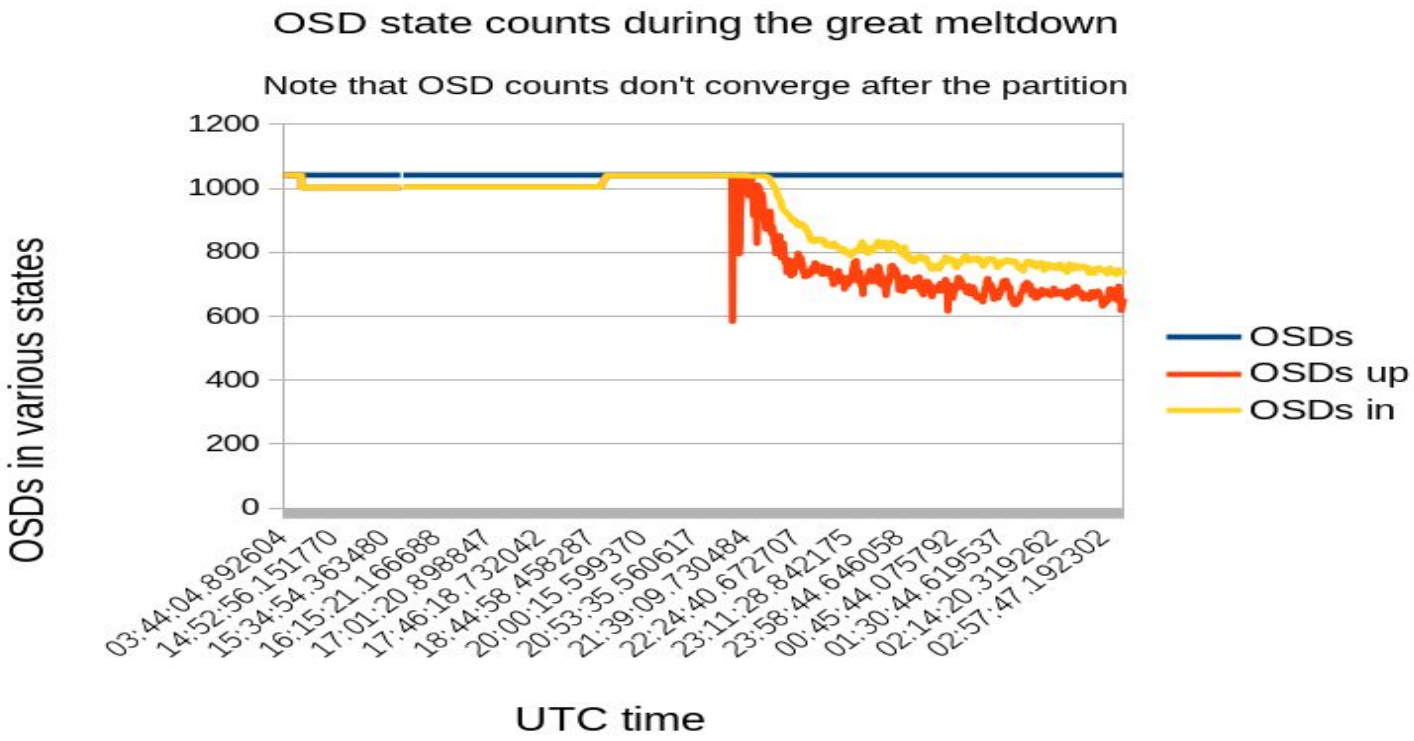
Day of the *Meltdown - a scale lab horror epic*

- Scale lab dynamically provisions VLANs for cloud tenants
- Juniper 5200 switches have some problem with VLAN provisioning, causing network partitioning
- Ceph monitors are cut off from each other and from the OSD hosts where the data resides
- OpenStack guests can’t access Ceph data on their Nova or Cinder volumes
- 5 hours ahead of schedule, 7 OSD hosts in rack c05 are removed
- Ceph tries to recover and can’t, filling up monitor database with state information and eventually overflowing system disk, monitors shut down, and Ceph loses quorum

What happened to the monitor database



What happened to OSD count



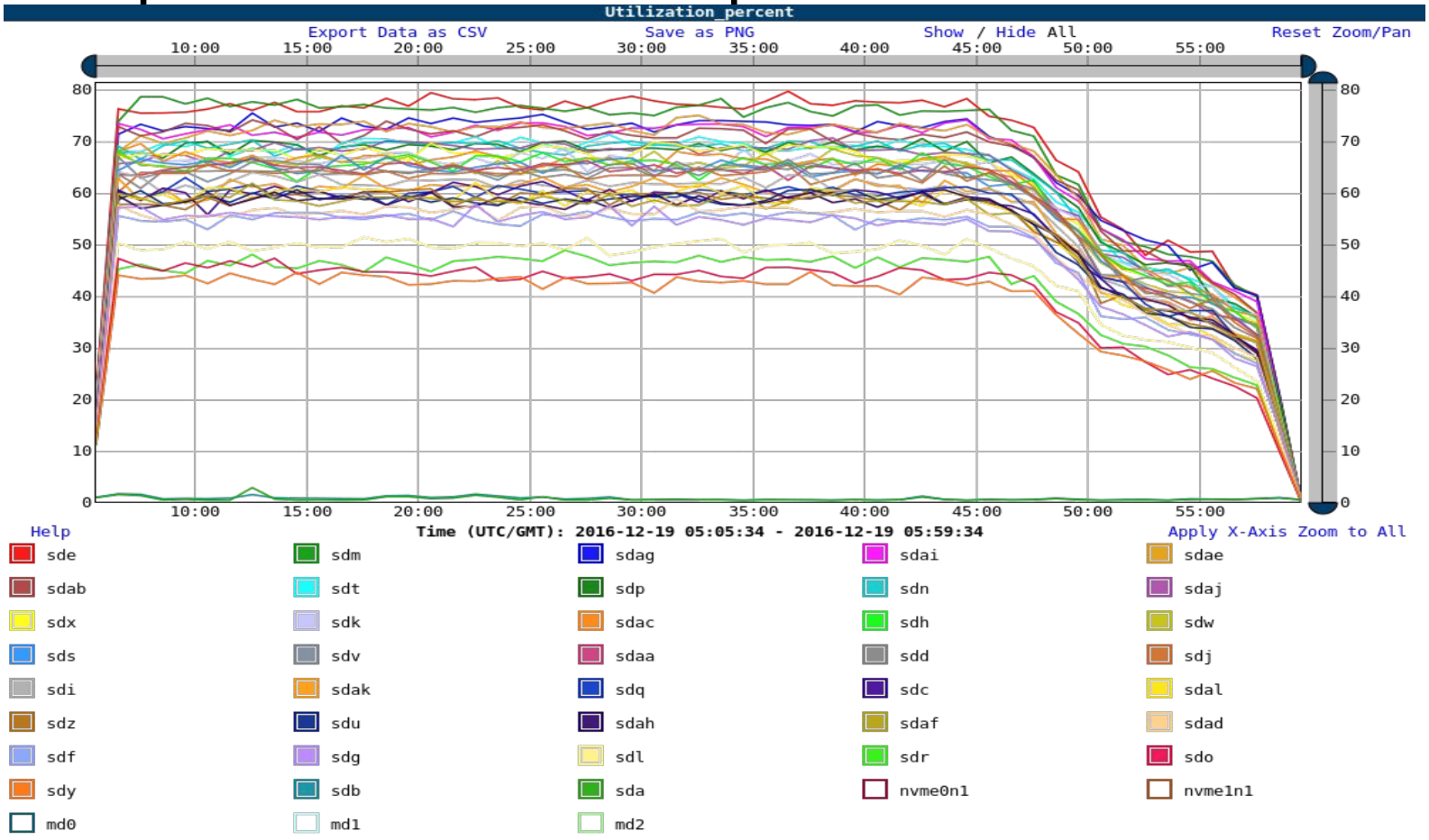
Root Cause Analysis

Ceph developer Kefu Chai examined a copy of one of the monitor databases. When the network is severely partitioned, OSDs can't talk to each other ("peering"), not enough free space, so it can't retire stale or damaged OSD maps and they build up, causing the database to grow.

Morals of story:

- Ceph is only as good as the network you give it
- If we do "ceph osd set noout" to prevent Ceph from trying to recover during a network outage, fix the network, then unset noout, Ceph would be fine
- You have to catch this problem before monitors run amok.

Ceph OSD utilization spread on one host wi 32K pgs



Cinder volume pool stats from readpgdump2.py

(Mark Nelson wrote this tool)

pool 20 'volumes' replicated size 3 min_size 2 crush_ruleset 0 object_hash rjenkins pg_num 49152 pgp_num 49152 last_change 60189 flags hashpspool stripe_width 0
removed_snaps [1~5]

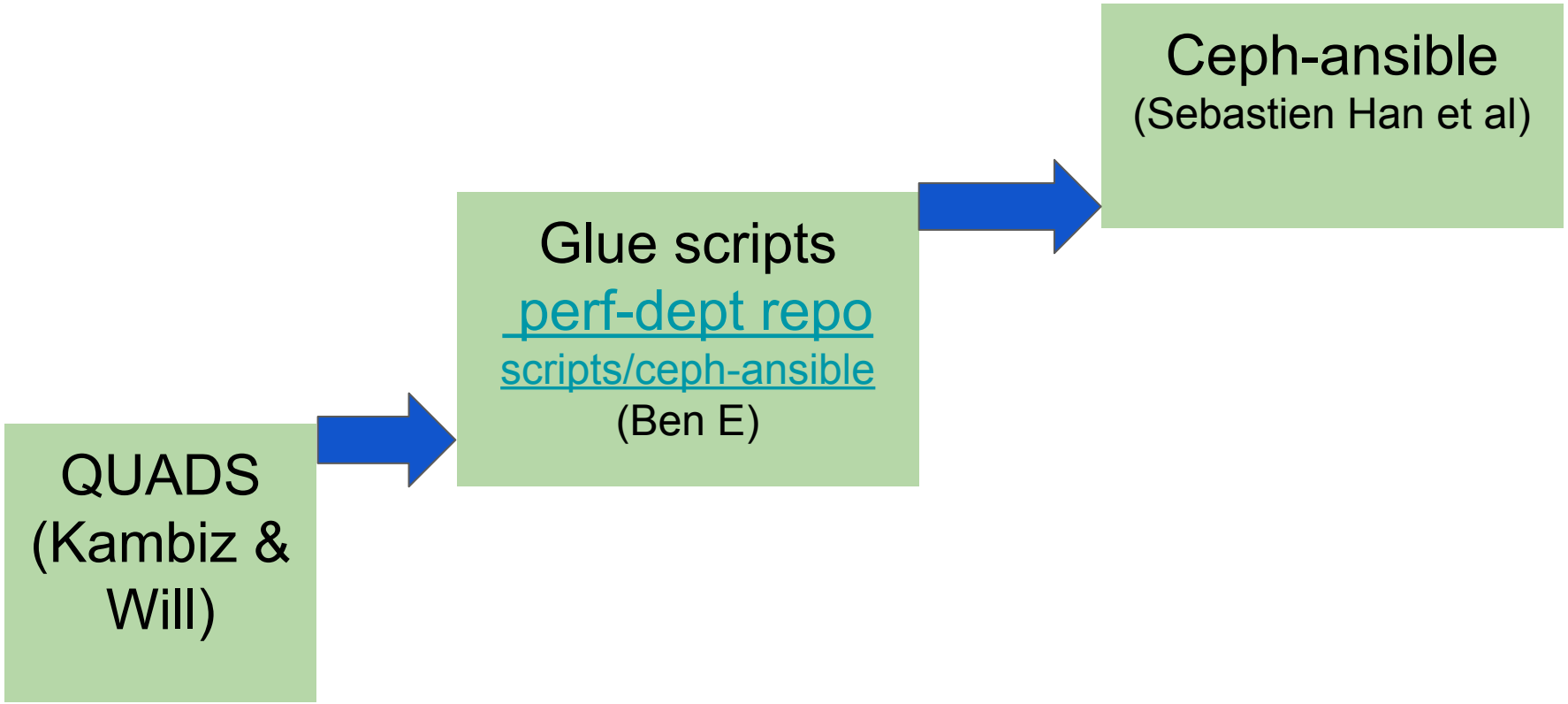
```
+-----------------------------------------------------------------------------+
| Pool ID: 20                                                                 |
+-----------------------------------------------------------------------------+
| Participating OSDs:  787                                                    |
| Participating PGs:  49152                                                   |
…
| OSDs in All Roles (Up)                                                      |
| PGs Per OSD: Min:  150.0, Max: 239.0, Mean: 187.4, Std Dev: 13.6            |
| 5 Most Subscribed OSDs: 699(239), 600(235), 54(232), 146(228), 466(228)     |
| 5 Least Subscribed OSDs: 17(150), 64(150), 277(153), 214(153), 455(153)     |
| Avg Deviation from Most Subscribed OSD:  21.6%                              |
```

The end

Long-term Challenges

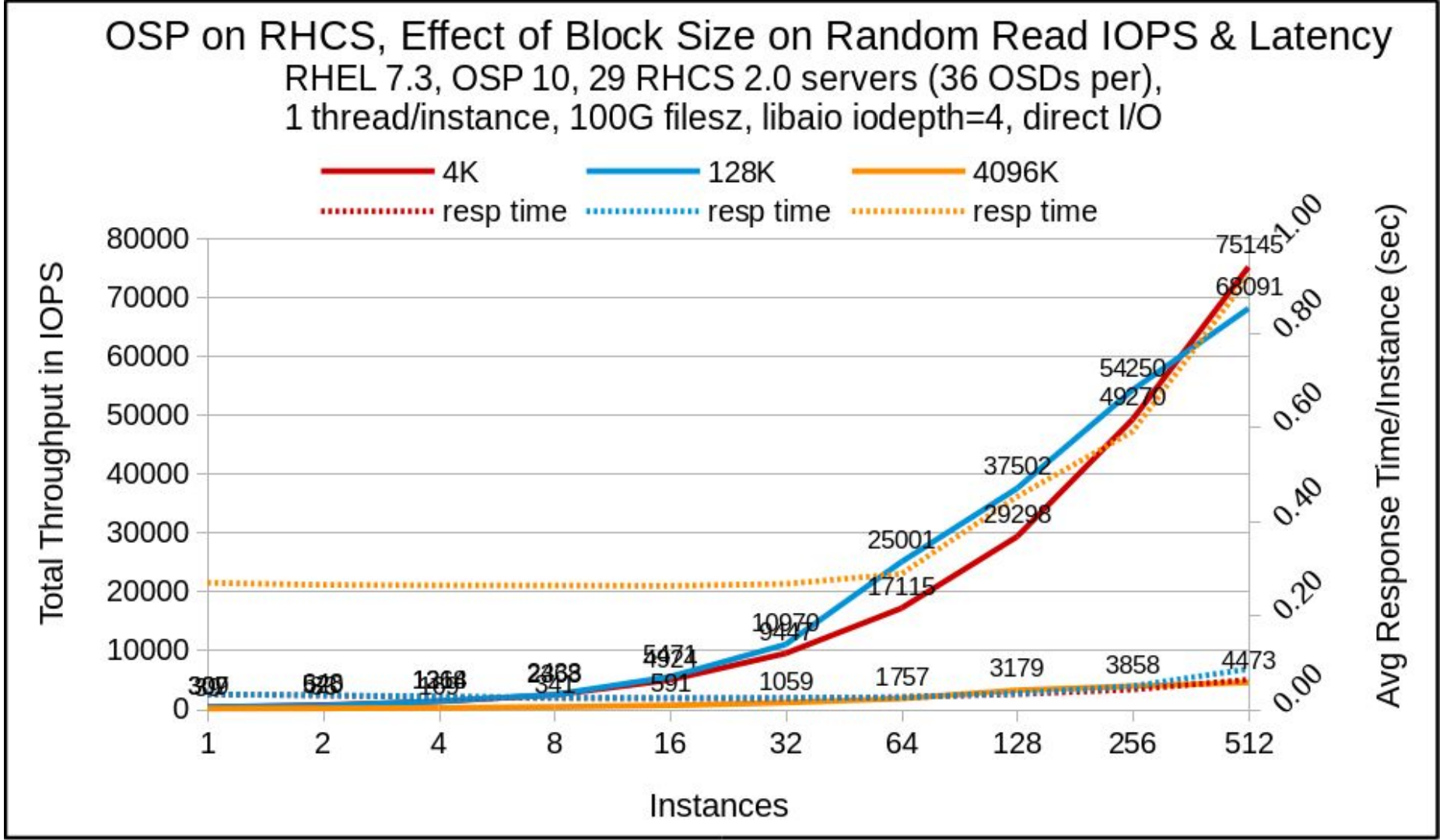
- Automate deployment of OpenStack + Ceph
  - Example: Get PG count approximation right if not specified by user
- Make Ceph rack-aware (crush maps)
  - Will increase inter-rack traffic but provide rack-level HA, typical customer requirement
- Increase hardware utilization to a more compelling level
  - Get 40-Gbps network hardware fully engaged
  - Even out HDD utilization
- I/O bandwidth balance between OpenStack compute hosts and Ceph
- Browbeat + pbench-fio on full-scale cluster to visualize results quickly

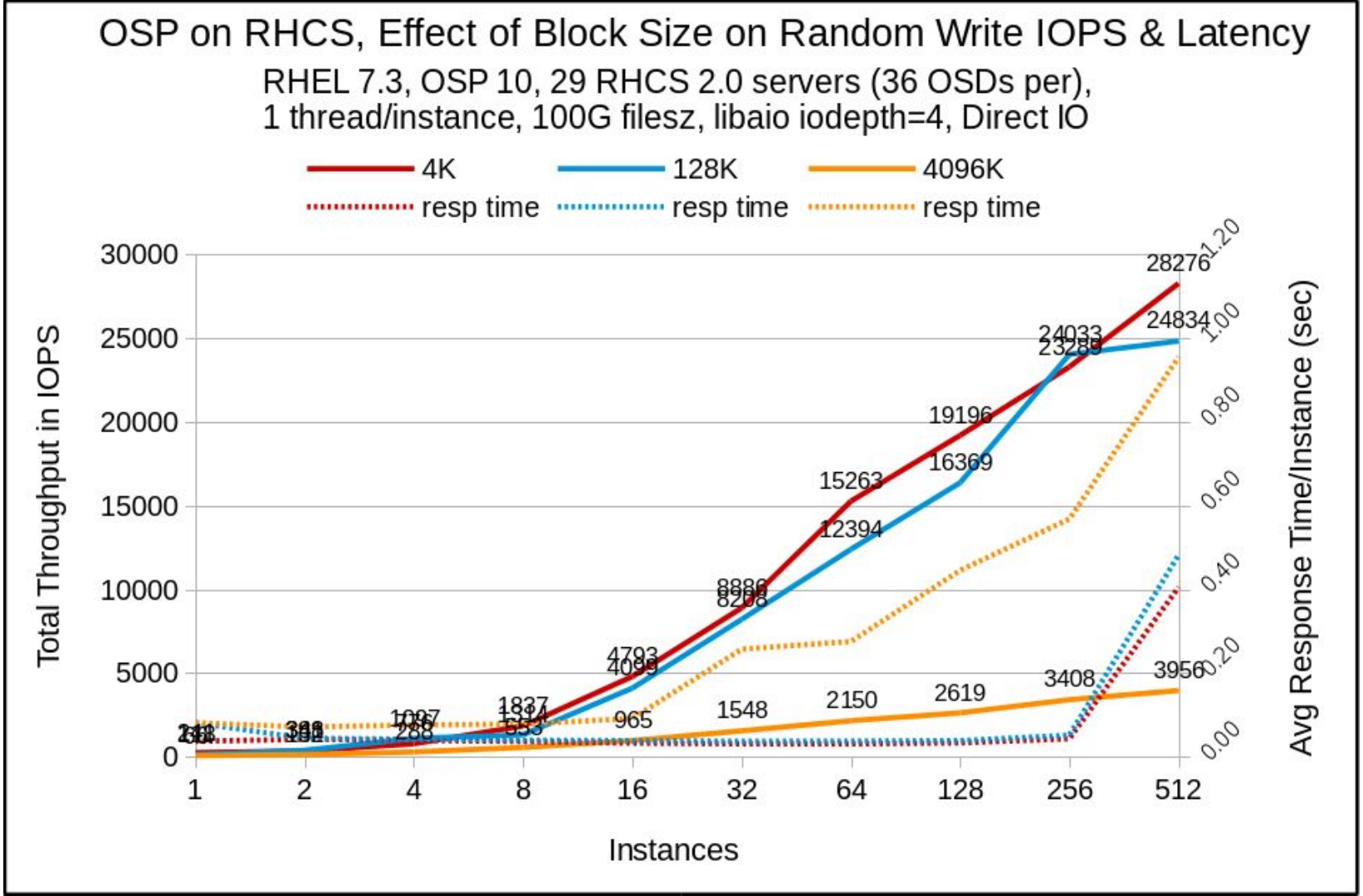
Ceph RHCS 2.0 Software deployment stack



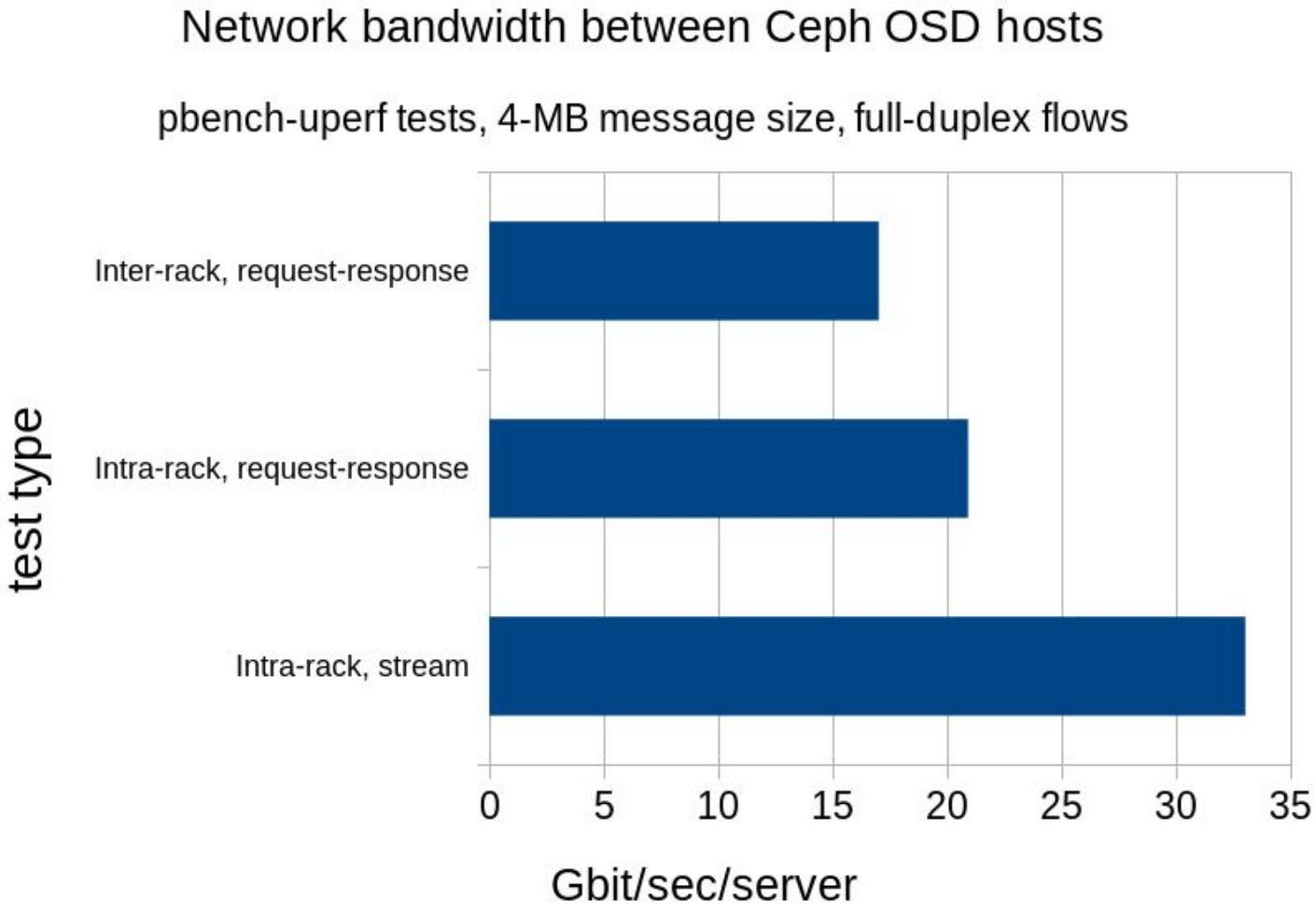
Tests performed so far

- pbench-uperf tests
  - Both stream and request-response tests
  - 4-MB request & response size
- Ceph low-level smoke test - rados bench - 4 MB objects
  - Need to use at least 16K Placement groups for storage pool to spread load across spindles, not default. Ceph PG calculator recommends ~32,000 PGs for 1000-OSD cluster
- CBT librbdfio benchmark - use Ceph more like VMs do
  - Peak bandwidth so far is 42 GB/s
  - Peak random IOPS so far is 140 KIOPS
- Pbench-fio tests from 512 OpenStack guests across 20 compute hosts
  - Peak read bandwidth so far is 18 GB/s
  - Peak random read IOPS so far is ~75K
  - Example utilization graph:

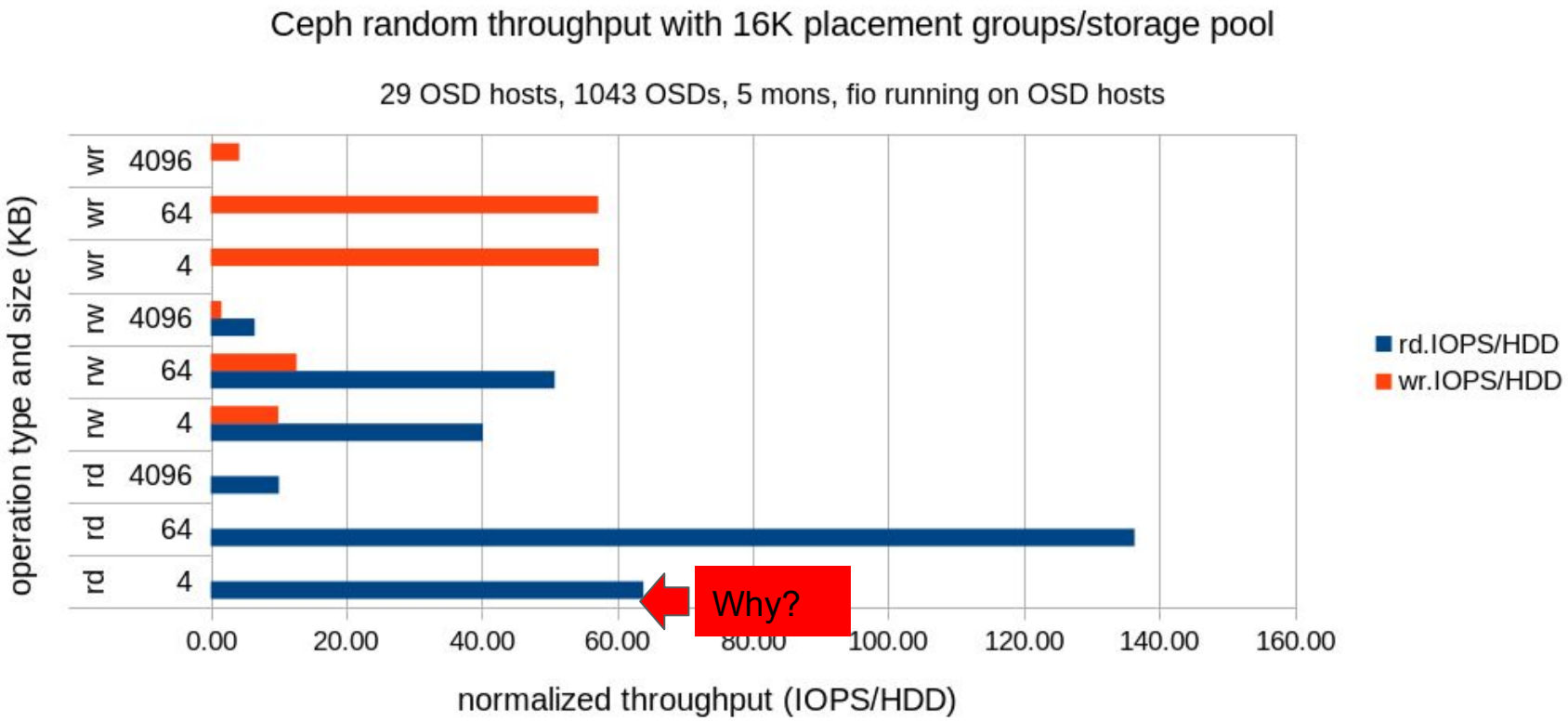

OSP on RHCS, Effect of Block Size on Random Read IOPS & Latency
RHEL 7.3, OSP 10, 29 RHCS 2.0 servers (36 OSDs per),
1 thread/instance, 100G filesz, libaio iodepth=4, direct I/O
4K
128K
4096K
resp time
resp time
resp time
Total Throughput in IOPS
Avg Response Time/Instance (sec)
Instances
80000
70000
60000
50000
40000
30000
20000
10000
0
1.00
0.80
0.60
0.40
0.20
0.00
1
2
4
8
16
32
64
128
256
512
75145
68091
54250
49270
37502
29298
25001
17115
10970
9447
5471
4924
591
1059
1757
3179
3858
4473



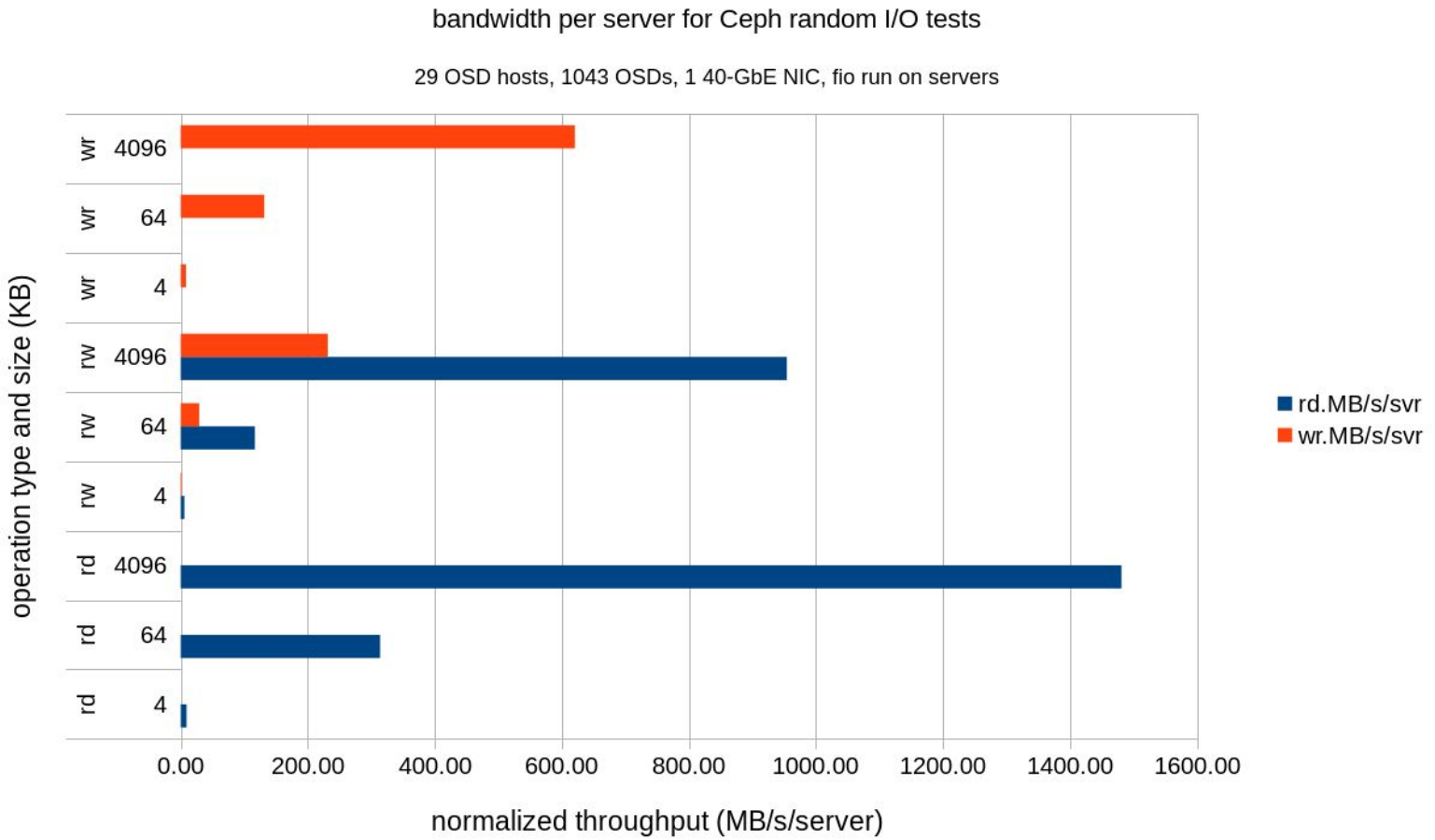
OSP on RHCS, Effect of Block Size on Random Write IOPS & Latency
RHEL 7.3, OSP 10, 29 RHCS 2.0 servers (36 OSDs per),
1 thread/instance, 100G filesz, libaio iodepth=4, Direct IO
4K
128K
4096K
resp time
resp time
resp time
Total Throughput in IOPS
Avg Response Time/Instance (sec)
Instances
30000
25000
20000
15000
10000
5000
0
1
2
4
8
16
32
64
128
256
512
1.20
1.00
0.80
0.60
0.40
0.20
0.00
28276
24834
24033
23289
19196
16369
15263
12394
8886
8208
4793
4099
1837
1514
1097
288
965
1548
2150
2619
3408
3956


Network bandwidth between Ceph OSD hosts
pbench-uperf tests, 4-MB message size, full-duplex flows
test type
Inter-rack, request-response
Intra-rack, request-response
Intra-rack, stream
0
5
10
15
20
25
30
35
Gbit/sec/server

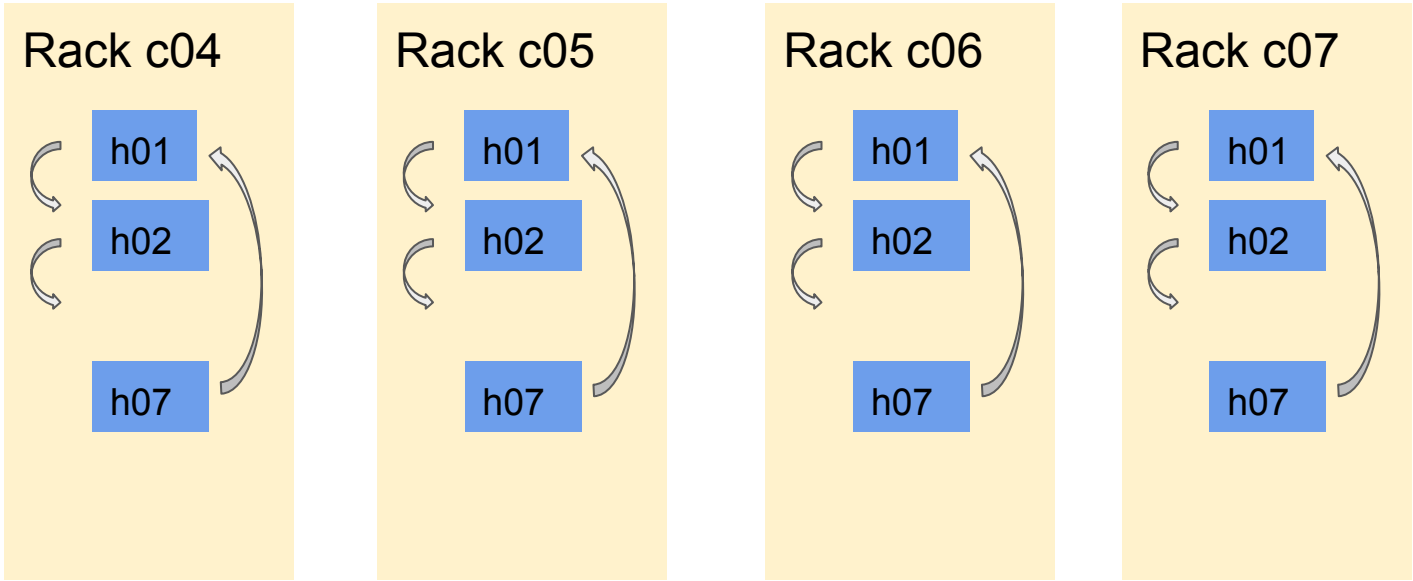
CBT (Ceph-only) results so far - random



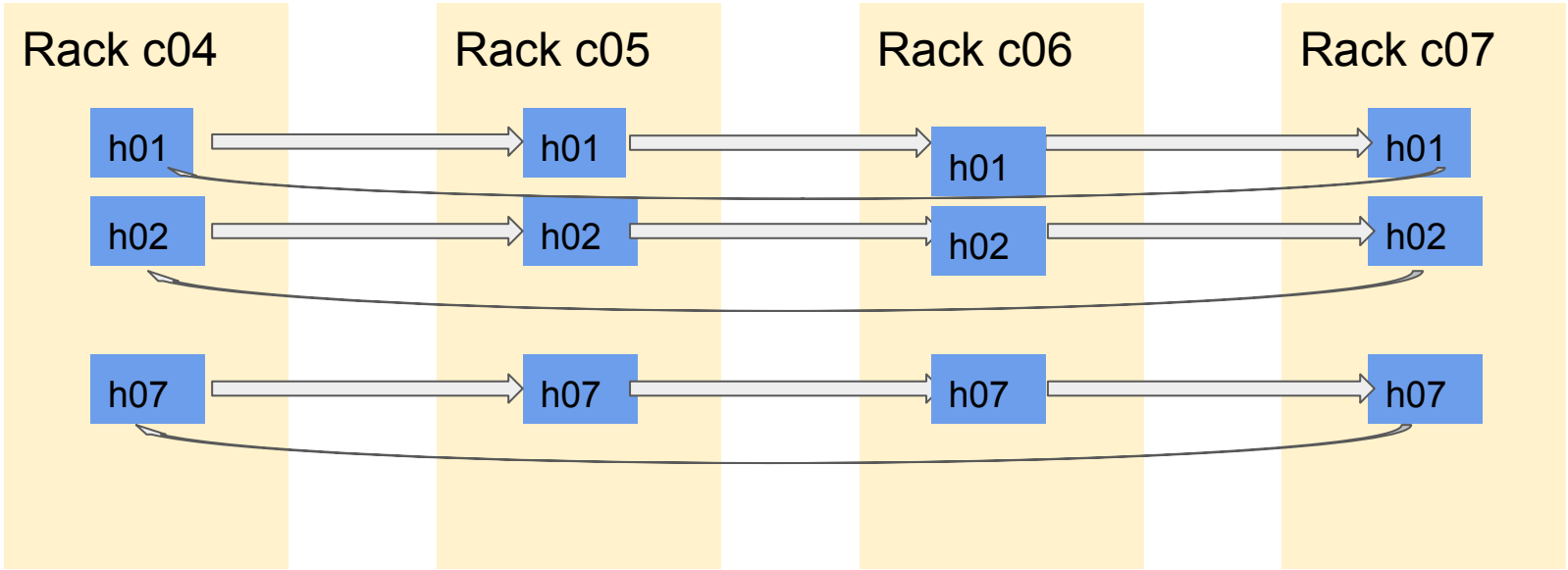
CBT (Ceph-only) results so far - bandwidth



Network testing - intra-rack



Network testing - inter-rack



inter-rack network throughput in exchanges/sec

payload 4-MB, 32 instances

