Syllabus

- May 18: Introduction
- May 25: Python tutorial
- Jun 1: NetCDF + Python
- Jun 8: NetCDF + Ferret
- Jun 15: Advanced processing NetCDF, Python
- After: User requests for projects

Week 1: Intro stuff

Outline

- Intro to UNIX
- 10 most import commands
- Departmental Computing Layout
- Computing Tips

UNIX, the layered OS

- Hardware, physical components of the computer
- Kernel, software interface layer to hardware
- Libraries, public methods to interface with the kernel
- Programs, call libraries to execute code on the hardware.

UNIX Filesystems

- UNIX has a 'rooted' filesystem. '/' is referred to as the root directory.
- Each node in the filesystem is either a file or a directory. Directories are special files...
- Two directory names are reserved
  - “.” refers to the local directory
  - “..” refers to the parent directory

Referencing files

- Local Reference
  - “./myscript.sh”
- Relative Reference
  - “../../bin/myscript.sh”
- Absolute Reference
  - “/usr/bin/myscript.sh”

Ex) “/home/akrherz/myscript.sh”

- Which of the following does not reference the above file?
  - “/home/./akrherz/../akrherz/./myscript.sh”
  - “/../home/akrherz/myscript.sh”
  - “/./home/../../../../../home/akrherz/myscript.sh”
  - “/home/akrherz/myscript.sh/../myscript.sh”

Base UNIX Filesystem

- /
  - bin (System Binaries)
  - etc (System Configs)
  - home (User Data)
  - lib (System Libs)
  - sbin (Admin Binaries)
  - tmp (Temp Space)
  - usr (User Binaries)
  - var (App Data)

Working within UNIX

- Shells
  - bash, csh, ksh, tcsh ...
- Scripts
  - shell scripts, perl scripts, python scripts, etc...
  - Executed at run-time
- Programs / Executables
  - pre-compiled. C++, FORTAN, etc...

Running Commands

- Commands need to either be
  - in your $PATH
    - ex) ls
  - referenced directly
    - ex) /bin/ls
- Commands can take arguments/flags
  - Here is a flag
    - ex) ls -l
  - Here is an argument
    - ex) ls myscript.sh
  - Here is both
    - ex) ls -l myscript.sh

Getting Help with your Program

- Most programs should have one of the following help options available
  - man command
    - ex) man ls
  - command –help
    - ex) ls --help

File Permissions / Ownerships

- $ ls -l qa4.css
  -rw-rw-r-- 1 akrherz users 2275 Dec 17 21:01 qa4.css
- Lots going on here!
  - “-rw-rw-r--” are file permissions
  - “1” is the number of directories+files
  - “akrherz” is the owner of the file
  - “users” is the group owner of the file
  - “2275” is the filesize in bytes
  - “Dec 17 21:01” is the file modification time

10 Commands you must know

- ls
- cd
- df
- du
- pwd
- w
- ln
- chmod
- chown
- tar

ls

- list directory contents
- Useful flags
  - -a “list all”
  - -d “don't walk dirs”
  - -l “long format”
  - -r “reverse output”
  - -R “recursive”
  - -t “sort by mod time”
- Common uses
  - ls -l
  - ls -latr
  - ls -d
  - ls -R

cd

- Built-in shell command
- changes your current working directory
- No meaningful flags
- Examples
  - cd ..
  - cd /tmp

df

- Reports filesystem disk space usage
- Useful Flags
  - -l “local filesystems”
  - -h “human readable”
  - -k “1k blocks”
- Examples
  - df -l
  - df -h
  - df /home

du

- Estimate file space usage
- Flags
  - -h “human readable”
  - -x “one filesystem”
  - -s “display summary”
- Examples
  - du -s -h *
  - du .

pwd

- Print name of the current working directory

- One example!
  - pwd

w

- show who is logged in and what are they doing
- No useful flags
- Should always be the first command you run after logging in
- Example:
  - w

ln

- make links between files
- One flag:
  - -s “source”
- Example:
  - ln -s realreal linked

chmod

- Change file permissions
- Flags:
  - -R “recursive”
- Octet based ex) 755
- Named base u+rw
- Examples
  - chmod +x myscript.sh
  - chmod 775 directory
  - chmod u+w,g-r,o-w file

chown

- Change file owner and group
- Flags:
  - -R “recursive”
- Typically, you would only run the command to change group permissions
- Examples
  - chown akrherz.gcp myfile.html
  - akrherz is the user
  - gcp is the group

tar

- Archiving Utility
- Lots of Flags:
  - -f “file”
  - -z “gzip input/output”
  - -c “create”
  - -x “extract”
  - -t “list”
  - -j “bzip input/output”
- Examples:
  - tar -cf myfile.tar .
  - tar -xf myfile.tar
  - tar -xzf myfile.tgz
  - tar -tzf myfile.tgz
  - tar -xf myfile.tar mydir

Our Computing System

- /home is NFS mounted and the same on most every machine
- /usr/local contains common programs and is the same on all systems
- User passwords are the same on all systems
- Many data drives are NFS mounted and available on the faster machines...

Tips for efficient computing

- Output from the 'ls' command should never cause the terminal window to scroll
- The more directories, the better!
- Use the /tmp directory for quick processing
- Place scripts in your $HOME directory, and data in a /something directory
- Use sym links, if needed

More tips

- Run your processing scripts physically close to your data. Understand what the command 'df' is telling you.
- Don't just run code to see what happens. This hurts your computing efforts and those of others
- Keep your $HOME directory clean and organized.
- Use tar to compress your output and to move to backups. Be careful when you run tar!

Assignment for Next Time

- I will email you a spreadsheet with all of the data shares listed.
- Log into each of those machines you have data on and produce a disk usage report for yourself
- Save those values in the spreadsheet. We will use it next time.