RUTGERS

A Quick Intro to the Amarel Cluster

https://rutgers.box.com/v/intro-amarel

Paul Arias

Office of Advanced Research Computing (OARC)

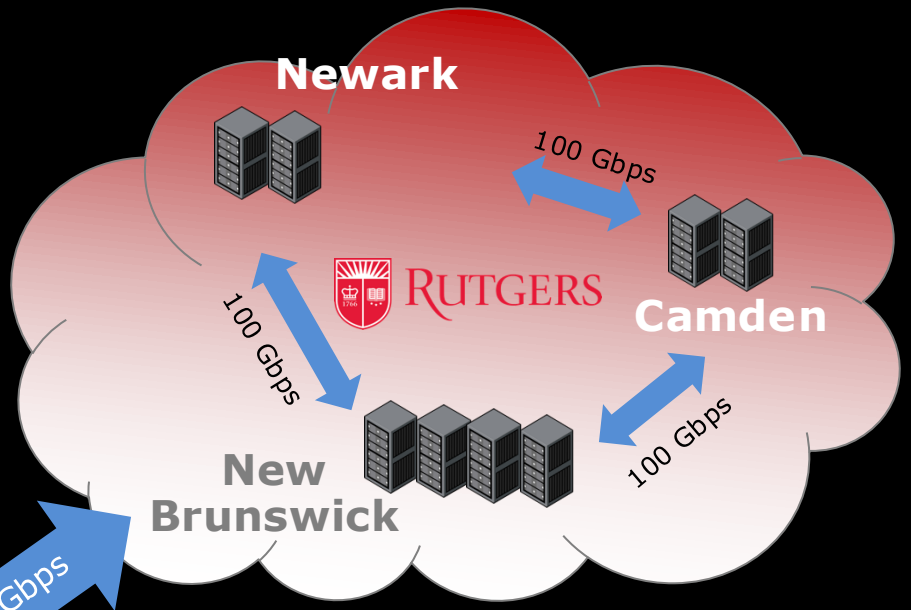
Jan 2025

- University-wide research computing support team
- Manage multiple clusters and storage systems
- Computational scientists are available for consultation and training
- We can help with developing proposals that make use of Rutgers research computing resources: computing, storage, networking, cloud services, etc.

E-mail help@oarc.rutgers.edu

One-Rutgers ACI Ecosystem

= distributed components of the Amarel cluster



100 Gbps

National & Commercial Cloud Services

XSEDE
Extreme Science and Engineering Discovery Environment
Open Science Grid
NIH
CloudLab
Microsoft Azure
geni
Exploring Networks Future
Chameleon
amazon web services
Google Cloud Platform

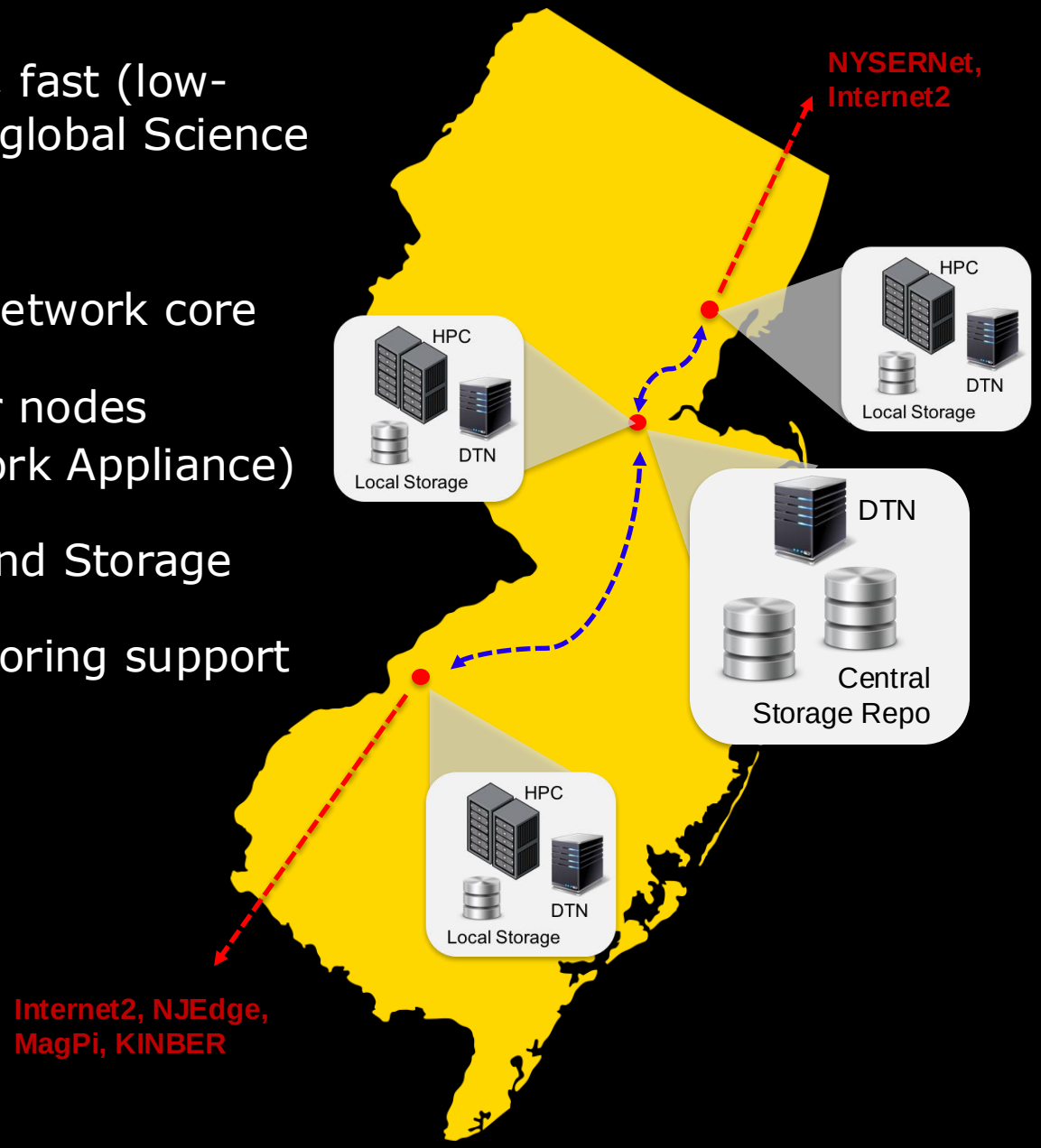
One-Rutgers Cloud Computing Ecosystem

Science DMZ
+
Software-Defined Networking

One-Rutgers Data DMZ

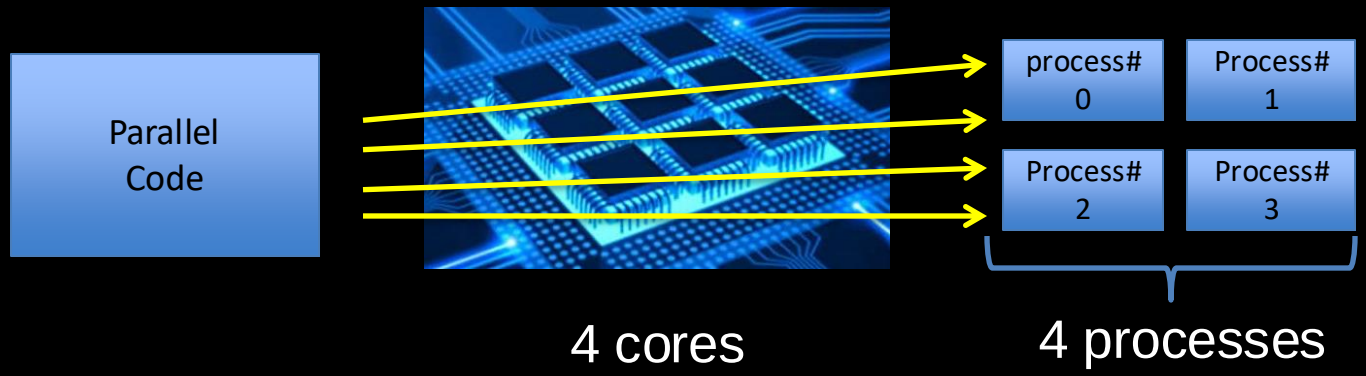
State-wide multi-campus, fast (low-latency) network, part of global Science DMZs:

- SDN-based 100 Gbps network core
- Dedicated data transfer nodes (FIONAs: Flash I/O Network Appliance)
- Advanced Computing and Storage
- Performance and monitoring support (perfSONAR, XDMoD)
- Testbed as a service
- NSF funded CC* (NSF OAC-1659232)



Parallel Computing

- Nearly all software is becoming increasingly parallel
- When you run a parallel program, copies of the parallel parts of the code are distributed amongst the available compute cores



- Each separate task can send & receive data among the other tasks (MPI & OpenMP)
- If tasks on separate nodes must communicate for the overall program to proceed, high-speed networking is needed (only MPI)

Research Computing

- Need for parallel computing or management of "big data" exceeds the capabilities of local workstations
- Dedicated computing resources are the next step
- Enhance or extend current research activities

Local workstations

2 to 8 cores
4 to 16 GB RAM

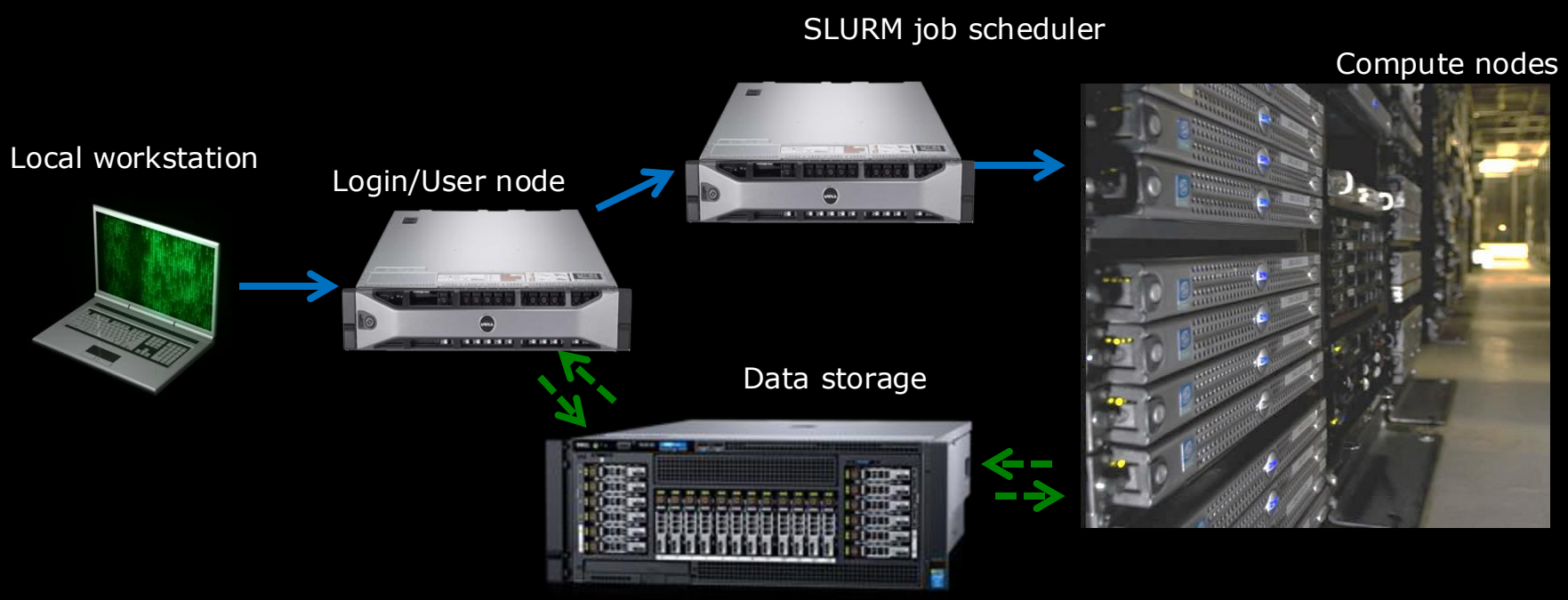
Advanced computing systems

10's to 1000's of cores
16 GB to 2 TB RAM

High-performance computing (HPC)
High-throughput computing (HTC)
Advanced data management systems

Using a Computer Cluster

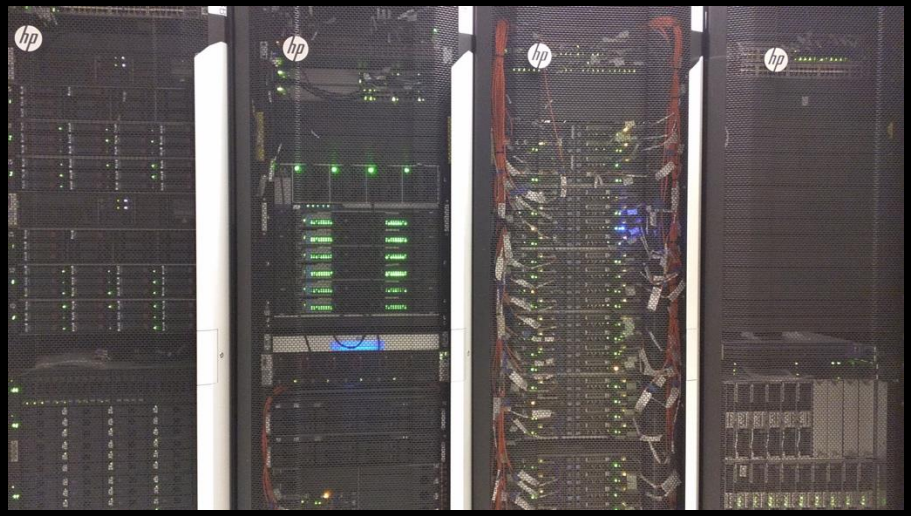
1. Connect to a cluster (SSH), setup your software to run there
2. Move input files/data to the cluster (rsync, scp, sftp)
3. Create a job script (requesting only the hardware you need)
4. Submit your job script to the cluster's resource manager
5. After you job has finished running, collect the output files



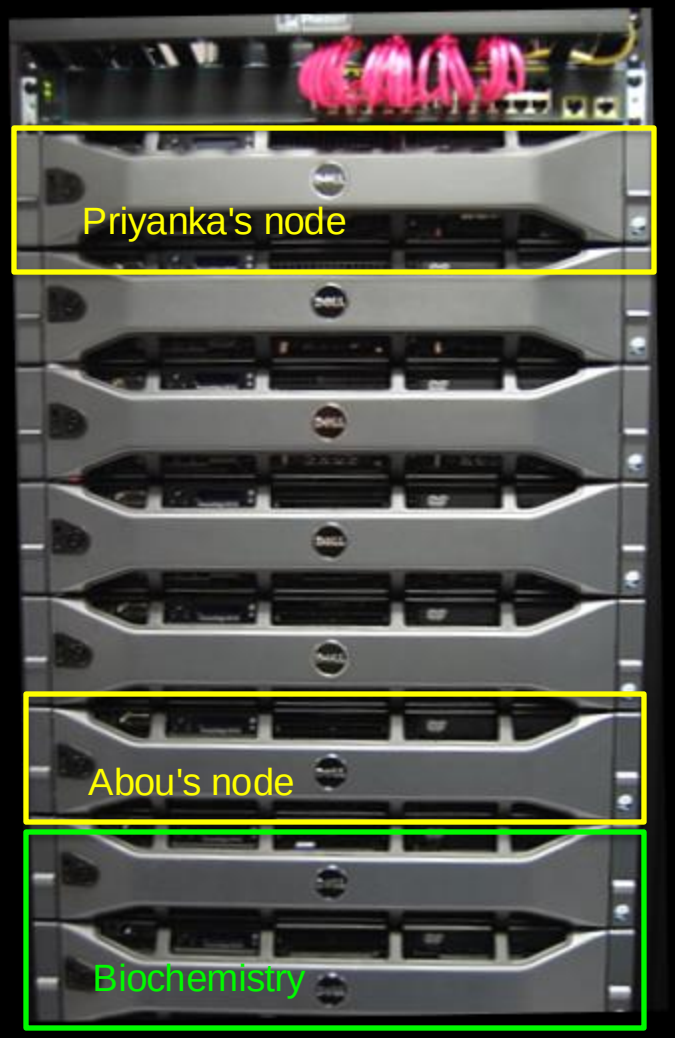
Technical Details (Sample)

- 2 40-core Xeon 6230 nodes each with 2 Nvidia Volta V100 GPUs onboard
- 4 24-core Xeon 4116 nodes each with 8 Nvidia RTX 2080Ti GPUs onboard
- 40+ CPU-only nodes, each with 40 Xeon 6230 (Cascade Lake) cores + 192 GB RAM
- 70+ CPU-only nodes, each with 32 Xeon 6130 (Skylake) cores + 192 GB RAM
- 52 CPU-only nodes, each with 28 Xeon e5-2680v4 (Broadwell) cores + 128 GB RAM
- 20 CPU-only nodes, each with 28 Xeon e5-2680v4 (Broadwell) cores + 256 GB RAM
- 4 28-core Xeon e5-2680v4 nodes each with 2 Nvidia Pascal P100 GPUs onboard
- 2 high-memory nodes, each with 56 e7-4830v4 (Broadwell) cores + 1.5 TB RAM
- 53 CPU-only nodes, each with 16 Xeon e5-2670 (Sandy Bridge) cores + 128 GB RAM
- 5 CPU-only nodes, each with 20 Intel Xeon e5-2670 (Ivy Bridge) cores + 128 GB RAM
- 26 CPU-only nodes, each with 24 Intel Xeon e5-2670 (Haswell) cores + 128 GB RAM
- 4 CPU-only nodes, each with 16 Intel Xeon e5-2680 (Broadwell) cores + 128 GB RAM
- 3 12-core Xeon e5-2670 nodes with 8 Nvidia Tesla M2070 GPUs onboard
- 2 28-core Xeon e5-2680 nodes with 4 Quadra M6000 GPUs onboard

- InfiniBand FDR (56 Gbps) and EDR (100 Gbps)
- 100 GB storage in /home with weekly snapshots
- 1 TB temporary, high I/O storage in /scratch
- 250 to 1000 GB fast local storage at /mnt/scratch



Community Model

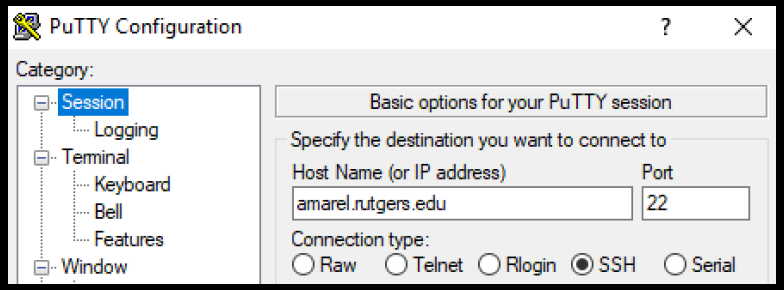


- Using Amarel is FREE, but about 50% of the cluster's compute nodes are "owned" by researchers
- Owners buy dedicated (highest priority) access to a nodes/GPUs using a low-cost 4-year plan
- OARC pays for power, cooling, repair or replacement, system administration, networking...
- When an owned node is not in use, general/main partition jobs can use idle CPU cores
- General/main partition jobs can be preempted by owner jobs, but that should occur rarely because only about 50% owned

Log-in to Amarel

Connect to your Amarel cluster account:

In Windows, launch an SSH client (e.g., PuTTY or MobaXterm):



Host Name: amarel.rutgers.edu

Click "Open"

Login as: [your NetID]

Off campus? Connect to the campus VPN first: https://soc.rutgers.edu/vpn

In OS X, launch a Terminal and SSH to Amarel:

Applications > Utilities > Terminal

ssh [your NetID]@amarel.rutgers.edu

In Linux or Unix, launch a Terminal and SSH to Amarel:

(in Ubuntu, Ctrl-Alt-T)

ssh [your NetID]@amarel.rutgers.edu

Shells and Bash

Bash (Bourne-again shell) = default shell (command processor) in Linux.

```
[gc563@amarel1 ~]$ which bash
/usr/bin/bash
[gc563@amarel1 ~]$ bash --version
GNU bash, version 4.2.46(1)-release (x86_64-redhat-linux-gnu)
Copyright (C) 2011 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>

This is free software; you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
[gc563@amarel1 ~]$
```

Bash uses 2 configuration files, .bashrc & .bash_profile and only .bashrc is processed every time a new shell instance is started.

Every time to connect to a different computer/server/node, you're starting a new shell.

Local workstation

Login/User node

Each compute node

The /scratch Filesystem

- Located at /scratch/*NetID*
- Physically located near execution nodes (varies)
- Temporary work directory for all jobs
- Specialized high-performance hardware
- Designed to handle very high I/O activity and large files
- 1 TB storage space (2 TB hard limit with 2-week grace period)
- Move files in, run jobs, move files out

Protecting Important Data

Free, backed-up storage options available to Rutgers students, staff, and faculty:

- Unlimited storage in box.rutgers.edu, https://box.rutgers.edu for all students, staff, faculty (15 GB file size limit, but larger files can be split before uploading)
- 5 TB of space in Office365 OneDrive, https://it.rutgers.edu/rutgers-connect/knowledgebase/onedrive-for-business for all staff, faculty, and RBHS students (10 GB file size limit, but larger files can be split before uploading)
- Unlimited storage in Google Drive for ScarletApps users, https://it.rutgers.edu/scarletapps (5 TB file size limit, but larger files can be split before uploading)
- Backed-up /projects storage on Amarel, $150/TB for 4 years for compute node owners

Also, be mindful of where it's safe to store your data:

https://box.rutgers.edu/data-classification-and-storage-matrix

Copy the Example Files

Copy example files to your /home directory:

```
cd  /home/[NetID]

cp  -r  /projects/oarc/users/training/intro.amarel  .

cd  intro.amarel

ls
```

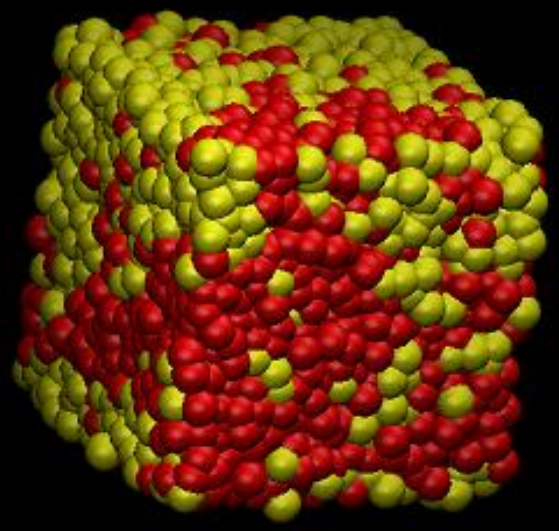
Don't forget the "."

It means copy the files "here"
(to my current location)

```
[gc563@amarel1 intro.amarel]$ ls
amber.example            gethostname.mpi.c      matlab.example
bashrc.modified.example  hello_world_mpi.c      namd.example
bashrc.new               hello_world_openmp.c   pi.integ.c
gaussian.example         lammps.example         pi.integ.mpi.c
gethostname.c            mathematica.example    run.hello_world_openmp
```

Run an Example Job

- LAMMPS is a molecular dynamics simulation program
- Cooling of a binary mixture
- 5,000 atoms (L-J interactions)
- 500,000 0.005 ps steps



```
cd intro.amarel/lammps.example
```

Edit run.lammps.binary (optional), then run the job:

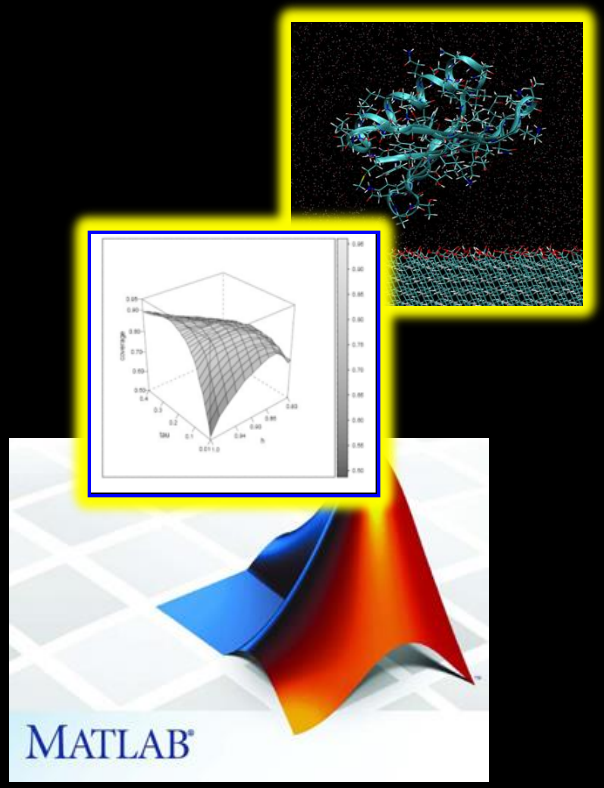
```
sbatch run.lammps.binary

squeue -u [NetID]

sstat --format=MaxRSS,MaxDiskRead,MaxDiskWrite -j [JobID]
```

Software

- Commercial & open-source software:

BLAST, Bowtie, BWA, CUDA Toolkit, Gaussian, GCC compilers, GROMACS, Intel compilers, LAMMPS, MATLAB, NAMD, PGI compilers, SAMtools, TopHat, TrinityRNAseq, and many more…

- Only current & previous major release versions are maintained
- Install your own software (libraries and executables)
- The OARC research support team can help you get this done



Using Software Modules

- To get a list of available software:

  module avail

  module spider (for more options & details)

- Check for prerequisites:

  module spider intel/17.0.4

- Load (add) modules:

  module load intel/17.0.4 mvapich2/2.1

- List your loaded modules:

  module list

- To clear-out your added modules:

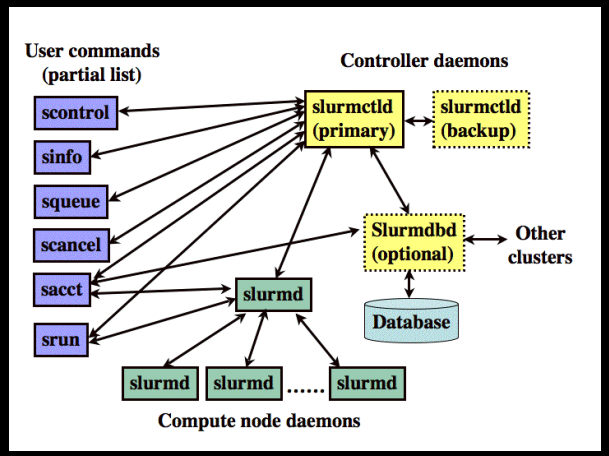
  module purge

You can add your 'module load ...' command(s) to your .bashrc file

Resource Management

SLURM = resource manager / job scheduler

- Enables scripting of computational tasks
- SLURM runs these tasks on compute nodes and returns the results (output files)
- If the cluster is full, SLURM holds your tasks and runs them when the resources are available
- SLURM ensures fair sharing of cluster resources (policy enforcement)



Basic SLURM Commands

Command(s)	Description
sinfo -a --summarize	View nodes and partition info
sbatch job-script [options]	Submit/setup a batch job
srun [options] program_name	Run a program (exe, application)
squeue -u NetID	Check status of job submissions
sstat -u jobID	Check status of a running job
sacct --format [options] -j jobID	See accounting details of current and completed jobs

https://slurm.schedmd.com/pdfs/summary.pdf

Job Script (MPI + GPUs)

```
#!/bin/bash
#SBATCH --clusters=amarel
#SBATCH --partition=gpu
#SBATCH --job-name=CH3Phx
#SBATCH --nodes=2
#SBATCH --ntasks=16
#SBATCH --gres=gpu:2
#SBATCH --cpus-per-task=1
#SBATCH --constraint=fdr,pascal
#SBATCH --mem=118G
#SBATCH --time=5:00:00
#SBATCH --output=slurm.%N.%j.out
#SBATCH --error=slurm.%N.%j.err
#SBATCH --mail-user=[NetID]@rutgers.edu
#SBATCH --mail-type=BEGIN,END,FAIL
#SBATCH --requeue
#SBATCH --export=ALL

srun --mpi=pmi2 my-exe input > output
sacct --format MaxRSS,Elapsed -j $SLURM_JOBID
```

This is a "long" example. You likely will not need all of these options.

Job Script (OpenMP/Java)

```
#!/bin/bash
#SBATCH --clusters=amarel
#SBATCH --partition=nonpre
#SBATCH --job-name=Efexa
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=16
#SBATCH --exclusive
#SBATCH --mem=118G
#SBATCH --time=5:00:00
#SBATCH --output=slurm.%N.%j.out
#SBATCH --error=slurm.%N.%j.err
#SBATCH --mail-user=[NetID]@rutgers.edu
#SBATCH --mail-type=BEGIN,END,FAIL
#SBATCH --export=ALL

OMP_NUM_THREADS=16
srun my-exe -t 16 input > output
sacct --format MaxRSS,Elapsed -j $SLURM_JOBID
```

Note: --mem=0 means "use all available RAM" but that's only the RAM that's not allocated to other jobs

Setup E-mail Notifications

```
#SBATCH --mail-user=[NetID]@rutgers.edu

#SBATCH --mail-type=[type]
```

Where [type] can be one of the following:

NONE, BEGIN, END, FAIL, REQUEUE, ALL

Note: only [NetID]@rutgers.edu will work

Non-Rutgers e-mail addresses or [NetID]@xxxxx.rutgers.edu will not work

Killing Batch Jobs

Kill just one, or just a few jobs:

```
scancel [jobID] [jobID] [jobID] ...
```

Kill all of your jobs:

```
scancel --user=[NetID]
```

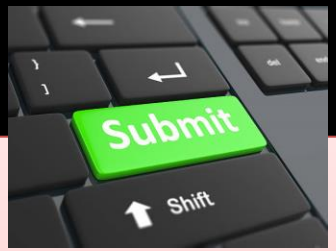
Kill all of your *queued* jobs:

```
scancel --user=[NetID] --state=PENDING
```

Kill all of your *running* jobs:

```
scancel --user=[NetID] --state=RUNNING
```

Batch vs. Interactive



Batch Job	**Interactive Job**
sbatch *job-script*	**srun** [required resources] *your.exe*
Starts when requested resources are available	Starts when requested resources are available
Runs "in the background" Use **srun** to launch tasks inside your script	You are actively logged-in (running a shell) on a compute node
Terminate batch jobs using **scancel** *jobID*	Terminate interactive jobs by simply logging-out (using **exit** or **CTRL-D**)
Useful for jobs that will run for a long time, and for jobs that don't require interaction or supervision	Useful for testing, compiling code, computational steering, etc.

Interactive Jobs

Useful for debugging applications or running short tests

Single-core / serial task example:

```
srun --time=2:00:00 --pty bash
```

exit (when finished)

Multi-core / parallel task example:

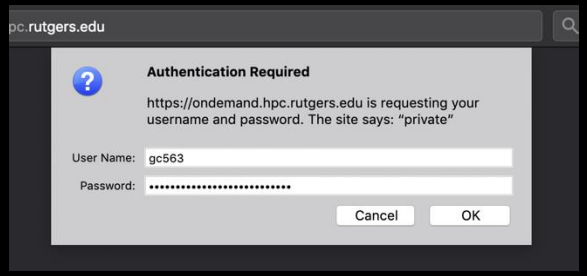
```
srun --ntasks=4 --mpi=pmi2 –time=2:00:00 --pty bash
```

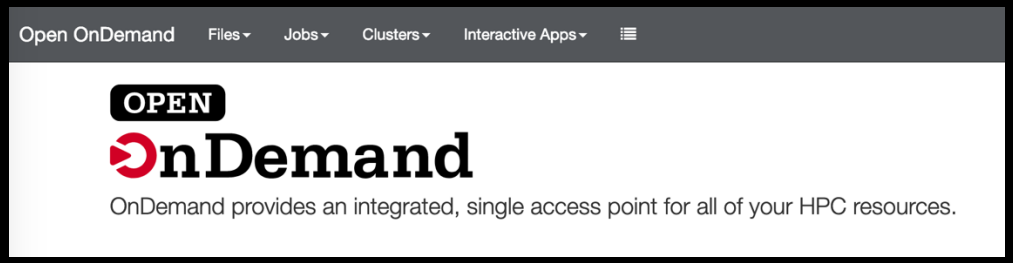
exit (when finished)

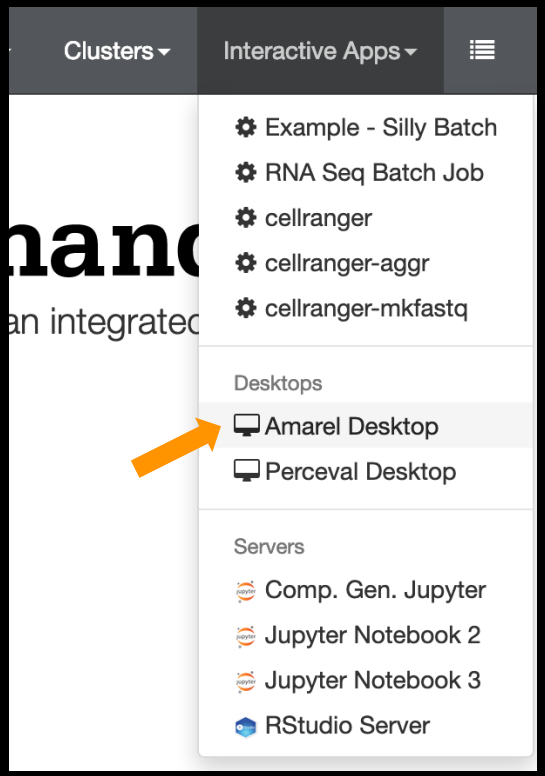
OnDemand

Open OnDemand offers web browser access to Amarel for running GUI-based applications

Go to https://ondemand.hpc.rutgers.edu/







This approach runs your session using a Singularity image (i.e., you're working inside a container), already on a compute node.

Tools/commands available from the command-line are limited to what's already installed in this Singularity image, so SLURM commands like srun, sbatch, sacct are not available.

FastX also offers web browser access to Amarel for running GUI-based applications

Go to https://amarel.rutgers.edu:3443

Confirm/accept the security exception

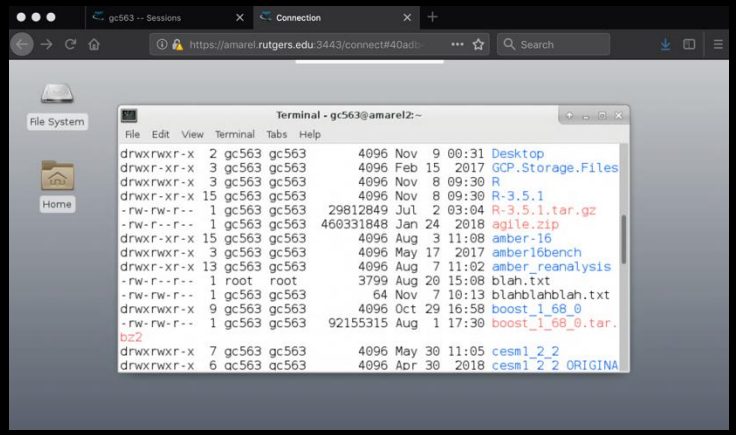
Log-in to Amarel

Choose 'Launch Session' and select an XFCE Desktop, then click the Launch button. You'll be launching a GUI on the Perceval login node using FastX.

On the desktop of your FastX session, right-click and choose 'Open Terminal Here'

In that terminal window, launch an interactive session on a compute node with X11 tunneling enabled as you normally would. For example,

```
srun --time=1:00:00 --pty bash
```



Use ALT-TAB to find minimized windows

Defaults and Limits

Default settings (in case you don't specify something):

--ntasks=1

--cpus-per-task=1

--mem-per-cpu=4G (per CPU core requested)

--time=00-00:02:00

Limits / upper bounds:

- Max # CPUs in-use per user for 'main' partition jobs = 504
- Max # of jobs a user can submit = 500 (this also applies to job arrays), owners have a 2000 job limit for each partition
- Currently no limit for owners, but system-wide limit of 10,000 jobs creates a variable limit

Getting Info from SLURM

SLURM can provide details about jobs and compute resources. Here are some examples:

```
squeue

sinfo

sinfo -p p_bubba_1

sinfo -n gpu037 --format="%7N %9P %.5a %.2l %.12g %.5D %.7T %6m %7G”

scontrol show partition p_foran

scontrol show job 8012395

sacct

sacct -j 8812395 --format=ReqMem,MaxRSS,NodeList,State,Elapsed,WorkDir

sacct --user=abc123 --starttime=2021-02-10 --
format=JobID,Partition,ReqMem,MaxRSS,NodeList,Elapsed,MaxDiskRead,MaxDiskWrite,State
,WorkDir%60
```

Software Installation

Most programs that you download and install in a Linux environment can be installed using a procedure similar to these steps:

```
wget https://www.python.org/ftp/python/3.6.8/Python-3.6.8.tgz
tar -zxf Python-3.6.8.tgz
cd Python-3.6.8
./configure --prefix=/home/[NetID]/python/3.6.8
make -j 4
make install
export PATH=/home/[NetID]/python/3.6.8/bin:$PATH
which python3
```

The actual steps may vary a little (see the README file), but the overall concept should be the same.

Software Installation

Configure your environment for using a new package:

These are just examples, you may not need all of these environment variables. Also, each long line here is a separate "export…" command.

```
export PATH=/home/[NetID]/python/3.6.8/bin:$PATH

export LD_LIBRARY_PATH=/home/[NetID]/python/3.6.8/lib:$LD_LIBRARY_PATH

export LIBRARY_PATH=/home/[NetID]/python/3.6.8/lib:$LIBRARY_PATH

export C_INCLUDE_PATH=/home/[NetID]/python/3.6.8/include:$C_INCLUDE_PATH

export CPLUS_INCLUDE_PATH=/home/[NetID]/python/3.6.8/include:$CPLUS_INCLUDE_PATH

export MANPATH=/home/[NetID]/python/3.6.8/share/man:$MANPATH
```

Add these lines to the end of your ~/.bashrc file to make the changes persist for all shells & subshells (interactive jobs)

Note: omit the :$------ part if a variable isn't already set

I can't install my software on Amarel (CentOS 7) because

(1) it requires newer core system libs (e.g., GLIBCXX_3.4.20+),

(2) my software was designed for a different OS,

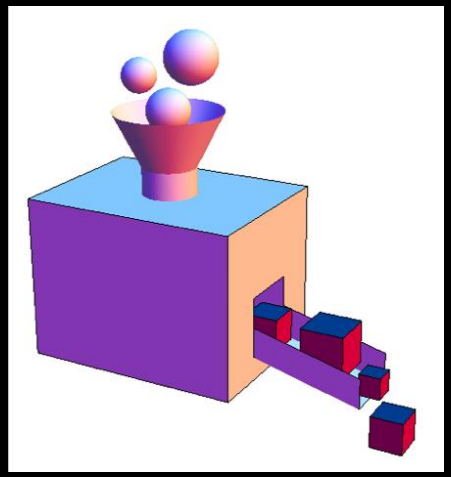
(3) I need root/admin privileges and apt-get or yum to install my software and a ton of dependencies.

1. Use a Linux workstation or VM where you have root/admin privileges, or the Sylabs Remote Builder (https://cloud.sylabs.io/builder) to create a Singularity image,
2. Install your software in that Linux OS image (e.g., Ubuntu, Fedora, many more),
3. Run your customized image as a container in a job run on Amarel

Job Arrays

Processing many input files with the same exe & job requirements

```
#SBATCH --partition=main
#SBATCH --job-name=ARRx1
#SBATCH --array=1-240
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=4
#SBATCH --mem=6000
#SBATCH --time=8:00:00
#SBATCH --requeue

OMP_NUM_THREADS=4
srun my-exe {$SLURM_ARRAY_TASK_ID}.in > {$SLURM_ARRAY_TASK_ID}.out
```



Input files:
1.in, 2.in, 3.in, ...

Output files:
1.out, 2.out, 3.out, ...

Note: sacct & sstat commands won't work for array jobs

Job Dependencies

You can defer the start of a job until some specified dependency has been satisfied (e.g., start job 2 after job 1 completes)

```
[gc563@amarel1 ~]$ sbatch run.matlab.MonteCarloPi
Submitted batch job 8827845 on cluster amarel
[gc563@amarel1 ~]$ sbatch --dependency=afterok:8827845 run.matlab.MatrixMultGPU
Submitted batch job 8827853 on cluster amarel

[gc563@amarel1 matlab.example]$ squeue -u gc563
             JOBID PARTITION     NAME     USER ST       TIME  NODES
   NODELIST(REASON)
           8827853       gpu  mtrxmlt    gc563 PD       0:00      1 (Dependency)
           8827845      main  mparfor    gc563  R       0:31      1 slepner071
```

after:jobid[:jobid...] job can begin after the specified jobs have started

afterany:jobid[:jobid...] job can begin after the specified jobs have terminated

afternotok:jobid[:jobid...] job can begin after the specified jobs have failed

afterok:jobid[:jobid...] job can begin after the specified jobs have run to completion with an exit code of zero (see the user guide for caveats).

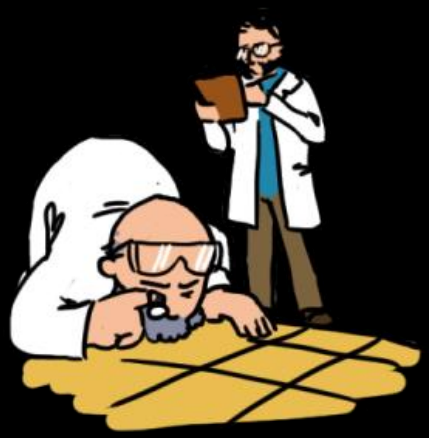
singleton jobs can begin execution after all previously launched jobs with the same name and user have ended. This is useful to collate results of a swarm or to send a notification at the end of a swarm.

Help Jobs Start Sooner

- Keep your jobs small, use minimum cores & memory
- Keep your jobs short (shorter wall time / time limit)
- Be flexible: highly specific hardware requirements may take longer to allocate
- Think about memory request boundaries (ask for about 8 GB less than the max)

Running Test Jobs

- Start with a small example to verify that your job will run successfully
- Test parallelization on 1 node (multiple cores) first
- How much memory does my job need? Run some test jobs and look at the detailed job output using `sacct --format=MaxRSS,MaxDiskRead,MaxDiskWrite,Elapsed \ --units=G -j [JobID]` to see how much memory, I/O, and time your job used
- *My job just dies without producing output!* Maybe you didn't allocate enough memory for it to run properly

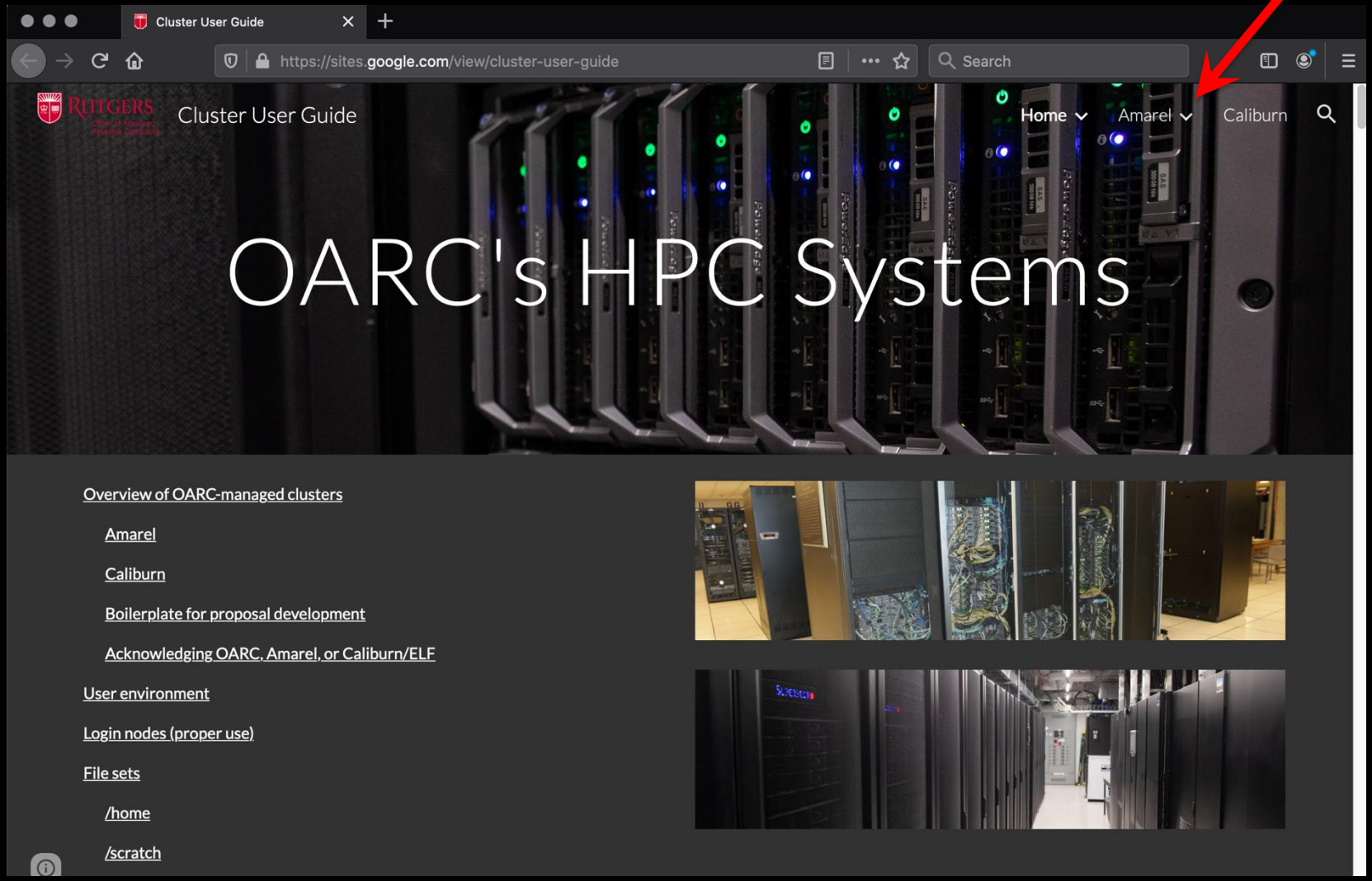


Sharing Resources

- The log-in / head / master node (amarel1, amarel2, etc.) is a shared system
- Avoid compute-intensive or memory-intensive operations there
- Do not run applications (R, Python, Perl, MATLAB, etc.) on the log-in node… that's what compute nodes are for
- On compute nodes, remember that you may not be the only user on a given node

Amarel User's Guide

https://sites.google.com/view/cluster-user-guide



Amarel Ownership

Note: ownership of Amarel resources is certainly not required because all Rutgers community members have **FREE** access

However, benefits of buying-in:

- Dedicated, immediate access to purchased resources via private job queue (partition)
- No limit on the # of jobs running
- Free 1 TB of project storage + access to additional storage purchases
- No indirect costs, this is an equipment purchase / 4-year ownership plan
- https://oarc.rutgers.edu/services/condo-model

Learning to Use Applications

https://libguides.rutgers.edu/graduatespecialist/workshops

https://libguides.rutgers.edu/graduatespecialist/workshops

RUTGERS University Libraries

Hours | My Account | Ask Us

Find | Services & Tools | Get Help | Places & Spaces | Info for... | About

Library » Research Guides » Graduate Specialist Program (New Brunswick Libraries) » Workshops

Graduate Specialist Program (New Brunswick Libraries): Workshops

Home page for the New Brunswick Libraries' Graduate Specialist Program

Home | Consulting Hours | Workshops | Workshops - Date order | Python | Text Analysis in R | NVivo

For workshops in chronological order click **here**.

See below for workshops by topic.

Additional workshops on High Performance Computing are offered by OARC. See their **calendar** and **resources** for more information.

- ✓ Python workshops
- ✓ Statistical software (SAS and R)
- ✓ NVivo workshops
- ✓ Digital humanities workshops

Python Workshops

☞ RSVP for the Python workshops.

Workshops are offered in either Alexander Library or LSM (with identical content). Participants in LSM-based workshops must bring their own laptops. At Alexander, you can either bring your own laptop, or use the desktops in the lab.

Python Basics and Data Exploration

- Friday, February 1 – 12:30-2:30 pm, Alexander Library Room 413 (Instructor, Hang Miao)
- Wednesday, February 13 – 12:00-2:00 pm, LSM Conference Room (Instructor, Aditya Vas)

This workshop will be an accelerated introduction to fundamental concepts such as variable assignment, data types, basic calculations, working with strings and lists, control structures (e.g. for-loops), functions. We will also start working with pandas, a popular data science library in Python, to explore a dataset on foodborne outbreaks reported to the CDC.

Data Manipulation and Analysis with Python

- Friday, February 8 – 12:30-2:30 pm, Alexander Library Room 413 (Instructor, Hang Miao)
- Monday, February 18 – 3:15-5:15 pm, LSM Conference Room (Instructor, Aditya Vas)

NVivo Workshops

☞ RSVP for the NVivo workshops.

Introduction to NVivo

- Tuesday, April 3 – 2:00-3:30 pm, Alexander Library Room 413 (Instructor, Cathy Wang)

This workshop is intended as a basic introduction to using NVivo, a software that supports qualitative and mixed methods research. The workshop focuses on introducing key mechanisms of the software that may be applied as required by different analytical approaches.

Advanced NVivo: Data Visualizations Using NVivo 12

- Tuesday, April 17 – 2:00-3:30 pm, Alexander Library Room 413 (Instructor, Cathy Wang)

This workshop focuses on the introduction of a suite of visualizations that help you gain deeper insights from your data by exploring and unearthing patterns, trends and connections.

Digital Humanities Workshops

Documentation & Notes

Amarel User's Guide

https://sites.google.com/view/cluster-user-guide

Final Notes:

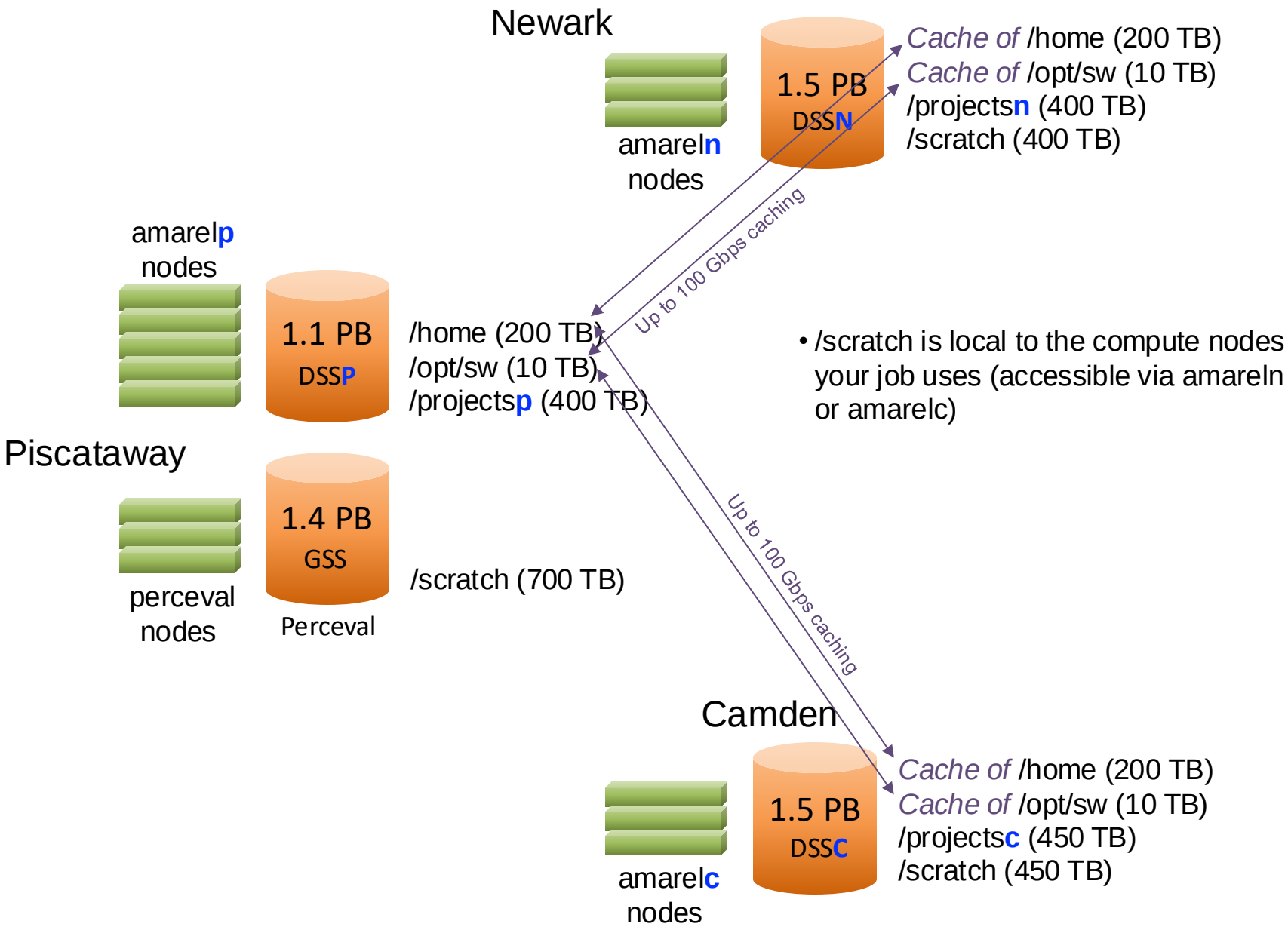
- Do not run compute-intensive or memory-intensive applications on the log-in nodes (amarel1, amarel2, etc.)
- For best performance, work in /scratch, then move files when finished
- Need help?

  E-mail help@oarc.rutgers.edu

  - Please remember to do the following in your emails:
  - Specify the cluster in question (Amarel, Perceval, Didact, etc.)
  - Specify your Job ID (if you have one)
  - Paste complete error messages in the email (if you have one)
  - Use an official Rutgers e-mail only (others will be disregarded)

Backup Slides

Storage on Amarel



- /scratch is local to the compute nodes your job uses (accessible via amareln or amarelc)

Research Commercialization

Transforming research into products, services and partnerships generating value for the University and enhancing economic development in the State of New Jersey.

Licensing and Intellectual Property	Manage technology lifecycle for Rutgers inventions from conception through patenting, marketing and licensing
SoCrates	Support for faculty development and distribution of software and creative works
New Ventures	Create and support startup companies based on university technology
TechAdvance Fund	Advance promising technologies toward commercialization through an early-stage-technology fund.
Internship	Opportunities for postdoctoral and senior graduate students interested in intellectual property, legal matters, business development, and/or academic technology transfer.

The SoCrates Program

- Educate faculty & staff
- Advise on Open Source licensing
- Patent prosecution (if applicable)
- Marketing and licensing
- Commercialization including start-up support
- Express licensing
- Sponsored Research and Grants support

Compile & Run, Example #1

Try compling and running gethostname.c (serial)

```
#include <stdio.h>
#include <sys/utsname.h>

int main ( )
{
    struct utsname uts;
    uname (&uts);
    printf (“My task ran on node %s.\n”, uts.nodename);
      return 0;
}
```

Compile & Run, Example #1

Compile commands:

```
module load intel/17.0.4
icc gethostname.c -o gethostname.exe
```

Run interactively:

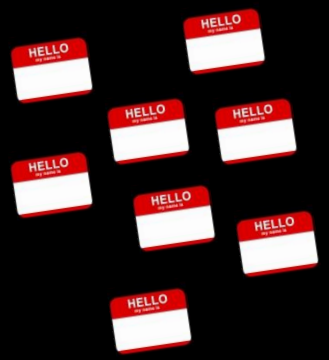
```
srun gethostname.exe
```

Or… SLURM job script for gethostname.exe (serial):

```
#SBATCH --job-name=gethost
srun gethostname.exe
```

Compile & Run, Example #2

Try compling and running gethostname.mpi.c (parallel)

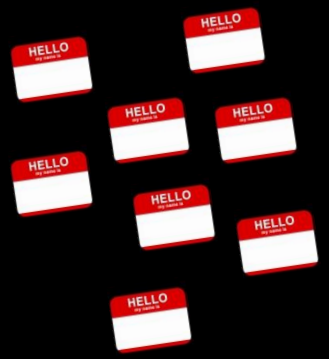



```
#include <stdio.h>
#include <sys/utsname.h>
#include <mpi.h>

int main (int argc, char *argv[])
{
    struct utsname uts;
    int rank;

    MPI_Init (&argc, &argv);
    MPI_Comm_rank (MPI_COMM_WORLD, &rank);

    uname (&uts);
    printf ("Task %d on node %s.\n", rank, uts.nodename);

    MPI_Finalize ();
        return 0;
}
```

Compile & Run, Example #2

Compile commands:

```
module load intel/17.0.4 mvapich2/2.1
mpicc gethostname.mpi.c -o gethostname.mpi.exe
```



Run interactively:

```
srun --ntasks=8 --mpi=pmi2 gethostname.mpi.exe
```

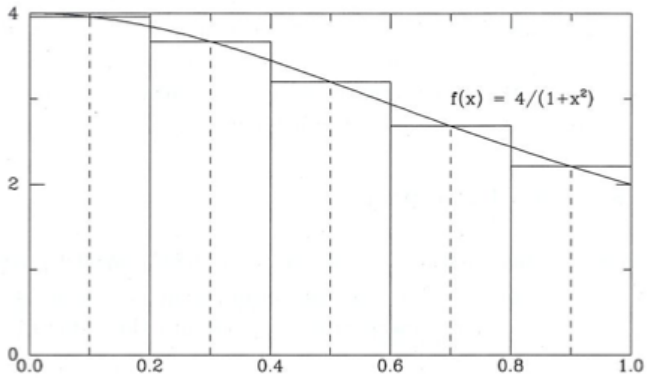
Or… SBATCH job script for gethostname.mpi.exe (parallel)

```
#SBATCH --job-name gethost2
#SBATCH --ntasks=8
srun --mpi=pmi2 gethostname.mpi.exe
```

Compile & Run, Example #3

Calculating pi by numerical integration:

$$\int_0^1 \frac{1}{1+x^2}\,dx = \arctan(1) - \arctan(0) = \arctan(1) = \frac{\pi}{4}$$



$$\pi \approx \left(\sum_{n=1}^{\#\,intervals} \frac{4}{(1+x^2)} \right) \Big/ \#\,intervals$$

$$\text{where } x = \frac{interval - 0.5}{\#\,intervals}$$

`pi.serial.c` vs. **`pi.mpi.c`**

Compile serial and parallel versions and compare performance.

Reservation for Today's Class

Special reservation for today = **xxxxxxx**

- For a batch job:

```
sbatch --reservation=xxxxxxxx my-job-script
```

- For an interactive job:

```
srun --reservation=xxxxxxxx --time=03:00:00 --pty bash
```