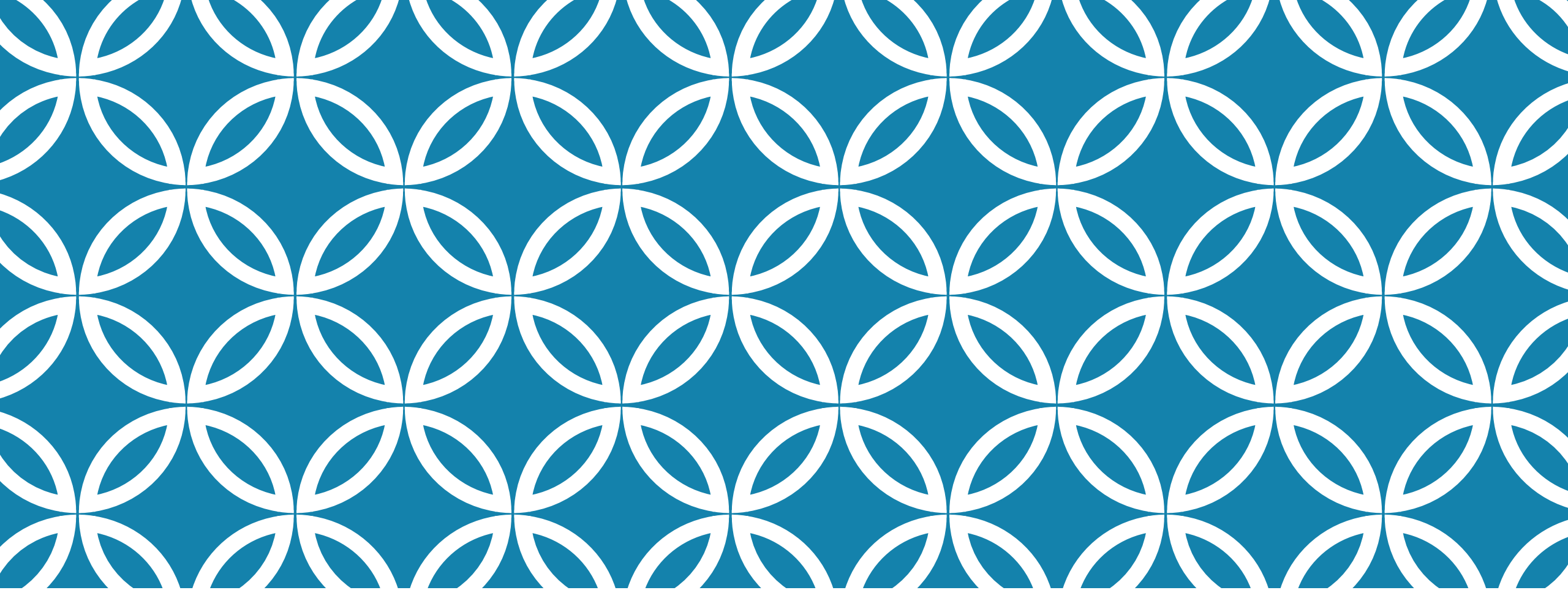


CS 5412/LECTURE 24. CEPH: A SCALABLE HIGH-PERFORMANCE DISTRIBUTED FILE SYSTEM

Ken Birman

Spring, 2019

HDFS LIMITATIONS

Although many applications are designed to use the normal "POSIX" file system API (operations like file create/open, read/write, close, rename/replace, delete, and snapshot), some modern applications find POSIX inefficient.

Some main issues:

- HDFS can handle big files, but treats them as sequences of fixed-size blocks. Many application are object-oriented
- HDFS lacks some of the "file system management" tools big-data needs

CEPH PROJECT

Created by Sage Weihl, a PhD student at U.C. Santa Cruz

Later became a company and then was acquired into Red Hat Linux

Now the "InkStack" portion of Linux offers Ceph plus various tools to leverage it, and Ceph is starting to replace HDFS worldwide.

Ceph is similar in some ways to HDFS but unrelated to it. Many big data systems are migrating to the system.

CEPH HAS THREE "APIs"

First is the standard POSIX file system API. You can use Ceph in any situation where you might use GFS, HDFS, NFS, etc.

Second, there are extensions to POSIX that allow Ceph to offer better performance in supercomputing systems, like at CERN.

Finally, Ceph has a lowest layer called RADOS that can be used directly as a key-value object store.

WHY TALK DIRECTLY TO RADOS? SERIALIZATION/DESERIALIZATION!

When an object is in memory, the data associated with it is managed by the class (or type) definition, and can include pointers, fields with gaps or other "subtle" properties, etc.

Example: a binary tree: the nodes and edges could be objects, but the whole tree could also be one object composed of other objects.

Serialization is a computing process to create a byte-array with the data in the object. Deserialization reconstructs the object from the array.

GOOD AND BAD THINGS

A serialized object can always be written over the network or to a disk.

But the number of bytes in the serialized byte array might vary. **Why?**

… so the "match" to a standard POSIX file system isn't ideal. **Why?**

This motivates Ceph.

KEY IDEAS IN CEPH

The focus is on two perspectives: object storage (ODS, via RADOS) for actual data, with automatic "striping" over multiple server for very large files or objects. Fault-tolerance is automatic.

MetaData Management. For any file or object, there is associated meta-data: a kind of specialized object. In Ceph, meta-data servers (MDS) are accessed in a very simple hash-based way using the CRUSH hashing function. This allows direct metadata lookup

Object "boundaries" are tracked in the meta-data, which allows the application to read "the next object." This is helpful if you store a series of objects.

CEPH: A SCALABLE, HIGH-PERFORMANCE DISTRIBUTED FILE SYSTEM

Original slide set from OSDI 2006

Sage A. Weil, Scott A. Brandt, Ethan L. Miller, Darrel D. E. Long

CONTENTS

GOALS

Scalability

- Storage capacity, throughput, client performance. Emphasis on HPC.

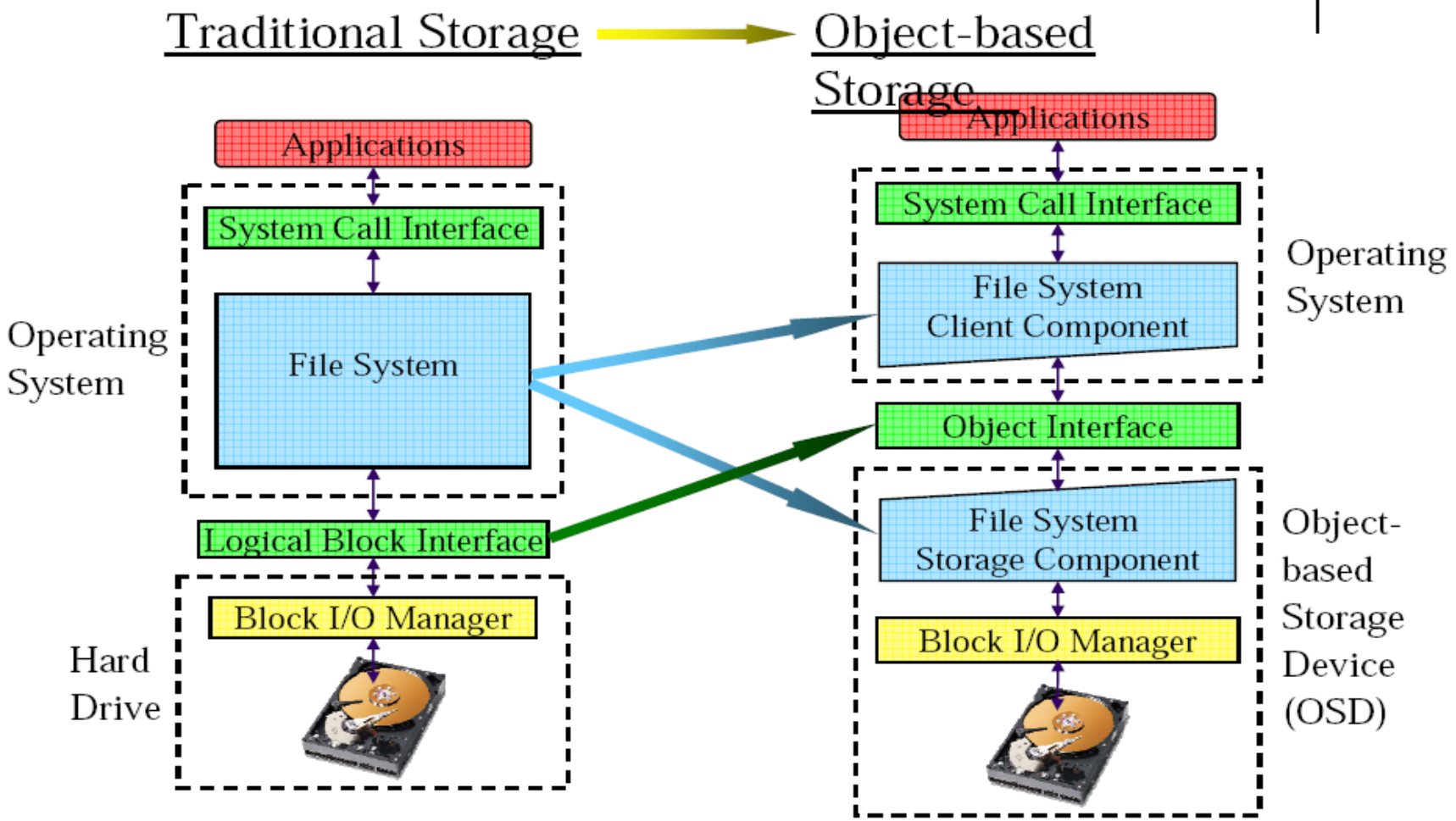
Reliability

- "…failures are the norm rather than the exception…"

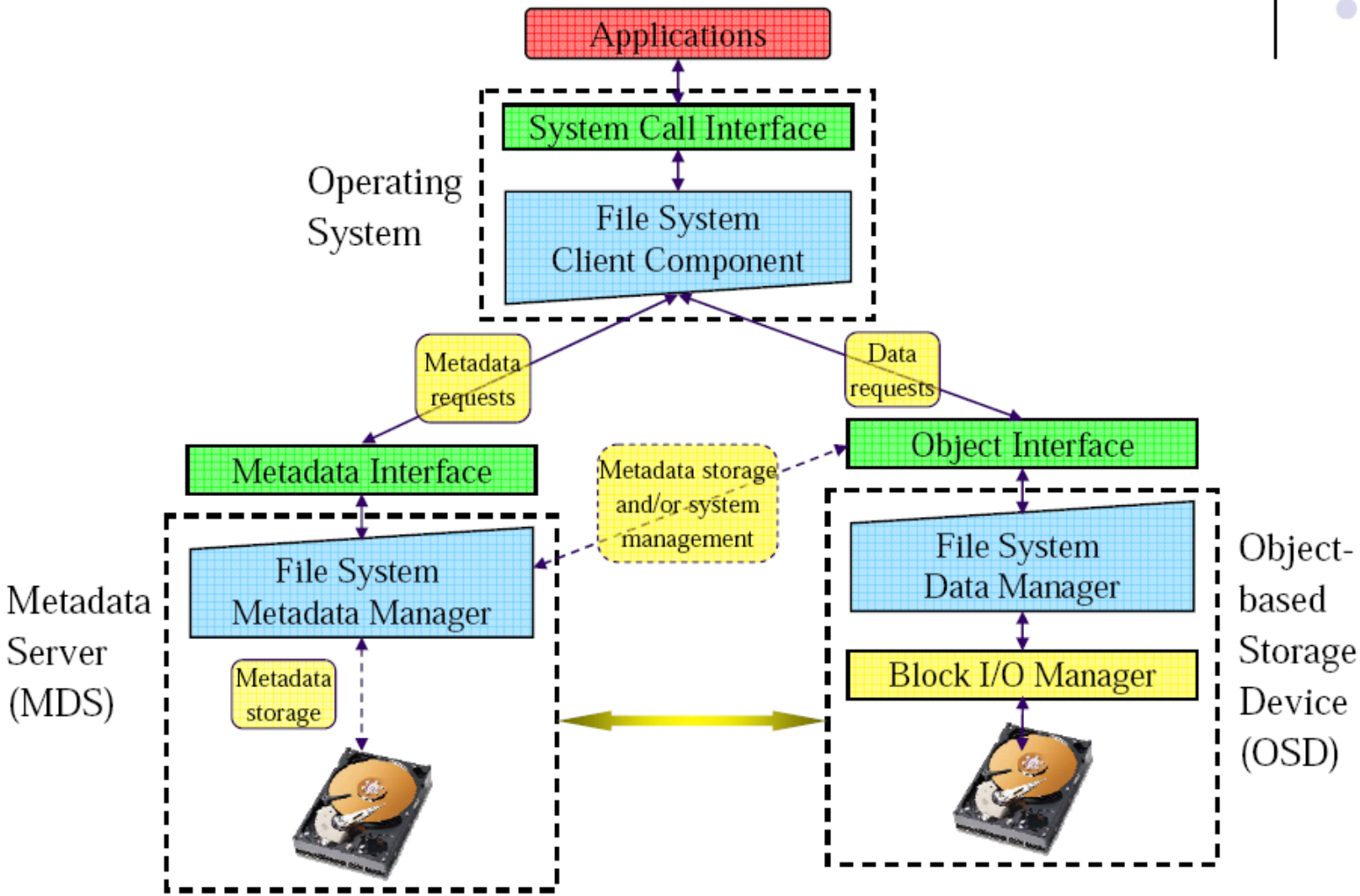
Performance

- Dynamic workloads

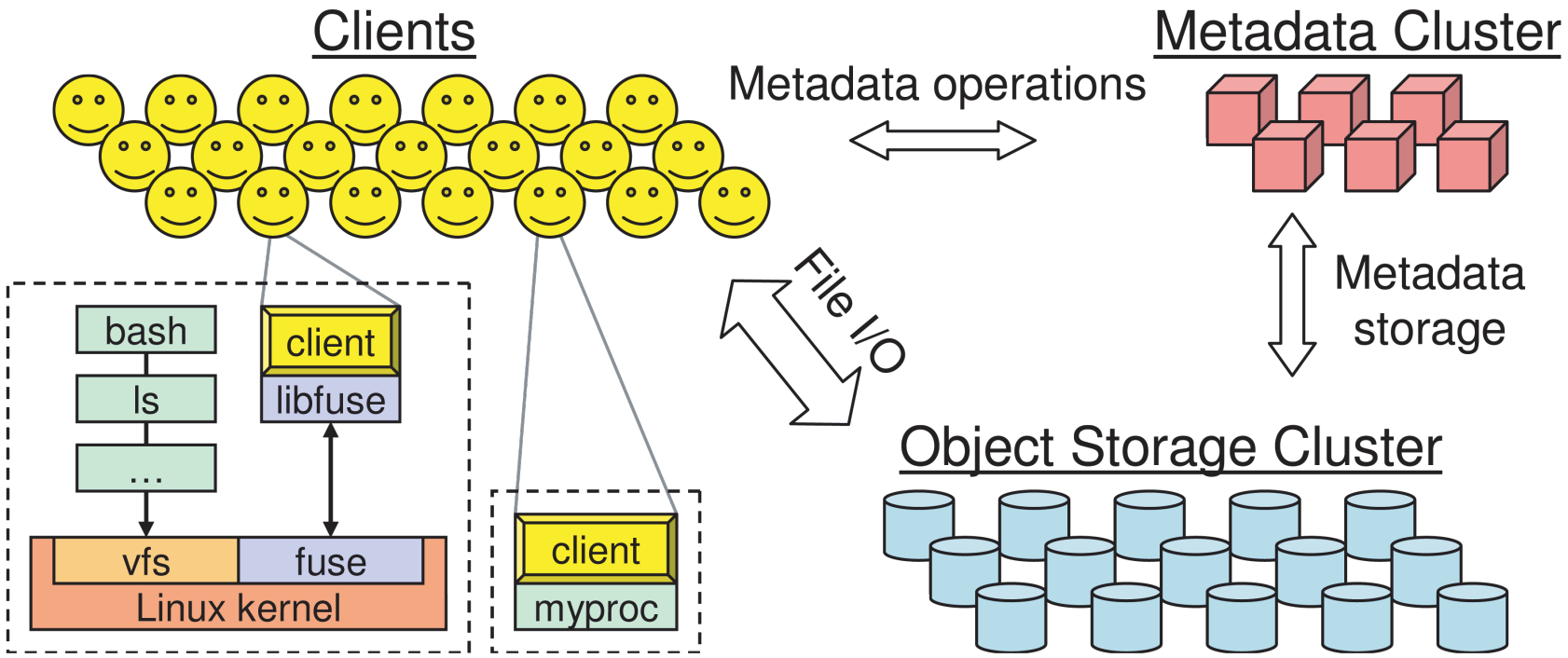
First Key Idea: Object-based Storage



Second Key Idea: Decoupled Data and Metadata



SYSTEM OVERVIEW



KEY FEATURES

Decoupled data and metadata

- CRUSH
  - Files striped onto predictably named objects
  - CRUSH maps objects to storage devices

Dynamic Distributed Metadata Management

- Dynamic subtree partitioning
  - Distributes metadata amongst MDSs

Object-based storage

- OSDs handle migration, replication, failure detection and recovery

CLIENT OPERATION

Ceph interface

- Nearly POSIX
- Decoupled data and metadata operation

User space implementation

- FUSE or directly linked

FUSE is a software allowing to implement a file system in a user space

CLIENT ACCESS EXAMPLE

Client sends open request to MDS

MDS returns capability, file inode, file size and stripe information

Client read/write directly from/to OSDs

MDS manages the capability

Client sends close request, relinquishes capability, provides details to MDS

SYNCHRONIZATION

Adheres to POSIX

Includes HPC oriented extensions

- Consistency / correctness by default
- Optionally relax constraints via extensions
- Extensions for both data and metadata

Synchronous I/O used with multiple writers or mix of readers and writers

DISTRIBUTED METADATA

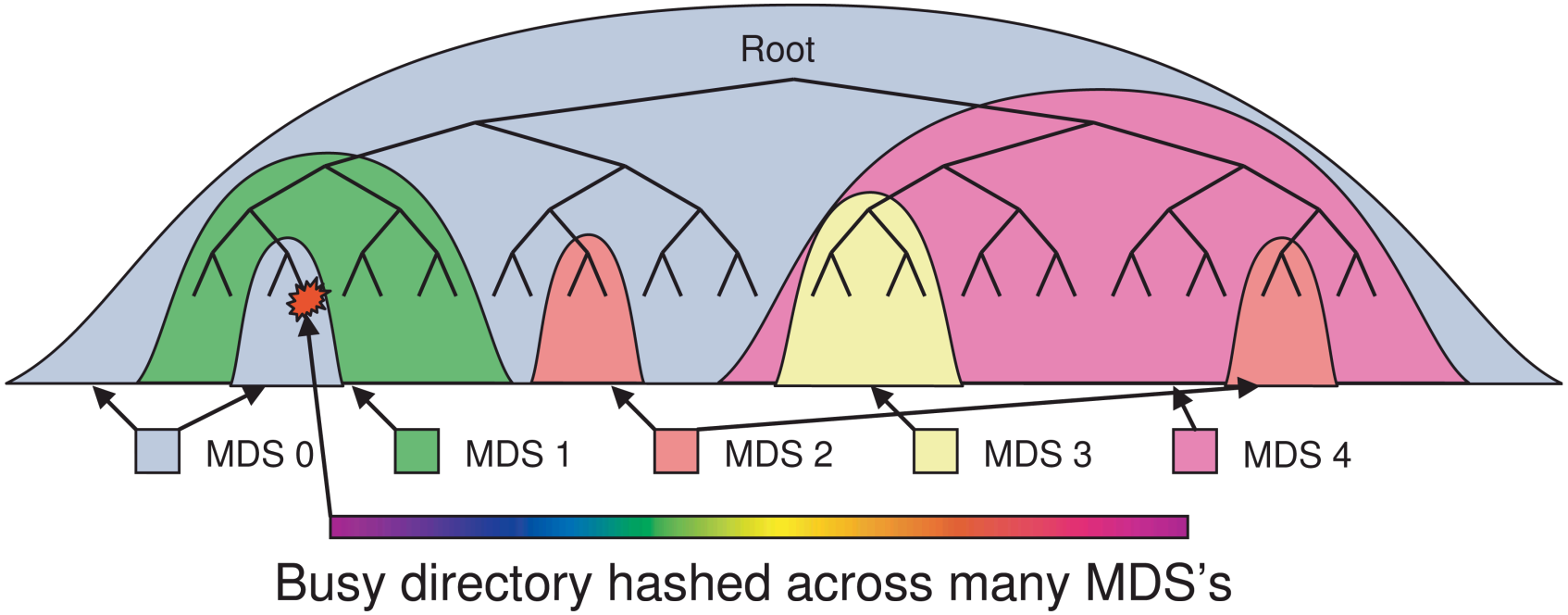
"Metadata operations often make up as much as half of file system workloads…"

MDSs use journaling

- Repetitive metadata updates handled in memory
- Optimizes on-disk layout for read access

Adaptively distributes cached metadata across a set of nodes

DYNAMIC SUBTREE PARTITIONING



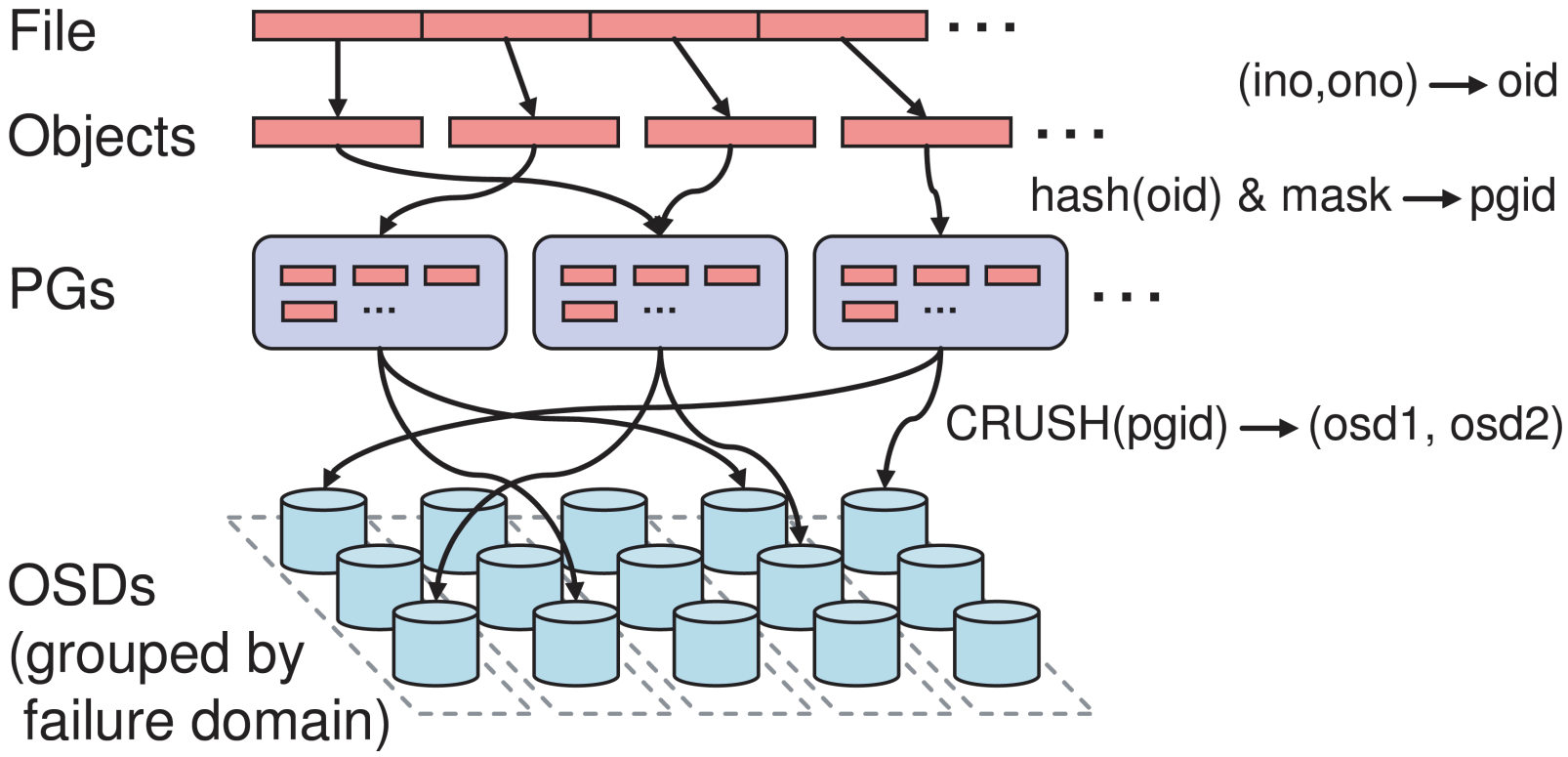
DISTRIBUTED OBJECT STORAGE

Files are split across objects

Objects are members of placement groups

Placement groups are distributed across OSDs.

DISTRIBUTED OBJECT STORAGE



CRUSH

CRUSH(x): (osdn1, osdn2, osdn3)

- Inputs
  - x is the placement group
  - Hierarchical cluster map
  - Placement rules
- Outputs a list of OSDs

Advantages

- Anyone can calculate object location
- Cluster map infrequently updated

DATA DISTRIBUTION

(not a part of the original PowerPoint presentation)

Files are striped into many objects

- (ino, ono) → an object id (oid)

Ceph maps objects into placement groups (PGs)

- hash(oid) & mask → a placement group id (pgid)

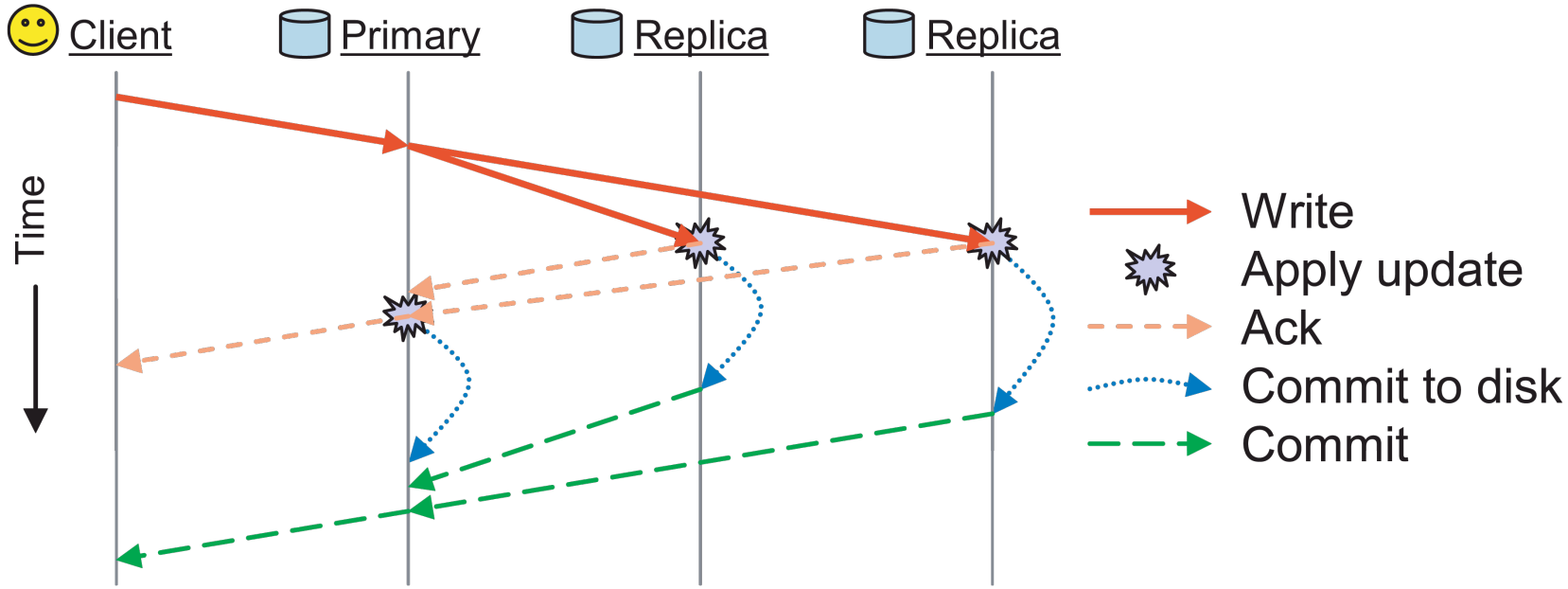
CRUSH assigns placement groups to OSDs

- CRUSH(pgid)→ a replication group, (osd1, osd2)

REPLICATION

Objects are replicated on OSDs within same PG

- Client is oblivious to replication



FAILURE DETECTION AND RECOVERY

Down and Out

Monitors check for intermittent problems

New or recovered OSDs peer with other OSDs within PG

ACRONYMS USED IN PERFORMANCE SLIDES

CRUSH: Controlled Replication Under Scalable Hashing

EBOFS: Extent and B-tree based Object File System

HPC: High Performance Computing

MDS: MetaData server

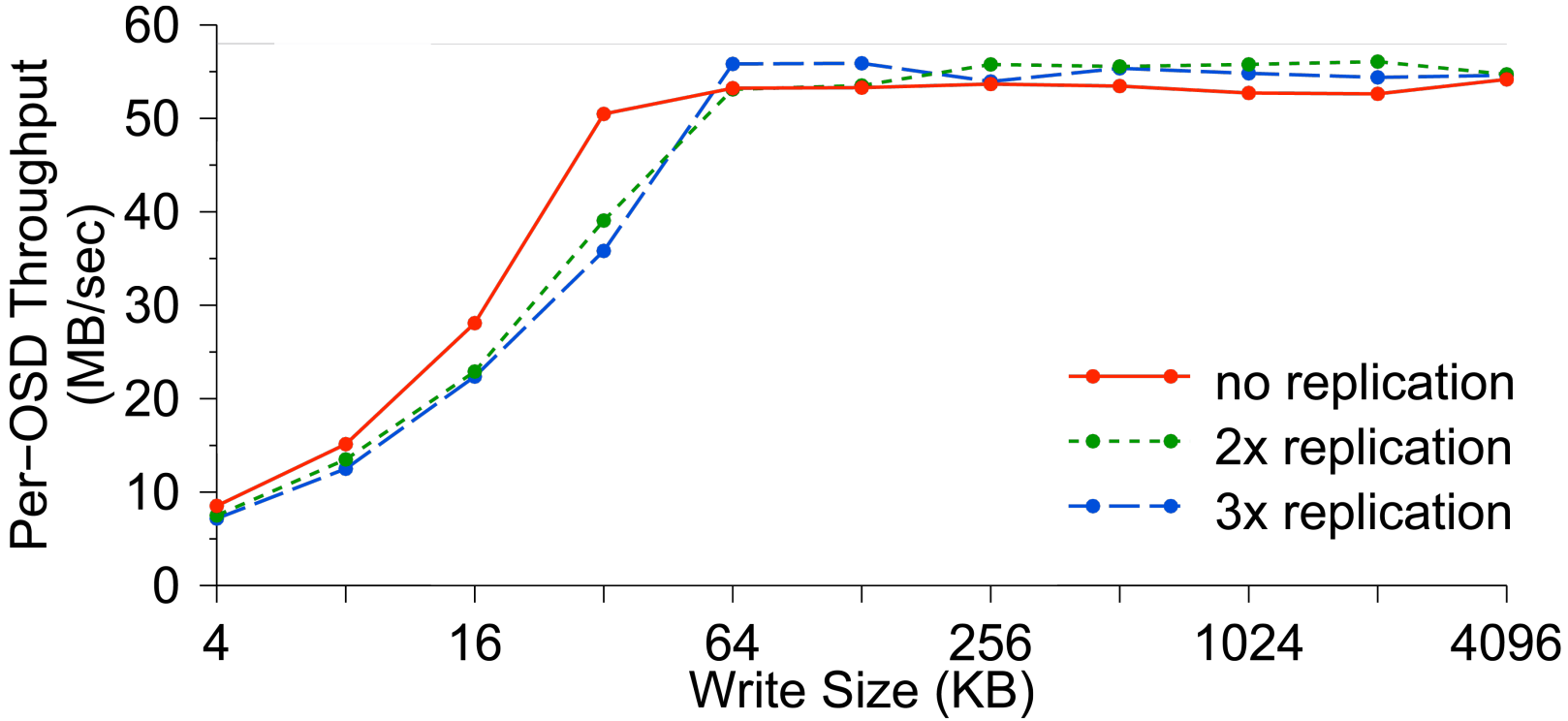
OSD: Object Storage Device

PG: Placement Group

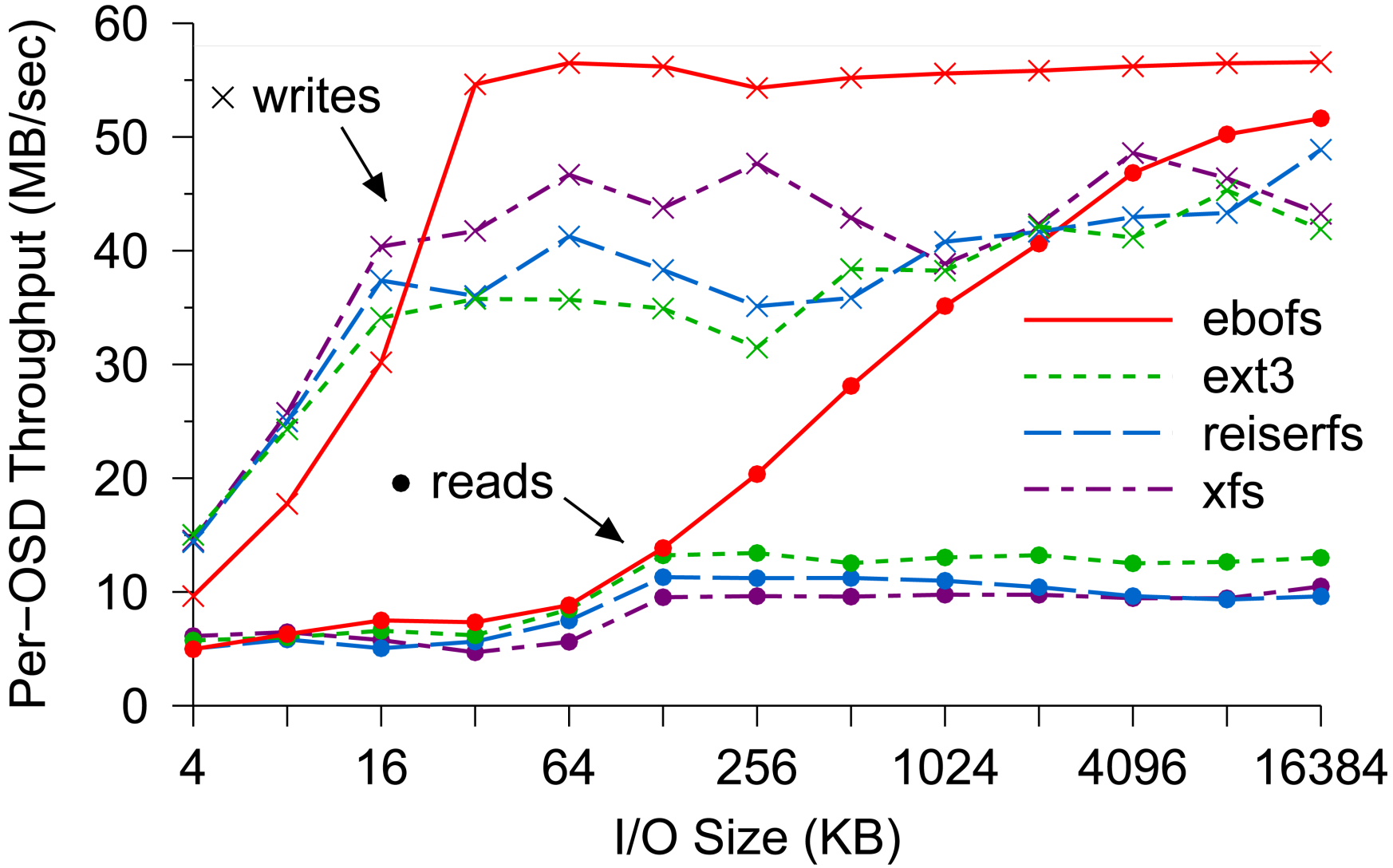
POSIX: Portable Operating System Interface for uniX

RADOS: Reliable Autonomic Distributed Object Store

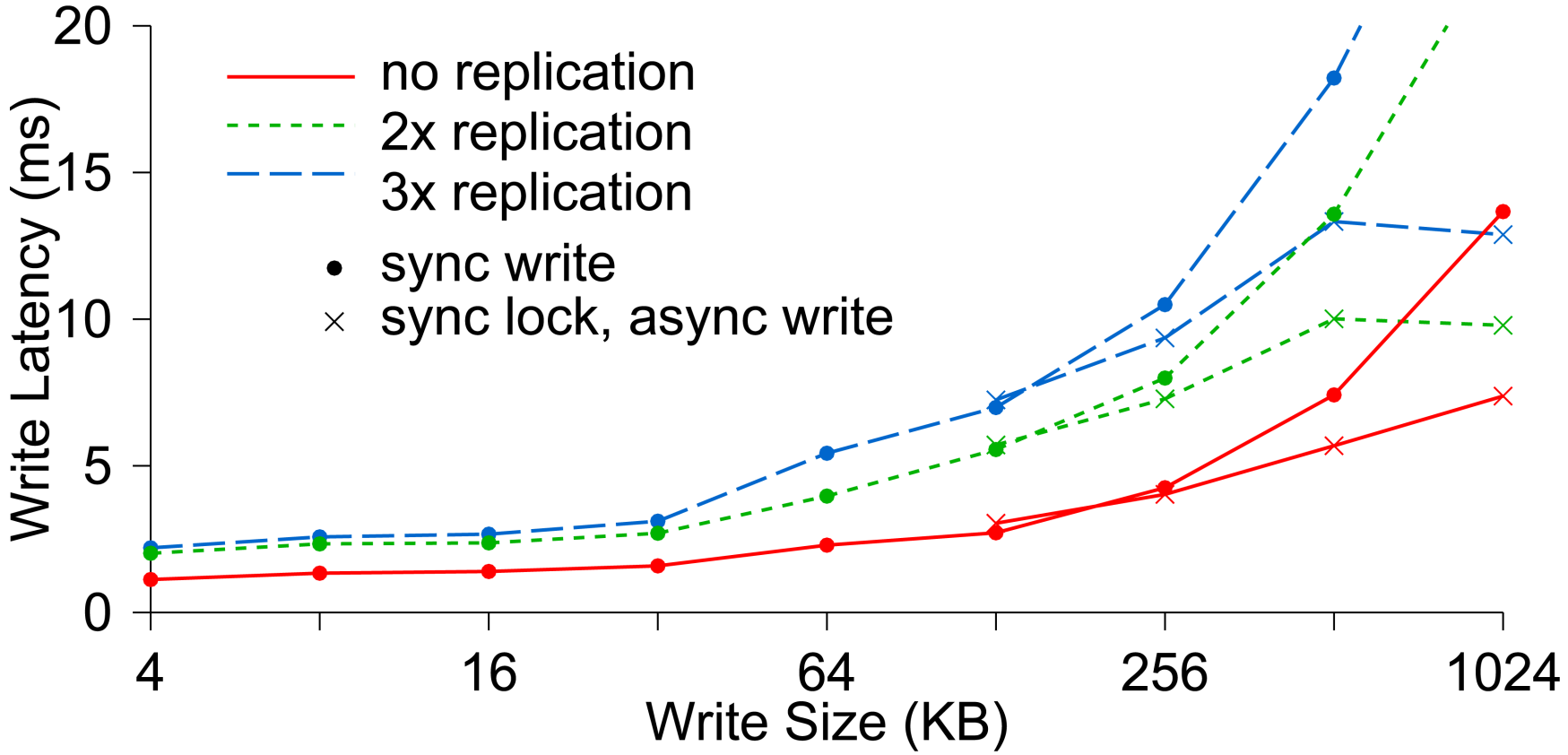
PER-OSD WRITE PERFORMANCE



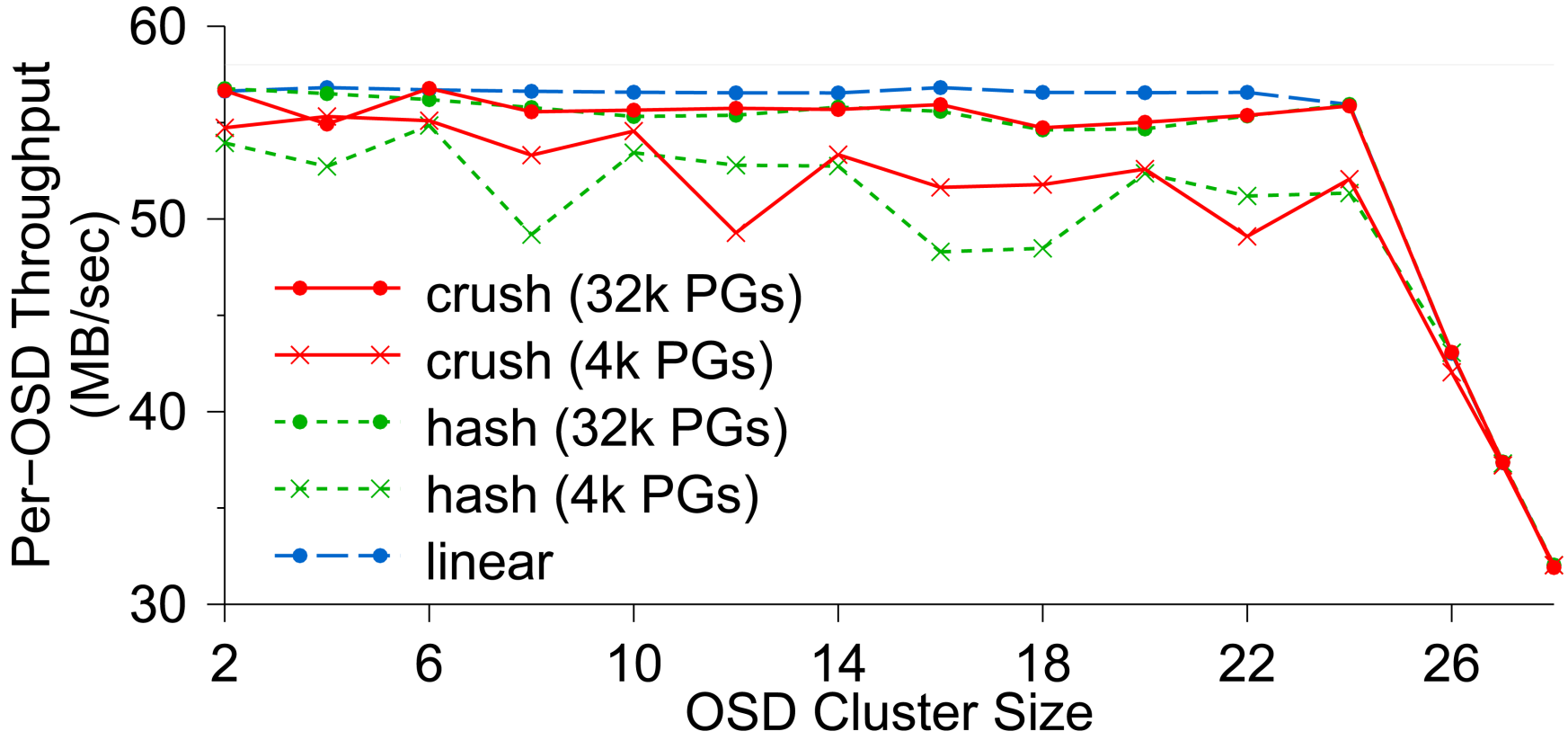
EBOFS PERFORMANCE



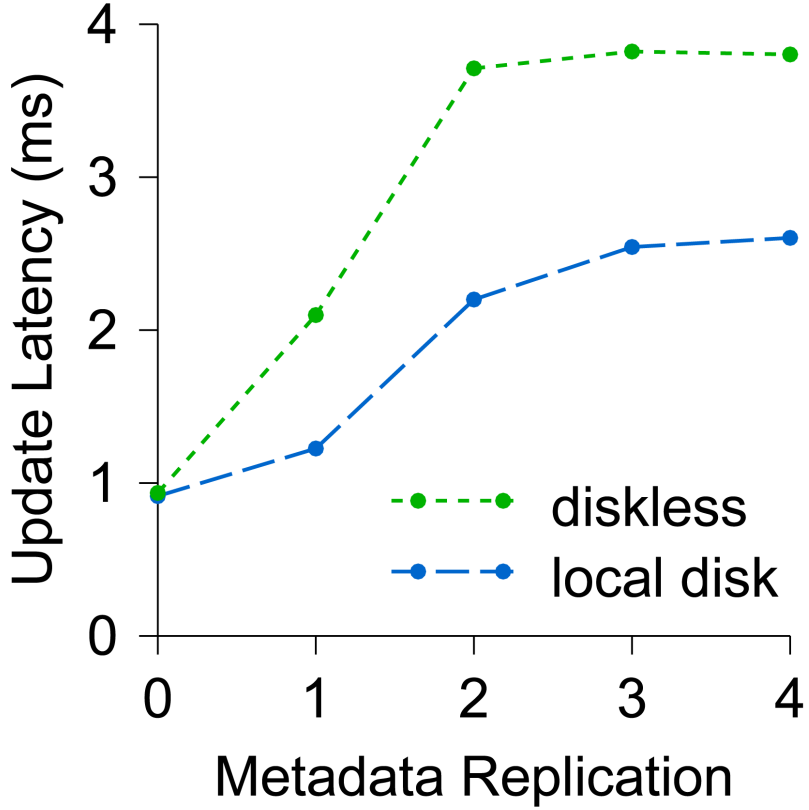
WRITE LATENCY

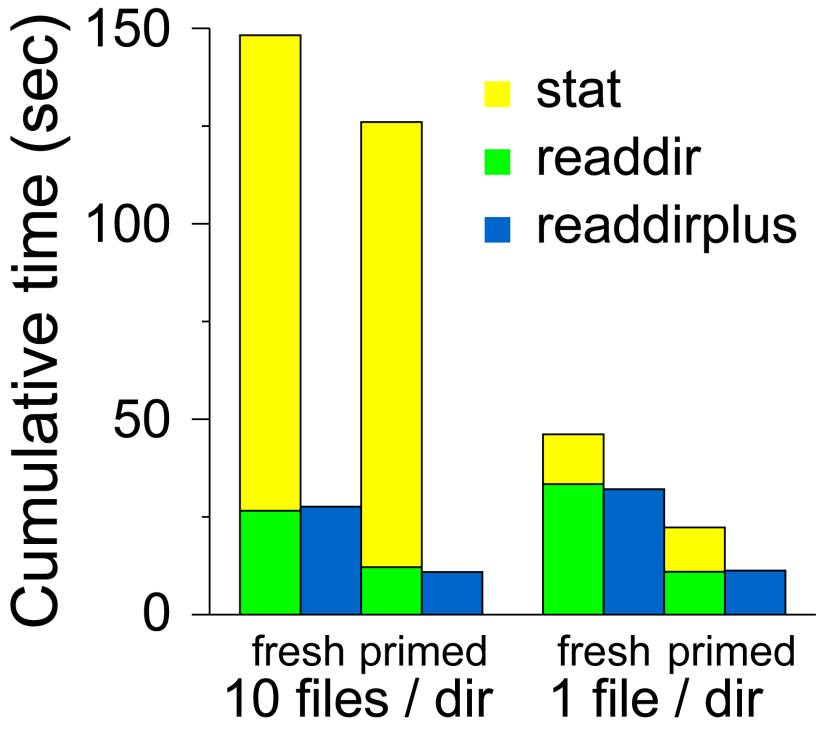


OSD WRITE PERFORMANCE



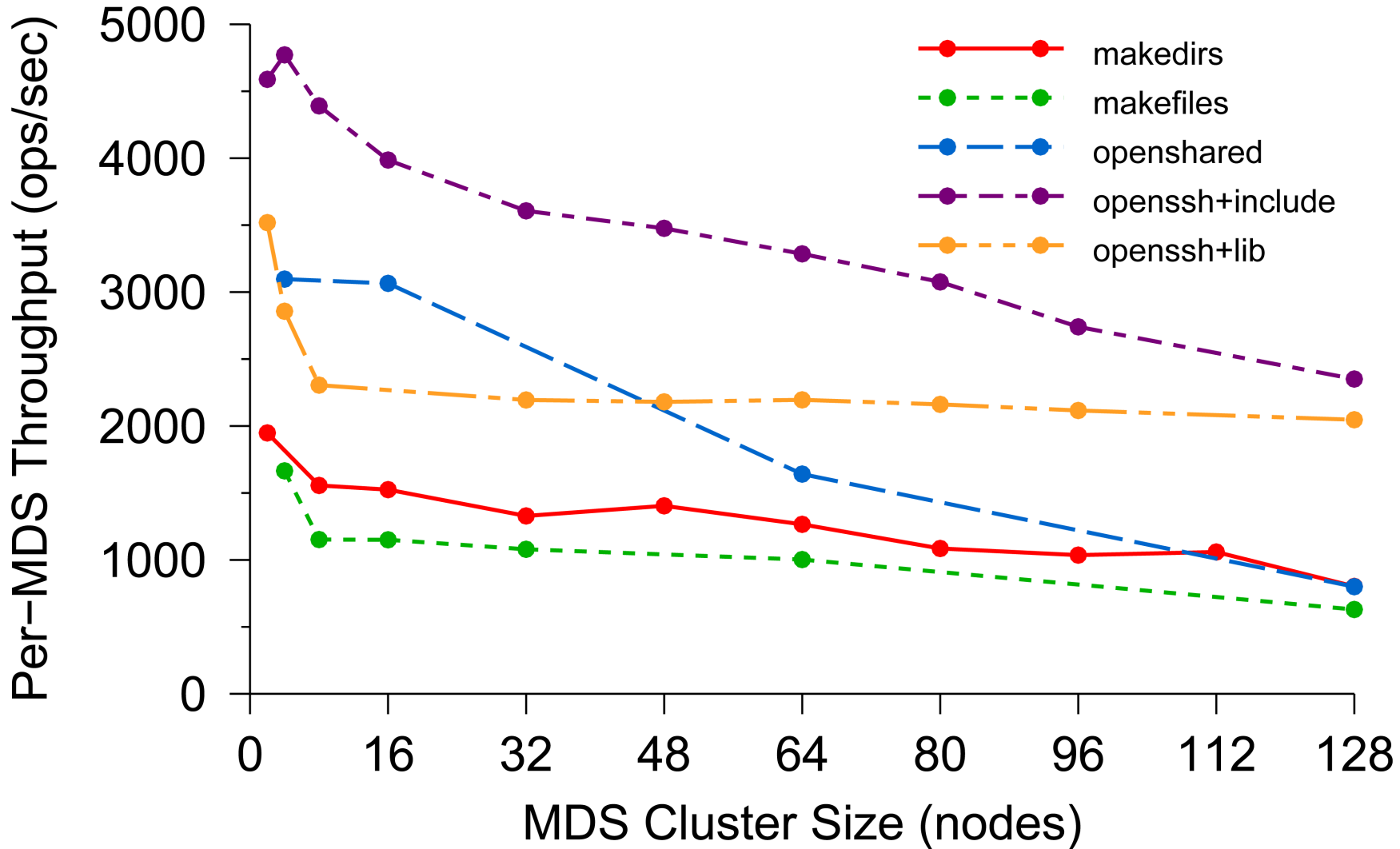
DISKLESS VS. LOCAL DISK



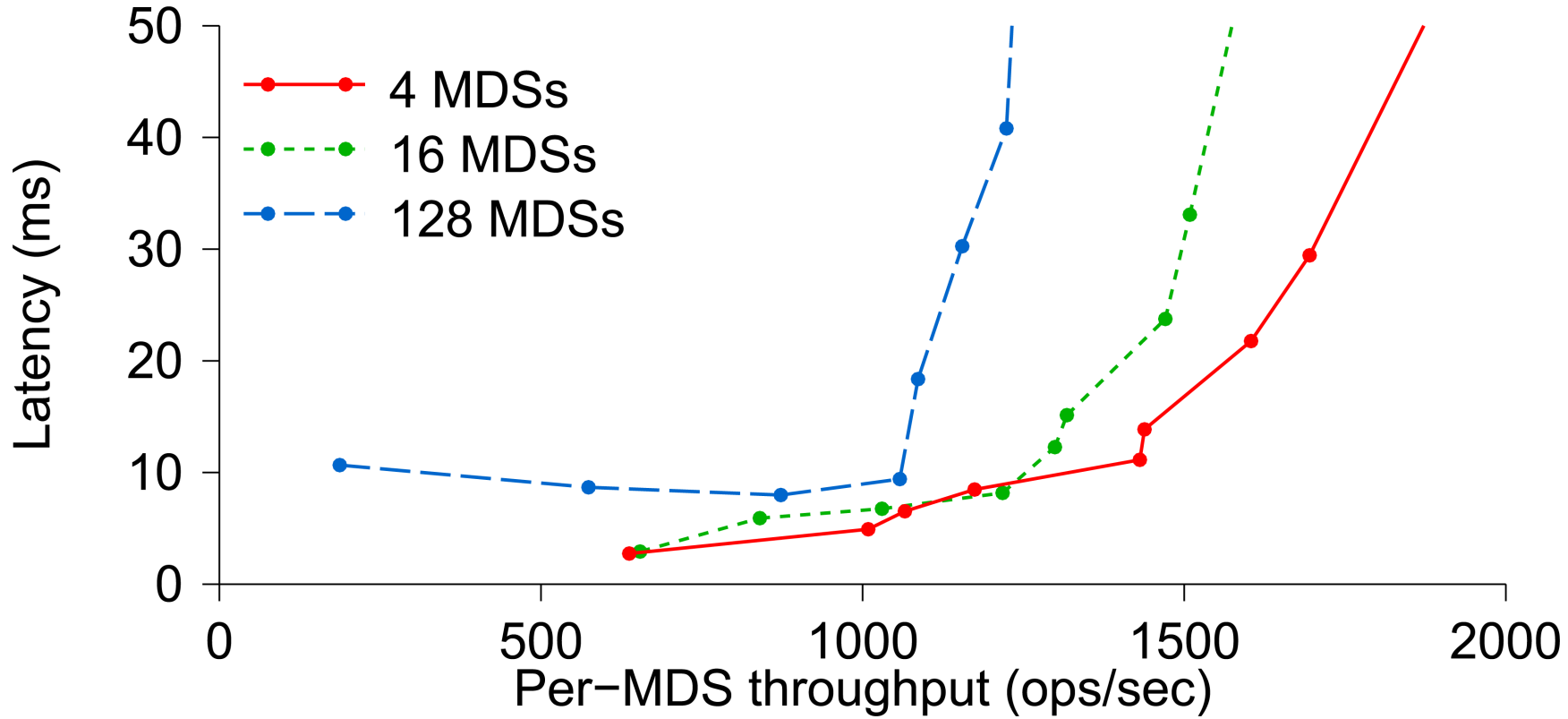


Compare latencies of (a) a MDS where all metadata are stored in a shared OSD cluster and (b) a MDS which has a local disk containing its journaling

PER-MDS THROUGHPUT



AVERAGE LATENCY



LESSONS LEARNED

(not a part of the original PowerPoint presentation)

Replacing file allocation metadata with a globally known distribution function was a good idea

- Simplified our design

We were right not to use an existing kernel file system for local object storage

The MDS load balancer has an important impact on overall system scalability but deciding which mtadata to migrate where is a difficult task

Implementing the client interface was more difficult than expected

- Idiosyncrasies of FUSE

CONCLUSION

Scalability, Reliability, Performance

Separation of data and metadata

- CRUSH data distribution function

Object based storage (some call it “software defined storage” these days)

CEPH IS WIDELY USED!

What has the experience been?

These next slides are from a high-performance computing workshop at CERN and will help us see how a really cutting-edge big-data use looks.

CERN is technically "aggressive" and very sophisticated. They invented the World Wide Web!


CERN

MANILA ON CEPHFS AT CERN

OUR WAY TO PRODUCTION

Arne Wiebalck

Dan van der Ster

OpenStack Summit

Boston, MA, U.S.

May 11, 2017

ABOUT CERN

European Organization for Nuclear Research

(**C**onseil **E**uropéen pour la **R**echerche **N**ucléaire)

- Founded in 1954

- World's largest particle physics laboratory

- Located at Franco-Swiss border near Geneva

- ~2'300 staff members
  >12'500 users

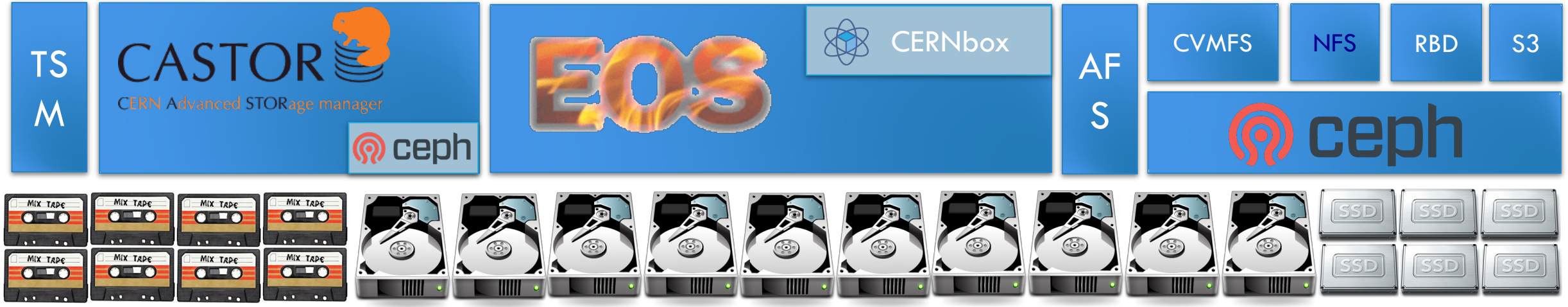
Primary mission:

Find answers to some of the fundamental questions about the universe!

THE CERN CLOUD AT A GLANCE

- Production service since July 2013
  - Several rolling upgrades since, now on Newton
- Two data centers, 23ms distance
  - One region, one API entry point
- Currently ~220'000 cores
  - 7'000 hypervisors (+2'000 more soon)
  - ~27k instances
- 50+ cells
  - Separate h/w, use case, power, location, …

SETTING THE SCENE: STORAGE



CASTOR
CERN Advanced STORage manager

HSM Data Archive

Developed at CERN

140PB – 25k tapes

EOS

Data Analysis

Developed at CERN

120PB – 44k HDDs

CERNBox

File share & sync

Owncloud/EOS

9'500 users

ceph

OpenStack backend

CephFS

Several PB-sized clusters

NFS Filer

OpenZFS/RBD/OpenStack

Strong POSIX

Infrastructure Services

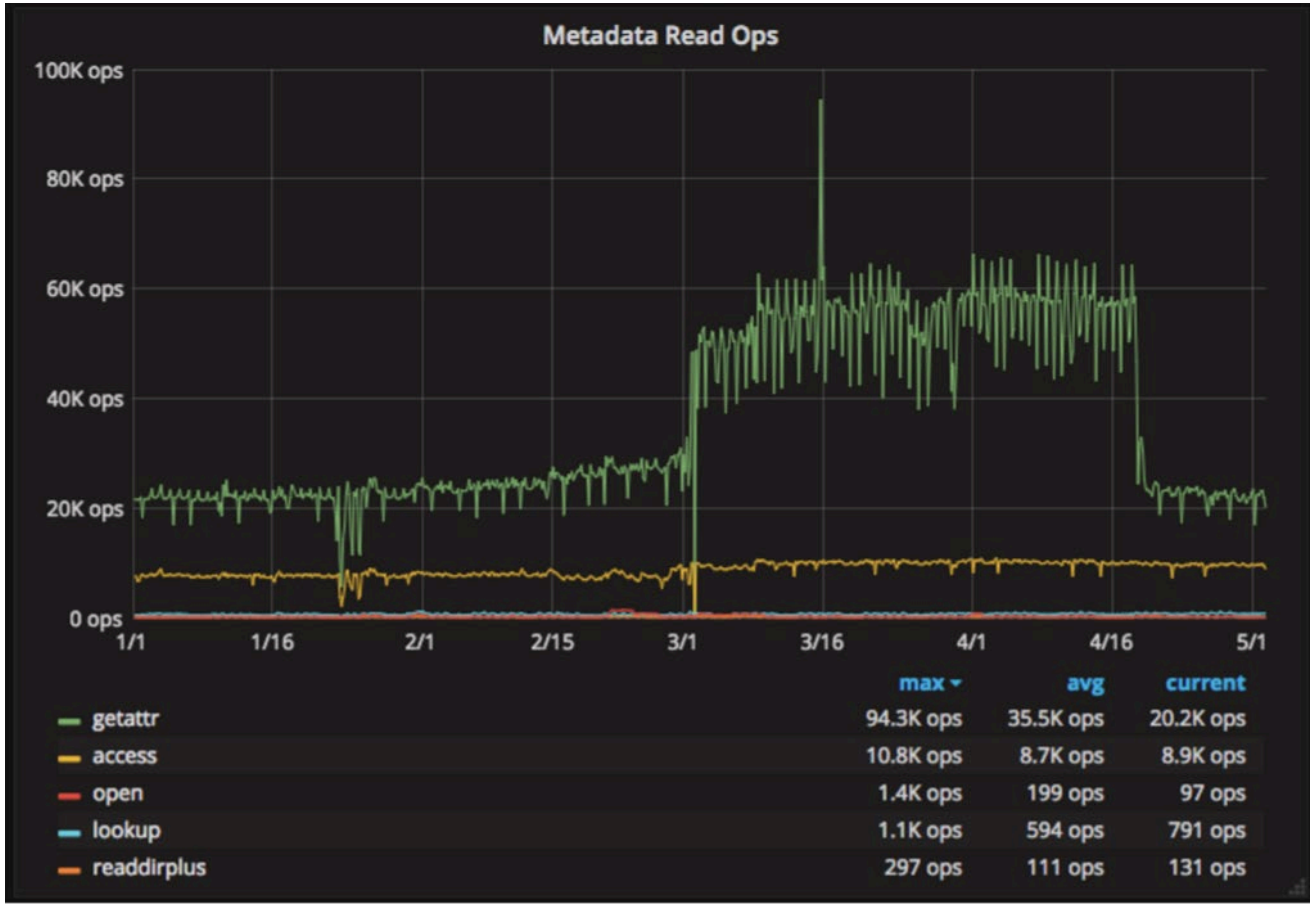
NFS FILER SERVICE OVERVIEW

- NFS appliances on top of OpenStack
  - Several VMs with Cinder volume and OpenZFS
  - ZFS replication to remote data center
  - Local SSD as accelerator (l2arc, ZIL)

- POSIX and strong consistency
  - Puppet, GitLab, OpenShift, Twiki, …
  - LSF, Grid CE, Argus, …
  - BOINC, Microelectronics, CDS, …

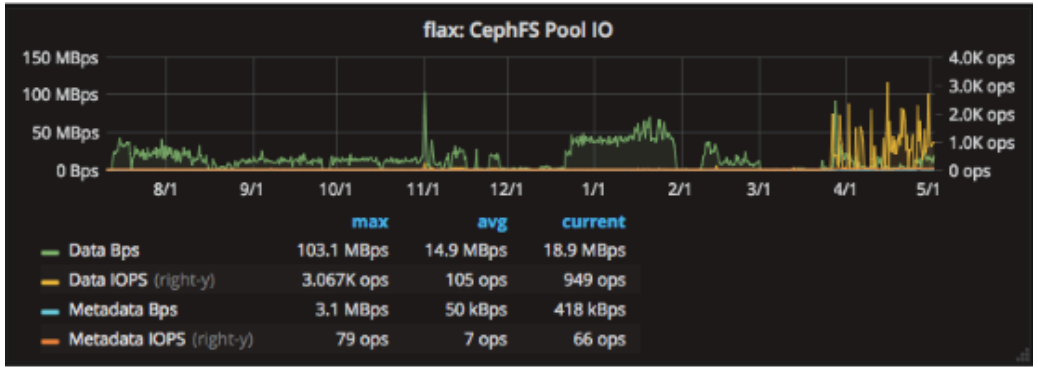
NFS FILER SERVICE LIMITATIONS

- Scalability
  - Metadata Operations (read)
- Availability
  - SPoF per NFS volume
  - ‘shared block device’
- Emerging use cases
  - HPC, see next slide



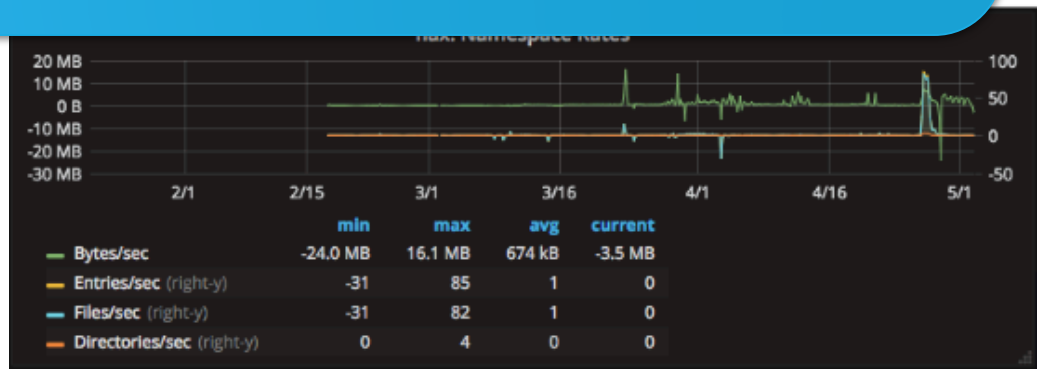
USE CASES: HPC

- MPI Applications
  - Beam simulations, accelerator physics, plasma simulations, computation fluid dynamics, QCD …
-
- Dedicated CephFS cluster in 2018
  - 3-node cluster, 150TB usable
  - RADOS: quite low activity
  - 52TB used, 8M files, 350k dirs
  - <100 file creations/sec



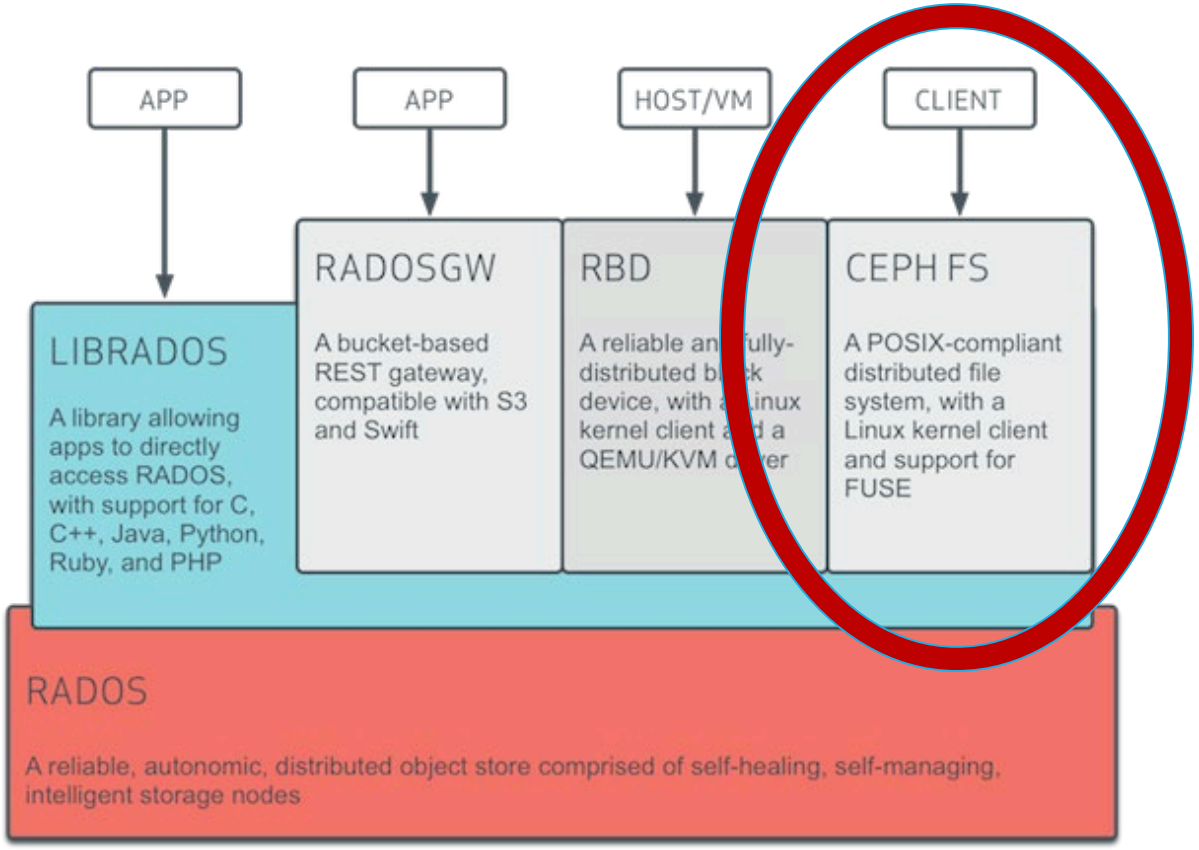
Can we converge on CephFS ?

(and if yes, how do we make it available to users?)



PART I: THE CEPHFS BACKEND

CEPHFS OVERVIEW



POSIX-compliant shared FS on top of RADOS

- Same foundation as RBD

Userland and kernel clients available

- 'ceph-fuse' and 'mount -t ceph'
- Features added to ceph-fuse first, then ML kernel

'jewel' release tagged production-ready

- April 2016, main addition: fsck
- 'Almost awesome' before, focus on object and block

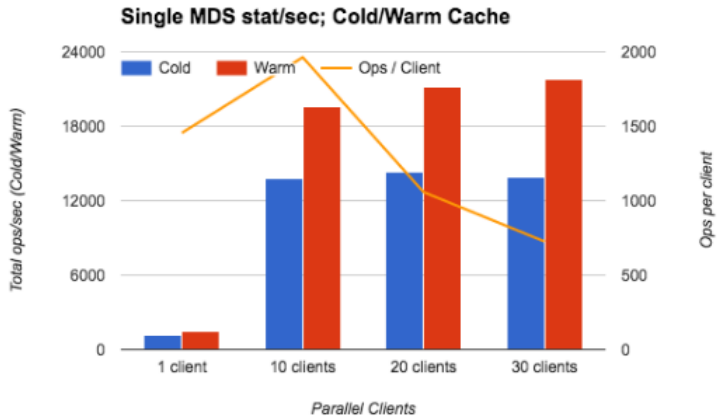
CEPHFS META-DATA SERVER (MDS)

- Crucial component to build a fast and scalable FS
  - Creates and manages inodes (persisted in RADOS, but cached in memory)
  - Tracking client inode 'capabilities' (which client is using which inodes)
- Larger cache can speed up meta-data throughput
  - More RAM can avoid meta-data stalls when reading from RADOS
- Single MDS can handle limited number of client reqs/sec
  - Faster network / CPU enables more reqs/sec, but multiple MDSs needed for scaling
  - MDS keeps nothing in disk, a flash-only RADOS pool may accelerate meta-data intensive workloads

CEPHFS MDS TESTING

- Correctness and single-user performance
  - POSIX compliance: ok! (Tuxera POSIX Test Suite v20090130)
  - Two-client consistency delays: ok! (20-30ms with fsping)
  - Two-client parallel IO: reproducible slowdown

- Mimic Puppet master
  - 'stat' all files in prod env from mutliple clients
  - 20k stats/sec limit



- Meta-data scalability in multi-user scenarios
  - Multiple active MDS fail-over w/ 1000 clients: ok!
  - Meta-data load balancing heuristics: todo …

CEPHFS ISSUES DISCOVERED

- ‘ceph-fuse’ crash in quota code, fixed in 10.2.5
  - http://tracker.ceph.com/issues/16066
- Bug when creating deep directores, fixed in 10.2.6
  - http://tracker.ceph.com/issues/18008
- Bug when creating deep directores, fixed in 10.2.6
  - http://tracker.ceph.com/issues/18008
- Objectcacher ignores max objects limits when writing large files
  - http://tracker.ceph.com/issues/19343
- Network outage can leave ‘ceph-fuse’ stuck
  - Difficult to reproduce, have server-side work-around, ‘luminous’ ?
- Parallel IO to single file can be slower than expected
  - If CephFS detects several writers to the same file, it switches clients to unbuffered IO

Desired:

Quotas

- should be mandatory for userland and kernel

QoS

- throttle/protect users

‘CEPHFS IS AWESOME!’

- POSIX compliance looks good
- Months of testing: no ‘difficult’ problems
  - Quotas and QoS?
- Single MDS meta-data limits are close to our Filer
  - We need multi-MDS!
- ToDo: Backup, NFS Ganesha (legacy clients, Kerberos)
- ‘luminous’ testing has started …

PART II: THE OPENSTACK INTEGRATION

LHC INCIDENT IN APRIL 2016

INTERNATIONAL POLITIQUE SOCIÉTÉ ÉCO CULTURE IDÉES PLANÈTE SPORT SCIENCES PIXELS

M Sciences

Le Monde

SCIENCES Vidéos Archéologie Supplément partenaire : Les Prix EDF Pulse Affaire de logique Astronor

Une fouine à l'origine d'une panne dans le plus grand accélérateur de particules du monde

Les réparations du LHC prendront plusieurs jours, rapporte le CERN.

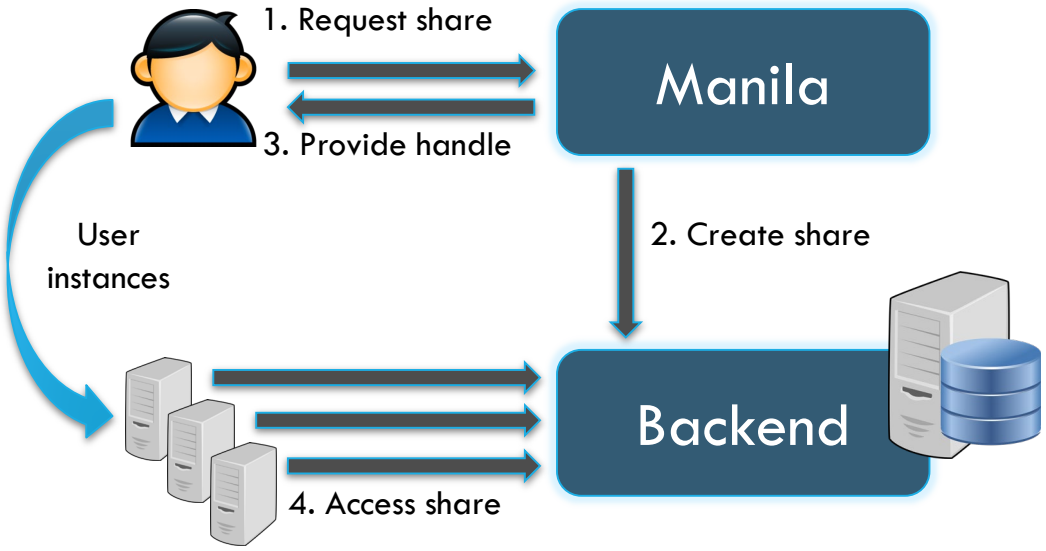
Le Monde.fr avec AFP | 30.04.2016 à 12h35

Abonnez vous à partir de 1 € Réagir Ajouter Partager (4 419) Tweeter

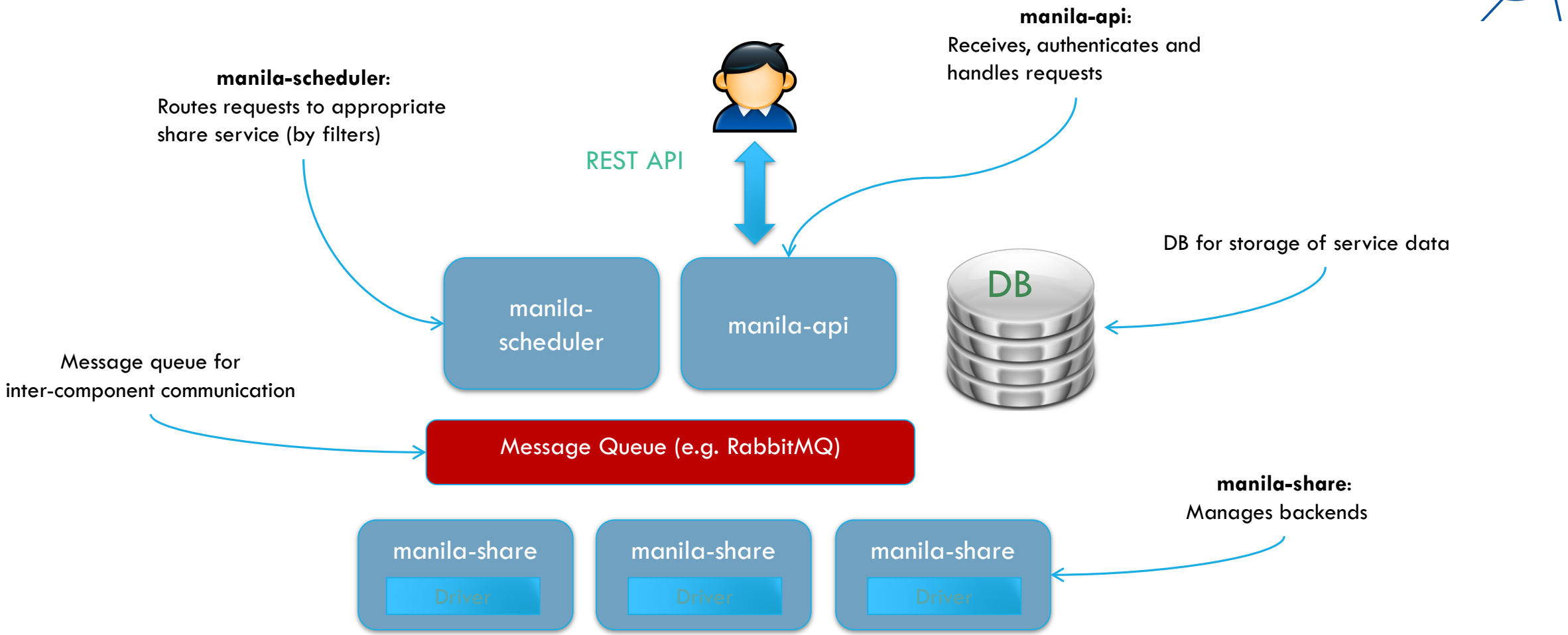
MANILA: OVERVIEW

- File Share Project in OpenStack
  - Provisioning of shared file systems to VMs
  - ‘Cinder for file shares’
- APIs for tenants to request shares
  - Fulfilled by backend drivers
  - Acessed from instances
- Support for variety of NAS protocols
  - NFS, CIFS, MapR-FS, GlusterFS, **CephFS**, …
- Supports the notion of share types
  - Map features to backends

MANILA
an OpenStack Community Project



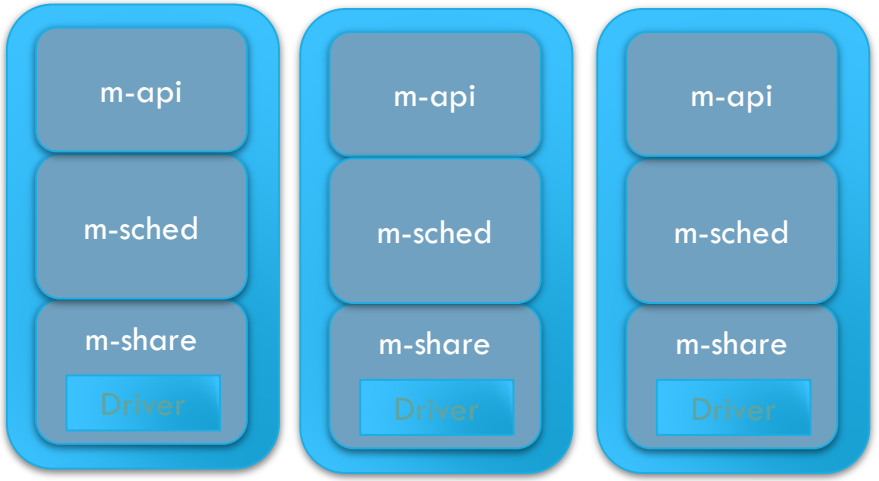
MANILA: COMPONENTS



(Not shown: **manila-data**: copy, migration, backup)

MANILA: OUR INITIAL SETUP

- Three controllers running
  m-{api, scheduler, share}
- Separate Rabbit cluster
- DB from external service
- Existing CephFS backend
- Set up and 'working' in <1h !

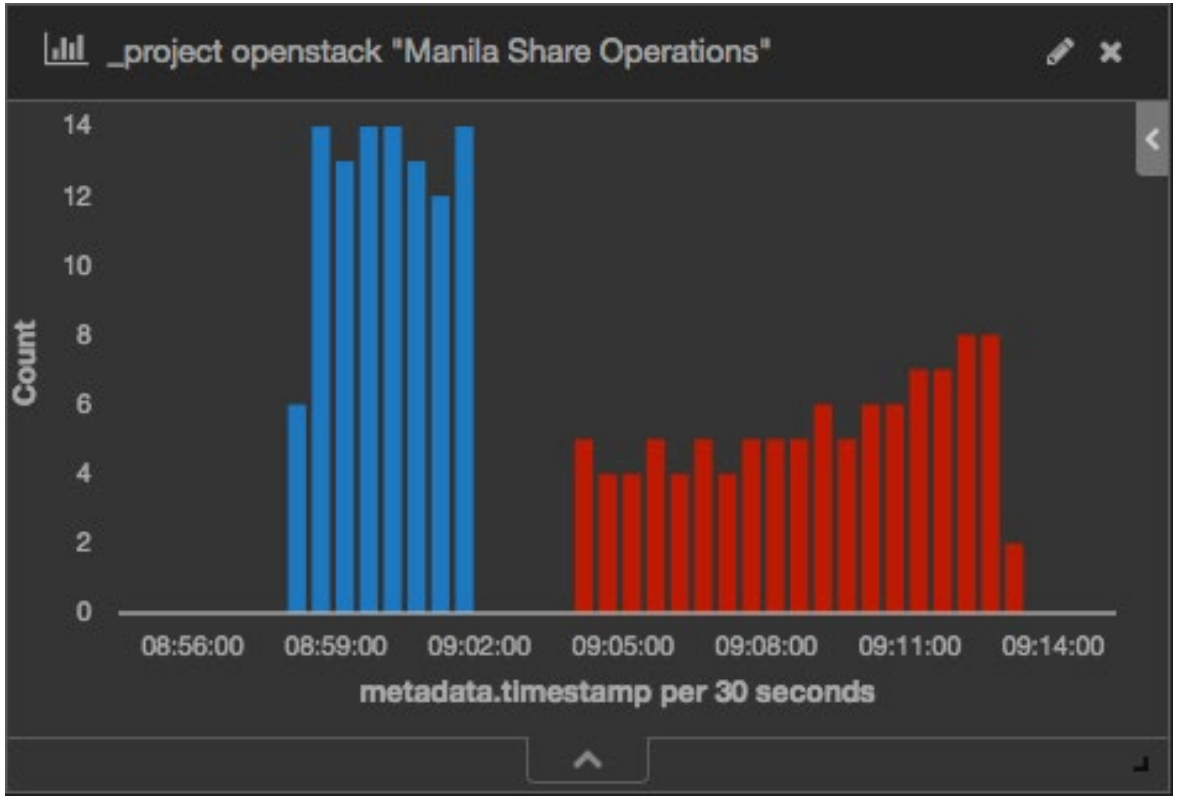


RabbitMQ

Our Cinder setup has been changed as well …

SEQUENTIAL SHARE CREATION/DELETION: OK!

- Create: ~2sec
- Delete: ~5sec
- “manila list” creates two auth calls
  - First one for discovery
  - Cinder/Nova do only one … ?

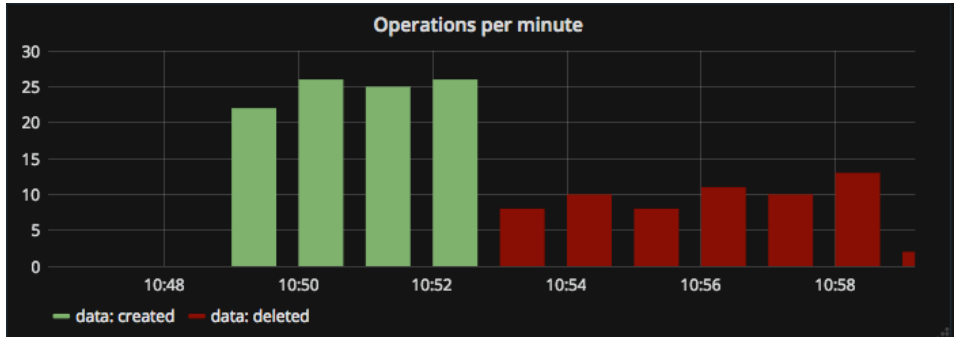


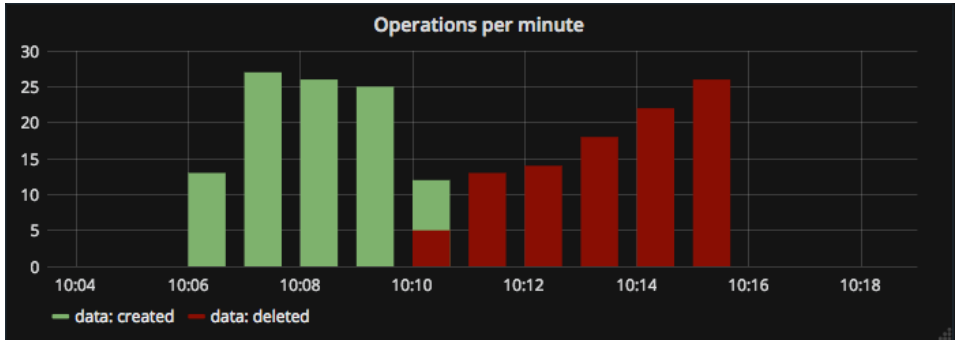
BULK DELETION OF SHARES: OK!

- *Sequentially*
  - manila delete share-01
  - manila delete share-02
  - manila delete share-03
  - …
- *In parallel*
  - manila delete share-01 share-02 …
  - Done with 100 shares in parallel

This is what users (and Magnum/Heat/K8s) do …

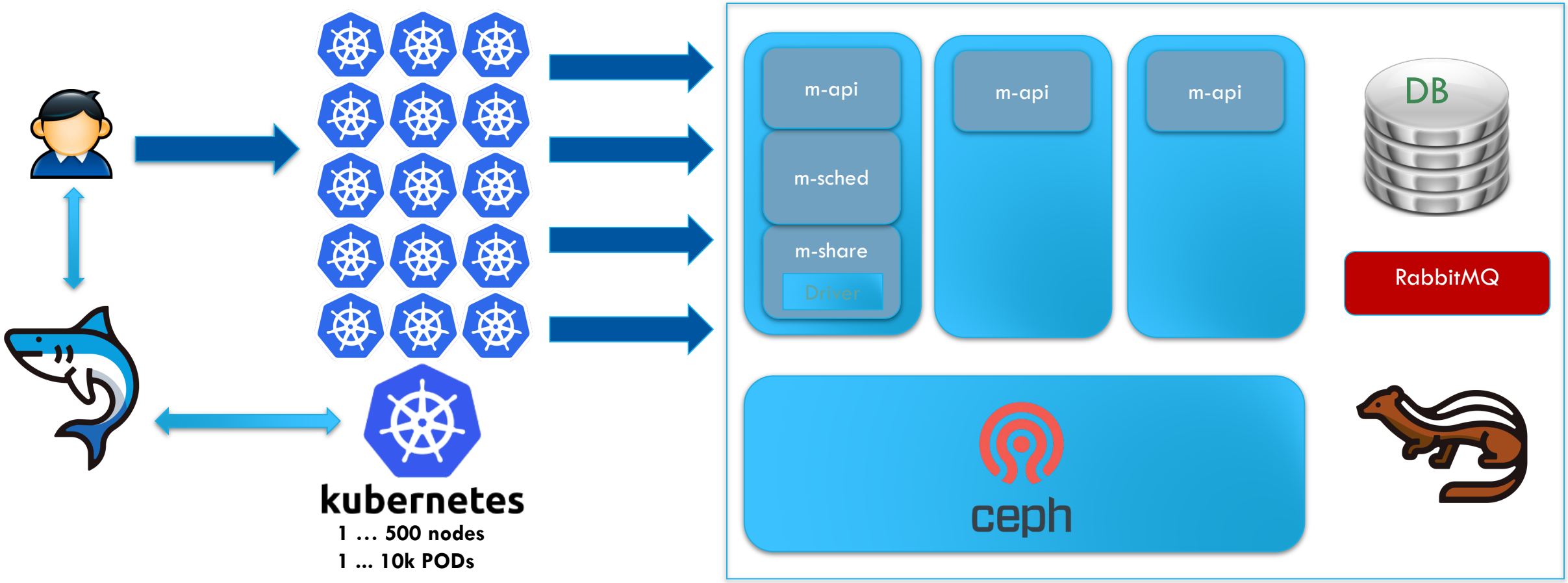
(… and this breaks our Cinder! Bug 1685818).





After successful 24h tests of constant creations/deletions …

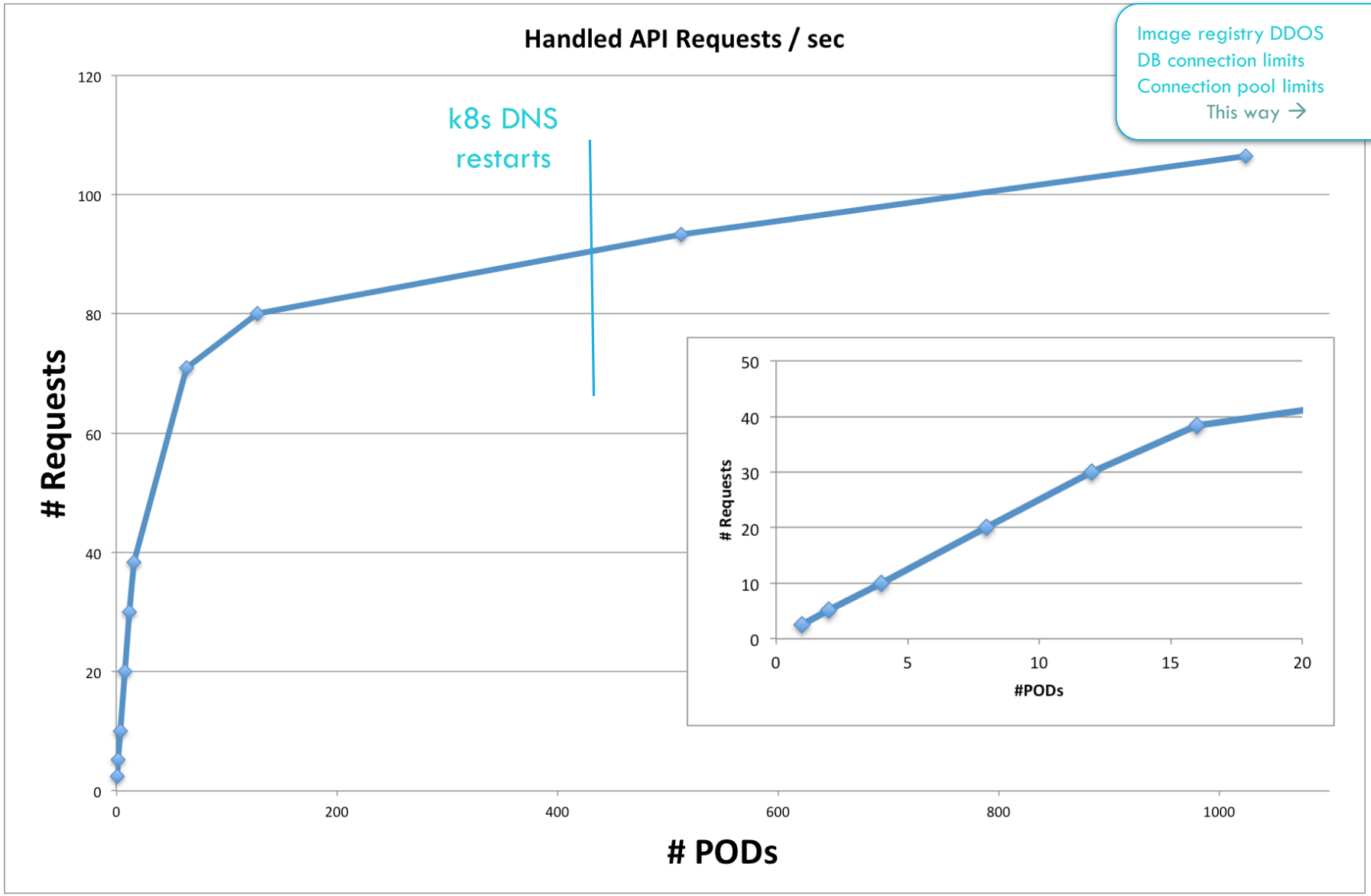
MANILA TESTING: #FOUINEHAMMER



Stressing the API (1)

PODs running
'manila list'
in a loop

~linear until API processes exhausted … ok!

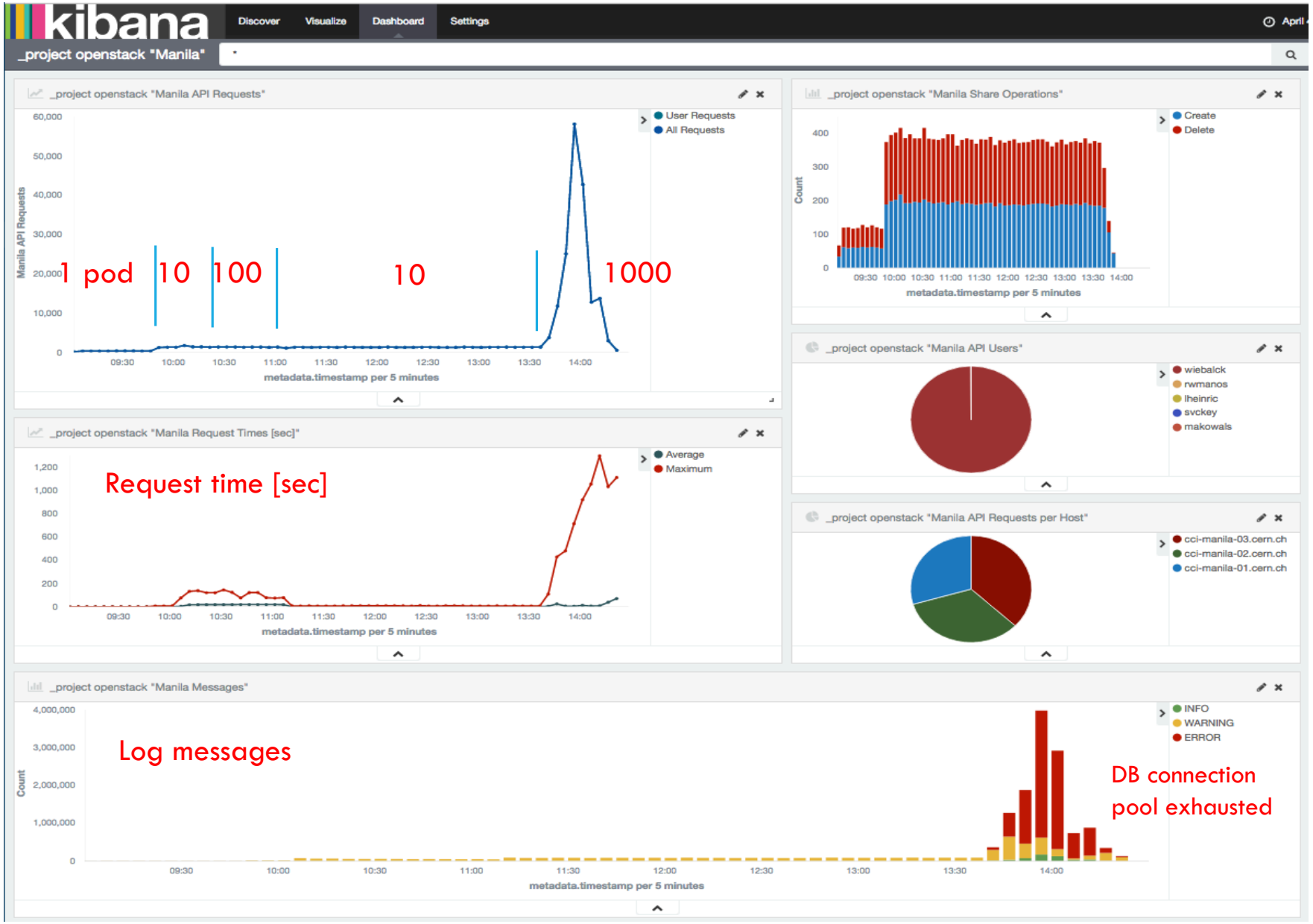


Stressing the API (2)

PODs running
‘manila create’
‘manila delete’
in a loop

~works until DB limits are reached …

ok!



COMPONENTS 'FOUINED' ON THE WAY

Right away: Image registry (when scaling the k8s cluster)

- 500 nodes downloading the image …
- Increased RAM on image registry nodes
- Used 100 nodes

~350 pods: Kubernetes DNS (~350 PODs)

- Constantly restarted: Name or service not known
- Scaled it out to 10 nodes

~1'000 pods: Central monitoring on Elastic Search

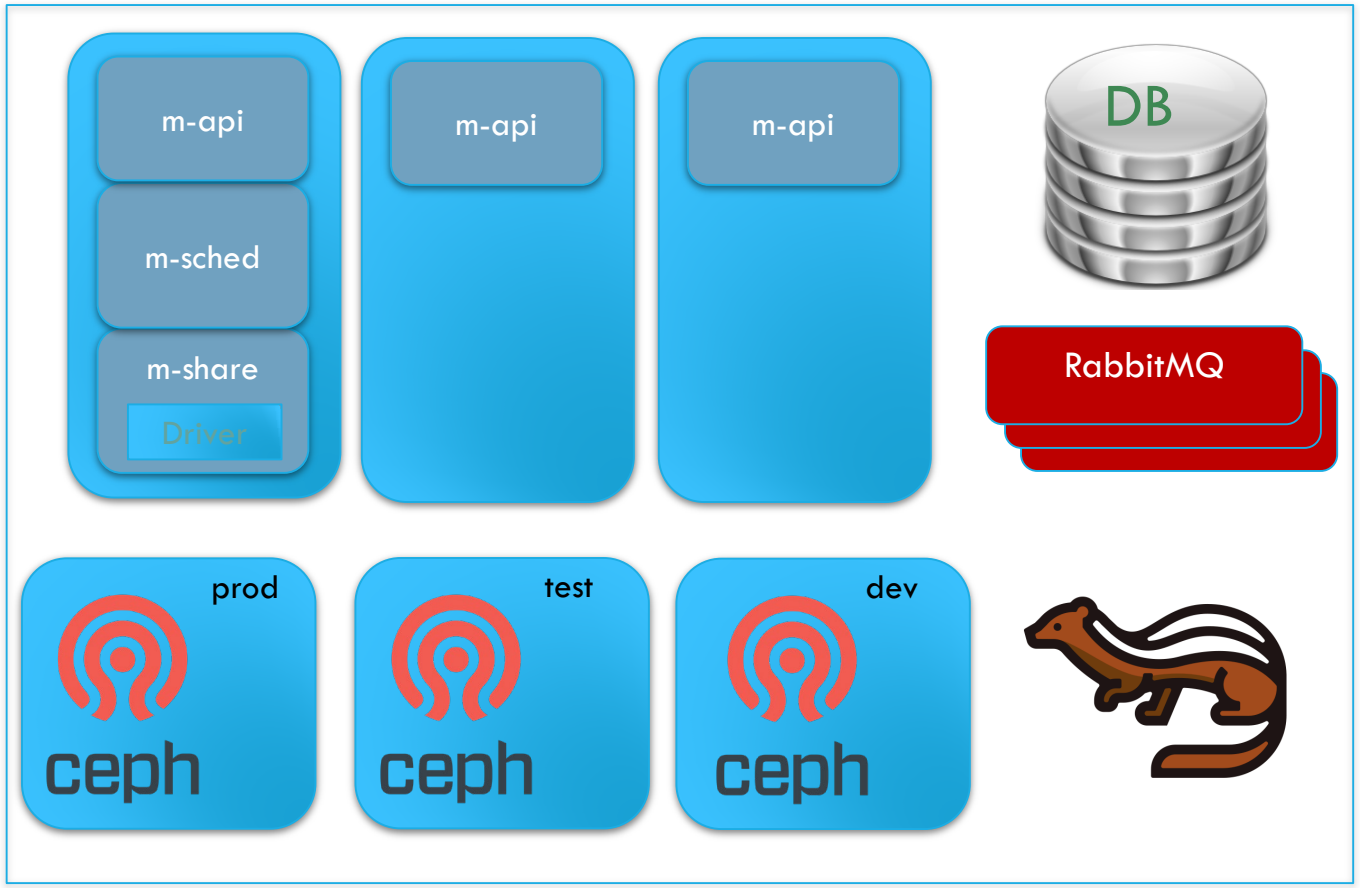
- Too many logs messages

~4'000 pods: Allowed DB connections (and the connection pool)

- Backtrace in Manila API logs

MANILA: OUR (PRE-)PROD SETUP

- Three virtual controllers
  - 4-core, 8GB
  - Newton
  - Puppet
- 3-node RabbitMQ cluster
  - V3.6.5
- Three share types
  - Test/Prod: Ceph 'jewel'
  - Dev: Ceph 'luminous'



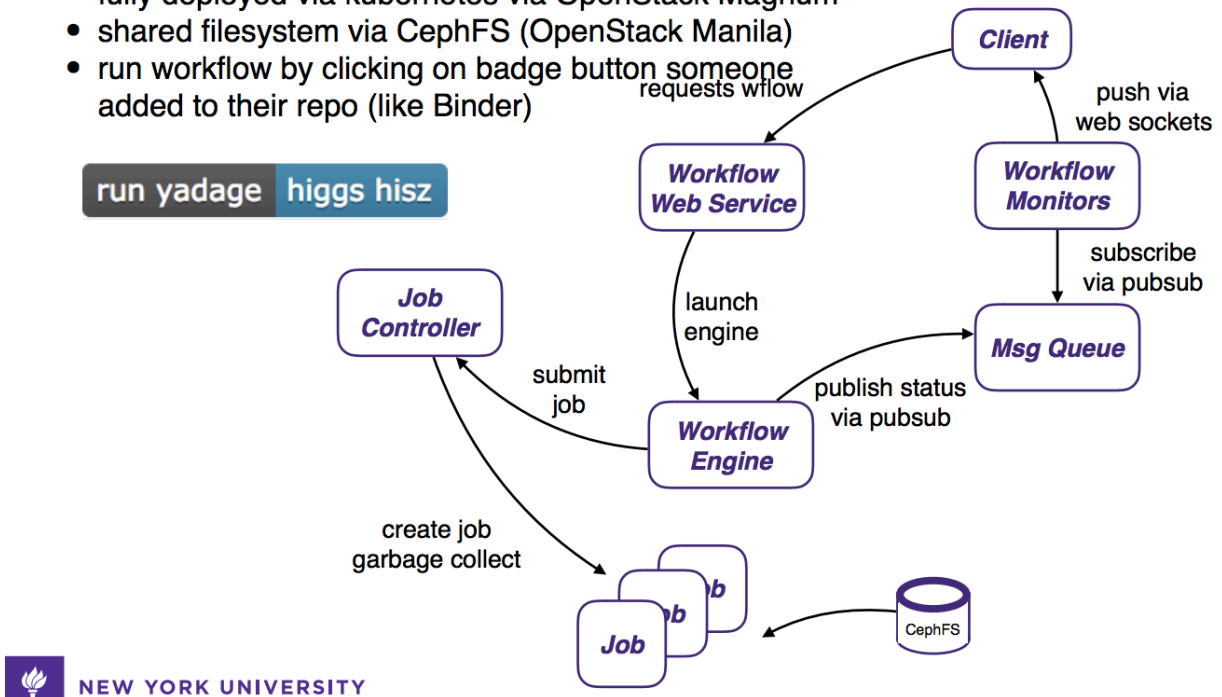
EARLY USER: CONTAINERS IN ATLAS

- Analysis workflows modeled as directed acyclic graphs, built at run-time
  - Graph's nodes are containers
- Addresses the issue of workflow preservation
  - Store the parametrized container workload
- Built using a k8s cluster
  - On top of Magnum
- Using CephFS to store intermediate job stages
  - Share creation via Manila
  - Leverageing CephFS integration in k8s

REANA infrastructure

- fully deployed via kubernetes via OpenStack Magnum
- shared filesystem via CephFS (OpenStack Manila)
- run workflow by clicking on badge button someone added to their repo (like Binder)

run yadage | higgs hisz



NEW YORK UNIVERSITY

Lukas Heinrich: Containers in ATLAS

‘MANILA IS ALSO AWESOME!’

- Setup is straightforward
- No Manila issues during our testing
  - Functional as well as stress testing
- Most features we found missing are in the plans
  - Per share type quotas, service HA
  - CephFS NFS driver
- Some features need some more attention
  - **OSC integration,** CLI/UI feature parity, AuthID goes with last share
- Welcoming, helpful team!

CERN

QUESTIONS?

Arne.Wiebalck@cern.ch
@ArneWiebalck