University of Colorado **Anschutz Medical Campus**

Introduction to Alpine

By Kevin Fotso
09/09/2025

Introduction:

This workshop will cover the following topics:

- What is Alpine and how to log into the Alpine system.
- Alpine filesystems and storage options
- Alpine data transfer options
- Alpine scheduler

Introduction:

This workshop will cover the following topics:

- *Alpine modules and software stack*
- *Alpine interactive and batch jobs*
- *Alpine resource monitoring will be covered during the workshop: “Software sustainability and efficiency on Alpine”*

Audience:

Users who are either new to Alpine or hoping to gain more Alpine experience

Alpine requirements

Reminder! Alpine cannot be used on sensitive data!

- No HIPAA or Protected Health Information (PHI)
- No General Data Protection Regulation (GDPR)
- No CMMC
- No FERPA
- No data that requires special security and compliance

PHI
PROTECTED HEALTH INFORMATION

DARPA

biobank uk
Enabling scientific discoveries that improve human health

CMMC
CYBERSECURITY MATURITY MODEL CERTIFICATION

DEPARTMENT OF DEFENSE
UNITED STATES OF AMERICA

DEPARTMENT OF EDUCATION
UNITED STATES OF AMERICA

FERPA
Family Educational Rights & Privacy Act

All data on Alpine must be 100% de-identified and make sure it is compliant with your IRB and/or grant funding agency

HPC

Benefits of HPC – Resource availability

Standard Laptop
~8GB RAM,
500GB hard drive,
Dual-core processor

Standard CPU Node on Alpine (amilan nodes)
64 cores, ~3.74GB/core
=240GB RAM

Alpine High Performance Compute Cluster
455 compute nodes for a total of 28,080 cores*

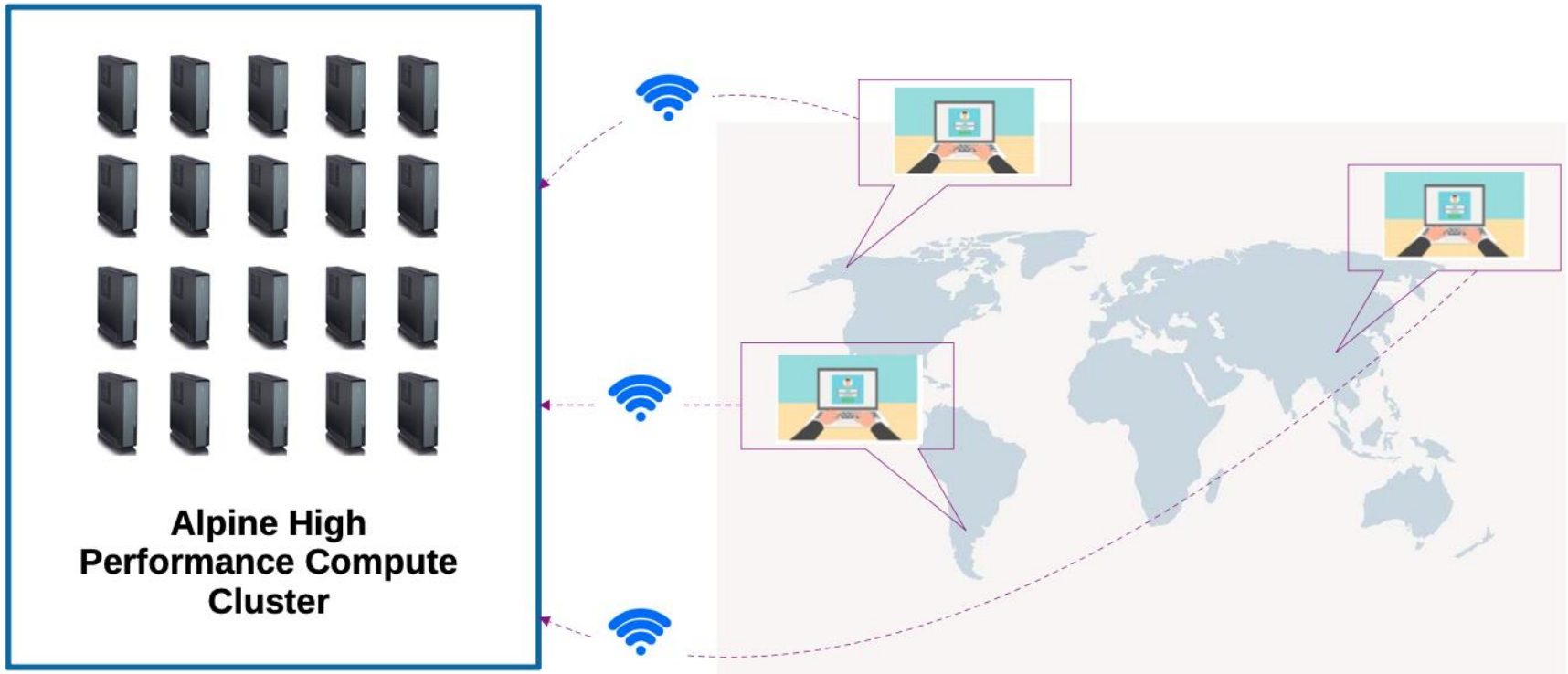
30 standard laptops = 1 CPU node on Alpine!

Alpine also has special nodes such as GPUs and high memory nodes which contain up to 2T of RAM.

* includes GPU & high mem

HPC

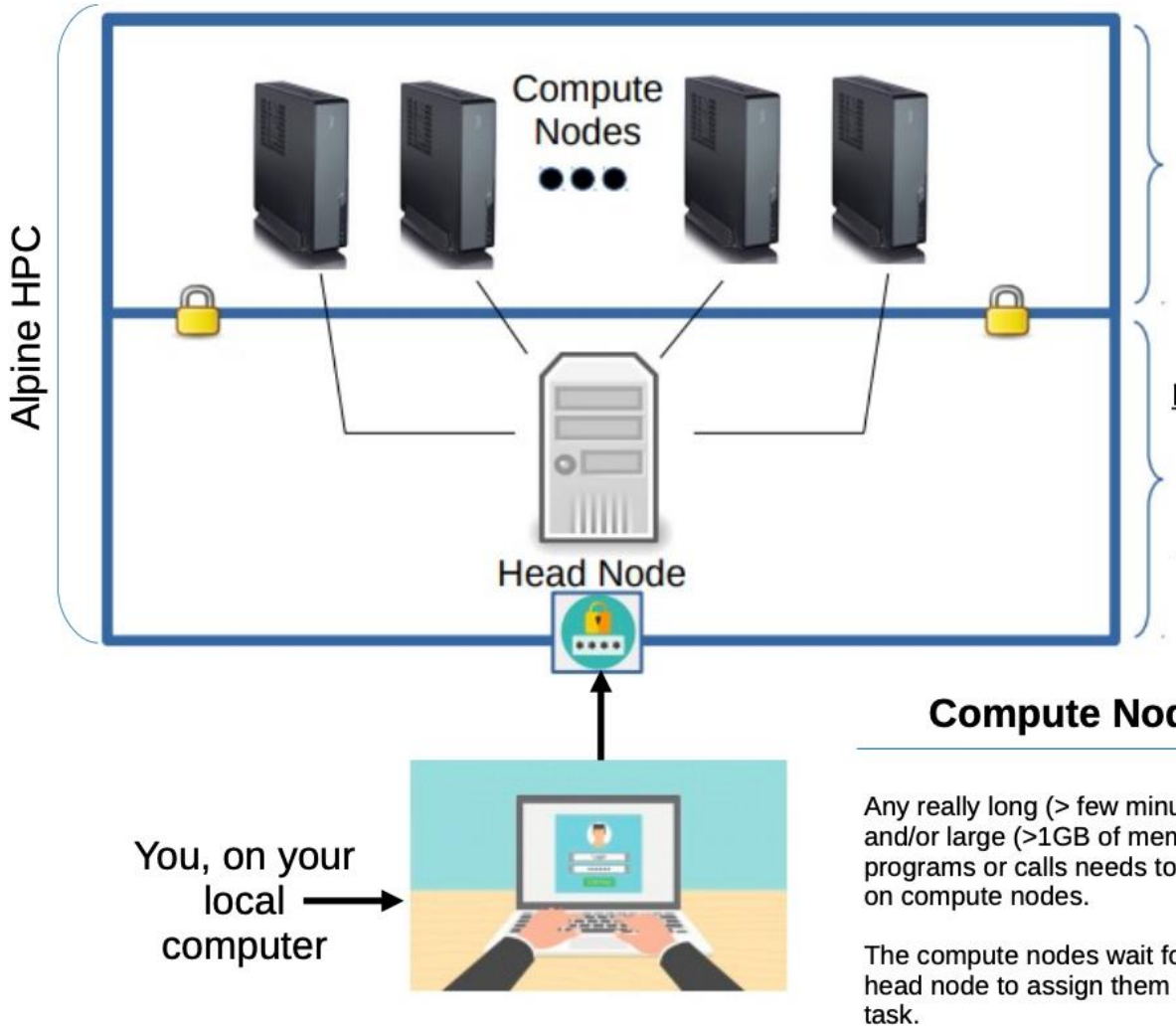
Benefits of HPC – Remote computation



Once you submit your job to Alpine or an HPC, you can turn off your computer. At this point, all the data, commands, scripts, etc… are copied over to the Boulder side where the power to the server is always on, therefore **turning off your computer after you submit your job is not at all tied to your local computer power**.

HPC

Anatomy of HPC



The compute nodes are where all the power of the HPC is harnessed. **This is where all the computationally heavy work occurs**.

Sometimes referred to as the **login node**. Its main purpose is to take your job requests and find compute resources for you to use.

This node is the master or brain of the HPC.

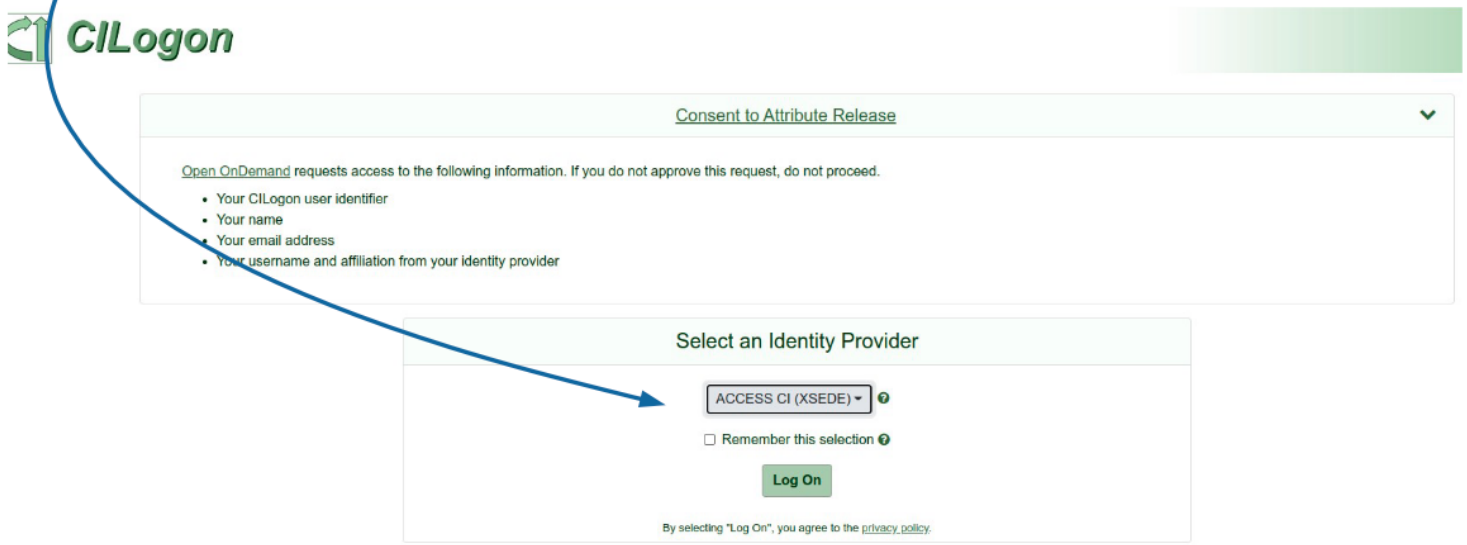
Compute Node	Head/Login Node
Any really long (> few minutes) and/or large (>1GB of memory) programs or calls needs to br run on compute nodes. The compute nodes wait for the head node to assign them a job or task.	Anytime a user makes calls through the command line on Alpine (except in acompile/sinteractive sessions), it is being performed on the head node. The head node is very small and limited in ability. Its main function is to work with the scheduler and delegate tasks and jobs to the compute nodes that users want to utilize

How to access Alpine

First let's login to Alpine!

1. Go to https://ondemand-rmacc.rc.colorado.edu

2. It should redirect you to CILogon which is how you authenticate your Alpine session. Make sure you select "ACCESS CI (XSEDE)" as your identity provide and then press "Log On"



University of Colorado Anschutz Medical Campus

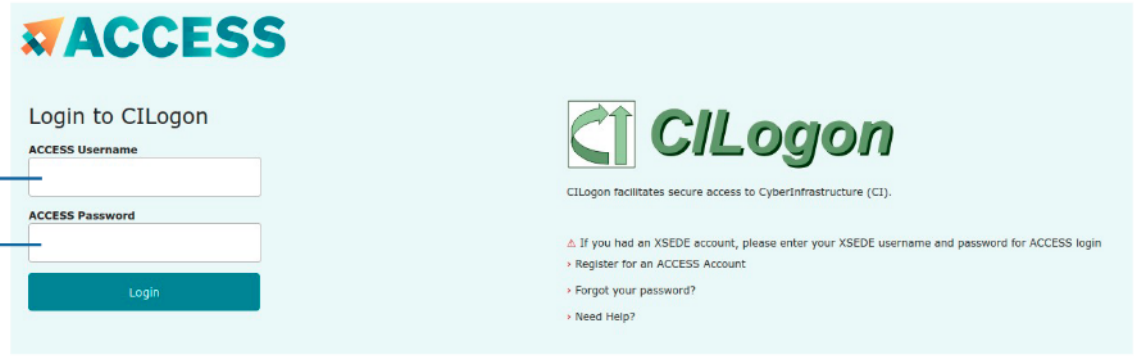
Source: https://github.com/kf-cuanschutz/CU-Anschutz-HPC-documentation/blob/main/Workshops/Alpine_Noob_Introduction_to_HPC_and_Alpine-120723.pdf

How to access Alpine

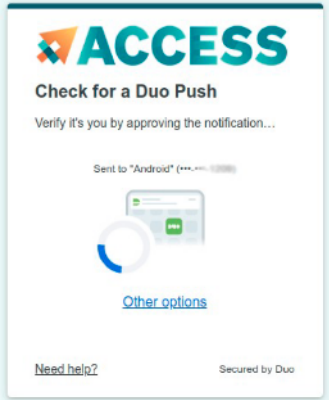
First let's login to Alpine!

3 Next it will take you to this page where you will put in your ACCESS ID and ACCESS password and press Login. This is NOT your CU Anschutz ID!!

ACCESS ID and ACCESS password **not** CU Anschutz credentials!!



4 This will prompt a DUO MFA push to your phone or whichever way you have DUO set up on your phone to authenticate. Accept the push sent to your device.



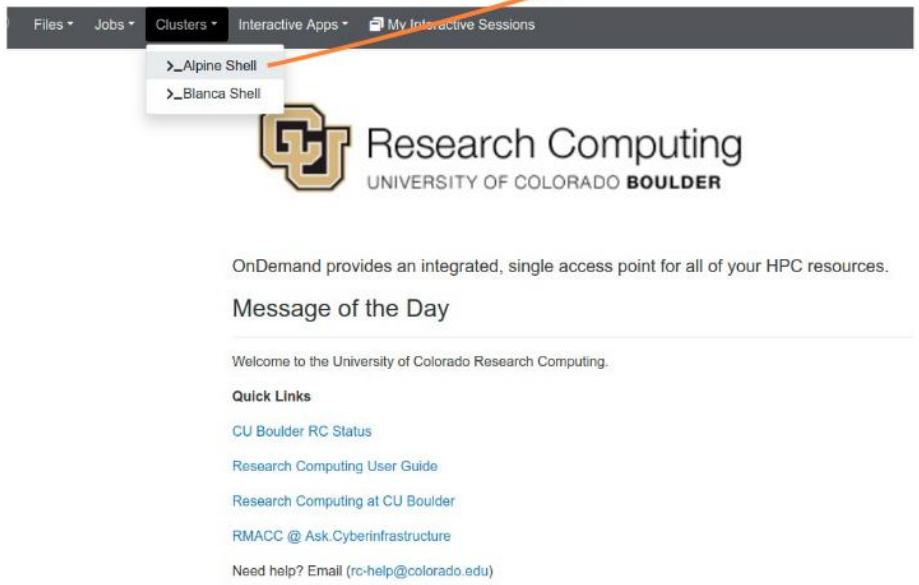
University of Colorado Anschutz Medical Campus

Source: https://github.com/kf-cuanschutz/CU-Anschutz-HPC-documentation/blob/main/Workshops/Alpine_Noob_Introduction_to_HPC_and_Alpine-120723.pdf

How to access Alpine

First let's login to Alpine!

5 This will take to the official CURC page. Let's select the Alpine terminal



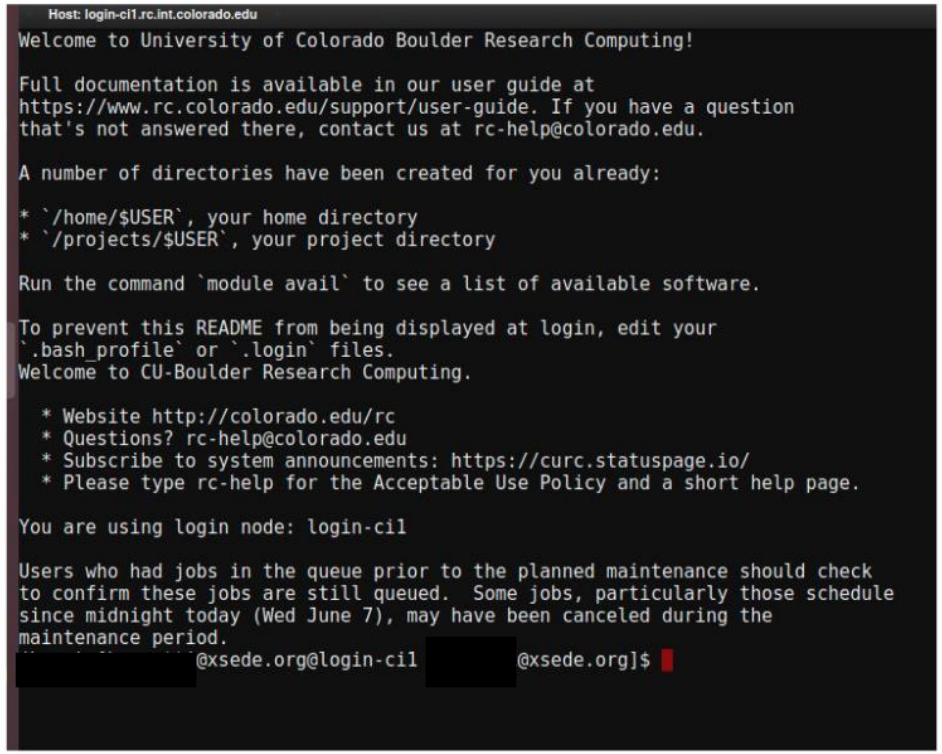
University of Colorado Anschutz Medical Campus

Source: https://github.com/kf-cuanschutz/CU-Anschutz-HPC-documentation/blob/main/Workshops/Alpine_Noob_Introduction_to_HPC_and_Alpine-120723.pdf

Make sure that you can access Alpine

First let's login to Alpine!

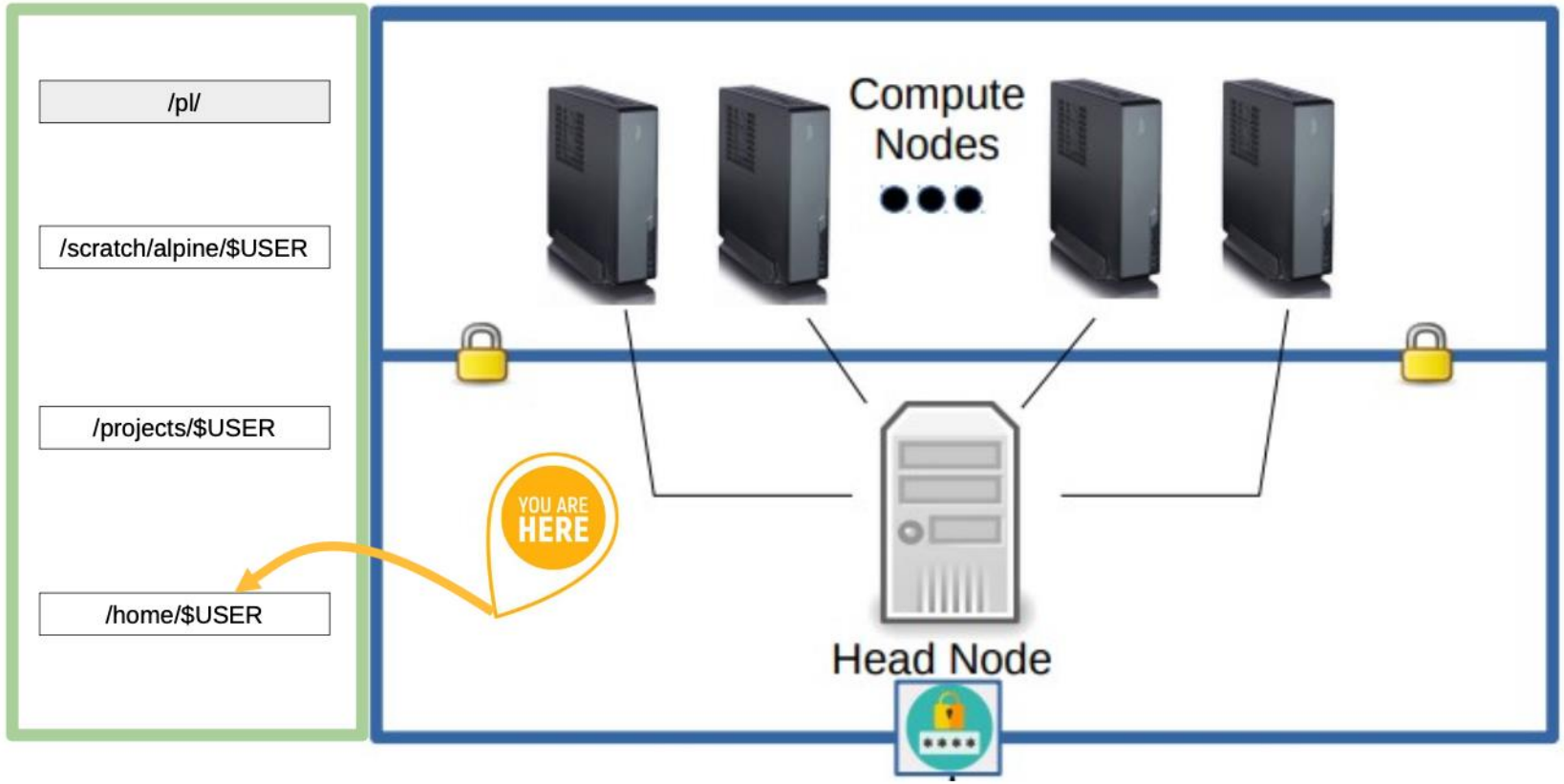
6 This will log you into the head node of Alpine. It will always default you to your home directory.



Source: https://github.com/kf-cuanschutz/CU-Anschutz-HPC-documentation/blob/main/Workshops/Alpine_Noob_Introduction_to_HPC_and_Alpine-120723.pdf

Storage

All 4 storage directories you have for storage are accessible by both the head/login node and the compute nodes



Every time you login to Alpine, **you will always be located in your /home/$USER directory on the head/login node**

University of Colorado Anschutz Medical Campus

Source: https://github.com/kf-cuanschutz/CU-Anschutz-HPC-documentation/blob/main/Workshops/Alpine_Noob_Introduction_to_HPC_and_Alpine-120723.pdf

Filesystems

Alpine HPC RoadMap

/home/$USER	/projects/$USER	/scratch/alpine/$USER	/pl/

Every time you login to Alpine, **you will always be located in your /home/$USER directory**

Filesystems

Alpine HPC RoadMap

/home/$USER	/projects/$USER	/scratch/alpine/$USER	/pl/

Every time you login to Alpine, **you will always be located in your /home/$USER directory**

It doesn't matter if you were in your scratch space when you logged out or where you were in any prior session; Alpine does not remember where you were last time, so it always puts you in your /home/$USER directory space

Filesystems

/home/$USER

/home/$USER is very, very small!
2GB total

For most users, you never want to do anything to or in home; this is where most your configuration files are located, such as your .bashrc, .bash_profile, .condarc, etc…

The only time you will do anything in your home space is if you want to modify these hidden configuration files

Do not place any data transfer here or store any data here!

Filesystems

/home/$USER	/projects/$USER	/scratch/alpine/$USER	/pl/

Let’s change directories into our projects directory...

```
cd /projects/$USER
```

Filesystems

/projects/$USER

This directory is slightly larger, at 250GB of space.

This should be used for more permanent storage of smaller files.

I recommend any local software installs (i.e. software you install yourself because it is not installed on Alpine) get installed here.

Additionally, all of your job scripts and code should be stored here

If small enough, reference data should be installed here; reference data meaning data that is used over and over again. In bioinformatics, this usually means genome index files, reference genome files (.fasta, .gtf), etc…

Again, not to be used for data storage of large files!

Filesystems

/home/$USER	/projects/$USER	/scratch/alpine/$USER	/pl/

Let’s change directories into our scratch directory...

```
cd /scratch/alpine/$USER
```

Filesystems

`/scratch/alpine/$USER`

This is your largest directory, at 10TB!

This is the location where most of your data should be transferred into when you are ready for analysis

Any computation where the software or program is expected to write out output files, should be redirected to this directory space

This has the best read/write performance and compute nodes will be able to handle large volumes of data from this locations.

Filesystems

`/scratch/alpine/$USER`

This is your largest directory, at 10TB!

This is the location where most of your data should be transferred into when you are ready for analysis

Any computation where the software or program is expected to write out output files, should be redirected to this directory space

This has the best read/write performance and compute nodes will be able to handle large volumes of data from this locations.

However, there are a few caveats…

- Storage is temporary – files are automatically deleted 90 days from the date of creation/transfer
- Do NOT have your only copy of data here or you risk losing it after the allotted time

Filesystems

Use the following command to keep track of the creation date of each file and folder:

mmlsattr -L <myfile>

Filesystems

Use the following command to keep track of the creation date of each file and folder:

mmlsattr -L <myfile>

- With <myfile> being the name of the file or folder that you wish to track.
- Remember that with Linux, any folder or file is considered as a file!

Filesystems

/home/$USER	/projects/$USER	/scratch/alpine/$USER	/pl/

For those of you that have a PetaLibrary allocation, you can access your allocation by using changing directories to the path that was given to you. Your allocation is mounted to your Alpine space automatically.

```
cd /pl/active/nameOfAllocation
```

Filesystems

/pl/

The benefits of having a petalibrary allocation is if you want permanent storage of data files so that you don't have to worry about constantly moving data in and out of your scratch space.

Consider a petalibrary allocation if:

- You have several TBs of data that you want permanently stored somewhere that Alpine has access to
- You don't want to manage moving files back and forth from scratch to your local computer

There is a cost associated with PL, which is ~$45/TB/year, or $65/TB/year if you want a second duplicate copy (recommended if you don't have a second copy save elsewhere) saved in case the hardware crashes

https://www.colorado.edu/rc/resources/petalibrary

Filesystems

/home/$USER

2GB

- Do not use this for anything regarding data transfer, storage, software installs, etc…
- One access pre-existing configuration files here if you need to modify them (.bashrc, .bash_profile, .condrc, etc...)

/projects/$USER

250GB

- Store all code, scripts, and sbatch files here
- Local software that you need to install yourself should go here
- If small enough, reference files that will be re-used, i.e. genome index, genome fasta or gtf annotation files, etc...

/scratch/alpine/$USER

10TB

- Location for all data transfers and large data to be used to computation and analysis
- Output files generated by software should be redirected here

There is a 90-day automated data purge from the date of file creation/transfer; not to be used for permanent storage or any data that is single copy

/pl/

1TB-??

- OPTIONAL – only available to those that are paying for an allocation
- Permanent data storage mounted to your Alpine user space
- There is a cost associated with it

What is Petalibrary?

- An Alpine service that supports the paid storage, archival, and sharing of research data.
- It is available at a subsidized cost to any researcher affiliated with the University of Colorado System (Boulder, Anschutz, Denver, Colorado Springs)

University of Colorado
Anschutz Medical Campus

Preliminary conditions

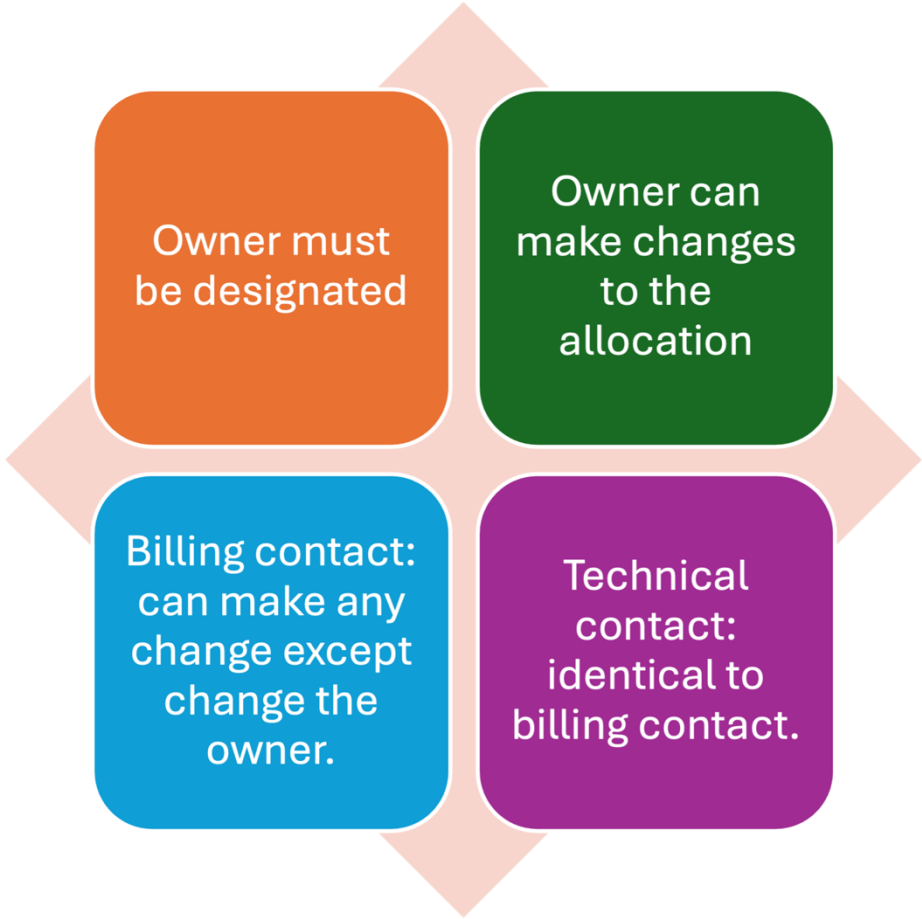
CREATION OF AN ACCESS GROUP

ALL MEMBER ADDED TO THE GROUP WILL NEED ACCESS/XSEDE ACCOUNTS.

```
/pl/active/<your_allocation_name>
/pl/archive/<your_allocation_name>
```

University of Colorado
Anschutz Medical Campus

Preliminary conditions



Application

- Application submitted here: https://www.colorado.edu/rc/resources/petalibrary

- The request form will need a speedtype: account# to which they plan to charge the allocation.

Billing

- On active storage: $45/TB/yr.
- ZFS Raidz2 allow for frequent read/write + parity.
- It is the responsibility of the PI to estimate and foresee the PL allotment for the year.

Billing

- On archive storage: $25/TB/yr.
- Tape-like storage for infrequently accessed data.
- Can be accessed from RC login node only and data transfer resources.
- Min size for any alloc is 1 TB.

Billing

- On archive storage: $25/TB/yr.
- Tape-like storage for infrequently accessed data.
- Can be accessed from RC login node only and data transfer resources.
- Min size for any alloc is 1 TB.
- **10,000 object (file/directory) limit.**
- **Owner will get and pay for the space even if they do not use it.**

Data redundancy

PL allocations are of single-copy nature.

All users should fill out the PL single copy acknowledgment.

Snapshots monitoring in place so that they are not missed unless on snapshot custom schedule

Data replication solution

- Replicated PetaLibrary active+archive allocation.
- CURC only takes data replication responsibility.

Data replication solution

- Replicated PetaLibrary active+archive allocation.
- CURC only takes data replication responsibility.
- Not subject to the 10,000 object (file/directory) limit.

Active+Archive replication

- Synchronization between active and archive happen every 15 min in theory but are not always guaranteed.
- Snapshots are also replicated from active to archive.
- The Boulder storage team maintains the second copy (e.g. archive) for you and you do not have direct access to it.
- $70/TB/year total. **(NOT $70 on top of the $45/$25 original rate!!)**

Archive DR

- CU Boulder storage team back it up monthly to an offsite location for you.
- More information here: https://curc.readthedocs.io/en/latest/storage/petalibrary/allocation_types.html#archive-dr-disaster-recovery

Petalibrary best practices.

- PL allocations fill up quickly due to snapshots if you use it on a daily basis.
- We recommend using your scratch filesystem as a buffer.

Petalibrary best practices.

- Transferring data into and out of your scratch space is typically much faster than doing so inside of your allocation.
- It is also more efficient to process large amount of data in your scratch filesystem.
- Once you are done, you can archive all the data you want to keep in your allocation.


University of Colorado Anschutz Medical Campus
Alpine compute

Now you are on the head/login node. What are your options?

Option A

Make a request to the the head node to login to a compute node so you can interactively run your code on a compute node, test code in real-time, or look at what software is installed on Alpine.

Option B

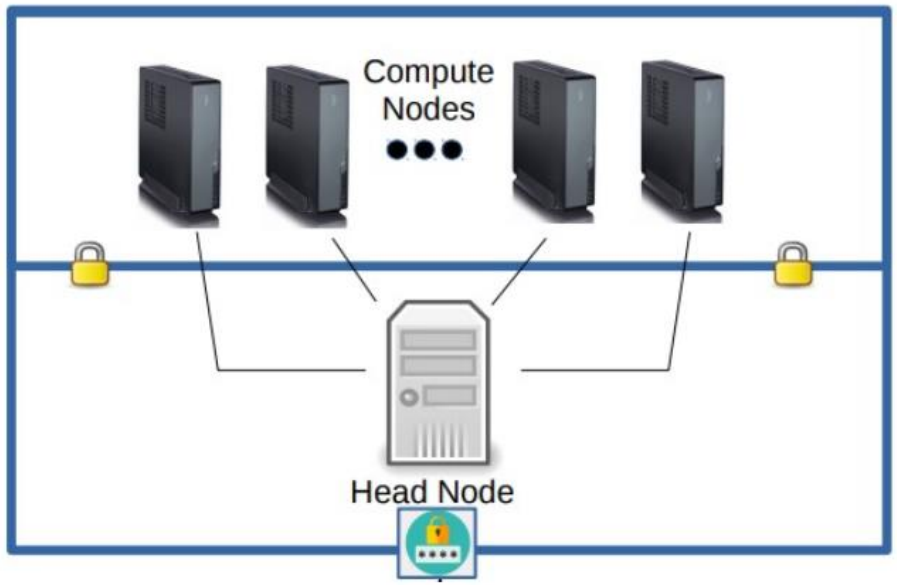
Stay on the head node because you want to submit a job remotely to a compute node.

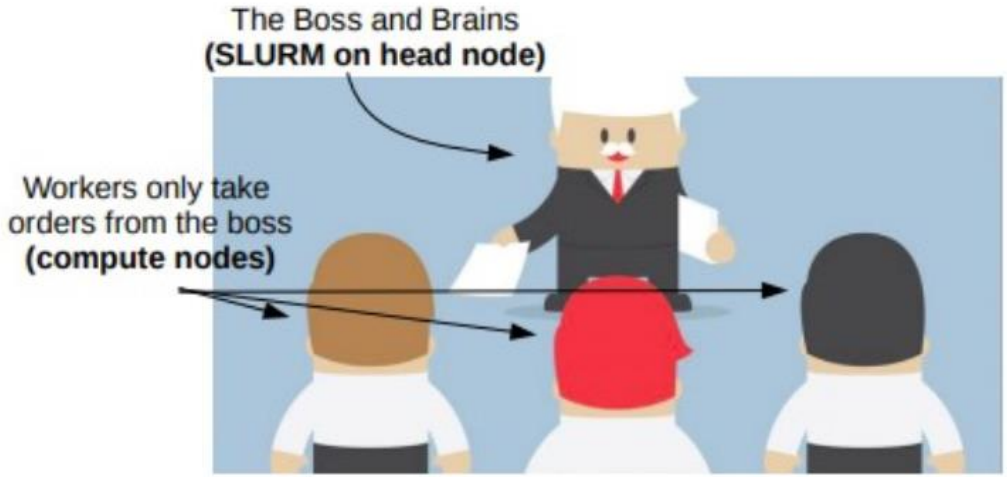
Regardless of which option you choose, you still need to communicate that to the head/login node, so how do we do that?

- You can communicate via **S**imple **L**inux **U**tility for **R**esource **M**anagement (SLURM) job scheduling language

- The head/login node and compute nodes all speak SLURM, so your task is to translate what you want into SLURM format.

- SLURM is a workload manager and is just one of the many flavor of job scheduling softwares available for HPCs. You may have heard of LSF (bsub, bqueues, etc…) or PBS (qsub, qstat, etc…) which are other flavors that are commonly used in Academia and all accomplish the same task – scheduling jobs by orchestrating communication between head/login nodes and compute nodes.



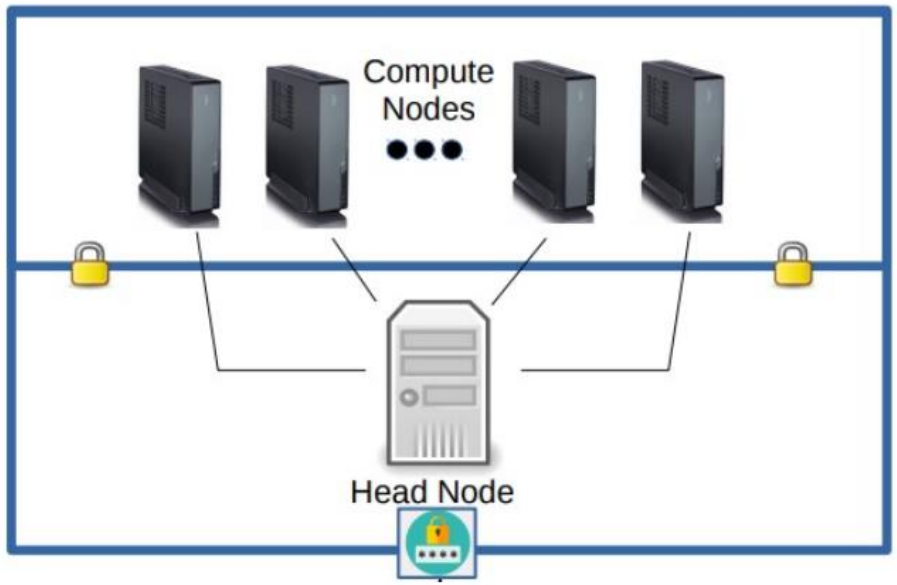


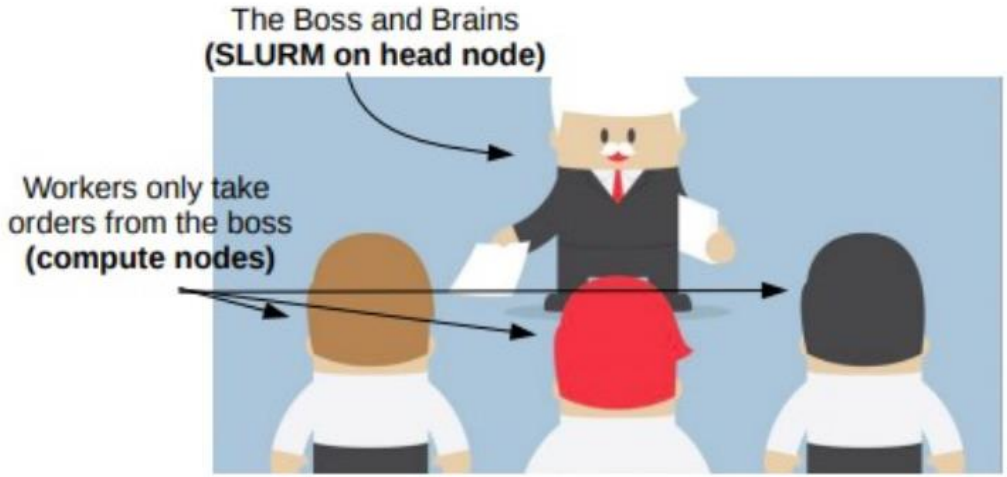
Regardless of which option you choose, you still need to communicate that to the head/login node, so how do we do that?

- You can communicate via **S**imple **L**inux **U**tility for **R**esource **M**anagement (SLURM) job scheduling language

- The head/login node and compute nodes all speak SLURM, so your task is to translate what you want into SLURM format.

- SLURM is a workload manager and is just one of the many flavor of job scheduling softwares available for HPCs. You may have heard of LSF (bsub, bqueues, etc…) or PBS (qsub, qstat, etc…) which are other flavors that are commonly used in Academia and all accomplish the same task – scheduling jobs by orchestrating communication between head/login nodes and compute nodes.



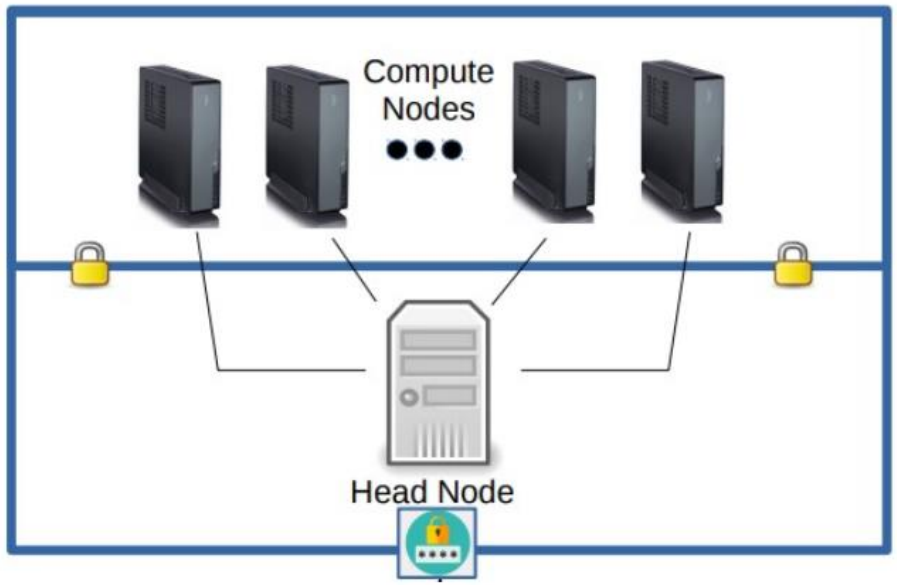


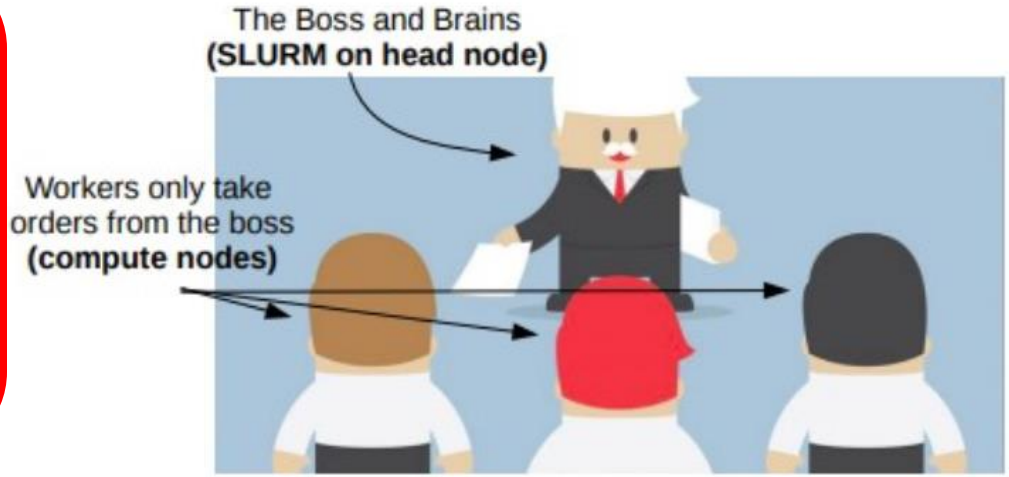
Regardless of which option you choose, you still need to communicate that to the head/login node, so how do we do that?

- You can communicate via **S**imple **L**inux **U**tility for **R**esource **M**anagement (SLURM) job scheduling language

- The head/login node and compute nodes all speak SLURM, so your task is to translate what you want into SLURM format.

- SLURM is a workload manager and is just one of the many flavor of job scheduling softwares available for HPCs. You may have heard of LSF (bsub, bqueues, etc…) or PBS (qsub, qstat, etc…) which are other flavors that are commonly used in Academia and all accomplish the same task – scheduling jobs by orchestrating communication between head/login nodes and compute nodes.





HPCs use job schedulers

- What is a job scheduler?
  - Responsible for executing multiple job requests (i.e. your scripts and programs) whether they come from the same or different user(s) and determining what resources are available, and whose jobs are next in line to be run

HPCs use job schedulers

- What is a job scheduler?
  - Responsible for executing multiple job requests (i.e. your scripts and programs) whether they come from the same or different user(s) and determining what resources are available, and whose jobs are next in line to be run
- Why do we need a scheduler?
  - “Fair-Share”
    - Sophisticated algorithm to schedule jobs fairly on a shared resource
    - The scheduler knows when each user made submitted requests so it can plan jobs effectively and know how long a job has been waiting in the queue or line
    - In the even all resources are being utilized, the “fair-share” policy is implemented meaning those users who have used the least amount of compute hours/resources within a window of time, get pushed to the front of the queue.
      - Prevents one individual from constantly using all the resources as the become available

HPCs use job schedulers

- What is a job scheduler?
  - Responsible for executing multiple job requests (i.e. your scripts and programs) whether they come from the same or different user(s) and determining what resources are available, and whose jobs are next in line to be run
- Why do we need a scheduler?
  - “Fair-Share”
    - Sophisticated algorithm to schedule jobs fairly on a shared resource
    - The scheduler knows when each user made submitted requests so it can plan jobs effectively and know how long a job has been waiting in the queue or line
    - In the even all resources are being utilized, the “fair-share” policy is implemented meaning those users who have used the least amount of compute hours/resources within a window of time, get pushed to the front of the queue.
      - Prevents one individual from constantly using all the resources as the become available
  - Efficiency
    - Jobs that are shorter or require less resources typically are in the queue shorter since resources become available more quickly and are more likely to fit into untapped resources
    - The scheduler know how many resources every user has requested and the time and memory constraints of those resources

Now you are on the head/login node. What are your options?

Option A

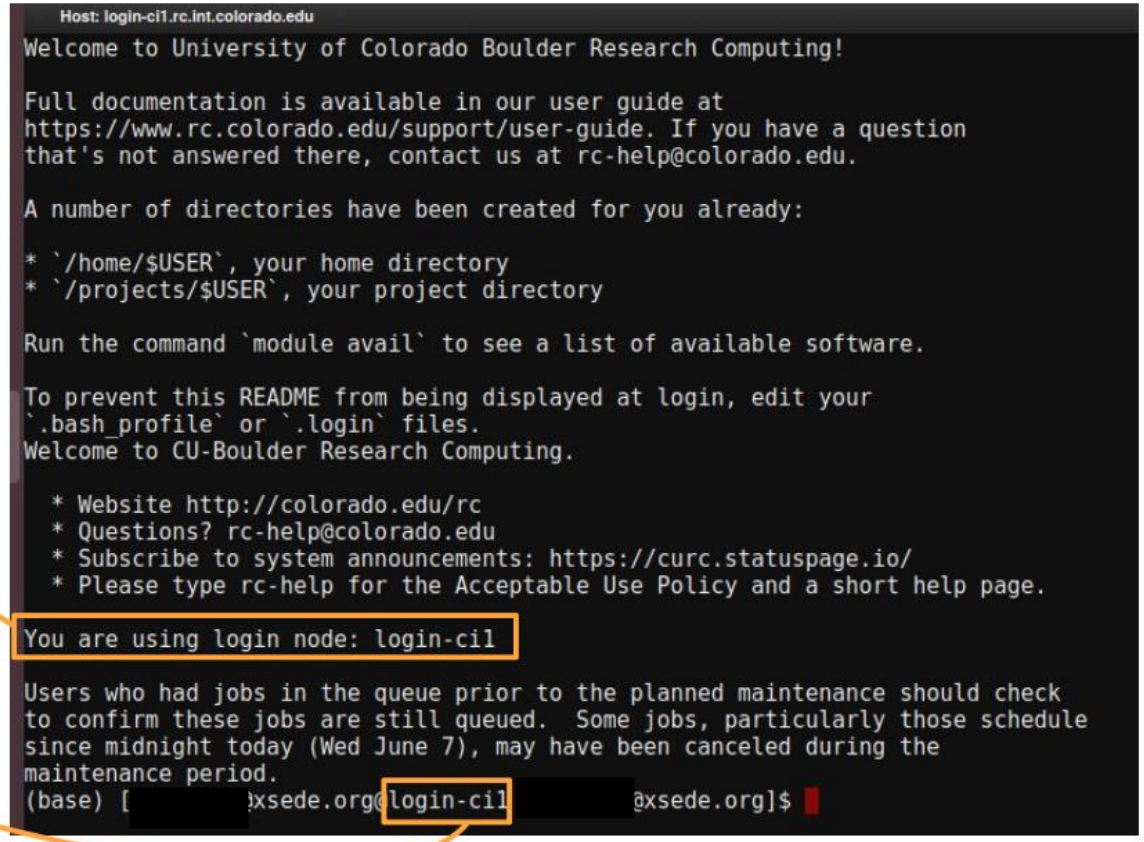
Make a request to the the head node to login to a compute node so you can interactively run your code on a compute node, test code in real-time, or look at what software is installed on Alpine.

Option B

Stay on the head node because you want to submit a job remotely to a compute node.

Option A: Request an interactive compute node

In case you get confused or are unsure if you are on the head/login node or a compute node you can check here! Anything that say `login-ci` followed by a number is the login/head node. I am on `login-ci1`, but you might have a different ci number



Request an short interactive compute node

```
[          @xsede.org@login-cli          @xsede.org]$ acompile
```

slurm workload manager

Now, we are inside of a compute node!

```
Welcome to University of Colorado Boulder Research Computing!

Full documentation is available in our user guide at
https://www.rc.colorado.edu/support/user-guide. If you have a question
that's not answered there, contact us at rc-help@colorado.edu.

A number of directories have been created for you already:

* `/home/$USER`, your home directory
* `/projects/$USER`, your project directory

Run the command `module avail` to see a list of available software.

To prevent this README from being displayed at login, edit your
`.bash_profile` or `.login` files.
Welcome to CU-Boulder Research Computing.

  * Website http://colorado.edu/rc
  * Questions? rc-help@colorado.edu
  * Subscribe to system announcements: https://curc.statuspage.io/
  * Please type rc-help for the Acceptable Use Policy and a short help page.

You are using login node: login-ci1

Users who had jobs in the queue prior to the planned maintenance should check
to confirm these jobs are still queued.  Some jobs, particularly those schedule
since midnight today (Wed June 7), may have been canceled during the
maintenance period.
(base) [brunetti@xsede.org@login-ci1 brunetti@xsede.org]$ acompile
acompile: submitting job... salloc --nodes=1 --partition=acompile --ntasks=1  --time=01:00:00  --qos=compile --job-name=acompile --bell --oversubscribe srun --pty /bin/bash
salloc: Granted job allocation 2650311
salloc: Nodes c3cpu-c15-u7-2 are ready for job
(base) [brunetti@xsede.org@c3cpu-c15-u7-2 brunetti@xsede.org]$
```

You can now see that I am on a compute node. Some of the numbers and characters may be a bit different as there are 455 different compute nodes on Alpine as of September 2025, but the big thing is you see it says `cpu` somewhere in there.

Now, we are inside of a compute node!

```
Welcome to University of Colorado Boulder Research Computing!

Full documentation is available in our user guide at
https://www.rc.colorado.edu/support/user-guide. If you have a question
that's not answered there, contact us at rc-help@colorado.edu.

A number of directories have been created for you already:

* `/home/$USER`, your home directory
* `/projects/$USER`, your project directory

Run the command `module avail` to see a list of available software.

To prevent this README from being displayed at login, edit your
`.bash_profile` or `.login` files.
Welcome to CU-Boulder Research Computing.

  * Website http://colorado.edu/rc
  * Questions? rc-help@colorado.edu
  * Subscribe to system announcements: https://curc.statuspage.io/
  * Please type rc-help for the Acceptable Use Policy and a short help page.

You are using login node: login-ci1

Users who had jobs in the queue prior to the planned maintenance should check
to confirm these jobs are still queued.  Some jobs, particularly those schedule
since midnight today (Wed June 7), may have been canceled during the
maintenance period.
(base) [bruneti@xsede.org@login-ci1 bruneti@xsede.org]$ acompile
acompile: submitting job... salloc --nodes=1 --partition=acompile --ntasks=1  --time=01:00:00  --qos=compile --job-name=acompile --bell --oversubscribe srun --pty /bin/bash
salloc: Granted job allocation 2650311
salloc: Nodes c3cpu-c15-u7-2 are ready for job
(base) [        @xsede.org@c3cpu-c15-u7-2          xsede.org]$
```

Notice that if you call `acompile` without any specifications it shows these parameters. This means you are requesting a CPU with 1 core (`--ntasks=1`), for 1 hour (`--time=01:00:00`), which is the default

This means that your session will automatically close/get killed if you use if longer than 1 hour.

The command below from the head node would not kill your job until 12 hours, which is also the maximum time you can request and interactive job

```
[          @xsede.org@login-cli          @xsede.org]$ acompile --time=12:00:00
```

Option A: Things we can do once we are interactively inside of a compute node...

- Check what software is installed on Alpine

- Install our own software

- Run software and code interactively

LMOD (CONTEXT)

- Modern environment module system for HPC
- Initially introduced by Robert McLay from TACC (UT Austin) in 2011.

LMOD (CONTEXT)

- Users have on HPC have different needs.
- Applications, compilers, libraries, versions being used might be different.

LMOD (benefits)

- Users do not need to know where software is installed
- Environment variable to interface packages can be set (e.g. picard).
- Very useful for software reproducibility

LMOD (benefits)

- Supports software hierarchy
- Users can set their environment modules at will.

Option A: Things we can do once we are interactively inside of a compute node...

- **Check what software is installed on Alpine**


```
module avail
```

Option A: Things we can do once we are interactively inside of a compute node...

- **Check what software is installed on Alpine**


```
module avail
```

```
base) [redacted]@xsede.org@c3cpu-c11-u17-2 [redacted]@xsede.org]$ module avail

------------------------------------------ /curc/sw/modules/slurm ------------------------------------------
   StdEnv    curc-quota/latest (D)    slurm/alpine (D)    slurm/blanca    slurm/core    slurmtools (D)

------------------------------------------ /usr/share/lmod/lmod/modulefiles/Core ------------------------------------------
   lmod    settarg

------------------------------------------ Compilers ------------------------------------------
   aocc/3.1.0 (D)    aocc/3.2.0    gcc/10.3.0    gcc/11.2.0 (D)    intel/2022.1.2 (m)    nvhpc_sdk/2022.229    nvhpc_sdk/2023.233 (D)

------------------------------------------ Independent Applications ------------------------------------------
   R/3.6.3                cuda/11.3          (g)    ghostscript/9.56.0          julia/1.6.6                paraview/5.0.1          rclone/1.58.0              tdom/0.9.2              (D)
   R/4.2.2          (D)   cuda/11.4          (g)    git-lfs/3.1.2               julia/1.8.1          (D)   paraview/5.6.0          rhel7for8/1.0              texlive/2021
   allinea/6.0.4    (m)   cuda/11.8          (g)    git/2.31.0                  lftp/4.8.4                 paraview/5.9.0          rocm/5.2.3          (g)    totalview/2016.
6.21
   anaconda/2020.11       cuda/12.1.1        (g,D)  gmsh/2.16.0                 loadbalance/0.2            paraview/5.10.0  (D)    rocm/5.3.0          (g)    ucx/1.10.1
   anaconda/2022.10 (D)   cudnn/8.1          (g)    gmsh/4.11.1           (D)   mambaforge/23.1.0-1        pdtoolkit/3.22             rocm/5.5.0          (g,D)  ucx/1.12.1              (D)
   arm-forge/19.1.3 (m)   cudnn/8.2          (g)    gnu_parallel/20160622       mathematica/9.0            pdtoolkit/3.25.1 (D)       ruby/2.3.1                 udunits/2.2.20
   autotools/2.69         cudnn/8.6          (g,D)  gnu_parallel/20210322 (D)   mathematica/11.1.0  (D)    perl/5.16.3                ruby/3.0.0          (D)    udunits/2.2.24
   autotools/2.71   (D)   curc-quota/latest         gnuplot/5.4.3               matlab/R2018b              perl/5.24.0         (D)    singularity/3.6.4 (D)      udunits/2.2.25
   chimerax/1.2.5         dmtcp/2.6.0               idl/8.7                     matlab/R2019b              perl/5.28.1                singularity/3.7.4          udunits/2.2.28          (D)
   cmake/3.5.2            eigen/3.4.0               imagemagick/6.9.12          matlab/R2020b              perl/5.36.0                slurmtools/0.0.0           valgrind/3.11.0
   cmake/3.9.2            emacs/25.3                jdk/1.7.0                   matlab/R2021b        (D)   pigz/2.7                   slurmtools/0.0.1           valgrind/3.17.0         (D)
   cmake/3.14.1           emacs/27.2         (D)    jdk/1.8.0_91                matlab/R2022b              python/2.7.18              subversion/1.8.16          vapor/3.3.0
   cmake/3.20.2           expat/2.1.1               jdk/1.8.0_281               maven/3.8.1                python/3.10.2    (D)       subversion/1.10.2          vapor/3.4.0             (D)
   cmake/3.25.0     (D)   expat/2.3.0        (D)    jdk/1.8.0                   ncl/6.3.0                  qchem/4010                 subversion/1.14.1 (D)      vtf3/1.43
   cube/3.4.3             ffmpeg/4.4                jdk/18.0.1.1          (D)   ncl/6.6.2            (D)   qt/5.6.0                   tcltk/8.6.5                zip/rhel7
   cube/4.3.4       (D)   gdb/8.1                   julia/0.6.2                 papi/5.4.3                 qt/5.9.1                   tcltk/8.6.11       (D)
   cuda/11.2        (g)   gdb/10.1           (D)    julia/1.6.0                 papi/5.5.1           (D)   qt/5.15             (D)    tdom/0.8.3

------------------------------------------ Bioinformatics ------------------------------------------
   alphafold/2.2.0        bbtools/39.01      bowtie2/2.5.0       cutadapt/4.2     homer/4.11      nextflow/22.10.6        plink2/2.00a2.3    sra-toolkit/3.0.0
   alphafold/2.3.1 (D)    bcftools/1.16      bwa/0.7.17          fastqc/0.11.9    htslib/1.16     nextflow/23.04   (D)    qiime2/2023.5      star/2.7.10b
   bamtools/2.5.2         bedtools/2.29.1    cellranger/7.1.0    gatk/4.3.0.0     multiqc/1.14    picard/2.27.5           samtools/1.16.1    trimmomatic/0.39

------------------------------------------ Lmod Internal Modules ------------------------------------------
   StdEnv    lmod    settarg

 Where:
   g:  built for GPU
   m:  built for host and native MIC
   D:  Default Module

se "module spider" to find all possible modules and extensions.
se "module keyword key1 key2 ..." to search for all possible modules matching any of the "keys".
```

Option A: Things we can do once we are interactively inside of a compute node...

- **Check what software is installed on Alpine**

module avail

Just a reminder! **I have to be inside of a compute node** to see this, not the head/login node!!

```
base) [          xsede.org@c3cpu-c11-u17-2          @xsede.org]$ module avail

------------------------------------------------------------------------------ /curc/sw/modules/slurm ------------------------------------------------------------------------------
   StdEnv    curc-quota/latest (D)    slurm/alpine (D)    slurm/blanca    slurm/core    slurmtools (D)

------------------------------------------------------------------------------ /usr/share/lmod/lmod/modulefiles/Core ------------------------------------------------------------------------------
   lmod    settarg

------------------------------------------------------------------------------ Compilers ------------------------------------------------------------------------------
   aocc/3.1.0 (D)    aocc/3.2.0    gcc/10.3.0    gcc/11.2.0 (D)    intel/2022.1.2 (m)    nvhpc_sdk/2022.229    nvhpc_sdk/2023.233 (D)

------------------------------------------------------------------------------ Independent Applications ------------------------------------------------------------------------------
   R/3.6.3                cuda/11.3          (g)      ghostscript/9.56.0          julia/1.6.6                paraview/5.0.1          rclone/1.58.0             tdom/0.9.2            (D)
   R/4.2.2          (D)   cuda/11.4          (g)      git-lfs/3.1.2               julia/1.8.1          (D)   paraview/5.6.0          rhel7for8/1.0             texlive/2021
   allinea/6.0.4    (m)   cuda/11.8          (g)      git/2.31.0                  lftp/4.8.4                 paraview/5.9.0          rocm/5.2.3          (g)   totalview/2016.
6.21
   anaconda/2020.11       cuda/12.1.1        (g,D)    gmsh/2.16.0                 loadbalance/0.2            paraview/5.10.0  (D)    rocm/5.3.0          (g)   ucx/1.10.1
   anaconda/2022.10 (D)   cudnn/8.1          (g)      gmsh/4.11.1           (D)   mambaforge/23.1.0-1        pdtoolkit/3.22          rocm/5.5.0          (g,D) ucx/1.12.1            (D)
   arm-forge/19.1.3 (m)   cudnn/8.2          (g)      gnu_parallel/20160622       mathematica/9.0            pdtoolkit/3.25.1 (D)    ruby/2.3.1                udunits/2.2.20
   autotools/2.69         cudnn/8.6          (g,D)    gnu_parallel/20210322 (D)   mathematica/11.1.0 (D)     perl/5.16.3             ruby/3.0.0          (D)   udunits/2.2.24
   autotools/2.71   (D)   curc-quota/latest           gnuplot/5.4.3               matlab/R2018b              perl/5.24.0       (D)   singularity/3.6.4 (D) udunits/2.2.25
   chimerax/1.2.5         dmtcp/2.6.0                 idl/8.7                     matlab/R2019b              perl/5.28.1             singularity/3.7.4         udunits/2.2.28      (D)
   cmake/3.5.2            eigen/3.4.0                 imagemagick/6.9.12          matlab/R2020b              perl/5.36.0             slurmtools/0.0.0          valgrind/3.11.0
   cmake/3.9.2            emacs/25.3                  jdk/1.7.0                   matlab/R2021b        (D)   pigz/2.7                slurmtools/0.0.1          valgrind/3.17.0     (D)
   cmake/3.14.1           emacs/27.2          (D)     jdk/1.8.0_91                matlab/R2022b              python/2.7.18           subversion/1.8.16         vapor/3.3.0
   cmake/3.20.2           expat/2.1.1                 jdk/1.8.0_281               maven/3.8.1                python/3.10.2    (D)    subversion/1.10.2         vapor/3.4.0         (D)
   cmake/3.25.0     (D)   expat/2.3.0         (D)     jdk/1.8.0                   ncl/6.3.0                  qchem/4010              subversion/1.14.1 (D)     vtf3/1.43
   cube/3.4.3             ffmpeg/4.4                  jdk/18.0.1.1          (D)   ncl/6.6.2            (D)   qt/5.6.0                tcltk/8.6.5               zip/rhel7
   cube/4.3.4       (D)   gdb/8.1                     julia/0.6.2                 papi/5.4.3                 qt/5.9.1                tcltk/8.6.11      (D)
   cuda/11.2        (g)   gdb/10.1            (D)     julia/1.6.0                 papi/5.5.1           (D)   qt/5.15           (D)   tdom/0.8.3

------------------------------------------------------------------------------ Bioinformatics ------------------------------------------------------------------------------
   alphafold/2.2.0        bbtools/39.01      bowtie2/2.5.0      cutadapt/4.2     homer/4.11      nextflow/22.10.6        plink2/2.00a2.3    sra-toolkit/3.0.0
   alphafold/2.3.1 (D)    bcftools/1.16      bwa/0.7.17         fastqc/0.11.9    htslib/1.16     nextflow/23.04   (D)    qiime2/2023.5      star/2.7.10b
   bamtools/2.5.2         bedtools/2.29.1    cellranger/7.1.0   gatk/4.3.0.0     multiqc/1.14    picard/2.27.5           samtools/1.16.1    trimmomatic/0.39

------------------------------------------------------------------------------ Lmod Internal Modules ------------------------------------------------------------------------------
   StdEnv    lmod    settarg

  Where:
   g:  built for GPU
   m:  built for host and native MIC
   D:  Default Module

se "module spider" to find all possible modules and extensions.
se "module keyword key1 key2 ..." to search for all possible modules matching any of the "keys".
```

Alpine software

- Refer to this list to look for all the already installed software on Alpine: https://curc.readthedocs.io/en/latest/clusters/alpine/software.html

- We will cover this topic more in depth during the workshop: Alpine module stack and miniforge.

Option A: Things we can do once we are interactively inside of a compute node...

- Check what software is installed on Alpine

- Install our own software

- Run software and code interactively

Option A: Things we can do once we are interactively inside of a compute node...

- **Install our own software**

You will notice that not all software is installed on Alpine but you can install Alpine yourself locally. There are typically two recommended ways of doing this:

  - Through a miniforge (preferred) or anaconda environment

  - Following the website instructions to install from source locally

Option A: Things we can do once we are interactively inside of a compute node...

- **Install our own software – using** miniforge or anaconda

Recall, that your home directory is really, really small (only 2GB), so we always want to make sure software is installed in our projects directory (`/projects/$USER/`). By default conda likes to install in home, so before do any conda work, we must change our `.condarc` file to include the following lines:

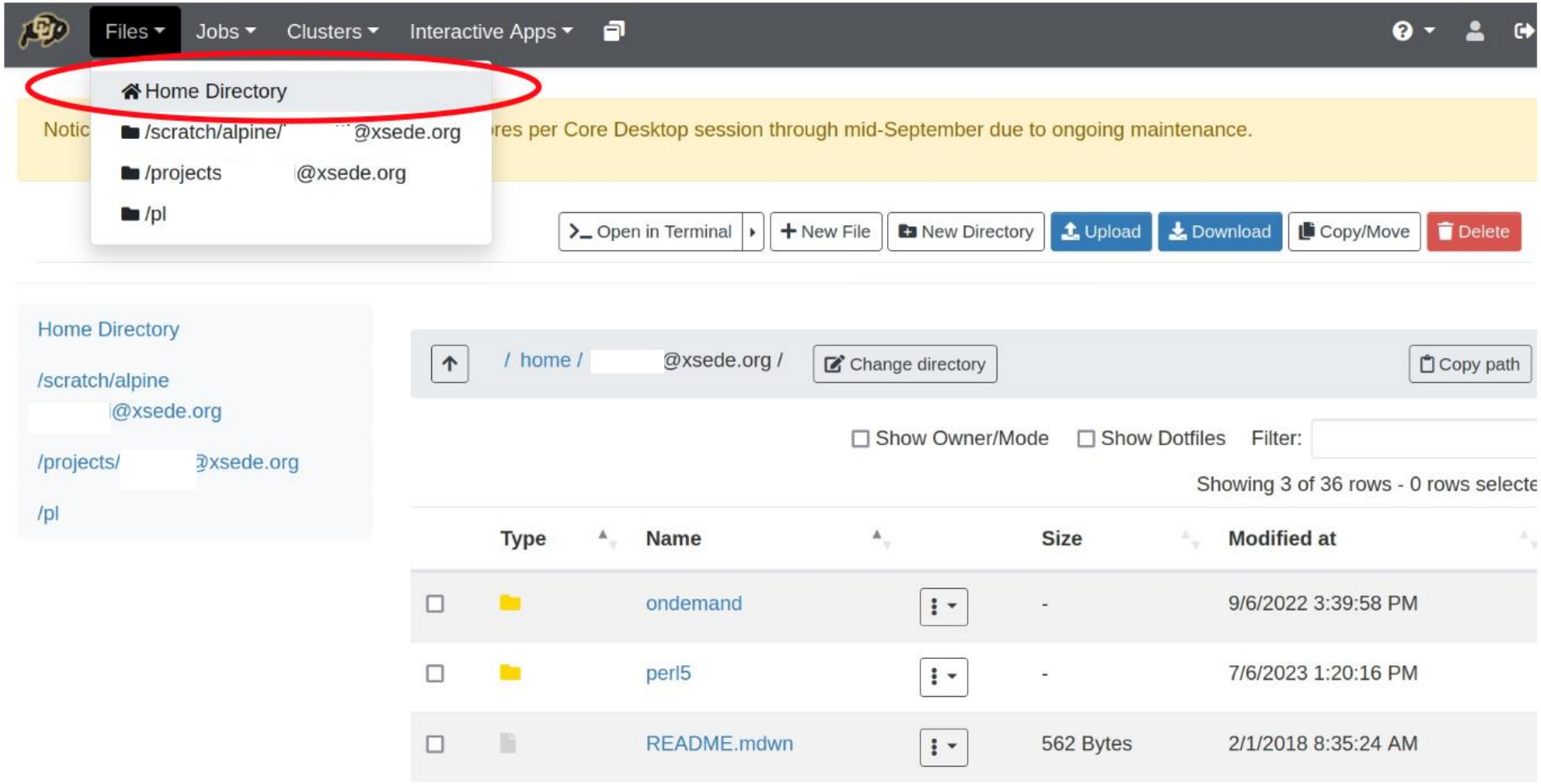
```
pkgs_dirs:
  - /projects/$USER/.conda_pkgs
envs_dirs:
  - /projects/$USER/software/anaconda/envs
```

This only every needs to be done 1 time for the lifetime of your Alpine account!

Option A: Things we can do once we are interactively inside of a compute node...

- **Through a** miniforge or anaconda environment

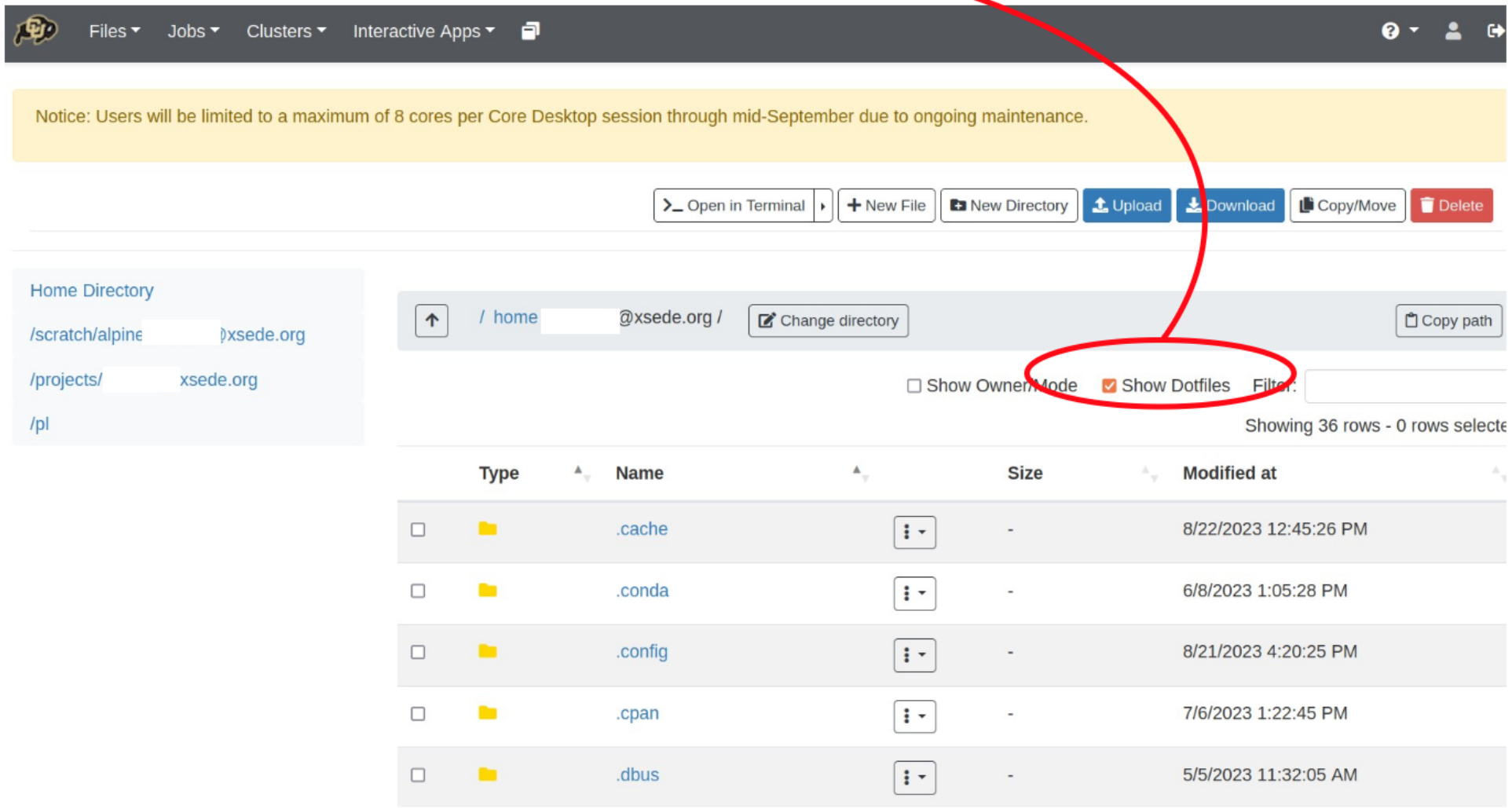
1 Navigate to your Home Directory within your OnDemand session:



Option A: Things we can do once we are interactively inside of a compute node...

- **Through a** miniforge or anaconda environment

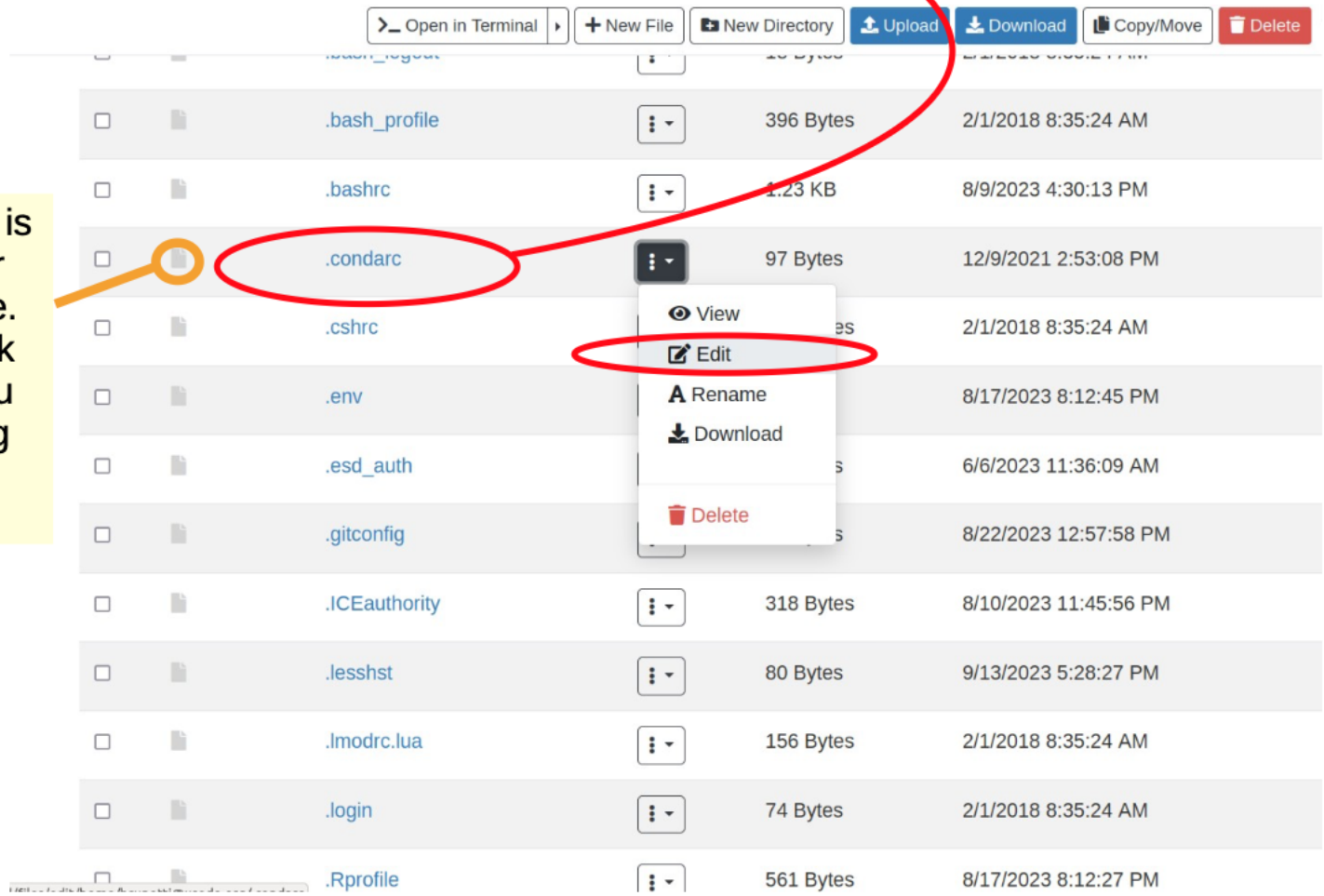
2 Make sure to click on the box that say “Show Dotfiles”



Option A: Things we can do once we are interactively inside of a compute node...

- **Through a** miniforge or anaconda environment

3 Scrolls down your list and find the file called .condarc and click the three dots and select Edit from the drop down menu

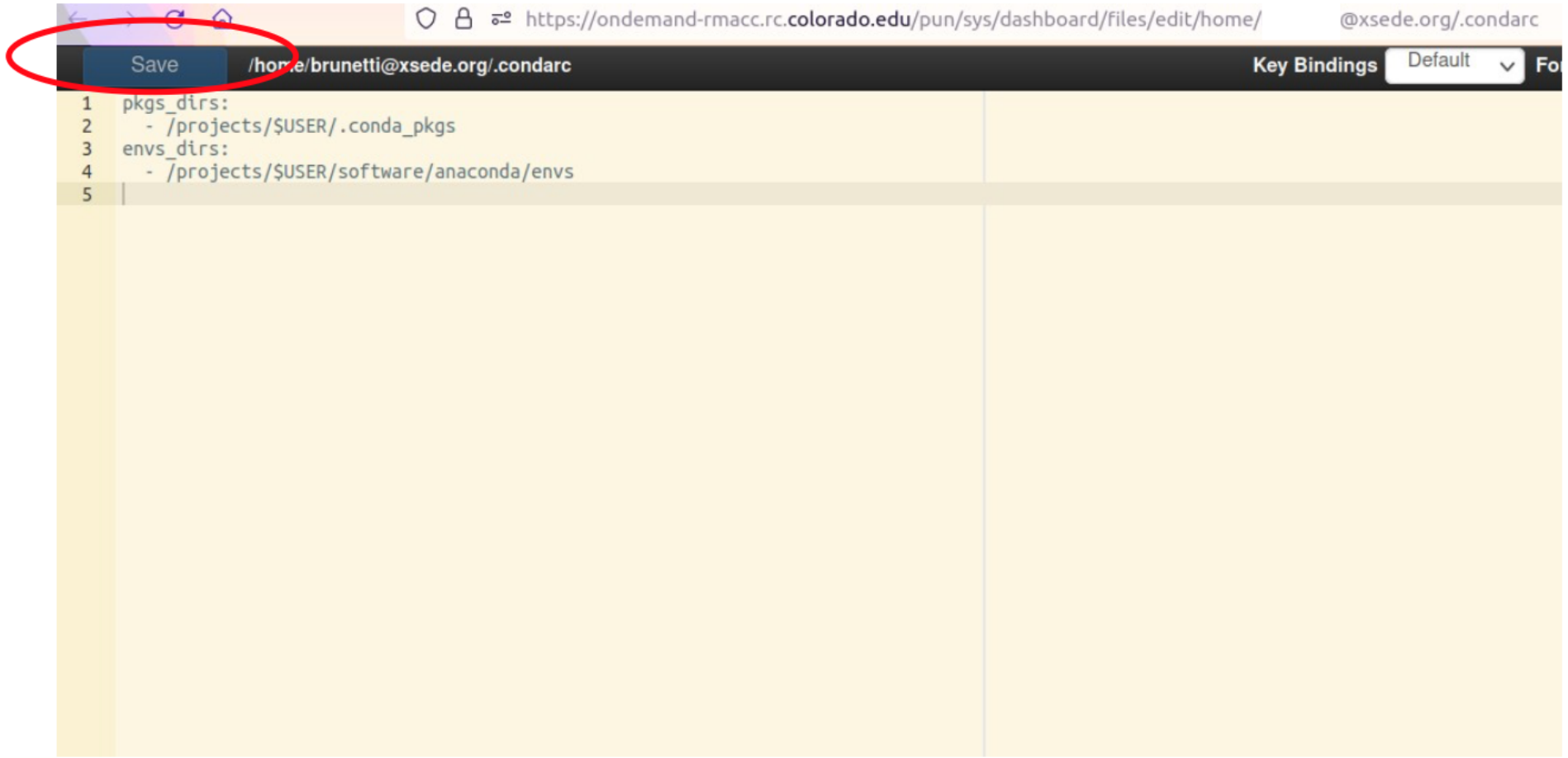


Be careful! There is a .condarc folder and a .condarc file. Make sure to click the file, which you can tell by seeing the little paper image to the left

Option A: Things we can do once we are interactively inside of a compute node...

- **Through a** miniforge or anaconda environment

4 This takes you to a text editor where you can copy and paste the following lines. Then be sure to press Save in the upper left hand corner



Option A: Things we can do once we are interactively inside of a compute node...

- **Through a** miniforge or anaconda environment

5 After you press save, you can exit the browser tab and you are done configuring conda!

Option A: Things we can do once we are interactively inside of a compute node...

- **Through a** miniforge or anaconda environment

5 After you press save, you can exit the browser tab and you are done configuring conda!

Now that conda is configured, let's go through a tutorial of how to install software that has a conda installation option.

Let's try to install the commonly used bioinformatics package, RSEM, which is a way to count/quantify reads mapping to a genome.

Option A: Things we can do once we are interactively inside of a compute node...

- **Through a** miniforge or anaconda environment

5 After you press save, you can exit the browser tab and you are done configuring conda!

Now that conda is configured, let's go through a tutorial of how to install software that has a conda installation option.

Let's try to install the commonly used bioinformatics package, RSEM, which is a way to count/quantify reads mapping to a genome.

Please refer to the miniforge guide for more details: https://github.com/kf-cuanschutz/CU-Anschutz-HPC-documentation/blob/main/miniforge_migration.md

Miniforge and package installation

- We will cover this topic more in depth during the workshop: “Alpine module stack and miniforge” on 9/15/25

- We will cover containers as an alternative way to install packages during the workshop: “How to use Containers on Alpine” on 9/29/25.

- We also have an existing guide here: https://github.com/kf-cuanschutz/CU-Anschutz-HPC-documentation/blob/main/Workshops/Container_lab_workshop_040125.pdf

Option A: Things we can do once we are interactively inside of a compute node...

- Check what software is installed on Alpine

- Install our own software

- Run software and code interactively

Option A: Things we can do once we are interactively inside of a compute node...

- **Run software and code interactively**

Once you are in a compute node you can also run code interactively, similar to how you have been doing on our command line. You may have noticed that Alpine also has several programming languages already installed that can be run on the command line and any other software (STAR, cutadapt, fastqc, Python, R, Ruby, Perl, C/C++, Java, Matlab, Mathematica, Julia, etc...)

module avail

```
------------------------------------------------------------ /curc/sw/modules/slurm ------------------------------------------
   StdEnv (L,D)    curc-quota/latest (L,D)    slurm/alpine (L,D)    slurm/blanca    slurm/core    slurmtools (D)

------------------------------------------------------- /usr/share/lmod/lmod/modulefiles/Core -------------------------------
   StdEnv    lmod (D)    settarg (D)

---------------------------------------------------------------- Compilers ---------------------------------------------------
   aocc/3.1.0 (D)    gcc/11.2.0        intel/2022.1.2     (m)    nvhpc_sdk/2022.229        nvhpc_sdk/2025.255
   aocc/3.2.0        gcc/13.2.0        intel/2024.2.1     (D)    nvhpc_sdk/2023.233 (D)
   gcc/10.3.0        gcc/14.2.0 (D)    nvhpc_sdk/2021.213        nvhpc_sdk/2025.251

------------------------------------------------------------ Independent Applications ----------------------------------------
   R/3.6.3                    emacs/27.2                 mathematica/11.1.0  (D)    rocm/5.5.0           (g)
   R/4.2.2                    emacs/30.1            (D)  matlab/R2018b              rocm/5.6.0           (g)
   R/4.4.0               (D)  expat/2.1.1                matlab/R2019b              rocm/6.1.0           (g,D)
   allinea/6.0.4         (m)  expat/2.3.0           (D)  matlab/R2020b              ruby/2.3.1
   anaconda/2020.11           ffmpeg/4.4                 matlab/R2021b         (D)  ruby/3.0.0           (D)
   anaconda/2022.10           gdb/8.1                    matlab/R2022b              singularity/3.6.4    (D)
   anaconda/2023.09      (D)  gdb/10.1              (D)  matlab/R2023b              singularity/3.7.4
   antlr/4.13.1               ghostscript/9.56.0         matlab/R2024b              slurmtools/0.0.1
   arm-forge/19.1.3      (m)  git-lfs/3.1.2              maven/3.8.1                spack/0.20.1
   autotools/2.69             git/2.31.0                 miniforge/24.11.3-0        subversion/1.8.16
   autotools/2.71        (D)  gmsh/2.16.0                ncl/6.3.0                  subversion/1.10.2
   chimerax/1.2.5             gmsh/4.11.1           (D)  nco/4.8.1                  subversion/1.14.1    (D)
   clustershell/1.9.2         gnu_parallel/20160622      papi/5.4.3                 swig/4.1.1
   cmake/3.5.2                gnu_parallel/20210322 (D)  papi/5.5.1            (D)  tcltk/8.6.5
   cmake/3.9.2                gnuplot/5.4.3              paraview/5.0.1             tcltk/8.6.11
   cmake/3.14.1               idl/8.7                    paraview/5.6.0             tcltk/9.0.1          (D)
   cmake/3.20.2               imagemagick/6.9.12         paraview/5.8.0             tdom/0.8.3
   cmake/3.25.0               jdk/1.7.0                  paraview/5.9.0             tdom/0.9.2           (D)
   cmake/3.27.7               jdk/1.8.0_91               paraview/5.10.0       (D)  texlive/2021
--More--
```

Option A: Things we can do once we are interactively inside of a compute node...

- **Run software and code interactively**

Let's say I wanted to test out some code in R in real-time. I can go inside of a compute node, load the programming module and start it by typing in the name of the programming language/software.

```
module load R/ 4.4.0
```

Followed by entering:

```
R
```

```
R version 4.2.2 (2022-10-31) -- "Innocent and Trusting"
Copyright (C) 2022 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

Now we are in an R session! We can install libraries, load libraries, write code and it will run; I am not going to talk about this much more in that this is really only good for testing since if you close your computer, turn it off etc… it will kill your session.

Option A: Things we can do once we are interactively inside of a compute node...

- **Run software and code interactively**

Let's say I wanted to test out some code in R in real-time. I can go inside of a compute node, load the programming module and start it by typing in the name of the programming language/software.

```
module load R/ 4.4.0
```

Do not use an R version lower than 4.4.0!

Followed by entering:

```
R
```

```
R version 4.2.2 (2022-10-31) -- "Innocent and Trusting"
Copyright (C) 2022 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

Now we are in an R session! We can install libraries, load libraries, write code and it will run; I am not going to talk about this much more in that this is really only good for testing since if you close your computer, turn it off etc… it will kill your session.

R on Alpine

- We will cover this topic more in depth during the workshop: “How to use R on Alpine” on 9/22/25

- We also have an existing guide here: https://github.com/kf-cuanschutz/CU-Anschutz-HPC-documentation/blob/main/Workshops/R_guide_on_Alpine_v2.pdf

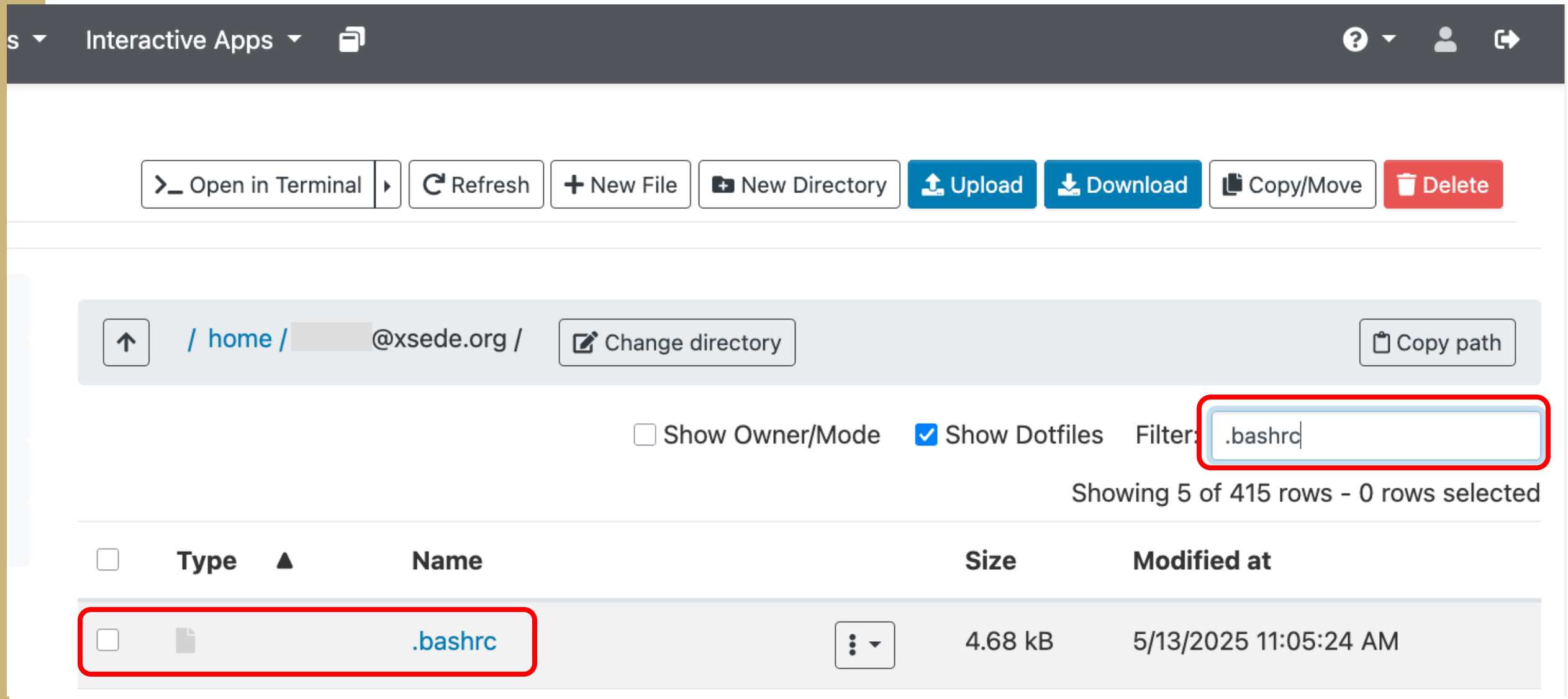
- Rstudio package installation is also covered here: https://github.com/kf-cuanschutz/CU-Anschutz-HPC-documentation/tree/main/Rstudio_related_scripts

Best practices on Alpine

- /home or /tmp have a very small size. Please always include the following in your $HOME/.bashrc or Slurm script.

```
export TMP=/gpfs/alpine1/scratch/$USER/cache_dir
mkdir -pv $TMP
export TEMP=$TMP
export TMPDIR=$TMP
export TEMPDIR=$TMP
export PIP_CACHE_DIR=$TMP
```

Best practices on Alpine



University of Colorado
Anschutz Medical Campus

Best practices on Alpine

Save /home/[redacted]@xsede.org/.bashrc

```
124  export TMP=/gpfs/alpine1/scratch/$USER/cache_dir
125  mkdir -pv $TMP
126  export TEMP=$TMP
127  export TMPDIR=$TMP
128  export TEMPDIR=$TMP
129  export PIP_CACHE_DIR=$TMP
```

Best practices on Alpine

- In general, always make sure to check your .cache in $HOME to make sure you are not running out of space.

- To clean up your .cache space, using the shell please run the following:

```
rm -Rv $HOME/.cache/*
```

Username aliasing

- Some packages that use perl won't install or run Alpine due to the @xsede.org username format.

- Please follow this guide if that happens to you:

  https://curc.readthedocs.io/en/latest/additional-resources/csu-xsede-usernames.html

Now you are on the head/login node. What are your options?

Option A

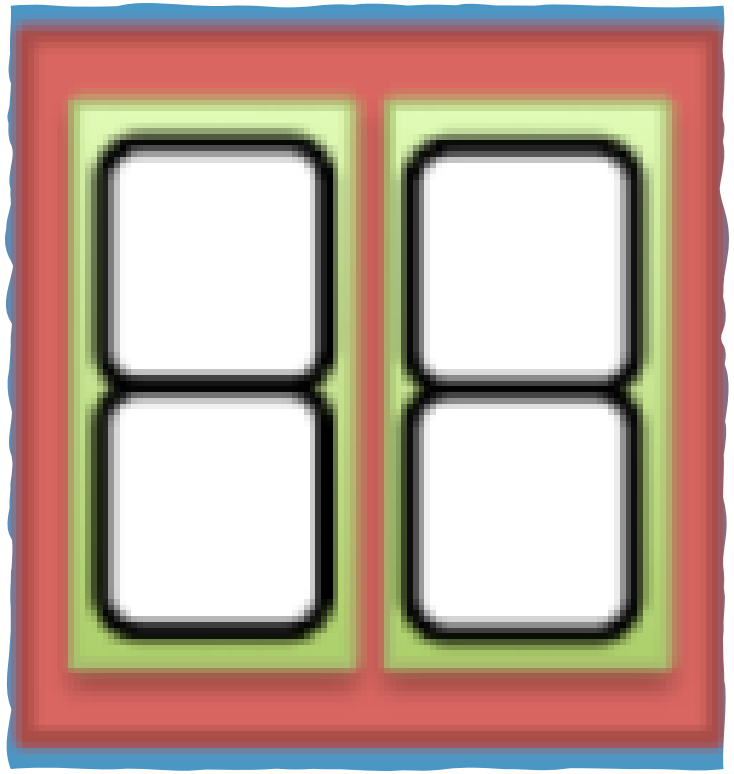
Make a request to the the head node to login to a compute node so you can interactively run your code on a compute node, test code in real-time, or look at what software is installed on Alpine.

Option B

Stay on the head node because you want to submit a job remotely to a compute node.

Elements of computing

Parts of the processing chip

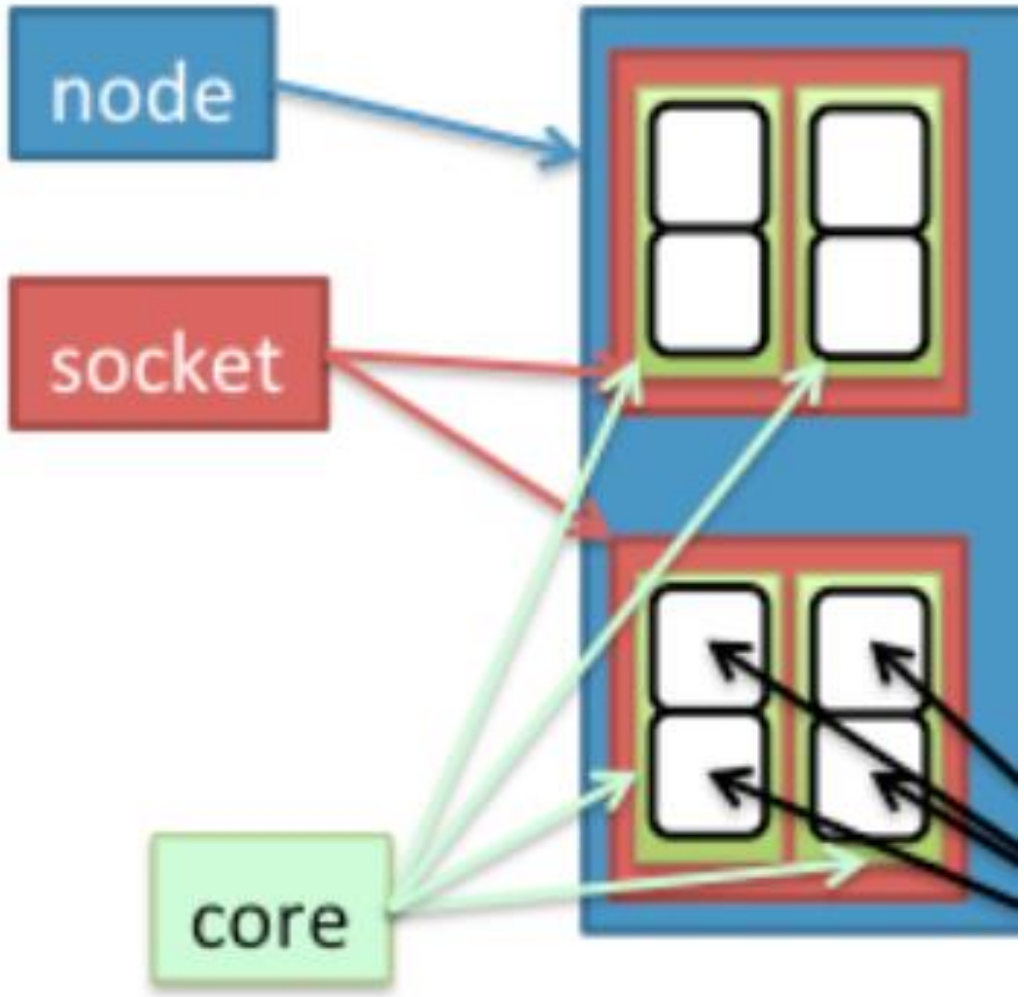


- A socket can be considered an entire computing chip in modern computers.
- Contains more than 1 core.
- On supercomputers a node usually contains more than 1 socket.

Elements of computing

A core contains an Arithmetic and Logic Unit **(ALU)** + Registers

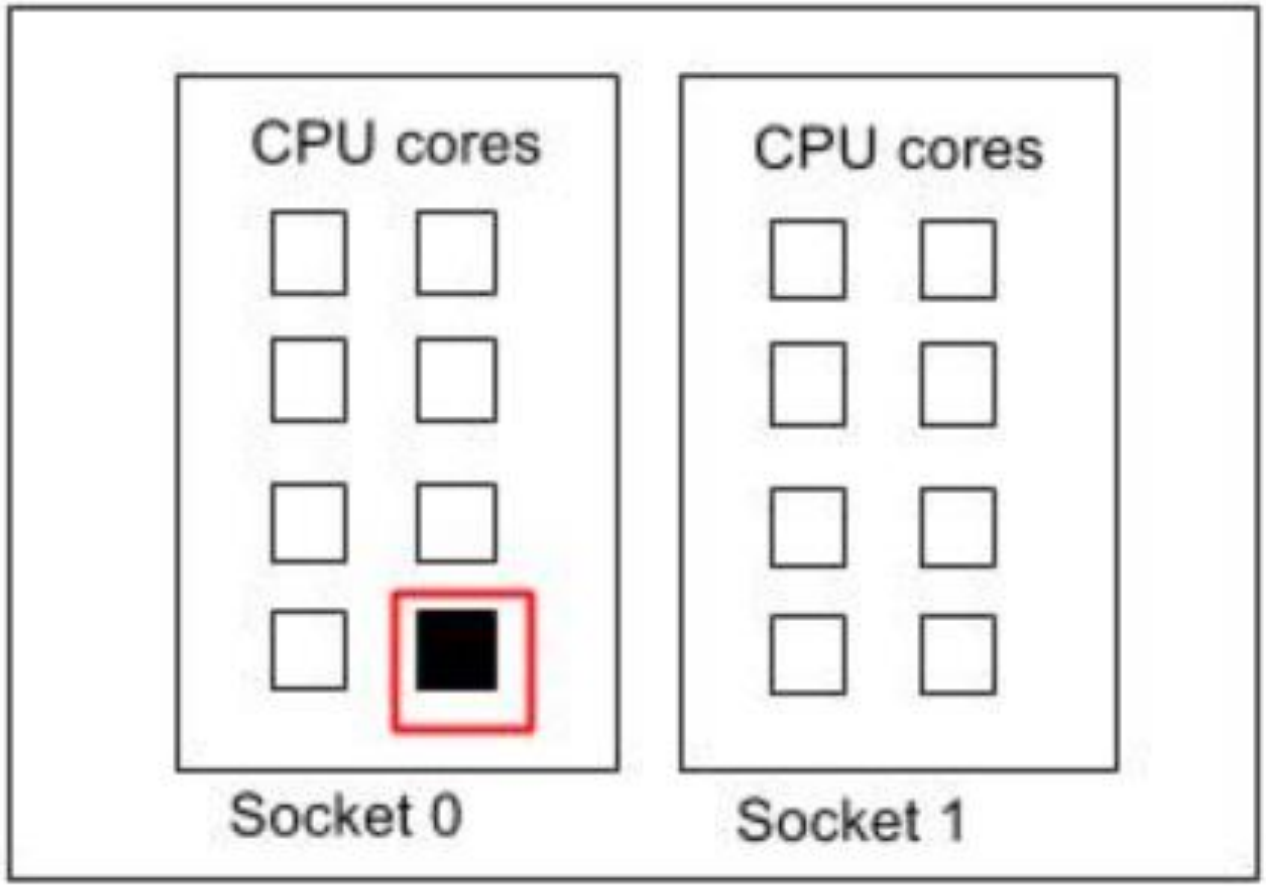
Parts of the processing chip



Source: https://hpc.nih.gov/apps/swarm.html

Elements of computing

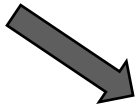
Parts of the processing chip



Source: https://hpc.nih.gov/apps/swarm.html

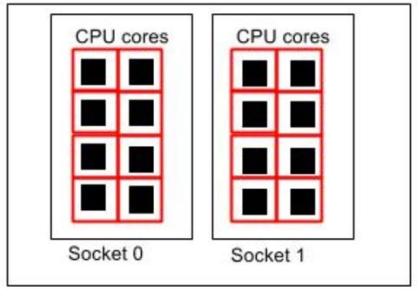
How does Slurm work internally?

- Bulma request 8 cores from Node 1

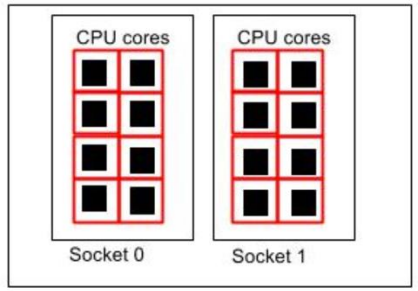


Slurm scheduler

Node 1

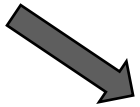


Node 2



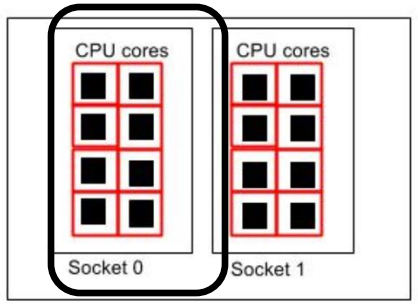
How does Slurm work internally?

- Bulma request 8 cores from Node 1

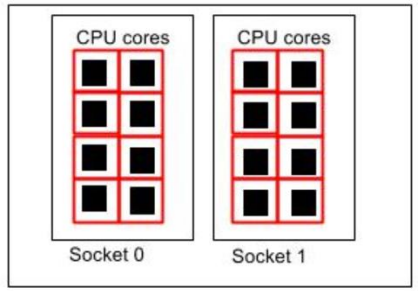


Slurm scheduler

Node 1

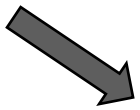


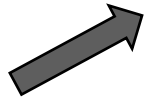
Node 2



How does Slurm work internally?

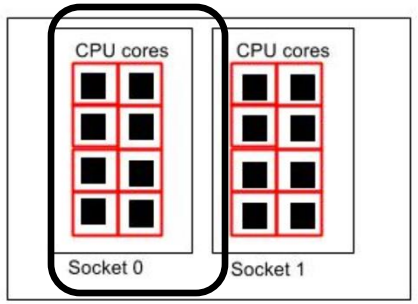
- Bulma request 8 cores from Node 1
- Dr. Brief requests 8 cores as well from Node 1
- Gohan requests 16 cores from Node 2
- No tragedy of the common!!!



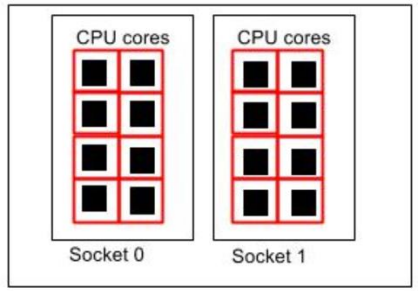


Slurm scheduler

Node 1

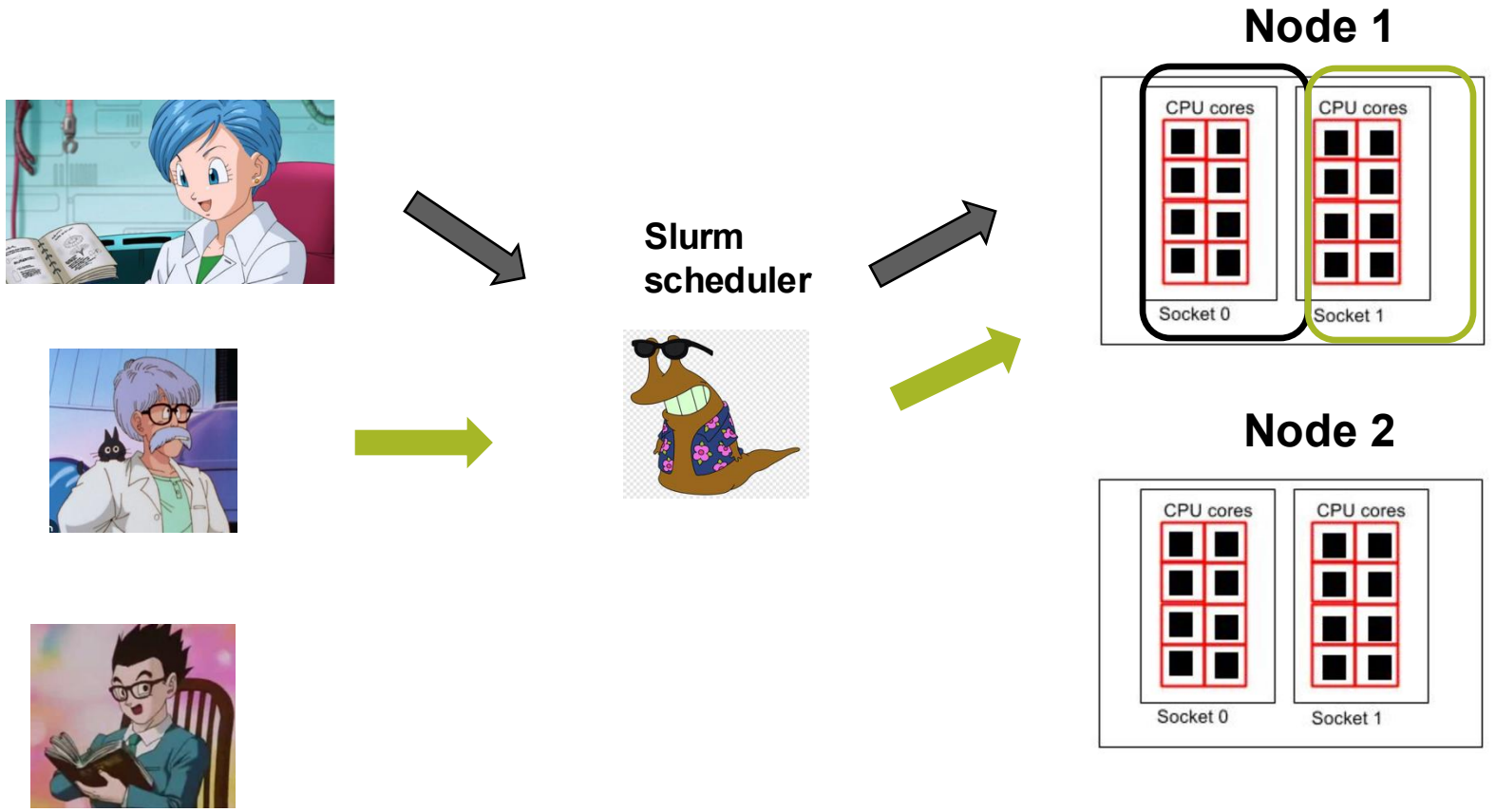


Node 2



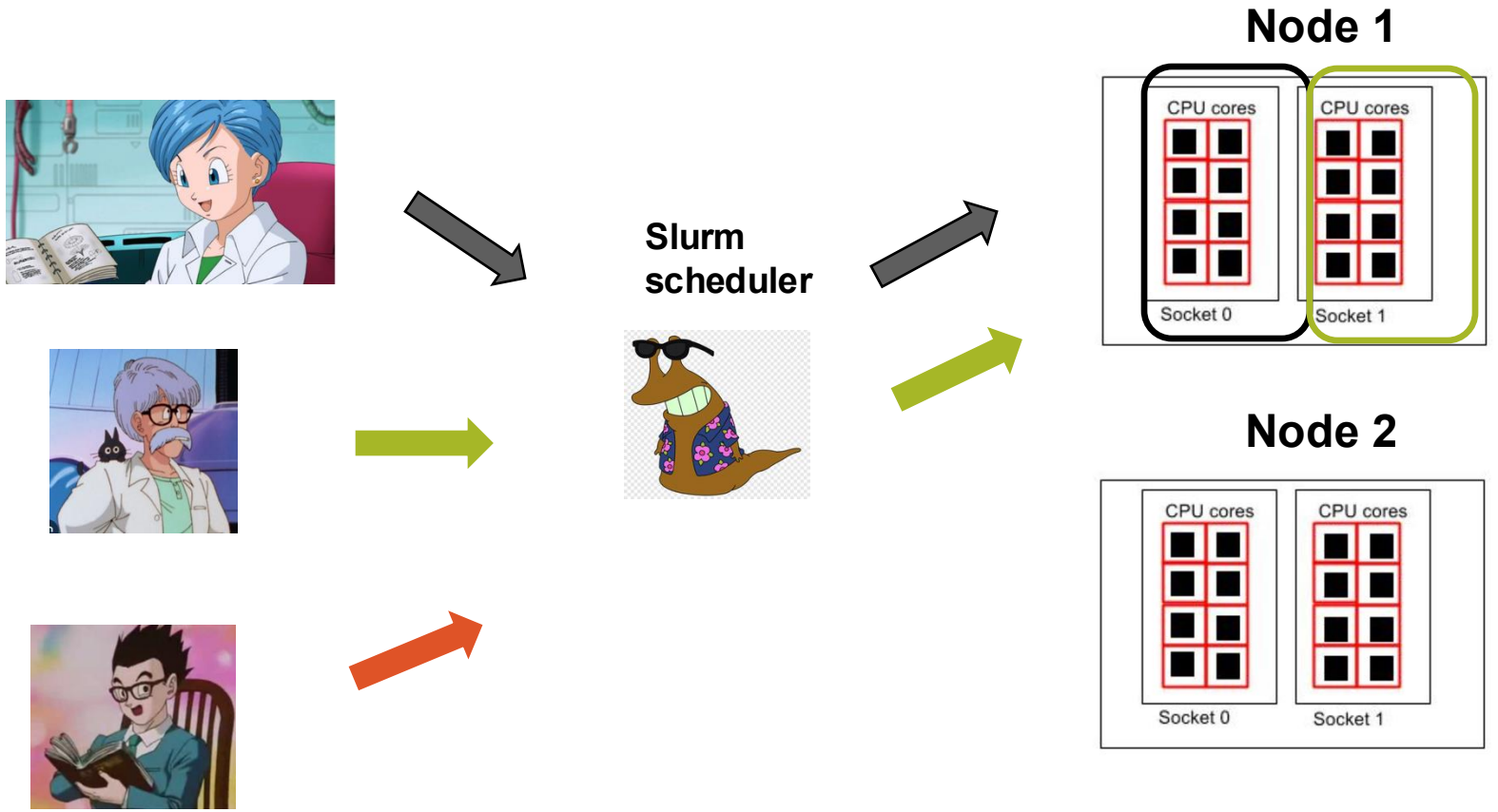
How does Slurm work internally?

- Bulma request 8 cores from Node 1
- Dr. Brief requests 8 cores as well from Node 1



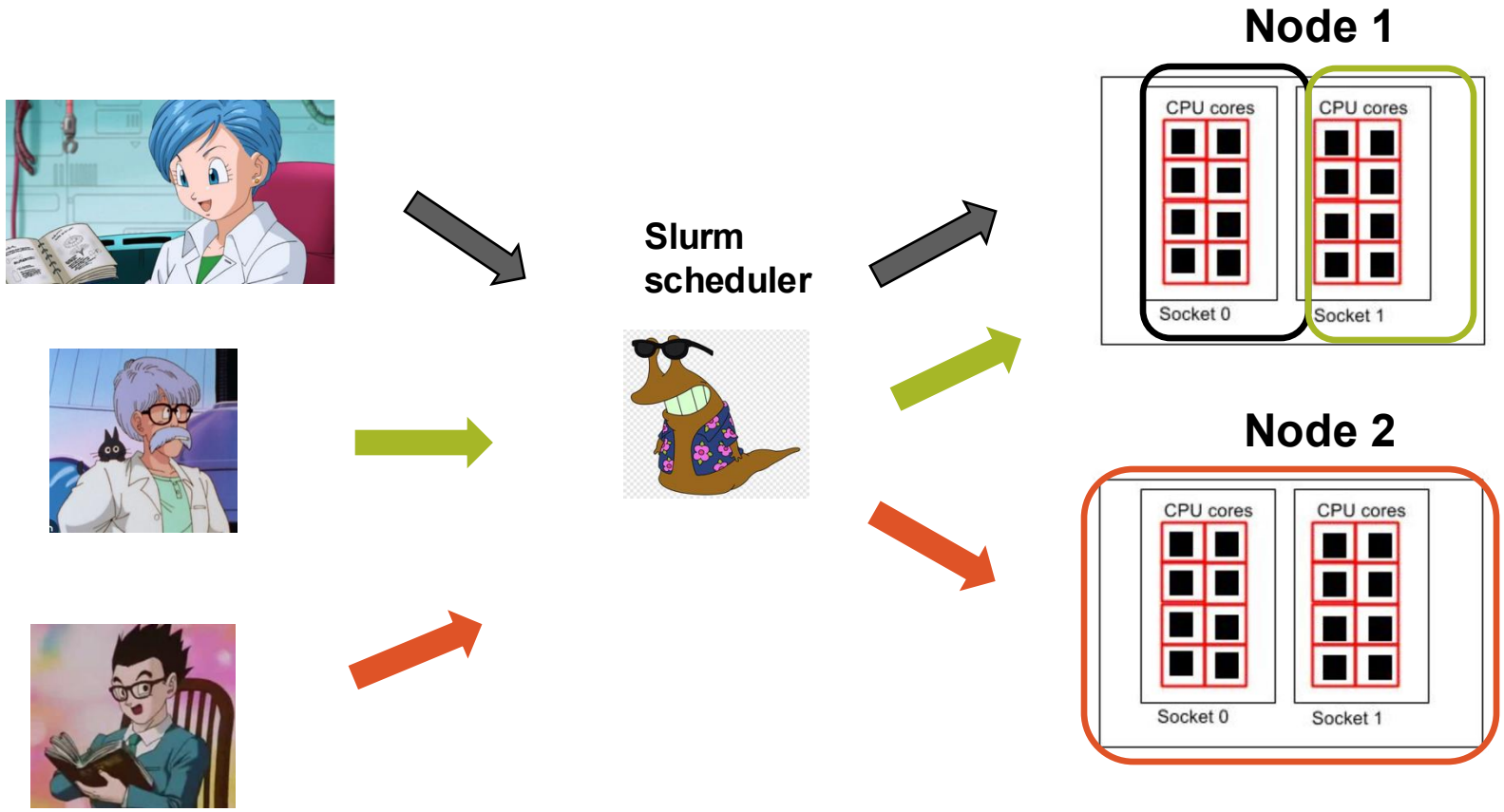
How does Slurm work internally?

- Bulma request 8 cores from Node 1
- Dr. Brief requests 8 cores as well from Node 1
- Gohan requests 16 cores from Node 2



How does Slurm work internally?

- Bulma request 8 cores from Node 1
- Dr. Brief requests 8 cores as well from Node 1
- Gohan requests 16 cores from Node 2
- No tragedy of the common!!!



Build a Slurm script

- Create the empty slurm script and modify it with your editor of choice (nano, vim, emac, Ondemand etc …)

```
[       @xsede.org@login-ci2 openmpi_hostname]$ touch hello_mpi.sh
 [      @xsede.org@login-ci2 openmpi_hostname]$
```

Build a Slurm script

```
#!/bin/bash

#SBATCH --partition=amilan
#SBATCH --job-name=first_mpirun
#SBATCH --output=first_mpirun-job.%j.out
#SBATCH --error=first_mpirun-job.%j.err
#SBATCH --account=amc-general
#SBATCH --qos=normal
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mail-type=ALL
#SBATCH --mail-user=
#SBATCH --time=00:00:01

module load gcc
module load openmpi

mpirun -n $SLURM_NTASKS hostname
```

Main CPU partition name

Partitions on Alpine

- amilan -> CPU
- amem -> High memory
- aa100 -> NVIDIA GPU partition
- al40 -> NVIDIA GPU partition
- ami100 -> AMD GPU partition

Build a Slurm script

```
#!/bin/bash

#SBATCH --partition=amilan
#SBATCH --job-name=first_mpirun
#SBATCH --output=first_mpirun-job.%j.out
#SBATCH --error=first_mpirun-job.%j.err
#SBATCH --account=amc-general
#SBATCH --qos=normal
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mail-type=ALL
#SBATCH --mail-user=
#SBATCH --time=00:00:01

module load gcc
module load openmpi

mpirun -n $SLURM_NTASKS hostname
```

Job name

Build a Slurm script

```
#!/bin/bash

#SBATCH --partition=amilan
#SBATCH --job-name=first_mpirun
#SBATCH --output=first_mpirun-job.%j.out
#SBATCH --error=first_mpirun-job.%j.err
#SBATCH --account=amc-general
#SBATCH --qos=normal
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mail-type=ALL
#SBATCH --mail-user=
#SBATCH --time=00:00:01

module load gcc
module load openmpi

mpirun -n $SLURM_NTASKS hostname
```

- Job output file.
- Similar to stdout in linux.

Build a Slurm script

```
#!/bin/bash

#SBATCH --partition=amilan
#SBATCH --job-name=first_mpirun
#SBATCH --output=first_mpirun-job.%j.out
#SBATCH --error=first_mpirun-job.%j.err
#SBATCH --account=amc-general
#SBATCH --qos=normal
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mail-type=ALL
#SBATCH --mail-user=
#SBATCH --time=00:00:01

module load gcc
module load openmpi

mpirun -n $SLURM_NTASKS hostname
```

%j for the jobID number

Build a Slurm script

```
#!/bin/bash

#SBATCH --partition=amilan
#SBATCH --job-name=first_mpirun
#SBATCH --output=first_mpirun-job.%j.out
#SBATCH --error=first_mpirun-job.%j.err
#SBATCH --account=amc-general
#SBATCH --qos=normal
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mail-type=ALL
#SBATCH --mail-user=
#SBATCH --time=00:00:01

module load gcc
module load openmpi

mpirun -n $SLURM_NTASKS hostname
```

Job error file.

Build a Slurm script

```
#!/bin/bash

#SBATCH --partition=amilan
#SBATCH --job-name=first_mpirun
#SBATCH --output=first_mpirun-job.%j.out
#SBATCH --error=first_mpirun-job.%j.err
#SBATCH --account=amc-general
#SBATCH --qos=normal
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mail-type=ALL
#SBATCH --mail-user=
#SBATCH --time=00:00:01

module load gcc
module load openmpi

mpirun -n $SLURM_NTASKS hostname
```

Account name. All AMC users should use the amc-general account!

Build a Slurm script

```
#!/bin/bash

#SBATCH --partition=amilan
#SBATCH --job-name=first_mpirun
#SBATCH --output=first_mpirun-job.%j.out
#SBATCH --error=first_mpirun-job.%j.err
#SBATCH --account=amc-general
#SBATCH --qos=normal
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mail-type=ALL
#SBATCH --mail-user=
#SBATCH --time=00:00:01

module load gcc
module load openmpi

mpirun -n $SLURM_NTASKS hostname
```

Quality of Service

Slurm Quality of service (qos)

- Used to modify or constrain characteristics that a job can have.
- **--qos=normal** corresponds to a walltime of 24 hours and is the default.
- **--qos=long** corresponds to a walltime of up to 7 days
- **--qos=mem** corresponds to high memory jobs only (up to 2TB)

Slurm Quality of service (qos)

- Used to modify or constrain characteristics that a job can have.
- **--qos=normal** corresponds to a walltime of 24 hours and is the default.
- **--qos=long** corresponds to a walltime of up to 7 days
- **--qos=mem** corresponds to high memory jobs only (up to 2TB)
- It is now mandatory to use #SBATCH --qos when interacting with Slurm!

Build a Slurm script

```
#!/bin/bash

#SBATCH --partition=amilan
#SBATCH --job-name=first_mpirun
#SBATCH --output=first_mpirun-job.%j.out
#SBATCH --error=first_mpirun-job.%j.err
#SBATCH --account=amc-general
#SBATCH --qos=normal
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mail-type=ALL
#SBATCH --mail-user=
#SBATCH --time=00:00:01

module load gcc
module load openmpi

mpirun -n $SLURM_NTASKS hostname
```

Number of nodes. Always use 1 unless you are using MPI, gnu parallel or sparkcluster!

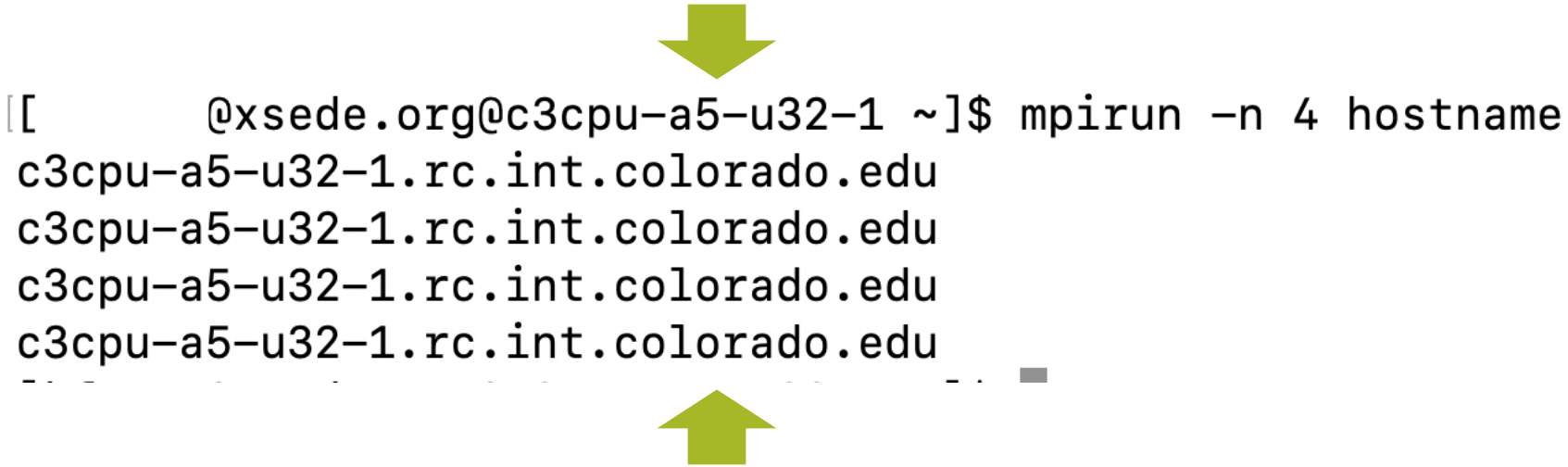
Build a Slurm script

```
#!/bin/bash

#SBATCH --partition=amilan
#SBATCH --job-name=first_mpirun
#SBATCH --output=first_mpirun-job.%j.out
#SBATCH --error=first_mpirun-job.%j.err
#SBATCH --account=amc-general
#SBATCH --qos=normal
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mail-type=ALL
#SBATCH --mail-user=
#SBATCH --time=00:00:01

module load gcc
module load openmpi

mpirun -n $SLURM_NTASKS hostname
```

Number of cores requested.

Slurm example on number of cores

- hostname is just a program that prints the name of the node I am working with



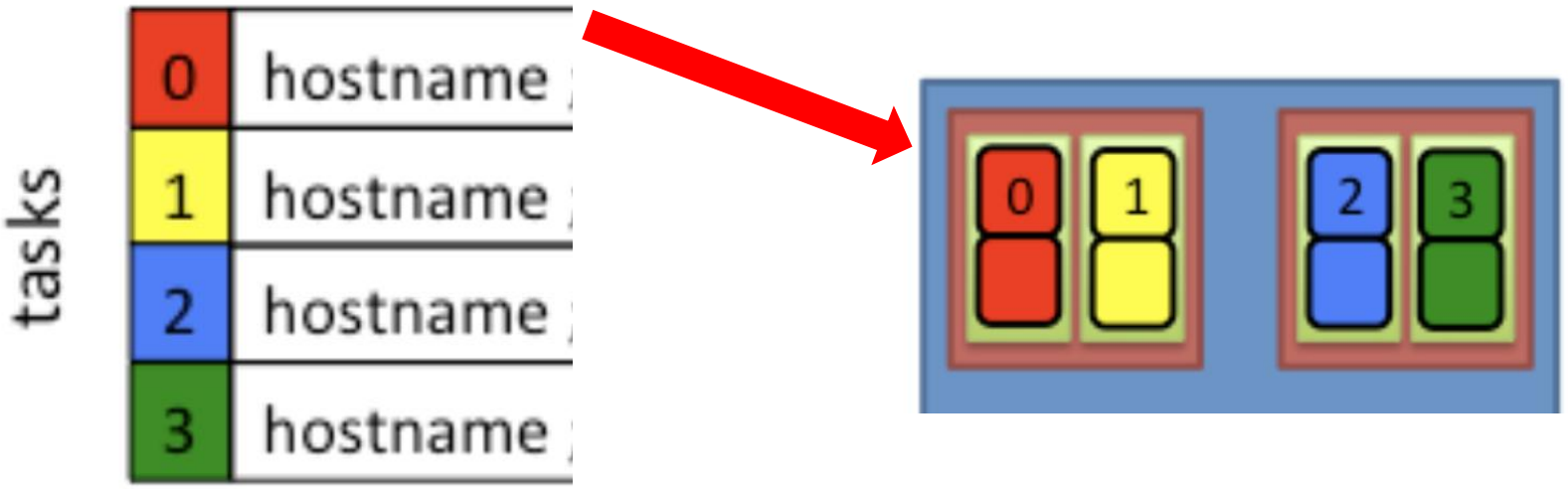
```
[      @xsede.org@c3cpu-a5-u32-1 ~]$ mpirun -n 4 hostname
c3cpu-a5-u32-1.rc.int.colorado.edu
c3cpu-a5-u32-1.rc.int.colorado.edu
c3cpu-a5-u32-1.rc.int.colorado.edu
c3cpu-a5-u32-1.rc.int.colorado.edu
```

- With mpirun, we schedule 4 tasks on the 4 cores,
  thus the name of the name of the node gets printed 4 times !!!

Slurm example on number of cores

Recap of what we did

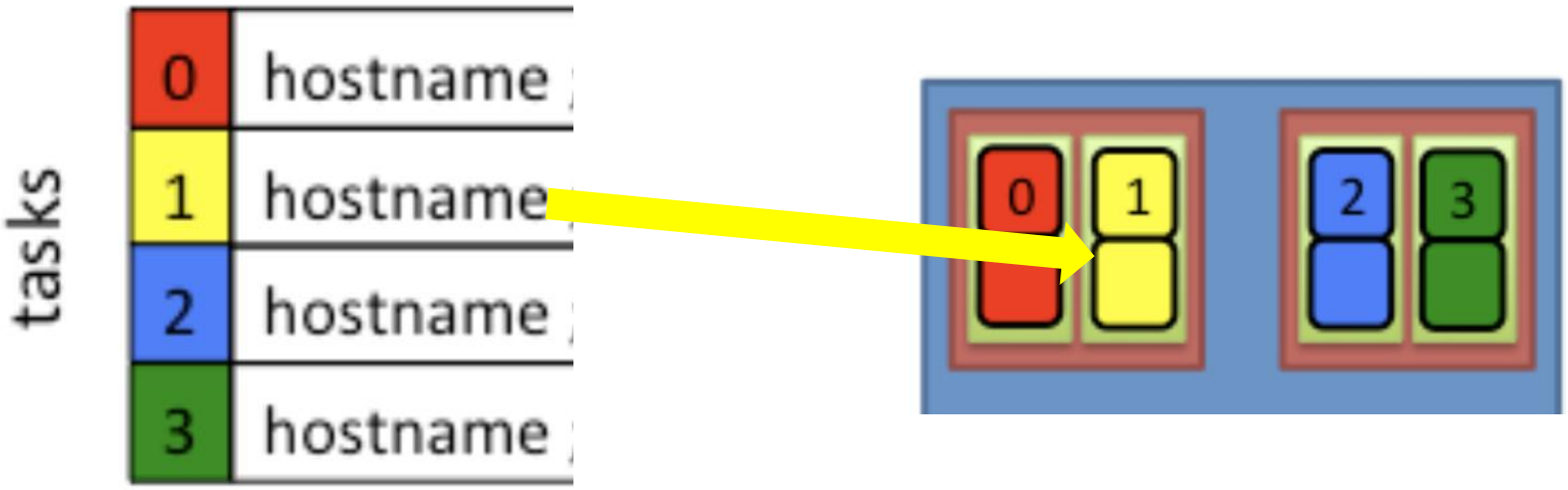
mpirun –n 4 hostname



Slurm example on number of cores

Recap of what we did

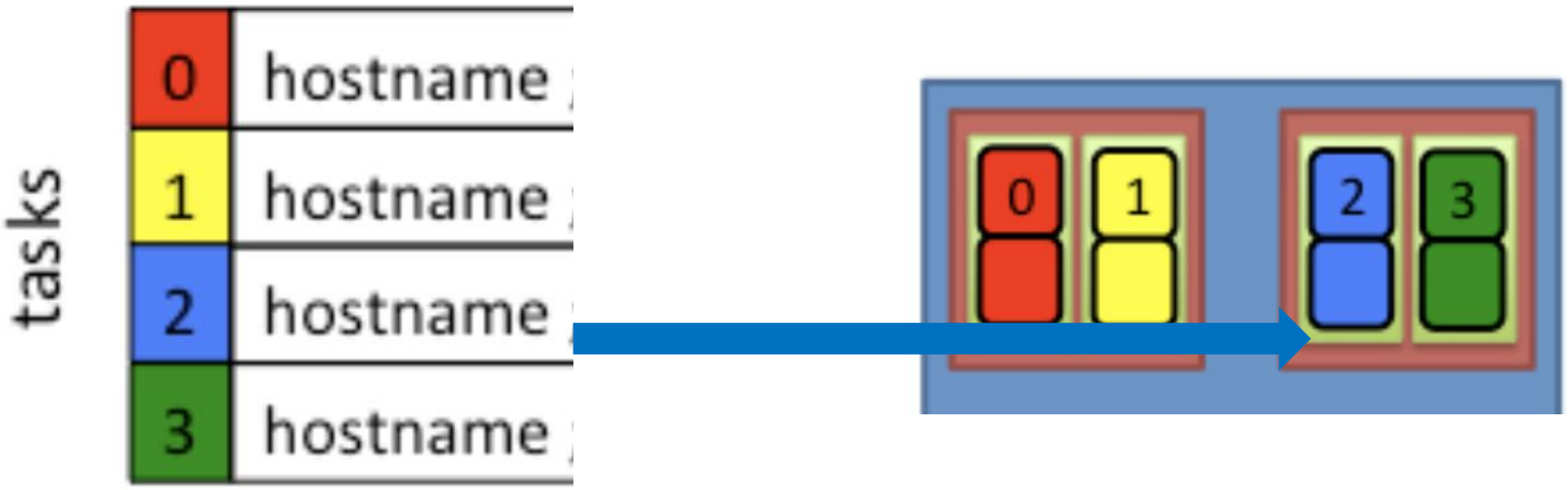
mpirun –n 4 hostname



Slurm example on number of cores

Recap of what we did

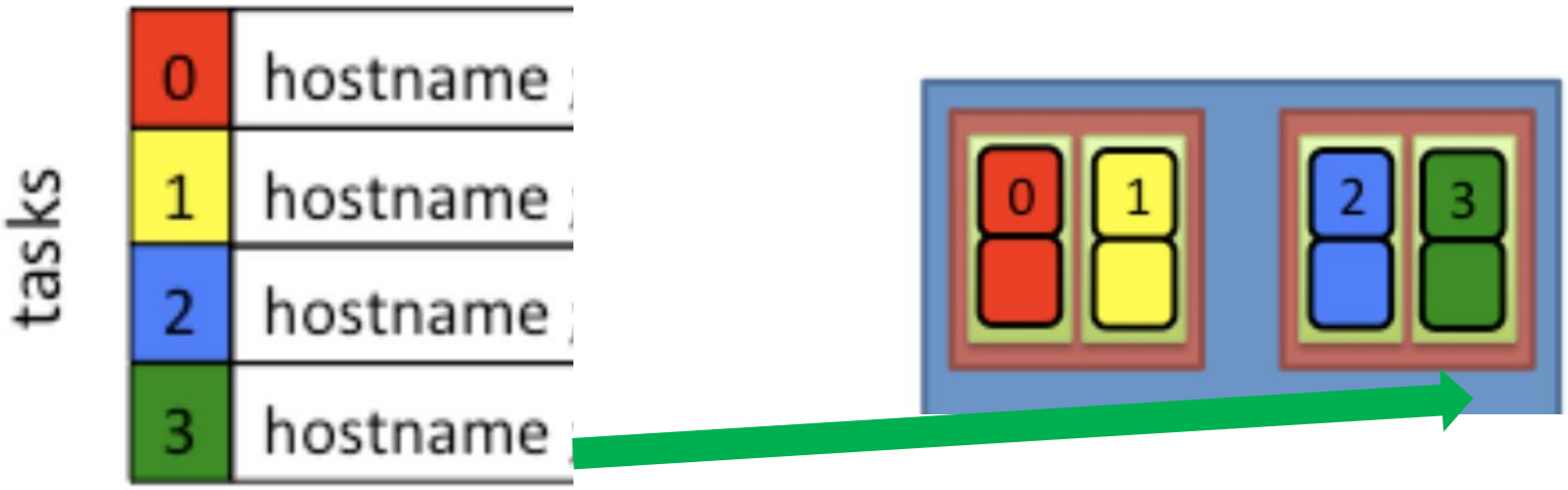
mpirun –n 4 hostname



Slurm example on number of cores

Recap of what we did

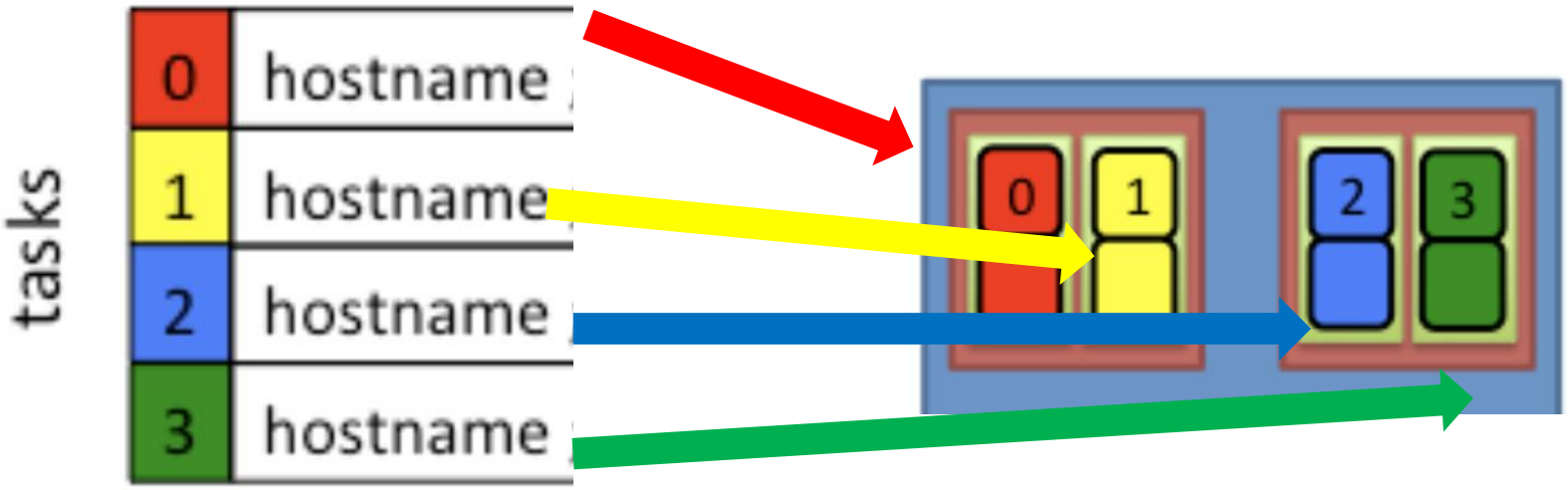
mpirun –n 4 hostname



Slurm example on number of cores

Recap of what we did

mpirun –n 4 hostname



Build a Slurm script

```
#!/bin/bash

#SBATCH --partition=amilan
#SBATCH --job-name=first_mpirun
#SBATCH --output=first_mpirun-job.%j.out
#SBATCH --error=first_mpirun-job.%j.err
#SBATCH --account=amc-general
#SBATCH --qos=normal
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mail-type=ALL
#SBATCH --mail-user=
#SBATCH --time=00:00:01

module load gcc
module load openmpi

mpirun -n $SLURM_NTASKS hostname
```

Request to receive an email notification.

Build a Slurm script

```
#!/bin/bash

#SBATCH --partition=amilan
#SBATCH --job-name=first_mpirun
#SBATCH --output=first_mpirun-job.%j.out
#SBATCH --error=first_mpirun-job.%j.err
#SBATCH --account=amc-general
#SBATCH --qos=normal
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mail-type=ALL
#SBATCH --mail-user=
#SBATCH --time=00:00:01

module load gcc
module load openmpi

mpirun -n $SLURM_NTASKS hostname
```

Duration of the slurm job.

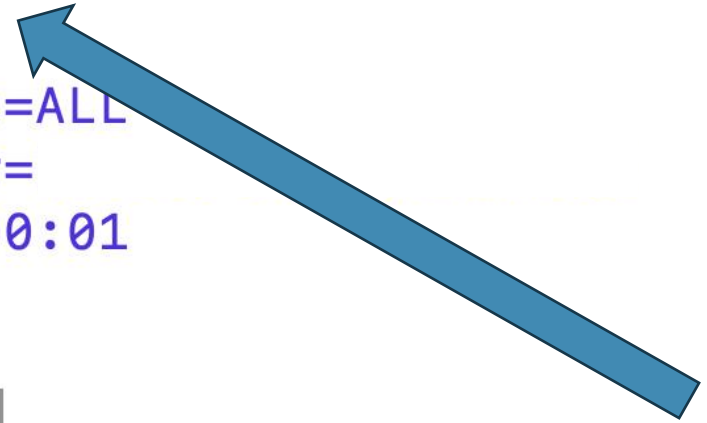
Build a Slurm script

```
#!/bin/bash

#SBATCH --partition=amilan
#SBATCH --job-name=first_mpirun
#SBATCH --output=first_mpirun-job.%j.out
#SBATCH --error=first_mpirun-job.%j.err
#SBATCH --account=amc-general
#SBATCH --qos=normal
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mail-type=ALL
#SBATCH --mail-user=
#SBATCH --time=00:00:01

module load gcc
module load openmpi

mpirun -n $SLURM_NTASKS hostname
```

Packages are not loaded automatically! You need to always load them.

Build a Slurm script

```
#!/bin/bash

#SBATCH --partition=amilan
#SBATCH --job-name=first_mpirun
#SBATCH --output=first_mpirun-job.%j.out
#SBATCH --error=first_mpirun-job.%j.err
#SBATCH --account=amc-general
#SBATCH --qos=normal
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mail-type=ALL
#SBATCH --mail-user=
#SBATCH --time=00:00:01

module load gcc
module load openmpi

mpirun -n $SLURM_NTASKS hostname
```



$SLURM_NTASKS is the slurm env variable for the 4 cores

Build a Slurm script

We use sbatch to submit the script

```
[      @xsede.org@login-ci2 openmpi_hostname]$ sbatch hello_mpi.sh
Submitted batch job 2998325
```

Build a Slurm script

We get output and error files, ending with %j.out and %j.err

```
sede.org@login-ci2 openmpi_hostname]$ ls -latr
total 184
drwxrws---. 44                                  1672 Sep 15 01:45 ..
-rw-r--r--.  1                                   400 Sep 15 01:52 hello_mpi.sh
-rw-r--r--.  1                                     0 Sep 15 01:52 first_mpirun-job.2998325.err
drwxr-sr-x.  2                                   122 Sep 15 01:52 .
-rw-r--r--.  1                                   140 Sep 15 01:52 first_mpirun-job.2998325.out
sede.org@login-ci2 openmpi_hostname]$ cat first_mpirun-job.2998325.out
c3cpu-c15-u1-2.rc.int.colorado.edu
c3cpu-c15-u1-2.rc.int.colorado.edu
c3cpu-c15-u1-2.rc.int.colorado.edu
c3cpu-c15-u1-2.rc.int.colorado.edu
sede.org@login-ci2 openmpi_hostname]$
```

Slurm job monitoring

- We will cover this topic more in depth during the workshop: “Software sustainability and efficiency on Alpine” on 10/06/25

Alpine data transfer

- Please refer to the workshop: “Globus and rclone data transfers on Alpine” given on 09/08/25 and posted here:

  https://github.com/kf-cuanschutz/CU-Anschutz-HPC-documentation/blob/main/Workshops/Globus_and_rclone_090825.pdf

- Please refer to this page as well under “Data transfers”:

  https://github.com/kf-cuanschutz/CU-Anschutz-HPC-documentation/tree/main

- Additionally, you may refer to this CU Boulder page on data transfers:

  https://curc.readthedocs.io/en/latest/compute/data-transfer.html#data-transfer

Alpine data transfer

- If you need access to ssh so that you can use scp, sftp or rsync, Open a ticket and fill out this form: https://ucdenverdata.formstack.com/forms/alpine_ssh_request_form

- We do one onboarding session per month.


University of Colorado Anschutz Medical Campus
THANK YOU