EOS Open Storage

evolution of an ecosystem for scientific data repositories

http://eos.cern.ch

Andreas-Joachim Peters
CERN IT-ST

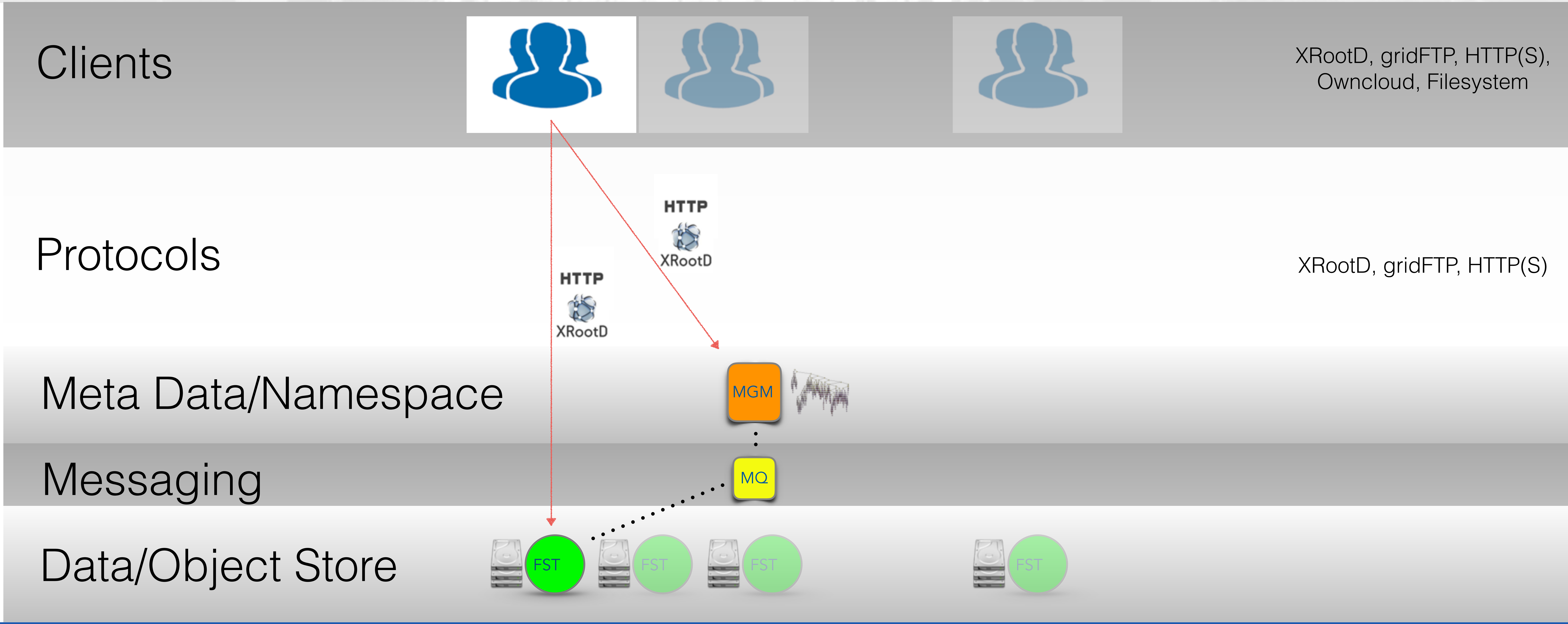
Overview

- **Introduction**
  - what is EOS?
  - history
  - architectural evolution
- EOS **service** at CERN
  - dimension & challenges
- EOS in a science **ecosystem**
  - EOS, CERNBox & SWAN
- EOS as a **filesystem**
- Evolution **data processing** - object storage models

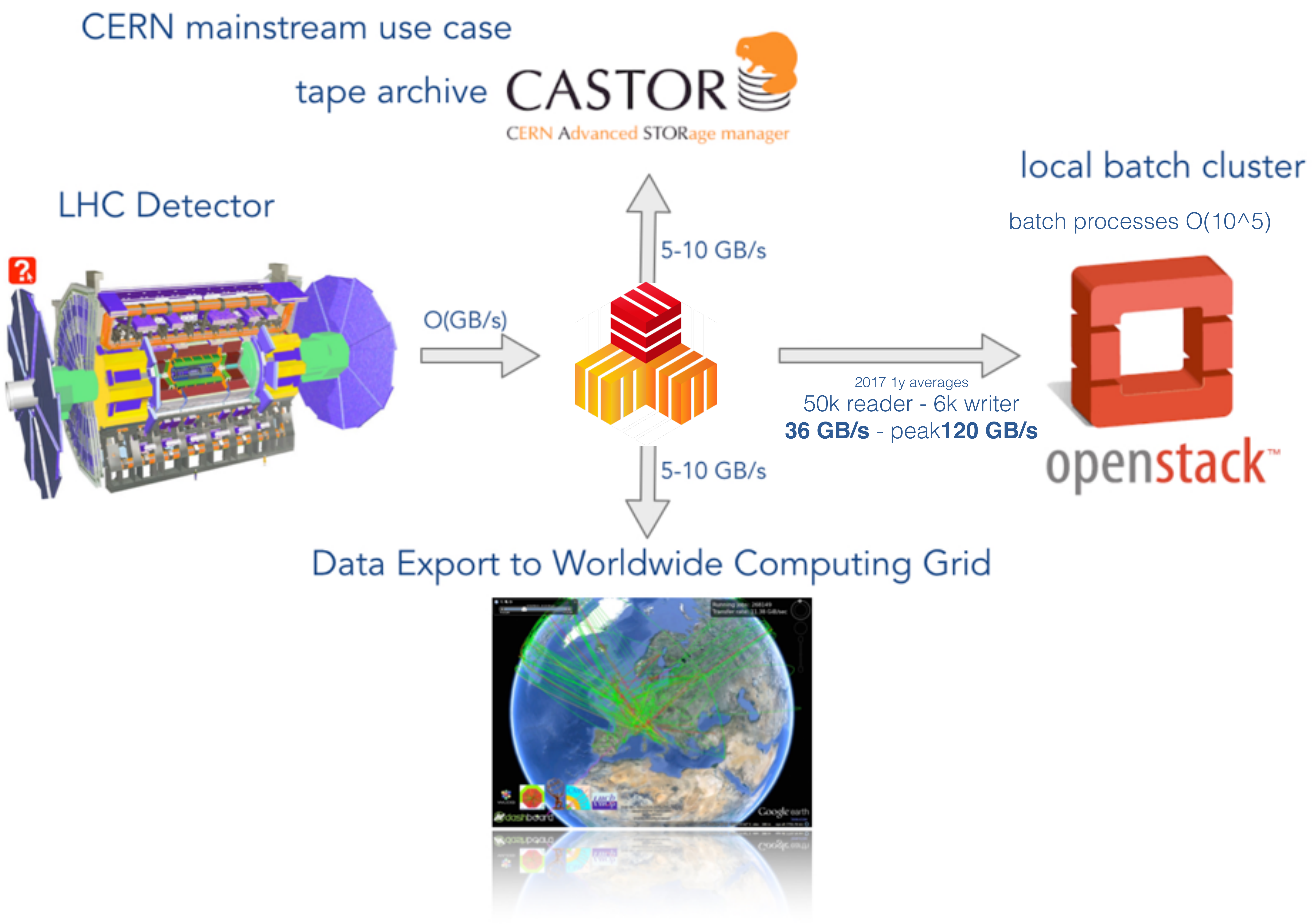
What is EOS ?

- **disk storage system** designed to serve **physics analysis** use cases
  high concurrency, pseudo-random access, LAN/WAN clients
- implemented as plug-ins into the **XRootD** framework
- native transport protocol is **XRootD** [optimized for latency compensation]
- code is written in C++ in IT-ST group at CERN

What is EOS ?


Clients
XRootD, gridFTP, HTTP(S), Owncloud, Filesystem
Protocols
HTTP
XRootD
HTTP
XRootD
XRootD, gridFTP, HTTP(S)
Meta Data/Namespace
MGM
Messaging
MQ
Data/Object Store
FST
FST
FST
FST

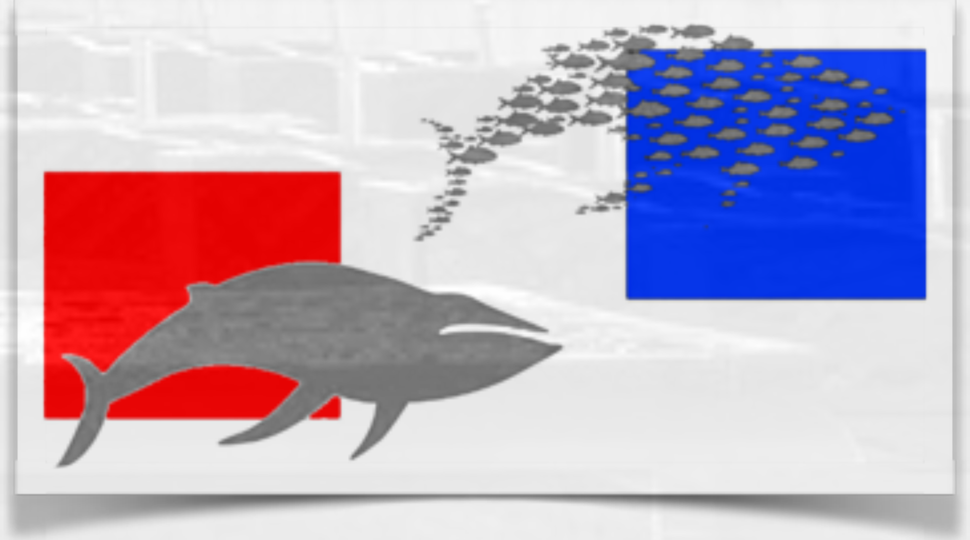
LHC Use Case



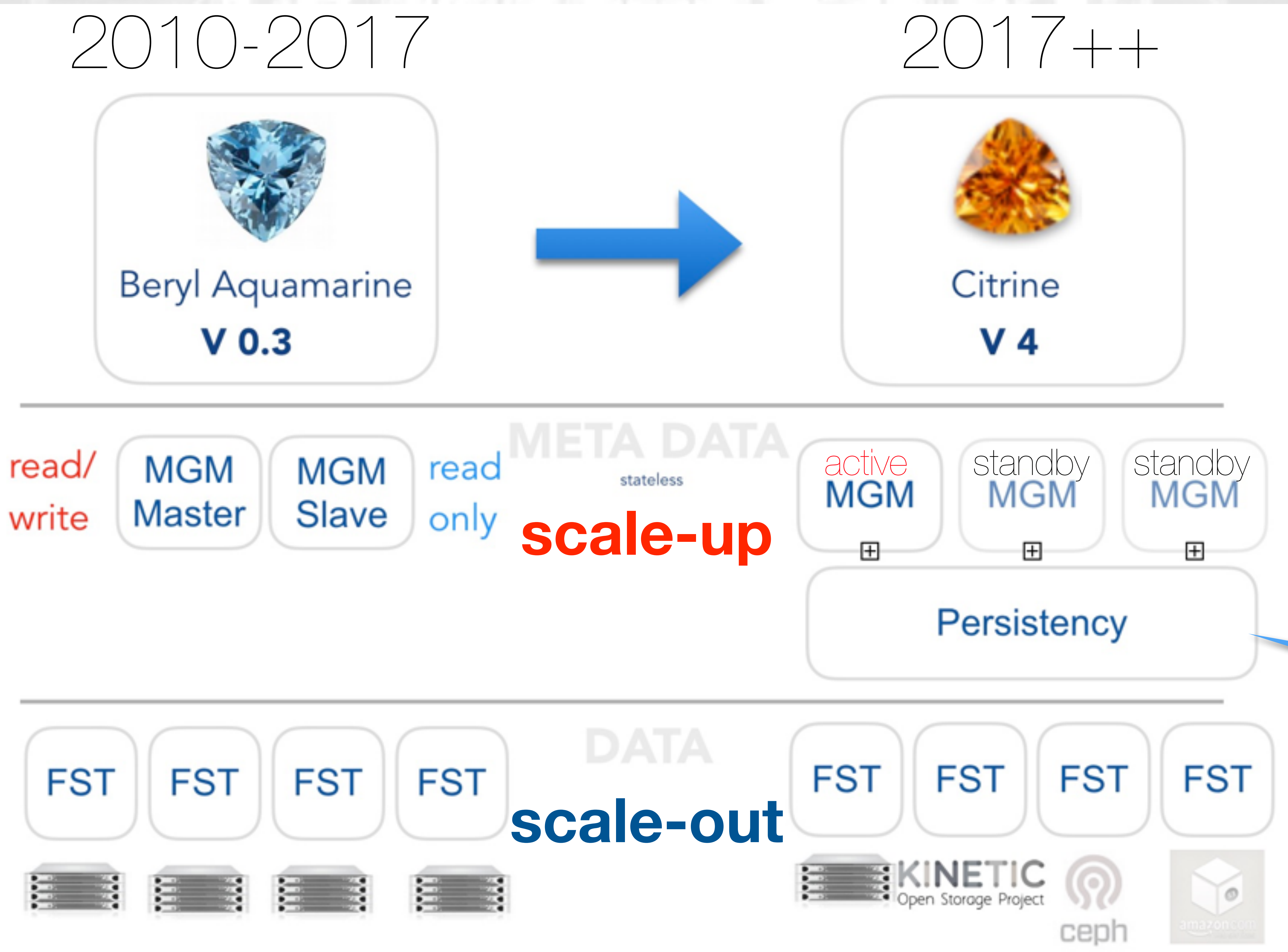
2017 1y averages
50k reader - 6k writer
36 GB/s read - peak **120 GB/s**

Project History

Introduction
Architectural evolution



stateful in-memory

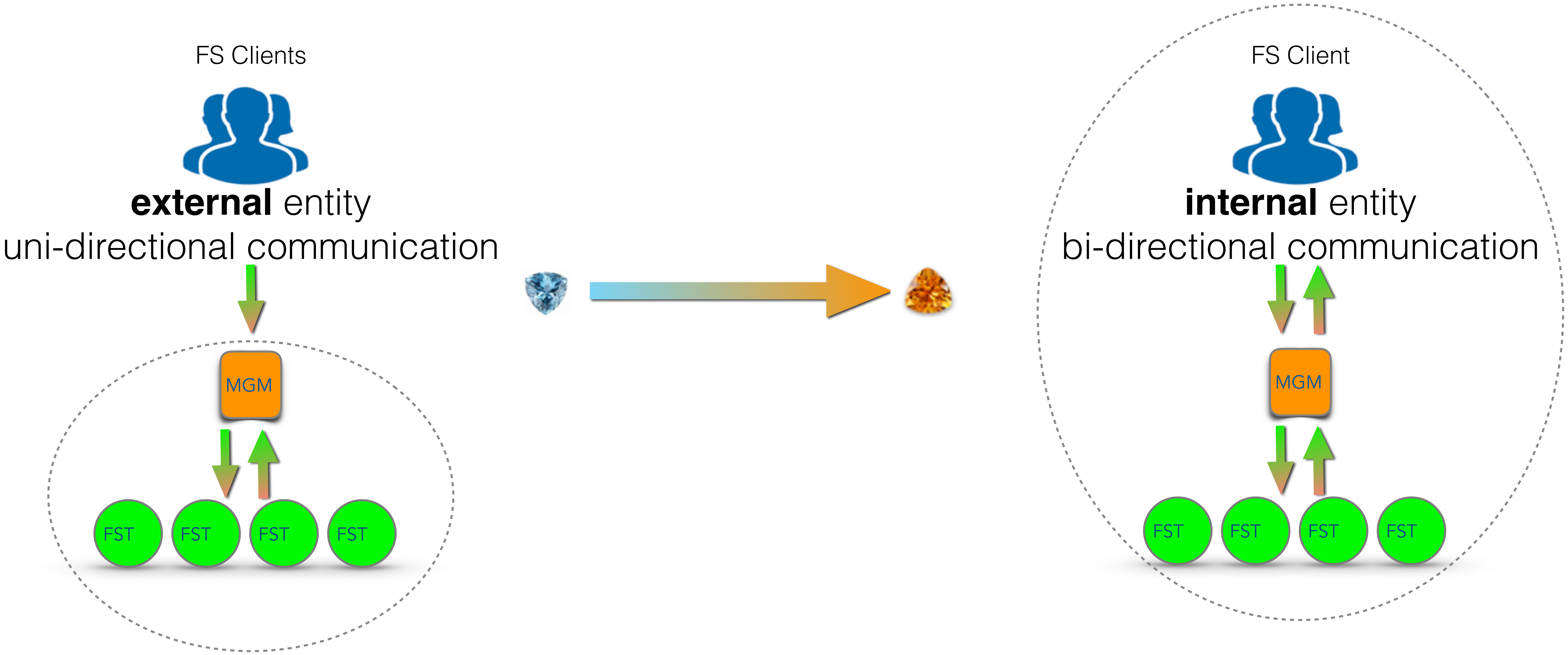


stateless + cache in-memory

Parallel Sessions
[Scaling the EOS namespace]

meta data service daemon becomes stateless

Introduction
Architectural evolution



better support of filesystem semantics requires
FS clients receiving call-backs

EOS service at CERN

designed as a 'lossy' service with two replica CERN/Wigner file replication

2017

statistics

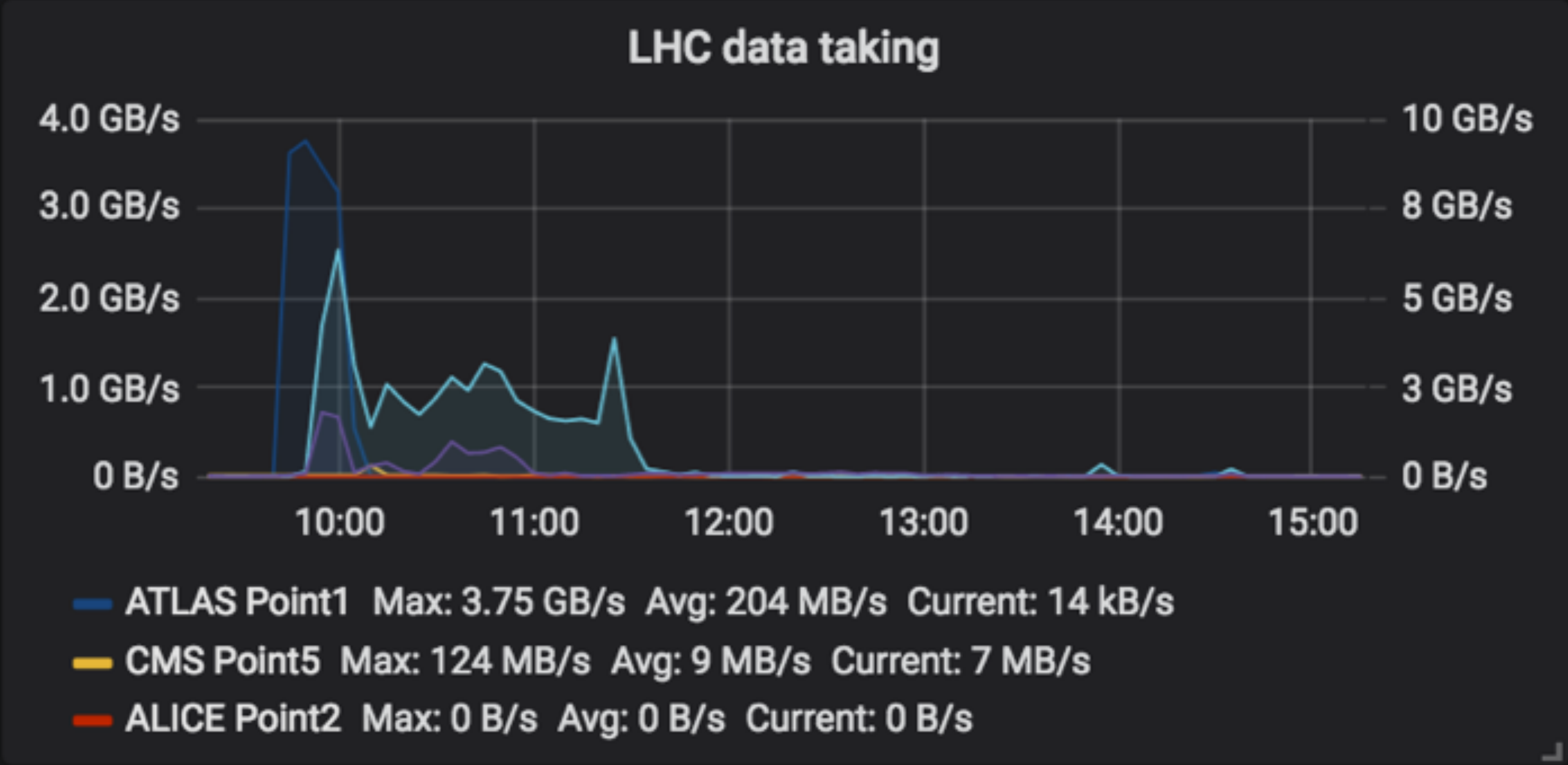
```
bytes read        1.00 EB/a
bytes written     0.25 EB/a
disk  IO          7.90 EB/a
hard disks        ~ 50k
streams           ~ 55k
fileloss rate     ~ O(10⁻⁶)/a
```

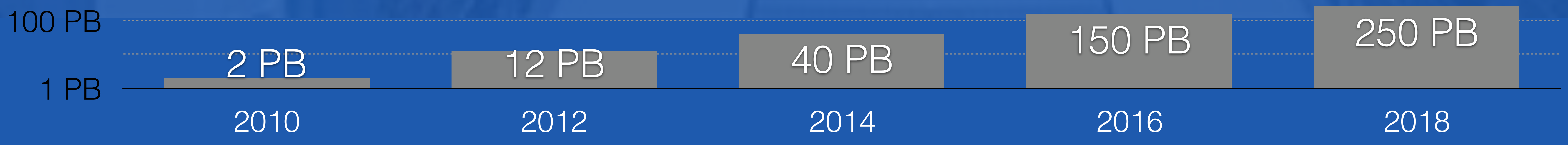
Parallel Sessions
Providing large-scale disk storage at CERN

Parallel Sessions
Disk failures in the EOS setup at CERN

Number of Files	Number of Directories	Total Space	Free Space	MGM # of open FDs
2.969 Bil	181 Mil	244 PB	66.2 PB	44775

Current Writers	Current Readers	IOPS	Write Throughput	Read Throughput
27.0 K	62.5 K	188 K	3.94 GB/s	35.2 GB/s





EOS Disk Space at CERN

EOS service at CERN

- *cheap* disk storage
- > 1.300 server, 50k disks
- JBOD with dual (geo-) replication
- 48-192 disks per standard head-node (batch server) in production
- new big server provide **2 PB** storage capacity
- **BEER** containerised batch jobs on EOS disk server
  EOS storage service daemon light-weight, allows to use 90% of free CPU resources

Parallel Sessions
Sharing server nodes for storage and compute

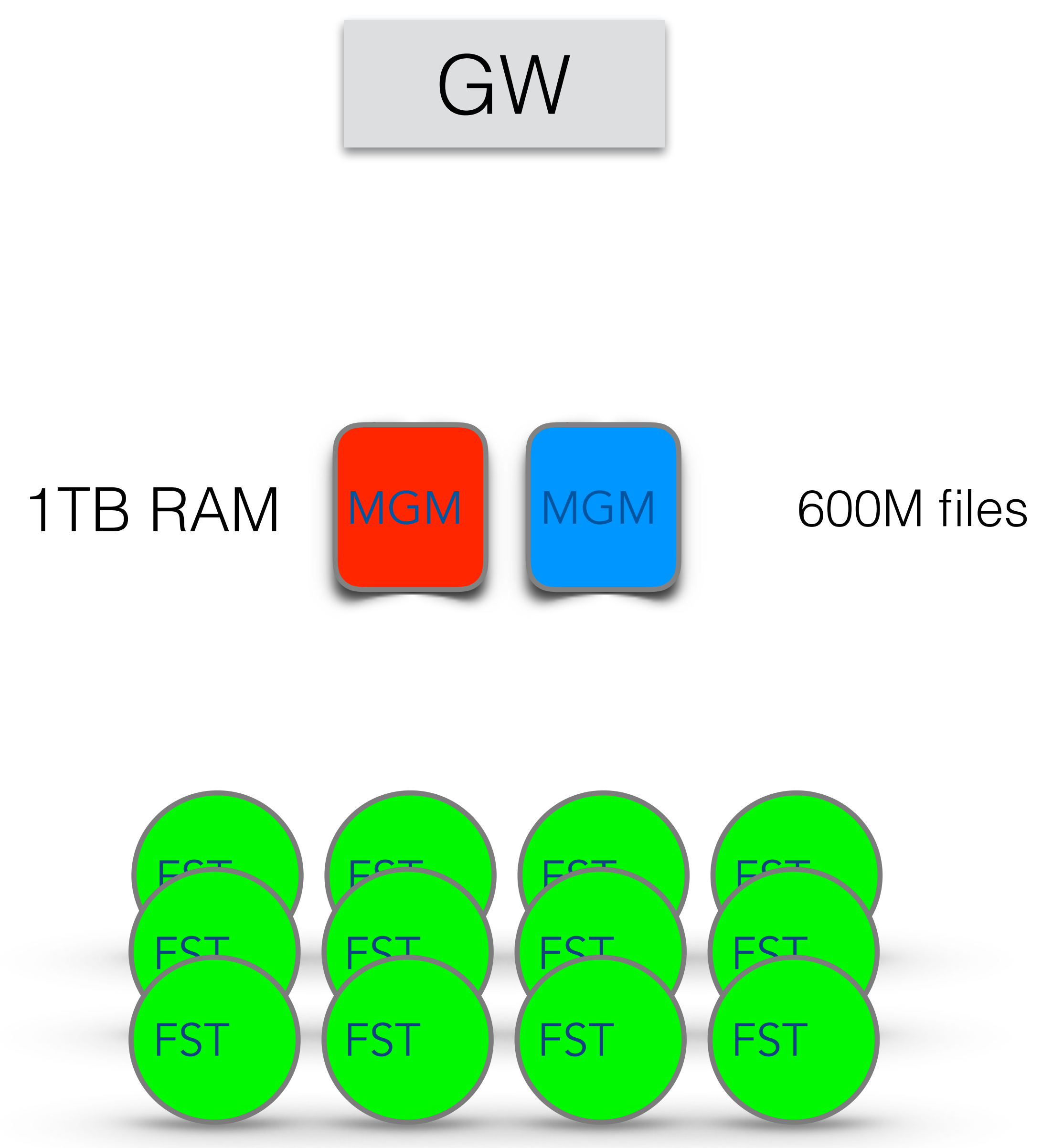
cheap volume storage on HDD

A new CERNBOX backend

Segmented high-available service model

CERNBOX now

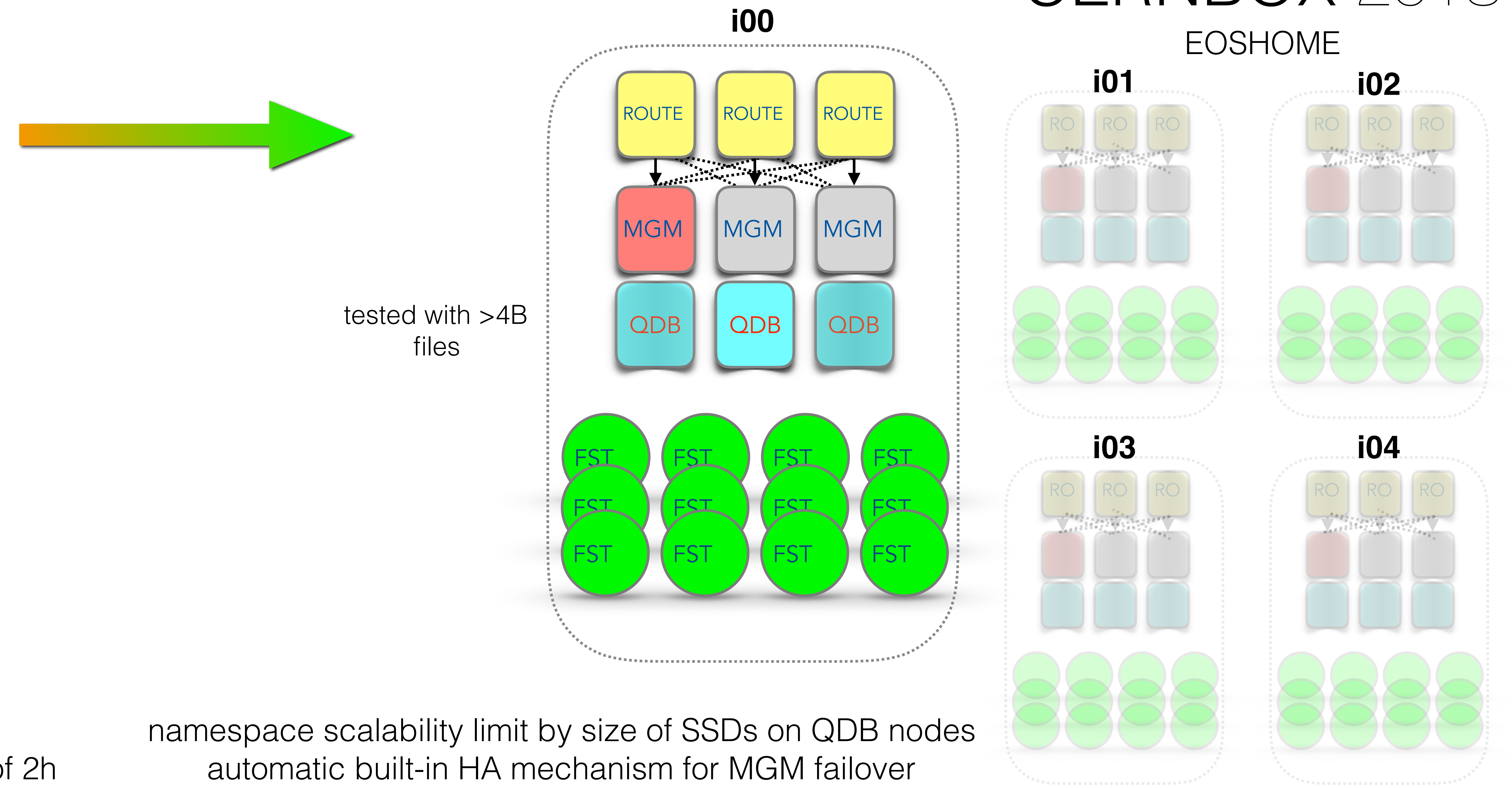
EOSUSER



at namespace scalability limit
availability constrained by infrequent long boot time of 2h

CERNBOX 2018

EOSHOME



namespace scalability limit by size of SSDs on QDB nodes
automatic built-in HA mechanism for MGM failover

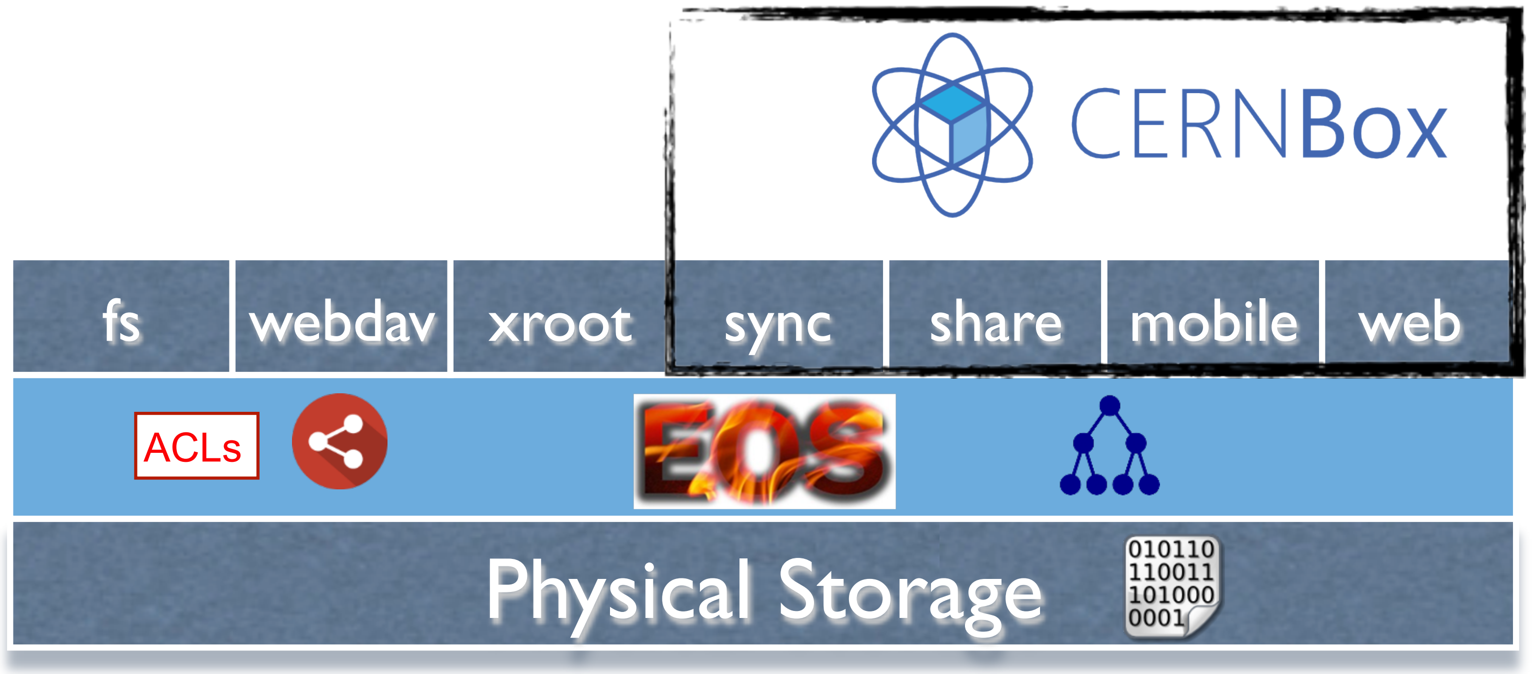
Since 2017 service running over scalability limit - new service architecture

2015++

EOS

CERNBox

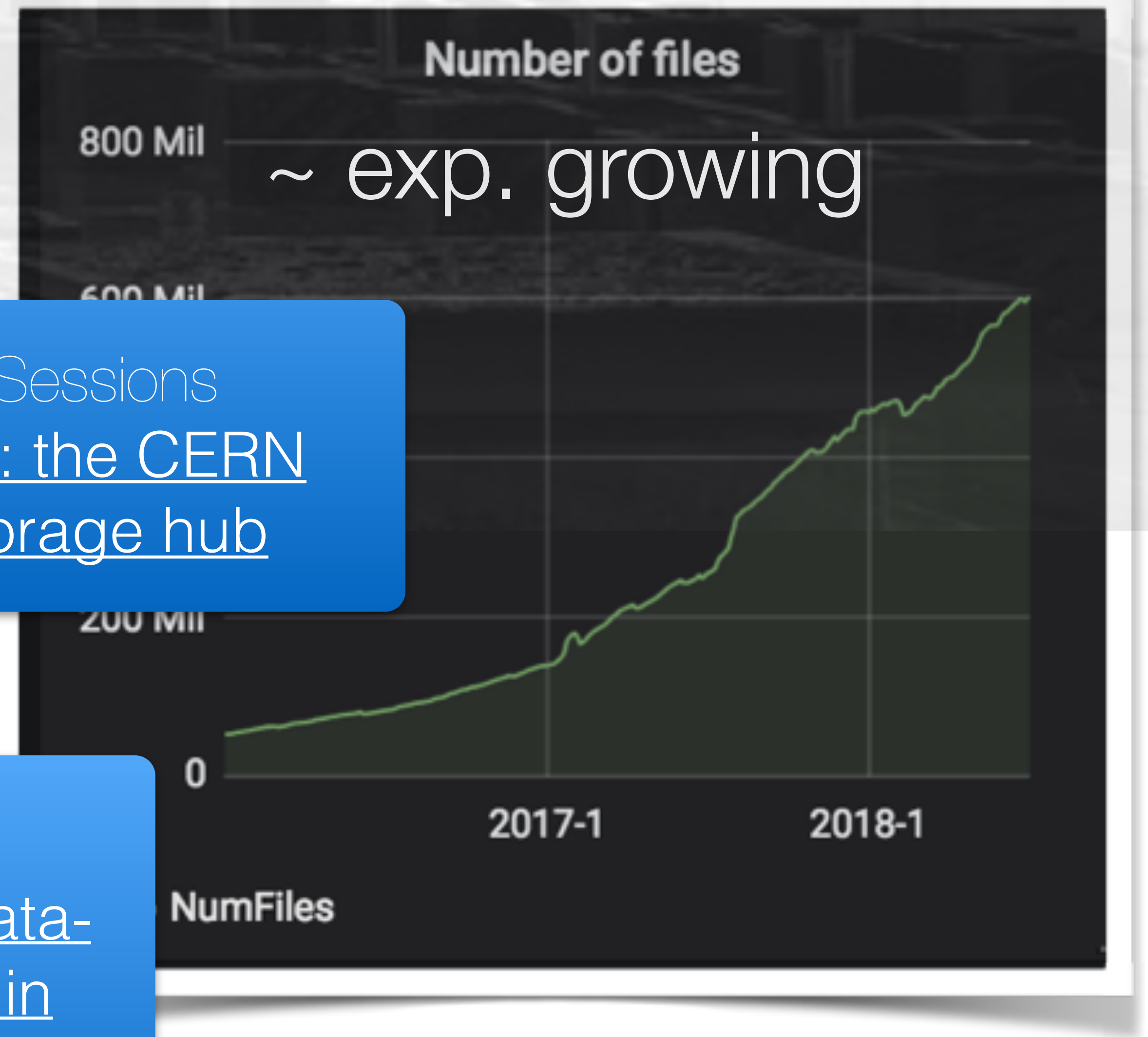
building block of an ecosystem for sc tific
data reposito ies



Poster Sessions
The EU Up to University Project

Parallel Sessions
Cloud Storage for data-intensive sciences in science and industry

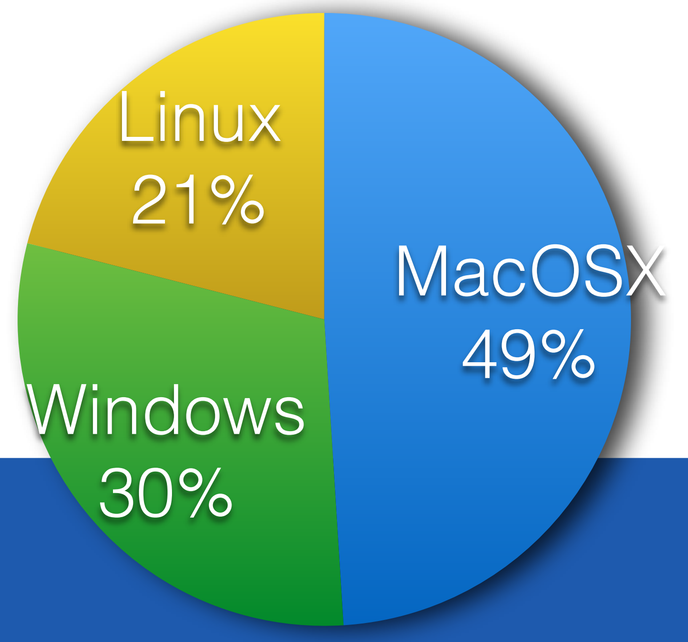
Parallel Sessions
CERNBox: the CERN cloud storage hub



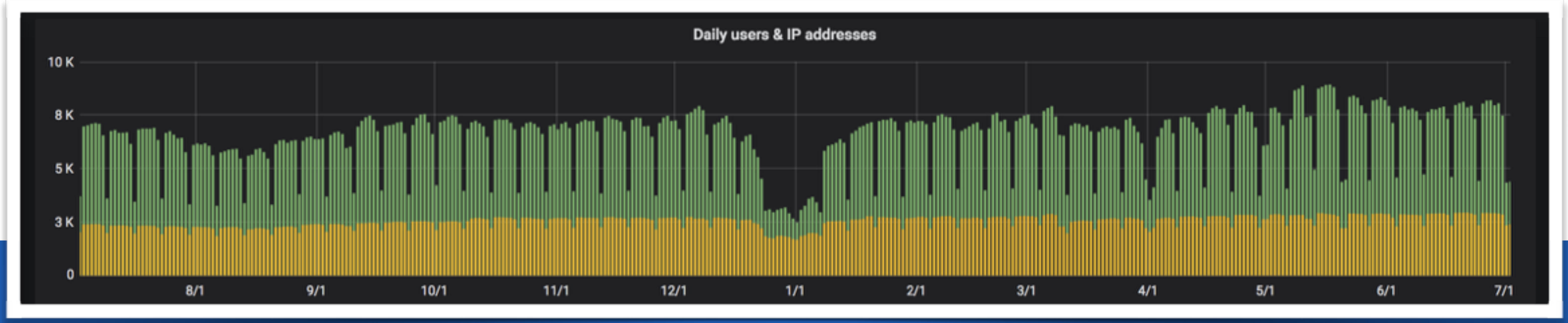
- “**dropbox**” for science
- **cloud storage**, **synchronisation** and **file sharing** service
- implemented as **web services** in front of EOS backend

Office

since mid 2017 support for collaborative editing



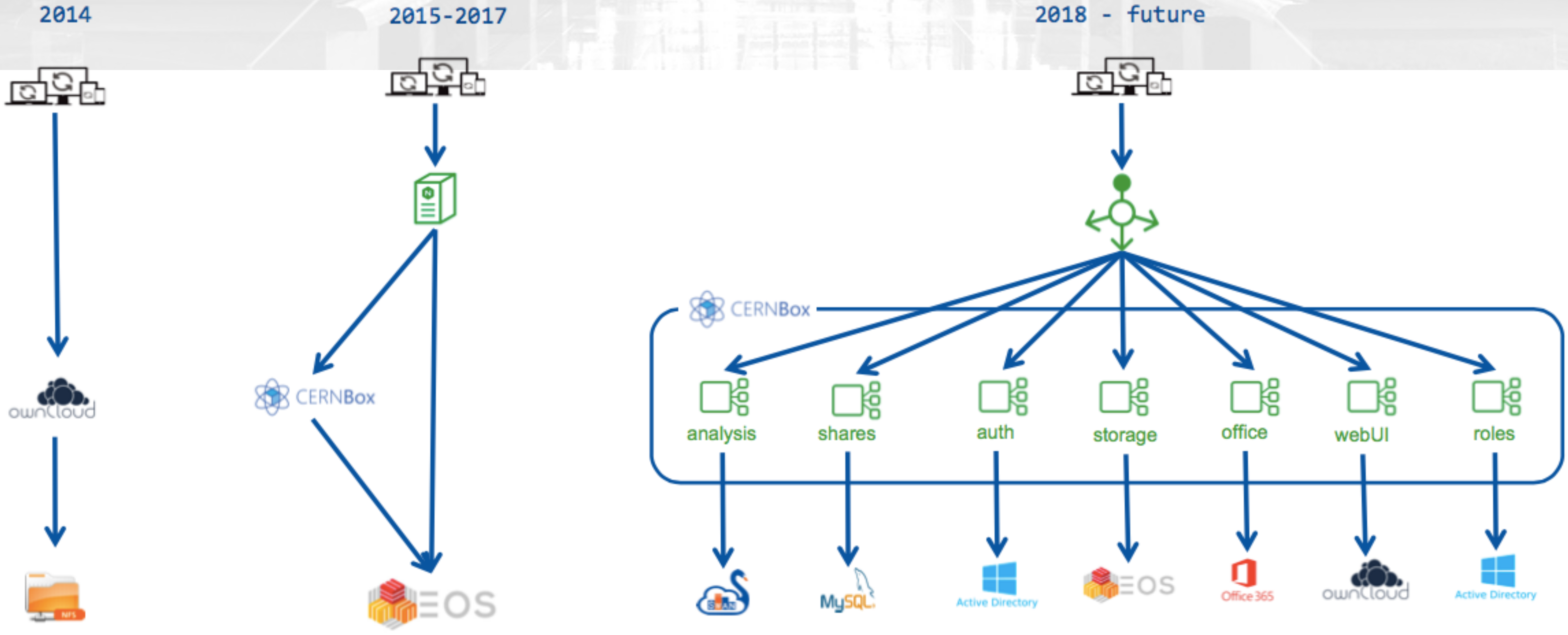
connected client platforms



~3.000 daily users, 9k connected devices

CERN

CERNBox Evolution



CERNBox refactored using micro service approach - boost performance & functionality

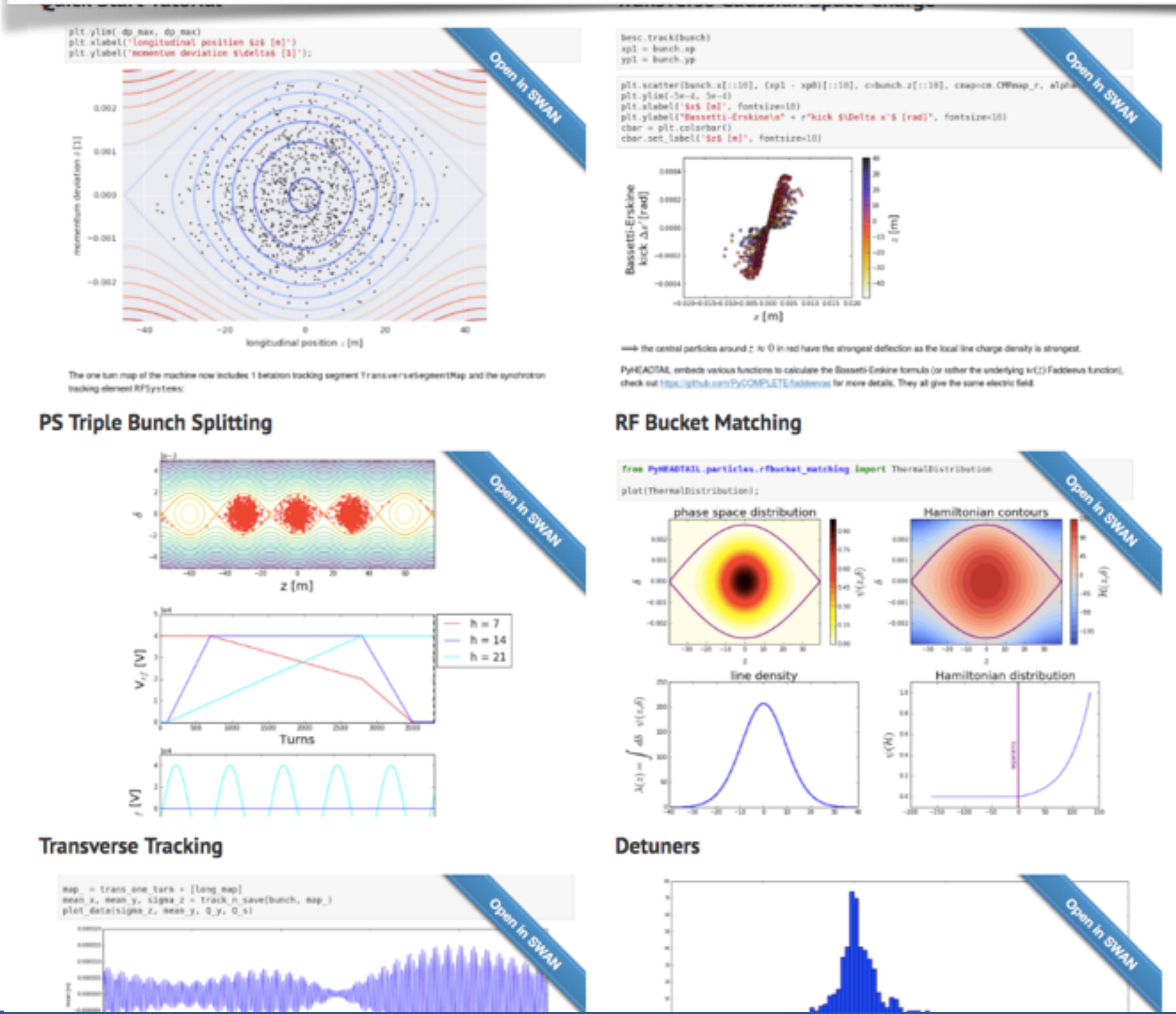
2016++

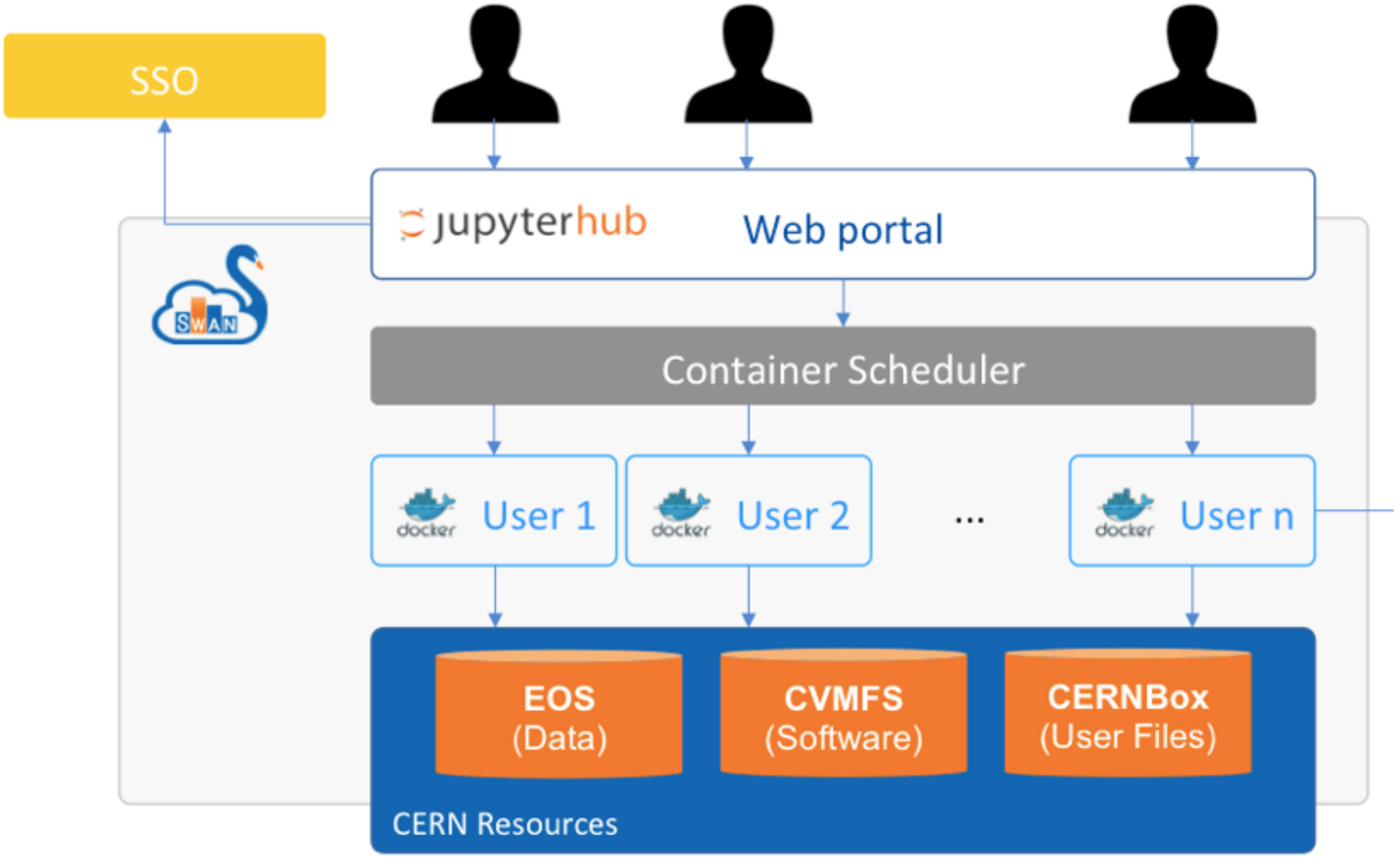
SWAN service

web-based analysis

swan.web.cern.ch

Parallel Sessions
Facilitating collaborative analysis in SWAN

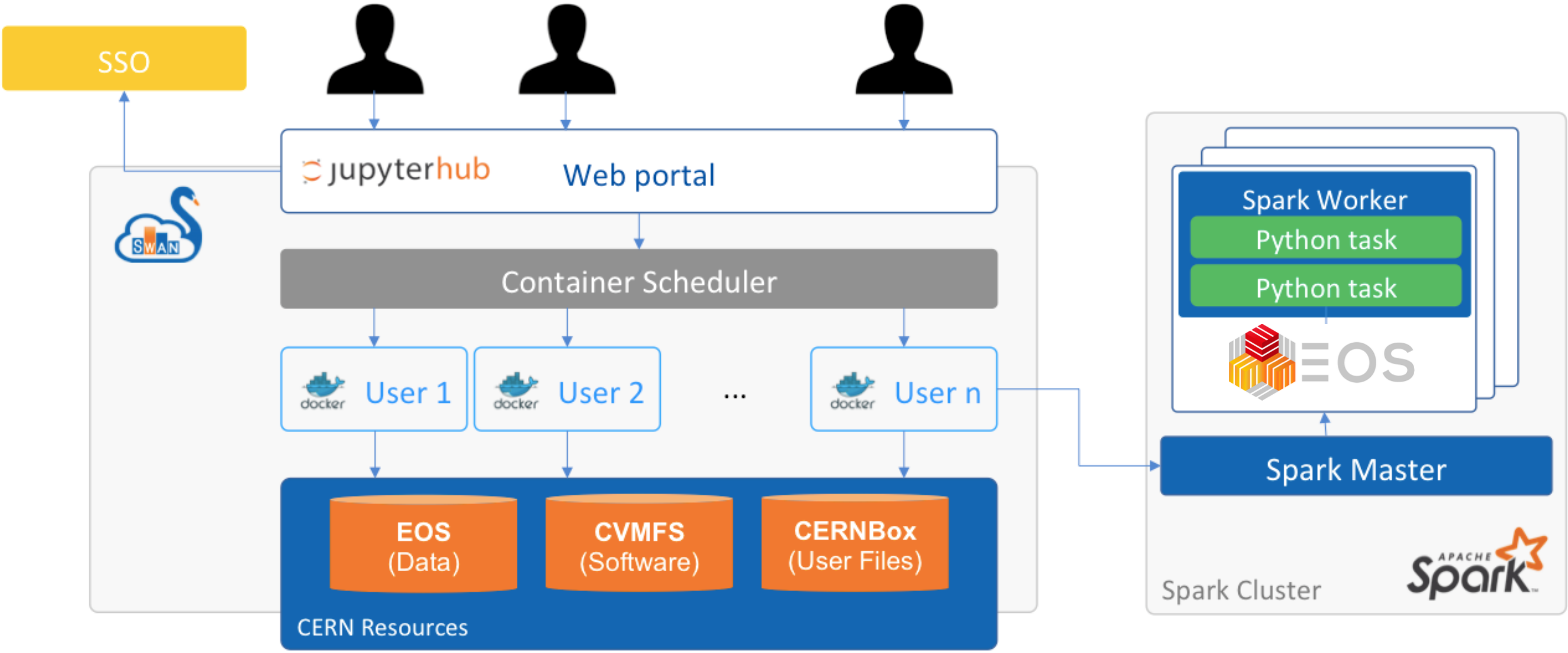




service architecture

SWAN provides interactive analysis front-end using JUPYTER notebooks

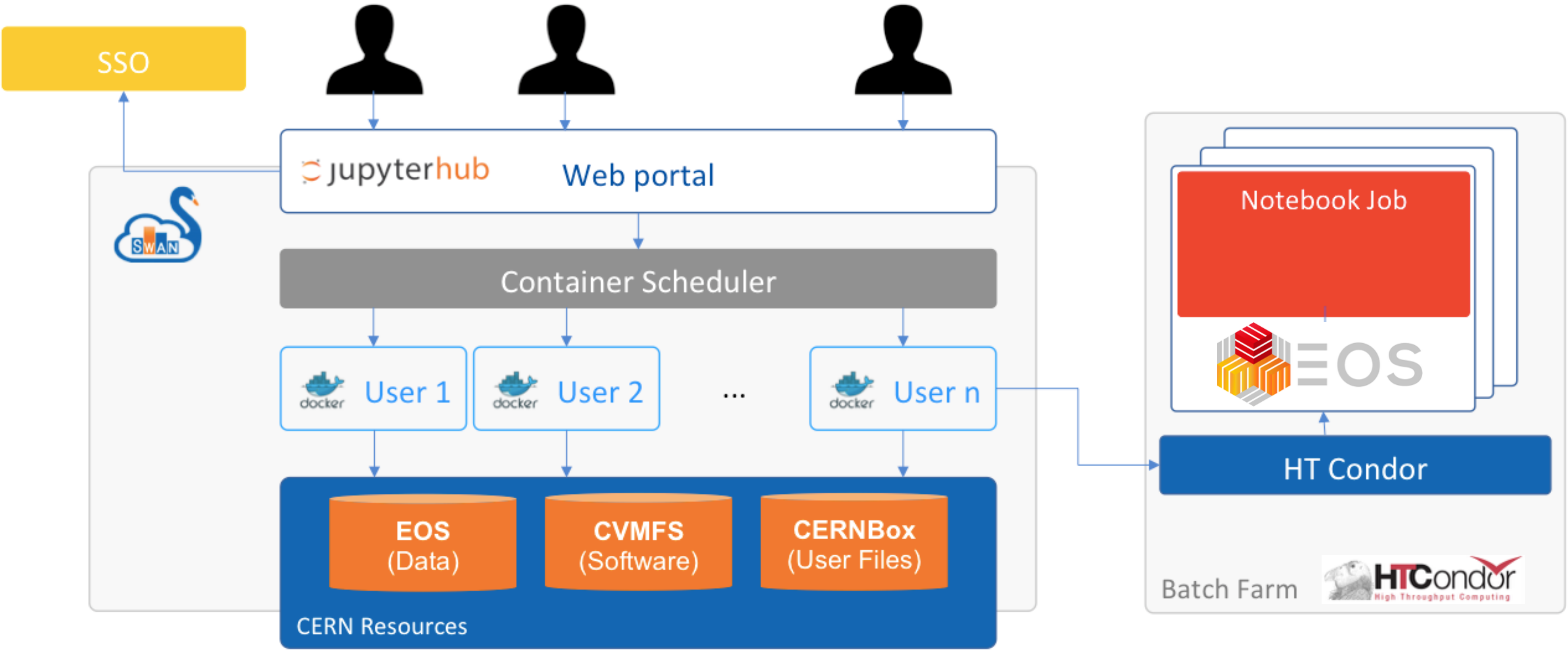
SWAN & compute



SWAN interfacing to Spark Cluster

SWAN & compute

2018++



next: SWAN interfacing to Batch Cluster

2017++

packaged eco-system
Science Box

EOS + CERNBox + SWAN

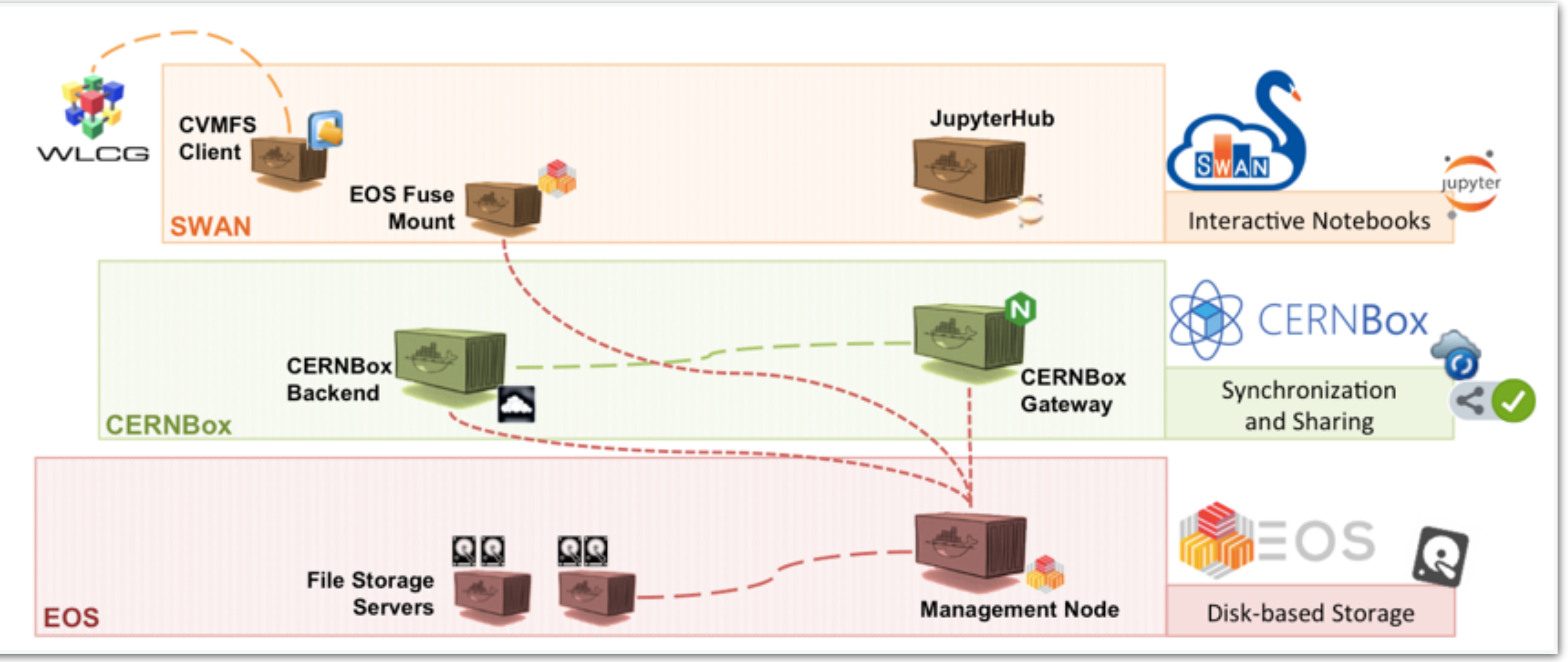
docker

One-Click Demo Deployment
- Single-box installation via **docker-compose**
- No configuration required
- Download and run services in 15 minutes

https://github.com/cernbox/uboxed

Production-oriented Deployment
- Container orchestration with **Kubernetes**
- Scale-out storage and computing
- Tolerant to node failure for high-availability

https://github.com/cernbox/kuboxed



Science Box provides an easy demo & production platform

2018++

EOS + Tape = EOSCTA

Parallel Sessions
CERN Tape Archive: From Development to Production Deployment

integrated support for tape into EOS file on tape=offline replica

- loose service coupling between EOS and CTA via protocol buffer interface & notification events
- no SRM, using XRootD protocol only for now - integrated with FTS
- pre-production service for ATLAS available

high disk capacity

EOSATLAS

TPC

low disk capacity

EOSATLASCTA

short file lifetime

Operation Model

Cern Tape Archive

Mid-term plan to migrate CASTOR data to CTA

eosxd - a filesystem client for EOS

Why this is important but difficult ...

- mounted filesystem access is required to enable storage access to any software *out of the box*

- filesystem development is difficult and lengthy

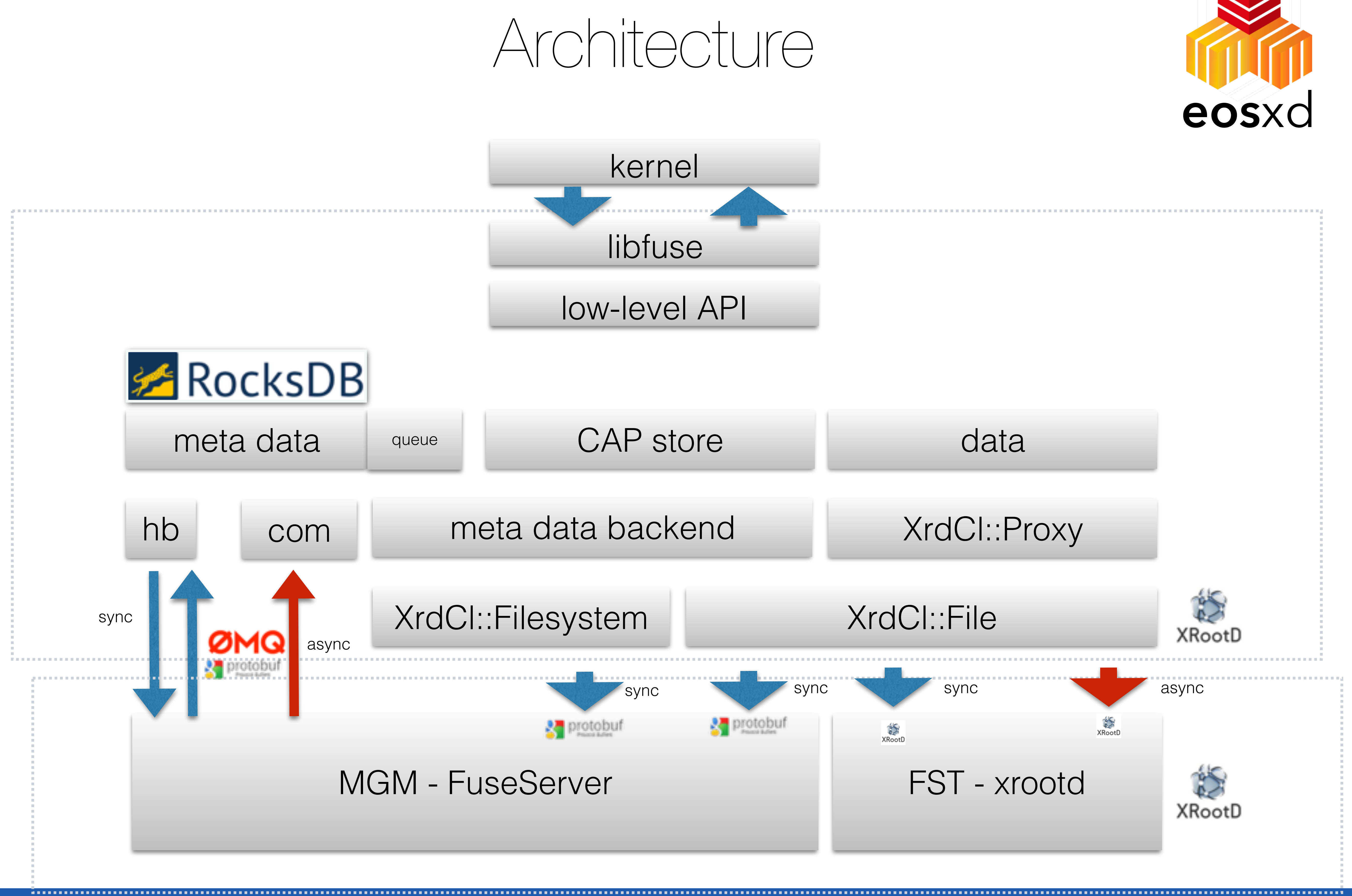
  AFS V1,2,3 - **35 years**
  NFS V1,2,3,4 - **34 years**
  cephfs - **12 years** - production version announced after 10 years!

- EOS filesystem client rewrite started Q4 2016: **eos**d => **eos**xd

Question: **how far can you get with a user-space filesystem implementation?**

eosxd

filesystem daemon



- enough **POSIX**ness
- file **locks**, byte-range locks
- hard **links** within directories
- rich **ACL** client support
- local **caching**
- **bulk deletion**/protection
- strong **security** & mount-by-key
- user,group & project **quota**
- implemented using **libfuse**

eosxd provides POSIXness very similar to AFS

2018++

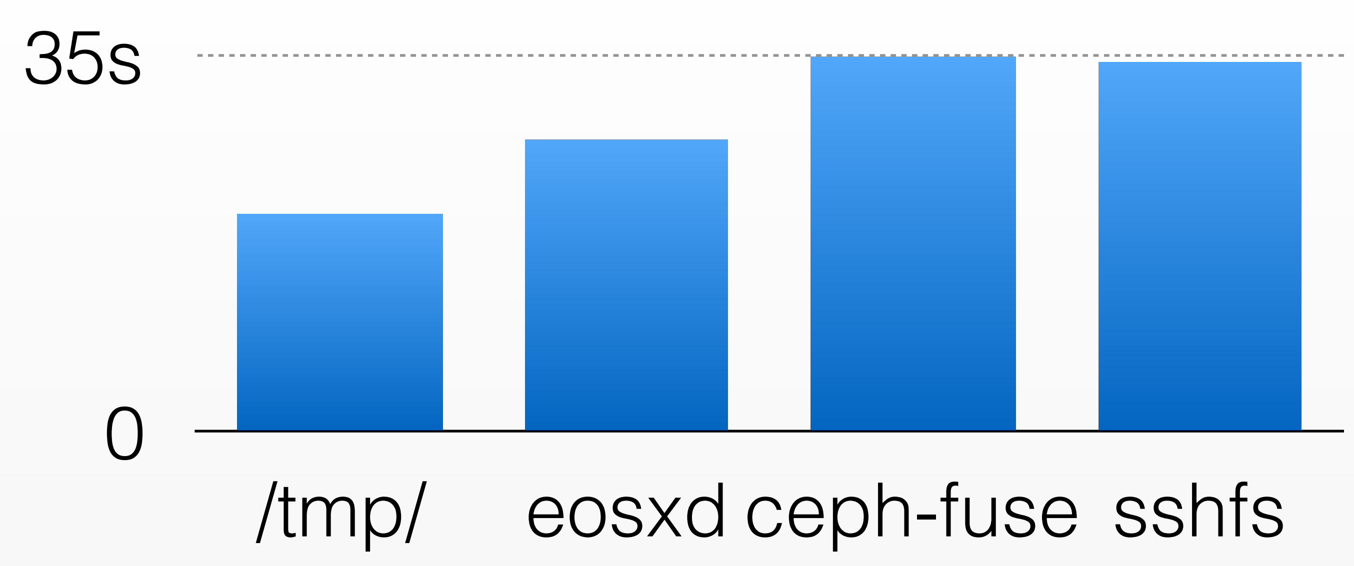
eosxd

FUSE filesystem daemon

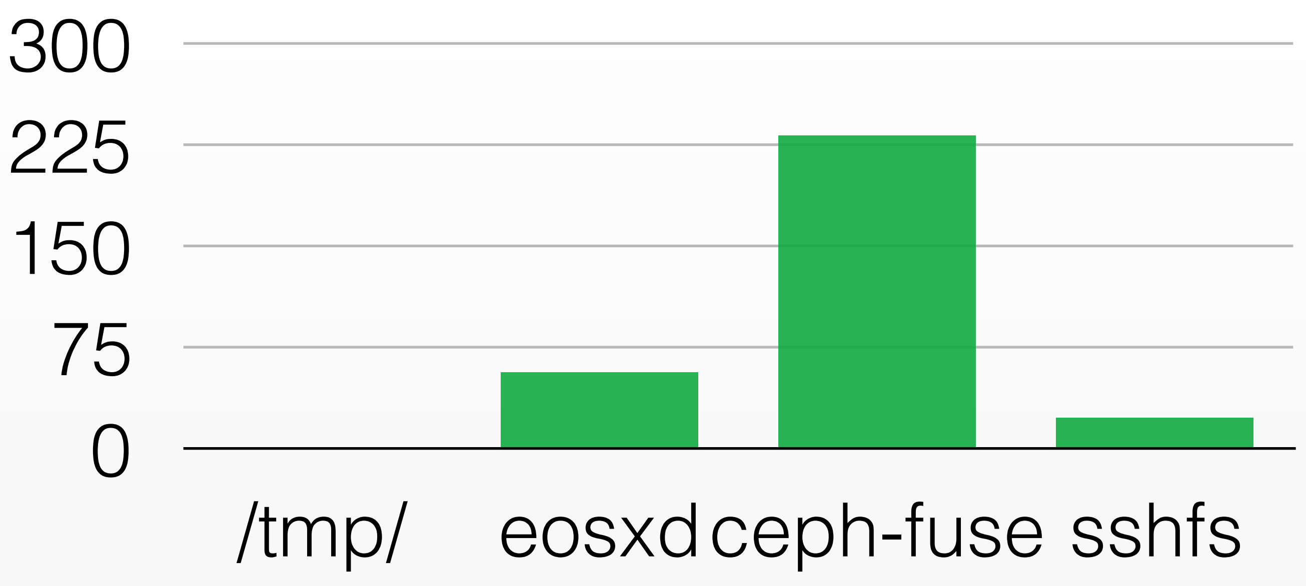
CAUTION

take benchmarks indicative

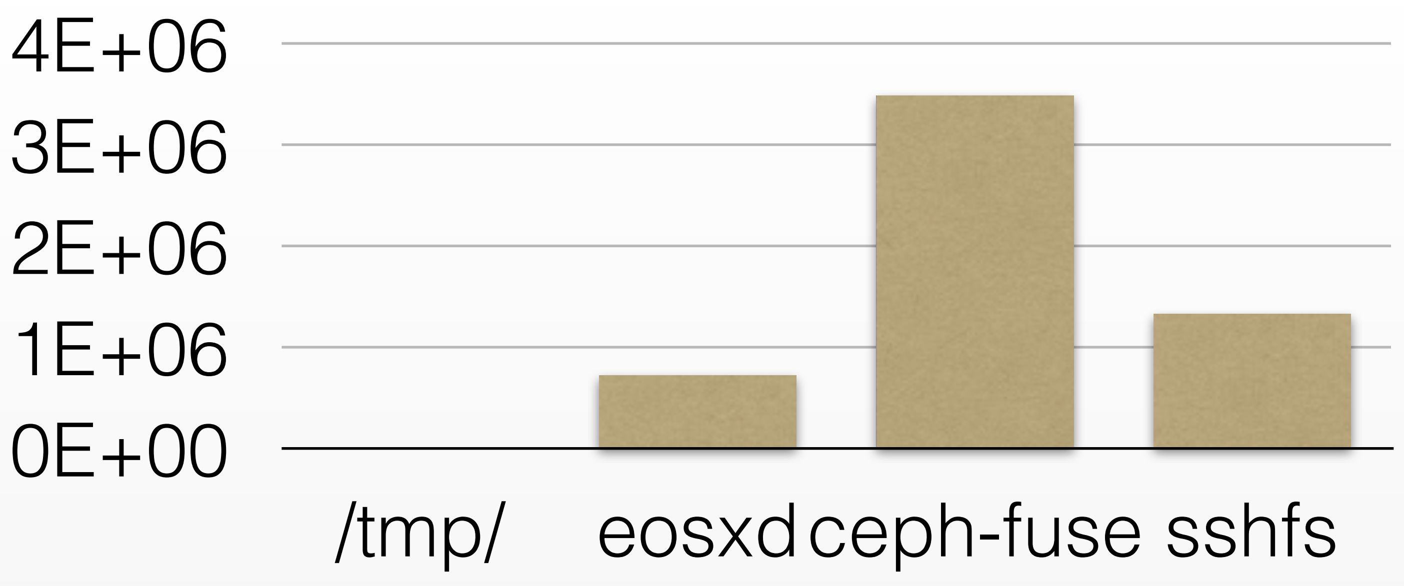
make -j 4



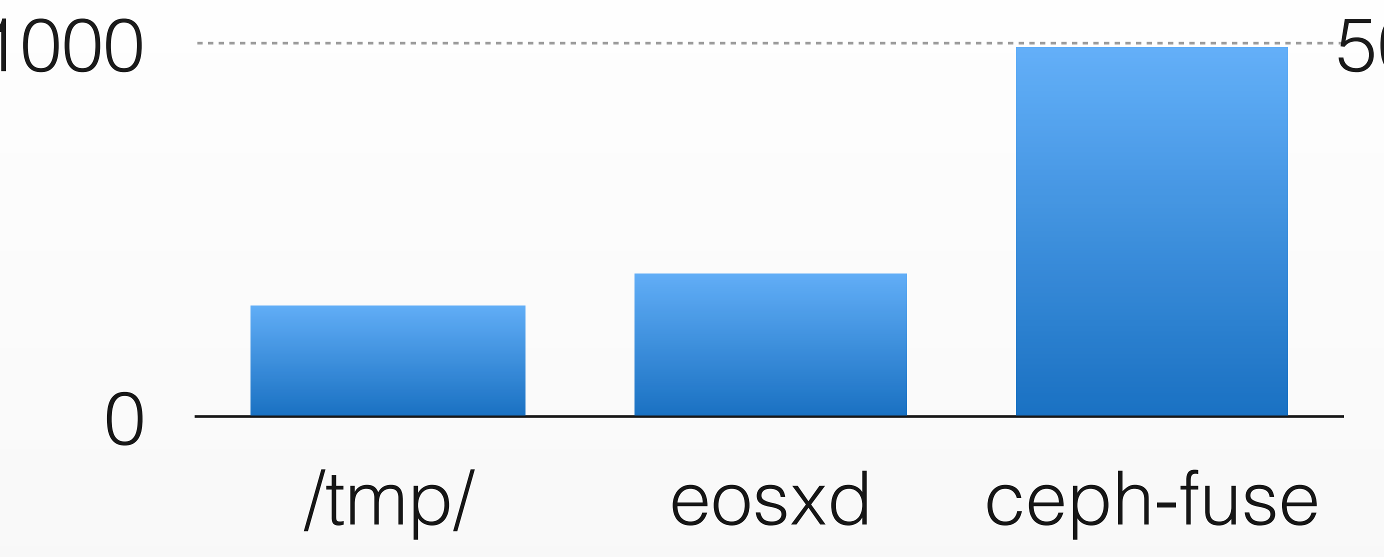
CPU consumption FS



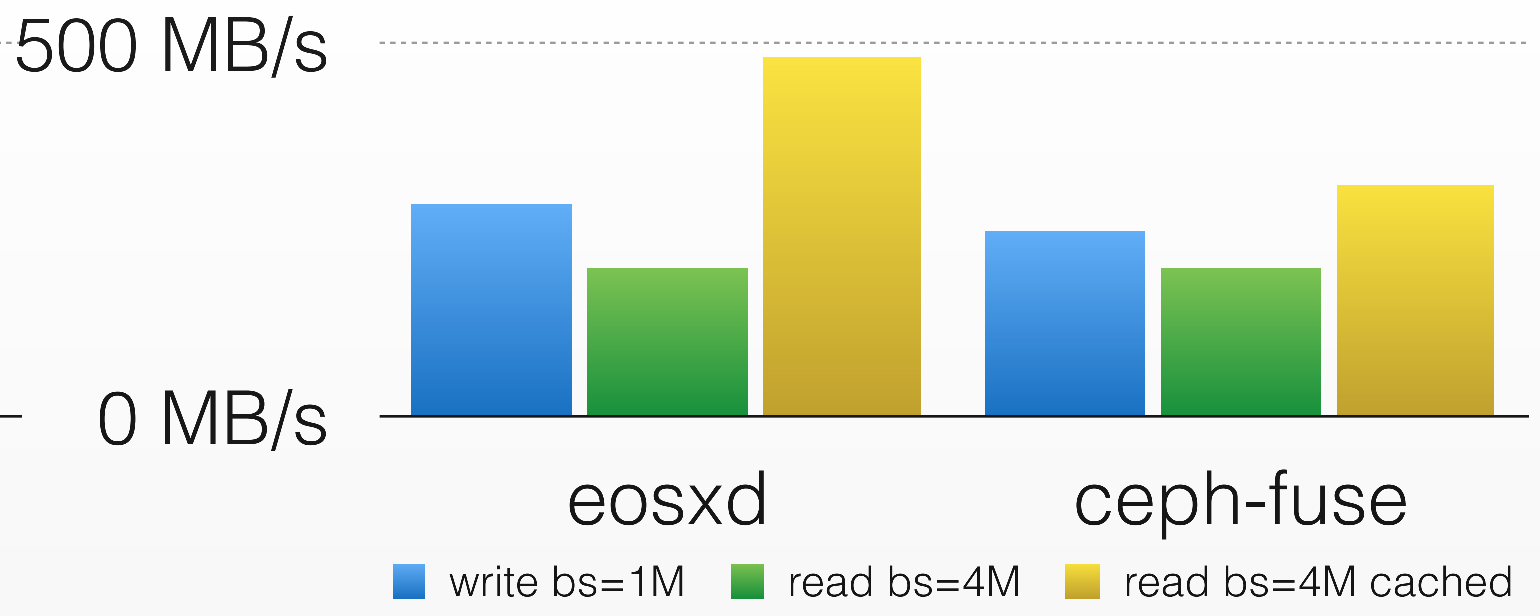
Context Switches



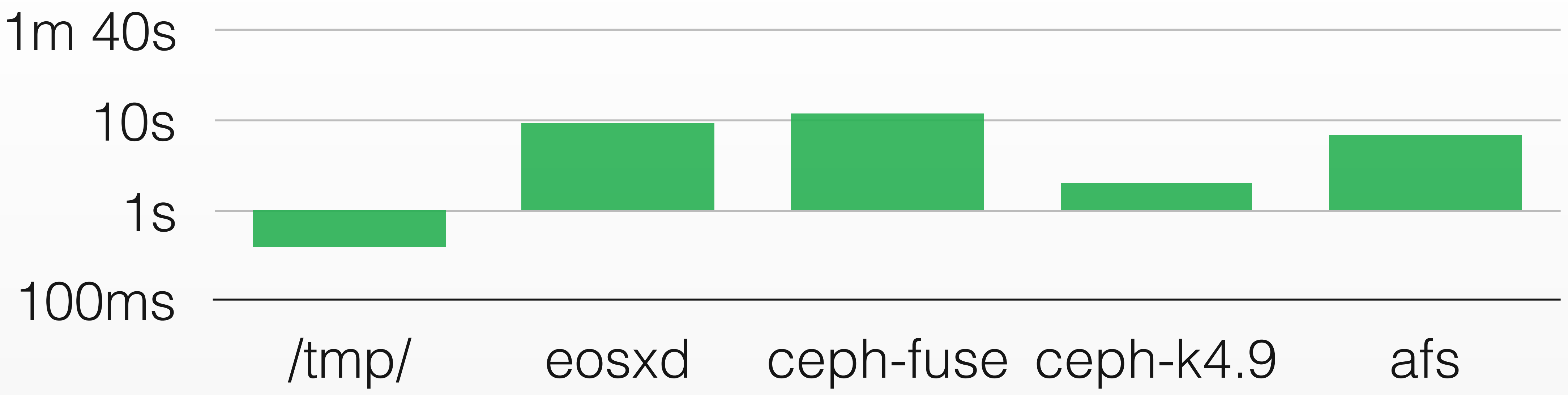
EOS rpm build



Streaming Write



untar



CERN

eosxd good performance with low resource usage

for a filesystem implemented in user space

eosxd

Strong Security Model

application runtime: **export KRB5CCNAME or X509xxx**

eosxd — **kerberos or X509 authentication** → eos

ACL per directory by mapped uid/gid

sys.acl=u:foo:rwx

before mount: **export XrdSecsssENDORSEMENT=<secret>**

eosxd — **shared secret authentication** → eos

ACL per directly by exported secret

sys.acl=k:B8E776C5-F5B2-4EF1-B2C3-64CB7C158FF3:rwx

clients **exports environment variables** in application context
to configure strong authentication - ***root*** role on client is **unavailable**

eosxd

sub-mount feature

glue external filesystems

- **Question:** can we integrate seemingly external filesystems into an EOS mount keeping their full performance?

- automount is a proven solution, but it has a static configuration and can not be configured by a user on the fly

```
/eos/user/f/foo/                    → EOS area
/eos/user/f/foo/software/root6      → software image
/eos/user/f/foo/hpc                 → manila share
/eos/user/f/foo/s3                  → S3 bucket
/eos/user/f/foo/backups             → backup snapshots
```

Short answer: yes we can!

2018++

EOS

eosxd

sub-mount feature

glue external filesystems

- allows **eos**xd to mount on-the fly any kind of filesystem described by a symbolic link in the EOS namespace
  - implemted: **squashfs** images with e.g. software distributions …
    - extremely space efficient file distribution with **zstd** compression, export millions of small files as a single image file
    - high-performance kernel module or FUSE module available

```
-rw-r--r--    1 nobody    nobody       256622592 Jun 29 18:04 .gcc-4.9.3.sqsh
lrwxrwxrwx    1 nobody    nobody               1 Jun 29 18:04 gcc-4.9.3 -> squashfuse:
```

  - envisaged: **external filesystem** areas e.g. high-performance *manila* shares, *s3* buckets etc. …
    - store cephx or s3 key as private extended attribute in EOS
  - envisaged: *restic* **backup snapshots** of user areas with restore password in extended attributes in EOS
    - browse/recover existing backups stored in an external instance without help from a service manager

WORK IN PROGRESS

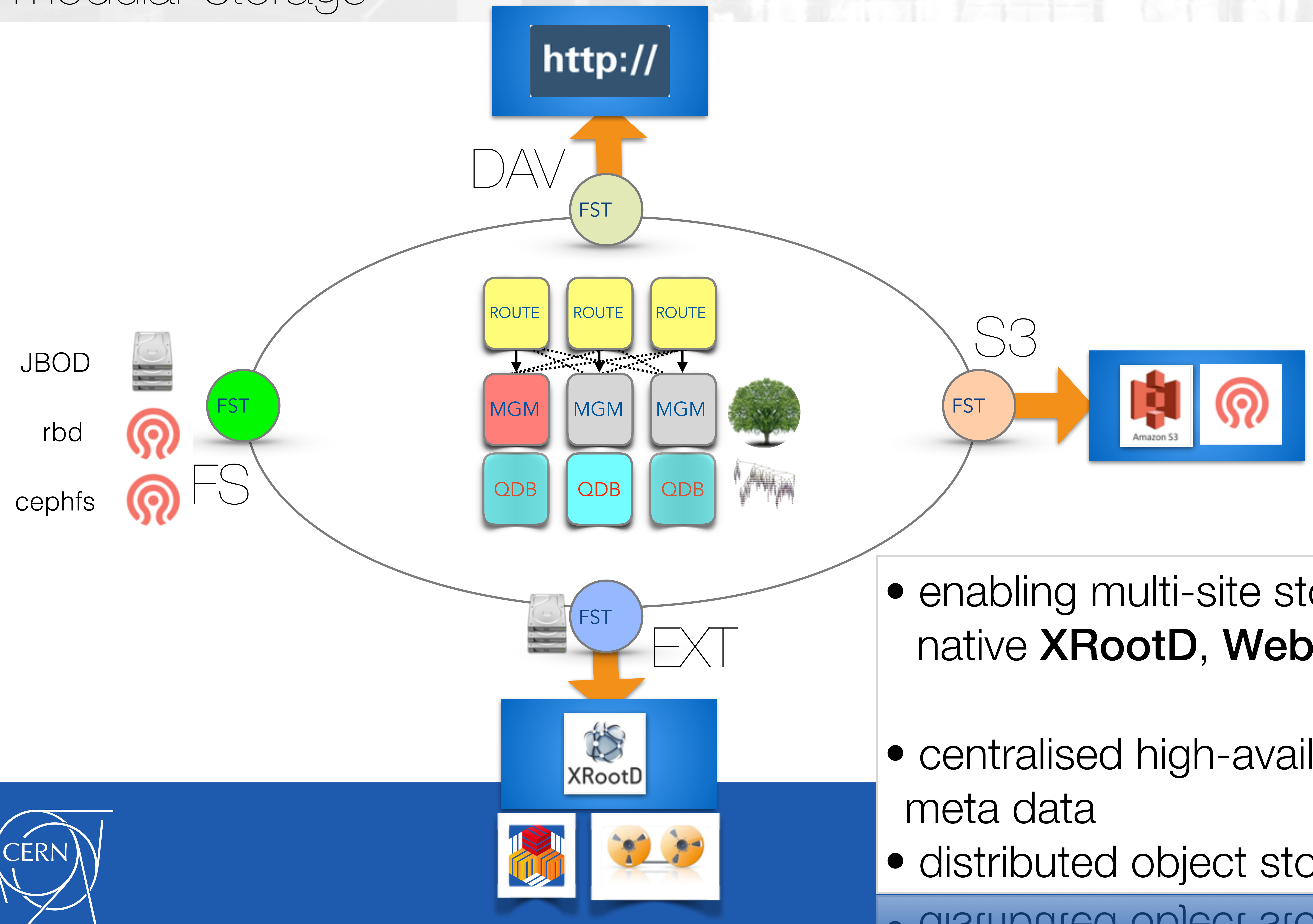
```
EOS Console [root://localhost] |/eos/wfe/submount/> squash
usage: squash new <path>                                     : create a new squashfs under <path>
       squash pack <path>                                    : pack a squashfs image
       squash unpack <path>                                  : unpack a squashfs image for modification
       squash info <path>                                    : squashfs information about <path>
       squash rm <path>                                      : delete a squashfs attached image and its smart link
```

CERN

eosxd **leverages performance of external optimised filesystems**

2018++ EOS modular storage

Distributed Storage Architecture



- enabling multi-site storage supporting native **XRootD**, **WebDav**, AWS/CEPH **S3** or **FS** storage
- centralised high-available namespace in KV store for meta data
- distributed object store for data

eXtreme DataCloud

CERN

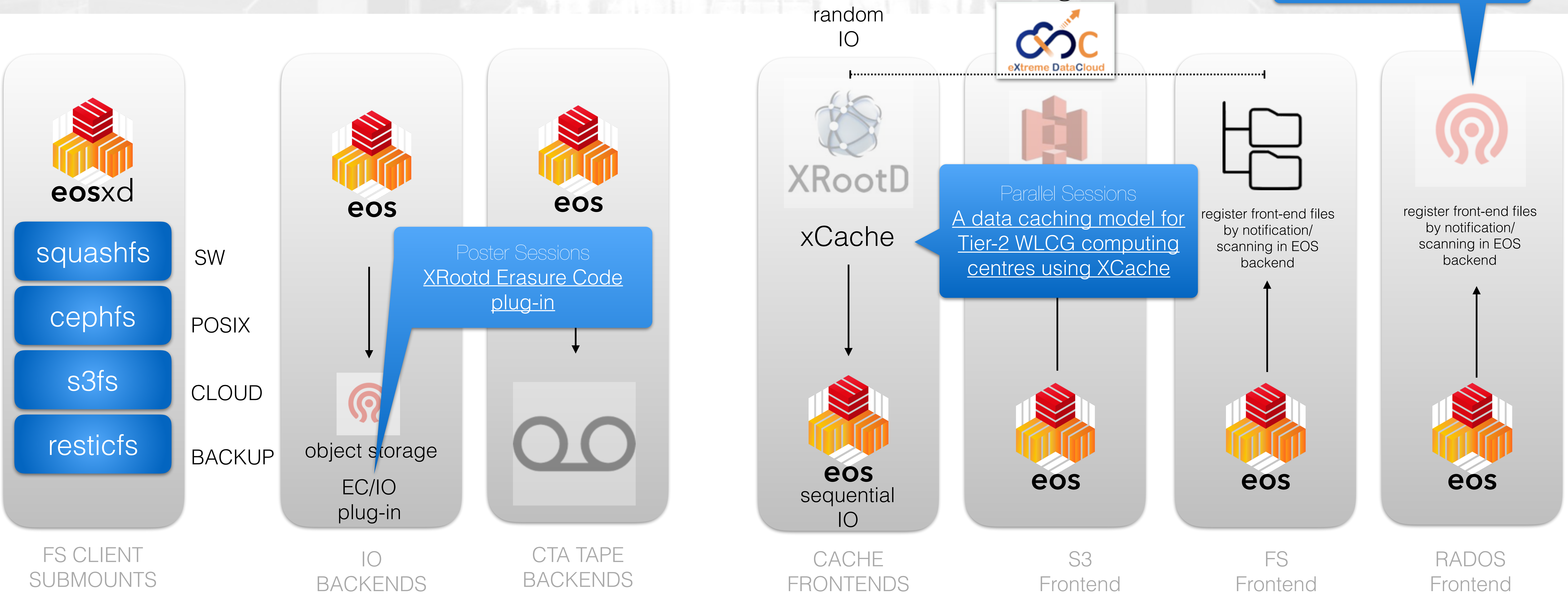
2018++

EOS

Modular Storage

client sub-mounts, IO backends, storage frontends

Parallel Sessions
Ceph File System for the CERN HPC infrastructure



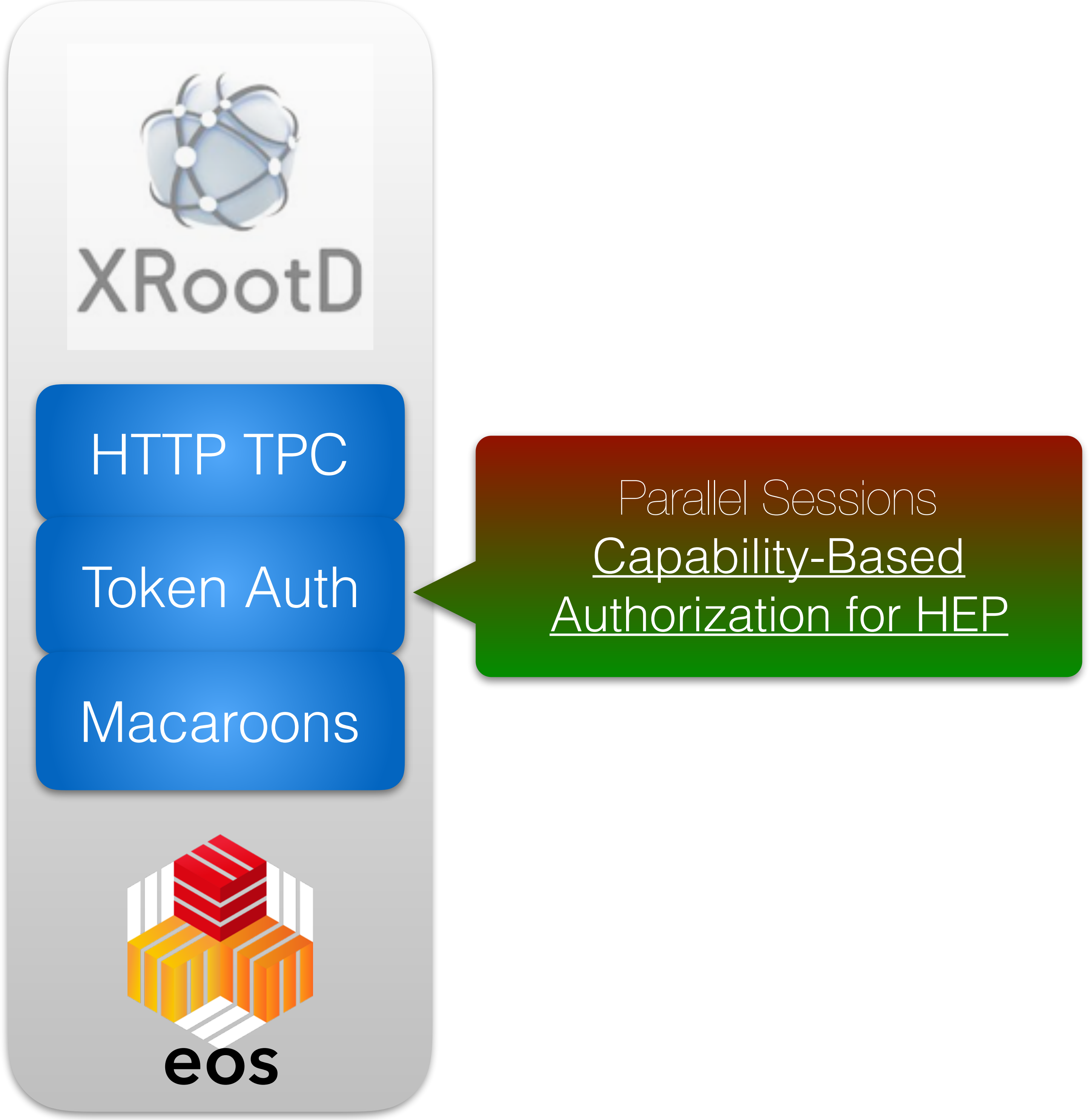
CERN

Storage modules allow extensions and replacement of custom low-level functionality with external solutions

Modular Storage

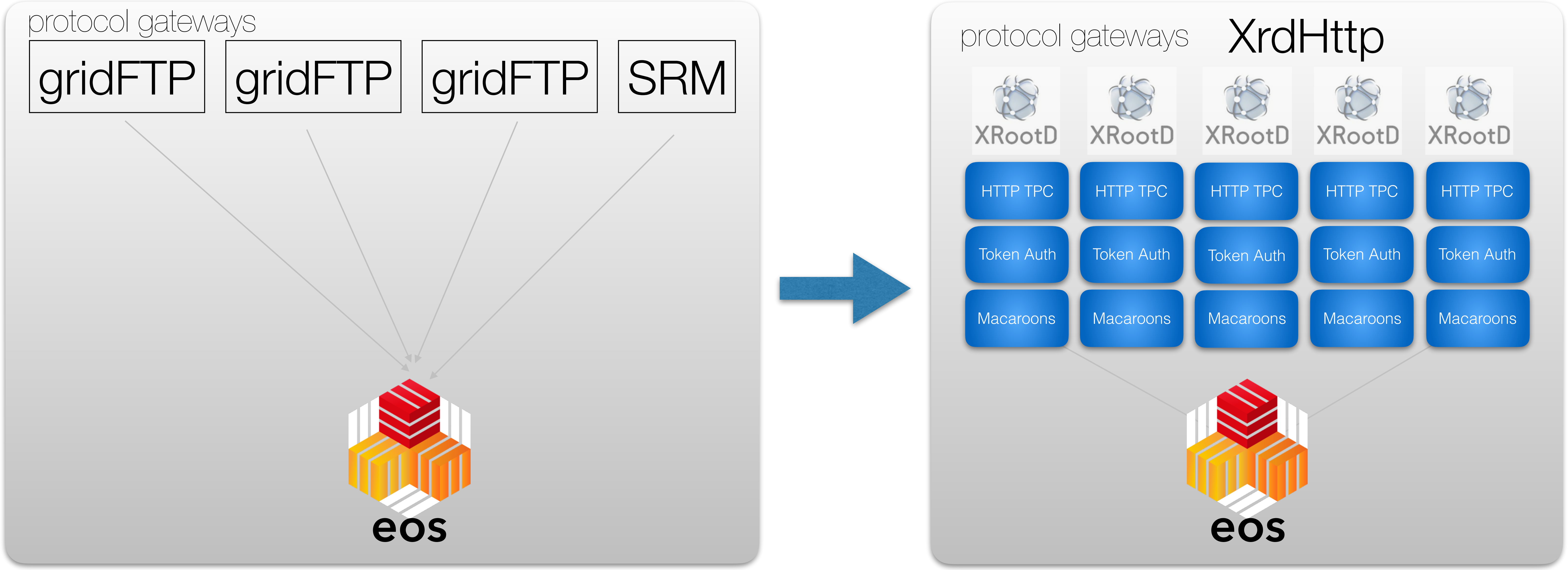
XRootD http ecosystem

http access



HTTP
Protocol Support

the end of FTP/SRM



XRootD is growing a complete set of plug-ins for HTTP enabled storage allowing decommissioning of gridFTP/SRM soon(ish)

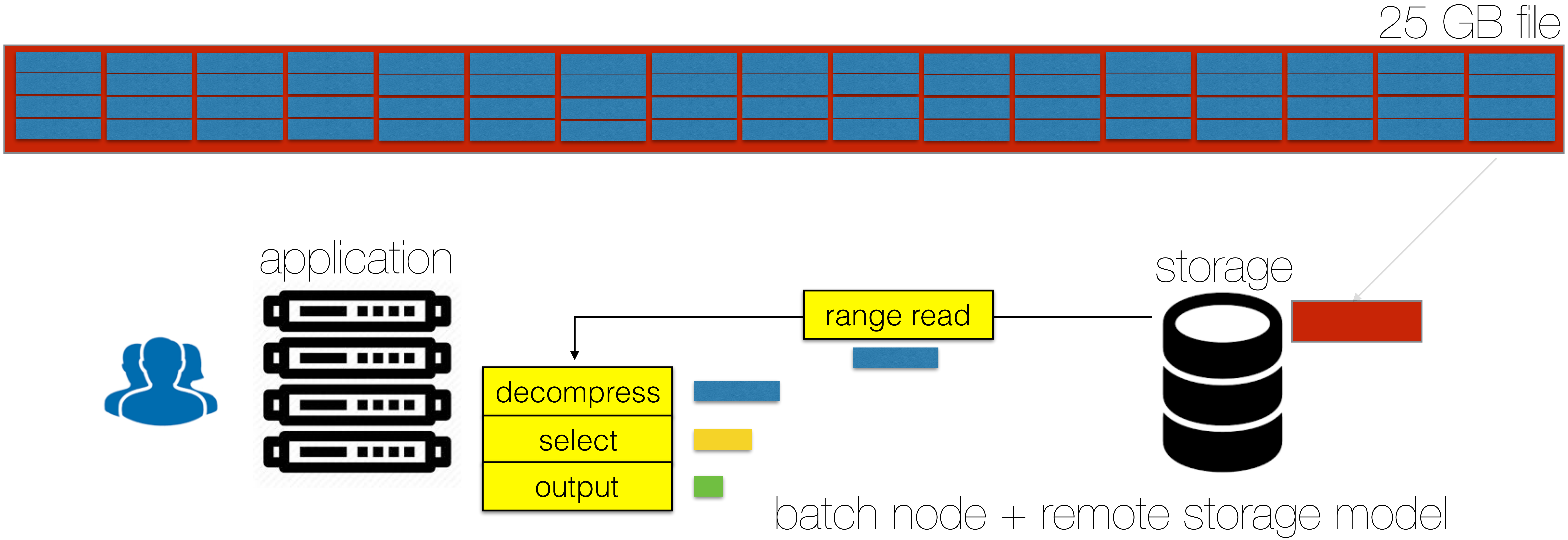
rEvolution
of data processing & storage
using object storage (?)

for a moment
assume

EOS → Exabyte-scale Object Storage

Our conventional file processing model

Parallel processing of a large file by e.g. 10k subtasks is not very scalable/efficient when using POSIX I/O.



Do we need to change this simple model?

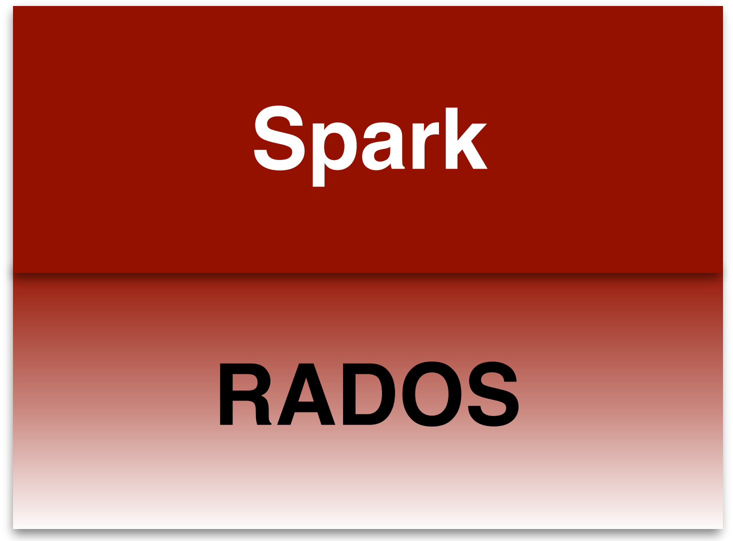
What are others proposing ...

Why Spark on Ceph? (Part 1 of 3)

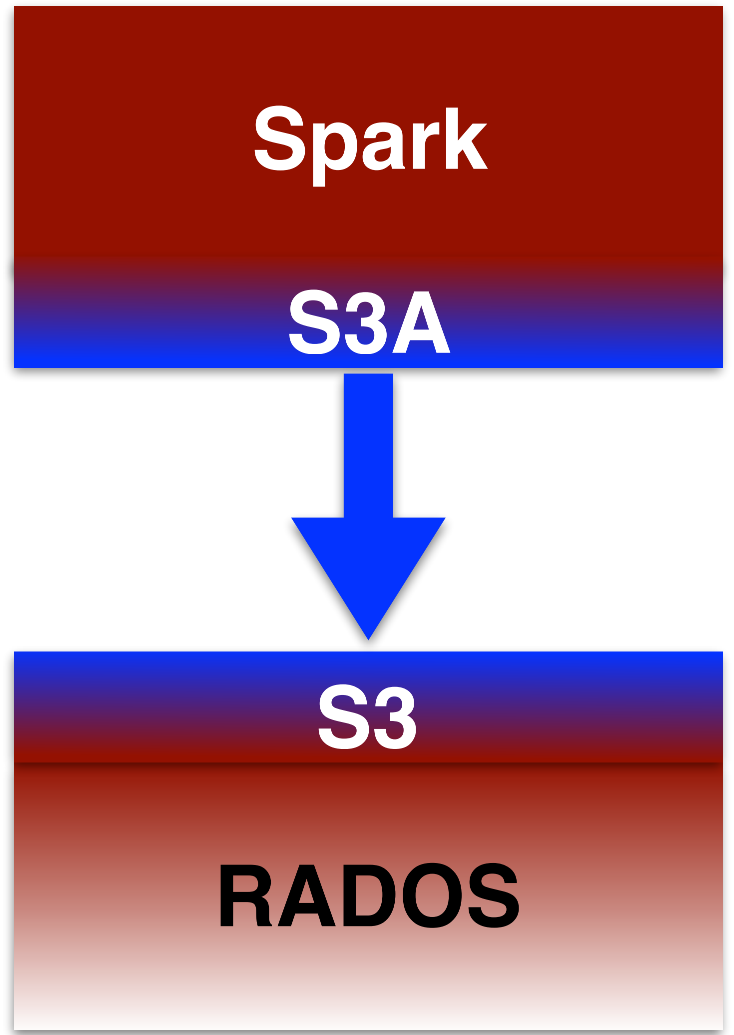
Posted on: June 25, 2018

https://redhatstorage.redhat.com/2018/06/25/why-spark-on-ceph-part-1-of-3/

Sounds HADOOP-like



but **means** only **S3 remote reading**



Conclusion in this article:

Not highest possible performance when **storage and compute** are **separated** but the most flexible model when you have many people sharing infrastructure.

We figured that out already. **That is what we did and do**!

positive+ CEPH S3 buckets can be configured to be index-less removing a scalability limitation [sacrifying listings & accounting]
negative- CEPH S3 for HEP analysis misses multi byte-range request and data flows via gateways. Good news: that could be fixed!

Most people mean S3 when they talk about Object Storage

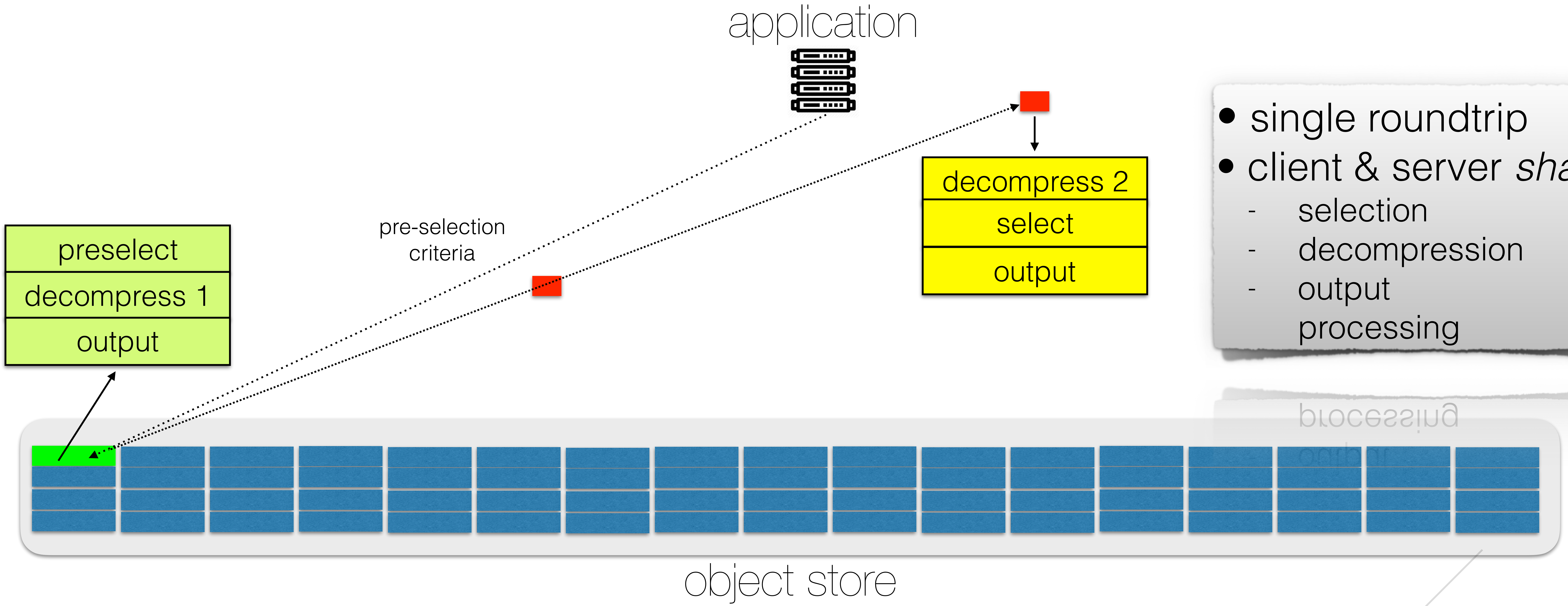
In fact applications know nothing about objects

Object Storage Usage Models

… mainly about Parallel IO

There are many ways to do the same thing with subtle differences in complexity & functionality. Which one is the best? … depends …

Data processing with application object awareness



- single roundtrip
- client & server *share*
  - selection
  - decompression
  - output processing

Allows to move some IO processing inside the object storage non-generic but use-case optimised approach - nice R&D

Summary & Outlook

- EOS has been under **steady evolution** since 8 years.
  - major promoter of XRootD as a framework and remote access protocol in HEP
  - CERN service had overrun **design limitation** in meta-data & data size during 2017 with visible impact
  - this year marks a major architectural change for scalability, availability & usability
- EOS converges towards an **integrative platform** of external storage components and services **for scientific data processing**
  - it leaves flexibility to integrate new ideas & requirements easily e.g. CERNBOX/SWAN/EOS eco-system
  - open to paradigm shift: leverage low-level components and implement high-level storage functionality
- Exabyte-scale Object Storage is an interesting technology to consider for LHC Run3
  - requires a detailed evaluation of the performance/cost model for storage and possible application benefit. Simplest approach is to build storage tiers and hide objects completely from applications. In this case: nothing visible will change for applications!

THANK YOU

QUESTIONS ?

Parallel Sessions
CERNBox: the CERN cloud storage hub

Parallel Sessions
Cloud Storage for data-intensive sciences in science and industry

Parallel Sessions
CERN Tape Archive: From Development to Production Deployment

Parallel Sessions
Sharing server nodes for storage and compute

Parallel Sessions
Facilitating collaborative analysis in SWAN

Poster Sessions
The EU Up to University Project

Parallel Sessions
Scaling the EOS namespace

Parallel Sessions
Providing large-scale disk storage at CERN

Parallel Sessions
Testing of complex, large-scale distributed storage systems

Parallel Sessions
Disk failures in the EOS setup at CERN

Parallel Sessions
Ceph File System for the CERN HPC infrastructure

Parallel Sessions
Capability-Based Authorization for HEP