EOS News & Storage Federations

Andreas-Joachim Peters
CERN - IT
Storage Group

andreas.Joachim.Peters@cern.ch

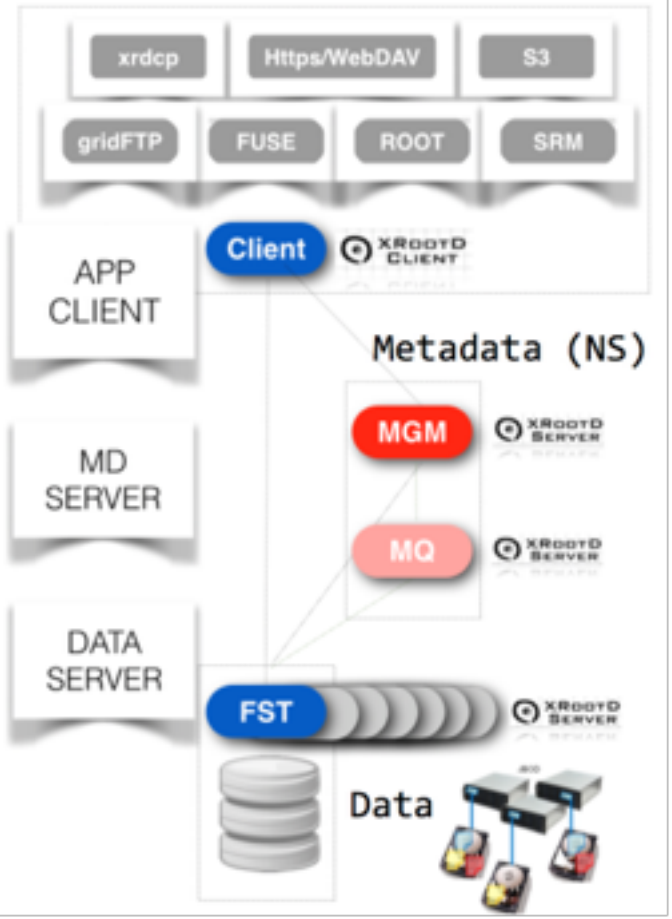
ALICE Tier 1/2/ Workshop 2017

Contents

- Introduction
- EOS@CERN
  - Alice EOS Instance
  - Disk Space Allocation
- Development
  - News
  - R&D
  - Roadmap / Milestones
- EOS via Docker … storage in a minute
- Storage Federations
  - implementation in EOS
  - examples & benchmarks
  - when (not) to use them

Introduction

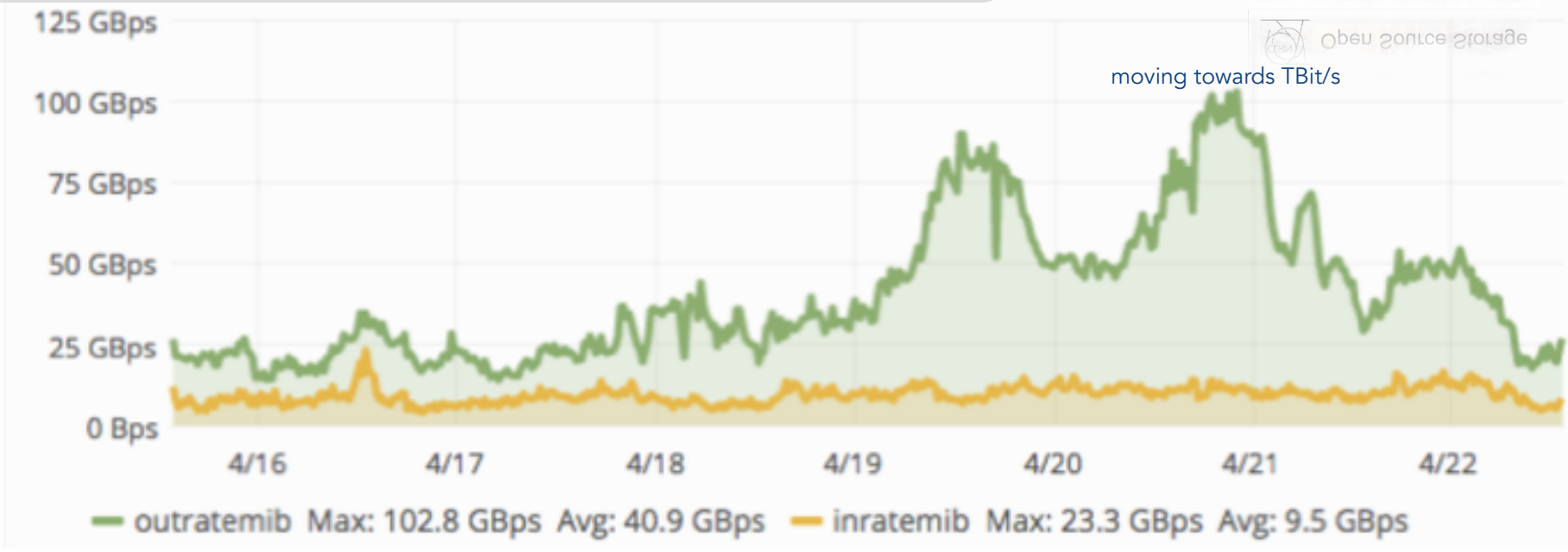
- Project started in 2010
- Licence free
- Simple and scalable solution
- Easy to operate
- In-memory namespace
- Secure access (krb5, gsi)
- Quotas (user/group)
- Network RAID (RAIN)
- Tuneable QoS
- Dev&Ops in CERN/IT-ST

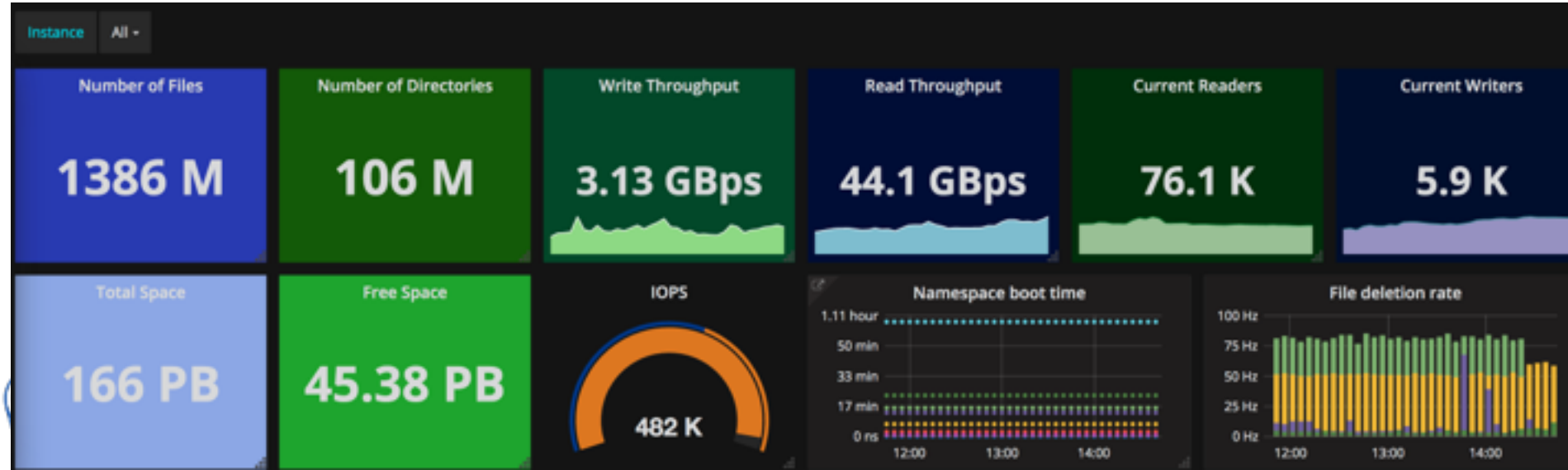


EOS at CERN

Open Source Storage

moving towards TBit/s





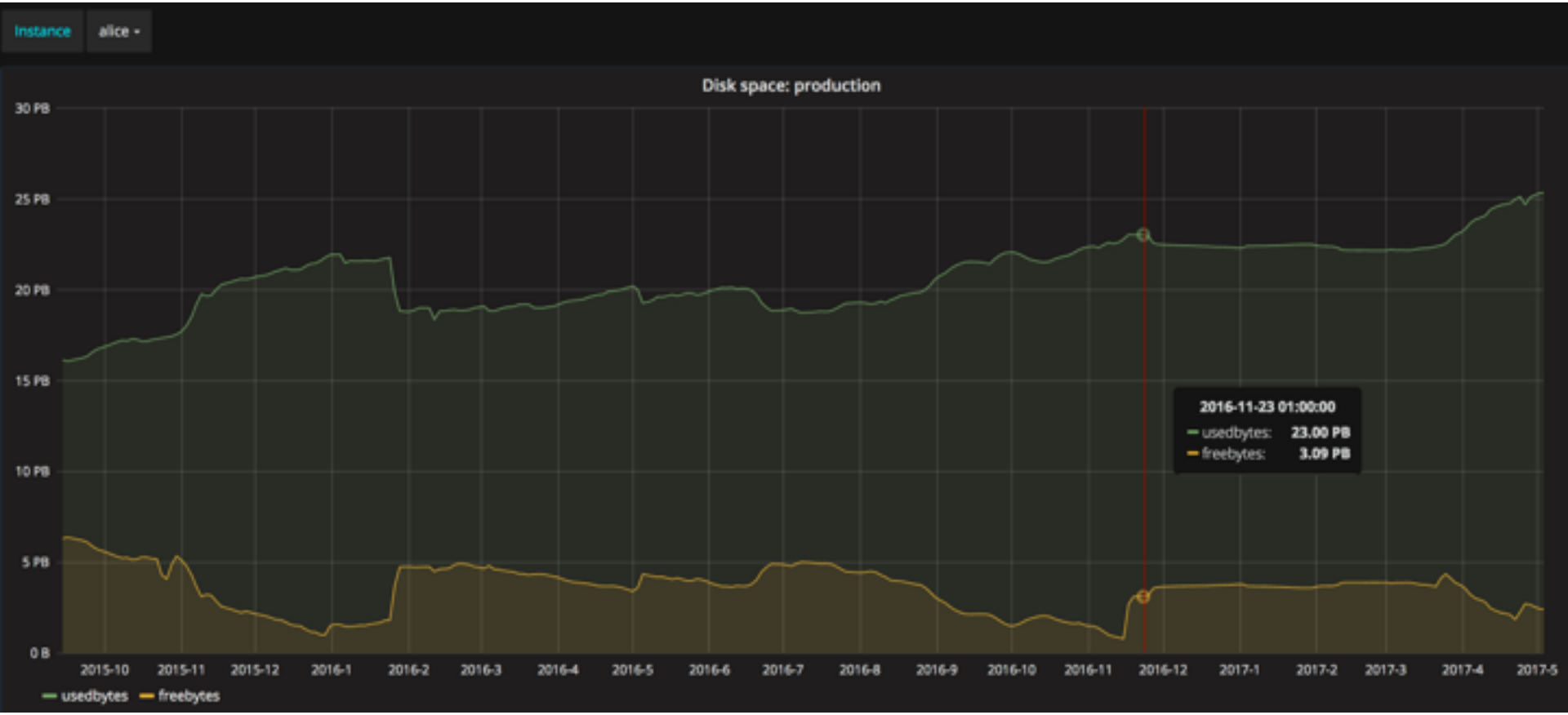
EOSALICE

namespace headnodes upgraded from 256G with HDD to 512G with SSD

s

EOSALICE

- Disk Space in Production (divide by 2)



CERN Disk Space Allocation

Experiment/Group	status Feb 2017	pledge 2017 (May)	request 2018
ALICE	17600	22400	27400
ATLAS	18500	25000	26000
CMS	22700	24600	26100
LHCb	7800	10900	12000

EOS + Castor

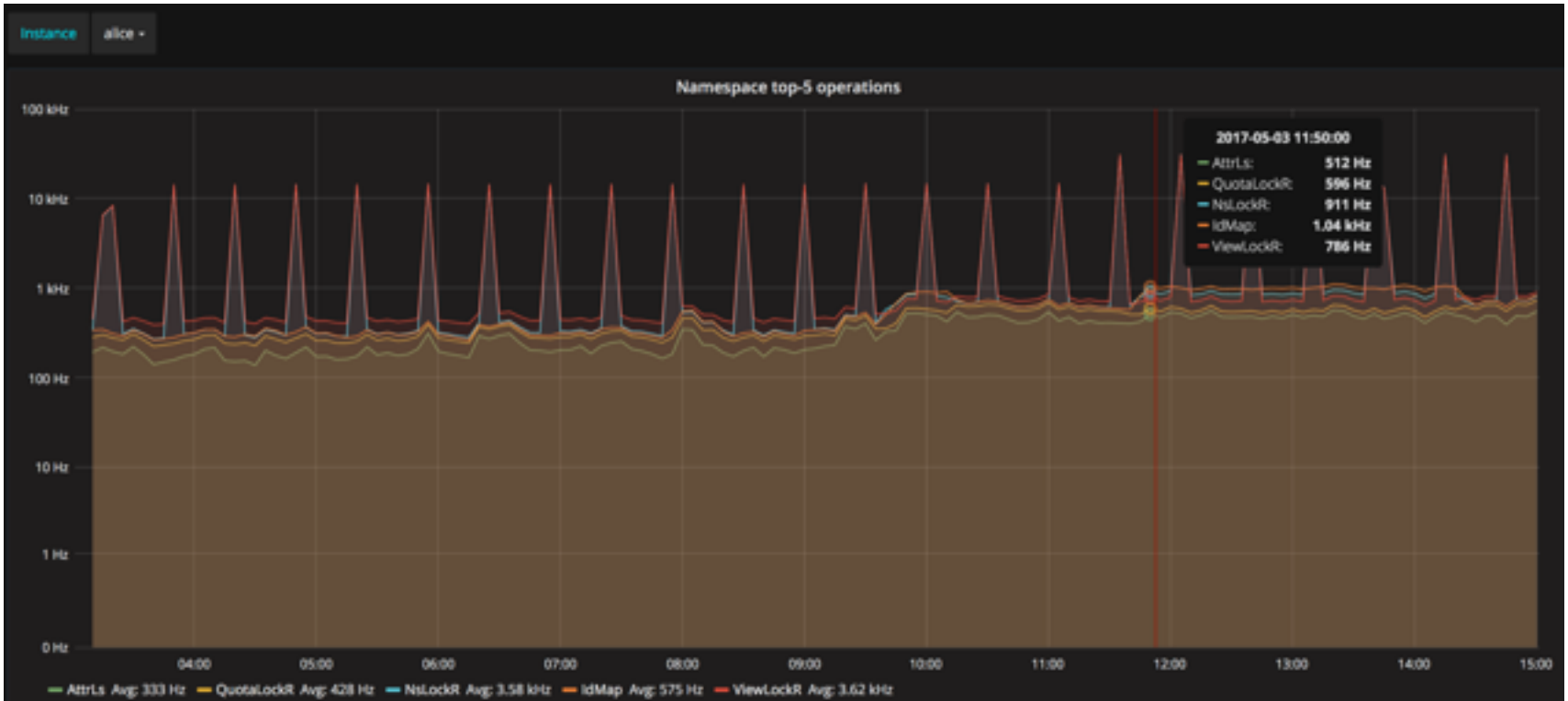
Additional disk server commissioning now
48 x 6 = **288 TB** server **5 GB/s** disk IO
100 PB for ALL@CERN (divide by 2)

EOSALICE

performance

Open Source Storage

~500 OpenRead/s (yearly average)
CERN hosts 1/3 of all ALICE GRID files - peak 50 GB/s read
ns limit: max. 1 MHz stat/s internal
ns limit: max. 65 kHz stat/s via XRootD protocol
ns limit: 0.8B files with Aquamarine version



Developments

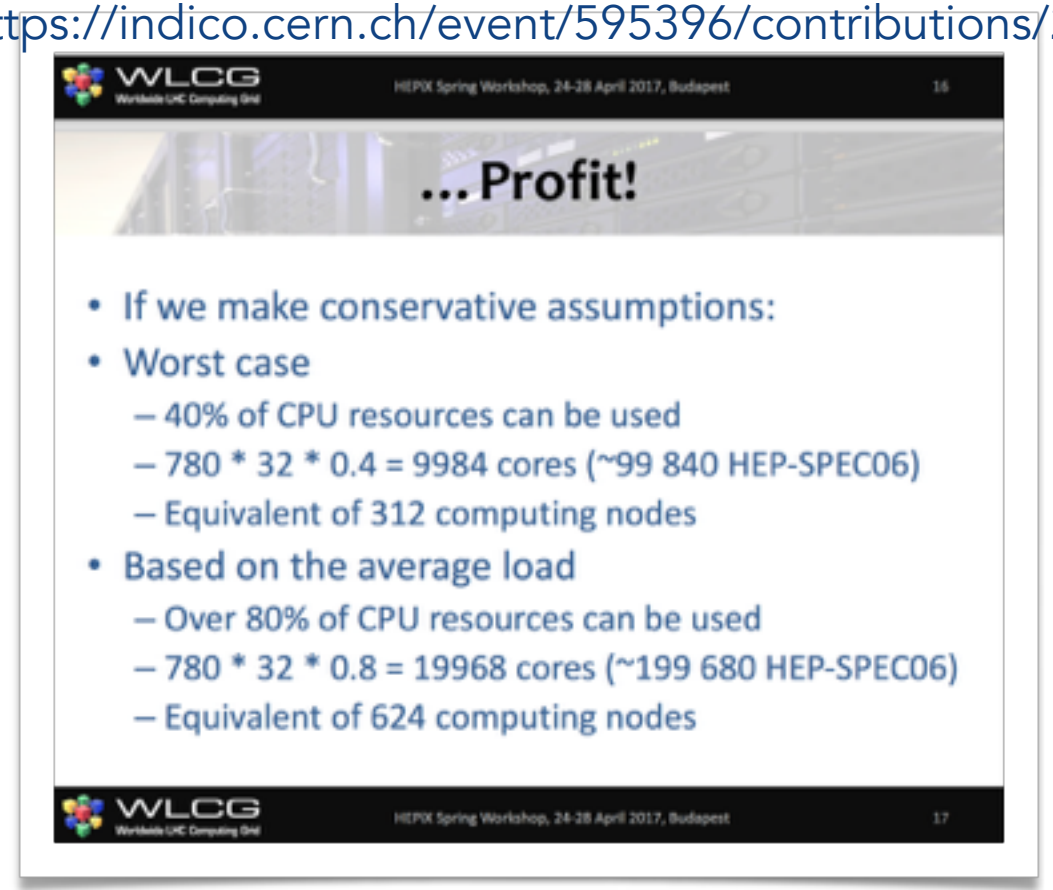
Development News

- boosted namespace load time 2-6x
- fixed master-slave failover & compaction issues
- CITRINE production ready
- CI - continuous integration platform on gitlab
  - build pipelines
    - **RPM** builds on SLC6, EL7, Fedora, OS X (client)
    - **DOCKER** image build
    - **automated testing** on every commit
  - coming: **kubernetes** cluster setups with **long-term testing**

details can be found on the EOS workshop page
https://indico.cern.ch/event/591485/

R&D

- hybrid computing/storage infrastructure
  - run batch jobs on EOS disk server
- interesting presentation at HEPIX
  https://indico.cern.ch/event/595396/contributions/2532584/



R&D

- CERN-IT extra-large disk server project
  - **8 x 24 x 6TB** disks connected to single front-end node [1.152 PB/node]
    - capacity/performance ratio ?
    - OS limitations handling 192 disks ?
    - RAID vs. ZRAID vs. Software EC
    - which network IF ?
    - which CPU type ?
    - TCO evaluation

EOS Architecture Aquamarine

Beryl Aquamarine

V 0.3.X

all instances today!

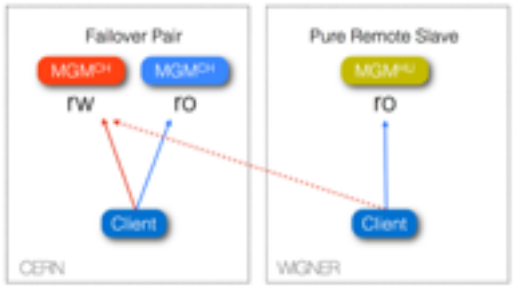
read/ write

MGM Master

MGM Slave

read only

META DATA

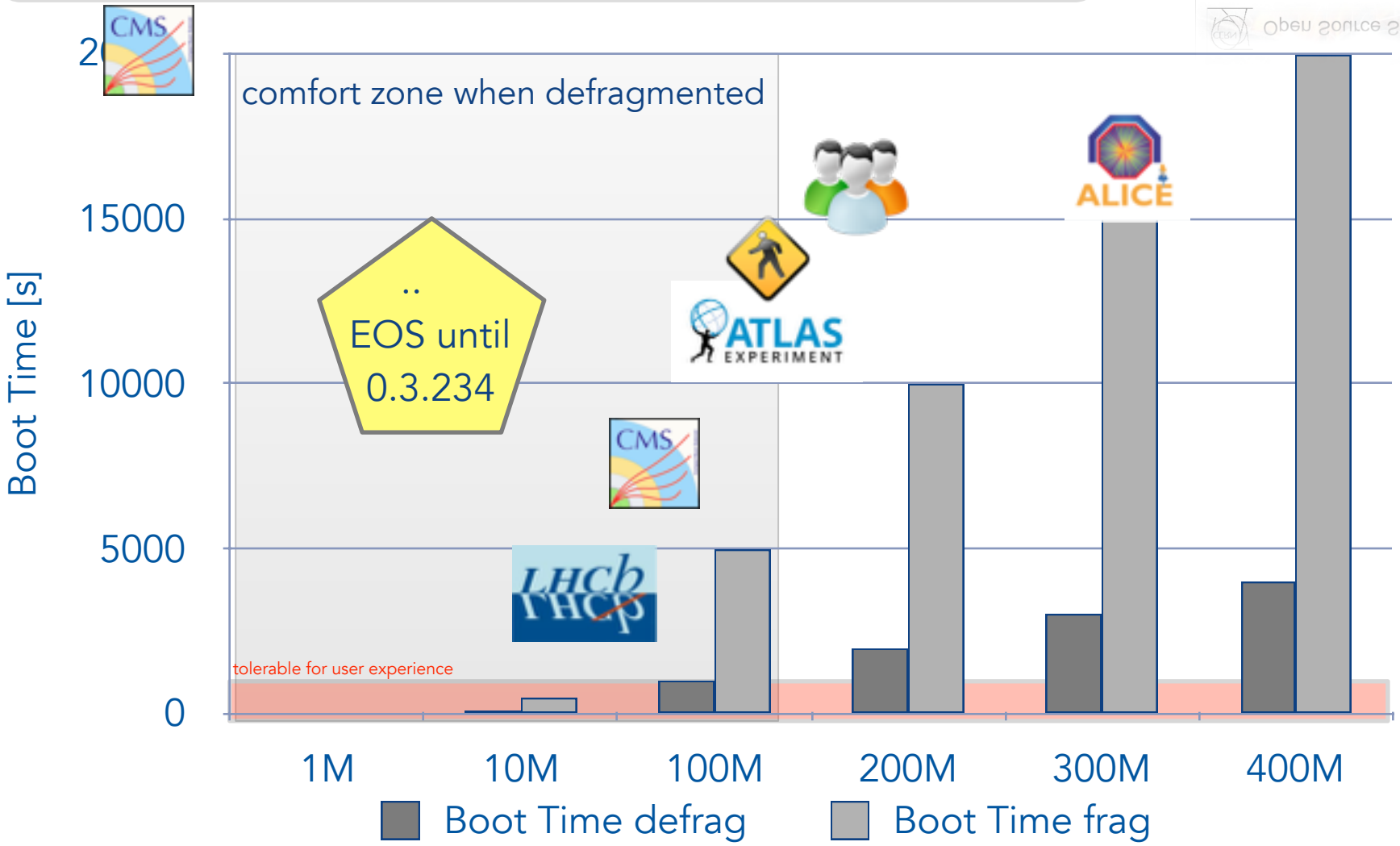


FST FST FST FST

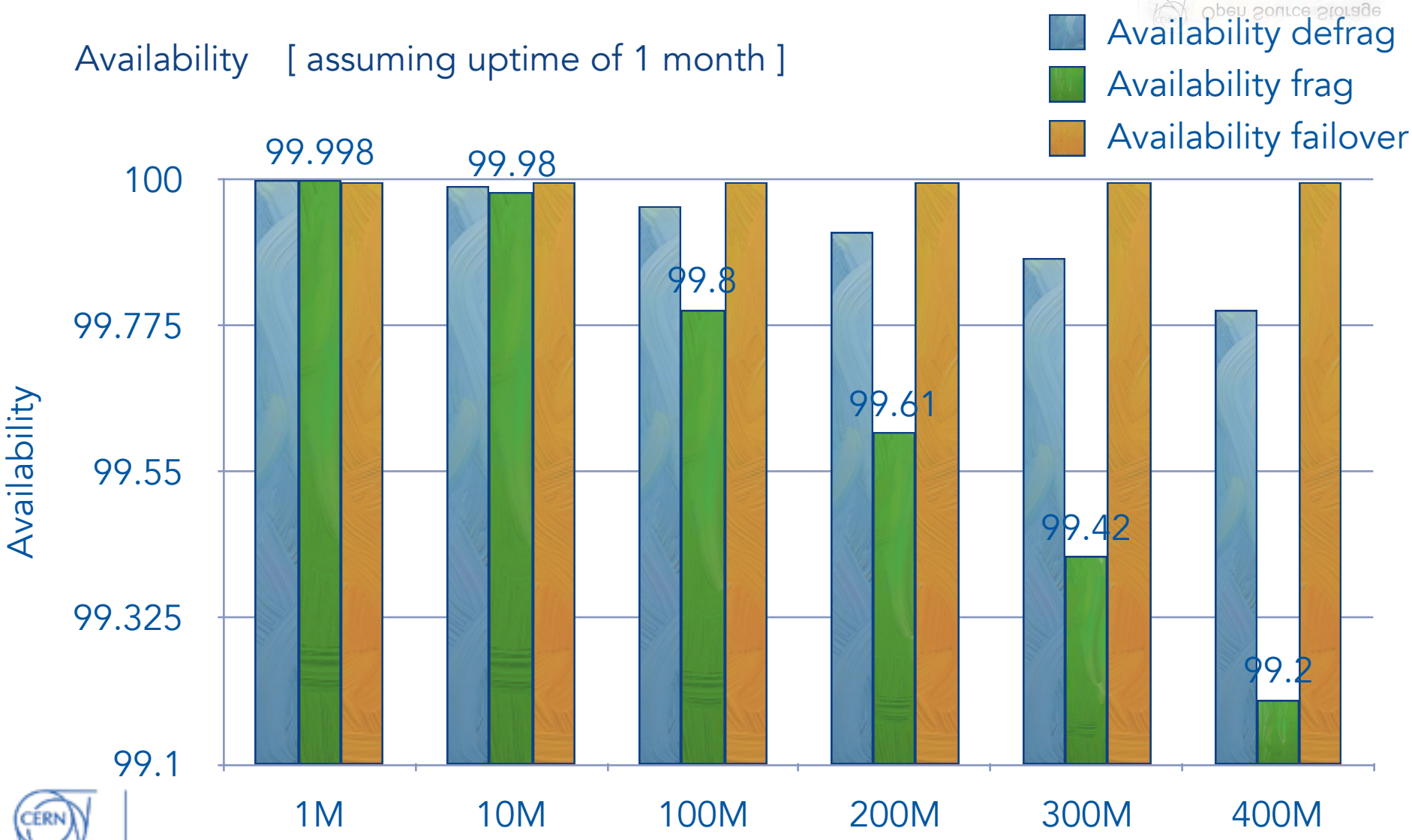
latest release **0.3.244**

major fixes in master/slave failover during last 3 month

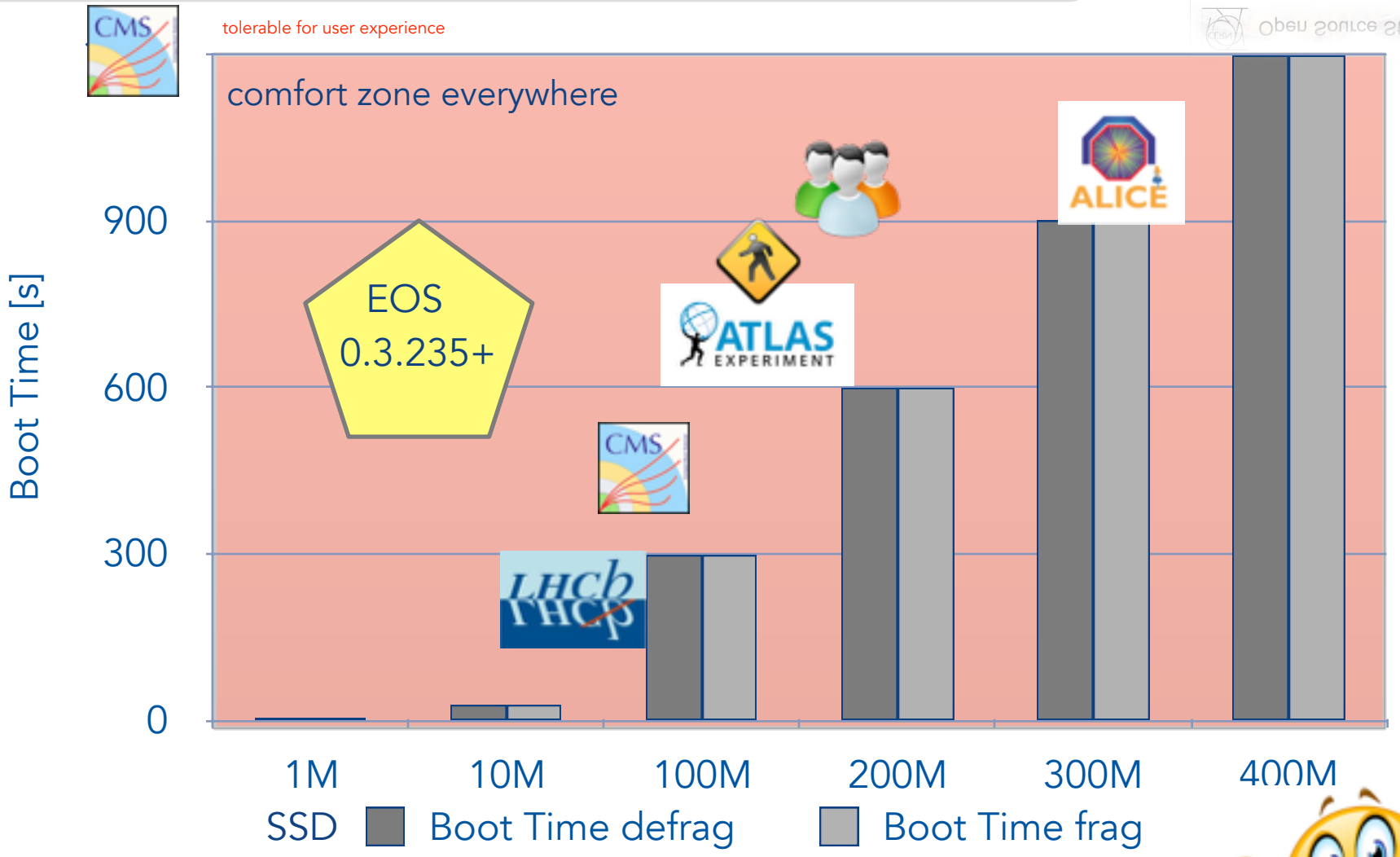
In-memory namespace



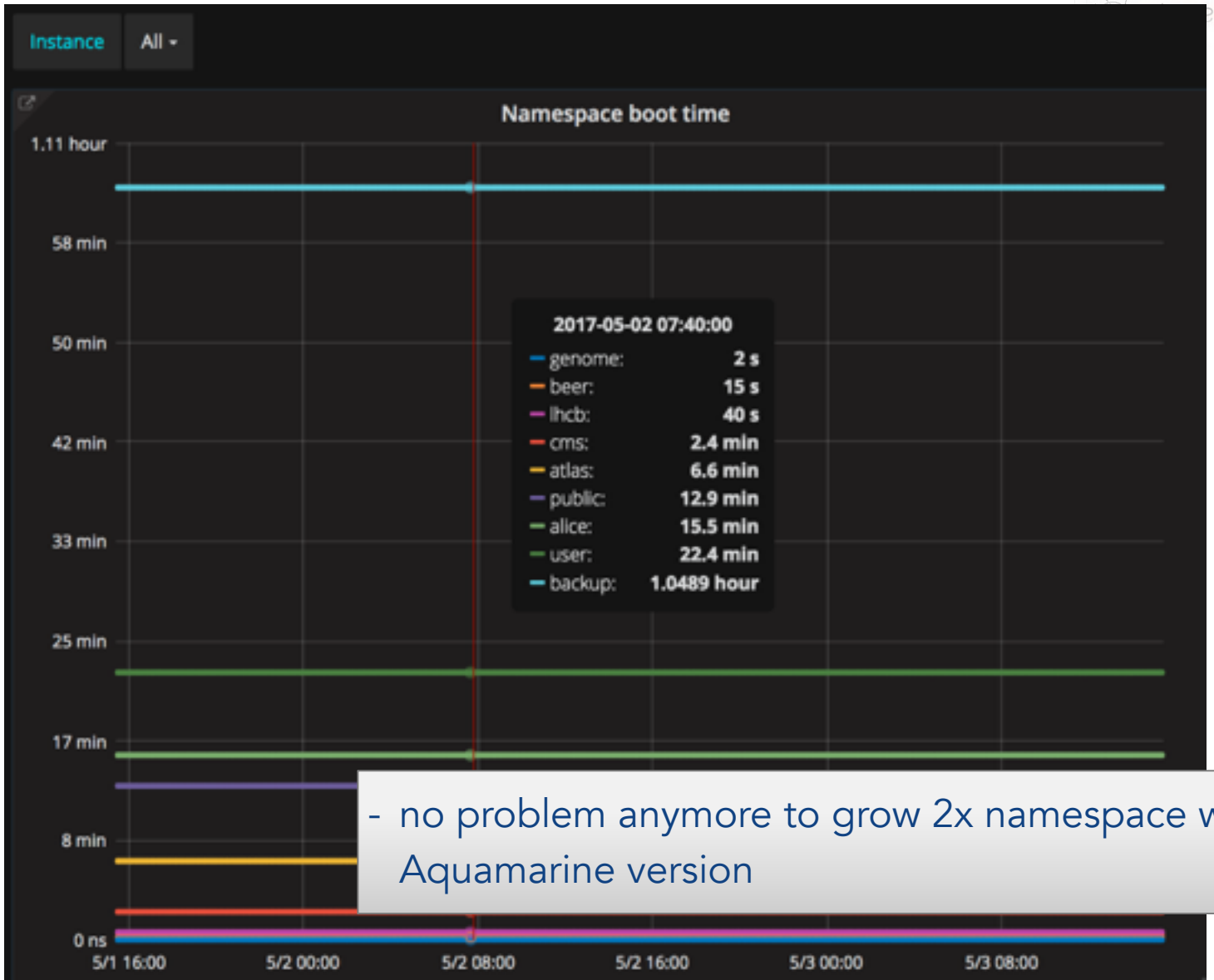
In-memory namespace



New Parallel NS Boot



New Parallel NS Boot



- no problem anymore to grow 2x namespace with Aquamarine version

EOS Architectural Evolution

Beryl Aquamarine

V 0.3.X

Citrine

V 4.X

read/ write

MGM Master

MGM Slave

read only

META DATA

stateless

MGM

MGM

MGM

Persistency

DATA

FST

FST

FST

FST

FST

FST

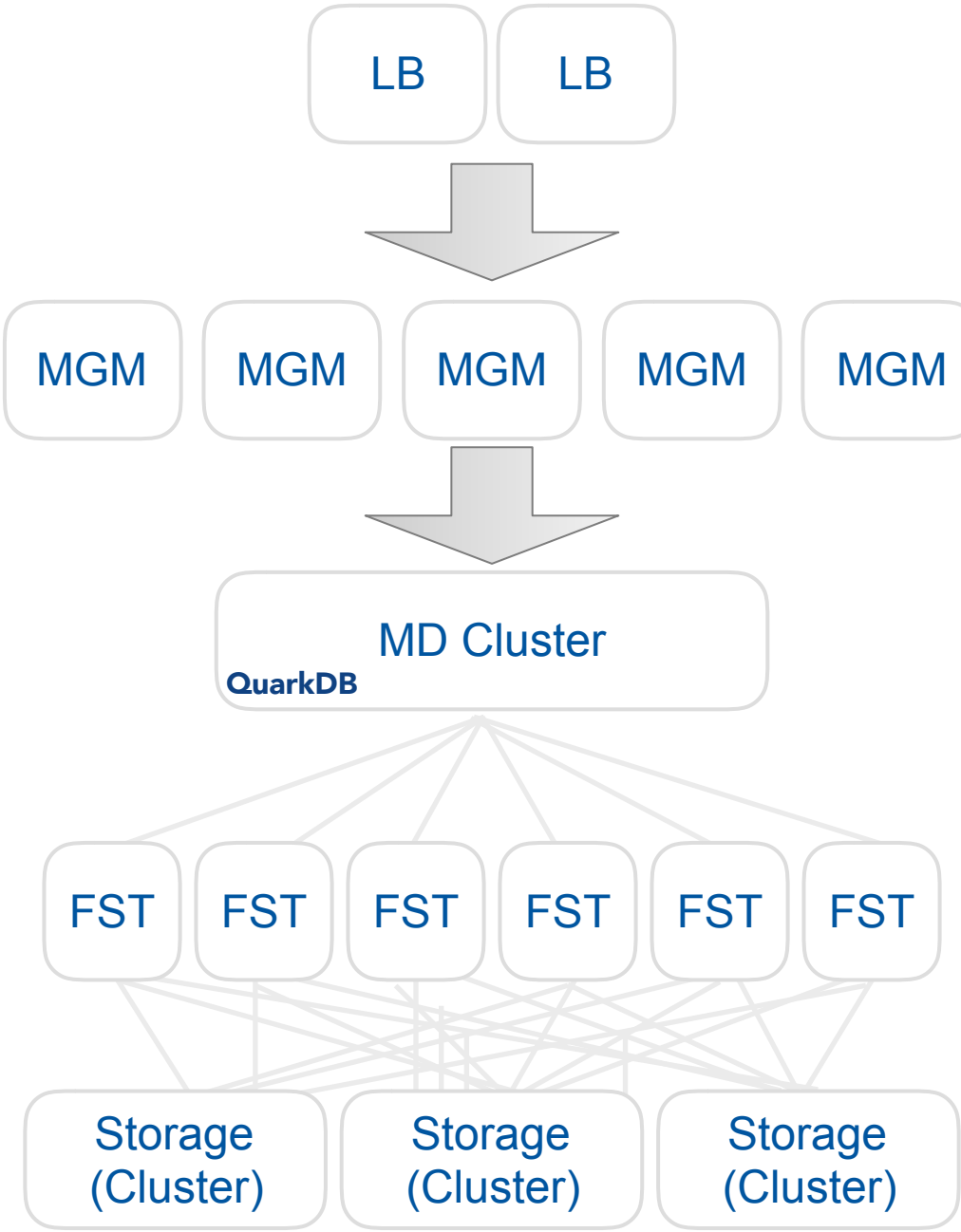
FST

FST

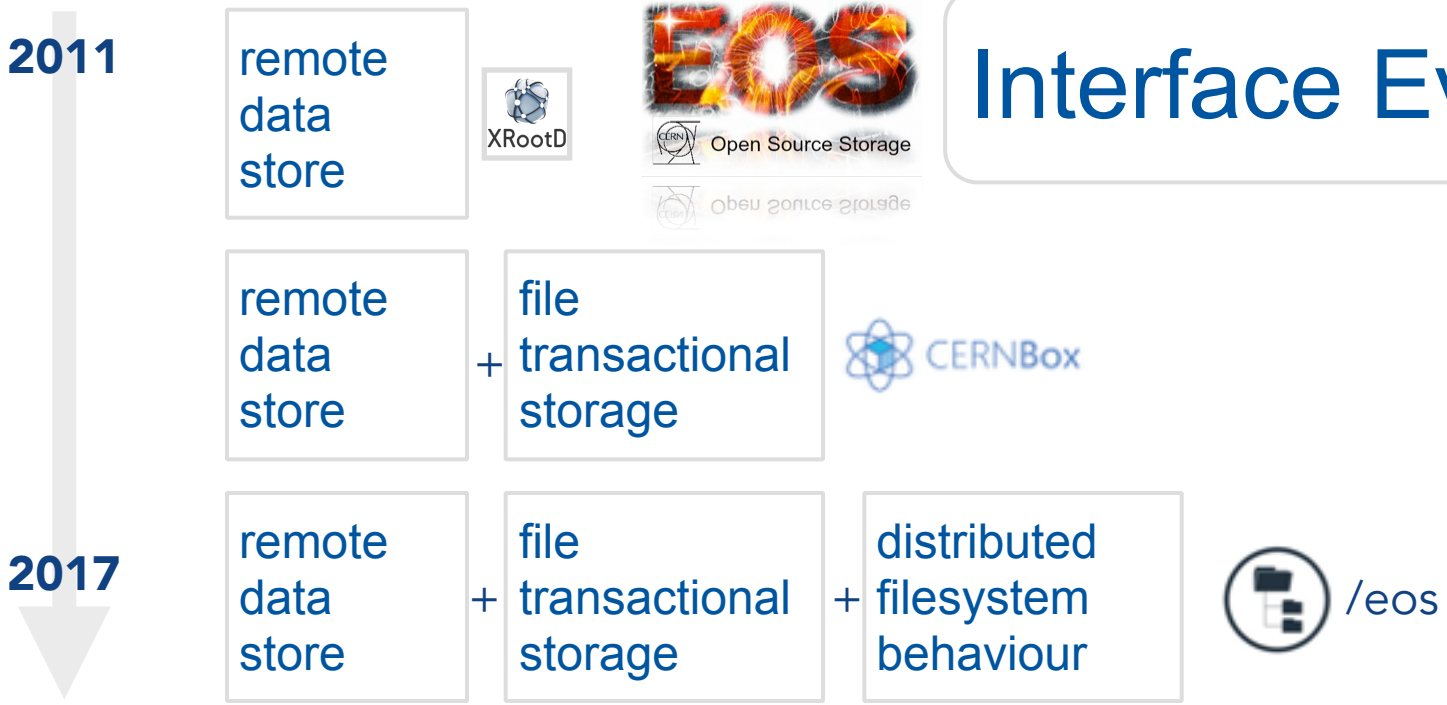
KINETIC Open Storage Project

ceph

amazon.com


LB
LB
MGM
MGM
MGM
MGM
MGM
MD Cluster
QuarkDB
FST
FST
FST
FST
FST
FST
Storage (Cluster)
Storage (Cluster)
Storage (Cluster)

Interface Evolution



Evolution

EOS has started 6 years ago as a remotely accessible data storage system with *posix-similar* **interface.** The interfaces has been extended to provide **file transaction functionality**. The most recent architectural change is to provide mounted filesystem semantics.

EOS FUSE Current Status

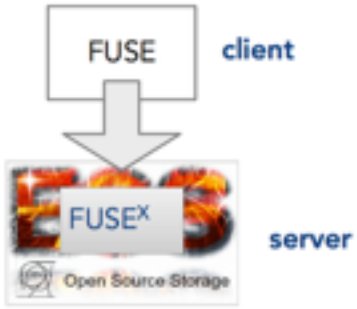
/eos mounted on lxplus and lxbatch

- significant amount of problems and obstacles
  - consistency, stability and kerberos integration
- experience triggered clean rewrite of FUSE client
- will be available this month
- useless for ALICE grid

V2 implementation

FUSE filesystem implemented as **pure client side** application without dedicated server side support.

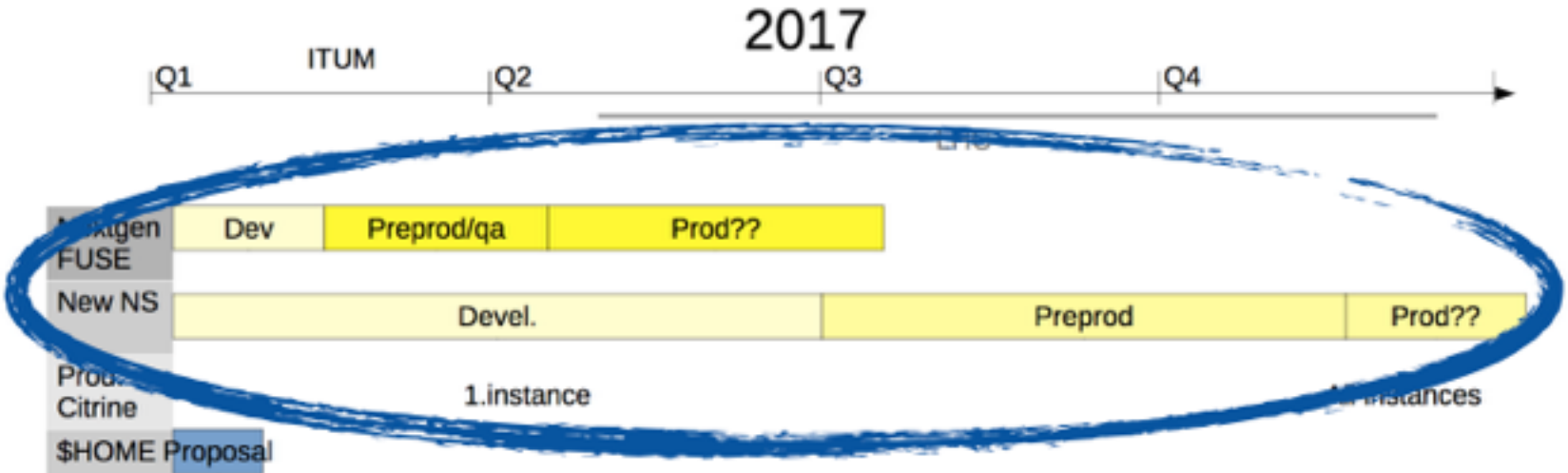
V3 implementation



Dedicated server-side support providing a fully asynchronous server->client communication, leases, locks, file inlining, local meta-data and data caching

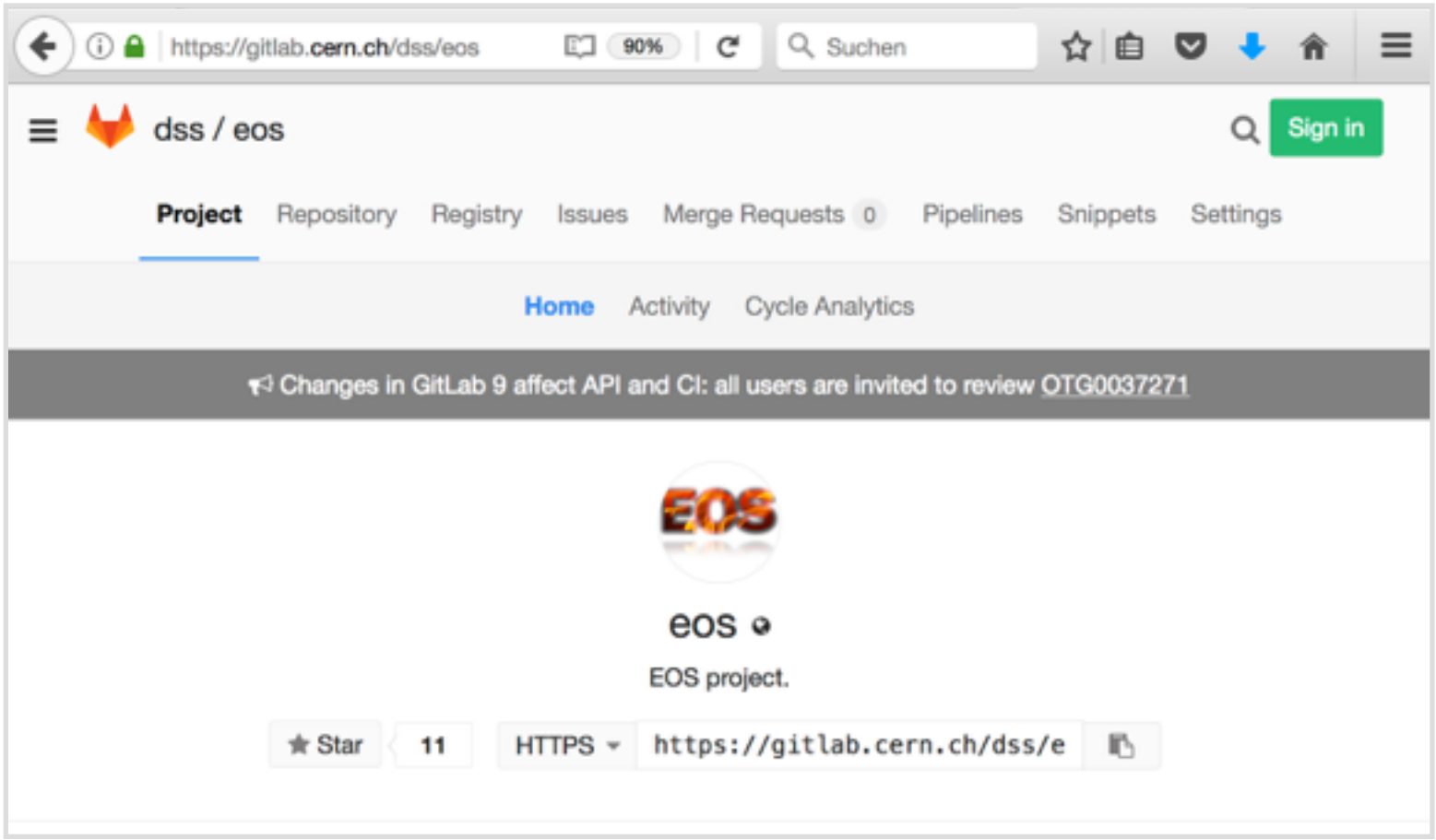
Roadmap & Milestones

- new FUSE client Q2
- first CITRINE instance at CERN **IPV6 /xroot4**
  - 15th of May update for LHCB still with in-memory namespace
  - based on experience migration of all instances to CITRINE after negotiation
  - Q3-Q4 planning move from in-memory to QuarkDB namespace **namespace scalability - demonstrate multi billion ns**



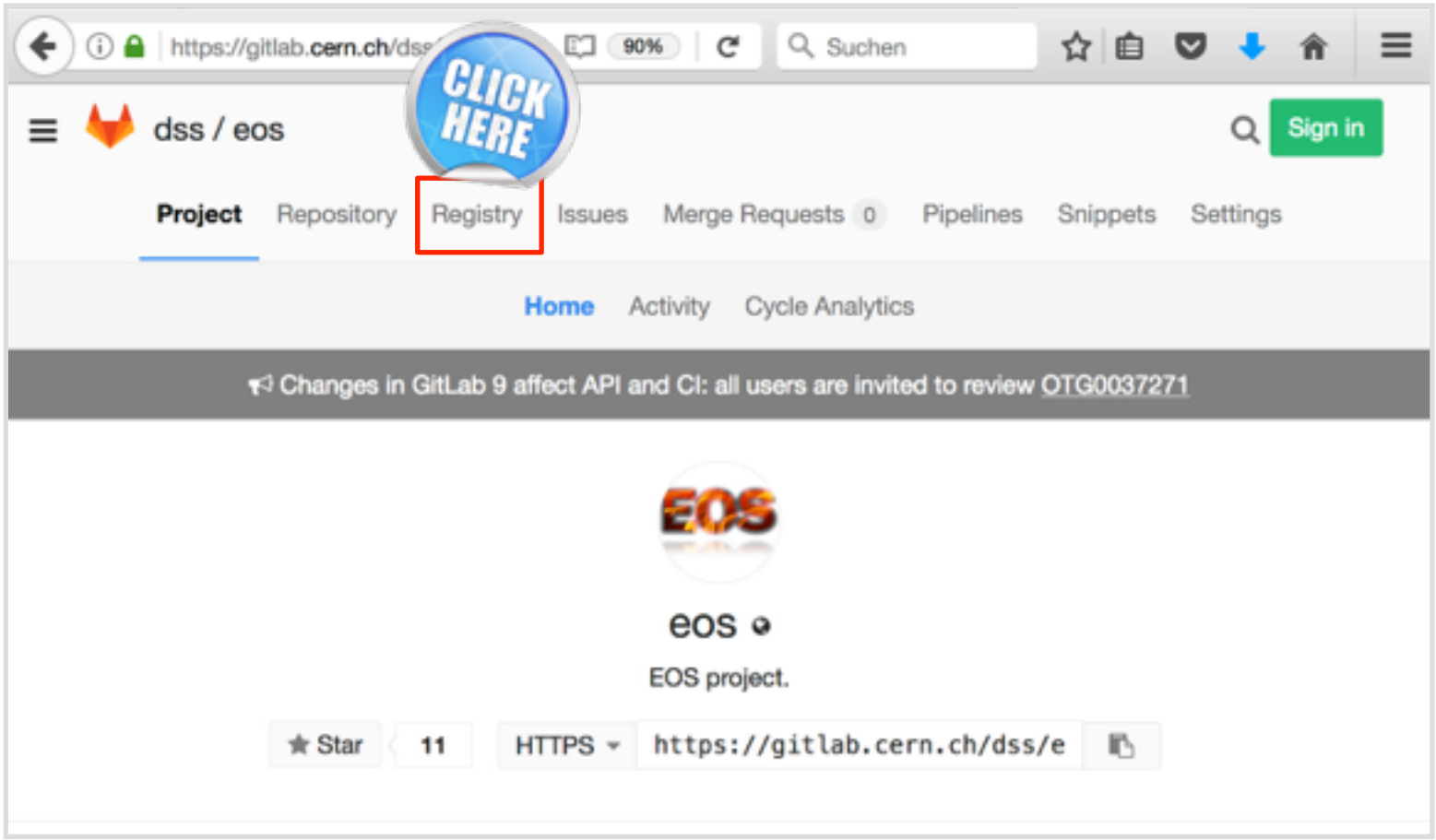
Continuos Integration

https://gitlab.cern.ch/dss/eos

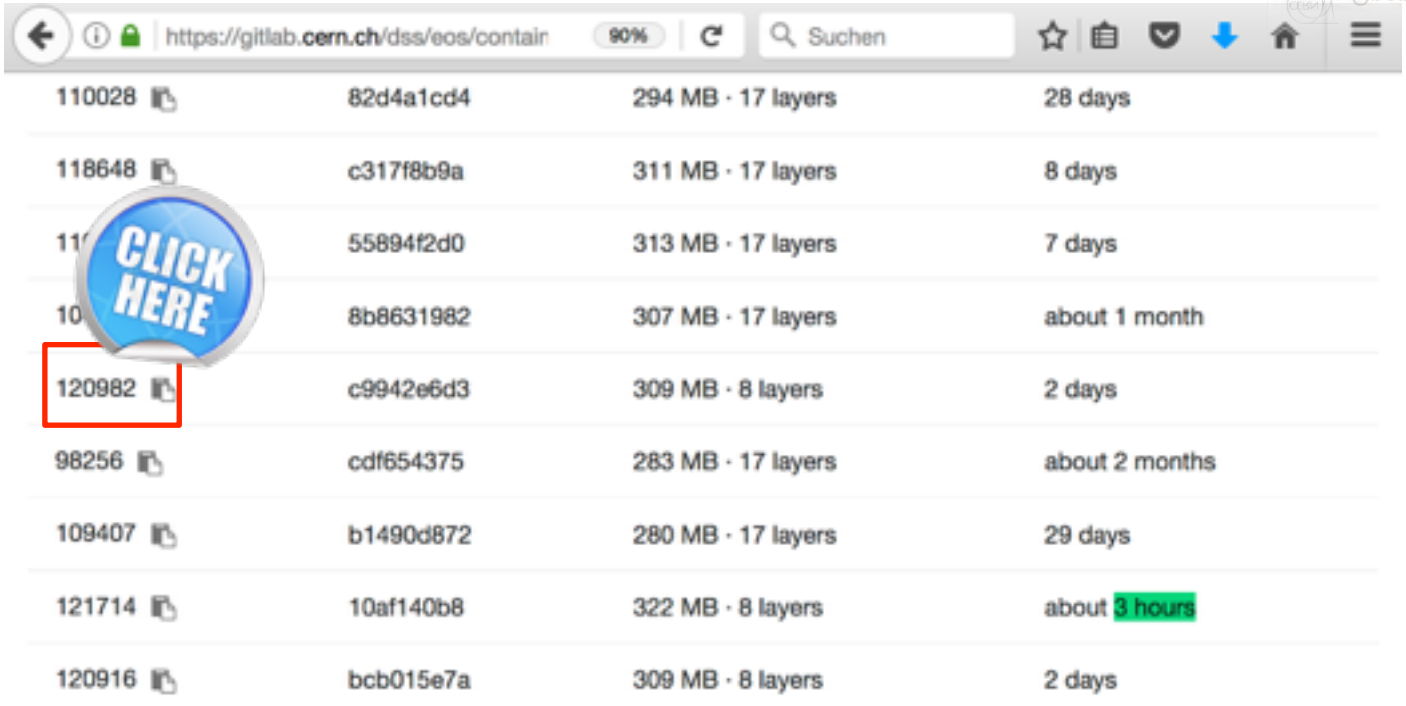


Continuos Integration

https://gitlab.cern.ch/dss/eos



Continuos Integration



get a CITRINE image e.g.

docker pull gitlab-registry.cern.ch/dss/eos:121629

```
[root@eos-docker ~]# docker images
REPOSITORY                          TAG                 IMAGE ID            CREATED
        SIZE
gitlab-registry.cern.ch/dss/eos     121629              0b153e8b6506        5 hours ago
        759.4 MB
```

EOS in DOCKER - 1 minute

git clone https://gitlab.cern.ch/eos/eos-docker.git

start a virtual instance with KDC, MGM, MQ, 6xFST

./eos-docker/scripts/start_services.sh

get a shell in the MGM service container

docker exec -it eos-mgm-test bash

run instance tests

eos-instance-test

```
[root@eos-docker tmp]# eos-docker/scripts/start_services.sh  -i 0b153e8b6506
468d806d6f2d8ad93980068l8d0df281ac73621ece9b9c739c36596c2a05fb17
700257d7437b3a0b72f0ecdccfae95557932f5b73871ad27b1ee5ccd7dede03a
Starting kdc... Done.
Initing kdc... Done.
Populating kdc... added admin1@TEST.EOS with password "tDp5mfkjBx"
added host/eos-mgm-test.eoscluster.cern.ch@TEST.EOS with password ">CzjpraSdm"
Done.
378267ed609c0806a73bc9a4163aff27573ed089cf29d84eb81043d5876c635d
2ecea01ccabb84442a024ef458dd0117519cece3b77b62085fa1a2a71b65bd42
success: set vid [  eos.rgid=0 eos.ruid=0 mgm.cmd=vid mgm.subcmd=set mgm.vid.auth
m.vid.cmd=map mgm.vid.gid=0 mgm.vid.key=<key> mgm.vid.pattern=<pwd> mgm.vid.uid=0
2d1ba4c7111c933d89d6ad282643cb4287b11c47f6a7a94149cbba1763fae651
02398a7e926966300016004397e59d419c26a13c256f7a64e13969ee222f7512
4e906031e8eb8b61bea2692aa8aec41c9921136676daed4eccd7aa6ebef51f21
7f42121461ed5c28961e87689fdcc0671f72d19a7936473387a3dc112e4e3607
75e76e25e2ac435cffde5476779162775cbf705720b10f274dc7e378e39a8446
152bffdaafdac36af00a69b3dd3b7e68eacbf9b54f479a15f3e09c6d7dcbb915
Starting fst1 ...
Starting fst2 ...
Starting fst3 ...
Starting fst4 ...
```

EOS in DOCKER - 1 minute

- currently the docker scripts are only to get a **one machine instance** for testing

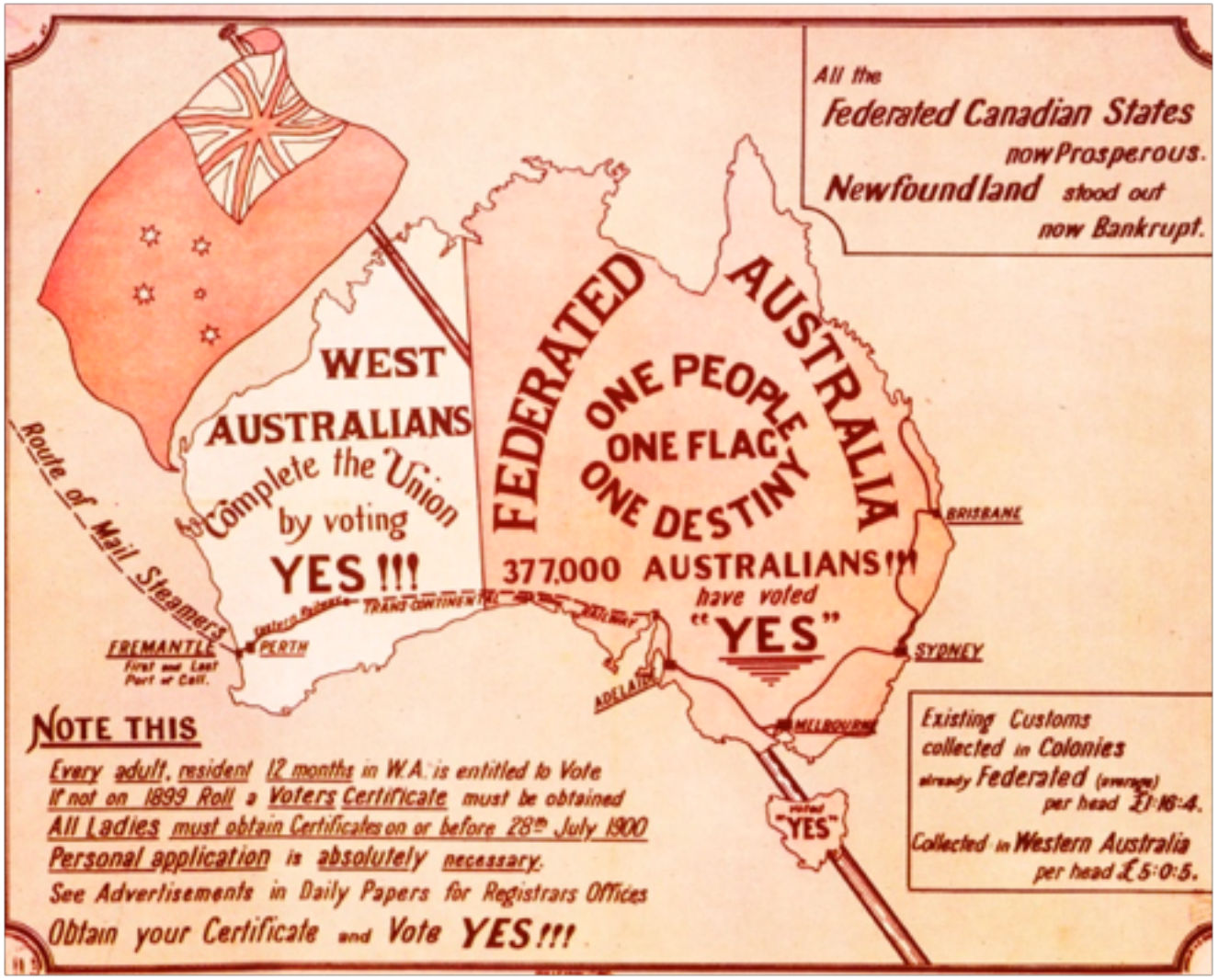
- the CERNBOX team is finishing a complete **dockerized CERNBOX**-like service package bundling EOS + OwnCloud

- prototyped a single host **ALICE docker** storage container with a pre-configured EOSALICE instance using the physical network inside the container

- interesting option to combine with **kubernetes** to simplify deployment in a storage federation …. integration is on the work plan …

- if there is a broader interest, we can integrate the work of AARNET which is deploying EOS only via docker containers and add ALICE specifics

Federations



Storage Federations

- driving idea is to
  - **reduce** the number of storage services to manage
  - **bundle** many small resources into a bigger single resource
  - **reduce** operation effort
    - few complex services (namespaces)
    - many trivial services (*object* storage servers)

EOS Storage Federation

3 types of sites

storage site type 1

FST FST

storage server

MGM Master

MGM Slave

active/passive meta data server

MQ

MGM Follower

FST FST FST

storage server

FST FST

storage server

storage site type 2

storage site type 3

EOS Placement Strategies

- CITRINE release provides **geographic-aware scheduling**
  - placement and access policies configurable for each directory
  - client and servers are geo-tagged
    (client:subnet server:configuration value)
    - access files as ‘close’ as possible
    - placement policies e.g.
      - hard-coded replication to defined locations
      - two replicas close
      - one close, one randomly
      - two replicas at maximum distance
      - …

EOS Federations

Examples

EOSALICE: 159 storage nodes at CERN & 85 at WIGNER

placement policy: if possible maximum distance e.g. one replica at CERN, one at Wigner

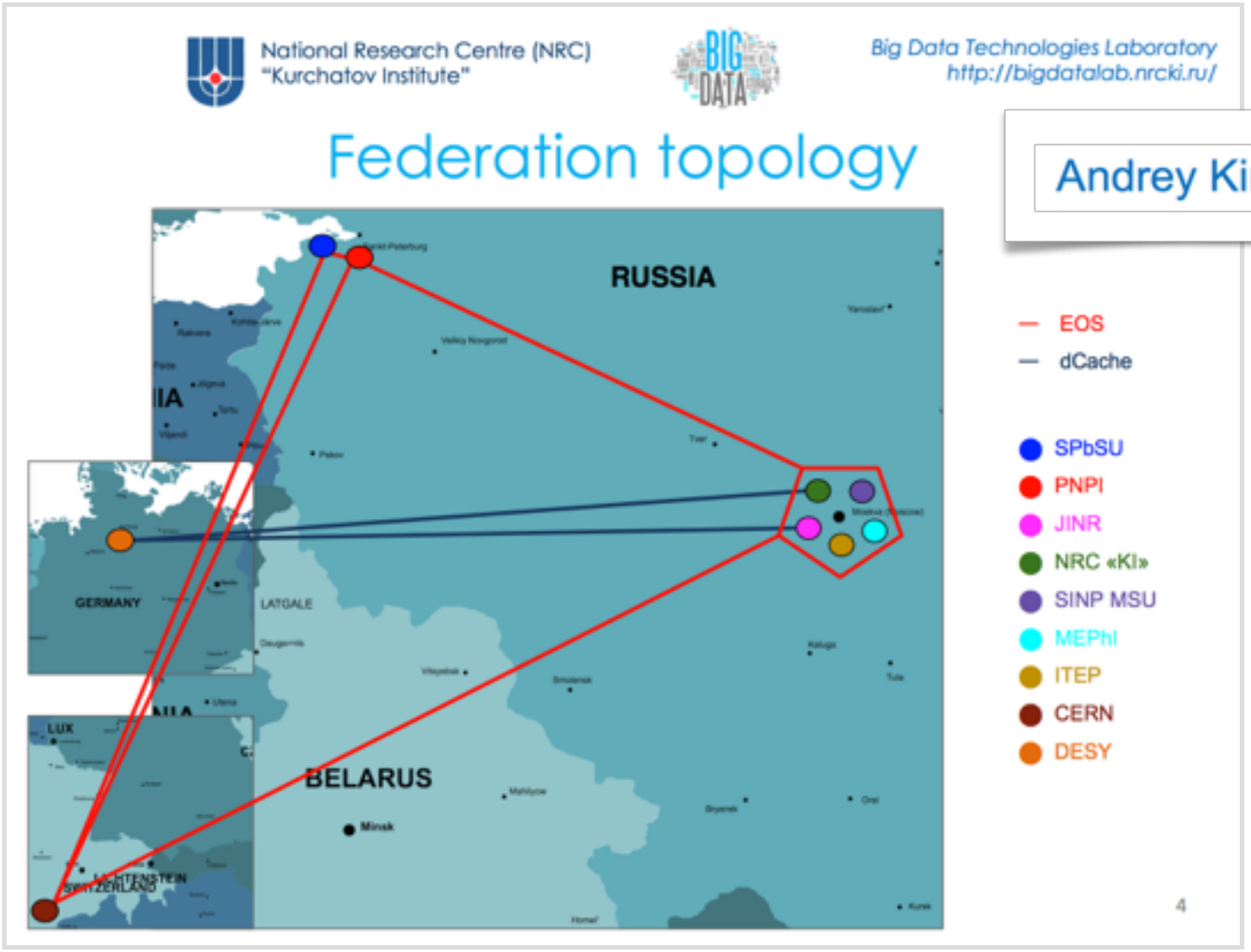
Wigner Computer Centre


```
[root@lxbse15c06 ~]# ping p05153065491511
PING p05153065491511.cern.ch (188.185.224.50) 56(84) bytes of data.
64 bytes from p05153065491511.cern.ch (188.185.224.50): icmp_seq=1 ttl=58 time=22.0 ms
64 bytes from p05153065491511.cern.ch (188.185.224.50): icmp_seq=2 ttl=58 time=22.1 ms
64 bytes from p05153065491511.cern.ch (188.185.224.50): icmp_seq=3 ttl=58 time=22.1 ms
64 bytes from p05153065491511.cern.ch (188.185.224.50): icmp_seq=4 ttl=58 time=22.1 ms
```

22ms rtt

EOS Federations

Examples



https://indico.cern.ch/event/595396/contributions/2532587/

EOS Federations

Examples

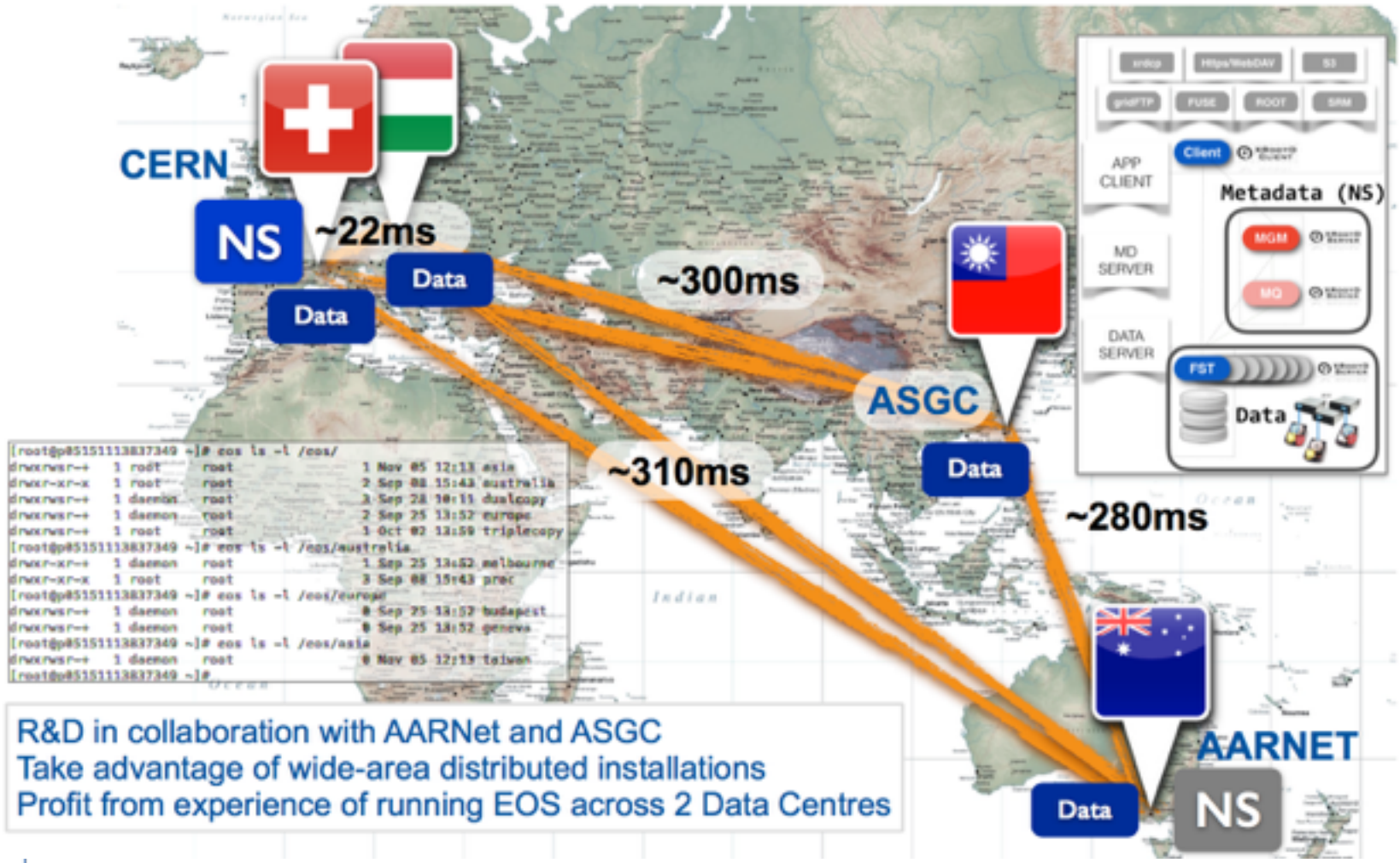
AARNET Federation in Australia with 65ms latency

- three storage locations
- one RW/RO namespace pair
- one remote RO namespace

EOS Federations

Examples

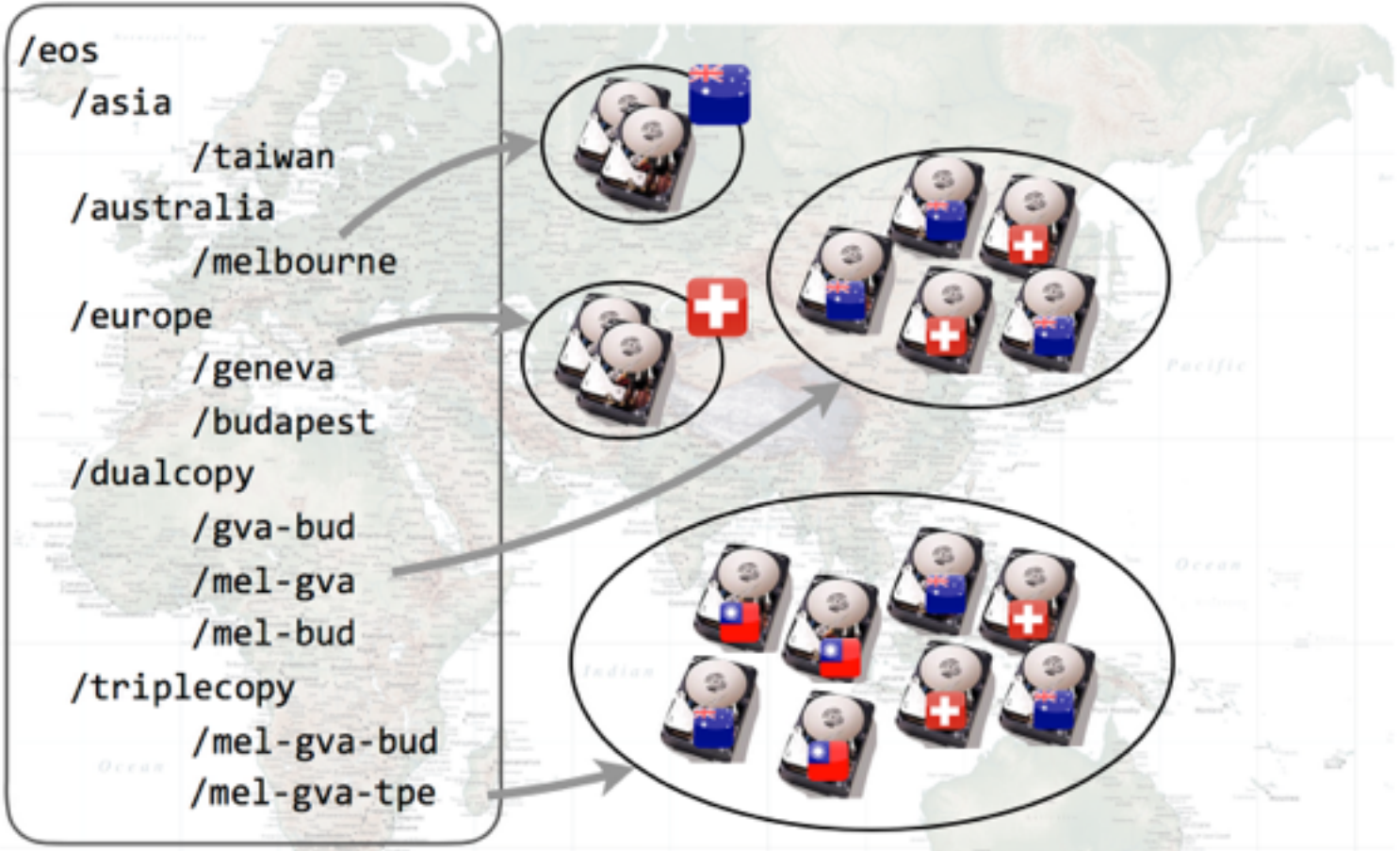
Worldwide distributed storage system with extreme latencies - R&D prototype



EOS Federations

Examples

Worldwide distributed storage system with extreme latencies - R&D prototype



Storage pools were created with filesystems from all four sites.
Files were replicated according to the different configured policy
(e.g. 3 replicas: MEL-GVA-TPE).

EOS Federations benchmarks

Andrey Kiryanov

presented at EOS workshop
presented at HEPIX sprint 2017

EOS Federations

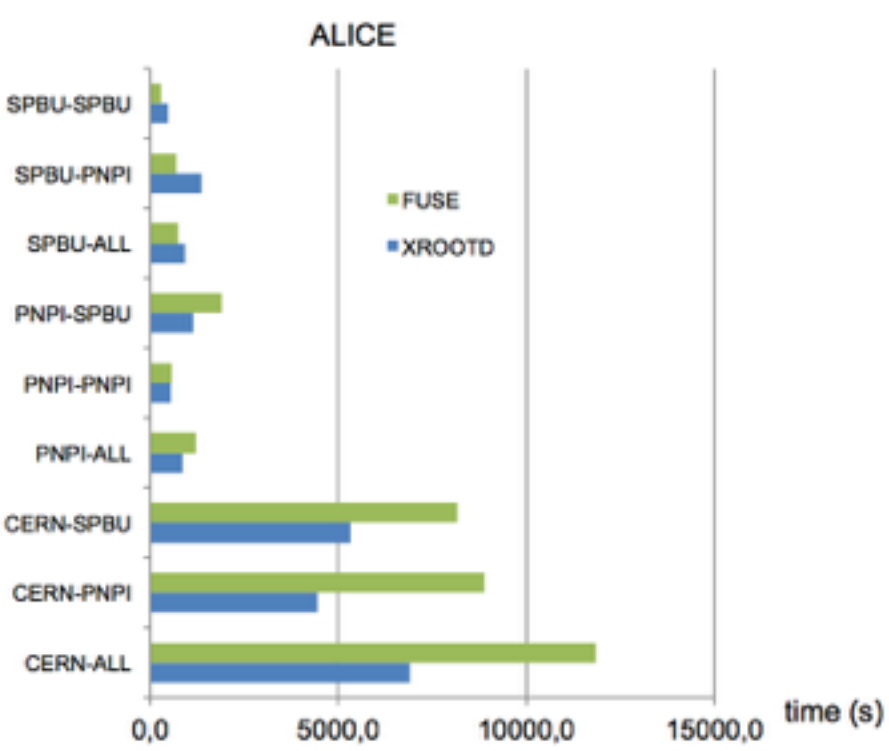
Examples

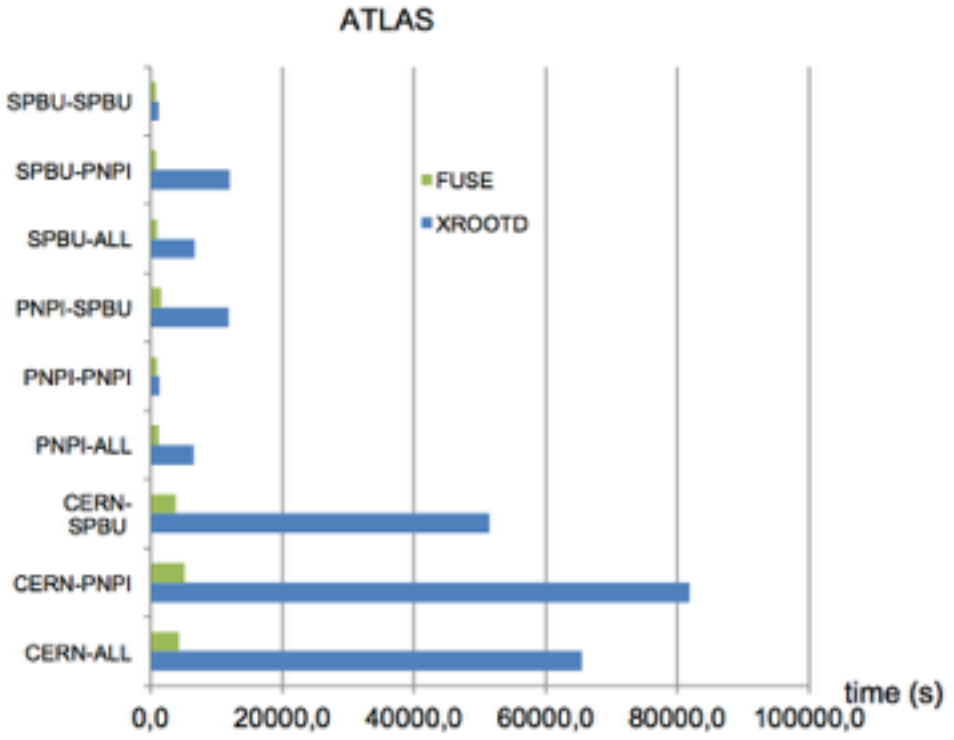
Andrey Kiryanov

National Research Centre (NRC)
"Kurchatov Institute"

Big Data Technologies Laboratory
http://bigdatalab.nrcki.ru/

Experiment-specific tests with EOS for two protocols: pure xrootd and locally-mounted file system (FUSE)



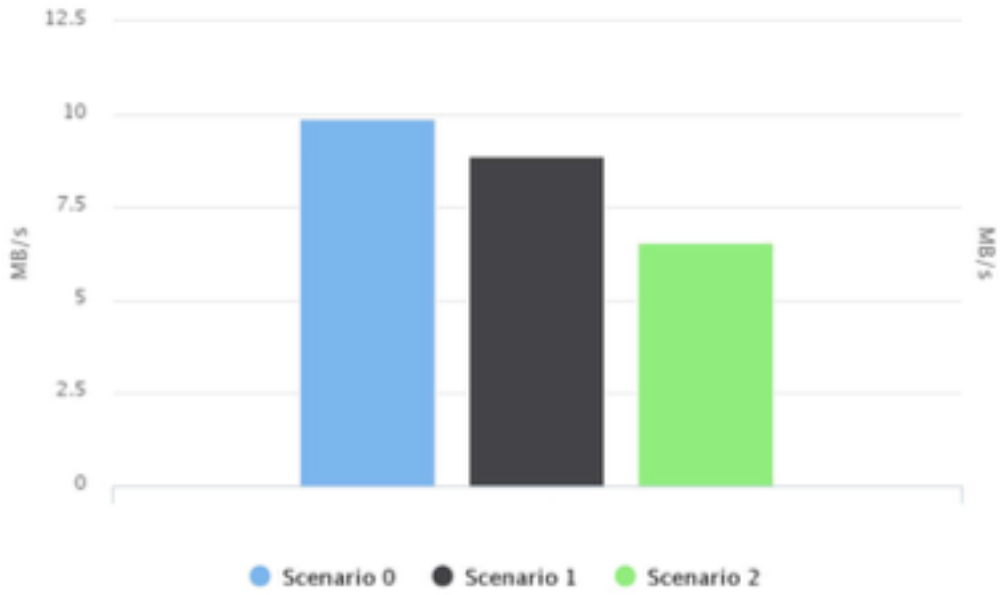


EOS Federations

Scenario 0: Dataset is randomly distributed among several sites;

Scenario 1: Dataset is located as close as possible to the client. If there's no close storage, the default reliable one is used (NRC "KI" in our tests);

Scenario 2: Dataset is located as in scenario 1 with secondary copies as in scenario 0.

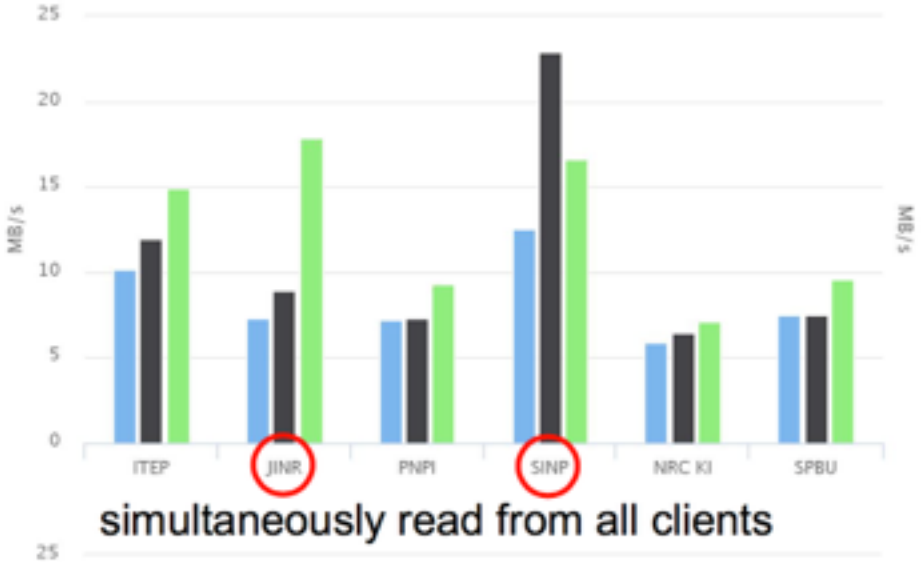


Data Upload Speed Comparison

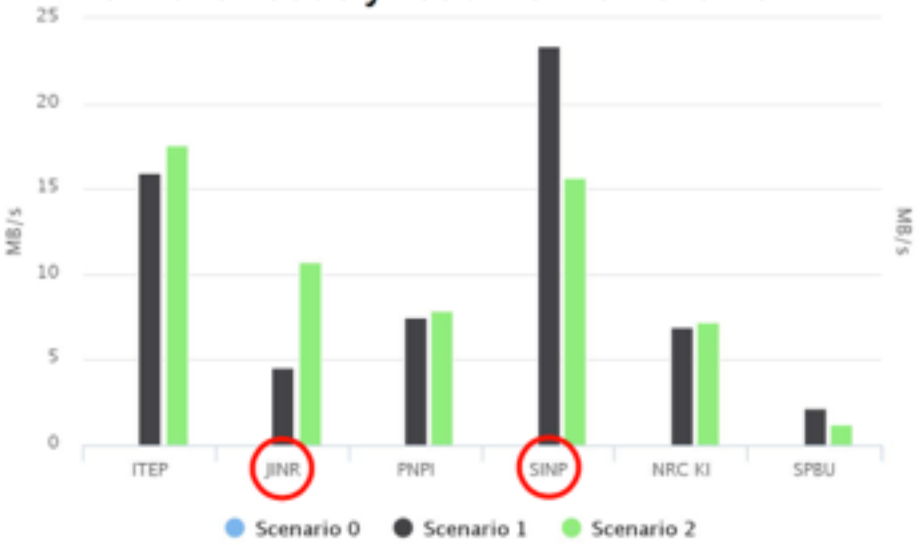
Andrey Kiryanov

EOS Federations

National Research Centre (NRC) "Kurchatov Institute"
Big Data Technologies Laboratory http://bigdatalab.nrcki.ru/
Andrey Kiryanov



simultaneously read from all clients



only one client reads data at any given time

ALICE read test

Read procedure is as follows:

Scenario 0: Files are scattered among several file servers

Scenario 1: All files are on the default file server at NRC "KI"

Scenario 2: All files are on the default file server at NRC "KI" with replicas that may end up on a closest file server

◯ – this client can find data on a closest file server

On both plots clients are shown on X axis

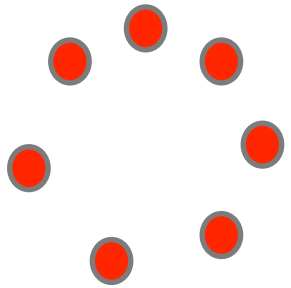
Impact of a system load is negligible at this scale.
Logistics optimization only makes sense for sites with a proper infrastructure.

Storage Federations

when to use them

- **doable** if all storage servers have sufficient bandwidth between each other and batch resources in the federation
  - the WIGNER experience
    - two site federated setup complicated operations
      - induces 100% storage overhead [2 replica]
        - cannot use erasure encoding
      - network links become saturated
      - cannot guarantee 50:50 resources CERN:Wigner
      - TCP window scaling, a lot of network tuning and network problems to debug
      - job efficiency worse for 22ms latency than in LAN (depends on application)

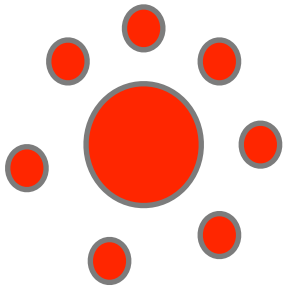
Storage Federations
when to use them



- **local cloud federation model** (regional federation)
  - storage peers are close (O(2) ms) and have sufficient network bandwidth
    - model requires efficient remote data access because:
      - impossible/inefficient to have a replica at each geo-graphic location
      - remote access will occur frequently

Storage Federations

when to use them



- **satellite federation model** (T1/T2 federation)
  - a large resource is federated with many significantly smaller resources
    - main storage volume and CPU provided by single resource
    - T1 provides namespace and storage
    - T2 attach to T1 namespace

Storage Federations

when not to use them

- federations of several big resources are orthogonal to the GRID model
  - job scheduling should be location aware
  - a federation abstracts locality and network becomes a bottle neck between large resources

- storage federation have to be complemente and integrated with CPU resource federations - no free lunch - sounds nice to have only few big virtual sites but ..

my personal conclusion:

- the ALICE grid model is the most space efficient federation (global policy for replication) - not operation wise
- regional federations of small contributors to storage and CPU make absolutely sense and help to reduce operational effort
- large resource federation have too high impact on volume and job efficiencies

EOS need-to-know summary

- running sites should **update to 0.3.244** if they rely on namespace failover
- sites interested in IPV6 can move to CITRINE release
  (first CERN deployment mid-may)
- many non-GRID-required features make EOS attractive as a site service (CERNBOX, multi-VO etc.) as GRID and non-GRID storage
- **support** - you are invited to get support in using EOS
  eos-support@cern.ch
- **documentation**
  - http://eos.readthedocs.io [en/citrine]
- **repositories**
  - https://gitlab.cern.ch/dss/eos.git https://github.com/cern-eos/eos
- **mailinglist**
  - eos-community@cern.ch [via egroups.cern.ch]

www.cern.ch

Thank You!