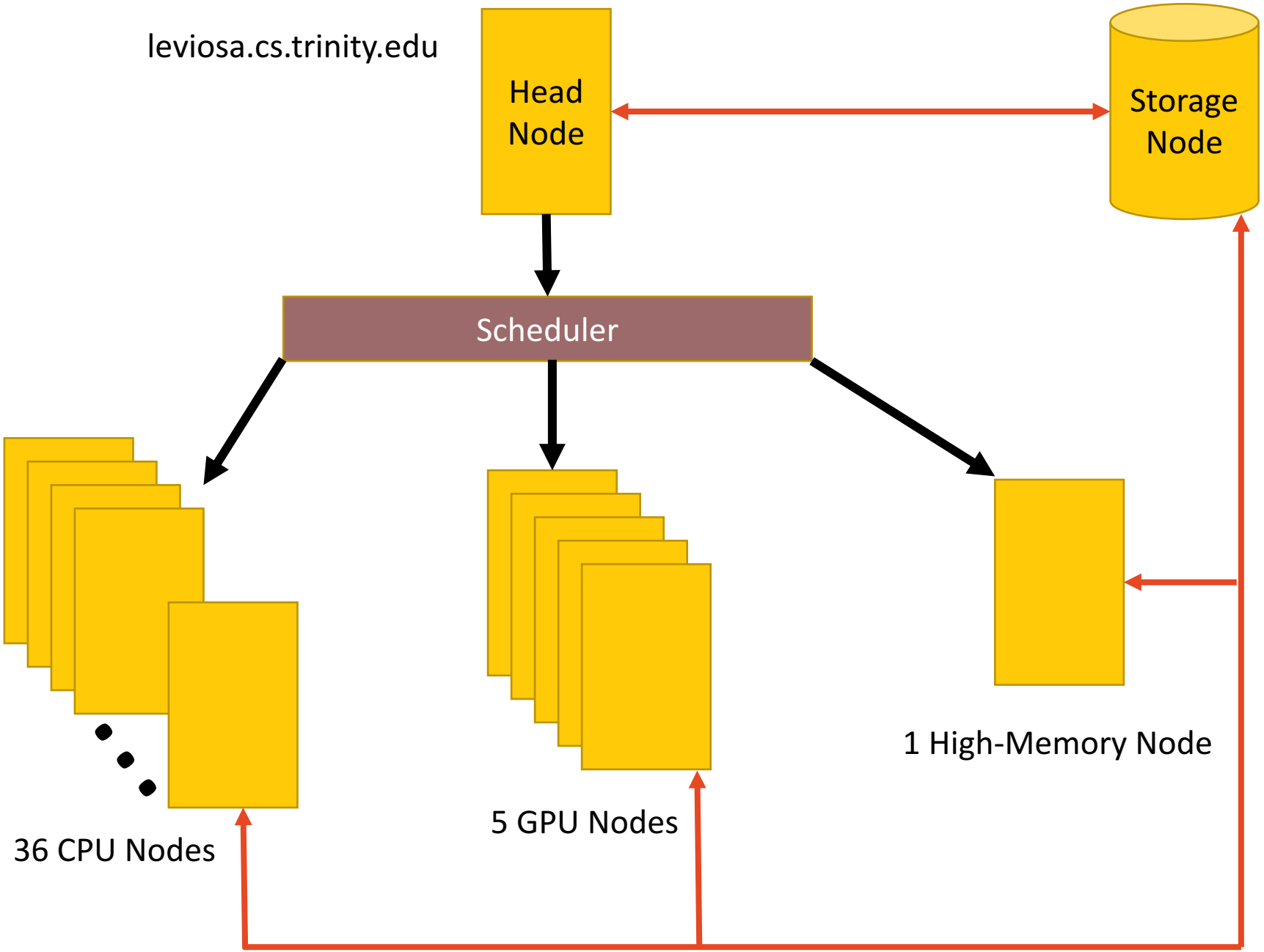
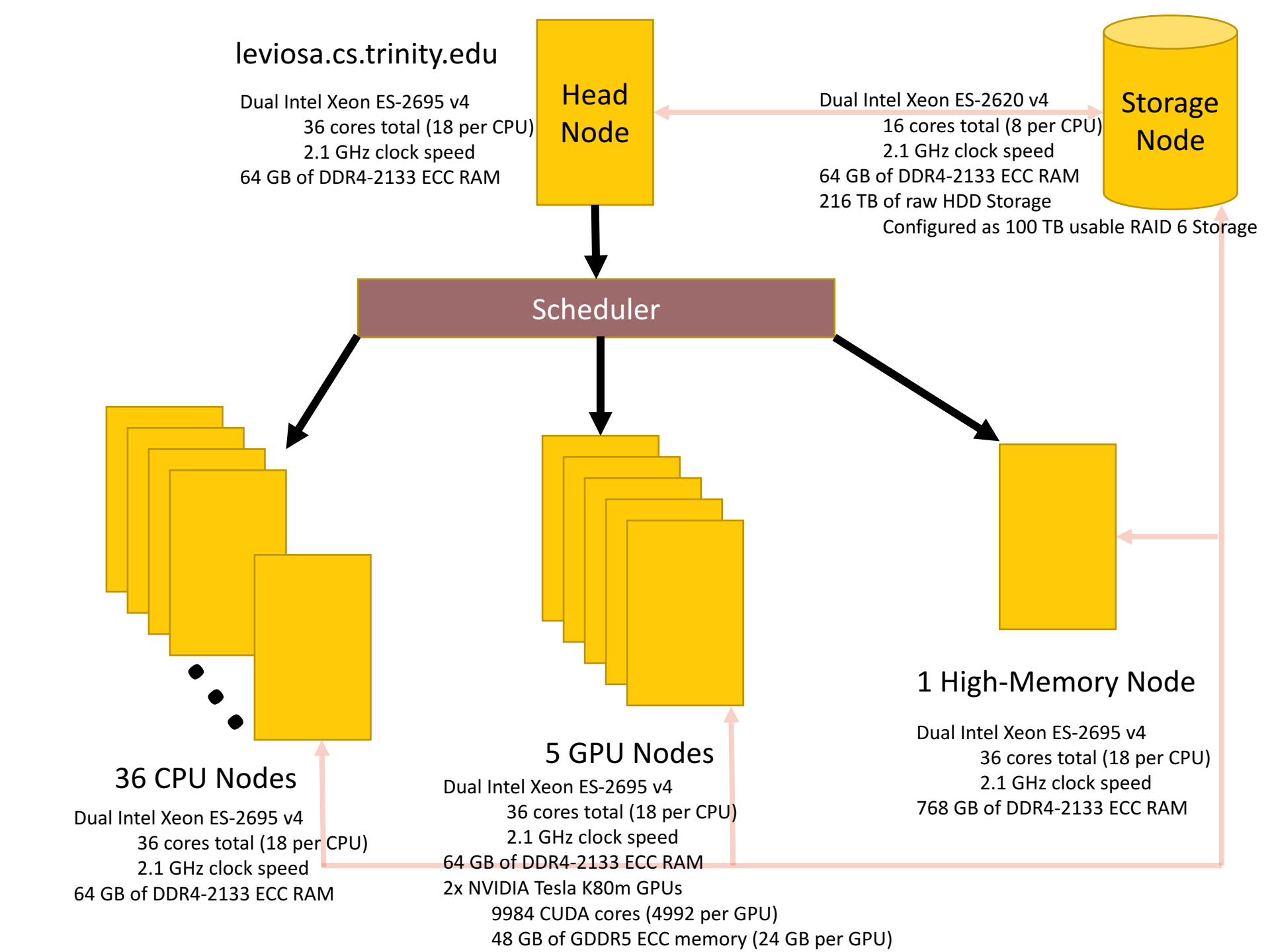
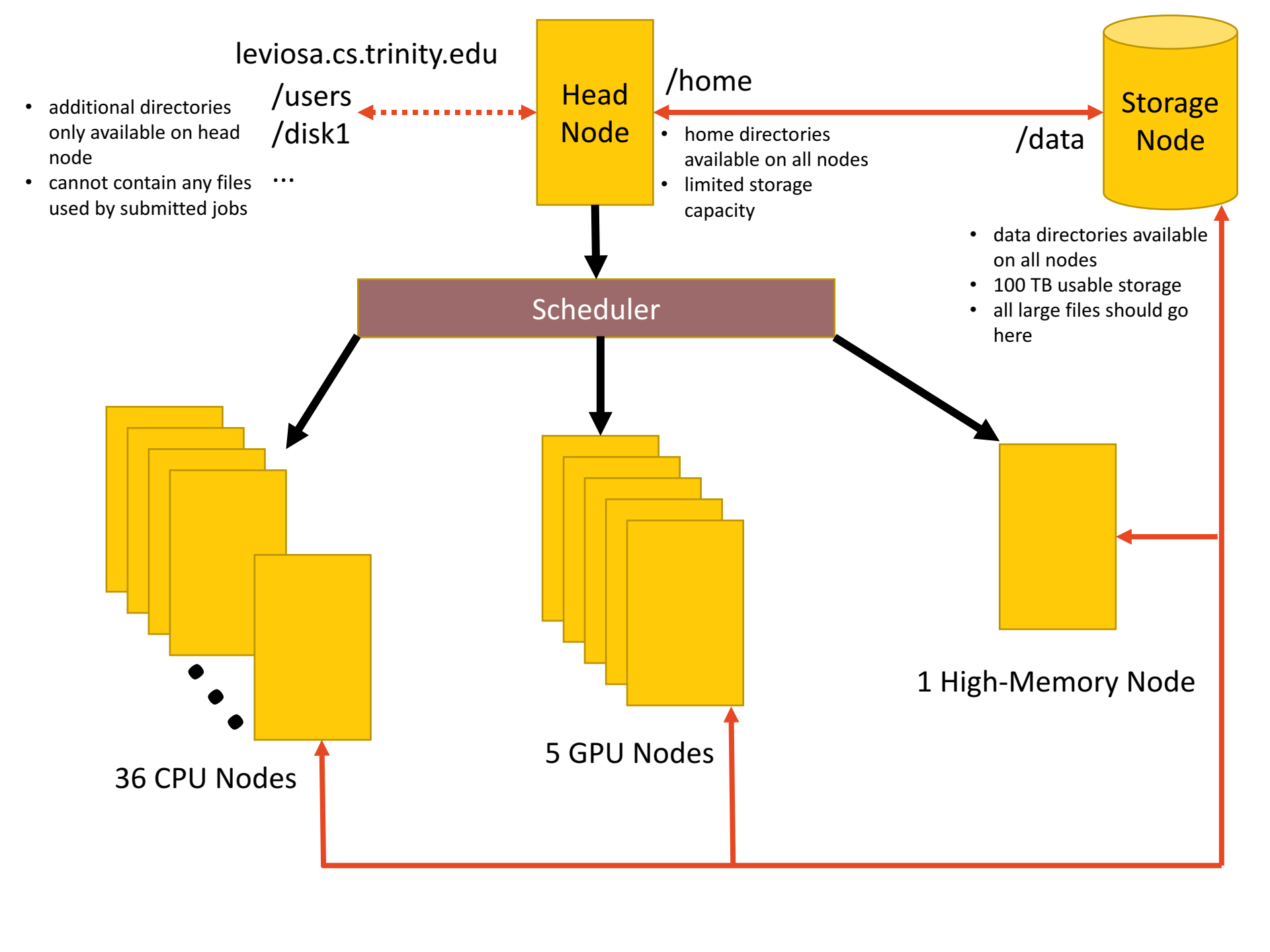
Trinity HPC Cluster Usage Overview

https://github.com/mhibbs-trinity/TU-HPC-Guide


leviosa.cs.trinity.edu
Head Node
Storage Node
Scheduler
36 CPU Nodes
5 GPU Nodes
1 High-Memory Node


leviosa.cs.trinity.edu
Dual Intel Xeon ES-2695 v4
36 cores total (18 per CPU)
2.1 GHz clock speed
64 GB of DDR4-2133 ECC RAM
Head Node
Dual Intel Xeon ES-2620 v4
16 cores total (8 per CPU)
2.1 GHz clock speed
64 GB of DDR4-2133 ECC RAM
216 TB of raw HDD Storage
Configured as 100 TB usable RAID 6 Storage
Storage Node
Scheduler
36 CPU Nodes
Dual Intel Xeon ES-2695 v4
36 cores total (18 per CPU)
2.1 GHz clock speed
64 GB of DDR4-2133 ECC RAM
5 GPU Nodes
Dual Intel Xeon ES-2695 v4
36 cores total (18 per CPU)
2.1 GHz clock speed
64 GB of DDR4-2133 ECC RAM
2x NVIDIA Tesla K80m GPUs
9984 CUDA cores (4992 per GPU)
48 GB of GDDR5 ECC memory (24 GB per GPU)
1 High-Memory Node
Dual Intel Xeon ES-2695 v4
36 cores total (18 per CPU)
2.1 GHz clock speed
768 GB of DDR4-2133 ECC RAM



- additional directories only available on head node
- cannot contain any files used by submitted jobs

- home directories available on all nodes
- limited storage capacity

- data directories available on all nodes
- 100 TB usable storage
- all large files should go here

OS & Software Information

- CentOS 7.3
  - derivative of RedHat Enterprise Linux
- TORQUE scheduling manager
- Core Linux software (e.g. cpp, vim, emacs, etc)
- Contact an administrator for help installing software
  - Neal Pape (npape@trinity.edu)
  - Matt Hibbs (mhibbs@trinity.edu)

Logging On

- Accounts are separate from other Linux systems
  - Send account requests to Neal Pape (npape@trinity.edu)

```
varys:~ mhibbs$ ssh mhibbs@leviosa.cs.trinity.edu
mhibbs@leviosa.cs.trinity.edu's password:
Last login: Wed May 24 21:58:34 2017 from lumos.cs.trinity.edu
[mhibbs@leviosa ~]$
```

Preparing to Submit Jobs

- The head node should only be used for small tasks
  - Compiling software
  - Viewing results
  - Copying files
  - etc
- All large jobs/tasks must be submitted to the scheduler in order for them to run on remote nodes
  - Prepare a shell script file
    - Include appropriate PBS directives
  - Submit job with qsub

Preparing Script File

- Bash shell script files
  - Start with `#!/bin/sh`
- PBS directives at the beginning of file
  - Comments that tell the scheduler how many resources to request for a job
- Linux commands to execute your program of interest

Example Script File

```
#!/bin/sh

#PBS -N keras_cifar10_example
#PBS -l nodes=1:ppn=36
#PBS -l walltime=72:00:00
#PBS -M mhibbs@trinity.edu
#PBS -m abe

python /data/hibbslab/projects/keras/keras/examples/cifar10_cnn.py
```

Submitting Jobs & Job Status

```
[mhibbs@leviosa ~]$ qsub run_keras_cifar10.sh
66.leviosa.cs.trinity.edu
```

```
[mhibbs@leviosa ~]$ qstat
Job ID                    Name             User            Time Use S Queue
------------------------- ---------------- --------------- -------- - -----
66.leviosa                 ...far10_example mhibbs                  0 R batch
```

Removing/Stopping Jobs

```
[mhibbs@leviosa ~]$ qstat
Job ID                    Name             User            Time Use S Queue
------------------------- ---------------- --------------- -------- - -----
66.leviosa                 ...far10_example mhibbs          00:06:51 R batch
[mhibbs@leviosa ~]$
[mhibbs@leviosa ~]$
[mhibbs@leviosa ~]$
[mhibbs@leviosa ~]$
[mhibbs@leviosa ~]$
[mhibbs@leviosa ~]$ qdel 66.leviosa
[mhibbs@leviosa ~]$ qstat
Job ID                    Name             User            Time Use S Queue
------------------------- ---------------- --------------- -------- - -----
66.leviosa                 ...far10_example mhibbs          00:24:00 C batch
[mhibbs@leviosa ~]$
[mhibbs@leviosa ~]$
```