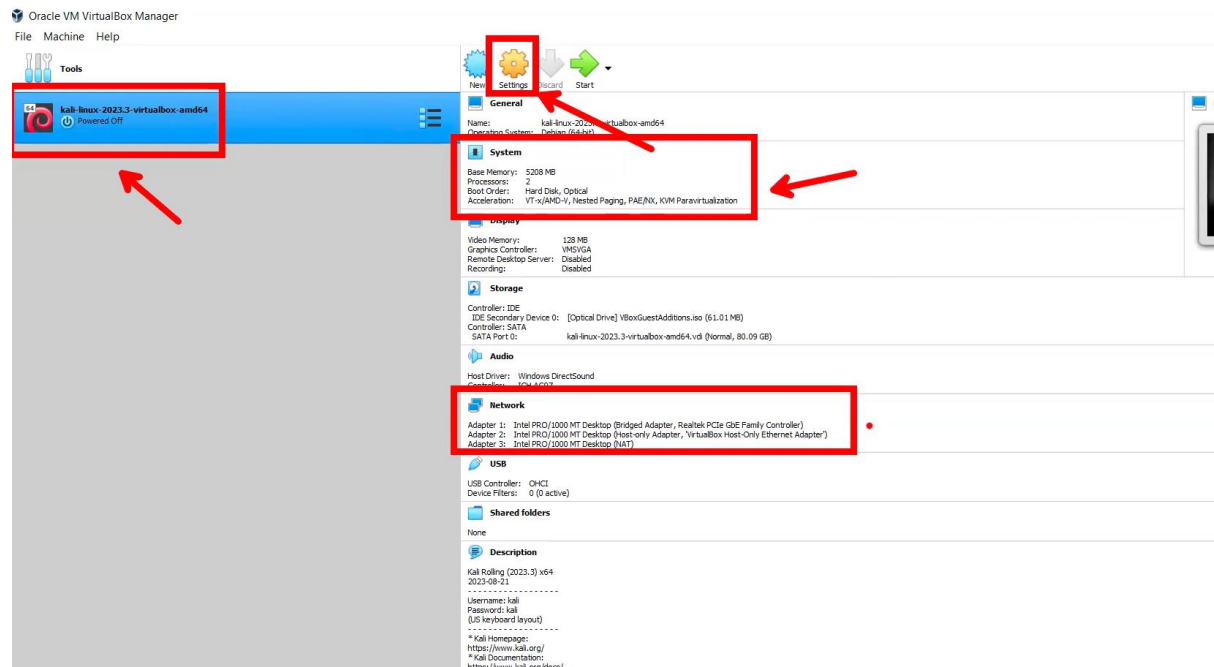
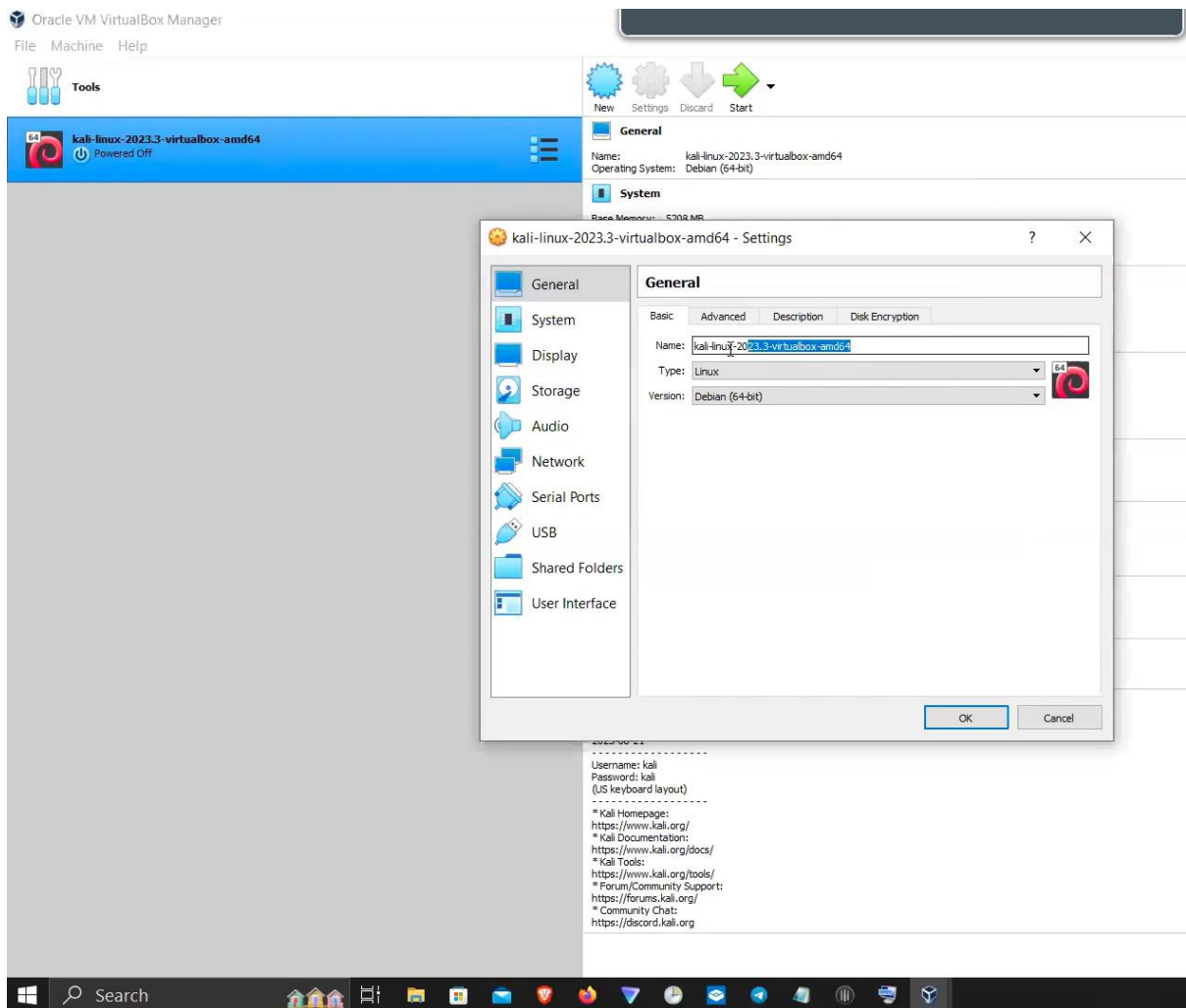


## Day 5 :

How you can change virtual machine system configuration



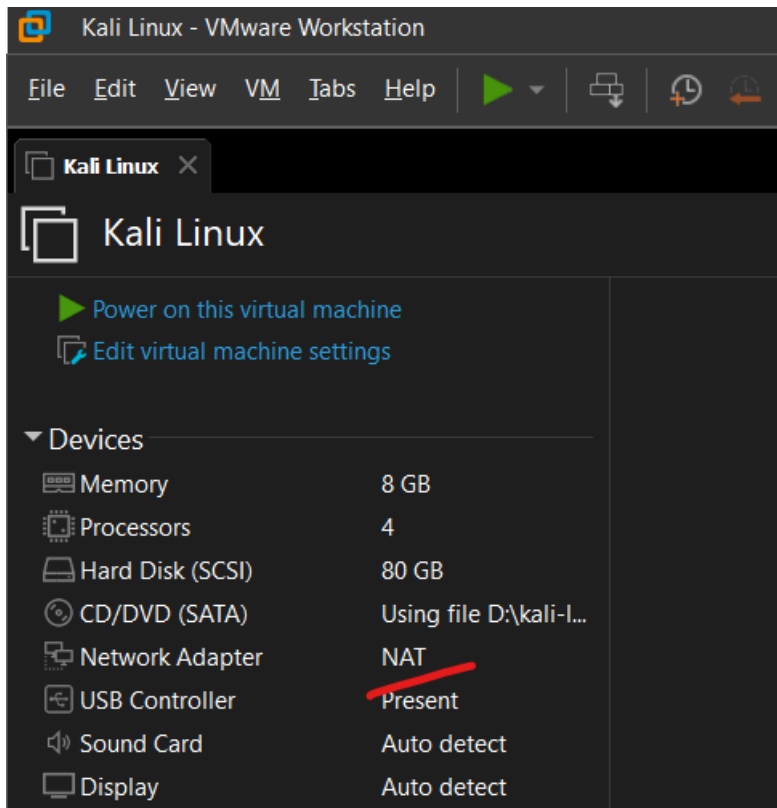


## NAT :

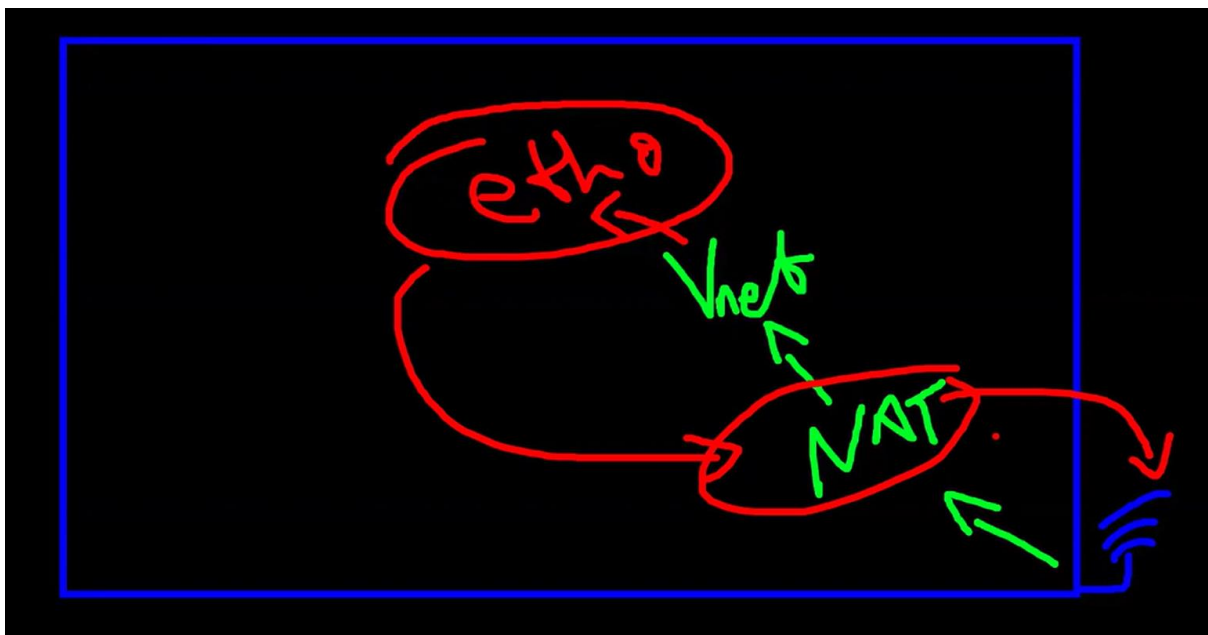
NAT stands for network address translation. It's a way to map multiple private addresses inside a local network to a public IP address before transferring the information onto the internet. Organizations that want multiple devices to employ a single IP address use NAT, as do most home routers. If you're connecting from your home right now, chances are your cable modem or DSL router is already providing NAT to your home.

By default NAT adapter is present in machine you set up in virtual machine, here we deploy kali machine in virtual box kali may have default network adapter is NAT.

Our Kali NAT is used to translate host machine network to virtual machine network, hence we can use network and other devices in network via virtual machine (kali linux).



Lets see In simple language how it works,



As shown in picture the rectangle is our machine, 3 blue line our Wi-Fi so internet comes to wifi then it goes NAT and NAT translate public internet to accessible private internet and future it creates virtual network Vnet for virtual machine operating system kali Linux and VM application crates eth0 adapter for kali linux which is virtual or we can say software based not physical ethernet.

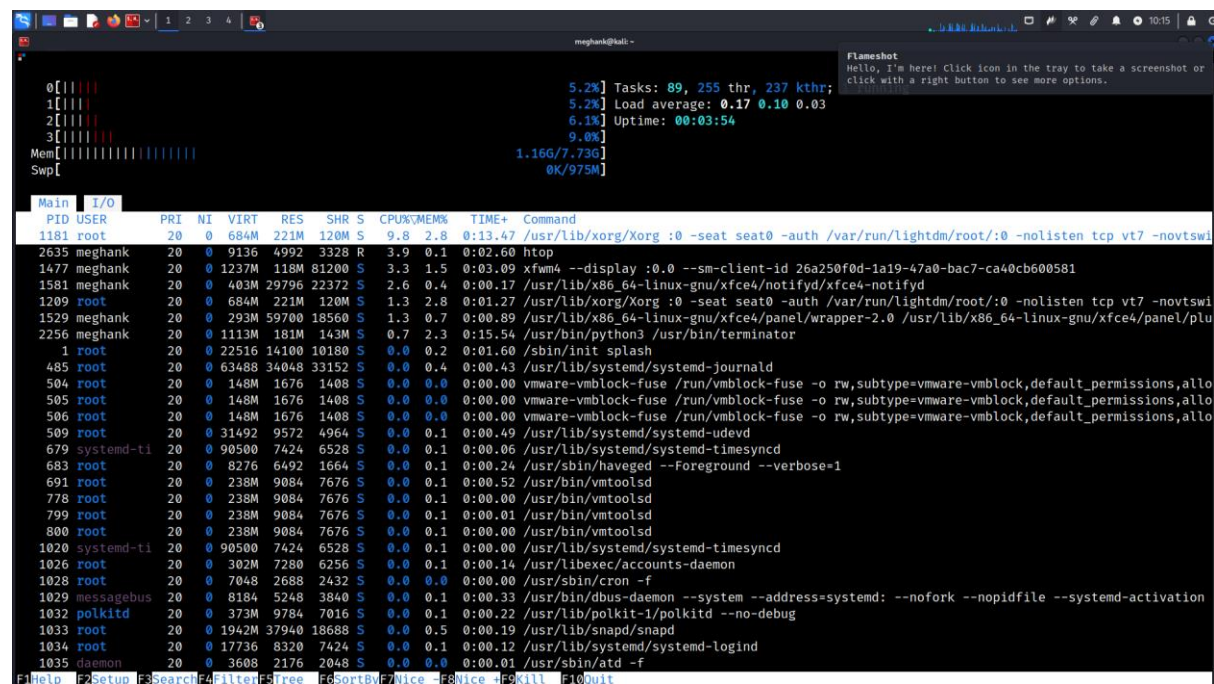
Host-Only adapter in Virtual Box :

A Host-Only Ethernet Adapter allows communication between the VM and the computer it's running on. For example, you can setup shared folders or run a web services on the VM and access it from the host. However it doesn't allow the VM to access anything else in your network, not even the internet.

htop :

The htop is a command-line utility that allows you to interactively monitor your system's vital resources or server processes in real-time

In a nutshell, htop is a useful command-line tool in the Linux environment to determine the cause of load by each process. It is similar to Task Manager in the Windows OS environment. It can be used to troubleshoot and kill a process that is utilizing excessive server resources

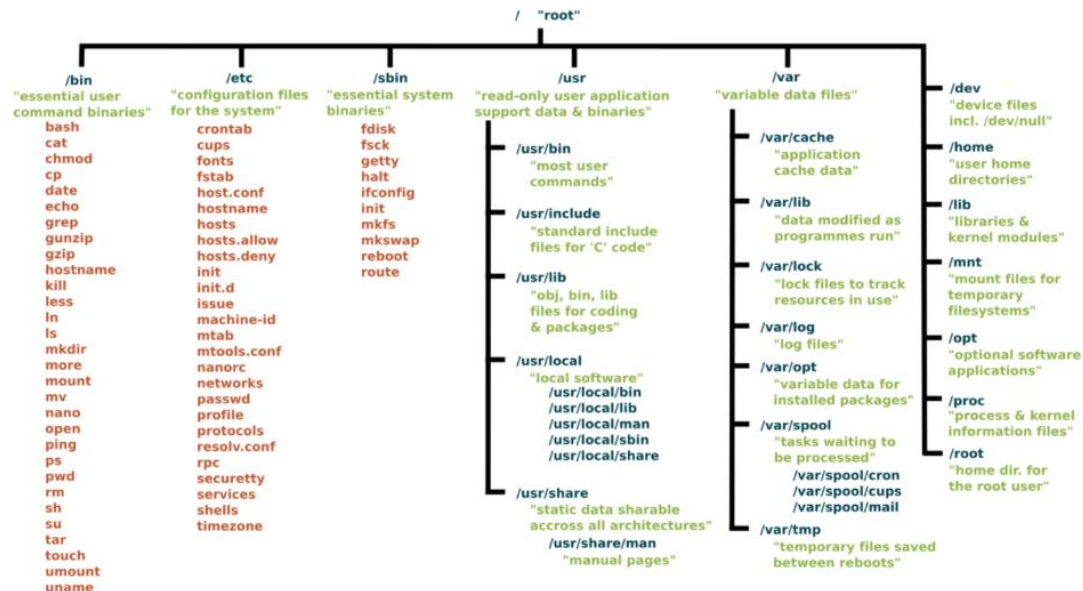


The screenshot shows the htop utility running in a terminal window. At the top, it displays system statistics: Tasks: 89, 255 thr, 237 kthr; Load average: 0.17 0.10 0.03; Uptime: 00:03:54; Memory: 1.166/7.73G; Swap: 0K/975M. Below this is a table of running processes. The table has columns for PID, USER, PRI, NI, VIRT, RES, SHR, S, CPU%, MEM%, TIME+, and Command. The processes listed include htop, xfwm4, xfce4-notifyd, xfce4-panel, sbin/init splash, sbin/python3, sbin/terminator, sbin/systemd-journald, vmware-vmtoolsd, sbin/haveged, sbin/vmtoolsd, sbin/cron, sbin/dbus-daemon, sbin/polkitd, sbin/snapd, sbin/systemd-timesyncd, sbin/daemon, and sbin/atd.

PID	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
1181	root	20	0	684M	221M	120M	S	9.8	2.8	0:13.47	/usr/lib/xorg/Xorg :0 -seat seat0 -auth /var/run/lightdm/root/:0 -nolisten tcp vt7 -novtswi
2635	meghank	20	0	9136	4992	3328	R	3.9	0.1	0:02.60	htop
1477	meghank	20	0	1237M	118M	81200	S	3.3	1.5	0:03.09	xfwm4 --display :0.0 --sm-client-id 26a250f0d-1a19-47a0-bac7-ca40cb600581
1581	meghank	20	0	403M	29796	22372	S	2.6	0.4	0:00.17	/usr/lib/x86_64-linux-gnu/xfce4/notifyd/xfce4-notifyd
1209	root	20	0	684M	221M	120M	S	1.3	2.8	0:01.27	/usr/lib/xorg/Xorg :0 -seat seat0 -auth /var/run/lightdm/root/:0 -nolisten tcp vt7 -novtswi
1529	meghank	20	0	293M	59700	18560	S	1.3	0.7	0:00.89	/usr/lib/x86_64-linux-gnu/xfce4/panel/wrapper-2.0 /usr/lib/x86_64-linux-gnu/xfce4/panel/plu
2256	meghank	20	0	1113M	181M	143M	S	0.7	2.3	0:15.54	/usr/bin/python3 /usr/bin/terminator
1	root	20	0	22516	14100	10180	S	0.0	0.2	0:01.60	/sbin/init splash
485	root	20	0	63488	34048	33152	S	0.0	0.4	0:00.43	/usr/lib/systemd/systemd-journald
504	root	20	0	148M	1676	1408	S	0.0	0.0	0:00.00	vmware-vmtoolsd-fuse /run/vmtoolsd-fuse -o rw,subtype=vmware-vmtoolsd,default_permissions,allo
505	root	20	0	148M	1676	1408	S	0.0	0.0	0:00.00	vmware-vmtoolsd-fuse /run/vmtoolsd-fuse -o rw,subtype=vmware-vmtoolsd,default_permissions,allo
506	root	20	0	148M	1676	1408	S	0.0	0.0	0:00.00	vmware-vmtoolsd-fuse /run/vmtoolsd-fuse -o rw,subtype=vmware-vmtoolsd,default_permissions,allo
509	root	20	0	31492	9572	4964	S	0.0	0.1	0:00.49	/usr/lib/systemd/systemd-udevd
679	systemd-ti	20	0	90500	7424	6528	S	0.0	0.1	0:00.06	/usr/lib/systemd/systemd-timesyncd
683	root	20	0	8276	6492	1664	S	0.0	0.1	0:00.24	/usr/sbin/haveged --Foreground --verbose=1
691	root	20	0	238M	9084	7676	S	0.0	0.1	0:00.52	/usr/bin/vmtoolsd
778	root	20	0	238M	9084	7676	S	0.0	0.1	0:00.00	/usr/bin/vmtoolsd
799	root	20	0	238M	9084	7676	S	0.0	0.1	0:00.01	/usr/bin/vmtoolsd
800	root	20	0	238M	9084	7676	S	0.0	0.1	0:00.00	/usr/bin/vmtoolsd
1020	systemd-ti	20	0	90500	7424	6528	S	0.0	0.1	0:00.00	/usr/lib/systemd/systemd-timesyncd
1026	root	20	0	302M	7280	6256	S	0.0	0.1	0:00.14	/usr/libexec/accounts-daemon
1028	root	20	0	7048	2688	2432	S	0.0	0.0	0:00.00	/usr/sbin/cron -f
1029	messagebus	20	0	8184	5248	3840	S	0.0	0.1	0:00.33	/usr/bin/dbus-daemon --system --address=systemd: --nofork --nopidfile --systemd-activation
1032	polkitd	20	0	373M	9784	7016	S	0.0	0.1	0:00.22	/usr/lib/polkit-1/polkitd --no-debug
1033	root	20	0	1942M	37940	18688	S	0.0	0.5	0:00.19	/usr/lib/snapd/snapd
1034	root	20	0	17736	8320	7424	S	0.0	0.1	0:00.12	/usr/lib/systemd/systemd-logind
1035	daemon	20	0	3608	2176	2048	S	0.0	0.0	0:00.01	/usr/sbin/atd -f

## Linux File Structure :

### Linux file system/ Directory



1. **/ (Root):** The starting point for the file system hierarchy. All other directories are subdirectories of the root directory.
2. **/bin:** Contains essential command binaries required for booting and repairing the system.
3. **/etc:** Holds system-wide configuration files and shell scripts used to initialize system settings for applications.
4. **/home:** Home directories for all users. Each user has a subdirectory named after their username.
5. **/var:** Contains variable data files such as logs, databases, and temporary files.
6. **/usr:** Contains user binaries, libraries, documentation, etc. It's a secondary hierarchy for read-only user data.
7. **/lib:** Contains essential shared libraries and kernel modules.
8. **/dev:** Contains device files which represent hardware components.
9. **/tmp:** Temporary storage for files. It's cleared on system reboot.
10. **/opt:** Optional application software packages.
11. **/sbin:** Contains system binaries essential for booting, restoring, and recovering the system.
12. **/srv:** Contains data for services provided by the system.
13. **/proc:** A virtual filesystem that provides detailed information about kernel and processes.
14. **/sys:** A virtual filesystem that provides an interface to kernel data structures.
15. **/run:** Contains runtime data for processes started since the last boot.
16. **/boot:** Contains files needed to start the boot process.
17. **/mnt:** Temporary mount points for mounting filesystems.
18. **/media:** Mount points for removable media like USB drives and CDs.

## File permissions :

File permissions are an essential part of managing and maintaining a Linux system because they control the access to files and directories and ensure the security of the Linux system and the data stored in the files and directories.

```
ls -l
total 72
drwxr-xr-x  5 root root  4096 Aug 17 10:47 ReconDog
lrwxrwxrwx  1 root root    7 Jun  2 05:15 bin -> usr/bin
drwxr-xr-x  3 root root  4096 Aug 29 09:57 boot
drwxr-xr-x 17 root root 3420 Aug 30 10:12 dev
drwxr-xr-x 196 root root 12288 Aug 30 10:12 etc
drwxr-xr-x  5 root root  4096 Jul 22 18:16 home
lrwxrwxrwx  1 root root    28 Jun 12 05:20 initrd.img -> boot/initrd.img-6.8.11-amd64
lrwxrwxrwx  1 root root    28 Jun 12 05:20 initrd.img.old -> boot/initrd.img-6.6.15-amd64
lrwxrwxrwx  1 root root    7 Jun  2 05:15 lib -> usr/lib
lrwxrwxrwx  1 root root    9 Jun  2 05:35 lib32 -> usr/lib32
lrwxrwxrwx  1 root root    9 Jun  2 05:15 lib64 -> usr/lib64
drwx-----  2 root root 16384 Jun  2 05:15 lost+found
drwxr-xr-x  3 root root  4096 Jun  2 05:15 media
drwxr-xr-x  2 root root  4096 Jun  2 05:15 mnt
drwxr-xr-x  4 root root  4096 Aug 13 06:33 opt
dr-xr-xr-x 339 root root    0 Aug 30 10:12 proc
drwx----- 28 root root  4096 Aug 29 10:35 root
drwxr-xr-x 38 root root   980 Aug 30 10:12 run
lrwxrwxrwx  1 root root    8 Jun  2 05:15 sbin -> usr/sbin
drwxr-xr-x  9 root root  4096 Aug 10 06:26 snap
drwxr-xr-x  3 root root  4096 Jun  2 05:42 srv
dr-xr-xr-x 13 root root    0 Aug 30 10:12 sys
drwxrwxrwt 17 root root   440 Aug 30 10:17 tmp
drwxr-xr-x 16 root root  4096 Jun  3 05:14 usr
drwxr-xr-x 13 root root  4096 Jul 29 19:00 var
lrwxrwxrwx  1 root root    25 Jun 12 05:20 vmlinuz -> boot/vmlinuz-6.8.11-amd64
lrwxrwxrwx  1 root root    25 Jun 12 05:20 vmlinuz.old -> boot/vmlinuz-6.6.15-amd64
```

Above image I marked at 2 places with red line one is at top at 'l' so l represent the file is link file or we can say it is shortcut bin is shortcut which is inside usr/bin.

And second mark is at 'd' d represent it is directory.

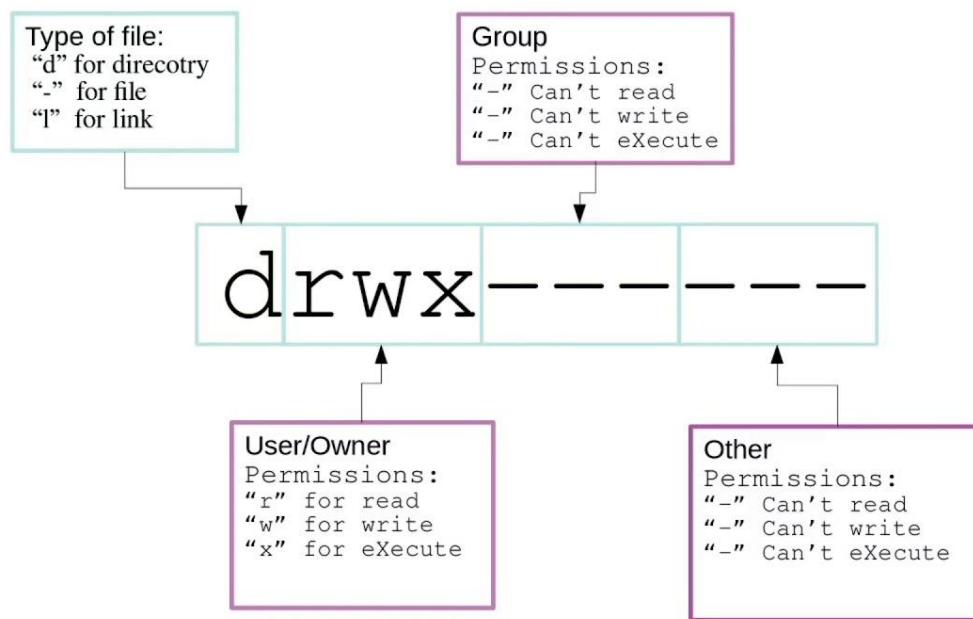
```
drwxr-xr-x  2 root root  4096 Jun 26 05:32 tmpfiles.d
-rw-r--r--  1 root root    938 Sep  9 2019 ts.conf
-rw-r--r--  1 root root  1260 Jan 27 2023 ucf.conf
drwxr-xr-x  4 root root  4096 Jul 17 05:34 udev
drwxr-xr-x  2 root root  4096 Aug 27 05:56 udisks2
drwxr-xr-x  3 root root  4096 Jun  2 05:23 ufw
```

If there is not d or l it is file other than directory or link file which can be any file.

Considering above picture there is also present "root root" which show username with root can only access that files or those are shows username and groupname.



Everything in one picture :



```
shum@sol:~$ ls -l
total 20
drwx----- 2 shum staff 4096 Jan 16 22:04 Mail
drwx----- 3 shum staff 4096 Jan 16 14:15 csc128
drwxr-xr-x 2 shum staff 4096 Jan 13 16:42 public
drwxr-xr-x 2 shum staff 4096 Jan 16 14:07 public_html
-rw-r--r-- 1 shum staff 628 Jan 15 20:04 verse
```

Annotations for the `ls -l` output:

- file type:** Points to the first character of the permission string (e.g., 'd' for directory).
- user (owner) name:** Points to the user name (e.g., 'shum').
- group name:** Points to the group name (e.g., 'staff').
- size:** Points to the file size in bytes (e.g., 4096).
- date/time last modified:** Points to the date and time (e.g., Jan 16 22:04).
- filename:** Points to the file name (e.g., Mail).
- number of hard links:** Points to the number before the user name (e.g., 2).
- permissions:**
  - user permissions:** Points to the first three characters of the permission string (e.g., 'rwx').
  - group permissions:** Points to the next three characters (e.g., '---').
  - other (everyone) permissions:** Points to the final three characters (e.g., '---').

Detailed breakdown of permissions (using `rwX` as an example):

- r:** readable
- w:** writeable
- X:** executable

```
-rw-r--r-- 1 root root 299 Jun 21 11:35 crack.txt
```

Above crack.txt file owner is root we can change ownership by using command chown

Chown stands for "change owner"

```
~ 10:57
> sudo chown meghank crack.txt
[sudo] password for meghank:

~ 10:57
> ls -lh crack.txt
-rw-r--r-- 1 meghank root 299 Jun 21 11:35 crack.txt
```

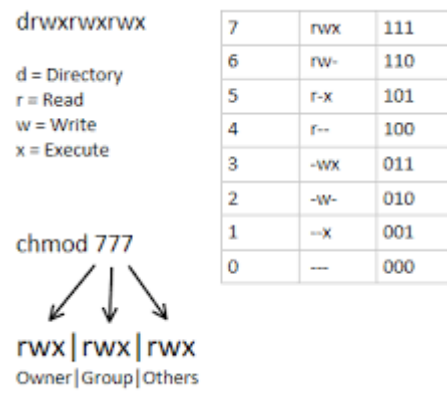
Here we are change user ownership only from root to Meghank user

### grep command :

grep searches for PATTERNS in each FILE. PATTERNS is one or more patterns separated by newline characters, and grep prints each line that matches a pattern. Typically PATTERNS should be quoted when grep is used in a shell command.

```
~ 10:43
> ls | grep d
Downloads
Videos
calendar.exe
data
demo.txt
encoder.exe
headers
hydra.restore
modified_vim.txt
nerd.txt
passwordlist.txt
practice_sed.txt
subd_for_ss.txt
```





Above picture ca easily clear picture for you about file permissions in linux files