

# Introduction To HPCC




## Faculty Seminars in Research and Instructional Technology

### May 8, 2012

Dirk Colbry  
colbrydi@msu.edu  
Research Specialist  
Institute for Cyber-Enabled Research

MICHIGAN STATE UNIVERSITY

© 2012 Michigan State University Board of Trustees.




# Agenda

- Introduction to iCER / HPCC
- Connecting to HPC
- Interactive Computing
- Using the Batch Scheduler

---

- Doing more faster
  - Pleasantly Parallel
  - Shared Memory Parallelization
  - Shared Network Parallelization

MICHIGAN STATE UNIVERSITY



## Science and Computation

- Pillars of Science:
  - Theory
  - Experimentation
  - Simulation
  - (Big) Data



MICHIGAN STATE  
UNIVERSITY



## What is Advanced Computing Hardware?

- Anything more advanced than your desktop
- Local resources
  - Lab, Department, Institution (HPCC)
- National resources
  - NSF (XSEDE, Blue Waters), DOE (Jaguar) , Others
- Commercial Resources (cloud computing)
  - Amazon, Azure, Liquid Web, Others

MICHIGAN STATE  
UNIVERSITY



## Why use Advanced Computing Hardware?

- Science takes too long
- Computation runs out of memory
- Need licensed software
- Need advanced interface (visualization)

## Types of problems

- CPU bound
  - Lots of computing (think simulation)
- Memory bound
  - Requires lots of memory (think genomics)
- I/O bound
  - Requires lots of data (think astronomy)

(many problems fall in more than one category)

## Types of Systems

### Local Resources

- Special Use Systems
  - SMP – Symmetric Multiprocessor (fat node)
  - GPGPU (General Purpose Graphics Processing Unit)
- Typical HPC cluster
  - Commodity computers
  - High speed backbone
  - High speed network storage

### National and Commercial

- Advanced HPC
  - Specialty hardware
  - High speed backbone
  - High speed storage
- Grid
  - Many HPC systems linked together by high speed network
- Cloud
  - Lots of definitions
  - Typically refers to computing as a service using highly flexible virtual machines

## 1957 MISTIC Mainframe

- MSU's first mainframe
- Hand built by grad students
  - Dick Reid
  - Glen Keeney

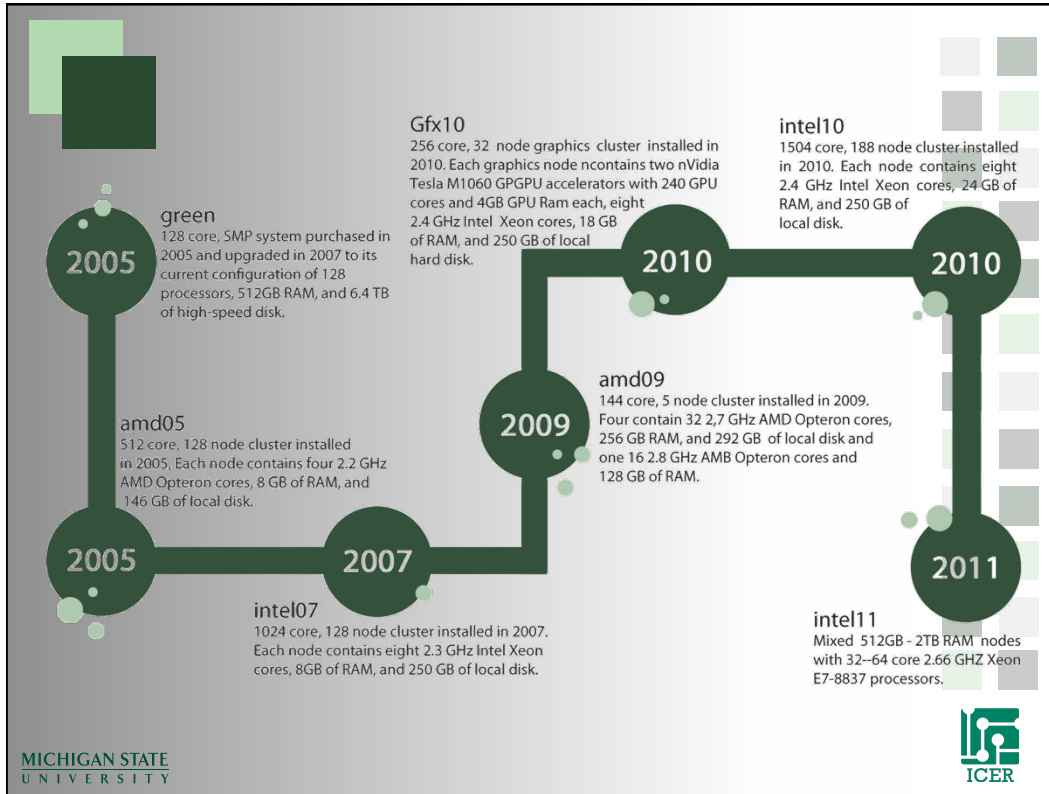


## After MISTIC

- 1957 MISTIC
- 1963-1973 CDC 3600
- 1967 Computer Science Department
- 1968 CDC 6500
- 1971 MERIT
- 1978 Cyber 750
- **2004 HPCC**
- **2009 ICER**

## 2004 MSU HPCC

- Provide a level of performance beyond what you could get and reasonably maintain as a small group
- Provide a variety of technology, hardware and software, that would allow for innovation not easily found



## 2009 iCER

The Institute for Cyber Enabled Research (iCER) at Michigan State University (MSU) was established to coordinate and support multidisciplinary resource for computation and computational sciences. The Center's goal is to enhance MSU's national and international presence and competitive edge in disciplines and research thrusts that rely on advanced computing.

## CSTAT for Computational Sciences

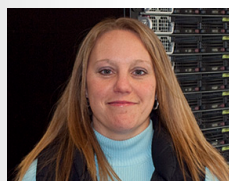
- CSTAT is a one-stop shop for researchers who have questions about using statistics in their research
- iCER is a one-stop shop for researchers who have questions about using computers in their research
- Both are under VPRGS and represent our F&A dollars at work

MICHIGAN STATE  
UNIVERSITY



## People

- iCER Director  
– Dr. Wolfgang Bauer
- HPCC Director  
– Dr. Bill Punch
- Administrative Assistant  
– Kelly Osborn



MICHIGAN STATE  
UNIVERSITY



## HPC Administrators

- Andrew Keen
- Jim Leikert
- Gregory Mason



MICHIGAN STATE  
UNIVERSITY



## HPC Programmers

- John Johnson
  - 50% Biology
  - 50% iCER
  - Primary liaison to Biology



MICHIGAN STATE  
UNIVERSITY





## iCER Research Specialist

Ben Ong and Me

- Research Consulting
- HPCC Programming
- Proposal Writing
- Training and Education
- Outreach



MICHIGAN STATE  
UNIVERSITY



## iCER Broad Goals

- Hardware (HPCC)
- Hardware Buy-in opportunities
- Education
- Consulting
- Collaboration
- Grant Support

MICHIGAN STATE  
UNIVERSITY



## Current iCER Programs

- HPCC Mid-Morning Breaks
- iCER Research Seminar
- Introduction to iCER department presentations
- Workshops
- Classroom presentations
- Study groups
  - Parallel programming
  - GPGPU
- One-on-one Consulting
- Fund matching

MICHIGAN STATE  
UNIVERSITY

[www.icer.msu.edu](http://www.icer.msu.edu)



## Bigger Science

- The goal of iCER is NOT:
  - Kflops / second
- Instead, the goal of iCER IS:
  - KSciences / second
- Doing More Science, Faster
  - Reducing the “Mean time to Science”
- iCER is designed to help researchers do their science and when appropriate scale them up to one of the national labs



MICHIGAN STATE  
UNIVERSITY





## We are here to help

- [www.hpcc.msu.edu/contact](http://www.hpcc.msu.edu/contact)
  - Questions
  - Schedule Consultations
  - Code Reviews
  - Programming help
  - Hardware Purchasing
  - Help with Grants
  - Support for Grants

## Online Resources

- [icer.msu.edu](http://icer.msu.edu) - iCER Home
  - [hpcc.msu.edu](http://hpcc.msu.edu) – HPCC Home
- [wiki.hpcc.msu.edu](http://wiki.hpcc.msu.edu) – HPCC User Wiki






# Connecting to the HPCC (The Basics)

<http://www.softwarecarpentry.org/>

MICHIGAN STATE UNIVERSITY


© 2010 Michigan State University Board of Trustees.



# HPCC account & NetID

- All HPCC resources use your MSU NetID username and password
- You need an account to log into HPCC systems
  - <https://www.hpcc.msu.edu/request>
  - Your P.I. needs to fill out the account request form

MICHIGAN STATE UNIVERSITY

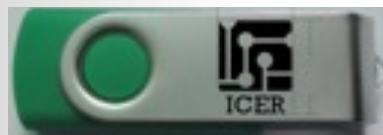


## URLs for Required Software

- PuTTY:
  - <http://www.chiark.greenend.org.uk/~sgtatham/putty/>
- Xming:
  - <http://www.straightrunning.com/XmingNotes/>
- Xming install:
  - <https://wiki.hpcc.msu.edu/x/swAk>
- WinSCP:
  - <http://winscp.net>

## Exercise: Portable HPCC

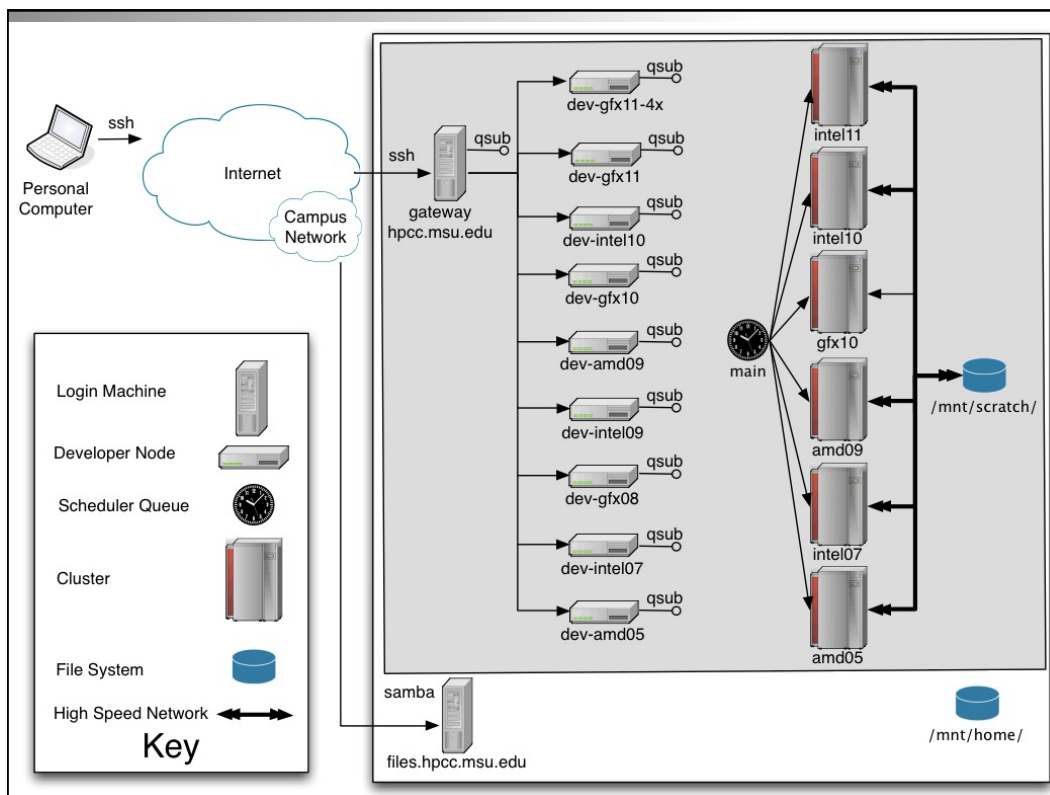
- Plug in your usb-thumb drive
- Open the thumb drive folder and select
  - PortableApps
- You should see a new menu in your system tray for navigating
- Start the pchat program



# HPCC Cluster Overview

- Linux operating system
- Primary interface is text based though Secure Shell (ssh)
- All machines in the main cluster are binary compatible (compile once run anywhere)
- A scheduler is used to manage jobs running on the cluster
- A submission script is used to tell the scheduler the resources required and how to run a job
- A module system is used to manage the loading and unloading of software configurations

MICHIGAN STATE  
UNIVERSITY



## Exercise: Connect to HPCC

- Use Xming on your thumb drive to log into gateway.hpcc.msu.edu
- ssh into a dev node (developer node)

```
>ssh dev-intel109
```

## Command Line Interface

- Command Line Interface (CLI)
- Shell
  - Program to run Programs
- Bash (Bourne Again Shell)
- Use it because:
  - many tools only have command-line interfaces
  - allows you to combine tools in powerful new ways

## Shell Navigation

- Basic Navigation commands:

<code>pwd</code>	print working directory
<code>cd</code>	change working directory
<code>ls</code>	list directory

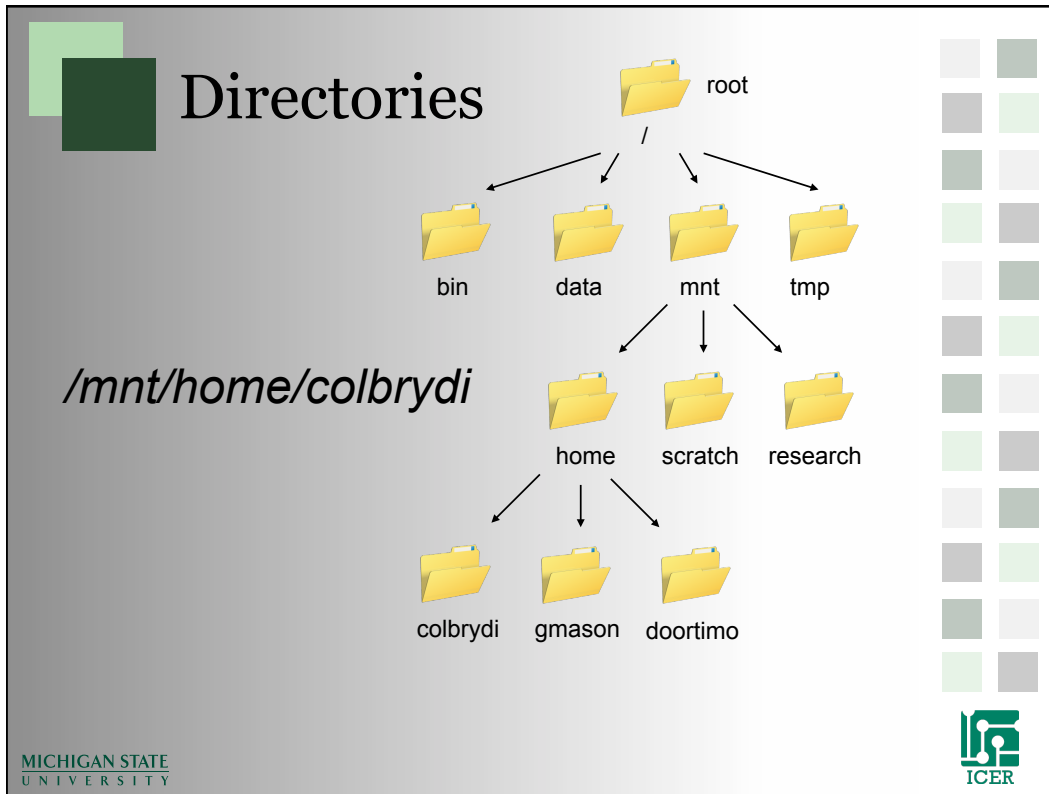
- Use the following symbols to indicate special directories:

<code>.</code>	current directory
<code>..</code>	parent directory
<code>~</code>	home directory
<code>-</code>	previous directory

## Exercise – Shell Navigation

- Show the path to the current directory  
> `pwd`
- Change to the scratch directory  
> `cd /mnt/scratch/`
- List the contents of the current directory:  
> `ls`
- Change back to home  
> `cd ~`





## Accounts

- Each user has 50Gigs of personal hard drive space
  - /mnt/home/username/
- Users have access to 33TB of scratch space
  - /mnt/scratch/username/
- Research group space is also available upon request
  - /mnt/research/groupname/

MICHIGAN STATE UNIVERSITY

ICER

## Example: File Manipulation

- Try Commands

mkdir	make directory
cp	copy file
cat	display contents of text file
rm	remove file

- See the contents of your “.bashrc” file
  - > **cat .bashrc**
- Make a directory called “hpcworkshop”, change to that directory and list the contents.

```
> mkdir hpcworkshop
> cd ./hpcworkshop
```

## Module System

- To maximize the different types of software and system configurations that are available to the users, HPCC uses a Module system
- Key Commands
  - **module avail** – show available modules
  - **module list** – list currently loaded modules
  - **module load** modulename – load a module
  - **module unload** modulename – unload a module

## Exercise – Module

- List loaded modules  
`>module list`
- Show available modules:  
`>module avail`
- Try an example (Shouldn't work):  
`>powertools`

## Exercise: getexample

- Load a newly available module:  
`>module load powertools`
- Show powertools (should work now):  
`>powertools`
- Run the “getexample” powertool  
`>getexample`
- Download the helloMPI example  
`>getexample helloworld`

## Standard in/out/err and piping

- You can redirect the output of a program to a file using “>” greater than character:
  - myprogram > output.txt
- You can also cause the output of the program to be the input of another program using the “|” pipe character:
  - myprogram | myotherprogram

## Exercise: redirection and piping

- Change to the helloworld directory:
 

```
> cd ~/hpccworkshop/helloworld
```

```
> ls -la
```
- Redirect the output of the ls command:
 

```
> ls -la > numOfLines
```

```
> Cat numOfLines
```
- Pipe Commands together
 

```
> wc -l * | sort -n
```

## SCP/SFTP – Secure File transfer

- WinSCP for Windows
- Command-line “scp” and “sftp” on Linux
- Many other scp and sftp clients out there as well
- Functions over SSHv2 protocol, very secure

## Exercise: Transfer a file

- Make a file called **minlines** using notepad++ on your thumb drive
- Put in the following lines:  

```
wc -l * | sort -n | head -1
```
- Open WinSCP on your thumb drive
- Copy the file **minlines** to the helloworld directory

# Home Directory Mapping

- Easy way to work with files at the HPCC
- No client to download, uses built-in functionality in Windows
- Not just for Windows.
  - Linux and Mac users can mount their home directory similar to NFS or AppleShare
- <http://wiki.hpcc.msu.edu/>

# File Permissions



	user	group	all
read	✓	✓	✗
write	✓	✗	✗
execute	✗	✗	✗



# Permissions

- Common Commands

chmod	Change permissions (change mode)
ls -a -l	List all long (including permissions)

# Example: permissions

- Change to the helloMPI directory  
`> cd ~/hpcworkshop/helloworld`
- show current permissions  
`> ls -la`
- change make the **minlines** file executable  
`> chmod u+x minlines`
- Check permissions again  
`> ls -la`
- Now you can run maxlines as a command  
`> ./minlines`



## Environment Variables

- Scripts also let you use environment variables
- These variables can be used by your script or program
- Use “export” and = to set a variable
- Use the \$ and {} to display the contents of a variable

## Example: Environment Variables




- Display all environment variables  
`>env`
- Display specific environment variable  
`>echo ${MACHTYPE}`
- Make a new variable  
`> export MYVAR="Hello World"`
- Use your variable  
`>echo ${MYVAR}`





# Interactive Computing


MICHIGAN STATE UNIVERSITY



# Running Jobs on the HPC

- Submission scripts are used to run jobs on the cluster
- The developer (dev) nodes are used to compile, test and debug programs
- However, the developer nodes are powerful systems too. **We don't want to waste their compute power.**

MICHIGAN STATE UNIVERSITY

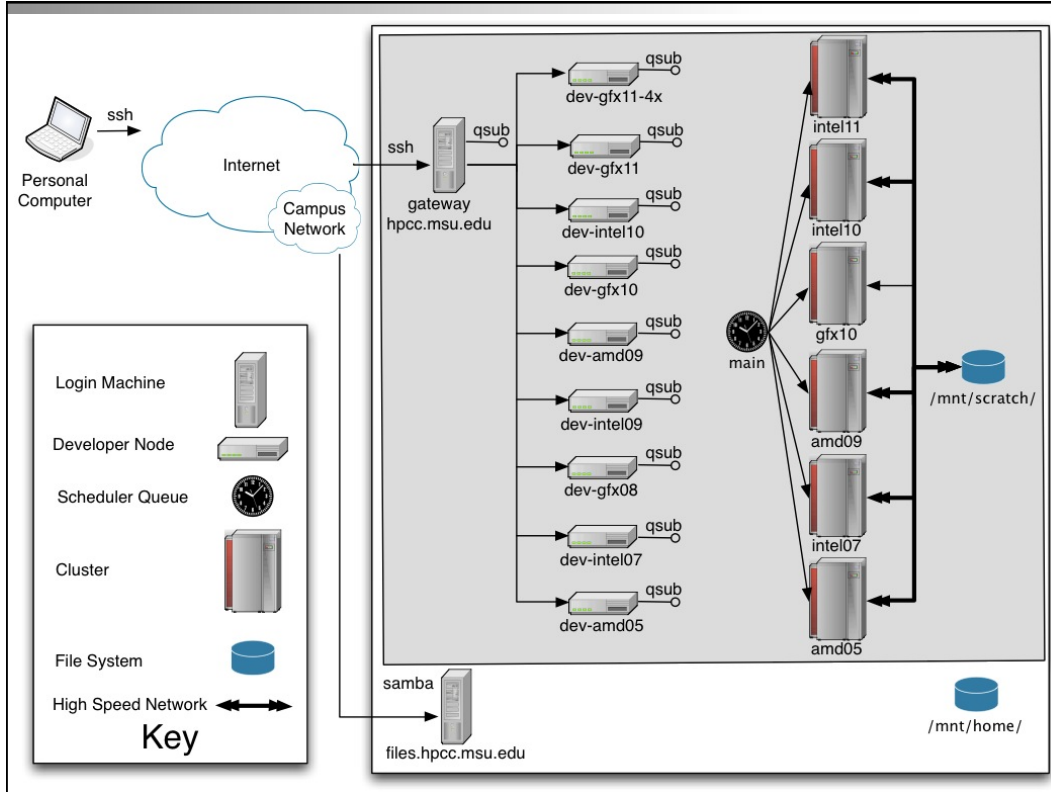


## Advantages of running Interactively

- You do not need to write a submission script
- You do not need to wait in the queue
- You can provide input to and get feedback from your programs as they are running

## Disadvantages of running Interactively

- All the resources on developer nodes are shared between all users.
- Any single process is limited to 2 hours of cpu time. If a process runs longer than 2 hours it will be killed.
- Programs that over utilize the resources on a developer node (preventing other to use the system) can be killed without warning.



## Developer Nodes

Name	Cores	Memory	GPUs	Notes
dev-amd05	4	8GB	-	
dev-intel07	8	8GB	-	
dev-gfx08	4	8GB	3	Graphics Node
dev-amd09	32	256GB	-	Fat Node
dev-intel09	8	32GB	-	Fastest Node
dev-gfx10	8	18GB	2	Graphics Node
dev-intel10	8	24GB	-	
dev-gfx11	4	8GB	2	Graphics Node
dev-gfx11-4x	8	18GB	4	Graphics Node

## Compiling

- Most users use the developer nodes for developing their software.
- If you are using a makefile you can compile using more processors with the `-j` option.
  - `make -j32`
  - Will make with 32 core threads
    - (use this on dev-amd09)

## Compilers

- By default we use the gnu compilers. However, lots of other compilers are available including Intel and Portland compilers.
- The module system always sets environment variables such that you can easily test with other compilers.
  - `{CC}`
  - `{FC}`
  - Etc.

## Exercise: Compile Code



- Make sure you are in the helloworld directory:
  - > `pwd`
- Run the intel compilers:
  - > `${CC} -O3 -o helloworld helloworld.cpp`
- Run the program:
  - > `./helloworld`

## Available Software

- Center Supported Development Software
  - Intel compilers, openmp, openmpi, mvapich, totalview, mkl, pathscale, gnu...
- Center Supported Research Software
  - Matlab, R, fluent, abaqus, HEEDS, amber, blast, ls-dyna, starp...
- Customer Software (module use.cus)
  - gromacs, cmake, cuda, imagemagick, java, openmm, siesta...
- For a more up to date list, see the documentation wiki:
  - <http://wiki.hpcc.msu.edu/>

## Running in the background

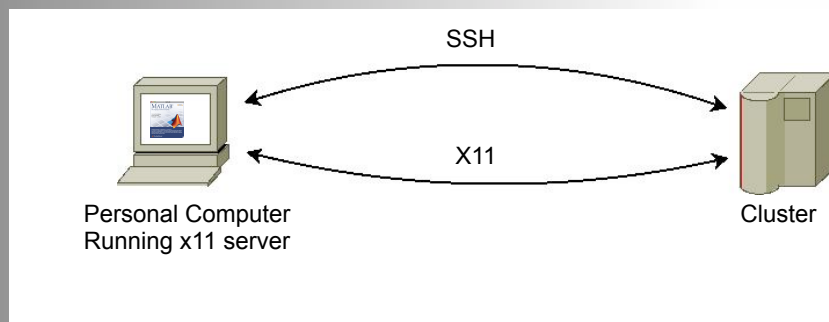
- You can run a program in the background by typing an “&” after the command.
- You can make a program keep running even after you log out of your ssh session by using “**nohup** command”
- You can run an entire session in the background even if you log in and out of your ssh session by using the “**screen**” command
- All three of these options are common to linux and tutorials can be found online

## CLI vs GUI

- CLI – Command Line Interface
- GUI – Graphical User Interface

## What is X11?

- Method for running Graphical User Interface (GUI) across a network connection.



## What is needed for X11

- X11 server running on your personal computer
- SSH connection with X11 enabled
- Fast Network connection
  - Preferably on campus

## Graphical User Interface

- X11 Windows: Install Xming
  - Installation instructions at:
   
<https://wiki.hpcc.msu.edu/x/swAk>
- `ssh -X username@hpc.msu.edu`
- Turn on x11 forwarding

## Exercise: Transfer a file

- Try one of the following Commands

xeyes	Test X11
matlab	Run Matlab

> **xeyes**

> **matlab &**

> **ps** <- Find the process ID ##### for matlab

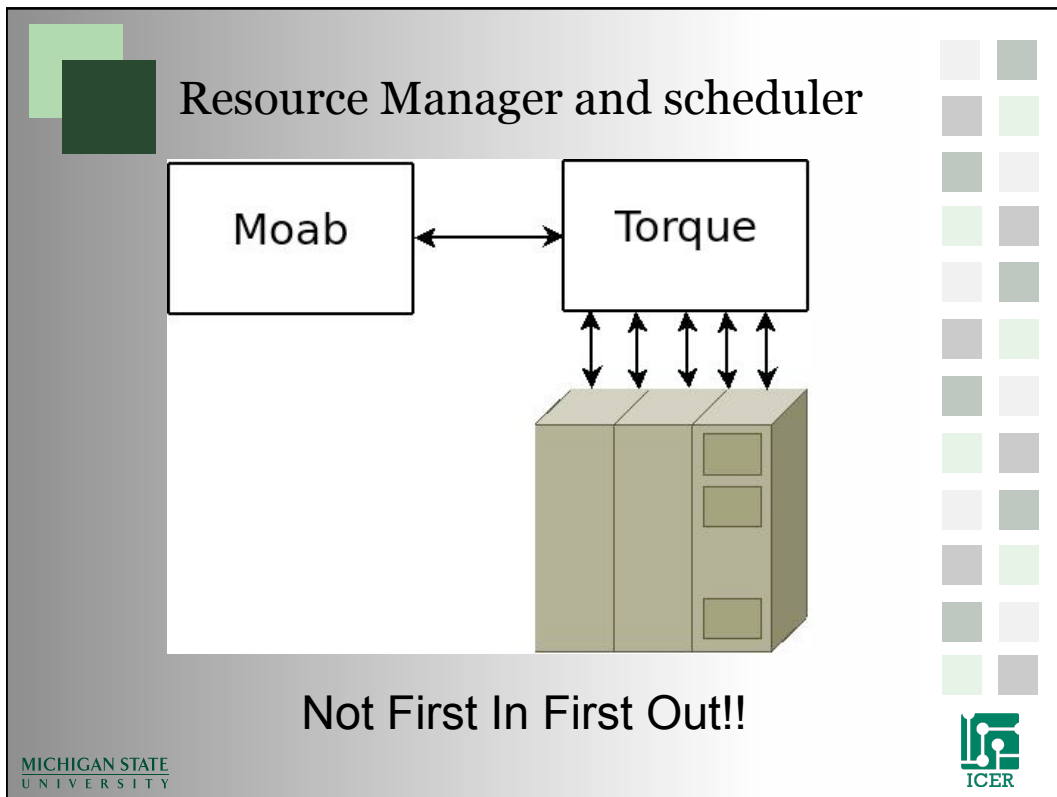
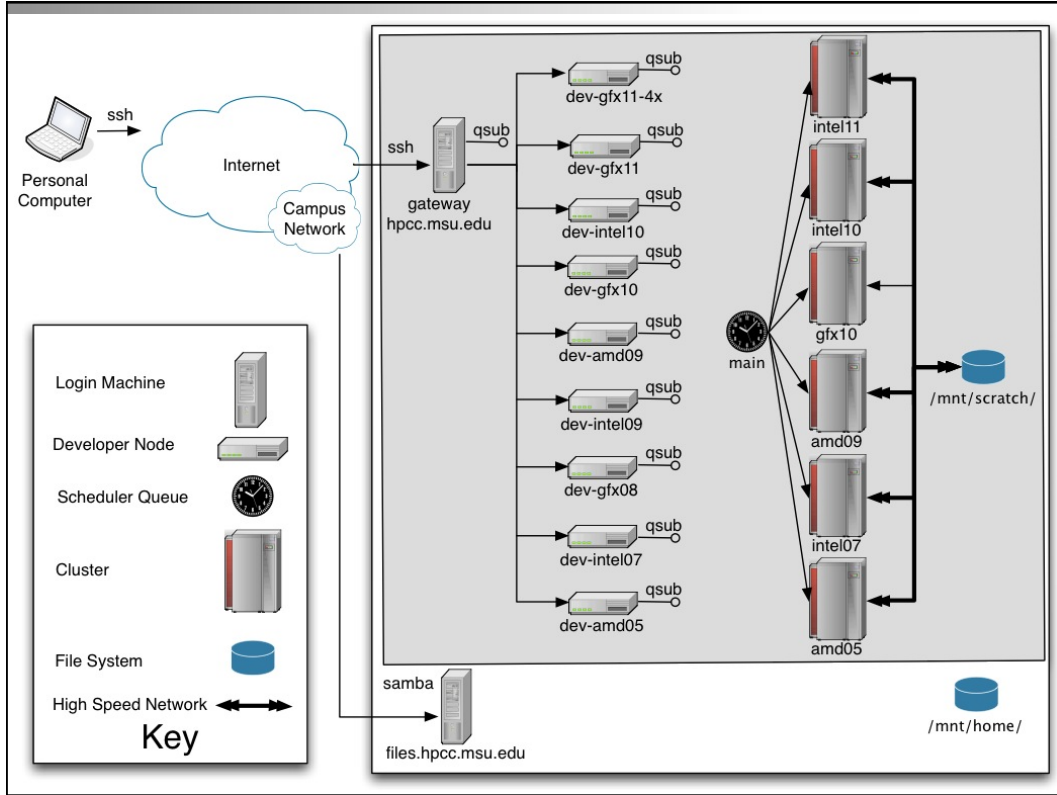
> **kill #####**



## Programs that can use X11

- matlab – Matrix Laboratory Software
- R - statistical computing and graphics
- firefox – Web browser
- totalview – C/C++/fortran debugger
- gedit, gvim, emacs – Text editors
- And others...

## Using The Batch Scheduler



## Schedulers vs Resource Managers

- Scheduler  
(Moab)
  - Tracks and assigns
    - Memory
    - CPUs
    - Disk space
    - Software Licenses
    - Power / environment
    - Network
- Resource Manager  
(PBS/Torque)
  - Hold jobs for execution
  - Put the jobs on the nodes
  - Monitor the jobs and nodes

## Common Commands

- **qsub** <Submission script>
  - Submit a job to the queue
- **qdel** <JOB ID>
  - Delete a job from the queue
- **showq** -u <USERNAME>
  - Show the current job queue
- **checkjob** <JOB ID>
  - Check the status of the current job
- **showstart** -e all <JOB ID>
  - Show the estimated start time of the job.

## Submission Script

- List of required resources
- All instructions needed to run the computation

## Typical Submission Script

```
#!/bin/bash -login
#PBS -l walltime=10:00:00,mem=3Gb,nodes=10:ppn=1
#PBS -j oe

cd ${PBS_O_WORKDIR}

./myprogram -my input arguments

qstat -f ${PBS_JOBID}
```

Shell Comment

Define Shell

Resource Requests

Shell Commands

Special Environment Variables

## Example: Submit a job

- Go to the top helloworld directory  
`>cd ~/hpccworkshop/helloworld`
- Create a simple submission script  
`>nano simple.qsub`
- See next slide for what to type...

## simple.qsub

```
#!/bin/bash -login
#PBS -l walltime=00:01:00
#PBS -l nodes=1:ppn=1,feature=gdb

cd ${PBS_O_WORKID}
./helloworld

qstat -f ${PBS_JOBID}
```

## Example: Submit a job, cont.

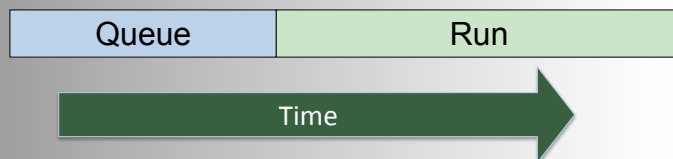
- Submit the file to the queue  

```
>qsub simple.qsub
```
- Record jobid number (#####) and wait at most 30 seconds
- Check the status of the queue  

```
>showq
```

## Submitting a job

- `qsub –arguments <Submission Script>`  
 – Returns the job ID. Typically looks like the following:
  - 5945571.cmgr01
- Time to job completion



## Example: Monitor a job

- Submit the file to the queue:

```
>qstat -f #####
```

- When will a job start:

```
>showstart -e all #####
```

## Scheduling Priorities

- Jobs that use more resources get higher priority (because these are hard to schedule)
- Smaller jobs are backfilled to fit in the holes created by the bigger jobs
- Eligible jobs acquire more priority as they sit in the queue
- Jobs can be in three basic states:
  - Blocked, eligible or running

## Cluster Resources

Year	Name	Description	ppn	Memory	Nodes	Total Cores
2005	amd05	Dual-core 2.2GHz AMD Opteron 275	4	8GB	96	384
2007	intel07	Quad-core 2.3GHz Intel Xeons E5345	8	8GB	124	992
2009	amd09	Sun Fire X4600 (Fat Node)	16	128GB	1	16
			32	256GB	4	128
2010	gfx10	Nvidia Cuda Node (no IB)	8	18GB	41	256
2010	intel10	Intel Xeon E5620 (2.40 GHz)	8	24GB	192	1536
2011	intel11	Intel Xeon 2.66 GHz E7-8837	32	512GB	1	32
			32	1TB	1	32
			64	2TB	2	128

MICHIGAN STATE  
UNIVERSITY



## System Limitations

- Scheduling
  - 5 eligible jobs at a time
  - 144 running jobs
  - 256 submitted jobs (increasing soon)
- Resources
  - 1 week of walltime
  - 144 cores (nodes \* ppn)
  - ppn=64
  - 1TB memory on a single core
  - ~200 GB Hard Drive

MICHIGAN STATE  
UNIVERSITY





## Job completion

- By default the job will automatically generate two files when it completes:
  - Standard Output:
    - Ex: jobname.o5945571
  - Standard Error:
    - Ex: jobname.e5945571
- You can combine these files if you add the join option in your submission script:
  - “#PBS -j oe”
- You can change the output file name
  - #PBS -o /mnt/home/netid/myoutputfile.txt

## Other Job Properties

- resources (-l)
  - Walltime, memory, nodes, processor, network, etc.
- #PBS -l feature=gpgpu,gbe
- #PBS -l nodes=2:ppn=8:gpu=2
- #PBS -l mem=16gb
- Email address (-M)
  - Ex: #PBS -M [colbrydi@msu.edu](mailto:colbrydi@msu.edu)
- Email Options (-m)
  - Ex: #PBS -m abe

Many others, see the wiki:  
<http://wiki.hpcc.msu.edu/>

## Requesting local disk

- Sometimes (not often) local disk is faster than scratch
- Users can use the following resource to request temporary local disk space:
  - #PBS -l file=10gb
- The directory to access this disk space is determined by the one time use environment variable
  - `${TMPDIR}`

## Advanced Environment Variables

- The scheduler adds a number of environment variables that you can use in your script:
  - `PBS_JOBID`
    - The job number for the current job.
  - `PBS_O_WORKDIR`
    - The original working directory which the job was submitted

Ex:

```
mkdir ${PBS_O_WORKDIR}/${PBS_JOBID}
```



# Getting Help

- Documentation and User Manual – [wiki.hpcc.msu.edu](http://wiki.hpcc.msu.edu)
- Contact HPCC and iCER Staff for:
  - Reporting System Problems
  - HPC Program writing/debugging Consultation
  - Help with HPC grant writing
  - System Requests
  - Other General Questions
- Primary form of contact - [www.hpcc.msu.edu/contact](http://www.hpcc.msu.edu/contact)
- HPCC Request tracking system – [rt.hpcc.msu.edu](http://rt.hpcc.msu.edu)
- HPCC Phone – (517) 353-9309
- HPCC Office – 1400 PBS
- Office Hours – Monday – Friday 9am-5pm