Introduction to High-Performance Computing (HPC)

Computer components

CPU : Central Processing Unit

cores : individual processing units within a CPU

Storage : Disk drives

HDD : Hard Disk Drive

SSD : Solid State Drive

Memory : small amount of volatile or temporary information storage

Computer components (my Macbook Pro)

Model Name: MacBook Pro

Number of Processors: 1

Total Number of Cores: 4

Memory: 16 GB

Data storage: 512 GB

"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/

Computer resources required for NGS Data Analysis

100s of cores for processing!

100s of Gigabytes or even Petabytes of storage!

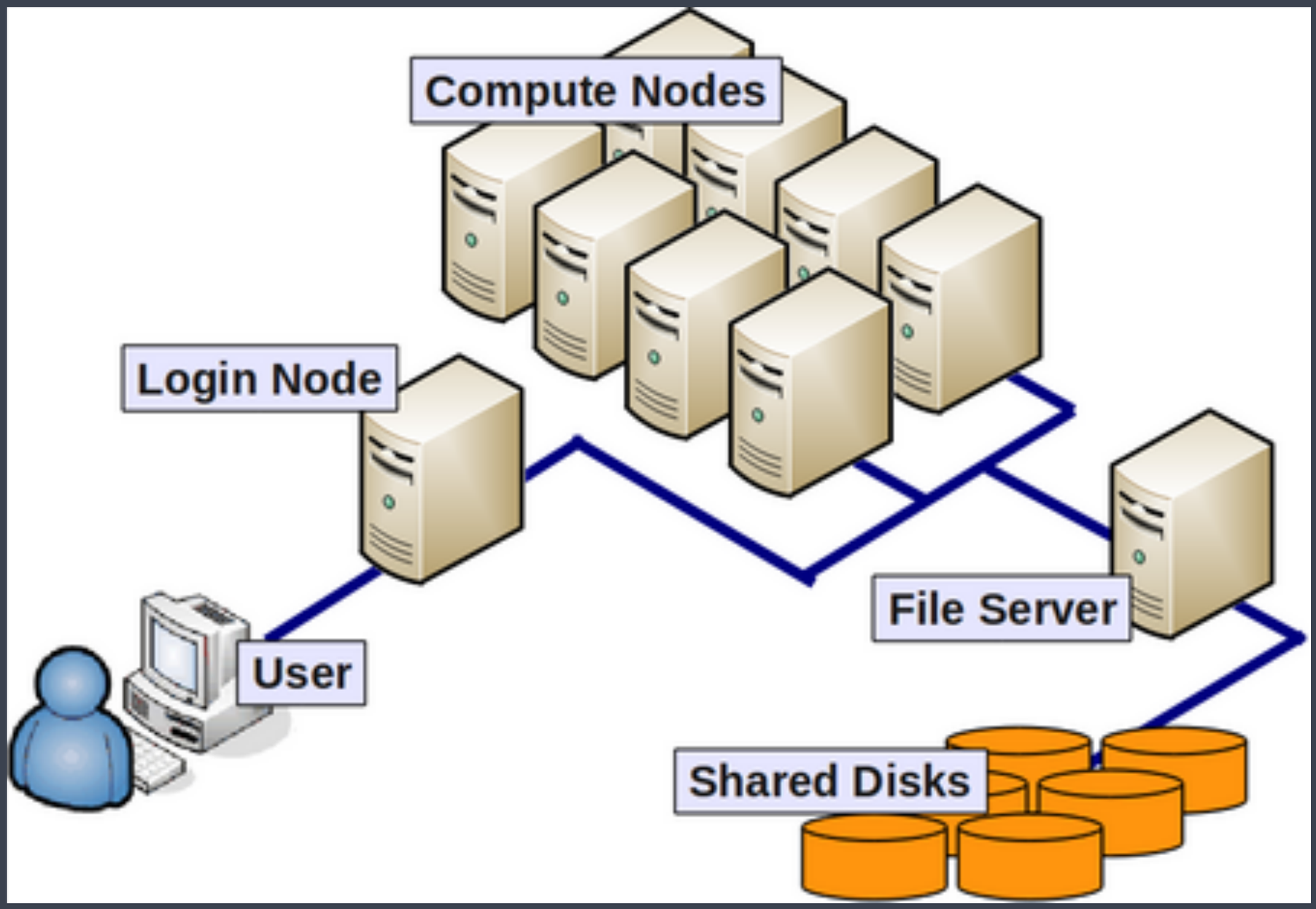
100s of Gigabytes of memory!

High-Performance Computing

Provides all the resources to run the desired Omics analysis in one place.

Provides software that is unavailable or unusable on your computer/local system

HPC cluster structure



HPC cluster components

Nodes: Individual computers in the cluster

Cores (threads): 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

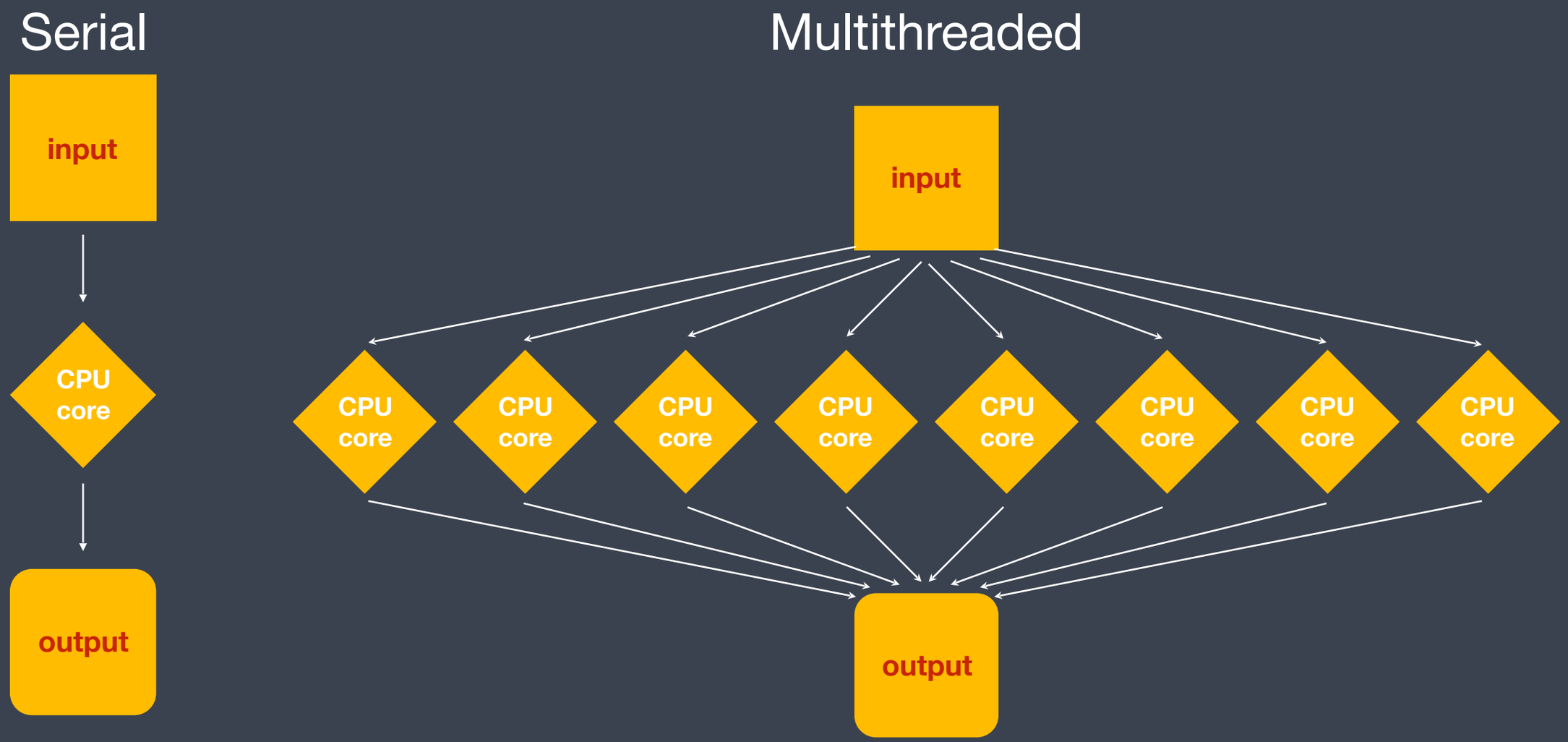
Parallel Computing

“Parallel computing is a form of computation in which many calculations are carried out simultaneously, operating on the principle that large problems can often be divided into smaller ones, which are then solved concurrently ("in parallel”)."

http://en.wikipedia.org/wiki/Parallel_computing

High-Performance Computing

For 1 sample

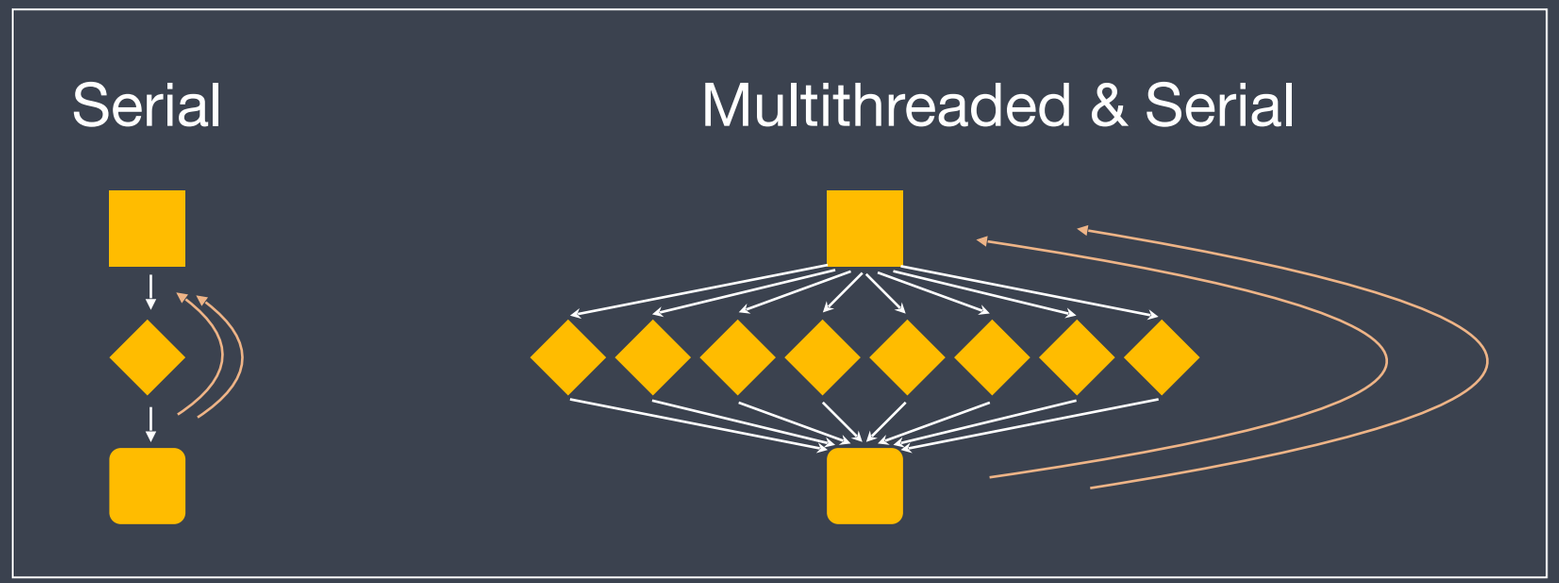


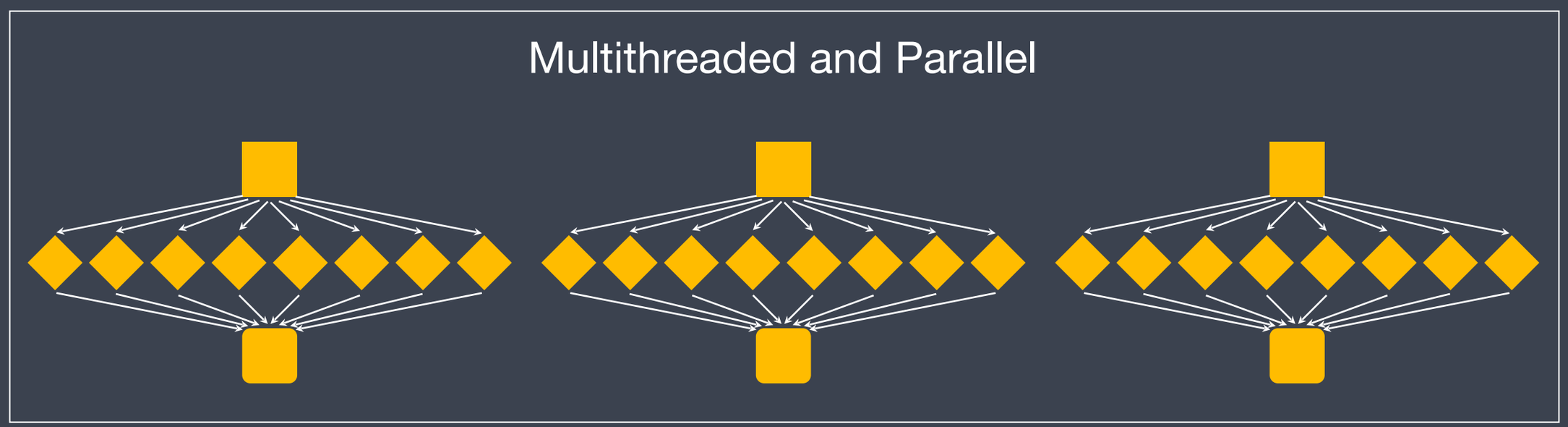
Faster and more efficient…

NGS data analysis is very amenable to this strategy

High-Performance Computing

For 3 samples





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, and there is an associated cost

Introduction to High Performance Computing for New Users

gRED IT

Information in slides courtesy of

Slaton Lipscomb

Architect, gRED IT

lipscomb.slaton@gene.com

Welcome to HPC @ Genentech!

- Old cluster = “Rescomp” is being phased out
- New cluster = “Rosalind” is in early production stage
- New cluster includes of the following changes:
  - the Lmod environmental module system
  - the Slurm scheduler
  - automated software build and installation framework with EasyBuild
  - CentOS 7 (Linux)

http://go.gene.com/rosalind

Rosalind Tech Specs

- 4 login nodes
- ~8500 cores on compute nodes
    - 291 nodes (RAM: 256 GB)
    - 6 high-memory nodes (RAM: 1 TB)
    - 8 gpu nodes (RAM: 128 GB)
- ~12 PB of new storage (on Rosalind)

  + existing 45 PB (Isilon NAS)

Using the cluster!

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**

Log in!

Open a terminal

```
ssh username@rosalind.gene.com
```

Welcome to Rosalind!

Where are you when you log in?

```
username@nl002:~ %
```

Logged into login node **`nl002`**. These nodes are not meant for heavy lifting!

```
username@nl002:~ % pwd
```

Land in the home directory. The “~” is short for the path to a user’s home directory, i.e. **`/gstore/home/username`**

Interactive Sessions

The **`srun`** command can be used to start an interactive session. It sends a request to the scheduler to give you access to a compute node.

```
username@nl002:~ % srun --pty -p defq
                   --qos=interactive --mem 8G bash
```

"`srun --pty`" is how interactive sessions are started

"`-p defq`" is the partition

"`--qos=interactive`" is the Slurm QoS policy

"`--mem 8G`" is the memory requested

```
username@nc026:~ %
```

No longer on the login node, but using the compute node **`nc026`**

2. Using installed software

LMOD: Software Modules

- Many tools that are available on the cluster are installed as environment modules
- The LMOD system adds the path to a given software package, as well as it's dependencies into the **$PATH** environment variable
- Allows for clean, easy loading, unloading and version tracking/switching.

LMOD: Software Modules

Using the LMOD system:

```
% module avail #to see software now available

% module spider #verbose software currently
available

% echo $PATH
```

Loading/Unloading Modules

- Loading modules

```
% module load fastqc    OR    % ml fastqc
```

- Which module version is loaded (if at all)?

```
% which fastqc
```

```
% echo $PATH
```

- Need help with the module?

```
% module help fastqc
```

- Unloading modules

```
% module unload fastqc
```

- Dump all modules

```
% module purge
```

3. The Job Scheduler, Slurm

Submitting Jobs

In an “interactive session”, programs can be called directly.

```
% fastqc -n 2 file1_1.fq file1_2.fq
```

What if you wanted to run the program and come back later to check on it?

You can do this by submitting a *batch job* with **`sbatch`**

```
% sbatch mybowtiejob.sh
```

Simple Linux Utility for Resource Management (SLURM)

- Fairly allocates access to resources (computer nodes) to users for some duration of time so they can perform work

- Provides a framework for starting, executing, and monitoring batch jobs

- Manages a queue of pending jobs; ensures that no single user or core monopolizes the cluster

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

The "sbatch" way of submitting jobs (1/2)

```
% sbatch -p defq --qos=short -n 2 --wrap="fastqc
  -t 2 file1.fq file2.fq"
```

Arguments used:

- -p (partition depends on which node type you want to use)
- --qos (Job runtime/walltime policies selected via Slurm QoS policy)
- -n (number of cores)
- --wrap (specifying the command you want to run)

-p available

- defq => compute nodes
- himem => high memory nodes (1 TB)
- gpu => gpu nodes

--qos available

- veryshort = 10 min
- short = 2 hr
- medium = 24 hr
- long = 3 days
- verylong = 14 days
- interactive = 6 days

The "sbatch" way of submitting jobs (2/2)

Recommended: write a job submission script

```
% sbatch completeSlurmJob.run
```

Creating a job submission script

```
#! /bin/sh                                  #Always at the top of the script

#SBATCH -p defq
#SBATCH --qos short
#SBATCH -n 4
#SBATCH --mem=8G
#SBATCH -o %j.out
#SBATCH -e %j.err
#SBATCH -J bowtie2_run1
#SBATCH --mail-type=ALL
#SBATCH --mail-user=<email>

module load bowtie2

bowtie -n 4 hg19 file1_1.fq file1_2.fq
```

Save as myJobScript.run

Run as `% sbatch myJobScript.run`

Note: *The* ***`sbatch`*** *options are specified differently here as compared to the first method*

`sbatch` options

```
#SBATCH -p #partition
#SBATCH --qos=<name> #QOS for length of job
#SBATCH -n X #number of cores
#SBATCH -N 1 #confine cores to 1 node (default: 1)
#SBATCH -J name_of_job (default: name of job script)
#SBATCH -o %j.out #out file
#SBATCH -e %j.err #error file
```

-p available

- defq => compute nodes
- himem => high memory nodes (1 TB)
- gpu => gpu nodes

--qos available

- veryshort = 10 min
- short = 2 hr
- medium = 24 hr
- long = 3 days
- verylong = 14 days
- interactive = 6 days

Managing jobs and getting information about submitted/running jobs

Job Monitoring

```
% squeue –u <username> –t RUNNING/ PENDING

% squeue –u <username> –p partition

% squeue –u <username> --start
```

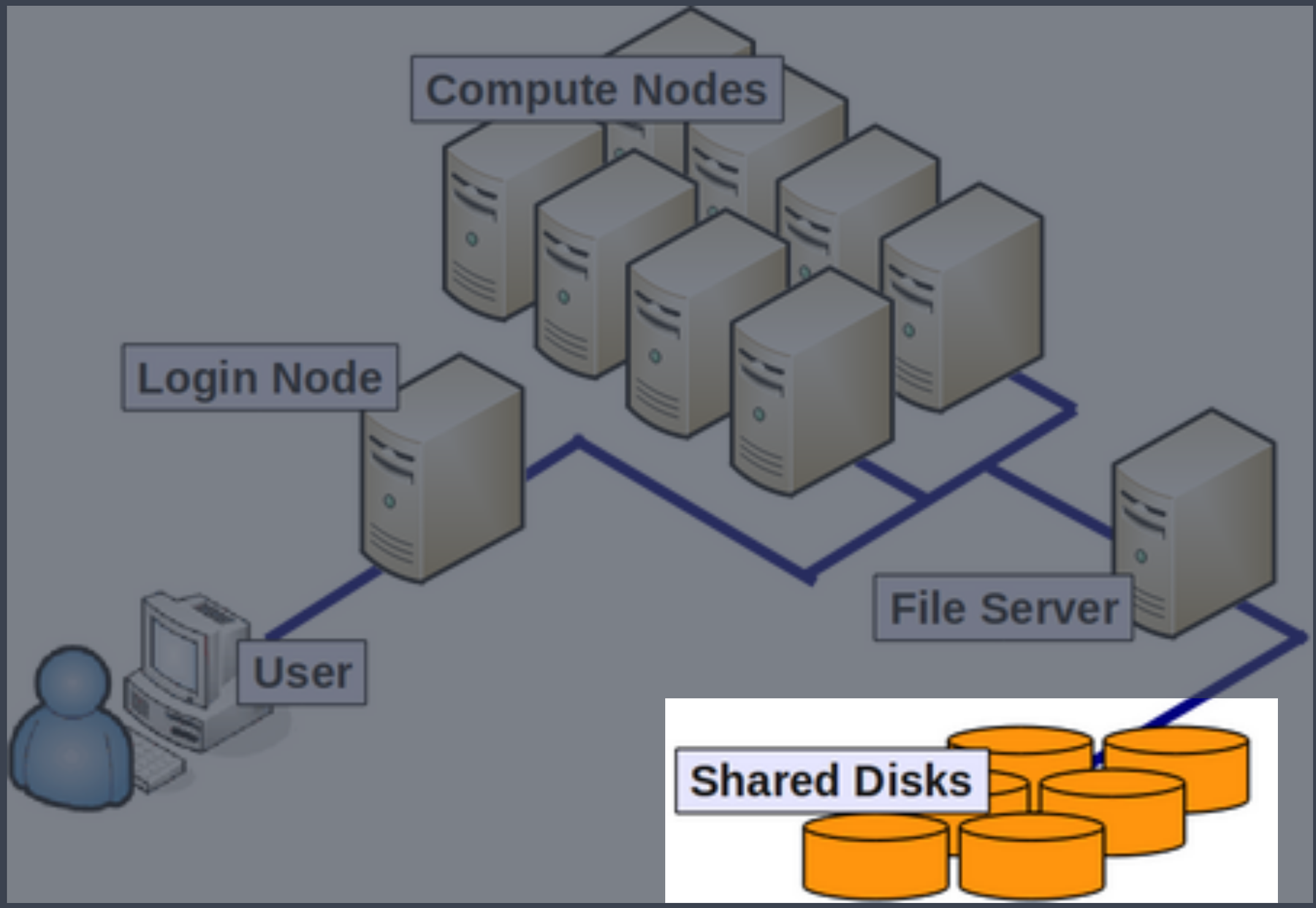
```
Detailed job info:
% scontrol show jobid <jobid>


Completed job statistics:
% sacct -j <jobid>
```

Cancelling/Pausing Jobs

```
% scancel <jobid>
% scancel -t PENDING
% scancel --name JOBNAME
% scontrol hold <jobid> #pause pending jobs
% scontrol release <jobid> #resume
% scontrol requeue <jobid> #cancel and rerun
```

4. Filesystems and storage

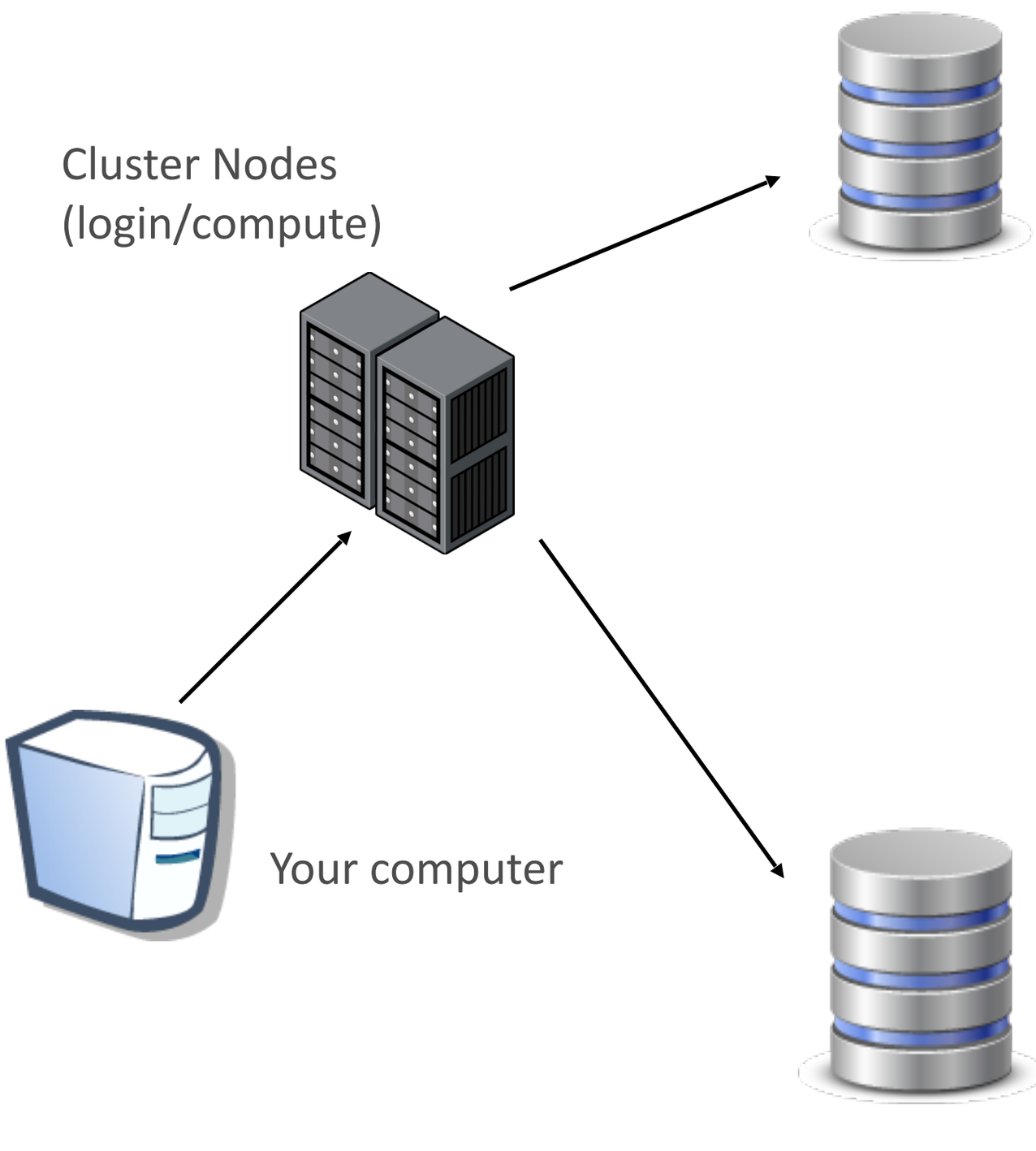
Filesystems and storage



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

Data Storage



/gstore (GPFS)

`/gstore/home/username`

- individual home directories
- path stored in $HOME
- 50 GB limit

`/gstore/scratch/u/username`

- path stored in $SCRATCH
- unlimited (within reason)
- deleted after 90 day if untouched

`/gstore/data/some-project`

- project or department specific

/gne (Isilon)

`/gne/data/some-project`

- project or department specific

More direction on storage

1. **`/gstore/home/username`** :: should be used for scripts, locally compiled or installed applications, documentation, papers, and other small files.

2. **`/gstore/scratch/u/username`** :: should be used for all active workflow (scientific computation) and in-progress data.

3. **`/gstore/data/some-project`** & **`/gne/data/some-project`** :: should be used for valuable intermediate or results data as well as raw instrument data.

More information

- http://go.gene.com/rosalind
- Get an account at http://go.gene.com/hpchelp
- Storage guide: https://rochewiki.roche.com/confluence/display/SCICOMP/Storage+Guide#StorageGuide-gstorescratch

Thanks!

- Slides adapted from Kristina Holton at HMS-RC