EASTERN WASHINGTON UNIVERSITY
start something big

Unix File System Command 3

Computer Science Department
Eastern Washington University
Yun Tian (Tony) Ph.D.

Recall Last Class

- ls –al list all directory contents in details
- cd .. change to parent directory
- pwd print working directory
- rmdir remove directory only if empty
- mkdir create new folder

Today

- cp, mv, rm, alias, clear, touch
- Deep understanding for User and Groups

Command cp

- **cp sourceFile(s) destination**
  - Make copies of files or entire directory.
  - where sourceFile(s) and destination specify the source and destination of the copy respectively
  - The behavior of cp depends on whether the destination is a file or a directory.

Command cp

- **cp sourceFile destination**
  - If the destination is a file, only one source file is allowed,
  - And cp makes a new file called destination that has the same contents as the source file.
  - E.g. cp program.c ../bakup/myprog.c_bak

Command cp

- **cp sourceFile(s) destination**
  - If the destination is a directory,
    - Many source files can be specified separated with space,
    - Each of which will be copied into the destination directory.
  - E.g.
    - **cp file1 file2 /tmp/data/file3 /home/ytian/files**

Command cp

- **cp -r source-directories destination-directory**
  - Copy entire directory (‘–r’ means copy recursively)
  - Including all files in the source folder and everything in its subdirectories.

Command cp

- More Examples
  - cp *.mp3 ~/Music/
  - Copies all .mp3 files from the current directory to /home/<username>/Music/
  - You can use wildcard(*) to copy multiple files.
    - '*' means zero or more of any characters in that spot.
  - We will talk more about wildcard later.

Command mv

- **mv [opts] <source> <destination>**
  - **Move** file or directory to another place
  - Or rename a file or directory
- Unlike cp, the move command is automatically recursive for directories.
  - Move all its subdirectories and files by default.

Command rm

- **rm [opts] <filename>**
  - **Removes(or delete)** the file called <filename>
- Using wildcards (more on this later) you can remove multiple files
- **rm *** removes all files in the current directory (cannot remove folders) .
- **rm *.jpg** removes every .jpg file in the current directory.
- **rm -i filename** prompt before deletion

Command rm

- Unlike in windows, once you delete a file from the command line, there is no easy way to recover the file.
- So be cautious when using **rm ***
- **Good practice is that,**
- Make an alias for your rm command by doing:

  alias rm='rm -i'

Command rm and alias

- Alias create a newAlias for your command name.
- **alias newAlias="regular-command“**
  - This create a temporary alias.
- E.g. alias rm=‘rm -i’
  - In this case, in current session each time you type rm, the shell actually executes ‘rm –i’ with the i option.
  - which asks you to confirm before delete anything.

Command rm and alias

- **alias newAlias="regular-command“**
  - This create a temporary alias.
- Another example
- alias dir=‘ls -al’
  - In this case, in current session each time you type dir, the shell actually executes ‘ls -al’ command.
    - dir is a popular DOS and windows command.

Command rm and rmdir

- **Let us go back to rm command**
- By default, rm cannot remove directories. Instead we use rmdir.
  - rmdir only removes empty directory.

Command rm and rmdir

- How to delete a directory and its contents?
  - To delete a directory and all its contents, we pass rm the option -r (which means recursive)

    rm -r /home/ytian/cscd240

  The command above delete folder cscd240 and all contents in it.
- **Very careful** when use this –r with wildcard *
  - Unix does not recover deleted files with a click.

Command clear and touch

- **clear**
  - Clear the terminal screen
- **touch [option] <file>**
  - Create an empty file
    - If the file does not exist, touch create it.
  - Adjust the timestamp of the specified file.
    - With no options uses the current date/time

File Extension in Unix

- File extensions (.exe, .txt, etc) often don't matter in UNIX.
- Using touch to create a file results in a blank plain-text file (so you don't need to add .txt to it).

Users and Groups

- Unix was designed to allow multiple people to use the same machine at the same time.
  - This raises some security issues.
  - How do we keep our coworkers from reading our email, deleting our programs and files?
    - Unix use permissions for different user categories on different files.

Users and Groups

- Access to files depends on the users account.
  - All accounts are presided over by the Superuser, or "root" account.
  - Each user has absolute control over any files he/she owns,
    - which can only be superseded by root.

Users and Groups

- Files can also be assigned to groups of users.
  - Allowing reading, modification and/or execution to be restricted to a subset of users.
  - E.g. students and teachers share a same server.
    - students cannot peek other students' homework.
    - But teacher may have the permission to access to students' homework.

Users and Groups

- Each file is assigned to a single user and a single group.
- Generally it takes root permission to change file ownership.
  - A regular user can't take ownership of someone else's files and can't pass ownership of their files to another user or a group they don't belong to.
- To see what groups you belong to use **groups**.

Permissions

- Advantage: normal users cannot change system files and cannot globally install programs.
  - Why is it advantage?
    - It restricts what malicious code can do.
- How do you change the permissions of your own files?
  - Next class uses chmod command.

Take Home Summary

- mv move file/directory to another place
  or rename a file/directory.
- rm deletes a **file (not a folder)**.
- alias creates a newName for a command.
- clear the terminal screen.
- touch creates a empty text file.
- More about User and groups

Next Class

- More File System commands
  - chmod
  - chgrp
  - chown