UNIX in 30 minutes (or less!)

Sean Caron

UNIX Systems Administrator
Center for Statistical Genetics

16 June, 2014

Agenda

- Client prep, logging in with SSH and X11
- $PATH and CWD, running X11 apps
- File and directory manipulation
- Process and storage management
- Editing text
- Program development, MOSIX 101
- Network connectivity
- Miscellaneous topics, references

Windows client prep

- Start X11-server (Xming) on workstation FIRST
  - Start menu -> All programs -> Xming -> Xming
- Run SSH client (PuTTY)
  - Category: Connection -> SSH -> X11
  - Check "Enable X11 forwarding"
  - Back up to Session:
  - Host name: seqshop-server.sph.umich.edu
  - Port 22
  - Click "Open"

Mac client prep (for reference)

- Start X11-server (X11.app) on workstation FIRST
  - Applications -> Utilities -> X11.app
  - If not present, check your OS X CD or download from Apple.
- Run SSH client (ssh)
  - Applications -> Utilities -> Terminal.app
  - ssh -Y username@seqshop-server.sph.umich.edu

Logging in

- Depending on your client, you will be prompted for both username and password, or just your password if username was specified earlier.
- You usually will have 3-5 tries to successfully log in before the server disconnects.
- After successfully logging in:
  - UNIX starts a shell under your user ID.
  - Per-user login script executes (if applicable).
  - You will receive a command prompt ($ or %).

PATH and CWD

- If a command is not internal to the shell, the shell searches a list of directories ($PATH).
- If it finds a program with the same name as the command you typed...
- It runs it.
- Otherwise, it says "Command not found".
- It usually doesn't check your current working directory (CWD).

PATH and CWD

- So: if we want to run a program in our current working directory, we use a dot-slash prefix:
  - ./myprogram [arg1] ... [argN]
- To show what's currently set as your PATH:
  - echo $PATH
- To edit your PATH:
  - setenv PATH /some/new/dir:$PATH
  - PATH=/some/new/dir:$PATH ; export PATH

Running X-applications

- Since you ran an X-server on your workstation and enabled X11-forwarding before connecting, you are not limited to just command line apps!
- You can run X11 GUI apps with output showing up on your desktop workstation!
- All required set up was done in SSH when you connected.
  - No need to adjust $DISPLAY.
  - No need to use xhost.

Running X11 applications

- nedit [file] &
  - GUI text editor.
- firefox &
  - Web browser.
- {xpdf|evince} [file] &
  - PDF viewer.
- xterm &
  - Open another terminal window.

File manipulation

- cd [directory]
  - Change directory to [directory].
- pwd
  - Show current working directory (CWD).
- ls [-l] [dir]
  - Show contents of directory.
  - If directory is not specified, will default to show contents of CWD.
  - Use [-l] flag for long listing format.

File manipulation

- cp [-R] [source] [destination]
  - Copy file from source to destination.
  - Use -R to copy entire directory tree recursively.
- mv [source] [destination]
  - Move (rename) file from source to destination.
- rm [-r] [file|directory]
  - Delete a file or directory.
  - Use -r argument to delete recursively (danger!)
  - NOTE: deleted files cannot be recovered!

File manipulation

- rmdir [directory]
  - Delete an (empty) directory.
- ln [-s] [source] [target]
  - Create a link from [source] to [target].
  - Use [-s] argument for symbolic link vs hard link.
  - Generally you will want to make symbolic links.
- cat [file]
  - Dump a (text) file to screen.

File manipulation

- {more|less} [file]
  - Dump a (text) file to screen with pagination.
- {head|tail} [-n lines] [file]
  - Dump the {first|last} [lines] of file to screen.
- touch [file]
  - Create an empty file.
- wc [file]
  - Count the number of words, lines, etc in a file.

File manipulation

- grep [pattern] [file]
  - Look for pattern (regular expression) in file.
- cut [-d ...] [-f ...] [file]
  - Cut a specified field from file given a specified delimiter.
- sort [options] [file]
  - Sort file per [options].
- diff [options] [file1] [file2]
  - Show line-by-line differences between two files.

Process management

- ps
  - Show running processes under your account.
- {ps -uaxc|ps -e}
  - Show running processes for the entire system.
- uptime, top, htop, dstat [-a], etc.
  - System load average, monitor processes interactively, monitor I/O interactively...
- kill [-SIGNAL] [pid]
  - Send [SIGNAL] (usually KILL) to process.

Storage management

- du [-s] [-h] [file|directory]
  - Show disk usage for specified file or directory.
  - Print just [-s]ummary, [-h]uman readable output.
- df [-k] [-l]
  - Show disk utilization for all mounted devices.
  - Print in units of [-k]ilobytes, [-l]ocal devices only.
  - Local devices excludes NFS mounts.

Editing text

- Many text editors available on UNIX from line editors to full-screen editors.
  - ed
  - vi, vim, etc
  - pico, nano, etc
  - emacs
- We'll concentrate (briefly) on pico/nano since it is easiest to learn to use.
  - NOTE: pico and nano are basically identical.

Editing text

- To begin:
  - {pico|nano}
  - {pico|nano} [file]
- Totally screen-oriented editor.
  - Use arrow keys to move the cursor.
  - Just start entering text or hitting Backspace with the cursor in the desired position for edits.
  - Help at the bottom.
- Ctrl-X exits. If file modified, it will ask to save.

Program development

- make [Makefile]
  - Run build script.
  - If [Makefile] not specified, looks for Makefile in CWD.
- cc [-o output] [file1.c] [file2.c] ... [fileN.c] [-llib..]
  - Build C program from source.
  - Output program will be [output] otherwise a.out.
  - Specify extra shared libraries if you need them.
  - Compiles and links in one step.

Program development

- c++ [-o output] [file1.cpp] ... [fileN.cpp] [-llib...]
  - Build C++ program from source.
- f77 [-o output] [file1.for] ... [fileN.for] [-llib...]
  - Build FORTRAN program from source.
- gdb [options] [prog [core|procID]]
  - Use GNU debugger against running process or left over core dump file.

IF ALL ELSE FAILS...

- UNIX has excellent built-in help!
  - man [section] [command]
  - Search [section] of the manual for help with [command].
  - If section omitted, it will pick a sensible default.
- Many commands also have built-in help!
  - ./command [-h|--help]
- Google is also a great resource!
- Hit Ctrl-C to kill what you're doing, right now.

MOSIX cluster 101

- We have provided a 4-node MOSIX cluster for everyone to use here at the workshop (MOSIX is just one of many clustering methodologies).
- Each node has dual 6-core CPUs and 128 GB RAM. They accept jobs from the gateway node (seqshop-server).
- MOSIX use in this workshop may be somewhat cloaked by job scripts and Makefiles (?)
- Just so you know...

MOSIX cluster 101

- mosrun -b [...] [command]
  - Run command in MOSIX native mode; MOSIX tries to find best node automatically; working directory is CWD.
- mosbatch -j[node] -E[WD] [...] [command]
  - Run command in MOSIX batch mode; user manually specifies node, working directory.
- mosps
  - Show running MOSIX processes under your account.

MOSIX cluster 101

- mosps -e uax
  - Show running MOSIX processes for all users across the entire system.
- mosmon
  - Graphical load monitor for MOSIX.
  - Good way to find node ID numbers.
- kill [-signal] [pid]
  - Send signal (usually KILL) to process.
  - NOTE: we kill MOSIX jobs just like local jobs!

Network connectivity

- ssh [-Y] user@host.fqdn
  - Secure remote login as user to host.
- sftp user@host.fqdn
  - Secure file copy using FTP-style interface.
- scp localfile user@host:/remotefile
- scp user@host:/remotefile localfile
  - Secure file copy using RCP-style interface
- wget URL
  - Pull file specified in URL (HTTP,FTP) to CWD.

Network connectivity

- ftp [host.fqdn]
  - Open FTP connection to host.
  - FTP will prompt for user name and password.
  - WARNING! FTP is insecure. It sends the user name and password in unencrypted plain text.
  - Anonymous FTP is sometimes encountered in downloading public (non-confidential) data sets (e.g. 1000 Genomes).
  - Generally: use of (non-anonymous) FTP should be avoided.

Miscellanea

- Suffix an ampersand to your command line to run it in the background:
  - xlogo &
- You can use screen to protect your jobs from being terminated when you get disconnected:
  - screen [-d] [-r] [...]
- Use a combination of the ampersand and the nohup command s.t. jobs run after disconnect:
  - nohup myjob &

Miscellanea

- Input redirection:
  - myprog < inputfile
- Output redirection:
  - stdout only: myprog [>|>>] outfile
  - stdout and stderr: myprog &> outfile
- Pipes:
  - prog1 | prog2 | prog3 | ...
  - Output of prog1 directly presented to prog2, output of prog2 directly presented to prog3...

Miscellanea

- The true power of UNIX shines when you start layering pipes, redirection and leveraging the ability to run processes in the background - all together!
- nohup ./myjob.sh &> myjob.out &
  - Run myjob in the background, all output goes to myjob.out, nohup protects against termination.

References

- The UNIX Programming Environment
  - Kernighan & Pike, 1984, Prentice Hall
  - A well-written classic, fundamentals of UNIX haven't changed...
- Learning the UNIX Operating System (5e)
  - Peek, Todino & Strang, 2001, O'Reilly Press
- Beginning the Linux Command Line
  - van Vugt, 2009, APress
- Look for: PrenHall, AWL, O'Reilly, Apress