LECTURE 15: SHELL COMMANDS

HOW TO CONNECT

- Mac/Linux:
  - No need to connect, just use your own command line(Terminal)
  - ssh –p 130 username@server
  - Enter password when prompted
- Windows:
  - Command Prompt
    - Not identical
  - We provide access to our UNIX terminal
    - Connect via an SSH client, such as PuTTY
    - https://www.putty.org/

SHELL COMMANDS BY EXAMPLES

- What's my default shell?
  - echo $SHELL
- We start with bash, change later if you like others
- Let's issue some commands
  - pwd
  - ls
  - less datafile.txt
  - mkdir CSE337
  - vim/vi notes1
  - cp notes1 tasks
  - mv notes1 tasks2

WHAT DOES THESE COMMANDS DO?

- pwd: show current directory I am in
- ls: list content of my current directory
- less: see content of file: “datafile.txt”
- mkdir: create a directory called
- cd: go into a directory
- vi create/edit a file
- cp copy a file to another file

SOME OTHER SHELL COMMANDS

- rm: remove file(s) (DANGEROUS with –R flag)
- man rm: display manual page
- scp: securely copy files
- mutt: read emails
- cal: calendar
- wc: word, character, line counts
- gzip: gnu zip
- gunzip: gnu unzip
- yppasswd to change password

COMMON UNIX COMMANDS IN MORE DETAIL

OUTLINE

- The ls command
  - UNIX Access permissions
- The chmod command
- The cd command
- mkdir and rmdir
- rm, more and less
- cp and mv

THE LS COMMAND

- ls lists the contents of a directory
  - ls <directory_name>
  - Works with multiple directories too:
    - ls <dir_name1> <dir_name2> <dir_name3>
- Command-line options
  - ls accepts a number of options that control how the output is presented to the user
  - -a option lists hidden files or directories whose names start with a . (dot)

LS COMMAND EXAMPLES

- ls testdir1/
- ls .
- ls
- ls ..
- ls ~
- ls /etc
- ls ~aaydin
- ls –l

LS -L OUTPUT

- The output is organized in columns. They are:
  - The file access permissions
  - Number of links to the file
  - Owner of the file
  - Group for the file
  - File size
  - Time of the last modification
  - The name of the file

UNIX FILESYSTEM ACCESS PERMISSIONS

- Every file has an owner and an owner group
- You own new objects that you create
- The owner can set the permissions of a file
- An owner group allows a file to be shared among members of the same project
  - The file owner specifies what the members of the group can do with the file

THE ROOT ACCOUNT

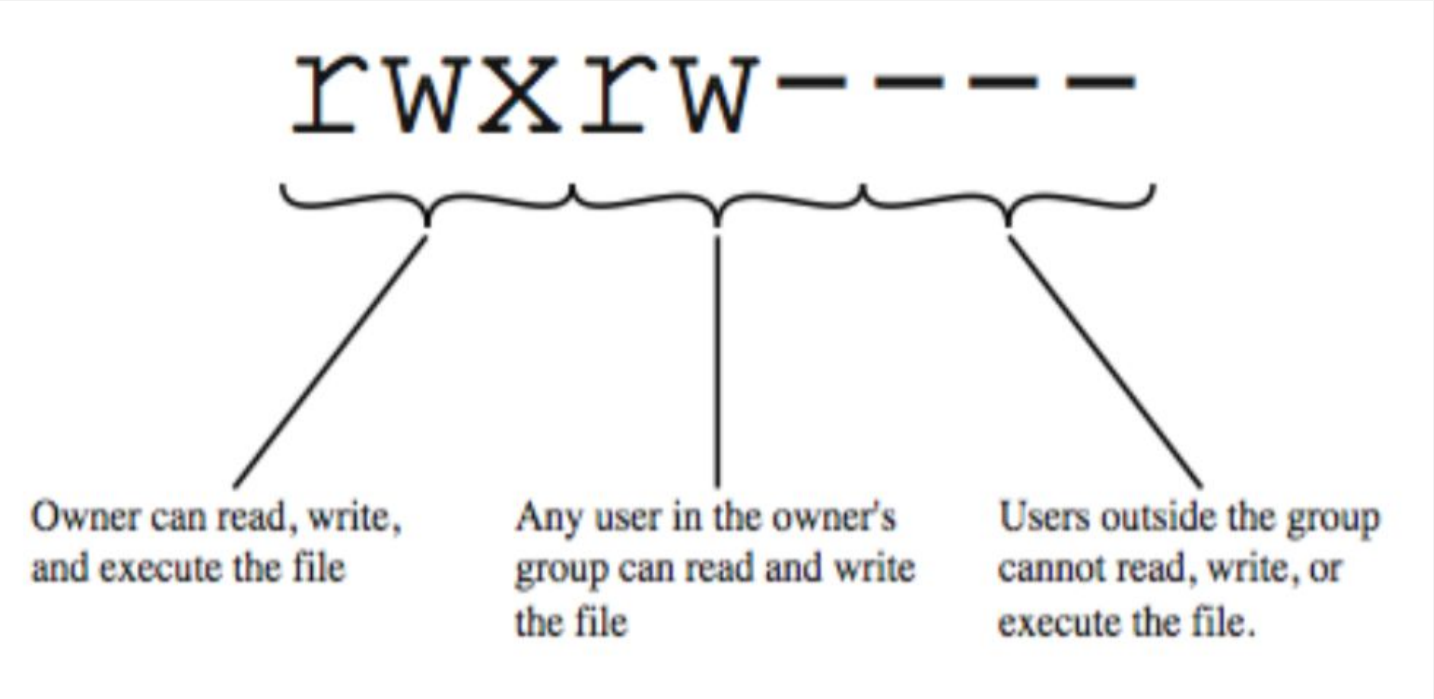
- UNIX's omnipotent administrative user(UID 0)
- Also known as superuser account, but actual username is root
- Traditional UNIX allows for superuser to any valid operation on any file or process
- Example root-only processes:
  - Setting hostname
  - Setting system clock
  - Shutting down system
  - Creating device files

DETERMINING FILE OWNERSHIPS

```
my-mac:lecture11 ali$ ls -l groups.pl
-rwxr-xr-x 1 ali  staff  579 Feb 28 19:22 groups.pl
```

- the groups.pl file is owned by user "ali", and group staff
- Letters and dashes in the first column symbolizes file permissions
- The first character is usually a dash(“-”) or “d”
  - dash means it’s a regular file
  - d means it’s a directory
- This is followed by 9 permission bits, for 3 sets of permissions

PERMISSION BITS



- 3 sets of permission
  - owner, group owner, and others

PERMISSION BITS

- Permissions for a file
  - Read: allow file open and read
  - Write: allow file content modification/truncation
  - Execute: allow file to be executed
- Permission for a directory
  - Read: allow content listing
  - Write: allow file creation, deletion, renaming
  - Execute: allow to enter the directory but not listing

THE CHMOD COMMAND

- Used to change the permissions on a file
  - chmod <who>=<mode> <file>
  - <who>: u, g, o (owner, group and other, respectively)
  - <mode>: new set of permissions, can be any combination of “r”(read), “w”(write) and ”x”(execution)
  - <file> is the name of the file or directory for which the permissions are being set
- Only owner of the file and superuser can run it

CHMOD EXAMPLES

- chmod g=rx lab1.cpp
- chmod u=rwx lab1.cpp
- chmod ug=rw lab1.cpp
- chmod o=rx lab1.cpp

NUMERICAL CODES

- chmod 777 lab1.cpp
- chmod 444 lab1.cpp
- chmod 464 lab1.cpp
- chmod 554 lab1.cpp

MORE CHMOD FORMS

- Permissions can also be added/ removed using chmod <who>+<mode> <file> or chmod <who>-<mode> <file>
- chmod o-w lab1.cpp removes write permission for others
- chmod u+w todo adds write for the owner of the file todo
- a-x removes execute for all categories
- g=u makes the group permissions the same as owner

THE CD COMMAND

- The cd command changes the current working directory to the specified directory. Its syntax:
  cd <directory_name>
- cd testdir1 (Change to the sub-directory testdir1)
- cd .. (Change to the parent of the current directory)
- cd ~ (Change to the user's home directory)
- cd /etc (Change to the directory /etc (absolute pathname))
- cd ~aaydin (Change to the user aaydin's home directory)

MKDIR AND RMDIR

- mkdir: used to create new directories
  mkdir <new_directory>
  - Example: mkdir testdir1 creates a new directory called testdir1 as a subdirectory in the current working directory
- rmdir: used to delete existing directories
  rmdir <directory_name>
  - The directory being deleted must be empty
  - Example: rmdir deletes the subdirectory testdir1

RM, MORE AND LESS

- rm: used to delete files. Its syntax: rm <file>
- more: used to view contents of files. Its syntax: more <file>
  - Displays content screen by screen. Enter to advance by one line, space to advance a full screen, q to quit
- less: =more, except it allows to go backwards using “b”. Its syntax: less <file>
  - Example: more readme.txt or less readme.txt

THE CP COMMAND

- Used to copy files: make a copy of src_file
- Common form: cp <src_file> <dest_file>
  - If dest file exists, it will be overwritten. If it does not exist, it is created. src_file not changed
- Another form: cp <src_file> <dest_dir>

FIN!