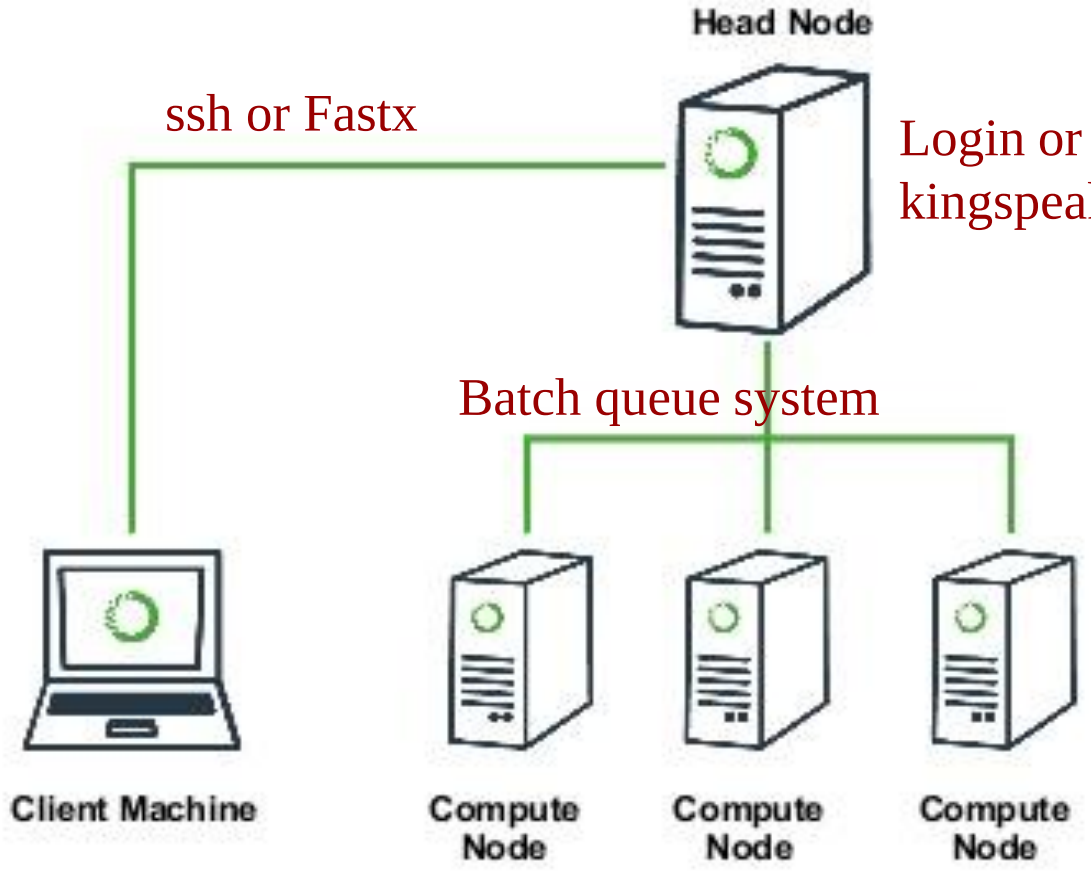
Introduction to Linux – Part 1

Anita Orendt and Wim Cardoen

Center for High Performance Computing

Cluster Architecture Diagram



Getting Started – Login using FastX

- Open web browser to:

  http://linuxclass.chpc.utah.edu:3000

- Enter temp login and password (or your uNID & password if you have a CHPC account) and hit “Log In” button
- Hit the “Launch Session” button
- Click on “xterm”, then hit the “Launch” button

FastX

- https://www.chpc.utah.edu/documentation/software/fastx2.php
- Persistence – you can disconnect without closing session, lets you resume sessions from other devices
- Licensed by CHPC
- Web based client option
- Desktop clients for windows, mac, and linux
- Available on all CHPC interactive nodes and the frisco nodes
- More efficient than simple X forwarding

Alternatives to FastX on Windows

- Ssh clients
  - PuTTY
    - http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html
  - XShell
    - http://www.netsarang.com/download/down_xsh.html
- For graphical programs also need X-forwarding tool
  - Xming (use Mesa version as needed for some apps)
    - http://www.straightrunning.com/XmingNotes/
  - Make sure X forwarding enabled in your ssh client

Alternatives to FastX on Mac/Linux

- Just open up a terminal or console window
- ssh *your_login*@linuxclass.chpc.utah.edu
- When running applications with graphical interfaces, use ssh –Y or ssh –X

Shell Basics

- A Shell is a program that is the interface between you and the operating system (OS – e.g, linux)
- Command line interface – CLI – versus a GUI – or a graphical user interface
- Type commands on command line, send command by pressing enter (or return), then the computer reads and executes the command and returns the results (NOTE – not all commands have output!)
- When commands are done they return to the PROMPT (more on prompts later)
- Commands can take flags that modify their behaviour

    - flags are formed with – (dash) and letters

- Commands can also sometimes require an argument – this defines the item upon which the command acts

Additional Shell Basics

- Linux is case sensitive!
- We will focus on two basic shells - slightly different command syntax
  - `bash` (Bourne again shell)
  - `tcsh` (TENEX C shell)
- While many shell commands are the same between shell types – there are syntax and behaviour differences
- Your account comes with a script that is executed upon login that sets a basic environment for your shell
- To check which shell you are using: echo $SHELL
  - Note $SHELL is an environment variable – more on these later
- To change shell for the session - enter name of shell you want at the prompt and hit enter

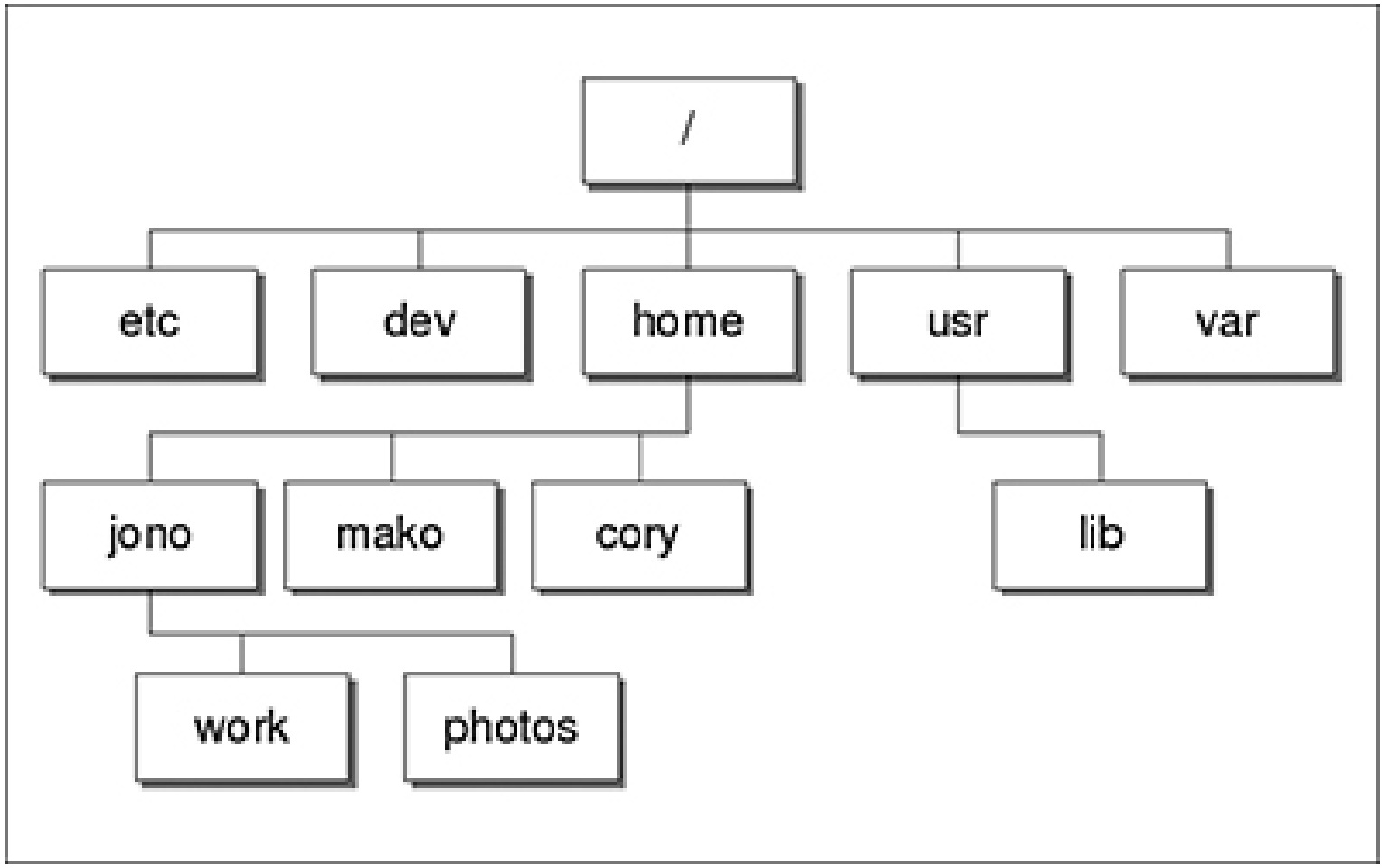
Other Useful Items

- Up/down arrows go through past commands
- **`history`** – provides list of all recent commands; can ! followed by number from history list will put that command at the prompt
- **Tab completion** – of commands, paths, filenames – very useful
- Can edit previous commands – up and down arrow to get to command; then right/left arrow, then delete any characters and type in new at cursor; cntrl-a gets to front of command line, cntrl-e to end of command line

Directory Structure

- The **file system** is a tree directory structure
- Levels are separated by /
- / --- refers to the “root” directory – the top level directory that contains all other directories
- The home directory is used to refer to a user’s base directory – this is where you will be upon login
- /path/from/root ➔ absolute path – has leading /
- path/without/leading/slash ➔ relative path from current location
- . ➔ current directory
- .. ➔ parent directory (up one level)

Directory Structure



At CHPC --- instead of /home, user directories located under /uufs/chpc.utah.edu/common/home

Login & Prompts

- When you first login you will see a prompt (the prompt is set by the login script)
  - [u0424091@linuxclass:~]$
  - [user33@linuxclass:~]$
- When you first login, you will be in your home directory
- To see your username: **`whoami`**
- To see your current directory: **`pwd`**
- Shortcuts
  - **`~`** ➔ your home directory
  - **`$HOME`** ➔ your home directory
  - **`~username`** ➔ someone else's home directory

Exercise

- Get logged in.
- What is your shell?
- What is your username?
- What is the path of your current directory?

Basic Directory Commands

- **`ls`** – list contents of a directory
  - Flags to change output To see all flags
    - ls --help
    - man ls
- **`mkdir`** – make directory (`mkdir test`)
- **`cd`** – move to directory (`cd test`)
  - **`cd`** without an argument moves you back to your home directory
  - **`cd ..`** -- moves you up one level
- **`rmdir`** – remove directory (`rmdir test`) – more on this later

More on ls flags

- -l : long
- -a : All (including hidden files, also called dot files)
- -r : Reverse ordering while sorting
- -t : Timestamp

Files & Filenames

- Directories can contain files and other directories
- Filenames may have an extension, like “homework.pdf”
- Extensions are useful for telling you what type of file it is – IF you follow the conventions (txt, pdf, jpg, etc)
  - The extensions also are used by the OS
  - The **`file`** command will tell you the file type
- Being careful with filenames can make your life easier – some guidelines:
  - Avoid special characters in names as you will have to handle these differently: space, tab, /, \, $, leading -
- Files that start with a “.” are hidden or “dot” files

Login Scripts & Environment Variables

- In your home directory are a a number of dot files - `.bashrc` and `.custom.sh`, `.tcshrc` and `.custom.csh` Depending on your shell choice, the appropriate pair of these are executed during login
- These set the environment (as environment variables) needed for you to work on CHPC resources
- Commands to check your environment: `env` or `printenv`

File commands

- **`cat`** – display contents of file
- **`less`** – display contents of file with page breaks (next page with space key) – can also look at **`more`**
- **`head`** – display top of file (default is 10 lines, change with -n)
- **`tail`** – display end of file (default is 10 lines, change with -n)
- **`grep`** – search for pattern in file (`grep "pattern" test1`)
- **`vi`** – edit file (more on this later)
- **`cp`** – copies file to a new name (`cp file1 file2`)
- **`mv`** – renames file to a new file (`mv old new`)
- **`touch`** – creates an empty file if file does not exist OR changes time stamp if it does (`touch file`)
- **`rm`** – deletes file (`rm file1`)
  - Note shells DO NOT have a trash bin; rm is final!

Wildcards

- more files can be specified via wildcards
- `*` - matches any number of letters including none
- `?` - matches any single character
- `[]` - encloses set of characters that can match the single given position
- – used within `[]` denotes range of characters

Examples:

```
*.csh , *.*sh , figure?.jpg , *.txt ,
figure[0-9].*
```

Exercise

- Make sure you are in your home directory, then make a directory called `IntroLinux1` and cd to that directory.
- Use “ls” to display the contents of MY IntroLinux1 directory: /uufs/chpc.utah.edu/common/home/u0028729/IntroLinux1
- Copy over the contents of my IntroLinux1 directory into the directory you are in.
- List contents of your IntroLinux1 directory – try different ls options, e.g. `ls -l`, `ls -ltr`, `ls -a`, and `ls -ltra`
- See what output you get when you do a ls of: `figure?.jpg`, `figure[0-9].*`
- Make a new directory called `Work` inside of `IntroLinux1` and copy all files with the `txt` extension from the `IntroLinux1` directory to your new directory
- Open man page for some command (e.g. `ls`) and see what these flags do

Exercise

- If you are not there already, cd into your `IntroLinux1` directory
- View `script.slurm` using `cat`, `less`, `head` and `tail`
- Vary number of lines viewed with `head` and `tail`
- Search for the string `SBATCH` in this file with `grep`
- Use the file command to tell you what the file type of `ShellReference.pdf`; copy this file to another filename, with a different extension and check the file type again

Command output redirection

- **>** redirects output to a file (instead of to screen)
  - will create file if it does not exist
  - will overwrite the previous contents if it does exist
  - `cat file1.dat > file4.dat`
- **>>** appends to a file
  - `cat file1.dat >> file3.dat`
- **|** - pipe – redirect command output to another command
  - `head -10 file.txt | tail -2`

Exercise

- In your `Work` directory, combine the contents of `geom1.txt` and `geom2.txt` into one file named `geom3.txt`
- Using `grep` and the file `states.dat` create a file `Mstates.dat` with only the states that start with the letter M
- Create the same file content using `head` and `tail`

File Permissions

- Shown with `ls -l`
- Permissions are read (r), write (w), execute or search for a directory (x)
- **`chmod`** – changes permissions of file or directory
- User (u), group (g), other (o), all (a)
- Examples:
  - `chmod g+x filename`
  - `chmod o-rwx *.c`
- Executable files (programs and scripts) must have executable permissions

Processes

- A Process is a running Linux program
  - Each process has a PID (Process ID)
- **`ps`** reports a snapshot of current processes
  - ps, ps x Display ALL of your processes
  - ps ax Display ALL processes
  - ps aux Display ALL processes (more detailed)
  - ps auxw Display ALL processes (more detailed & unlimited width)
  - ps –eFwww Also displays ALL processes
- **`kill PID`** kills the process with the specified PID

Some other useful commands

- `pushd` *`directory_name`*`, popd` *`directory_name`*`, dirs`
  - "pushes" and "pops" directory_name on to / off of a stack
- `wc` – e.g. `wc -l file.txt`
  - Prints line (-l), word (-w), character (-m) or byte (-c) count of file
- `cut` – e.g. `cut -f 2 -d : file.txt`
  - Prints selected columns of lines from file to standard output (screen)
- `du` – e.g. `du -hs`
  - Reports file space usage; -s give summary of total usage, -h gives it in "human readable" format of K, M, G
- `df` – e.g. `df -h`
  - Reports file system disk space usage
- `ln` – e.g. `ln -s ~/bin/prog.exe prog1.exe`
  - create a link between files (-s symbolic)

On your own – Use and explore options of these commands

Have Questions?

- Anita: anita.orendt@utah.edu
- Wim: wim.cardoen@utah.edu
- CHPC has an issue tracking system: helpdesk@chpc.utah.edu
- Slides and files: ~u0028729/IntroLinux1
- Some useful websites

  http://swcarpentry.github.io/shell-novice/

  http://linuxcommand.org/