Linux Architecture, Linux File System, Linux Basic Commands

Overview

- History of Linux
- Linux Architecture
- Linux File System
- Demonstration
- Linux Access
- Linux Commands
- File Permission
- Editors
- Package Management
- System Handling Commands
- Conclusion and Questions

History

- 1969
  - Dennis Ritchie and Ken Thompson developed the C language and the Unix operating system at AT&T Bell Labs
- Unix Features
  - Multiuser & Multitasking System
  - Hierarchical directory structure
  - Programming Facility
  - Documentation

History

1980s

- There were many Companies develop their own unix
- Richard Stallman started GNU project
- Many Commands we use today in Linux are GNU tools

1990s

- In 1991 Linus Torvalds writes a kernel and Share the code
- Combination of this kernel and GNU tools becomes Linux

Distribution

A Linux distribution is a collection of software on top of a Linux kernel. A distribution can bundle server software, system management documentation and many desktop applications

A distribution aims to provide a common look and feel, secure and easy software management and often a specific operational purpose

Popular Distribution

Red Hat

- RHEL (Commercial product with support)
- Fedora (Open source product)
- Ubuntu
- Debian

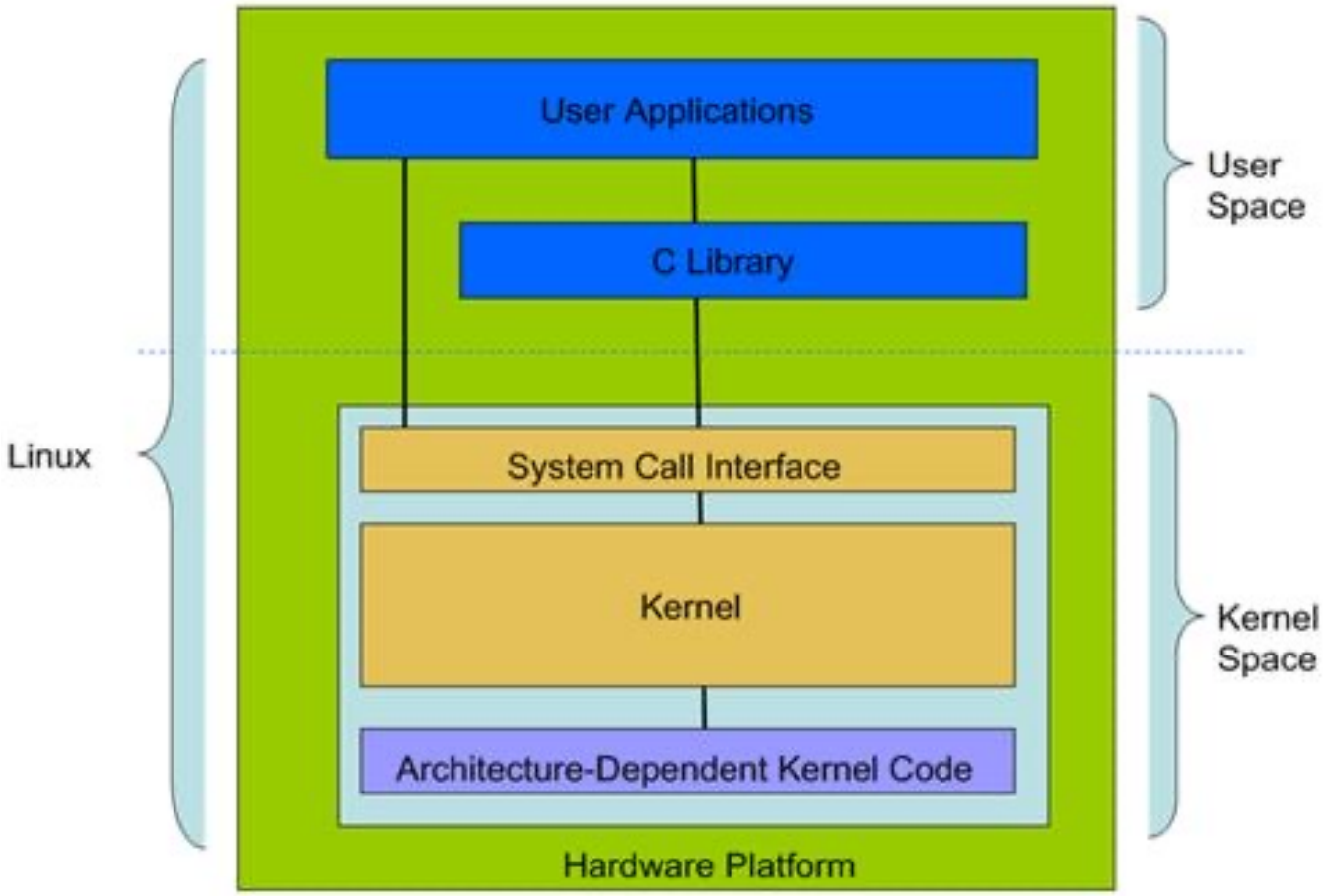
Other

- CentOS
- Linux Mint/ Linux Lite
- openSUSE
- Kali
- Slackware
- Solus

Linux System Architecture

Divided in to two levels

- User space
- Kernel space

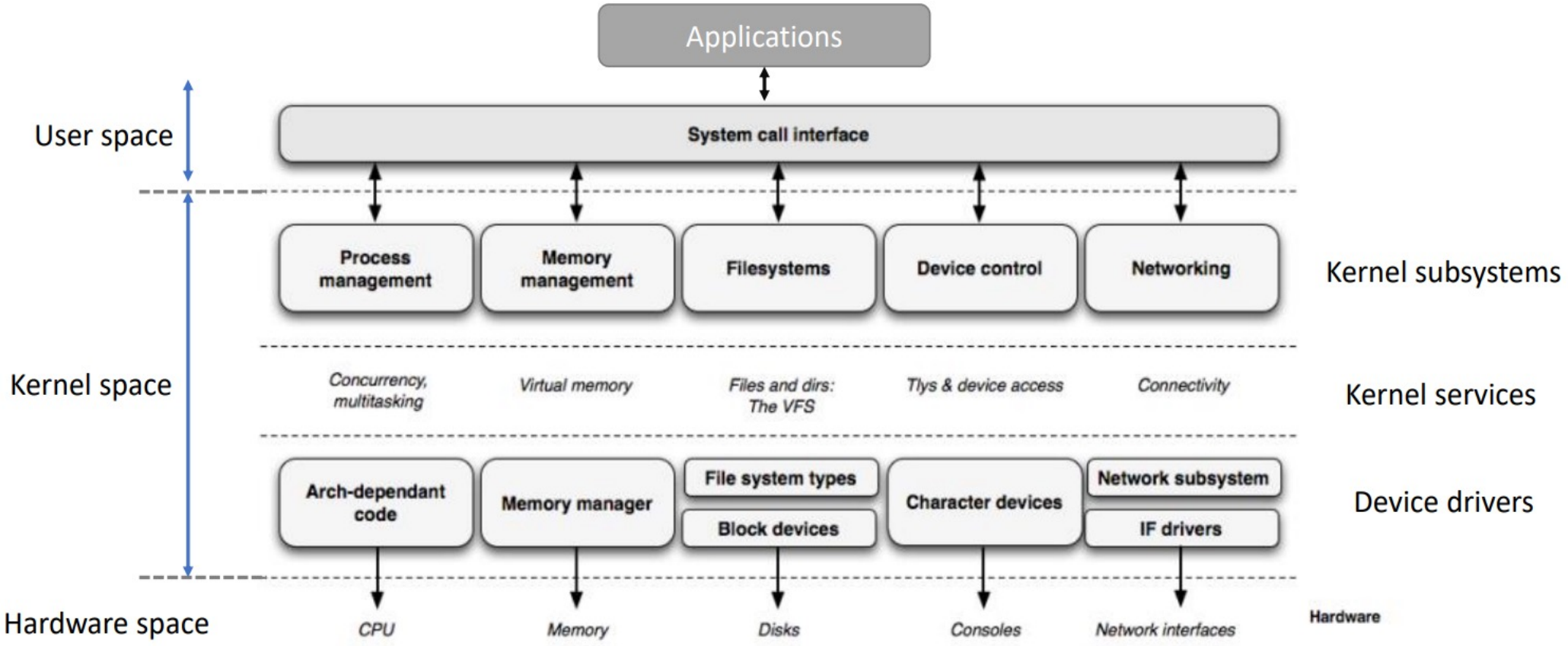


User Space

- GNU C Library is Implemented
- User applications are executed
- This included shells

- Features of Shell
  - Interface between user and kernel
  - Can be more than one
  - User can swap between them
  - Command line and GUI

Kernel Space



Kernel Space

- System Call Interface
  - Provides platform to perform functions from user space to kernel space
  - Architecture dependent
- Kernel Subsystem
  - Process Management
  - Memory Management
  - File System
  - Device Control
  - Networking

Kernel Space

- Process Management
  - Create and destroy processes
  - Communication among different processes
  - Controls how processes share CPU
- Memory management
  - Managed in 'Pages’
  - Controls available memory
  - Controls physical and Virtual memory mappings

Kernel Space

Device Control

- All device control operations are performed by code called a device driver
- Device diver registers devices with the kernel and it handles I/O requests of the device

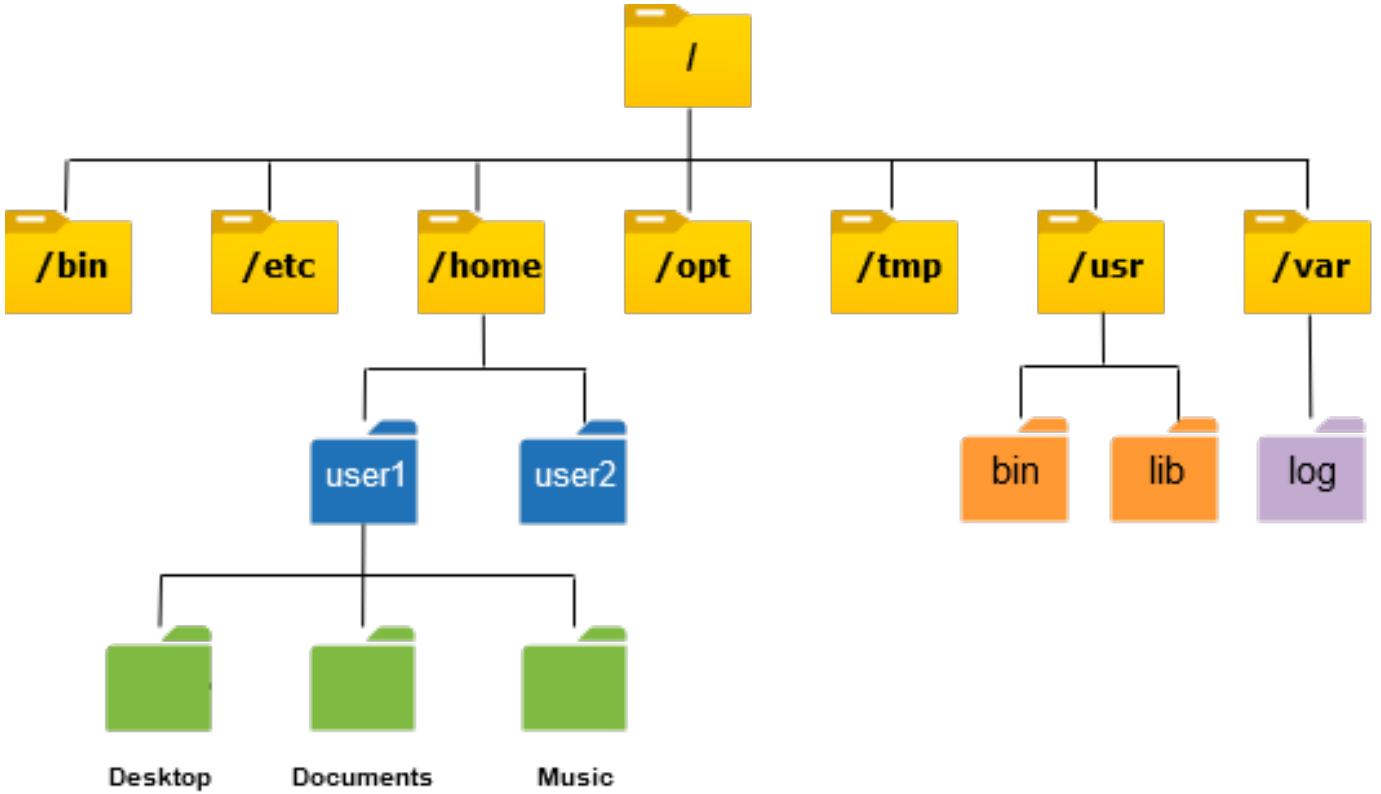
Networking

- Network operations are not specific to a process and incoming packets are asynchronous
- Kernel is in charge of delivering data packets across program and network interfaces

File System

- Linux abstracts file systems operations through the Virtual File System (VFS)
  - Provides an interface for user mode programs to interact with the file system
  - Provides an interface that file systems have to implement
- Handles “mounting”, I/O requests that get implemented (eventually) by a device driver
- Supports more than one file system types
  - Ext2 / Ext3 / Ext4
  - Fat32
  - NTFS

Directory Structure



Directory Structure

- /bin : Common programs, shared by the system, the system administrator and the users
- /boot : The start-up files and the kernel, vmlinuz. In some recent distributions also grub data. Grub is the Grand Unified Boot loader and is an attempt to get rid of the many different boot-loaders we know today
- /dev : Contains references to all the CPU peripheral hardware, which are represented as files with special properties
- /etc : Most important system configuration files are in /etc, this directory contains data similar to those in the Control Panel in Windows
- /home : Home directories of the common users

Directory Structure

- /lib: Library files, includes files for all kinds of programs needed by the system and the users
- /root : The administrative user's home directory. Mind the difference between /, the root directory and /root, the home directory of the root user
- /usr : Programs, libraries, documentation etc. for all user-related programs
- /var: Storage for all variable files and temporary files created by users, such as log files, the mail queue, the print spooler area, space for temporary storage of files downloaded from the Internet, or to keep an image of a CD before burning it
- /mnt : Standard mount point for external file systems, e.g. a CD-ROM or a digital camera

Linux Installation

- Demonstration

https://ws.learn.ac.lk/wiki/Nmm2022/Agenda/Linuxsetup

Types of Users

- Root User
  - The super user
- Normal User
  - Other users that has access
- System User
  - An account used by an application

The Super User

- By default, one account can do anything: root
- Some Linux distributions disable logging in as this user
- Root is powerful
  - It can change (or delete) any file
  - It can perform any function
- Root is dangerous
  - Inexperienced users can break a system
  - Root can be exploited by attackers
- Limit what Root can do remotely – if you allow at all.

Become Another User

- su: “substitute user identity”
  - Syntax: su [options] [username]
  - Give password
  - Quit the shell by typing “exit”
- sudo: executes a single command as another user
  - sudo syntax: sudo [options] [-u user] command
  - If no user is specified, root is assumed

User Process

- Programs you run, typically interactively
- Often-used programs have short, cryptic names
  - ls, cp, rm, pwd, cd, cat, less, mkdir, mv, rm, man
- Hundreds of programs included in base systems
- Thousands of programs can be downloaded, free
- Thousands more can be purchased

Common Commands

- man: display the manual
- ls: list the contents of a directory
- pwd: print working directory
- cd: change directory
- mkdir: make a directory
- cp: copy
- mv: move
- rm: remove

The Format of a Command

- Commands are programs
- Options modify commands
  - Typically a dash followed by a letter (-v)
  - Some utilities also allow dash dash word (--verbose)
- Commands act on Parameters (ls -al /etc)
- Spaces are critical “-- help” != “--help”

Find/ Edit Past Commands

- Try your up arrow
- Now type history
- Run a past command by typing !number
- Looking for something in particular?
  - history | grep command-name
- Don't retype commands
  - It takes longer
  - It can lead to errors

Viewing Configuration Files

- If you want to look, but not touch
  - cat <filename> displays a files contents
  - more <filename> displays with pagination
  - less <filename> paginates with search & more
- Changing files usually requires an editor

Linux Editors

- To edit text files in CLI mode you need editors
- There are lot of editors available
  - Emacs
  - Nano
  - Vi
  - Vim
- We will look at vi as it is one of the most powerful editor and
  once you are used to it, It is very easy

vi Editor

- Starting Vi
  - Opening or creating a file
    vi filename
- Vi Modes of Operation
  - Command Mode :
    - Allows the entry of commands to manipulate text
    - Default mode when vi starts
    - Use Escape key to move into command mode
  - Insert Mode :
    - Puts anything you type into the current file
    - To get into insert mode, commands area (append) and i (insert)

File Permission

- Linux is a Multiuser System
- Different users can access/execute different files
- Checking file permissions

  ls -l

Users, Groups & Privilege Types

- Linux understands Users and Groups
- A user can belong to several groups
- A file can belong to only one user and one group at a time
- Only root can change the ownership of a file
- Privilege Types
  - Read
  - Write
  - Execute

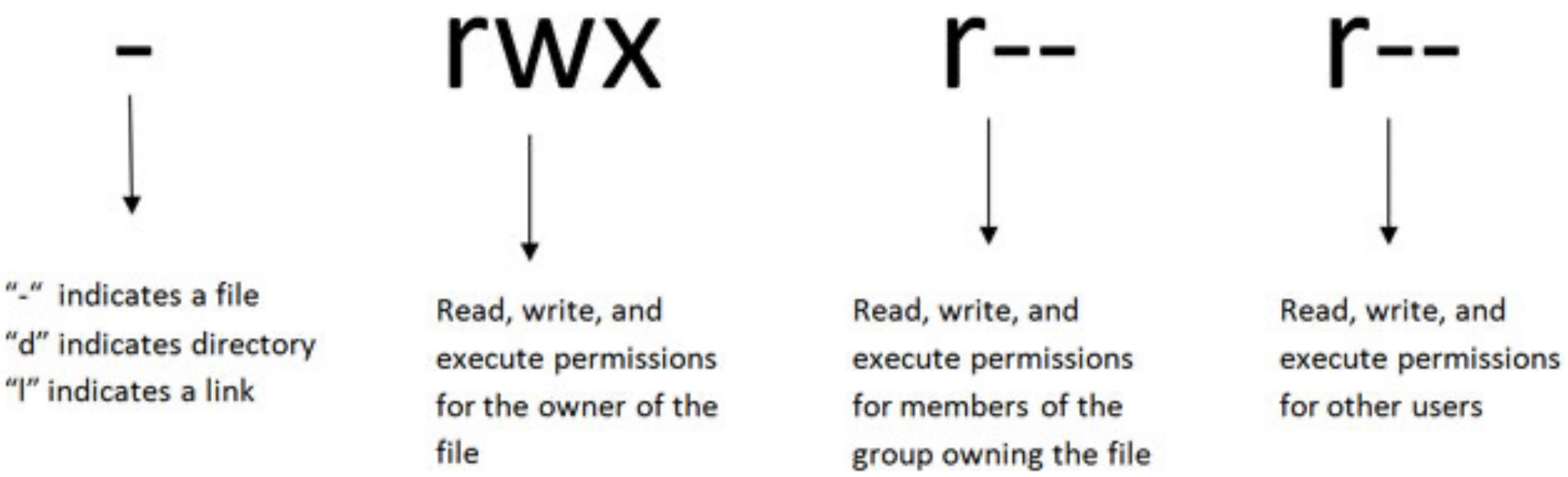
A Program

- A program mostly run by a user, when the system starts or by another process.
- Before the program can execute the kernel inspects several things:
  - Is the file containing the program accessible to the user or group of the process that wants to run it?
  - Does the file containing the program permit execution by that user or group (or anybody)?
  - In most cases, while executing, a program inherits the privileges of the user/process who started it.

File Permission in Detail

- When we type:
  ls -l /usr/bin/top
- We'll see:
  -rwxr-xr-x 1 root root 68524 2011-12-19 07:18 /usr/bin/top
- What does all this mean?

File Permission in Detail



File Permission in Detail

-rwxr-xr-x 1 root root 68524 2011-12-19 07:18 /usr/bin/top

- 1 : Link count
- First root : owner
- Second root : group
- 68524 : Size (In bytes)
- 2011-12-19 : Modification date
- 07:18 : Modification time
- /usr/bin/top : file name

Access Rights

- Files are owned by a user and a group (ownership)
- Files have permissions for the user, the group, and other
- “other” permission is often referred to as “world”
- The permissions are Read, Write and Execute (r, w, x)
- The user who owns a file is always allowed to change its permissions

Changing File Permissions

- File permissions can be change using “chmod” Command
- There are two ways to use this command
  - Symbolic mode
  - Absolute mode

Symbolic Mode

- Uses letters and “+”,”-” to give permissions
- Letters are used as following
  - u: user
  - g: group
  - o: other
  - r: read
  - w: write
  - e: execute
  - + : to add a permission
  - - : to remove a permission

Symbolic mode (Examples)

testfile has permissions of -r--r--r—

- $ chmod g+x testfile ==> -r--r-xr—
- $ chmod u+wx testfile ==> -rwxr-xr—
- $ chmod ug-x testfile ==> -rw--r--r—

u=user, g=group, o=other (world)

Absolute Mode

We use octal (base eight) values represented like this

Number	Permission Type
0	No Permissions
1	Execute
2	Write
3	Execute+Write
4	Read
5	Read+Execute
6	Read+Write
7	Read+Write+Execute

- For each column, User, Group or Other you can set values from
0 to 7

Symbolic Mode (Example)

testfile has permissions of -r--r--r—

- $ chmod 445 testfile ==> -r--r-xr—
- $ chmod 754 testfile ==> -rwxr-xr—
- $ chmod 644 testfile ==> -rw--r--r—

Software Management (the CLI)

- dpkg is the Debian/Ubuntu software manager
  - dpkg --get-selections: see what's installed
  - dpkg -reconfigure: reconfigure a package
  - dpkg --purge: remove software & its config files
- apt is the best way to use dpkg
  - apt-cache search: see what's available
  - apt-get update: get a new list of what's available
  - apt-get install: install software & its dependencies

Scripting

- Shells like bash and Korn have support for programming constructs that can be saved as scripts
- Many Linux commands are scripts
- To run a script user should have execute privilege
- You can do conditions, loops, Pass variables and many programming concepts in your scripts

Package Management

- Package management system is derived from the same system used by the Debian GNU/Linux distribution. The package files contain all of the necessary files, meta-data, and instructions to implement a particular functionality or software application on your Ubuntu computer.

Package Management (Installing/ Removing)

- **Install a Package**: Installation of packages using the apt tool is quite simple. For example, to install the nmap network scanner, type the following:

  sudo apt install nmap

- **Remove a Package**: Removal of a package (or packages) is also straightforward. To remove the package installed in the previous example, type the following:

  sudo apt remove nmap

Package Management (Updating)

- **Update the Package Index**: The APT package index is essentially a database of available packages from the repositories defined in the /etc/apt/sources.list file and in the /etc/apt/sources.list.d directory. To update the local package index with the latest changes made in the repositories, type the following:

```
sudo apt update
```

Package Management (Upgrade)

- **Upgrade Packages**: Over time, updated versions of packages currently installed on your computer may become available from the package repositories (for example security updates). To upgrade your system, first, update your package index as outlined above, and then type:

```
sudo apt upgrade
```

System Handling commands

- **View Running Processes in Linux:**

```
$ top
```

```
Output
top - 15:14:40 up 46 min,  1 user,  load average: 0.00, 0.01, 0.05
Tasks:  56 total,   1 running,  55 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   1019600k total,   316576k used,   703024k free,     7652k buffers
Swap:        0k total,        0k used,        0k free,   258976k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
    1 root      20   0 24188 2120 1300 S  0.0  0.2   0:00.56 init
    2 root      20   0     0    0    0 S  0.0  0.0   0:00.00 kthreadd
    3 root      20   0     0    0    0 S  0.0  0.0   0:00.07 ksoftirqd/0
    6 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
    7 root      RT   0     0    0    0 S  0.0  0.0   0:00.03 watchdog/0
    8 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 cpuset
    9 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 khelper
   10 root      20   0     0    0    0 S  0.0  0.0   0:00.00 kdevtmpfs
```

The first several lines of output provide system statistics, such as CPU/memory load and the total number of running tasks.

You can see that there is 1 running process, and 55 processes that are considered to be *sleeping* because they are not actively using CPU cycles.

System Handling commands

- Analyze usage for the current directory and any subdirectories

“du” will analyze usage for the current directory and any subdirectories. The default output of du running in a nearly-empty home directory looks like this:

```
$ du
```

```
Output
4     ./.cache
8     ./.ssh
28     .
```

System Handling commands

- **View the system hostname and version**

The “uname” tool is most commonly used to determine the processor architecture, the system hostname and the version of the kernel running on the system.

```
$uname
```

System Handling commands

- **Memory Information:**

On Linux you can use the command cat /proc/meminfo to determine how much memory the computer has. This command displays the information stored in the meminfo file located in the /proc directory.

$ cat /proc/meminfo

System Handling commands

- **Get CPU Informaiton:**

You can simply view the information of your system CPU by viewing the contents of the **/proc/cpuinfo** file with the help of cat command as follows:

```
$ cat /proc/cpuinfo
```

System Handling commands

- **Get network related information :**

Netstat command displays various network related information such as network connections, routing tables, interface statistics, masquerade connections, multicast memberships etc.,

netstat -a | more : To show both listening and non-listening sockets.

netstat -at : To list all tcp ports.

netstat -au : To list all udp ports.

netstat -l : To list only the listening ports.

netstat -lt : To list only the listening tcp ports.

netstat -lu : To list only the listening udp ports.

System Handling commands

- **Checking the network connectivity:**

Ping is short for **Packet Internet Groper**. This command is mainly used for checking the network connectivity among host/server and host. The ping command takes the URL or IP address as input and transfers the data packet to a specified address along with a **"PING"** message. Then, it will get a reply from the host/server. This time is known as **"latency"**.

```
$ ping 8.8.8.8

$ ping www.google.com
```

System Handling commands

- **Tracepath:**

The tracepath command in Linux allows to trace the path to the destination path determining MTU along this path using UDP port or any other ports that will not require any superuser permissions.

The general syntax of Linux tracepath is:

tracepath [destination]

When the options are given along with the tracepath command, the syntax is given below:

tracepath [-n] [-b] [-l pktlen] [-m max] [-port] destination

Thank You