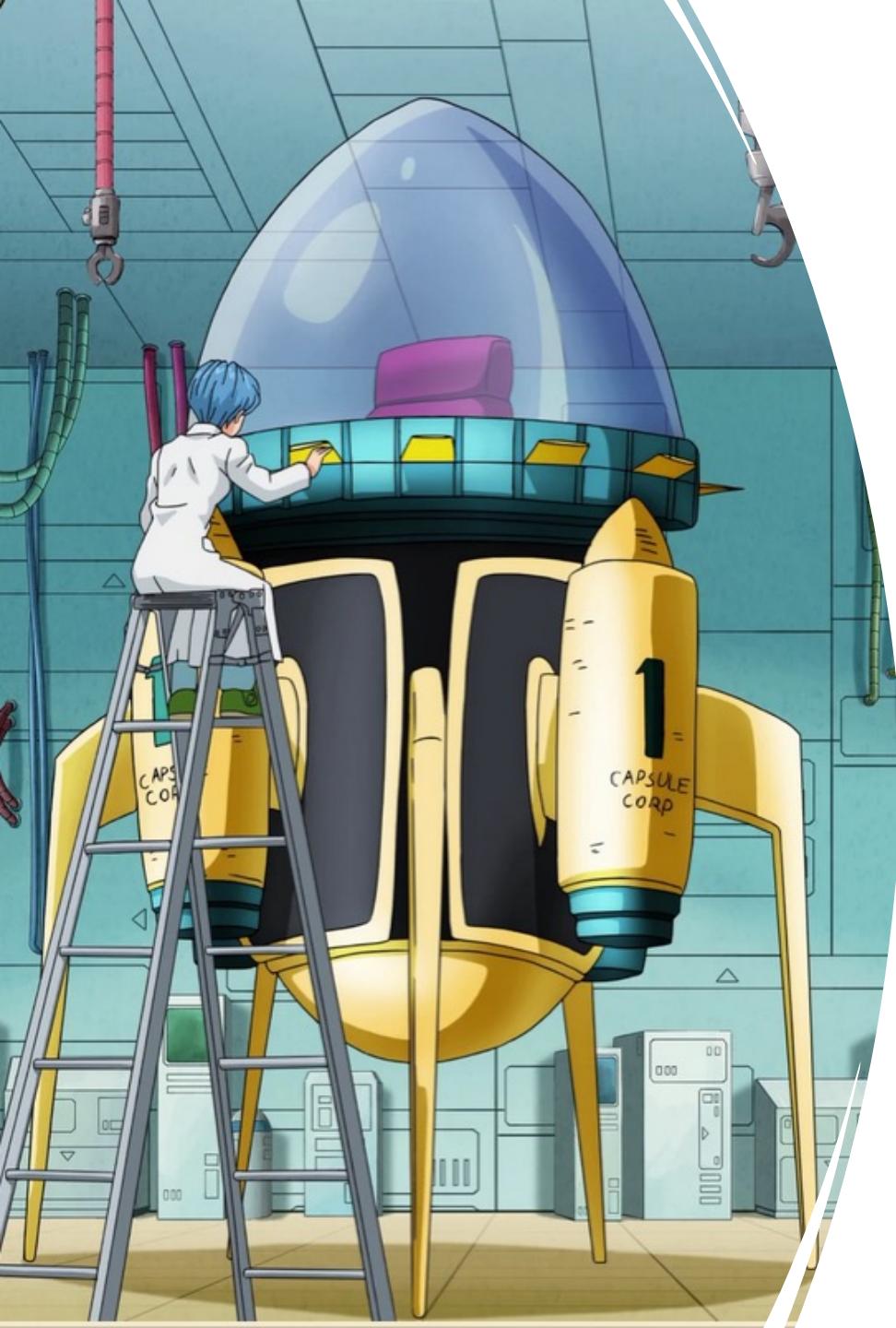


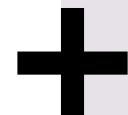
Introduction to Alpine

Kevin Fotso





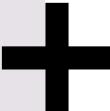
I- Part 1) Elements of computing



Parts of the processing chip (1)

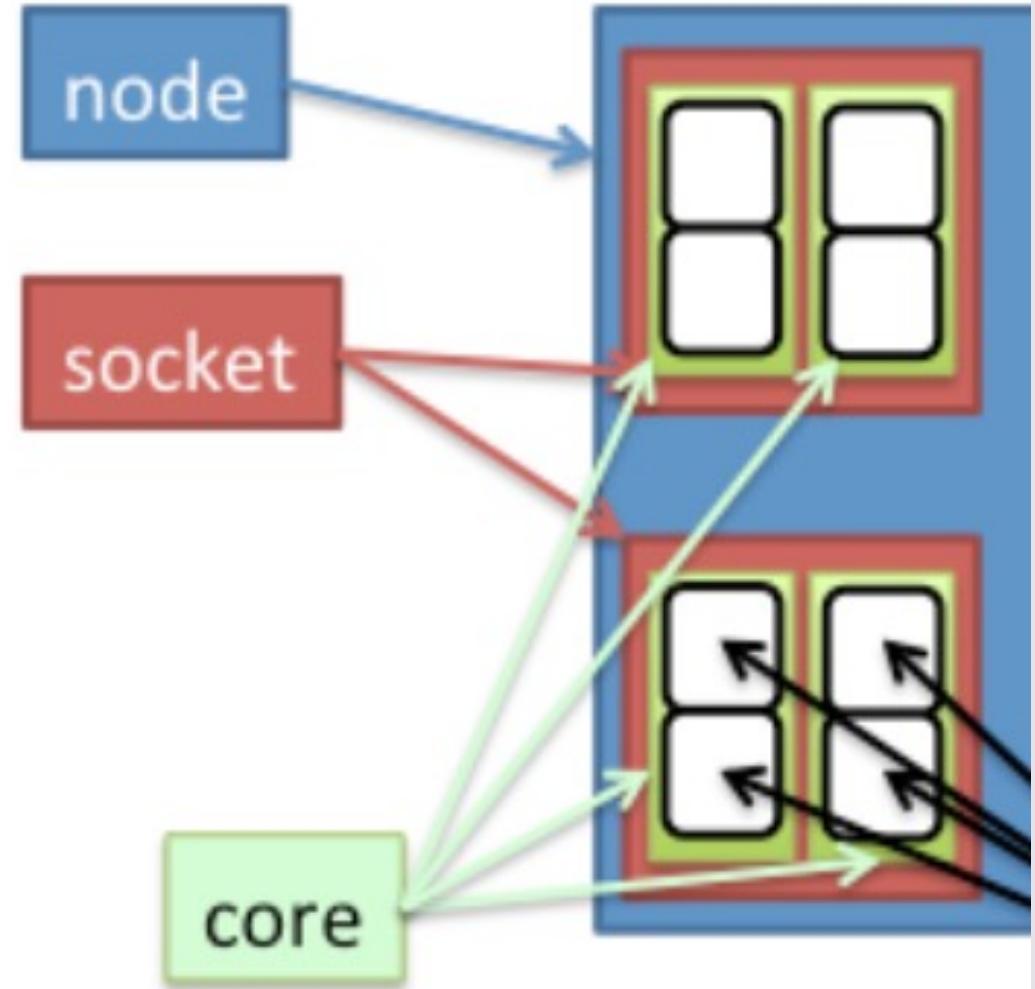


- A socket can be considered an entire computing chip in modern computers.
- Contains more than 1 core.
- On supercomputers a node usually contains more than 1 socket.

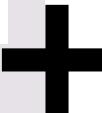
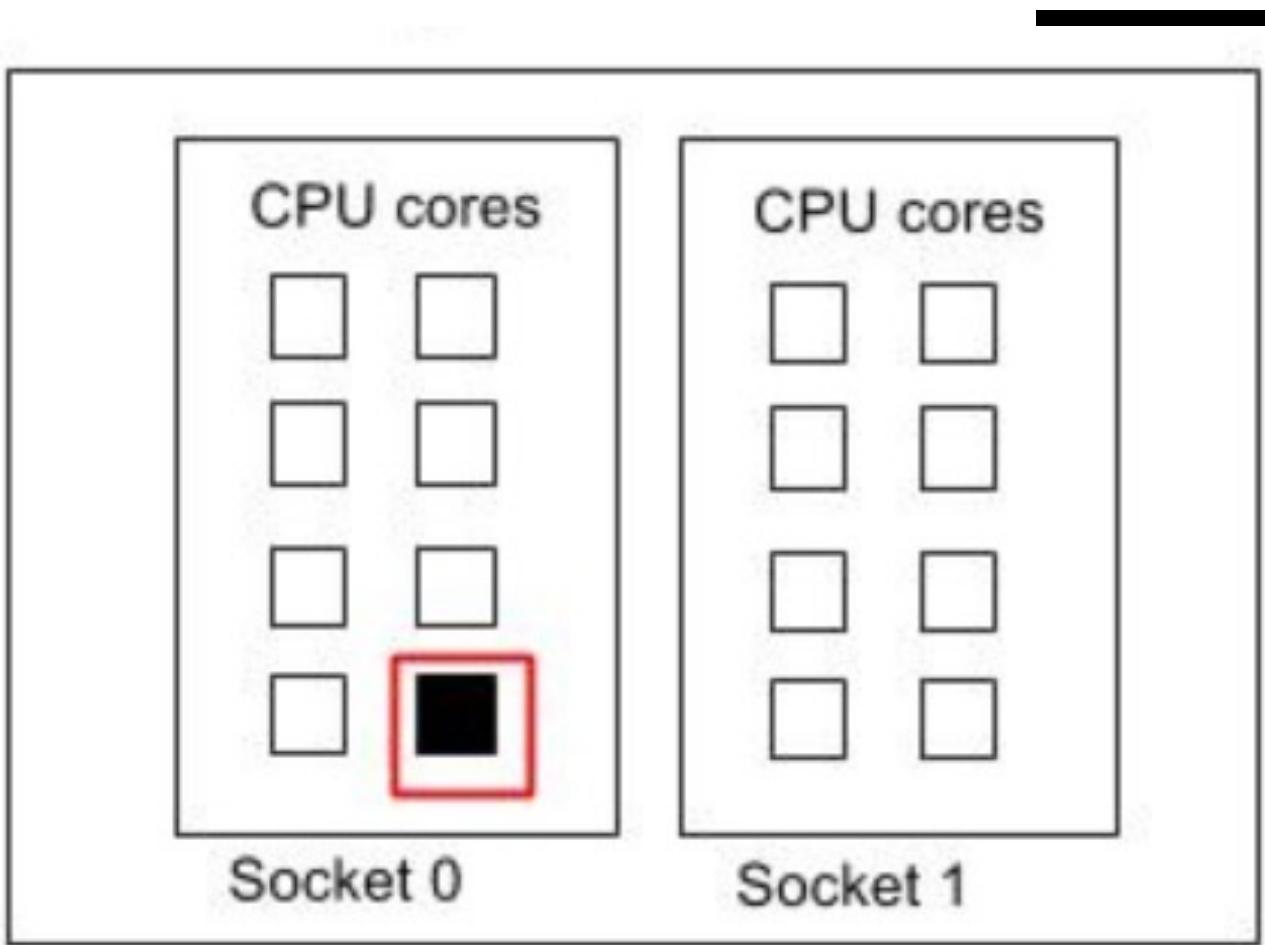


Parts of the processing chip (2)

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

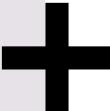
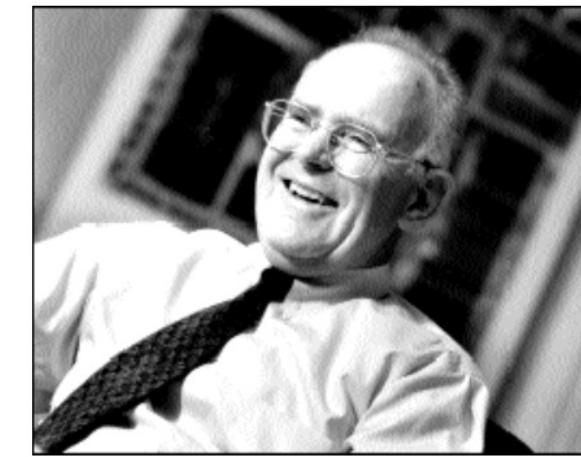
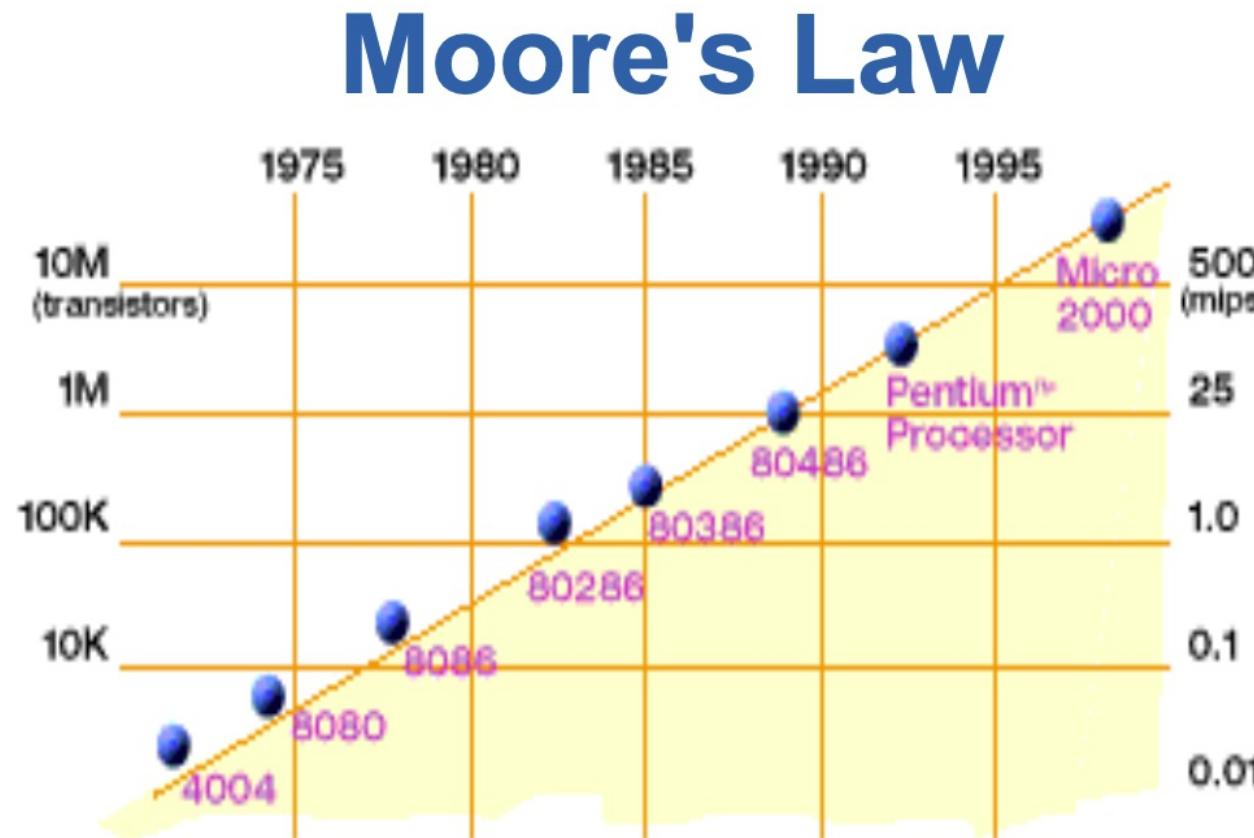


Parts of the processing chip (2)



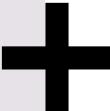
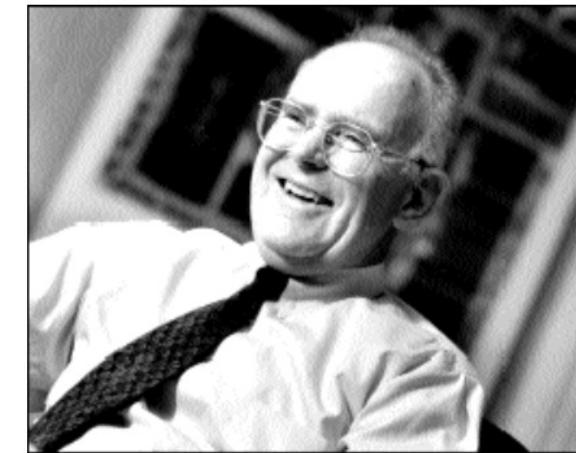
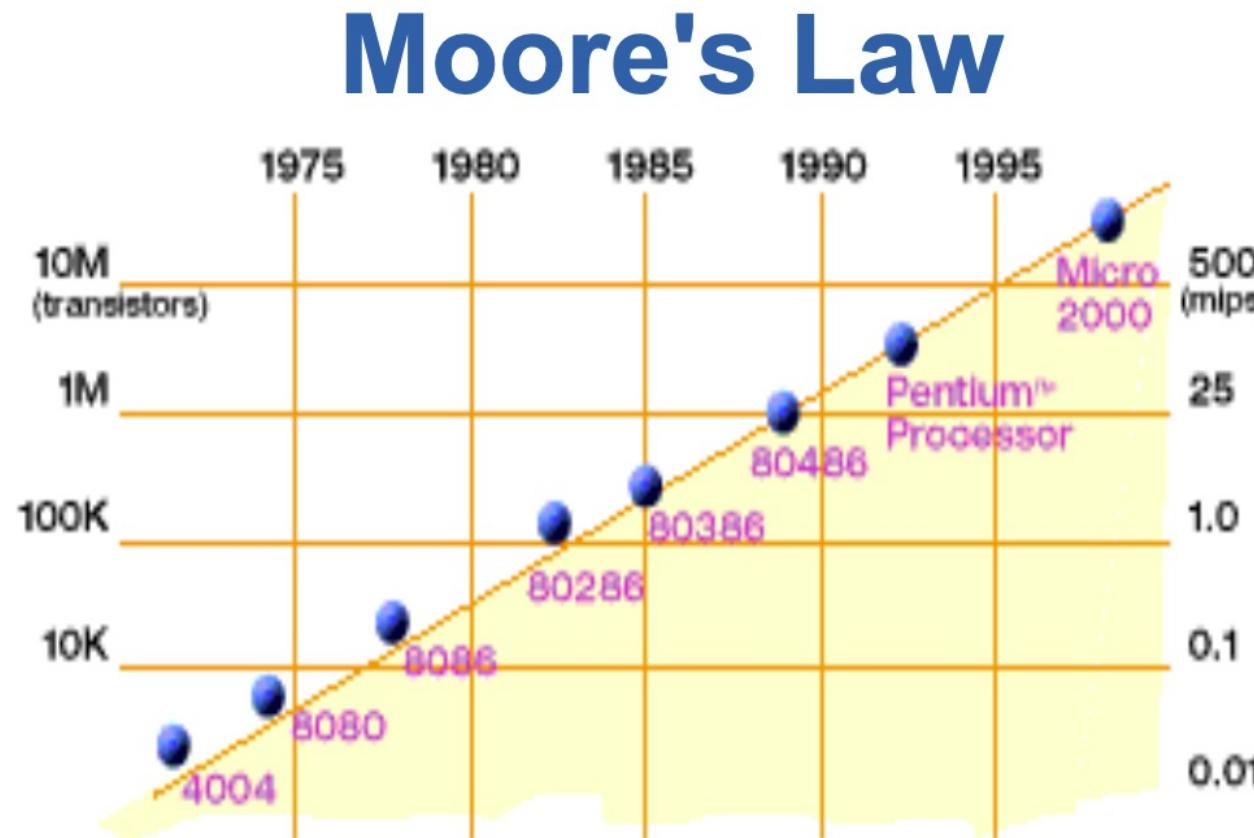
Moore's law (1)

- Transistor count doubles in a chip every 18 months.



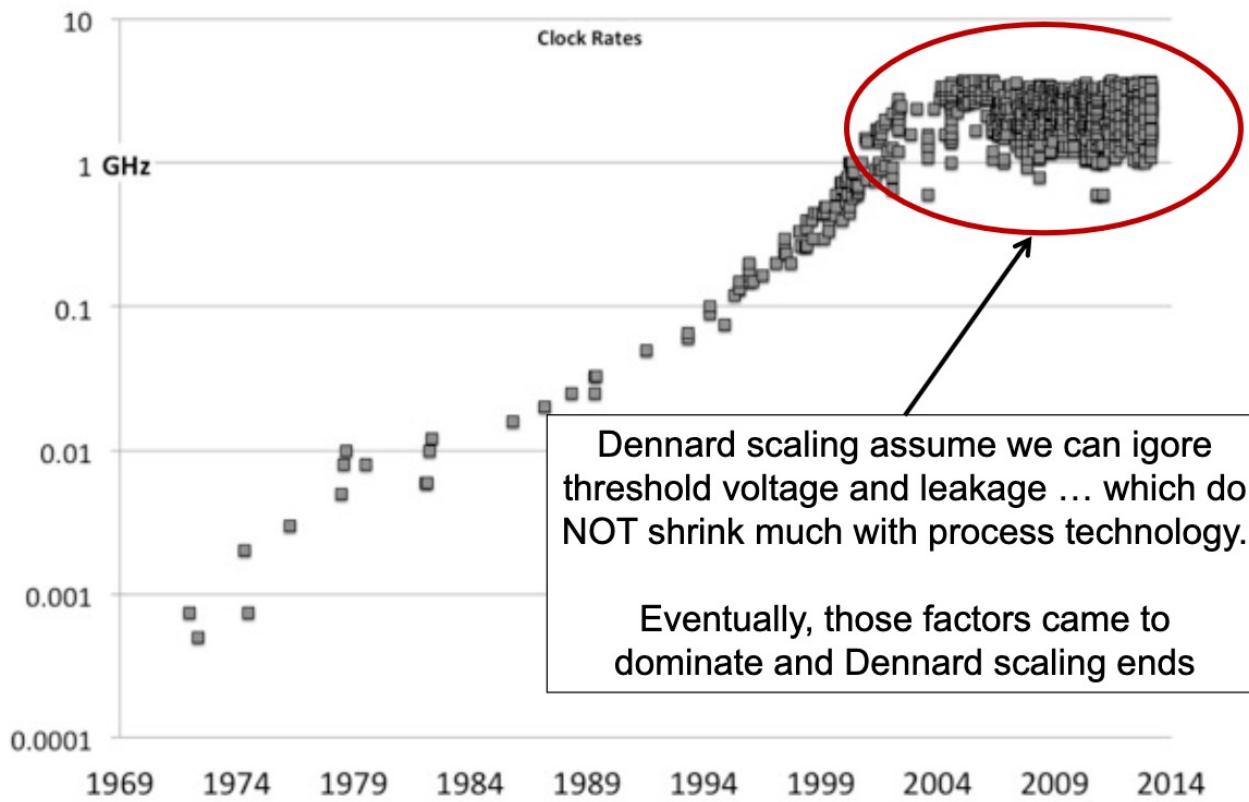
Moore's law (2)

- Worked until around 2010 where it started to plateau

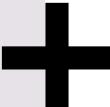


Moore's law plateau

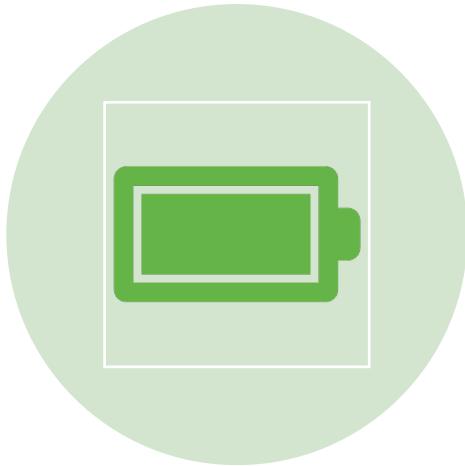
CPU Frequency (GHz) over time (years)



- Number of cycles per seconds plateaued in the 2010s
- Why?



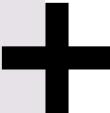
Why?

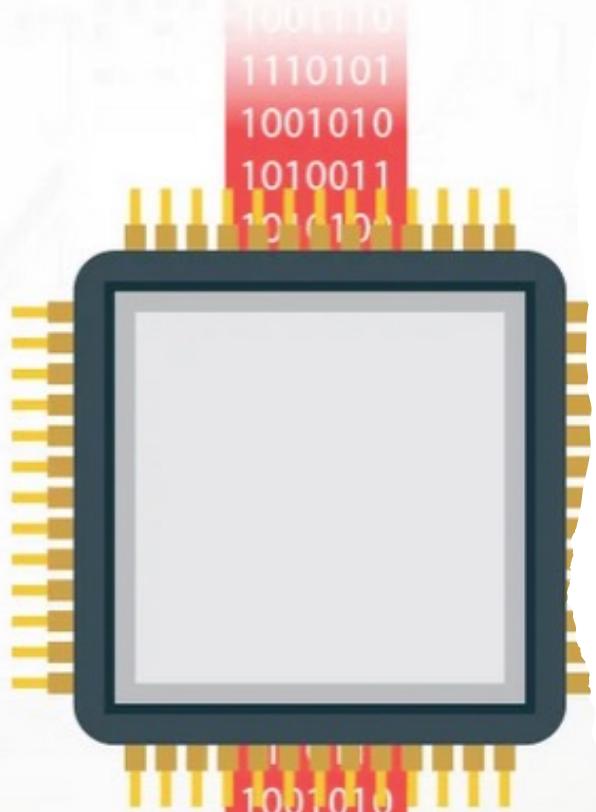


TRADITIONALLY, POWER DENSITY STAYS CONSTANT AS CHIPS FEATURE GOT SMALLER.



HOWEVER AROUND 2010, CLOCK FREQUENCY COULD NOT BE RAISED FURTHER BECAUSE THE HEAT DISSIPATION OR POWER WOULD HAVE GONE TOO FAR.



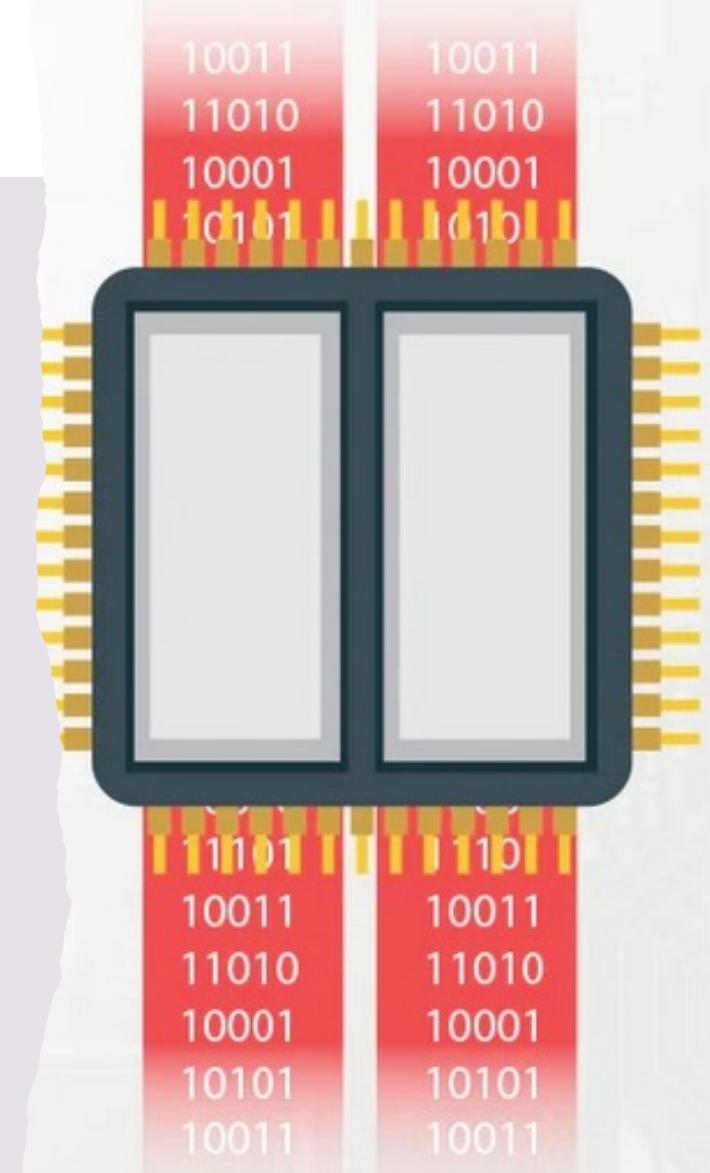


Single-core

Increase need of parallelism

Reduce power by adding cores of small frequency

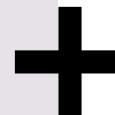
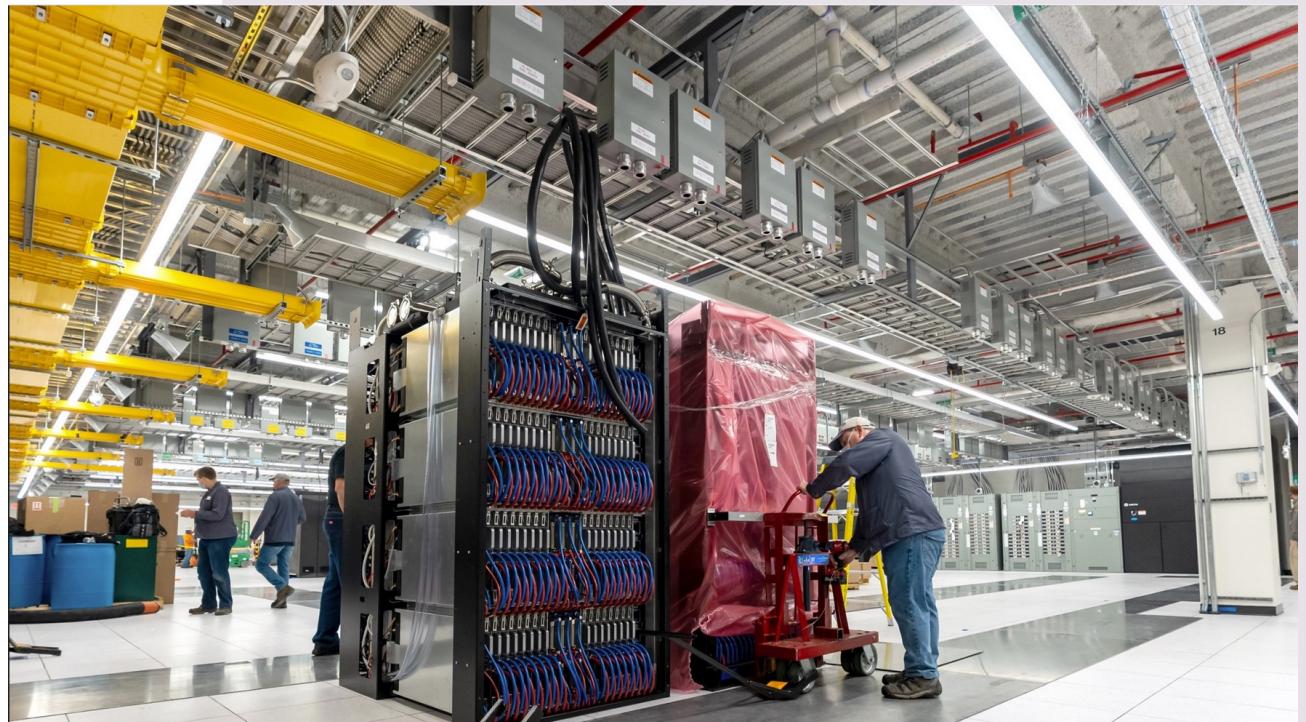
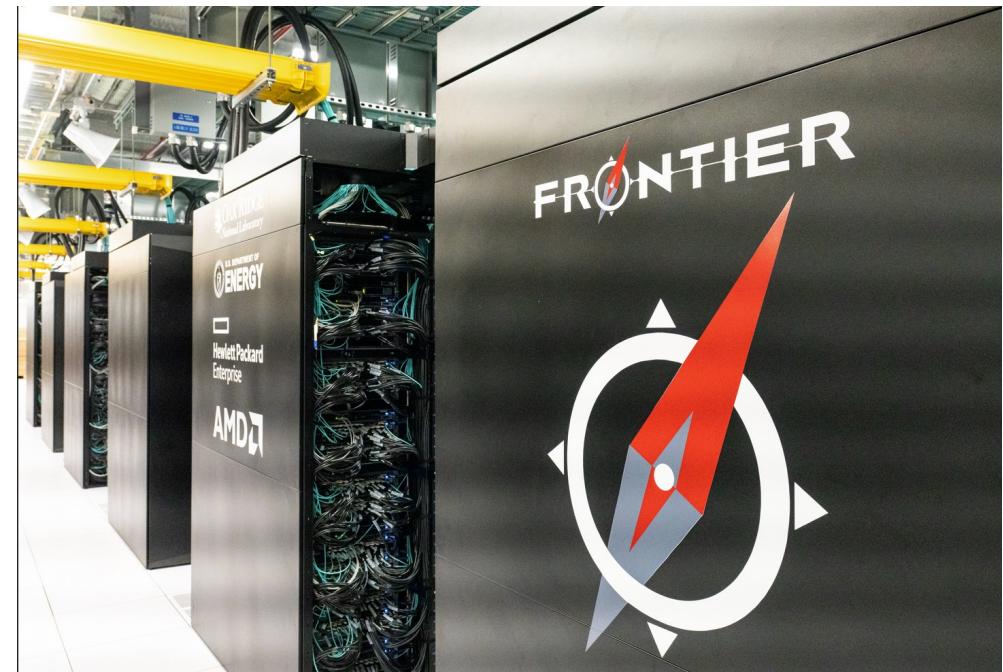
2 cores at half frequency f can achieve the same throughput as a single core of frequency f



Dual Core

A look at an HPC cluster(1)

- Frontier supercomputer OAK Ridge National lab.
[AMD Instinct MI250X GPU; 8,699,904 Cores]
- Most powerful at the moment benchmarking with Linpack (e.g. $Ax=b$)



A look at an HPC cluster(2)



June 2023: The TOP 10 Systems (52% of the Total Performance of Top500)

- Alpine has >18,000 cores!

Rank	Site	Computer	Country	Cores	Rmax [Pflops]	% of Peak	Power [MW]	GFlops/Watt
1	DOE / OS Oak Ridge Nat Lab	Frontier, HPE Cray Ex235a, AMD 3rd EPYC 64C, 2 GHz, AMD Instinct MI250X , Slingshot 10	USA	8,699,904	1,194	71	22.7	52.6
2	RIKEN Center for Computational Science	Fugaku, ARM A64FX (48C, 2.2 GHz), Tofu D Interconnect	Japan	7,299,072	442.	82	29.9	14.8
3	EuroHPC /CSC	LUMI, HPE Cray EX235a, AMD 3rd EPYC 64C, 2 GHz, AMD Instinct MI250X , Slingshot 10	Finland	1,268,736	304.	72	2.94	52.3
4	EuroHPC/CINECA	BullSequana XH2000, Xeon Platinum 8358 32C 2.6GHz, NVIDIA A100 (108C) , Quad-rail NVIDIA HDR100	Italy	1,824,768	239.	78	7.4	32.2
5	DOE / OS Oak Ridge Nat Lab	Summit, IBM Power 9 (22C, 3.0 GHz), NVIDIA GV100 (80C) , Mellanox EDR	USA	2,397,824	149.	74	10.1	14.7
6	DOE / NNSA Livermore Nat Lab	Sierra, IBM Power 9 (22C, 3.1 GHz), NVIDIA GV100 (80C) , Mellanox EDR	USA	1,572,480	94.6	75	7.44	12.7
7	National Super Computer Center in Wuxi	Sunway TaihuLight, SW26010 (260C) , Custom Interconnect	China	10,649,000	93.0	74	15.4	6.05
8	DOE / OS NERSC - LBNL	Perlmutter HPE Cray EX235n, AMD EPYC 64C 2.45GHz, NVIDIA A100 , Slingshot 10	USA	706,304	64.6	71	2.59	27.4
9	NVIDIA Corporation	Selene NVIDIA DGX A100, AMD EPYC 7742 (64C, 2.25GHz), NVIDIA A100 (108C) , Mellanox HDR	USA	555,520	63.4	80	2.64	23.9
10	National Super Computer Center in Guangzhou	Tianhe-2A NUDT, Xeon (12C), MATRIX-2000 (128C) + Custom Interconnect	China	4,981,760	61.4	61	18.5	3.32

Anatomy of supercomputing



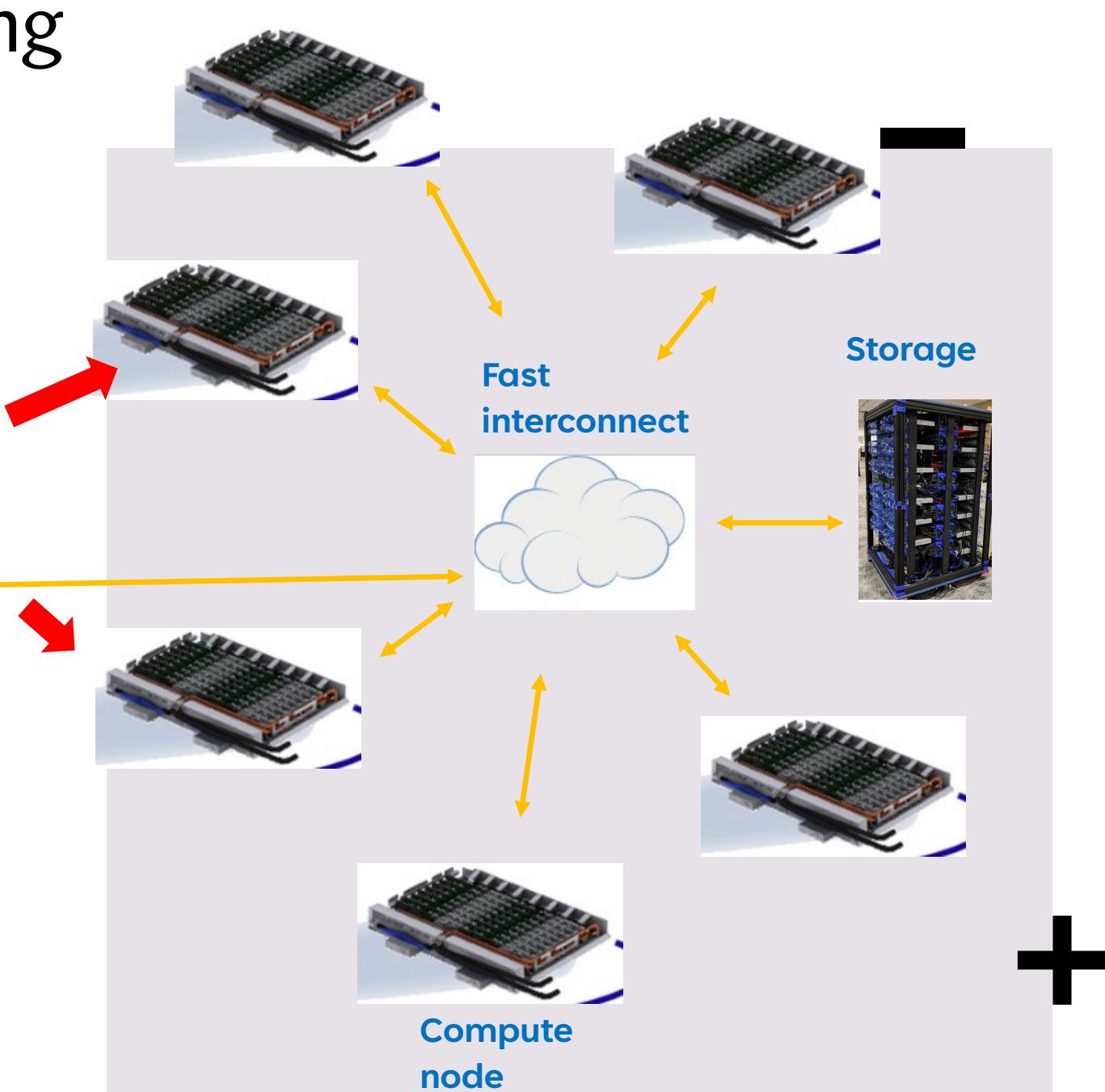
Login node



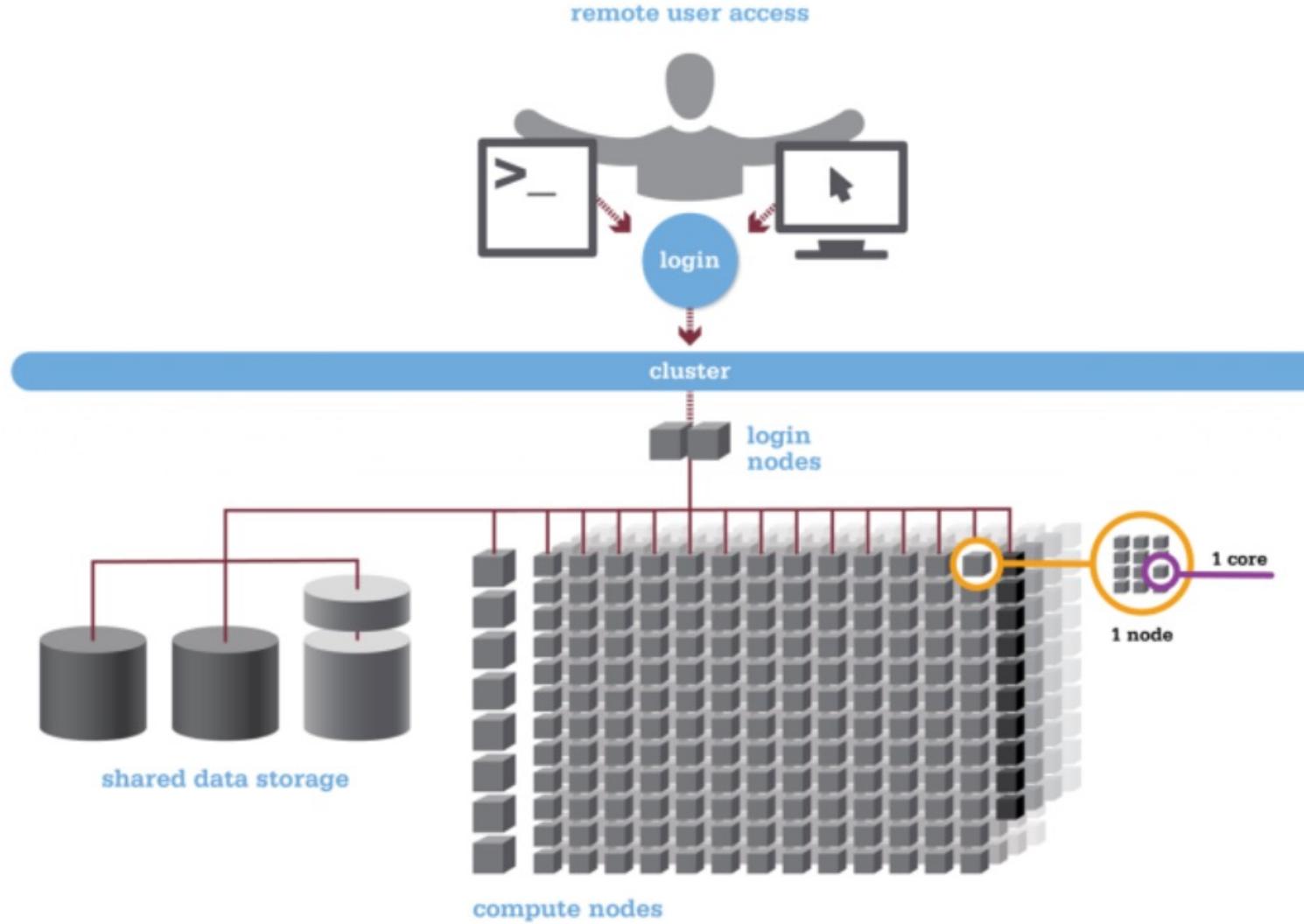
Slurm:
scheduler



Laptop or
PC



Architecture of a supercomputer

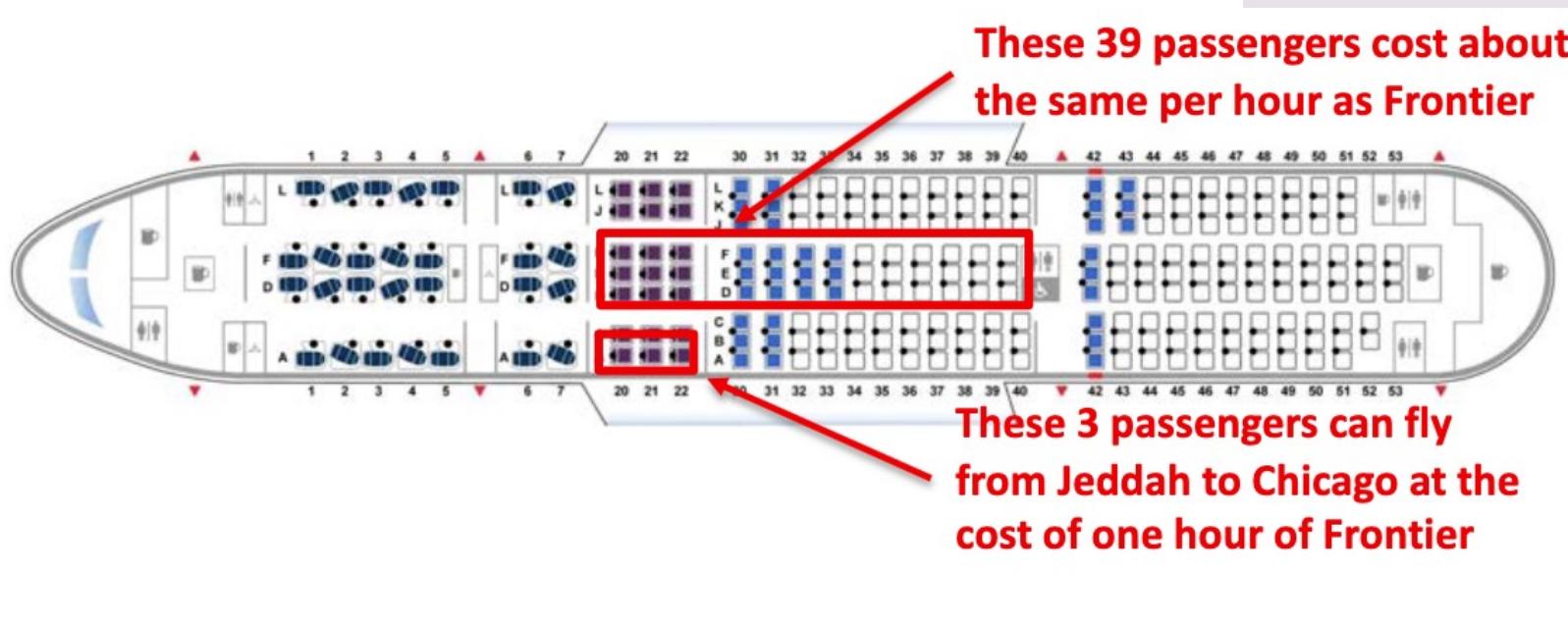


- Login nodes. To log into the system, cd into directories, look at files etc ...
- Compute node. Dedicated to do the computation
- The slurm scheduler controls access to the compute nodes to avoid a tragedy of the commons ...

Why Slurm?



- Allows for better tracking of resource used per user
- Better management of resource across user base
- More sustainable and less tragedy of the commons
- Widely used and supported by applications in case of bugs.



Why Slurm?

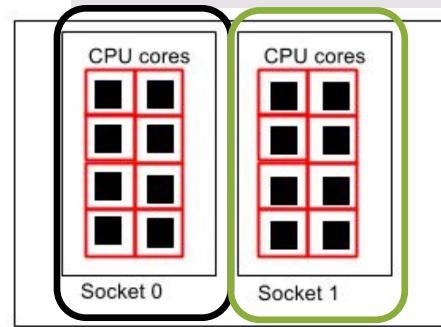
- Bulma request 8 cores from Node 1
- Dr. Brief requests 8 cores as well from Node 1
- Gohan requests 16 cores from Node 2
- No tragedy of the common!!!



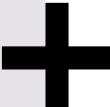
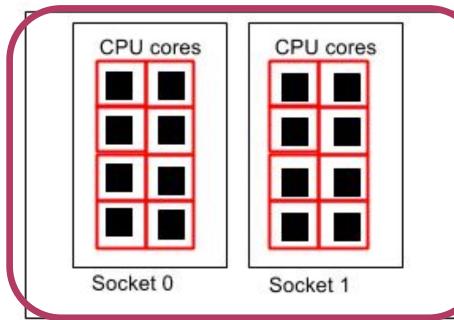
Slurm
scheduler



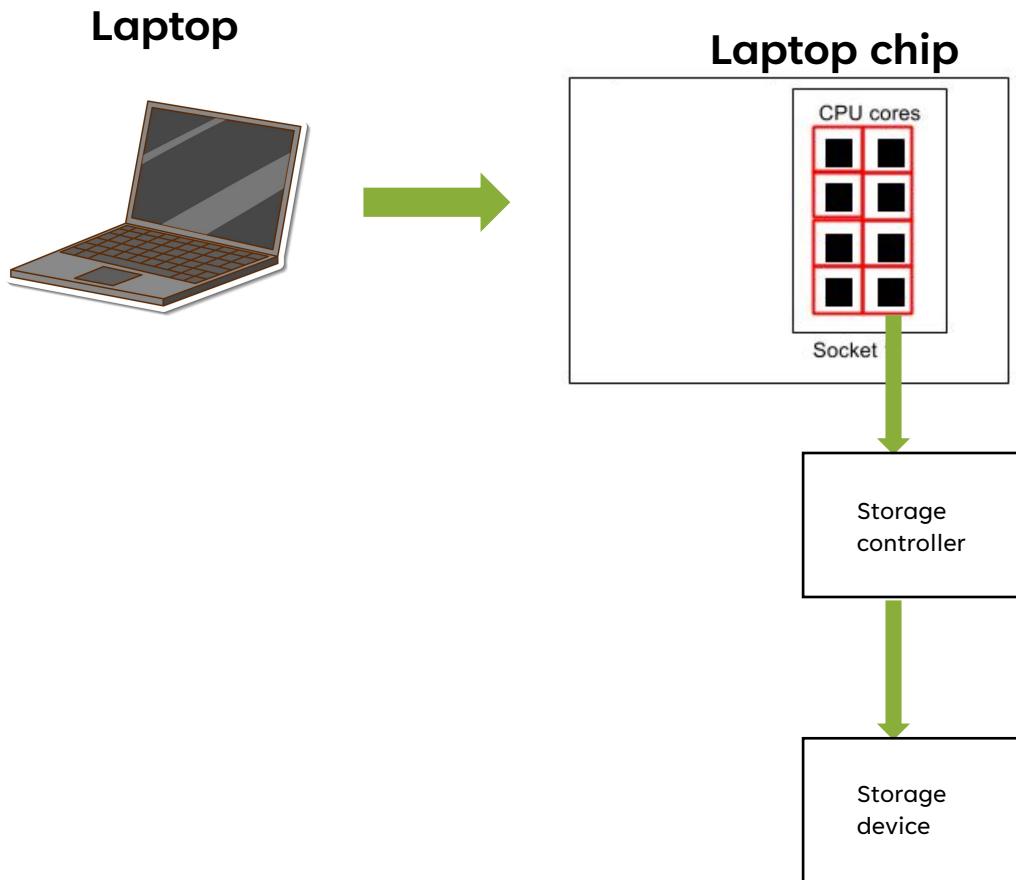
Node 1



Node 2



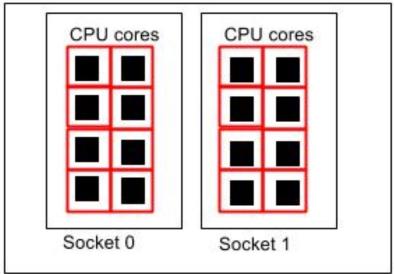
Overview of storage [PC]



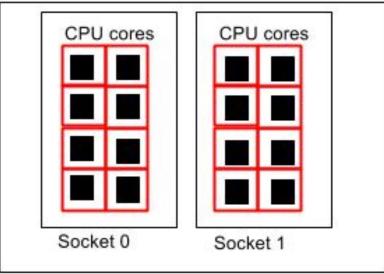
- Compute core talks to the storage controller which talks to the storage device to get the data
- Path between software to storage is short.
- Low latency and low bandwidth.

Overview of storage [HPC]

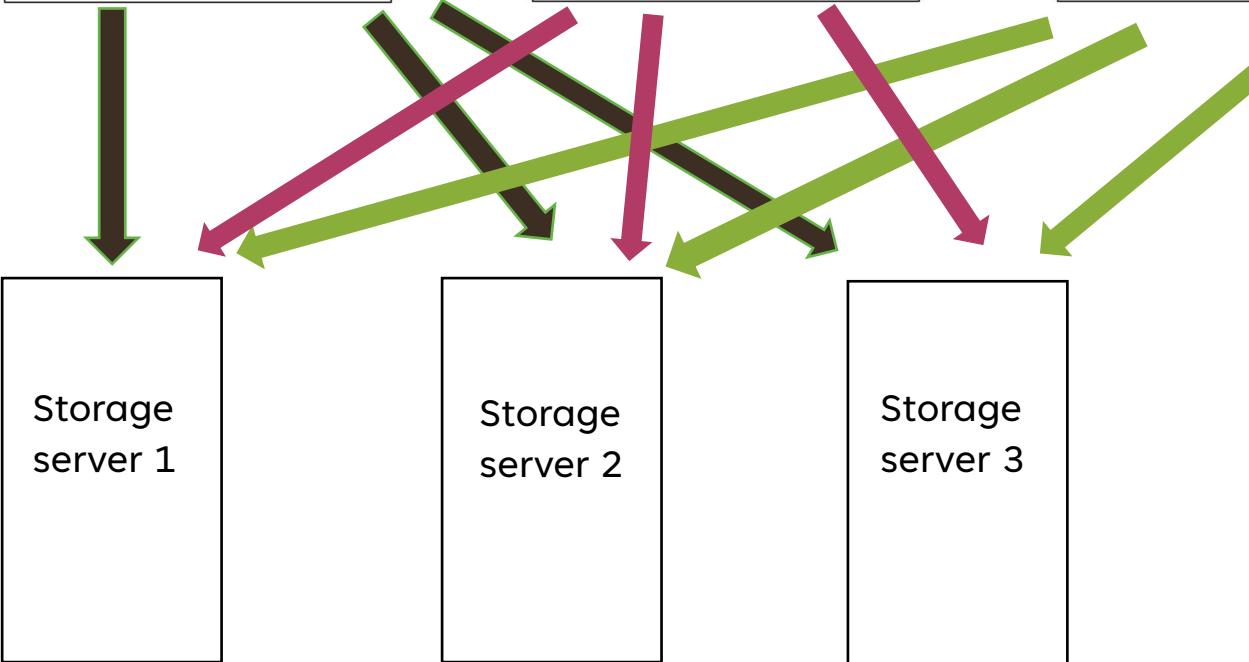
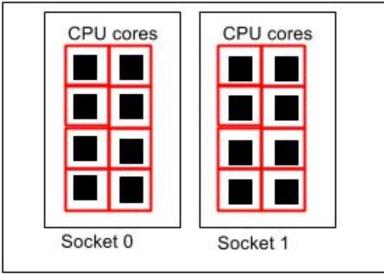
Node 1



Node 2



Node 3



- Files distributed across multiple servers.
- Much Higher bandwidth.
- In general higher latency.

Filesystems on High Performance Computing

- Network filesystem (NFS):
Allows users to access files and directories located on remote servers



GPFS High performance filesystem

- Optimized for heavy I/O pipelines
- Optimized for heavy parallelized pipelines



II- Part 2) Intro to Alpine



Official Github pages:

- CU Anschutz HPC official Github page:

<https://github.com/kf-cuanschutz/CU-An documentation>.

- Note: You can submit pull requests as well.

The screenshot shows the GitHub repository page for kf-cuanschutz / CU-Anschutz-HPC-documentation. The repository is public and has 2 issues, 1 branch, and 0 tags. It contains 63 commits from kf-cuanschutz. The commits are listed below, showing updates to README.md, Globus-local-entry-point-files, Office-hours-presentation-files, ALDEEx2-R-package-installation.md, Alpine-cluster-maintenance.md, Alpine-pipeline-opt-FAQ.md, EcholocateR.md, MATLAB-kernel-on-Jupyterlab.md, README.md, and cellRangerRkit.md. The repository has 6 stars, 1 watching, and 0 forks. It also includes sections for About, Releases, and Packages.

About
All documentation associated with bioinformatics tutorial on the Alpine cluster

Readme

Activity

6 stars

1 watching

0 forks

Releases
No releases published
Create a new release

Packages
No packages published
Publish your first package

- CU Boulder curc doc:

<https://curc.readthedocs.io/en/latest/access/logging-in.html>

- Tickets submitted by emailing: rc-help@colorado.edu. You may email AMC HPC Support as well: hpcsupport@cuanschutz.edu.

Logging into Alpine(1)

Alpine Office Hours_.pdf
422 KB

<User Name>,
Congratulations on your new Alpine account! You are now ready to start using Alpine.

To help you navigate Alpine and get your workflows started quickly, we have compiled a few quick instructions and tips on how to get the most out of your experience. For a fully comprehensive overview of all documentation related to Alpine, please visit the following site: <https://curc.readthedocs.io/en/latest/index.html>. We recommend you bookmark this site and save this email message for future use.

Please note, all functionality, commands, and directives can only be used after requesting an `acompile` node on the command line. This can be done by typing `acompile` and pressing enter once you are logged into the terminal. From there, all SLURM and module commands will work as expected.

Introduction to the SLURM job scheduler

Alpine utilizes the SLURM job scheduling software to manage job submissions to the compute nodes. The Boulder CURC team has provided tutorials and documentation for how to write your job scripts in order for the head node to allocate and assign your jobs with the appropriate resources you are requesting.

- Useful SLURM script examples: <https://curc.readthedocs.io/en/latest/clusters/alpine/examples.html>
- Commonly used SLURM commands on Alpine: <https://curc.readthedocs.io/en/latest/running-jobs/slurm-commands.html>
- SLURM flags and partitions on Alpine: <https://curc.readthedocs.io/en/latest/running-jobs/job-resources.html>
- How to write and submit a SLURM job script: <https://curc.readthedocs.io/en/latest/running-jobs/batch-jobs.html>
- Setting up an interactive job with SLURM: please use the `acompile` or `sinteractive` commands to spin up an interactive job; specify the `-h` flag to see all options for these commands.

All Anschutz users should specify `#SBATCH --account=amc-general` in their job submission scripts in order to get the Anschutz priority boost enabled on their jobs.

Logging into Alpine

- For Anschutz users, you will need to login to Alpine by following the [set of instructions outlined here in this link](#) using the Open OnDemand portal.
- For now, directly accessing Alpine from the terminal using ssh is not yet enabled. We will notify you when this feature is available.

Your directories on Alpine

The paths below use the built-in shell variable `$USER` to represent your username. You can access them directly using the paths shown below, since `$USER` will be replaced with your username by the shell. If you are unsure of your username, you can find it by typing `echo $USER` on the Alpine terminal. For Anschutz users, this may have the `@xsede.org` or `@access.org` extension.

/home/\$USER

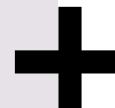
- 2 GB per user
- Not meant to be used for data storage or for reading and writing to/from compute nodes!
- It should be used for `.bashrc`, `.bash_profile`, `.ssh`, etc...

/projects/\$USER

- 250 GB per user
- Used for very small data (<1 GB) and local software installs
- When compiling software, please use a `compile` node, not the `head` node. [For more information on nodes please visit node types here.](#)



- Once the user's EUA application is approved, they will receive an onboarding email similar to what is above.



Logging into Alpine(2)

<User Name>,
Congratulations on your new Alpine account! You are now ready to start using Alpine.

To help you navigate Alpine and get your workflows started quickly, we have compiled a few quick instructions and tips on how to get the most out of your experience. For a fully comprehensive overview of all documentation related to Alpine, please visit the following site: <https://curc.readthedocs.io/en/latest/index.html>. We recommend you bookmark this site and save this email message for future use.

Please note, all functionality, commands, and directives can only be used after requesting an acompile node on the command line. This can be done by typing acompile and pressing enter once you are logged into the terminal. From there, all SLURM and module commands will work as expected.

Introduction to the SLURM job scheduler
Alpine utilizes the SLURM job scheduling software to manage job submissions to the compute nodes. The Boulder CURC team has provided tutorials and documentation for how to write your job scripts in order for the head node to allocate and assign your jobs with the appropriate resources you are requesting.

- Useful SLURM script examples: <https://curc.readthedocs.io/en/latest/clusters/alpine/examples.html>
- Commonly used SLURM commands on Alpine: <https://curc.readthedocs.io/en/latest/running-jobs.html>
- SLURM flags and partitions on Alpine: <https://curc.readthedocs.io/en/latest/running-jobs/job-reservation.html>
- How to write and submit a SLURM job script: <https://curc.readthedocs.io/en/latest/running-jobs/basic.html>
- Setting up an interactive job with SLURM: please use the acompile or sinteractive command options for these commands.

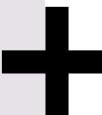
All Anschutz users should specify `#SBATCH --account=amc-general` in their job submission script.

Logging into Alpine

- For Anschutz users, you will need to login to Alpine by following the [set of instructions outlined here](#) using the Open OnDemand portal.
- For now, directly accessing Alpine from the terminal using ssh is not yet enabled. We will notify you when it is available.

Your directories on Alpine
The paths below use the built-in shell variable \$USER to represent your username. You can access them directly using the paths shown below, since \$USER will be replaced with your username by the shell. If you are unsure of your username, you can find it by typing `echo $USER` on the Alpine terminal. For Anschutz users, this

-
- Then they will open the following link in listed in the email and follow the procedure.



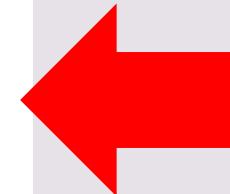
Logging into Alpine(3)

Logging into Alpine from the Shell App

Note

Make sure you already have your XSEDE/ACCESS user name and password set up before proceeding and Duo 2-factor authentication set up for your ACCESS/XSEDE account

1. Visit <https://ondemand-rmacc.rc.colorado.edu> You will be redirected to CILogon. From there, make sure you select the ACCESS CI (XSEDE) as your identity provider and then click the "Log On" button.



CILogon

Consent to Attribute Release

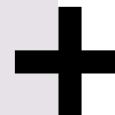
OOD RMACC requests access to the following information. If you do not approve this request, do not proceed.

- Your CILogon user identifier
- Your name
- Your email address
- Your username and affiliation from your identity provider

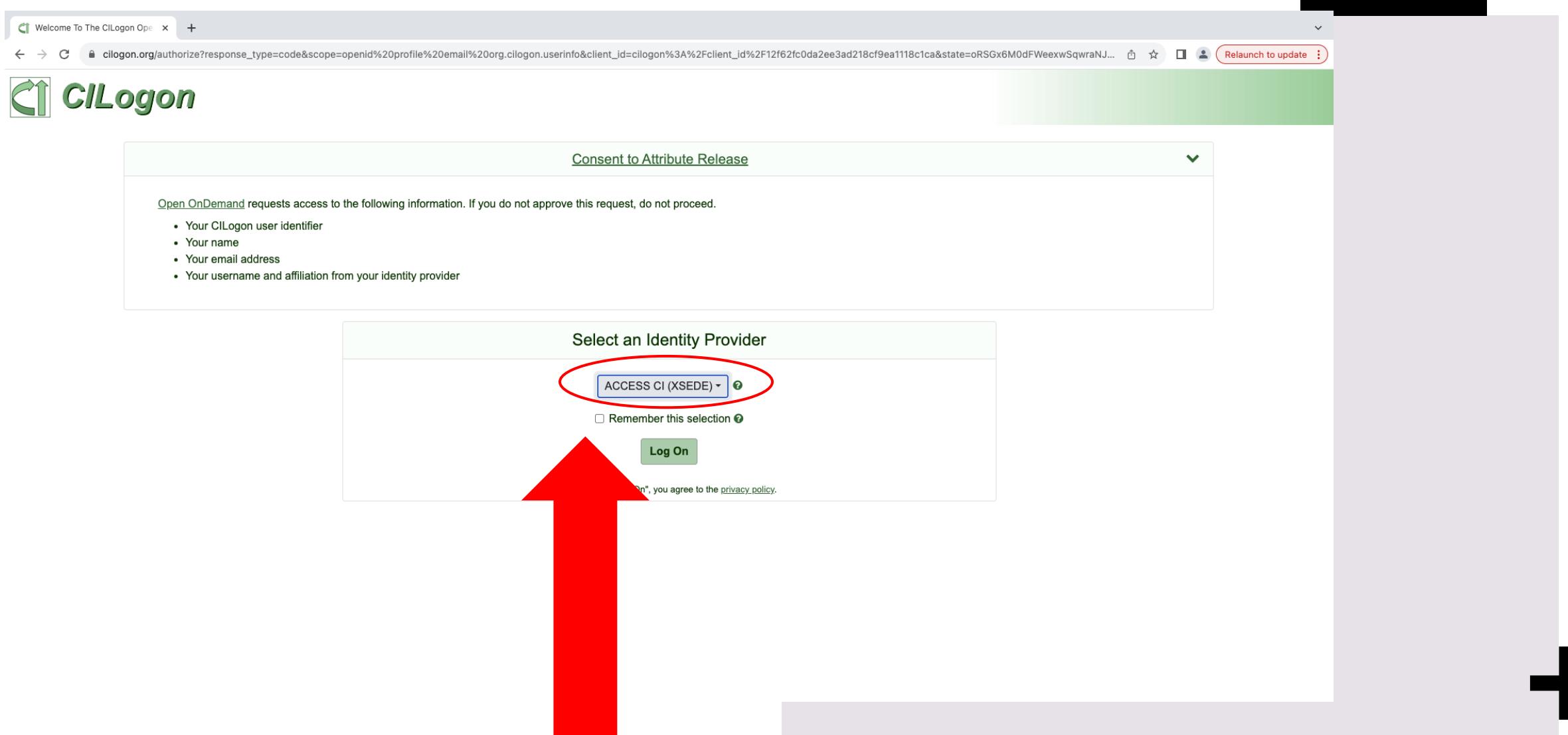
Select an Identity Provider

ACCESS CI (XSEDE)

Remember this selection

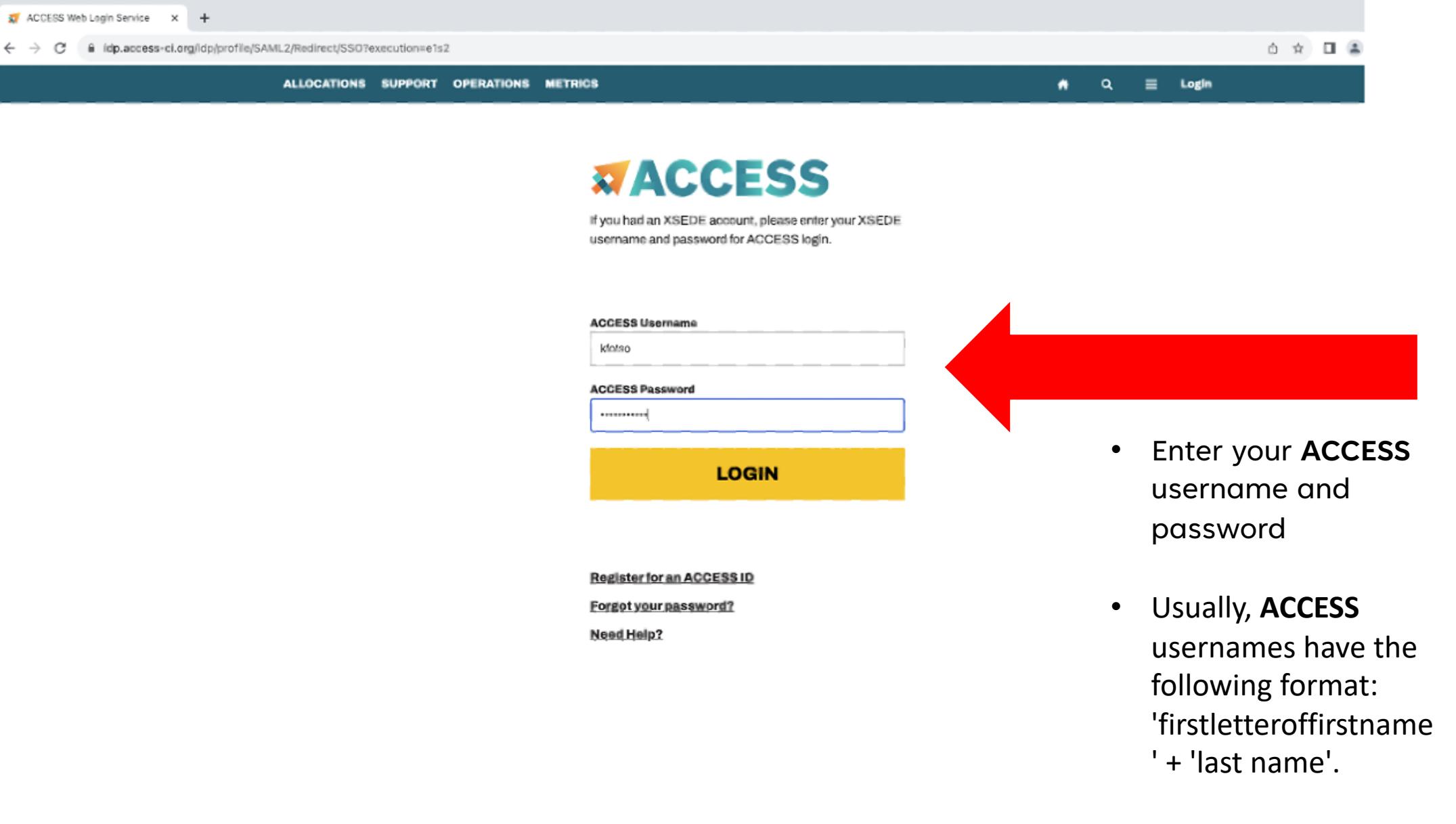


Logging into Alpine(4)



- Make sure to select ACCESS CI (XSEDE) Identity Provider **and NOT CU Anschutz !!!**

Logging into Alpine(5)



If you had an XSEDE account, please enter your XSEDE username and password for ACCESS login.

ACCESS Username

ACCESS Password

LOGIN

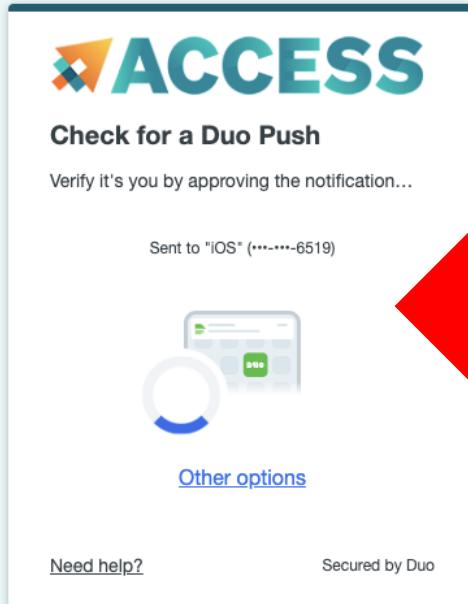
[Register for an ACCESS ID](#)
[Forgot your password?](#)
[Need Help?](#)

- Enter your **ACCESS** username and password
- Usually, **ACCESS** usernames have the following format: 'firstletteroffirstname' + 'last name'.

Logging into Alpine(6)

Duo Security

api-616d966d.duosecurity.com/frame/v4/auth/prompt?sid=frameless-db928ffc-8566-42c6-a72a-64e4f50995d5



ACCESS

Check for a Duo Push

Verify it's you by approving the notification...

Sent to "iOS" (*****-6519)

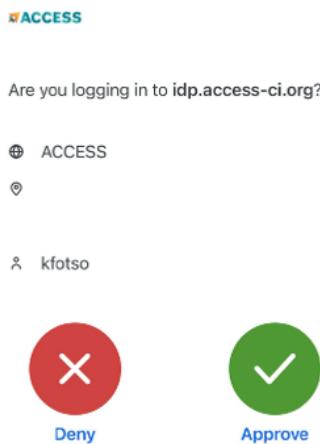
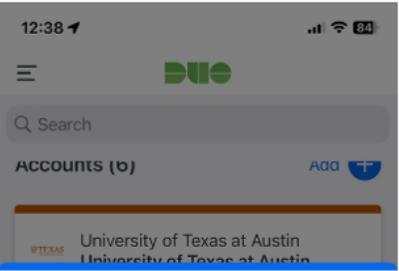
Other options

Need help?

Secured by Duo

Make sure to check the DUO push notification on your phone.

Logging into Alpine(7)



The authentication confirmation should look similar to the following on your phone.

Logging into Alpine(2)

The screenshot shows the OnDemand web interface for the University of Colorado Research Computing system. At the top, there is a navigation bar with icons for Files, Jobs, Clusters (which is currently selected and highlighted in black), Interactive Apps, and My Interactive Sessions. A red circle highlights the 'Clusters' dropdown menu, which is open to show two options: '_Alpine Shell' and '_Blanca Shell'. Below the navigation bar is the CU Boulder Research Computing logo, featuring a gold 'CU' monogram and the text 'Research Computing UNIVERSITY OF COLORADO BOULDER'. A main message states: 'OnDemand provides an integrated, single access point for all of your HPC resources.' Below this is a 'Message of the Day' section with the text: 'Welcome to the University of Colorado Research Computing.' Underneath, there is a 'Quick Links' section with several blue hyperlinks: 'CU Boulder RC Status', 'Research Computing User Guide', 'Research Computing at CU Boulder', 'RMACC @ Ask.Cyberinfrastructure', and 'Need help? Email (rc-help@colorado.edu)'. A large black plus sign icon is located in the bottom right corner of the page.

OnDemand provides an integrated, single access point for all of your HPC resources.

Message of the Day

Welcome to the University of Colorado Research Computing.

Quick Links

[CU Boulder RC Status](#)
[Research Computing User Guide](#)
[Research Computing at CU Boulder](#)
[RMACC @ Ask.Cyberinfrastructure](#)
Need help? Email (rc-help@colorado.edu)

On the login node (1)

```
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.
```

```
bash: /usr/bin/python: No such file or directory
```

```
Welcome to CU-Boulder Research Computing.
```

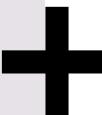
```
* Website http://colorado.edu/rc  
* Questions? rc-help@colorado.edu  
* Subscribe to system announcements: https://curc.statuspage.io/  
* Please type rc-help for the Acceptable Use Policy and a short help page.
```

```
You are using login node: login-ci1
```

```
bash: /usr/bin/python: No such file or directory  
[kfotso@xsede.org@login-ci1 ~]$ █
```



Name of the username



On the login node (2)

```
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.
```

```
bash: /usr/bin/python: No such file or directory
```

```
Welcome to CU-Boulder Research Computing.
```

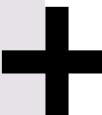
```
* Website http://colorado.edu/rc  
* Questions? rc-help@colorado.edu  
* Subscribe to system announcements: https://curc.statuspage.io/  
* Please type rc-help for the Acceptable Use Policy and a short help page.
```

```
You are using login node: login-ci1
```

```
bash: /usr/bin/python: No such file or directory  
[kfotso@xsede.org@login-ci1 ~]$
```



Name of the login node
(hostname)



Storage(1)

Home filesystem (2G). Backed up + for hosting config files.

Project filesystem (250G). Backed up -> for package installation

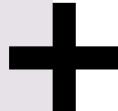
“cd .snapshot” to access those back ups

‘curc-quota’ or ‘du’ to check on space

Storage(2)

```
[kfotso@xsede.org@login-ci1 ~]$ curc-quota
```

	Used	Avail	Quota	Limit
/home/kfotso@xsede.org	289M	1.8G	2.0G	
/projects/kfotso@xsede.org	144G	107G	250G	
/scratch/alpine1	7786G	20825G	28611G	



Storage(3)

- Scratch space (10 TB)
- GPFS filesystem
- Very suitable for parallel application + heavy I/O
- Purged every 90 days

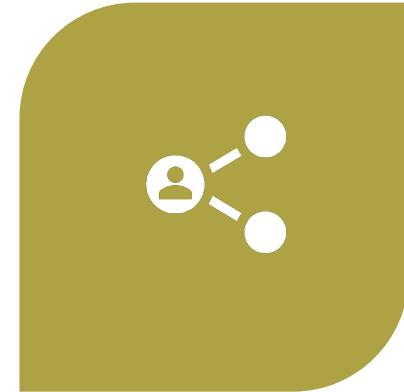
Petalibrary

+

Preliminary conditions (1)

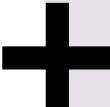


CREATION OF AN ACCESS
GROUP

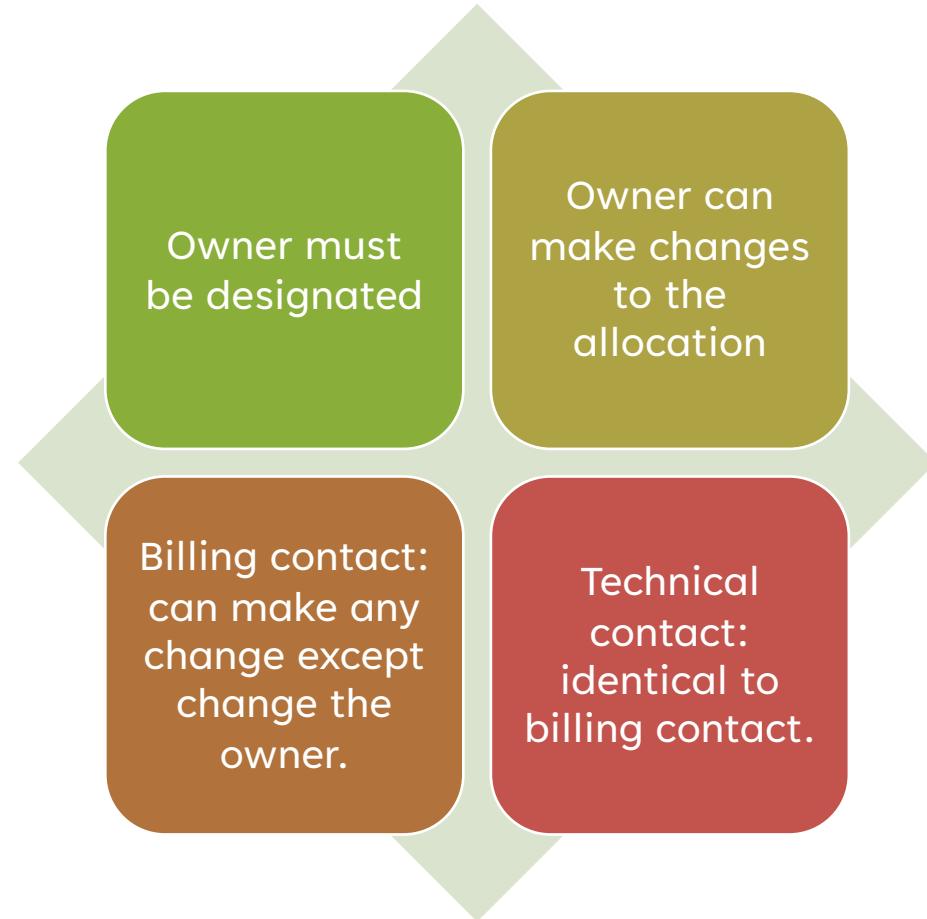


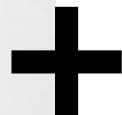
ALL MEMBER ADDED TO THE
GROUP WILL NEED
ACCESS/XSEDE ACCOUNTS.

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



Preliminary conditions (2)





Application

- Application submitted here:
<https://www.colorado.edu/rcc/resources/petalibrary>
- The request form will need a speedtype: account# to which they plan to charge the allocation.

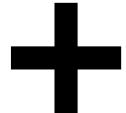
Billing (1)

- On active storage: \$45/TB/yr.
- ZFS Raidz2 allow for frequent read/write + parity.
- **It is highly suggested that total TB size for the year is determined in advanced by the owner.**



Billing (2)

- On archive storage: \$20/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



Terms of usage

- Data will need to be in full compliance with term of service.
- No PHI data, no FERPA, no ITAR, no GDPR, no export control and data/software that comply with IRB requirements.
- More information here:
<https://www.colorado.edu/rc/resources/petalibrary/tos>
- Data classification:
<https://www.cu.edu/security/data-classification>.
- HIPAA identifiers:
<https://www.dhcs.ca.gov/dataandstats/data/Pages>ListofHIPAAIdentifiers.aspx>

```
mirror_mod = modifier_obj
# Set mirror object to mirror
mirror_mod.mirror_object = active_obj
if operation == "MIRROR_X":
    mirror_mod.use_x = True
    mirror_mod.use_y = False
    mirror_mod.use_z = False
elif operation == "MIRROR_Y":
    mirror_mod.use_x = False
    mirror_mod.use_y = True
    mirror_mod.use_z = False
else:
    mirror_mod.use_x = False
    mirror_mod.use_y = False
    mirror_mod.use_z = True

#selection at the end -add
modifier_obj.select= 1
active_modifier_obj.select=1
bpy.context.scene.objects.active = bpy.context.scene.objects[("Selected" + str(modifier_obj)).split('.')[0]]
modifier_obj.select = 0
bpy.context.selected_objects.append(active_modifier_obj)
data.objects[one.name].select = 1
print("please select exactly one object to mirror")
#----- OPERATOR CLASSES -----
#----- OPERATOR CLASSES -----
```



```
types.Operator:
    def execute(self):
        # X mirror to the selected object.mirror_mirror_x"
        if self.mirror_x:
            mirror_x()
        if self.mirror_y:
            mirror_y()
        if self.mirror_z:
            mirror_z()

    def invoke(self, context):
        if context.active_object is not None:
```

Data redundancy (1)



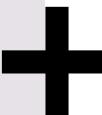
PL allocations are of single-copy nature.



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



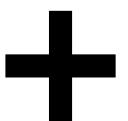
Sort term back up solutions (1)

- Microsoft OneDrive: 5TB per person through A3 licensing.
- S3 buckets: customer paid & rates located here:
<https://aws.amazon.com/s3/pricing/?p=pm&c=s3&z=4>
- Multiple PL allocations as though customer requested 2 allocations (X2 price).



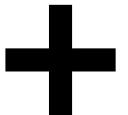
Sort+mid term back up solutions (2)

- [BETA]
Replicated PetaLibrary active
+archive allocation.
- Nothing is charged during
beta period but will likely look
like Multiple PL allocations.
- CURC **only takes data
replication responsibility** for
BETA



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.
- \$65/TB/year total. **(NOT \$65 on top of the \$45/\$20 original rate!!)**





1

Creation of Globus Connect
Personal endpoint.



2

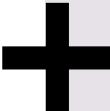
CU Boulder endpoint for
Anschutz was CU Boulder
Research Computing
ACCESS.



3

Current ACCESS point
(DTN23) is 10X faster than
the legacy one
“CU Boulder Research
Computing”

Files transfer (Globus)



Globus (2)

Collections

Get Globus Connect Personal

CU Boulder Research Computing

QUICK FILTERS RECENTLY USED ADMINISTERED BY YOU IN USE SHAREABLE BY YOU SHARED WITH YOU

COLLECTION	HA	MANAGED	STATUS	ROLE
CU Boulder Research Computing (internal - Legacy - Do Not Use) Managed GCSv4 Host			requires activation	
CU Boulder Research Computing - OLD Managed GCSv4 Host			requires activation	
CU Boulder Research Computing (Legacy - Do Not Use) Managed GCSv4 Host			requires activation	
CU Boulder Research Computing Managed Mapped Collection (GCS) on CU Boulder Research Computing DTN23			ready	

A red oval highlights the fourth row, which corresponds to the collection circled in the previous screenshot.

A large black plus sign is located in the bottom right corner of the interface.

Globus(3)

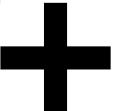
- More info here:

<https://curc.readthedocs.io/en/latest/compute/data>

[transfer.html?highlight=globus#globus-transfers](#)

The screenshot shows a sidebar menu for "Research Computing University of Colorado Boulder". The sidebar includes sections for "Frequently Asked Questions", "ACCESSING RC RESOURCES" (Logging In, Duo 2-factor Authentication, RMACC Access to Alpine, SSH Key-Based Authentication for Anschutz Medical Campus), "THE COMPUTE ENVIRONMENT" (Node types, Filesystems, The Modules System), and "Data Transfer" (Globus transfers, Guest Collections (Globus Shared Endpoints), Globus to AWS S3 Bucket connection, Filezilla, Secure Copy (scp), Rsync, Interactive file transfer with sftp). The main content area is titled "Data Transfer" and discusses various file transfer methods supported by Research Computing.

The screenshot shows the "Data Transfer" page from the CURC documentation. It features a heading "Data Transfer", a paragraph about supported file transfer methods, and a video player with the title "Data Transfers" showing a view of the Flatirons in Boulder, Colorado. Below the video, there is a "Watch on YouTube" button and the "Be Boulder." logo.



II) - Alpine computing



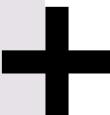
Scheduler Slurm



acompile --ntasks=1 --time=00:30:00 to build packages and do some testing. (max 12 hours)



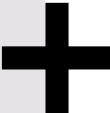
sinteractive --ntasks-per-node=2 --nodes=2 --partition=atesting to test pipelines. (max 1 hour)



```
[kfotso@xsede.org@login-ci1 kfotso@xsede.org]$ sinfo --partition=aa100
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
aa100          up 1-00:00:00      1  resv  c3gpu-a9-u33-1
aa100          up 1-00:00:00      7  mix   c3gpu-a9-u31-1,c3gpu-a9-u35-1,c3gpu-c2-u
aa100          up 1-00:00:00      4  alloc  c3gpu-a9-u29-1,c3gpu-c2-u[7,13,15]
```

sinfo

Can be used to get
information about a node.



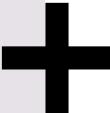
Package availability (1)

Some packages that have been built and accessible through Imod.

Adding new packages through Imod takes a lot of round of approval so it is recommended to build them locally.

Solutions: (cmake+make), Anaconda, pip, containers, spack etc ...

Submit a ticket at rc-help@Colorado.edu so that I can build it for you locally.



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.



List of officially available bio modules

- “acompile”
- “module avail”

`alphafold/2.2.0`
`alphafold/2.3.1 (D)`
`bamtools/2.5.2`
`bbtools/39.01`

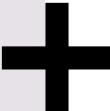
`bcftools/1.16`
`bedtools/2.29.1`
`bowtie2/2.5.0`
`bwa/0.7.17`

`cellranger/7.1.0`
`cutadapt/4.2`
`fastqc/0.11.9`
`gatk/4.3.0.0`

`htslib/1.16`
`multiqc/1.14`
`nextflow/22.10.6`
`nextflow/23.04 (D)`

`picard/2.27.5`
`plink2/2.00a2.3`
`qiime2/2023.5`
`samtools/1.16.1`

`sra-toolkit/3.0.0`
`star/2.7.10b`
`trimmomatic/0.39`

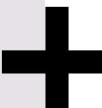


Slurm example

```
#!/bin/bash ← Your shell is bash  
  
#SBATCH --partition=amilan ← Partition is amilan (CPU)  
#SBATCH --job-name=example-job ← Job name when it gets queued  
#SBATCH --output=example-job.%j.out ← Output file name  
#SBATCH --time=01:00:00 ← How long my job runs?  
#SBATCH --qos=normal  
#SBATCH --nodes=1  
#SBATCH --ntasks=4  
#SBATCH --mail-type=ALL  
#SBATCH --mail-user=youridentikey@colorado.edu
```

```
module purge  
module load anaconda  
conda activate custom-env
```

```
pythonmyscript.py
```



Slurm example

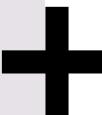
```
#!/bin/bash

#SBATCH --partition=amilan
#SBATCH --job-name=example-job
#SBATCH --output=example-job.%j.out
#SBATCH --time=01:00:00
#SBATCH --qos=normal
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mail-type=ALL
#SBATCH --mail-user=youridentikey@colorado.edu
```

Quality of service
How many nodes I want to run on?

```
module purge
module load anaconda
conda activate custom-env
```

```
python myscript.py
```



Slurm example

```
#!/bin/bash

#SBATCH --partition=amilan
#SBATCH --job-name=example-job
#SBATCH --output=example-job.%j.out
#SBATCH --time=01:00:00
#SBATCH --qos=normal
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mail-type=ALL
#SBATCH --mail-user=youridentikey@colorado.edu

module purge
module load anaconda
conda activate custom-env

python myscript.py
```

Total cores used

Email when job begins, ends and fails

Email
address

Slurm example

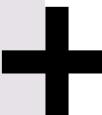
```
#!/bin/bash

#SBATCH --partition=amilan
#SBATCH --job-name=example-job
#SBATCH --output=example-job.%j.out
#SBATCH --time=01:00:00
#SBATCH --qos=normal
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mail-type=ALL
#SBATCH --mail-user=youridentikey@colorado.edu
```

module purge ← Reinitialize module list
module load anaconda ← Load anaconda
conda activate custom-env ← Activate environment
python myscript.py ← Running pipeline command

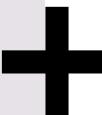
Slurm cheatsheet (1)

Slurm script command	Description
<code>#!/bin/bash</code>	Sets the shell that the job will be executed on the compute node
<code>#SBATCH --ntasks=1</code> <code>#SBATCH --n1</code>	Requests for 1 processors on task, usually 1 cpu as 1 cpu per task is default.
<code>#SBATCH --time=0-05:00</code> <code>#SBATCH -t 0-05:00</code>	Sets the maximum runtime of 5 hours for your job
<code>#SBATCH --mail-user= <email></code>	Sets the email address for sending notifications about your job state.
<code>#SBATCH --mail-type=BEGIN</code> <code>#SBATCH --mail-type=END</code> <code>#SBATCH --mail-type=FAIL</code> <code>#SBATCH --mail-type=REQUEUE</code> <code>#SBATCH --mail-type=ALL</code>	Sets the scheduling system to send you email when the job enters the following states: BEGIN,END,FAIL,REQUEUE,ALL
<code>#SBATCH --job-name=my-named-job</code>	Sets the Jobs name



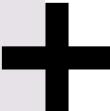
Slurm cheatsheet(2)

Slurm script command	Description
#SBATCH --ntasks=X	Requests for X tasks. When cpus-per-task=1 (and this is the default) this requests X cores. When not otherwise constraint these CPUs may be running on any node
#SBATCH --nodes=X	Request that a minimum of X nodes be allocated to this job
#SBATCH --nodes=X-Y	Request that a minimum of X nodes and a maximum of Y nodes be allocated to this job
#SBATCH --cpus-per-task=X	Request that a minimum of X CPUs per task be allocated to this job
#SBATCH --tasks-per-node=X	Requests minimum of X task be allocated per node



Slurm cheatsheet(3)

Slurm script commands	Description of effects
<code>#SBATCH --ntasks=1</code> <code>#SBATCH --cpus-per-task=1</code>	Requests 1 CPU (Serial) cpus-per-task is set to 1 by default and may be omitted.
<code>#SBATCH --cpus-per-task=X</code> <code>#SBATCH --ntasks=1</code> <code>#SBATCH --nodes=1</code>	Requests for X CPUs in 1 task on 1 node (OpenMP) Both ntasks and nodes are set to 1 by default and may be omitted
<code>#SBATCH --ntasks=X</code> <code>#SBATCH --tasks-per-node=X</code> <code>#SBATCH --cpus-per-task=1</code>	Requests for X CPUs and tasks on 1 node cpus-per-task is set to 1 by default and may be omitted.
<code>#SBATCH --ntasks=X</code> <code>#SBATCH --nodes=1</code> <code>#SBATCH --cpus-per-task=1</code>	Requests for X CPUs and tasks on 1 node cpus-per-task is set to 1 by default and may be omitted.



Get information about jobs

```
[kfotso@xsede.org@login-ci1 ~]$ squeue -l --me
```

JOBID	PARTITION	NAME	USER	STATE
2158225	acompile	acompile	kfotso@x	RUNNING

TIME	TIME_LIMI	NODES	NODELIST(REASON)
0:16	3:00	1	c3cpu-c11-u21-2

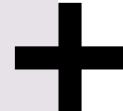
Monitor resources

```
[kfotso@xsede.org@login-ci1 ~]$ module load slurmtools
[kfotso@xsede.org@login-ci1 ~]$ jobstats $USER 2
job stats for user kfotso@xsede.org over past 2 days
jobid      jobname   partition   qos       account    cpus state start-date-time elapsed    wait
-----
2064187    sinterac  atesting_+  testing    amc-gener+  48   TIMEOUT 2023-06-20T23:32:52 01:00:04  0 hrs
2071952    vep_loft  aa100      normal    amc-gener+  64   COMPLETE 2023-06-21T13:47:55 00:26:42  3 hrs
```

- Allows to get information about a past jobs

```
[kfotso@xsede.org@login-ci1 ~]$ seff 1451164
Job ID: 1451164
Cluster: alpine
User/Group: kfotso@xsede.org/kfotsopgrp@xsede.org
State: COMPLETED (exit code 0)
Nodes: 1
Cores per node: 48
CPU Utilized: 26-03:21:39
CPU Efficiency: 94.06% of 27-19:00:48 core-walltime
Job Wall-clock time: 13:53:46
Memory Utilized: 412.77 GB
Memory Efficiency: 41.28% of 999.98 GB
```

- To get more computational information about the job efficiency



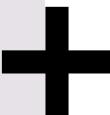
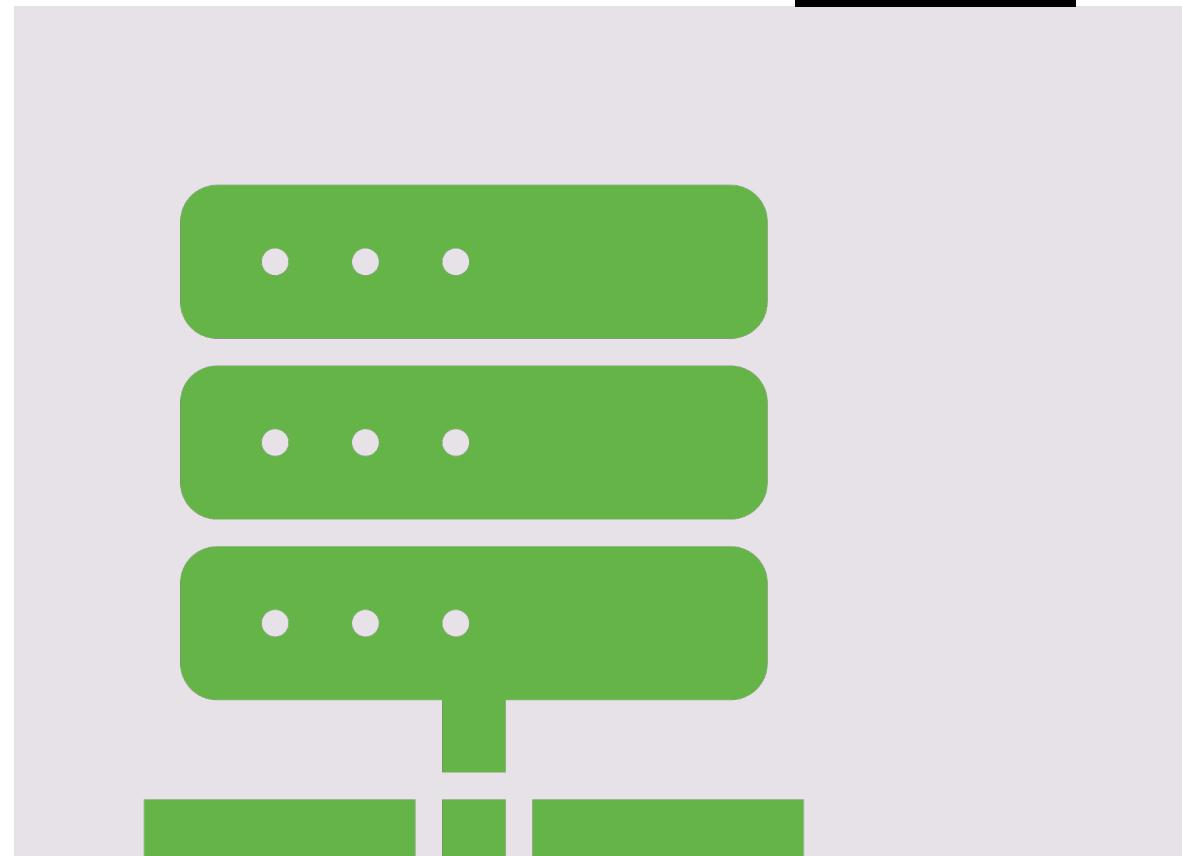
Slurm Quality of service (qos)

- 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 1TB)



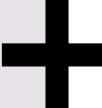
Fairshare overview

- Difference between the portion of computing resource that has been promised and the amount of resources that has been consumed.
- Level fairshare of 1 indicates average priority compared to other users in that account (amc-general)
- **module load slurmtools; levels \$USER**



Job priority calculation formula

```
Job_priority =  
    site_factor +  
    (PriorityWeightAge) * (age_factor) +  
    (PriorityWeightAssoc) * (assoc_factor) +  
    (PriorityWeightFairshare) * (fair-share_factor) +  
    (PriorityWeightJobSize) * (job_size_factor) +  
    (PriorityWeightPartition) * (partition_factor) +  
    (PriorityWeightQOS) * (QOS_factor) +  
    SUM(TRES_weight_cpu * TRES_factor_cpu,  
        TRES_weight_<type> * TRES_factor_<type>,  
        ...)  
    - nice_factor
```

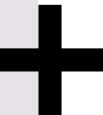


Check fairshare

Host: login-ci1.rc.int.colorado.edu

```
[kfotso@xsede.org@login-ci1 ~]$ levelfs $USER
LevelFS for user kfotso@xsede.org and institution amc:
Account          LevelFS_User      LevelFS_Inst
-----
amc-general      0.194275        4.750220
[kfotso@xsede.org@login-ci1 ~]$
```

- 0.19 means that my priority will be low
- On the other hand 4.75 means that priority for the institution is high



Service Units (SU)

- It is the number of core hours used.

```
[kfotso@xsede.org@login-ci1 ~]$ suuser $USER 10
SU used by user kfotso@xsede.org in the last 10 days:
Cluster|Account|Login|Proper Name|TRES Name|Used|
alpine|amc-general|kfotso@xsede.org|Kevin Fotso|billing|8393|
```

- suacct to get the number of core hours used by institution

```
Host: login-ci1.rc.int.colorado.edu
[kfotso@xsede.org@login-ci1 ~]$ suacct amc-general 180
SU used by account (allocation) amc-general in the last 180 days:
Cluster|Account|Login|Proper Name|TRES Name|Used
alpine|amc-general|||billing|1806360
alpine| amc-general|acozart@xsede.org|Abigail Cozart|billing|573
alpine| amc-general|agillen@xsede.org|Austin Gillen|billing|40320
alpine| amc-general|agray@xsede.org|Alyx Gray|billing|22
```

Best practices on Alpine (1)

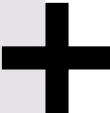
- /home or /tmp have a very small size. Please always include the following in your slurm script.

```
export TMP=/scratch/alpine/$USER
```

```
export TEMP=/scratch/alpine/$USER
```

```
export TMPDIR=/scratch/alpine/$USER
```

```
export TEMPDIR=/scratch/alpine/$USER
```



Best practices on Alpine (2)

- /home or /tmp have a very small size. Please always include the following in your slurm script.

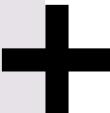
```
export TMP=/scratch/alpine/$USER
```

```
export TEMP=/scratch/alpine/$USER
```

```
export TMPDIR=/scratch/alpine/$USER
```

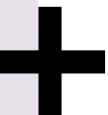
```
export TEMPDIR=/scratch/alpine/$USER
```

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



Best practices on Alpine (3) - Username Aliasing

- `/home/foo@xsede.org` → `/home/.xsede.org/foo/`
- `/projects/foo@xsede.org` → `/projects/.xsede.org/foo/`
- `/scratch/alpine/foo@xsede.org` → `/scratch/alpine/.xsede.org/foo/`



Best practices on Alpine (4) – Anaconda config

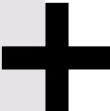
- Edit the .condarc file located in \$HOME/.condarc with the editor of choice (e.g. nano, vim etc ...)
- Paste the following 4 lines in the file and exit.

pkgs_dirs:

- /projects/.xsede.org/foo/.conda_pkgs

envs_dirs:

- </projects/.xsede.org/foo/software/anaconda/envs>



Brief mention of parallelism

- OpenMP (multithreading).
- MPI (Parallel distribution between multiple nodes)
- GNU parallel (embarrassingly parallel)
- Job arrays (max 1000 on Alpine)
- Load balancing (developed by CU Boulder)
- GPU computing



Slurm example to see the number of cores(1)

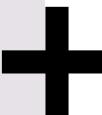
- Here, we are requesting 4 cores for a 2 minutes



```
[kfotso@login-ci2 ~]$ acompile --ntasks=4 --time=00:02:00
acompile: submitting job... salloc --nodes=1 --partition=acompile --ntasks=4 --time=00:02:00 --qos=compile --job-name=acompile --bell
--oversubscribe srun --pty /bin/bash
salloc: Granted job allocation 2998293
salloc: Nodes c3cpu-a5-u32-1 are ready for job
```



- We are getting node c3cpu-a5-u32-1 and our jobID is 2998293



Slurm example to see the number of cores(2)

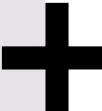
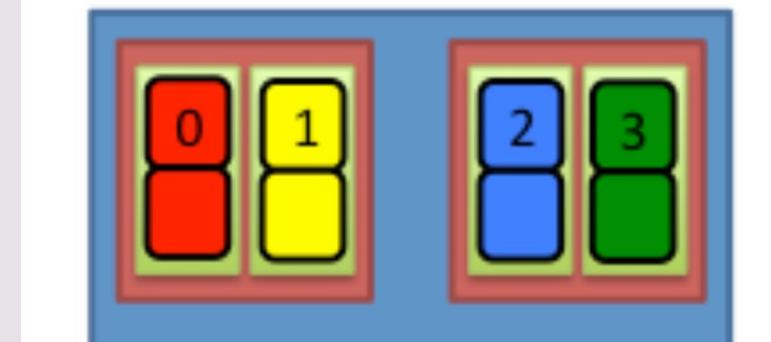
- We want load compiler gcc because openmpi depends on it



```
[kfotso@xsede.org@c3cpu-a5-u32-1 ~]$ module load gcc  
[kfotso@xsede.org@c3cpu-a5-u32-1 ~]$ module load openmpi
```



- We want to use openmpi because we want to demonstrate parallelism on the 4 cores we have reserved



Slurm example to see the number of cores(3)

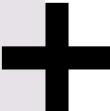
- hostname is just a program that prints the name of the node I am working with



```
[kfotso@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 node gets printed 4 times !!!



Slurm example to see the number of cores(4)

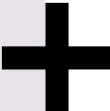
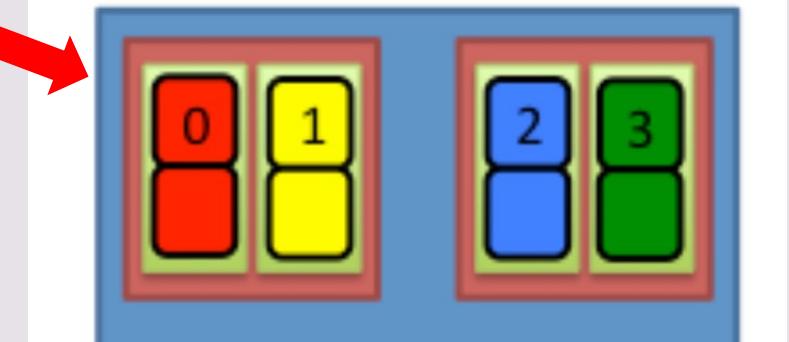
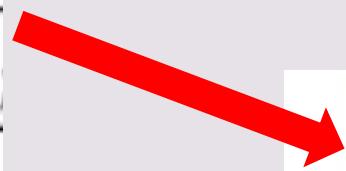
Recap of what we did

`mpirun -n 4 hostname`



tasks

0	hostname
1	hostname
2	hostname
3	hostname



Slurm example to see the number of cores(4)

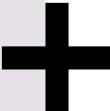
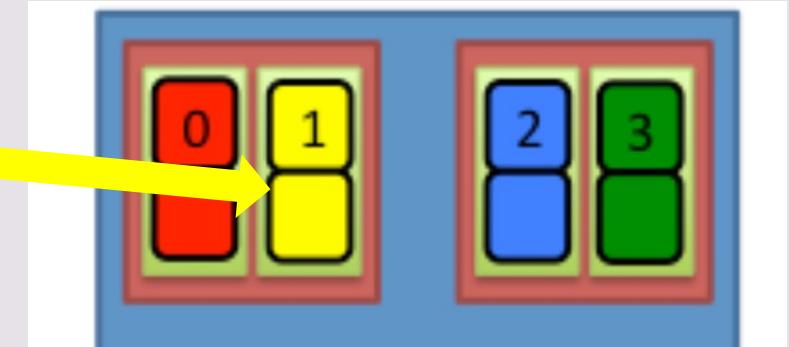
Recap of what we did

`mpirun -n 4 hostname`



tasks

0	hostname
1	hostname
2	hostname
3	hostname



Slurm example to see the number of cores(4)

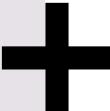
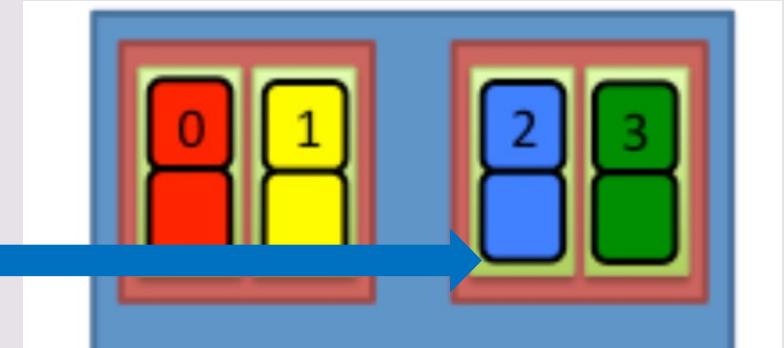
Recap of what we did

`mpirun -n 4 hostname`



tasks

0	hostname
1	hostname
2	hostname
3	hostname



Slurm example to see the number of cores(4)

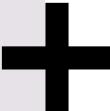
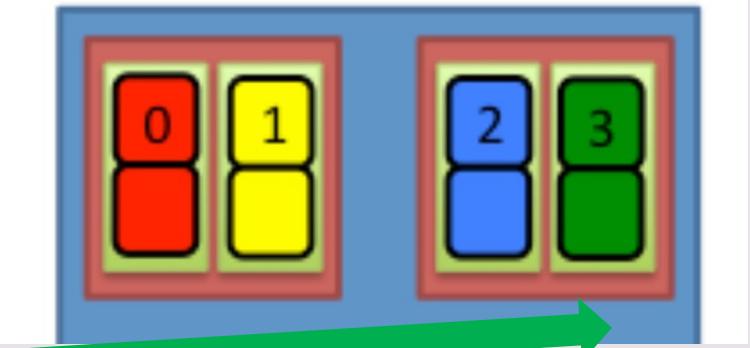
Recap of what we did

`mpirun -n 4 hostname`



tasks

0	hostname
1	hostname
2	hostname
3	hostname



Slurm example to see the number of cores(4)

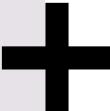
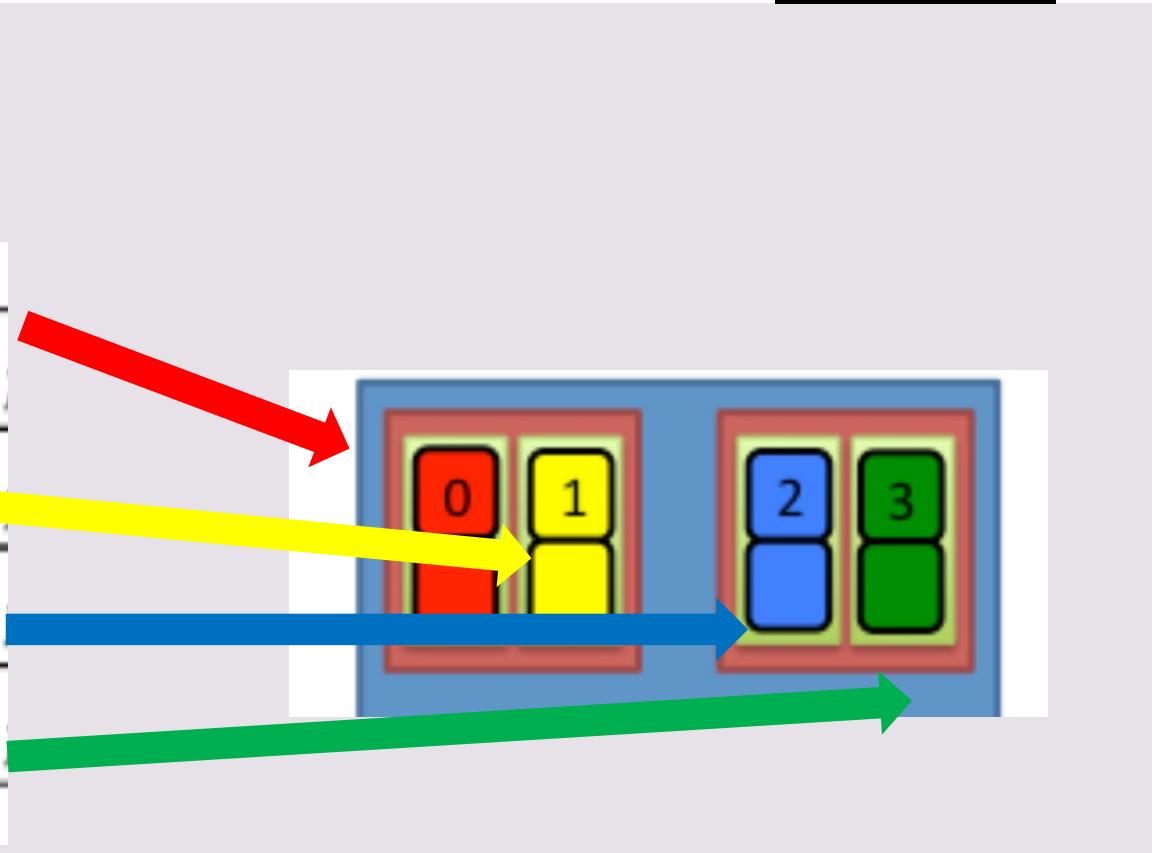
Recap of what we did

`mpirun -n 4 hostname`



tasks

0	hostname
1	hostname
2	hostname
3	hostname



Slurm example to see the number of cores(5)

- Congratulations!!! You just wrote your first HPC parallel code program!

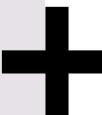


Build the corresponding slurm script (1)

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



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



Build the corresponding slurm script

```
#!/bin/bash

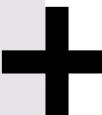
#SBATCH --partition=amilan
#SBATCH --job-name=first_mpirun
#SBATCH --output=first_mpirun-job.%j.out
#SBATCH --error=first_mpirun-job.%j.err
#SBATCH --account=amc-general
#SBATCH --qos=normal
#SBATCH --nodes=1
#SBATCH --ntasks=4
#SBATCH --mail-type=ALL
#SBATCH --mail-user=kfotsotagne@unm.edu
#SBATCH --time=00:00:01

module load gcc
module load openmpi

mpirun -n $SLURM_NTASKS hostname
```

Account name is
amc-general

Job runs for 1
second!!



Build the corresponding slurm script

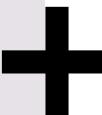
```
#!/bin/bash

#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=kfotsotagne@unm.edu
#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

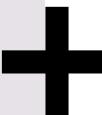


Build the corresponding slurm script (4)

We use sbatch to submit the script



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



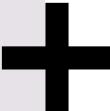
Build the corresponding slurm script

(5)

The .out file shows the name of the nodes 4X.
.err file is empty so no error.



```
[kfotso@xsede.org@login-ci2 openmpi_hostname]$ ls -latr
total 184
drwxrws---. 44 kfotso@xsede.org kfotsogrp@xsede.org 1672 Sep 15 01:45 ..
-rw-r--r--. 1 kfotso@xsede.org kfotsogrp@xsede.org 400 Sep 15 01:52 hello_mpi.sh
-rw-r--r--. 1 kfotso@xsede.org kfotsogrp@xsede.org 0 Sep 15 01:52 first_mpirun-job.2998325.err
drwxr-sr-x. 2 kfotso@xsede.org kfotsogrp@xsede.org 122 Sep 15 01:52 .
-rw-r--r--. 1 kfotso@xsede.org kfotsogrp@xsede.org 140 Sep 15 01:52 first_mpirun-job.2998325.out
[kfotso@xsede.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
[kfotso@xsede.org@login-ci2 openmpi_hostname]$
```



Ondemand MATLAB

RC will conduct its monthly planned maintenance on Wednesday, September 6. Please visit curc.statuspage.io for details.

Notice: Users will be limited to a maximum of 8 cores per Core Desktop session through mid-September due to ongoing maintenance.

[Home](#) / [My Interactive Sessions](#) / [MATLAB \(Presets\)](#)

Interactive Apps

- Desktops
- Core Desktop (Presets)
- GUIs
- MATLAB (Presets)

Servers

- Jupyter Session (Custom)
- Jupyter Session (Presets)
- RStudio Server (Custom)
- RStudio Server (Presets)
- VS Code-Server (Custom)
- VS Code-Server (Presets)

MATLAB (Presets)

This app will launch a MATLAB GUI on a CURC node. You will be able to interact with MATLAB through a VNC session. GPU based options are not meant for computationally intensive workflows. Additionally, please keep in mind that these GPU based options are a shared resource amongst all users. Thus, significant computation by one user can affect other users of this service.

MATLAB version

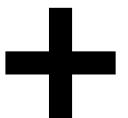
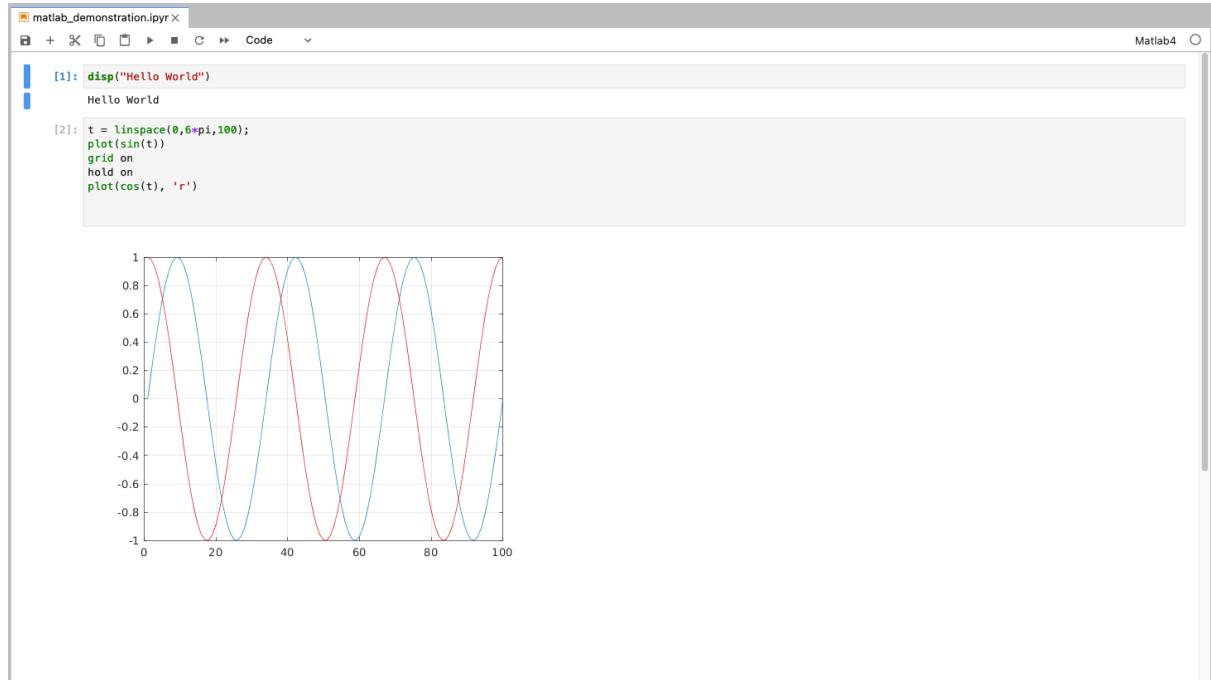
R2021b

Configuration

- 2 cores, 1 hour, K80 GPU
- 2 cores, 1 hour, RTX8000 GPU
- 2 cores, 12 hours, K80 GPU
- 2 cores, 12 hours, RTX8000 GPU
- 4 cores, 4 hours, K80 GPU
- 4 cores, 4 hours, RTX8000 GPU

Ondemand Jupyterlab with MATLAB kernel

- Instructions here:
<https://github.com/kf-cuanschutz/CU-Anschutz-HPC-documentation/blob/main/MATLAB-kernel-on-Jupyterlab.md>
- Possibility to use MATLAB with ssh (if registered)
- MATLAB Ondemand coming soon.



Rstudio with Ondemand

The screenshot shows the XSEDE OnDemand interface for launching an RStudio Server (Custom) session. The top navigation bar includes links for Files, Jobs, Clusters, Interactive Apps (selected), My Interactive Sessions, Help, and Logout.

The left sidebar under "Interactive Apps" shows a list of servers:

- JupyterHub 1 (Presets)
- JupyterHub 2 (Custom)
- RStudio Server (Custom) (selected)
- RStudio Server (Presets)
- JupyterHub 1 (Presets)
- JupyterHub 2 (Custom)
- RStudio Server (Custom) (selected)
- RStudio Server (Presets)

The main content area displays the "RStudio Server (Custom)" configuration page. It includes the following fields:

- RStudio Version:** Rstudio 2023.03.0, R 4.2.2
- Cluster:** Alpine
- Account:** amc-general
- Partition:** ahub
- Number of cores:** 4
- Memory [GiB]:** 4
- QoS Name:** (empty field)

A large black plus sign icon is located in the bottom right corner of the page.

VS Code on Ondemand-rmacc

Home / My Interactive Sessions / VS Code-Server (Custom)

Interactive Apps
Desktops
Core Desktop (Presets)
GUIs
MATLAB (Presets)
Servers
Jupyter Session (Custom)
Jupyter Session (Presets)
RStudio Server (Custom)
RStudio Server (Presets)
VS Code-Server (Custom)
VS Code-Server (Presets)

VS Code-Server (Custom)

This app will launch a [VS Code](#) server using [Code-Server](#). For more information on possible settings for this application, see [Running Custom Interactive applications](#) in our documentation. Additionally, for more information on installing VS Code extensions, please see our [Installing VS Code-Server Extensions](#) section of the documentation.

Cluster

Alpine

Code-Server version

✓ 4.16.1

4.14.1

Account

amc-general

Partition

ahub

QoS Name

interactive

Time

4

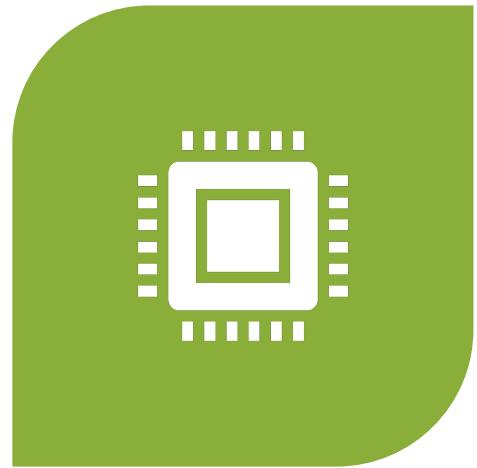
Number of cores

1

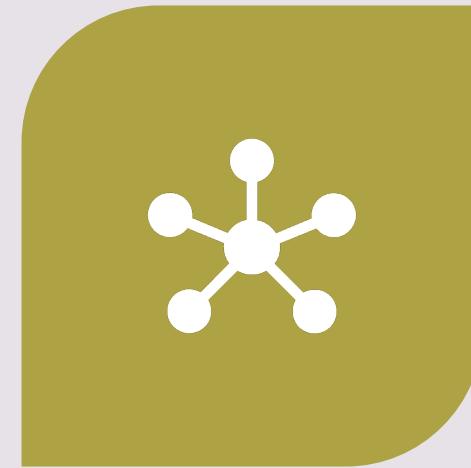
Launch

* The VS Code-Server (Custom) session data for this session can be accessed under the [data root directory](#).

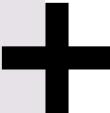
GPU nodes on Alpine



12 NVIDIA GPU NODES
(3 GPU EACH)



8 AMD GPU NODES (3
GPU EACH)



GPU partition on Alpine

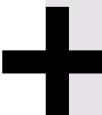
aa100 -> NVIDIA
RELATED PARTITION;
MAX WALLTIME 24
HOURS

ami100 -> AMD
RELATED PARTITION;
MAX WALLTIME 24
HOURS

atesting_a100 ->
NVIDIA TESTING
PARTITION; MAX
WALLTIME 1 HOUR

atesting_mi100 ->
AMD TESTING
PARTITION; MAX
WALLTIME 1 HOUR

Important: to request
more than 24 hours
walltime use:
`--qos=long`

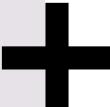


Access to NVIDIA Job partition

```
#!/bin/bash

#SBATCH --job-name=full_genome_nn_hp_opt_gpu
#SBATCH --output=full_genome_nn_hp_opt_gpu.out
#SBATCH --error=full_genome_nn_hp_opt_gpu.err
#SBATCH --partition=aa100
#SBATCH --nodes=1
#SBATCH --ntasks=45
#SBATCH --gres=gpu:1
#SBATCH --account=amc-general
#SBATCH --time=24:00:00
```

- To access an AMD partition add: **--partition=ami100**
- Here, the number of gpus are allocated with: **--gres=gpu:1**

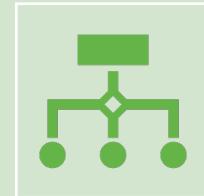


Debug nodes for NVIDIA A100 and AMD MI100



[NVIDIA] sinteractive

```
--partition=atesting_a100 --qos=testing  
--time=00:05:00 --gres=gpu:1 --ntasks=2
```



[AMD] sinteractive

```
--partition=atesting_mi100 --qos=testing  
--time=00:05:00 --gres=gpu:1 --ntasks=2
```

Users are now limited to up to 2/3
of the GPU partition (not per node)

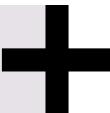


Package availability for ML (2)

- Cuda versions 11.2, 11.3, 11.4, 11.8 and 12.1.1 on Alpine.
- Only cudnn 8.1, 8.2 and 8.6 on Alpine.

GPU

Version	Python version	Compiler	Build tools	cuDNN	CUDA
tensorflow-2.13.0	3.8-3.11	Clang 16.0.0	Bazel 5.3.0	8.6	11.8
tensorflow-2.12.0	3.8-3.11	GCC 9.3.1	Bazel 5.3.0	8.6	11.8
tensorflow-2.11.0	3.7-3.10	GCC 9.3.1	Bazel 5.3.0	8.1	11.2
tensorflow-2.10.0	3.7-3.10	GCC 9.3.1	Bazel 5.1.1	8.1	11.2
tensorflow-2.9.0	3.7-3.10	GCC 9.3.1	Bazel 5.0.0	8.1	11.2
tensorflow-2.8.0	3.7-3.10	GCC 7.3.1	Bazel 4.2.1	8.1	11.2
tensorflow-2.7.0	3.7-3.9	GCC 7.3.1	Bazel 3.7.2	8.1	11.2
tensorflow-2.6.0	3.6-3.9	GCC 7.3.1	Bazel 3.7.2	8.1	11.2
tensorflow-2.5.0	3.6-3.9	GCC 7.3.1	Bazel 3.7.2	8.1	11.2
tensorflow-2.4.0	3.6-3.8	GCC 7.3.1	Bazel 3.1.0	8.0	11.0
tensorflow-2.3.0	3.5-3.8	GCC 7.3.1	Bazel 3.1.0	7.6	10.1



Pytorch installation on Alpine [NVIDIA] (1)

1. After you log into the Alpine cluster, please load the slurm modules and request allocation so that you can install the packages:

```
module load slurm/alpine  
acompile --ntasks=4
```



2. Load anaconda, create your environment with python 3.10 and activate it.

```
module load anaconda  
conda create -n pytorch_env python=3.10  
conda activate pytorch_env
```



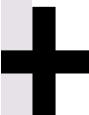
3. Install pytorch, pytorch-cuda. You could also install torchvision and torchaudio if needed for your workflow.

```
conda install pytorch==2.0.0 torchvision==0.15.0 torchaudio==2.0.0 pytorch-cuda=11.8 -c pytorch -c nvidia
```



4. Install cuda-toolkit 11.8.0

```
conda install -c "nvidia/label/cuda-11.8.0" cuda-toolkit
```



Pytorch installation on Alpine [NVIDIA] (2)

5. Install nvidia-cudnn 8.6.0

```
pip install nvidia-cudnn-cu11==8.6.0.163
```

6. To test that your installation is working you will need to exit "acompile" first and load on the NVIDIA gpu debug partition on Alpine"

```
conda deactivate
exit
sinteractive --partition=atesting_a100 --qos=testing --time=00:05:00 --gres=gpu:1 --ntasks=2
module load anaconda
conda activate pytorch_env
python
```

7. Finally, run the following:

```
>>>import torch
>>>print(torch.cuda.is_available())
True
```

8. Make sure to exit the GPU debug node partition after testing the installation.

```
$ exit
exit
```

Tensorflow installation on Alpine [NVIDIA] (1)

1. After you log into the Alpine cluster, please load the Slurm modules and request allocation so that you can install the packages:

```
module load slurm/alpine  
acompile --ntasks=4 --time=01:30:00
```

2. Load anaconda, create your environment with python 3.9 and activate it.

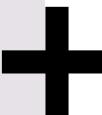
```
module load anaconda  
conda create -n tf_env python=3.9  
conda activate tf_env
```

3. Install cudnn 8.6.0

```
pip install nvidia-cudnn-cu11==8.6.0.163
```

4. Install cuda-toolkit 11.8.0

```
conda install -c "nvidia/label/cuda-11.8.0" cuda-toolkit
```



Tensorflow installation on Alpine [NVIDIA] (2)

5. Install Tensorflow 2.12.0. Also note that tensorflow, cuda and cudnn have to follow a strict versioning: <https://www.tensorflow.org/install/source#gpu>

```
python3 -m pip install tensorflow==2.12.0
```

6. Export the correct paths by following this guide here: <https://www.tensorflow.org/install/pip>

```
mkdir -p $CONDA_PREFIX/etc/conda/activate.d
echo 'CUDNN_PATH=$(dirname $(python -c "import nvidia.cudnn;print(nvidia.cudnn.__file__)"))' >> $CONDA_PREFIX/etc/conda/activate.d/env_vars.sh
echo 'export LD_LIBRARY_PATH=$CONDA_PREFIX/lib/:$CUDNN_PATH/lib:$LD_LIBRARY_PATH' >> $CONDA_PREFIX/etc/conda/activate.d/env_vars.sh
source $CONDA_PREFIX/etc/conda/activate.d/env_vars.sh
export LD_LIBRARY_PATH=$CONDA_PREFIX/lib/python3.9/site-packages/nvidia/cudnn/lib:$LD_LIBRARY_PATH
export LD_LIBRARY_PATH=$CONDA_PREFIX/lib:$LD_LIBRARY_PATH
export PATH=$CONDA_PREFIX/bin:$PATH
export XLA_FLAGS=--xla_gpu_cuda_data_dir=$CONDA_PREFIX
```

7. Install Tensorrt and export PATH:

```
pip install nvidia-tensorrt==8.4.1.5
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$CONDA_PREFIX/lib/python3.9/site-packages/tensorrt
```

Tensorflow installation on Alpine [NVIDIA] (3)

8. We link libnvinfer.so.8 to libnvinfer.so.7 :

```
ln -s $CONDA_PREFIX/lib/python3.9/site-packages/tensorrt/libnvinfer.so.8 $CONDA_PREFIX/lib/python3.9/site-packages/tensorrt/libnvinfer.so.7  
ln -s $CONDA_PREFIX/lib/python3.9/site-packages/tensorrt/libnvinfer_plugin.so.8 $CONDA_PREFIX/lib/python3.9/site-packages/tensorrt/libnvinfer_plugin.so.7
```

9. To test that your installation is working you will need to exit "acompile" first and load on the NVIDIA GPU debug partition on Alpine"

```
conda deactivate  
exit  
sinteractive --partition=atesting_a100 --qos=testing --time=00:05:00 --gres=gpu:1 --ntasks=2  
module load anaconda  
conda activate tf_env  
python3 -c "import tensorflow as tf; print(tf.config.list_physical_devices('GPU'))"
```

10. Make sure to exit the GPU debug node partition after testing the installation.

```
$ exit  
exit
```

Jupyterhub with GPUs

Home / My Interactive Sessions / Jupyter Session (Custom)

Interactive Apps

- Desktops
- Core Desktop (Presets)
- GUIs
- MATLAB (Presets)
- Servers
- Jupyter Session (Custom)**
- Jupyter Session (Presets)
- RStudio Server (Custom)
- RStudio Server (Presets)
- VS Code-Server (Custom)
- VS Code-Server (Presets)

Jupyter Session (Custom)

This app will launch a Jupyter Notebook or JupyterLab session. For more information on possible settings for this application, see [Running Custom Interactive applications](#) in our documentation.

Cluster: Alpine

Account: amc-general

Partition: atesting_a100

Time: 1

Number of cores: 8

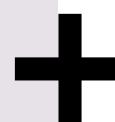
QoS Name: normal

Use JupyterLab instead of Jupyter Notebook?

Launch

* The Jupyter Session (Custom) session data for this session can be accessed under the [data root directory](#).

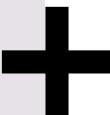
OnDemand version: 2.0.32



Containers (1)

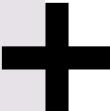


- Apptainer/Singularity can now be built directly on the cluster
- Can be built either from a definition file or converted from a docker image.
- e.g: `apptainer build -f R_spack_env.sif R_spack_env.def`



Containers (2)

- apptainer available on the compute node(s) automatically (`/usr/bin/apptainer`)
- “module load singularity” works as well
- `export ALPINE_SCRATCH=/gpfs/alpine1/scratch/$USER`
- `export SINGULARITY_TMPDIR=$ALPINE_SCRATCH/singularity/tmp`
- `export SINGULARITY_CACHEDIR=$ALPINE_SCRATCH/singularity/cache`
`mkdir -pv $SINGULARITY_CACHEDIR $SINGULARITY_TMPDIR`



Questions?



Let's a go!!

