HIGH PERFORMANCE RESEARCH COMPUTING

HPRC Primer

Introduction to Linux Using Grace

September 05, 2025

TAMU users: If you are outside campus, activate VPN by connect.tamu.edu

High Performance
Research Computing
DIVISION OF RESEARCH

Computing Resources

The HPRC group currently administers five HPC clusters:

- ACES
- FASTER
- Grace
- Launch
- ViDaL

You will need one of two options to use them:

Credentials	Clusters	Who
HPRC Account	FASTER, Grace	Texas A&M faculty/students/staff
ACCESS ID	Launch, and ACES	Researcher or educator at a U.S. academic, non-profit research, or educational institution

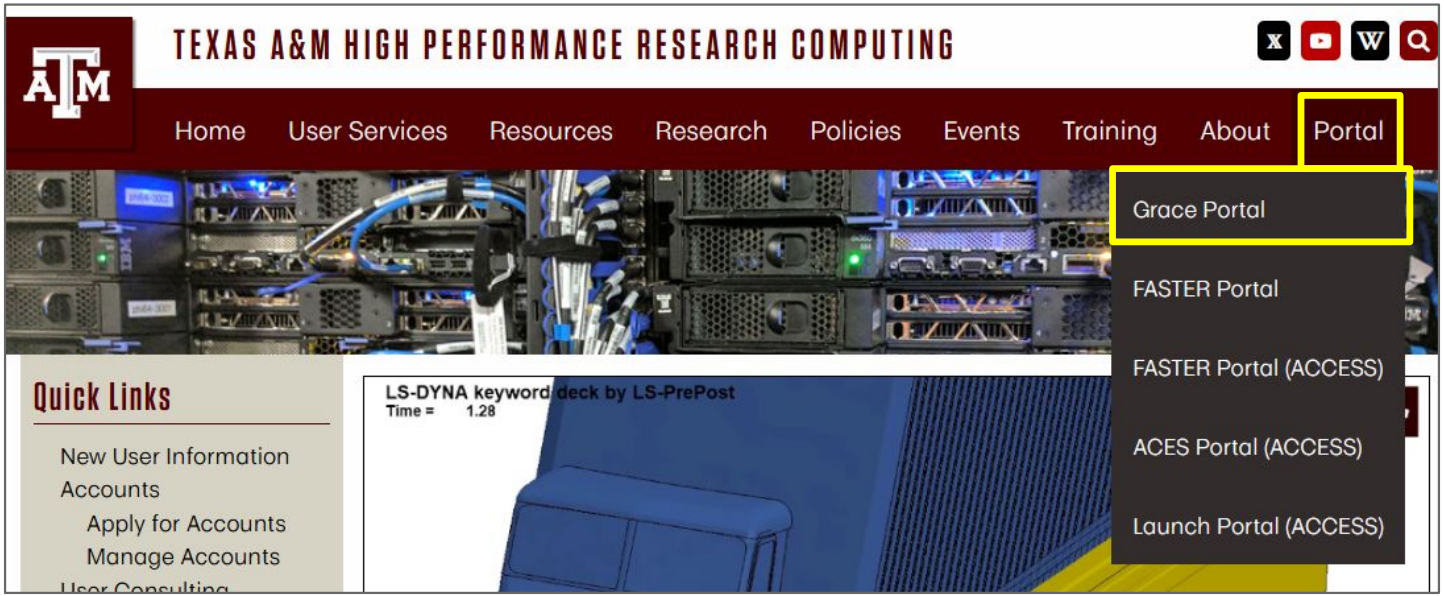
Link to our Knowledge Base: https://hprc.tamu.edu/kb/

Your Login Password

- Do **NOT** share your password
- Do **NOT** share your account
- Texas law and TAMU regulations prohibit the sharing and/or illegal use of computer passwords and accounts

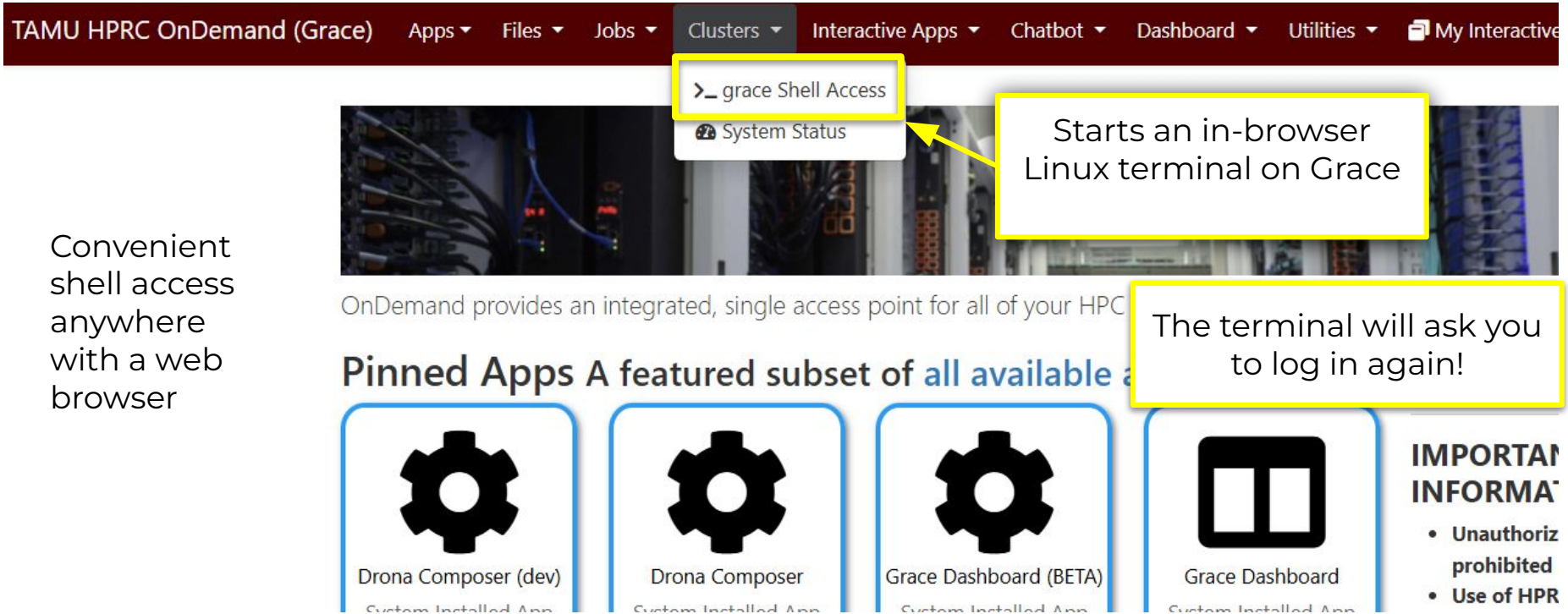
HPRC Portal

- HPRC webpage: hprc.tamu.edu
  - Grace portal: portal-grace.hprc.tamu.edu



Linux Using the Portal - Shell Access

Convenient shell access anywhere with a web browser



Linux Using SSH

If you're using a terminal application on your own computer instead of the Portal, you can connect to HPRC clusters using the ssh command:

```
ssh -X NetID@faster.hprc.tamu.edu
```

Mac users may need to use ssh -Y to enable X11 so you can view images and use GUI software:

```
ssh -Y NetID@faster.hprc.tamu.edu
```

You may see something like this the first time you connect:

```
Host key not found from the list of known hosts.
    Are you sure you want to continue connecting (yes/no)?
```

Type **yes**, hit enter and you will then see the following:

```
Host faster.hprc.tamu.edu' added to the list of known hosts.
    NetID@faster.tamu.edu's password:
```

Where Am I?

pwd command (print working directory)

Linux commands in green for you to type

```
pwd
```

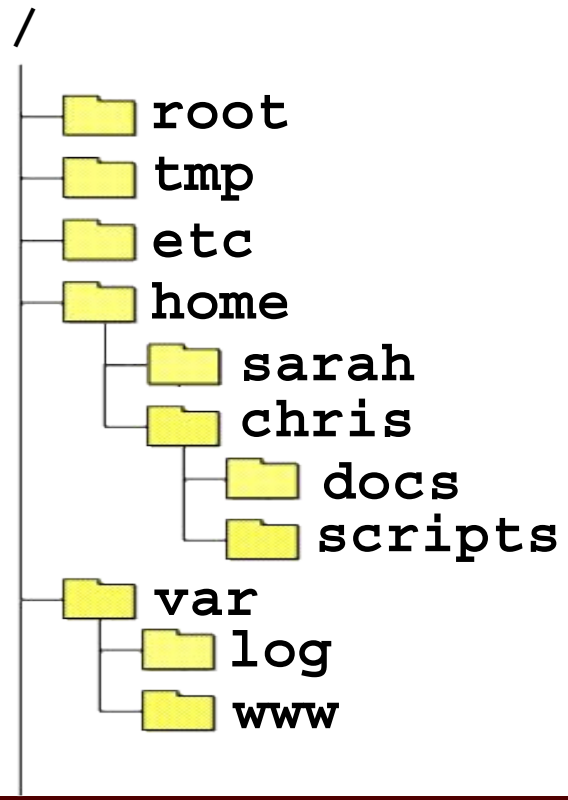
command output in blue

```
/home/username
```

list contents of your working directory

```
ls
```

Navigating the Linux Directory Structure




```
/
/root
/tmp
/etc
/home
/home/sarah
/home/chris
/home/chris/docs
/home/chris/scripts
/var
/var/log
/var/www
```

Common Directory Commands

```
mkdir my_dir
```

mkdir to make a new directory

```
cd my_dir
```

cd to change to another directory

```
cd ..
```

cd back out of the current directory

```
rmdir my_dir
```

rmdir to remove an empty directory

Changing Directories: cd

Return to your home directory:

```
cd
cd ~
cd ~/
cd $HOME
```

Switch to the parent directory of the current directory:

```
cd ..
```

Return to previous directory:

```
cd -
```

```
cd $HOME
mkdir temp
mkdir temp/dir1
cd temp
pwd
cd dir1
pwd
cd ../..
pwd
cd -
pwd
cd ..
pwd
cd ~
pwd
```

Absolute vs. Relative Path

```
/
/root
/tmp
/etc
/home
/home/sarah
/home/chris/project
/home/chris/docs/README
/var
/var/log
/var/www
```

If you are in the `project` directory

```
pwd
```

```
/home/chris/project
```

The relative path to the README file is `../docs/README`

```
ls ../docs/README
```

The absolute path to the README file `/home/chris/docs/README`

```
ls /home/chris/docs/README
```

Common Commands

Let's start working with content in our directories.
Start with these basic commands:

cat	Writes file content on the standard output*
echo	Display a text string on the standard output
touch	Creates a new empty file
nano	Creates a new file or edit an existing file (text editor)
rm	Remove a file

Let's print some output and make a new file:

```
echo "Hello World"
touch new.txt
nano new.txt
cat new.txt
```

*Usually "standard output" just means your screen, but it can be moved

History of Your Commands

Your commands are saved to a file in your home directory (.bash_history)
You can use the up/down arrows to scroll through previous commands
Type **`history`** to see your previously entered commands

`history`	`History of your commands`
**`history	tail`**	`See the last 10 commands`

Search your command history using | and grep

```
history | grep echo
```

Linux Commands Have Options

Leave a space between the command and the options

Spell out a full option with a double-dash:

```
ls --all
```

`--all` show all files, including hidden files which begin with '.'

Single dash lets you abbreviate:

```
ls -a -l
```

`-a` (shorter version of `--all`)
`-l` show file details

You can also combine (short) options behind one single dash:

```
ls -al
```

`-a` (same function as above)
`-l` (same function as above)

Remember directory shortcuts:

`.` current working directory
`..` parent directory

Search for Linux Commands Options

Search the manual page for the Linux command ls

```
man ls
```

`f`	move down (forward) one page
`b`	move up (back) one page

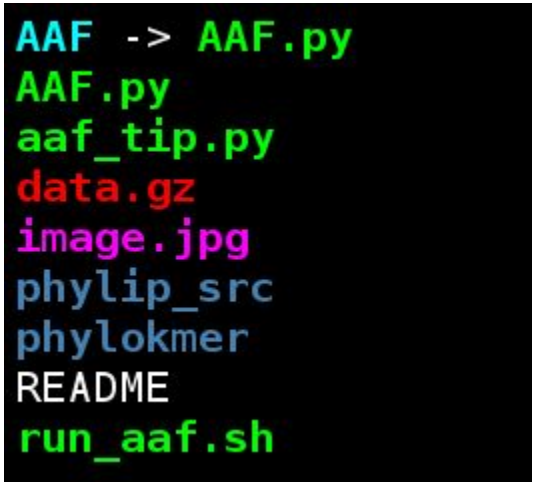
(Sometimes mouse scroll wheel and arrow keys work, too)

`/all`	search the man page for the text 'all'
`n`	search forward for next found match
`N`	search backwards next found match
`g`	go to first line
`G`	go to last line
`q`	quit

Linux Terminal Attributes

Depending on your terminal, you've probably been seeing different colors as you navigate.

File and directory names are colored based on their attributes such as permissions and extension (file type).



TURQUOISE	Symbolic link
GREEN	Executable file
RED	Compressed files
PURPLE	Image files
BLUE	Directories
WHITE	Text files

Note: These colors are not Linux-universal and can depend on the different terminal emulator or shell.

Changing Attributes: chmod

```
chmod [options] [permission mode] [target_file]
```

```
0 = No permission
1 = Execute permission
2 = Write permission
3 = Write and execute permissions
4 = Read permission
5 = Read and execute permissions
6 = Read and write permissions
7 = Read, write, and execute permissions
```

```
u = user
g = group
o = other

r = read
w = write
x = execute
-x = remove executable permissions
+x = enable executable permissions
```

Note the permissions display format is - `uuugggooo`

Changing Attributes: chmod

Set limits on who can modify files and directories with '`chmod`'

Follow the instructions at right to make some example files and check their details.

```
mkdir data
cd data
touch file1.txt
touch file2.txt
ls -l
```

You should see a bunch of dashes and letters to the left. Those are the permissions.

1. To change the user's permissions of file1.txt to read, write, execute:
   (will be `-rwxrw-r--`)

```
chmod u+rwx file1.txt
```

2. To change the permissions of file2.txt to read and execute for all and write for the user:
   (will be `-rwxr-xr-x`)

```
chmod 755 file2.txt
```

(see next slide for what this number means)

3. To remove the execute permissions of file2.txt for all "other" users:
   (will be `-rwxr-xr--`)

```
chmod o-x file2.txt
```

Shell Script Exercise

A *script* will let you perform multiple commands at once.

We've created an example script, which you can copy and run yourself.

Navigate to your home directory

```
cd $HOME
```

Copy the script to your home directory

```
cp /scratch/training/fall_2025_primers/my_script.sh .
```

Shell Script Exercise

View (or edit) the shell script

```
nano my_script.sh
```

```
#!/bin/bash
# HPRC shell script exercise

my_var="People"

echo "Howdy $my_var" > output.txt

mkdir script_output

mv output.txt script_output

cd script_output

cat output.txt
```

make your shell script executable

```
chmod 755 my_script.sh
```

run your shell script

```
./my_script.sh
```

Shell Script Explanation

The “shebang”; all bash scripts must have this at the very top so the computer knows how to run it.

Pound signs start comments. They’re for you to leave notes; the computer doesn’t do anything with them. (The shebang is the exception!)

```
#!/bin/bash
# HPRC shell script exercise

my_var="People"

echo "Howdy $my_var" > output.txt

mkdir script_output

mv output.txt script_output

cd script_output

cat output.txt
```

Shell Script Explanation

A “variable.” Call later with ‘$’ to reuse stored data.

The ‘>’ redirects the output to the filename you provide.

(Commands we’ve seen previously)

```
#!/bin/bash
# HPRC shell script exercise

my_var="People"

echo "Howdy $my_var" > output.txt

mkdir script_output

mv output.txt script_output

cd script_output

cat output.txt
```

Shell Script Exercise

View (or edit) the shell script

```
nano my_script.sh
```

make your shell script executable

```
chmod 755 my_script.sh
```

run your shell script

```
./my_script.sh
```

```
#!/bin/bash
# HPRC shell script exercise

my_var="People"

echo "Howdy $my_var" > output.txt

mkdir script_output

mv output.txt script_output

cd script_output

cat output.txt
```

Exit your terminal

```
exit
```

exit the terminal session

To fully log out of the Grace portal, you need to exit the browser.

High Performance
Research Computing
DIVISION OF RESEARCH

Give us feedback on the class with this survey:
https://u.tamu.edu/hprc_shortcourse_survey

Thank you

Questions?

HPRC Survey

Need Help?

First check the FAQ https://hprc.tamu.edu/kb/FAQ/Accounts/

- Grace User Guide https://hprc.tamu.edu/kb/User-Guides/Grace
- Email your questions to help@hprc.tamu.edu

Help us help you -- provide the following info:

- Which cluster you're using
- Your username
- Job id(s) if any
- Location of your jobfile, input/output files
- Application used, if any
- Module(s) loaded, if any
- Error messages
- Steps you have taken, so we can reproduce the problem

Continued Learning

Intro to HPRC Video Tutorial Series

HPRC's Knowledge Base