




1



## CLUSTER ADMINISTRATION & SUPPORT

SYSTEM ENGINEERS | OIT SERVICES

**hpcadmin@auburn.edu**  
[hpc.auburn.edu](http://hpc.auburn.edu)

**HPC Systems Engineers:** Kris Garner  
Frank Henderson  
Matt Smith  
Keenan Terry

**Director:** Jeff Stallworth

**Associate Director:** Bradley Morgan

3

3

TODAY'S TOPIC...

## **HPC FUNDAMENTALS**

- HPC Overview
- Linux Command Line Basics
- Research Software
- Job Submission

4

4

## REMOTE EVENT GUIDELINES

- Please ensure your microphone is muted during the presentation.
- Chat is the preferred method for asking any questions you may have.
- Please feel free to post to the chat at any time.
- Use the “raise hand” feature if you would like to ask a question or make a comment with audio.
- Breakout rooms can be used during labs. If you get stuck on a lab, please use the “raise hand” feature and an admin will assist you.
- Don't be shy—questions and comments are welcome and encouraged.

[hpc.auburn.edu](http://hpc.auburn.edu) | [hpcadmin@auburn.edu](mailto:hpcadmin@auburn.edu)

5

5

## SLIDES & LAB EXERCISES



<https://aub.ie/hpctrain>

hpc.auburn.edu | hpcadmin@auburn.edu

6

6

## FLIGHT CHECK

### CLUSTER ACCOUNT & CONNECTION

WINDOWS	
PuTTY	putty.org
SecureCRT	auburn.edu/download
WSL	Windows Subsystem for Linux
WinSCP	(File Transfer)
Emulation	VMWare Workstation VirtualBox

MAC OS	
Terminal	Applications -> Utilities -> Terminal
Third Party	Prompt (iOS)
FileZilla	(File Transfer)
Transmit	(File Transfer)
Emulation	VMWare Fusion VirtualBox

LINUX	
Terminal	gnome kde
Runlevel	CLI (3)
FileZilla	(File Transfer)

AU HPC SSH connections require **DUO Two-Factor Authentication**

**VPN is required** for off-campus and wireless connections, and some wired campus subnets.

hpc.auburn.edu | hpcadmin@auburn.edu

8

8

## HPC TERMINOLOGY

### # MANAGEMENT COMPONENTS



- **Cluster:** Integrated collection of computing resources
  - Node: A single encapsulated computer
  - Network: Communication between nodes
  - Storage: Various places to store your files
  - Scheduler: Resource allocation

[hpc.auburn.edu](mailto:hpc.auburn.edu) | [hpcadmin@auburn.edu](mailto:hpcadmin@auburn.edu)

26

26

## HPC TERMINOLOGY

### # WORKLOAD COMPONENTS



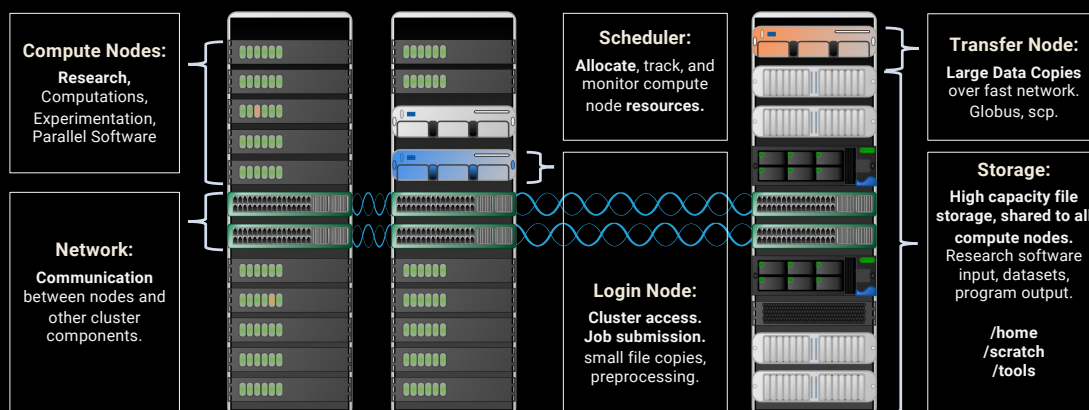
- **Processor:** The main computational chip (CPU)
- **Core:** Processing subcomponents within a processor
- **Process:** Running program
- **Thread:** Parallelization (within the confines of a node, shared memory)
- **Message Passing (e.g. MPI):** Highly Scalable parallelization (across multiple nodes, distributed memory)

[hpc.auburn.edu](mailto:hpc.auburn.edu) | [hpcadmin@auburn.edu](mailto:hpcadmin@auburn.edu)

27

27

## HPC CLUSTER COMPONENTS



hpc.auburn.edu | hpcadmin@auburn.edu

28

28

## LINUX COMMAND LINE BASICS

hpc.auburn.edu | hpcadmin@auburn.edu

29

29

## WHY LINUX?

- Ubiquity
  - The world's most capable supercomputers all use Linux
- Research Software
  - Reproducibility
  - Non-proprietary tools and libraries
  - Reduce costs
- Portability & Collaboration
  - Facilitates adaptivity and customization
- Important Tool for Computational Research

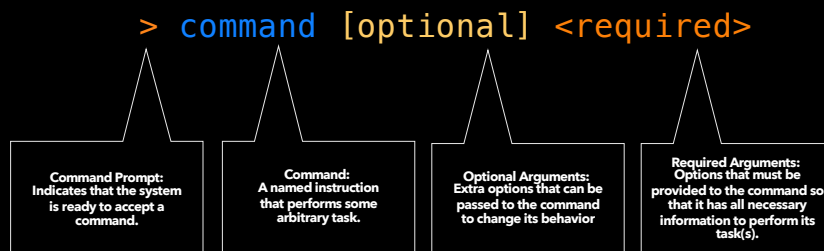
hpc.auburn.edu | hpcadmin@auburn.edu

32

32

## LINUX COMMAND LINE BASICS

# GENERALIZED COMMAND LINE SYNTAX



hpc.auburn.edu | hpcadmin@auburn.edu

33

33

# ***BREAK***

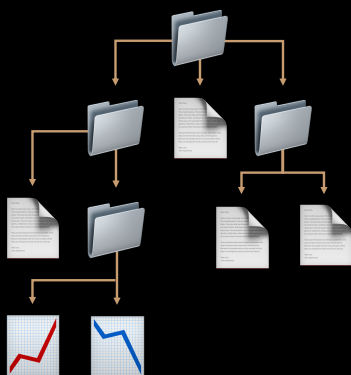
hpc.auburn.edu | hpcadmin@auburn.edu

34

34

## LINUX COMMAND LINE BASICS

### # FILE SYSTEM



- Provide a Method to Store Data
- Logically Organize Data into Files and Folders (Directories)
- Hierarchical Structure
- Critical for Practical Use of Computer Systems

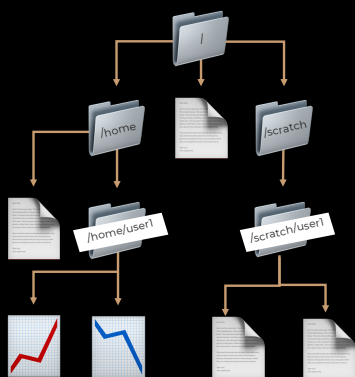
hpc.auburn.edu | hpcadmin@auburn.edu

35

35

# LINUX COMMAND LINE BASICS

## # FILE SYSTEM



- We need a method for moving around the file system without a mouse
- Subdirectories are delineated by a Forward Slash (/)
  - directory/subdir1/subdir2/...
- The topmost directory is the Root Directory (/)
  - Similar to C:\ on Windows

hpc.auburn.edu | hpcadmin@auburn.edu

36

36

# LINUX COMMAND LINE BASICS

## # FILE SYSTEM: GUI VS. COMMAND LINE

hpc.auburn.edu | hpcadmin@auburn.edu

37

37

## LINUX COMMAND LINE BASICS

hpcuser@easley01:Documents > **ls -al**

List all files and directories in the current location, with lots of details

```
total 5
drwxrwxr-x 5 hpcuser hpcuser 512 Jun 16 10:53 .
drwxrwxr-x 8 hpcuser hpcuser 512 Jun 16 10:53 ..
-rw-rw-r-- 1 hpcuser hpcuser  0 Jun 16 10:53 File.pdf
-rw-rw-r-- 1 hpcuser hpcuser  0 Jun 16 10:53 File.png
-rw-rw-r-- 1 hpcuser hpcuser  0 Jun 16 10:53 File.txt
drwxrwxr-x 2 hpcuser hpcuser 512 Jun 16 10:53 Folder1
drwxrwxr-x 2 hpcuser hpcuser 512 Jun 16 10:53 Folder2
drwxrwxr-x 2 hpcuser hpcuser 512 Jun 16 10:53 Folder3
```

Special directory: current directory

Special directory: previous directory

Column 1:  
File Permissions

Column 3:  
File Owner

Column 5:  
File Size

Column 7:  
File Name

Column 2:  
References

Column 4:  
File Group

Column 6:  
File Creation  
Date

hpc.auburn.edu | hpcadmin@auburn.edu

38

38

## LINUX COMMAND LINE BASICS

# FILE SYSTEM: NAVIGATING, LISTING, AND VIEWING

COMMAND	DESCRIPTION
<b>pwd</b>	"Present Working Directory" (e.g. where am I?)
<b>ls</b>	Show a list of files and directories
<b>cd</b>	Change present working directory to specified path
<b>cat</b>	Display file contents
<b>less</b>	Display file contents page-by-page
<b>more</b>	Display file contents page-by-page

hpc.auburn.edu | hpcadmin@auburn.edu

39

39

# LINUX COMMAND LINE BASICS

## # FILE SYSTEM: NAVIGATION DEMO

```

hpcuser@easley01:~ > pwd
/home/hpcuser
hpcuser@easley01:~ > cd Documents
hpcuser@easley01:Documents > pwd
/home/hpcuser/Documents
hpcuser@easley01:Documents > ls -l
File.pdf
File.png
File.txt
Folder1
Folder2
Folder3

```

Where am I? (Present Working Directory)

The command output tells us we are in /home/hpcuser, which is a "home directory"

The "cd" command will change our present working directory

Okay, now where am I?

We're now currently in a subdirectory named "Documents"

Let's use the "ls" command to show the files that are in this "Documents" folder

Here are the files and subdirectories inside the "Documents" folder

hpc.auburn.edu | hpcadmin@auburn.edu

40

40

# LINUX COMMAND LINE BASICS

## # FILE SYSTEM: CREATING, MODIFYING, AND DELETING FILES

COMMAND	DESCRIPTION
<b>touch &lt;filename&gt;</b>	Create an empty file
<b>mkdir &lt;directory&gt;</b>	Create an empty directory
<b>cp &lt;source&gt; &lt;target&gt;</b>	Copy file(s)
<b>mv &lt;source&gt; &lt;target&gt;</b>	Move file(s)
<b>rm &lt;filename&gt;</b>	Delete file(s) or directories
<b>rmdir &lt;directory&gt;</b>	Delete an empty directory
<b>&gt;</b>	Output redirect to file (destructive)
<b>&gt;&gt;</b>	Output redirect to file (append)

hpc.auburn.edu | hpcadmin@auburn.edu

41

41

# LINUX COMMAND LINE BASICS

## # FILE SYSTEM: CREATING FILES AND DIRECTORIES

```

> mkdir newdir
> ls -l
...
newdir
> cd newdir
> pwd
/home/hpcuser/Documents/newdir
> touch new.txt
> ls -l
-rw-rw-r-- 1 hpcuser hpcuser 0 Aug  5 16:48 new.txt

```

Make a new directory named "newdir"

List the contents of the current directory, with single-column formatting

Confirm the "newdir" directory was created

Change our present working directory to the newly created directory

Further confirm that "newdir" is the current directory

Create a new, empty file named "new.txt"

List the current directory contents, and we should now see our new file

hpc.auburn.edu | hpcadmin@auburn.edu

42

42

# LINUX COMMAND LINE BASICS

## # FILE SYSTEM: CREATING FILES AND DIRECTORIES

```

> echo "Hi!" >> new.txt
> cat new.txt
Hi!
> cp new.txt old.txt
> mv new.txt newer.txt
> ls
newer.txt  old.txt

```

Use output redirection operator "&gt;&gt;" to append contents to an existing file

Print the entire contents of the file "new.txt" to the screen

Make a copy of the file "new.txt" called "old.txt"

Change the filename of "new.txt" to "newer.txt"

List the directory to see what the previous commands have done

hpc.auburn.edu | hpcadmin@auburn.edu

43

43

# LINUX COMMAND LINE BASICS

## # FILE SYSTEM: SPECIAL FILES AND DIRECTORIES

FILENAME \ COMMAND	DESCRIPTION
.	Special directory: current directory
..	Special directory: previous directory
.<filename>	Special file: hidden file
~/.bashrc	Special file: shell configuration
~/.bash_profile	Special file: shell configuration
ls -al	List current directory and show all special and hidden files

hpc.auburn.edu | hpcadmin@auburn.edu

44

44

# LINUX COMMAND LINE BASICS

## # FILE SYSTEM: SPECIAL DIRECTORIES

```

> cd .
> pwd
/home/hpcuser/Documents/newdir
> cd ..
> pwd
/home/hpcuser/Documents/

```

← Change the current directory to special directory "."

← Nothing happened, Special directory "." (dot) refers to the current directory.

← Change our present working directory to special directory ..

← Present working directory is now one level back in the hierarchy. Special directory .. (dot dot) refers to the previous directory.

hpc.auburn.edu | hpcadmin@auburn.edu

45

45

## LINUX COMMAND LINE BASICS

# FILE SYSTEM: MOVING DATA WITH SECURE COPY (SCP)

- WinSCP: <https://auburn.edu/download>
- Filezilla: <http://filezilla-project.org>
- SCP Command Line Tool

### SCP GENERAL SYNTAX

```
> scp <source> <userid>@easley.auburn.edu:<destination>
```

### SCP EXAMPLE

```
> scp -r /my/local/data <userid>@easley.auburn.edu:~/project_data
```

46

## LINUX COMMAND LINE BASICS

# FILE SYSTEM: MOVING DATA WITH GLOBUS



- <https://globus.org>

47

## SHELLS & SHELL SCRIPTS

- What is a **Shell**?
  - Many different flavors: bash, csh, zsh ...
  - Interprets the commands you type
- What is a **Shell Script**?
  - Can be conceptualized as small computer programs
  - Automate processes, set and display variables, perform computations ...

48

## SHELLS & SHELL SCRIPTS

### # EDITORS

- nano
- vi \ vim
- emacs
- Edit offline and upload
  - notepad, notepad++, then scp
- Edit offline directly
  - VSCode, PyCharm

49

## SHELL SCRIPTS

```
#!/bin/bash
# this is an example shell script ...
# print a string
echo "Hello World!"

# declare STRING variable and assign it a value ...
STRING="Hello World"

# print a variable value to the screen
echo $STRING

# you can enter commands in a shell script like you enter them on the
# command line
ls -al
cd ~
```

Shell interpreter (shbang, hashbang) line, tells Linux what shell to use for the script

After the shbang line, any other line starting with # is considered a **comment**.

A shell script can run any command you would normally run on the command line. Here, we use echo to print text to the screen.

Shell scripts can also use named variables. Allows you to reference values multiple times within the script.

Assigning or changing the variable value will cascade to any of the references.

Shell scripts are essential for:  
**Automating** repetitive tasks  
**Submitting jobs** with dynamic configurations or parameters  
 Performing **pre and post processing operations**

50

## LAB EXERCISE 1

# BASIC LINUX COMMANDS

51

# RESEARCH SOFTWARE

[hpc.auburn.edu](http://hpc.auburn.edu) | [hpcadmin@auburn.edu](mailto:hpcadmin@auburn.edu)

52

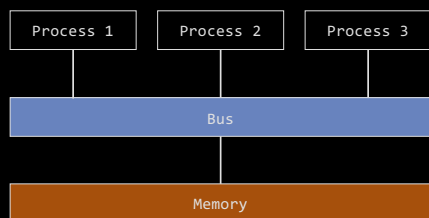
52

## Research Software

- Dependencies are highly dynamic
- Requires manipulation of your shell environment
- Typically requires system-specific compilation
- Can be installed and used in home directory
  - Or installed by HPC Admins: <https://aub.ie/hpcsw>
- Many popular software, compilers, and libraries already available...

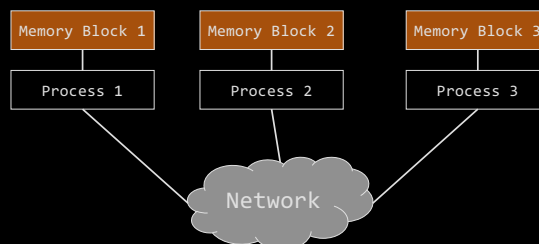
53

## PARALLEL PROGRAMMING MODELS



### Shared Memory

- All processes have access to the same memory address space.
- Communication with memory via internal bus.
- Parallelism (typically) through **threading**.



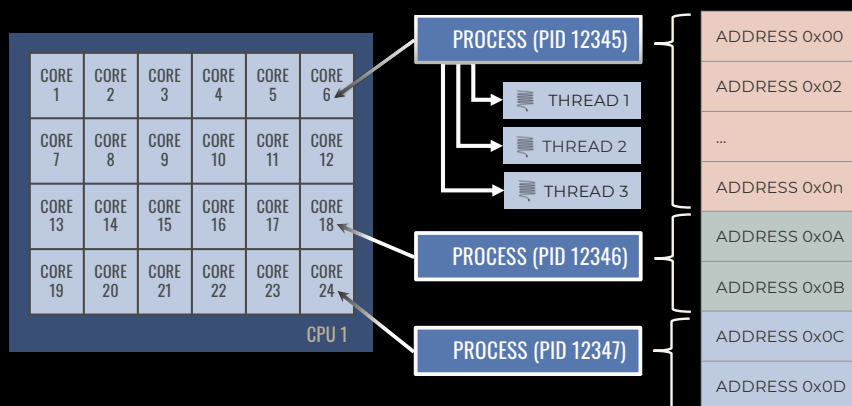
### Distributed Memory

- Each process has its own dedicated memory.
- Communication is over an arbitrary network.
- Parallelism through **message passing**.

54

## HPC TERMINOLOGY

### # SOFTWARE CONTEXT



- A process and its threads all share the same memory space.
- Linux refers to running software programs as **processes**.
- Each process is given a unique **process identifier**, known as a PID.
- The execution of a process is assigned to a specific **CPU Core**.
- A process can spawn **multiple threads** to improve concurrency and parallelization.

hpc.auburn.edu | hpcadmin@auburn.edu

55

55

## ENVIRONMENT

### Traditional Method

```
$ echo $PATH
$ echo $LD_LIBRARY_PATH
$ export PATH=$PATH:/home/username/custom/bin
$ export LD_LIBRARY_PATH=/home/username/custom/lib:$LD_LIBRARY_PATH
```

### Environment Module Method

```
$ module load <software_name>
```

### Basic Commands

```
$ module avail [search_string]
$ module show <software_name>
$ module purge (must reload slurm)
$ module swap <old_software> <new_software>
$ module list
```

56

## LAB EXERCISE 2

# RESEARCH SOFTWARE

64

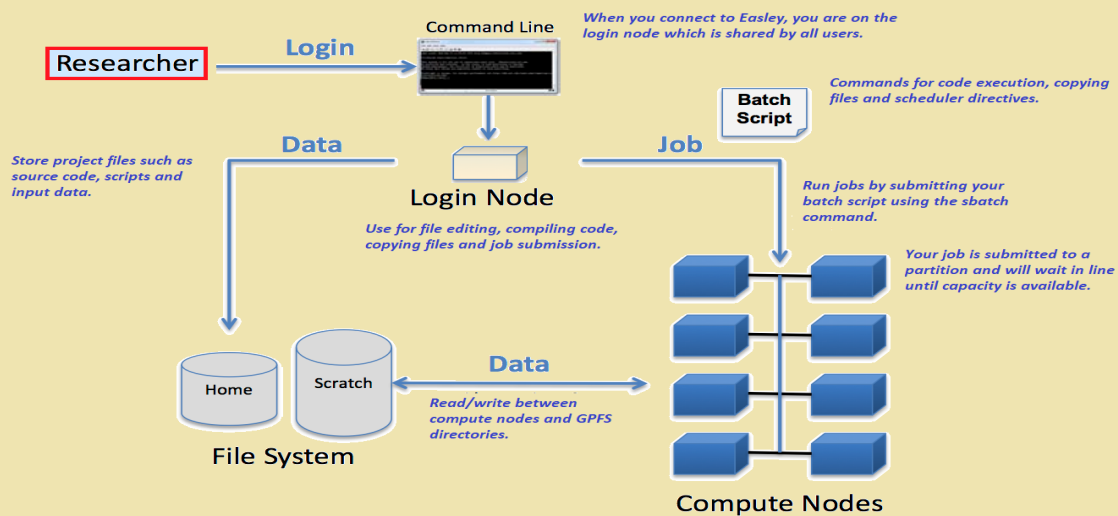
# JOB SUBMISSION

hpc.auburn.edu | hpcadmin@auburn.edu

65

65

## Job Submission Overview



66

## WHAT IS A SCHEDULER?

- An HPC cluster needs a way for users to access its computational capacity in a fair and efficient manner. It does this using a scheduler.
- The scheduler takes user requests in the form of jobs and allocates resources to these jobs based on availability and cluster policy.

67

## WHY DO WE NEED IT?

- Potentially 100s of researchers and 1000s of jobs.
- How to fairly allocate shared resources?
- How to monitor the resources to determine availability?
- How to arbitrate contention for resources?

68

## WHY SLURM?

- Slurm is an open source, fault-tolerant, and highly scalable cluster management and job scheduling system.
- The industry standard used by many of the top universities and research institutions worldwide.

69

## WHAT IS A JOB?

- A means of interacting with the scheduler in order to get to the computational capacity of the compute nodes.
- A job consists of
  1. Resource requirements
  2. Length of time resources are needed
  3. What commands to run

71

## PARTITIONS

- Define different types of computers.
  - general, amd, bigmem, and gpu
  - The general partition is the default.
- Implement priority access to capacity.

72

## Slurm: Partitions

- **Community partitions**
  - Contain ALL capacity of a specific type.
  - For example, the general partition consists of ALL standard nodes.
- **Investor partitions**
  - Contain ALL capacity of a specific specialty type.
  - For example, the investor\_amd partition consists of ALL amd nodes.
- **Dedicated partitions**
  - Contain ALL capacity of a specific type for a specific investor.
  - For example, the vza0113\_std partition consists of ALL standard node capacity purchased by vza0113.

73

## How to submit a job?

### Submitting Jobs

```
$ sbatch -p general -N1 --ntasks=2 -t2:00 ./job.sh
```

```
$ cat job.sh
```

```
#!/bin/bash
#SBATCH --job-name=myJob          # job name
#SBATCH -n10                     # number of tasks across all nodes
#SBATCH --partition=hpcadmin_lab  # name of partition to submit job
#SBATCH --time=01:00:00          # Run time (D-HH:MM:SS)
#SBATCH --output=job-%j.out       # Output file. %j is replaced with job ID
#SBATCH --error=job-%j.err       # Error file. %j is replaced with job ID
#SBATCH --mail-type=ALL          # will send email for begin,end,fail
#SBATCH --mail-user=terrykd@auburn.edu
...
```

75

## JOB SCHEDULING

### Viewing the Partitions

```
$ sinfo
$ sinfo -t idle
$ /tools/scripts/my_capacity
```

### Monitoring Jobs

```
$ squeue
$ scontrol show job <jobid>
$ squeue -u <userid>
```

### Estimated Start Time

```
$ sbatch --test-only <job.sh>
```

76

# **LAB EXERCISE 3**

## **JOB SUBMISSION**

77

# **BEST PRACTICES**

82



## DELETING FILES

- Use remove ( rm ) command with caution
- There is no recycle bin in Linux
- When a file is removed, it's gone
- Linux will not ask 'Are you sure?'
- It assumes that you know what you're doing

83

## RUNNING SOFTWARE

- Do not run jobs on the login node except as a test.
- This means only short jobs using small amounts of memory to ensure that your code will run.
- Processes that violate this will be killed.

84

## STORAGE QUOTAS

### **Pay attention to your disk usage.**

- Each user has a 2TB home directory.
- Your jobs will not run as expected when this limit is reached.
- Use the `checkquota` command to see your usage.

85

## SCRATCH STORAGE

- **Do not use the scratch directory for long-term storage.**
- The scratch directory is for temporary, work-in-progress files only.
- It is not backed up.
- You run the risk of losing all your work if scratch is used for long-term storage of data files and output files.

86

## WHERE TO GO FOR HELP

- Email [hpcadmin@auburn.edu](mailto:hpcadmin@auburn.edu)
- Include job #, job sub command, errors and any other pertinent information
- Request an appointment
- Email to schedule a time to meet

87

## REMINDERS

88

## CITATIONS & ACKNOWLEDGEMENTS

*Please help show the importance of HPC resources in research at Auburn University by citing such in any publication or presentation that are made possible using the CASIC, Hopper, or Easley Clusters.*

<https://aub.ie/citehpc>

89



## **AUBURN HPC v. ALABAMA SUPERCOMPUTER**

- *Two distinct entities.*
- *Please give credit to the appropriate institution.*

90

## Spring Maintenance: March 27 - 31

- One of two maintenance periods.
- Needed to maintain cluster.
- Includes upgrades and other maintenance tasks that are difficult to do with jobs running.

Until the maintenance period is over, you must specify a wall-time in your job submission that ensures that your job will end before the maintenance period begins or else your job will not run.

91

## Next First Friday: March 3 @ 9:30-11:30am

- **Special Topics Class**
  - Data Management ( AU Library )
  - Data Transfer
  - Research software
  - MPI
  - GPU
  - Slurm

Please let us know a topic of interest to you!

92

## COURSE EVALUATION

<http://aub.ie/hpceval>

93



## HELP ...

- Documentation: [aub.ie/hpcdocs](http://aub.ie/hpcdocs)
- Email: [hpcadmin@auburn.edu](mailto:hpcadmin@auburn.edu)
  - General questions & feedback welcome
  - Support requests, please include:
    - job id · script\data paths · errors · description
  - direct support by appointment

94