UT Southwestern Medical Center
Lyda Hill Department of Bioinformatics

BioHPC

Introduction to Linux and the Command Line Interface (CLI)

[web] portal.biohpc.swmed.edu
[email] biohpc-help@utsouthwestern.edu

August 2021

There are numerous study resources

- https://www.linuxcommand.org/
- https://linoxide.com/
- https://itsfoss.com/
- https://linuxjourney.com/
- https://wizardzines.com/comics/
- https://linuxhandbook.com/
- https://www.freecodecamp.org/news/the-linux-commands-handbook/

Please follow along…

Different methods to access a Linux terminal:

1. The Nucleus Web terminal on the BioHPC portal (VPN required):

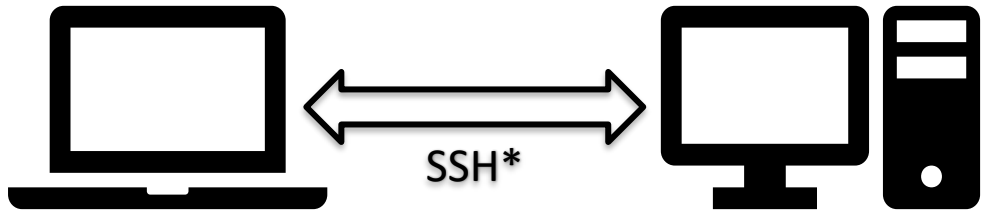
   https://portal.biohpc.swmed.edu/terminal/ssh/

2. PuTTY, WSL, MobaXterm or any other SSH* client from your PC

3. Install Linux as the guest OS in a VM (VirtualBox)

4. Terminal from your MacBook:

```
ssh <username>@nucleus.biohpc.swmed.edu
```



What is Linux, anyway?

- Putting it simply, Linux is just a **kernel** (the core of the OS).
- A Linux **distribution** (or **distro**) includes, in addition to the kernel, drivers (modules), a shell, package managers, and a desktop environment (Gnome, KDE).
- There are numerous flavors of Linux around for you to choose.
- What about BioHPC? What is the distro do BioHPC machines run on?
  - Red Hat Enterprise Linux (RHEL) 7.6
  - Gnome 3 desktop environment
  - Bourne again shell (Bash)
  - Modular environment
  - SLURM scheduler

The Bourne Again Shell (Bash)

A shell provides you with an CLI to the Linux system. The shell reads your commands and tells Linux to execute them. When your command finishes executing, the shell displays that command's output.

The prompt, $, which is called the command prompt, is issued by the shell. While the prompt is displayed, you can type a command.

The shell reads your command after you press Enter. It determines the command you want executed by looking at the first word of your command. A word is an unbroken set of characters. Spaces and tabs separate words.

The shell is, after all, a real programming language, complete with variables, control structures, and so forth. No matter how complicated a script gets, it is still just a list of commands executed sequentially.

There are other types of shell, but we shall focus exclusively on Bash, which is the default shell in BioHPC.

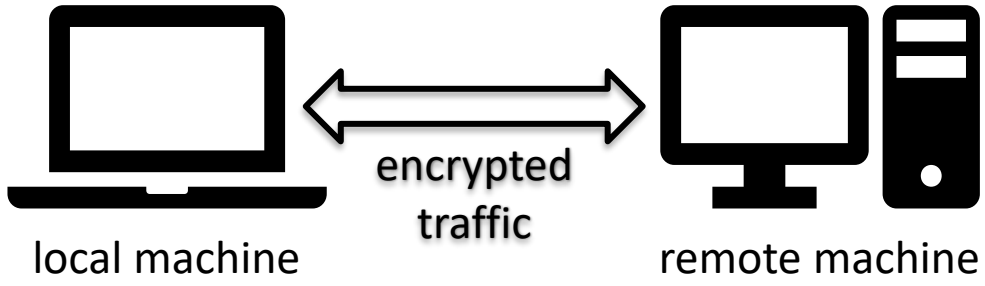
If you get into shell scripting, give shellcheck a try.

The secure shell (SSH)

Most of your interactions with the Nucleus cluster will likely be through SSH.

Most modern GNU/Linux distributions have an OpenSSH client installed by default. Mac OS X also has SSH. PuTTY is recommended for MS Windows. MobaXterm is very popular and feature-rich, too.

```
$ ssh s191529@nucleus.biohpc.swmed.edu
```



The Command Line Interface

```
[s191529@Nucleus006 ~]$ echo -n Hello!
```

user

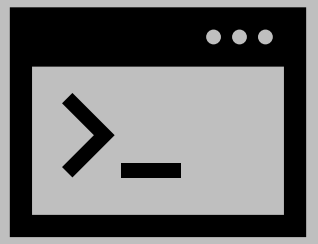
host

working directory

command

options

argument



This is a hands-on task!

```
[s191529@Nucleus006 ~]$ sudo su -
[s191529@Nucleus006 ~]#
```

Logging into Nucleus

```
$ ssh s191529@nucleus.biohpc.swmed.edu
s191529@nucleus.biohpc.swmed.edu's password:

Current BioHPC Storage Quotas:

  FILE    |         SPACE USAGE          |         NUMBER OF FILES
 SYSTEM   |     USED     SOFT     HARD   |       USED       SOFT       HARD
----------|------------------------------|-----------------------------------
User quotas for s191529
----------|------------------------------|-----------------------------------
home2     |   18100M   51200M   71680M   |       122k          0          0
work      |   1.143T       5T       7T   |      35912          0          0
----------|------------------------------|-----------------------------------
Group quotas for biohpc_admin
----------|------------------------------|-----------------------------------
project   |   23.51T       0k       0k   |   12888410          0          0
archive   |   20.12T      30T      40T   |   12459884          0          0

*/home2 are backed up twice per week (Mon and Wed)
*/work are backed up weekly
*/project and /archive have no backup, PI can request backup

[s191529@Nucleus005 ~]$
```

Navigating the file systems

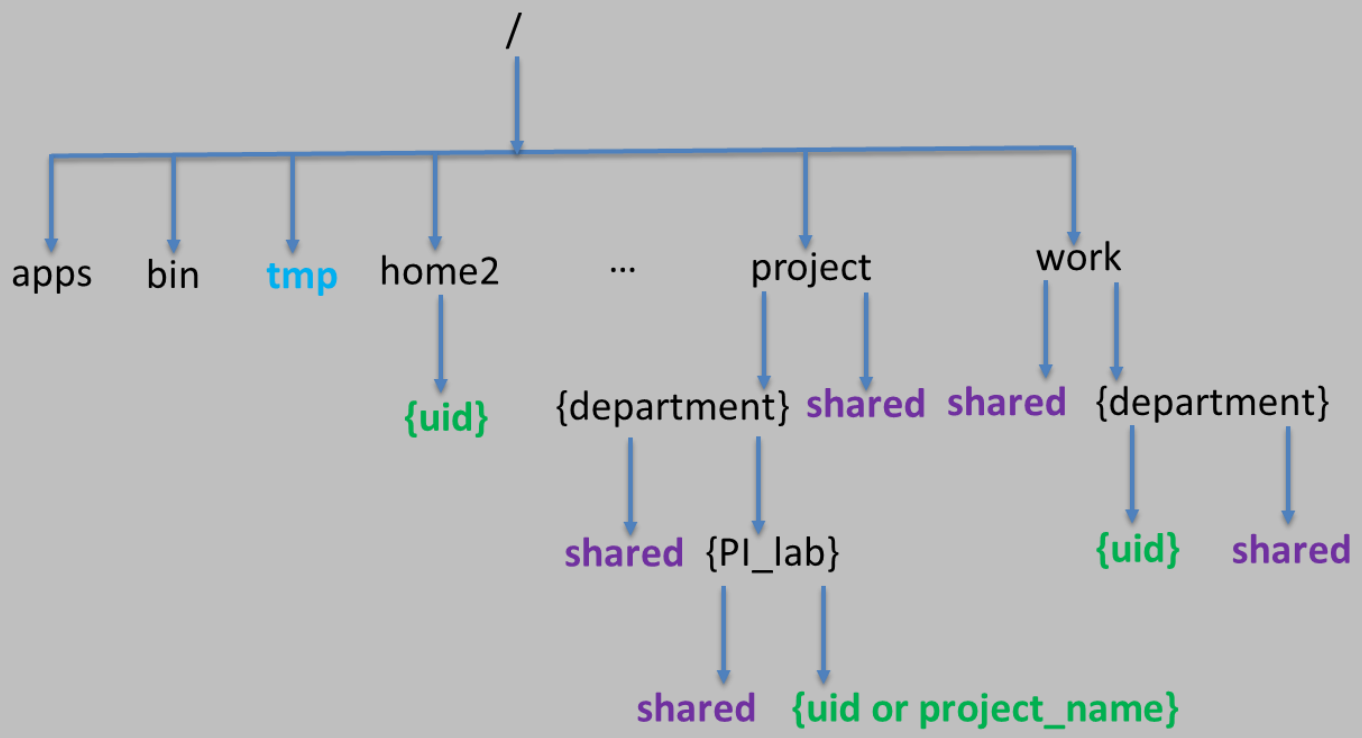
```
$ pwd
/home2/s191529

$ ls

$ ls /home2/s191529

$ ls ~

$ ls .

$ ls /work/biohpcadmin/s191529

$ ls /archive/biohpcadmin/s191529

$ ls /project/biohpcadmin/s191529

$ ls -lt

$ ls -lah
```

pwd – print working directory

ls – list contents of a directory

Filenames that start with **.** are hidden. You can view them with the **ls** command and the **-a** option (**a** for all).

Navigating the file systems



Everything in Linux is a file. Files on a Linux system are arranged in a **hierarchical directory structure**. The first directory in the filesystem is named the **root** directory.

Navigating the file systems

```
$ cd /work/biohpcadmin/s191529

$ cd ..

$ cd s191529

$ cd ~

$ cd -
```

1. Can you navigate into your personal folders in BioHPC?
2. What is your current directory?
3. What does the "/" at the beginning of the path mean?
4. Can I enter someone else's folder?
5. What is an absolute or a relative path?

cd – change directory

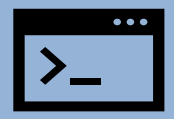
Shortcuts to help you out:

- **.** This is the directory you are currently in.
- **..** Takes you to the directory above your current one.
- **~** This directory defaults to your home directory.
- **-** This will take you to the previous directory you were just at.
- Finally, the **Up Arrow** brings the last command you hit.

How much storage is a directory occupying?

```
$ $ du -sh Documents/misc/

$ touch myfile.txt

$ file myfile.txt

$ quota -a

$ id s191529

$ lfs quota -g 1001 /project -h
```

1. Find you GID.
2. Check out your quota in BioHPC filesystems.
3. Create a file in /project, /work and /home2 folders.



du – disk usage (**-h** – human readable; **-s** – summarize)

touch – create a file, change timestamps

file – determine file type

In Linux, file extensions aren't required.

Some useful commands

```
$ df -h

$ who

$ whoami

$ top

$ exit

$ history

$ man history

$ !990

$ !!
```

history – Linux's shell saves a history of the commands you run

df – report file system disk space usage

top – display Linux processes

who – show who is logged in

how-to-use-the-history-command-on-linux

Exploring the file systems and environment variables

```
$ cd /project/shared/biohpc_training/

$ cat c475_r0ck_4m_1_r16h7.txt

$ file RJ_WS

$ clear

$ reset

$ echo 'Hello!'

$ export MESSAGE=Hello!

$ env

$ echo $MESSAGE

$ unset MESSAGE
```

clear – clear the terminal screen

reset – reinitialize the terminal

env – check environment variables

echo – display a line of text

cat – concatenate files and print on the standard output

More on environment variables

```
$ env

$ echo $SHELL
/bin/bash

$ echo $HOME
/home2/s191529

$ echo $USER
s191529

$ echo $PATH
/home2/s191529/.local/bin:/cm/shared/apps/slurm/
16.05.8/sbin:/cm/shared/apps/slurm/16.05.8/bin:/
usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin

$ export PATH=/home2/s191529/bin:$PATH
```

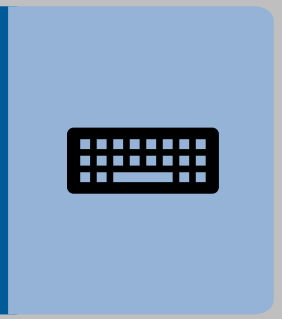
$PATH variable is one of the most important and tells the shell where your programs are

The module system on BioHPC modifies this **PATH** so that programs are made available to the user. One can also manually edit their **PATH**

Viewing files

```
$ cat HD728.R1.fastq

$ wc -l HD728.R1.fastq

$ less HD728.R1.fastq

$ head HD728.R1.fastq

$ tail HD728.R1.fastq
```

CTRL + C	CTRL + Z
CTRL + U	CRTL + K
CTRL + W	CTRL + Y



The command **cat** is not good for large files.

less – is more

wc – print newline, word, and byte counts for each file

more – is a filter for paging through text one screenful at a time, but you shall use **less**

Command to navigate through **less**:

- **q** – quit
- **Page Up/Down** & **Up/Down**
- **/** – search
- **h** – help

Permissions

```
$ chmod 700 blah.txt

$ chmod 770 blah.txt

$ chmod 777 blah.txt

$ chmod o-x blah.txt

$ chmod o-w blah.txt

$ chmod o-r blah.txt
```

---	---	---
000	000	000
rwx	rwx	rwx
111	111	111
user	group	other

There are 3 things you can do to a **file**:

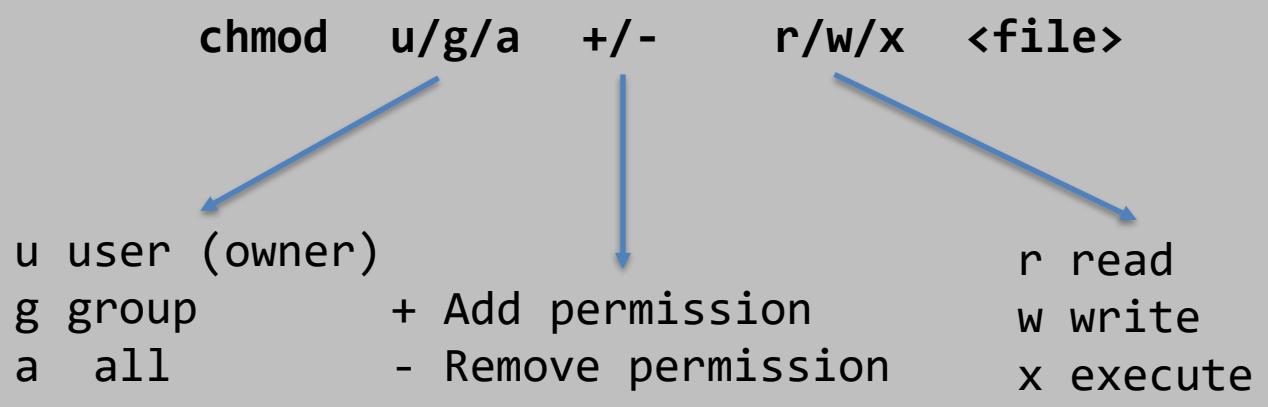
- **r**ead – r
- **w**rite – w
- e**x**ecute – x

Permissions are set for the **user**, the **group** and the **others**. The **user** is the **owner** of that file, which in turn might also belong to a **group**.

Directories also have **rwx** permissions:

- **r** – can list files
- **w** – can create files
- **x** – can navigate into

Permissions




```
chmod g+rw script.sh  # Add read/write permissions for the group
chmod a+x script.sh   # Add execute permission for everyone
chmod g-x script.sh   # Remove execute permission for the group

chmod 700 script.sh   # ?
chmod 640 script.sh   # ?
```

Navigating through BioHPC filesystems

- Do permissions in BioHPC make sense?
- Where to store a file so that every member of your lab has access to that file?
- Where to store a file so that every member of your department has access to that file?
- Can you change the permissions of any file or directory?

Octal	Binary	File Mode
0	000	---
1	001	--x
2	010	-w-
3	011	-wx
4	100	r--
5	101	r-x
6	110	rw-
7	111	rwx

Text editors

- **Vim**
  - Cryptic commands! Cheat sheet on the portal:
    http://www.washington.edu/computing/unix/vi.html

- **Emacs**
  - An extensible, customizable text editor: http://www.gnu.org/software/emacs/tour/

- **Nano**
  - Easier to use: http://mintaka.sdsu.edu/reu/nano.html

- **Any text editor from your PC or Mac**
  - Mount your directories as network drives:
    https://portal.biohpc.swmed.edu/content/guides/biohpc-cloud-storage/

Working with files and folders

```
$ mkdir -p s191529/temp

$ mv blah.txt bla.txt

$ mv blah.txt s191529/temp/

$ cp -r  ~/cuda-samples .

$ cp -r cuda-samples/ s191529/temp/

$ cp -r cuda-samples/* s191529/temp/

$ ls -l s191529/temp/

$ rm -rf s191529/temp/

$ rm -i cuda-samples/*
```

mkdir – create an empty directory

cp – copy from source to destination (use the **-r** option to copy recursively)

mv – move from source to destination (or rename a file or directory)

rm – remove file or directory (use the “r” option to remove recursively)

Use the **-i** option (interactive) to prompt you before overwriting a file. By default, **cp** will apply your ownership and primary group to files.

Working with files and folders

1. Create a folder and a file in that folder.
2. Create another folder and copy the file you created into the new created folder.
3. Delete the file you created in one of those folders.
4. Delete the empty folder.

Be very cautious of your ability to destroy files!

There is **no Recycling Bin** to restore your files.

Once files are deleted by the CLI, it is generally very difficult to recover them.

Make sure important data is backed up!

Redirection operators

```
$ cat ~/.bashrc

$ cat < ~/.bashrc

$ cat ~/.bashrc > bashrc.txt

$ echo ‘The Matrix has you!’ >> bashrc.txt

$ ls -l /bin/usr > error.txt 2>&1

$ ls -l /bin/usr > /dev/null 2>&1
```

File Descriptor	Name	Standard Stream
0	Standard Input	stdin
1	Standard Output	stdout
2	Standard Error	stderr

< – input redirection

> – standard output redirection

>> – output append operator

*“To begin, **/dev/null** is a special file called the null device in Unix systems. Colloquially it is also called the **bit-bucket** or the **blackhole** because it immediately discards anything written to it and only returns an end-of-file (**EOF**) when read.”*

Finding files and directories

```
$ find . -name README.md

$ find . -iname readme.md

$ find /home2/s191529 -name hello_py.slurm

$ find /home2/s191529 -type d -iname pgi

$ find . -name "*.h"

$ find . -type f -perm 755

$ find . -type d -perm 777 -print -exec chmod 755 {} \;

$ find . -type f -name ".*"

$ find . -group biohpc_admin
```

find – search for files in a directory hierarchy

The **find** command doesn't stop at the directory you are searching, it will look inside any subdirectories that directory may have as well

Wildcards

***** **Match any number of characters:**

`$ ls *`	Any file
`$ ls notes*`	Any file beginning with notes
`$ ls *.txt`	Any file ending in .txt
`$ ls *2019*`	Any file with 2019 somewhere in its name

? **Match a single character:**

`$ ls data_00?.txt`	Matches data_001, data_002, data_00A, etc.

[] **Match a set of characters (bracket expression):**

`$ ls data_00[0123456789].txt`	
`$ ls data_00[0-9].txt`	Matches data_001 – data_009, not data_00A

Transferring files

```
$ scp s191529@nucleus.biohpc.swmed.edu:~/config.yaml .

$ scp file.txt s191529@nucleus.biohpc.swmed.edu:~

$ scp -r dir s191529@nucleus.biohpc.swmed.edu:~

$ rsync s191529@nucleus.biohpc.swmed.edu:~/config.yaml .

$ rsync file.txt s191529@nucleus.biohpc.swmed.edu:~

$ rsync -r dir s191529@nucleus.biohpc.swmed.edu:~
```

scp – secure copy (remote file copy program)

Any similarity with the **cp** command? It's almost the same except that you have to specify username and IP address with colon "**:**".

rsync – a fast, versatile, remote (and local) file-copying tool

The Pipe |

```
$ command1 | command2 | command3 | .... | command

$ ls -l | more

$ sort RJ_WS | uniq
```

Whenever we work with **|**, **|** command will take STDOUT of command and transfer it to STDIN of the subsequent command.

Bonus commands:
wget – network downloader
curl – transfer a URL

Questions?