

# Beginner's Introduction to Computing at CARC

Matthew Fricke

Version 0.1

# Goals

- 1) Basic Linux Literacy
- 2) Ability to run programs on CARC compute nodes

I hope to spend about an hour on each with a 15 minute break.

# Outline

- High Performance Computing Overview
- Logging in
- The BASH Shell and Scripts
- The Slurm Job Scheduler
- Storage at CARC
- Transferring data to and from CARC
- Accessing software and the module system
- Parallelization

# High Performance Computing

- What is high performance computing?
  - Really just means something that is a lot more powerful than your desktop or laptop.
  - Hardware:
    - That might mean more and faster processors to do the calculations more quickly (eg 400 CPUs instead of 4)
    - More RAM so you can work on bigger problems (3,000 GB instead of 8)
    - Bigger file systems so you can process larger datasets
    - More and bigger GPUs to accelerate your computations (12 GPUs at a time instead of 1)
  - People:
    - Someone else to manage the systems and keep them running and secure
    - Someone to answer your questions and help with problems

# Logging into Wheeler



First login to the Linux **workstation** in front of you.

Use your CARC username and password.

Keven, Tannor, and Jose can help you login if you have trouble.

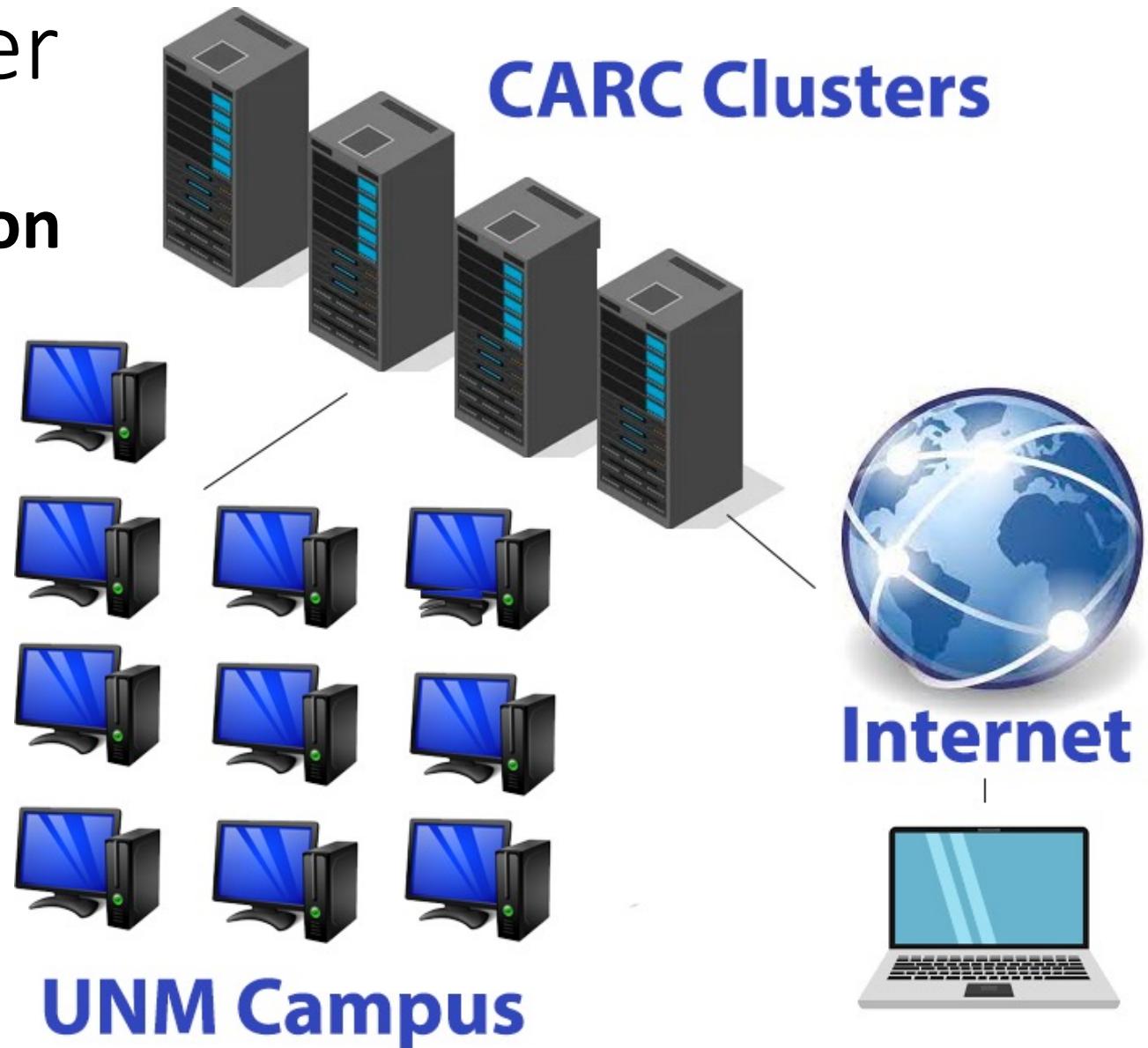
This is an “important step” so don’t let me move on until you have logged in

# Logging into Wheeler

First login to the **workstation** in front of you.

You will always login to CARC cluster remotely.

These clusters don't even have monitors.



# Logging into Wheeler

We are going to use a program called **secure shell**.

**Secure shell (ssh)** is now built into every major operating system (Windows, OSX, and Linux).

You don't need third party programs like putty anymore.

# Logging into Wheeler



```
ssh mfricke@wheeler.alliance.unm.edu
```

Should prompt you for a password...

Don't let me move on until you are able to login.

# Logging into Wheeler

```
Matthew — ssh wheeler — 82x27
ast login: Tue Jun 14 14:47:24 2022 from fricke.co.uk
-----
Welcome to Wheeler

Be sure to review the "Acceptable Use" guidelines posted on the CARC website

For assistance using this system email help@carc.unm.edu.

Tutorial videos can be accessed through the CARC website: Go to
http://carc.unm.edu, select the "New Users" menu and then click
"Introduction to Computing at CARC".

Warning: By default home directories are world readable. Use the chmod command
to restrict access.

Don't forget to acknowledge CARC in publications, dissertations, theses and
presentations that use CARC computational resources:

"We would like to thank the UNM Center for Advanced Research Computing,
supported in part by the National Science Foundation, for providing the
research computing resources used in this work."

Please send citations to publications@carc.unm.edu.

-----
tarting SSH Key Agent...
gent pid 19486
fricke@wheeler:~ $
```

Please enter the following command



```
cp -r /projects/shared/workshops/beginner/mystuff ~/
```

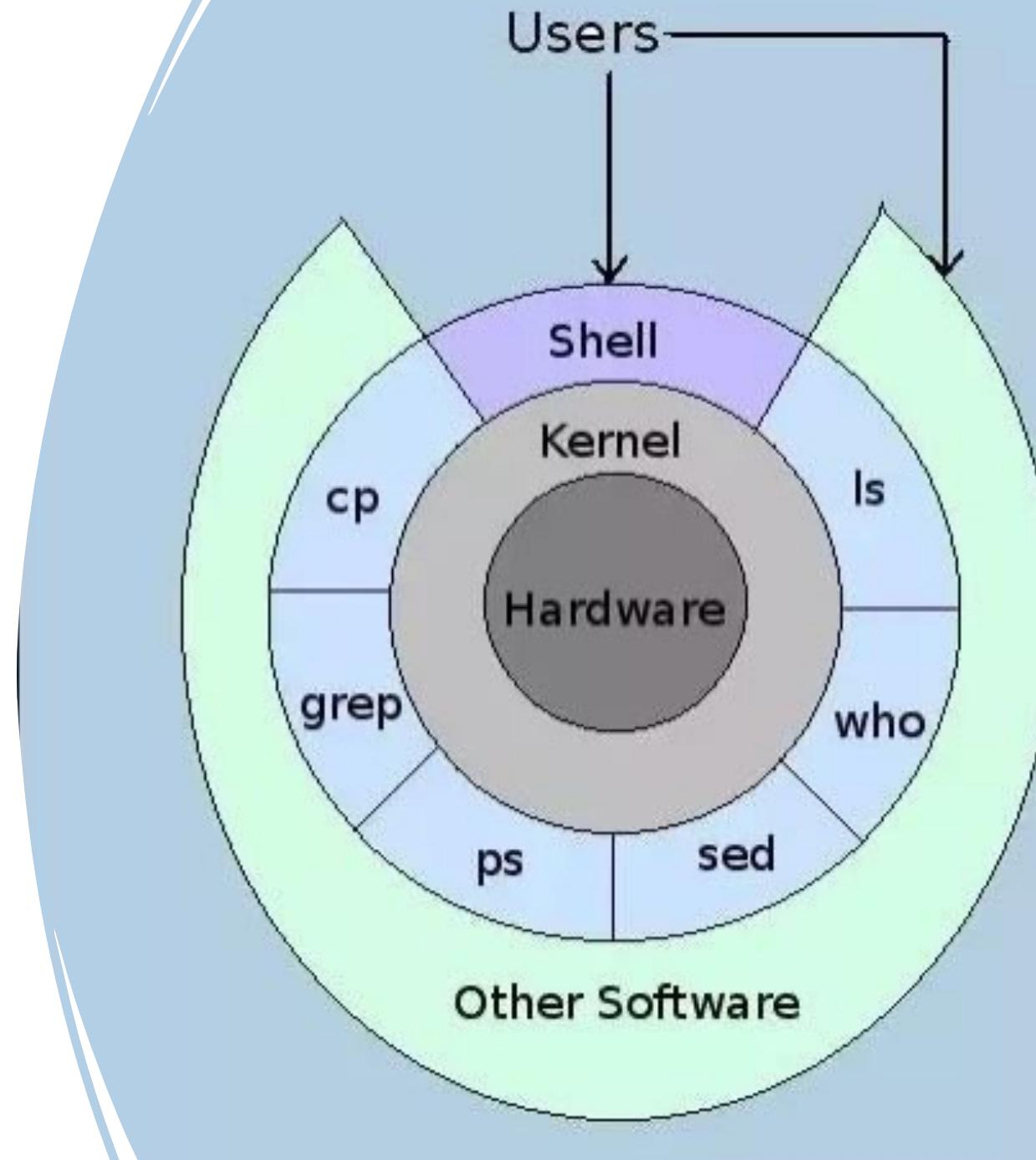
We will come help you if you have any trouble.

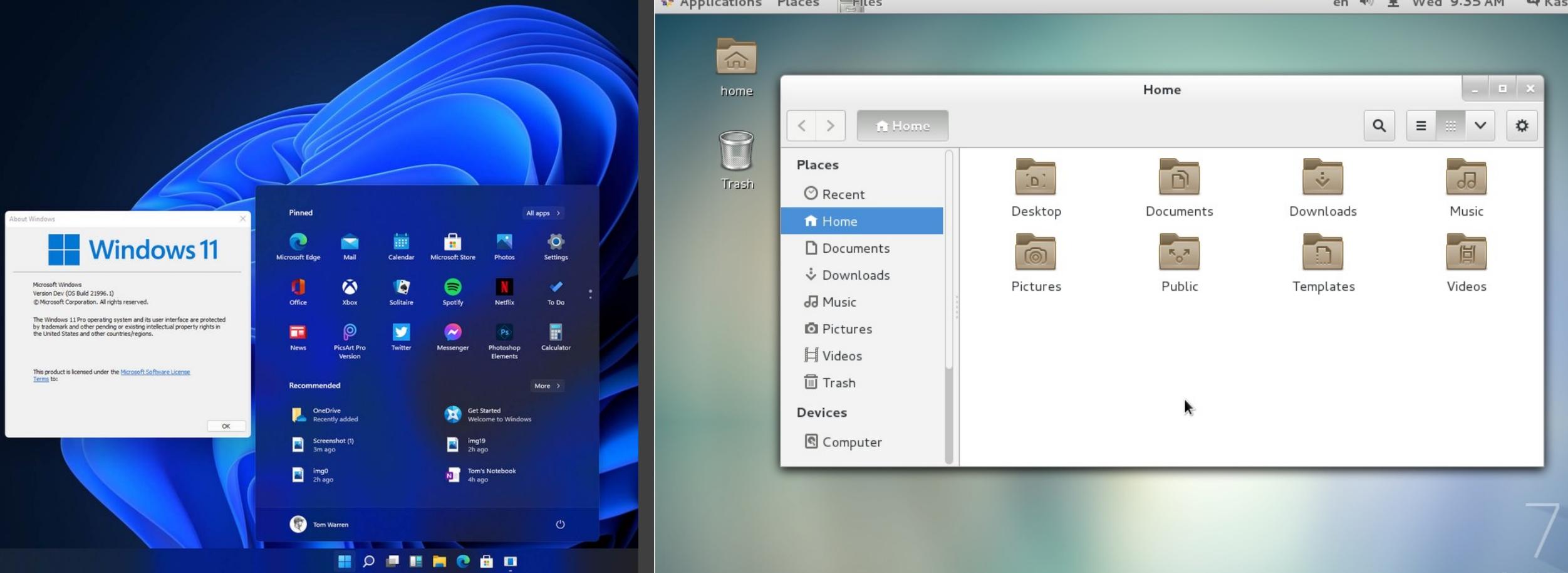
(Later I will go over what this command does)

# Linux and the BASH Shell

---

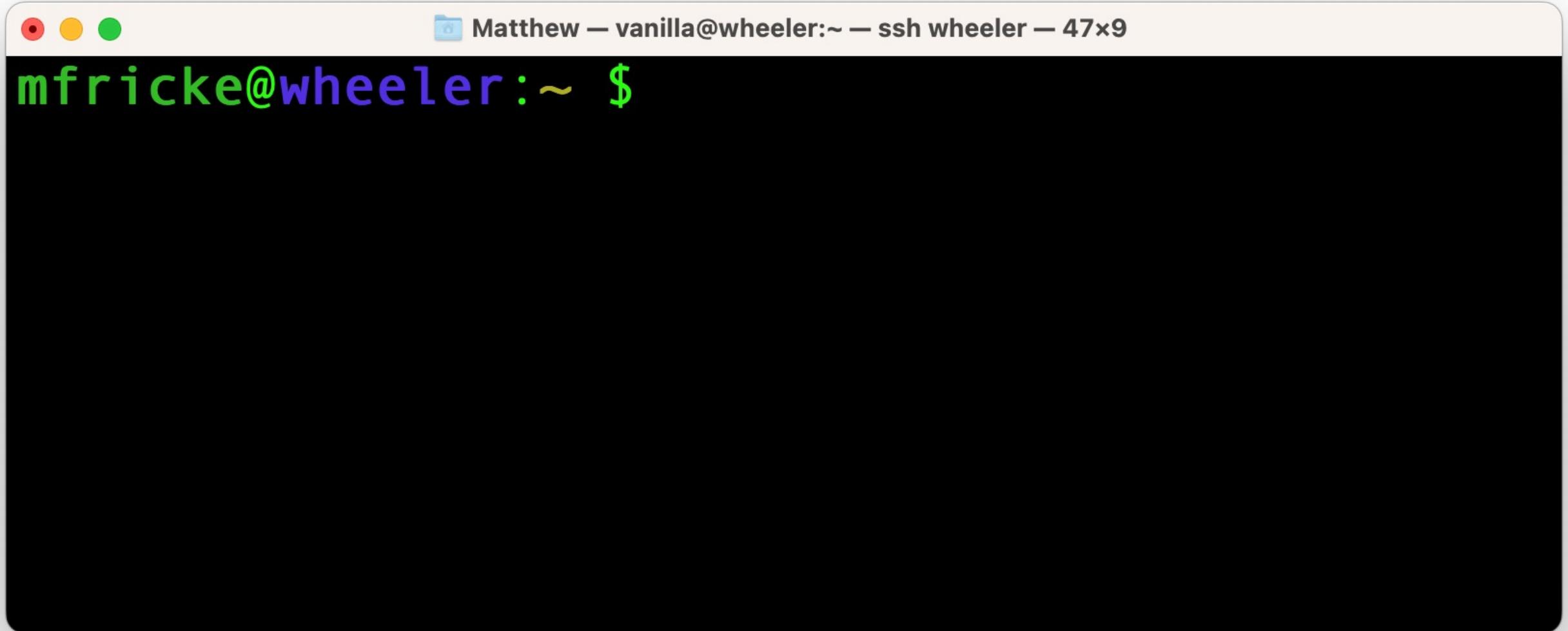
- The Kernel manages access to the hardware in a computer.
- An Operating System (OS) is the Kernel plus useful programs provided by the OS.
- The “shell” is the outermost layer of the OS.
- It is where the user interacts with the OS.





# Graphical Shells (GUIs)

# Logging into Wheeler



A screenshot of a macOS terminal window. The window has a white header bar with three colored circular icons (red, yellow, green) on the left. The title bar contains the text "Matthew — vanilla@wheeler:~ — ssh wheeler — 47x9". The main area of the terminal is black and shows the command prompt "mfricke@wheeler:~ \$".



**YOU ARE AT THE  
RIGHT PLACE**

# Linux and the BASH Shell

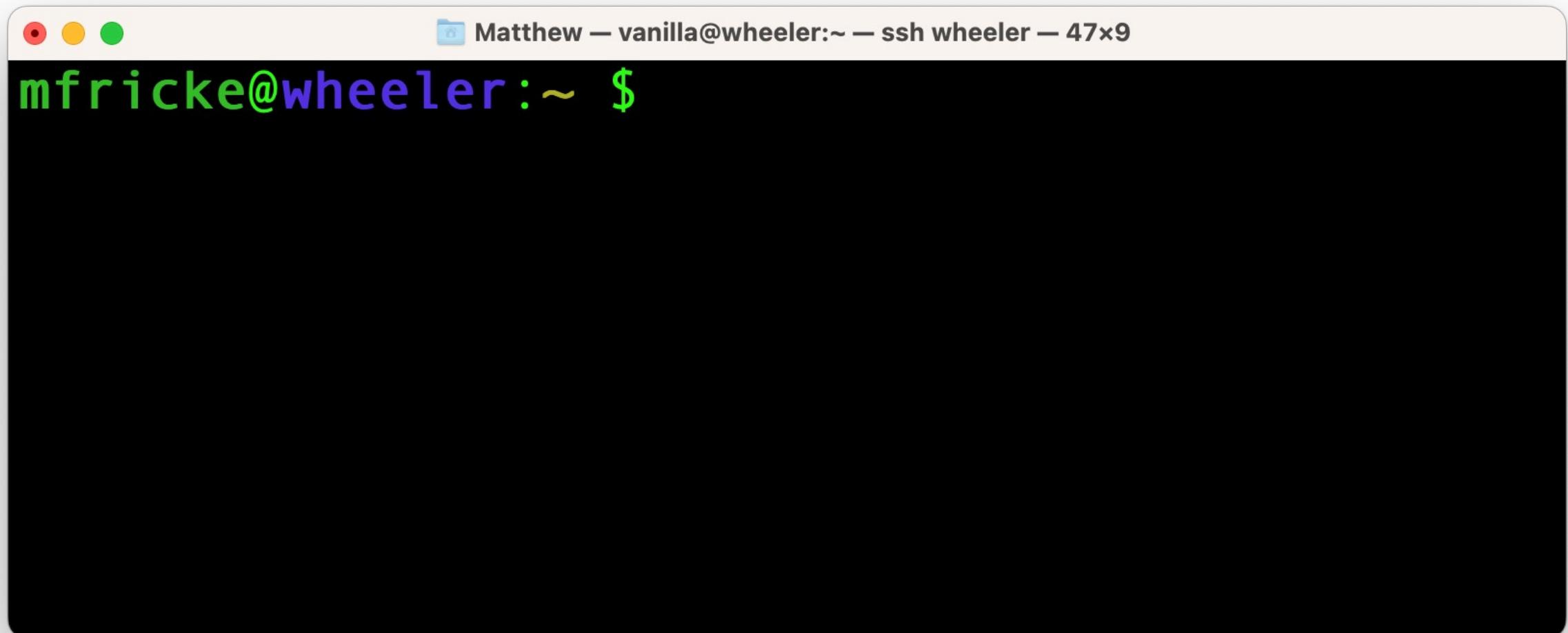


**WHERE THERE IS A SHELL, THERE IS A WAY!**

# The Borne-Again Shell (BASH)

---

Written in 1976 by Stephen Bourne for UNIX version 7.



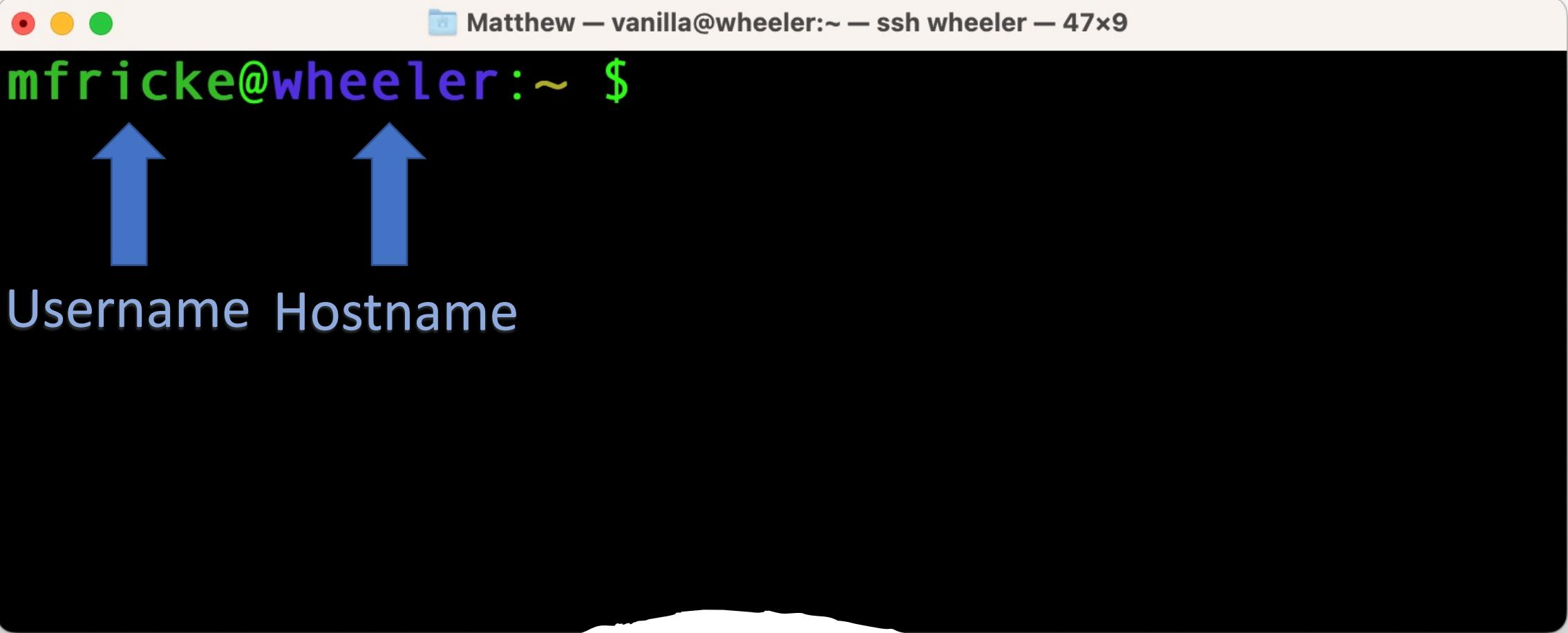
A screenshot of a macOS terminal window. The window title bar shows three colored circular icons (red, yellow, green) on the left, followed by a folder icon and the text "Matthew — vanilla@wheeler:~ — ssh wheeler — 47x9". The main terminal area is black and contains a single line of text: "mfricke@wheeler:~ \$". The text is color-coded: "mfricke@wheeler:" is in green, "@" is in blue, and the prompt ":~ \$" is in green. The window has rounded corners and a thin gray border.

mfricke@wheeler:~ \$

↑

Username

Understanding the BASH prompt...



mfricke@wheeler:~ \$

↑      ↑

Username Hostname

Understanding the BASH prompt...



Matthew — vanilla@wheeler:~ — ssh wheeler — 47x9

mfricke@wheeler:~ \$



This is the current working directory.  
“~” is short for home directory

Understanding the BASH prompt...



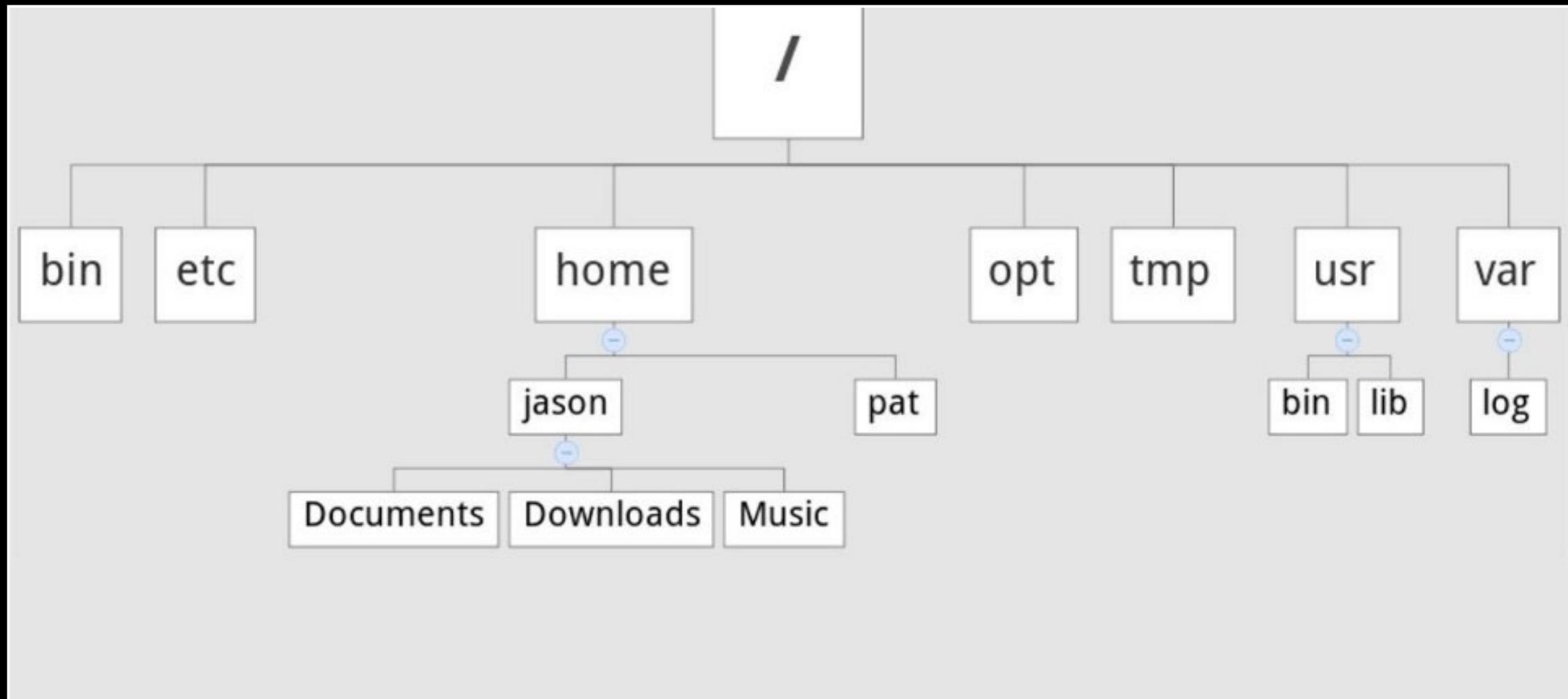
Matthew — vanilla@wheeler:~ — ssh wheeler — 47x9

mfricke@wheeler:~ \$



“\$” means this user is standard user  
(i.e. not a system administrator)

Understanding the BASH prompt...



Example Filesystem Tree



Matthew — vanilla@wheeler:~ — ssh wheeler — 42x13

```
[vanilla@wheeler ~]$ pwd  
/users/vanilla  
[vanilla@wheeler ~]$ █
```

Figuring out where you are in the  
filesystem... |



Matthew — vanilla@wheeler:~ — ssh wheeler — 42x13

```
[vanilla@wheeler ~]$ ls  
mystuff  wheeler-scratch  
[vanilla@wheeler ~]$
```

Figuring out where you are in the  
filesystem...



Matthew — vanilla@wheeler:~ — ssh wheeler — 49x13

```
[vanilla@wheeler ~]$ tree
```

```
.
├── mystuff
│   ├── myfile1
│   └── myfile2
└── wheeler-scratch -> /wheeler/scratch/vanilla
```

```
2 directories, 2 files
```

```
[vanilla@wheeler ~]$
```

Figuring out where you are in  
the filesystem...



Matthew — vanilla@wheeler:~ — ssh wheeler — 49x13

```
[vanilla@wheeler ~]$ tree
```

```
.
├── mystuff
│   ├── myfile1
│   └── myfile2
└── wheeler-scratch -> /wheeler/scratch/vanilla
```

```
2 directories, 2 files
```

```
[vanilla@wheeler ~]$
```

Figuring out where you are in  
the filesystem...



```
[vanilla@wheeler ~]$ tree
```



```
└── mystuff
    ├── myfile1
    └── myfile2
└── wheeler-scratch -> /wheeler/scratch/vanilla
```

This . means the current directory

2 directories, 2 files

```
[vanilla@wheeler ~]$
```

Figuring out where you are in  
the filesystem...

# “Absolute” paths vs “relative” paths

- A path is a list of directories and/or files. It is a path through the directory tree that tells one how to get somewhere in the filesystem.
- An absolute path tells one how to get to the destination from starting from the root of the filesystem. E.g “/users/vanilla/mystuff/”
- A relative path specifies how to get there \*starting from the current working directory\*. E.g vanilla/mystuff/



Matthew — vanilla@wheeler:~ — ssh wheeler — 49x13

```
[vanilla@wheeler ~]$ ls mystuff/
myfile1 myfile2
[vanilla@wheeler ~]$ █
```

Figuring out where you going...

 Matthew — vanilla@wheeler:~ — ssh wheeler — 49x13

```
[vanilla@wheeler ~]$ ls /users/vanilla/mystuff
myfile1 myfile2
[vanilla@wheeler ~]$
```

Figuring out where you going...

```
[vanilla@wheeler ~]$ ls ./mystuff/  
myfile1 myfile2  
[vanilla@wheeler ~]$ ls ~/mystuff/  
myfile1 myfile2  
[vanilla@wheeler ~]$
```

Figuring out where you going...

```
[vanilla@wheeler ~]$ ls -a
.
..
.
.addressbook
.addressbook.lu
.bashrc
.cache
.comsol
.config
.flexlmrc
.modulesbeginenv
mystuff
.oracle_jre_usage
.pinerc
.pki
.rhosts
.shosts
.spack
.ssh
```

Figuring out where you going...

```
[vanilla@wheeler ~]$ ls -l
total 4
drwxr-xr-x 2 vanilla users 4096 Jun 14 22:05 mystuff
lrwxrwxrwx 1 vanilla users    24 Jun 14 21:20 wheeler-scratch ->
/wheeler/scratch/vanilla
[vanilla@wheeler ~]$
```

Figuring out where you going...

```
[vanilla@wheeler ~]$ ls -l mystuff/
total 473704
-rw-r--r-- 1 vanilla users 483165473 Jun 14 23:20 myfile1
-rw-r--r-- 1 vanilla users          0 Jun 14 22:05 myfile2
[vanilla@wheeler ~]$
```

Figuring out where you going...

```
[vanilla@wheeler ~]$ ls -lh mystuff/
total 463M
-rw-r--r-- 1 vanilla users 461M Jun 14 23:20 myfile1
-rw-r--r-- 1 vanilla users     0 Jun 14 22:05 myfile2
[vanilla@wheeler ~]$
```

Figuring out where you going...

```
[vanilla@wheeler ~]$ du -s
```

```
499704 .
```

```
[vanilla@wheeler ~]$ du -sh
```

```
488M .
```

```
[vanilla@wheeler ~]$
```

Figuring out where you going...

```
[vanilla@wheeler ~]$ df -h
```

Filesystem	Size	Used	Avail	Use%	Mounted on
devtmpfs	24G	0	24G	0%	/dev
tmpfs	24G	64K	24G	1%	/dev/shm
tmpfs	24G	968M	23G	5%	/run
tmpfs	24G	0	24G	0%	/sys/fs/cgroup
/dev/mapper/centos-root	930G	567G	363G	61%	/
/dev/sdc2	836G	72G	764G	9%	/tmp
/dev/md126p1	2.0G	333M	1.7G	17%	/boot
172.17.2.254:/mnt/wheeler-scratch	37T	28T	8.7T	77%	/wheeler/scratch
172.17.2.255:/mnt/wheeler-scratch2	37T	28T	9.0T	76%	/wheeler/scratch2
beegfs_nodev	110T	51T	60T	46%	/carc/scratch
chama:/home/homes	65T	36T	30T	55%	/users
chama:/home/carc_projects	65T	36T	30T	55%	/projects

Figuring out where you going...

```
[vanilla@wheeler ~]$ quota -s
Disk quotas for user vanilla (uid 659):
  Filesystem   space   quota   limit   grace   files   quota   limit   grace
chama:/home/homes          488M    100G    200G
[vanilla@wheeler ~]$
```

Figuring out where you going...

```
[vanilla@wheeler ~]$ stat mystuff/myfile1
  File: 'mystuff/myfile1'
  Size: 483165473 Blocks: 947408      IO Block: 65536   regular file
Device: 28h/40d Inode: 9232782834205560540 Links: 1
Access: (0644/-rw-r--r--) Uid: ( 659/ vanilla) Gid: ( 100/ users)
Access: 2022-06-14 22:05:27.503289000 -0600
Modify: 2022-06-14 23:20:26.945918000 -0600
Change: 2022-06-14 23:20:48.754917000 -0600
 Birth: -
```

Figuring out what you've got...

```
[vanilla@wheeler ~]$ find -name myfile2  
./mystuff/myfile2
```

```
[vanilla@wheeler ~]$ find -name "myfile*"  
./mystuff/myfile1  
./mystuff/myfile2  
./mystuff/myfile3  
./mystuff/myfile0
```



Wildcard

Figuring out what you've got...

```
[vanilla@wheeler ~]$ cd mystuff/  
[vanilla@wheeler ~/mystuff]$
```

Use the tab key to autocomplete

Going somewhere new...

Now it is your turn...



- For this path:  
`/projects/shared/workshops/beginner/vecadd`
- What are the names of the files in that directory?
- When were they last modified?
- How large are the files?

You can find this information with the `ls` command.

Now it is your turn...



- For this path:

/projects/shared/workshops/beginner/vecadd

Now “cd” into that directory using <tab> autocomplete.

- Now you know how to find your way around filesystems using bash
- Let's see how to modify the filesystem.
  - In bash to move a file we use the `mv` command.
  - To copy a file it is `cp`.
  - To copy files from CARC to a personal computer use `scp` or `rsync`.

```
[vanilla@wheeler beginner]$ pwd  
/projects/shared/workshops/beginner  
[vanilla@wheeler beginner]$ cd ~  
[vanilla@wheeler ~]$ pwd  
/users/vanilla  
[vanilla@wheeler ~]$
```

First return to your home  
directory...

```
[vanilla@wheeler ~]$ cd mystuff  
[vanilla@wheeler ~/mystuff]$ mv myfile1 myfile0  
[vanilla@wheeler ~/mystuff]$ ls  
myfile0  myfile2  myfile3  
[vanilla@wheeler ~/mystuff]
```

Modifying the filesystem...  
moving a file.

```
[vanilla@wheeler ~/mystuff]$ cp myfile0 myfile1  
[vanilla@wheeler ~/mystuff]$
```



Source   Destination

```
[vanilla@wheeler ~/mystuff]$ ls  
myfile0 myfile1 myfile2 myfile3  
[vanilla@wheeler ~/mystuff]$
```

Modifying the filesystem...  
copying a file.

```
[vanilla@wheeler ~/mystuff]$ mkdir mynewdir  
[vanilla@wheeler ~/mystuff]$
```



New directory name

Modifying the filesystem...  
create a new directory.

```
[vanilla@wheeler ~]$ cp -r mystuff mystuff2  
[vanilla@wheeler ~]$
```



Source



Destination

```
[vanilla@wheeler ~]$ ls  
mystuff  mystuff2  wheeler-scratch
```

Copying a whole directory tree...

```
[vanilla@wheeler ~]$ exit  
Lycaon:~ matthew$ scp vanilla@wheeler.alliance.unm.edu:~/mystuff/myfile3 Desktop/
```



Source

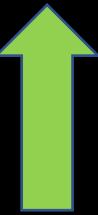


Destination

```
(vanilla@wheeler.alliance.unm.edu) Password:  
myfile3          100%   40      2.0KB/s  00:00
```

Copying data to a personal  
computer from CARC...

```
Lycaon:~ matthew$ scp -r vanilla@wheeler.alliance.unm.edu:~/mystuff Desktop/
```



Source



Destination

```
(vanilla@wheeler.alliance.unm.edu) Password:
```

myfile1	100%	1024KB	6.5MB/s	00:00
myfile2	100%	2048KB	382.5KB/s	00:05
myfile3	100%	40	3.2KB/s	00:00
myfile0	100%	1024KB	8.8MB/s	00:00

Copying data to a personal  
computer from CARC...

```
Lycaon:~ matthew$ scp -r Desktop/mystuff vanilla@wheeler.alliance.unm.edu:~/
```



Source



Destination

```
(vanilla@wheeler.alliance.unm.edu) Password:
```

myfile1	100%	1024KB	591.5KB/s	00:01
myfile0	100%	1024KB	2.0MB/s	00:00
myfile2	100%	2048KB	2.1MB/s	00:00
myfile3	100%	40	2.1KB/s	00:00

To copy from a personal  
computer to CARC...

```
ssh vanilla@wheeler.alliance.unm.edu
```

Log back into wheeler...

```
[vanilla@wheeler ~]$ file mystuff/myfile0  
mystuff/myfile0: data
```

```
[vanilla@wheeler ~]$ file mystuff/myfile3  
mystuff/myfile3: ASCII text
```

Figuring out file types ...

```
[vanilla@wheeler ~]$ cat mystuff/myfile3  
Welcome to the CARC Beginner's Workshop
```

Text files ...

```
[vanilla@wheeler ~]$ nano mystuff/myfile3
```

The screenshot shows a terminal window titled "Matthew — vanilla@wheeler:~ — ssh wheeler — 71x23". Inside the terminal, the command "nano mystuff/myfile3" is run, opening the specified file. The file content is "Welcome to the CARC Beginner's Workshop". The bottom of the screen displays the nano editor's command-line interface with various keyboard shortcuts.

GNU nano 2.3.1 File: mystuff/myfile3

Welcome to the CARC Beginner's Workshop

[ Read 1 line ]

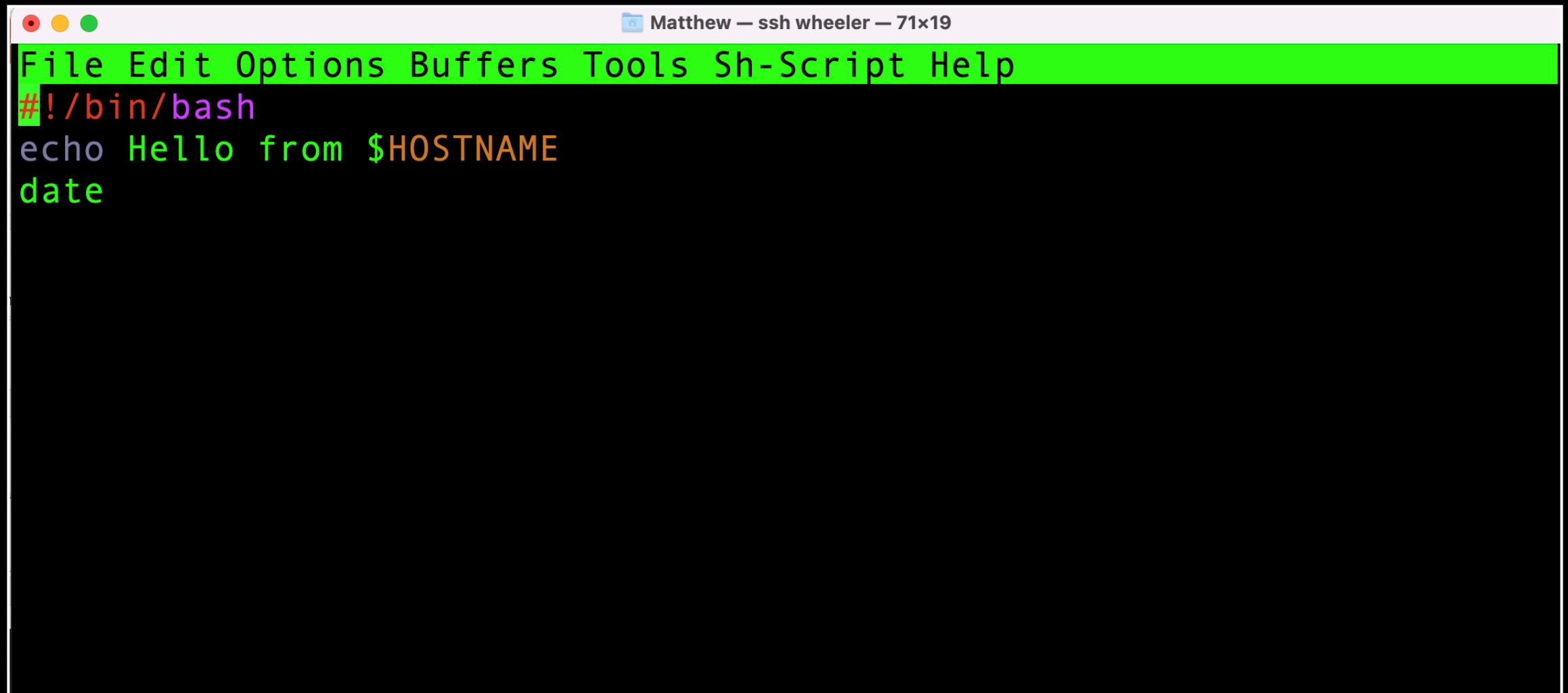
^G Get Help ^O WriteOut ^R Read Fil ^Y Prev Pag ^K Cut Text ^C Cur Pos  
^X Exit ^J Justify ^W Where Is ^V Next Pag ^U UnCut Te ^T To Spell

```
[vanilla@wheeler ~]$ date  
Wed Jun 15 03:08:15 MDT 2022
```

```
[vanilla@wheeler ~]$ echo Hello from $HOSTNAME  
Hello from wheeler
```

```
[vanilla@wheeler ~]$ hostname  
wheeler
```

Programs we will use as examples...



A screenshot of a terminal window titled "Matthew — ssh wheeler — 71x19". The window has a dark background and a light green header bar. The header bar contains the following menu items: File, Edit, Options, Buffers, Tools, Sh-Script, and Help. Below the header bar, the script content is displayed in white text on a black background. The script starts with a shebang line "#!/bin/bash", followed by "echo Hello from \$HOSTNAME", and ends with "date".

```
#!/bin/bash
echo Hello from $HOSTNAME
date
```

# Shell Scripts

# 15 Minute Break

# Software Access

Lmod  
Modules

Conda

```
[vanilla@wheeler ~]$ module spider matlab
```

```
- matlab:
```

```
- Versions:
```

```
  matlab/R2017a
  matlab/R2018b
  matlab/R2019a
  matlab/R2020a
  matlab/R2021a
```

Getting access to software...

```
[vanilla@wheeler ~]$ module load matlab/R2021a
Lmod has detected the following error: Matlab may only be run on compute
nodes. wheeler is not a compute node. Exiting...
While processing the following module(s):
  Module fullname      Module Filename
  -----      -----
matlab/R2021a      /opt/local/modules/matlab/R2021a.lua
```

Getting access to software...

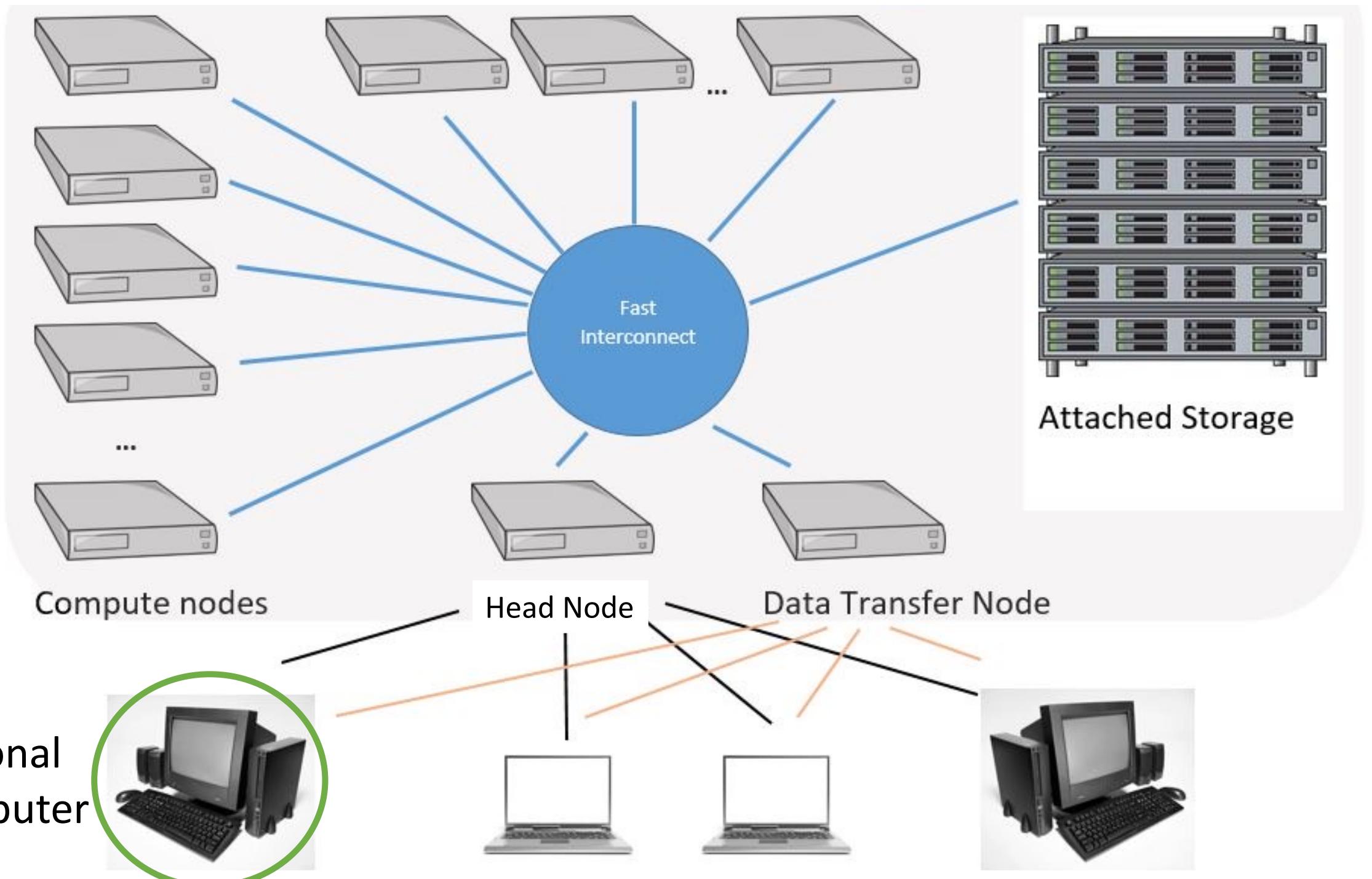
```
[vanilla@wheeler ~]$ module load matlab/R2021a
Lmod has detected the following error: Matlab may only be run on compute
nodes. wheeler is not a compute node. Exiting...
While processing the following module(s):
  Module fullname      Module Filename
  -----      -----
matlab/R2021a      /opt/local/modules/matlab/R2021a.lua
```

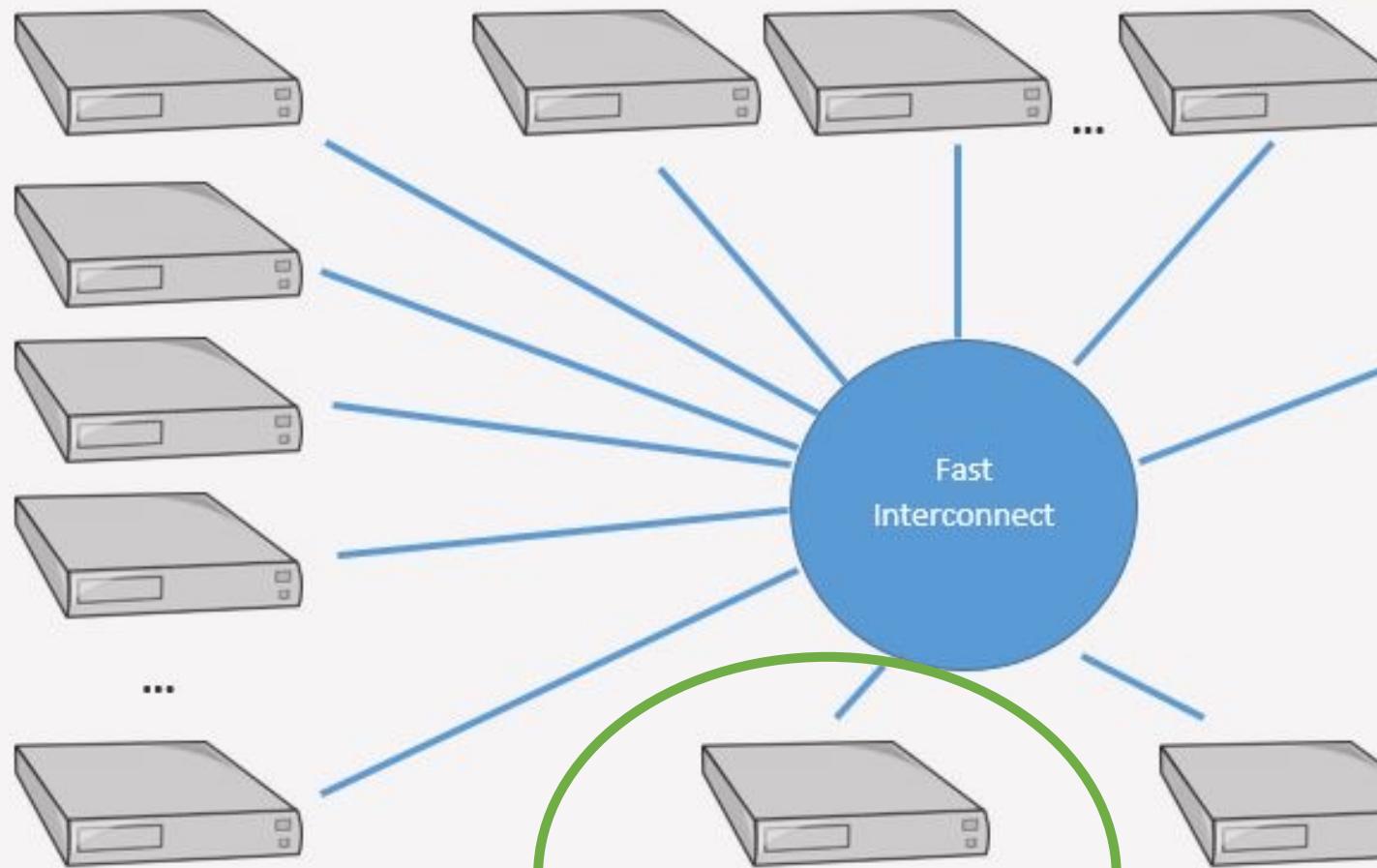
What is a compute node?

Getting access to software...

# HPC Cluster







Compute nodes

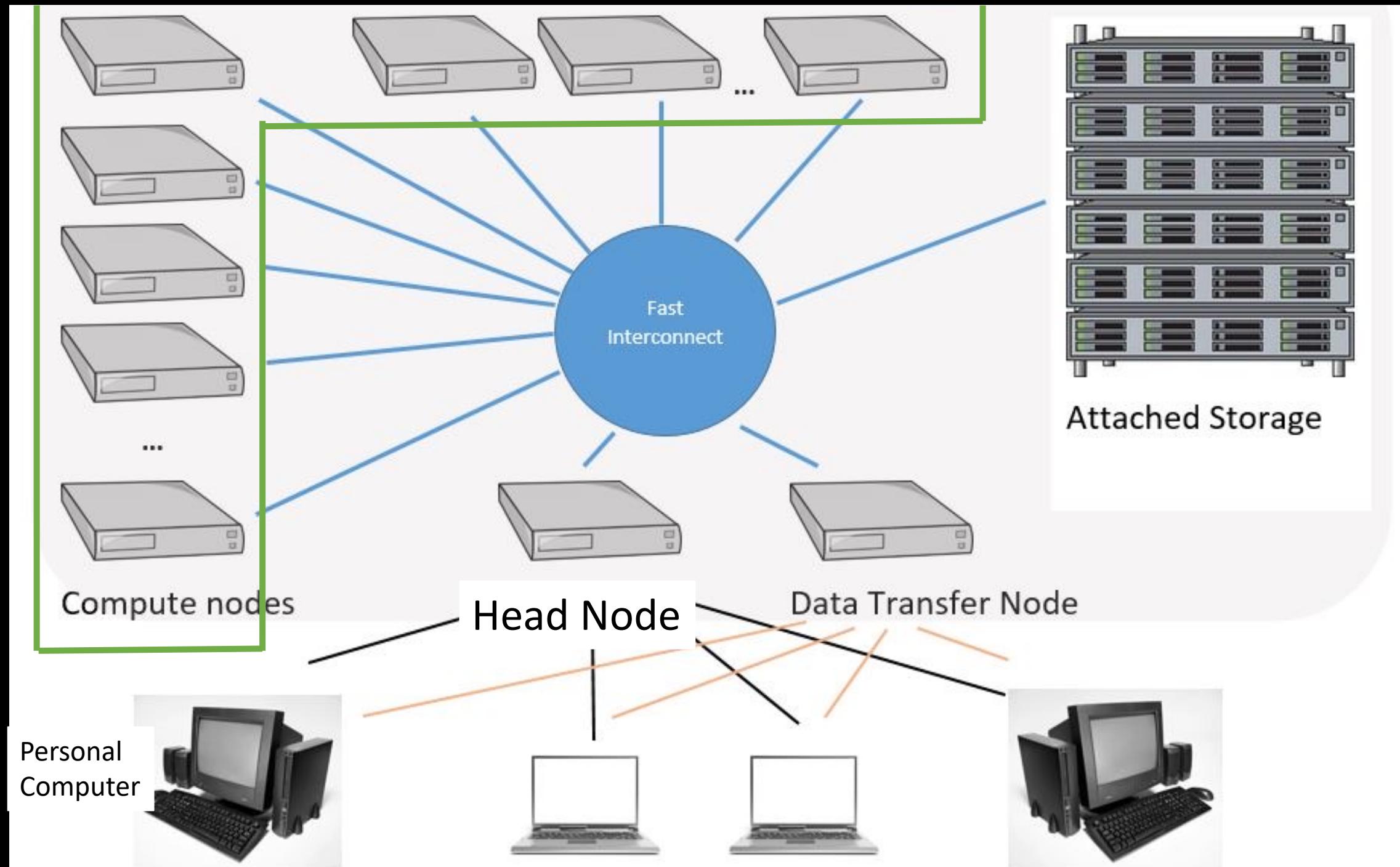
Head Node

Data Transfer Node

Personal Computer



Attached Storage





**WARNING !!**

Never run computations on the head node

Always use compute nodes

```
[vanilla@wheeler ~]$ qgrok
```

queues	free	busy	offline	jobs	nodes	CPUs
normal	0	299	1	97	300	2400
debug	4	0	0	0	4	32
totals:	4	299	1	97	304	2432

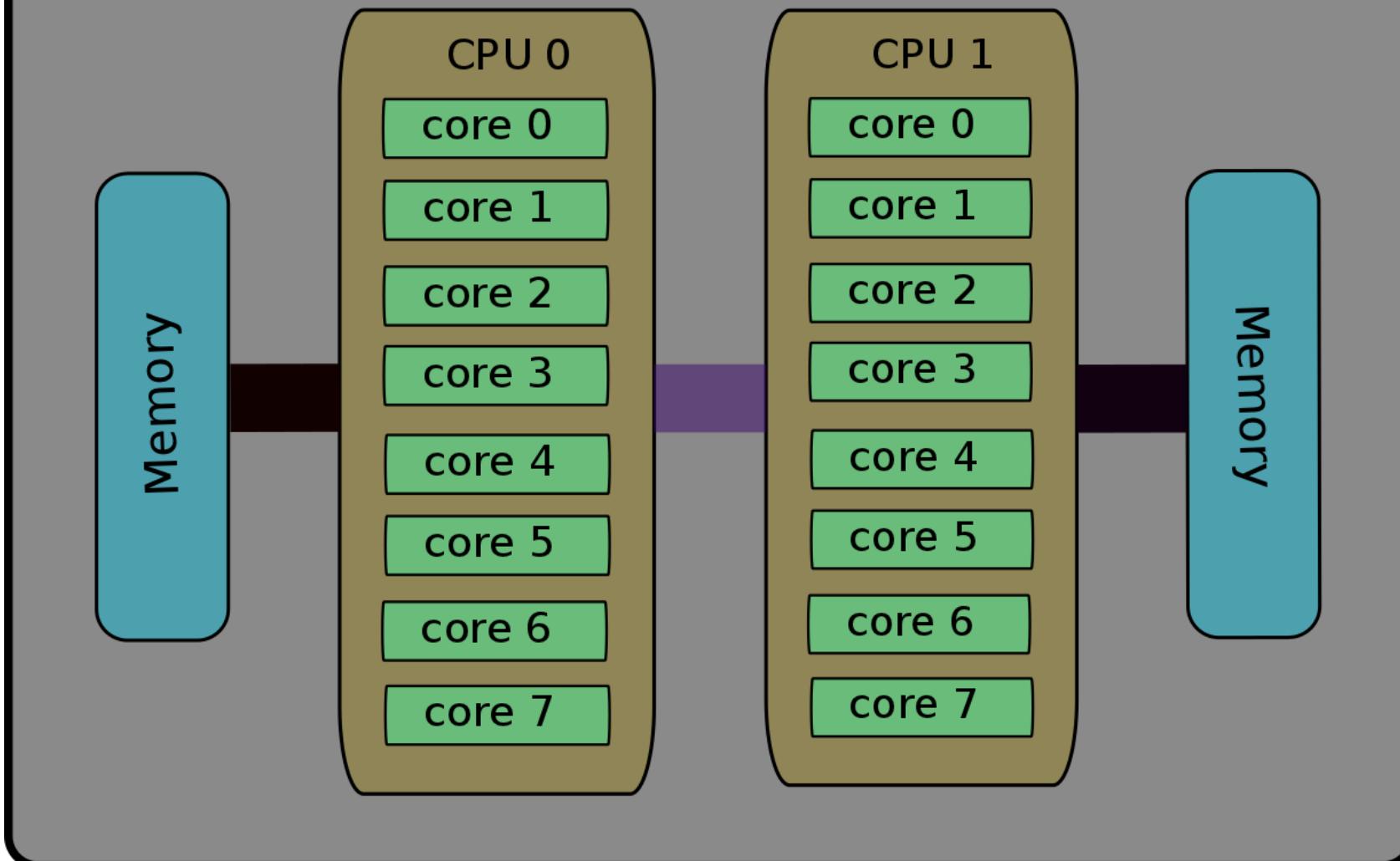
Compute nodes and partitions...

```
qmfricke@hopper:~ $ qgrok
```

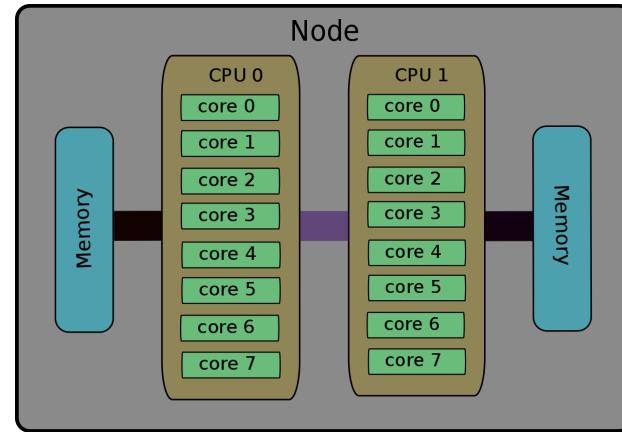
queues	free	busy	offline	jobs	nodes	CPUs
general	12	0	0	0	12	384
condo	29	1	7	0	37	1184
bugs	2	0	0	0	2	64
pcnc	2	0	0	0	2	64
pathogen	0	1	0	1	1	32
tc	7	0	3	0	10	320
gold	2	0	0	0	2	64
fishgen	1	0	0	0	1	32
neuro-hsc	13	0	1	0	14	448
cup-ecs	2	0	0	0	2	64
tid	0	0	1	0	1	32
biocomp	0	0	1	0	1	32
chakra	0	0	1	0	1	32
totals:	70	2	14	1	86	2752

Compute nodes and partitions...

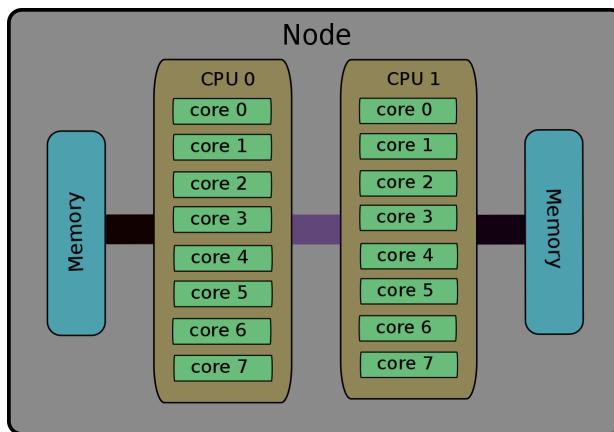
# Node



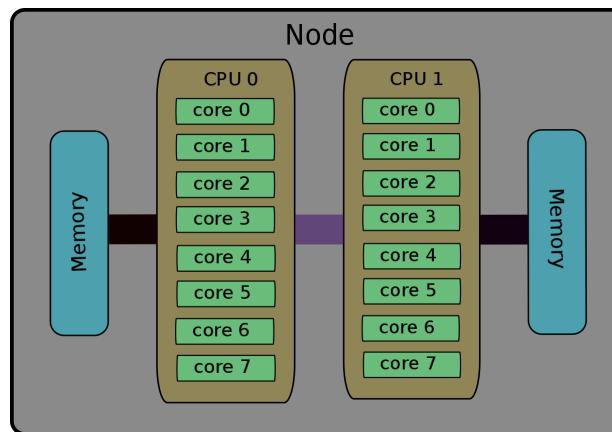
# Head Node (wheeler)



# Compute Nodes

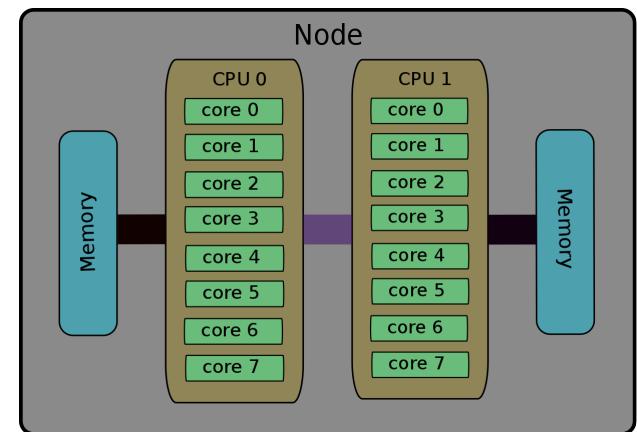


wheeler001



wheeler002

...



wheeler304

```
[vanilla@wheeler ~]$ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                8
On-line CPU(s) list:  0-7
Thread(s) per core:   1
Core(s) per socket:   4
Socket(s):             2
NUMA node(s):          2
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 26
Model name:            Intel(R) Xeon(R) CPU X5550 @ 2.67GHz
```

Wheeler has 8 cores per node....

```
mfricke@hopper:~ $ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                64
On-line CPU(s) list:  0-63
Thread(s) per core:   2
Core(s) per socket:   16
Socket(s):             2
NUMA node(s):          2
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 85
Model name:            Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz
```

Hopper has 32 real cores (64 virtual cores) per node....

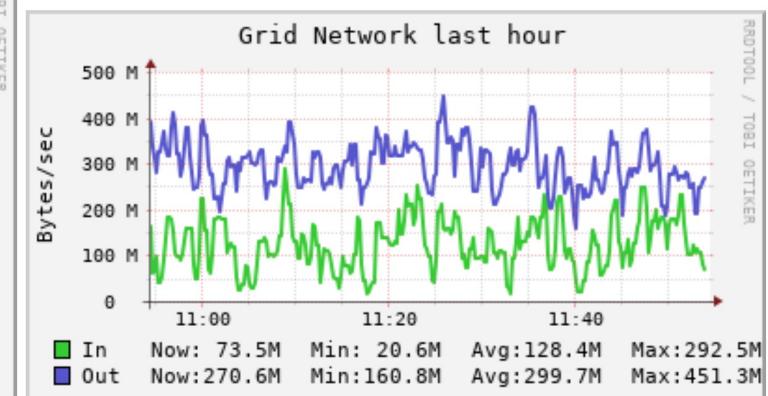
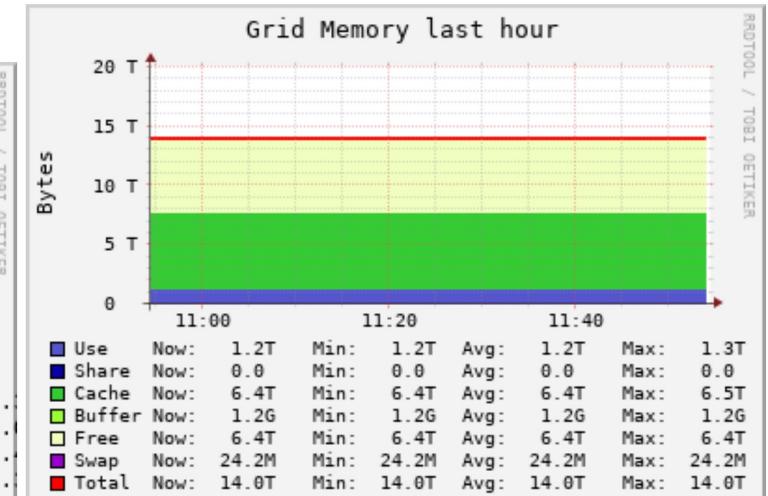
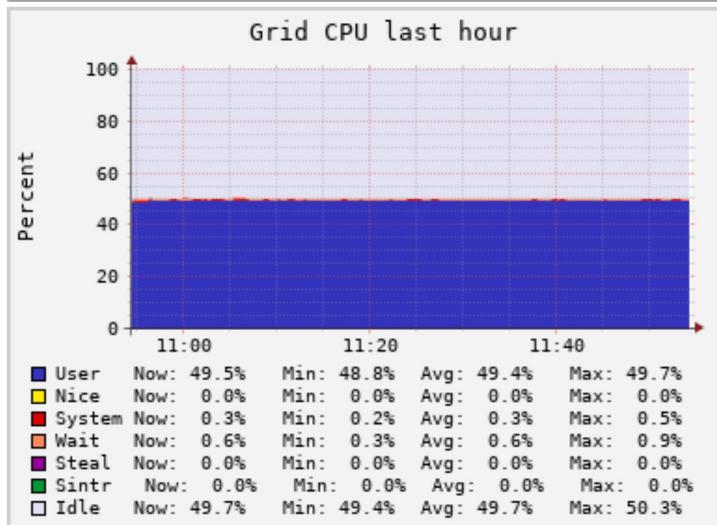
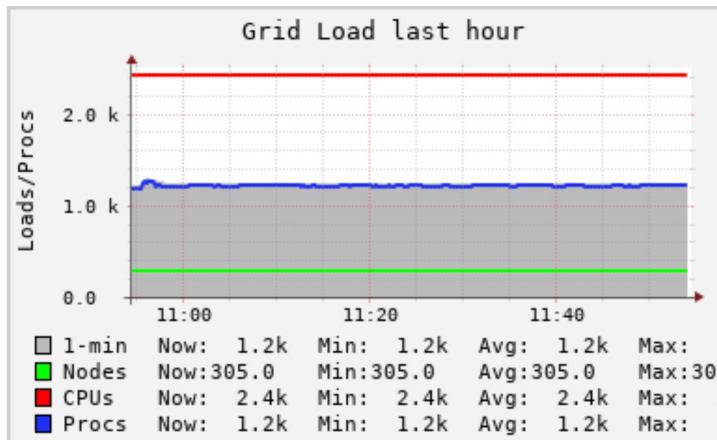
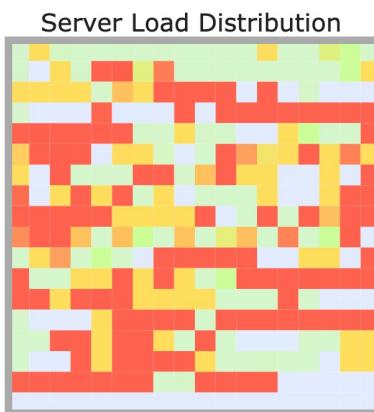
# Architecture

Wheeler Cluster Grid > Wheeler Cluster > --Choose a Node

Overview of @ 2022-03-03 18:54

CPUs Total: **2440**  
Hosts up: **305**  
Hosts down: **0**

Current Load Avg (15, 5, 1m):  
**51%, 51%, 51%**  
Avg Utilization (last hour):  
**51%**





Technology, IT etc.

# SLURM

means

Simple Linux Utility for Resource Management

ENJOY

# Slurm

SODA

IT'S HIGHLY ADDICTIVE!

VOTED #1

SOFT DRINK OF THE 31<sup>ST</sup> CENTURY!



SLURM MCKENZIE

```
[vanilla@wheeler ~]$ squeue
```

	JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
	159914	normal	co-mcpdf	nsharma2	CG	2-00:00:26	1	wheeler257
	159915	normal	co-mcpdf	nsharma2	CG	2-00:00:26	1	wheeler257
	159916	normal	co-mcpdf	nsharma2	CG	2-00:00:26	1	wheeler257
	159917	normal	co-mcpdf	nsharma2	CG	2-00:00:26	1	wheeler257
	159918	normal	co-mcpdf	nsharma2	CG	2-00:00:26	1	wheeler257
	159919	normal	co-mcpdf	nsharma2	CG	2-00:00:26	1	wheeler257
	159912	normal	co-mcpdf	nsharma2	CG	2-00:00:28	1	wheeler257
	159913	normal	co-mcpdf	nsharma2	CG	2-00:00:28	1	wheeler257
	166800_[21-100%10]	normal	Jannat	jannat	PD	0:00	1	(JobArrayTaskLimit)
	167067	normal	WINDENER	rubeldas	PD	0:00	36	
	(QOSMaxCpuPerUserLimit)							
	167068	normal	WINDENER	rubeldas	PD	0:00	24	

Slurm....

```
[vanilla@wheeler ~]$ srun --partition debug --nodes 2 hostname
srun: Account not specified in script or
~/default_slurm_account, using latest project
wheeler302.alliance.unm.edu
You have not been allocated GPUs. To request GPUs, use the -G
option in your submission script.
wheeler301.alliance.unm.edu
[vanilla@wheeler ~]$ srun --partition debug --nodes 2 hostname
```

The srun command...

# We finally used the HPC Cluster!



```
[vanilla@wheeler ~]$ srun --partition debug --nodes 2 hostname
srun: Account not specified in script or
~/default_slurm_account, using latest project
wheeler302.alliance.unm.edu
You have not been allocated GPUs. To request GPUs, use the -G
option in your submission script.
wheeler301.alliance.unm.edu
[vanilla@wheeler ~]$ srun --partition debug --nodes 2 hostname
```

The srun command...

```
[vanilla@wheeler ~]$ srun --partition debug --ntasks 8 hostname
srun: Account not specified in script or ~/.default_slurm_account, using
latest project
wheeler302.alliance.unm.edu
wheeler302.alliance.unm.edu
wheeler302.alliance.unm.edu
wheeler302.alliance.unm.edu
wheeler302.alliance.unm.edu
wheeler302.alliance.unm.edu
You have not been allocated GPUs. To request GPUs, use the -G option in your
submission script.
wheeler302.alliance.unm.edu
wheeler302.alliance.unm.edu
```

The srun command...

```
[vanilla@wheeler ~]$ cp -r /projects/shared/workshops/beginner/vecadd ~  
[vanilla@wheeler ~]$
```

Review, what does this command do?

```
[vanilla@wheeler ~]$ cd vecadd/  
[vanilla@wheeler ~/vecadd]$ module load openmpi/4.1.2-q2zi
```

What do these commands do?

```
[vanilla@wheeler ~/vecadd] $ srun --partition debug --ntasks 4 vecaddmpi
```

Now run the program with  
“srun”...

```
[vanilla@wheeler ~]$ qgrok
```

queues	free	busy	offline	jobs	nodes	CPUs
normal	0	299	1	97	300	2400
debug	4	0	0	0	4	32
totals:	4	299	1	97	304	2432

srun is good but HPC centers are  
busy!

# Workflow

Head Node

User 1

Program A

Script A

User 2

Program B

Script B

Compute Node 01

Compute Node 02

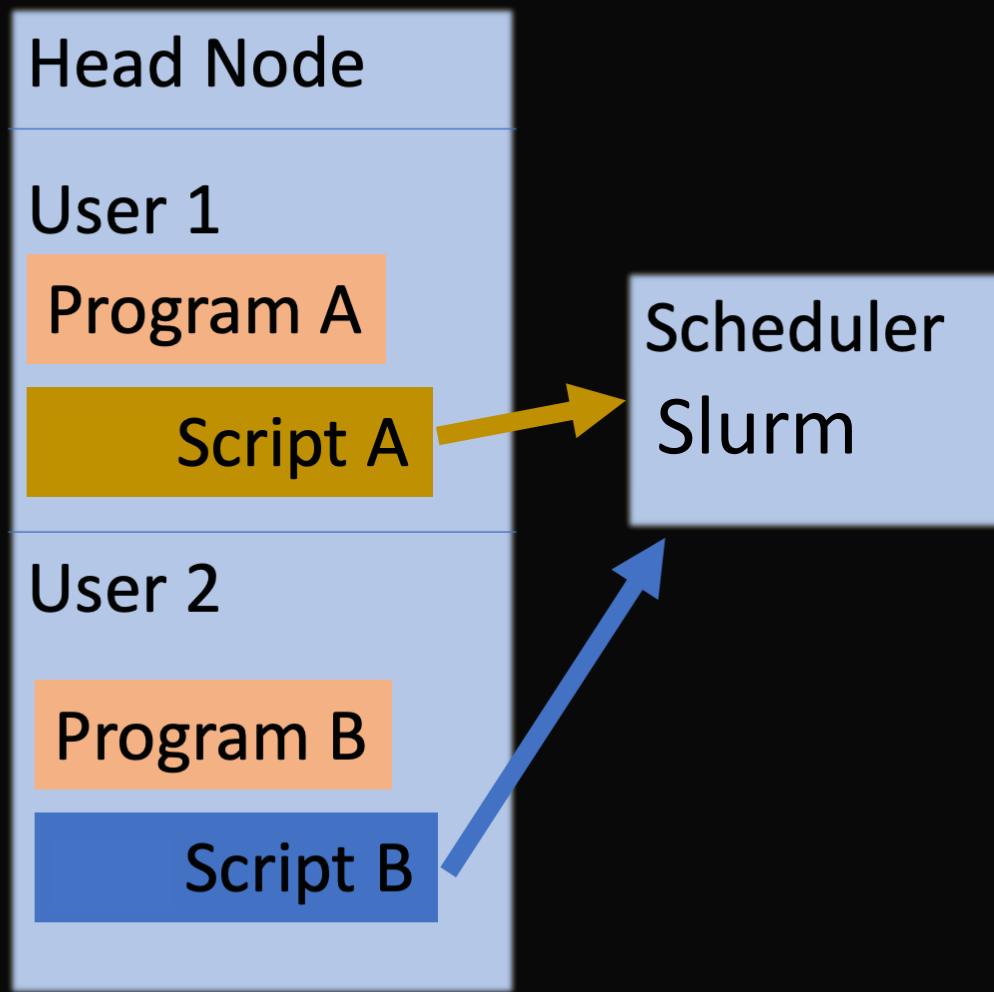
Compute Node 03

Compute Node 04

Compute Node 05

Shared filesystems – All nodes can access the same programs and write output

# Workflow



Compute Node 01

Compute Node 02

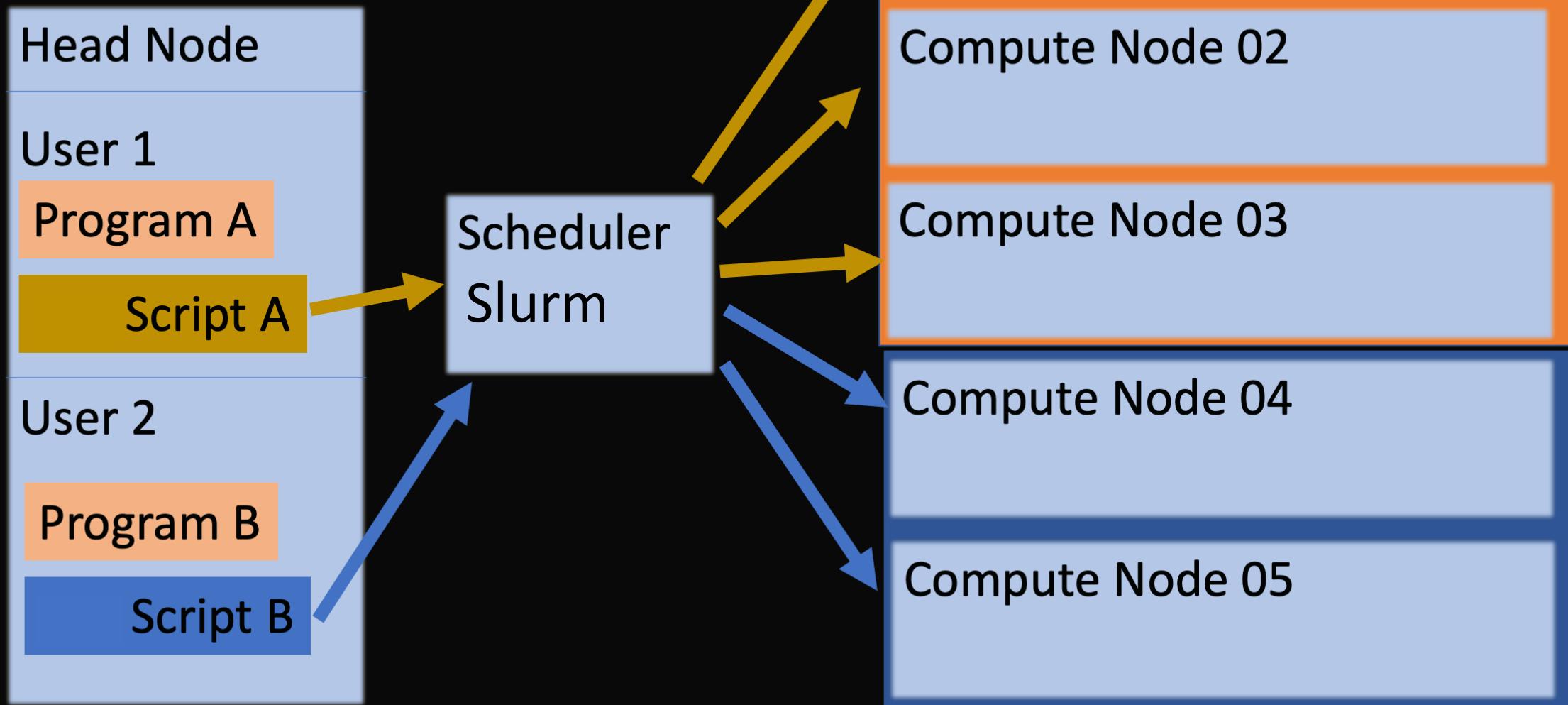
Compute Node 03

Compute Node 04

Compute Node 05

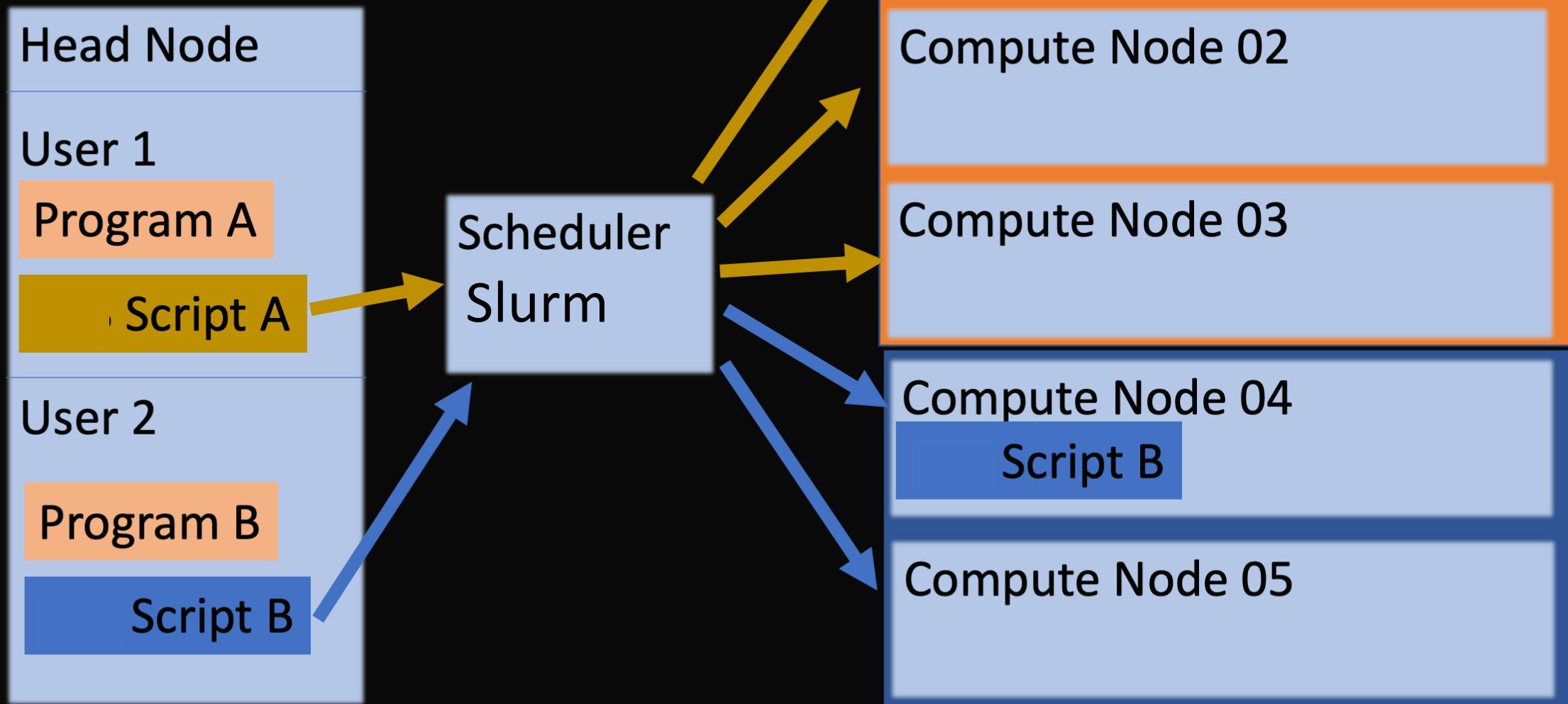
Shared filesystems – All nodes can access the same programs and write output

# Workflow



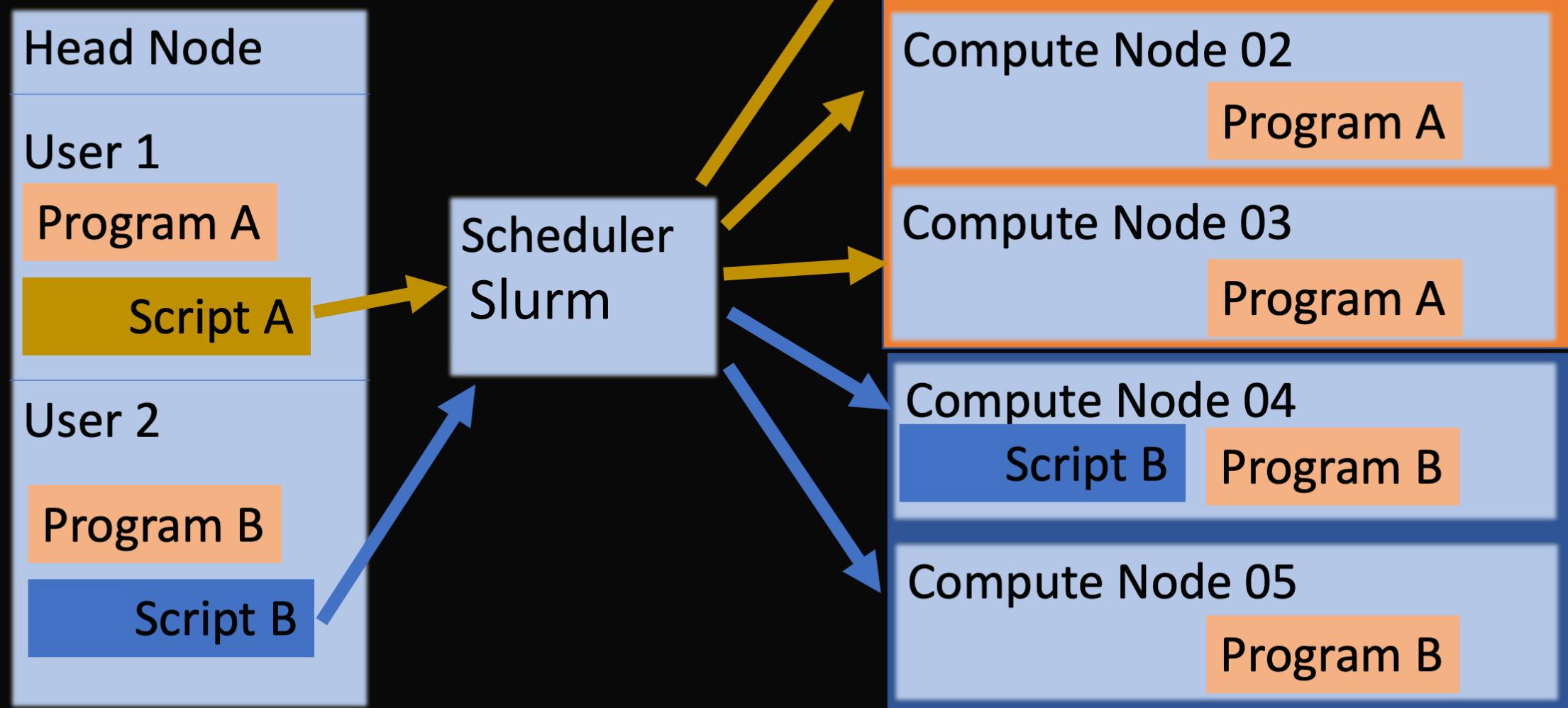
Shared filesystems – All nodes can access the same programs and write output

# Workflow



Shared filesystems – All nodes can access the same programs and write output

# Workflow



Shared filesystems – All nodes can access the same programs and write output

qgrok

sbatch

squeue -u USERNAME

Scheduler

# Running Programs on Compute Nodes

- qgrok
- Intro to the Slurm Scheduler
- The srun command
- sbatch
- sinfo
- squeue
- squeue -u username --start
- <https://www.cism.ucl.ac.be/Services/Formations/slurm/2016/slurm.pdf>
- <https://www.nrel.gov/hpc/assets/pdfs/slurm-advanced-topics.pdf>

The screenshot shows a terminal window with the following details:

- Title Bar:** Matthew — ssh wheeler — 71x19
- Menu Bar:** File Edit Options Buffers Tools Sh-Script Help
- Content:** A Slurm script with syntax highlighting. The code includes Slurm directives (#SBATCH) and a command (sleep).

```
#!/bin/bash
#SBATCH --job-name=demo
#SBATCH --ntasks=4
#SBATCH --time=00:10:00
#SBATCH --mem-per-cpu=4G
#SBATCH --mail-user=yourusername@unm.edu
#SBATCH --mail-type=All

# Enter the commands you want to run below here:
sleep 60
echo Hello from node $HOSTNAME
```

## Slurm Script

```
[vanilla@wheeler ~/vecadd]$ cat vecaddmpi.sh
#!/bin/bash
#SBATCH --job-name=vecaddmpi
#SBATCH --ntasks=4
#SBATCH --time=00:10:00
#SBATCH --mem-per-cpu=4G
#SBATCH --mail-user=mfricke@unm.edu
#SBATCH --mail-type=All
#SBATCH --output=vecaddmpi.out

module load openmpi/4.1.2-q2zi
srun ./vecaddmpi
```

Slurm Script

```
vanilla@wheeler:~/vecadd $ sbatch vecaddmpi.sh
sbatch: Using account 2016199 from ~/.default_slurm_account
Submitted batch job 167571
```

```
vanilla@wheeler:~/vecadd $ squeue --me
              JOBID PARTITION      NAME     USER   ST      TIME  NODES
NODELIST(REASON)
        167571    normal  vecaddmp  vanilla  R      0:07      1 Wheeler145
```

## Slurm Script

```
vanilla@wheeler:~/vecadd $ tail -f vecaddmpi.out
You have not been allocated GPUs. To request GPUs, use the -G option in your
submission script.
Assigning compute node to rank 1.
ComputeNode: Starting with rank 1.
ComputeNode (1): Waiting for vectors from dataserver with rank 3...
Assigning compute node to rank 2.
ComputeNode: Starting with rank 2.
ComputeNode (2): Waiting for vectors from dataserver with rank 3...
Will try to allocate a vector of size 1 GB.
```

## Slurm Script

```
vanilla@wheeler:~/vecadd $ tail -f vecaddmpi.out
You have not been allocated GPUs. To request GPUs, use the -G option in your
submission script.
Assigning compute node to rank 1.
ComputeNode: Starting with rank 1.
ComputeNode (1): Waiting for vectors from dataserver with rank 3...
Assigning compute node to rank 2.
ComputeNode: Starting with rank 2.
ComputeNode (2): Waiting for vectors from dataserver with rank 3...
Will try to allocate a vector of size 1 GB.
```

## Slurm Script

# Useful Slurm Commands

squeue --me --long	shows information about jobs you submitted
squeue --me --start	shows when slurm expects your job to start
scancel jobid	cancels a job
sacct	shows your job history

```
vanilla@wheeler:~/vecadd $ seff 167573
Job ID: 167573
Cluster: wheeler
User/Group: mfricke/users
State: COMPLETED (exit code 0)
Nodes: 1
Cores per node: 4
CPU Utilized: 00:01:03
CPU Efficiency: 78.75% of 00:01:20 core-walltime
Job Wall-clock time: 00:00:20
Memory Utilized: 39.55 MB (estimated maximum)
Memory Efficiency: 0.24% of 16.00 GB (4.00 GB/core)
```

Slurm Script

# CARC Resources

- Tutorial Videos
- Written Tutorials



## QuickBytes

Quickbytes are tutorials designed to help CARC users.

- Linux-Intro
- Running jobs
  - Logging in
  - SSH keys and Config file
  - Transferring data
  - PBS/TORQUE
  - Sample PBS script
  - Submitting jobs
  - Check running jobs
  - Managing modules
  - Intro to Slurm
  - Converting PBS to Slurm
  - Intro to Slurm accounting at CARC

A screenshot of a YouTube channel page for 'UNMCARC'. The channel name is 'QuickBytes'. It has 16 videos and 2,870 views, last updated on Jan 5, 2022. A note on the page states: 'Short Tutorials on CARC Systems - DUE TO THE RECENT WHEELER UPGRADE THESE VIDEOS WILL BE REPLACED SOON.' The channel has a logo for 'UNM CENTER FOR ADVANCED RESEARCH COMPUTING'. There are 6 video thumbnails listed:

1. Intro to the UNM Center for Advanced Research Computing (5:53)
2. Projects and Accounts (8:47)
3. Logging into CARC Systems (12:48)
4. Storage Systems (13:11)
5. Transferring data (21:56)
6. Environment Modules (6:04)

A red 'SUBSCRIBE' button is located below the video list.

# Getting Help

help@carc.unm.edu

Office hours

# Bonus – sview and graphical programs