Some important linux commands:

To know free memory:

```
rootjinesh@DESKTOP-KN25QO6:~$ free -h
               total        used        free      shared  buff/cache   available
Mem:            15Gi       421Mi        14Gi       3.0Mi       604Mi        14Gi
Swap:          4.0Gi          0B       4.0Gi
rootjinesh@DESKTOP-KN25QO6:~$
```

To check which program is taking most memory and then sort them on basis of memory in descending order:

```
rootjinesh@DESKTOP-KN25QO6:~$ ps aux --sort -%mem | head
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         280  0.3  0.4 2266484 77060 ?       Ssl  09:05   0:00 /usr/bin/dockerd -H fd:// --containerd=/run/container
d/containerd.sock
root         200  0.3  0.3 2244068 49536 ?       Ssl  09:05   0:00 /usr/bin/containerd
root         253  0.1  0.1 107160 21072 ?        Ssl  09:05   0:00 /usr/bin/python3 /usr/share/unattended-upgrades/unatt
ended-upgrade-shutdown --wait-for-signal
root         190  0.2  0.1  30080 19608 ?        Ss   09:05   0:00 /usr/bin/python3 /usr/bin/networkd-dispatcher --run-s
tartup-triggers
root          66  0.3  0.0  39636 15312 ?        S<s  09:05   0:00 /lib/systemd/systemd-journald
systemd+      94  0.3  0.0  26200 13096 ?        Ss   09:05   0:00 /lib/systemd/systemd-resolved
root           1  0.8  0.0 165932 11400 ?        Ss   09:05   0:01 /sbin/init
rootjin+     652  0.0  0.0  17076  9336 ?        Ss   09:05   0:00 /lib/systemd/systemd --user
root         194  0.2  0.0  15332  7384 ?        Ss   09:05   0:00 /lib/systemd/systemd-logind
rootjinesh@DESKTOP-KN25QO6:~$
```

Then we can also use curl to download content of file from internet.

To know stat of a file:

```
rootjinesh@DESKTOP-KN25QO6:~$ stat a.txt
  File: a.txt
  Size: 219             Blocks: 8          IO Block: 4096   regular file
Device: 820h/2080d      Inode: 8851        Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1000/rootjinesh)   Gid: ( 1000/rootjinesh)
Access: 2025-02-25 09:20:19.091710566 +0530
Modify: 2025-02-25 09:20:14.534856289 +0530
Change: 2025-02-25 09:20:14.534856289 +0530
 Birth: 2025-02-25 09:20:14.534856289 +0530
rootjinesh@DESKTOP-KN25QO6:~$
```

To see disk usage:

```
rootjinesh@DESKTOP-KN25QO6:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
none            7.8G     0  7.8G   0% /usr/lib/modules/5.15.167.4-microsoft-standard-WSL2
none            7.8G  4.0K  7.8G   1% /mnt/wsl
drivers         238G  206G   32G  87% /usr/lib/wsl/drivers
/dev/sdc       1007G  7.6G  949G   1% /
none            7.8G   80K  7.8G   1% /mnt/wslg
none            7.8G     0  7.8G   0% /usr/lib/wsl/lib
rootfs          7.8G  2.4M  7.8G   1% /init
none            7.8G  504K  7.8G   1% /run
none            7.8G     0  7.8G   0% /run/lock
none            7.8G     0  7.8G   0% /run/shm
tmpfs           4.0M     0  4.0M   0% /sys/fs/cgroup
none            7.8G   76K  7.8G   1% /mnt/wslg/versions.txt
none            7.8G   76K  7.8G   1% /mnt/wslg/doc
C:\             238G  206G   32G  87% /mnt/c
D:\             932G  230G  703G  25% /mnt/d
rootjinesh@DESKTOP-KN25QO6:~$
```

To install stress:

```
rootjinesh@DESKTOP-KN25QO6:~$ sudo apt install stress
[sudo] password for rootjinesh:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  stress
0 upgraded, 1 newly installed, 0 to remove and 4 not upgraded.
Need to get 18.4 kB of archives.
After this operation, 52.2 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/universe amd64 stress amd64 1.0.5-1 [18.4 kB]
Fetched 18.4 kB in 1s (34.1 kB/s)
Selecting previously unselected package stress.
(Reading database ... 66497 files and directories currently installed.)
Preparing to unpack .../stress_1.0.5-1_amd64.deb ...
Unpacking stress (1.0.5-1) ...
Setting up stress (1.0.5-1) ...
Processing triggers for man-db (2.10.2-1) ...
rootjinesh@DESKTOP-KN25QO6:~$
```

To now perform stress test:

```
rootjinesh@DESKTOP-KN25QO6:~$  stress --vm 1 --vm-bytes 256M --timeout 30s
stress: info: [896] dispatching hogs: 0 cpu, 0 io, 1 vm, 0 hdd
stress: info: [896] successful run completed in 30s
rootjinesh@DESKTOP-KN25QO6:~$
```

To see packages we have installed on our computer:

```
rootjinesh@DESKTOP-KN25QO6:~$ dpkg -l | grep mysql
```

Here we are specifically searching for packages related to sql.

Difference between upgrade and update:

`sudo apt update`

- **What it does:** This command refreshes your system's knowledge of available packages and their versions. Think of it like checking the menu at a restaurant. It doesn't bring you the food yet, but it tells you what's available.
- **How it works:** It contacts the software repositories (online sources for packages) that your system is configured to use. It downloads the latest information about the packages in those repositories, but it doesn't actually install or upgrade any software.
- **Why it's important:** You need to run `sudo apt update` before you install new software or upgrade existing software. It ensures that your system has the most up-to-date information about what's available.

`sudo apt upgrade`

- **What it does:** This command actually installs the newest versions of the packages that are already installed on your system. It's like ordering the food you saw on the menu.
- **How it works:** After you've run `sudo apt update`, your system knows about the latest versions of packages. `sudo apt upgrade` compares the versions of the packages you have installed with the latest versions available in the repositories. It then downloads and installs the updates for any packages that have newer versions.
- **Why it's important:** Running `sudo apt upgrade` regularly keeps your system up-to-date with the latest security patches, bug fixes, and new features.

Now to create a new user:

```
rootjinesh@DESKTOP-KN25QO6:~$ sudo adduser jinesh
[sudo] password for rootjinesh:
Adding user `jinesh' ...
Adding new group `jinesh' (1001) ...
Adding new user `jinesh' (1001) with group `jinesh' ...
Creating home directory `/home/jinesh' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for jinesh
Enter the new value, or press ENTER for the default
        Full Name []: Jinesh Ranawat
        Room Number []: 1
        Work Phone []: 1
        Home Phone []: 1
        Other []: 1
  the information correct? [Y/n] Y
rootjinesh@DESKTOP-KN25QO6:~$
```

Now to switch user:

```
rootjinesh@DESKTOP-KN25QO6:~$ su - jinesh
Password:
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 5.15.167.4-microsoft-standard-WSL2 x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Tue Feb 25 09:55:51 IST 2025

  System load:  0.08               Processes:             35
  Usage of /:   0.8% of 1006.85GB  Users logged in:       1
  Memory usage: 3%                 IPv4 address for eth0: 172.17.71.253
  Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

This message is shown once a day. To disable it please create the
/home/jinesh/.hushlogin file.
jinesh@DESKTOP-KN25QO6:~$
```

To delete a user:

```
rootjinesh@DESKTOP-KN25QO6:~$ ps -u jinesh
    PID TTY          TIME CMD
rootjinesh@DESKTOP-KN25QO6:~$ userdel -r jinesh
userdel: Permission denied.
userdel: cannot lock /etc/passwd; try again later.
rootjinesh@DESKTOP-KN25QO6:~$ sudo userdel -r jinesh
[sudo] password for rootjinesh:
userdel: jinesh mail spool (/var/mail/jinesh) not found
rootjinesh@DESKTOP-KN25QO6:~$ su - jinesh
su: user jinesh does not exist or the user entry does not contain all the required fields
rootjinesh@DESKTOP-KN25QO6:~$
```

Now to create a user without specifying password and then if we want to set password for user:

```
rootjinesh@DESKTOP-KN25QO6:~$ sudo pkill -9 -u jinesh
pkill: invalid user name: jinesh
rootjinesh@DESKTOP-KN25QO6:~$ sudo useradd -m jineshtry
rootjinesh@DESKTOP-KN25QO6:~$ sudo passwd jineshtry
New password:
Retype new password:
passwd: password updated successfully
rootjinesh@DESKTOP-KN25QO6:~$
```

To see details like when was the user logged in:

```
rootjinesh@DESKTOP-KN25QO6:~$ sudo apt install finger
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  finger
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 16.9 kB of archives.
After this operation, 51.2 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/universe amd64 finger amd64 0.17-17 [16.9 kB]
Fetched 16.9 kB in 1s (32.5 kB/s)
Selecting previously unselected package finger.
(Reading database ... 66497 files and directories currently installed.)
Preparing to unpack .../finger_0.17-17_amd64.deb ...
Unpacking finger (0.17-17) ...
Setting up finger (0.17-17) ...
Processing triggers for man-db (2.10.2-1) ...
rootjinesh@DESKTOP-KN25QO6:~$ finger rootjinesh
Login: rootjinesh                       Name:
Directory: /home/rootjinesh             Shell: /bin/bash
On since Tue Feb 25 09:05 (IST) on pts/1   58 minutes 20 seconds idle
     (messages off)
No mail.
No Plan.
rootjinesh@DESKTOP-KN25QO6:~$
```

related to user informations

```
sudo usermod -a -G c406cohort jineshtrying
sudo groupadd c406cohort
sudo usermod -a -G c406cohort jineshtrying
history
cd /etc/
ls -lrt
sudo cat group
cd /home
sudo mkdir -p /home/grouptry/jinesh/a
sudo chmod 400 /home/grouptry/jinesh/a
sudo chmod 400 /home/grouptry
sudo chgrp c406cohort /home/grouptry
sudo chgrp c406cohort /home/grouptry/jinesh/a
ls -lrt
chmod 040 grouptry/
sudo chmod 040 grouptry/
sudo chmod 2775 grouptry/
history
```

- **Use `chmod` when:**
  - You want to change the read, write, or execute permissions for the owner, group, or others.
  - You want to fine-tune who can access a file and what they can do with it.
  - Examples:
    - Making a file executable (`chmod +x myfile.sh`).
    - Giving a group write access to a directory (`chmod g+w mydirectory`).
    - Restricting access to a sensitive file (`chmod 600 sensitive.txt`).

- **Use `chown` when:**
  - You need to change the owner of a file (e.g., after creating it as root).
  - You want to change the group associated with a file (e.g., to allow a specific group access).
  - Examples:
    - Taking ownership of a file you created (`chown yourusername myfile.txt`).
    - Giving ownership of a shared directory to a specific group (`chown :sharedgroup mydirectory`).

Information about groups:

In Linux, a group is a way to manage permissions for multiple users at once. Instead of assigning permissions to each user individually, you can create a group, assign users to that group, and then set permissions for the group on files and directories. This simplifies administration, especially when dealing with many users who need similar access levels.

2. Creating a Group:

You use the `groupadd` command:

```bash
sudo groupadd <group_name>
```

- `sudo` : Necessary for administrator privileges.
- `groupadd` : The command to create a group.
- `<group_name>` : The name you want to give to the group (e.g., "developers", "students", "sales").

Example:

```bash
sudo groupadd webadmins
```

3. Adding Users to a Group:

You use the `usermod` command:

```
Bash
sudo usermod -aG <group_name> <username>
```

- `sudo`: Administrator privileges required.
- `usermod`: The command to modify user settings.
- `-aG`: `-a` appends the user to the group (important!), and `-G` specifies the group. Without `-a`, the user would be *removed* from any other secondary groups.
- `<group_name>`: The name of the group you want to add the user to.
- `<username>`: The username of the user you want to add.

4. Setting Permissions for a Group:

You use the `chmod` command:

```
Bash
sudo chmod g<permissions> <file_or_directory>
```

- `sudo`: Administrator privileges needed.
- `chmod`: The command to change file permissions.
- `g<permissions>`: Specifies the permissions for the *group*. Here's how to specify the permissions:
  - `r`: Read permission.
  - `w`: Write permission.
  - `x`: Execute permission.
  - `+`: Add a permission.
  - `-`: Remove a permission.
  - `=`: Set the permission exactly (removes any other permissions).

Examples:

- **Give the group read and write access to a file:**

```
Bash
sudo chmod g+rw myfile.txt
```

- **Give the group read-only access to a directory and its contents (recursively):**

```
Bash
sudo chmod -R g+r,g-w,o-rwx mydirectory/
```

5. Setting Ownership (Important for Group Permissions):

For the group permissions to work correctly, the file or directory needs to be *owned* by the group. You use the `chown` command for this:

Bash

```
sudo chown :<group_name> <file_or_directory>
```

The colon `:` before the group name means "change only the group; leave the owner as is." If you need to change both, provide both:

Bash

```
sudo chown <username>:<group_name> <file_or_directory>
```

Bash

```
sudo chown :webadmins website_files/
```

Complete Example:

Let's say you want to create a group called "editors", add users "alice" and "bob" to it, and give them read and write access to a directory called "documents":

Bash

```
sudo groupadd editors
sudo usermod -aG editors alice
sudo usermod -aG editors bob
sudo chown :editors documents/
sudo chmod -R g+rw,o-rwx documents/
```

This will create the group, add the users, make the "documents" directory owned by the "editors" group, and give the group read and write access (and remove all permissions for others).

About set gid command:

Key takeaway: The Setgid bit

The most significant part of this command is the `2` in the first digit, which sets the *setgid* bit. When the setgid bit is set on a directory:

1. **New files and subdirectories created within that directory inherit the group ownership of the directory, not the group ownership of the user who created them.** This is crucial for shared directories where multiple users in the same group need to collaborate.
2. **It also affects how `chmod` behaves.** When the setgid bit is set, and a new file or subdirectory is created, the group ID of the new file or subdirectory is set to the group ID of the containing directory.

Let's break down each of these commands and explain the syntax and keys involved in SSH key generation and usage.

1. ssh-keygen -t rsa -b 4096 -C "ranawatjinesh@gmail.com"

* **ssh-keygen:** This is the command-line utility for generating, managing, and converting SSH keys.

* **-t rsa:** This option specifies the type of key to generate. RSA is a widely used and secure algorithm. Other options exist (like ed25519), but RSA is a good default.

* **-b 4096:** This sets the key length (number of bits). 4096 bits is a strong key size, providing good security. While 2048 bits is still generally considered secure, 4096 is recommended for new keys. Larger key sizes offer more security but can be slightly slower.

* **-C "ranawatjinesh@gmail.com":** This adds a comment to the key. It's a good practice to include your email address or some other identifier so you can easily distinguish between different keys you might have. This comment is stored within the public key.

* *Result:* This command generates two files:

  * id_rsa: This is your private key. *Keep this file secret!* Never share it with anyone.

  * id_rsa.pub: This is your public key. You'll share this key with the servers you want to access.

2. cd .ssh/

* **cd:** This is the command to change the current directory.

* **.ssh/:** This is the standard directory where SSH keys and configuration files are stored. It's usually located in your home directory (e.g., /home/yourusername/.ssh).

3. ls -lrt

* **ls:** This is the command to list files and directories

* **-l:** This option displays the output in a long listing format, showing details like permissions, owner, group, size, and modification time.

* **-r:** This option reverses the order of the listing.

* **-t:** This option sorts the listing by modification time, with the most recently modified files listed last.

* *Together:* ls -lrt lists the files in the current directory (which is now .ssh/) in a long listing format, sorted by modification time (newest last).

4. less id_rsa.pub

* **less:** This is a pager program that allows you to view the contents of a file one page at a time. It's useful for viewing long files.

* **id_rsa.pub:** This is your public key file. You'll copy the contents of this file and add it to the authorized_keys file on the remote server you want to access.

5. less id_rsa

* **less:** Same as above.

* **id_rsa:** This is your private key file. You should never share the contents of this file.

6. cd

* **cd (without any arguments):** This changes the current directory back to your home directory.

7. eval "$(ssh-agent -s)"

* **eval:** This command evaluates a string as a command.

* **ssh-agent:** This is a program that holds your private keys in memory so you don't have to enter the passphrase every time you connect to a server.

* **-s:** This option tells ssh-agent to output shell commands that set the necessary environment variables.

* **$(...):** This is command substitution. It executes the command inside the parentheses and captures its output.

* *Together:* ssh-agent -s outputs commands to set environment variables. The eval command then executes those commands, which starts the ssh-agent and makes it available in your current shell session. This is crucial for ssh-add to work.

8. sudo apt install openssh-client

* **sudo:** Executes the command with root privileges.

* **apt install openssh-client:** This installs the OpenSSH client package, which includes the ssh command, ssh-agent, and other related utilities. This is necessary if you don't already have these tools installed.

9. eval "{ssh-agent -s}"

* *Same as command 7:* This starts the ssh-agent again. It's likely redundant in this sequence of commands, as it was already run earlier. However, it ensures that ssh-agent is running before you try to add your key.

Key takeaway: The core idea is to generate a key pair (ssh-keygen), start the ssh-agent (eval "$(ssh-agent -s)"), add your private key to the agent (ssh-add), and then copy the public key (id_rsa.pub) to the remote server's authorized_keys file. This allows you to log in to the server without a password.

Now to follow the above mentioned steps:

To create a key:

```
rootjinesh@DESKTOP-KN25QO6:~$ ssh-keygen -t ed25519 -C "ranawatjinesh@gmail.com"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/rootjinesh/.ssh/id_ed25519):
Created directory '/home/rootjinesh/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/rootjinesh/.ssh/id_ed25519
Your public key has been saved in /home/rootjinesh/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:sNHfDHkzT1Z3FAK+wTUvulIUsmZyUl9d8IxtJ6Xdgfo ranawatjinesh@gmail.com
The key's randomart image is:
+--[ED25519 256]--+
|          o o.=+=X|
|         .. *.+.+@=|
|       oo.=o*=.*.O|
|        ++..***.o.|
|       . S .++ .  |
|           . .E   |
|          . .     |
|           .      |
|                  |
+----[SHA256]-----+
rootjinesh@DESKTOP-KN25QO6:~$
```

To evaluate if ssh is working or not:

```
rootjinesh@DESKTOP-KN25QO6:~$ eval "$(ssh-agent -s)"
Agent pid 2969
rootjinesh@DESKTOP-KN25QO6:~$
```

Now we have started ssh agent on our machine and now we would add our private key file:

```
rootjinesh@DESKTOP-KN25QO6:~$ ssh-add ~/.ssh/id_ed25519
Identity added: /home/rootjinesh/.ssh/id_ed25519 (ranawatjinesh@gmail.com)
rootjinesh@DESKTOP-KN25QO6:~$
```

```
rootjinesh@DESKTOP-KN25QO6:~$ ssh-keygen -t ed25519 -C "ranawatjinesh@gmail.com"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/rootjinesh/.ssh/id_ed25519):
Created directory '/home/rootjinesh/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/rootjinesh/.ssh/id_ed25519
Your public key has been saved in /home/rootjinesh/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:sNHfDHkzT1Z3FAK+wTUvulIUsmZyUl9d8IxtJ6Xdgfo ranawatjinesh@gmail.com
The key's randomart image is:
+--[ED25519 256]--+
|          o o.=+=X|
|         .. *.+.+@=|
|       oo.=o*=.*.O|
|        ++..***.o.|
|       . S .++ .  |
|           . .E   |
|          . .     |
|           .      |
|                  |
+----[SHA256]-----+
rootjinesh@DESKTOP-KN25QO6:~$ eval "$(ssh-agents -s)"
Command 'ssh-agents' not found, did you mean:
  command 'ssh-agent' from deb openssh-client (1:8.9p1-3ubuntu0.10)
Try: sudo apt install <deb name>
rootjinesh@DESKTOP-KN25QO6:~$ eval "$(ssh-agent -s)"
Agent pid 2969
rootjinesh@DESKTOP-KN25QO6:~$ ssh-add ~/.ssh/id_ed25519
Identity added: /home/rootjinesh/.ssh/id_ed25519 (ranawatjinesh@gmail.com)
rootjinesh@DESKTOP-KN25QO6:~$ cat ~/.s
.second.sh.swp           .selected_editor         .ssh/                    .sudo_as_admin_successful
rootjinesh@DESKTOP-KN25QO6:~$ cat ~/.ssh/id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIMOpNZdw/8lYCFt5BUqywgKr+cHtWKXoUB8SJ2Zfe8eg ranawatjinesh@gmail.com
rootjinesh@DESKTOP-KN25QO6:~$
```

```
  887  history
rootjinesh@DESKTOP-KN25QO6:~$ stty -a
speed 38400 baud; rows 41; columns 156; line = 0;
intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol = <undef>; eol2 = <undef>; swtch = <undef>; start = ^Q; stop = ^S; susp = ^Z; rprnt = ^R;
werase = ^W; lnext = ^V; discard = ^O; min = 1; time = 0;
-parenb -parodd -cmspar cs8 -hupcl -cstopb cread -clocal -crtscts
-ignbrk -brkint -ignpar -parmrk -inpck -istrip -inlcr -igncr icrnl ixon -ixoff -iuclc -ixany -imaxbel -iutf8
opost -olcuc -ocrnl onlcr -onocr -onlret -ofill -ofdel nl0 cr0 tab0 bs0 vt0 ff0
isig icanon iexten echo echoe echok -echonl -noflsh -xcase -tostop -echoprt echoctl echoke -flusho -extproc
rootjinesh@DESKTOP-KN25QO6:~$
```

To change display setting:

```
rootjinesh@DESKTOP-KN25QO6:~$ stty -a
speed 38400 baud; rows 41; columns 156; line = 0;
intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol = <undef>; eol2 = <undef>; swtch = <undef>; start = ^Q; stop = ^S; susp = ^Z; rprnt = ^R;
werase = ^W; lnext = ^V; discard = ^O; min = 1; time = 0;
-parenb -parodd -cmspar cs8 -hupcl -cstopb cread -clocal -crtscts
-ignbrk -brkint -ignpar -parmrk -inpck -istrip -inlcr -igncr icrnl ixon -ixoff -iuclc -ixany -imaxbel -iutf8
opost -olcuc -ocrnl onlcr -onocr -onlret -ofill -ofdel nl0 cr0 tab0 bs0 vt0 ff0
isig icanon iexten echo echoe echok -echonl -noflsh -xcase -tostop -echoprt echoctl echoke -flusho -extproc
rootjinesh@DESKTOP-KN25QO6:~$ stty rows 26 row columns 158
stty: invalid argument 'row'
Try 'stty --help' for more information.
rootjinesh@DESKTOP-KN25QO6:~$ stty 26 row columns 158
stty: invalid argument '26'
Try 'stty --help' for more information.
rootjinesh@DESKTOP-KN25QO6:~$ stty rows 26  columns 158
rootjinesh@DESKTOP-KN25QO6:~$ stty -a
speed 38400 baud; rows 26; columns 158; line = 0;
intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol = <undef>; eol2 = <undef>; swtch = <undef>; start = ^Q; stop = ^S; susp = ^Z; rprnt = ^R;
werase = ^W; lnext = ^V; discard = ^O; min = 1; time = 0;
-parenb -parodd -cmspar cs8 -hupcl -cstopb cread -clocal -crtscts
-ignbrk -brkint -ignpar -parmrk -inpck -istrip -inlcr -igncr icrnl ixon -ixoff -iuclc -ixany -imaxbel -iutf8
opost -olcuc -ocrnl onlcr -onocr -onlret -ofill -ofdel nl0 cr0 tab0 bs0 vt0 ff0
isig icanon iexten echo echoe echok -echonl -noflsh -xcase -tostop -echoprt echoctl echoke -flusho -extproc
rootjinesh@DESKTOP-KN25QO6:~$
```

To send a file to printer : “lpr” (if we haven’t installed then we may need to install it.)

To do kernel level coding:

```
dmesg
dmesg | tail -50
dmesg | grep "USB"
```

For networking related things:

```
rootjinesh@DESKTOP-KN25QO6:~$ iperf
Usage: iperf [-s|-c host] [options]
Try `iperf --help' for more information.
rootjinesh@DESKTOP-KN25QO6:~$ iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size:  128 KByte (default)
------------------------------------------------------------
```

To know about data transfer:

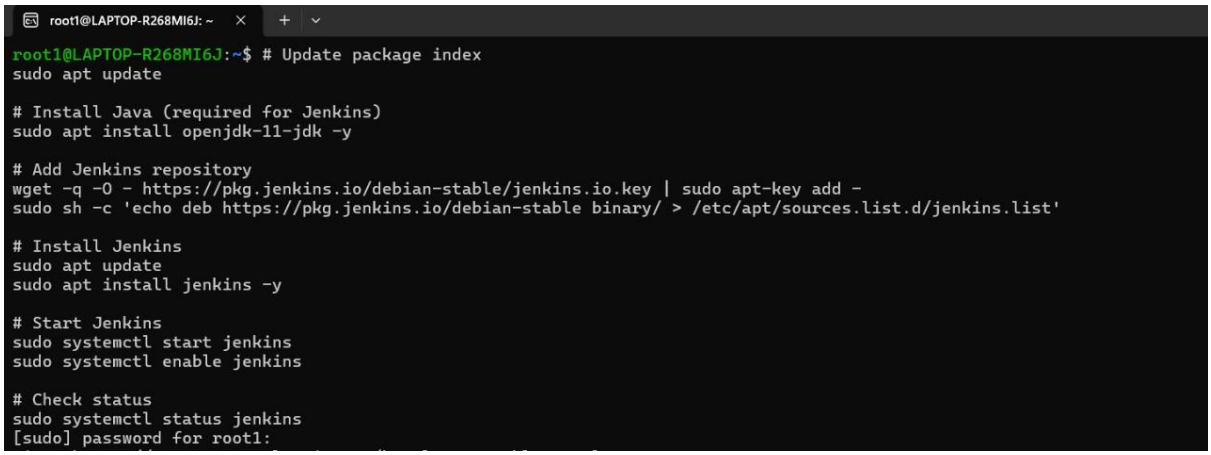
```
rootjinesh@DESKTOP-KN25QO6:~$ sudo tcpdump -i any
[sudo] password for rootjinesh:
tcpdump: data link type LINUX_SLL2
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel
```

Using dig we can get DNS server details:

```
rootjinesh@DESKTOP-KN25QO6:~$ dig google.com

; <<>> DiG 9.18.30-0ubuntu0.22.04.2-Ubuntu <<>> google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 62562
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;google.com.                    IN      A

;; ANSWER SECTION:
google.com.             26      IN      A       142.250.192.142

;; Query time: 0 msec
;; SERVER: 10.255.255.254#53(10.255.255.254) (UDP)
;; WHEN: Tue Feb 25 12:00:14 IST 2025
;; MSG SIZE  rcvd: 44
```

To install Jenkins in wsl:

```
# Update package index

sudo apt update

# Install Java (required for Jenkins)

sudo apt install openjdk-11-jdk -y

# Add Jenkins repository

wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add -

sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'

# Install Jenkins

sudo apt update

sudo apt install jenkins -y

# Start Jenkins

sudo systemctl start jenkins

sudo systemctl enable jenkins

# Check status

sudo systemctl status Jenkins

# Download the Jenkins repository key

curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee /usr/share/keyrings/jenkins-keyring.asc > /dev/null

# Set up the Jenkins repository with proper key verification

echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] https://pkg.jenkins.io/debian-stable binary/" | sudo tee /etc/apt/sources.list.d/jenkins.list > /dev/null
```



```
E: Package 'jenkins' has no installation candidate
Failed to start jenkins.service: Unit jenkins.service not found.
Failed to enable unit: Unit file jenkins.service does not exist.
Unit jenkins.service could not be found.
root1@LAPTOP-R268MI6J:~$ # Download the Jenkins repository key
curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee /usr/share/keyrings/jenkins-keyring.asc > /dev/null

# Set up the Jenkins repository with proper key verification
echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] https://pkg.jenkins.io/debian-stable binary/" | sudo tee /etc/apt/sources.list.d/jenkins.list
> /dev/null
root1@LAPTOP-R268MI6J:~$ # Update package lists
sudo apt update

# Install Java (required for Jenkins)
sudo apt install fontconfig openjdk-17-jre -y

# Install Jenkins
sudo apt install jenkins -y
Hit:1 https://download.docker.com/linux/ubuntu noble InRelease
Hit:2 http://archive.ubuntu.com/ubuntu noble InRelease
```

and to get the password of Jenkins, and store it in some file.

```
root1@LAPTOP-R268MI6J:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
c9ca3453bdfc4bbfb65dd7506a6cdf02
root1@LAPTOP-R268MI6J:~$
```

To add Jenkins in docker:

Add Jenkins user to Docker group (so Jenkins can run Docker commands)

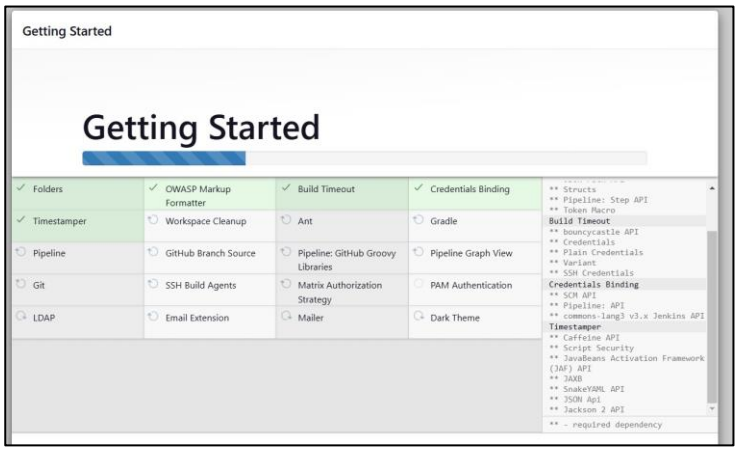
sudo usermod -aG docker jenkins

Restart Jenkins to apply group changes

Sudo systemctl restart Jenkins

```
root1@LAPTOP-R268MI6J:~$ sudo usermod -aG docker jenkins
root1@LAPTOP-R268MI6J:~$ sudo systemctl restart jenkins
root1@LAPTOP-R268MI6J:~$
```

- after adding Jenkins user we can run this command to verify jenkins and would paste our password here
- http://localhost:8080/login?from=%2F
- pw: c9ca3453bdfc4bbfb65dd7506a6cdf02
- then u'll select the installed plugins



And then after installation we just paste our name and password and it would open Jenkins for us: