



EOS on CephFS

friend-or-foe

Andreas-Joachim Peters
Dan van der Ster
CERN IT-ST



Motivation [1]

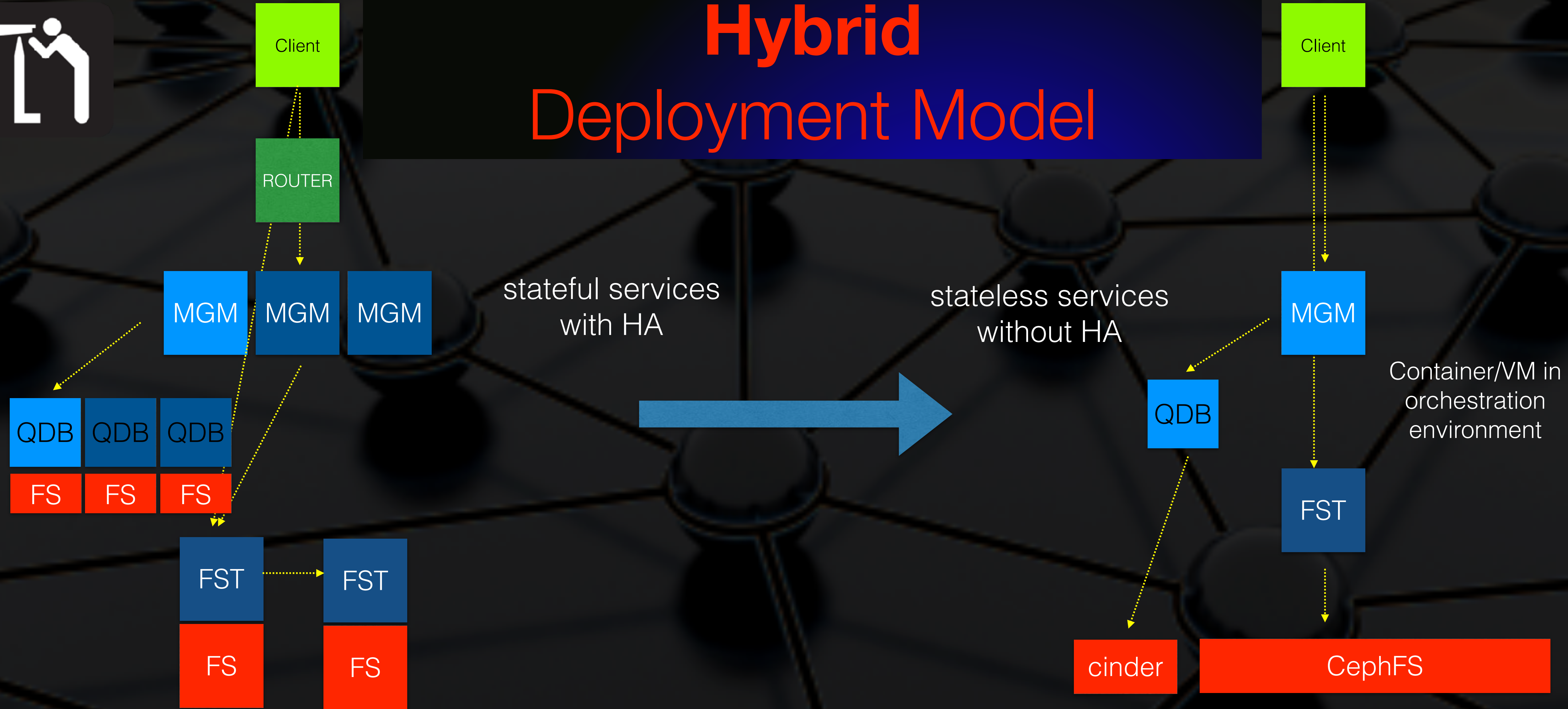
- **CephFS** is part of CEPH, an [Open Source storage platform implementing object storage](#) on a single distributed computing cluster [Wikipedia]
- **CephFS** client is [part of Linux kernel](#), high-performant, very stable and nearly POSIX compliant
- **Ceph** is used **already in many production sites** to provide [block storage](#) to virtual infrastructures
 - if storage hardware is already optimised or tailored to support Ceph, it is a [natural choice](#) to use it also as Big Data storage in the context of HEP - it is inefficient to use it on very old hardware
- **EOS** adds as a [high-level service](#) additional **security** features, extended **quota** and **user management, remote access, token access & TPC** for root:// and https:// protocol, CERNBOX for **sync&share** and CTA for **tape integration**
- **CephFS** adds to EOS the ability to store files of “infinite” [size](#), possibility for [ultra-high-bandwidth](#) access to individual files and a very [robust reliability layer](#) for data

Motivation [2]

- **EOS + CephFS** allows a **complete virtualisation** of the storage environment
 - all services can be deployed in **container environments**
 - most of the **HA** functionality of **EOS** can be delegated to an orchestration service making sure, services are running 'somewhere'
 - HA of data is delegated to **CephFS**, **EOS** can be used to provide an **additional redundancy layer** over computer centres/sites
- **CephFS** can also be used only for a fraction of an **EOS** storage area, which needs to be **tuned for particular IO use cases**

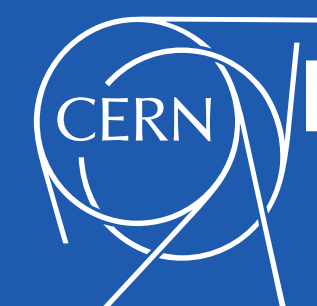
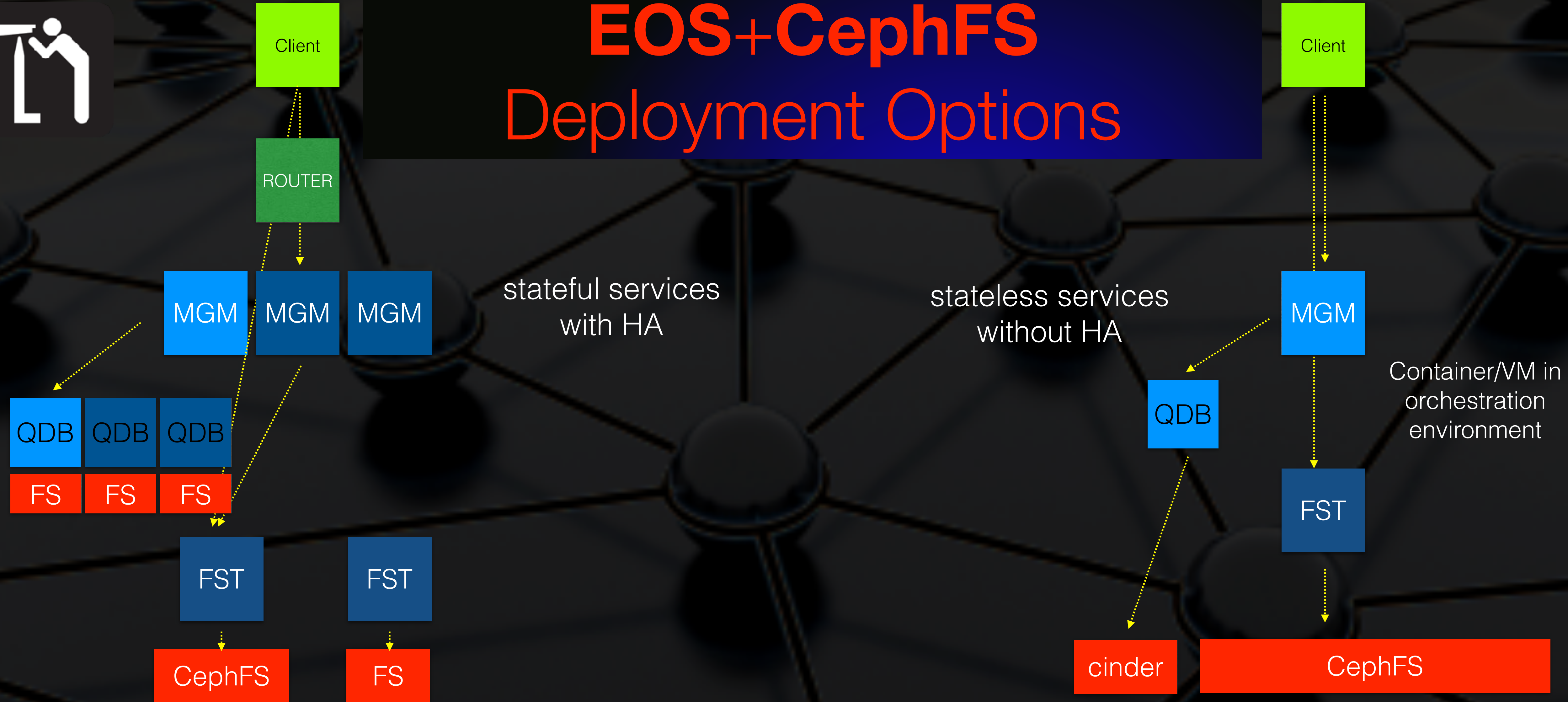


Hybrid Deployment Model





EOS+CephFS Deployment Options



native EOS deployment

physical nodes - local or remote FS

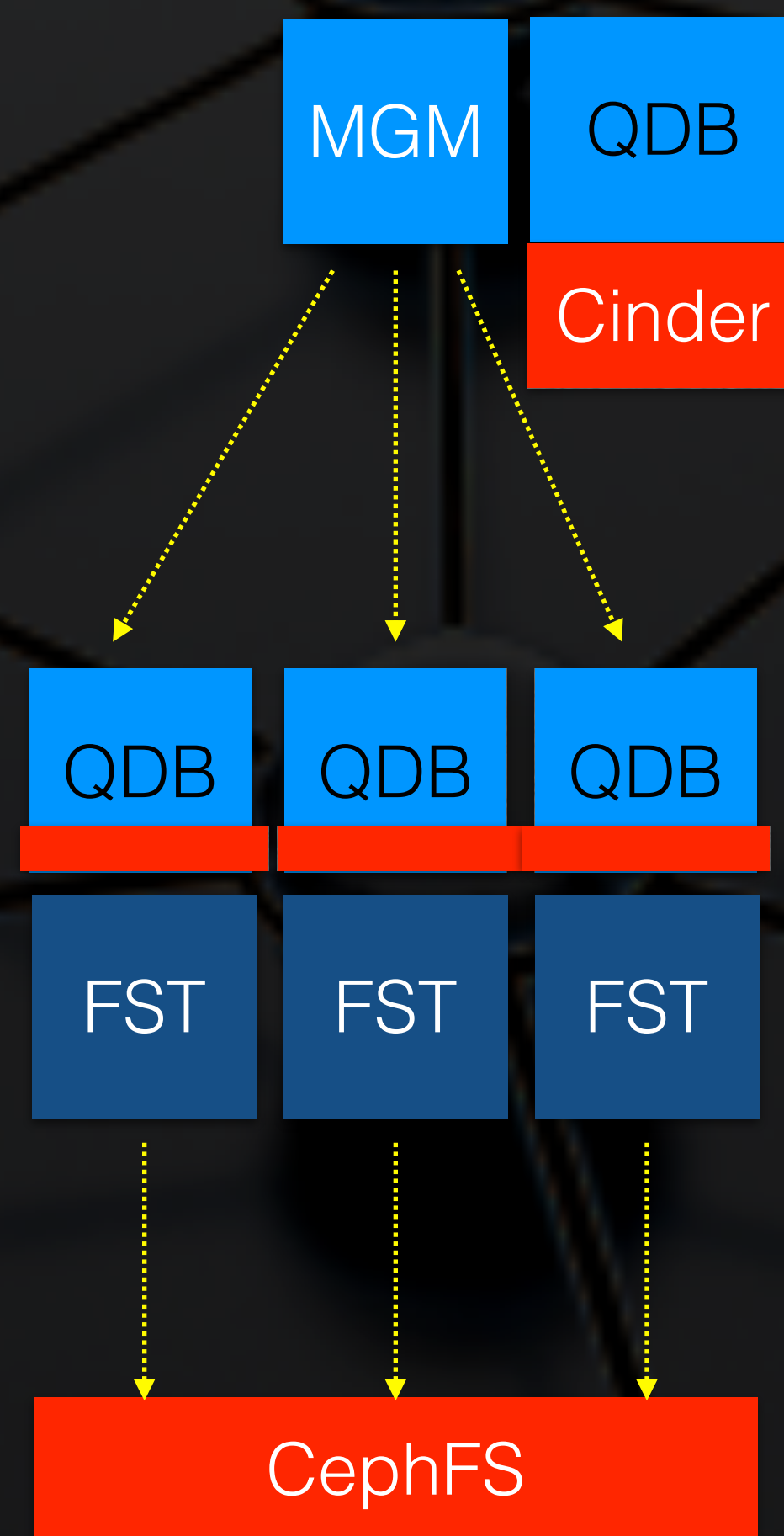
fabric deployment with CephFS

virtual nodes + backend - fabric HA



Deployment Prototype

Hybrid



@OpenStack

pure virtual
deployment
4xVM 20 core - 58GB

Ceph Leviton SSD cluster

3 x **600-750 MB/s** [IO per FST]
creation @**500 Hz**
tested up to **80M files**



2 GB/s storage front-end in four CERN Openstack VMs

Configuration of CephFS in EOS

Native

- usage of CephFS instead of hard disks was done with the [kernel CephFS mount](#) on FST nodes and a directory for an FST filesystem owned by daemon:daemon
- if **IO bw** to CephFS should be **scaled** out, CephFS can be mounted on several FSTs and each FST manages a subdirectory inside CephFS

host	port	id	path	schedgroup	geotag	boot	configstatus	drain	active	health
st-120hd-100gb009.cern.ch	1095	479	/cephfs/eos/data-09/	ceph.0	0513::EC	booted	rw	nodrain	online	OK
st-120hd-100gb010.cern.ch	1095	480	/cephfs/eos/data-10/	ceph.0	0513::EC	booted	rw	nodrain	online	OK
st-120hd-100gb011.cern.ch	1095	481	/cephfs/eos/data-11/							OK
st-120hd-100gb012.cern.ch	1095	482	/cephfs/eos/data-12/							OK
st-120hd-100gb013.cern.ch	1095	483	/cephfs/eos/data-13/							OK
st-120hd-100gb014.cern.ch	1095	484	/cephfs/eos/data-14/							OK
st-120hd-100gb015.cern.ch	1095	485	/cephfs/eos/data-15/							OK
st-120hd-100gb016.cern.ch	1095	486	/cephfs/eos/data-16/							OK

type	name	groupsize	groupmod	N(fs)	N(fs-rw)	sum(usedbytes)
spaceview	ceph	10	10	8	8	2.96 PB

host	port	id	path	schedgroup	geotag	boot	configstatus	drain	active	health
st-120hd-100gb009.cern.ch	1095	1	/data00	default.0	0513::EC	booted	rw	nodrain	online	no mdstat
st-120hd-100gb009.cern.ch	1096	2	/data01	default.1	0513::EC	booted	rw	nodrain	online	no mdstat
st-120hd-100gb009.cern.ch	1097	3	/data02	default.2	0513::EC	booted	rw	nodrain	online	no mdstat
st-120hd-100gb009.cern.ch	1098	4	/data03	default.3	0513::EC	booted	rw	nodrain	online	no mdstat
st-120hd-100gb009.cern.ch	1099	5	/data04	default.4	0513::EC	booted	rw	nodrain	online	no mdstat
st-120hd-100gb009.cern.ch	1100	6	/data05	default.5	0513::EC	booted	rw	nodrain	online	no mdstat
st-120hd-100gb009.cern.ch	1101	7	/data06	default.6	0513::EC	booted	rw	nodrain	online	no mdstat
st-120hd-100gb009.cern.ch	1102	8	/data07	default.7	0513::EC	booted	rw	nodrain	online	no mdstat

Configuration of CephFS in EOS

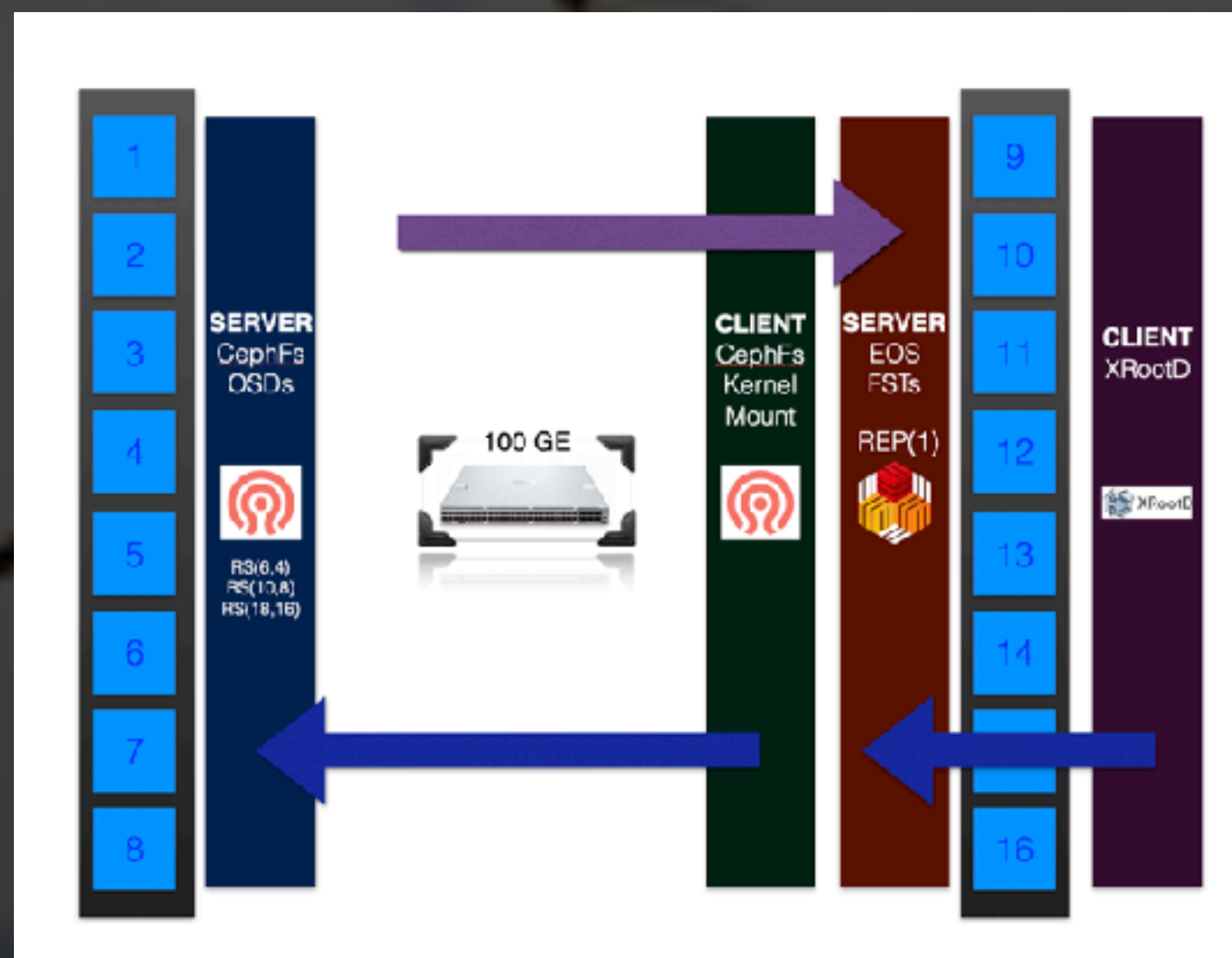
- in the described deployment model each **FST** is a **point of failure**
- **EOS4** supports to transparently **move a filesystem** from one node to another if the backend is shared

```
eos fs mv -force 479 st-120hd-100gb010.cern.ch:1095
```

- currently this functionality is **manually**, it is straight forward to add this as a convenience service to the MGM server
- moves can be done **without any impact for readers** and eosxd clients - **xrdcp** will support in a future version to re-initiate a complete upload if an upload is failed during a transfer instead of retrying the current disk server
- for node maintenance filesystems can be configured to be read-only and the **mv** can be done when the last writer has finished

Experiences with CephFS & EOS

- we have deployed a **6 PB CephFS+EOS** cluster with **100GE** technology and evaluated performance with various **CephFS** erasure coding configurations
- **results** will be published and presented at **vCHEP 2021**
- the **impact** on throughput performance of EOS front-end **is small** if there are no network bottlenecks



Experiences with CephFS & EOS

EOS fsck vs the CephFS MDS

- **MDS** grants caps to CephFS clients so they can read/write/cache/etc
 - each cap requires the inode to be in memory on the MDS, a few kB each
 - Caps are granted on demand and recalled according to MDS memory pressure
 - MDS has a tunable memory target, 4GB by default, allowing around 1M inodes to be cached
- **EOS fsck** needs to stat all files in the FST: roughly equivalent to find /cephfs
- **Problem:**
 - Caps can be granted at up to 30-40kHz per client, but recall is limited by default to 5kHz per client
 - MDS memory usage will quickly exceed the mds_target_memory, going OOM if poorly configured
- **Solution:**
 - `ceph config set mds mds_recall_caps_max 30000`
 - more aggressive caps recall is now the default: <https://github.com/ceph/ceph/pull/38574>

Experiences with CephFS & EOS

CephFS throughput limitations

- each **client limits** its in-flight write op bytes **to 100MiB** by default:
 - removed this artificial limitation by setting `objecter_inflight_op_bytes = 1GiB`


Experiences with CephFS & EOS

CephFS throughput limitations

- in one test write rates dropped from nominal **25GiB/s** to below **3GiB/s**:
- iperf tests look OK, disks are all ~idle. Where is the bottleneck?
- we found one disk (osd.256) with 40x latency from other drives!
- likely caused by a poor SATA connection
- we marked the drive out of the crush map and 25GiB/s returned immediately
- visible in ceph internal perf metrics, but no HEALTH_WARN for this

<https://tracker.ceph.com/issues/49505>

```
[root@eosprojectx-ec ~]# ceph osd perf | sort -n -k3 | tail
 78          50          50
427          50          50
 85          52
227          52
335          53
252          54
 30          57
455          59
186          64
256          2306       2306
[root@eosprojectx-ec ~]#
```



Recommendations

- in such a setup you should use a **CephFS EOS area only via EOS**
- the namespace in CephFS created by FSTs is not attractive and there is no original ownership of files visible in the backend
- in principle FSTs could run on OSDs, however there is a certain **risk of kernel deadlocks** under memory pressure when CephFS is mounted on OSDs
- a possible optimisation for the future could be a **local redirect** to a read-only mount on client side if the data privacy policy allows that
- we have seen a **performance bottleneck** of approx. **6 GiB/s** per 100GE FST and the network layout has to foresee the additional impact of a front-end layer

Summary

- reasons to use or not use **CephFS+EOS** can be manifold
 - **deployment** and tuning of CephFS alone can be challenging or not (it was) - same is true for EOS - once done it is all clear and easy
- the **integration** and **configuration** of CephFS inside EOS is **trivial**
- we can do several small **improvements** to deal better with shared filesystems under FSTs - automatic failover mechanism
 - it could be an option to run EOS without it's own namespace and use it mainly as protocol gateway with real-time configuration options and comfortable user management
 - implement a VFS namespace plug-in
- **experience** so far is quite **positive**
- to be continued ...

EVERYTHING YOU NEED
..... ☕ ☕ ☕ ☕ ☕
TO KNOW ABOUT

EOS Open Storage

 WORKSHOP '21

 LATEST V4.8.35

 Install

Virtual Workshop 1.-5. March 2021

eos.web.cern.ch