Pirogov Russian National Research Medical University

Building a computer cluster for bioinformatics and biomedical research from zero to production

V. Kotliar, D. Bolitin, D. Syrko, Pirogov Russian National Research Medical University, RU-117997, Moscow, Russia

E-mail: kotliar_vv@rsmu.ru

To build a computer cluster for **bioinformatics** and **biomedical research** is a very complex task. Such cluster has to seamlessly combine **different types of the software stack** which is used for computations and it should provide the easiest way for organizing **complex workflows** for scientific research. On the other side it has to be **as simple as possible for usage** to allow researchers with no or basic knowledge in information technology to perform their tasks.

This work present one of the possible architecture for a such system and a cluster software stack which could be used to build and operate it using a computer cluster of the Institute of Translational Medicine from Pirogov Russian National Research Medical University as an example.

Introduction RSMU

- 8300 STUDENTS OF HIGH MEDICAL SCHOOL
- 11500 POSTGRADUATES AND PROFESSIONAL STUDENTS
- 700 INTERNATIONAL STUDENTS
- 12000 PARTICIPANTS OF EVENTS FOR PROSPECT

Pirogov Russian National Research Medical University is one of the oldest medical higher school in Russia. In 2016 the University celebrated its 110th anniversary. The School of Biomedicine opened at the University in 1963. Today the University is a complex of educational, scientific and medical divisions and centers, offering educational programs of all levels in clinical medicine, biomedicine, psychology, social work and pharmacy.

Faculty of Biomedicine provides educational programs of Master degree ("Bioinformatics") (department of bioinformatics)

Introduction ITM

The Institute focuses on **applied** medical and biomedical research and its **early implementation in practice**. New means of rehabilitation, diagnostic kits, and means of treatment from the idea to practical implementation are the tasks of the Institute. It works in close cooperation with research departments of the Russian Academy of Sciences, pharmaceutical and biotechnological companies and clinical centers.

The institute includes the following divisions:
Analytical Center in Biomedicine
Laboratory of Medical Genomics
Laboratory of Microbiology and Biological Safety
Laboratory of Molecular Pharmacology
Laboratory of Redox Regulation
Center of Genomic Technologies
Laboratory of Experimental Oncology
Laboratory of Electrophysiology
Laboratory of Biological Testing
Laboratory of Molecular Oncology
Laboratory of Chemistry of Natural Compounds
Department of Cell Technology and Regenerative Medicine
Department of Medical Biophysics
Department of Medical Nanobiotechnology
Department of Medical Chemistry and Toxicology
Department of Molecular Technologies
Department of Neurocomputer Interfaces
Department of Regenerative Medicine

Building a cluster

Tasks what we really went through :

1. Describe requirements
   - Access
   - Interfaces for computing
   - Data usage
2. Design architecture
   - Software stack
   - Hardware components
3. Setup a pre-cluster
   - Building, Engineering (power + cooling + fire alarm and fire fighting, racks + PDU, UPS, Access control + CCTV), Networking
   - Hardware install
   - Commissioning installed pre-cluster environment
     - Stress tests for compute: memory, CPU, GPU – cooling, power setup, hardware
     - Stress tests for storages: memory, CPU, network, disks
4. Cluster setup
   - Install core HA-cluster
   - Install base network and cluster services
   - Install storage system
   - Install compute system
   - Install frontend nodes
   - Testing and performance measurements

Building a cluster: base software stack

Requrements:

1. Access
   - Jupyter Hub (web)
   - Rstudio (web)
   - Ssh access (with X11 forwading)
2. Sessions
   - Could be long but limited by time
   - Have access to BIG CPU/GPU/RAM
   - Resource share managed among many users
3. Storage
   - Strict homes and projects access
   - Projects share based on Linux groups (posix+acl)
   - Strict quota checking
   - Auto snapshots as much as possible
   - Auto mounts for each volume, no idle connections
   - Compression enable

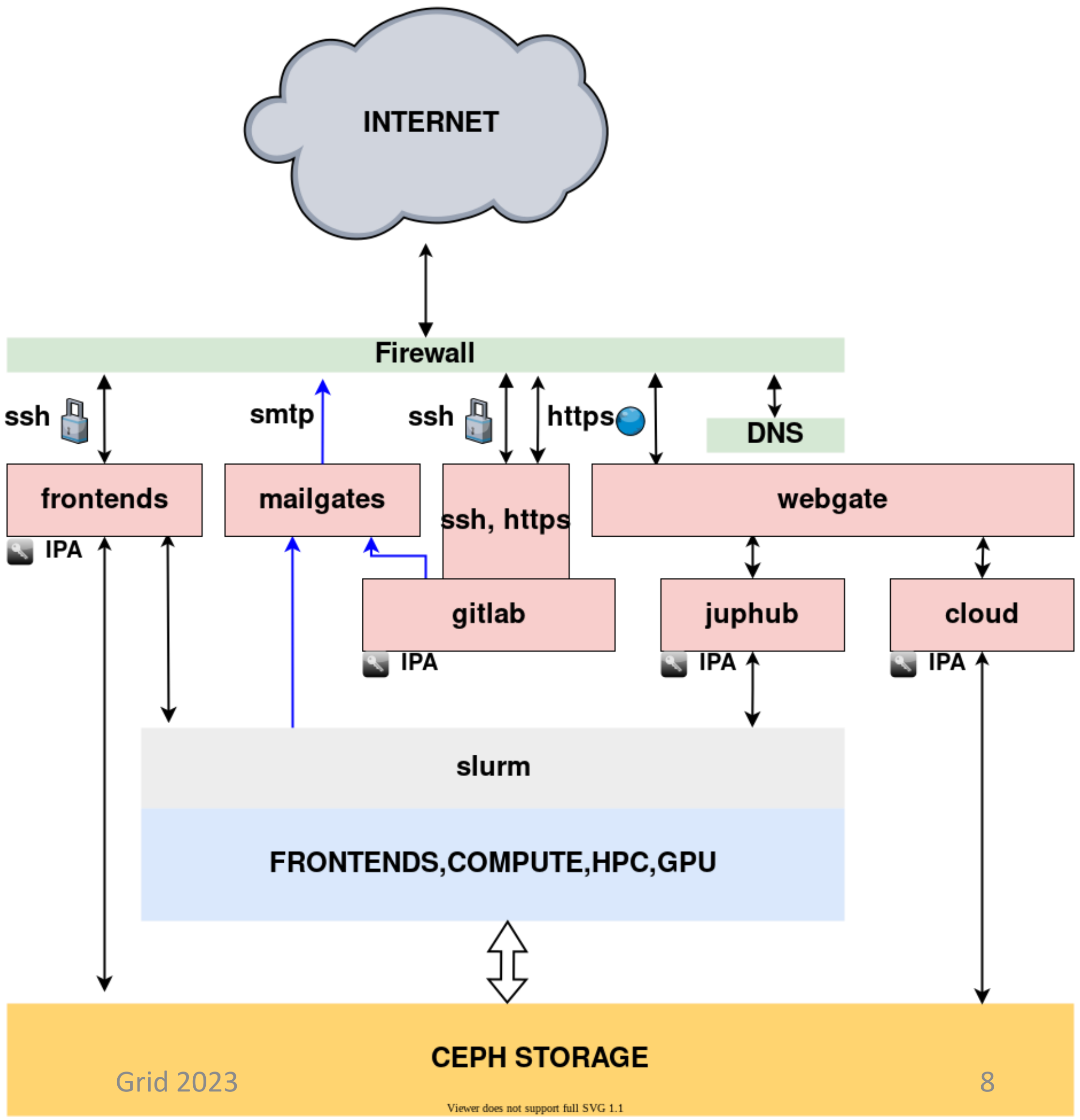
Building a cluster: additional software stack

Requrements:

1. Compute
   - Containers support
     - OCI
     - Docker
     - Nvidia GPU
     - Registry + CI
   - Python environments support
   - R support (multi versions in mind)
   - Flow tools support
2. Collaborative storage
   - Exchange file with external labs
   - Like google docs functional
3. Storage archive
   - Copy archive data to tapes
   - Backup critical data

Building a cluster: architecture

- Base OS Ubuntu 20.04 LTS
- CephFS with sub volumes groups
- 4 node types
- Self-build slurm with containers, gpu, mpi
- Jupyter hub with slurm spawners
- Let’s encrypt for SSL, ssh keys only



Building a cluster: hardware

Compute nodes(**920CPU 8GPU**)
- 8 compute 2xCLX 4210R 2P 10C/20T 2.4G, 384GB RAM, 320CPU
- 2 HPC 2x6258R (2.70 GHz) 1536GB RAM, 224CPU
- 3 HPC 2x6240 (2.60 GHz) 1536GB RAM, 216CPU
- 4 GPU 2x4210R, 10 cores, 13.75M Cache, 2.40 GHz, 384GB RAM, 8GPU (2x2x2x2) NVIDIA Tesla V100 32GB CoWoS HBM2

Storage nodes
- RAW объем **3168TB** (22x12x12)
- 4 x cephfs mds servers (ceph-osd+ceph-mds) 2 active + 2 standby
- 14 x ceph storages (ceph-osd)
- 5 x ceph mons (ceph-osd+ceph-mon)

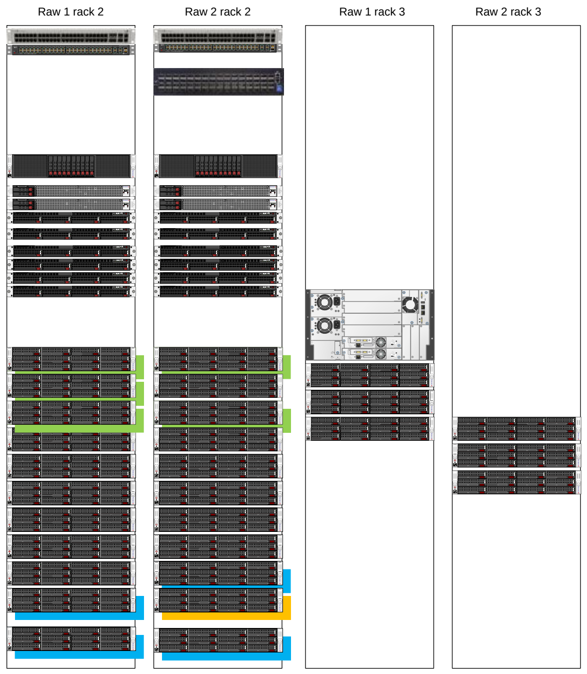
Tape storage system HPE
- MSL6480 for 80 tapes
- Cartridges size without compression **948TB** (79*12)
- Two tape drives LTO8 SAS3
- Disks cache 92TB (two system by 45TB)
- Network 10Gb/s

Frontends
- 2 2xCLX 4210R 2P 10C/20T 2.4G, 384GB RAM, 80CPU

Core servers
- 2 2xCLX 4210R 2P 10C/20T 2.4G, 384GB RAM - direct connect (100Gb/s)

Network equipment
- Data **100Гбс** (NVIDIA Mellanox MSN4600-CS2FC Spectrum-3 Based 100GbE 2U Open Ethernet Switch with Cumulus Linux 64 QSFP28)
- Generic 2x10Гбс

Building a cluster: infrastructure

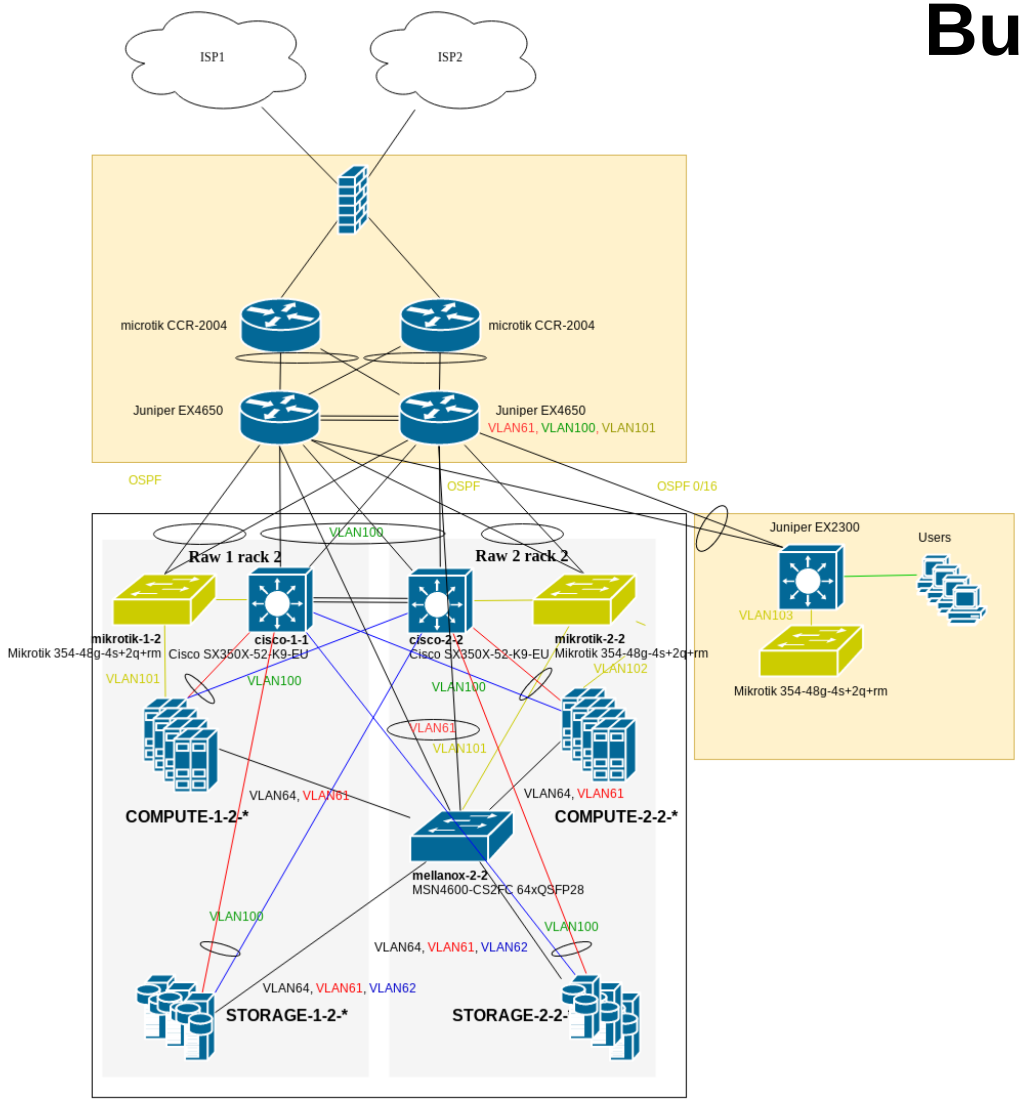


Racks
- 7kW per rack (84kW in total)
- Up for 12 by project (4 used)

Power
- One Input
- One UPS for 220kVa/176 kW
- Two PDUs per rack 32A

Cooling
- Two system Tecnair UPA 662 (2x30,75kW each)
- Total cooling capacity 123 kW (3+1 Redundancy)

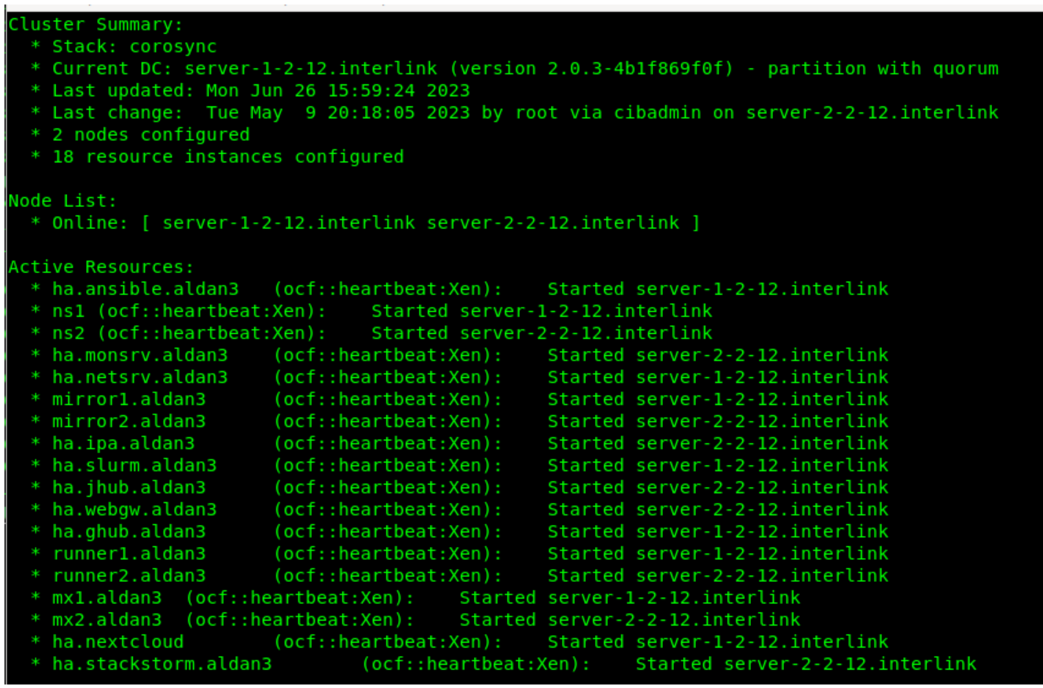
Building a cluster: network



- IPv4 based
- Port forwarding for a few external services
- Cluster has isolated and autonomous network
- VLAN based isolation for access, data and storage segments on switch and on Linux level
- TAG(LAG, ML-LAG) LACP where possible

Cluster setup: core services on HA cluster

- Classical two node setup: pacemaker + corosync + DRBD + XEN virtualization
- Some services on two nodes (service level of HA)
- Some services on one node with DRBD(master/slave)+XEN live migration
- Base core services
  - netsrv, ansible, monsrv
  - ns, mx, mirrors, webgw
  - stackstorm
- Cluster core services:
  - ipa, slurm,
  - jhub
- Other cluster services:
  - ghub, runners
  - nextcloud



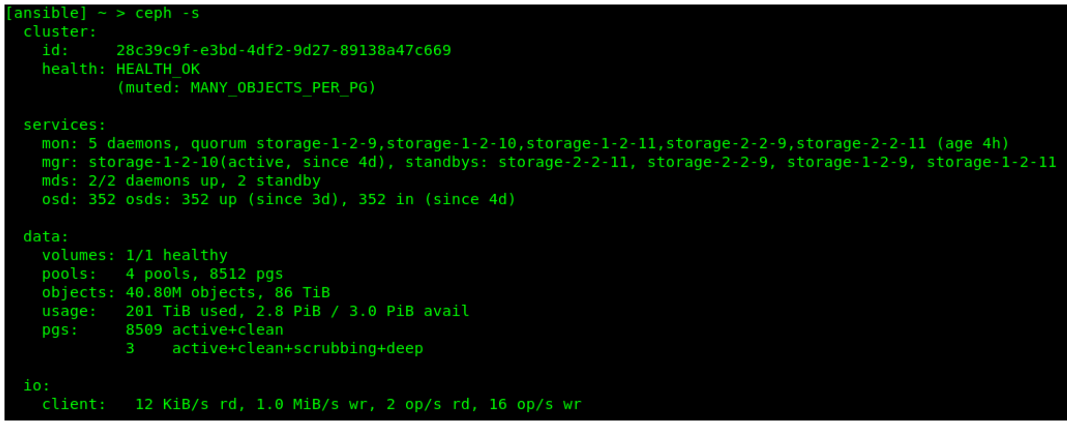
```
Cluster Summary:
  * Stack: corosync
  * Current DC: server-1-2-12.interlink (version 2.0.3-4b1f869f0f) - partition with quorum
  * Last updated: Mon Jun 26 15:59:24 2023
  * Last change:  Tue May  9 20:18:05 2023 by root via cibadmin on server-2-2-12.interlink
  * 2 nodes configured
  * 18 resource instances configured

Node List:
  * Online: [ server-1-2-12.interlink server-2-2-12.interlink ]

Active Resources:
  * ha.ansible.aldan3   (ocf::heartbeat:Xen):    Started server-1-2-12.interlink
  * ns1 (ocf::heartbeat:Xen):    Started server-1-2-12.interlink
  * ns2 (ocf::heartbeat:Xen):    Started server-2-2-12.interlink
  * ha.monsrv.aldan3    (ocf::heartbeat:Xen):    Started server-2-2-12.interlink
  * ha.netsrv.aldan3    (ocf::heartbeat:Xen):    Started server-1-2-12.interlink
  * mirror1.aldan3      (ocf::heartbeat:Xen):    Started server-1-2-12.interlink
  * mirror2.aldan3      (ocf::heartbeat:Xen):    Started server-2-2-12.interlink
  * ha.ipa.aldan3       (ocf::heartbeat:Xen):    Started server-2-2-12.interlink
  * ha.slurm.aldan3     (ocf::heartbeat:Xen):    Started server-1-2-12.interlink
  * ha.jhub.aldan3      (ocf::heartbeat:Xen):    Started server-2-2-12.interlink
  * ha.webgw.aldan3     (ocf::heartbeat:Xen):    Started server-2-2-12.interlink
  * ha.ghub.aldan3      (ocf::heartbeat:Xen):    Started server-1-2-12.interlink
  * runner1.aldan3      (ocf::heartbeat:Xen):    Started server-1-2-12.interlink
  * runner2.aldan3      (ocf::heartbeat:Xen):    Started server-2-2-12.interlink
  * mx1.aldan3  (ocf::heartbeat:Xen):    Started server-1-2-12.interlink
  * mx2.aldan3  (ocf::heartbeat:Xen):    Started server-2-2-12.interlink
  * ha.nextcloud        (ocf::heartbeat:Xen):    Started server-1-2-12.interlink
  * ha.stackstorm.aldan3        (ocf::heartbeat:Xen):    Started server-2-2-12.interlink
```

Cluster setup: storage

- Main system CEPH FS (with lz4 compression)
- Standard Installation based on ansible with Docker containers (17.2.6 quincy (stable))
- Two pools with subvolume groups
  - home with replication 3
  - projects with EC8:3
- Subvolume groups snapshots every hour with retention 12h7d3w3m (based on zfs- auto-snapshot https://github.com/witxka/ceph-auto-snapshot)
- osd 1 nvme 2TB per 3 HDD 12TB wal+db

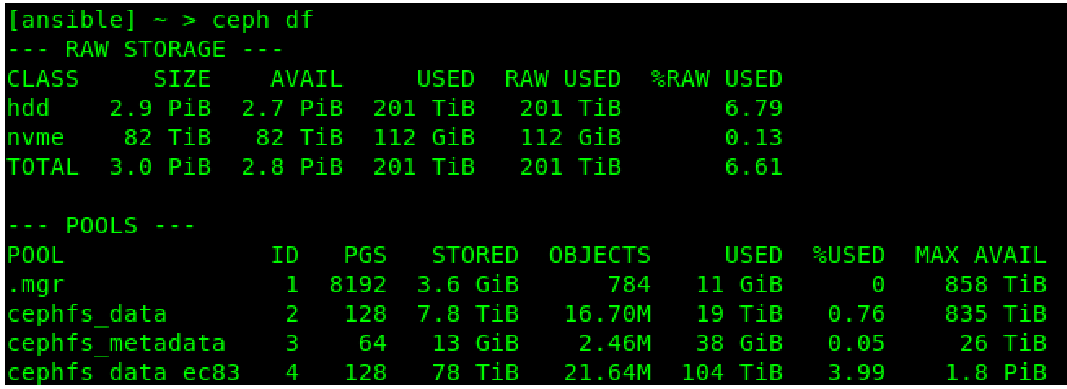


```
[ansible] ~ > ceph -s
  cluster:
    id:     28c39c9f-e3bd-4df2-9d27-89138a47c669
    health: HEALTH_OK
            (muted: MANY_OBJECTS_PER_PG)

  services:
    mon: 5 daemons, quorum storage-1-2-9,storage-1-2-10,storage-1-2-11,storage-2-2-9,storage-2-2-11 (age 4h)
    mgr: storage-1-2-10(active, since 4d), standbys: storage-2-2-11, storage-2-2-9, storage-1-2-9, storage-1-2-11
    mds: 2/2 daemons up, 2 standby
    osd: 352 osds: 352 up (since 3d), 352 in (since 4d)

  data:
    volumes: 1/1 healthy
    pools:   4 pools, 8512 pgs
    objects: 40.80M objects, 86 TiB
    usage:   201 TiB used, 2.8 PiB / 3.0 PiB avail
    pgs:     8509 active+clean
             3    active+clean+scrubbing+deep

  io:
    client:   12 KiB/s rd, 1.0 MiB/s wr, 2 op/s rd, 16 op/s wr
```

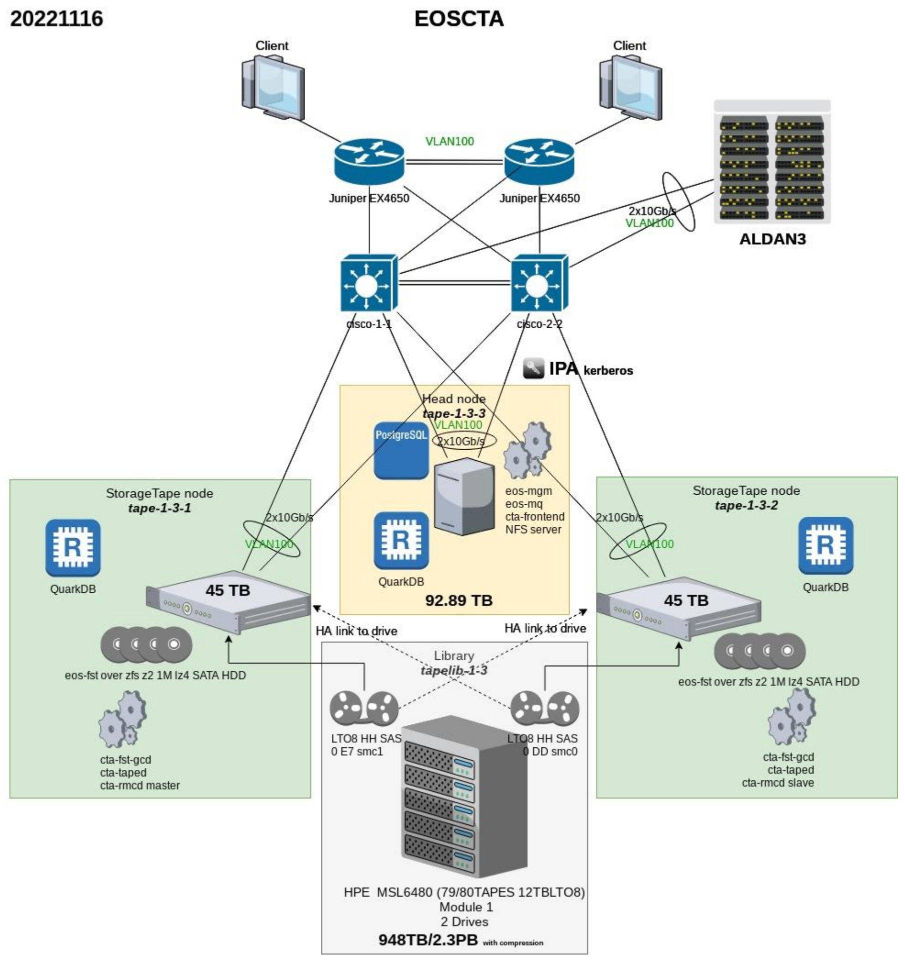


```
[ansible] ~ > ceph df
--- RAW STORAGE ---
CLASS     SIZE    AVAIL     USED  RAW USED  %RAW USED
hdd    2.9 PiB  2.7 PiB  201 TiB   201 TiB       6.79
nvme    82 TiB   82 TiB  112 GiB   112 GiB       0.13
TOTAL  3.0 PiB  2.8 PiB  201 TiB   201 TiB       6.61

--- POOLS ---
POOL                 ID   PGS   STORED  OBJECTS     USED  %USED  MAX AVAIL
.mgr                  1  8192  3.6 GiB      784   11 GiB      0    858 TiB
cephfs_data           2   128  7.8 TiB   16.70M   19 TiB   0.76    835 TiB
cephfs_metadata       3    64   13 GiB    2.46M   38 GiB   0.05     26 TiB
cephfs_data_ec83      4   128   78 TiB   21.64M  104 TiB   3.99    1.8 PiB
```

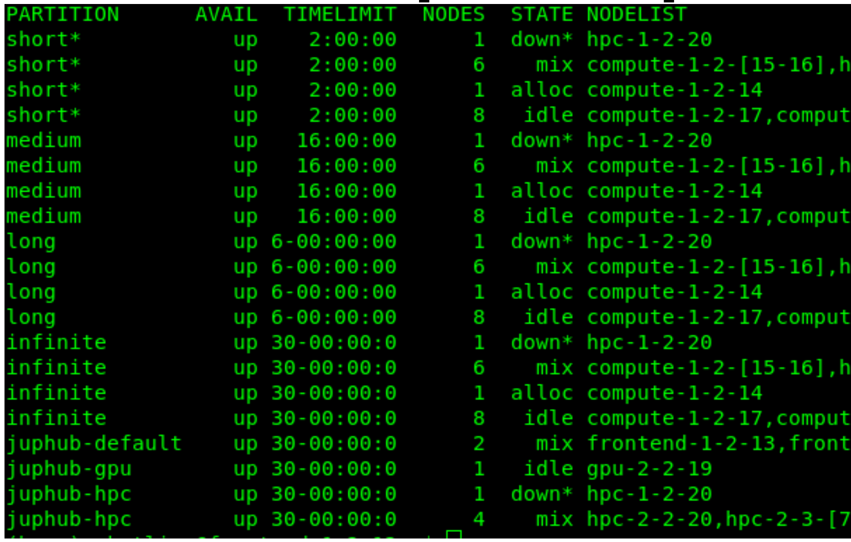
Cluster setup: archive

- CERN TAPE ARCHIVE
- EOS based
- Everything in containers (no k8s)
- All components independent except cluster IPA
- No CEPH used for tape scheduler



Cluster setup: compute

- Slurm as entry point on the cluster
  - Accounting DB
  - Configless setup
  - Cgroups isolation
  - CPU default oversubscribing on (for meantime)
- Several types of nodes
  - frontends (slurm + ssh(cgroups limits))
  - compute (default)
  - HPC (RAM and CPU intensive)
  - GPU (GPU applications)
- Partitions based on timelimits
  - Short partition - more priority
  - Node type selection by feature
  - For juphub partitions custom job_submit.lua for slurm scheduler
  - Standard GPU usage with GRES (no oversubscribing)
- Rootless containers support
  - Build-in OCI support from SLURM (fastest)
  - Enroot from Nvidia with GPU support
  - Podman in rootless mode (docker containers)
- OpenMPI support with Mellanox OFED (100Gb/s Ethernet)
- Two nvme on nodes (RAID0) for local scratch and TMP_DIR (sorting algorithms)



```
PARTITION       AVAIL  TIMELIMIT  NODES  STATE NODELIST
short*             up    2:00:00      1  down* hpc-1-2-20
short*             up    2:00:00      6    mix compute-1-2-[15-16],h
short*             up    2:00:00      1  alloc compute-1-2-14
short*             up    2:00:00      8   idle compute-1-2-17,comput
medium             up   16:00:00      1  down* hpc-1-2-20
medium             up   16:00:00      6    mix compute-1-2-[15-16],h
medium             up   16:00:00      1  alloc compute-1-2-14
medium             up   16:00:00      8   idle compute-1-2-17,comput
long               up 6-00:00:00      1  down* hpc-1-2-20
long               up 6-00:00:00      6    mix compute-1-2-[15-16],h
long               up 6-00:00:00      1  alloc compute-1-2-14
long               up 6-00:00:00      8   idle compute-1-2-17,comput
infinite           up 30-00:00:0      1  down* hpc-1-2-20
infinite           up 30-00:00:0      6    mix compute-1-2-[15-16],h
infinite           up 30-00:00:0      1  alloc compute-1-2-14
infinite           up 30-00:00:0      8   idle compute-1-2-17,comput
juphub-default     up 30-00:00:0      2    mix frontend-1-2-13,front
juphub-gpu         up 30-00:00:0      1   idle gpu-2-2-19
juphub-hpc         up 30-00:00:0      1  down* hpc-1-2-20
juphub-hpc         up 30-00:00:0      4    mix hpc-2-2-20,hpc-2-3-[7
```

- User's access from Jupyter hub over slurm
  - Extensions for slurm
  - Rstudio
  - Mamba for python environments through gui
  - Jupyter python environments support
- Standard ssh access by keys with limits on cpu and mem.

Maintenance

- Services oriented monitoring with telegram notifications (small teams to operate in push mode)
- Auto install for security updates
- Cluster defense against cooling and power cuts
- Users onboarding to a new cluster
- Only system libraries support – no software support for users (anyway helping in environment setup)
- regular system upgrades once per three months
- Base components (ceph, jupyter, slurm) software upgrades once per 1-2 year

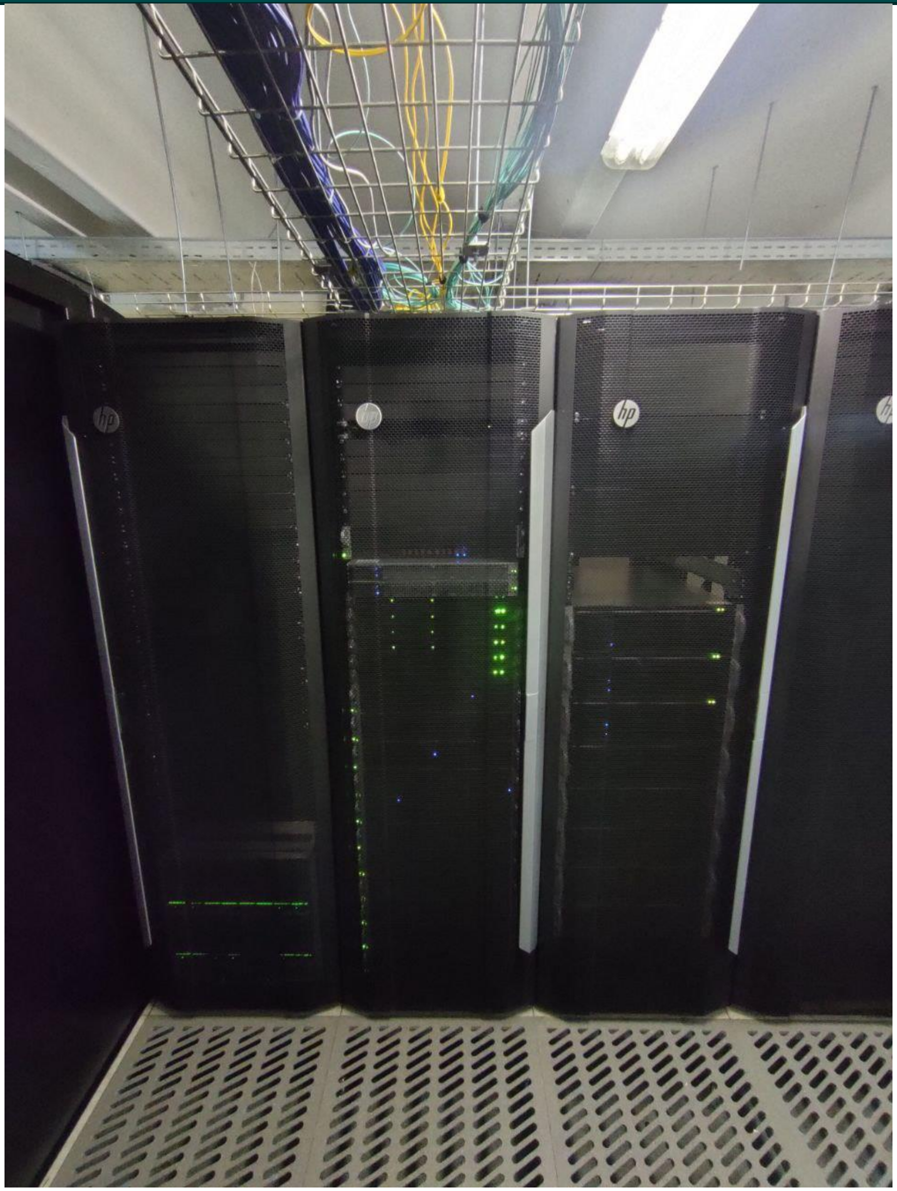
Plans

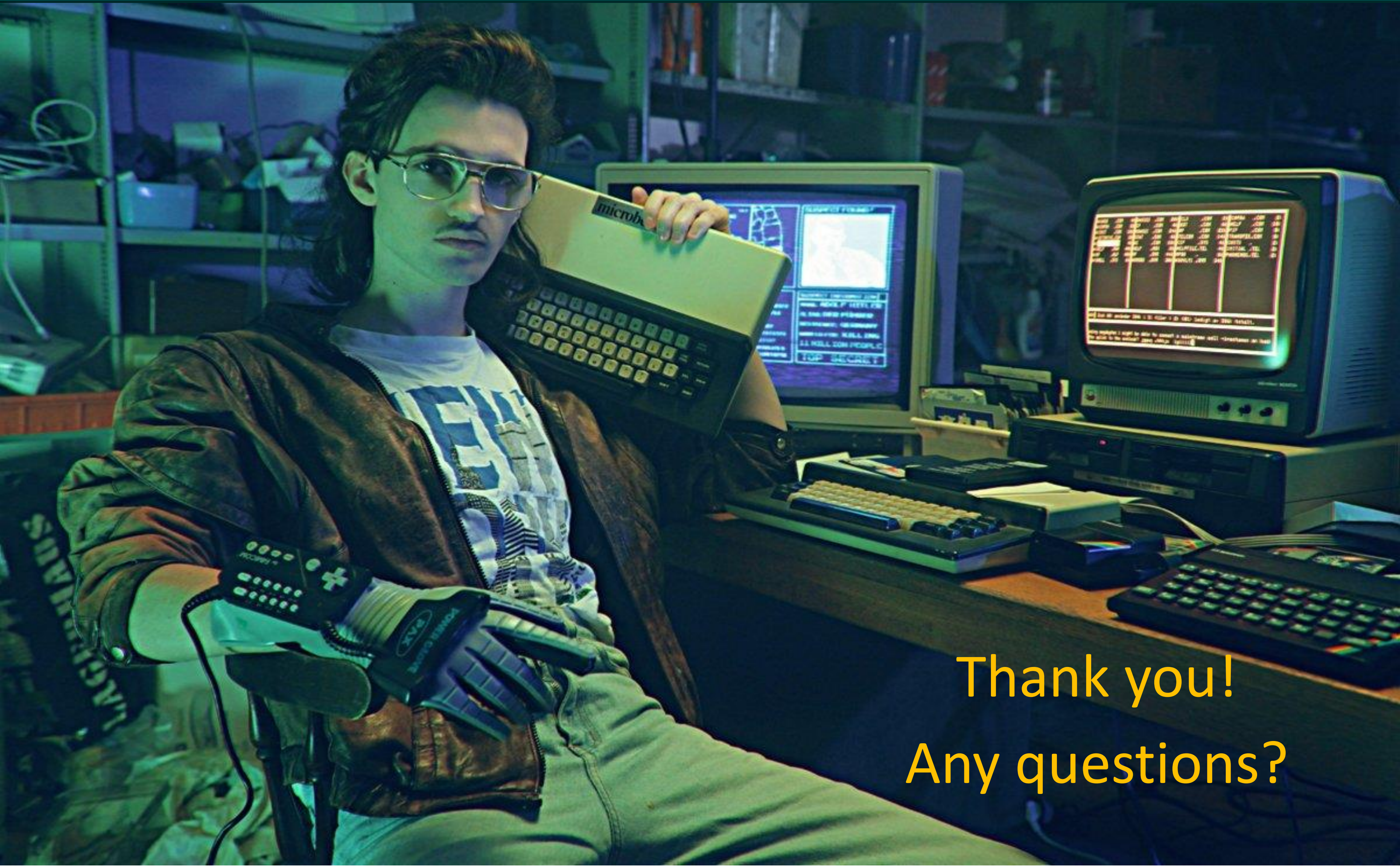
- Add more workflow based scenarios for system management
- Chat-ops usage with integrated workflows
- Add more usage patterns to the cluster
    - sequenators integration
    - students labs integration
    - Integration with other laboratories
- Increase efficiency for resources usage

Photos

Photos

Photos





Thank you!

Any questions?