Introduction to Slurm & Slurm batch scripts

Ashley Dederich & Emilie Parra
Research Consulting & Faculty Engagement
Center for High Performance Computing
ashley.dederich@utah.edu

Overview of Talk

- What is Slurm, and why use it?
- Preparing a Slurm job
  - Accounts and Partitions
  - CHPC Storage Resources
  - Slurm Environment Variables
- Slurm batch directives
- Basic Slurm Commands
- Running an Interactive Batch job
- Using GPU Nodes
- Job Priority & Performance

Overview of Talk

- What is Slurm, and why use it?
- Preparing a Slurm job
  - Accounts and Partitions
  - CHPC Storage Resources
  - Slurm Environment Variables
- Slurm batch directives
- Basic Slurm Commands
- Running an Interactive Batch job
- Using GPU Nodes
- Job Priority & Performance

Re-cap of Resources

- **CHPC resources:**
  - HPC clusters:
    - General Environment: notchpeak, kingspeak, lonepeak, ash
    - Protected Environment (PE): redwood
    - Others
  - VM (Windows, Linux)
  - Storage
  - Services
- **Condominium mode:**
  - HPC Cluster = CHPC-owned nodes (general nodes) + PI-owned nodes (owner nodes)
  - All CHPC users have access to CHPC-owned resources for free. Some clusters (notchpeak) need allocations (peer-reviewed proposals)
  - Owners (PI group) have the highest priority using owner nodes
  - All CHPC users have access to owner nodes in Guest mode for free (jobs subject to preemption)

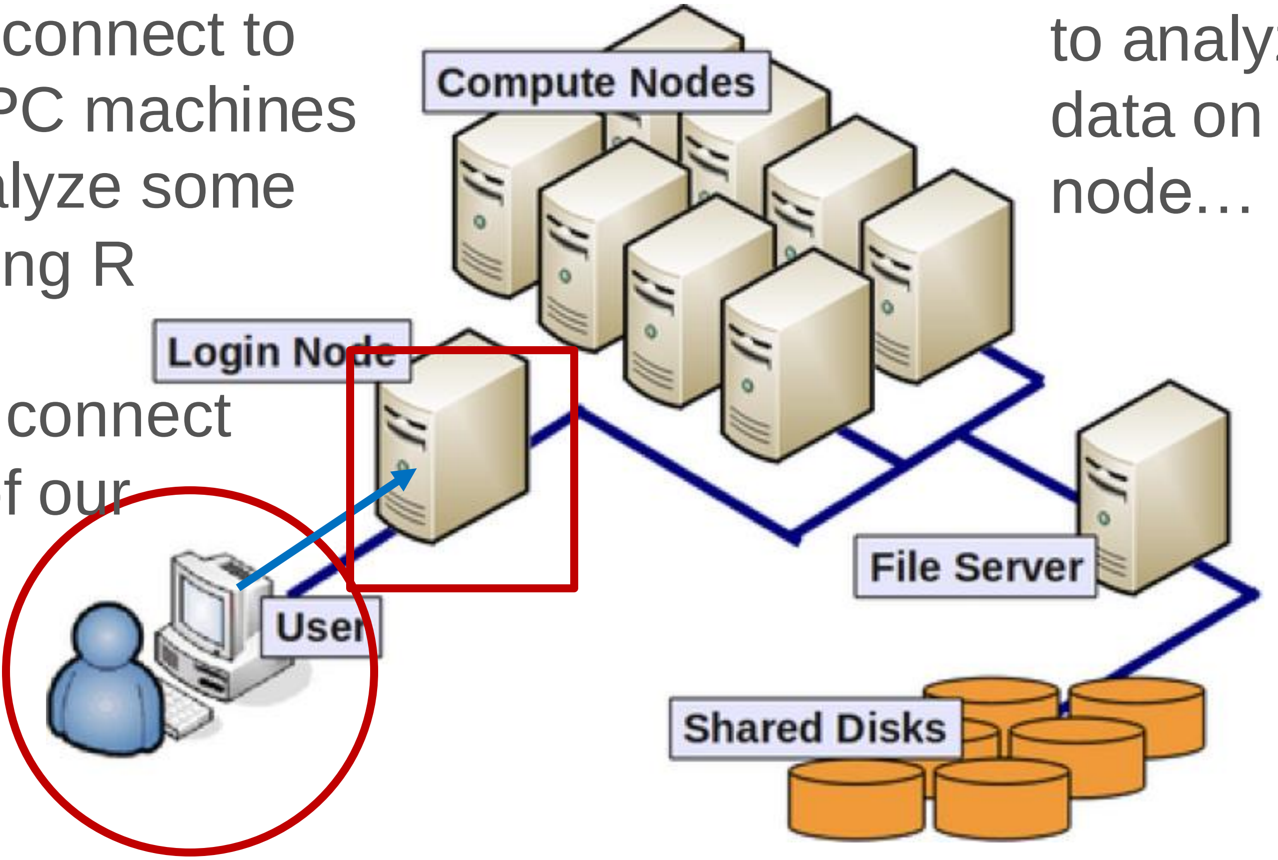
What is Slurm

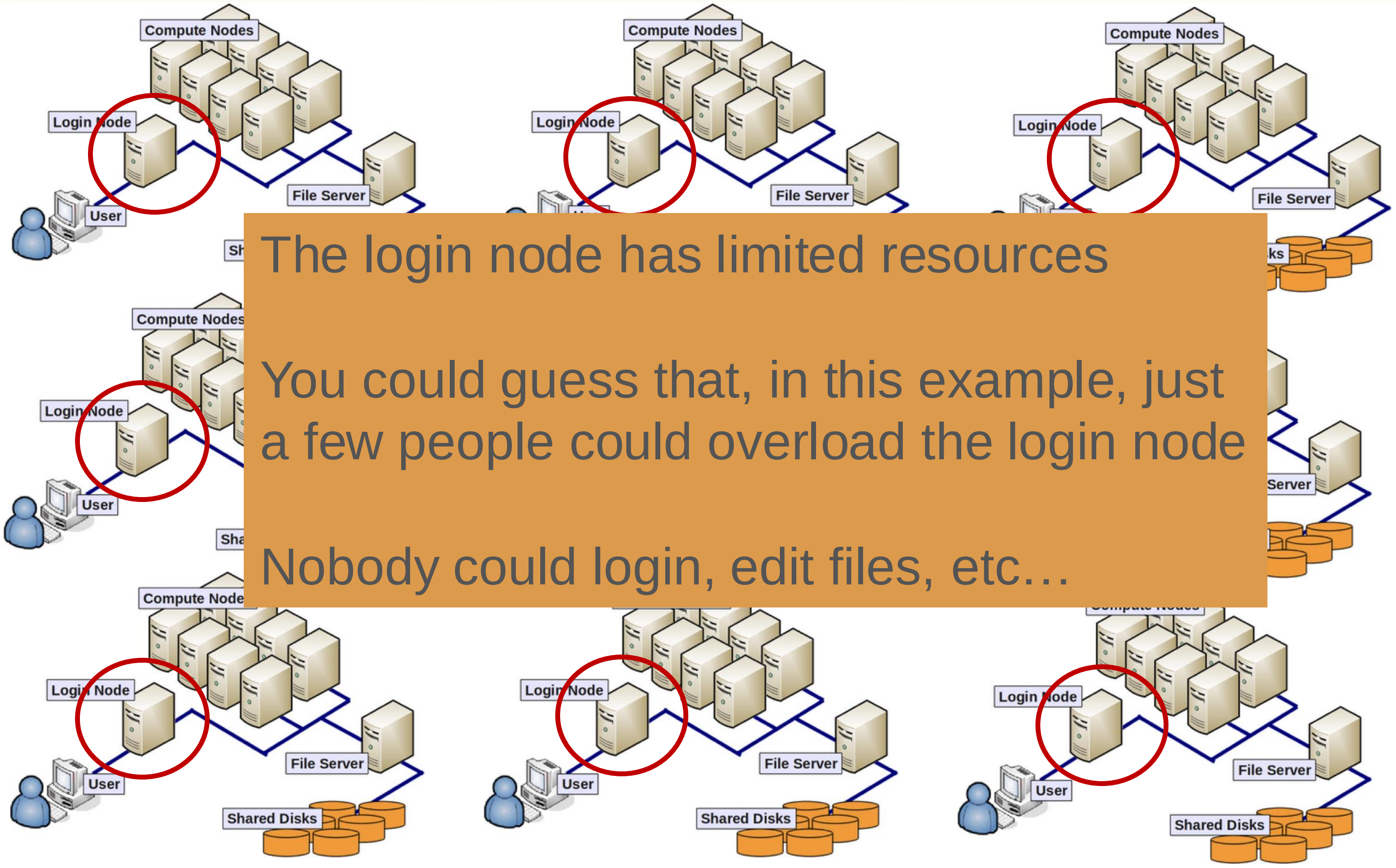
…and why use it?

Goal: you, the user, want to connect to the CHPC machines and analyze some data using R

You don't want to analyze your data on the login node…

So, you connect to one of our clusters



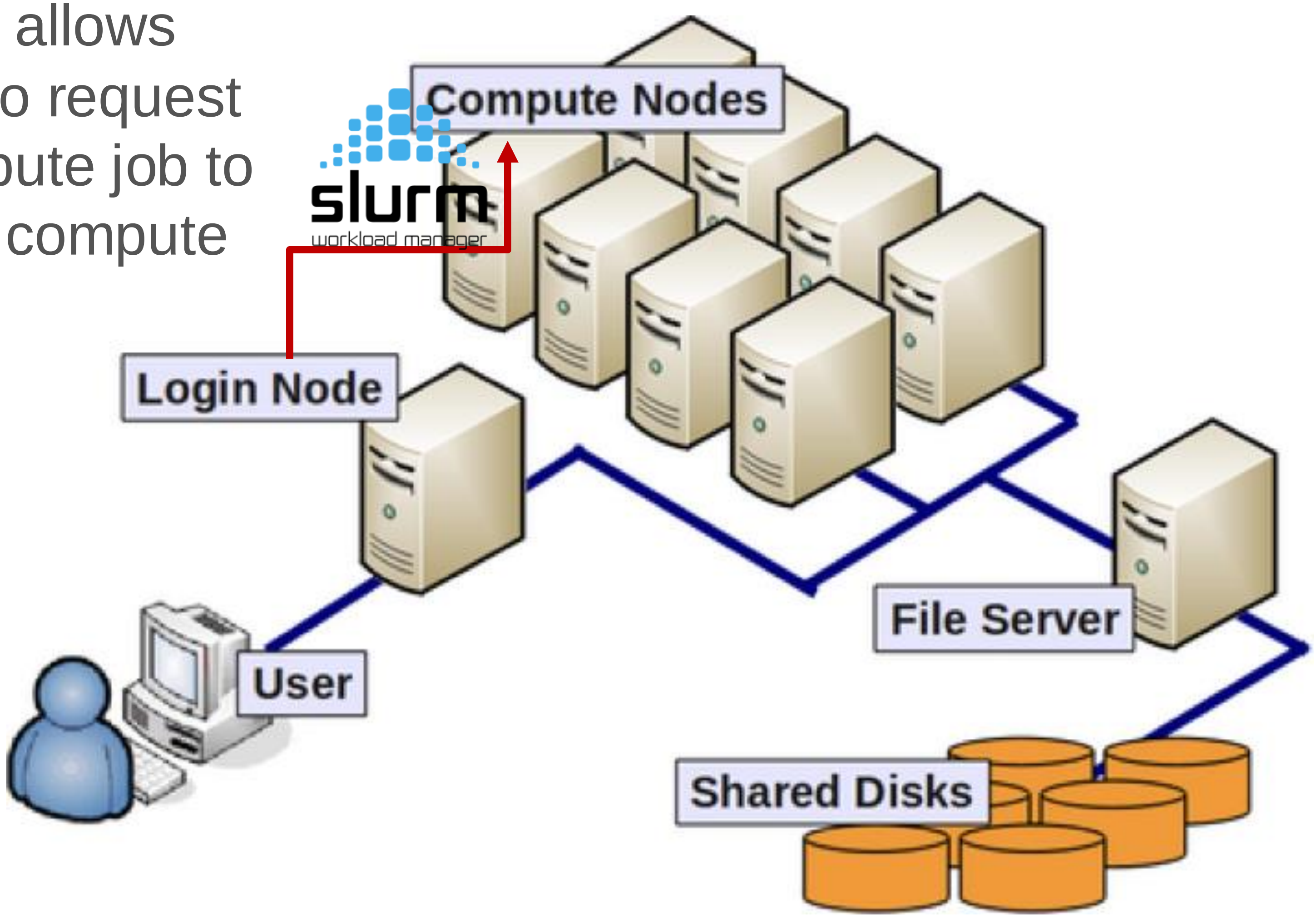


The login node has limited resources

You could guess that, in this example, just a few people could overload the login node

Nobody could login, edit files, etc…

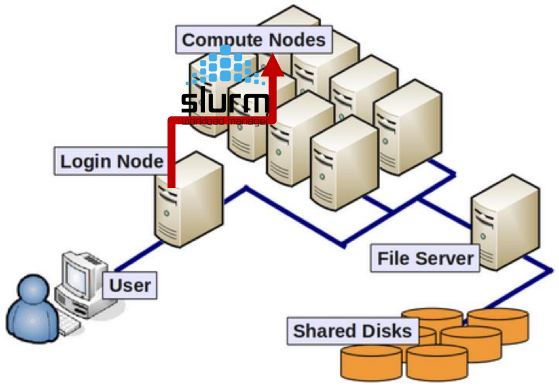
*Slurm allows users to request a compute job to run on compute nodes



What is Slurm

…and why use it?

- So, how do we ask Slurm to submit a job?
- We need to ask for the correct resources
- But first, we need to know what those resources are…



Overview of Talk

- What is Slurm, and why use it?
- Preparing a Slurm job
  - Accounts and Partitions
  - CHPC Storage Resources
  - Slurm Environment Variables
- Slurm batch directives
- Basic Slurm Commands
- Running an Interactive Batch job
- Using GPU Nodes
- Job Priority & Performance

```
#!/bin/bash
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#set up the temporary directory
SCRDIR=/scratch/general/vast/$USER/$SLURM_JOB_ID
mkdir -p $SCRDIR
#copy over input files
cp file.input $SCRDIR/.
cd $SCRDIR
#Set up whatever package we need to run with
module load <some-module>
#Run the program with our input
myprogram < file.input > file.output
#Move files out of working directory and clean up
cp file.output $HOME/.
cd $HOME
rm -rf $SCRDIR
```

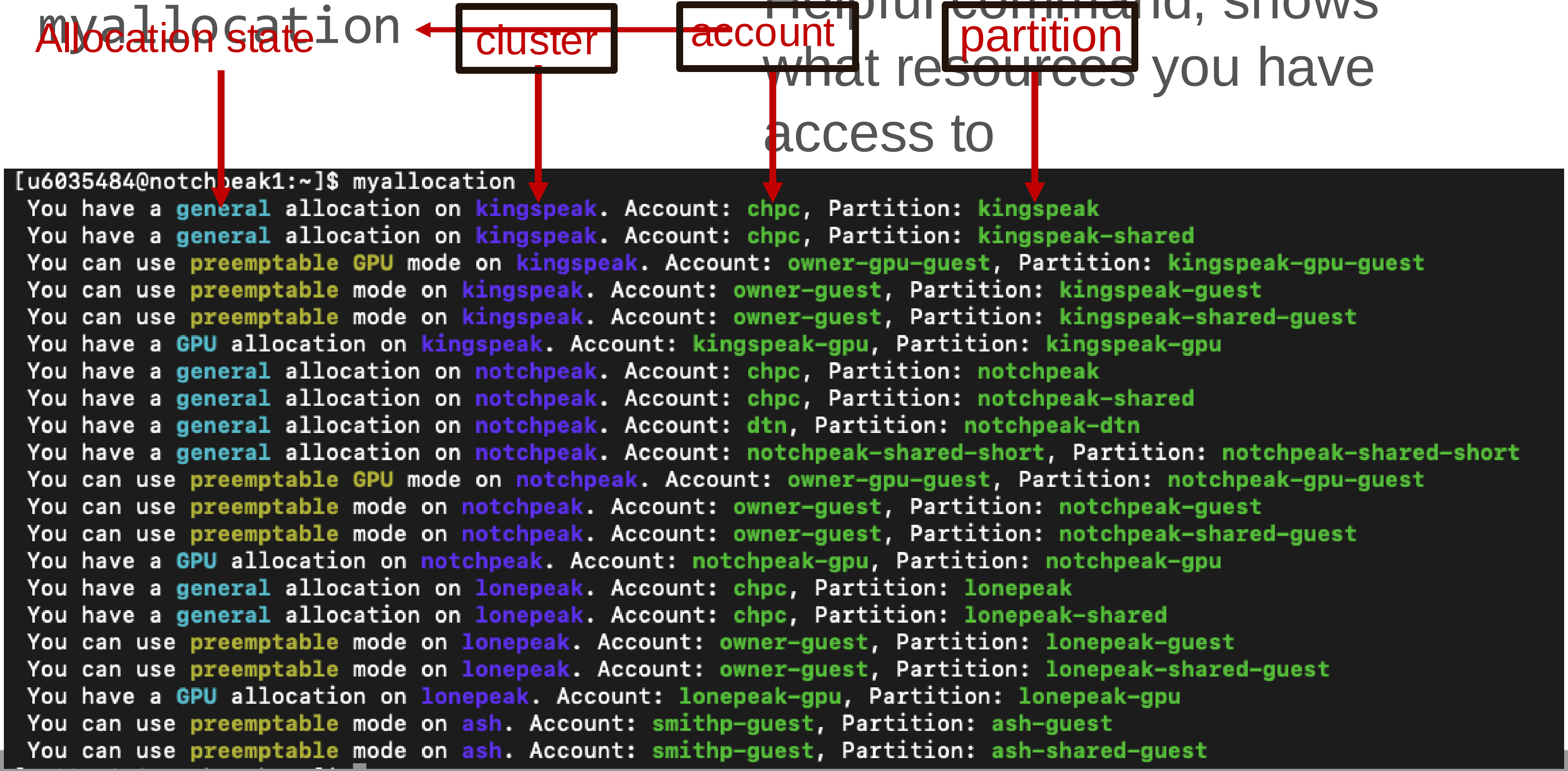
```
#!/bin/tcsh
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#set up the scratch directory
set SCRDIR /scratch/local/$USER/$SLURM_JOB_ID
mkdir -p $SCRDIR
#move input files into scratch directory
cp file.input $SCRDIR/.
cd $SCRDIR
#Set up whatever package we need to run with
module load <some-module>
#Run the program with our input
myprogram < file.input > file.output
#Move files out of working directory and clean up
cp file.output $HOME/.
cd $HOME
rm -rf $SCRDIR
```

```
#!/bin/bash
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
```

```
#!/bin/tcsh
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
```

Preparing a Slurm Job

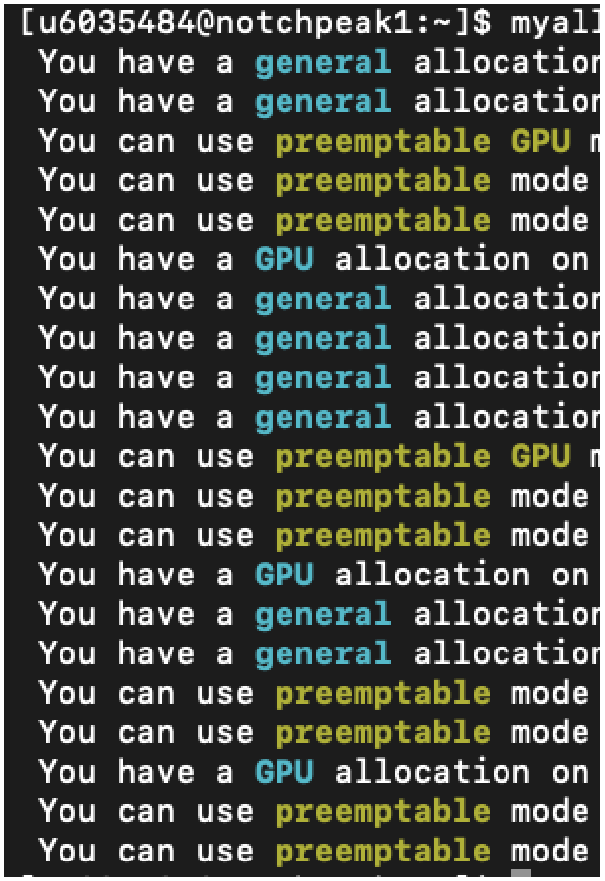
- `myallocation` – Helpful command; shows what resources you have access to



```
[u6035484@notchpeak1:~]$ myallocation
 You have a general allocation on kingspeak. Account: chpc, Partition: kingspeak
 You have a general allocation on kingspeak. Account: chpc, Partition: kingspeak-shared
 You can use preemptable GPU mode on kingspeak. Account: owner-gpu-guest, Partition: kingspeak-gpu-guest
 You can use preemptable mode on kingspeak. Account: owner-guest, Partition: kingspeak-guest
 You can use preemptable mode on kingspeak. Account: owner-guest, Partition: kingspeak-shared-guest
 You have a GPU allocation on kingspeak. Account: kingspeak-gpu, Partition: kingspeak-gpu
 You have a general allocation on notchpeak. Account: chpc, Partition: notchpeak
 You have a general allocation on notchpeak. Account: chpc, Partition: notchpeak-shared
 You have a general allocation on notchpeak. Account: dtn, Partition: notchpeak-dtn
 You have a general allocation on notchpeak. Account: notchpeak-shared-short, Partition: notchpeak-shared-short
 You can use preemptable GPU mode on notchpeak. Account: owner-gpu-guest, Partition: notchpeak-gpu-guest
 You can use preemptable mode on notchpeak. Account: owner-guest, Partition: notchpeak-guest
 You can use preemptable mode on notchpeak. Account: owner-guest, Partition: notchpeak-shared-guest
 You have a GPU allocation on notchpeak. Account: notchpeak-gpu, Partition: notchpeak-gpu
 You have a general allocation on lonepeak. Account: chpc, Partition: lonepeak
 You have a general allocation on lonepeak. Account: chpc, Partition: lonepeak-shared
 You can use preemptable mode on lonepeak. Account: owner-guest, Partition: lonepeak-guest
 You can use preemptable mode on lonepeak. Account: owner-guest, Partition: lonepeak-shared-guest
 You have a GPU allocation on lonepeak. Account: lonepeak-gpu, Partition: lonepeak-gpu
 You can use preemptable mode on ash. Account: smithp-guest, Partition: ash-guest
 You can use preemptable mode on ash. Account: smithp-guest, Partition: ash-shared-guest
```

Allocation State

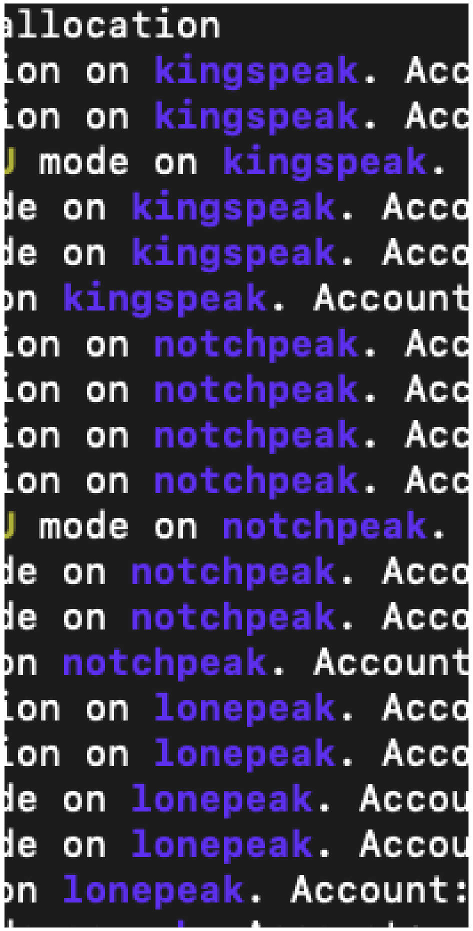
- Three allocation states:
  - General: you can run jobs on that cluster with no issues
  - Preemptable: you can still run jobs, but they are subject to *preemption*.
  - Owner: you own a node and have priority access to it (preempt guest jobs)
- Preemption: your job will run on that node until another job requests that same node, at which point your job is automatically cancelled.



```
[u6035484@notchpeak1:~]$ myall
 You have a general allocation
 You have a general allocation
 You can use preemptable GPU m
 You can use preemptable mode
 You can use preemptable mode
 You have a GPU allocation on
 You have a general allocation
 You have a general allocation
 You have a general allocation
 You have a general allocation
 You can use preemptable GPU m
 You can use preemptable mode
 You can use preemptable mode
 You have a GPU allocation on
 You have a general allocation
 You have a general allocation
 You can use preemptable mode
 You can use preemptable mode
 You have a GPU allocation on
 You can use preemptable mode
 You can use preemptable mode
```

Cluster

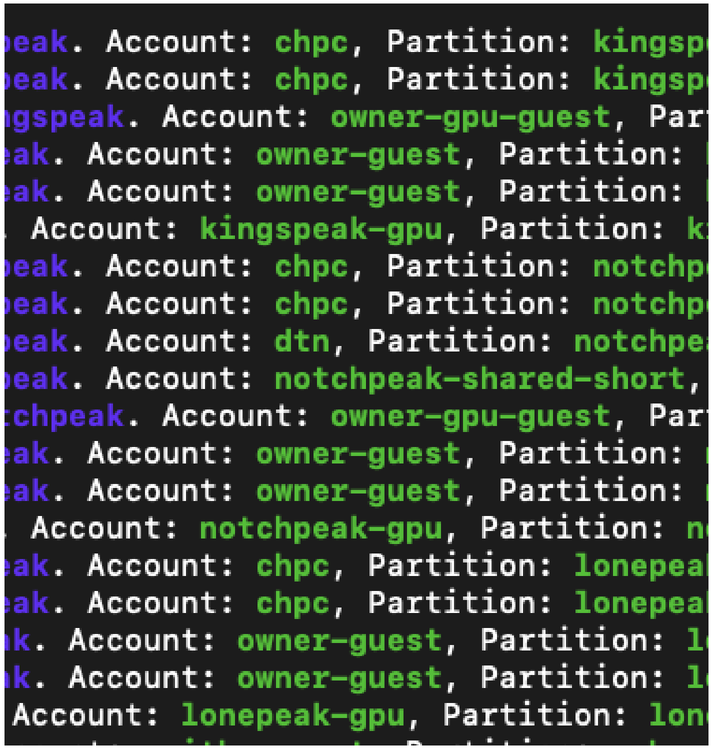
- We currently have four *general environment* clusters:
  - Notchpeak
  - Kingspeak
  - Lonepeak
  - Ash (guest access only)
- We have one *protected environment* cluster:
  - Redwood



```
allocation
ion on kingspeak. Acc
ion on kingspeak. Acc
J mode on kingspeak.
de on kingspeak. Acco
de on kingspeak. Acco
on kingspeak. Account
ion on notchpeak. Acc
ion on notchpeak. Acc
ion on notchpeak. Acc
ion on notchpeak. Acc
J mode on notchpeak.
de on notchpeak. Acco
de on notchpeak. Acco
on notchpeak. Account
ion on lonepeak. Acco
ion on lonepeak. Acco
de on lonepeak. Accou
de on lonepeak. Accou
on lonepeak. Account:
```

Account

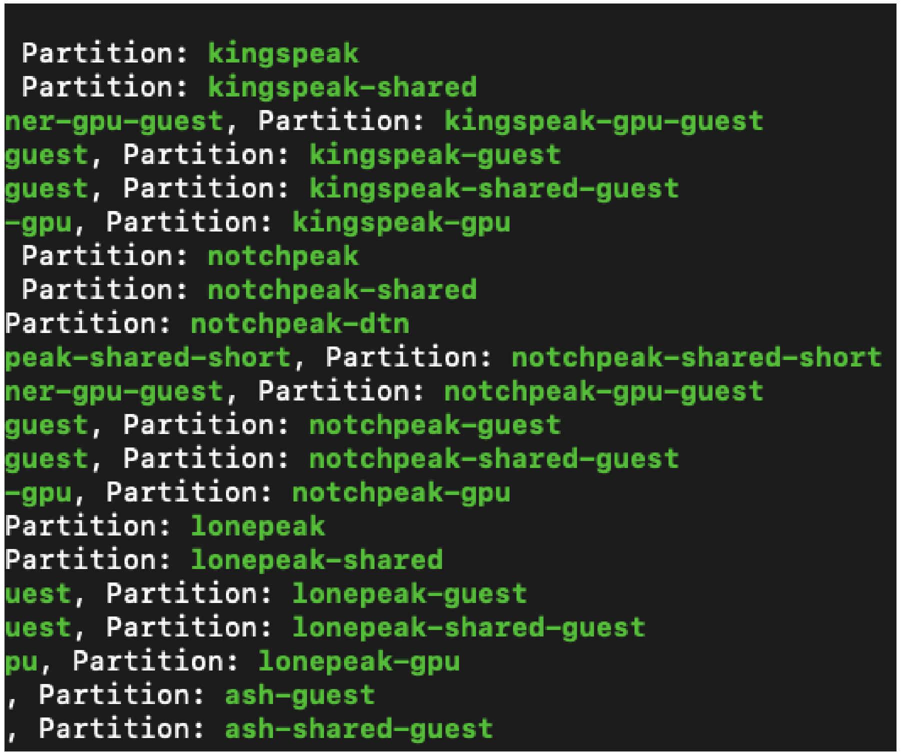
- **Account**: to limit and track resource utilization at user/group level.
- A user/group can have multiple Slurm accounts
  - each represents different privileges.



```
eak. Account: chpc, Partition: kingsp
eak. Account: chpc, Partition: kingsp
gspeak. Account: owner-gpu-guest, Par
ak. Account: owner-guest, Partition:
ak. Account: owner-guest, Partition:
 Account: kingspeak-gpu, Partition: k
eak. Account: chpc, Partition: notchp
eak. Account: chpc, Partition: notchp
eak. Account: dtn, Partition: notchpe
eak. Account: notchpeak-shared-short,
chpeak. Account: owner-gpu-guest, Par
ak. Account: owner-guest, Partition:
ak. Account: owner-guest, Partition:
 Account: notchpeak-gpu, Partition: n
ak. Account: chpc, Partition: lonepea
ak. Account: chpc, Partition: lonepea
k. Account: owner-guest, Partition: l
k. Account: owner-guest, Partition: l
Account: lonepeak-gpu, Partition: lon
```

Partition

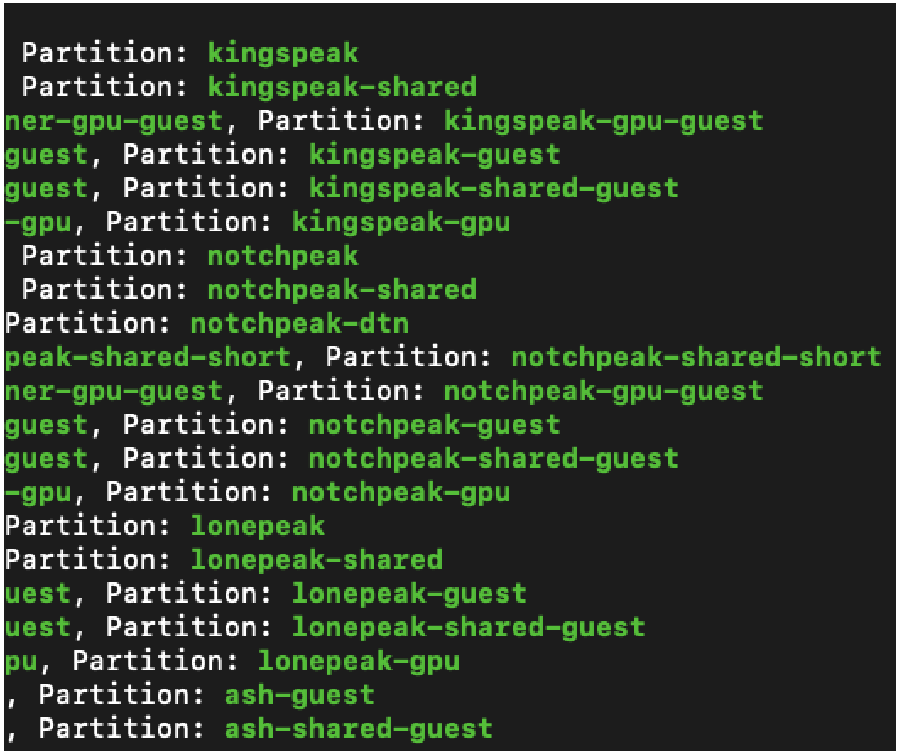
- Refers to a set of nodes with specific resources:
- <cluster> : whole node(s) to yourself
- <cluster>-shared: share a node with other job(s)
- <cluster>-guest: use owner nodes, subject to preemption
- <cluster>-shared-guest: share owner nodes with other jobs, subject to preemption
- <cluster>-gpu: use nodes with GPUs



```
 Partition: kingspeak
 Partition: kingspeak-shared
ner-gpu-guest, Partition: kingspeak-gpu-guest
guest, Partition: kingspeak-guest
guest, Partition: kingspeak-shared-guest
-gpu, Partition: kingspeak-gpu
 Partition: notchpeak
 Partition: notchpeak-shared
Partition: notchpeak-dtn
peak-shared-short, Partition: notchpeak-shared-short
ner-gpu-guest, Partition: notchpeak-gpu-guest
guest, Partition: notchpeak-guest
guest, Partition: notchpeak-shared-guest
-gpu, Partition: notchpeak-gpu
Partition: lonepeak
Partition: lonepeak-shared
uest, Partition: lonepeak-guest
uest, Partition: lonepeak-shared-guest
pu, Partition: lonepeak-gpu
, Partition: ash-guest
, Partition: ash-shared-guest
```

Partition

- Refers to a set of nodes with specific resources:
- <cluster> : whole node(s) to yourself
- <cluster>-shared: share a node with other job(s)
- <cluster>-guest: use owner nodes, subject to preemption
- <cluster>-shared-guest: share owner nodes with other jobs, subject to preemption
- <cluster>-gpu: use nodes with GPUs *



```
 Partition: kingspeak
 Partition: kingspeak-shared
ner-gpu-guest, Partition: kingspeak-gpu-guest
guest, Partition: kingspeak-guest
guest, Partition: kingspeak-shared-guest
-gpu, Partition: kingspeak-gpu
 Partition: notchpeak
 Partition: notchpeak-shared
Partition: notchpeak-dtn
peak-shared-short, Partition: notchpeak-shared-short
ner-gpu-guest, Partition: notchpeak-gpu-guest
guest, Partition: notchpeak-guest
guest, Partition: notchpeak-shared-guest
-gpu, Partition: notchpeak-gpu
Partition: lonepeak
Partition: lonepeak-shared
uest, Partition: lonepeak-guest
uest, Partition: lonepeak-shared-guest
pu, Partition: lonepeak-gpu
, Partition: ash-guest
, Partition: ash-shared-guest
```

Exception: GPU partitions are all in Shared mode (even with no "-shared" in names)

Node Sharing

- Use **Shared Partition** wherever possible
  - Save your group allocations/credits
  - Shorten queueing time for you and others
  - Help increase utilization and save energy/environment

https://www.chpc.utah.edu/documentation/software/node-sharing.php

```
#!/bin/bash
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
```

```
#!/bin/tcsh
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
```

#SBATCH --time=02:00:00 specifies wall time of a job in Hours:Minutes:Seconds

#SBATCH -t 02:00:00
also works

```
#!/bin/bash
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
```

```
#!/bin/tcsh
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
```

#SBATCH --nodes=1 specifies number of nodes

#SBATCH -N 1
also works

```
#!/bin/bash
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
```

```
#!/bin/tcsh
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
```

#SBATCH --ntasks=8 total number of tasks (cpu cores) (or -n)

#SBATCH -n 8
also works

```
#!/bin/bash
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
```

```
#!/bin/tcsh
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
```

#SBATCH --mem=32GB specifies total memory *per node*

#SBATCH --mem=0
gives you memory of whole node

```
#!/bin/bash
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
```

```
#!/bin/tcsh
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
```

#SBATCH -o outputs standard output in the form slurmjob-<JOBID>.out-<NODEID>

#SBATCH -e outputs error messages in the form slurmjob-<JOBID>.err-<NODEID>

```bash
#!/bin/bash
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#set up the temporary directory
SCRDIR=/scratch/general/vast/$USER/$SLURM_JOB_ID
mkdir -p $SCRDIR
#copy over input files
cp file.input $SCRDIR/.
cd $SCRDIR
```

```tcsh
#!/bin/tcsh
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#set up the scratch directory
set SCRDIR /scratch/local/$USER/$SLURM_JOB_ID
mkdir -p $SCRDIR
#move input files into scratch directory
cp file.input $SCRDIR/.
cd $SCRDIR
```

Now, we will discuss the best way to stage your files for analysis

Overview of Talk

- What is Slurm, and why use it?
- Preparing a Slurm job
  - Accounts and Partitions
  - CHPC Storage Resources
  - Slurm Environment Variables
- Slurm batch directives
- Basic Slurm Commands
- Running an Interactive Batch job
- Using GPU Nodes
- Job Priority & Performance

CHPC Storage Resources

Home

- Free
- Automatically provisioned
- 50GB soft limit

Scratch

- Free
- For intermediate files required during a job
- **vast** – 50TB/user quota
- **nfs1** – no quota

Group

- Needs to be purchased by PI
- By the TB

```
#!/bin/bash
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#set up the temporary directory
SCRDIR=/scratch/general/vast/$USER/$SLURM_JOB_ID
mkdir -p $SCRDIR
#copy over input files
cp file.input $SCRDIR/.
cd $SCRDIR
```

Points to your uNID

```
#!/bin/tcsh
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#set up the scratch directory
set SCRDIR /scratch/local/$USER/$SLURM_JOB_ID
mkdir -p $SCRDIR
#move input files into scratch directory
cp file.input $SCRDIR/.
cd $SCRDIR
```

Points to your uNID

Create an environmental variable that points to scratch path

Slurm Environment Variables

- Some useful environment variables:
  - $SLURM_JOB_ID
  - $SLURM_SUBMIT_DIR
  - $SLURM_NNODES
  - $SLURM_NTASKS
- Can get them for a given set of directives by using the **env** command inside a script (or in a srun session).

See: https://slurm.schedmd.com/sbatch.html#SECTION_OUTPUT-ENVIRONMENT-VARIABLES

```bash
#!/bin/bash
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#set up the temporary directory
SCRDIR=/scratch/general/vast/$USER/$SLURM_JOB_ID
mkdir -p $SCRDIR
#copy over input files
cp file.input $SCRDIR/.
cd $SCRDIR
```

```tcsh
#!/bin/tcsh
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#set up the scratch directory
set SCRDIR /scratch/local/$USER/$SLURM_JOB_ID
mkdir -p $SCRDIR
#move input files into scratch directory
cp file.input $SCRDIR/.
cd $SCRDIR
```

Create the scratch directory

```
#!/bin/bash
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#set up the temporary directory
SCRDIR=/scratch/general/vast/$USER/$SLURM_JOB_ID
mkdir -p $SCRDIR
#copy over input files
cp file.input $SCRDIR/.
cd $SCRDIR
```

```
#!/bin/tcsh
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#set up the scratch directory
set SCRDIR /scratch/local/$USER/$SLURM_JOB_ID
mkdir -p $SCRDIR
#move input files into scratch directory
cp file.input $SCRDIR/.
cd $SCRDIR
```

Copy over input files and move on over to $SCRDIR

```
#!/bin/bash
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#set up the temporary directory
SCRDIR=/scratch/general/vast/$USER/$SLURM_JOB_ID
mkdir -p $SCRDIR

#copy over input files
cp file.input $SCRDIR/.
cd $SCRDIR
#Set up whatever package we need to run with
module load <some-module>
```

```
#!/bin/tcsh
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#set up the scratch directory
set SCRDIR /scratch/local/$USER/$SLURM_JOB_ID
mkdir -p $SCRDIR

#move input files into scratch directory
cp file.input $SCRDIR/.
cd $SCRDIR
#Set up whatever package we need to run with
module load <some-module>
```

Load the desired modules

```
#!/bin/bash
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#set up the temporary directory
SCRDIR=/scratch/general/vast/$USER/$SLURM_JOB_ID
mkdir -p $SCRDIR
#copy over input files
cp file.input $SCRDIR/.
cd $SCRDIR
#Set up whatever package we need to run with
module load <some-module>
#Run the program with our input
myprogram < file.input > file.output
```

```
#!/bin/tcsh
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#set up the scratch directory
set SCRDIR /scratch/local/$USER/$SLURM_JOB_ID
mkdir -p $SCRDIR
#move input files into scratch directory
cp file.input $SCRDIR/.
cd $SCRDIR
#Set up whatever package we need to run with
module load <some-module>
#Run the program with our input
myprogram < file.input > file.output
```

Run the program you need to

```
#!/bin/bash
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#set up the temporary directory
SCRDIR=/scratch/general/vast/$USER/$SLURM_JOB_ID
mkdir -p $SCRDIR
#copy over input files
cp file.input $SCRDIR/.
cd $SCRDIR
#Set up whatever package we need to run with
module load <some-module>
#Run the program with our input
myprogram < file.input > file.output
#Move files out of working directory and clean up
cp file.output $HOME/.   <- Copy output to your $HOME
cd $HOME                 <- Move back to $HOME
rm -rf $SCRDIR           <- Remove $SCRDIR
```

```
#!/bin/tcsh
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#set up the scratch directory
set SCRDIR /scratch/local/$USER/$SLURM_JOB_ID
mkdir -p $SCRDIR
#move input files into scratch directory
cp file.input $SCRDIR/.
cd $SCRDIR
#Set up whatever package we need to run with
module load <some-module>
#Run the program with our input
myprogram < file.input > file.output
#Move files out of working directory and clean up
cp file.output $HOME/.   <- Copy output to your $HOME
cd $HOME                 <- Move back to $HOME
rm -rf $SCRDIR           <- Remove $SCRDIR
```

```
#!/bin/bash
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
#SBATCH --mem=32G
#SBATCH -o slurmjob-%j
#SBATCH -e slurmjob-%j
#set up the temporary
SCRDIR=/scratch/gener
mkdir -p $SCRDIR
#copy over input files
cp file.input $SCRDIR
cd $SCRDIR
#Set up whatever package we need to run with
module load <some-module>
#Run the program with our input
myprogram < file.input > file.output
#Move files out of working directory and clean up
cp file.output $HOME/.
cd $HOME
rm -rf $SCRDIR
```

```
#!/bin/tcsh
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-shared-guest
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=8
j.out-%N
j.err-%N
irectory
ocal/$USER/$SLURM_JOB_ID
o scratch directory
/.
cd $SCRDIR
#Set up whatever package we need to run with
module load <some-module>
#Run the program with our input
myprogram < file.input > file.output
#Move files out of working directory and clean up
cp file.output $HOME/.
cd $HOME
rm -rf $SCRDIR
```

Done! Let’s call this file

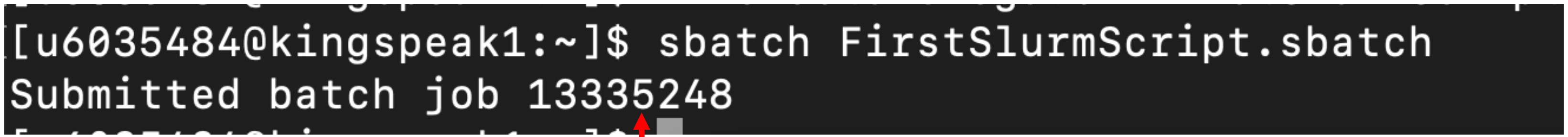
FirstSlurmScript.sbatch

Overview of Talk

- What is Slurm, and why use it?
- Preparing a Slurm job
  - Accounts and Partitions
  - CHPC Storage Resources
  - Slurm Environment Variables
- Slurm batch directives
- Basic Slurm Commands
- Running an Interactive Batch job
- Using GPU Nodes
- Job Priority & Performance

Basic Slurm commands

- **sbatch FirstSlurmScript.sbatch -** launch a batch job



```
[u6035484@kingspeak1:~]$ sbatch FirstSlurmScript.sbatch
Submitted batch job 13335248
```

Job ID

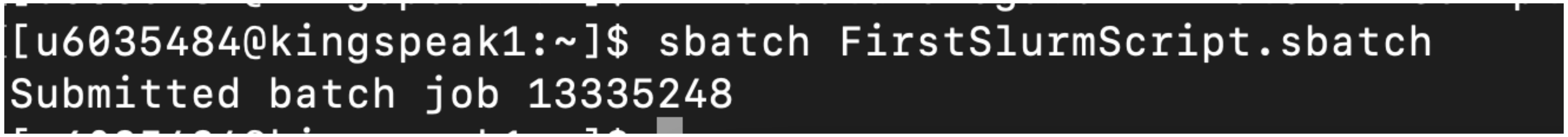
Basic Slurm commands

- **sbatch FirstSlurmScript.sbatch -** launch a batch job
- **squeue** - shows all jobs in queue
  - **squeue --me** - shows only your jobs
  - **squeue -u <uNID>** - shows only your jobs
  - **mysqueue* -** shows job queue per partition and associated accounts you have access to on the cluster

*CHPC developed programs. See CHPC Newsletter 2023 Summer

Basic Slurm commands

- **sbatch FirstSlurmScript.sbatch -** launch a batch job
- **squeue** - shows all jobs in queue
  - **squeue --me** - shows only your jobs
  - **squeue -u <uNID>** - shows only your jobs
  - **mysqueue*** **-** shows job queue per partition and associated accounts you have access to on the cluster
- **scancel <jobid>** - cancel a job



```
[u6035484@kingspeak1:~]$ sbatch FirstSlurmScript.sbatch
Submitted batch job 13335248
```

```
scancel 13335248
```

*CHPC developed programs. See CHPC Newsletter 2023 Summer

Basic Slurm commands

- **sbatch FirstSlurmScript.sbatch -** launch a batch job
- **squeue** - shows all jobs in queue
  - **squeue --me** - shows only your jobs
  - **squeue -u <uNID>** - shows only your jobs
  - **mysqueue* -** shows job queue per partition and associated accounts you have access to on the cluster
- **scancel <jobid>** - cancel a job
- **sinfo** - shows all partitions/nodes state
  - **mysinfo* -** info on partitions/nodes and associated accounts you have access to on the cluster

*CHPC developed programs. See CHPC Newsletter 2023 Summer

Basic Slurm commands

- **sbatch FirstSlurmScript.sbatch -** launch a batch job
- **squeue** - shows a
  - **squeue --me** - sh
  - **squeue -u <uNID**
  - **mysqueue*** **-** sho ts you have access t
- **scancel <jobid>**
- **sinfo** - shows all p
  - **mysinfo*** **-** info or ve access to on the c
- **salloc** – start an i
  - Works with same

note – all of these commands only work on the cluster you are *currently* logged into.

To recognize a different cluster, use these flags:

-M all
-M kingspeak
--clusters all
--clusters kingspeak

*CHPC developed programs. See CHPC Newsletter 2023 Summer

Running interactive batch jobs

- An interactive command is launched through the **`salloc`** command

Running interactive batch jobs

- An interactive command is launched through the **`salloc`** command

```
salloc --time=8:00:00 --ntasks=4 --nodes=1 --mem=16G
--account=<account>  --partition=kingspeak-shared
```

Running interactive batch jobs

- An interactive command is launched through the **`salloc`** command

```
salloc --time=8:00:00 --ntasks=4 --nodes=1 --mem=16G
--account=<account>  --partition=kingspeak-shared
```

- Use of FastX connection is highly recommended
  - support GUI applications
  - keep your sessions alive

Running interactive batch jobs

- An interactive command is launched through the **`salloc`** command

```
salloc --time=8:00:00 --ntasks=4 --nodes=1 --mem=16G
--account=<account>  --partition=kingspeak-shared
```

- Use of FastX connection is highly recommended
  - support GUI applications
  - keep your sessions alive

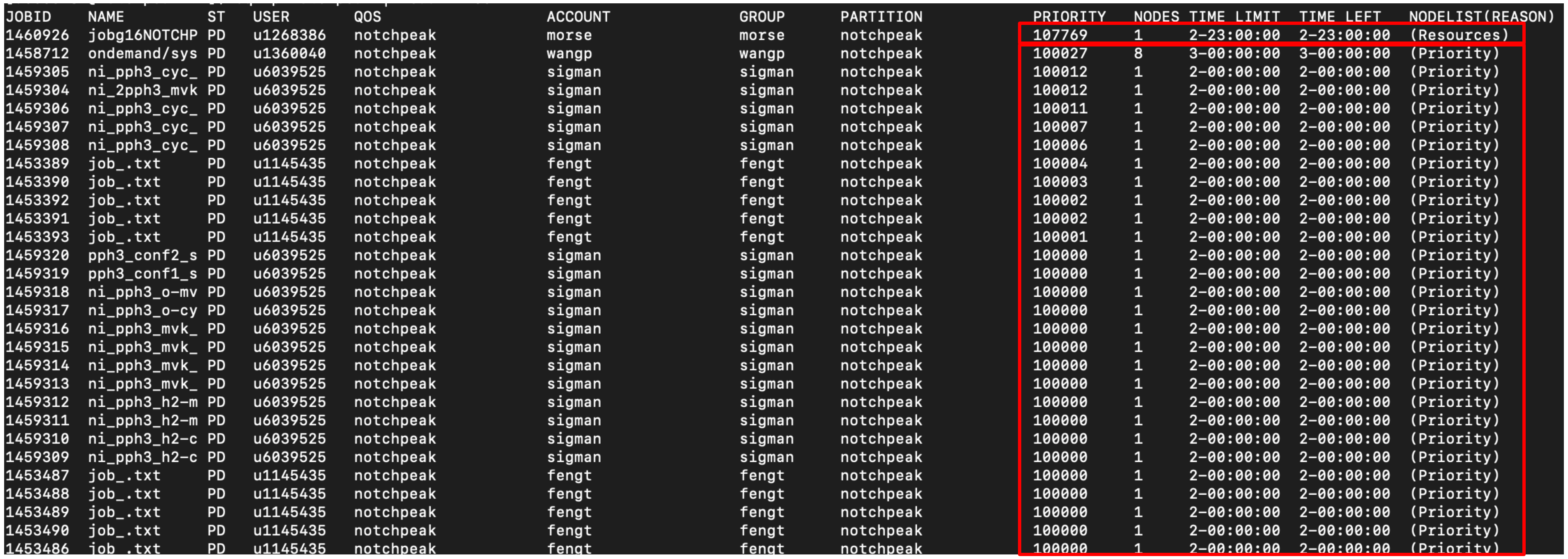
Open OnDemand is another option to start interactive sessions

Overview of Talk

- What is Slurm, and why use it?
- Preparing a Slurm job
  - Accounts and Partitions
  - CHPC Storage Resources
  - Slurm Environment Variables
- Slurm batch directives
- Basic Slurm Commands
- Running an Interactive Batch job
- Using GPU Nodes
- Job Priority & Performance

Job Priority

- Slurm assigns each job a priority score
- Priority score = how fast your job will start



```
JOBID    NAME         ST  USER      QOS                ACCOUNT            GROUP      PARTITION           PRIORITY   NODES TIME_LIMIT  TIME_LEFT   NODELIST(REASON)
1460926  jobg16NOTCHP PD  u1268386  notchpeak          morse              morse      notchpeak           107769     1     2-23:00:00  2-23:00:00  (Resources)
1458712  ondemand/sys PD  u1360040  notchpeak          wangp              wangp      notchpeak           100027     8     3-00:00:00  3-00:00:00  (Priority)
1459305  ni_pph3_cyc_ PD  u6039525  notchpeak          sigman             sigman     notchpeak           100012     1     2-00:00:00  2-00:00:00  (Priority)
1459304  ni_2pph3_mvk PD  u6039525  notchpeak          sigman             sigman     notchpeak           100012     1     2-00:00:00  2-00:00:00  (Priority)
1459306  ni_pph3_cyc_ PD  u6039525  notchpeak          sigman             sigman     notchpeak           100011     1     2-00:00:00  2-00:00:00  (Priority)
1459307  ni_pph3_cyc_ PD  u6039525  notchpeak          sigman             sigman     notchpeak           100007     1     2-00:00:00  2-00:00:00  (Priority)
1459308  ni_pph3_cyc_ PD  u6039525  notchpeak          sigman             sigman     notchpeak           100006     1     2-00:00:00  2-00:00:00  (Priority)
1453389  job_.txt     PD  u1145435  notchpeak          fengt              fengt      notchpeak           100004     1     2-00:00:00  2-00:00:00  (Priority)
1453390  job_.txt     PD  u1145435  notchpeak          fengt              fengt      notchpeak           100003     1     2-00:00:00  2-00:00:00  (Priority)
1453392  job_.txt     PD  u1145435  notchpeak          fengt              fengt      notchpeak           100002     1     2-00:00:00  2-00:00:00  (Priority)
1453391  job_.txt     PD  u1145435  notchpeak          fengt              fengt      notchpeak           100002     1     2-00:00:00  2-00:00:00  (Priority)
1453393  job_.txt     PD  u1145435  notchpeak          fengt              fengt      notchpeak           100001     1     2-00:00:00  2-00:00:00  (Priority)
1459320  pph3_conf2_s PD  u6039525  notchpeak          sigman             sigman     notchpeak           100000     1     2-00:00:00  2-00:00:00  (Priority)
1459319  pph3_conf1_s PD  u6039525  notchpeak          sigman             sigman     notchpeak           100000     1     2-00:00:00  2-00:00:00  (Priority)
1459318  ni_pph3_o-mv PD  u6039525  notchpeak          sigman             sigman     notchpeak           100000     1     2-00:00:00  2-00:00:00  (Priority)
1459317  ni_pph3_o-cy PD  u6039525  notchpeak          sigman             sigman     notchpeak           100000     1     2-00:00:00  2-00:00:00  (Priority)
1459316  ni_pph3_mvk_ PD  u6039525  notchpeak          sigman             sigman     notchpeak           100000     1     2-00:00:00  2-00:00:00  (Priority)
1459315  ni_pph3_mvk_ PD  u6039525  notchpeak          sigman             sigman     notchpeak           100000     1     2-00:00:00  2-00:00:00  (Priority)
1459314  ni_pph3_mvk_ PD  u6039525  notchpeak          sigman             sigman     notchpeak           100000     1     2-00:00:00  2-00:00:00  (Priority)
1459313  ni_pph3_mvk_ PD  u6039525  notchpeak          sigman             sigman     notchpeak           100000     1     2-00:00:00  2-00:00:00  (Priority)
1459312  ni_pph3_h2-m PD  u6039525  notchpeak          sigman             sigman     notchpeak           100000     1     2-00:00:00  2-00:00:00  (Priority)
1459311  ni_pph3_h2-m PD  u6039525  notchpeak          sigman             sigman     notchpeak           100000     1     2-00:00:00  2-00:00:00  (Priority)
1459310  ni_pph3_h2-c PD  u6039525  notchpeak          sigman             sigman     notchpeak           100000     1     2-00:00:00  2-00:00:00  (Priority)
1459309  ni_pph3_h2-c PD  u6039525  notchpeak          sigman             sigman     notchpeak           100000     1     2-00:00:00  2-00:00:00  (Priority)
1453487  job_.txt     PD  u1145435  notchpeak          fengt              fengt      notchpeak           100000     1     2-00:00:00  2-00:00:00  (Priority)
1453488  job_.txt     PD  u1145435  notchpeak          fengt              fengt      notchpeak           100000     1     2-00:00:00  2-00:00:00  (Priority)
1453489  job_.txt     PD  u1145435  notchpeak          fengt              fengt      notchpeak           100000     1     2-00:00:00  2-00:00:00  (Priority)
1453490  job_.txt     PD  u1145435  notchpeak          fengt              fengt      notchpeak           100000     1     2-00:00:00  2-00:00:00  (Priority)
1453486  job_.txt     PD  u1145435  notchpeak          fengt              fengt      notchpeak           100000     1     2-00:00:00  2-00:00:00  (Priority)
```

Job Priority

- Combination of four factors add to **base priority (QOS)**
  - Time in queue (most important)
  - Fairshare
  - Job size
  - # jobs in last 2 weeks
- Only **5 jobs per user** per slurm account (qos) will accrue priority
- `sprio` gives job priority for all jobs
  - `sprio -j <JOBID>` for a given job
  - `sprio -u <UNID>` for user's jobs

https://www.chpc.utah.edu/documentation/software/slurm.php#priority

Slurm Documentation at CHPC

https://www.chpc.utah.edu/documentation/software/slurm.php

https://www.chpc.utah.edu/documentation/software/serial-jobs.php

https://www.chpc.utah.edu/documentation/software/node-sharing.php

https://www.chpc.utah.edu/usage/constraints/

https://www.chpc.utah.edu/documentation/guides/index.php#GenSlurm

Other good documentation sources

http://slurm.schedmd.com/documentation.html

http://slurm.schedmd.com/pdfs/summary.pdf

http://www.schedmd.com/slurmdocs/rosetta.pdf

Getting Help

- CHPC website documentation
  - www.chpc.utah.edu
    - Getting started guide, cluster usage guides, software manual pages, CHPC policies
- Email: helpdesk@chpc.utah.edu
- Help Desk: 405 INSCC
- We use chpc-hpc-users@lists.utah.edu for sending messages to users