CS Essentials

Session 1: Linux Essentials

Oxford University
CompSoc

Administration

- Every Thursday 7-8:30pm, weeks 2-7
- Materials will be available online right before the session
- Linux Mint Cinnamon

Course outline

Session 1 (This one) : Linux Command Line & Permissions

Session 2 : Vim Text Editor

Sessions 3 & 4 : Bash scripting

Sessions 5 & 6 : $\LaTeX$

What is UNIX?

- the mother of the OS
- there are many UNIX-like systems, in that they offer similar interfaces
- laid the foundation for Linux
- the commands are mainly the same
- the main idea: Linux is an OS kernel designed like the UNIX kernel

Why Linux?

- free
- not made for profit
- a lot of desktop environments to choose from

What is a Linux Distribution?

Linux is used by programmers, different organizations around the world to createan OS to fit their requirements. Many of them have their Linux OS private to prevent hacking. However, some of the variations are free. A few examples:

- Ubuntu
- Fedora
- Debian
- Mint

Why Mint?

- stable
- designed for people with basic computer tasks
- designed for home PCs, can be used in an office too
- everything you need is already installed
- it is the transition to more powerful distros

Introduction to Mint

File manager represents the GUI - Graphical User Interface.
The terminal represents the CLI - Command Line Interface.
Why use the terminal instead of file manager?

- commands offer more options and are flexible
- faster loading
- does not consume RAM, compared to GUI

Open the terminal : Ctrl + Alt + T

Commands

1. whoami
Tells you what your username is.

2. pwd - print working directory
Shows you the path from the home directory to the file in which you currently are.

Commands

3.1 ls - list files
Lists files and directories in bare format (you won't be able to view details like file type, size, modified date and time etc.)

3.2 ls -l
Shows file or directory size, modified date and time, file or folder name and owner of file and its permissions.

3.3 ls -a
Shows all files again, but includes the ones that are hidden (the ones starting with a dot).

Commands

4.1. cd - change directory

Use this command to navigate through directories.

- To go through more files at once write cd name1/name2, which means go to the file name2 which is in name1, which is in the current directory.
- To go into a directory that has special characters in its name, you need to put a backslash before every special character.
- If you just type cd, it will return you to the home directory.

4.2 cd ..

Takes you into the previous directory.

4.3 cd -

Tells you the file you just changed from and also takes you back to it.

Commands

5. mkdir - make directory

Takes one argument, the name of the directory you want to create.

6. rmdir - remove directory

Also takes one argument, the file you want to delete.
Warning! It only works for empty directories, you cannot delete the directory and all the files in it using this command. This is why we have..

7. rm - remove

Removes only files and not directories.

- Note: you can use rm -r to remove a directory and all the files in it recursively.

Commands

8. touch

Creates other files than directories.

9. man & help

- man takes one argument (a command) and gives you the manual of that certain command
- help also takes one argument and tells you the way in which you can use a certain command
- Difference? Help is a bash command. It uses internal bash structures to store and retreive information. (works only for bash commands)
- use q to exit the help or man page

Commands

10. cp - copy

Takes two arguments, the file you want to copy and the directory you want to put it in.

11. mv - move

This command works exactly like cp, only it moves the file instead of copying it.

- Note: can be used to rename a file

Relative vs Absolute path

Main idea: you can navigate through files using either the relative or absolute path.

1. Absolute path
   - specify the location of a file starting from the root directory
   - how to write an absolute path-name :
     - start at the root directory (write /) and work down to the file you want
     - write a slash (/) after every file (except the last one)
2. Relative path
   - it is defined as the path related to the directory you are currently in
   - therefore, starts from the current directory and doesn't have a / at the beginning

Cheat sheet

- ls - list files (bare format)
- ls - l - list files (with size, modified date and time etc.)
- ls - a - list files (including hidden)
- cd - change directory
- cd .. - go to previous directory
- cd - - tells the file you came from and it also takes you back
- whoami - shows username
- pwd - print working directory
- mkdir - make directory
- rmdir - remove directory
- rm - remove
- rm -r - remove directories and all files in it
- touch - create files
- man & help & info
- cp - copy
- mv - move

The sudo and su commands

Sudo - super user do

- allows you to run commands with root privilages
- the root user is a user who has access to all commands and files
- asks for your password every time you use it (unless you use two sudo commands at an interval less than 15 minutes)

Su - switch user

- if you just type su it switches the current user to the root user
- asks for password once and runs all the commands you type after that with root privilages (don't recommend)

SSH

- SSH = Secure Shell
- the most common way to connect to a remote server
- provides a safe and secure way of executing commands, making changes remotely
- how to log in using SSH: type ssh username@remote_host (the username you have on that server)

Permissions

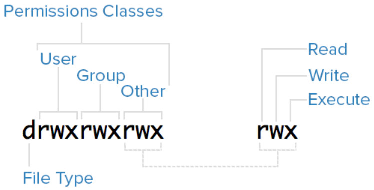
- the UNIX-like OS such as Linux are multi-user
- e.g if your computer is attached to a network or the Internet, remote users can log in via SSH and operate the computer
- because of that, a method to protect users from each other was devised: permissions
- each file and directory is assigned access rights for the owner of the file, the members of a group of related users and everybody else
- rights can be assigned to read, write and execute a file

Permissions

- recall ls -l : shows all details of the files from the directory we are in

  mode owner group file_size last_modified file_name

- the first part (the mode) is the important one



https://assets.digitalocean.com/articles/linux_basics/mode.png

Permissions

1. Read
   - for normal files, allows an user to view the contents of a file
   - for directories, allows an user to view the names of the files in the directory
2. Write
   - for normal files, allows an user to modify and delete a file
   - for directories, allows an user to delete the directory, modify its contents (create, delete and rename files in it) and modify the contents of files that the user can read

Permissions

3. Execute
    - for normal files, allows an user to execute a file (the user must also have read permission)
    - therefore, the execute permission must be set for executable programs

Changing permissions

1. chmod - change mode

Takes two arguments, the options and the file name.
The options you have are:

- u - owner (user)
- g - group
- o - other
- a - all (the same as if you write ugo)
- r - read
- w - write
- x - execute
- + - add permission
- - - remove permission
- = - set permission

Changing permissions

2. chown - change owner

Takes two arguments, the new owner and the file name.

3. chgrp - change group

Works in the same way, takes the new group and the file name.

Useful tricks

- use the top arrow to go back to what you previously wrote in the terminal; you can use it multiple times and it updates as you go
- use the bottom arrow if you went back and now you want to see what you wrote more recently
- use Tab to fill up the terminal
- use Ctrl + C to quit an ongoing process
- if Ctrl + C doesn't work, use Ctrl + Z to force quit it
- use the command clear if the terminal gets too filled up

Useful tricks

- use the command exit to exit the terminal
- used to using Ctrl + S? Press Ctrl + Q to unfreeze the terminal
- use Ctrl + A to move to the beginning of the line
- use Ctrl + E to move at the end of the line
- copy and paste are no longer as we have known them! Ctrl + Shift + C to copy and Ctrl + Shift + V to paste
- run more than one command at a time: command1;command2
- run more commands but only if the previous was successful: command1&&command2

Thank you!