New Users Training

Introduction to FASRC clusters

Learning objectives 1 – FASRC account

- Learn how to request an FASRC account
- Activate your new account
- How to modify your account or add groups

Learning objectives 2 – Intro to HPC

- What is high-performance computing (HPC)? How is it different from a desktop/laptop?
- Laptop vs. Cannon
- Why HPC?
- FASRC clusters
- Cluster architecture
- Job scheduler
- Choose compute resources for jobs
    - Memory, cores
    - Partitions, file systems
- Storage
- Data Management
- Cluster customs and responsibilities

Learning objectives 3 – Documentation and help

- FASRC docs
- GitHub User_codes
- Office hours
- Tickets

Request FASRC account

https://docs.rc.fas.harvard.edu/kb/quickstart-guide/

1. Request an account using Account Request Tool https://portal.rc.fas.harvard.edu/request/account/new
   - Use Harvard Key option
2. Set FASRC password https://portal.rc.fas.harvard.edu/p3/pwreset/
3. Set two-factor authentication https://docs.rc.fas.harvard.edu/kb/openauth/
4. Set FASRC VPN (needed for mounting storage, OOD, level 3 data, license server access) https://docs.rc.fas.harvard.edu/kb/vpn-setup/
5. Review intro training

How to modify your account

- Change labs: https://docs.rc.fas.harvard.edu/kb/change-lab-group/
- Add a lab:
  - Portal gives access to lab storage: https://docs.rc.fas.harvard.edu/kb/additional-groups/
  - If you work for more than 1 PI, and need access to lab slurm account (more on slurm later), send a ticket
- Never request a second account!!
- Membership in the FASRC mailing-list is required
- Account needs to be used in the last 12 months to be active
- After 12 months of inactivity
  - Account is disabled, but nothing is deleted
  - Can be reactivated with PI/admin approval

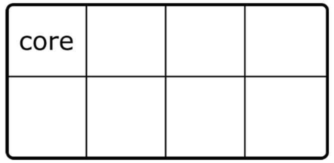
What is HPC?

- HPC: High performance computing
- HPC: biggest and fastest computing machines right now
- Supercomputers: rule of thumb - at least 100 times as powerful as a PC (personal computer)
- Jargon: other terms
  - Supercomputing
  - Cyberinfrastructure (CI)
  - Cluster computing

Laptop vs. Cannon

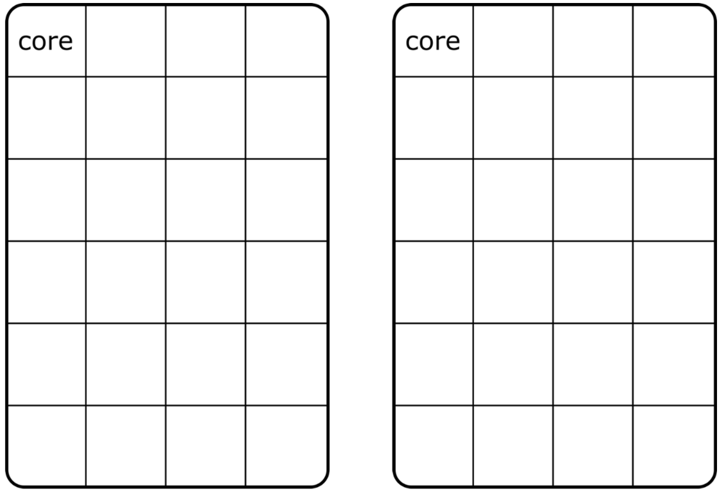
MacBook Pro

- 1 CPU (processor)
- 4-12 cores per CPU
- Memory: 16-96 GB



Cannon typical nodes

- 2 CPUs
- 24-56 cores **per CPU**
- Memory: 184-2000 GB
- **1260+ nodes!!!**



Why HPC?

- Size: problems that can't fit on a desktop/laptop, for example 500+ GB of RAM or 100s of cores
- Speed: problems that take months on a laptop may take a few hours on a supercomputer
- Amount: need 1000s of runs

45 miles/hour

600 miles/hour

What about FASRC clusters?

Massachusetts Green HPC Center (MGHPCC)

Cannon cluster

From https://www.servethehome.com/the-harvard-cannon-powered-by-lenovo-neptune/

FASRC clusters: Cannon and FASSE

Cannon

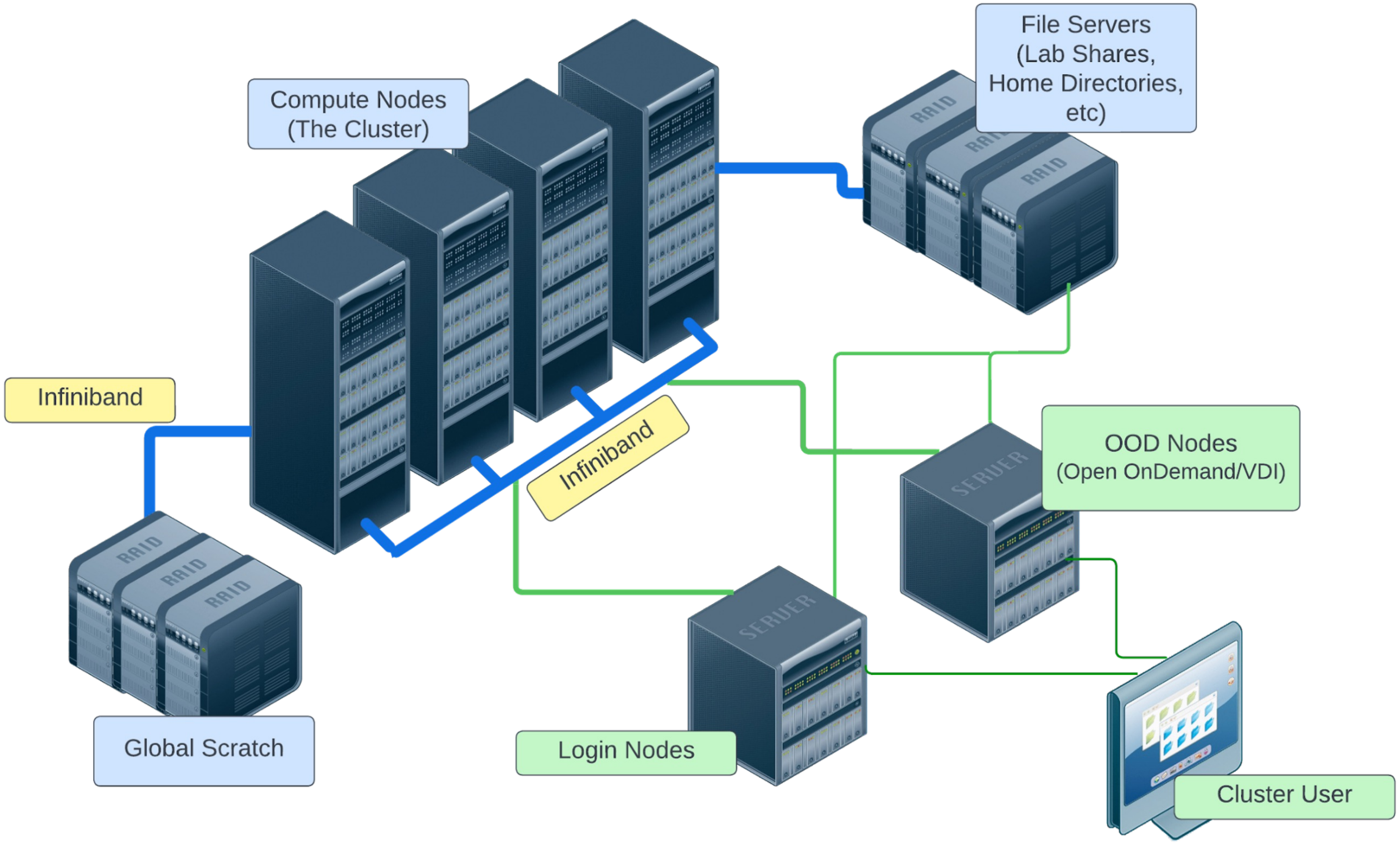
- General purpose
- Only level 1 and 2 data

FASSE

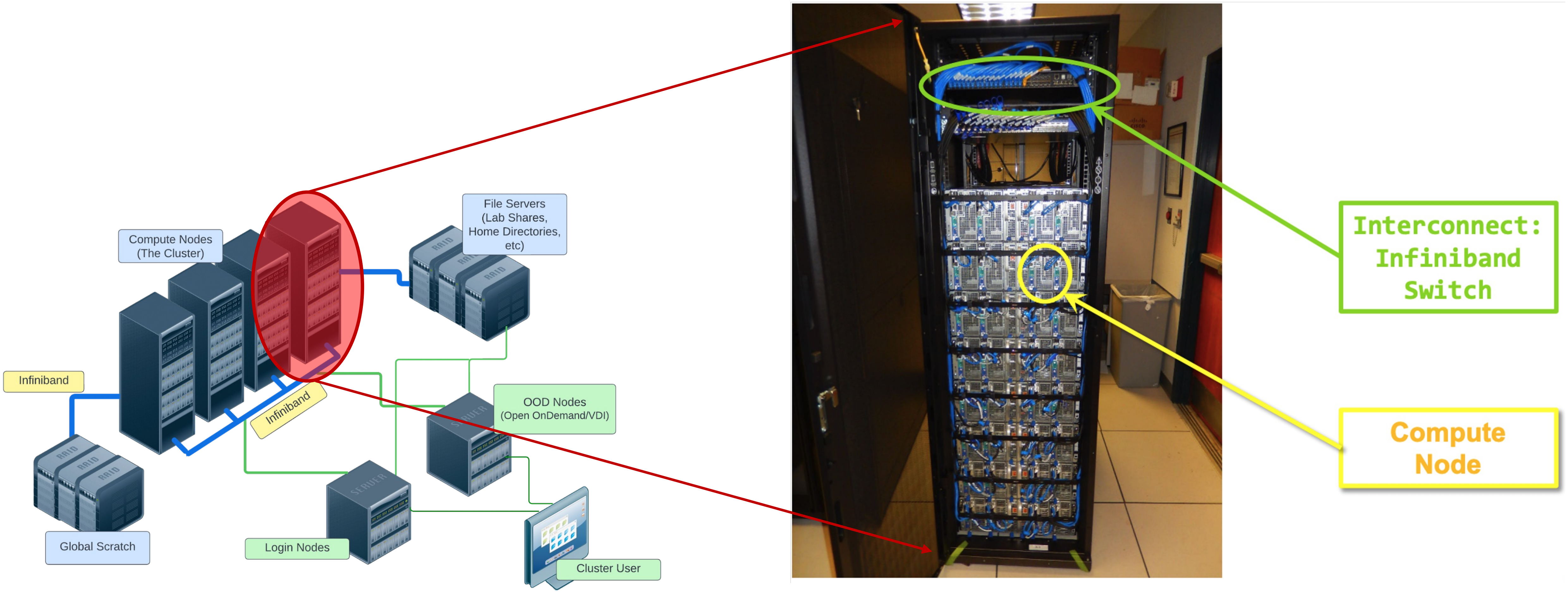
- FAS Secure Environment
- Secure multi-tenant environment
- Analysis of sensitive datasets with DUAs and IRBs
- Level 3 data, no level 4 data
- PI/lab responsibility to know their data
- https://policy.security.harvard.edu/
- https://docs.rc.fas.harvard.edu/kb/data-use-agreements/
- https://security.harvard.edu/
- https://docs.rc.fas.harvard.edu/kb/fasse/

PUBLIC	Public information (Level 1)	Level 1 Harvard Systems
LOW	Low Risk information (Level 2) is information the University has chosen to keep confidential but the disclosure of which would not cause material harm.	Low Risk Systems (L2)
MEDIUM	Medium Risk information (Level 3) could cause risk of material harm to individuals or the University if disclosed or compromised.	Medium Risk Systems (L3)
HIGH	High risk information (Level 4) would likely cause serious harm to individuals or the University if disclosed or compromised.	High Risk Systems (L4)
LEVEL 5	Reserved for extremely sensitive Research Data that requires special handling per IRB determination.	Level 5 Systems

Cluster architecture

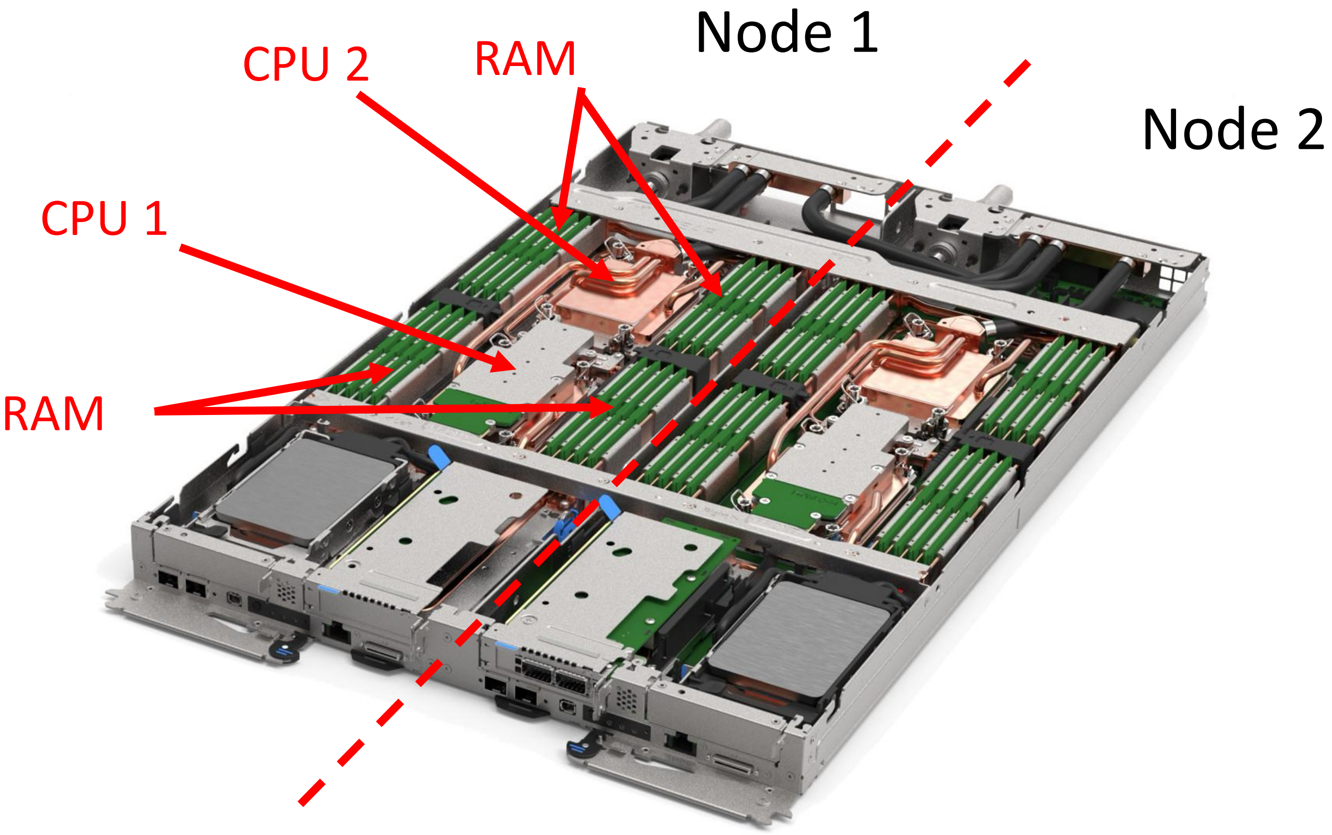


Rack



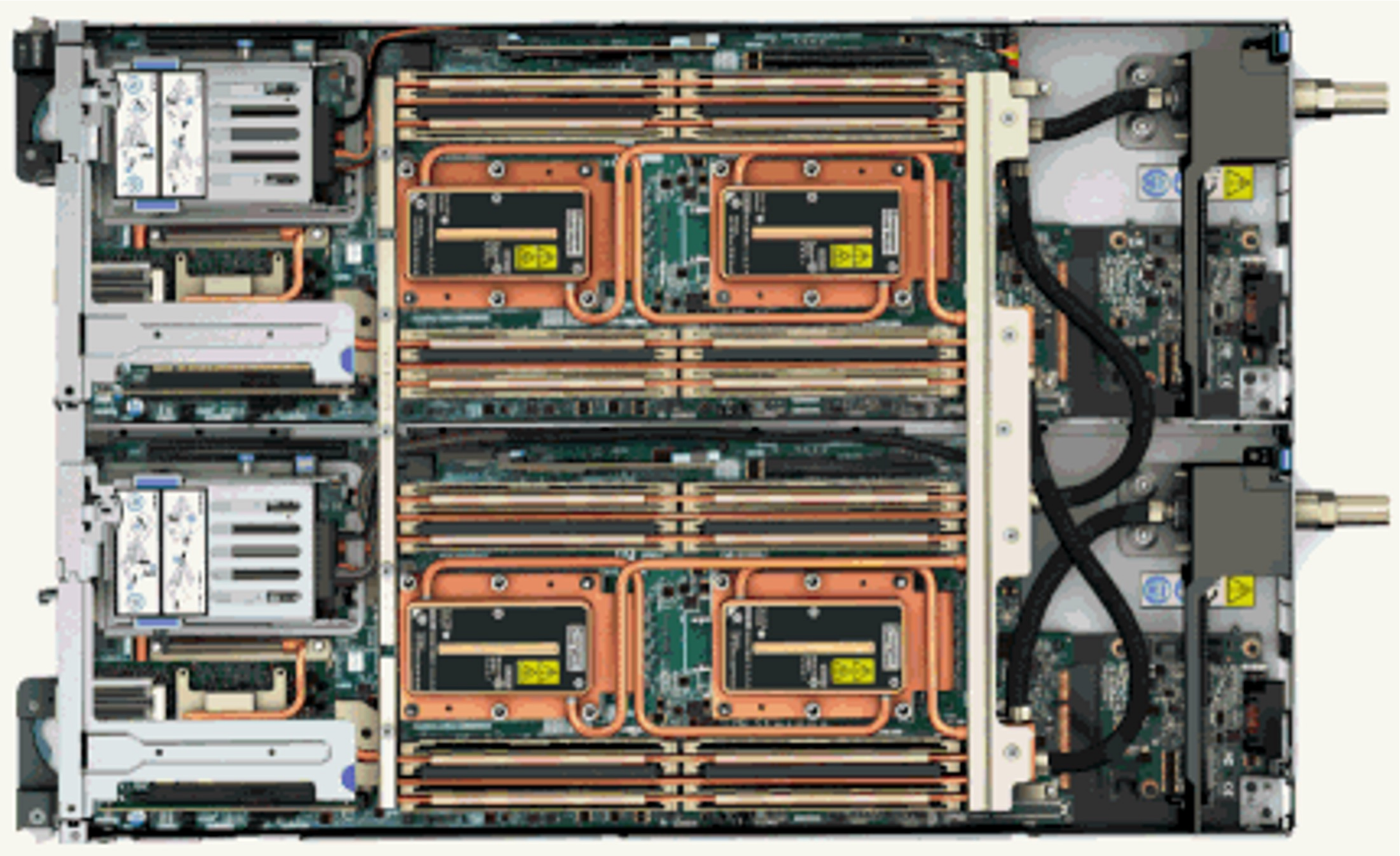
From HPC@LSU training (http://www.hpc.lsu.edu/training/weekly-materials/2022-Fall/HPC_UserEnv1_Fall2022.pdf)

Node



From https://lenovopress.lenovo.com/lp1603-thinksystem-sd650-v3-server

Node water cooling



From https://www.rc.fas.harvard.edu/blog/cannon-makes-top-500-list/

Node, processors, core

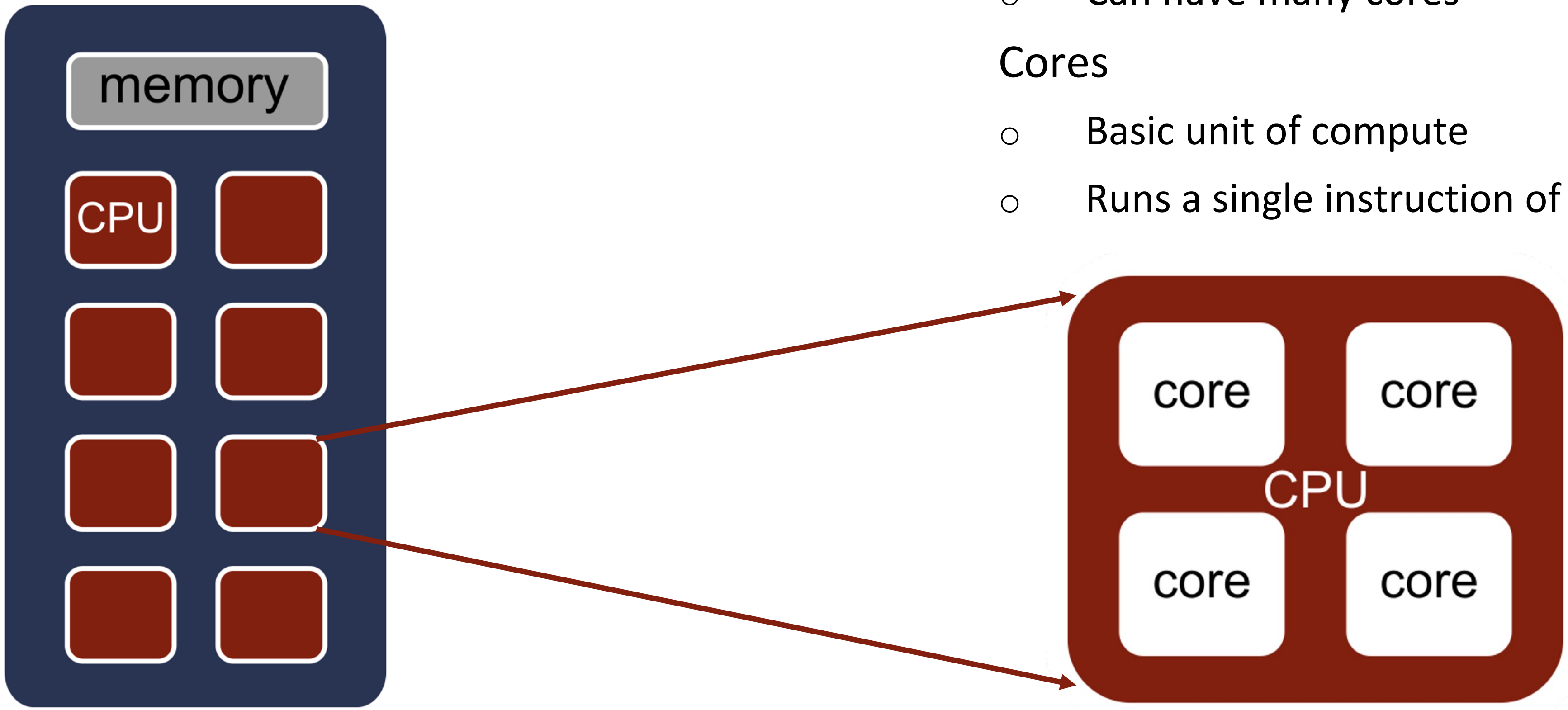
Node: a computer in the cluster

CPU

- Central processing unit, processor
- Can have many cores

Cores

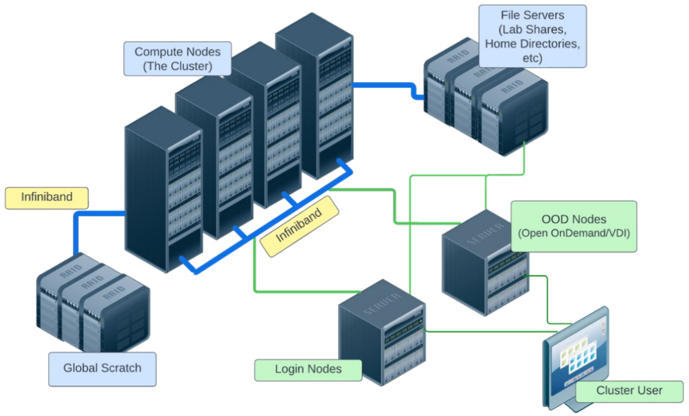
- Basic unit of compute
- Runs a single instruction of code



Nomenclature summary

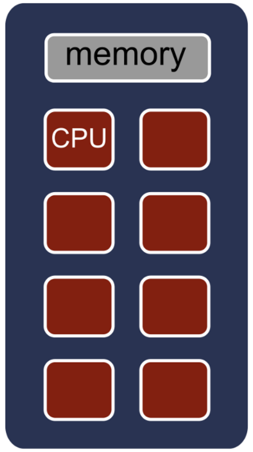
Cluster

Top level unit of a supercomputer



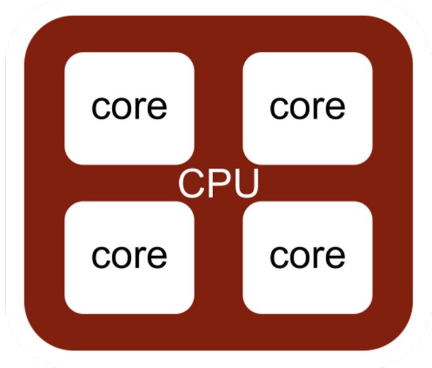
Node

One host in the cluster (i.e., one computer)



Core

Basic unit of computer



New term: Job

A user's request to use a certain amount of resources for a specific amount of time

Glossary: https://docs.rc.fas.harvard.edu/kb/glossary/

Adapted from HPC@LSU training (http://www.hpc.lsu.edu/training/weekly-materials/2021-Summer/HPC_UserEnv_2021_Summer_session_2.pdf)

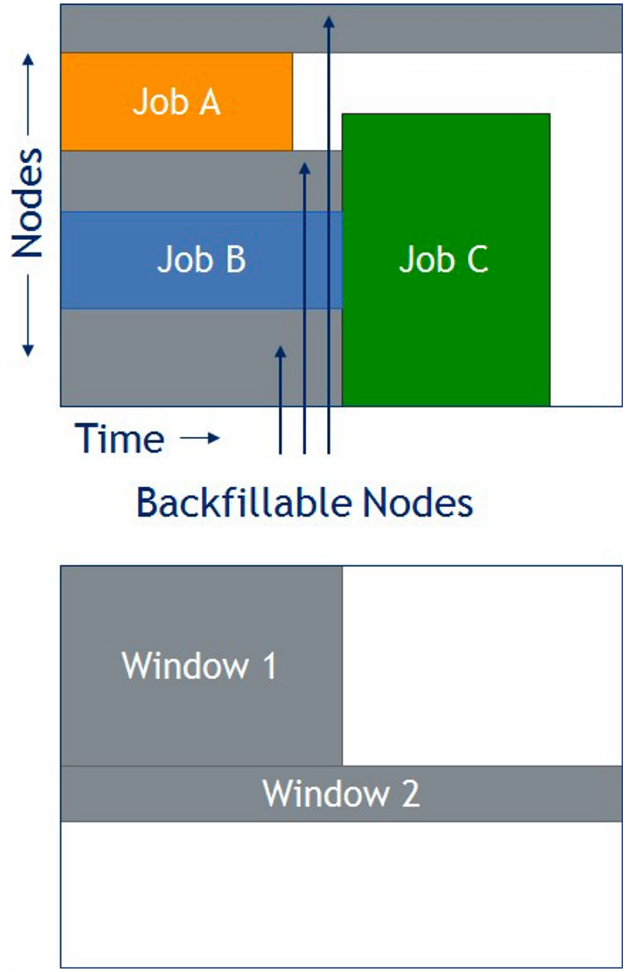
Job scheduler

- The Cluster is a multi-tenant environment, so how can everyone use it fairly?
- Job scheduler!
- Slurm: Simple Linux Utility for Resource Management
  - Manages job queue for a cluster of resources
  - Prioritizes jobs
  - Provides status of running, queue, completed and failed jobs
  - Determines the order jobs are executed
  - On which node(s) jobs are executed

Adapted from HPC@LSU training (http://www.hpc.lsu.edu/training/weekly-materials/2021-Summer/HPC_UserEnv_2021_Summer_session_2.pdf)

Job management philosophy

- Prioritize workload
- Backfill idle node to maximize cluster use



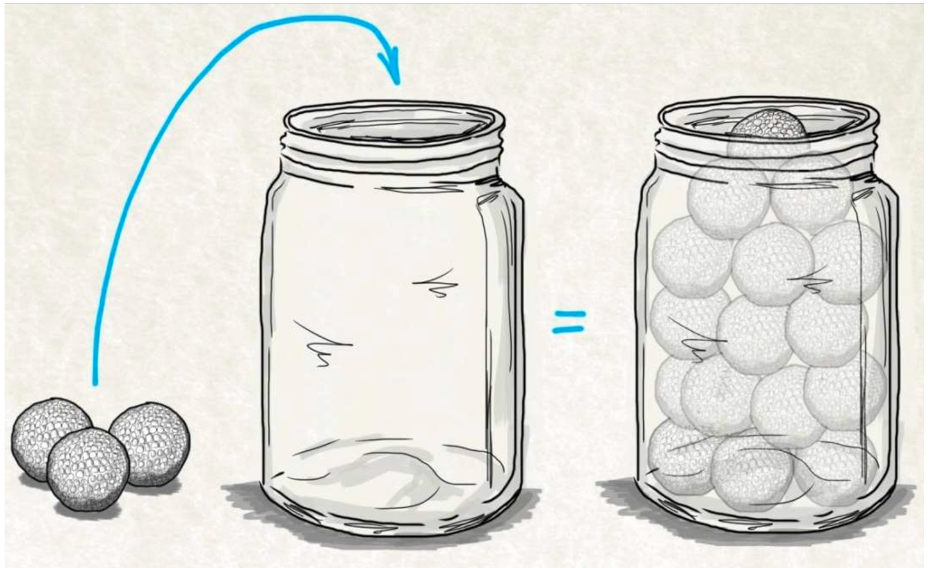
Job Priority

- **Not** first come, first served
- Job with higher priority scheduled ahead of jobs with lower priority
- Priority depends on
  - Fairshare
  - Amount of time pending
  - Group priority

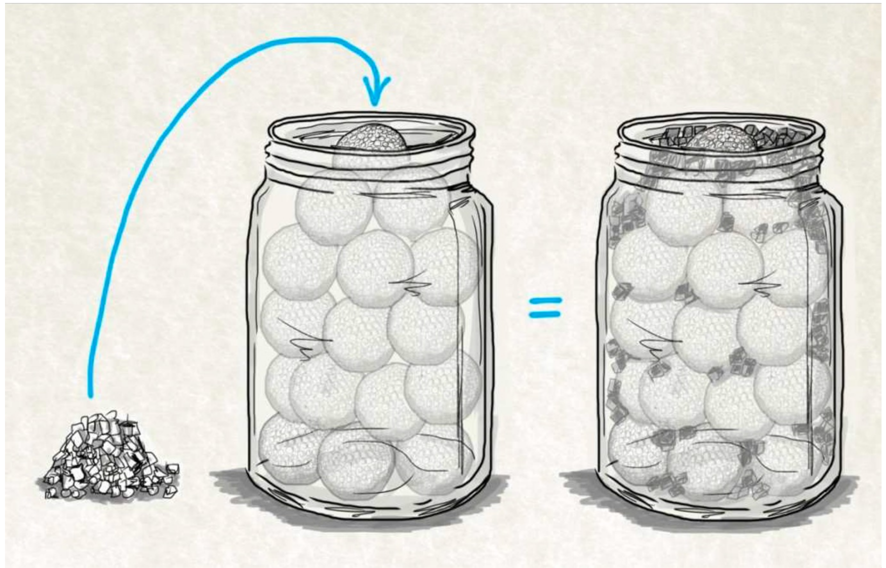
Adapted from HPC@LSU training (http://www.hpc.lsu.edu/training/weekly-materials/2021-Summer/HPC_UserEnv_2021_Summer_session_2.pdf)

How to maximize cluster usage?

1. Fill in high-priority jobs



2. Backfill with low-priority jobs



Adapted from HPC@LSU training (http://www.hpc.lsu.edu/training/weekly-materials/2021-Summer/HPC_UserEnv_2021_Summer_session_2.pdf)

Choosing computational resources

- How do we choose memory, cores, partitions, and file systems?
- First time ever running on a cluster?
  - Run a test case choosing similar resources as the machine you are currently using
  - Check how efficient your job was and adjust it accordingly
- Increasing a job/analysis/simulation?
  - Run for a small test case
  - Increase size by 1.5, 2.0, 2.5x and check how job scaled
  - Then you can have a rough estimation of how much a first trial production job of ~10x would require

Cannon partitions

Documentation: https://docs.rc.fas.harvard.edu/kb/running-jobs/

Partitions	sapphire	shared	gpu	test	gpu_test	serial_requeue	gpu_requeue	bigmem	intermediate	bigmem_ intermediate	unrestricted	**pi_lab**
Time Limit	3 days	3 days	3 days	12 h	12 h	3 days	3 days	3 days	3-14 days	3-14 days	no limit	**varies**
# Nodes	192	288	36	12	14	varies	varies	4	12	3	8	**varies**
# Cores / Node	112	48	64 + 4 A100	112	64 + 8 A100 MIG	varies	varies	112	112	64	48	**varies**
Memory / Node (GB)	1004	184	1004	1004	375	varies	varies	2000	1004	2000	184	**varies**

FASSE partitions

Documentation: https://docs.rc.fas.harvard.edu/kb/fasse/

Partitions	fasse	fasse_gpu	test	serial_requeue	fasse_bigmem	fasse_ultramem	remoteviz	**pi_lab**
Time Limit	7 days	7 days	12 h	7 days	7 days	7 days	7 days	**varies**
# Nodes	42	4	5	varies	17	1	1	**varies**
# Cores / Node	48	64 + 4 A100	48	varies	64	64	32	**varies**
Memory / Node (GB)	184	499	184	varies	499	2000	373	**varies**

Which partitions can I use?

Documentation: https://docs.rc.fas.harvard.edu/kb/convenient-slurm-commands/

```
[jharvard@boslogin02 ~]$ spart
Partition              State    Cores    GPUs    Average Mem/Node(GB)    Nodes    Time Limit
bigmem                 UP       448      0       2015                    4        3-00:00:00
bigmem_intermediate    UP       192      0       2015                    3        14-00:00:00
gpu                    UP       2304     144     1007                    36       3-00:00:00
gpu_requeue            UP       9184     698     772                     156      3-00:00:00
gpu_test               UP       896      112     503                     14       12:00:00
intermediate           UP       1344     0       1007                    12       14-00:00:00
remoteviz              UP       32       0       377                     1        3-00:00:00
sapphire               UP       21504    0       1007                    192      3-00:00:00
serial_requeue         UP       88300    690     438                     1457     3-00:00:00
shared                 UP       13824    0       188                     288      3-00:00:00
test                   UP       1344     0       1007                    12       12:00:00
ultramem               DRAIN    192      0       2015                    3        3-00:00:00
unrestricted           UP       384      0       188                     8        UNLIMITED
```

Storage

Tier storage documentation: https://www.rc.fas.harvard.edu/services/data-storage/

	Home Directories	Lab Directory (Startup)	Local Scratch	Global Scratch	Tier Storage
Mount Point	$HOME /n/home#/$USER /n/home_fasse/$USER	/n/holylabs/LABS/pi_lab	/scratch	$SCRATCH /n/holyscratch01/pi_lab	/n/pi_lab
Size Limit	100GB	4TB	70+ GB/node	2.4PB total	Based on Tier
Availability	All cluster nodes + Desktop/laptop	All cluster nodes	Local compute node only	All cluster nodes	All cluster nodes/ mountable
Retention Policy	Indefinite	Indefinite	Job duration	90 days	Indefinite
Backup	Hourly snapshot + Daily Offsite	No backup	No backup	No backup	Depending on Tier
Performance	Moderate. Not suitable for high I/O	Moderate. Not suitable for high I/O	Suited for small file I/O intensive jobs	Appropriate for large file I/O intensive jobs	Depending on Tier
Cost	Free	Free max of 4TB	Free	Free	Paid

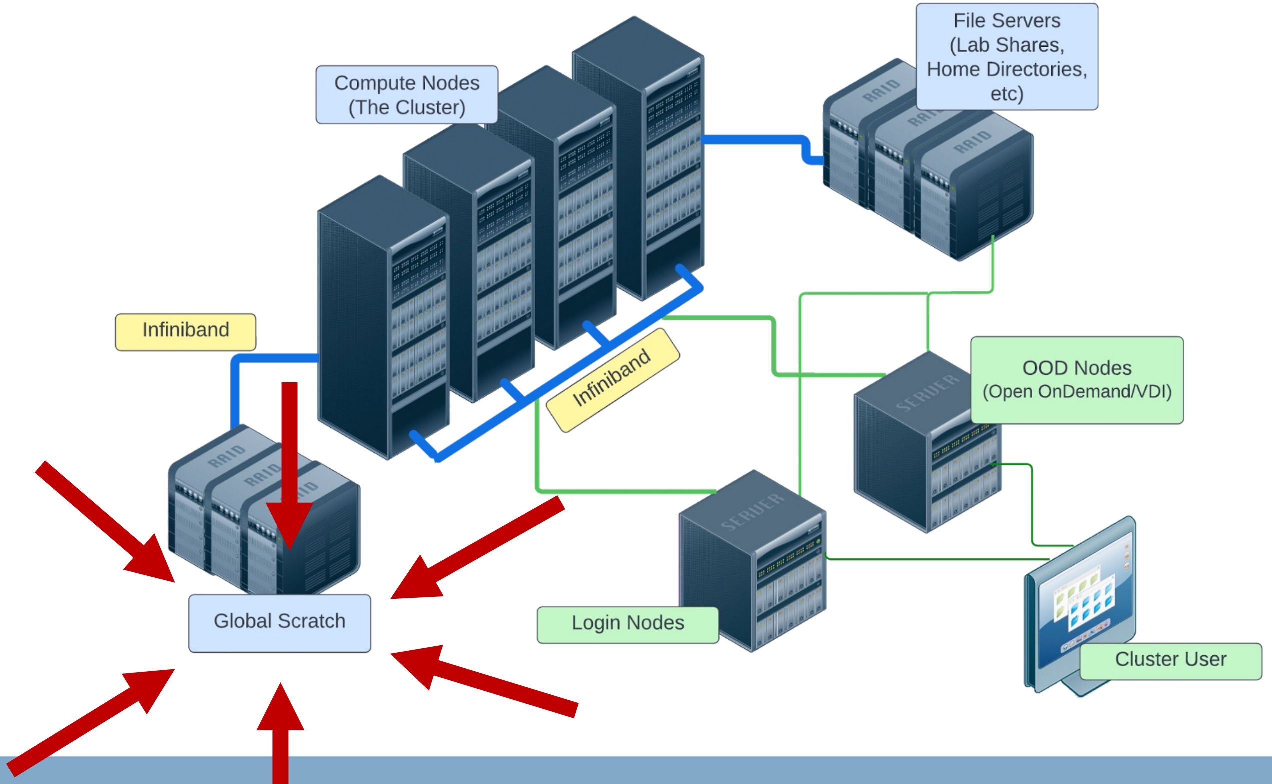
Storage schematics

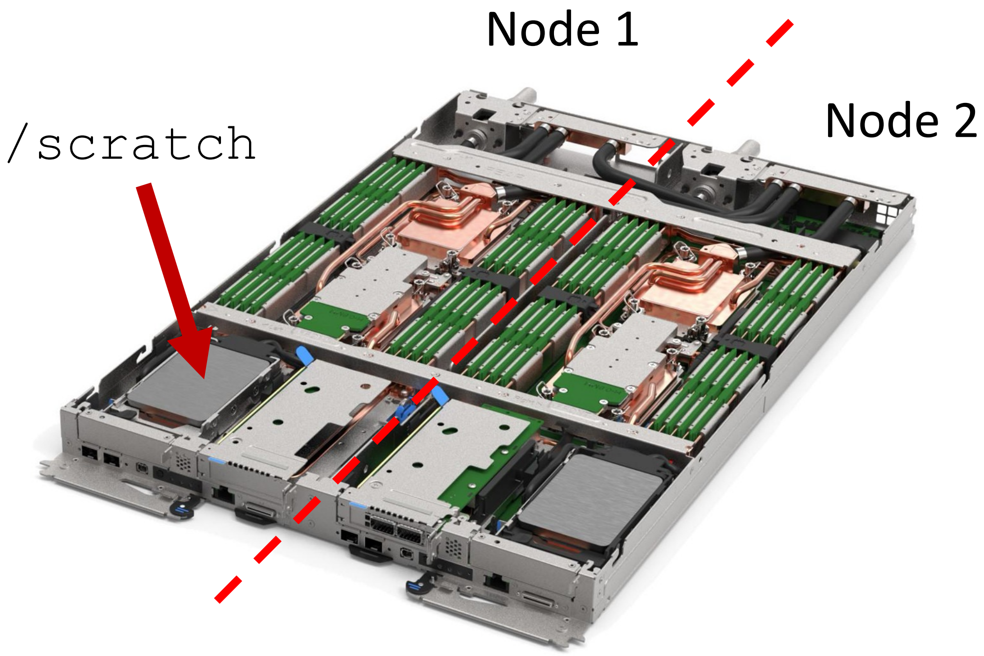
Global Scratch

- Networked scratch
- Global variable: `$SCRATCH`
- Path: `/n/holyscratch01/pi_lab`

Local Scratch

- Storage on the node
- Path: `/scratch`



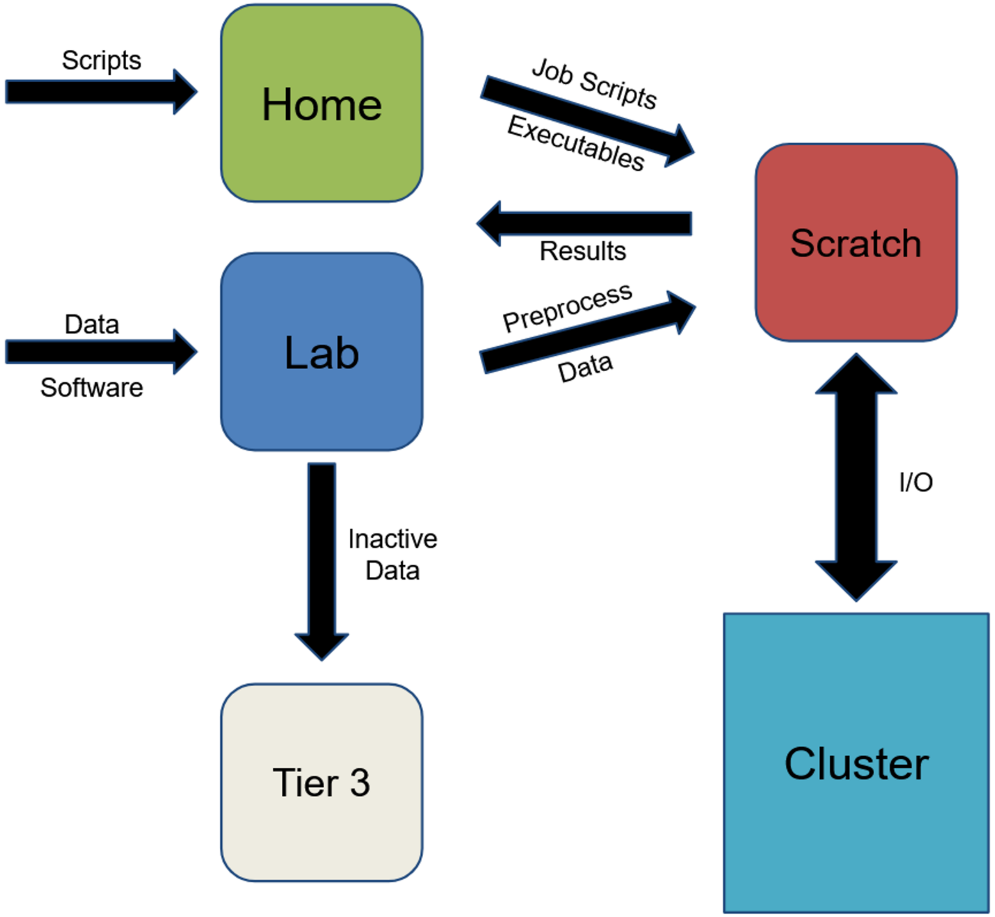


From https://lenovopress.lenovo.com/lp1603-thinksystem-sd650-v3-server

Data management

Documentation: https://docs.rc.fas.harvard.edu/kb/data-management-best-practices/

- Home
  - Backed up with daily snapshots (up to 2 weeks)
  - “Valuable” and small code
- Global scratch
  - Temporary storage
  - Copy job scripts and executables for jobs
  - Input data, output results
  - Do not have multiple jobs hitting the same file!!
- Lab storage
  - Permanent storage
  - If you have code here and not backed up, use version control (`git`)!!



Cluster customs and responsibilities (1)

Documentation: https://docs.rc.fas.harvard.edu/kb/responsibilities/

- Don't run anything on the login nodes
- Be as accurate as possible for memory requests
- Keep job counts reasonable: 10,000 job limit per user (scheduled or running)
- Request at least 10 minutes
- Don't overwhelm scheduler: wait 0.5 to 1 sec for `sbatch` and `sacct` commands

Cluster customs and responsibilities (2)

Documentation: https://docs.rc.fas.harvard.edu/kb/responsibilities/

- Use appropriate partition
- Use `serial_requeue` and `gpu_requeue` when possible
- Heavy I/O should be done on `/scratch` and `$SCRATH`
- Keep at most 1000 files per directory (i.e., folder)
- No production work on test partitions
- Poorly behaved jobs will be terminated
- Don’t mine digital currency or misuse Harvard resources

Acknowledge using the FASRC Clusters

Documentation: https://docs.rc.fas.harvard.edu/kb/attribution/

- If you publish work performed on FASRC clusters, acknowledge it:

"The computations in this paper were run on the FASRC cluster supported by the FAS Division of Science Research Computing Group at Harvard University."

Survey

Please, fill out our course survey. Your feedback is essential for us to improve our trainings!!

http://tinyurl.com/FASRCsurvey

FASRC documentation

- FASRC docs: https://docs.rc.fas.harvard.edu/
  - User search!!
- GitHub User_Codes: https://github.com/fasrc/User_Codes/
- Getting help
  - Office hours: https://www.rc.fas.harvard.edu/training/office-hours/
  - Ticket
    - Portal: http://portal.rc.fas.harvard.edu/rcrt/submit_ticket (requires login)
    - Email: rchelp@rc.fas.harvard.edu

Upcoming training sessions

Training calendar: https://www.rc.fas.harvard.edu/upcoming-training/

Getting started on the FASRC clusters with Open OnDemand

- Audience
  - New users not familiar with command-line interface
  - Wants to use a GUI
- Requirements
  - Single-node jobs
  - Working FASRC account with cluster access
- Content
  - Access Open OnDemand
  - Launch Jupyter, Rstudio Server, Remote Desktop
  - Install Rstudio Server packages
  - Install python packages for Jupyter
  - Launch software from Remote Desktop

Getting started on the FASRC clusters with command line interface (CLI)

- Requirement: working FASRC account with cluster access
- Audience
  - Users familiar with command-line interface
  - New to Cannon and FASSE, but familiar with HPC systems
- Content
  - Submit interactive job with salloc
  - Submit batch job sbatch
  - Monitor jobs
  - Cluster software overview (modules, spack)

Thank you :)

FAS Research Computing