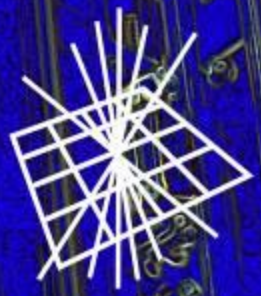


GridPP

UK Computing for Particle Physics

Ceph object storage at RAL

George Vasilakakos, Alastair Dewhurst, James Adams, Bruno Canning, Ignacy Debicki, Shaun de Witt

Science & Technology Facilities Council

Our Ceph Grid Cluster

- Ceph Hammer (0.94.1.)
  - 47 storage nodes
  - 21 x ~100TB + 26 x ~120TB
  - 2-3GB of RAM per OSD
  - 2x10GbE networking
    - One for public, one for cluster (not yet in place)
- 1286 OSDs, 5196TB raw storage space
- 3 physical monitors
- 3 physical gateways
  - 40 logical cores
  - 128GB RAM
  - 4x10 GbE links

Our Ceph Grid Cluster

- 3 physical gateways
  - Each gateway will provide all three interfaces (S3, XrootD, GridFTP) to the object store
  - S3 interface to be deployed by the end of the month
    - Using civetweb as shipped with Ceph
    - Have successfully tested IPv6
  - GridFTP and XrootD with Ceph backend support and x.509 proxies are next
    - Some bugs on the Ceph side (libradosstriper), fixes have been merged, next Ceph release should resolve this
  - GridFTP, developed by Ian Johnson at RAL, for FTS transfers
  - XrootD, developed by Sebastien Ponce at CERN, for worker nodes to access the object store

Monitoring

- Nagios checks in place
  - Overall cluster health, OSDs, MONs, S3 functional test
- Intent is to use ELK for logging
- Monitoring with InfluxDB and Grafana
  - Ceph is distributed, Ganglia can only go so far
  - We chose InfluxDB because of the free-form tagging
  - Custom Python scripts by summer student Ignacy Debicki
    - Query Ceph and report to InfluxDB, highly configurable
    - https://github.com/stfc/ceph-InfluxDB-metricsCollector
  - Small virtualised instance (temporary)
    - ~250k data points per minute, in one big batch, were too much
  - Work in progress

Services under development

- S3/Swift Object Store
  - Aim to provide it for anyone who wants it
  - ATLAS Event Service will likely be first user
  - Looking at CVMFS Stratum 0
  - Climate physics group has shown interest
- XrootD for Tier-1 batch farm
  - XrootD 4.2.0 adds support for Ceph backends
  - Used for the batch farm to access data on the Ceph obj. store
    - Plan to deploy local XrootD gateways on worker nodes to avoid bottlenecks
- GridFTP
  - Also now supports Ceph backends
  - Used for WAN transfers via FTS
- XrootD and GridFTP are interoperable
  - Same pool, metadata, authentication
- We also aim to provide some space to RAL's Castor team
  - Castor 2.1.16 supports Ceph backends
  - We plan to follow CERN in switching to Ceph for Castor tape buffers

Erasure Coding

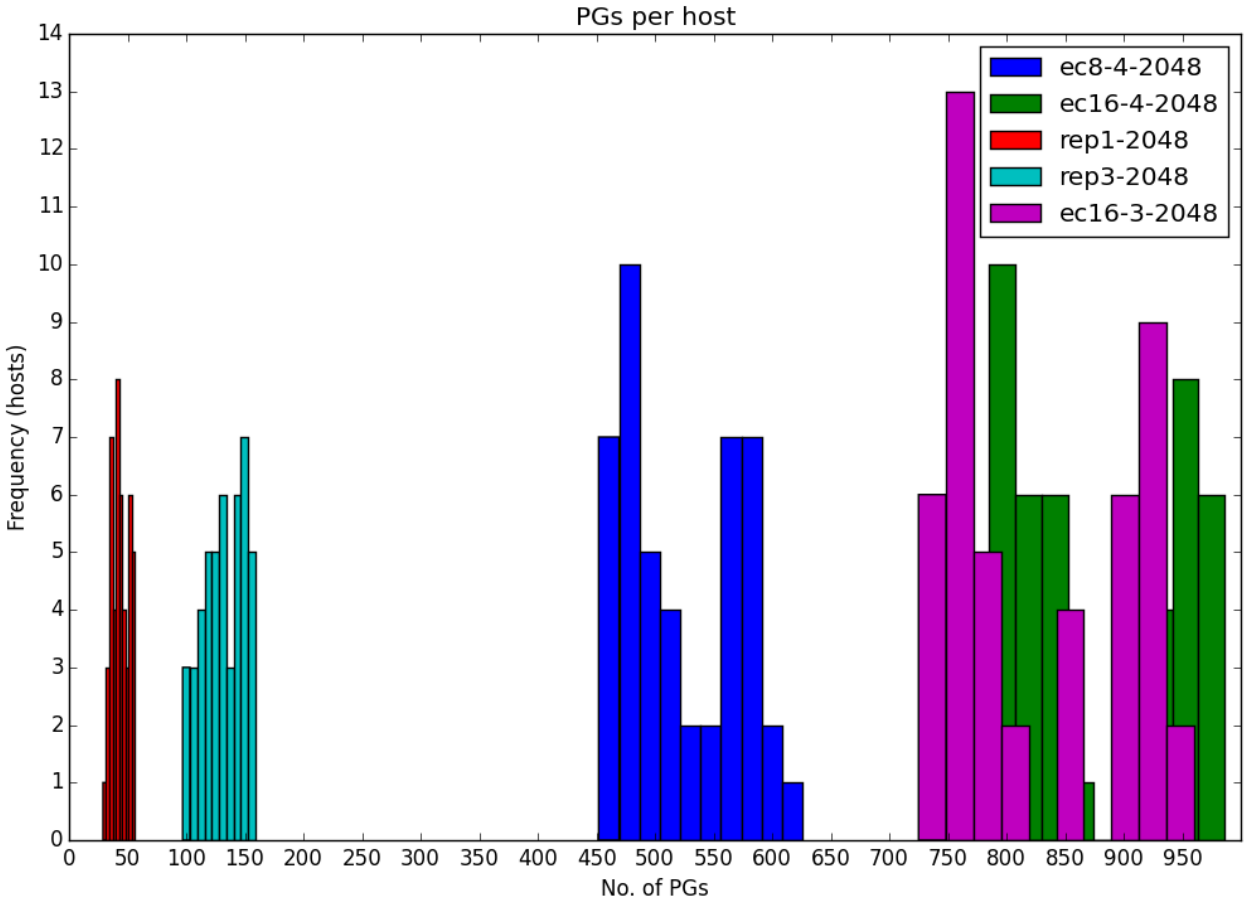
- Why Erasure Coding
  - More efficient use of disk space
  - At the cost of CPU time and extra concurrency
  - At 16+3 we incur 18.75% overhead but can tolerate 3 concurrent failures, as opposed to keeping 4 copies
- We plan to use 16+3, initially with 2048 PGs
  - If we lose 4 disk drives (at random) at the same time, the chances of a placement group using all of them (and hence losing data) is only 0.007% and if this did happen we predict unto 35TB of data would be lost.
  - CRUSH map and ruleset protect against loss of data by complete failure of any single server
- EC not without its Challenges
  - PGs' acting sets are large, more on following slides
  - Higher overall latencies
    - First OSD in acting set has to compute EC chunks for all IOs
  - Higher recovery load

Data placement

- We are using a simple CRUSH ruleset that tries to choose one OSD from each host
  - Hosts touched by single op = (replication factor|k+m value of EC profile)
  - The higher the number the higher the variance in number of PGs assigned to nodes
  - Our cluster has 2 different types of nodes, most of the variance is down to this: 120TB nodes get 20% more PGs than 100TB nodes.
- Starting our main data pools with 2048 PGs
  - Need to achieve a sane distribution over ~1300 OSDs
  - Not going for the suggested amount of PGs (too large)
  - Increase as needed when expanding cluster (stay at a few PGs per OSD)
- With a 16+3 pool and 47 nodes, ~40% of nodes will participate in each operation
  - Large amounts of inter-node communication
  - Slows down peering process
  - Large amounts of threads (~90k) per storage node
- We expect cluster networking and more storage nodes to improve situation
  - Split network contention between internal traffic and client IO
  - Spread PGs out over more hosts

Data placement

Histogram shows number of PGs per SN



Distribution analysis tool by apprentice Jack Harper at RAL: https://github.com/stfc/ceph-pg-analyst

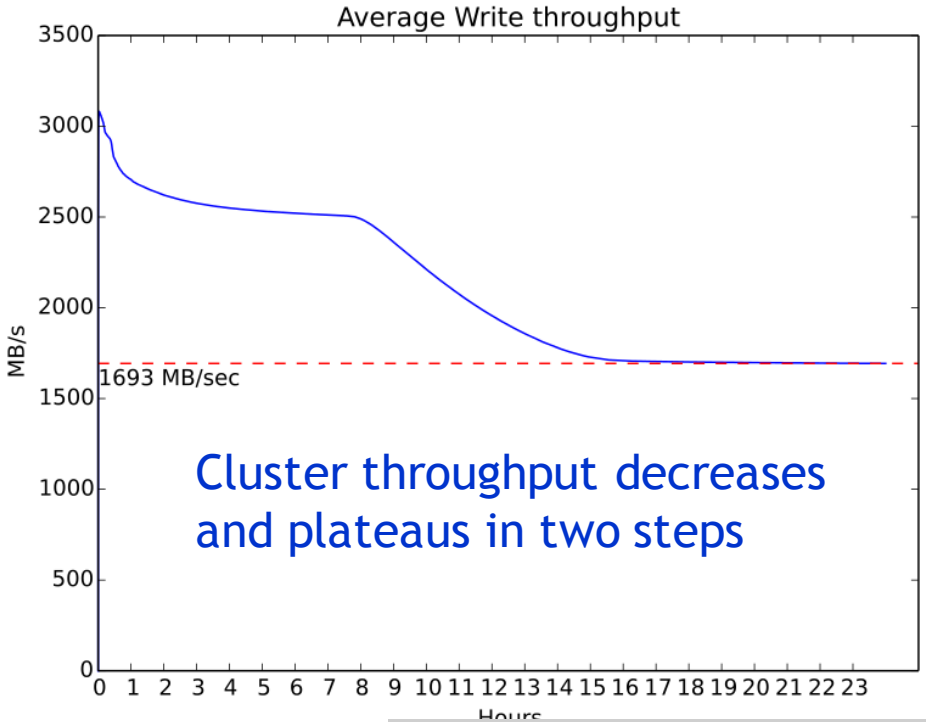
Observations with fewer nodes

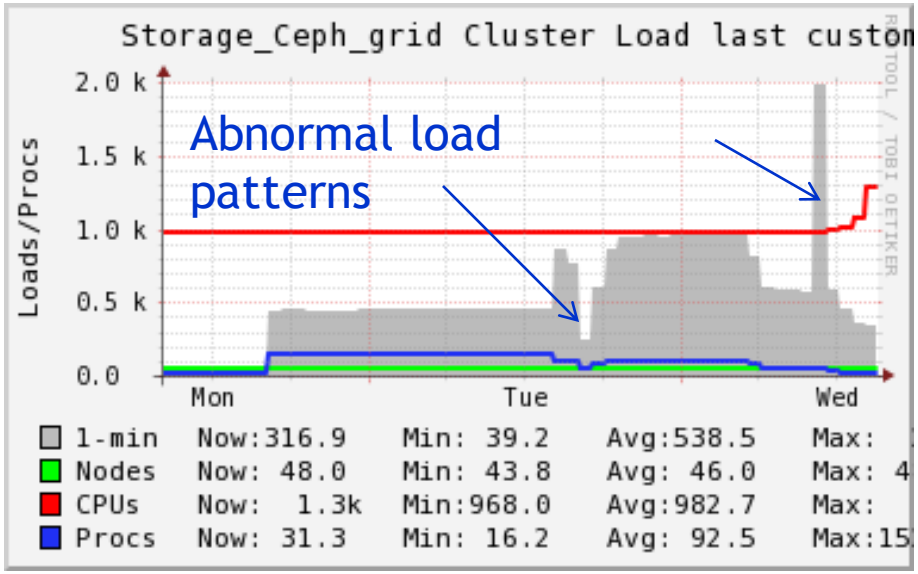
- Procuring network equipment for cluster backend
  - We were loaned some really fast, new hardware to test with
  - Not enough ports for all nodes
  - Tested with only the larger 21 nodes (34 OSDs each)
- Tried EC 16+3, didn't work, went down to 16+2
  - Pathological case: 18 out of 21 nodes touched for each IO op.
  - Tried generating some IO using rados bench from the monitors
  - Despite using cluster network IO contention was too much
    - Load spikes
    - Ever-increasing swap usage
    - Most OSD processes in D states
    - Ceph bogged down with "slow requests"
    - OSDs couldn't even keep up peering traffic, ended up being "wrongly marked down"

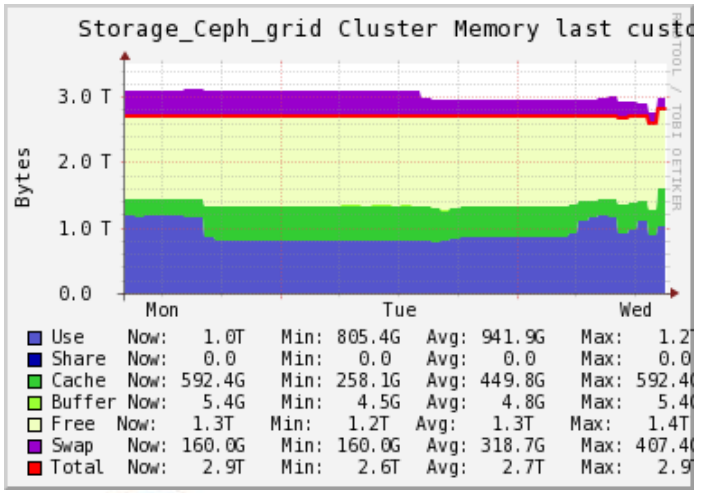
Observations with fewer nodes

- The 3 monitors were used to generate a write load on the cluster
- Ceph provides rados bench, a benchmarking tool that access the object at the lowest level (RADOS)
- Each monitor was set to write 4MB objects to the cluster, continuously for 24 hours, using 32 threads.
- The plots present aggregated measurements of
  - Throughput
  - Load
  - Network
  - Memory

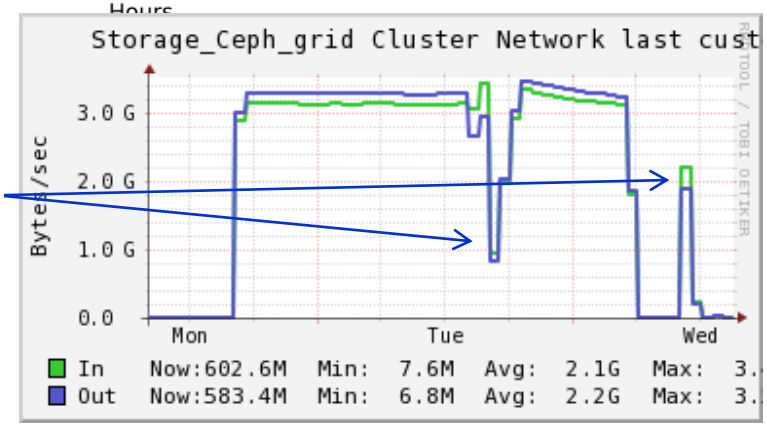
Observations with fewer nodes







Similarly erratic behaviour



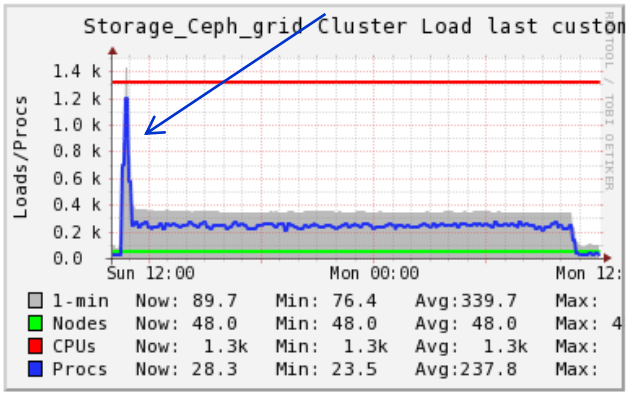
Observations with fewer nodes

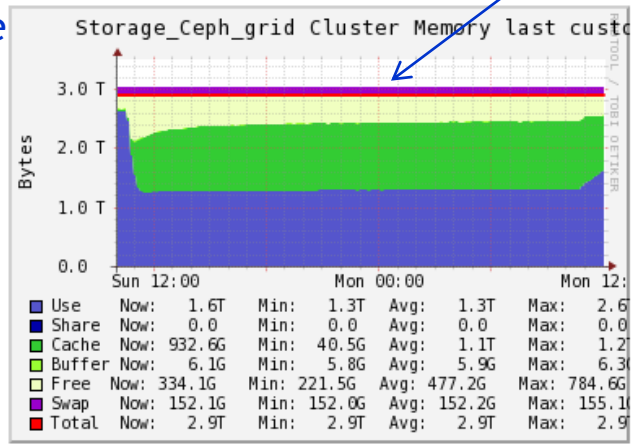
- Reconfigured cluster
  - Used all 47 storage nodes
  - Removed “cluster” network configuration
  - Using single 10GbE connection for client I/O, cluster background I/O, cluster communication
  - Erasure-coded 16+3 pool created successsfully
- Repeated test
  - Much better results
  - More throughput
    - Mostly down to having more nodes
    - Better stability: no OSDs were marked down

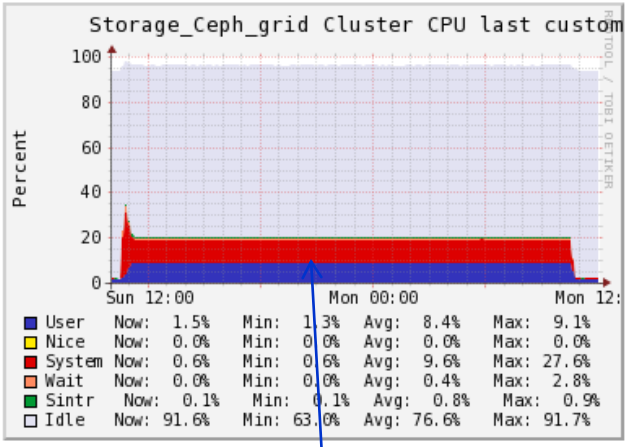
Observations with fewer nodes

Some swap usage, but it's static

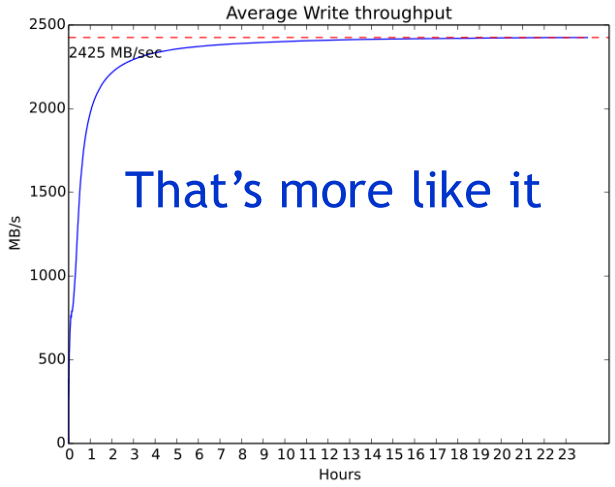
After initial load spike, very stable



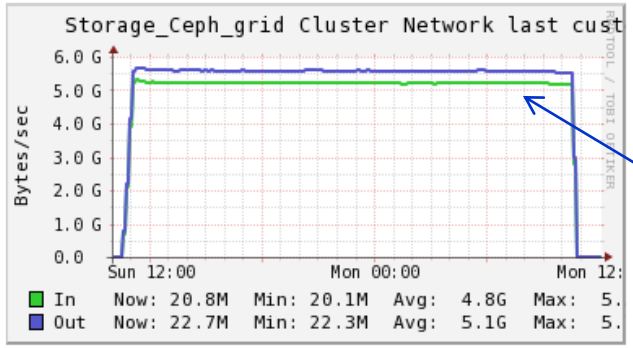




Almost no waiting



That's more like it



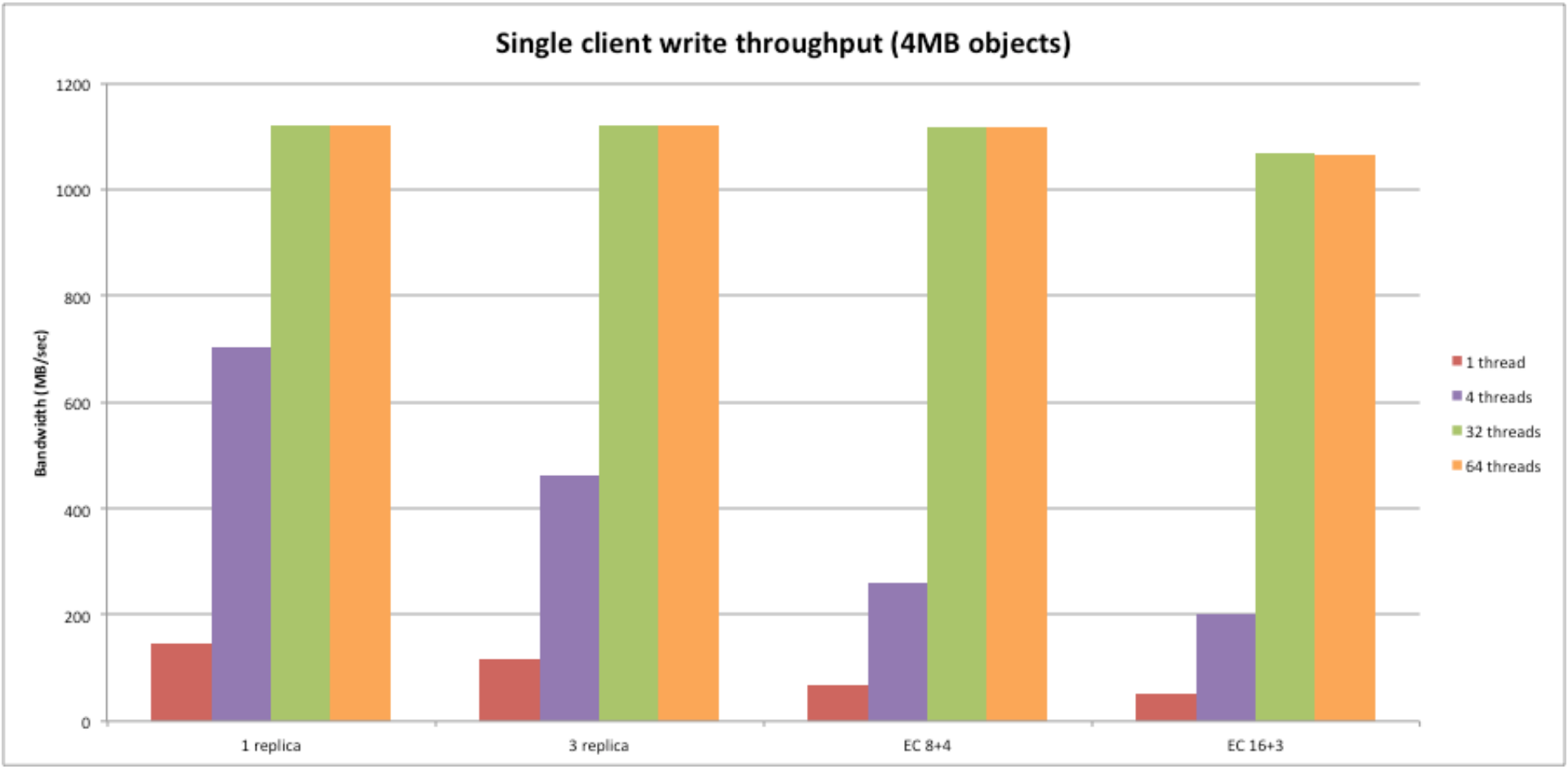
More bandwidth used and more stable

Everything goes smoothly

EC performance tests

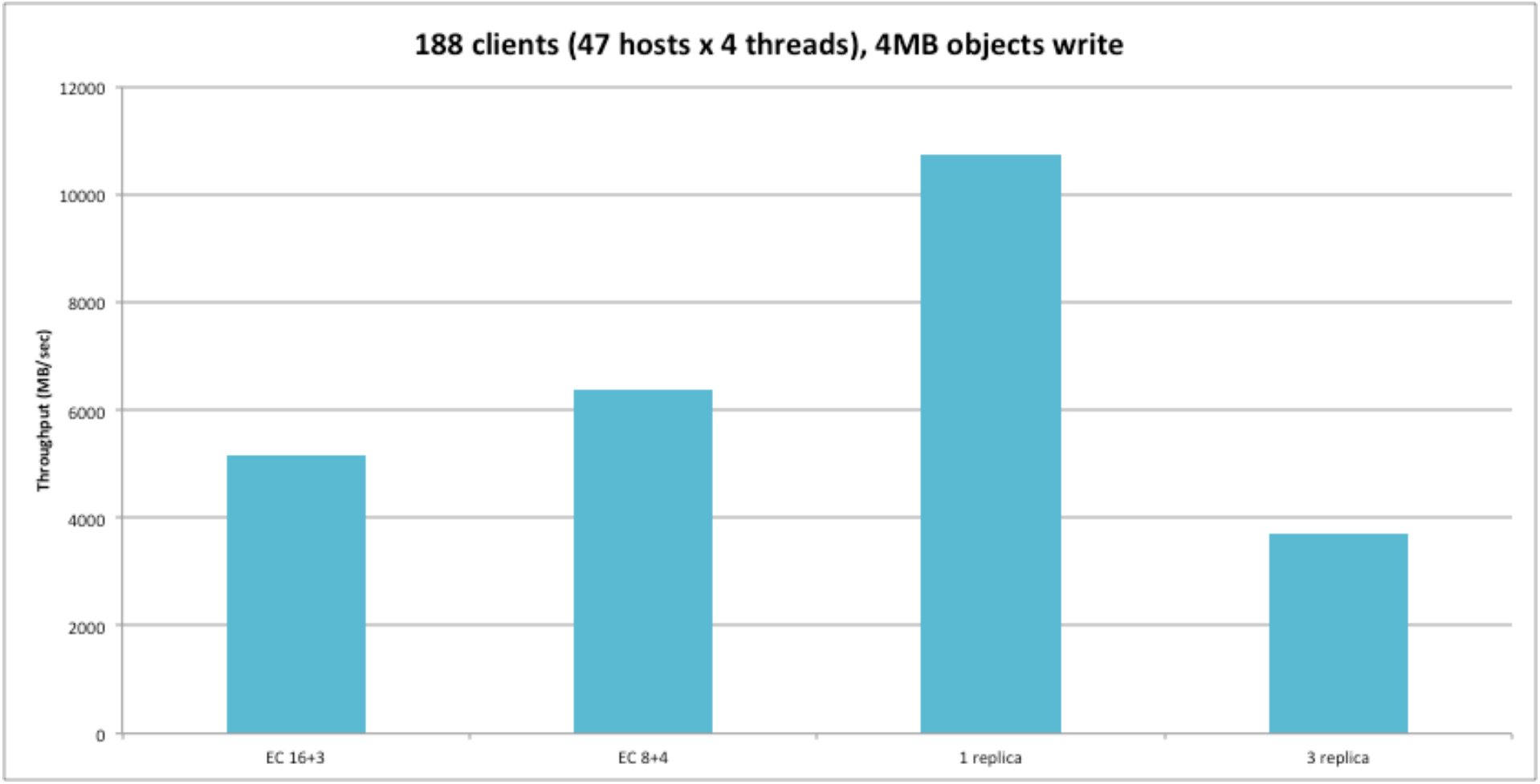
- We benchmarked the cluster using rados bench under various scenarios
- We tried to stick to the methodology of tests presented by CERN during last HEPiX
  - Aim is to be able to compare results
  - We tested mainly with erasure coded pools
  - Tried schemas which may be used in a production system
- We intend to use the object store with striping
  - We don't want a 1GB file stored as one object, but sharded in many parts
- Also interested in
  - many-small-writes performance
  - concurrency

EC performance tests



EC performance tests

EC performance tests



Future Work

- Still have extensive testing to do with EC
  - Fault tolerance & recovery
  - Stress-testing using the gateways
  - Tuning
- Test the gateways
  - Scale-testing of GridFTP and XrootD
  - Set up load balancing scheme across the gateways
- Would like to establish a baseline for individual node performance and identify bottlenecks
- Use our ElasticSearch instance for logging
- Get the cluster networking in place

Questions?