Introduction to High-Performance Computing (HPC)

HPC cluster

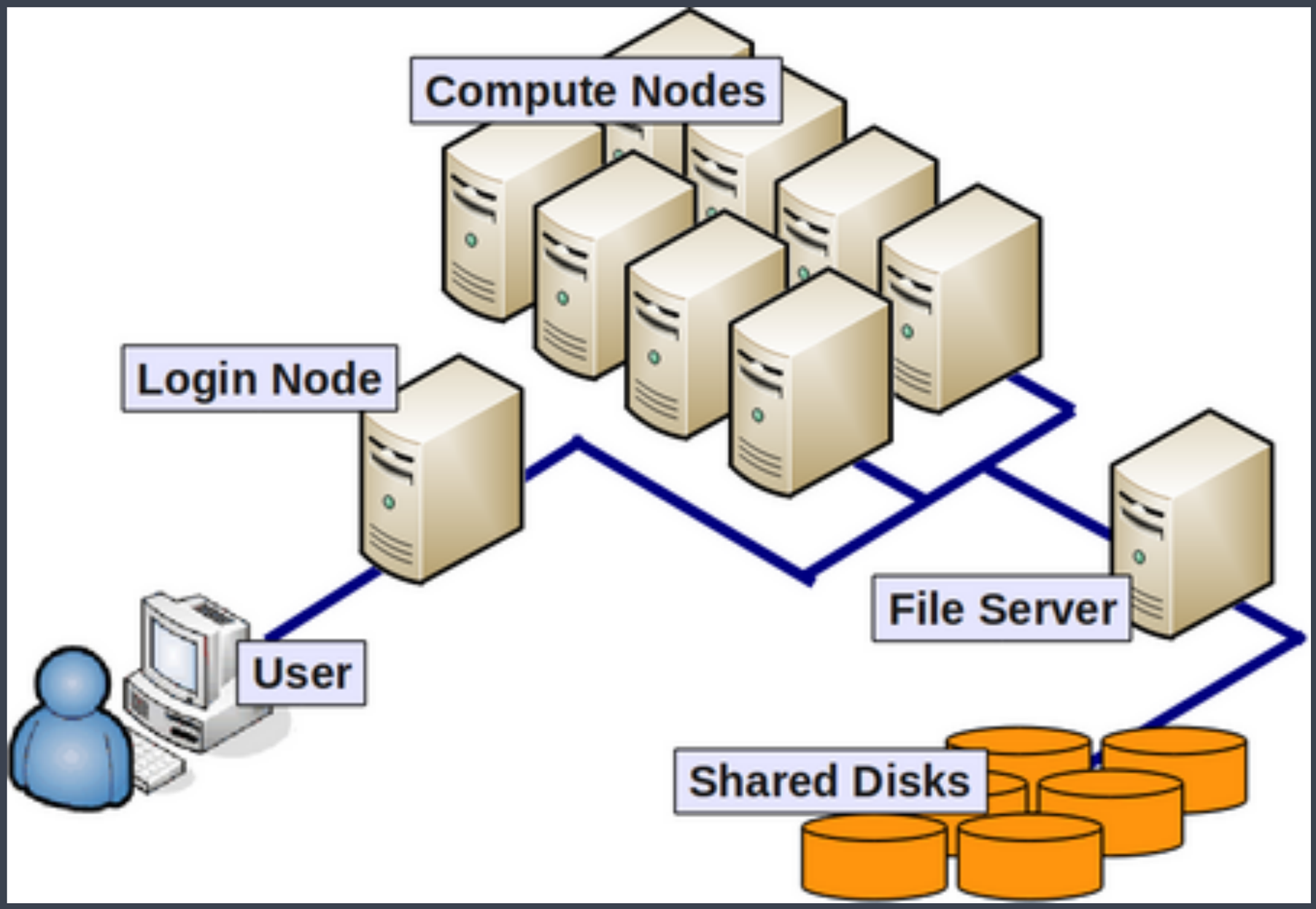
“High Performance Computing most generally refers to the practice of aggregating computing power in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation in order to solve large problems in science, engineering, or business.”

http://insidehpc.com/hpc-basic-training/what-is-hpc/

Need for HPC cluster?

- Resources: Your computer does not have the resources to run the desired NGS analysis
- Speed: You want to produce results faster than your computer can
- Software: You need software that is unavailable or unusable on your computer

HPC cluster



Need for HPC cluster?

- Resources: Your computer does not have the resources to run the desired NGS analysis
- Speed: You want to produce results faster than your computer can
- Software: You need software that is unavailable or unusable on your computer

HPC cluster components

Nodes: Individual computers in the cluster

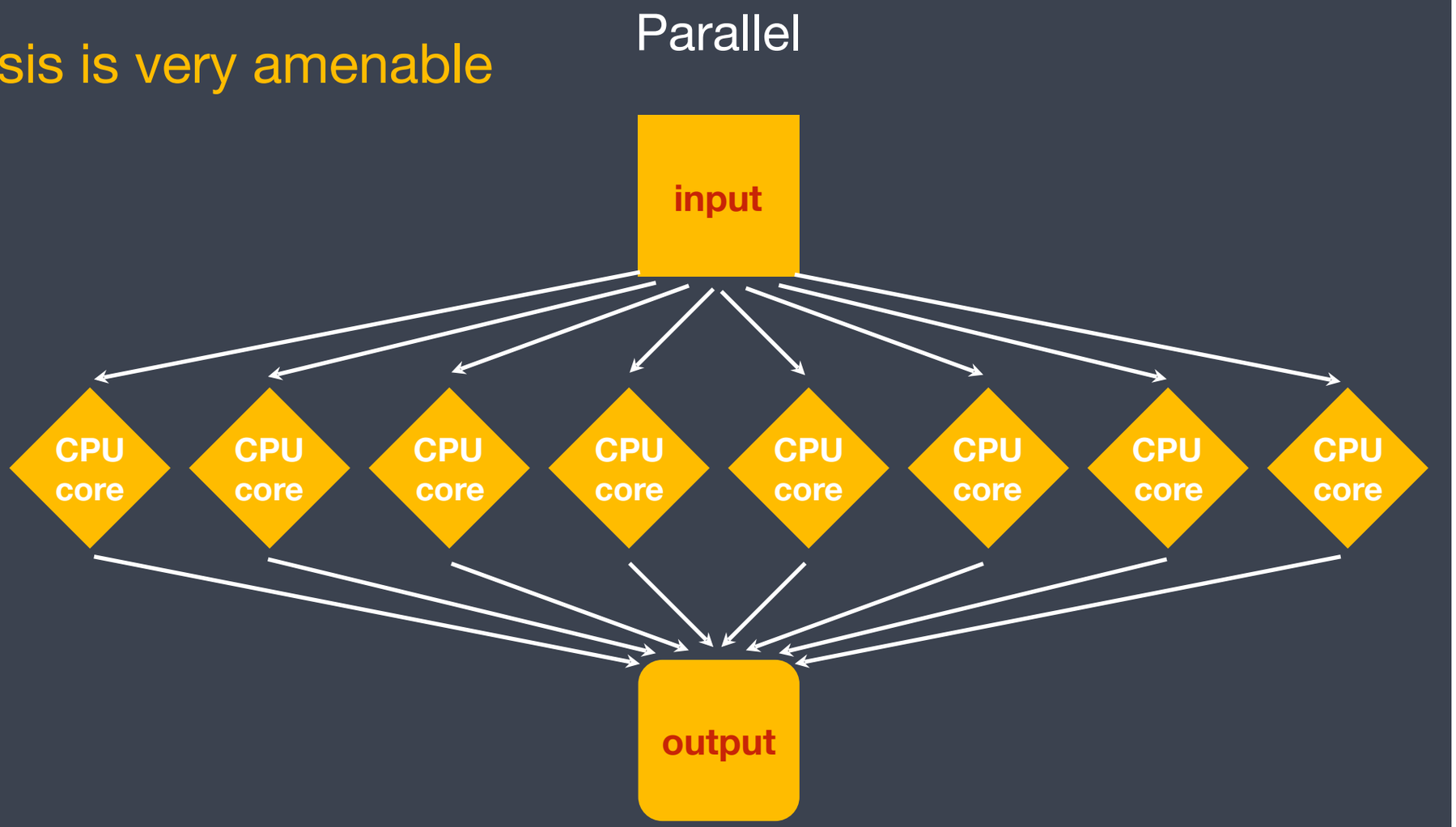
Cores: individual processing units available within each CPU of each Node

e.g. a “Node” with eight “quad”-core CPUs = 32 cores for that node.

Shared disk: storage that can be shared (and accessed) by all nodes

Multi-threading

NGS data analysis is very amenable to this strategy.



Faster and more efficient

HPC cluster

- multi-user, shared resource
- lots of nodes = lots of processing capacity + lots of memory
- a system like this requires constant maintenance and upkeep.

Orchestra
HIGH PERFORMANCE COMPUTING CLUSTER

Please see our Orchestra status page for known issues.

The **Orchestra** platform provides UNIX-based high performance computing, web hosting, and database hosting services at Harvard Medical School.

Orchestra and its associated services are managed by the **Research Computing Group**, part of the HMS Information Technology Department.

Please submit a request via the RC web site for help with Orchestra or feedback. You can also subscribe to the mailing list for Orchestra users.

Wiki page: https://wiki.med.harvard.edu/Orchestra

Introduction to High Performance Computing and Orchestra

HMS Research Computing
Spring 2016

What is Orchestra?

- Tech specs:
  - Over 550 compute nodes
  - Over 7,500 cores
  - Over 42TB RAM
- CentOS 6 Linux
- LSF scheduler
- Total 28+PB storage

Using Orchestra!

1. Logging in to remote machines (securely)

- When logging in we used the “ssh” command,
  ssh stands for Secure SHell

- **ssh** is a protocol for data transfer that is secure, i.e the data is encrypted as it travels between your computer and the cluster (remote computer)

- Commonly used commands that use the **ssh** protocol for data transfer are, **scp** and **sftp**

2. Using & installing software

Using Software Environment Modules

- Most “software” on Orchestra is installed as an environment module.
- An environment module makes the necessary modifications in $PATH to make sure the program runs without any issues.
- Allows for clean, easy loading, including most dependencies, and switching versions.

Using Software Environment Modules

```
$ module avail
$ module avail seq/
$ module avail stats/
$ module avail seq/bowtie/
```

What software environment modules are available on Orchestra?

- Loading modules

```
$ module load seq/bowtie/2.0.6
```

- See all modules that have been loaded

```
$ module list
```

- Unloading modules

```
$ module unload seq/bowtie/2.0.6
```

3. The Job Scheduler, LSF

LSF: Fair Sharing

- **L**oad **S**haring **F**acility: distributes jobs across Orchestra fairly
- Ensures that no single user or core monopolizes Orchestra
- Users are assigned dynamic priorities
- Queues also have priorities
- Submitting lots of jobs reduces your fairshare priority
- Even if many jobs are pending, your jobs will start quickly provided you have not submitted many jobs

Choosing the proper resources for your job with the appropriate `bsub` options

The “bsub”

```
$ bsub –q queue –W hr:min job/command
```

- Necessary to specify:
  - -q (queue)
  - -W (runtime in hr:min)

bsub options

```
-W 3:00          #time requested
-q multicore     #name of the queue
-n 4             #number of cores requested
-R “rusage[mem=8000]”   #memory requested in MB
-J jobname
-a openmpi       #type of mpi run
-Is              #interactive shell
-e %J.err        #send  errors to file
-o %J.out        #send screen output to file
-N               #notify when job completes
```

Runtime Limit (`bsub -W`)

- In hours:minutes
- Runtimes are subject to the maximum time permitted per queue (see table)
- If your job exceeds your runtime, your job will be killed ☹
- Running many jobs that finish quickly (less than a few minutes) is suboptimal and may result in job suspension, contact RC to learn how to batch jobs

Multithreading (`bsub -n`)

- A single CPU can execute multiple processes (threads) concurrently
- “-n” indicates how many cores are requested
- Jobs that are overefficient (use more cores than reserved) jeopardize the health of a node
- Reserve the same amount of cores in your job and your bsub!

CPU Limit

- Amount of seconds the cluster works on your job (calculated by LSF)
- Ncores * Runlimit (-n * -W)
- Common error:

  bsub –q short –W 8:00 tophat –p 8

  tophat asks for 8 cores (-p 8) but only 1 core requested (no –n), job killed in 1 hour (8/8)

Reserving Memory (`bsub -R`)

- Most nodes have 90GB memory available over all cores, some have more
- Make a resource request with

  -R “rusage[mem=8000]” (memory requested in MB)
- Memory multiplies by cores requested, so

  -n 4 –R “rusage[mem=16000]” reserves 64GB memory
- Asking for more memory may cause jobs to pend longer
- TERM_MEMLIMIT errors indicate that not enough memory was reserved

Shared Queues (`bsub -q`)

- ***mpi*** queue if you have an MPI parallel job
- ***priority*** queue if you have just one or two jobs to run
- ***mcore*** queue if you have multi-core jobs to run.
- ***short*** queue if your jobs will take less than 12 hours to run.
- Else: ***long*** queue.

https://wiki.med.harvard.edu/Orchestra/ChoosingAQueue

Job submission scripts

Shell Scripts: The Basics

```
#! /bin/sh              #always at the top


#commands with options
tophat -p 4 -o ./mytophatdir1 hg19 file1_1.fastq file1_2.fastq
tophat -p 4 -o ./mytophatdir2 hg19 file2_1.fastq file2_2.fastq
```

Save as **`myshellscript.sh`**

Run this script in an interactive session as **`sh myshellscript.sh`**

OR

Submit it as a job to the LSF scheduler

```
$ bsub -q mcore -W 12:00 -n 4 -N -e %J.err -o %J.out sh myshellscript.sh
```

Creating a Job Submission Script

```
#! /bin/sh
#BSUB -q mcore
#BSUB -W 12:00
#BSUB -o %J.out
#BSUB -e %J.err
#BSUB –N
#BSUB -n 4
#BSUB –R “rusage[mem=12000]”

module load seq/tophat2.1.1

tophat -p 4 -o ./mytophatdir1 hg19 file1_1.fastq file1_2.fastq
```

Save as **myshellscript.lsf**

Run as **$ bsub < myshellscript.lsf**

Orchestra will notify you when the job is done, or if there is an error

Managing jobs and getting information about submitted/running jobs

Monitoring Jobs

- List info about jobs/their status:

```
mfk8$loge:~$ bjobs
```

-r (running jobs)

-p (pending jobs)

-l (command entered, long form)

Terminating Jobs

- Terminate a job (jobid given at submission)

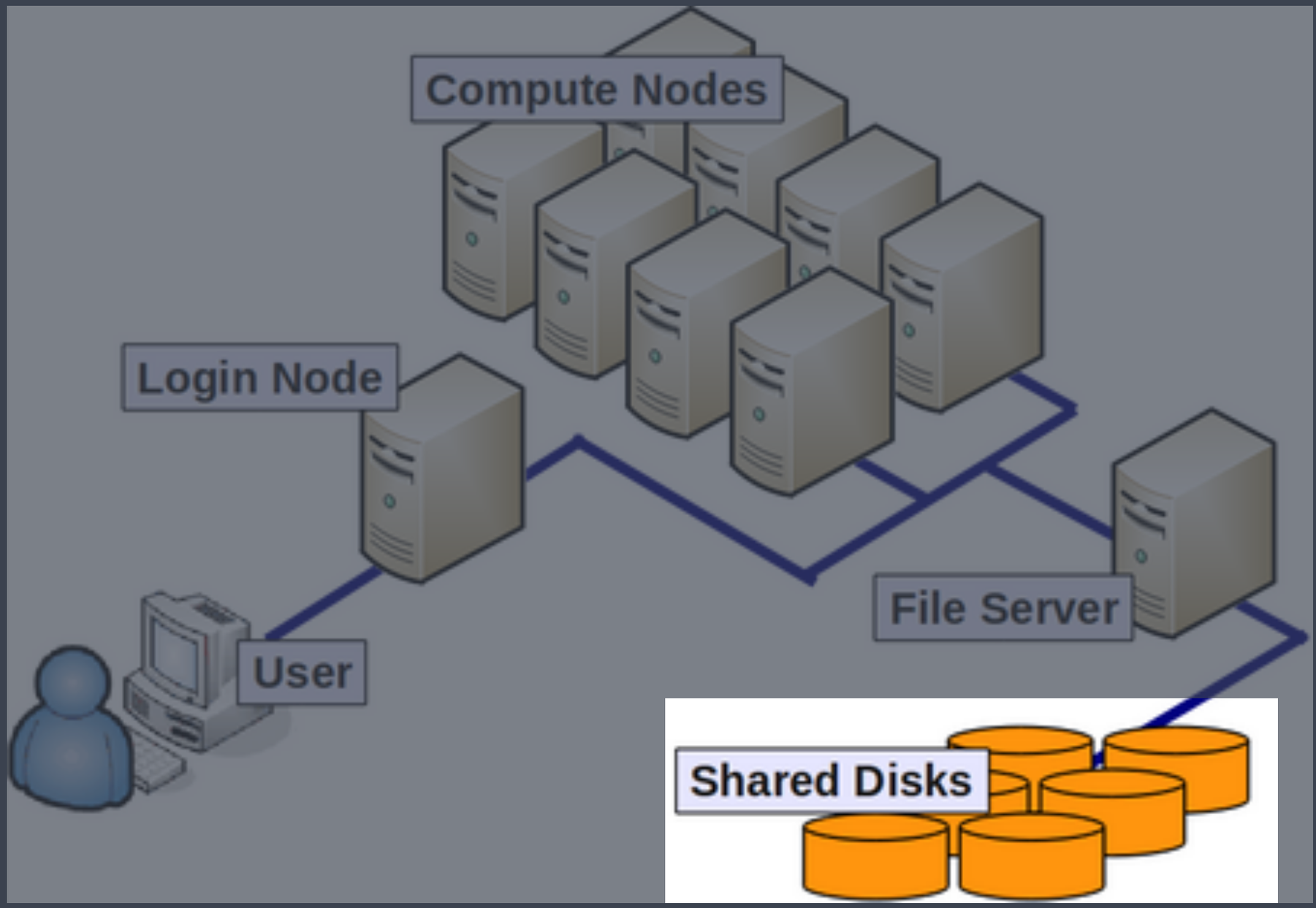
```
mfk8$loge:~$ bkill jobid
```

- Terminate all jobs (be careful!)

```
mfk8$loge:~$ bkill 0
```

4. Filesystems and storage

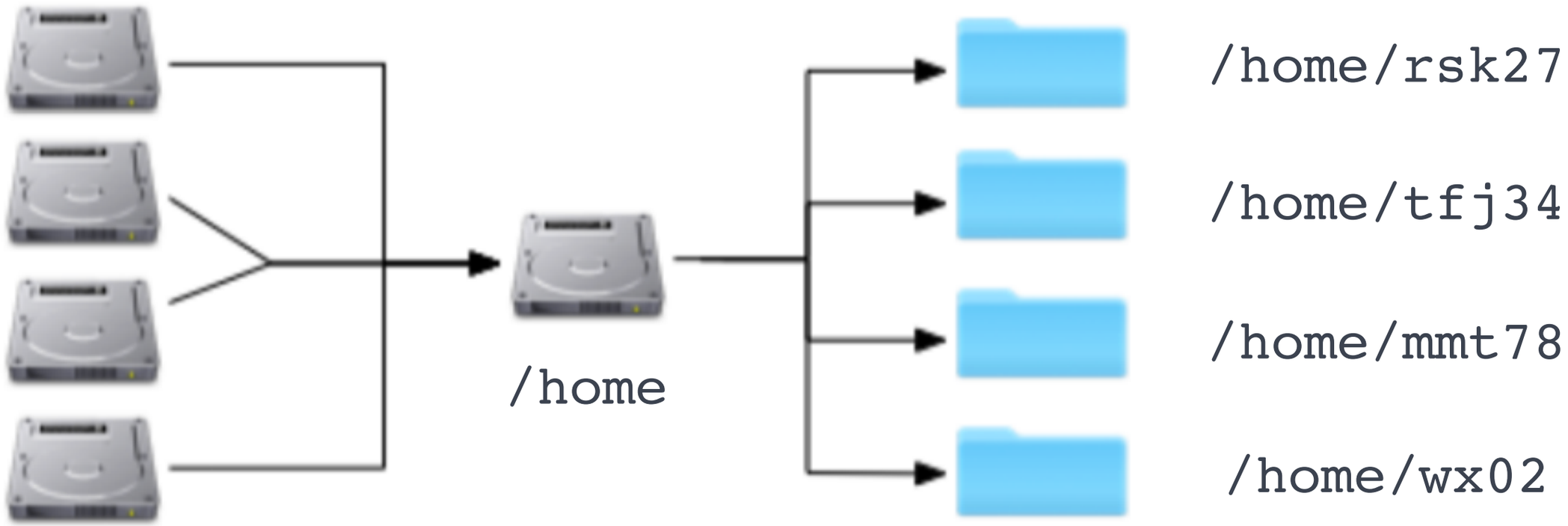
Filesystems and storage



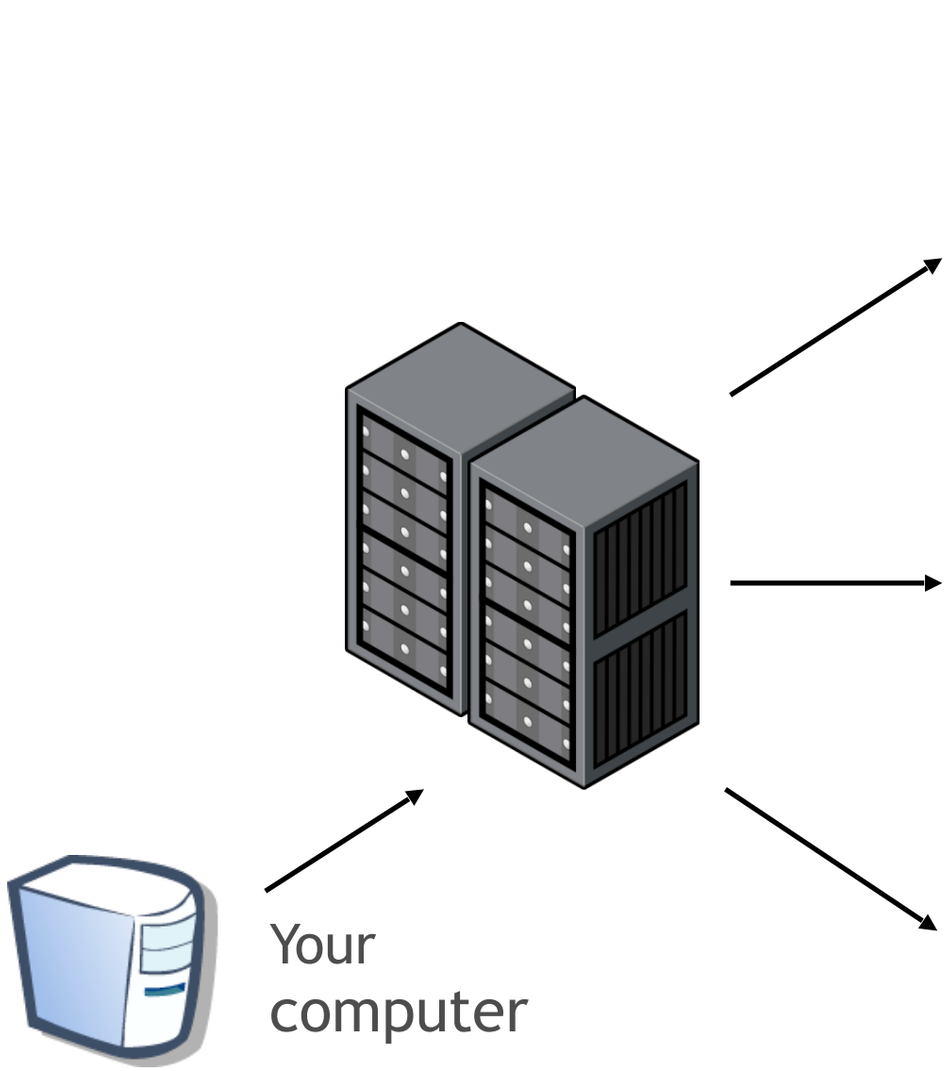
Filesystems and storage

- Storage on HPC systems is organized differently than on your personal machine
- Physical disks are bundled together into a virtual volume; this volume may represent a single filesystem, or may be divided up, or partitioned, into multiple filesystems
- Filesystems are accessed over the internal network

Filesystems and storage



Storage on Orchestra



/home

- /home/user_id
- quota: 100GB per user
- Backup: 14-day snapshots + 60 day weekly snapshots

/groups, /n/data1, /n/data2

- /groups/your_lab/your_dir
- quota: expandable
- Backup: 14-day snapshots + 60-day weekly snapshots

/n/scratch2

- /n/scratch2/user_id
- quota: 5TB per user
- Backups: none
- Run workflows here

Checking storage usage

- To check your storage available:

```
$ quota
```

- Home directory (`/home/username`): you get 100GB.

- Group directories: space varies, can increase.

```
/groups/groupname

/n/data1

/n/data2
```

For more direction:

- https://wiki.med.harvard.edu/Orchestra/NewUserGuide
- http://rc.hms.harvard.edu
- rchelp@hms.harvard.edu
- Office Hours: Wednesdays **1-3p Gordon Hall 500**