HPC–101
Onboarding

Director: Gwen Jacobs
Sean Cleveland, Jennifer Geis, Ouida Meier,
Ron Merrill, David Schanzenbach

Information Technology Services
Cyberinfrastructure
University of Hawai'i

https://www.hawaii.edu/its/ci/
uh-hpc-help@lists.hawaii.edu

July 17, 2017

Outline

Outline

Parallel Computing

- High Performance Compute
  - Each separate process can send and receive data amongst other processes (MPI & OpenMP)
  - If processes must communicate for the overall program to proceed, high-speed networking is needed (only MPI)
- High Throughput Compute
  - *Pleasantly Parallel* – Processes are independent and no communication is necessary

Outline

Cray CS300 – History

2014

- Jul. – University of Hawai'i (UH) selects Cray's proposal
- Oct. – Delivered 184 nodes, 3,800 cores
- Dec. – Passed testing and accepted

2015

- Jan. – Early adopter testing started (8 users)
- Apr. – Opened for general use

2016

- Mar. – 92 nodes added, 5,876 cores total

2017

- Jan. – 5 nodes added, 5,992 cores total
- May – 5 nodes added, 6,092 cores total
- Jun. – 452+ users have been granted access to the cluster

Cray CS300 – Compute Nodes

Community Nodes[1]

Type	Cores	Ivy-Bridge	Haswell	Broadwell	128 GB[2]	256 GB[2]	1 TB[2]
Standard	20	178			178		
Large	40	6					6
GPU[3]	20		1		1		

Condo Nodes[1]

Type	Cores	Ivy-Bridge	Haswell	Broadwell	128 GB[2]	256 GB[2]	1 TB[2]
Standard	24		62		62		
Standard	20			5	5		
Custom	20		37			33	4
GPU[3]	20		1		1		

[1] Nodes are diskless and run CentOS linux

[2] Usable RAM: ≈ 118 GB of 128 GB, ≈ 246 GB of 256 GB, ≈ 1003 GB of 1 TB

[3] 2 x Nvidia Tesla K40

Cray CS300 – Storage

Two storage options are currently available on the Cray CS300

1. Lustre®
   - High performance parallel filesystem
   - Available on all the cluster nodes
   - Freely available to all users
2. ValueStorage
   - Network attached scale out storage
   - Only available from the login nodes
   - Available as a for fee service

Cray CS300 – Storage → Lustre®

- The Cray CS300 has ≈ 582TB of storage space
- Primarily used as scratch space for jobs (Input and Output)
- User do not have a usage quota (soft or hard)
- Certain directories are subject to a 90 day purge policy
- **Data is not backed up! Users are responsible for their own data**
- Lustre® utilizes RAID 6 arrays, and is fairly robust but . . .
  RAID is not a backup

Cray CS300 – Storage → ValueStorage

- 500TB of scale out storage
- Purchased annually in 0.5TB increments

ValueStorage Pricing

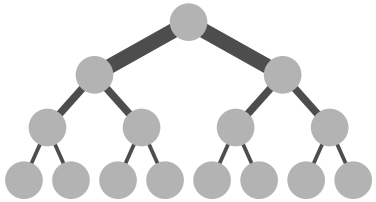
Product	Annual Cost
0.5TB	$65.00
0.5TB + Replication	$130.00

More Information

All prices are subject to change

Cray CS300 – Network

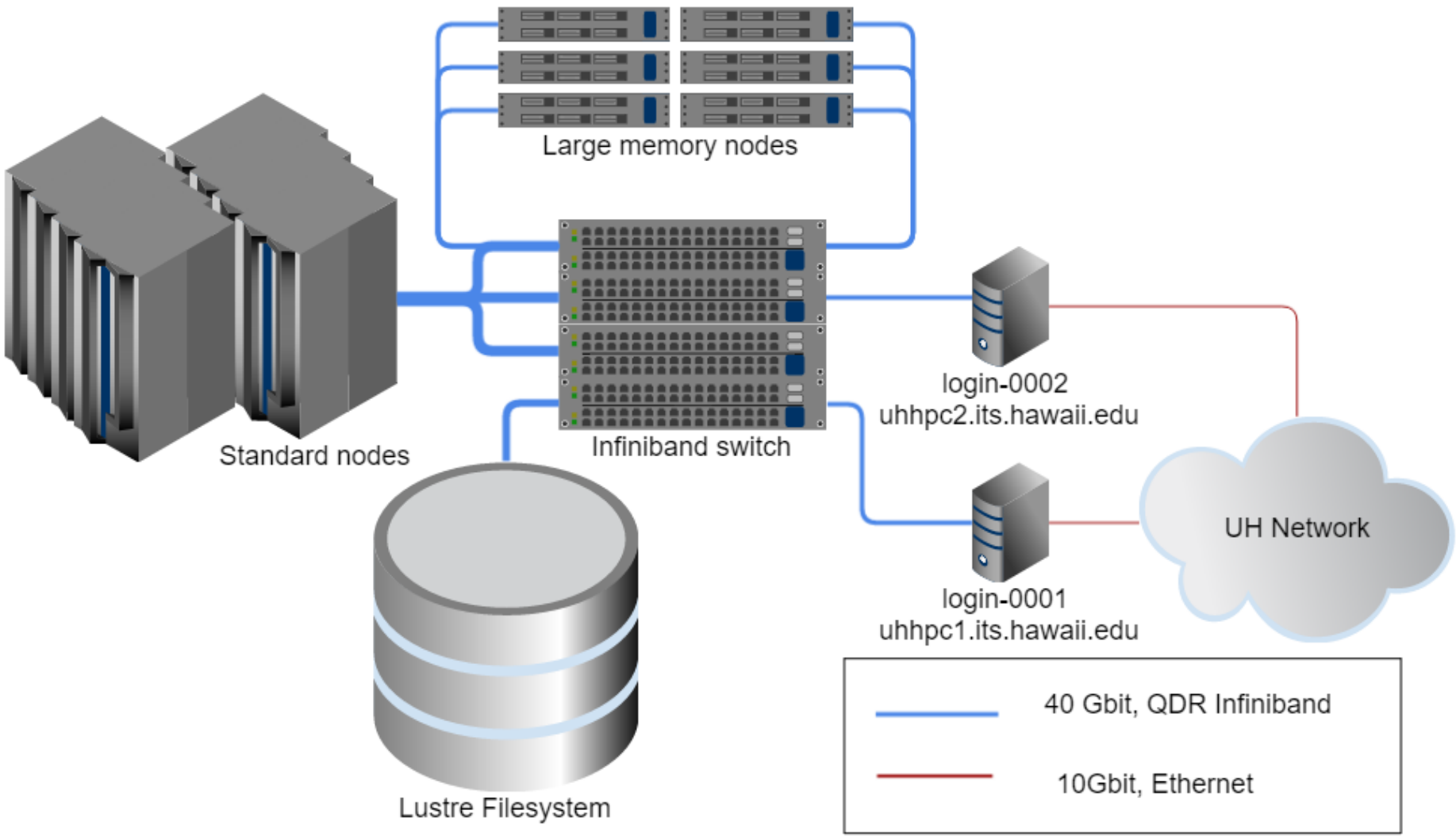
- 40Gb Infiniband inter-connects (QDR)
  - High speed inter-connect between compute nodes, Lustre® storage and Login nodes
  - low latency ($\approx 1.3\mu$s)
  - Utilizes the *fat tree network topology*



Source: https://en.wikipedia.org/wiki/Fat_tree

- 10Gb login node internet connectivity
  - Speed test from UH to CERN clocked transfer speeds up to 2+ Gb/s

Cray CS300 – Layout



Outline

Community Resources

- The initial investment in the cluster provides resources for free to all faculty, staff, and students affiliated with the University of Hawai'i
- All users can run on the publicly accessible partitions: *community.q, exclusive.q, lm.q, gpu.q, sb.q, kill.q, htc.q*
- For some users, the publicly available resources may not be enough . . .

Condo Model

- The Condo model allows users to buy nodes (*condos*) to incorporate into the cluster
- Node owners are provided with priority access to their purchased hardware
- All nodes have a *5 year warranty*
  - Once a nodes warranty has expired, it will be removed from the cluster
- Condo owners are given early access to purchased nodes
  - Access is granted as soon as the funds land in our accounts
  - The 5 year warranty will not begin until newly ordered nodes are installed
- Condo owners are also given the option to purchase 1TB of Lustre® storage per node purchased

Service Units

- In some cases, users will not require owning a node, but still need priority access
- An alternative to purchasing a node, is to purchase service units (SU)
- SUs come in two flavors:
  - Standard node units
    - 20 core hours, with access to 128GB of ram on a standard node
  - Large memory node units
    - 40 core hours, with access to 1TB of ram on a large memory node
- A minimum order totaling $500 is required

Price Card

Condo Price Card

Product	Cost	Product	Cost
Standard	$6,600.00	GPU Node	Contact for pricing
Large Memory	$20,000.00	1 TB Lustre® Storage for 5 years	$600.00

Service Unit Price Card

Product	Cost	Minimum Order
Standard node	$0.50 per SU	1,000 SU ($500.00)
Large memory node	$2.00 per SU	250 SU ($500.00)

All prices are subject to change

Outline

Overview – Cluster Interaction

- Connecting to a cluster @ UH
  - Login to the cluster
  - Verify user permissions
- User directories
- Transferring files
  - Globus
- Software
  - Modules
  - Acquiring software
  - Compilers
- Managing user jobs
  - Job scheduler
  - Using SLURM
  - Partitions
  - Submitting jobs (Examples)

Connecting to a cluster @ UH

- To connect to the cluster, an SSH client is required
- Linux and MacOS X, typically have a SSH client already installed
- Windows typically needs a 3rd party SSH client
- Suggested SSH clients for Windows include:
  - SSH Secure Shell (SSH 3.2.9)
  - Putty
  - MobaXterm

- The Cray CS300 has two login nodes:
  - uhhpc1.its.hawaii.edu
  - uhhpc2.its.hawaii.edu

Let's attempt to login!

Login to the Cray CS300

Windows

- If SSH 3.2.9 installed (Lab PCs have it installed)
- Open the start menu, and type "SSH" and you should see a program called "SSH Secure File Terminal Client"
- Click "Quick Connect" and enter the following information:

  Host Name: uhhpc1.its.hawaii.edu –OR– uhhpc2.its.hawaii.edu
  User Name: Your UH User name e.g., user99
  Port: 22

- Press "Connect"
- Enter your UH user password when prompted and press the return key

Login to the Cray CS300

Mac & Linux

- Open a terminal window
- Enter one of the following:
  - ssh <UH User name>@uhhpc1.its.hawaii.edu
  - ssh <UH User name>@uhhpc2.its.hawaii.edu
  - **Example:** ssh user99@uhhpc1.its.hawaii.edu
- Enter your UH user password when prompted and press the return key

On Initial Login . . .

Validate that all system permissions are correct for your user

1. Test that you can list files in your home: '`ls -la`'
2. Test making a file in your home: '`touch test.txt`'
3. Go into ~/lus: '`cd ~/lus/`'
4. Test making a file in your lus directory: '`touch test.txt`'
5. Go into ~/apps: '`cd ~/apps/`'
6. Test making a file in your apps directory: '`touch test.txt`'

Result

Did you get any errors? Let us know if you did

Notes:

- On login you are placed in /home/<username>/
- By default, ~ is equivalent to /home/<username>/

Overview – Cluster Interaction

- Connecting to a cluster @ UH
  - Login to the cluster
  - Verify user permissions
- User directories
- Transferring files
  - Globus
- Software
  - Modules
  - Acquiring software
  - Compilers
- Managing user jobs
  - Job scheduler
  - Using SLURM
  - Partitions
  - Submitting jobs (Examples)

User Directories

Home

```
[user99@login ~]$ ls -l
total 0
lrwxrwxrwx 1 user99 user99 23 Jan 15 20:38 apps -> /lus/scratch/usr/user99
lrwxrwxrwx 1 user99 user99 19 Jan 15 20:38 lus -> /lus/scratch/user99
lrwxrwxrwx 1 root   root   37 Jan 15 20:41 purge -> /lus/scratch/log/purge/current/user99
```

- ~/ is not on the Lustre® filesystem and **should not be used for job data!**
- ~/lus/ is a symlink to the Lustre® scratch
  - This is where all your job data files should live
  - Items in this directory ***are*** subject to our 90 day purge policy
- ~/apps/ is a symlink to where programs should be stored
  - Items in this directory ***are not*** subject to our 90 day purge policy
  - Directory is monitored for abuse
- ~/purge/ is typically a dead symlink
  - Symlink becomes active if the user has files that are part of the next automatic purge
  - When ~/purge/ is active, the directory containing two files – *purge_list.txt* & *totals.txt*
  - An email is sent to users if they have files that will be purged
  - Email notification is sent out 14 days before the purge takes place

User Directories

Filesystems

```
[user99@login ~]$ df -h
Filesystem                                  Size  Used Avail Use% Mounted on
10.10.0.3:/ha_cluster/home                  1.8T  888G  851G  52% /home
10.12.0.51@o2ib:10.12.0.52@o2ib:/scratch    582T  429T  125T  78% /lus/scratch
```

- /home/<username> exists on a NFS mounted filesystem
  - Only has 1.8TB of useable space
  - Using all this space may cause problems for the entire cluster
  - Not a high performance filesystem and small in size
- /lus/scratch/ is the Lustre® filesystem
  - Has 582TB of useable space
  - ~/apps/, ~/lus/, and ~/purge/ all point to directories on this filesystem
  - High performance and a lot more space for users to use
  - No hard or soft quotas are in place
  - Utilization is managed through the 90 day purge policy

Overview – Cluster Interaction

- Connecting to a cluster @ UH
  - Login to the cluster
  - Verify user permissions
- User directories
- Transferring files
  - Globus
- Software
  - Modules
  - Acquiring software
  - Compilers
- Managing user jobs
  - Job scheduler
  - Using SLURM
  - Partitions
  - Submitting jobs (Examples)

Available File Transfer Protocols

- The cluster has the following options for transferring files:
  - scp (RCP+SSH protocol)
  - rsync (rsync protocol with SSH transport)
  - SFTP (SSH FTP protocol) – Filezilla, Cyberduck
  - Globus (Grid FTP protocol)
- All options are widely used, and have clients that can be found for on most major operating systems

SFTP, scp, and rsync are fairly common on Linux systems,
but Globus is not as common . . .

Globus

What is Globus?

The Globus transfer service provides high-performance, secure, file transfer and synchronization between endpoints.

Globus handles all the difficult aspects of data transfer, allowing application users to easily start and manage transfers between endpoints, while automatically tuning parameters to maximize bandwidth usage, managing security configurations, providing automatic fault recovery, and notifying users of completion and problems.

Definition

An ***endpoint*** is one of the two file transfer locations – either the source or the destination – between which files can move. Once a resource (such as a server, cluster, storage system, laptop, or other system) is defined as an endpoint, it will be available to authorized users who can transfer files to or from this endpoint.

Please visit the Cyberinfrastructure Globus quick start guide for instructions on how to use Globus: http://go.hawaii.edu/jMW

---

https://www.globus.org/file-transfer

Overview – Cluster Interaction

- Connecting to a cluster @ UH
  - Login to the cluster
  - Verify user permissions
- User directories
- Transferring files
  - Globus
- Software
  - Modules
  - Acquiring software
  - Compilers
- Managing user jobs
  - Job scheduler
  - Using SLURM
  - Partitions
  - Submitting jobs (Examples)

Modules

Modules

A tool to help users manage their Unix or Linux shell environment, by allowing groups of related environment-variable settings to be made or removed dynamically.[a]

[a] https://en.wikipedia.org/wiki/Environment_Modules_(software)

- We globally install frequently requested software packages and create modules for all users to access
- Access to modules is via the **module** command
  - 'module avail' – list installed modules
  - 'module show <module name>' – Show what actions a module performs
  - 'module load <module name>' – Loads the named module
  - 'module purge' – Unload all loaded modules
- Installing software in your ~/apps directory is suggested to prevent us from being a bottleneck

Acquiring Software – Binaries and/or Source

- You can transfer software source, binaries or scripts into your ~/apps directory on the Cray CS300
  - Binaries compiled as x86_64 (64-bit) for CentOS 6.5 or RHEL6.5 should work
- You may also download tar or zipped software/source code directly from the login nodes using tools like **wget** & **curl**
- You may also clone source repositories using the correct software revision tool: **git**, **svn**, **hg**, **cvs**, etc.

Compilers

- We have the Intel® & GNU (gcc, g++) compilers
- Compiling must take place on a compute node
  - Interactive sessions are useful for compiling software
  - Sandbox nodes mirror the environment the compute nodes provide and are ideal for compilation
  - Login nodes **do not** load all the software and libraries found on the compute nodes
- Intel® compilers are recommended for best performance
  - Intel® 2013 compilers:
    - module load intel/ics – Loads Intel® compilers: **icc**, **ifort**, **icpc**
    - module load intel/impi – Loads Intel® MPI wrapper: **mpiicc**, **mpiifort**, **mpiicpc**
  - Intel® 2016 compilers:
    - We have 2 floating seats for Intel® 2016 compiler
    - intel_2016/ics
    - intel_2016/impi

Overview – Cluster Interaction

- Connecting to a cluster @ UH
  - Login to the cluster
  - Verify user permissions
- User directories
- Transferring files
  - Globus
- Software
  - Modules
  - Acquiring software
  - Compilers
- Managing user jobs
  - Job scheduler
  - Using SLURM
  - Partitions
  - Submitting jobs (Examples)

Managing User Jobs

User jobs all come in different shapes and sizes:

- Require multiple nodes working in concert towards a common goal (MPI)
- Require a single node, in which they use multiple threads work together (OpenMP, pthreads)
- Require a lot of cores to process a lot of data in an identical manner, yet none of the inputs have dependencies on another (HTC)

The Cray CS300 is capable of handling many different types of jobs, but with so many users in a multi-user environment, how do we impose order on this chaos?

This looks like a job for a ***job scheduler***!

Job Schedulers

Purpose

To control and prioritize the execution order of unrelated jobs

Basic features expected of a job scheduler:

- Provides a user interface for users to request resources and monitor work
- Allocates access to resources for the requested duration of time
- Starts, monitors and terminates work on allocated resources
- Arbitrates contention for resources by managing queues of pending work

The Cray CS300 uses the **S**imple **L**inux **U**tility for **R**esource **M**anagement or simply known as the *SLURM scheduler*

---

https://en.wikipedia.org/wiki/Slurm_Workload_Manager

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

SLURM

How are jobs scheduled?

User submitted jobs are assigned a priority using a fairshare algorithm. Factors such as the following are all used to assign a priority to a given job:

- Runtime
- Amount of resources requested
- Age of job
- Amount of core hours a user has used in recent history

SLURM commands

SLURM has a series of commands, each of which allow users to interact with the job scheduler

- ***srun*** – Used to submit a job for execution or initiate job steps in real time
- ***sbatch*** – Used to submit a job script for later execution. The script could contain one or more srun commands
- ***squeue*** – Reports the state of jobs or job steps
- ***scancel*** – Used to cancel a pending or running job or job step. It can also be used to send an arbitrary signal to all processes associated with a running job or job step
- ***sinfo*** – Reports the state of partitions and nodes managed by Slurm. It has a wide variety of filtering, sorting, and formatting options
- ***sacct*** – Used to report job or job step accounting information about active or completed jobs
- ***scontrol*** – The administrative tool used to view and/or modify Slurm state. Note that many scontrol commands can only be executed as user root

– Examples usage of the SLURM commands can be seen on schedmd's quickstart

– Each command should have a 'man' page, or displays help when the -h flag is used

---

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

Partitions

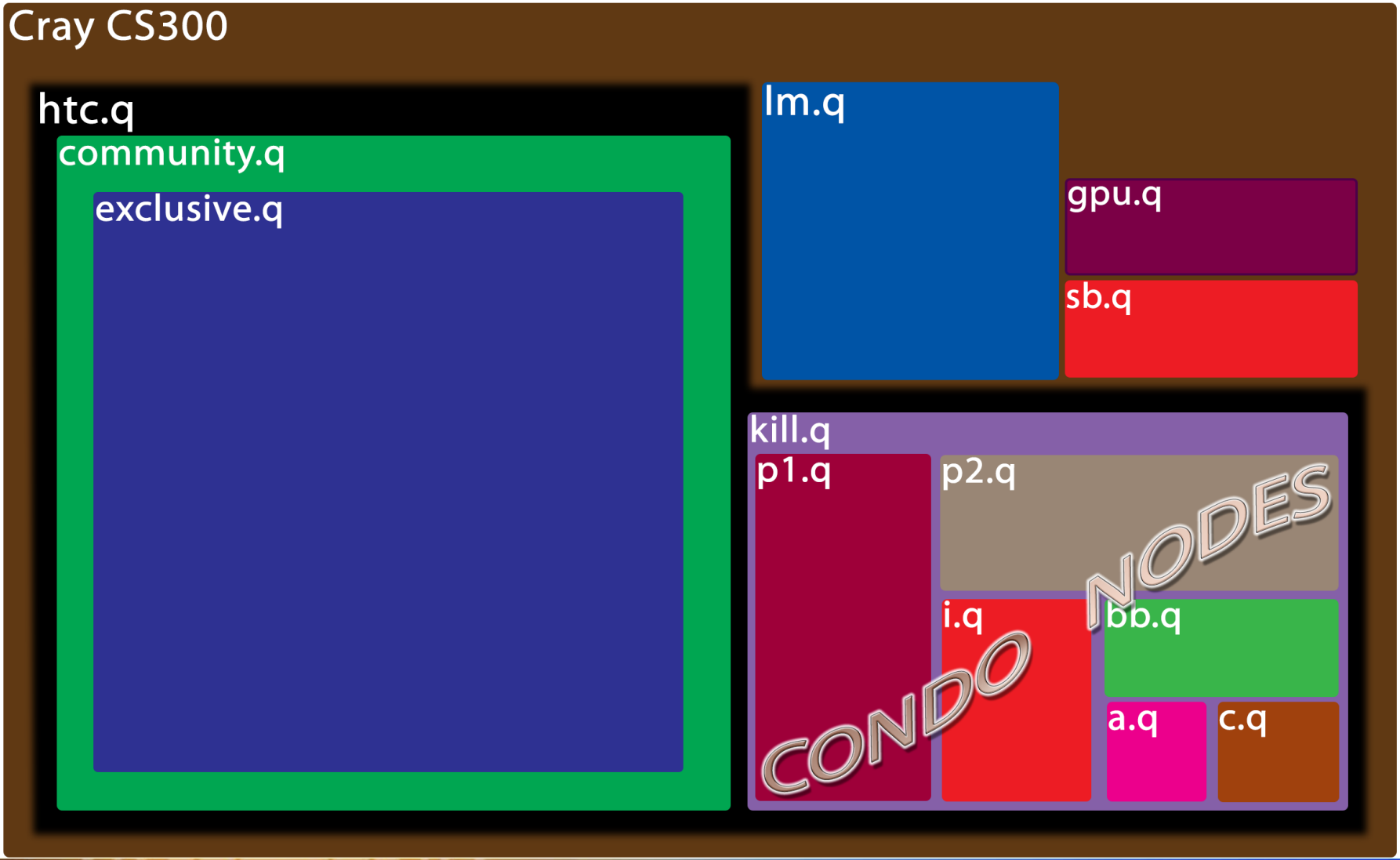
What is a partition?

A partition can be thought of as a group of nodes/resources divided into possibly overlapping sets. Each partition can be considered as a job queue, each of which has an assortment of constraints such as job size limit, job time limit, users permitted to use it, etc. Priority-ordered jobs are allocated nodes within a partition until the resources (nodes, processors, memory, etc.) within that partition are exhausted.[a]

[a] http://slurm.schedmd.com/quickstart.html

- The Cray CS300 currently has seven public partitions:
  community.q, **exclusive.q**, **lm.q**, **gpu.q**, **sb.q**, **kill.q**, **htc.q**
- Jobs submitted to kill.q and htc.q can be preempted by jobs in other partitions

Partitions


Cray CS300
htc.q
community.q
exclusive.q
lm.q
gpu.q
sb.q
kill.q
p1.q
p2.q
i.q
bb.q
a.q
c.q
CONDO
NODES

Partition Details

Partition	Time	Nodes per job	Priority	Shared	Preempt Mode	Memory per CPU (MB)
community.q	**Default:** 0-00:10:00 **Max:** 3-00:00:00	**Min:** 1 **Max:** 20	10	NO	OFF	**Default:** 3250 **Max:** ∞
exclusive.q	**Default:** 0-00:10:00 **Max:** 3-00:00:00	**Min:** 1 **Max:** 20	10	EXCLUSIVE	OFF	**Default:** ∞ **Max:** ∞
gpu.q	**Default:** 0-00:10:00 **Max:** 3-00:00:00	**Min:** 1 **Max:** 1	10	NO	OFF	**Default:** 3250 **Max:** ∞
kill.q	**Default:** 0-00:10:00 **Max:** 3-00:00:00	**Min:** 1 **Max:** 20	10	NO	REQUEUE	**Default:** 3250 **Max:** ∞
htc.q	**Default:** 0-00:10:00 **Max:** 3-00:00:00	**Min:** 1 **Max:** 1	1	NO	REQUEUE	**Default:** 3250 **Max:** ∞
lm.q	**Default:** 0-00:10:00 **Max:** 3-00:00:00	**Min:** 1 **Max:** 1	10	NO	OFF	**Default:** ∞ **Max:** ∞
sb.q	**Default:** 0-00:05:00 **Max:** 0-01:00:00	**Min:** 1 **Max:** 2	10	NO	OFF	**Default:** 3250 **Max:** ∞

Partition details also available on the Cyberinfrastructure website or by using the following command:

```
[login ~]$ scontrol show partition <partition name>
```

Interactive Job with SLURM

Interactive session

```
[login ~]$ srun --immediate --partition sb.q --nodes 1 --cpus-per-task 1 --tasks-per-node 1 --time 0-01:00:00 --pty /bin/bash
```

```
[login ~]$ srun -I -p sb.q -N 1 -c 1 -n 1 -t 0-01:00:00 --pty /bin/bash
```

Interactive sessions terminate when the specified time has elapsed or if you give the **exit** command

SLURM sbatch – Submission Script File – GPU

```
[login lus]$ cat gpu.slurm

#!/bin/bash
#SBATCH --job-name=GPU_example
#SBATCH --partition=gpu.q
#SBATCH --time=3-00:00:00 ## time format is DD-HH:MM:SS
## task-per-node x cpus-per-task should not typically exceed core count on an individual node
#SBATCH --nodes=1
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=10
#SBATCH --mem=11000 ## max amount of memory per node you require
#SBATCH --gres=gpu:NV-K40:2  ## request both GPUs in the GPU node
### To request only 1 of the two GPUs in the node, you would do: gpu:NV-K40:1
#SBATCH --error=hello-%A.err ## %A - filled with jobid
#SBATCH --output=hello-%A.out ## %A - filled with jobid
## Useful for remote notification
#SBATCH --mail-type=BEGIN,END,FAIL,REQUEUE,TIME_LIMIT_80
#SBATCH --mail-user=user@test.org

source ~/.bash_profile #if you want to use modules or need environment variables, source your bash profile
module load GPGPU/cuda/samples/7.5

## All options and environment variables found on schedMD site: http://slurm.schedmd.com/sbatch.html
export OMP_NUM_THREADS=${SLURM_CPUS_PER_TASK}

bandwidthTest; cudaOpenMP; deviceQuery; simpleMultiGPU
```

SLURM sbatch – Submission Script File (MPI Job)

```
[login lus]$ cat mpi.slurm

#!/bin/bash
#SBATCH --job-name=MPI_example
#SBATCH --partition=exclusive.q
## 3 day max run time for community.q, kill.q, exclusive.q, and htc.q.  1 Hour max run time for sb.q
#SBATCH --time=3-00:00:00 ## time format is DD-HH:MM:SS
## task-per-node x cpus-per-task should not typically exceed core count on an individual node
#SBATCH --nodes=4
#SBATCH --tasks-per-node=20
#SBATCH --cpus-per-task=1
##SBATCH --mem=11000 # Memory should not be set for jobs in exclusive.q
#SBATCH --error=hello-%A.err ## %A - filled with jobid
#SBATCH --output=hello-%A.out ## %A - filled with jobid
## Useful for remote notification
#SBATCH --mail-type=BEGIN,END,FAIL,REQUEUE,TIME_LIMIT_80
#SBATCH --mail-user=user@test.org

source ~/.bash_profile #if you want to use modules or need environment variables, source your bash profile

## All options and environment variables found on schedMD site: http://slurm.schedmd.com/sbatch.html
## Intel MPI manual: https://software.intel.com/en-us/mpi-refman-lin-html
export OMP_NUM_THREADS=${SLURM_CPUS_PER_TASK}
export I_MPI_FABRICS=shm:tmi
export I_MPI_PMI_LIBRARY=/opt/local/slurm/default/lib64/libpmi.so

srun  -n ${SLURM_NTASKS}  ./hello_mpi.intel
```

SLURM sbatch – Submission Script File (Non-MPI Job)

```
[login lus]$ cat hello_world.slurm

#!/bin/bash
#SBATCH --job-name=example
#SBATCH --partition=community.q
## 3 day max run time for community.q, kill.q, exclusive.q, and htc.q.  1 Hour max run time for sb.q
#SBATCH --time=3-00:00:00 ## time format is DD-HH:MM:SS
## task-per-node x cpus-per-task should not typically exceed core count on an individual node
#SBATCH --nodes=1
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=5
#SBATCH --mem=11000 ## max amount of memory per node you require
#SBATCH --error=hello-%A.err ## %A - filled with jobid
#SBATCH --output=hello-%A.out ## %A - filled with jobid
## Useful for remote notification
#SBATCH --mail-type=BEGIN,END,FAIL,REQUEUE,TIME_LIMIT_80
#SBATCH --mail-user=user@test.org

source ~/.bash_profile #if you want to use modules or need environment variables, source your bash profile

## All options and environment variables found on schedMD site: http://slurm.schedmd.com/sbatch.html
export OMP_NUM_THREADS=${SLURM_CPUS_PER_TASK}

./hello_world
```

SLURM sbatch – Executing & Monitoring Jobs

To execute a submission script you use the **sbatch** command

Example

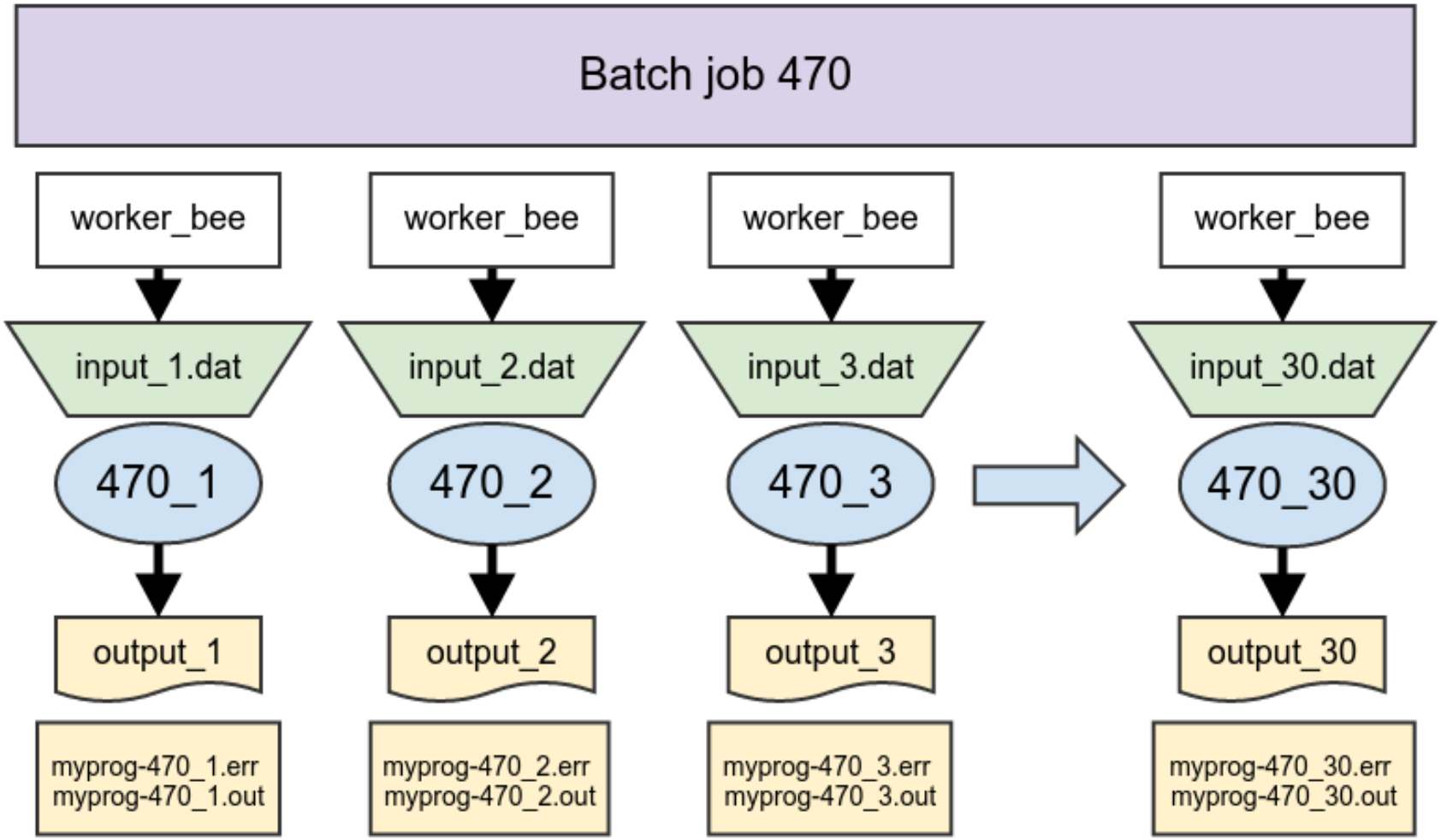
```
[login lus]$ sinfo
PARTITION     AVAIL  TIMELIMIT  NODES  STATE NODELIST
community.q   up     3-00:00:00 2      idle  compute-[0001-0002]

[login lus]$ sbatch hello_world.slurm
sbatch: Submitted batch job 469

[login lus]$ squeue
JOBID PARTITION    NAME     USER    ST TIME  NODES  NODELIST(REASON)
469   community.q  example  user99  R  00:01 1      compute-0001
```

What if I want to run many jobs with identical parameters, but on different inputs? SLURM provides a mechanism for this called a **Job Array**.

Slurm Job Array



SLURM sbatch – Submission Script File (Job Array)

```
[login lus]$ cat job_array.slurm

#!/bin/bash
#SBATCH --job-name=example
#SBATCH --partition=community.q
## 3 day max run time for community.q, kill.q, exclusive.q, and htc.q.  1 Hour max run time for sb.q
#SBATCH --time=3-00:00:00 ## time format is DD-HH:MM:SS
## task-per-node x cpus-per-task should not typically exceed core count on an individual node
#SBATCH --nodes=1
#SBATCH --tasks-per-node=1
#SBATCH --cpus-per-task=5
#SBATCH --mem=11000 ## max amount of memory per node you require
#SBATCH --error=myprog-%A_%a.err ## %A - filled with jobid. %a - filled with job arrayid
#SBATCH --output=myprog-%A_%a.out ## %A - filled with jobid. %a - filled with job arrayid
## Useful for remote notification
#SBATCH --mail-type=BEGIN,END,FAIL,REQUEUE,TIME_LIMIT_80
#SBATCH --mail-user=user@test.org

source ~/.bash_profile #if you want to use modules or need environment variables, source your bash profile

## All options and environment variables found on schedMD site: http://slurm.schedmd.com/sbatch.html

export OMP_NUM_THREADS=${SLURM_CPUS_PER_TASK}

./worker_bee -i input_${SLURM_ARRAY_TASK_ID}.dat -o output_${SLURM_ARRAY_TASK_ID}
```

SLURM sbatch – Executing & Monitoring Job Arrays

Example

```
[login lus]$ sinfo
PARTITION     AVAIL  TIMELIMIT  NODES  STATE NODELIST
community.q   up     3-00:00:00 2      idle  compute-[0001-0002]

[login lus]$ sbatch --array=1-30 job_array.slurm
sbatch: Submitted batch job 470

[login lus]$ squeue
JOBID      PARTITION    NAME     USER    ST   TIME    NODES  NODELIST(REASON)
470[3-30]  community.q  example  user99  PD   00:00   1      (resource)
470_1      community.q  example  user99  R    00:01   1      compute-0001
470_2      community.q  example  user99  R    00:05   1      compute-0002
```

Outline

Problem Reporting

- Always email UH-HPC-Help@lists.hawaii.edu
- For batch jobs . . .
  - Job ID, path to submission script, submission command, error file location, output files
- For other problems . . .
  - State the problem, command issued, host, directory, remote host, error messages

Outline

Cluster Etiquette

- Never run anything on the login nodes
- Try to request only the resources that your job needs. Leaving unneeded resources available, allows other users to use them
- Test your application in an interactive session before scheduling a long running job
  - This can help identify what resources your job will require, and potentially aid you in correctly sizing what resources you request.
- Use the nodes in the sb.q to validate your slurm script runs as intended
  - This helps to reduce the chance of you having a small error that kills your job prematurely after waiting for resources
- Always run jobs from within ~/lus/ directory to ensure output is written there and does not fill up the system filesystem

Outline

Login Node Usage Policy

The login nodes are a shared resource and are the only access to the cluster for hundreds of user and are meant to provide the following functionality for all users:

- Providing ssh shell access
- Facilitate the transfer files to and from the cluster:
  Globus, sftp, scp, rsync
- Launching and monitoring SLURM jobs (batch and interactive)
- Modifying text files with a text editor: vi/vim, emacs, nano

- If we identify or are notified that a user is running computation on the login nodes, the application will be killed
- If we determine that a user repeatedly violates the login node policy, even after being warned, the repeat offender can have their cluster account disabled

Login node usage policy is subject to change
Users will be notified via email prior to changes taking effect

Lustre® Filesystem Purge Policies

Due to users not having any quotas on the Lustre® filesystem, we need some policy in place which removes older files from the system. To accomplish this, we have currently implemented a purge policy, for any file that is older than 90 days.

- Which of my directories are subject to the purge policy?
  - **Answer:** All files and folders found in ~/lus/ are subject to the 90 day purge policy. Symlinks are excluded from the purge, and are not followed by the purge bot.
- How frequently are files for purging identified?
  - **Answer:** Taking into account the fill rate of the Lustre® filesystem, a purge is performend as usage approaches 80%.

Purge policy is subject to change
Users will be notified via email prior to changes taking effect

Questions?

FAQ

- Are files on the cluster backed up?
  - **Answer: NO!** User files on the cluster ***are not backed up***. It is up to you as the user to validate and maintain your own backups. The CI–Team takes no responsibility for any data that is lost due to human or mechanical error.
- Help I have been trying to compile an application without success, can the CI–Team help?
  - **Answer:** We are willing to help, just send us an email at *uh-hpc-help@lists.hawaii.edu* with a description of what you have attempted as well as a link to the software you are trying to compile, and/or where it is located on the cluster.
- HELP! My job needs more time and does not support check-pointing. What can I do?
  - **Answer:** We understand that jobs may not all fit within the 3 day timelimit or unforeseen issues during the execution can cause a job to run long. The CI–Team does review and potentially approves requests to extend jobs in a one off fashion. A request to extend a job can be denied, since we need to take into account factor such as: how busy the cluster is, the number of time extension requests a user has made in recent history.
    If it is known before the job is submitted that more than 3 days is required, we would ask that you contact us first so we can verify that you truly will need more time.
    If a job is already running and just doesn't look like it will complete in time, please send us a request with the jobid number and what you want the job extended to e.g., please extend my jobs run time to 7 days.
    Please remember the CI–Team is small and we are also human. Depending on when you make your request we may not be able to immediately respond or act. It is best to make your request at the first signs you need more time, in hopes that you can provide us with ample time to act.

FAQ

- Can I run a display program on the cluster?
  - **Answer:** Yes, but not from the login nodes. To follow cluster usage policies, please run X11 applications from a compute node. Please see the following steps.

Interactive session with X11

1. Connect via SSH using the -Y option, X11 forwarding enabled
2. run srun.x11 to start a session on a node

```
[local ~]$ ssh -Y user99@uhhpc1.its.hawaii.edu
[login ~]$ srun.x11 --partition sb.q --nodes 1 --cpus-per-task 1 --tasks-per-node 1 --time 0-01:00:00
[compute-0001 ~]$ xterm
```