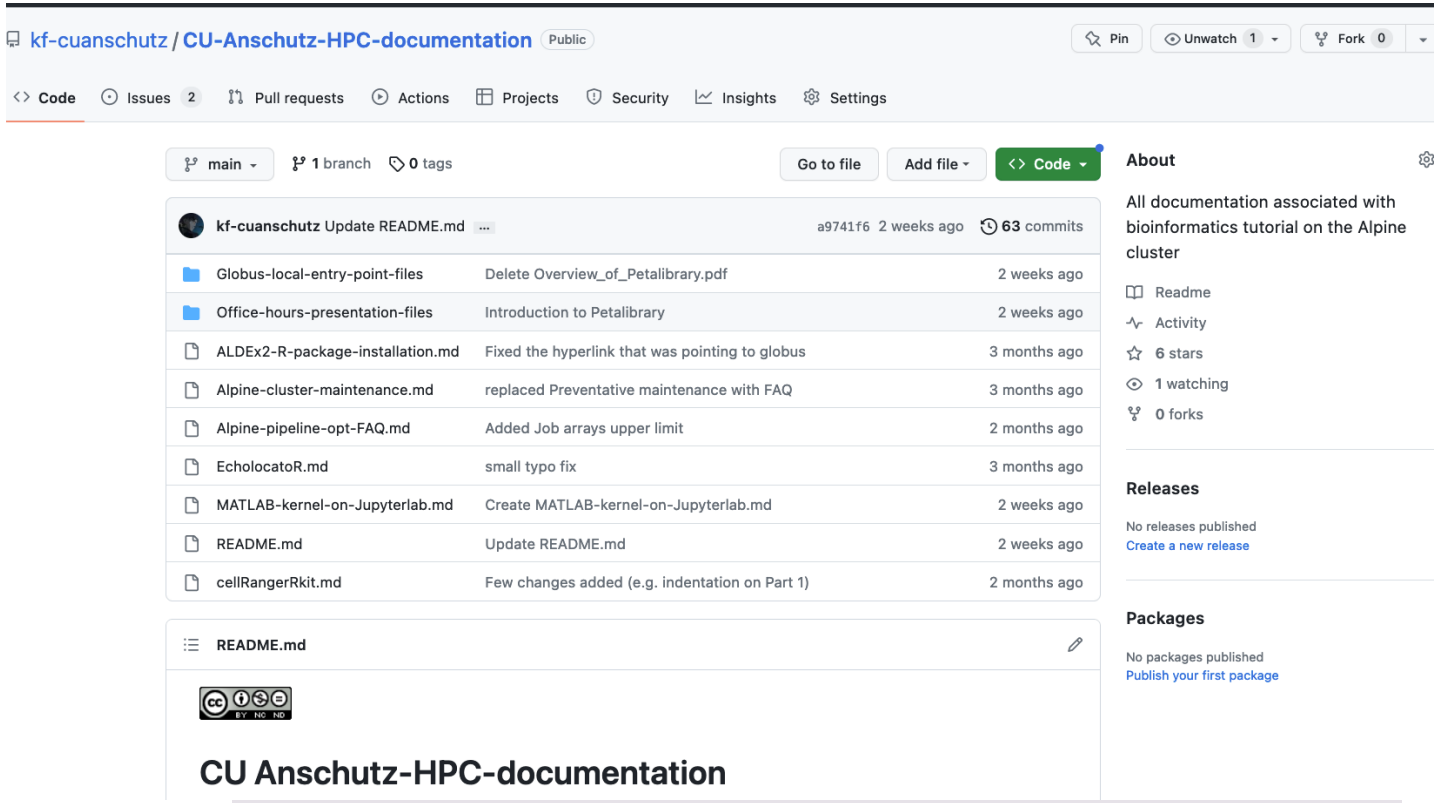
Introduction to Alpine

Kevin Fotso

Alpine computing

Official Github pages:

– CU Anschutz HPC official Github page.



– CU Boulder curc doc:

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

Hardware (1)

- 317 compute nodes and 18,080 nodes officially.
- 184 CPU nodes (HDR IB interconnect)
- 12 high memory nodes (1TB)
- 8 NVIDIA A100 GPU and 8 AMD GPU MI100 nodes. (3 GPUs per node) + (2X25 Ethernet interconnect)
- NVIDIA GPU tend to be more busy but AMD GPU are popular.

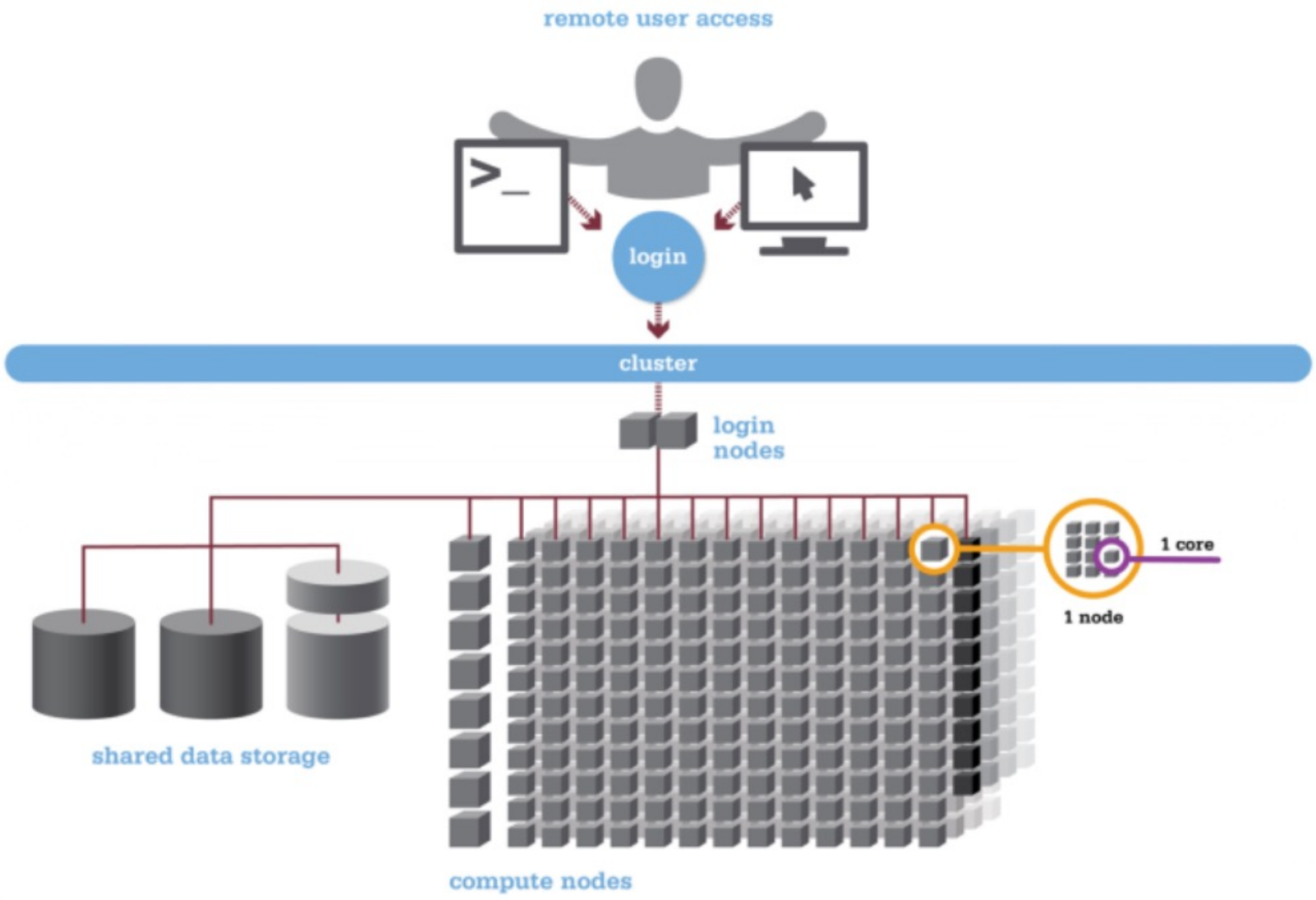
Hardware (2)

GPU debug nodes are now available with --qos=atesting.

1 hour and up to 2 GPUs.

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

Architechture 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 ...

Scheduler Slurm

- **acompile --ntasks=1 --time=00:30:00** to build packages and do some [redacted] testing.
- **sinteractive --ntasks-per-node=2 --nodes=2 --partition=atesting** to test pipelines
- NVIDIA gpu partitions are aa100, amc and atesting_a100.
- AMD gpu partitions are ami100 and atesting_mi100

```
[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.

+

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
```

- Partition is the type of node
- Qos is the quality of service
- ntasks are the number of cores
- Sbatch slurm script

Slurm cheatsheet (1)

Slurm script command	Description
#!/bin/bash	Sets the shell that the job will be executed on the compute node
#SBATCH --ntasks=1 #SBATCH --n1	Requests for 1 processors on task, usually 1 cpu as 1 cpu per task is default.
#SBATCH --time=0-05:00 #SBATCH -t 0-05:00	Sets the maximum runtime of 5 hours for your job
#SBATCH --mail-user= <email>	Sets the email address for sending notifications about your job state.
#SBATCH --mail-type=BEGIN #SBATCH --mail-type=END #SBATCH --mail-type=FAIL #SBATCH --mail-type=REQUEUE #SBATCH --mail-type=ALL	Sets the scedualing system to send you email when the job enters the follwoing states: BEGIN,END,FAIL,REQUEUE,ALL
#SBATCH --job-name=my-named-job	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
#SBATCH --ntasks=1 #SBATCH --cpus-per-task=1	Requests 1 CPU (Serial) cpus-per-task is set to 1 by default and may be omitted.
#SBATCH --cpus-per-task=X #SBATCH --ntasks=1 #SBATCH --nodes=1	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
#SBATCH --ntasks=X #SBATCH --tasks-per-node=X #SBATCH --cpus-per-task=1	Requests for X CPUs and tasks on 1 node cpus-per-task is set to 1 by default and may be omitted.
#SBATCH --ntasks=X #SBATCH --nodes=1 #SBATCH --cpus-per-task=1	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       TIME TIME_LIMI  NODES NODELIST(REASON)
2158225  acompile acompile kfotso@x  RUNNING       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
```

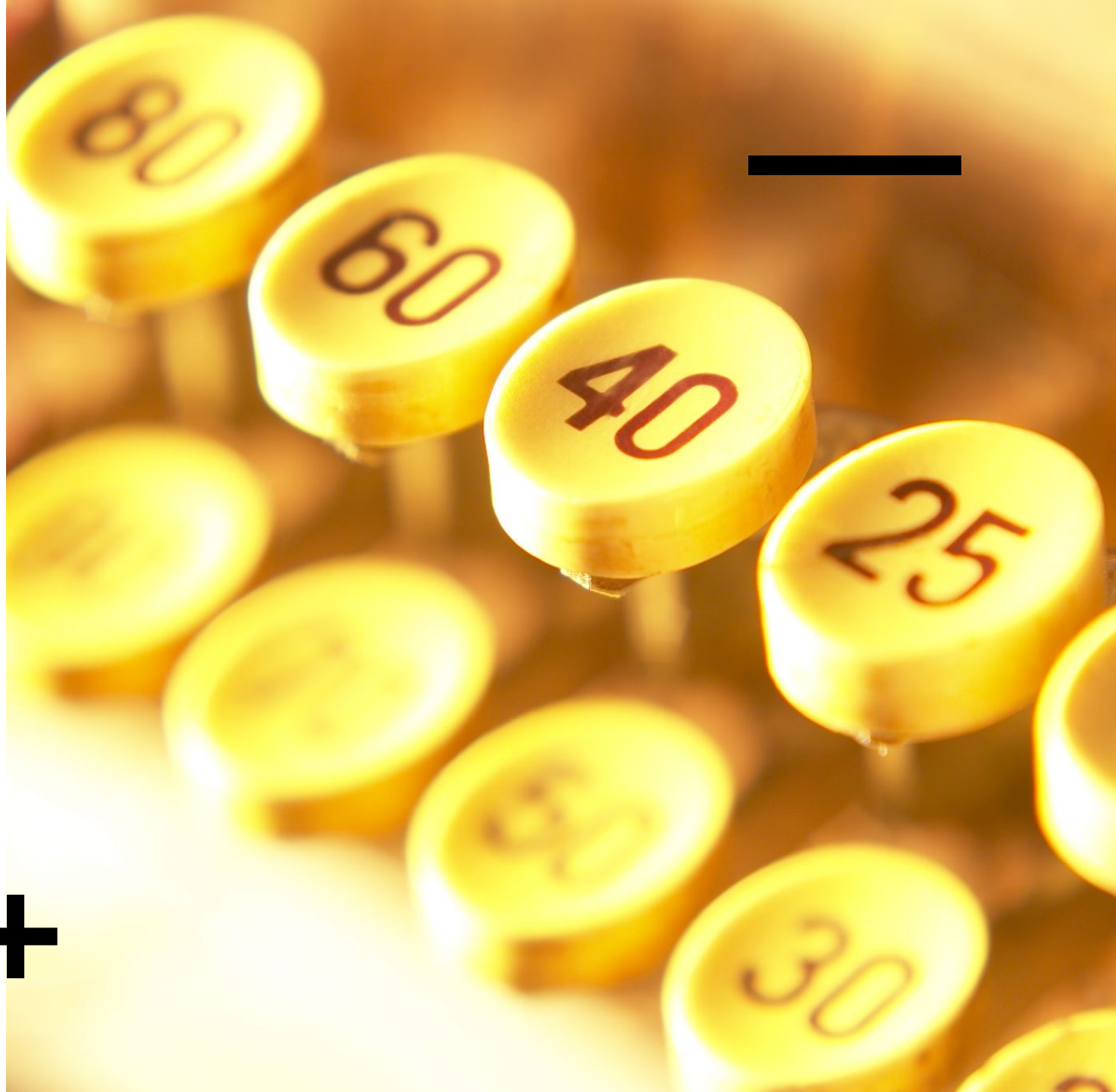
- Allows to get information about a past jobs

```
[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)

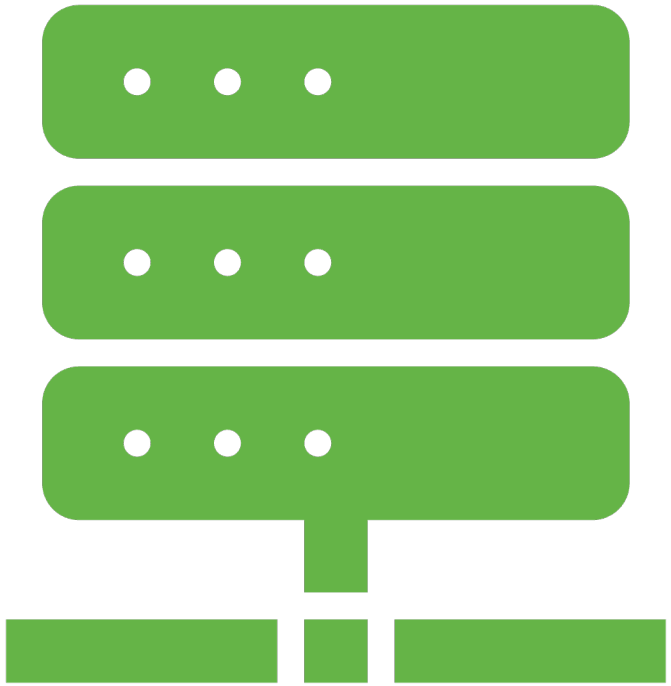


Fairshare overview

- 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; levelfs $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
```

Package availability (1)

Some packages that have been built and accessible through lmod.

Adding new packages through lmod 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.

Containers (1)

- Singularity only and it needs to be built offline and then imported back to the cluster.
- Can be built either from a definition file or converted from a docker image.
- e.g. `sudo singularity -v build splice_conda_v7.sif splice_conda.def`

Containers (2)

- module load singularity
- 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?