



SECTION 1

Linux Fundamentals

1. Basic definitions and Concepts in Linux

Operating system:

A software that manages the hardware, other software and devices in a computer system. e.g.: Windows, Unix, Linux. IOS, Android etc.

- Linux is an open source Operating System, you do not have to pay to use this Operating system*
- Linux was developed by *Linus Torvalds* in 1991 during his college time as an alternative to the paid versions of *nix system.

Open Source:

A program, tool or package which is available to all for free. Even the source code for the program is available on internet which can be customized and use without any approval or License.

- The code used to create Linux is free and available to all. You can always edit it and make your own version of Linux.
- It is secure, flexible, and has excellent community support.
- Devices such as, Mobile phone, tablets, cameras, wearables and most web services available on Internet uses Linux OS.



Linux Distributions

There are many variants of Linux available in market and we can use any of them as per our requirement and used cases.

Some of the most common Distributions are given below-

- Amazon Linux (native to AWS)
- Ubuntu (*one of biggest community support, and best for beginners*)
- RedHat (*Commercial version of Linux*)
- Fedora (*Based on Linux and a testing version of RedHat that can used for free*)
- CentOS (*Based on Redhat Linux and free to use*)

Linux Kernel

Kernel is the heart of Linux Operating System. It is the interface between hardware and its processes. It takes care of memory management, process management and other hardware drivers

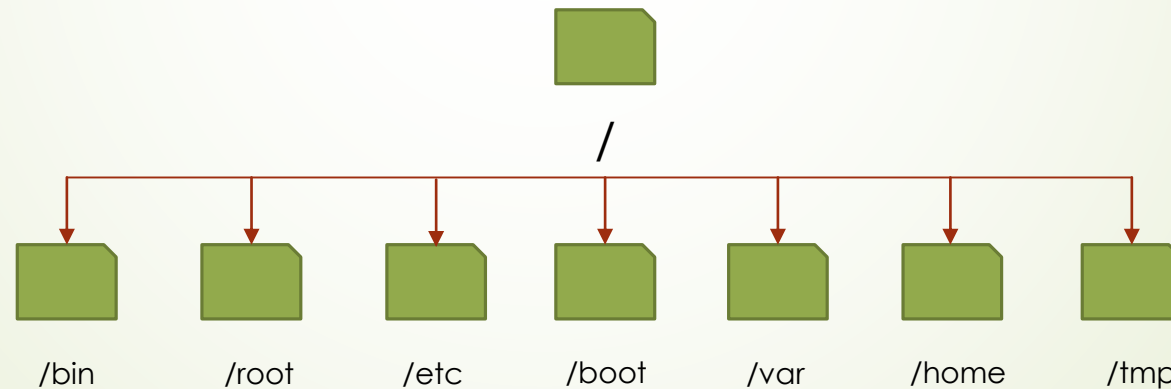
2. Creating a Linux Virtual Machine in AWS

➤ AWS Management Console

- Create a new account on AWS, it will ask for credit/debit card details.
- All Labs that we will perform in this course can be completed free of cost (Most of the services are free of cost for one year , we just need to ensure not to cross the free eligible tier)
- Select Ubuntu distribution while creating the VM.
- Install putty on Windows machine to connect to the Linux servers.
- Once connected, we can proceed with the next topic of the course.

3. Managing File System

- All files are stored in Linux in a single tree of directory known as file system hierarchy.
- The root '/' is at the top of hierarchy and all other directories & subdirectories start with the root (/) directory.
- Different directories are available in Linux by default and have specific purpose.





Important Directories on the system and their purpose

LOCATION	PURPOSE
/home	Home directory of regular users where they store their personal data and files.
/root	Home directory for the root user (superuser)
/etc	Configuration files specific to the system
/boot	Stores the files needed to start the system (booting process)
/var	Variable data specific to the system, files that dynamically change e.g.: log files
/tmp	A world writable space for temporary files
/bin	Executable binary files, ready to run programs

File Management Commands

COMMAND	USAGE
pwd	To get the present working directory, the path that we are using currently
ls	File listing. Can be used with different flags such as 'a', 'l' etc.
cd	Changing the current directory or to move to a different directory
touch	To create an empty file
cat	To see the contents inside a file
mkdir	To create a directory
cp	To copy files/ directories from one location to another
mv	To move and Rename a file or directory
rm	To Remove a file or directory
ip r	To get the IP address of the server

4. Managing Users and Groups

Points to Discuss

- Creating a new user and a group

```
# useradd # groupadd
```

Set the password for the user

```
# passwd
```

- Removing a user and a group

```
# userdel # groupdel
```

- Checking the available users in the system

```
# cat /etc/passwd
```

- Checking the available groups in the system

```
# cat /etc/group
```

- Checking associated groups with a user

```
# id # cat /etc/group
```

- Modify a user setting (associated roles)

```
# usermod
```


Explaining the important files

➤ */etc/passwd*

```
myuser:x:1001:1001::/home/myuser:/bin/bash
```

- It begins with the username **myuser**.
- There is an **x** for the password field indicating that the system is using shadow passwords.
- A UID greater than 999 is created. Under Red Hat Enterprise Linux 7, UID s below 1000 are reserved for system use and should not be assigned to users.
- A GID greater than 999 is created. Under Red Hat Enterprise Linux 7, GID s below 1000 are reserved for system use and should not be assigned to users
- Next field is showing the comment that we added during user creation.
- The home directory for **myuser** is set to **/home/myuser/**.
- The default shell is set to **/bin/bash**.

5. Gaining Privileges

- The su command
 - This command helps you to switch from the current user to the specified user.
 - If nothing {no argument} is given to the command, then it will switch to the root user account
- The sudo command
 - When trusted users run a command with sudo, they are prompted for their own password
 - Only users listed in /etc/sudoers file, can use sudo.
 - If you want to avoid giving password every time, add the following for the user

```
# john ALL=(ALL) ALL
```

6. Linux File System Permissions

- File /Directory permission is basically divided in three categories-
 - read (r)
 - write (w)
 - execute (x)
- Effect of permissions on files and directories

Permissions	Effect on files	Effect on directories
read (r)	Contents of the file can be read	Contents of the directory can be listed
write (w)	Contents of the file can be changed	Any file in the directory can be created or deleted
exec (x)	Files can be executed as commands	Contents of the directory can be accessed (depends on the permissions of file inside the directory)

- Viewing file/directory permissions and their ownership
 - # `ls -l <filename>`
- When you check directory permissions using this command, it will show the permissions of entire content inside the directory
 - To check permission of directory itself use `ls -ld <directory name>`

Changing the Permissions

- `chmod` is the command to change the permissions of a file or directory.

- *Using symbolic method*

```
# chmod WhoWhatWhich file/directory
who is u, g, o, a (for user, group, other, all)
what is +, -, = (for add, remove, set exactly)
which is r, w, x (for read, write, executable)
```

- *Numeric Method*

```
# chmod PPP file/directory
p is the sum of r, w, x
r = 4; w = 2; x = 1
```



Examples

1. Create a file and provide full permissions (everything)
2. Create a file with only read permission for all, and then add executable permission group.
3. Create a directory with read, write and execute permissions for user; read and execute for group and nothing for others
4. Create nested directories and update the files permissions recursively.



Changing the Ownership

- chown is the command to change the user/group ownership of a file or directory.

```
# chown user:group file/directory
```

- Examples:

1. Create a user account and name it 'alpha', Now create a file and change its user ownership to alpha user.
2. Create nested directories with some files in it and change user ownership to alpha user for all files recursively.
3. Create a group called 'teamalpha' and change the group ownership of the file to this new group.
4. Create another user & group and name them 'beta' & 'teambeta' respectively. Now change the user and group ownership of all the files inside the nested directory with this new user:group pair.

Understand the permission with scenarios

Username	Groups
alpha	alpha, coder
beta	beta, coder
gamma	gamma, gamer
delta	delta, gamer

Permissions	User	Group	File/directory
drwxrwxr-x	beta	coder	dir (below files are available in this directory)
-rw-rw-r--	alpha	Alpha	File1
-rw-r--rw-	alpha	coder	File2
-rw-rw-r--	Beta	coder	File3
-rw-r-----	Beta	coder	file4



Query and Solution

1. Who can access the file1?
2. Can beta view/change/modify the content of file2?
3. Who can delete all the files in the directory?
4. Will gamma be able to update contents of file2 and file3?
5. Who all cannot read the content of file4.

7. Installing and updating packages in Linux

➤ RPM Package Manager

- RPM is a standard way to package software for distribution.
- It helps to keep the track of packages and help with the supporting packages.

`httpd-tools-2.4.6-7.el7.x86_64.rpm`

Name of package; Version; Release.Arch

➤ Some common comand

```
# yum repolist
# yum list installed
# yum list "package-name"
# yum info "package-name"
# yum update
# yum list kernel
```



Examples

- Yum search for the webserver `httpd/nginx`
- Check the info of this package
- Install the package
- Remove the above installed package from the machine.

Examining the rpm files

- `rpm -q` is used to query packages available in the system
- If you want, we can query all the packages at once using `-a`
- For any specific package, just add the name at the end of `rpm -q`
- To install any package using rpm; use `rpm -ivh <rpm file>;`
- you can also use `yum localinstall <rpmfile name>` to keep the yum history intact.
- **Example-**
 - Check if nginx package is installed on machine using rpm.
 - Download the rpm for nginx from internet
 - Check the installed package
 - Remove the package using rpm command

8. Accessing Linux Filesystem

- To get an overview of Linux filesystem mount points and the amount of available free space use **df** command.
- To check the disk space of any partition, use **du** command.
- To check the existing partition and its filesystem use **blkid** command
- To mount a partition, (volume) on the system use **mount** command.
- Use **umount** command to unmount a partition.



Working with Linux filesystem


- Create a new volume in AWS Account
- Attach this volume to your EC2 instance.
- Once attached, check on the server the details of this newly created volume using **lsblk/blkid** command.
- Create the filesystem on this volume (XFS, EXT4)
- Mount this volume to a path in your system. /mnt/mydata
- Make permanent entry of this volume so it will sustain the reboot.

Managing Links

Hard links:

- A hard link is a new directory entry with reference to an existing file on the system.
- Every file in a filesystem has a hard link by default.
- To save space, instead of copying file, a new hard link can be created to reference the same file.
- **# ln** command is used to create a hard link.

```
# ln file1.txt /tmp/hardlinkfile1.txt
```
- All hard links referencing the same file has the same permissions, like count, user/group permissions, time stamps and file content.
- Even if the original file gets deleted, the content of the file is still available as long as atleast one hard link exists.



Soft links or symbolic link:

- A soft link is a special file type that points to an existing file or directory.
- **# ln -s** command is used to create a soft link.

```
# ln file2.txt /tmp/softlinkfile2.txt
```

- When the original file gets deleted, the soft link still points to file but the file is gone. Such kind of soft links are called dangling soft links.
- A soft link can point to a directory, and we can even do **cd** to a soft link directory and it will work just fine.

9. Configuring and Working with OpenSSH

What is OpenSSH

- The openSSH secure Shell, (ssh) is used to securely run a shell on a remote machine.
- If you have an account on the remote system that offers SSH and want to access the machine from your local laptop, we generally use **ssh** command for it.

```
# ssh user@hostmachine
```

- If you want to check the currently logged in users on your machine, use **w** command.

Key based Authentication

- SSH can be done using username/password and using the encrypted keys. The keys are considered as the most secure way to login to the system.
- This is a combination of two keys, public and private. The public key gets saved inside the remote machine (on first login/or when the administrator saves it) and the private key must be available with the user.
- **.ssh** is the directory which stores all the ssh related keys (public) for each user.



Configuring the ssh key based authentication

- We can use the command `ssh-keygen` to create a new key pair. (There are a lot of ways to create a key pair and the ultimate result is just to have a key pair).
- During the key generation, you can specify the passphrase which is nothing but a password for added security. This gives you two factor authentications.
- Two files, that get created using above command are `id_rsa` and `id_rsa.pub`.
- `id_rsa` is the private key which should be available with the user only and the other one is public key which the administrator will save in the user's `.ssh` directory.
- Save the public key in `authorized_keys` file in `.ssh` directory. (if the file is not available then you can create the one for the user)
- Give the owner and group permission of entire `.ssh` directory to the user account.
- At last, the private key should have minimal permission only (400 works just fine).

Understanding the ssh config file

- ▶ **/etc/ssh/sshd_config** is the configuration file which we can customize as per our requirements with respect to ssh.
- ▶ *Prohibit the root user to login in using ssh:* It is advisable to disable root user ssh login due to security reasons. *The username root is the most common and always available on the system.*

Modify the line **PermitRootLogin** to **no** to disable ssh for root user.

- ▶ *Prohibit password authentication using ssh:* This makes the use of key based authentication mandatory for all ssh connections. It is also one of the most advisable point for ssh since hacking password is quite normal in today's scenario.

Modify the line **PasswordAuthenticaiton** to **no** to disable ssh for root user.

*For any modification to take effect, we need to restart the ssh service (sshd). *

10. Linux Booting Process

1. BIOS: Basic Input Output System

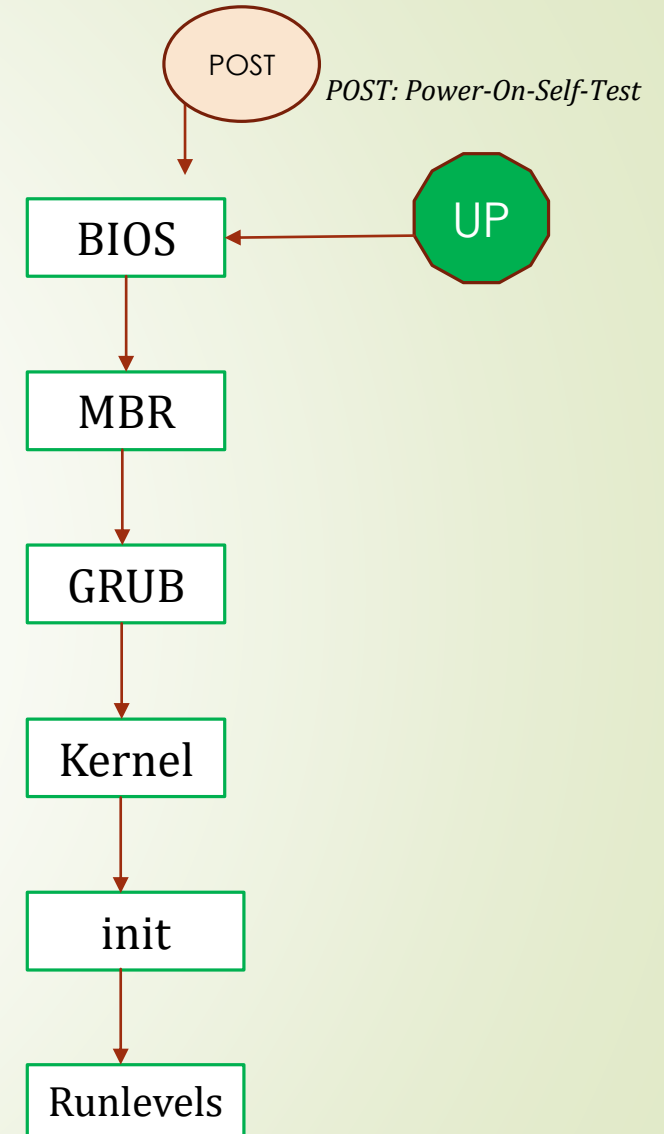
- It performs the system integrity checks, all the connected peripherals.
- It also searches, load and execute the boot loader program.
- Once bootloader is detected and loaded in memory, BIOS gives the control to MBR.

2. MBR: Master Boot Record

- It is located in first sector of the bootable disk.
- MBR is 512 Bytes in size and has three components-
 - Primary Boot Loader (446 bytes)
 - Partition table information (64 bytes)
 - MBR validation check (2 bytes)
- It contains information about GRUB Boot Loader.

3. GRUB: Grand Unified Bootloader

- If there are multiple kernel images available, we can select the required one from here.
- If nothing is selected, it will boot the default kernel image



4. Kernel:

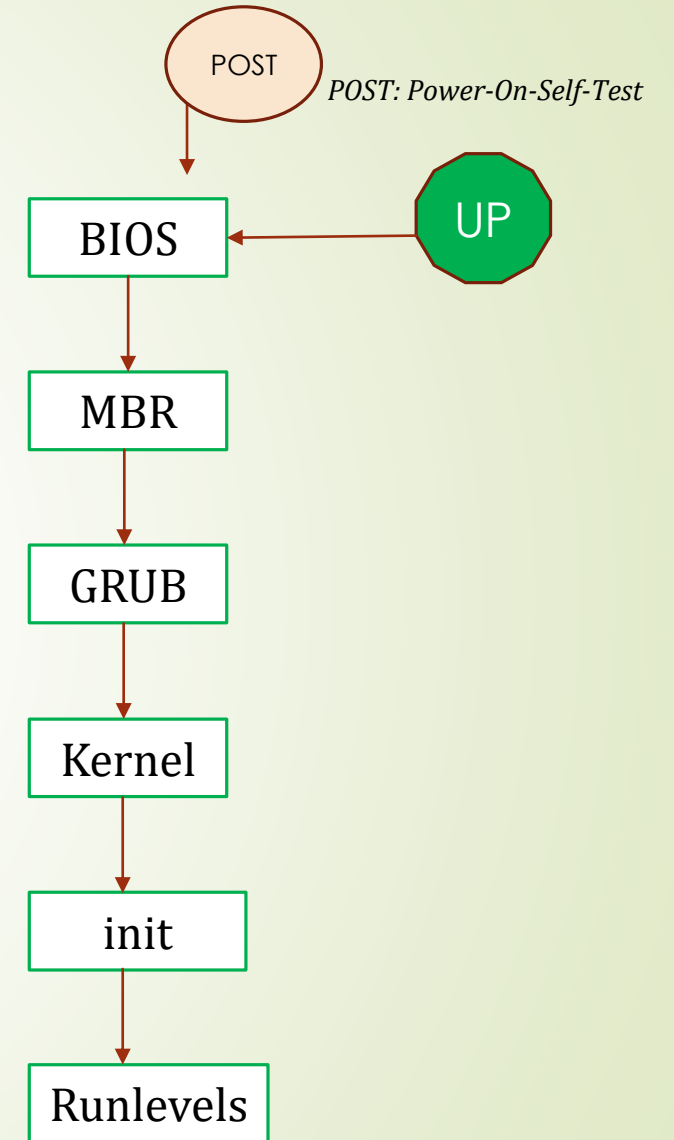
- Once kernel is loaded into memory, it starts the very first process of the system i.e., 'init process'.
- Since init is the first program to be loaded it has the process id as 1.

5. Init:

- Looks at the `/etc/inittab` file and select the Linux Run level
- Following are the available run levels
 - 0 – halt
 - 1 – Single user mode
 - 2 – Multiuser mode without network file system
 - 3 – Full multiuser mode
 - 4 – Unused
 - 5- Multiuser mode with GUI
 - 6 - reboot

6. Runlevels:

- Depends on what runlevel has been selected or was default for init, all the processes that were set to start in that particular runlevel initiates.



11. Important Concepts

Managing the logs:

- By default, most of the system logs are present in `/var/log` directory in their individual folders.
- `logrotate` service is used to keep rotating the log files to keep the space filling up due to continuous logs

Log File	Purpose
<code>/var/log/messages</code>	Most syslog messages are stored here, The exceptions are messages related to authentication and email processing
<code>/var/log/secure</code>	For security and authentication related messages
<code>/var/log/maillog</code>	The log file with mail server related logs
<code>/var/log/cron</code>	The logs related to periodic tasks
<code>/var/log/boot.log</code>	Messages related to system startup

- The default system logs can be viewed using `journalctl` command in latest Linux distributions.



Working with Running Processes

- By default, there are many processes started by the kernel also at the same time many user managed processes run in the system.
- We can always check the running process and get their details by using command **ps**.
- We can get the process id, config files, and other important details with the use of ps command with certain flags.
- To kill a process, we can use the **kill** command. There are many signals that we can provide to kill command as per the requirement.



System CPU and Memory Utilization

- To check the CPU utilization of a system, we use the **top** command. This command provides almost all the important system details that we need to monitor our machine.
- *Load Average:* Load average is calculated based on the CPU core and the number of processes waiting, to be executed by the CPU at a particular time. The more is load average, the more processes are running, and system might perform slow.
- To get the memory utilization of a system, use **free** command. We can use free command with **-g/-m** flags to get the details in more human readable format.
- *Swap Space:* Swap disk/space is a special partition that we create on our machine to complement the system RAM. When the system RAM is fully exhaust, this swap space can be used to keep unwanted pages/files to free up RAM space.
- System Memory and CPU utilization can also be determined by seeing the **/proc** files.

Automate system tasks using Cron Job

- Cron job or sometimes refer to as cron tasks are the scheduled jobs that gets executed at the defined time.
- We can specify the command/script that needs to execute at a particular time using **crontab** command.
- Crontab takes the time in following format-

Minute Hour DayOfMonth Month DayOfWeek

Eg: 0 2 10 * *

will execute the command on 10th of every month at 2:00 AM

- To list the current configured cron jobs; use the command **crontab -l**
- To create a new cron job, use the command **crontab -e**



Working with vi editor

- One of the most powerful editing tools in Linux.
- This is used to view, edit, modify the files available in the system.
- There are three modes in vi editor to work
- Command mode; vi starts with the command mode and then you can provide relevant options to jump on either Visual mode or Insert mode.
- Visual mode; This is useful to perform some bulk modifications for e.g.: commenting a lot of lines at ones.
- Insert mode; Almost everything we do in vi editor is done in Insert mode.
- Once you are with the editing/ modification of your file. Press **ESC** followed by **:wq** to save the file.
- In case you want to quit without saving the files (to discard your changes) use the same key combination except the **w**.