# Introduction to the Linux Command Line

Presented by Oralee Nudson
Systems Administrator
GINA

## Overview

- Brief Intro to the Linux Operating System
- Connecting to Remote Systems
- Working with Files
- File and Directory Permissions
- File I/O and Redirection
- Working with Processes
- Working with Large Data Sets
- Additional tools

# Linux Overview

- Unix-like Operating System (OS) developed by Linus Torvalds in 1991
- Open Source Software
- "Runs on more computer hardware platforms than any other OS" (wikipedia.org)
- Runs on Supercomputers, embedded systems, in the Amazon Web Services cloud
- The shell is a command line interface to the OS
  - Open a "terminal" window
  - Edit files
  - Launch processes or jobs
  - Check the status of running processes
  - Send signals to processes
  - Common shells: bash, ksh, tcsh, csh

# Connecting to Remote Systems

- Linux comes with command line versions of **ssh**, **sftp** and **scp**.
- Windows requires downloading a terminal and file transfer program (and optionally an X11 server). Common tools include PuTTY (text only), FileZilla (file transfer), and Xming (graphics support).
- Login with:
  - ssh –X –Y username@systemname.gina.alaska.edu
  - Example: ssh –X –Y onudson@bootcamp.gina.alaska.edu
- Copy files with:
  - scp myfiles.tar.gz username@systemname.gina.alaska.edu:~/exampleData/
  - Use a GUI: filezilla, fetch, winscp
- Logging in frequently? Set up your public/private ssh keys!

# Navigating the File System

- Linux is a collection of files and directories (think of folders)
- The top directory is called the "root" or "/"
- Some directories contain actual files, others provide access to hardware devices

# Practice Common Commands

```
$ pwd
$ ls
$ mkdir datasets
$ ls
$ cd datasets
$ ls
$ touch one-file
$ ls
$ rm one-file
$ ls
$ cd ..
$ ls
$ rmdir datasets
$ rm –rf datasets
$ ls
```

## Working with Files

- Quickly view the contents of a file with:
    - cat filename
    - less filename
        - Exit with "q"
- Documentation for shell commands
    - "man" pages
    - info
- View images with the "`display`" command
- Common Linux Text Editors
    - vim or gvim
    - Emacs
    - nano
    - nedit (X11 enabled only)

# vim text editor

- Text only editor, no graphics support
- Great tool to use when logging onto remote systems
- Three modes: command, insert, and last line
- Try it!
    - vim hello-world.txt
    - Hit the "escape" key for command mode
    - Hit the "i" key for insert mode
    - Enter your text
    - Hit the "escape" key followed by ":" for last line mode
    - Exit by entering last line mode and typing "wq" then hitting the enter key.

# File and Directory Permissions

- Permissions control access to files and directories
  - View permissions with the "ls –al" listing of your directory and files
  - Three categories of access:
    - user
    - group (type "`groups`" to determine which you belong to)
    - other
  - Three categories of permissions:
    - read
    - write
    - execute
  - Use "`chmod`" to modify access permissions
    - chmod u+r myDir (add read permissions for myself)
    - chmod g+rx myFile (add group read & execute permissions)
    - chmod 750 myFile (add group read & execute permissions)
    - chmod go-rwx myFile (remove group and other permissions)

# File and Directory Permissions

- Security Awareness:
  - World write permissions are discouraged.
  - Never share your login credentials (username & password) with others.
  - What else?

# File Input/Output & Redirection

- Three forms of input/output:
  - “stdin” from keyboard or a file
  - “stdout” to screen or a file
  - “stderr” to screen or a file
- Redirect I/O with
  - Greater/Less Than Symbols, “>” or “>>” or “<“
  - Pipes, “|”
- Tie stdout and stderr together with “2>&1”
  - # In bash:
  - ./generate-output.bash > my-data.20160517 2>&1

## Special Shell Characters

- "*" matches anything
- "?" matches a single character
- "&" backgrounds a running process

# Working with Active Processes

- “ps” allows you to view process statuses
  - Useful variations “ps –elf” and “ps –aux”
- “top” to view what’s eating up all the CPU resources!
  - Exit with “q”
- Send a signal:
  - CTRL+c (kill)
  - CTRL+z (suspend)
- Search with “grep”, then “sort”

# Working with Active Processes

Try it!

```
$ sleep 1000
$ ctrl-z
$ ps
$ fg
$ ctrl-c
$ sleep 1000 &
$ ps
$ fg
```

## Working with Active Processes

Try it!

```
# edit a new file called sleep-time.sh containing:
    #!/bin/bash
    echo “hello there.  I’m tired...”
    sleep 1005
    exit
$ chmod 700 sleep-time.sh
$ ./sleep-time.sh
```

# Working with Active Processes

- "kill" to terminate processes
- "man kill"
- Send particular signals, e.g. "kill –KILL 3039"
- Try it!
  - sleep 2000 &
  - ps
  - kill <pid>
  - ps

## Customizing the User Environment

- Environment Variables store short strings of information
- Important variables: $PATH, $HOME, $CENTER
- The shell auto-expands variables
- Set with
  - bash: export PATH=${PATH}:/home/onudson/bin
- View with “echo $PATH”

UNIVERSITY OF ALASKA FAIRBANKS

# User Environment

- Customize your login by modifying your $HOME "." files
- Example for bash users:
  - Add the following to your ~/.profile file:
    export PS1="Good Morning!% "
  - Then source the file with ". ~/.profile"

# Working with large datasets?

- Try tarballs and compression to save space!
  - Create a new tarball with: “tar –cvf may2016data.tar myData/*”
  - Compress the tarball with: “gzip may2016data.tar”

  - Extract a gzipped file with: “gunzip may2016data.tar.gz” or “unzip filename.gz”
  - Untar a tarball with: “tar –xvf may2016data.tar”

  - Can you tar and compress with one single command?
  - Can you extract and untar with one single command?

- Transfer large data sets to a remote system using…
  - rsync –avz ~/mayData/* username@bootcamp.gina.alaska.edu:~/mayData

UNIVERSITY OF ALASKA FAIRBANKS

# Fun with looping

- Need to automate a repetitive task or iterate through a list 100 times?  Try a loop!
  - for i in {1..100}; do echo $i; done
  - for i in {1..100}; do echo “Hello!  I am on count number $i”; done
  - for i in one.txt two.txt three.txt; do echo “stuff goes here” > $i; done

UNIVERSITY OF ALASKA FAIRBANKS

# The "find" command

- Looking for a file but you don't quite remember its full name?
  - find /System –name key*
  - find /System -type f
  - find /System -type f | wc -l
  - find /System -type f –exec grep foo {} \;

UNIVERSITY OF ALASKA FAIRBANKS

## Practice!

- https://cmdchallenge.com/

- https://github.com/blahah/command_line_bootcamp