



# Illinois Campus Cluster: Basics of Access and Usage

Gautham Narayan and Jay Alameda  
(Based on material from Bruno Abreu)



**National Center for  
Supercomputing Applications**

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

September 20, 2023

# Fall 2023 CAPS Computing Boot Camp

September 21, 2023 - Introductory Topics

- 8:30-9:00 Breakfast
- 9:00-9:15 Campus Cluster Overview (Jay)
- 9:15-9:30 CAPS Partition Overview (Jay)
- 9:30-10:30 Campus Cluster Access and Usage Overview (Jay)
- 10:30-11:00 Break
- 11:00-12:30 SSH Keys setup and use, X11 (Gautham)
- 12:30-1:30 Lunch



# Fall 2023 CAPS Computing Boot Camp

September 22, 2023 - Advanced Topics

- 8:30-9:00 Breakfast
- 9:00-10:30 Parallelization (Cynthia)
- 10:30-11:00 Break
- 11:00-12:30 Astronomical Data Analysis with Jupyter Notebooks (Srini)
- 12:30-1:30 Lunch

Ask questions early and often - better to avoid confusion!



# Campus Cluster Overview



Illinois Campus Cluster Program

AT THE UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

## Overview

~ 20 min

- What is the ICC?
- Who is it for?
- Examples of research applications
- Why should I use it?

- How can I access it?
- How can I run my codes and applications?
- What else is available at ICC?
- Support material and further learning

## Access and Usage

~ 55 min

- Q&A session (~ 15 min)



NCSA | NATIONAL CENTER FOR SUPERCOMPUTING APPLICATIONS

# What is the ICCP?

<https://campuscluster.illinois.edu/>



Technical resource that enables **advanced computing, data, and networking** for researchers on campus

- Highly trained professional staff from NCSA
- Continuous deployment model: add/retire hardware as needed
- It is on the smaller scale of computation clusters, but you can have a lot of memory
- Located at *1011 W Springfield Avenue* – **Advanced Computation Building**

**I ILLINOIS**  
Technology Services

**I ILLINOIS**  
NCSA | National Center for  
Supercomputing Applications



NCSA | NATIONAL CENTER FOR SUPERCOMPUTING APPLICATIONS

# ICC Resources: Hardware

## Traditional CPU Nodes

### Intel Xeon 6248

- 20 cores / 40 threads processor
- 2.5 GHz base frequency
- 192 to 384 GB of RAM

### AMD EPYC 7702

- 64 cores / 128 threads processor
- 2.0 GHz base frequency
- 256 GB to 4 TB GB of RAM

## Mixed CPU/GPU Nodes

Traditional CPU Node type capable of accepting 1-4 GPUs

- **NVIDIA GeForce RTX**
- **NVIDIA A6000, A40 and A100**
- **NVIDIA Tesla T4**  
16 to 48 GB GPU Memory  
2,560 to 10,752 CUDA cores

## Storage

- **DDN SFA 14KX**  
420 x 8 TB SAS (2.5 PB usable)
- **Samsung PM1725**  
12 x 3.5 TB NVME (19.2 TB usable for metadata)
- 6 x **Dell R740**: I/O Core Servers
- 3 x **Dell R6515**: Export Servers



# Examples of research applications



## Dark Energy Survey maps the skies

Dr. Felipe Menanteau and the DES team uses ICCP to try to understand where everything came from and if we are alone.



## What makes us humans?

Drs. Tandy Warnow and Michael Nute are researching what makes us different from other species and from one another by producing biological sequence analyses as it relates to genetic trees.



## Unraveling the Mysteries of the Brain

Dr. Xiaohui Chen and his team are mining data to help neuroscientists find the mechanism that makes neurodegenerative diseases tick.



## The Future of Water

Dr. Maria L. Chu studies how a variety of factors alter how water moves in the watershed, affecting its quantity and quality.

... and many other scientific applications: <https://campuscluster.illinois.edu/science/teams/>



# CAPS Partition Overview

You have been added to a “partition” attributed to the CAPS investment.

Details: [https://caps.ncsa.illinois.edu/get-involved/comp\\_res\\_camp\\_clus/](https://caps.ncsa.illinois.edu/get-involved/comp_res_camp_clus/)

- Regular memory nodes
- High memory nodes
- Node-local storage (~8 TB/node)

Queue names:

- caps (all machines)
- caps-main (regular memory)
- caps-highmem (high memory)

Note **best practices** on this page (more later).





# Outline

## Overview

~ 20 min

- What is the ICC?
- Who is it for?
- Examples of research applications
- Why should I use it?

- How can I access it?
- How can I run my codes and applications?
- What else is available at ICC?
- Support material and further learning

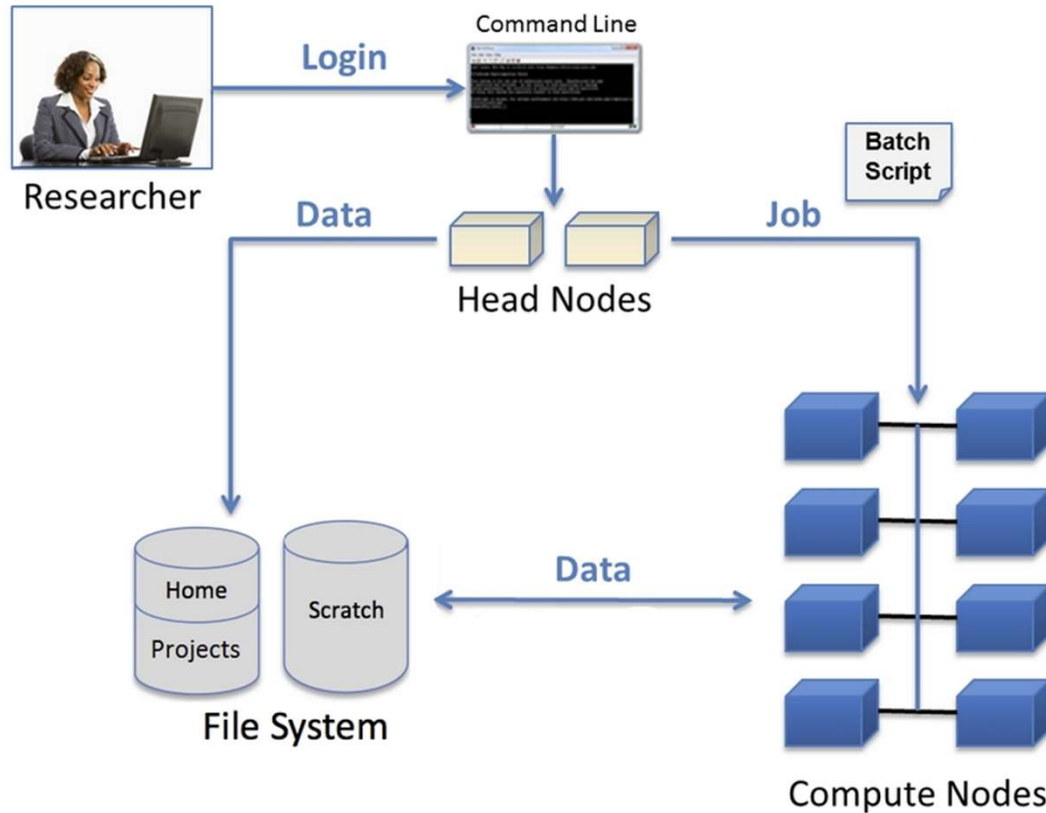
## Access and Usage

~ 55 min

- Q&A session (~ 15 min)



# ICC Access and Usage Overview



**We will discuss each one of these components with 8 hands-on exercises:**

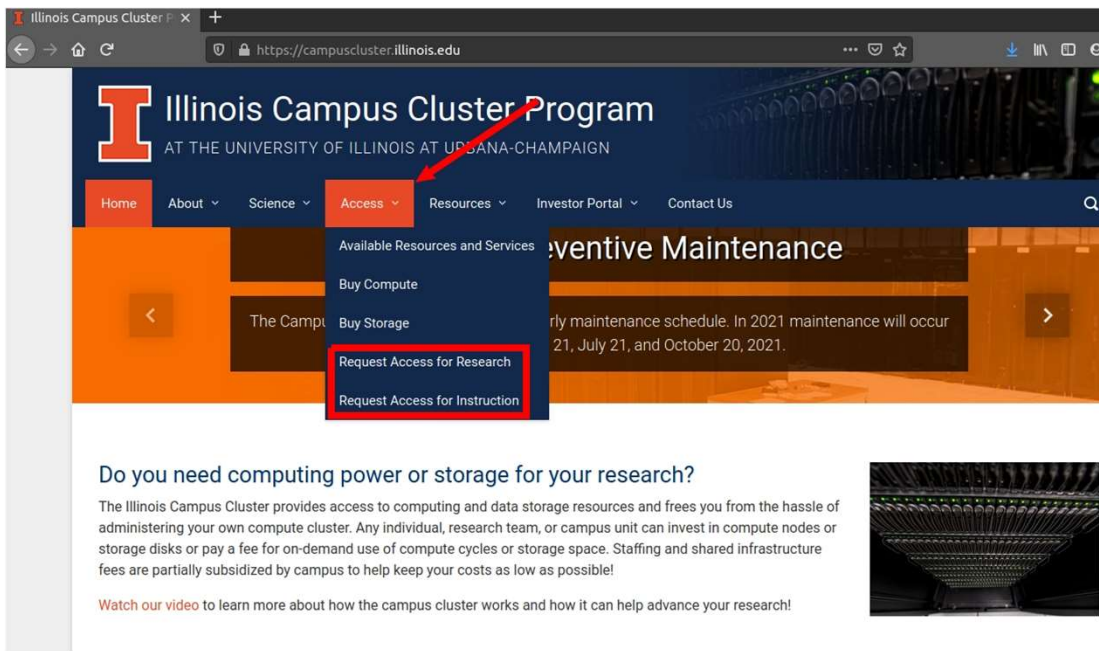
1. Accessing via SSH
2. The file system
3. Copying and moving file
4. Compiling code
5. Using modules
6. Job scripts
7. Keeping track of your jobs
8. Transferring files



# Requesting an account

If you don't have an account yet, you can request it. Check to see if your department is an investor already!

<https://campuscluster.illinois.edu/>



Illinois Campus Cluster Program  
AT THE UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Home About Science Access Resources Investor Portal Contact Us

Available Resources and Services  
Buy Compute  
Buy Storage  
Request Access for Research  
Request Access for Instruction

Do you need computing power or storage for your research?

The Illinois Campus Cluster provides access to computing and data storage resources and frees you from the hassle of administering your own compute cluster. Any individual, research team, or campus unit can invest in compute nodes or storage disks or pay a fee for on-demand use of compute cycles or storage space. Staffing and shared infrastructure fees are partially subsidized by campus to help keep your costs as low as possible!

Watch our video to learn more about how the campus cluster works and how it can help advance your research!

## List of Investors:

<https://campuscluster.illinois.edu/about/investors/>

- **Access for Research** is likely to be your option

1. You will be redirect to **Shibboleth**
2. Authenticate with your NetID and password
3. Fill out the form
  - Info about you
  - Info about Professor/Advisor
    - Name, NetID and queue



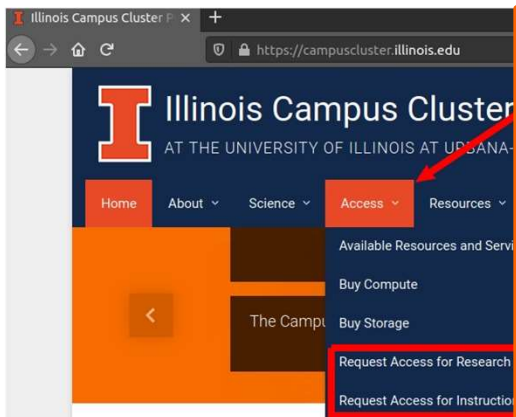
# Requesting an account

If you don't have an account yet, you can request it. Check to see if your department is an investor already!

<https://campuscluster.illinois.edu/>

**List of Investors:**

<https://campuscluster.illinois.edu/about/>



You are already added to the CAPS partition on the Illinois Campus Cluster! (so nothing to worry about here)

• **Research** is likely to be

redirect to **Shibboleth** with your NetID and

form about you

- Info about Professor/Advisor
  - Name, NetID and queue

## Do you need computing power or storage for your research?

The Illinois Campus Cluster provides access to computing and data storage resources and frees you from the hassle of administering your own compute cluster. Any individual, research team, or campus unit can invest in compute nodes or storage disks or pay a fee for on-demand use of compute cycles or storage space. Staffing and shared infrastructure fees are partially subsidized by campus to help keep your costs as low as possible!

Watch our video to learn more about how the campus cluster works and how it can help advance your research!

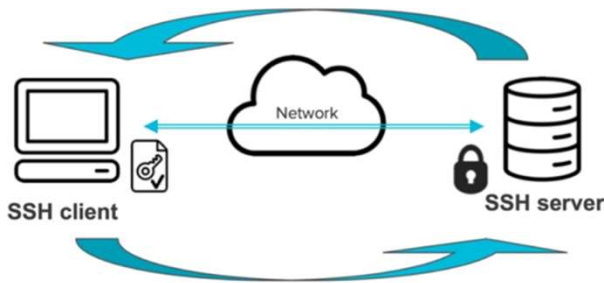


# SSH-client access

SSH access is usually done via *command line*, so:

## SSH? Secure Shell

Provides a secure channel over an unsecured network by using *client-server* architecture



## 1. Open your Terminal

### Mac OS

- Use Spotlight Search (Command+Space)
  - or
  - Use Finder
- to search for "Terminal"

### Windows OS

Start -> Windows System -> Command Prompt

### Linux OS

- Ctrl+Alt+T
- or
- Search for "Terminal" in Applications

## 2. Connect using *ssh*

```
ssh -l <YourUsername> cc-login.campuscluster.illinois.edu
```



# Exercise 1: Access via SSH client

## a) Open your Terminal

"Terminal" on *Spotlight Search* or *Finder*



Start -> Windows System  
-> Command Prompt



Ctrl+Alt+T



## b) Connect to ICC

```
ssh -l <YourUsername> cc-login.campuscluster.illinois.edu
```

You may get: *The authenticity of host 'blah.blah.blah (10.10.10.10)' can't be established.  
RSA key fingerprint is a4:d9:a4:d9:a4:d9a4:d9:a4:d9a4:d9a4:d9a4:d9a4:d9. YES  
Are you sure you want to continue connecting (yes/no)?*

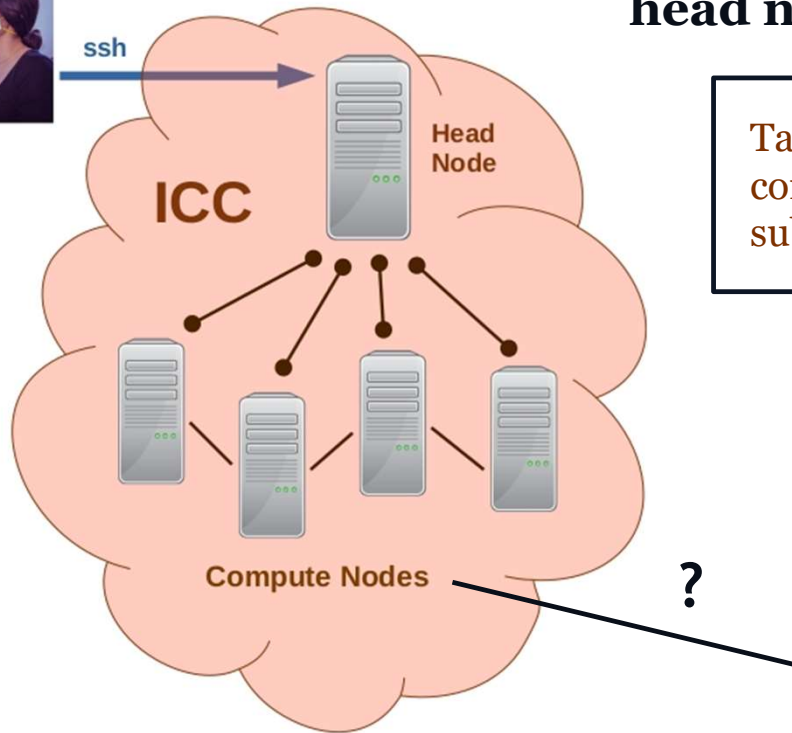


# Head and Compute Nodes

You (local machine)



ssh



SSH directs you to one of the cluster's **head nodes** by default

Tasks such as file editing, code compilation, data backup, job submission, and tracking



**Do not run your applications here**

Once you are here, your Terminal is essentially a Bash Shell in the cluster environment (Linux)

LOADING...



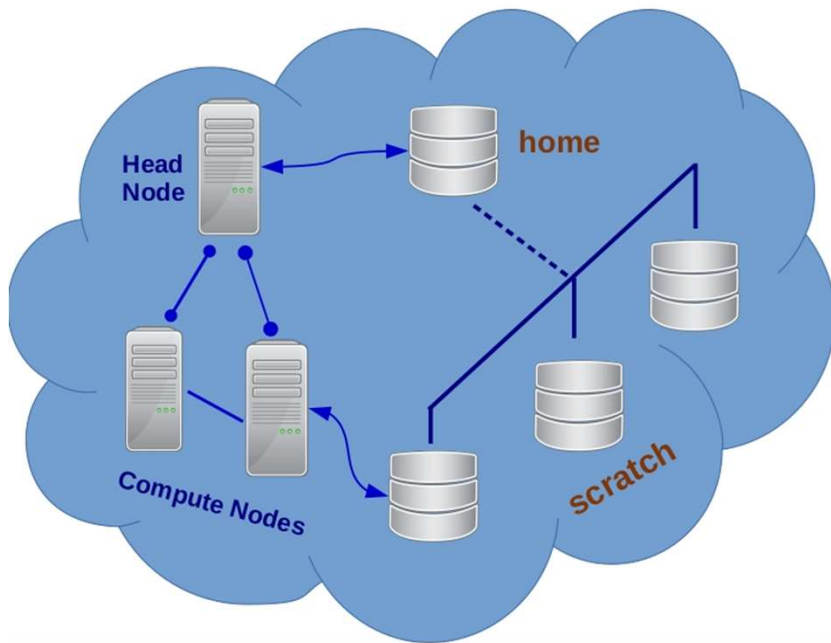
The **compute nodes** are where your applications are going to run - *we will get there soon!*



# File System

- Nodes perform **computing work** with light (*Head*) or heavy (*Compute*) loads

↓  
data manipulation



## Main Storage Areas

Name	Quota	Purge Policy	Snapshots
<b>home</b>	5 GB/user	No Purge	Daily for 30 days
<b>scratch</b>	10 TB/user	Daily Purge for files older than 30 days	No



Files older than 30 days in the **scratch** area are **automatically purged**.

**Backup consistently!**





## Exercise 2: Navigating the file system

a) Use **pwd** to verify your current folder/area

b) Use **ls** to see what files/folders you have there

c) Use **cd** to change current directory to **scratch**. Repeat a, b.

```
cd <target_directory>
```

d) Use **quota** to verify your usage in each area

Symbol	Target
.	Current directory
..	Parent directory
~	Home directory
-	Previously accessed directory



# Exercise 3: Copying and moving files

a) Use `mkdir` to create a new folder in your home area named *HeLLoWorld*

```
mkdir <new_folder_name>
```

b) Use `cp` to copy the *HeLLoWorld.cpp* file located at */home/babreu/ICCPWorkshop* to your home folder

```
cp <original_file_location> <copy_location>
```

c) Use `mv` to move this file to the folder you created in a)

```
mv <current_file_location> <target_file_location>
```

d) Use `cp` to copy the file that you just moved from *HeLLoWorld* to home

e) Verify that copies are indeed in both folders using `ls`

f) Remove the copy in your home folder using `rm`

```
rm <file_location>
```

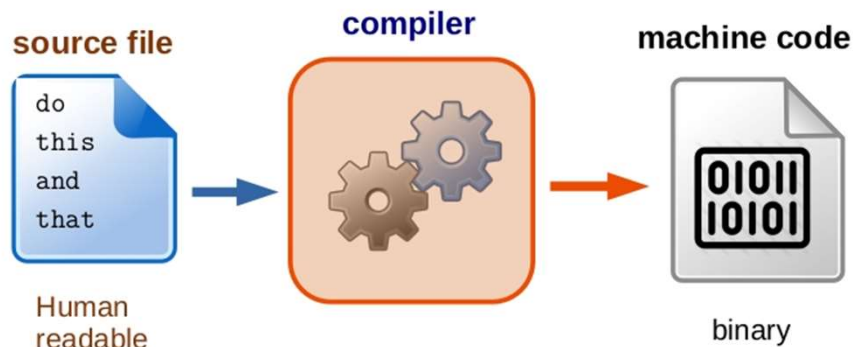
**Tip:** after each item, use `ls` to check if it worked!



# Compiling code

GNU compilers are by default available when you log in the **Head Nodes**

- **gcc**: compiles C code
- **g++**: compiles C++ code
- **gfortran**: compiles Fortran code



## General compiling syntax is

```
compiler <source_code_file> -o <binary_file>
```

Language	GNU compiler command
C	<code>gcc MyProg.c -o MyProg.exe</code>
C++	<code>g++ MyProg.cpp -o MyProg.exe</code>
Fortran	<code>gfortran MyProg.f -o MyProg.exe</code>



Many other compilers are available in ICC, but if you want to use them, you need to **load** them – *we will cover that too!*



## Exercise 4: Compiling code

**a)** Go to the directory where the file from the previous exercise is located

**b)** Check the version of the GNU compiler that is available to you in the Head Node

```
gcc --version
```

**c)** Compile *HelloWorld.cpp* using **g++**

```
g++ HelloWorld.cpp -o HelloWorld.exe
```

**d)** Verify that the binary has been generated in your folder



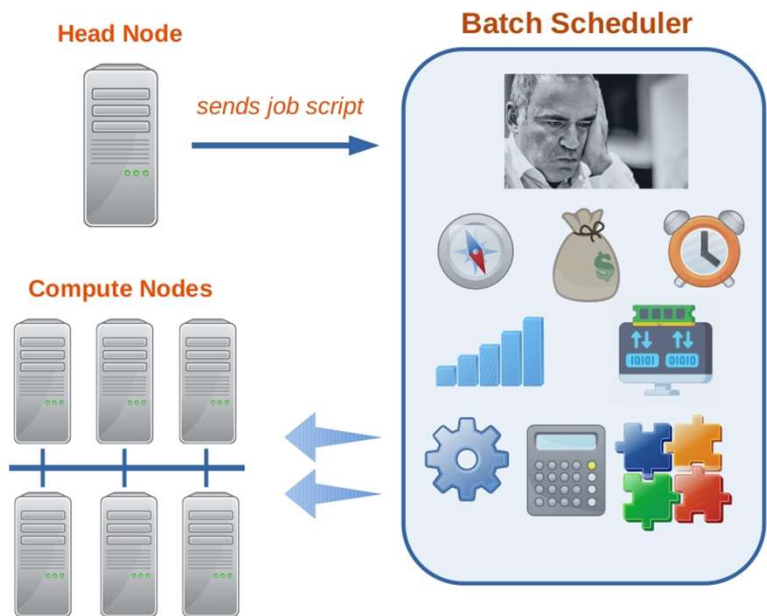
# Batch schedulers (Slurm)



Now that we know how to

- Log in to ICC
- Navigate file system
- Load modules
- Compile code

It is time to use the cluster's power!



**Batch processing** is "the running of jobs that can run without end-user interaction, or can be scheduled to run as resources permit."

The **batch scheduler** is a computer application that weights in a number of factors and resources to determine *where, how and when* a certain request is going to run in the **compute nodes**

In ICC, this application is **SLURM**:

<https://slurm.schedmd.com/>



# Batch job scripts

For SLURM to do its magic, we need to provide it with some **information...**

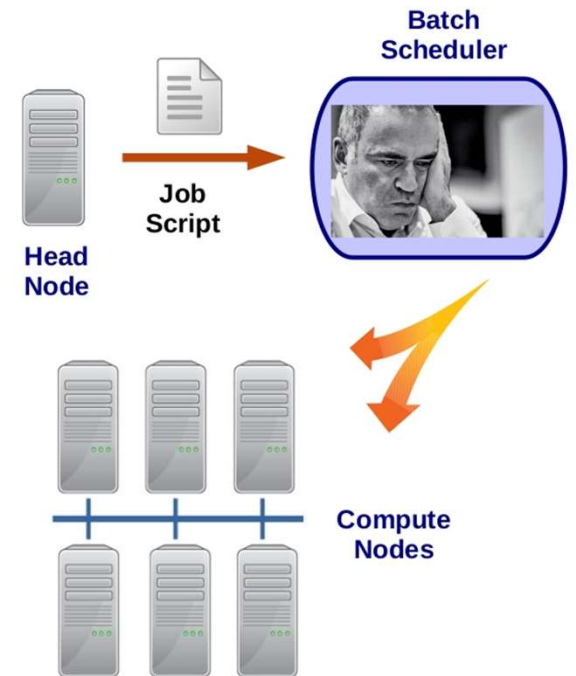
Job Script

```
#!/bin/bash
#SBATCH --time=XX:YY:ZZ
#SBATCH --nodes=N
#SBATCH --ntasks-per-node=M
#SBATCH --partition=Where
#SBATCH --output=OutFile
#SBATCH --error=ErrorFile
#SBATCH --mail-user=Email
#SBATCH --mail-type=When
./HelloWorld.exe
```

We will send bash commands

- Required time
- Number of nodes
- Number of tasks per node
- Where to run it
- Output file
- Error file
- Who to email
- When to email

• What to run



# Exercise 5: Job script for HelloWorld

**a)** Copy the file *HelloWorld.sbatch* located at */home/alameda/HelloTest* to your *HelloWorld* folder

```
cp /home/alameda/HelloTest/HelloWorld.sbatch .
```

**b)** Use **cat** to verify the information that is being passed to the scheduler

```
cat HelloWorld.sbatch
```

**c)** Submit your job scrip to the batch scheduler using **sbatch**

```
sbatch HelloWorld.sbatch
```

**d)** Take note of your job ID



# Exercise 6: Keeping track of your job

- Use the **squeue** commands to monitor the status of your job

## Useful SLURM commands

Command	Action
<code>sbatch &lt;script_file&gt;</code>	Submit job script to the scheduler
<code>squeue -a</code>	List status of all jobs in the batch system
<code>squeue -u &lt;Username&gt;</code>	List status of all your jobs
<code>squeue -j &lt;JobID&gt;</code>	Lists information about a job
<code>scancel &lt;JobID&gt;</code>	Kills a job





# File transfers using SCP

- There is a **specific node** for transferring files over the network

## Sending file from ICC to your machine:

```
scp <Username>@cc-xfer.campuscluster.illinois.edu:<path_to_file> <destination>
```

## Sending file from your machine to ICC

```
scp <path_to_file> <Username>@cc-xfer.campuscluster.illinois.edu:<destination>
```

## Secure Copy Protocol



- Works basically like SSH
- Safe channel to copy/transfer data from or to your local machine
- Command line interface

There are **many** other options available for managing files at ICC

<https://campuscluster.illinois.edu/resources/docs/storage-and-data-guide/>



# Exercise 7: Transfer files with SCP

a) Log out of your cluster session using **logout**

b) Use **scp** to copy the *HelloWorld.out* file to your local machine

```
scp <Username>@cc-xfer.campuscluster.illinois.edu:~/HelloWorld/HelloWorld.out .
```

Your cluster /home  
folder

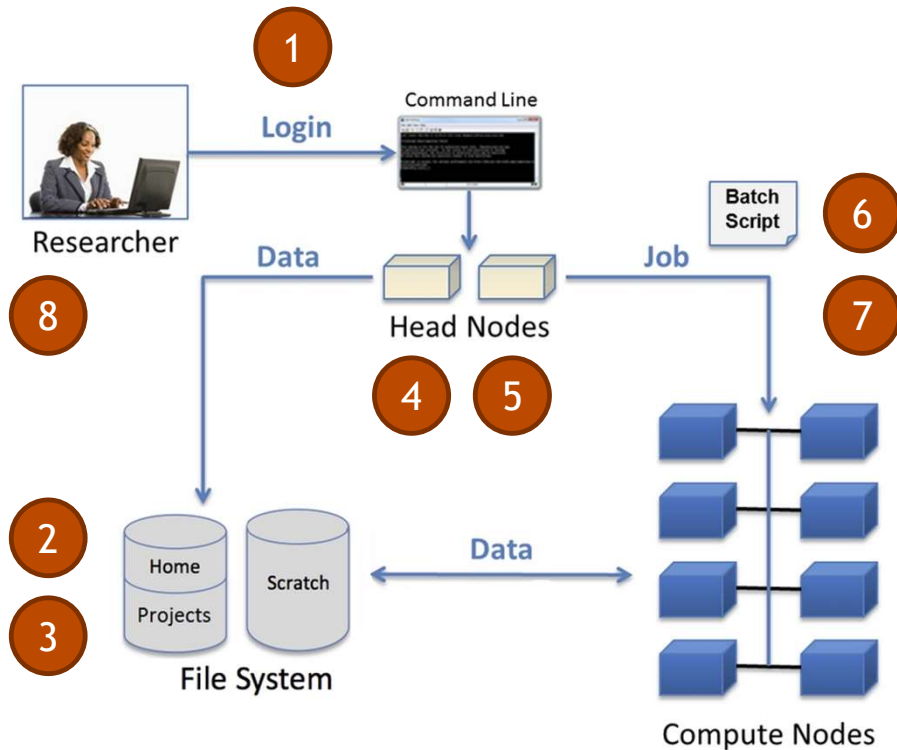
Where you opened  
your terminal

c) Look for the file in your machine and try to open it using your text editor



# Cooling down

We now have a pretty good idea about navigating the Illinois Campus Cluster...



1. Login using SSH client
2. Understand and use the file system
3. Copy and move files around
4. Compile code
5. Load and unload modules
6. Write and submit a job script
7. Keep track of your jobs
8. Transfer files to your local machine



# Doing a bit more

i.e. stretch goals



**National Center for  
Supercomputing Applications**

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

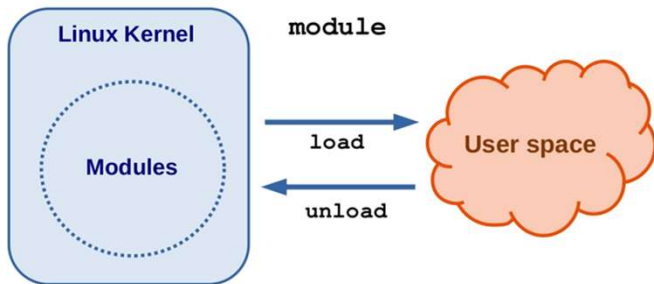
# Working with modules

- What if you want to use a different compiler, or even a different version of GCC?



**Modules** allow you to dynamically load and unload components of the operating system as you need them!

The same approach applies to loading a variety of software and libraries, as well as compilers



## Useful module options

Command	Description
<code>module list</code>	Lists current modules loaded in your session
<code>module avail</code>	Lists all available modules
<code>module help &lt;module_file&gt;</code>	Information about <i>module_file</i>
<code>module load &lt;module_file&gt;</code>	Loads <i>module_file</i> to your environment
<code>module unload &lt;module_file&gt;</code>	Removes <i>module_file</i> from your environment



# Exercise 8: Working with modules

a) Verify the modules currently loaded in your shell environment

```
module list
```

b) Check the list of modules that are available to you

```
module avail
```

c) Try to use the Intel compiler to compile HelloWorld.cpp

```
icc HelloWorld.cpp -o HelloWorld_Intel.exe
```

*Make sure you are in  
the right folder!*

d) Load the intel/18.0 module and repeat c)

```
module load intel/18.0
```

e) Unload the intel/18.0 module

```
module unload intel/18.0
```



# Note re. Anaconda on the cluster

Conda is available on the cluster as a module - note that you can use **apropos** to search

```
[gsn@cc-login2 ~]$ module apropos conda
anaconda/2      :      Name: Anaconda 5.2.0 (Python 2.7.15)
anaconda/2      : Description: Anaconda is a distribution of the Python and R programming languages
anaconda/2      :      URL: https://www.anaconda.com/
anaconda/2018-Dec/2 :      Name: Anaconda 2018.12 (Python 2.7.15)
anaconda/2018-Dec/2 : Description: Anaconda is a distribution of the Python and R programming languages
anaconda/2018-Dec/2 :      URL: https://www.anaconda.com/
anaconda/2018-Dec/3 :      Name: Anaconda 2018.12 (Python 3.6.8)
anaconda/2018-Dec/3 : Description: Anaconda is a distribution of the Python and R programming languages
anaconda/2018-Dec/3 :      URL: https://www.anaconda.com/
```

**BUT !!!**

- a) you are probably part of a research group that should use a consistent set of packages, so have your PI create a shared **.condarc** for the group
- b) conda will create a package cache in **\$HOME**. This will destroy your quota.

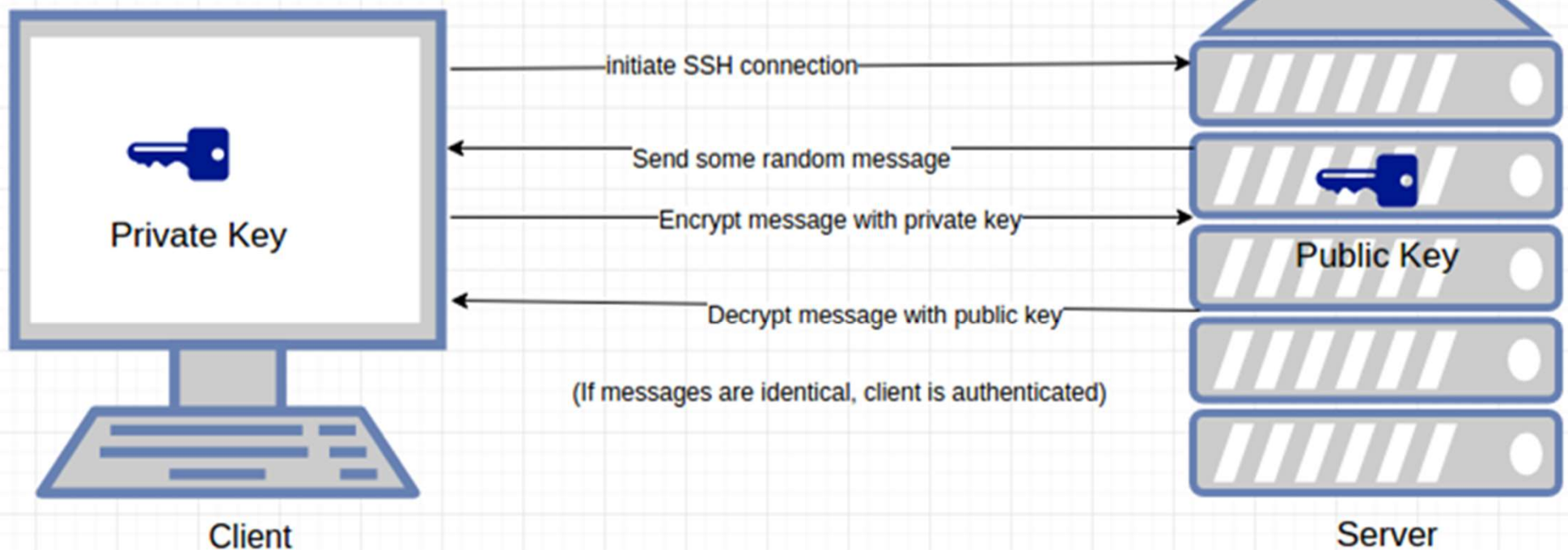


# Exercise 9: Setting up SSH keys

It's going to get really annoying to have to login repeatedly (trust us)

There are two ways to help with this:

1. Make logging in repeatedly easier with **ssh keys**
2. Using **screen/tmux**





## Exercise 9: Setting up SSH keys

- a) Open another terminal on your local machine (keep the terminal connected to the campus cluster open)
- b) On your local machine, generate a public-private key pair (you can skip to c) if you have done this before

**ssh-keygen**  
(and follow the prompts)

- c) Verify that the files were created with

```
ls ~/.ssh/
```

you should find a <filename> and <filename>.pub  
(typically something like id\_rsa if you didn't change it)



## Exercise 9: Setting up SSH keys

d) Copy your **public key** to the campus cluster (or whatever you machine you want to login to). You can use **scp** but it's safer to use **ssh-copy-id**

```
ssh-copy-id <username>@cc-login.campuscluster.illinois.edu
```

**Never ever copy your private keys on a computer somebody else has root on. If you do, you just shared your keys with that person.**

Your public key can be used for multiple services - you don't (generally) need a separate key for different services, unless required to... e.g. if you work on LSST, and get a compute account at NERSC, they will require a different key pair

Different local machine? Generate a different keypair - basically keep your private key machine specific. On your own time read this:

<https://goteleport.com/blog/how-to-use-ssh-agent-safely/>



# Exercise 9: Setting up SSH keys

e) On your local machine, make sure ssh-agent is running

```
ps -u <username> | grep ssh-agent
```

```
if you see nothing, run  
eval `ssh-agent -s`
```

f) Add your key

```
ssh-add
```

We'll set this up so you only have to do this once each time on your local machine, until you restart it.



## Exercise 9: Setting up SSH keys

g) Now try logging into the campus cluster as usual - you shouldn't need a password!

To not have to manage the ssh-agent stuff manually, stick the script here in your `.bashrc` or `.zshrc`

<https://gist.github.com/johnvey/3442925>

h) While you are at it, you might want to create a nice file for your aliases. Add this to the END of your `.bashrc/.zshrc`

```
if [ -f ~/.aliases ]; then
    . ~/.aliases
fi
```

and create a `.aliases` file with this inside

```
alias cclogin='ssh -X <username>@cc-login.campuscluster.illinois.edu'
```



## Exercise 10: Configuring SSH

Of course, you might want different keys for different servers, or to set aliases for hosts

You can configure SSH to do these sorts of things by editing the text file **.ssh/config**

Most of the time, if you need some special configuration, the remote server admins will tell you

Notice the different identity files for my private keys

```
Host slacl
  Hostname s3dflogin.slac.stanford.edu
  IdentityFile ~/.ssh/id_rsa
  IdentitiesOnly yes

Host slacd
  Hostname rubin-dev1
  ProxyJump slacl
  IdentityFile ~/.ssh/id_rsa
  IdentitiesOnly yes

Host observer2.ctio.noao.edu
  HostKeyAlgorithms=+ssh-dss

Host aperturesshduo.stsci.edu
  ServerAliveInterval 120
  Port 8222
  ProxyCommand none

Host *.stsci.edu
  ProxyCommand ssh -qq aperturesshduo.stsci.edu -W %h:%p
  ForwardX11 yes
  ForwardX11Trusted yes
  ServerAliveInterval 120
  #If your home username differs from your STScI AD username:
  #User myuser

Host cori*.nersc.gov dtn*.nersc.gov
  ForwardX11 yes
  ForwardX11Trusted yes
  IdentityFile ~/.ssh/nersc
  IdentitiesOnly yes
  ForwardAgent yes

Host *
  AddKeysToAgent yes
```



# Exercise 10: Configuring SSH

You can also use this to alias host names

Notice how Host and Hostname are set differently here

```
Host cc
  Hostname cc-login1.campuscluster.illinois.edu
  User gsn
  ServerAliveInterval 120
  ServerAliveCountMax 120
  StrictHostKeyChecking no
```

This lets you do:

```
ssh cc
scp cc
sftp cc
etc
```



## Exercise 11: screen/tmux: Alternatives to multiple logins

Sometimes you want to start an interactive job, logout and have the job keep running.

Sometimes you want multiple windows to e.g. code in one window, test in another, look at the file system in the third. Kinda like tabs in a browser.

**screen** (older, more stable, more likely to be installed) and **tmux** (newer, easier to use, but not on campus cluster much sadface) do this for you

To make it slightly more obvious to see what is going on, you might want to:

```
git clone https://github.com/gnarayan/vimrc.git  
cp vimrc/.screenrc ~
```



## Exercise 11: screen/tmux: Alternatives to multiple logins

Launch **screen** in a terminal on the campus cluster

This looks just like your regular shell. Run a `ls` or something in it.

The magic starts when you hit **Ctrl + a c**

**Ctrl + a n/p** to go to next/previous tab.

```
[gsn@cc-login2 ~]$ ls
21B_2half.txt  L14_halos.dat  bayesn  conda-bld  scratch  vimrc  yse_decam_notes.txt
[gsn@cc-login2 ~]$ pwd
/home/gsn
[gsn@cc-login2 ~]$ █
```

```
[ cc-login2 ][ (0*$bash) 1-$ bash
```

```
[gsn@cc-login2 ~]$ decamv20.0
pipedata=/projects/caps/uiucsn/decam/photpipe/v20.0/DECAMNOAO/YSE
gsn@cc-login2(YSE,bash)% cddata
gsn@cc-login2(YSE,bash)% pwd
/projects/caps/uiucsn/decam/photpipe/v20.0/DECAMNOAO/YSE
gsn@cc-login2(YSE,bash)% █
```

```
[ cc-login2 ][ 0-$ bash (1*$bash)
```



## Exercise 11: screen/tmux: Alternatives to multiple logins

The real trick with screen is you can **detach** from it, and reconnect later

To show you how that works, let's fake a long process - within the terminal running screen, run:

```
c=1  
while true; do echo "Hello $c"; let c=c+1; sleep 1; done
```

Detach (not quit) from the screen with **Ctrl + a d**

This puts you back at the terminal from which you launched screen

**screen -ls** - shows you the list of screen sessions you are running

**screen -r <PID>** - reconnects you to the screen session

The best part is your job is still running.

```
[gsn@cc-login2 ~]$ screen -ls  
There is a screen on:  
    21646.pts-13.cc-login2 (Detached)  
1 Socket in /var/run/screen/S-gsn.  
[gsn@cc-login2 ~]$ screen -r 21646
```



# Exercise 12: X11 Forwarding

When you login to the campus cluster you are running processes on the remote machines  
But astrophysics is very visual, and you want to be able to look at your output  
If you login via ssh, you might notice:

```
gsn@golubh3(YSE,bash)% ds9 -zscale ds9 2022xiw.221015.1141836_ooi_z_v1_S4.sw.fits
application-specific initialization failed: couldn't connect to display ""
Unable to initialize window system.
gsn@golubh3(YSE,bash)% xclock
Error: Can't open display:
```

So what's going on...

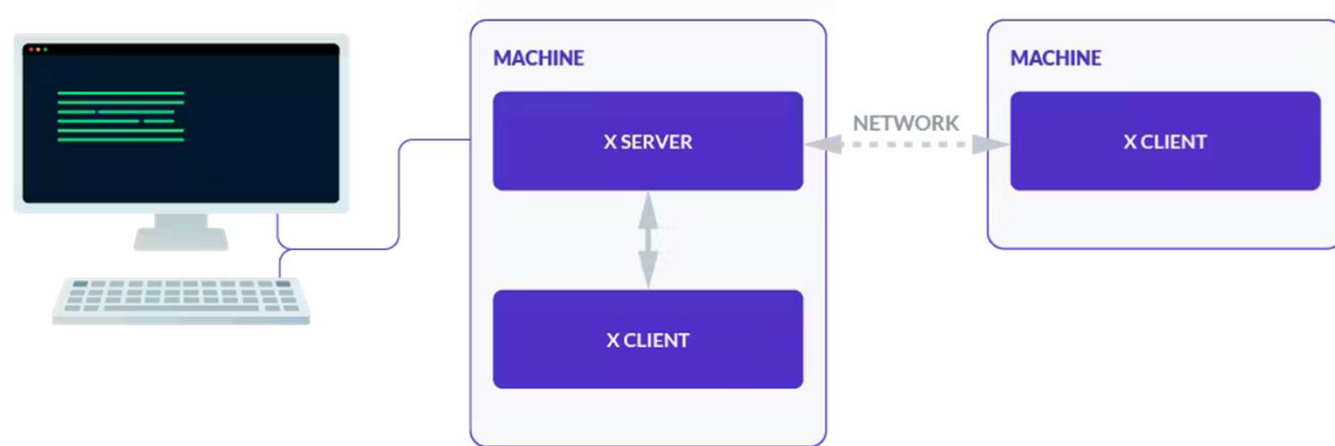


# Exercise 12: X11 Forwarding

X11 refers to the 11th edition of the X Window System; an open source graphics protocol developed in the early days of the internet.

It provides a basic framework for creating custom GUIs that can display graphics on both local and remote display devices.

**X11 forwarding** is a mechanism that allows a user to start up remote applications, and then forward the application display to their local machine.



# Exercise 12: X11 Forwarding

The remote server will have to have X11 forwarding enabled - this is normal for more shared compute environments.

Next, you'll need a X11 server locally. If you are on a Linux machine, you already have a X11 server installed by default.

Mac: <https://www.xquartz.org/index.html>

Windows: <https://sourceforge.net/projects/xming/>

If you are on a Mac, installing XQuartz will also setup your **DISPLAY** environment variable

If you are on Windows and using WSL2, you'll also need this in your .bashrc

```
export DISPLAY="$(hostname).local:0"
```



# Exercise 12: X11 Forwarding

If you setup the **cclogin** alias, use it else

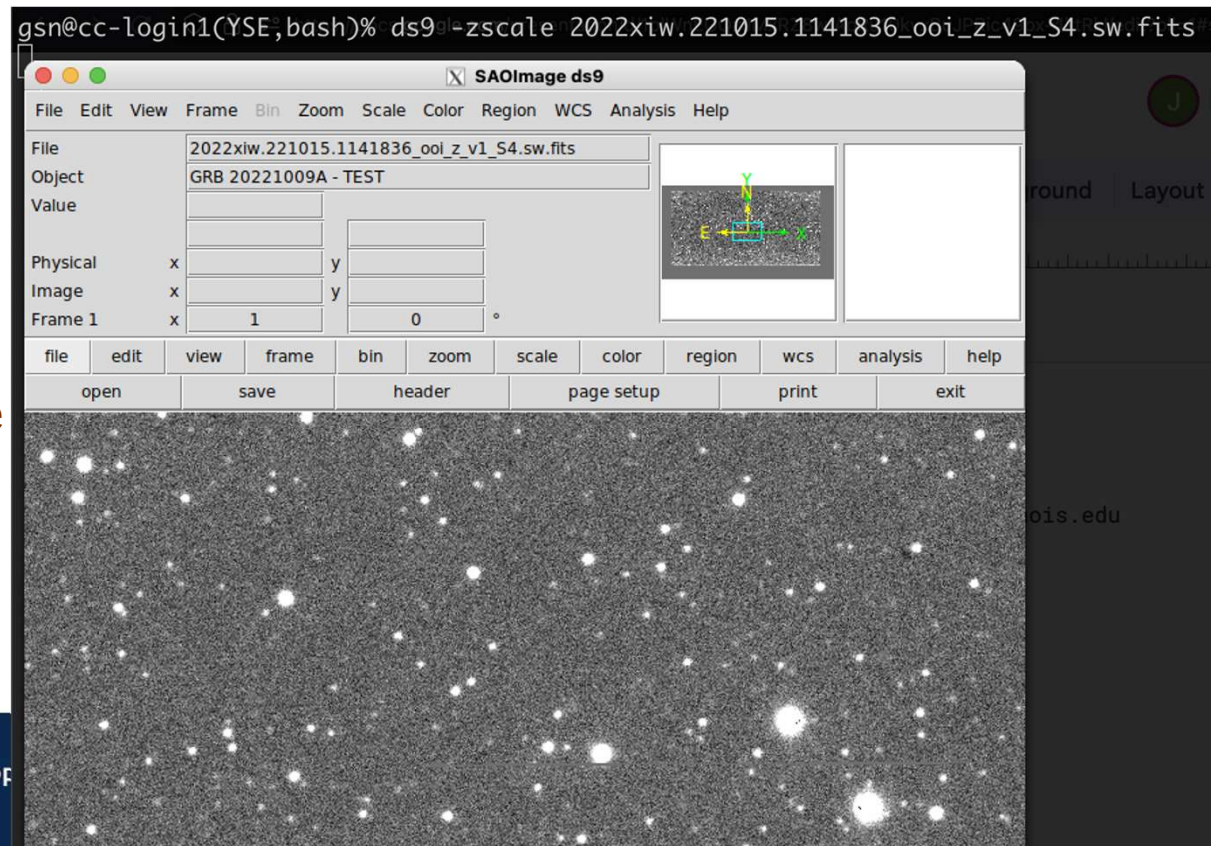
```
ssh -X <username>@cc-login.campuscluster.illinois.edu
```

- the **-X** enables X11 forwarding

Try some X11 app like `xclock`

Now you should be able to see images/plots/any GUI

These will render on your local machine with the data forwarded to your local X11 server over SSH from the remote machine



# Being a good citizen on the cluster

Clusters are shared compute environments, and you are part of a research group. Chances are that the files you create are going to be needed by other people in your group. Unix creates files with very restrictive permissions by default - only you can read or write your own files.

Solution: Add **umask 002** to the end of your `.bashrc` file.

This changes default permissions to be group readable/writable, and world readable, but not writable. This will make your PI less likely to want to strangle you.

Use **chmod** to fix permissions in specific directories that need to be more restrictive (This is why we are doing this now, at the end, instead of before creating `.ssh` keys....)



# Being a good citizen on the cluster

Again, do not run long jobs on the head node/login node. If you do, it'll probably be killed automatically and we'll all point at you and laugh.

To run a program for testing over a short amount of time, you can use interactive jobs. You can open an interactive job like that with this alias (add to your `.alias` file):

```
alias node='srun --partition=caps --time=03:00:00 --nodes=1 --ntasks-per-node=16 --pty  
/bin/bash'
```

```
alias longnode='srun --partition=caps --time=24:00:00 --nodes=1 --ntasks-per-node=16 --pty  
/bin/bash'
```

Use **longnode** judiciously - test long job stability, not run in production.

You can also submit to the **caps-highmem** partition if your job needs more memory.



# Being a good citizen on the cluster

Even with these precautions, it is possible that all the nodes are being used. You can check the status of the **caps** partition with:

```
alias capsstatus='sinfo --partition caps -N'
```

You can check the status of your own jobs with

```
alias sq='squeue --format="%.18i %.9P %.50j %.8u %.8T %.10M %.9l %.6D %R" -u <username>'
```

Once you are sure you are not the problem yourself

```
squeue | grep caps
```

And see the username column to see who is using the cluster. If you have a time-critical job, send them a message on slack. **If your job isn't time-critical, submit to the queue and wait.**





# Stretching

**A few other exercises that can help developing the concepts we have just learned:**

- Edit the job script to include your email address that will be notified when your job starts running and when it terminates
- Write a simple code in your local machine. Transfer it to the cluster, compile it. Write a job script for your executable and run it in the Compute Nodes. Transfer the output back to your machine and analyze it
- Use one of the other options available at <https://campuscluster.illinois.edu/resources/docs/storage-and-data-guide/> to manage your files with a graphic interface
- Load and unload different modules: if you have some experience with Python, try to use the console and perform some simple Python 3 tasks.
- Fire up an interactive session on the Compute Nodes using **srun**: <https://campuscluster.illinois.edu/resources/docs/user-guide/#jobs>
- Get creative and use your temporary allocation to learn more about the cluster!



# Resources

- **ICCP official *Getting Started* webpage**

<https://campuscluster.illinois.edu/resources/docs/start/>

- **ICCP official *User Guide***

<https://campuscluster.illinois.edu/resources/docs/user-guide/>

- **ICCP System Info: Hardware, Storage, Cluster Monitor**

<https://campuscluster.illinois.edu/about/system-info/>

- **Self-paced *Getting Started on ICC – HPC Moodle* page**

<https://www.hpc-training.org/xsede/moodle/>



# Resources

- **SLURM Documentation**

<https://slurm.schedmd.com/quickstart.html>

- **Basic Linux commands for beginners**

<https://maker.pro/linux/tutorial/basic-linux-commands-for-beginners>

- **Environment Modules documentation**

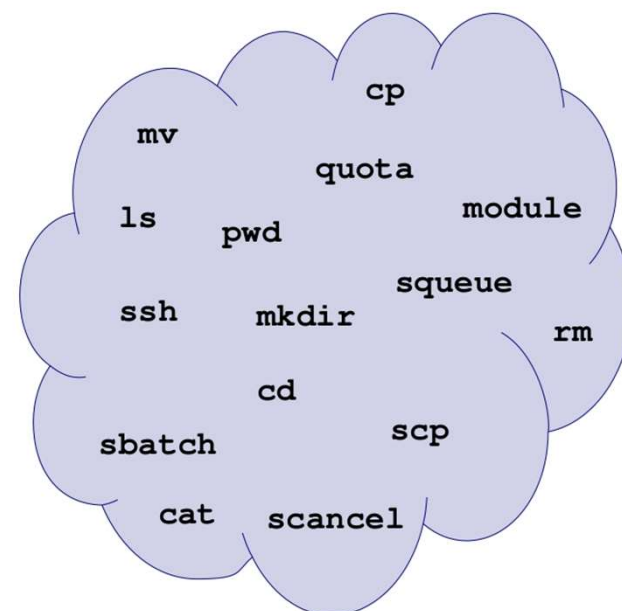
<https://modules.readthedocs.io/en/latest/>

- **Examples of Best Practices for running jobs in general**

<https://docs.nersc.gov/jobs/best-practices/>

[https://dccn-hpc-](https://dccn-hpc-wiki.readthedocs.io/en/latest/docs/cluster_howto/best_practices.html)

[wiki.readthedocs.io/en/latest/docs/cluster\\_howto/best\\_practices.html](https://dccn-hpc-wiki.readthedocs.io/en/latest/docs/cluster_howto/best_practices.html)



# Getting help

**If you are having issues using the Illinois Campus Cluster and cannot find a solution in the official documentation:**

- **Getting Started**

<https://campuscluster.illinois.edu/resources/docs/start/>

- **User Guide**

<https://campuscluster.illinois.edu/resources/docs/user-guide/>

## **Reach out!**

[help@campuscluster.illinois.edu](mailto:help@campuscluster.illinois.edu) or ASK ON THE CAPS SLACK CHANNEL



# Q&A

Have further questions about this workshop?  
[gsn@illinois.edu](mailto:gsn@illinois.edu) or [alameda@illinois.edu](mailto:alameda@illinois.edu)



Feedback form

<https://forms.gle/GeBgSgTLknkd8F3s6>



**National Center for  
Supercomputing Applications**

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN