CS4414 Recitation 3

Linux and Linux Commands

09/10/2021

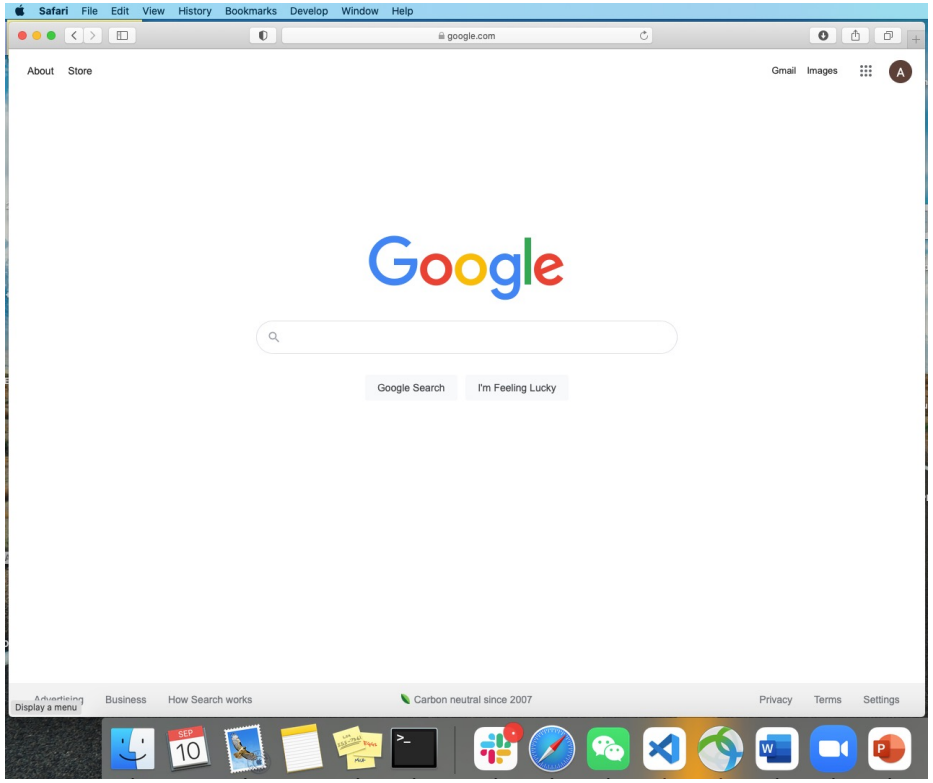
Alicia Yang

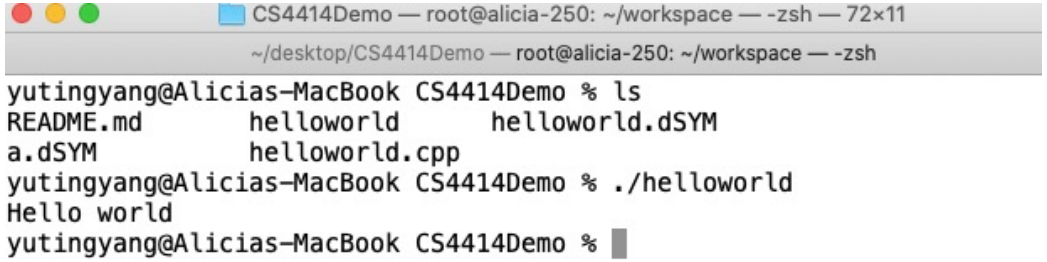
Operating System

What is Operating System?

Operating System

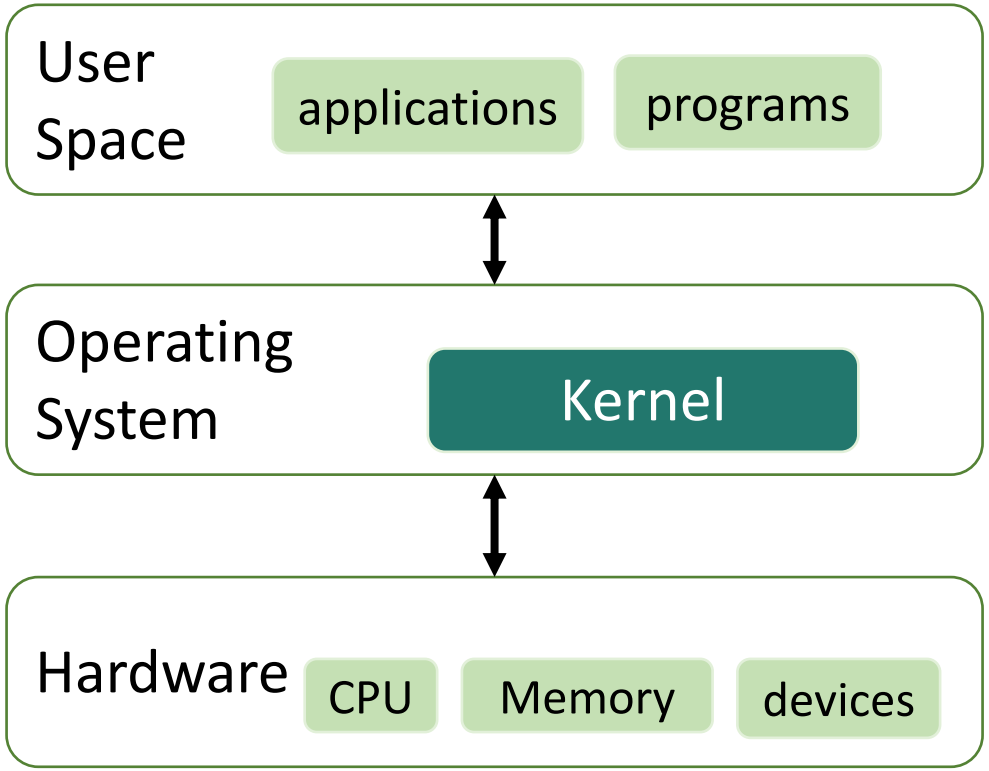
- A system software that **manages** computer hardware, software resources, and **provides common services** for computer programs
- Think of Operating System as an app for hardware.





Operating System

- How does the processor on your computer know that you are asking it to run a file? What is it that makes the computer hardware work like that?
- The operating system or the kernel does this work.



Linux System

- Linux(GNU/Linux) is an operating system, like MacOS, Windows.
- Linux is an **Unix-like operating system**
- Unlike other operating systems, Linux is **free and open source.**

  Programmers can use Linux kernel to design their own custom operating system

- Applications like Google server, Android use Linux

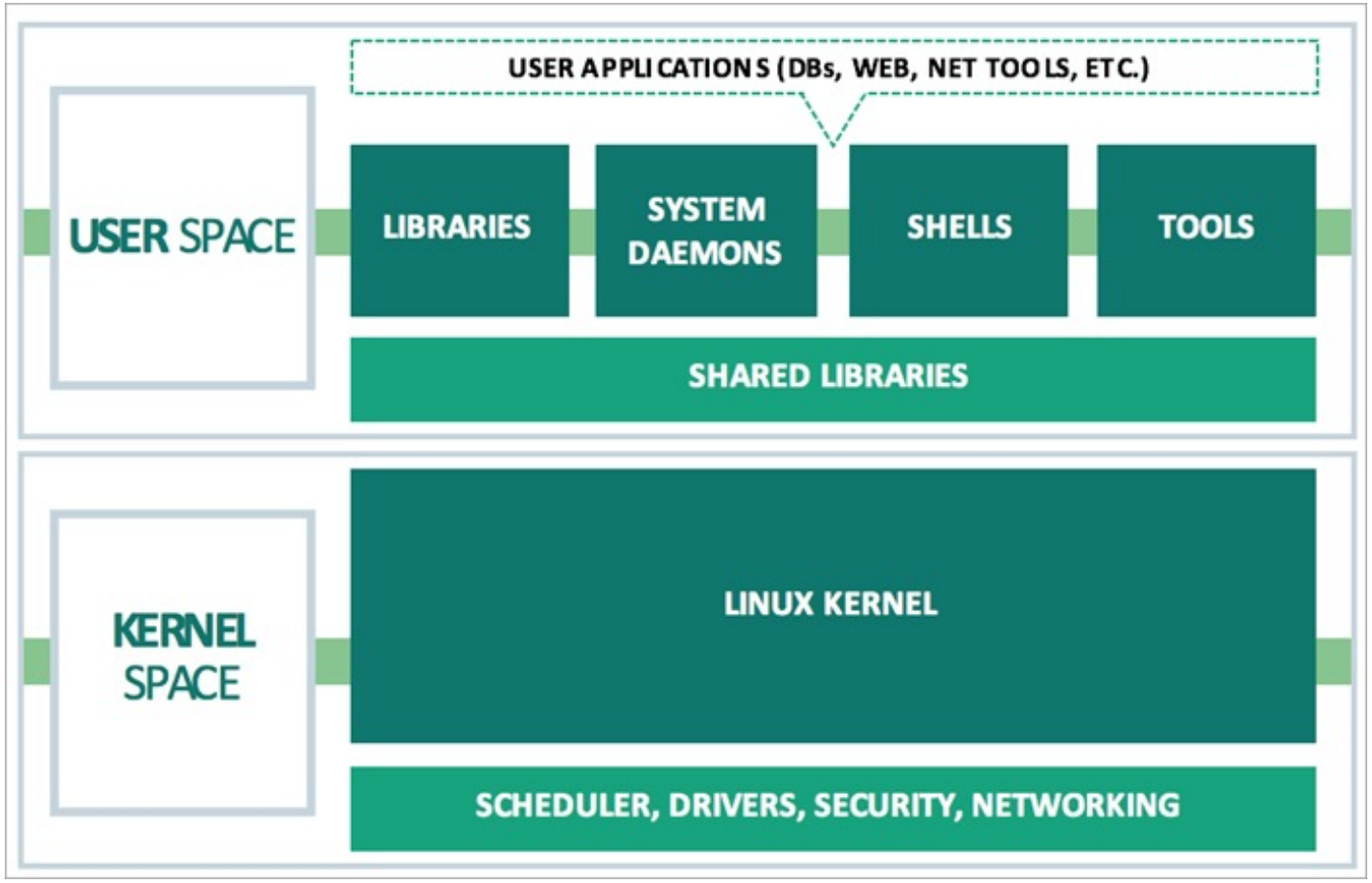
Linux Kernel

- Receives requests from user program

  Then relays the requests to computer hardware.

Linux Kernel

- Receiving requests from user program, and relaying the requests to computer hardware.

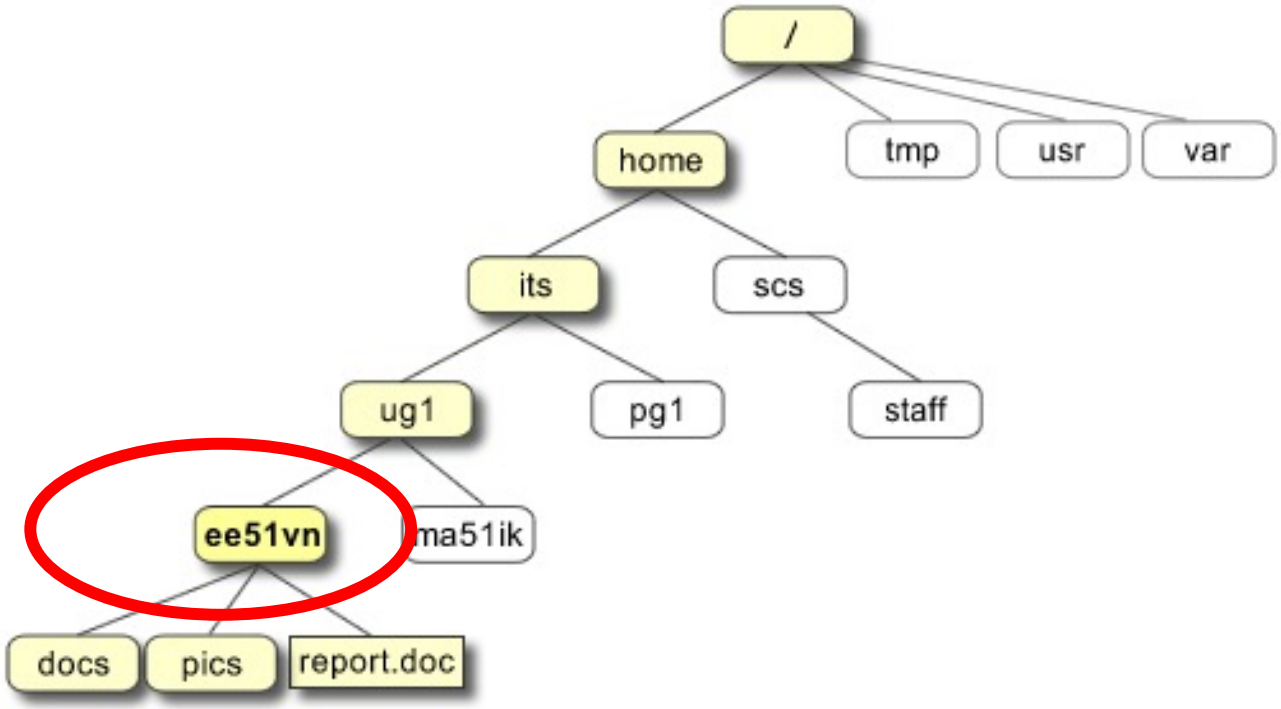


File Structure

- Absolute path:

  What is the absolute path of the folder ee51vn?

  /home/its/ug1/

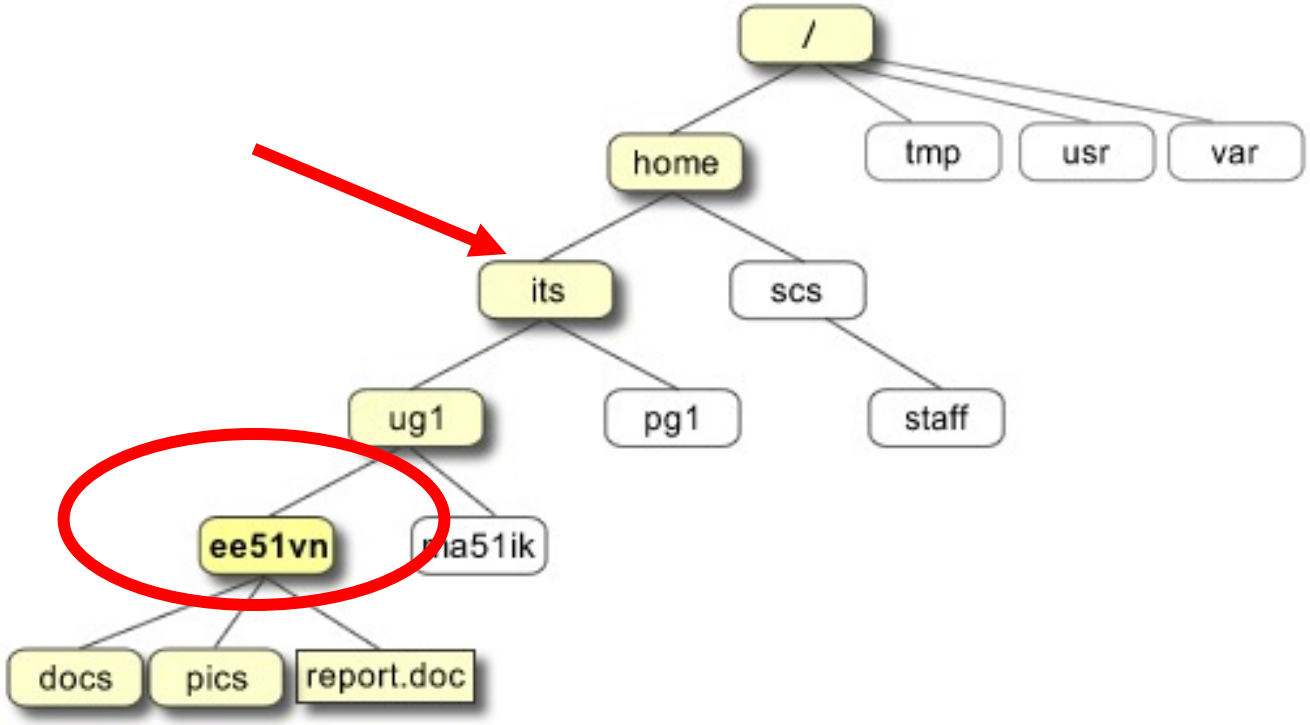


File Structure

- Relative path

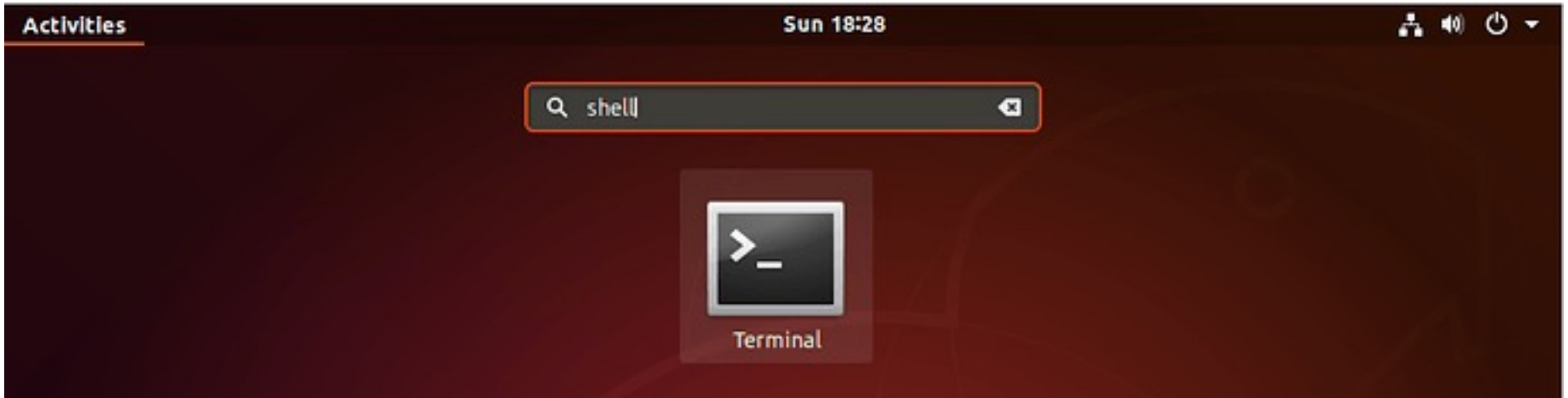
  Suppose current directory is at /home/its, what is the relative path of the folder ee51vn?

  ./ug1/



Linux Commands

Open a terminal, let's get started!



Basic Commands

- **echo $SHELL**
  - Within a terminal, there's a **shell**.
  - Shell is **a part of the operating system,** defines how the terminal behaves after a command.
  - Examples: bash, zsh (~/.bash_profile **set the environment** for shell, same for ~/.zsh_profile)
- **lsb_release –a** Display **Linux** distribution
- **free –g** Display how many **space freed/used**

Basic Commands

- **which g++** shows which compiler is running
- **uname** basic information about the operating system name and system hardware
  - uname –s print kernel name
  - uname –a print all information
  - …
- **man uname** ‘man’(manual) command like [help] can print details of cmd’s optional argument

Directory and Navigation

- **pwd** — get current directory
- **ls** — show what's in current directory
- **ls [directory]** — show what's in specific [directory]
- **ls -l** — '-' is the argument pass to the command, [-l] which indicates we are going to do a long listing
- **cd [directory]** — move to another directory
  - **cd /** — root directory
- **mkdir [directory]** — create a directory

Directory and Files

- **echo "This is a test”** ‘echo’ prints its arguments back out again
- **mv [file1] [directory1]** move file1 to directory1
- **rm [file1]** remove file1
- **rmdir [directory]** remove empty directory
- **rm –r [directory]** remove [directory] and all files in the [directory]

Output Redirection

- **echo "This is a test" > test_1.txt** ‘>’ redirect the content to the file
- **cat < test_1.txt** ‘<’ display the content in file
- **cat test_1.txt test_2.txt** ‘cat’ can concatenate/link the [file2] and [file1], then display
- **./helloworld > test_1.txt** write output from ‘helloworld’-program to file

Wildcard and alias

- **?** Wildcard: matches a single character.
- ***** wildcard: matches any character or set of characters
- Alias
  - **alias clean='rm -f *~'** Defile alias
  - **touch a~ b~ x~** Create some files with ~ ending
  - Type **clean** Clean the files with ~ ending

Permission

- **sudo** command for super user to execute (careful when using this cmd)
- **ls -l [file]** shows the permission of a [file]
- **chmod [who][+,-,=][permissions] filename** change the permissions
  - **chmod u-r filename** remove read permission from [file]
  - **chmod a+x filesname** add execute permission to [file]
  - **chmod 750 ~/example.txt** is equivalent to chmod u=rwx,g=rx,o= ~/example.txt

Permission Reference

https://en.wikipedia.org/wiki/Chmod

Reference	Class	Description
u	user	file owner
g	group	members of the file's group
o	others	users who are neither the file's owner nor members of the file's group
a	all	all three of the above, same as ugo

Operator	Description
+	adds the specified modes to the specified classes
-	removes the specified modes from the specified classes
=	the modes specified are to be made the exact modes for the specified classes

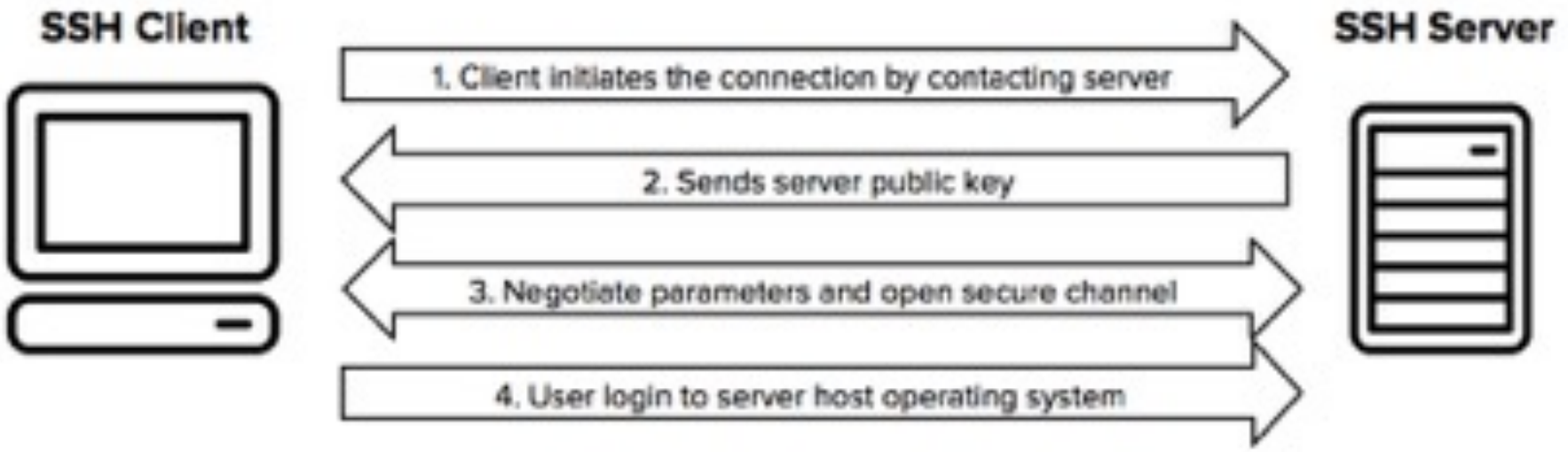
Mode	Name	Description
r	read	**r**ead a file or list a directory's contents
w	write	**w**rite to a file or directory
x	execute	e**x**ecute a file or recurse a directory tree

Processes

- **ps aux** — **S**how all the processes
  - **ps aux | grep** — Grep(search the output of input)
- **sleep 10** — Sleep for 10 secs
  - **sleep 10 &** — Turns the sleep process into background
- **Ctr+ c** — Send a signal to the process, which will terminate the process
- **ps** — show all the processes running

Connecting to remote servers

- **ssh sample.ssh.com**



Connecting to remote servers

- **ssh sample.ssh.com**
  - To log in to a remote computer called sample.ssh.com
- **scp [source_file_directory] [destination directory]**
  - **scp path/file host:path** : copy the file to the remote host
  - **scp host:path/file path** : fetch the file from the host, and puts it in the path
  - **scp -r localpath host:path** : copy the whole folder at localpath to host:path
  - More details of configure public key authentication:

    https://www.ssh.com/academy/ssh/scp

Compilation

Commpilation

--- g++ command and options

- -g - turn on debugging (so GDB gives more friendly output)
- -Wall - turns on most warnings
- -O or -O2 - turn on optimizations
- -o <name> - name of the output file
- -c - output an object file (.o)
- -I<include path> - specify an include directory
- -L<library path> - specify a lib directory
- -l<library> - link with library lib<library>.a

Writing Shell Script

Shell Script

- **cat ~/../../xxx.sh** Show the script in bash script
- **source xxx.sh** Run the script
- Editing Tools:
  - vi, emacs, nano
  - more details : https://linuxcommand.org/lc3_wss0010.php

CMake

CMake

- What is Cmake file used for?

Make is a tool to simplify building executable from different modules of a project. A **makefile** is a text file that is used or referenced by the 'make' command to build the targets.

- How to write make file?

Cmake

Default makefile

```
default:
        g++ main.cpp –o out
```

Generic makefile

```
target:  dependency1 dependency2 ... dependencyn
    <tab> command
```

Cmake --- makefile with variables

```
# the compiler: gcc for C program, define as g++ for C++
  CC = gcc

  # compiler flags:
  #  -g    - this flag adds debugging information to the executable file
  #  -Wall  - this flag is used to turn on most compiler warnings
  CFLAGS  = -g -Wall

  # The build target
  TARGET = myprogram

  all: $(TARGET)

  $(TARGET): $(TARGET).c
          $(CC) $(CFLAGS) -o $(TARGET) $(TARGET).c
```

Cmake

Run Makefile:

Type **make**

Run clean command:

Type **make clean**

Cmake

demo

- Run Makefile:

Type **make**

- Run clean command:

Type **make clean**

Code: https://github.com/aliciayuting/CS4414Demo.git

Resources : https://www.cosmiclearn.com/cplusplus/stdqueue.php

https://en.cppreference.com/w/

https://www.geeksforgeeks.org/queue-of-pairs-in-c-stl-with-examples/