Enabling Research and Discovery
SRCC
Stanford University srcc.stanford.edu

A filesystem coming of age: live hardware upgrade practices

LAD22
LUSTRE ADMIN & DEV WORKSHOP
Paris 26-28.09.2022

Stéphane Thiell
Research Storage Lead, Stanford Research Computing

Lustre update

at Stanford Research Computing

Lustre based computer clusters

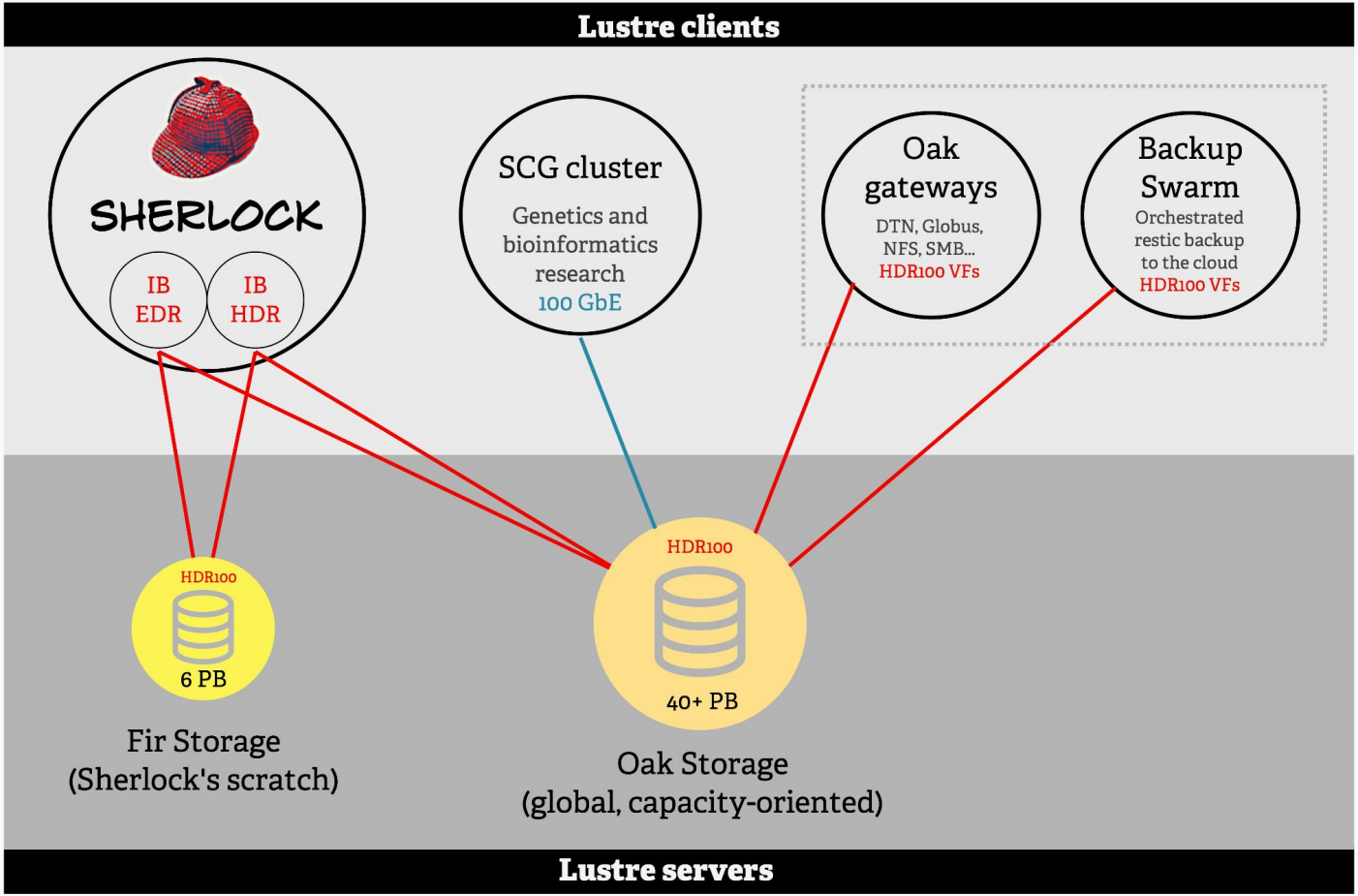
- **Sherlock** 
  - Shared computer cluster for sponsored research (condo model)
  - ~1,500 compute nodes across two IB fabrics (EDR, HDR 200Gb/s)
  - 125+ multi-GPU nodes with a few using LNet Multi-Rail
  - https://www.sherlock.stanford.edu/
- **SCG**
  - Cluster resources for the **G**enetics **B**ioinformatics **S**ervice **C**enter
  - ~80 compute nodes (10/25/40/100 GbE) + 1 SGI UV300
  - https://login.scg.stanford.edu/configuration/

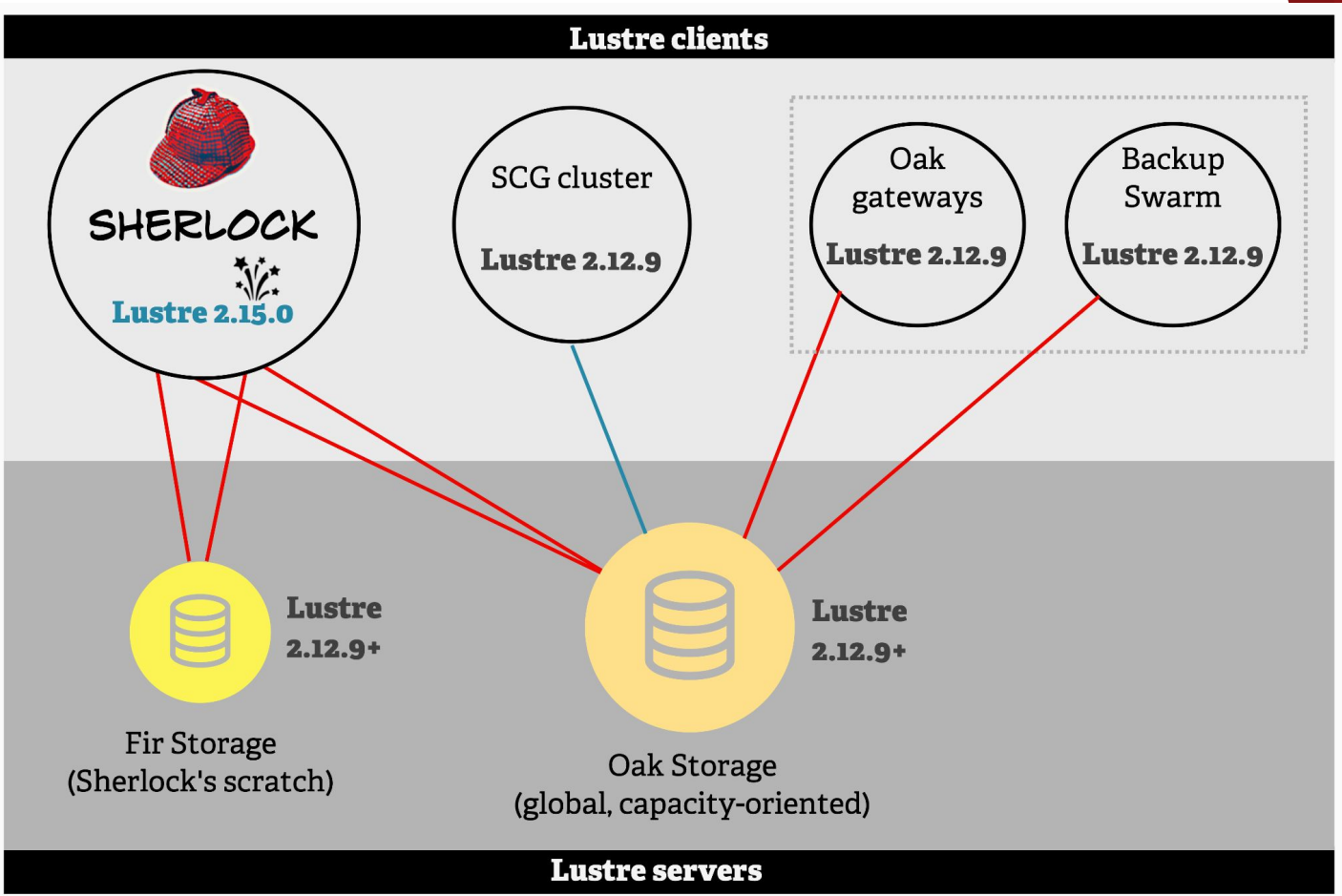
Lustre storage systems

- **Fir**
  - Sherlock's **scratch** filesystem
  - High-performance storage system for **temporary data**
    - intermediate job files, staged datasets, pre-processed data, etc.
    - automatically **purged** based on **data_version** after **90 days**
  - 4 MDS, 16 OSS, mdraid/ldiskfs backend, ~**6 PB** usable
  - Enforcing **project/directory quotas**
  - Deployed in 2018: all OSS are currently **HDD based**

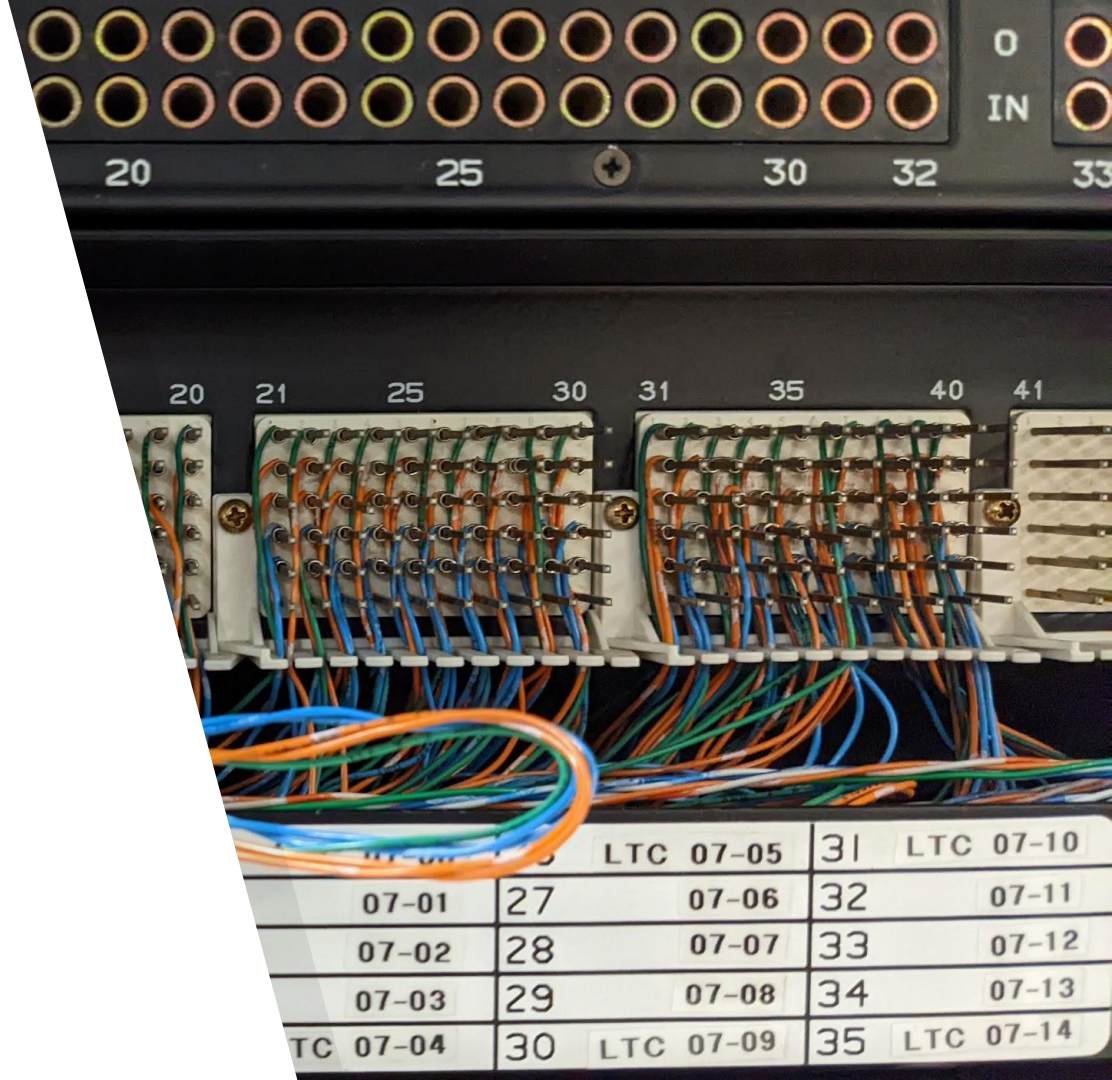
Lustre storage systems (cont'd)

- **Oak**
  - Global, **capacity**-oriented Lustre filesystem
  - **Large I/O cells**, each of up to 11 PB usable
  - 4 MDS, 14 OSS, mdraid/ldiskfs backend, **40+ PB** global usable
  - Fixed inodes / volume ratio: **150,000 inodes per TB**
  - QoS: **NRS TBF per GID** enabled on OSS
  - Offered to researchers as a **service** (for a fee) since 2017


Lustre clients
SHERLOCK
IB EDR
IB HDR
SCG cluster
Genetics and bioinformatics research
100 GbE
Oak gateways
DTN, Globus, NFS, SMB...
HDR100 VFs
Backup Swarm
Orchestrated restic backup to the cloud
HDR100 VFs
HDR100
6 PB
Fir Storage
(Sherlock's scratch)
HDR100
40+ PB
Oak Storage
(global, capacity-oriented)
Lustre servers


Lustre clients
SHERLOCK
Lustre 2.15.0
SCG cluster
Lustre 2.12.9
Oak gateways
Lustre 2.12.9
Backup Swarm
Lustre 2.12.9
Lustre 2.12.9+
Fir Storage
(Sherlock's scratch)
Lustre 2.12.9+
Oak Storage
(global, capacity-oriented)
Lustre servers

Oak's hardware lifecycle



Oak's hardware lifecycle

- Add new storage
  - When needed to accommodate demand and replace old storage
- Remove old storage
  - In accordance with the lifetime / end of warranty of the hardware

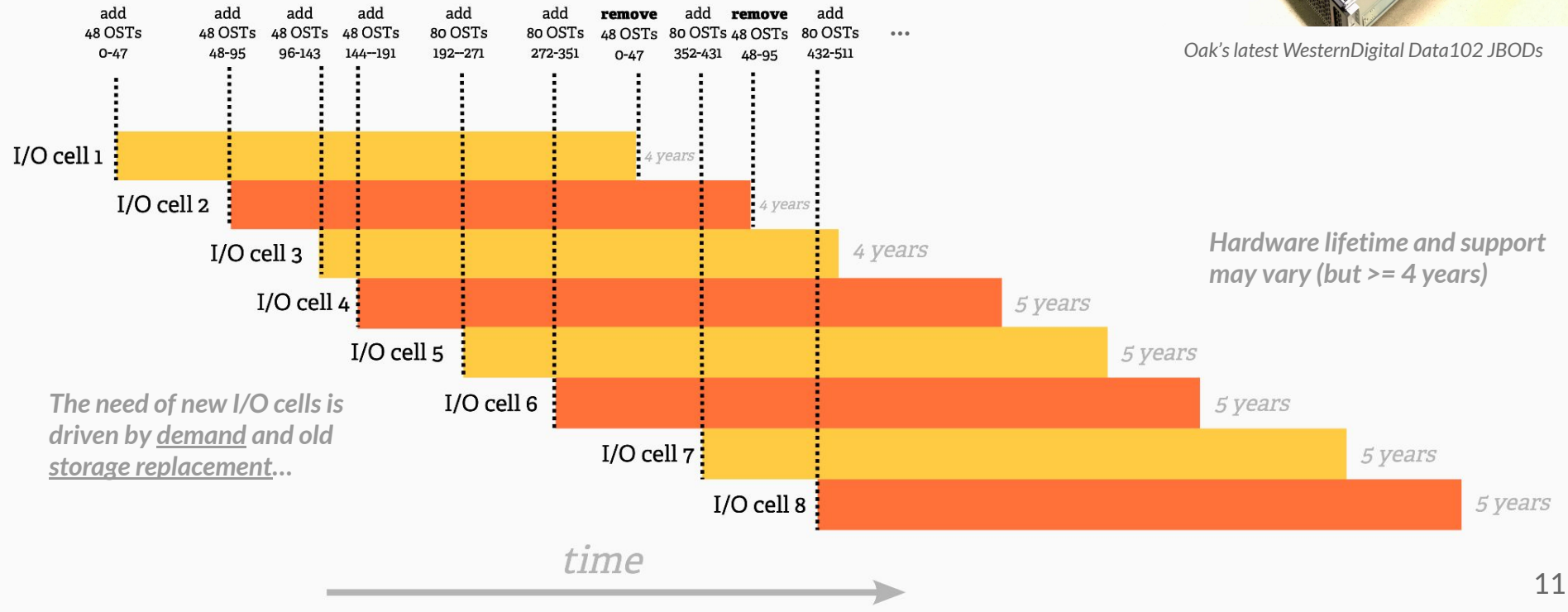
Oak's hardware lifecycle **methods**

- Adding new storage
  - Add new OSTs (always by using increasing OST indexes)
- Removing old storage
  - Set the OSTs in "**no create**" mode well in advance if possible
    `lctl set_param -P osp.*osc_name*.max_create_count=0`
  - Empty the OSTs by **migrating data** (with lfs find + lfs_migrate)
  - **Deactivate** the empty OSTs
    `lctl conf_param *ost_name*.osc.active=0`
  - **Permanently remove** the OSTs from the filesystem with lctl del_ost
    - Removed (lower) OST indexes are not used again on Oak

See also:
Lustre Manual 14.9.3. Removing an OST from the File System

Oak storage lifecycle overview

Oak's latest WesternDigital Data102 JBODs



Hardware lifetime and support may vary (but >= 4 years)

The need of new I/O cells is driven by demand and old storage replacement…

Permanently removing OSTs?

- lctl **del_ost** --target *fsname*-OST*xxxx*
  - Allow permanent OST removal **without** rewriting the configuration (writeconf) → **no down time needed!**
  - Development done in collaboration with **Andreas Dilger** (Whamcloud), thanks!
    - Jira ticket: **LU-7668** [Landed in **Lustre 2.16**]
  - Patch for **LU-15000** (llog: read canceled records in llog_backup) required when **backporting to 2.12 or 2.15**
    - Thanks to **Etienne Aujames** (DDN) for fixing this bug!
  - lctl del_ost will be part of a paper submitted to HPCSYSPROS22
    - *Overcoming HPC System Management Challenges: An Open Source Approach*

sighpc

Fir’s flash upgrade

Fir: upgrade goals

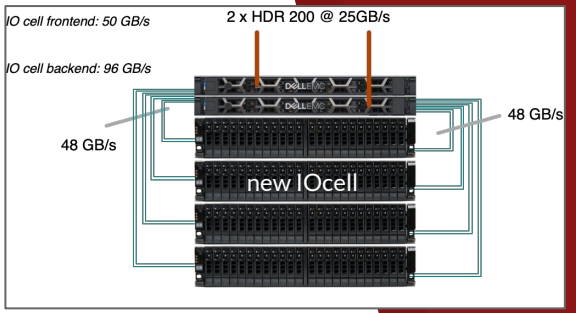
- #1: increase IOPS
- #2: increase bandwidth
- #3: increase volume

How? **Replace rotating disks with (larger) SSDs**

- replace existing JBODs with JBOFs (w/SSD drives)
- replace existing servers (bandwidth limited)

Fir: in-place flash upgrade

- Replace existing 60-HDD JBODs with 24-SSD JBOFs
- Each JBOD replaced with 2 JBOF (960 slots → 768)
  - Seagate Exos E 2u24 platform
  - need larger SSDs to maintain volume
    - Nytro 3332 15.36TB SAS SSD
    - Use mdraid with RAID layout 10+2 (vs. 8+2 with HDD today)
- Replace each server with one, faster server



Fir: *The way*

- Principle of **minimal disruption**
  - Users have data in /scratch
  - They don't have time to transfer it to another filesystem, or to modify their scripts and workflows
  - So, we bring new performance in by upgrading the existing filesystem in place.
- **Zero downtime, zero disruption, twice the performance.**

Fir: step by step **upgrade**

1. **empty** an IO cell (using the **purge**!)
   set the IO cell as "**no create**" and wait for 90 days, most files will have been purged, then migrate remaining data to other IO cells with **lfs_migrate**
2. **retire** emptied IO cell from the filesystem
   - with **lctl del_ost** of course!
3. **replace** retired servers and JBODs with new hardware (same U space)
4. **add** new IO cell to the filesystem
5. rinse and repeat

Fir: upgrade one IO cell at a time

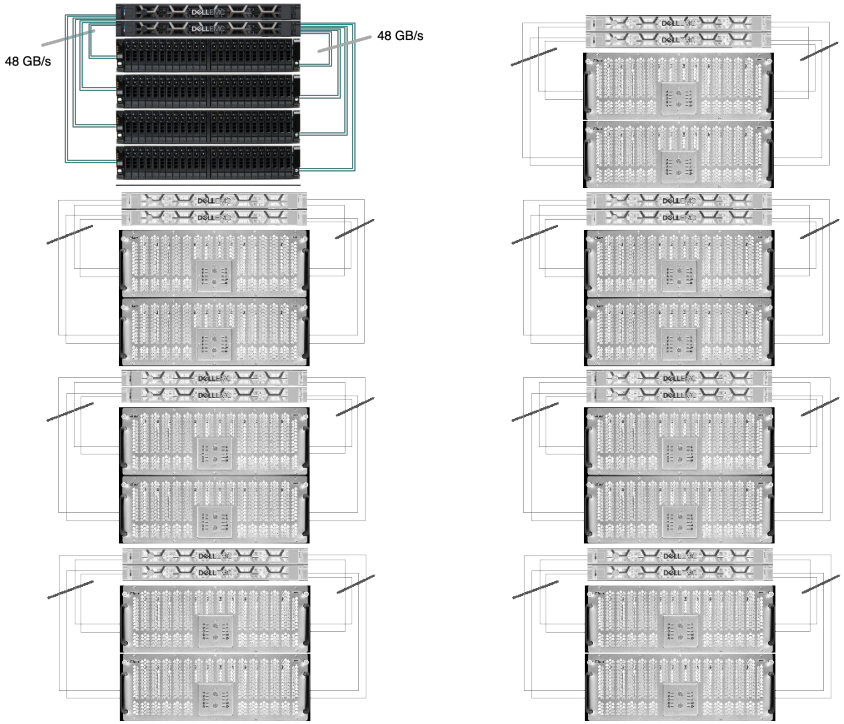
No downtime, no disruption for the users

Performance will increase gradually
as IO cells are replaced over FY23-24

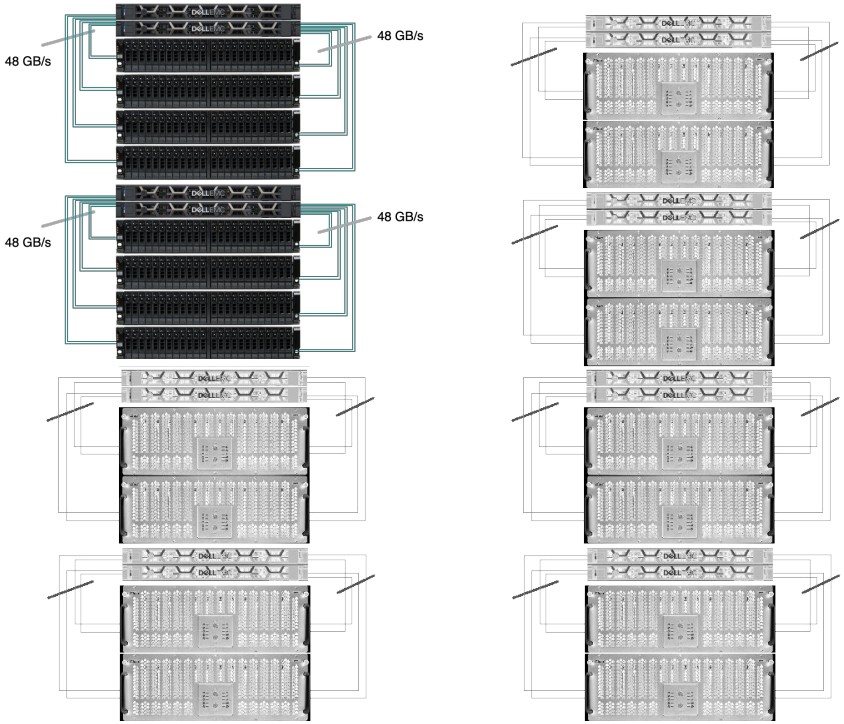
Performance benefits (ultimately)

- usable volume **x1.6**
- frontend bandwidth **x2**
- backend bandwidth **x6**
- IOPS (on OSS cache miss) **x1000** (estim.)

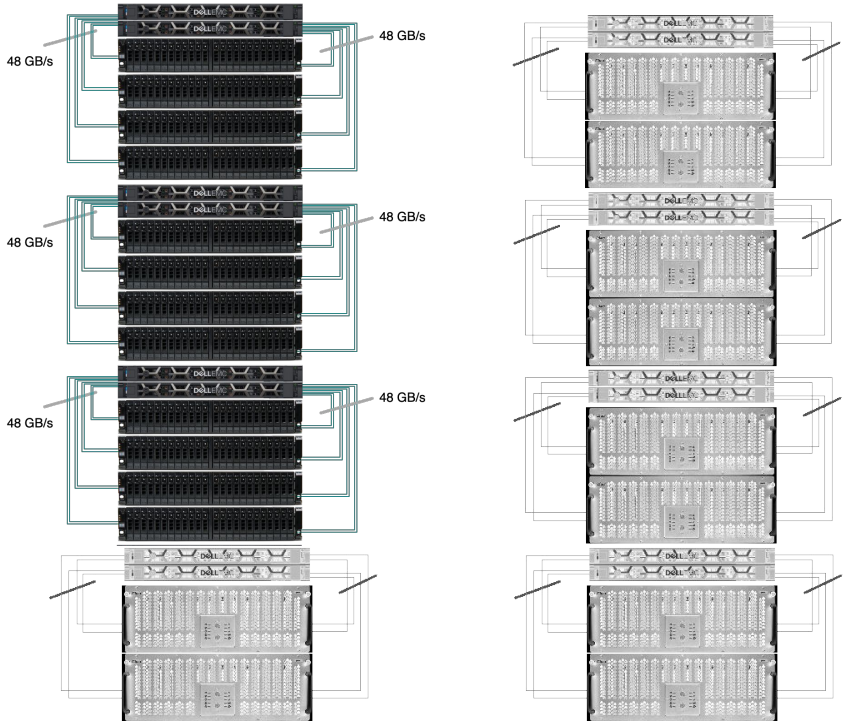
Fir: **in-place**, one IO cell at a time **upgrade**



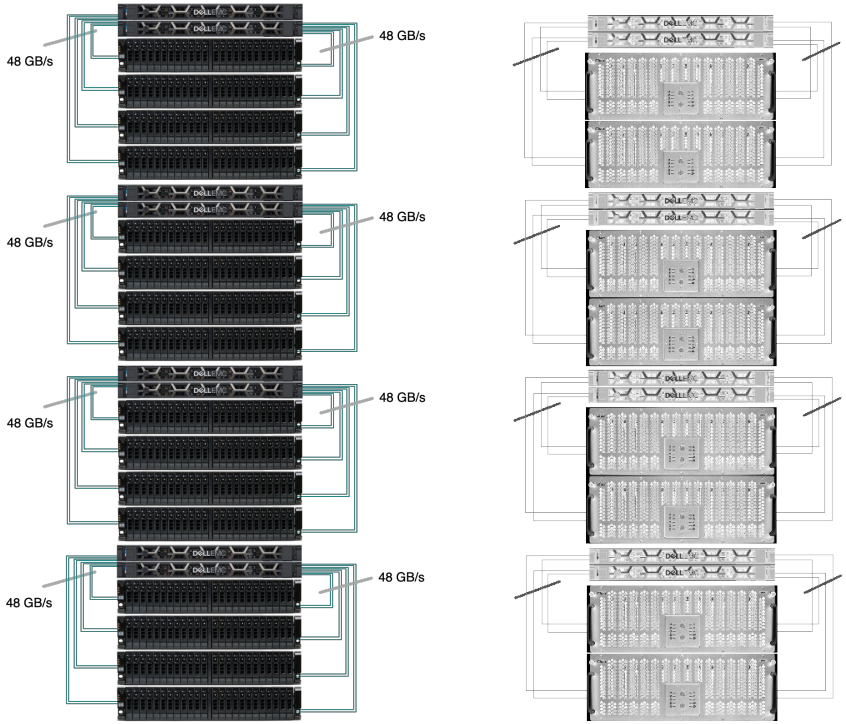
Fir: **in-place**, one IO cell at a time **upgrade**



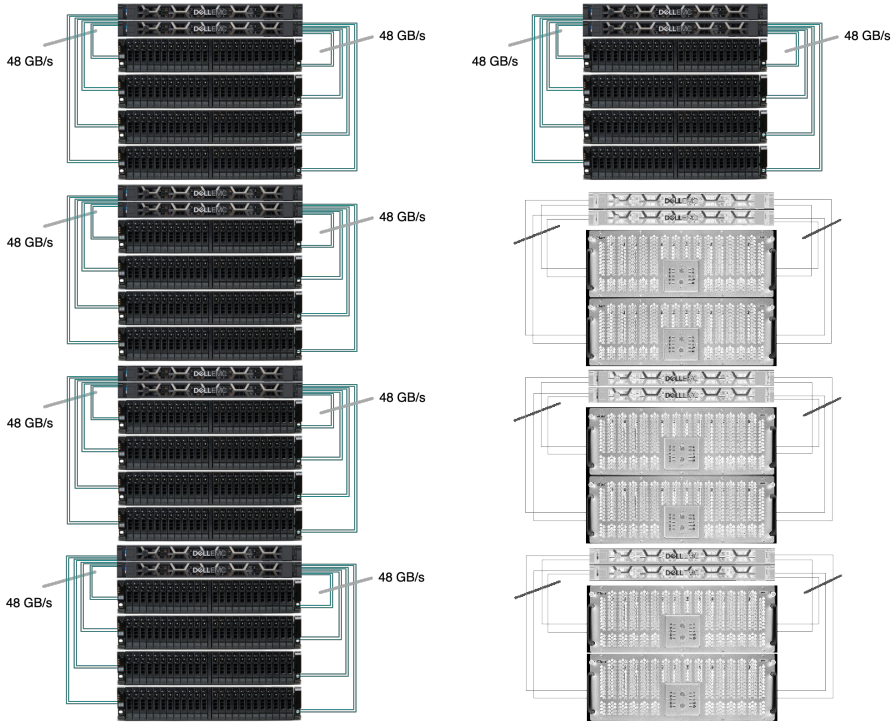
Fir: **in-place**, one IO cell at a time **upgrade**



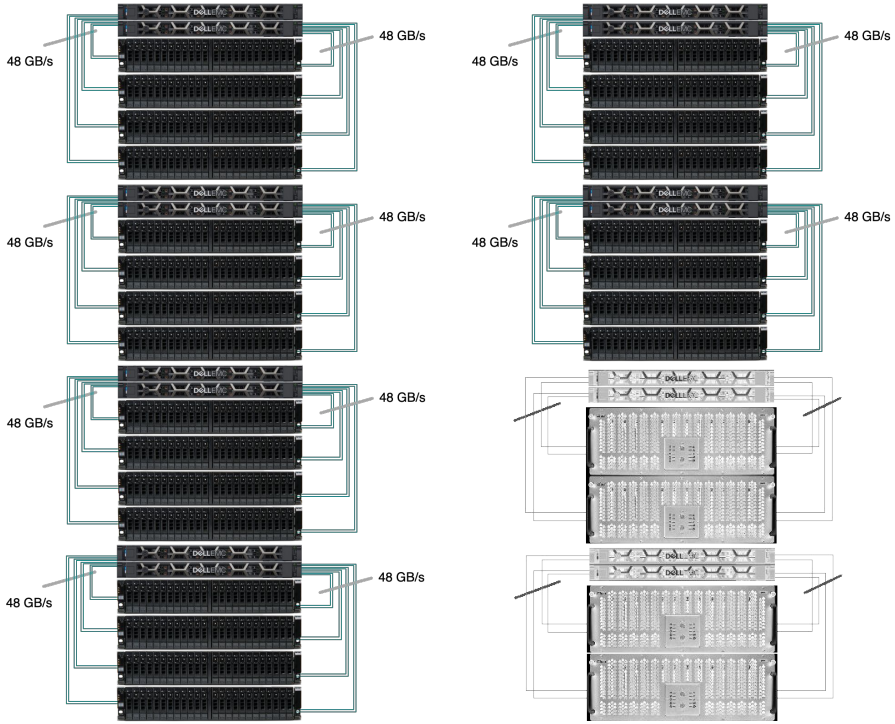
Fir: **in-place**, one IO cell at a time **upgrade**



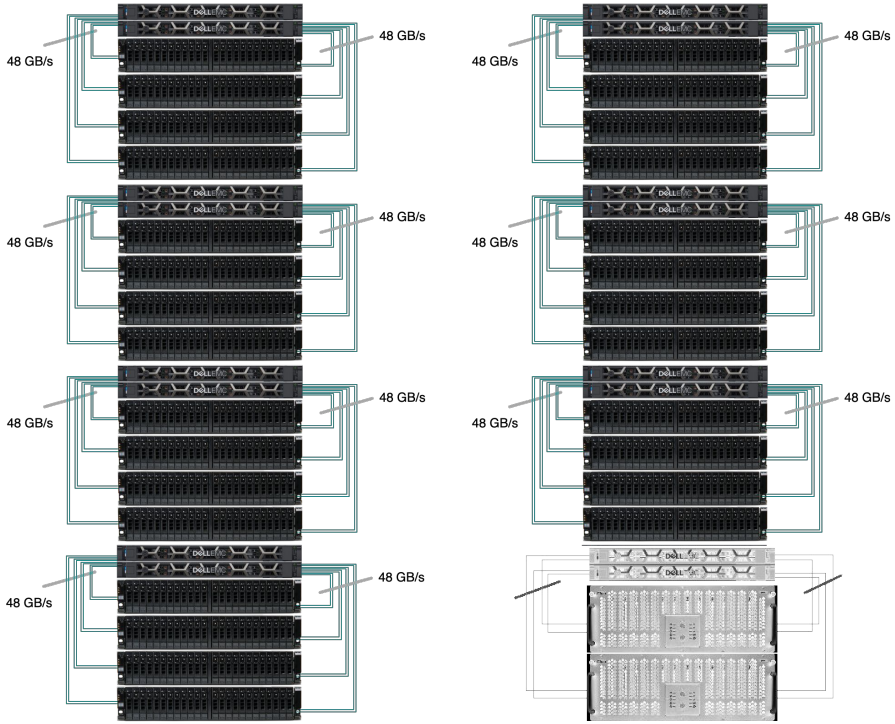
Fir: **in-place**, one IO cell at a time **upgrade**



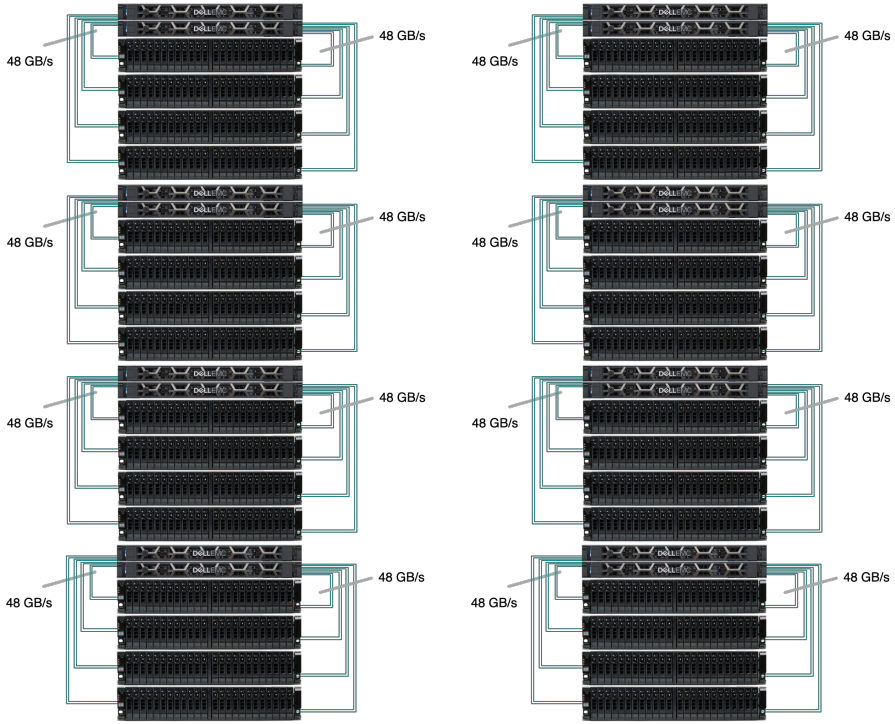
Fir: **in-place**, one IO cell at a time **upgrade**



Fir: **in-place**, one IO cell at a time **upgrade**



Fir: **in-place**, one IO cell at a time **upgrade**



Fir 2.0 specs (ultimately full-flash)

Hardware

without metadata and management

- 2 IO racks
- 8 IO cells
  - 2 servers ea.
  - 4 JBOFs ea.
- 768 15.36TB SSDs

Usable space: **9.8 PB**

IO cell bandwidth

- 96GB/s backend SSD/SAS
- 25GB/s frontend Infiniband

Total bandwidth

- 768GB/s backend SSD/SAS
- 400GB/s frontend Infiniband

Stanford

THANKS!

Any question?

sthiell@stanford.edu

GitHub: https://github.com/stanford-rc/

https://facts.stanford.edu/

Stanford
University