COMP 2400

Prof. Chris GauthierDickey

Week 1: Beginning Unix

UNIVERSITY OF DENVER

What is Unix?

- An operating system developed in 1969 at Bell Labs
- A collection of tools (decades of tools, really) designed to be as interoperable as possible
- One of the few operating systems not designed to be commercial

UNIVERSITY OF DENVER

What to expect

- In this class, we'll learn a variety of topics, mostly based around Unix
  - filesystem, vi, svn, regexes, bash, makefiles, compilation, REXX
  - unix utilities
    - grep, find, cut, sort, sed
- This class will differ slightly from the past in that we'll use REXX

UNIVERSITY OF DENVER

Original Unix

- Developed in 1969 at Bell Labs
    - Ken Thompson, Dennis Ritchie, Douglas McIlroy and Joe Ossanna
- Developed at the same time as C
- During the late 70s and early 80s it became widely adopted at universities
    - BSD (Berkeley)
    - Solaris, HP-UX, and AIX (IBM)

UNIVERSITY OF DENVER

More about Unix

- Unix has a few great ideas, to say the least:
  - multi-tasking and multi-user out of the box (nice they figured this out way back then)
  - much of the configuration was stored in text
  - hierarchical file system--a tree!
  - Files! Everything (almost) looks like a file and can be read from and written to

UNIVERSITY OF DENVER

- Inter-process communication: processes can talk to each other through pipes
- lots of software tools that can be strung together: sort, grep, more...
- jobs! Running multiple things--stuff in the background

UNIVERSITY OF DENVER

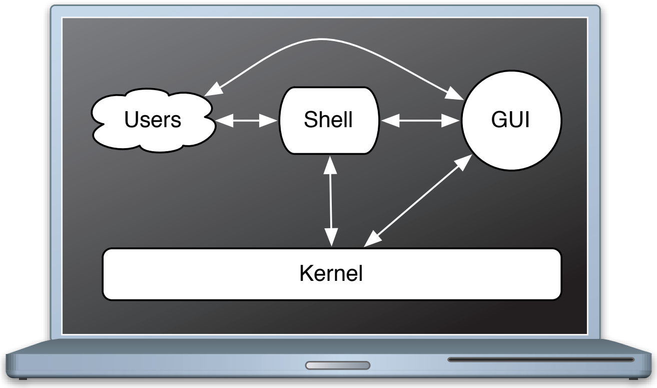
Interacting with Unix

- You can interact with Unix in a variety of ways:
  - The shell, which is a command interpreter
    - bash, csh, rexx
  - An interactive command, running inside of a shell
    - vi, emacs, ftp, ssh
  - A GUI, like we all know and love
    - X-Windows, Gnome, KDE, Aqua

UNIVERSITY OF DENVER

The Unix OS

- The kernel is the core part of the operating system
- We interact with a shell or GUI that talks to the kernel
- Programs interact with each other and the kernel



UNIVERSITY OF DENVER

It's a big ocean with a lot of shells

- Over the years, a variety of shells have been created
  - The original shell, by Steve Bourne, was called 'sh'
  - The c-shell, or csh, came out of Berkley, had more advanced job control, and was better for interactive use
    - Then there's zsh, a hybrid of sh and csh
    - tcsh, a more modern csh, but with some bugs

UNIVERSITY OF DENVER

- The Korn shell (David Korn from Bell Labs) was created in the early 80s.
  - Backwards compatible with sh and included features of the csh such as command history
  - Was designed to be a programming language
    - had associate arrays, floating-point arithmetic
- bash, the Bourne-again shell (1987), from the FSF which many modern Unixes use (and we'll use on Linux)
  - Has the best of csh and bash and ksh

UNIVERSITY OF DENVER

Files

- Files are broadly classified into 3 types
  - Regular files: typically data files of some sort
  - Executable files: files with the 'execute' bit set which can be run by the shell
  - Directories: Yes, these look like files too, but are folders
- The Unix filesystem is organized as a tree since directories can contain other directories

UNIVERSITY OF DENVER

Unix Filesystem

- The top level of the Unix file system, called the root, is referred to by the character '/'.
  - It doesn't have a name really
  - Every file can be located and referred to by starting at the root
- The root file system has a set of directories required to live in it given by the File System Hierarchy Standard

- bin: essential command binaries
- boot: static files of the boot loader
- dev: device files
- etc: host-specific configuration files
- lib: essential shared libraries and kernel modules
- media: mount point for removable media
- mnt: mount point for mounting a temp. file system
- opt: add-on application software packages
- sbin: essential system binaries
- srv: data for services provided by the system
- tmp: temporary files
- usr: secondary hierarchy
- var: variable data (changes without admin input)

UNIVERSITY OF DENVER

More root directories!

- home is a symbolic link to the home directories (often under /usr/home)
- root is the home directory for the super-user
- lib<qualifier> is a special library directory of different formats, where <qualifier> is the library type

UNIVERSITY OF DENVER

Files and Directories

- Files and directories are separated with the forward-slash '/'
- Your home directory lives under /home/<username>, such as /home/bob/
  - You can refer to your home directory by ~ (tilde)
- You can find out who you are and where your home directory is by asking:
  - whoami: tells the shell to print your username
  - pwd: tells the shell to print your current directory

UNIVERSITY OF DENVER

More on files

- Files can be given their full pathname when you're accessing them--all the way from the root
- You can also access files from a relative location
  - You always have a 'working directory', and file access is relative to it
  - If you begin with '/' it's the full pathname
  - Using ../ you can back up the hierarchy one level
    - sort ../filename

UNIVERSITY OF DENVER

The Working Directory

- As noted, pwd will tell the shell to print your working directory
- cd will change directories to a new one
  - cd can take arguments absolute or relative paths as arguments
    - cd without arguments goes to your home directory
    - cd .. goes up one level, cd ../.. goes up two levels, etc
    - cd ~/ will refer to files starting at your home directory

UNIVERSITY OF DENVER

Which shell am I using?

- By default, the bash prompt has a $ while csh has a %, though the super-user usually has a # at the end
    - Note: the prompt is easily changed
    - You can determine your shell by typing: echo $SHELL
- You can change your shell by running chsh...sometimes, it depends on the system.
    - Alternately, you can make whatever shell you start with spawn a process that runs the shell of your choice

UNIVERSITY OF DENVER

Shell commands

- Shell commands consist of words separated by space
  - Arguments follow the command
  - Quotes can group words with space into a single argument
  - Double quotes, "", will interpret variables
    - try echo "$PATH" and echo '$PATH' to see the difference
    - Usually, you want to use double quotes

UNIVERSITY OF DENVER

How are they used?

- Unix is a collection of tools!
    - One way it does this is through the definitions of standard input, output, and error
    - All programs will use standard input for input by default, and print to the screen for output and errors
    - These streams can be changed and redirected to other programs!

UNIVERSITY OF DENVER

Input/Output

- An important Unix concept to understand are the streams used for input and output
  - What's a stream? Just sequential data coming or going
  - Unix has 3 streams that are standard:
    - standard Input, standard output, standard error

UNIVERSITY OF DENVER

Why Standard I/O?

- Why is this useful?
  - By itself, taking input from standard i/o is somewhat useful
  - Used with redirection it's quite powerfull

UNIVERSITY OF DENVER

Standard Streams

- Standard input is your keyboard or the default device used for input on your system
  - It's not the mouse!
- Standard output is the screen, or terminal window
- Standard error is also to the screen

UNIVERSITY OF DENVER

Your First Unix Utilities

- cat: copy input to output
- grep: search for strings in the input
- sort: sorts lines of the input
- cut: extracts columns from the input
- sed: performs regular expression searching and replacing on input
- tr: translates characters in the input to other characters

UNIVERSITY OF DENVER

Must love cats

- Well, cat is useful, who doesn't want to copy input to output?!
  - But all these utilities will take input from standard input if you don't give them an argument
- Try it with no arguments
  - You'll need to hit CTRL-D to exit, but type a line, hit return and then observe what cat does

UNIVERSITY OF DENVER

The ls command

- ls will list your directory
  - By default, it only lists the file names
- We can give ls arguments to get more info:
  - For example, ls -l, or ls -la
    - -l gives us the long format, -a lists all the files, including the hidden ones (those that begin with a .): you can combine them into -la

UNIVERSITY OF DENVER

The long listing with -l

- If we do ls -l, what do we see?
  - drwxr_xr_x 5 bob student 186 Sept 6 21:30 mydir
- The 'd' stands for directory, if it's a '_' it is a regular file
- The rwx come in triples, the first is permissions for the user, then the group, then everyone else
  - r means the file can be read, w means it can be written to, x means it can be executed (or you can use cd if it's a directory)

UNIVERSITY OF DENVER

- In our example, it means that mydir can be read, written to, and executed by the user, it can be read and executed by the group, and it can be read and executed by everyone else

- A _ means it *does not* have that permission, so in our example, the group cannot write to that directory

    - ie, they can't add files to it

UNIVERSITY OF DENVER

More on ls -l

- The next value tells you how many files and directories live beneath the file--so a file always has 1, directories have more than 1
- The next value is the user who owns the file
- The next value is the group the file belongs to
- The next value is the size of the file in blocks
    - You can see it in bytes by passing -h to ls

UNIVERSITY OF DENVER

More on ls -l (cont.)

- The next value is the date the file was created
- Finally, the last value is the file name

- Now, try ls -la
    - You will see the 'hidden' files, or those that begin with a .
        - You'll also see the files . and .. which are links to this directory and the one higher up

UNIVERSITY OF DENVER

Back to cats

- Note that cat will take an argument which is a file
  - Try to run cat filename, where filename is something in your directory
  - Try to cat a directory--what happens?

Man and More

- Like all operating systems, Unix has built-in help
  - We call it the manual pages
  - You can access them through the command 'man'
- For example, try 'man cat' and it will give you all the things you can do with cat
  - You can concatenate multiple files, for example, into a single output!

UNIVERSITY OF DENVER

More or Less

- more is another Unix tool that allows you to paginate through text
  - It will print as much text from a file that fits on your screen and then prompt you to continue
  - more only goes forward
- less is more than more (yes, purposely)
  - You can go backwards and forwards through files
  - You can search files and open multiple files

UNIVERSITY OF DENVER

Your first Unix trick

- Now that you know a few commands, let's have some fun
  - By default, your shell is probably tcsh but we want to use bash
    - Make sure you have bash: do a 'which bash'
      - It should print out the path to bash
  - tcsh will run .cshrc from your home directory when you first log in--we want it to run bash instead
- echo "exec \`which bash\`\n" > ~/.cshrc

UNIVERSITY OF DENVER

What does this do?

- First, echo will just print out what you type to standard output
  - "exec `which bash`\n" is the argument to echo
  - The "" will cause things to be interpreted inside of them
  - The ` is a back-tick, and will execute a command and the output will be placed where the ` live
    - `which bash` will be replaced with the path to where bash lives on the system
    - \n is the end-of-line marker on Unix

UNIVERSITY OF DENVER

Continued explanation

- Exec will execute the given command and terminate the current one
- Thus, in this case, we run bash but terminate tcsh, so we're left in bash
- Finally, the > will redirect the standard output from the screen to a file specified after >
    - In this case, it's the startup file for tcsh: .cshrc and the ~/ tells it to look in your home directory
    - Run cat ~/.cshrc and you should see the command echoed commands there

UNIVERSITY OF DENVER