Lecture 2: Linux Shell & Files

Lecture Participation Poll #2

Log onto pollev.com/cse374
Or
Text CSE374 to 22333
**Administrivia**

- HW 1 will release Monday
- Class webpage coming later today
- Class discussion board available
- Linux accounts will be available later this afternoon
  - Username = uwnetid
  - Password = tempPassword
- Meet some of your TAs!
  - Dixon
  - Andres
  - Tom
  - Leah
Computer Model

- OS: Linux
  - Interface: shell (bash)
  - Process
  - Process
  - Process
  - Process

- Users (many)

- One OS (CentOS) controls the computer.
- One filesystem stores data.
- Many processes are run. (A program runs one or many processes.)
- A shell is one process that allows for command line interface.
- Many users
What is the OS?

- Memory Management
- Processor Management
- Device Management
- File Management
- Security
- Control over system performance
- Job accounting
- Error detecting aids
- Coordination between other software and users
Bash Language

Bash acts as a language interpreter
- commands are subroutines with arguments
- bash interprets the arguments and calls subroutine
- bash has its own variables and logic

<table>
<thead>
<tr>
<th>Bash</th>
<th>Java</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interpreted</td>
<td>Compiled</td>
</tr>
<tr>
<td>Esoteric variable access</td>
<td>Highly structured &amp; strongly typed</td>
</tr>
<tr>
<td>everything is a string</td>
<td>Strings have library processing</td>
</tr>
<tr>
<td>easy access to files and programs</td>
<td>Data structures and libraries</td>
</tr>
<tr>
<td>good for quick &amp; interactive programs</td>
<td>good for large complex programs</td>
</tr>
</tbody>
</table>
Meet the Linux Shell

▪ Text based interface for Linux operating system

▪ We will be using the “Bash” shell
  - There are different versions, but for this course we will only be using bash

▪ Use echo $SHELL to check which shell you are using

▪ Bash in a unix shell and command language that is the default login shell for most Linux and MacOS

▪ Interpreted, not compiled
  - You’re on your own when things go wrong
Commands in the Shell

- The shell is a text-based interface that takes commands instead of clicks

- Commands are pre-existing programs
  - `command name` <options> <input || output>

- To learn about an individual command use “man”
  - `command name` man
  - Short for “manual page”
  - Can also use the --help option
Shell Interaction Basics

1. Open the terminal application on your local computer
2. Connect to Klaatu Linux server with
3. `ssh <username>@klaatu.cs.washington.edu`
4. Enter in your password, you will not see characters as you type

Basic Interactions:

- You can use copy and paste with your usual short cuts
- You can navigate through your executed commands by using the up and down arrows
  - Convenient way to rerun commands or to fix small errors in previous command
- The history command will print the commands you’ve used this session to the terminal
Running Programs

▪ You can run a program by typing its path into the terminal

▪ Some folders are globally visible, so you only need the program’s name
  - /bin/ is globally visible because it is in the PATH shell variable

▪ To run a program in the current directory you need to give the path
  - ./local_program
  - Running local_program by itself will not work because it’s not globally visible

▪ All commands are bash files that are executed when you hit “enter” on a terminal line
  - You can write and execute your own! More on that later
Processes in the Shell

- Programs running in the shell are called “processes”
  - We refer to the code/instructions as the “program”
  - We can run a given program many times, creating many processes
  - Terminal can only run one process in the foreground at a time
    - Use the “&” special character to launch a process in the background
      - EX: emacs &

- Bash Shell has many built in programs
  - Commands like cd and ls

- Processes have Input and Outputs
  - Inputs come in two main forms: arguments and stdin
    - Arguments are strings separated by spaces given after the command
      - EX: cp my/src dest/folder
        - Arguments: “my/src” and “dest/folder”
        - Arguments with spaces need to be wrapped in quotes EX: echo “hello world”
    - Stdin or Standard Input is a stream that the user enters into the terminal
  - Outputs can be stdout, stderr or a directed to an output file
  - All redirections & string expansions or substitutions are done by the shell before the command
## Useful Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Operation</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ls</code></td>
<td>See folder contents</td>
<td><code>ls -l</code></td>
</tr>
<tr>
<td><code>cd &lt;folderName&gt;</code></td>
<td>Move into given folder</td>
<td><code>cd cs Downloads</code></td>
</tr>
<tr>
<td><code>cp &lt;source&gt; &lt;destination&gt;</code></td>
<td>Make a copy of given file in given destination</td>
<td><code>cp file.txt myDir/</code></td>
</tr>
<tr>
<td><code>mv &lt;oldName&gt; &lt;newname&gt;</code></td>
<td>Rename or move given existing file to given name/destination</td>
<td><code>mv fil.txt file.txt</code></td>
</tr>
<tr>
<td><code>cat &lt;fileName&gt;</code></td>
<td>Print file contents to terminal window</td>
<td><code>cat file.txt</code></td>
</tr>
<tr>
<td><code>touch &lt;filename&gt;</code></td>
<td>Create empty file with given name</td>
<td><code>touch file.txt</code></td>
</tr>
<tr>
<td><code>echo &lt;string&gt;</code></td>
<td>Print given string to terminal window</td>
<td><code>echo “hello world”</code></td>
</tr>
<tr>
<td><code>pwd</code></td>
<td>Print working directory</td>
<td><code>pwd</code></td>
</tr>
<tr>
<td><code>mkdir &lt;directoryName&gt;</code></td>
<td>Create an empty directory at location specified</td>
<td><code>mkdir ~/newDir</code></td>
</tr>
<tr>
<td><code>exit</code></td>
<td>Exit the shell</td>
<td><code>exit</code></td>
</tr>
</tbody>
</table>
Other Useful Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Operation</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>pico &lt;fileName&gt;</td>
<td>Create or edit files</td>
<td>pico filename</td>
</tr>
<tr>
<td>echo &lt;text&gt;</td>
<td>Print text</td>
<td>echo hello world</td>
</tr>
<tr>
<td>pwd</td>
<td>Print working directory’s absolute path</td>
<td>pwd</td>
</tr>
<tr>
<td>touch &lt;filename&gt;</td>
<td>Create empty file</td>
<td>touch filename</td>
</tr>
<tr>
<td>mkdir</td>
<td>Create empty directory</td>
<td>mkdir</td>
</tr>
<tr>
<td>find -name &lt;filename&gt;</td>
<td>Search for file</td>
<td></td>
</tr>
<tr>
<td>exit</td>
<td>Exit the shell</td>
<td></td>
</tr>
</tbody>
</table>
Linux Demo

Recorded Demo from 374 Sp 20 Instructor Megan Hazen
Files

- A collection of data used for long term storage
  - Stored on a hard drive
  - Hard drive is the physical portion of a computer that stores large amounts of data sits outside the CPU

- Files have...
  - Name
    - Unique string within the folder
  - Type
    - Indicated by the extension at the end of the name
  - Content
    - Data contained within the file
  - Location
    - Folder trail from drive to name
    - "breadcrumb"

<table>
<thead>
<tr>
<th>Name</th>
<th>Date Modified</th>
<th>Size</th>
<th>Kind</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lecture1-Intro.pptx</td>
<td>10/7/20</td>
<td>4.9 MB</td>
<td>PowerP...(pptx)</td>
</tr>
<tr>
<td>Lecture2-Shell.pptx</td>
<td>11:49 PM</td>
<td>1.5 MB</td>
<td>PowerP...(pptx)</td>
</tr>
<tr>
<td>Lecture3-Shell2.pptx</td>
<td>10/7/20</td>
<td>2.4 MB</td>
<td>PowerP...(pptx)</td>
</tr>
<tr>
<td>Lecture4-grep.pptx</td>
<td>10/7/20</td>
<td>989 KB</td>
<td>PowerP...(pptx)</td>
</tr>
<tr>
<td>Lecture5-...ping.pptx</td>
<td>Yesterday</td>
<td>959 KB</td>
<td>PowerP...(pptx)</td>
</tr>
<tr>
<td>Lecture6-Regex.pptx</td>
<td>Yesterday</td>
<td>954 KB</td>
<td>PowerP...(pptx)</td>
</tr>
<tr>
<td>Lecture7-...oToC.pptx</td>
<td>11:13 PM</td>
<td>1.9 MB</td>
<td>PowerP...(pptx)</td>
</tr>
<tr>
<td>Lecture8-...nters.pptx</td>
<td>11:15 PM</td>
<td>2.5 MB</td>
<td>PowerP...(pptx)</td>
</tr>
<tr>
<td>Lecture9-malloc.pptx</td>
<td>11:17 PM</td>
<td>858 KB</td>
<td>PowerP...(pptx)</td>
</tr>
</tbody>
</table>

Finder GUI view of folder

ls –l view of folder
Linux File Permissions

Permission Groups
- **u** – Owner
- **g** – Group
- **o** – Others
- **a** – All users

Permission Types
- **r** – read – a user’s ability to read the contents of the file.
- **w** – write – a user’s capability to write or modify a file or directory.
- **x** – execute – a user’s capability to execute a file or view the contents of a directory.

reading **ls -l**
- `rw_rw_rw` = owner, group and all users have read & write permissions
- first character is either a - or a d : d means “directory”, “-” means file

**chmod** <group>+|-<permission> <file>
- **chmod a-rw file1**: remove read and write permissions on file1 for all users
- **chmod a+rw file1**: add read and write permissions on file1 for all users

https://www.linux.com/training-tutorials/understanding-linux-file-permissions/
File System

- Files contain other files, branching out from the root “/” forming a tree-like hierarchy
- Files are located with a path of folders separated by “/” this is called the “file path”
- Paths starting with “/” are called absolute paths
  - Start searching from the root of the file system
  - EX: /usr/documents/myFiles/myFile.txt
- Paths that do NOT start with “/” are called relative paths
  - Starts searching from current directory
  - EX: myFiles/myFile.txt
- `pwd` command will print the current directory

https://homepages.uc.edu/~thomam/Intro_Unix_Text/File_System.html
Demo: File Manipulation
Questions?

Lecture Participation Poll #2
Log onto pollev.com/cse374
Or
Text CSE374 to 22333