

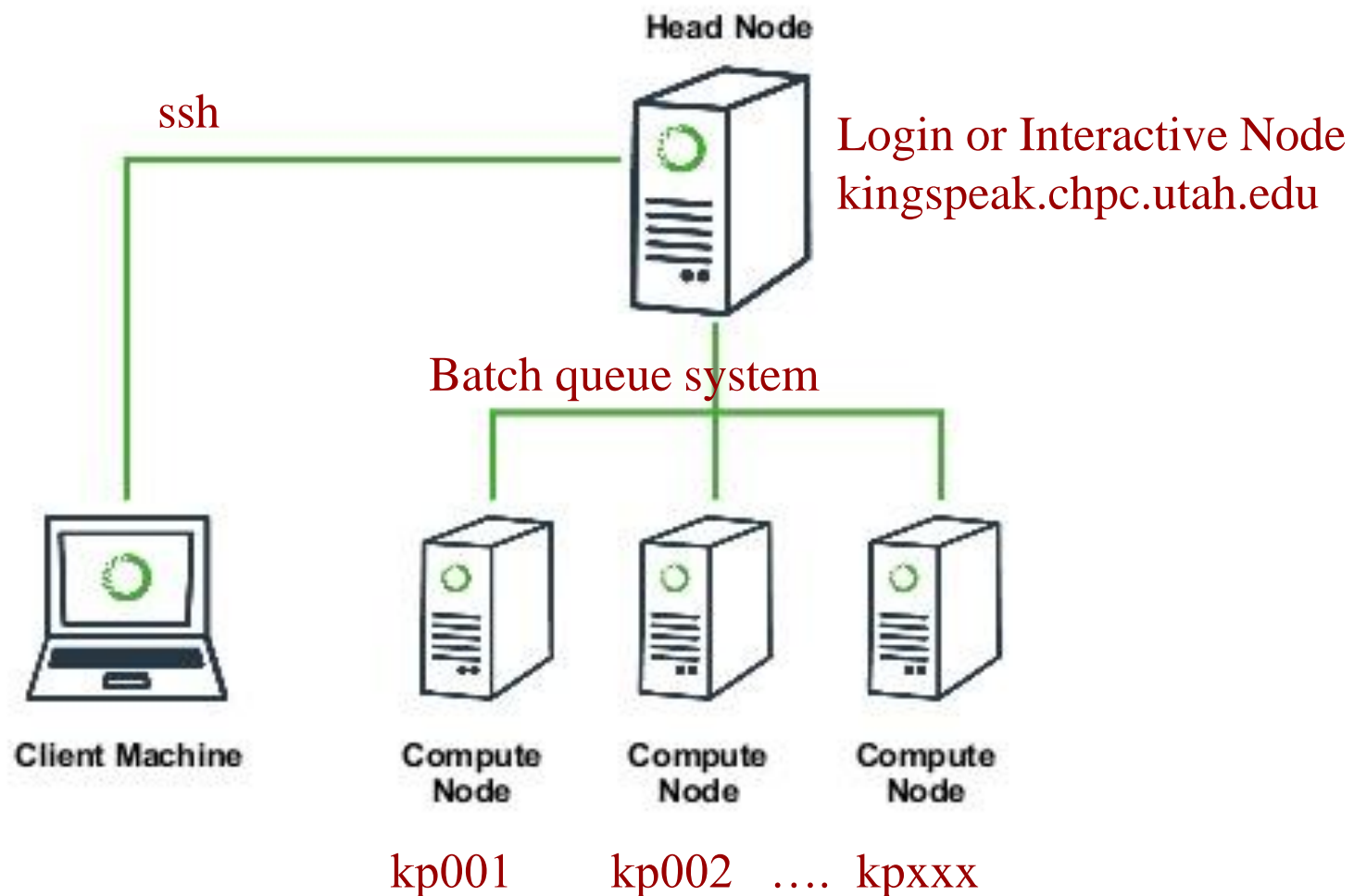
Introduction to Linux – Part 1

Anita Orendt and Wim Cardoen

Center for High Performance Computing

31 August 2017

Cluster Architecture Diagram



FastX

- <https://www.chpc.utah.edu/documentation/software/fastx2.php>
- Remote graphical sessions in much more efficient and effective way than simple X forwarding
- Persistence - can be disconnected from without closing the session, allowing users to resume their sessions from other devices.
- Licensed by CHPC
- Desktop clients exist for windows, mac, and linux
- Web based client option
- Server installed on all CHPC interactive nodes and the frisco nodes.

Windows – alternatives to FastX

- Need ssh client
 - PuTTY
 - <http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>
 - XShell
 - http://www.netsarang.com/download/down_xsh.html
- For X applications 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

Linux or Mac Desktop

- Just need to open up a terminal or console
- When running applications with graphical interfaces, use `ssh -Y` or `ssh -X`

Getting Started - Login

- Download and install FastX if you like (required on windows unless you already have PuTTY or Xshell installed)
- If you have a CHPC account:
 - `ssh unid@linuxclass.chpc.utah.edu`
- If not get a username and password:
 - `ssh userXX@linuxclass.chpc.utah.edu`

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, 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 letter
- ❑ 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
 - `csh/tcsh` (C, TENEX C shell)
 - `sh/bash` (Bourne, Bourne again)
- ❑ 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

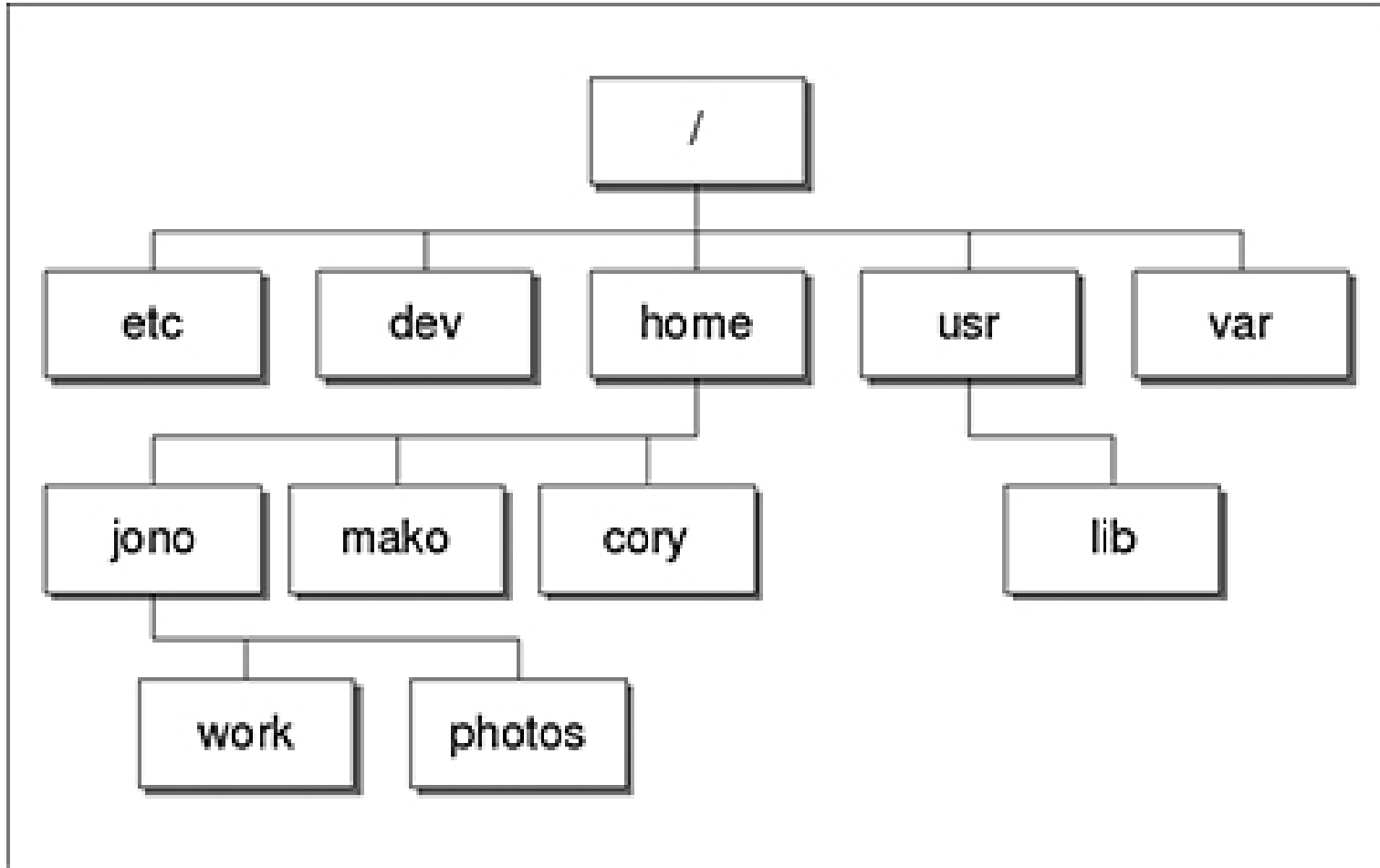
Other Useful Items

- 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

- ❑ / --- refers to the “root” directory – the top level directory that contains all other directories
- ❑ There is a tree directory structure – levels are separated by /
- ❑ The home directory is used to refer to a user’s base directory – this is where you will be upon login
 - ❑ If you have a CHPC account this is in
/uufs/chpc.utah.edu/common/home/<yourusername>
 - ❑ On linuxclass it will be /home/<yourusername>
- ❑ /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 we have /uufs/chpc.utah.edu/common/home under which we have all user directories

Login & Prompts

- ❑ When you first login you will see a prompt (the prompt is set by the login script)
 - ❑ `[u0028729@kingspeak1 ~]$`
 - ❑ `[userxx@linuxclass:~]$`
- ❑ When you first login, you will be in your home directory
- ❑ To see your username: **whoami**
- ❑ To see your current directory: **pwd**
- ❑ Shortcuts
 - ❑ **~yourusername** → your home directory
 - ❑ **\$HOME** → your home directory

Exercise

- Download and install FastX if you do not yet have it on your desktop.
- Login --
- 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

- ❑ Within directories you can have other directories and also files
- ❑ Filenames are often name.extension
- ❑ Files that start with a . are hidden or dot files
- ❑ 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:
 - ❑ Do not use white spaces or other special characters in names as you will have to handle these differently

Login Scripts & Environment Variables

- In your home directory are 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
- ❑ **more** – display contents of file with page breaks (next page with space key) – can also look at **less**
- ❑ **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 and then make a directory called `IntroLinux1` and change into this directory
- ❑ Look at the contents of one of MY directories:
`/uufs/chpc.utah.edu/common/home/u0028729/IntroLinux1`
- ❑ Copy over the contents of this directory into the directory you are in
- ❑ List contents of this directory – see difference of a normal `ls`, `ls -l`, `ls -ltr`, 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 already, move into your `IntroLinux1` directory
- ❑ **View** `script.slurm` using `cat`, `more`, `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

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

Exercise

- ❑ In the `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`
- ❑ User (u), group (g), other (o), all (a)
- ❑ Permissions are read (r), write (w), execute or search for a directory (x)
- ❑ **chmod** – to change permissions of file or directory
- ❑ Format `chmod g+x file`
- ❑ 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

- **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 parts 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
- ❑ Brett: brett.milash@utah.edu
- ❑ CHPC has an issue tracking system:
issues@chpc.utah.edu
- ❑ Slides and files `~u0028729/IntroLinux1`
- ❑ Some useful websites
<http://swcarpentry.github.io/shell-novice/>
<http://linuxcommand.org/>