# GACRC Sapelo2 Cluster New User Training

*Georgia Advanced Computing Resource Center (GACRC)*
*Enterprise Information Technology Services (EITS)*


UNIVERSITY OF GEORGIA
Georgia Advanced Computing Resource Center

# Agenda

- GACRC
- Sapelo2 Cluster
    - Cluster Diagram and Overview
    - Directories
    - Computational Partitions
    - Software Environment
- Job Submission Workflow
- Useful Commands: sq --me, sacct-gacrc -X
- GACRC wiki and User Support
- Appendices

# GACRC

- A high-performance-computing (HPC) center at UGA
- Advanced computing environment for the UGA research and education community
  - HPC infrastructure location in the Boyd Data Center
  - Comprehensive collection of scientific, engineering and business applications
  - Consulting and training services

- GACRC Wiki: http://wiki.gacrc.uga.edu
- Help and Support: http://help.gacrc.uga.edu
- Main website: http://gacrc.uga.edu
- Kaltura Channel (tutorial videos): https://kaltura.uga.edu/channel/GACRC/176125031

# Running Computations on Sapelo2

- You can run your computations/programs/scripts in **jobs** on the Sapelo2 Cluster
- There are a few different types of **jobs** on the cluster depending on how you want to run your computation
- A few we will talk about today are:

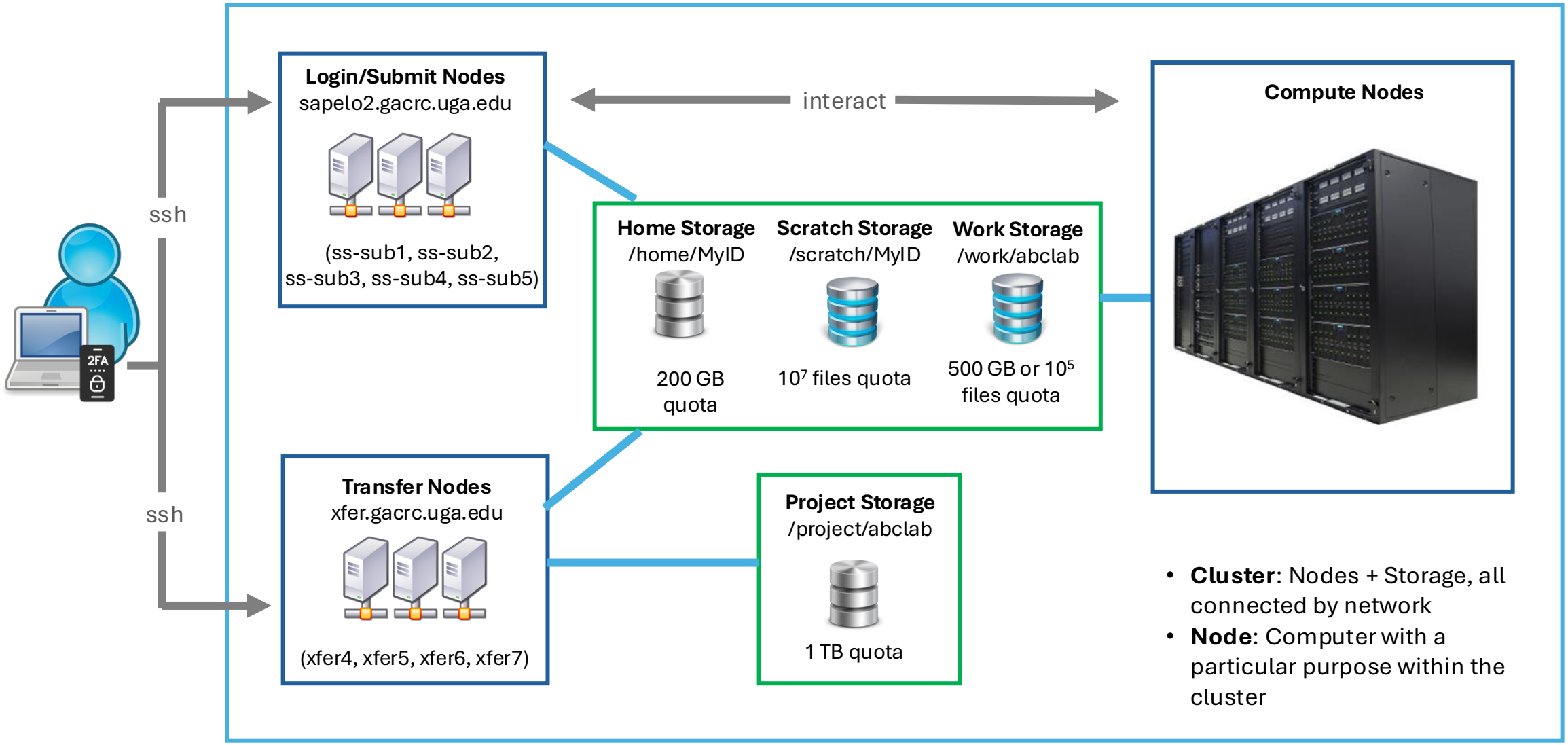
Submission script jobs (batch jobs)

Interactive jobs

OnDemand jobs

- The **job** queueing system is called **Slurm**

# Sapelo2 Cluster




- **Cluster**: Nodes + Storage, all connected by network
- **Node**: Computer with a particular purpose within the cluster

# Storage Spaces (Directories)

| Directory | Name | Quota | Accessible from | Intended Use | Backed-up | Important Notes |
|---|---|---|---|---|---|---|
| /home/MyID | Home | 200GB | **Login**<br>**Transfer**<br>**Compute** | Static data, e.g.<br>1. Scripts, source codes<br>2. Local software | Yes | Not for storing data of your jobs! |
| /scratch/MyID | Scratch | $10^7$ files | **Login**<br>**Transfer**<br>**Compute** | Temporary files needed for currently running jobs | No | Clean up when your job finishes!<br>Subject to "30-day purge" policy |
| /work/abclab | Work | 500GB<br>$10^5$ files | **Login**<br>**Transfer**<br>**Compute** | Input files needed for repeated jobs | No | Clean up when your job finishes!<br>Group sharing is possible |
| /project/abclab | Project | 1TB (initial) | **Transfer** | Temporary data parking | Yes | Group sharing is possible |
| /lscratch | Local Scratch | 3.5 TB | **Compute** | Jobs with heavy disk I/O operations | No | Clean up when job exits from node! |

# Scratch 30-Day Purge Policy

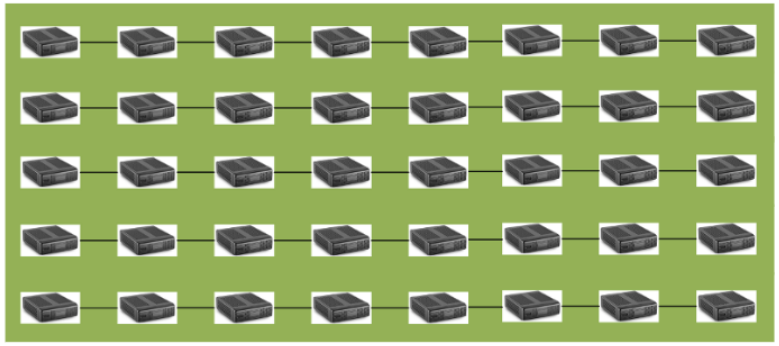
- https://wiki.gacrc.uga.edu/wiki/Disk_Storage#Scratch_file_system

- Any file that is not accessed or modified by a compute job within **30 days** will be automatically deleted off the /scratch file system.
- Measures circumventing this policy will be monitored and actively discouraged.

- Copy files in /scratch that you want to keep to /project when the job is done
  - Can use Globus or fpsync on xfer nodes to do this quickly
- Move all unnecessary files and folders to /scratch/trash/$USER
- When you archive data using tar on /scratch, please do not use z option (compression option).
  - After you archive data with tar, you can use gzip to compress it.

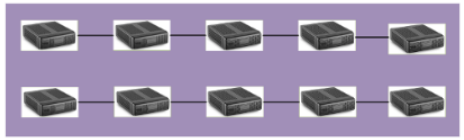
# Computational Partitions

**Compute Nodes**



**batch** for up to 7 days
**batch_30d** for up to 30 days

→ For regular jobs up to 740GB per node

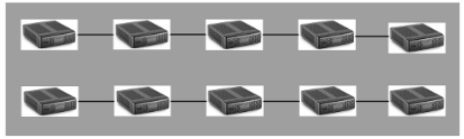


**highmem_p** for up to 7 days
**highmem_30d_p** for up to 30 days

→ For high memory jobs up to 990GB per node

**hugemem_p** for up to 7 days
**hugemem_30d_p** for up to 30 days

→ For huge memory jobs up to 3000GB per node



**gpu_p** for up to 7 days
**gpu_30d_p** for up to 30 days

→ For GPU enabled jobs

**inter_p** for interactive jobs

→ For interactive jobs

Compute nodes are divided into groups called **partitions**. A **partition** is a collection of compute nodes for a particular computing need.

# Computational Partitions

*subject to change

https://wiki.gacrc.uga.edu/wiki/Job_Submission_partitions_on_Sapelo2

| Type | Partition | Time limit | Max jobs Running* | Max jobs Submit* | Notes |
|---|---|---|---|---|---|
| Regular | batch | 7 days | 250 | 10,000 | Regular nodes |
| | highmem_p | | 6 | 100 | For running high memory jobs |
| | hugemem_p | | 4 | 4 | For running huge memory jobs |
| | gpu_p | | 8 | 20 | For running GPU-enabled jobs |
| Longer | batch_30d | 30 days | 1 | 2 | 30-day partition counterparts |
| | highmem_30d_p | | | | |
| | hugemem_30d_p | | 4 | 4 | |
| | gpu_30d_p | | 2 | 2 | |
| Interactive | inter_p | 2 days | 3 | 20 | Regular nodes, for interactive jobs. |
| Buy-in | name_p | variable | | | Partitions that target different groups' buy-in nodes. The name string is specific to each group. |

# Computational Partitions - continued

| Partition | Total Nodes | Max Mem(GB) /Single-node job | Cores /Node | Processor Type | GPU Cards /Node |
|---|---|---|---|---|---|
| batch<br>batch_30d | 1 | 740 | 384 | AMD EPYC Turin | N/A |
| | 64 | | 128 | AMD EPYC Genoa | |
| | 120 | 500 | | AMD EPYC Milan | |
| | 4 | 250 | 64 | AMD EPYC Milan | |
| | 123 | 120 | 64 | AMD EPYC Rome | |
| | 25 | | 32 | AMD EPYC Naples | |
| | 40 | 180 | 32 | Intel Xeon Skylake | |
| highmem_p<br>highmem_30d_p | 10 | 500 | 32 | AMD EPYC Naples | N/A |
| | 12 | 990 | 32 | AMD EPYC Milan | |
| | 2 | | 128 | | |
| hugemem_p,<br>hugemem_30d_p | 3 | 3000 | 48 | AMD EPYC Genoa | N/A |
| | 2 | 2000 | 32 | AMD EPYC Rome | |
| gpu_p<br>gpu_30d_p | 14 | 1000 | 64 | AMD EPYC Milan | 4 NVIDIA A100 |
| | 12 | | | Intel Xeon SapphireRapids | 4 NVIDIA H100 |
| | 12 | 740 | 128 | AMD EPYC Genoa | 4 NVIDIA L4 |

# Software Environment

- Software is installed as “software modules”
- Naming format of software modules: **Name/Version-Toolchain**
  - e.g., **Python/3.12.3-GCCcore-13.3.0**
- Module commands:
  - ml spider *pattern* : Search module names matching a *pattern*
  - ml *ModuleName* : Load a module into your submission script
  - ml : List modules currently loaded
  - ml -*ModuleName* : Remove a module
  - ml purge : Remove all currently loaded modules

**NOTE: DO NOT LOAD/USE MODULES ON THE LOGIN/SUBMIT NODES! (ss-sub1, ss-sub2, ss-sub3, etc…)**

# Software Environment - toolchains

- When you load more than one software module, you must pay attention to their toolchain compatibility
- Having modules loaded with incompatible toolchains can cause errors in your job, such as:

```
Lmod has detected the following error:
These module(s) exist but cannot be loaded as requested
```

- The toolchain compatibility table can be found at https://wiki.gacrc.uga.edu/wiki/Available_Toolchains_and_Toolchain_Compatibility

# Job Submission Workflow

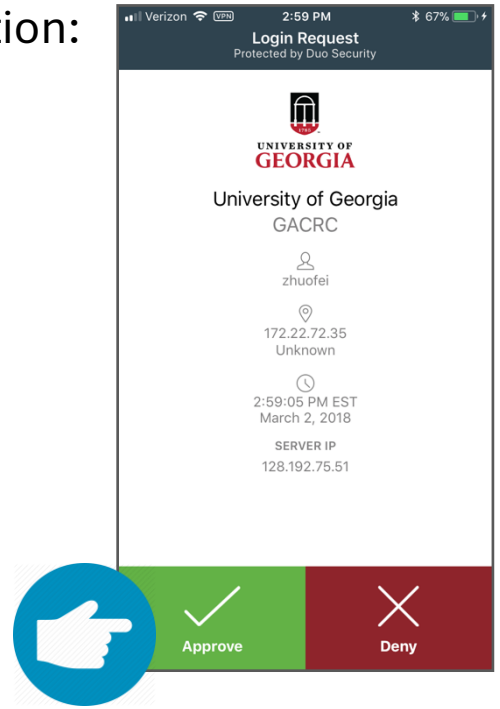
1. Log on to Login node using MyID and password, and two-factor authentication with Archpass Duo:
   **ssh MyID@sapelo2.gacrc.uga.edu**
2. On Login node, change directory to your scratch space: **cd /scratch/MyID**
3. Create a working subdirectory for a job : **mkdir training**
4. Change directory to training : **cd training**
5. Transfer data from local computer to training : use Globus to transfer data to the cluster
   Transfer data on cluster to training : use Globus or log on to Transfer node and then use cp/mv
6. Make a job submission script in training : **nano sub.sh**
7. Submit a job from training : **sbatch sub.sh**
8. Check job status : **sq --me** and **sacct-gacrc -X** or Cancel a job : **scancel jobID**

# Step 1: Login to login/submit node – Mac users - 1

1. Open Terminal utility
2. Type in the command line: ssh MyID@sapelo2.gacrc.uga.edu
3. You will be prompted for your MyID password
4. Sapelo2 access requires ID verification using two-factor authentication with Archpass Duo. If you are not enrolled in Archpass Duo, please refer to https://uga.teamdynamix.com/TDClient/3190/eitsclientportal/KB/Article/154538/ArchPass-powered-by-Duo on how to enroll

# Step 1: Login to login/submit node – Mac users - 2

Duo authentication:




Steps on the terminal utility on Mac:

```
[zhuofei@localhost ~]$
[zhuofei@localhost ~]$ ssh zhuofei@sapelo2.gacrc.uga.edu   ← Log on
Password:                                   ← Input MyID password!
......
Enter a passcode or select one of the following options:
 1. Duo Push to XXX-XXX-5758    ← Select Duo authentication option!

Passcode or option (1-5): 1
Success. Logging you in...
Last login: Tue Sep 15 11:22:42 2020 from 128.192.75.65

zhuofei@ss-sub1 ~$          ← I am on login node ss-sub1!
```

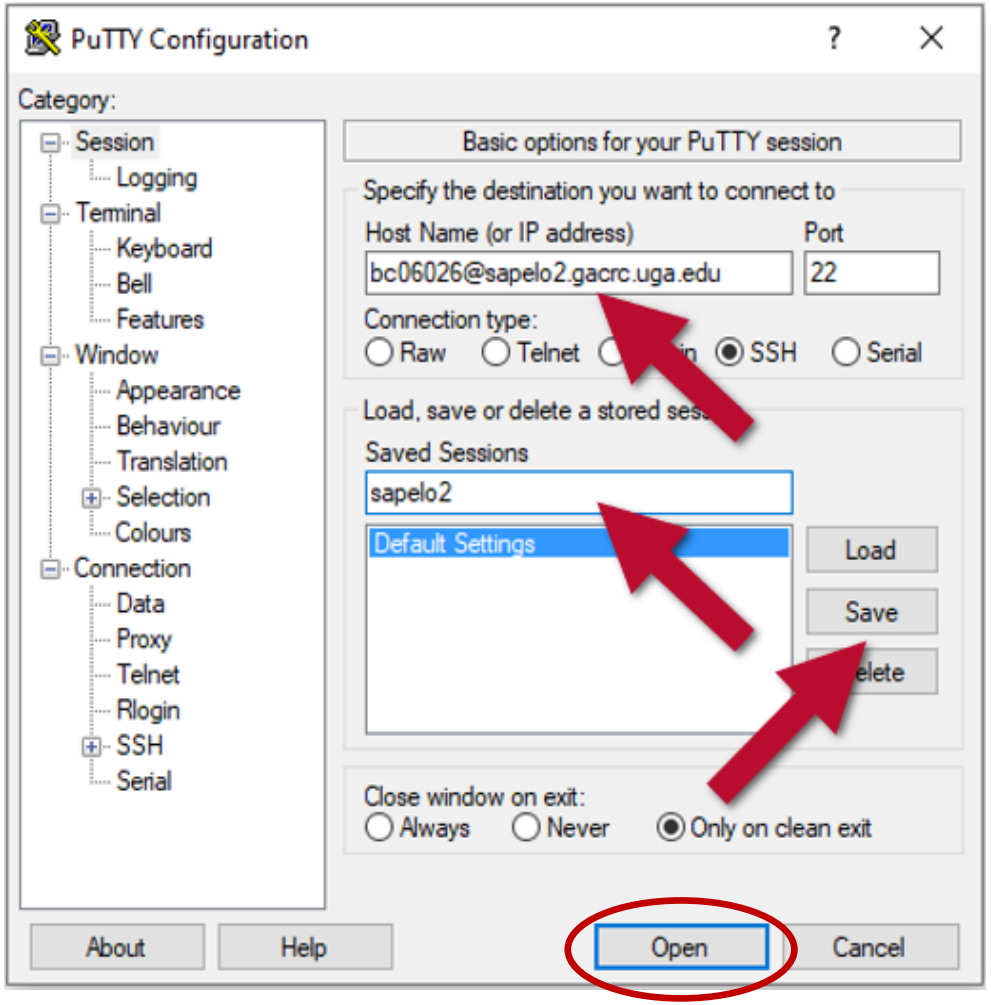
# Step 1: Login to login/submit node – Windows users - 1

1. Download and install PuTTY: https://www.chiark.greenend.org.uk/~sgtatham/putty/
2. Detailed downloading and installation instructions:

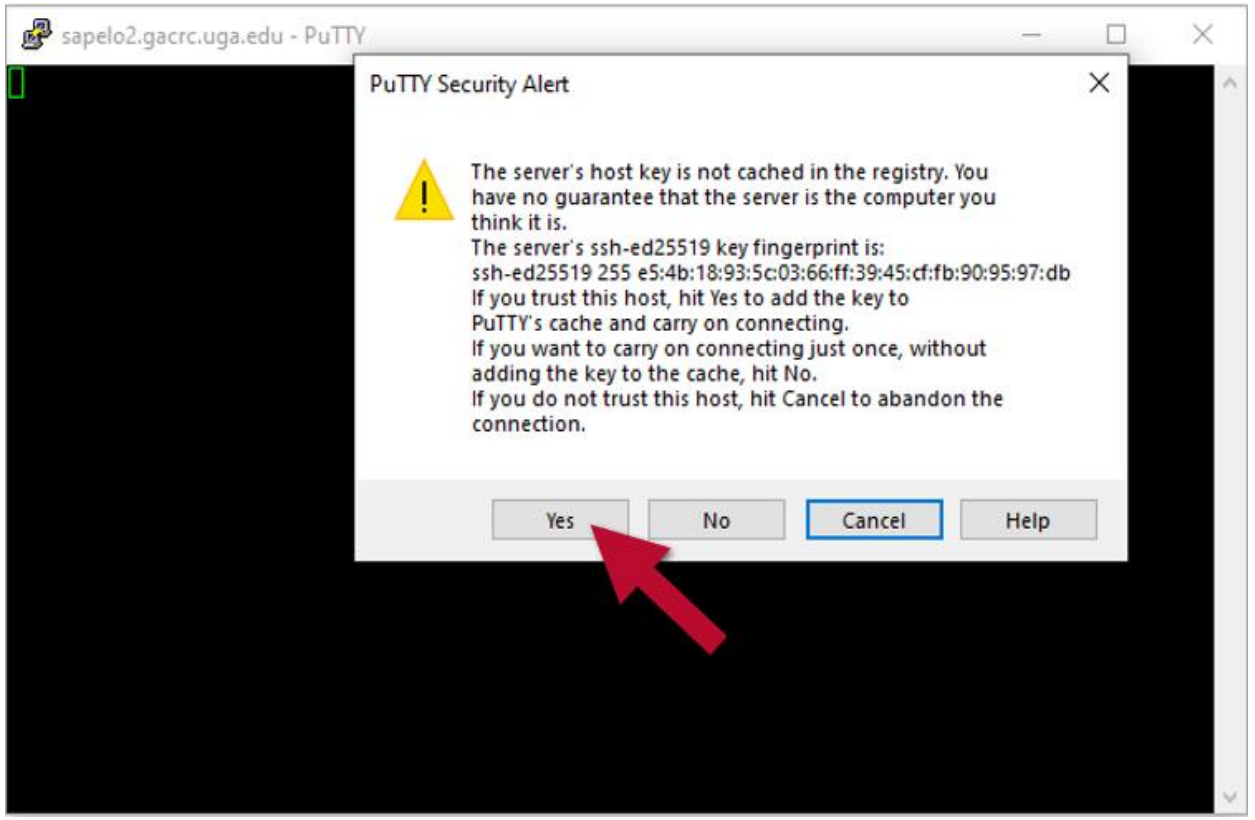
   https://wiki.gacrc.uga.edu/wiki/How_to_Install_and_Configure_PuTTY
3. Detailed configuring and usage instructions:

   https://wiki.gacrc.uga.edu/wiki/How_to_Install_and_Configure_PuTTY#Configuring_PuTTY

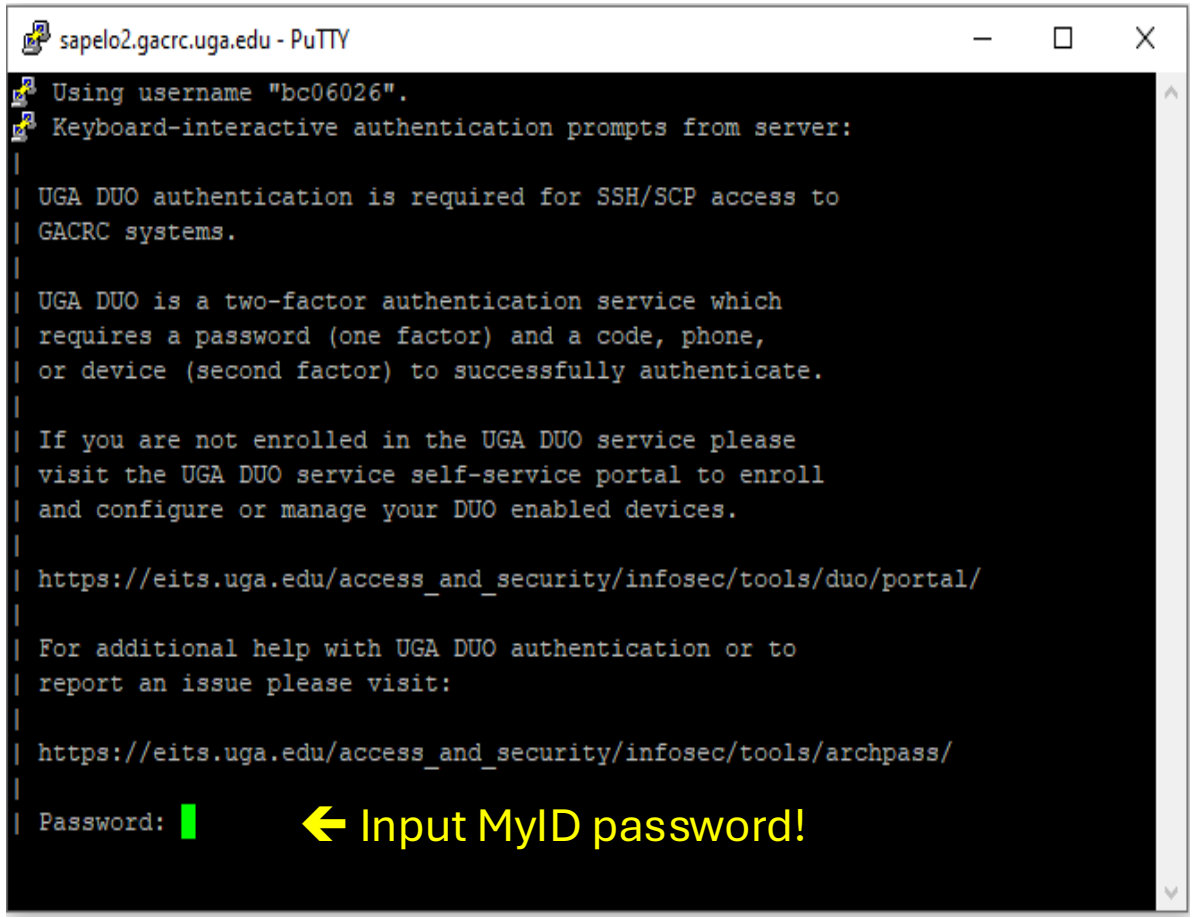
# Step 1: Login to login/submit node – Windows users - 2

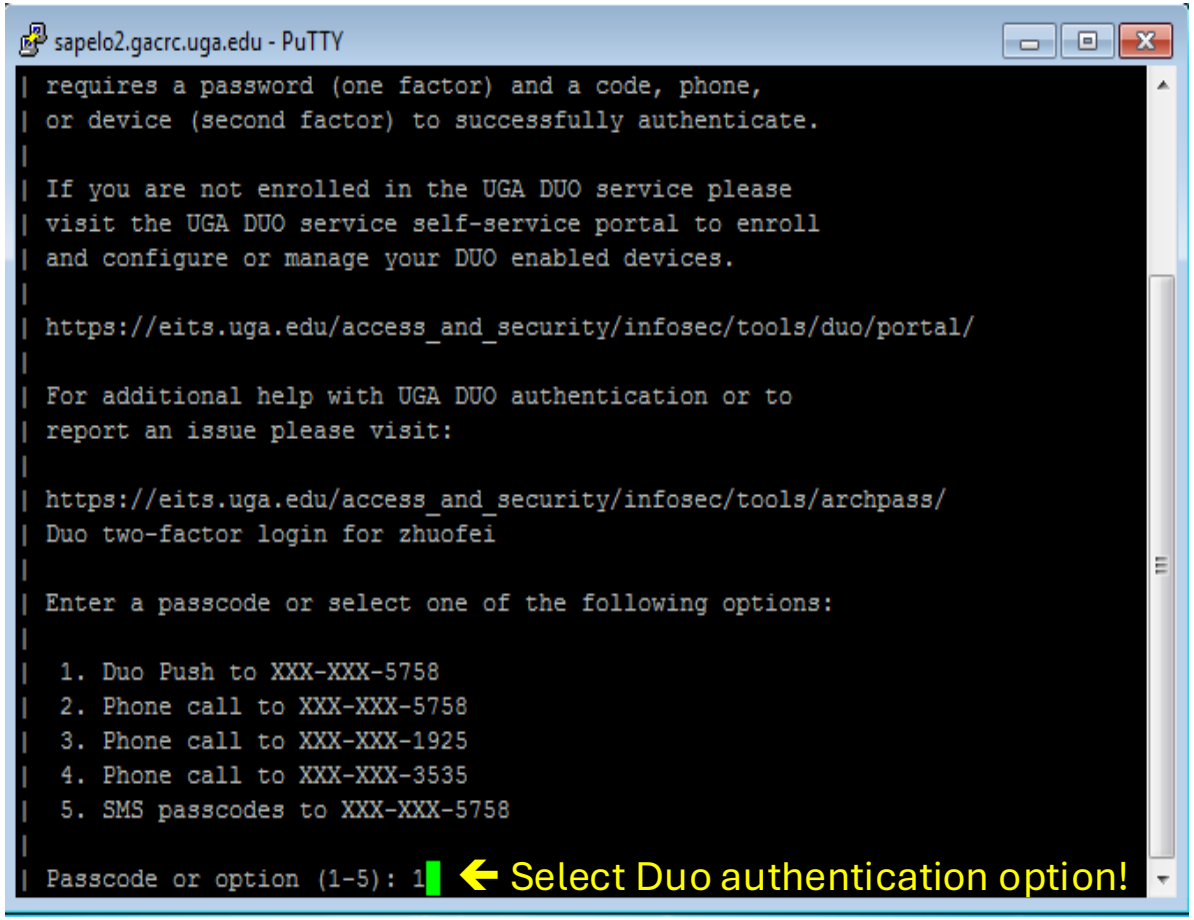



The first time you connect to a login node, PuTTY will give you this security alert window. Please click "Yes" or "Accept"



# Step 1: Login to login/submit node – Windows users - 3



# Step 2: Change Directory to “scratch” dir

- Once logged on, your current location will be your home directory

```
zhuofei@ss-sub1 ~$ pwd
/home/zhuofei                    ← this is my home directory!
```

- Use cd command to change your current directory to /scratch/MyID

```
zhuofei@ss-sub1 ~$ cd /scratch/zhuofei/
zhuofei@ss-sub1 zhuofei$ pwd
/scratch/zhuofei                 ← this is my scratch space!
```

- Use ls command to take a look in /scratch/MyID

```
zhuofei@ss-sub1 zhuofei$ ls
apps  input_1.txt
```

# Step 3: Create a subdirectory

- Use the mkdir command to create a new directory within /scratch/MyID

```
zhuofei@ss-sub1 zhuofei$ mkdir training
zhuofei@ss-sub1 zhuofei$ ls
apps  input_1.txt  training
```

# Step 4: Change directories to training dir

- Use the cd command to change your directory from /scratch/MyID to /scratch/MyID/training
- Can use a relative path or an absolute path

- Using a relative path (relative to where we currently are which is /scratch/MyID):

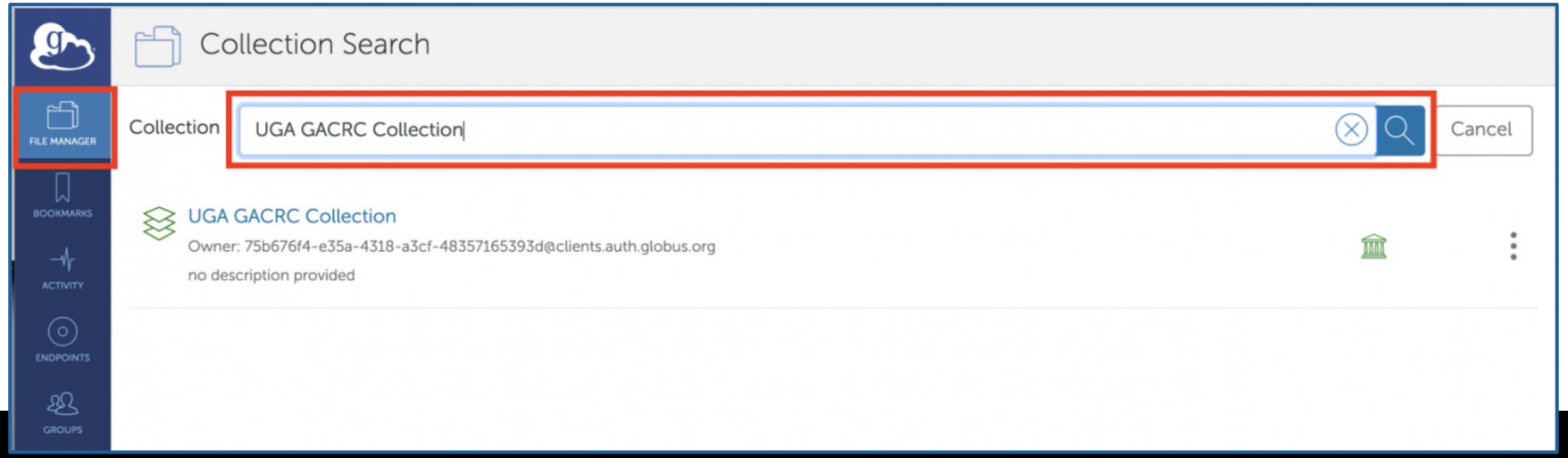
```
cft07037@ss-sub1 cft07037$ cd training
cft07037@ss-sub1 cft07037$ pwd
/scratch/cft07037/training
```

- Using an absolute path (works no matter where you currently are):

```
cft07037@ss-sub1 cft07037$ cd /scratch/cft07037/training
cft07037@ss-sub1 cft07037$ pwd
/scratch/cft07037/training
```

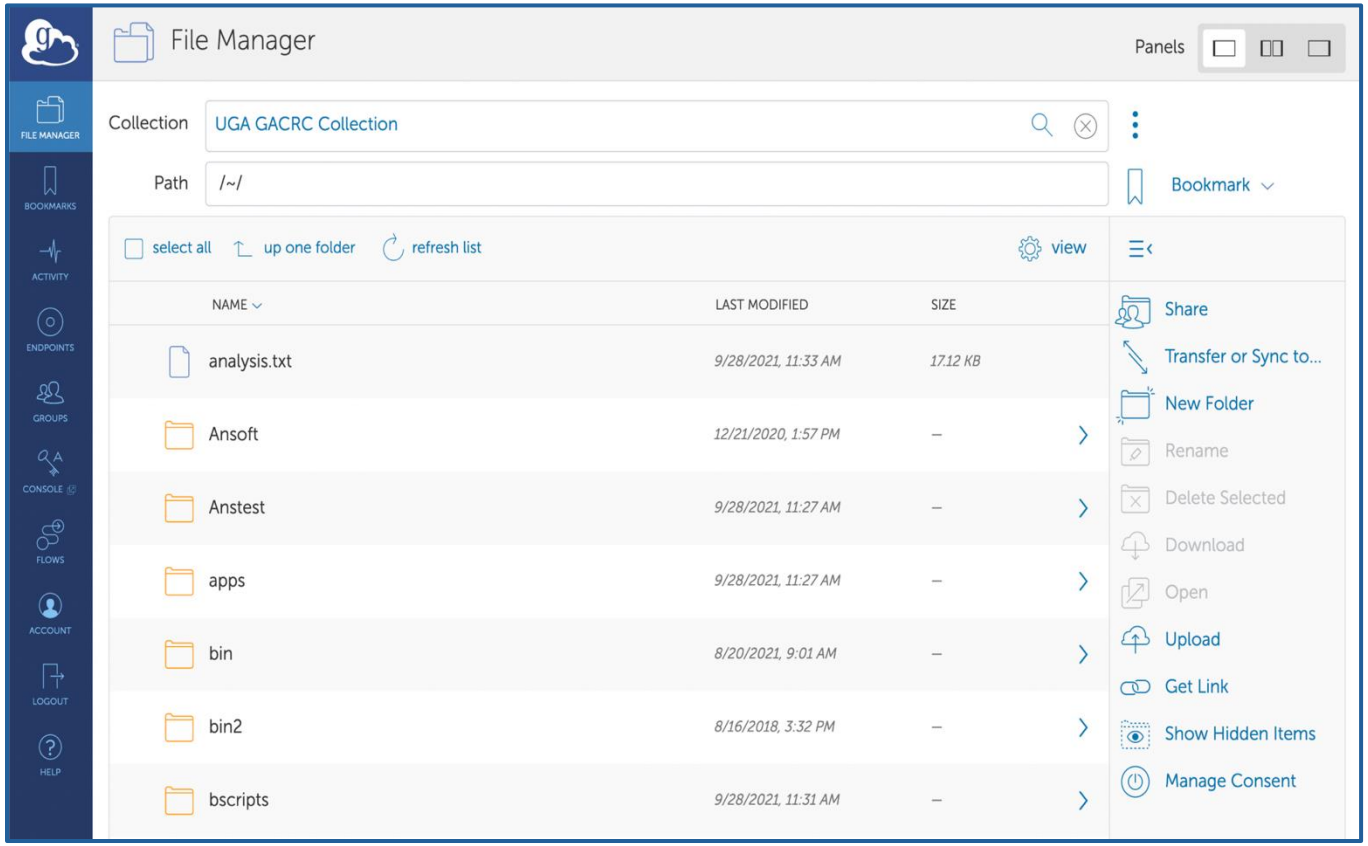
# Step 5: Transfer data to the Cluster - 1

- Globus is a website you can use to transfer files between your local computer and the Cluster and is free for UGA members
- Create a Globus Identity Account at globus.org following the instructions at https://wiki.gacrc.uga.edu/wiki/Globus#Getting_Started
- From the File Manager tab at globus.org, search for the “UGA GACRC Collection”



# Step 5: Transfer data to the Cluster - 2

- When you open the UGA GACRC Collection in Globus, by default you will be in your home directory. You can change to any of your or your lab's other directories by typing a path in the Path bar.
- Use the **Upload/Download** options on the right for transferring **small** or a **few** files.
- Use the **Transfer or Sync** option for **many** or **large** files (requires Globus Connect Personal on your computer: https://wiki.gacrc.uga.edu/wiki/Globus_Connect_Personal)

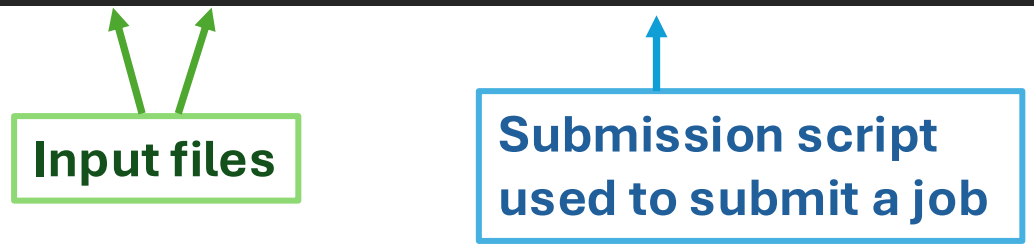


# Step 6: Make a job submission script - 1

- Copy over input files and submission script

```
$ cp -r /usr/local/training/Bowtie2/* .
```

```
cft07037@ss-sub1 training$ pwd
/scratch/cft07037/training
cft07037@ss-sub1 training$ cp -r /usr/local/training/Bowtie2/* .
cft07037@ss-sub1 training$ ls
index  myreads.fq  sub.sh
```



# Step 6: Make a job submission script - 2

- ”nano” is a simple text editor in Linux. You are welcome to use others like “vim” or “emacs”
- Open the sub.sh file with a Linux text editor: `$ nano sub.sh`

```
 GNU nano 2.9.8                                                    sub.sh

#!/bin/bash
#SBATCH --job-name=testBowtie2          # Job name (testBowtie2)
#SBATCH --partition=batch               # Partition name (batch, highmem_p, or gpu_p)
#SBATCH --nodes=1                       # Number of compute nodes for resources to be spread out over (increase only if using MPI enabled software)
#SBATCH --ntasks=1                      # 1 task (process) for below commands
#SBATCH --cpus-per-task=1               # CPU core count per task, by default 1 CPU core per task
#SBATCH --mem=4G                        # Memory per node (4GB); by default using M as unit
#SBATCH --time=1:00:00                  # Time limit hrs:min:sec or days-hours:minutes:seconds
#SBATCH --output=%x_%j.out              # Standard output log, e.g., testBowtie2_12345.out
#SBATCH --mail-user=cft07037@uga.edu    # Where to send mail
#SBATCH --mail-type=END,FAIL            # Mail events (BEGIN, END, FAIL, ALL)

ml Bowtie2/2.5.2-GCC-11.3.0             # Load software module and run bowtie2 below

bowtie2 -x index/lambda_virus -U myreads.fq
```

# Step 6: Make a job submission script - 3

- Submission scripts do not have to be complex!

This script works the same way as sub.sh, but with only required Slurm headers specified:

```
#!/bin/bash

#SBATCH --partition=batch
#SBATCH --ntasks=1
#SBATCH --mem=4G
#SBATCH --time=1:00:00

ml Bowtie2/2.4.1-GCC-8.3.0

bowtie2 -x index/lambda_virus -U myreads.fq
```

The only required Slurm headers are:

- --partition
- --ntasks
- --mem
- --time

(Slurm will let you know if you forgot one when you try to submit your submission script)

Some default values if not specified:

- --cpus-per-task=1
- --nodes=1 (this may not default to 1 if you have requested more than one process with ntasks)

# Step 7: Submit your job

- Submission scripts can be submitted to the Cluster with the command sbatch
- Be sure to submit jobs from within your /scratch/MyID directory

```
cft07037@ss-sub1 training$ pwd ← check current location
/scratch/cft07037/training
cft07037@ss-sub1 training$ ls ← view files in current location
index  myreads.fq  sub.sh
cft07037@ss-sub1 training$ sbatch sub.sh ← submit job
Submitted batch job 32860
```

**sub.sh** is a job submission script to

1. specify computing resources:
2. load software using ml *moduleName*
3. run software

# Step 8: Monitor Job with sq

- The command sq --me will show you information about your pending and running jobs

```
cft07037@ss-sub1 training$ sq --me
JOBID    TIME   TIME_LIMIT   NAME          PARTITION    USER      NODES  CPUS   MIN_MEMORY  PRIORITY  STATE     NODELIST(REASON)
4618668  1:18   4:00:00      test-job      highmem_p    cft07037   1      1      200G         5993      RUNNING   d4-11
4618666  1:21   1:00:00      Bowtie2-2cpu  batch        cft07037   1      2      4G           5991      RUNNING   c1-23
4618665  5:25   2:00:00      testBowtie2   batch        cft07037   1      1      4G           5991      RUNNING   c1-23
```

# Step 8: Monitor Job with sq --help

- Running sq --help will show you the help page for this command:

```
cft07037@ss-sub1 training$ sq --help

Usage: sq [OPTIONS]

Description: sq - preformatted wrapper for squeue.  See man squeue for more information.
    -j                          Displays squeue output for a given job
    --me                        Displays squeue output for the user executing this command
    -p                          Displays squeue output for a given partition
    -u                          Displays squeue output for a given user
    -T                          Displays submit and start time columns
    -h, --help                  Displays this help output
```

# Step 8: Monitor Job with sacct-gacrc

- The command sacct-gacrc -X will show you information about your pending, running, and completed jobs

```
cft07037@ss-sub1 training$ sacct-gacrc -X
     JobID         JobName      User  Partition NNode NCPUS    ReqMem     CPUTime     Elapsed  Timelimit      State ExitCode   NodeList
---------- --------------- --------- ---------- ----- ----- -------- ---------- ---------- ---------- ---------- -------- ----------
   4618312        interact   cft07037    inter_p     1     1       2Gn   00:07:01   00:07:01   12:00:00  COMPLETED      0:0       c4-20
   4618665     testBowtie2   cft07037    batch       1     1       4Gn   00:23:12   00:23:12   01:00:00  COMPLETED      0:0       c1-23
   4618666    Bowtie2-2cpu   cft07037    batch       1     2       4Gn   00:20:24   00:10:12   01:00:00  COMPLETED      0:0       c1-23
   4618668        test-job   cft07037   highmem_p    1     1     200Gn   00:09:46   00:09:46   04:00:00  COMPLETED      0:0       d4-11
```

# Step 8: Monitor Job with sacct-gacrc options

- You can use the --starttime and --endtime options to modify the timeframe sacct-gacrc uses to show your jobs (default is midnight of previous day to your current time)

```
[cft07037@ss-sub1 ~]$ sacct-gacrc -X --starttime 2025-01-17 --endtime 2025-01-29
```

# Step 8: Monitor Job with sacct-gacrc --help

- Running sacct-gacrc --help will show you the help page for this command:

```
cft07037@ss-sub1 training$ sacct-gacrc --help

Usage: sacct-gacrc [OPTIONS]

Description: preformatted wrapper for sacct.  See man sacct for more information.
    -E, --endtime               Display information about jobs up to a date, in the format of yyyy-mm-dd (default: now)
    -j, --jobs                  Display information about a particular job or jobs (comma-separated list if more than one
job)
    -r, --partition             Display information about jobs from a particular partition
    -S, --starttime             Display information about jobs starting from a date in the format of yyyy-mm-dd (default:
Midnight of today)
    -T                          Display the end time of a particular job or jobs
    -u, --user                  Display information about a particular user's job(s) (default: current user)
    -X, --allocations           Only show one line per job (do not display job steps)
    --debug                     Display the sacct command being executed
    --prio                      Display the priority of a particular job or jobs
    -h, --help                  Display this help output
```

# Step 8: Monitor Job with seff

- You can check the resource usage of a completed job with seff JobID

```
[cft07037@ss-sub3 workdir]$ seff 34714807
Job ID: 34714807
Cluster: tc2
User/Group: cft07037/gacrc-appadmin
State: COMPLETED (exit code 0)
Nodes: 1
Cores per node: 3
CPU Utilized: 00:13:48
CPU Efficiency: 98.22% of 00:14:03 core-walltime
Job Wall-clock time: 00:04:41
Memory Utilized: 493.74 MB
Memory Efficiency: 12.05% of 4.00 GB
```

← How many CPU cores were given to your job

← How well your job used all of its CPU cores

← How much memory your job actually used

← How much memory was requested

# Step 8: Cancel Job with scancel

- You may cancel pending or running jobs with the command scancel JobID where you would replace JobID with your job's ID:

```
[cft07037@ss-sub1 workdir]$ scancel 34714806
```

- There is no way to undo canceling a job, so use this command wisely!

# Monitoring jobs overview

- sq --me for details of pending and running jobs
- sacct-gacrc -X for details of pending, running, and completed jobs
- seff to check the resource usage of a finished job

# Interactive jobs

- Interactive jobs are an alternative way to use software on the Sapelo2 cluster rather than using a submission script
- Interactive jobs will place you directly onto a compute node (from the login/submit node you were on originally)
- Pros:
  - Can load software directly from command line
  - Can run computations directly from command line
  - More hands-on and interactive than a submission script job
- Cons:
  - You must remain in the interactive job for the duration of your computations/processes (if you leave your interactive job, it will automatically end) so interactive jobs are better for short jobs
    - Whereas submission script jobs will continue to run in the background which makes them more versatile (for short or long jobs)

# Start an Interactive job

- To start an interactive job, run the command interact from a login/submit node
  - This will queue you up for an interactive job with default resources:

```
[cft07037@ss-sub3 workdir]$ interact

srun --pty  --cpus-per-task=1 --job-name=interact --ntasks=1 --nodes=1 --partition=inter_p --time=12:00:00 --mem=2GB /bin/bash -l

[cft07037@c4-16 workdir]$
```

- To start an interactive job will specific resources, type the desired resource after the interact command. For example, to request 7GB of memory (rather than use the default 2GB):

```
[cft07037@ss-sub3 workdir]$ interact --mem=7GB

srun --pty  --cpus-per-task=1 --job-name=interact --ntasks=1 --nodes=1 --partition=inter_p --time=12:00:00 --mem=7GB /bin/bash -l

[cft07037@c4-16 workdir]$
```

- You can leave an interactive job to go back to the login/submit node with the command exit
- You can type interact --help to see all of the options for this command

# Helpful GACRC Links

- GACRC Wiki http://wiki.gacrc.uga.edu
- Kaltura channel (GACRC video trainings) https://kaltura.uga.edu/channel/GACRC/176125031
- Connecting https://wiki.gacrc.uga.edu/wiki/Connecting#Connecting_to_Sapelo2
- Software https://wiki.gacrc.uga.edu/wiki/Software_on_Sapelo2
- Running Jobs https://wiki.gacrc.uga.edu/wiki/Running_Jobs_on_Sapelo2
- Monitoring Jobs https://wiki.gacrc.uga.edu/wiki/Monitoring_Jobs_on_Sapelo2
- Sample submission scripts
  https://wiki.gacrc.uga.edu/wiki/Sample_batch_job_submission_scripts_on_Sapelo2
- Transferring Files with Globus https://wiki.gacrc.uga.edu/wiki/Globus
- Linux Commands Help https://wiki.gacrc.uga.edu/wiki/Command_List
- Open OnDemand https://wiki.gacrc.uga.edu/wiki/OnDemand
- Other Trainings Offered https://wiki.gacrc.uga.edu/wiki/Training

# GACRC Ticketing Help/Support

https://uga.teamdynamix.com/TDClient/2060/Portal/Requests/ServiceCatalog?CategoryID=18080

- For Job Troubleshooting help:
  - Please include as many details as you can including:
    - Your UGA MyID
    - Your JobID
    - The full path to your submission script
    - The command you use to submit the job
- For Software Installations:
  - Include
    - Specific name and version of software
    - Download website
- For Training Sign Ups:
    - Include
      - Name of training
      - Date which you'd like to sign up for (find updated training dates on our wiki)

# Example Submission Script: Single Core Job

```
#!/bin/bash
#SBATCH --job-name=testBowtie2                  # Job name (testBowtie2)
#SBATCH --partition=batch                       # Queue name (batch)
#SBATCH --nodes=1                               # Run all processes on a single node
#SBATCH --ntasks=1                              # Run in a single task using one CPU core on a single node
#SBATCH --mem=4G                                # Job memory limit (4 GB)
#SBATCH --time=1:00:00                          # Time limit hrs:min:sec or days-hours:minutes:seconds
#SBATCH --output=%x_%j.out                      # Standard output log, e.g., testBowtie2_1234.out
#SBATCH --error=%x_%j.err                       # Standard error log, e.g., testBowtie2_1234.err


cd $SLURM_SUBMIT_DIR                            # Change directory to job submission directory
ml Bowtie2/2.4.1-GCC-8.3.0                      # Load software module and run bowtie2 below
bowtie2  -x  ./index/lambda_virus  -U  ./myreads.fq  -S  output.sam
```

# Example Submission Script: Multi-threaded Job

```
#!/bin/bash
#SBATCH --job-name=testBowtie2                  # Job name (testBowtie2)
#SBATCH --partition=batch                       # Queue name (batch)
#SBATCH --nodes=1                               # Run all processes on a single node
#SBATCH --ntasks=1                              # Run in a single task using 8 CPU cores on a single node
#SBATCH --cpus-per-task=8                       # Number of CPU cores per task (8)
#SBATCH --mem=10G                               # Job memory limit (10 GB)
#SBATCH --time=1:00:00                          # Time limit hrs:min:sec or days-hours:minutes:seconds
#SBATCH --output=%x_%j.out                      # Standard output log, e.g., testBowtie2_1234.out
#SBATCH --error=%x_%j.err                       # Standard error log, e.g., testBowtie2_1234.err


cd $SLURM_SUBMIT_DIR
ml Bowtie2/2.4.1-GCC-8.3.0
bowtie2 --threads 8 -x ./index/lambda_virus -U ./myreads.fq -S output.sam       # Run bowtie2 with 8 threads (for 8 CPU cores)
```

# Example Submission Script: MPI Job

```
#!/bin/bash
#SBATCH --job-name=testBowtie2                  # Job name (testBowtie2)
#SBATCH --partition=batch                       # Queue name (batch)
#SBATCH --nodes=2                               # Run all processes on 2 nodes
#SBATCH --ntasks-per-node=8                     # Run 8 processes on EACH node (total of 16 processes)
#SBATCH --cpus-per-task=1                       # Number of CPU cores PER task (total of 16 CPU cores)
#SBATCH –mem-per-cpu=10G                        # Job memory limit (10 GB)
#SBATCH --time=1:00:00                          # Time limit hrs:min:sec or days-hours:minutes:seconds
#SBATCH --output=%x_%j.out                      # Standard output log, e.g., testBowtie2_1234.out
#SBATCH --error=%x_%j.err                       # Standard error log, e.g., testBowtie2_1234.err


cd $SLURM_SUBMIT_DIR
ml foss/2019b                                                   # Load toolchain module
srun -n 16 ./myProgram.x                        # Run your program binary compiled with OpenMPI with 16 ranks
```

# Example Submission Script: Array Job

```
#!/bin/bash
#SBATCH --job-name=testBowtie2Array                  # Job name (testBowtie2)
#SBATCH --partition=batch                            # Queue name (batch)
#SBATCH --ntasks=1                                   # Run 1 process on 1 node
#SBATCH --cpus-per-task=1                            # Number of CPU cores per task
#SBATCH --mem=4G                                     # Job memory limit (4 GB)
#SBATCH --time=1:00:00                               # Time limit hrs:min:sec or days-hours:minutes:seconds
#SBATCH --output=%x_%j.out                           # Standard output log, e.g., testBowtie2_1234.out
#SBATCH --error=%x_%j.err                            # Standard error log, e.g., testBowtie2_1234.err
#SBATCH --array=0-9                                  # Array element range from 0 to 9, i.e. 10 element jobs


cd $SLURM_SUBMIT_DIR
ml Bowtie2/2.4.1-GCC-8.3.0                                                  # Original data is split into 10 pieces and run in each element job
bowtie2 -x ./index/lambda_virus -U ./myreads_$SLURM_ARRAY_TASK_ID.fq -S output_$SLURM_ARRAY_TASK_ID.sam
```

# Example Submission Script: GPU Job

```
#!/bin/bash
#SBATCH --job-name=amber_test                  # Job name (testBowtie2)
#SBATCH --partition=gpu_p                      # Queue name (gpu partition)
#SBATCH --gres=gpu:A100:1                      # Requests one GPU device
#SBATCH --ntasks=1                             # Run 1 process on 1 node
#SBATCH --cpus-per-task=2                      # Number of CPU cores per task
#SBATCH --mem=40G                              # Job memory limit (40 GB)
#SBATCH --time=10:00:00                        # Time limit hrs:min:sec or days-hours:minutes:seconds
#SBATCH --output=%x_%j.out                     # Standard output log, e.g., testBowtie2_1234.out
#SBATCH --error=%x_%j.err                      # Standard error log, e.g., testBowtie2_1234.err

cd $SLURM_SUBMIT_DIR
ml Python/3.11.3-GCCcore-12.3.0
python train.py
```

# General Dos and Don’ts

- Do NOT use Login nodes to run CPU/memory intensive tasks directly - submit jobs to Compute nodes!
- Do NOT use Login nodes to transfer data between your local computer and cluster - use Transfer nodes!
- Do NOT park data in Scratch or Local Scratch - clean up when job finishes or exits from node
- Do NOT park data permanently in Project - download data to your local drive
- NO large memory job running on batch partition - use highmem_p
- NO small memory job running on highmem_p partition - use batch
- In general, number of threads you want to run with a parallel job = number of cores requested
- When you archive data using tar on /scratch, please do not use the z option (compression option). After you archived data with tar, you can use gzip to compress it.

# File Storage Guidelines - 1

- No directory should have too many files inside! A good rule of thumb is to keep no more than a few tens of thousands of files (<10000 would be even better) in any single directory which is accessed frequently

All files are in ONE single dir!

Files are organized in subdirs

# File Storage Guidelines - 2

- When you need to get rid of some large number of files:
  - Move unwanted files that are in your /scratch dir into /scratch/trash/MyID with mv command

- If you need to back up many files onto the /project filesystem, especially relatively small files, please first create a tarball (.tar file) with those files and then transfer the tarball to /project, to reduce the number of files in /project.
  - The tar command to create a tarball of files in /work or /scratch can be run in a batch job or in an interactive job. Please do not run commands like tar and gzip on the Sapelo2 login nodes.