HARVARD UNIVERSITY

FASRC

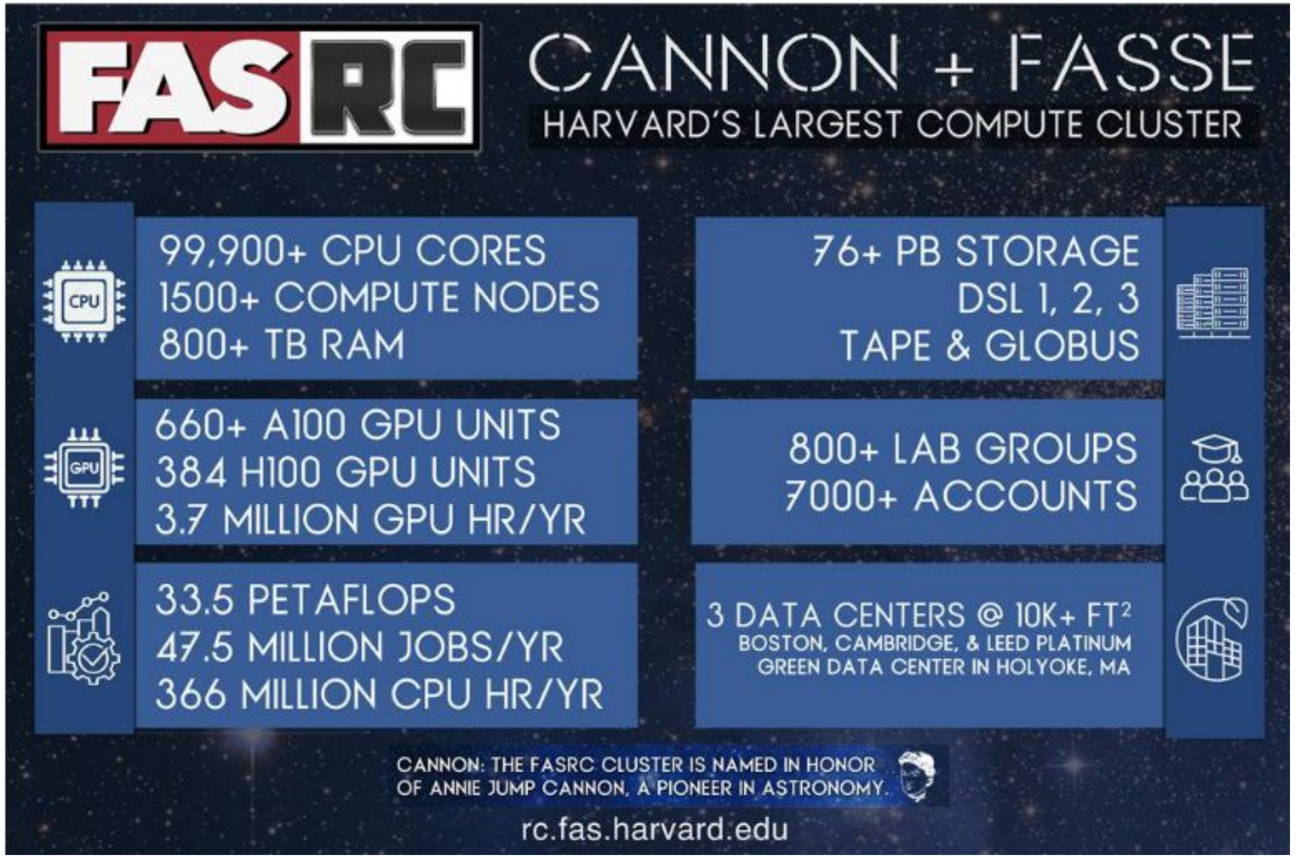
FAS Research Computing
Division of Science
https://rc.fas.harvard.edu

New Users Training

Introduction to FASRC clusters

What is FASRC?

- Compute Clusters
- FASSE Clusters
- Access and Authentication
- Data Storage
- Infrastructure
- Security
- Research Data Management
- Support and Training



FASRC

CANNON + FASSE

HARVARD'S LARGEST COMPUTE CLUSTER

99,900+ CPU CORES
1500+ COMPUTE NODES
800+ TB RAM

76+ PB STORAGE
DSL 1, 2, 3
TAPE & GLOBUS

660+ A100 GPU UNITS
384 H100 GPU UNITS
3.7 MILLION GPU HR/YR

800+ LAB GROUPS
7000+ ACCOUNTS

33.5 PETAFLOPS
47.5 MILLION JOBS/YR
366 MILLION CPU HR/YR

3 DATA CENTERS @ 10K+ FT^2
BOSTON, CAMBRIDGE, & LEED PLATINUM
GREEN DATA CENTER IN HOLYOKE, MA

CANNON: THE FASRC CLUSTER IS NAMED IN HONOR OF ANNIE JUMP CANNON, A PIONEER IN ASTRONOMY.

rc.fas.harvard.edu

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)?
- 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 - https://docs.rc.fas.harvard.edu
- GitHub User Codes - https://github.com/fasrc/User_Codes
- Office Hours - https://rc.fas.harvard.edu/training/office-hours
- Tickets
  - Send email to rchelp@rc.fas.harvard.edu

FASRC Account

Request FASRC account

Quick start guide: 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/

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 6-12 months of inactivity
  - Account is disabled, but nothing is deleted
  - Can be reactivated with PI/admin approval

Intro to HPC

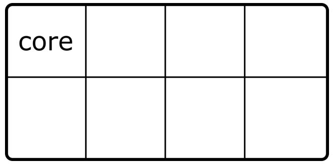
What is HPC?

- 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

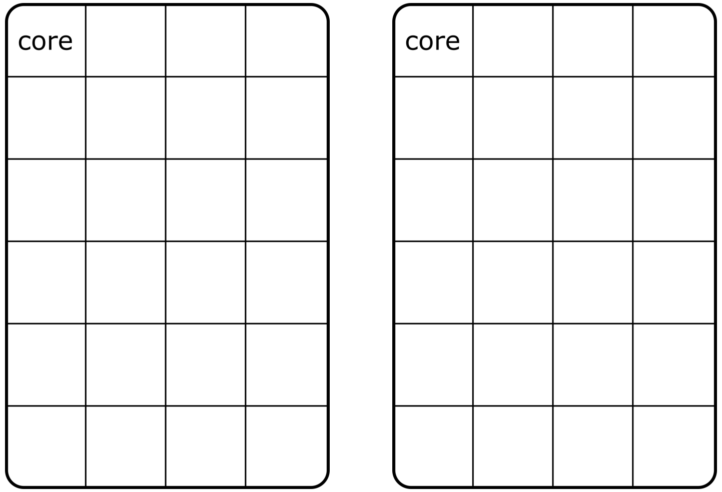
MacBook Pro

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



Cannon typical nodes

- 2 CPUs
- 48-112 cores per node
- Memory: 184-2000 GB
- **1400+ 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

FASRC clusters

Massachusetts Green HPC Center (MGHPCC)

Cannon cluster

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

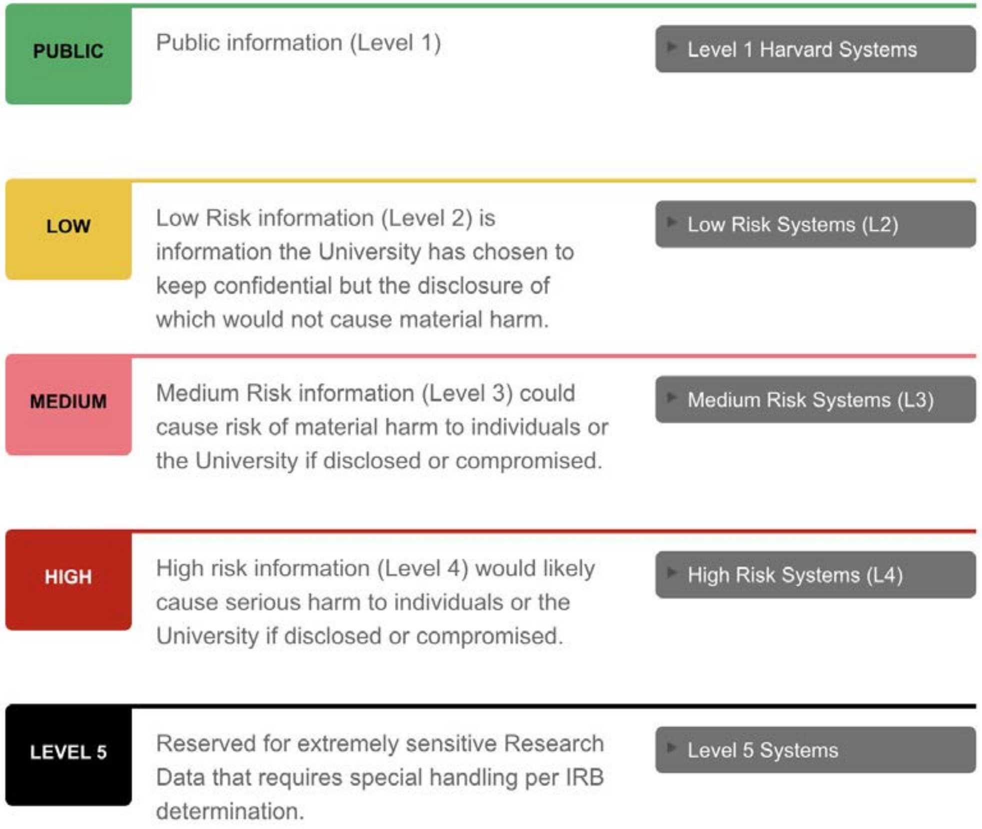
FASRC clusters: Cannon and FASSE

Cannon

- General purpose
- Only level 1 and 2 data

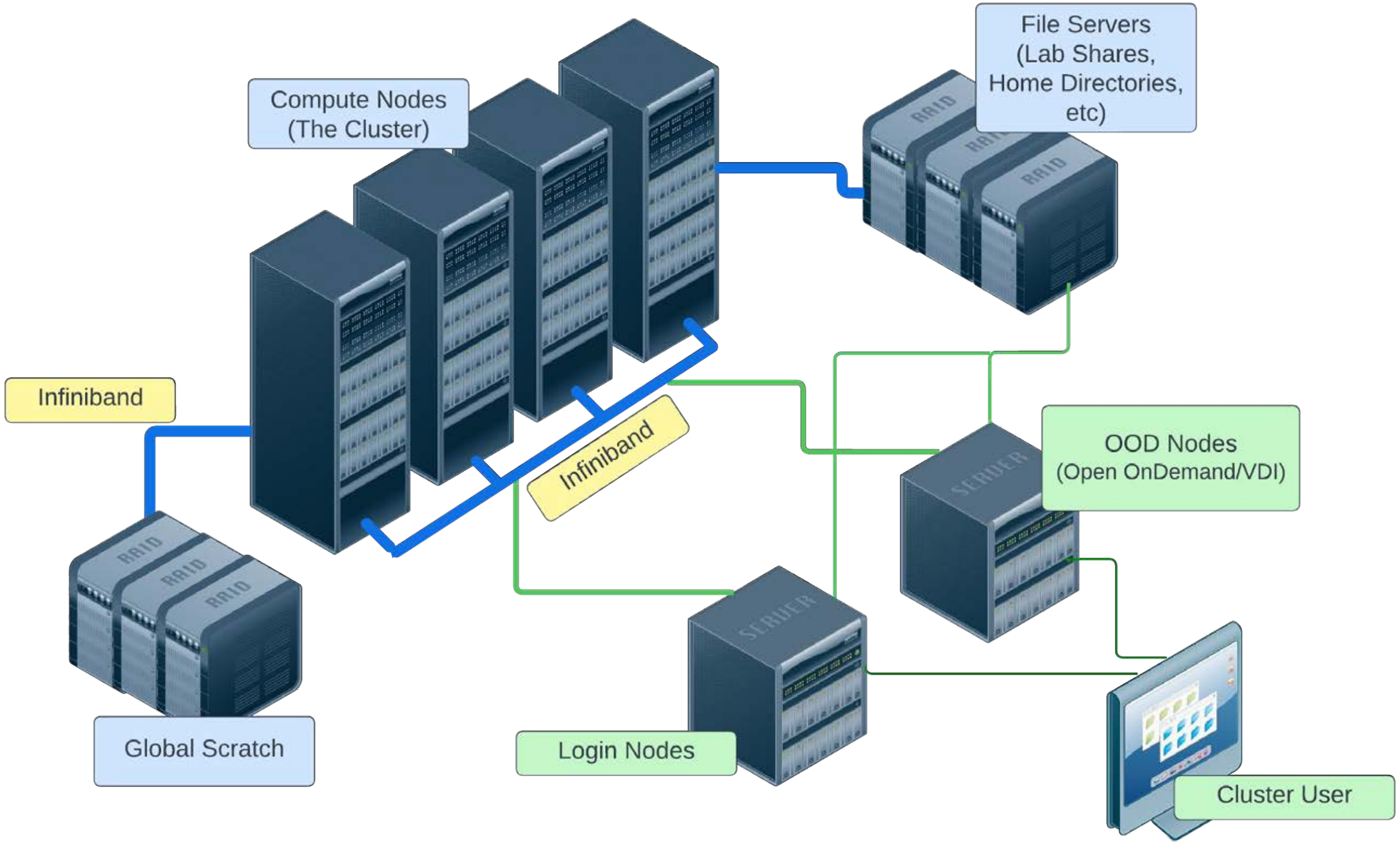
FASSE

- FAS Secure Environment https://docs.rc.fas.harvard.edu/kb/fasse/
- 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
- Information Security and Data Privacy
- DUA: https://docs.rc.fas.harvard.edu/kb/data-use-agreements/

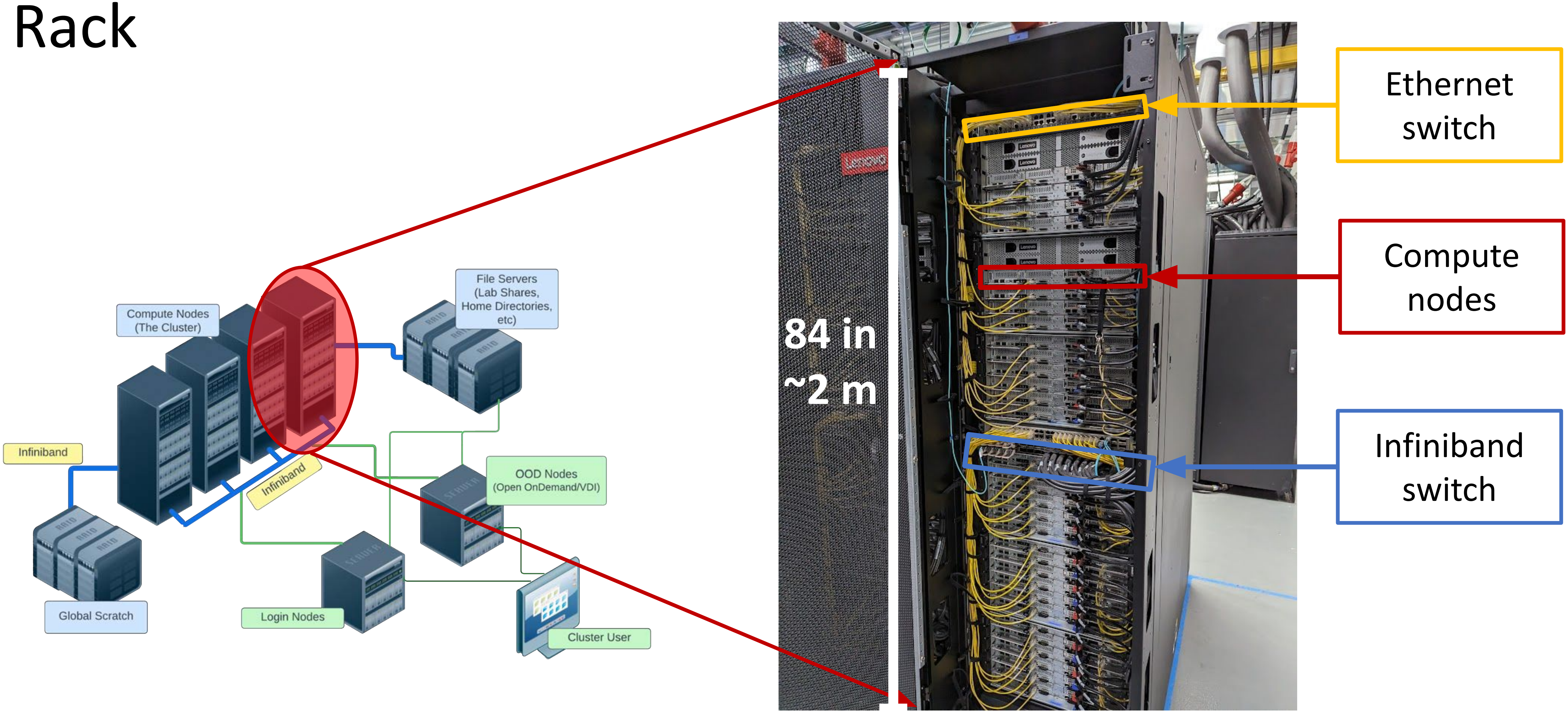


PUBLIC	Public information (Level 1)	Level 1 Harvard Systems
LOW	Low Risk information (Level 2) is information the University has chosen to keep confidential but the disclosure of which would not cause material harm.	Low Risk Systems (L2)
MEDIUM	Medium Risk information (Level 3) could cause risk of material 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



Compute Nodes (The Cluster)

File Servers (Lab Shares, Home Directories, etc)

Infiniband

Infiniband

OOD Nodes (Open OnDemand/VDI)

Global Scratch

Login Nodes

Cluster User

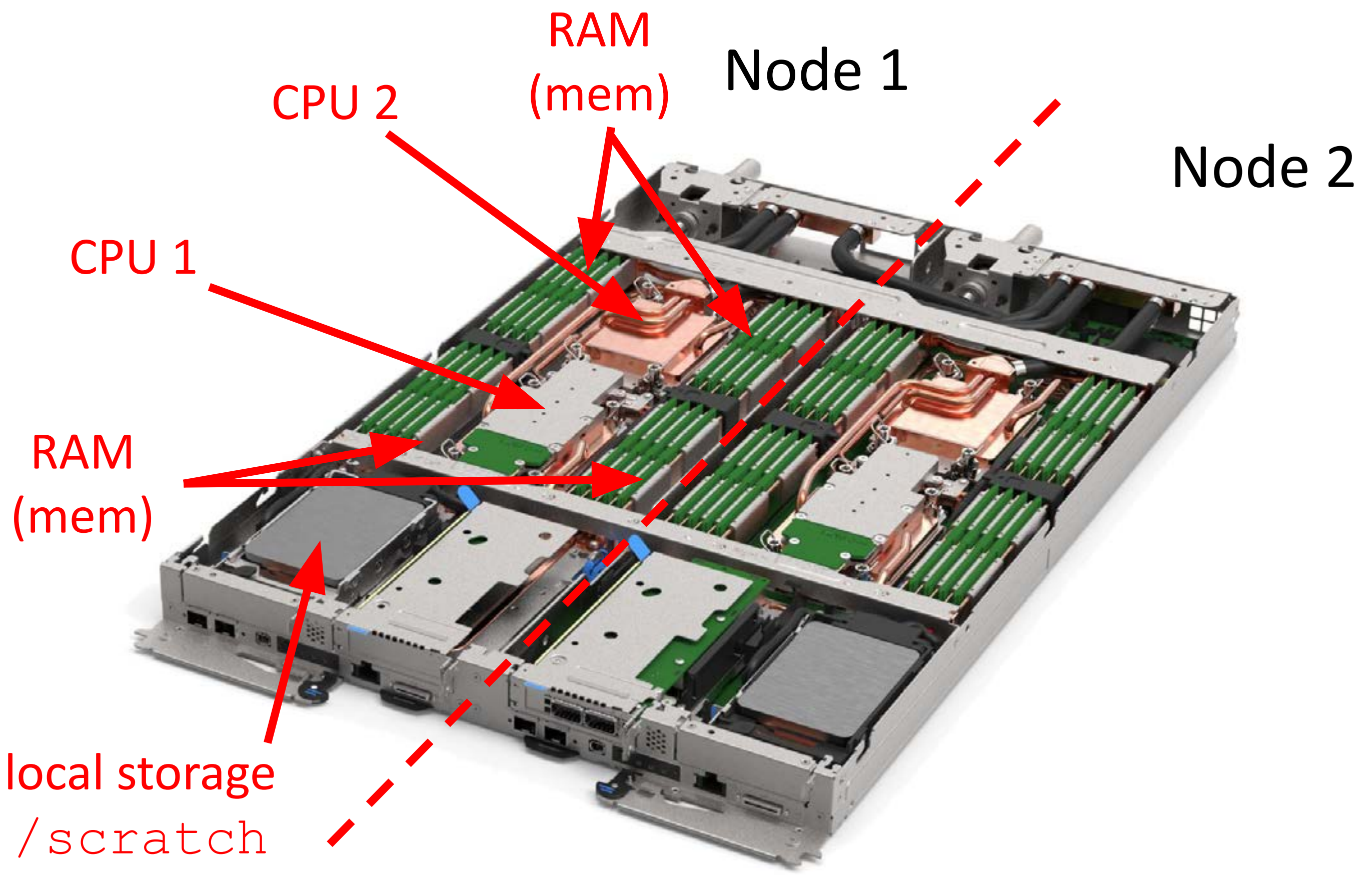
84 in
~2 m

Ethernet switch

Compute nodes

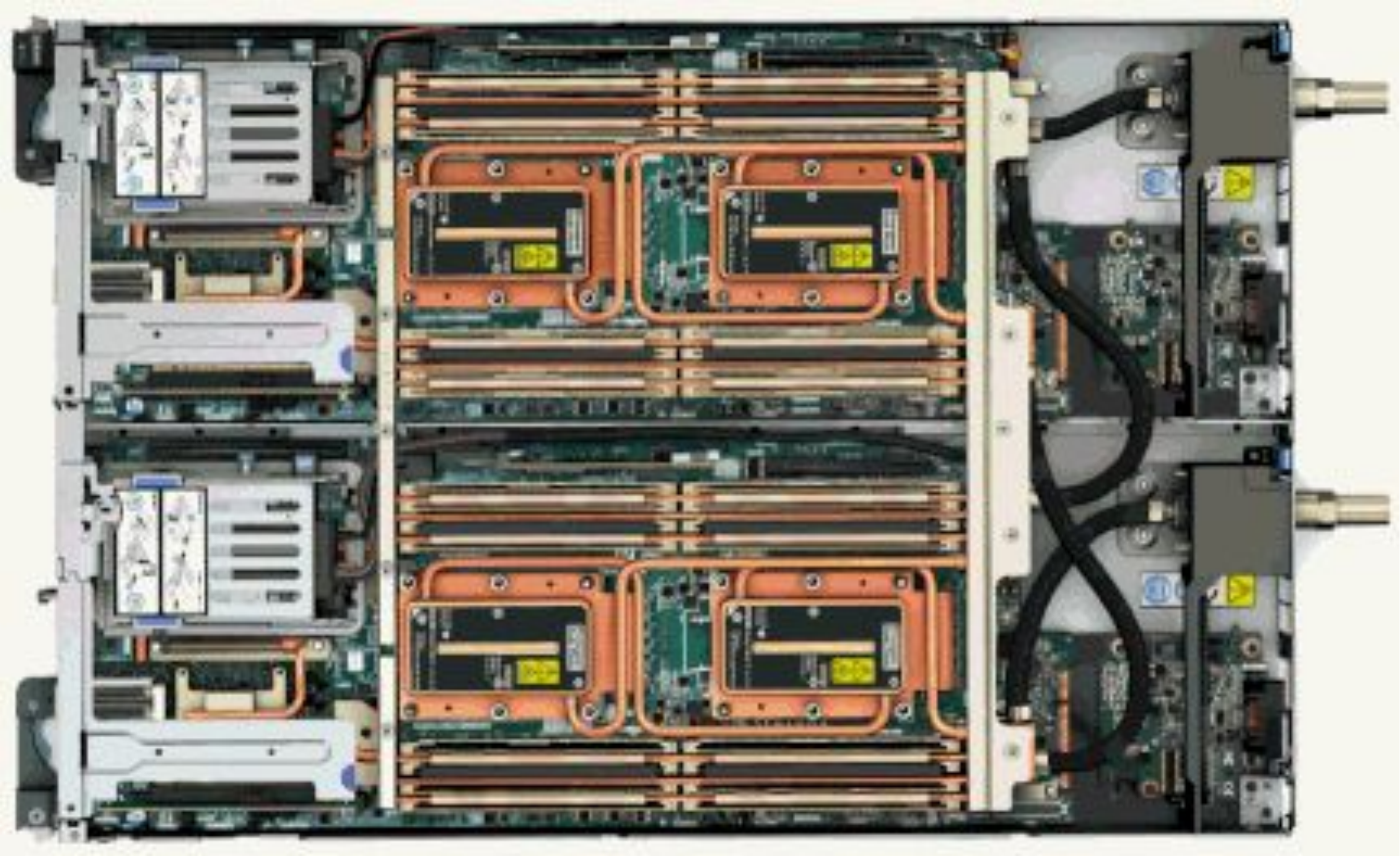
Infiniband switch

CPU Node components



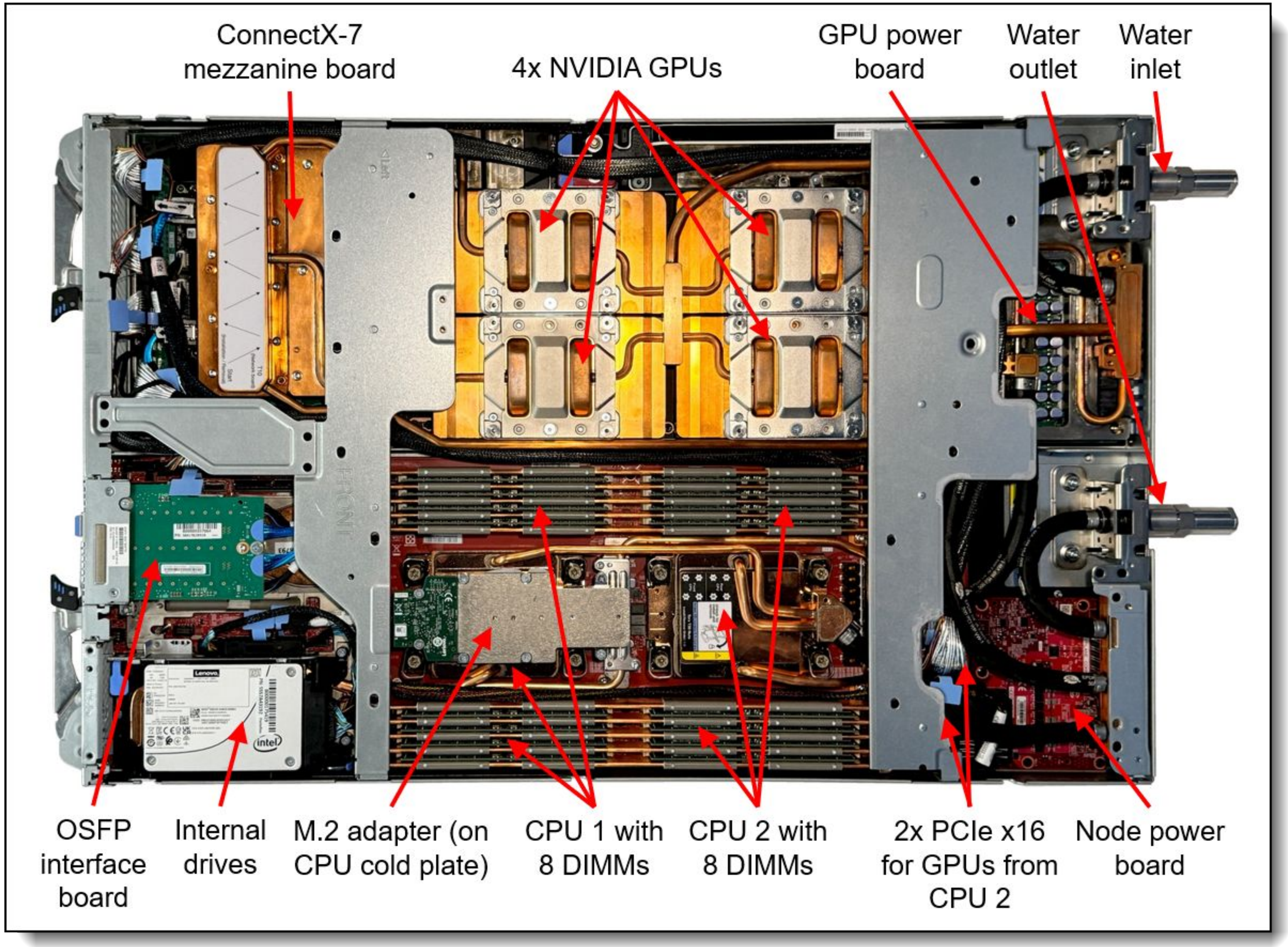
From https://lenovopress.lenovo.com/lp1603-thinksystem-sd650-v3-server

CPU nodes and water cooling



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

GPU node


ConnectX-7 mezzanine board
4x NVIDIA GPUs
GPU power board
Water outlet
Water inlet
OSFP interface board
Internal drives
M.2 adapter (on CPU cold plate)
CPU 1 with 8 DIMMs
CPU 2 with 8 DIMMs
2x PCIe x16 for GPUs from CPU 2
Node power board

Node, processors, core

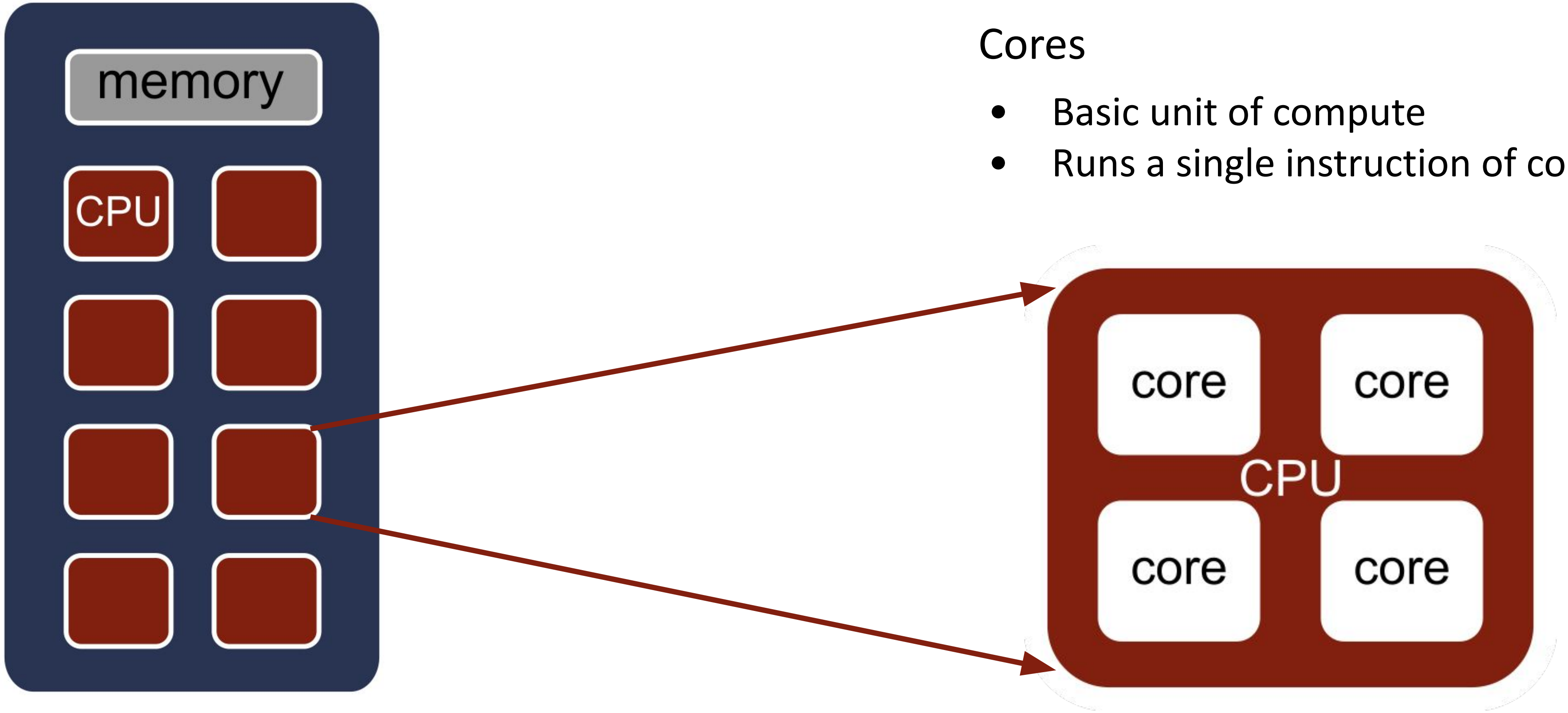
Node: a computer in the cluster

CPU

- Central processing unit, processor
- Can have many cores

Cores

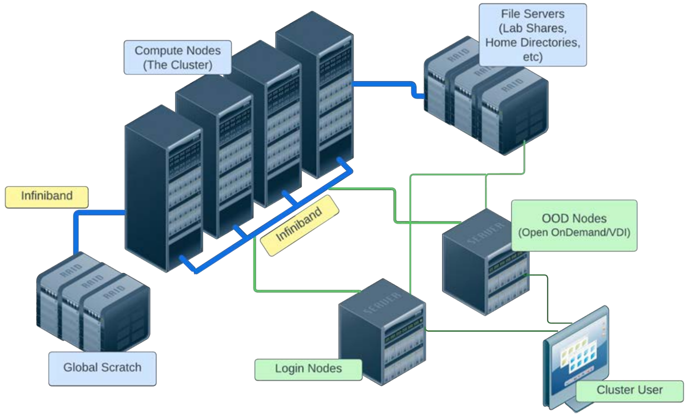
- Basic unit of compute
- Runs a single instruction of code



Nomenclature summary

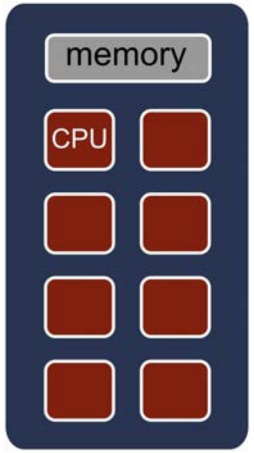
Cluster

Top level unit of a supercomputer


File Servers (Lab Shares, Home Directories, etc)
Compute Nodes (The Cluster)
Infiniband
Infiniband
OOD Nodes (Open OnDemand/VDI)
Global Scratch
Login Nodes
Cluster User

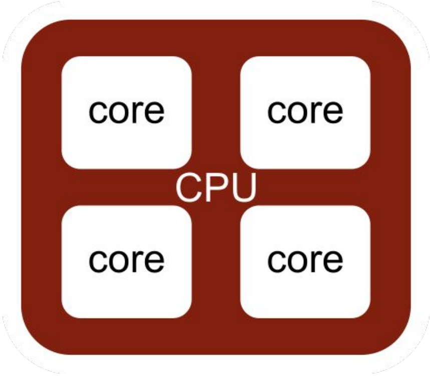
Node

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

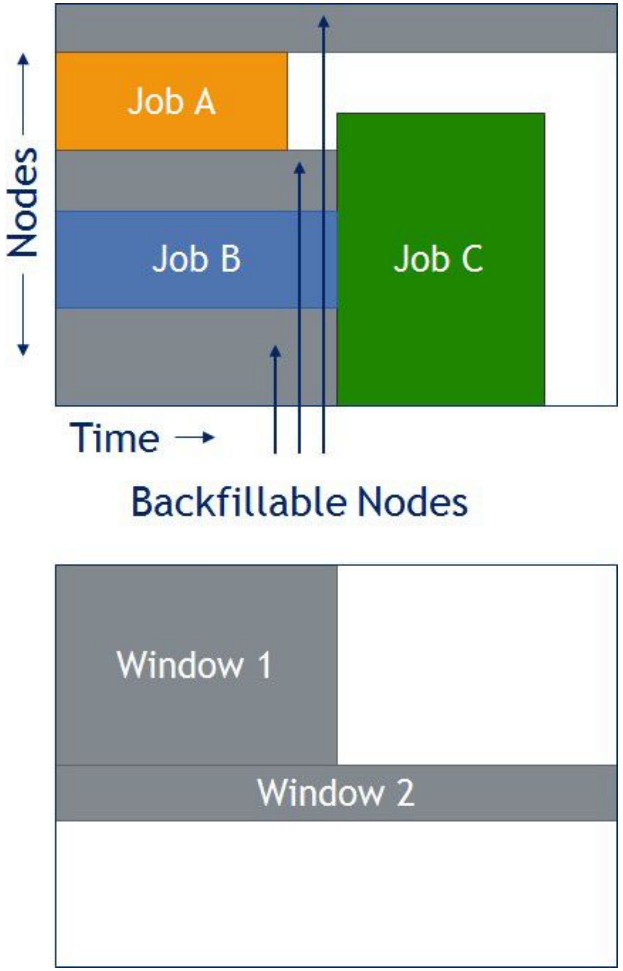
Adapted from HPC@LSU training (http://www.hpc.lsu.edu/training/weekly-materials/2021-Summer/HPC_UserEnv_2021_Summer_session_2.pdf)

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
  - Group Fairshare
  - Amount of time pending

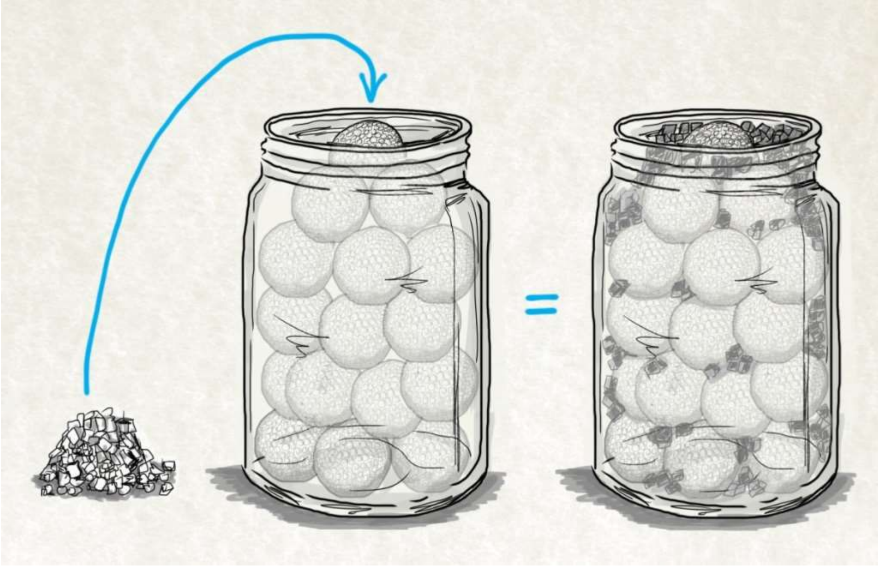


Adapted from HPC@LSU training
(http://www.hpc.lsu.edu/training/weekly-materials/2021-Summer/HPC_UserEnv_2021_Summer_session_2.pdf)

How to maximize cluster usage?

1. Fill in high-priority jobs

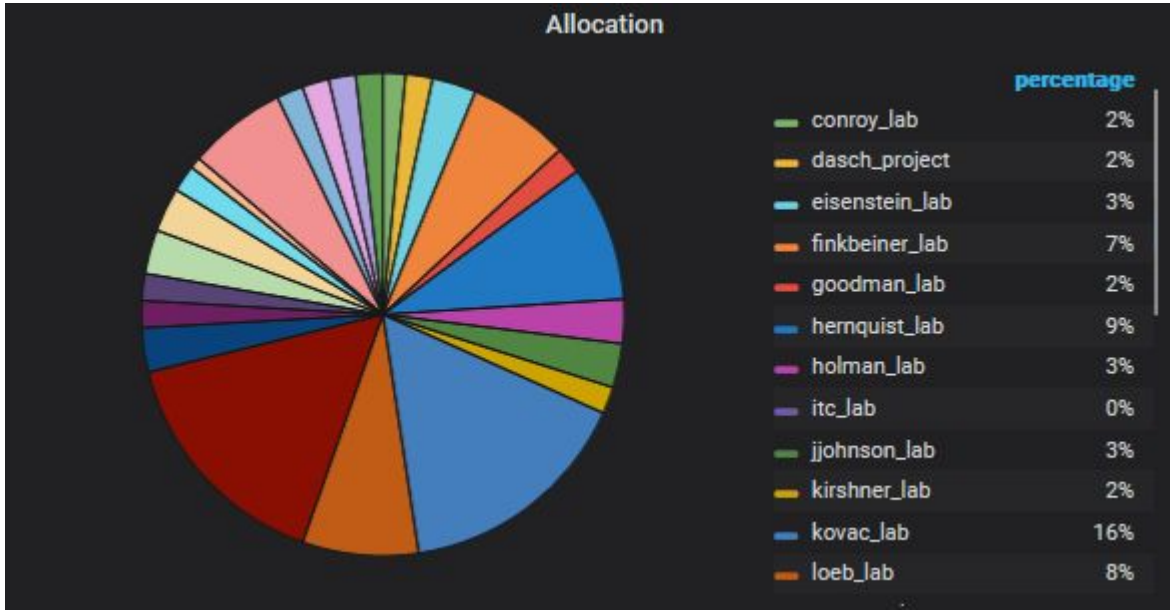
2. Backfill with low-priority jobs



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

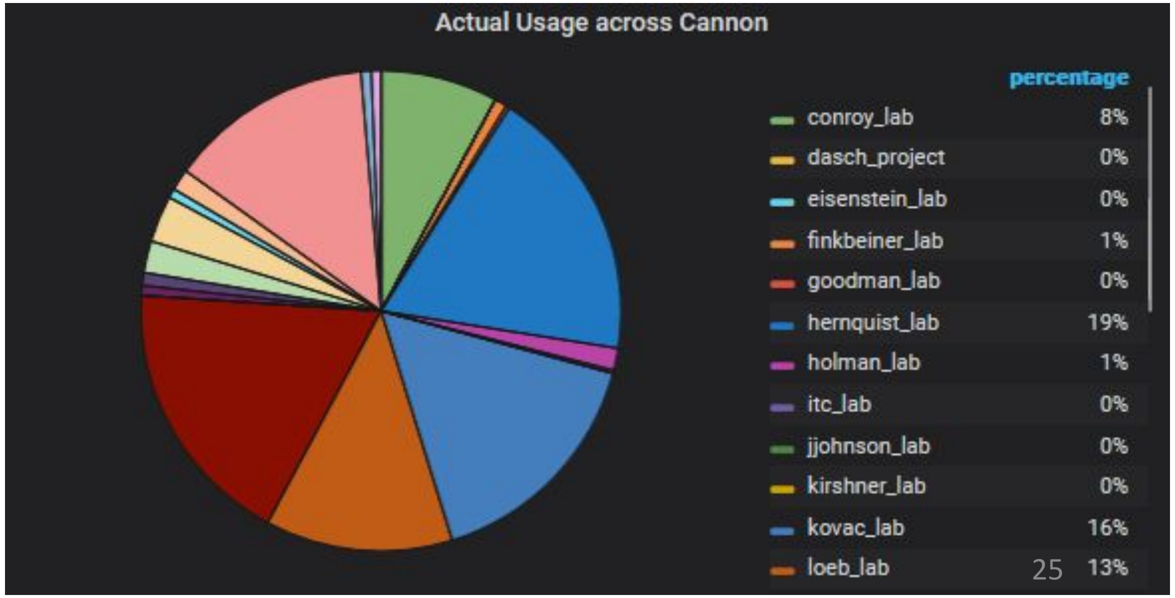
Fairshare

- A method for ensuring the equitable use of a cluster
- The fraction of the cluster a user/group gets
- The score assigned by Slurm to a user/group based on usage
- Priority that users/groups get based on usage



Allocation

	percentage
conroy_lab	2%
dasch_project	2%
eisenstein_lab	3%
finkbeiner_lab	7%
goodman_lab	2%
hernquist_lab	9%
holman_lab	3%
itc_lab	0%
jjohnson_lab	3%
kirshner_lab	2%
kovac_lab	16%
loeb_lab	8%



Actual Usage across Cannon

	percentage
conroy_lab	8%
dasch_project	0%
eisenstein_lab	0%
finkbeiner_lab	1%
goodman_lab	0%
hernquist_lab	19%
holman_lab	1%
itc_lab	0%
jjohnson_lab	0%
kirshner_lab	0%
kovac_lab	16%
loeb_lab	13%

General fairshare principles

- Fairshare is affected by how much cpu, memory, GPU, and time you request in order to run your calculations.
- GPUs eat up fairshare 200x-500x as fast as CPUs
- Your usage affects every person in your group
- Using either the test or the gpu_test partition does not affect fairshare, but is limited in terms of time and the number of jobs you can submit
- Leaving a session running in OOD consumes resources. You should cancel it if you're not still using it.

Things you can do to decrease your impact on fairshare

- Use fewer GPUs. They are 200x-500x more "expensive" than CPUs in terms of fairshare.
- Request less memory.
- Request fewer CPUs.
- Run your jobs for a shorter time period.
- Leaving a session running in OOD consumes resources. You should cancel it if you're not still using it.
- Wait. Coordinate jobs with lab members. Space jobs accordingly.

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 (~1h)
  - Increase size by 1.5, 2.0, 2.5x and check how job scales
  - Then you can have a rough estimation of how much a first trial production job of ~10x would require

Cannon “test” partitions

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

Partitions	test	gpu_test
Time Limit	12 h	12 h
# Nodes	18	14
# Cores / Node / GPU	112	112 + 8 A100 MIG
Memory / Node (GB)	990	487

Cannon “cpu” partitions

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

Partitions	sapphire	shared	intermediate	unrestricted
Time Limit	3 days	3 days	3-14 days	14+ days - none
# Nodes	186	310	12	8
# Cores / Node	112	48	112	48
Memory / Node (GB)	990	184	990	184

Cannon “gpu” partitions

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

Partitions	gpu	gpu_h200
Time Limit	3 days	3 days
# Nodes	36	24
# Cores / Node / GPU	64 + 4 A100	112 + 4 H200
Memory / Node (GB)	990	990

Cannon “big memory” partitions

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

Partitions	bigmem	bigmem_intermediate
Time Limit	3 days	3 days
# Nodes	4	3
# Cores / Node	112	64
Memory / Node (GB)	1988	2000

Cannon HSPH partitions

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

Partitions	hsph	hsph_gpu
Time Limit	3 days	3 days
# Nodes	56	2
# Cores / Node / GPU	112	96 + 4 H100
Memory / Node (GB)	990	1500

Cannon “requeue” partitions

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

Partitions	serial_requeue	gpu_requeue
Time Limit	3 days	3 days
# Nodes	varies	varies
# Cores / Node	varies	varies
Memory / Node (GB)	varies	varies

Cannon “other” partitions

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

Partitions	remoteviz	PI_lab
Time Limit	3 days	varies
# Nodes	down	varies
# Cores / Node / GPU	32 + V100 for rendering	varies
Memory / Node (GB)	373	varies

FASSE partitions

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

Partitions	test	fasse	serial_requeue	fasse_gpu	fasse_bigmem	fasse_ultramem	remoteviz	**pi_lab**
Time Limit	12 h	7 days	7 days	7 days	7 days	7 days	7 days	**varies**
# Nodes	5	42	varies	4	17	1	1	**varies**
# Cores / Node	48	48	varies	64 + 4 A100	64	64	32	**varies**
Memory / Node (GB)	184	184	varies	487	499	2000	373	**varies**

cpu

gpu

memory

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

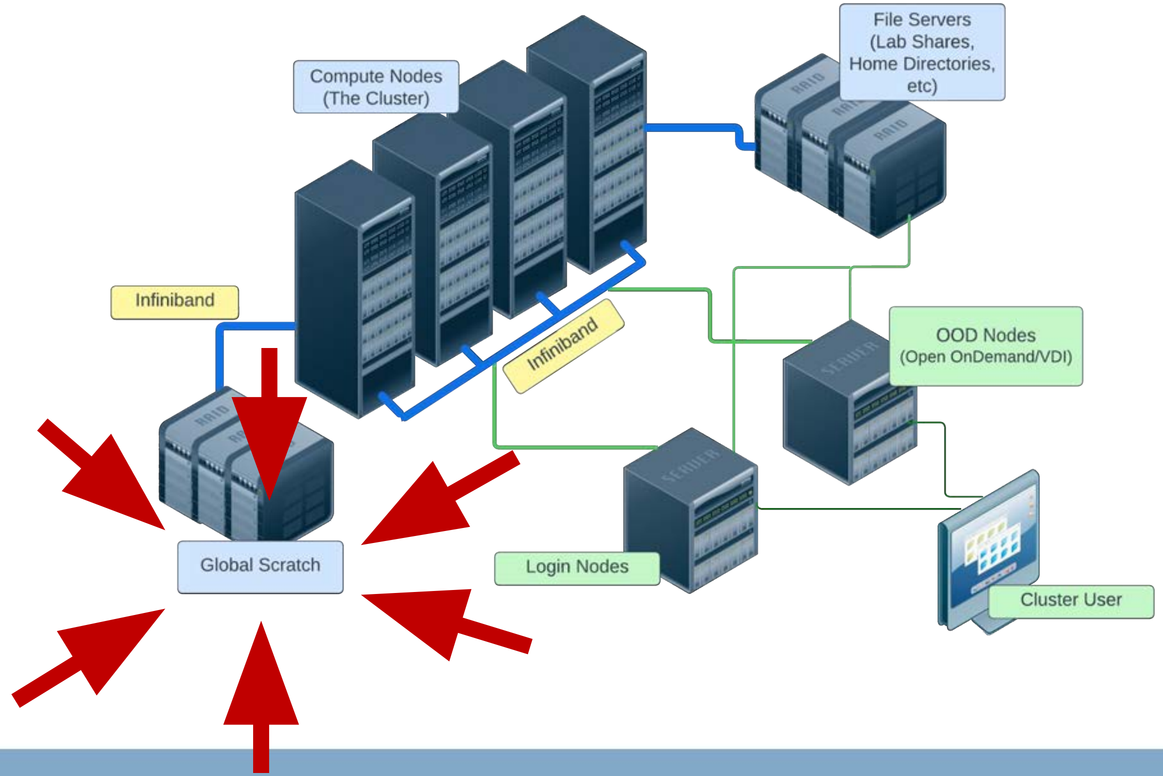
Tier storage documentation: https://www.rc.fas.harvard.edu/services/data-storage/

	Home Directories	Lab Directory (Startup)	Local Scratch	Global Scratch	Tier Storage
Mount Point	$HOME /n/home#/$USER /n/home_fasse/$USER	/n/holylabs/pi_lab	/scratch	$SCRATCH /n/netscratch/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

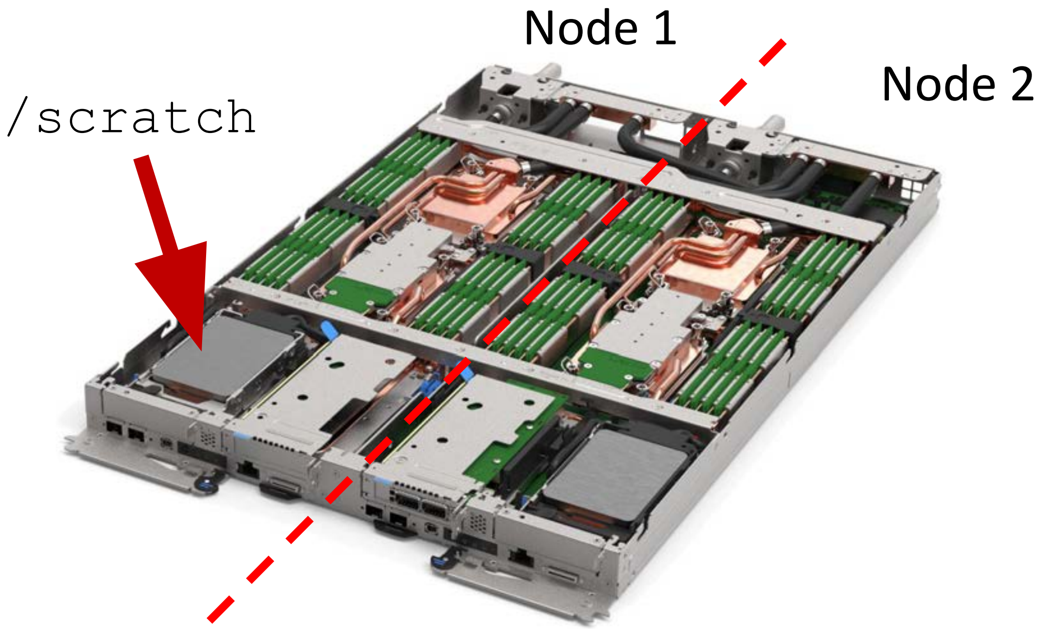
Global Scratch

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


Compute Nodes (The Cluster)
File Servers (Lab Shares, Home Directories, etc)
Infiniband
Infiniband
OOD Nodes (Open OnDemand/VDI)
Global Scratch
Login Nodes
Cluster User

Local Scratch

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

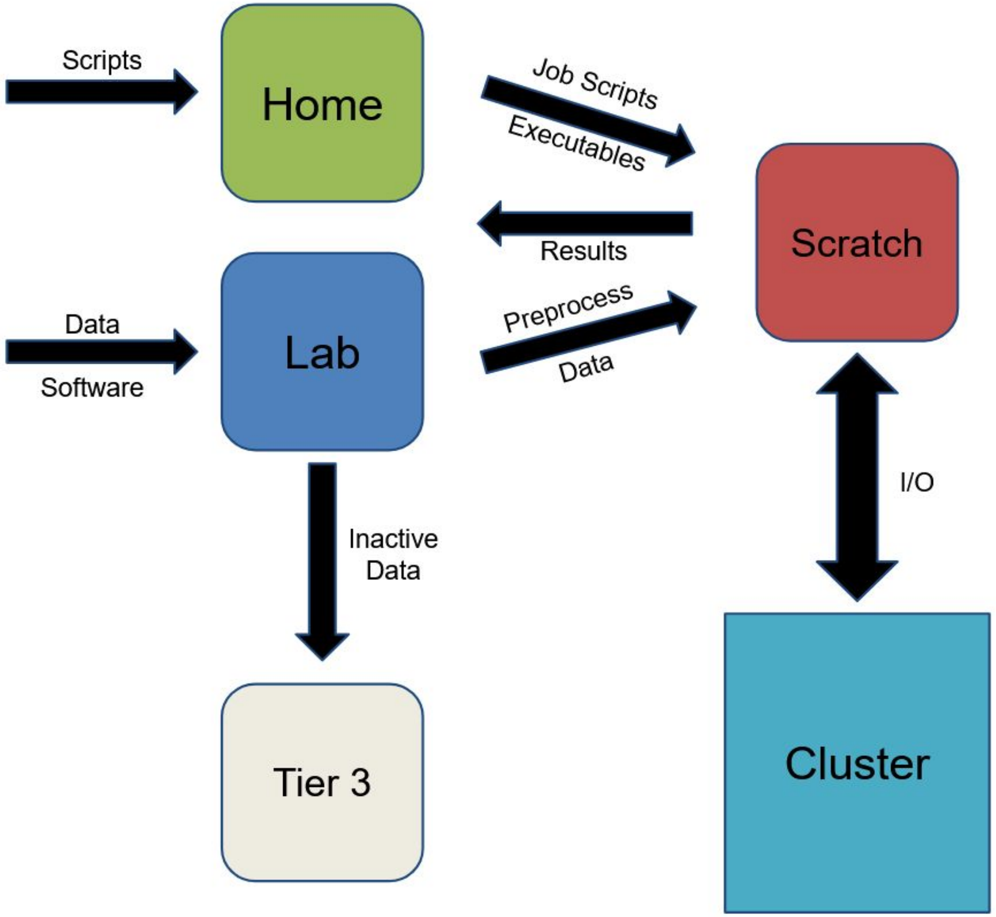


From https://lenovopress.lenovo.com/lp1603-thinksystem-sd650-v3-server

Data management

Documentation:
https://docs.rc.fas.harvard.edu/kb/data-storage-workflow-rdm/

- 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)!!
- Specific training about Research Data Management at FASRC - check out Training Calendar!



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: 10,100 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 `$SCRATCH`
- 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, please 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.”*

Training session evaluation

Please, fill out our training session evaluation. Your feedback is essential for us to improve our trainings!!

https://tinyurl.com/FASRC-training

FASRC documentation

- FASRC docs: https://docs.rc.fas.harvard.edu/
  - If searching on Google add FASRC to your search
- GitHub User_Codes: https://github.com/fasrc/User_Codes/
- Getting help
  - Office hours: https://www.rc.fas.harvard.edu/training/office-hours/
  - Ticket: send email to rchelp@rc.fas.harvard.edu
    - include as much detail as possible
    - please send screenshots, full pathnames, refer to previous relevant tickets, etc.

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)

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

Thank you :)

FAS Research Computing