

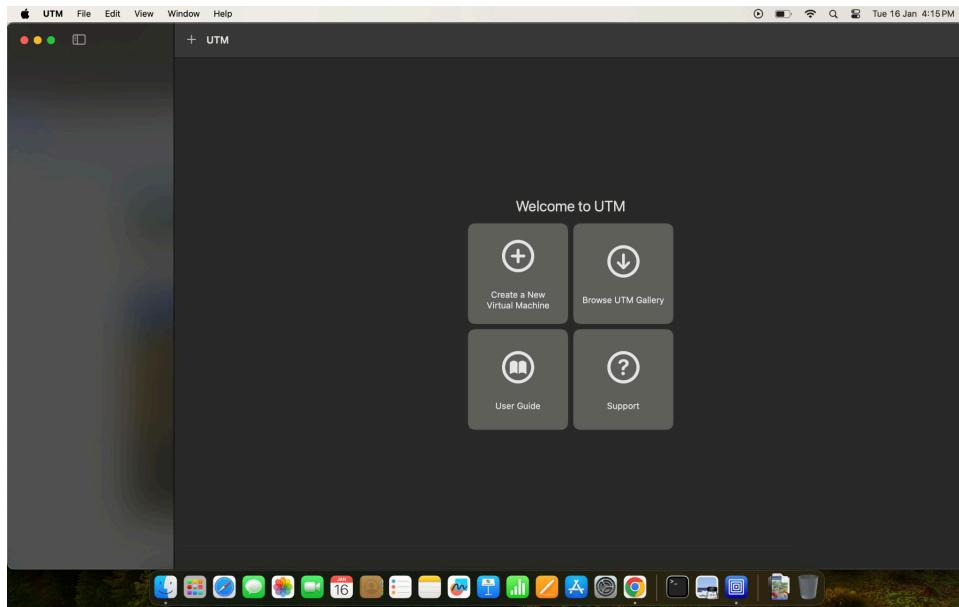
1)

a) Knowing the basics of virtual box.

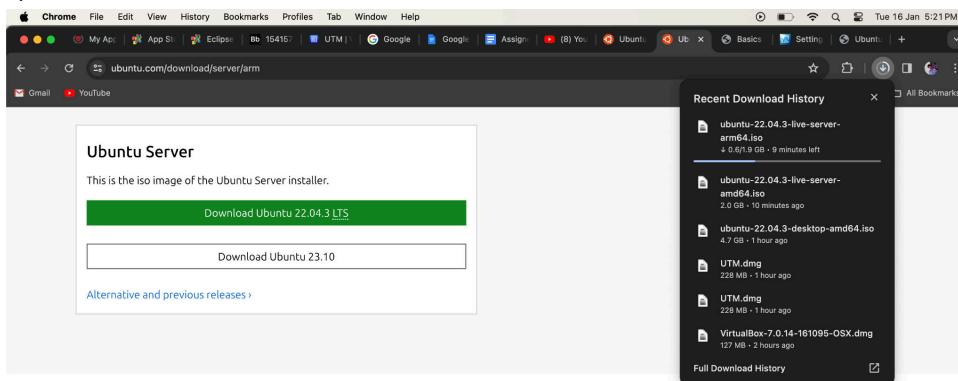
b)



c)



d)



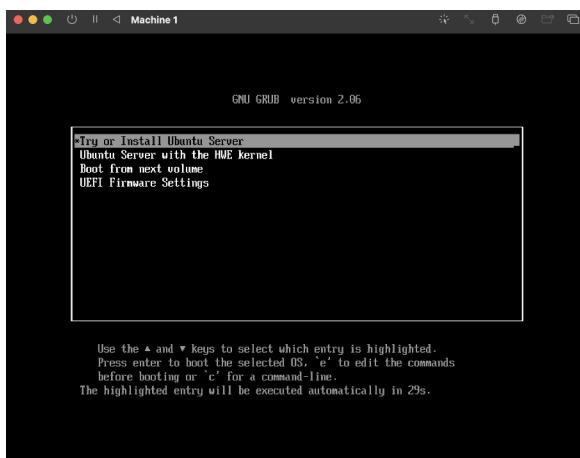
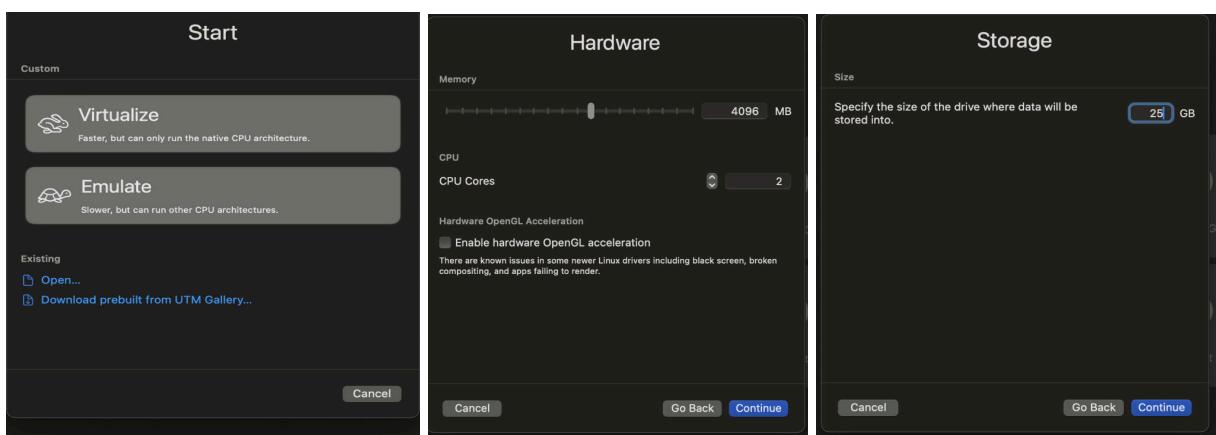
Containers, databases, web and more

Ubuntu Server for ARM includes everything you are looking for in a server operating system, including:

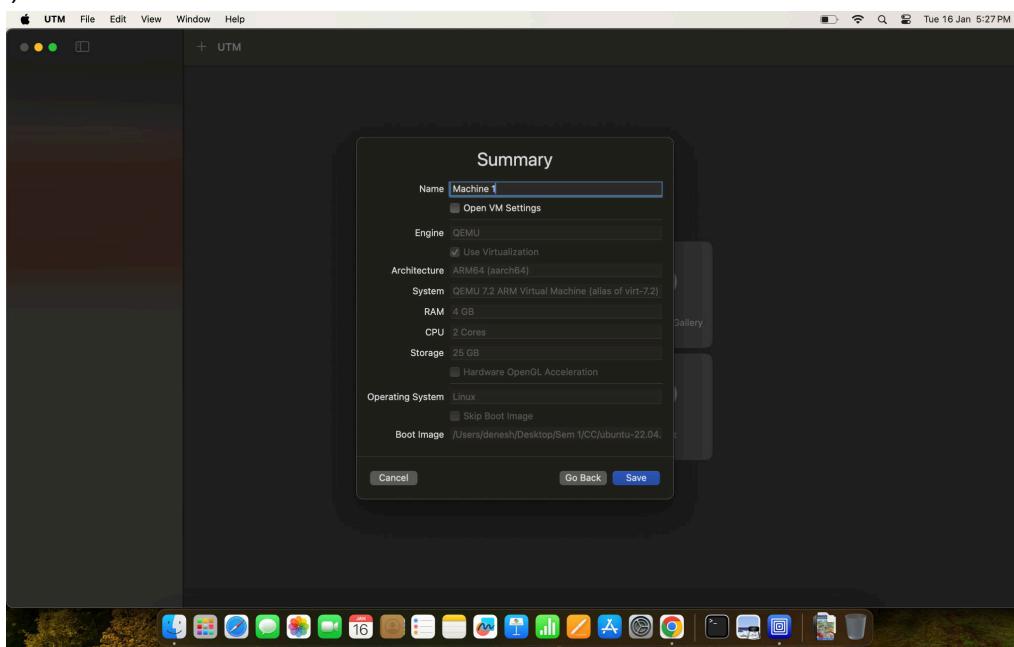
- The LXD container hypervisor, giving you instant access to isolated, secured environments running with bare metal performance.



e)



f)



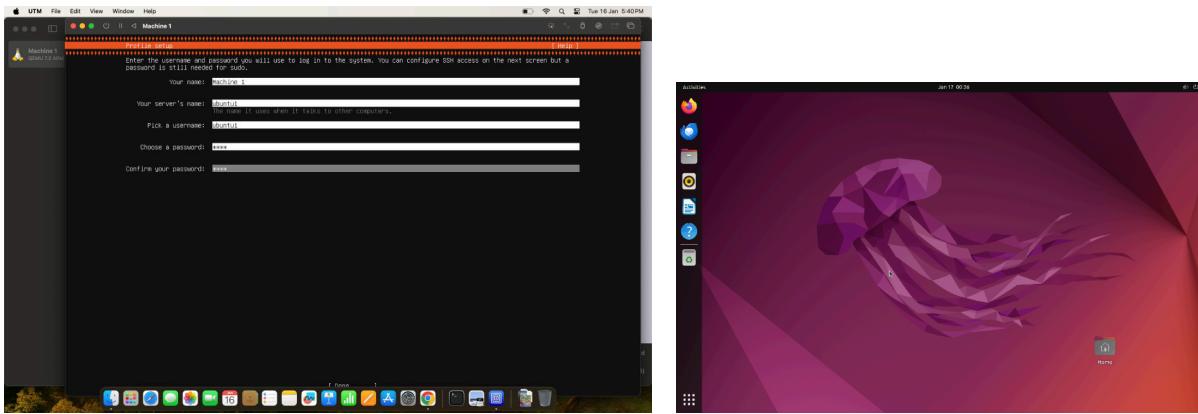
```

f) + UTM Tue 16 Jan 5:27 PM
Machine 1

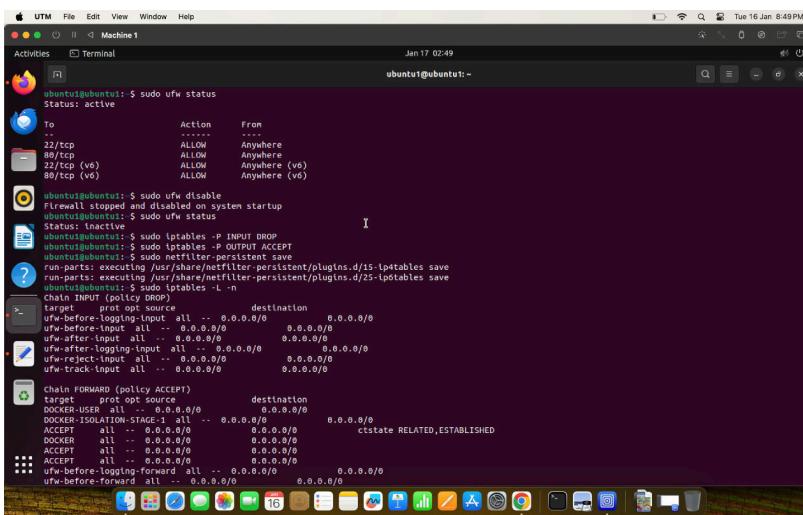
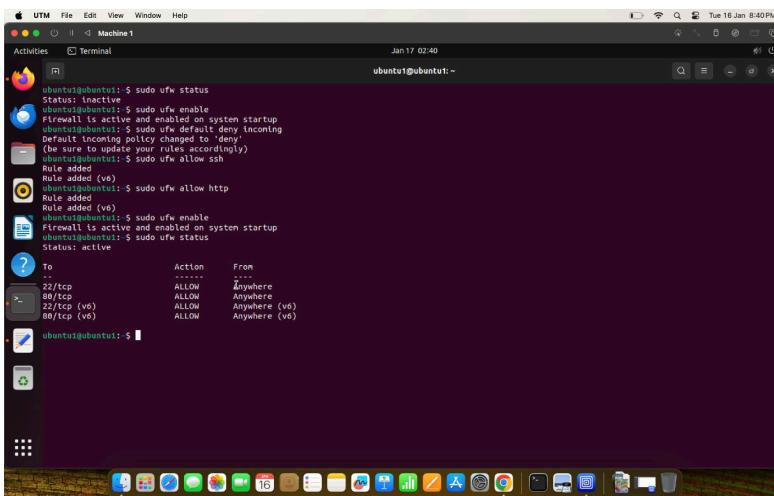
Get: 791 http://ports.ubuntu.com/ubuntu-ports jammy-updates/main arm64 libcanberra-pulse arm64 0.30-10ubuntu1.22.04.1 [11.4 kB]
Get: 791 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libcanberra-pulse arm64 0.30-10ubuntu1.22.04.1 [11.4 kB]
Get: 792 http://ports.ubuntu.com/ubuntu-ports jammy-updates/main arm64 libcurl4-arm64-openssl arm64 2.3.3.4-0ubuntu5 [490 kB]
Get: 793 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libcurl4-arm64-openssl arm64 2.3.3.4-0ubuntu5 [490 kB]
Get: 794 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libcurl4-arm64-openssl arm64 2.3.3.4-0ubuntu5 [494.4 kB]
Get: 795 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libcurl4-common arm64 1.29.4-0fss-40u1d1 [4,582 kB]
Get: 796 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libcurl4-common arm64 1.29.4-0fss-40u1d1 [4,582 kB]
Get: 797 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libcurl4-openssl45 arm64 1.29.4-0fss-40u1d1 [10.2 kB]
Get: 798 http://ports.ubuntu.com/ubuntu-ports jammy-updates/main arm64 libcurl4-openssl45 arm64 1.29.4-0fss-40u1d1 [10.2 kB]
Get: 799 http://ports.ubuntu.com/ubuntu-ports jammy-updates/main arm64 libcurl4-openssl45 arm64 1.29.4-0fss-40u1d1 [10.2 kB]
Get: 800 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libdante-dump-perl all 1.25-1 [25.9 kB]
Get: 801 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libdante-patching-3.0-2 arm64 2.3.41-0build2 [60.6 kB]
Get: 801 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libdanteconf arm64 3-0.5afesgrncl-build2 [15.6 kB]
Get: 802 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libdanteconf-common arm64 3-0.5afesgrncl-build2 [15.6 kB]
Get: 803 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libdanteconf-common arm64 3-0.5afesgrncl-build2 [15.6 kB]
Get: 804 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libdanteconf-common arm64 3-0.5afesgrncl-build2 [15.6 kB]
Get: 805 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libde-book-0.1-1 arm64 0.1.3-20u1d2 [14.1 kB]
Get: 806 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libencodings-locale-perl all 1.05-1 [11.9 kB]
Get: 807 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libeuro arm64 0.01-0build1 [28.5 kB]
Get: 808 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libeurobeam-0.1-4 arm64 0.01-0build1 [114 kB]
Get: 808 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libeurobeam-0.1-4 arm64 0.01-0build1 [114 kB]
Get: 809 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libenvironc 0.2.0-10u1d2 [9,039 kB]
Get: 810 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libespeak-ng arm64 1.50-0fss-10 [206 kB]
Get: 811 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libetonerk-0.1-1 arm64 0.1.10-30u1d1 [1,239 kB]
Get: 812 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libevent-2.1-7 arm64 2.1.12-stable-10u1d2 [14.1 kB]
Get: 813 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libext2fs-extractfs-all arm64 1.4.3-10u1d2 [14.1 kB]
Get: 814 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libhexagon-0.1-1 arm64 0.1.2-38u1d2 [655 kB]
Get: 815 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libhttp-system-simple-perl all 1.30-1 [23.2 kB]
Get: 816 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libhttp-basedir-perl all 0.09-1 [15.7 kB]
Get: 817 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libluri-perl all 5.10-1 [70.8 kB]
Get: 818 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libluri-desktopentry-perl all 5.62-2 [17.0 kB]
Get: 819 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libluri-distro-perl all 6.05-1 [34.0 kB]
Get: 820 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libluri-listing-perl all 6.14-1 [11.2 kB]
Get: 821 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libluri-npmenvinfo-perl all 0.31-1 [43.5 kB]
Get: 822 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libluri-npmenvinfo-perl all 0.31-1 [43.6 kB]
Get: 823 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libfont-afm-perl all 1.20-3 [13.6 kB]
Get: 824 http://ports.ubuntu.com/ubuntu-ports jammy-updates/main arm64 libfontconfig1 arm64 2.13.3-0ubuntu1.22.04.05 [277 kB]
Get: 825 http://ports.ubuntu.com/ubuntu-ports jammy-updates/main arm64 libfontconfig1 arm64 2.13.3-0ubuntu1.22.04.05 [277 kB]
Get: 826 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libfontconfig1 arm64 2.13.3-0ubuntu1.22.04.05 [277 kB]
Get: 827 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libfontconfig1 arm64 2.24.33-20ubuntu2 [14.8 kB]
Get: 828 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libgail-common arm64 2.24.33-20ubuntu2 [128 kB]
Get: 829 http://ports.ubuntu.com/ubuntu-ports jammy-updates/main arm64 libgdk-pixbuf2.0-bin arm64 2.42.0-0fss-ubuntu2.0 [14.1 kB]
Get: 830 http://ports.ubuntu.com/ubuntu-ports jammy-updates/main arm64 libglx-amber-drivers arm64 21.3.3-0ubuntu1.22.04.1 [1,850 kB]
Get: 831 http://ports.ubuntu.com/ubuntu-ports jammy-updates/main arm64 libglx-amber-drivers arm64 21.3.3-0ubuntu1.22.04.1 [1,850 kB]
Get: 832 http://ports.ubuntu.com/ubuntu-ports jammy-updates/main arm64 libgpg-error arm64 1.16.0-1.20ubuntu0.1 [105 kB]
Get: 833 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libgxphoto2-110m all 2.5.27-10u1d2 [14.0 kB]
Get: 834 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libgcode arm64 0.8.3-16u11d2 [207 kB]
Get: 835 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libgcode-common arm64 0.8.3-16u11d2 [23.1 kB]
Get: 836 http://ports.ubuntu.com/ubuntu-ports jammy/main arm64 libgssapi-sasl-2.0-20 arm64 1.4.0-1-20u1d1 [47.7 kB]
Get: 837 http://ports.ubuntu.com/ubuntu-ports jammy-updates/main arm64 libigfxr3d arm64 2.24.33-10ubuntu1.22.04.05 [63.4 kB]
Get: 838 http://ports.ubuntu.com/ubuntu-ports jammy-updates/main arm64 liblightdm-4-bin arm64 4.0.3-3ds-10ubuntu1.22.04.1 [2,774 kB]
62% [838 libgtk4-4-bin 1,394 kB/2,774 kB 50%]
2.725 kB/s 1min 37s_

```

g)



h)



i)

UTM File Edit View Window Help

Machine 1

Activities Terminal Jan 17 02:49 ubuntu1@ubuntu1:~

```
Chain ufw-after-output (1 references)
target  prot opt source      destination
Chain ufw-before-forward (1 references)
target  prot opt source      destination
Chain ufw-before-input (1 references)
target  prot opt source      destination
Chain ufw-before-logging-forward (1 references)
target  prot opt source      destination
Chain ufw-before-logging-input (1 references)
target  prot opt source      destination
Chain ufw-before-logging-output (1 references)
target  prot opt source      destination
Chain ufw-before-output (1 references)
target  prot opt source      destination
Chain ufw-reject-forward (1 references)
target  prot opt source      destination
Chain ufw-reject-input (1 references)
target  prot opt source      destination
Chain ufw-reject-output (1 references)
target  prot opt source      destination
Chain ufw-track-forward (1 references)
target  prot opt source      destination
Chain ufw-track-input (1 references)
target  prot opt source      destination
Chain ufw-track-output (1 references)
target  prot opt source      destination
```

Machine 1

File Applications Dock

j)

UTM File Edit View Search Terminal Help

Machine 2

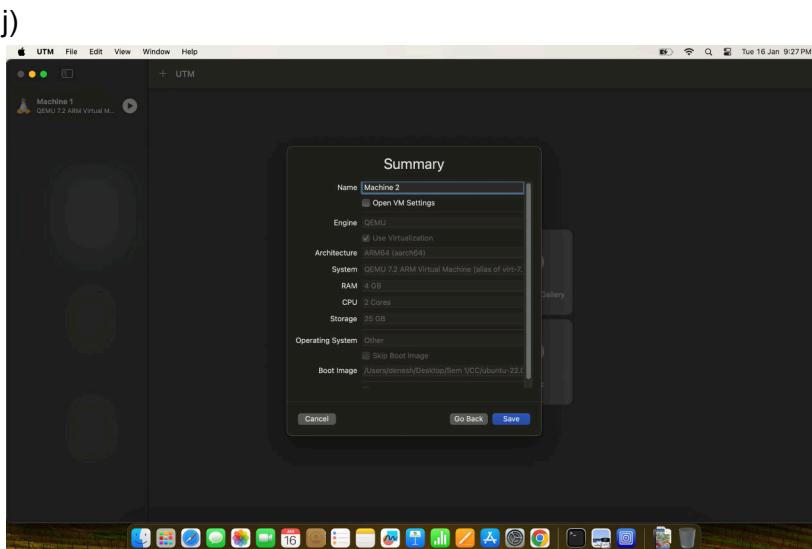
Activities Terminal Jan 17 04:55 ubuntu2@ubuntu2:~

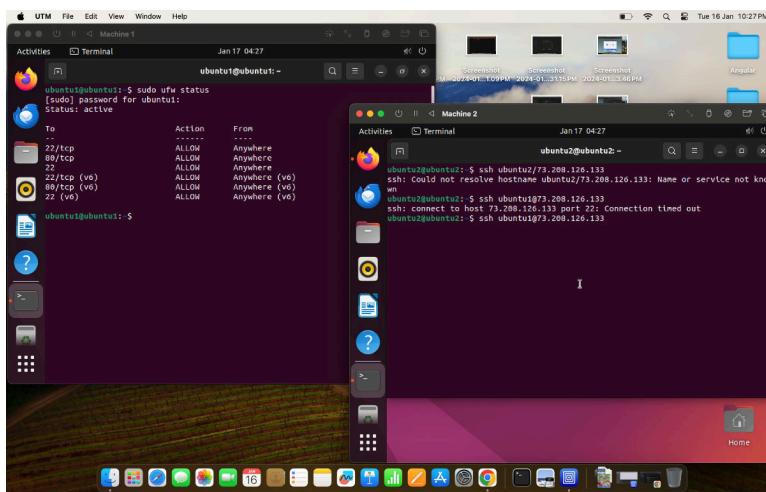
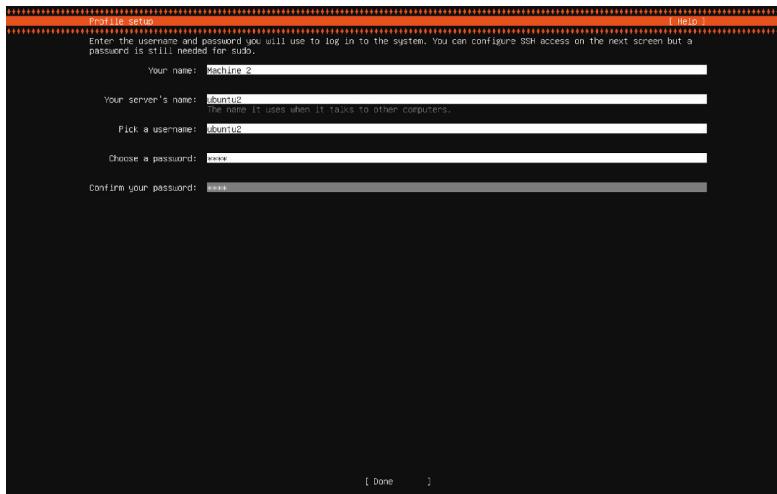
```
Chain ufw-user-limit-accept (0 references)
target  prot opt source      destination
ACCEPT  all  -- anywhere      anywhere
Chain ufw-user-logging-forward (0 references)
target  prot opt source      destination
Chain ufw-user-logging-input (0 references)
target  prot opt source      destination
Chain ufw-user-logging-output (0 references)
target  prot opt source      destination
Chain ufw-user-output (1 references)
target  prot opt source      destination
ubuntu2@ubuntu2:~$ sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
Bad argument '22' for '-p' or '--proto' option, try 'iptables -h' or 'iptables --help' for more information.
ubuntu2@ubuntu2:~$ sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
Bad argument '22' for '-p' or '--proto' option, try 'iptables -h' or 'iptables --help' for more information.
ubuntu2@ubuntu2:~$ sudo iptables -t nat -A PREROUTING -p tcp --dport 22 -j DNAT --to-destination 127.0.0.1:22
Bad argument '22' for '-p' or '--proto' option, try 'iptables -h' or 'iptables --help' for more information.
ubuntu2@ubuntu2:~$ sudo iptables -t nat -A PREROUTING -p tcp --dport 22 -j DNAT --to-destination 127.0.0.1:22
Bad argument '22' for '-p' or '--proto' option, try 'iptables -h' or 'iptables --help' for more information.
ubuntu2@ubuntu2:~$ sudo ufw allow ssh
Skipping adding existing rule
ubuntu2@ubuntu2:~$ sudo ufw enable
Skipping adding existing rule
ubuntu2@ubuntu2:~$ sudo ufw allow ssh
Skipping adding existing rule
ubuntu2@ubuntu2:~$ sudo apt-get services
Command 'firewall-cmd' not found, but can be installed with:
sudo apt install firewalld
ubuntu2@ubuntu2:~$ sudo apt list firewalld
ubuntu2@ubuntu2:~$ sudo apt list Firewall
ubuntu2@ubuntu2:~$ sudo apt update
ubuntu2@ubuntu2:~$ sudo apt install firewalld
ubuntu2@ubuntu2:~$ sudo apt install Firewall

Listing... Done
ubuntu2@ubuntu2:~$ sudo apt install firewalld
ubuntu2@ubuntu2:~$ sudo apt list Firewall
ubuntu2@ubuntu2:~$ sudo apt update
ubuntu2@ubuntu2:~$ sudo apt install firewalld
ubuntu2@ubuntu2:~$ sudo apt list Firewall
```

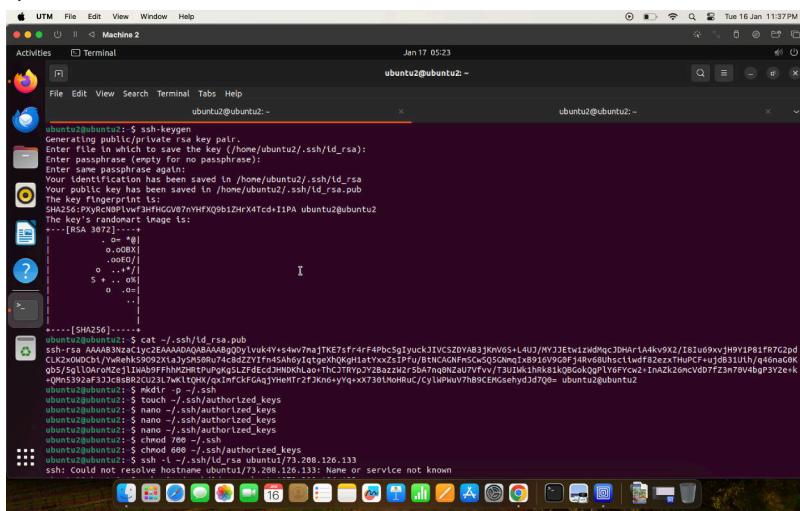
Machine 2

File Applications Dock

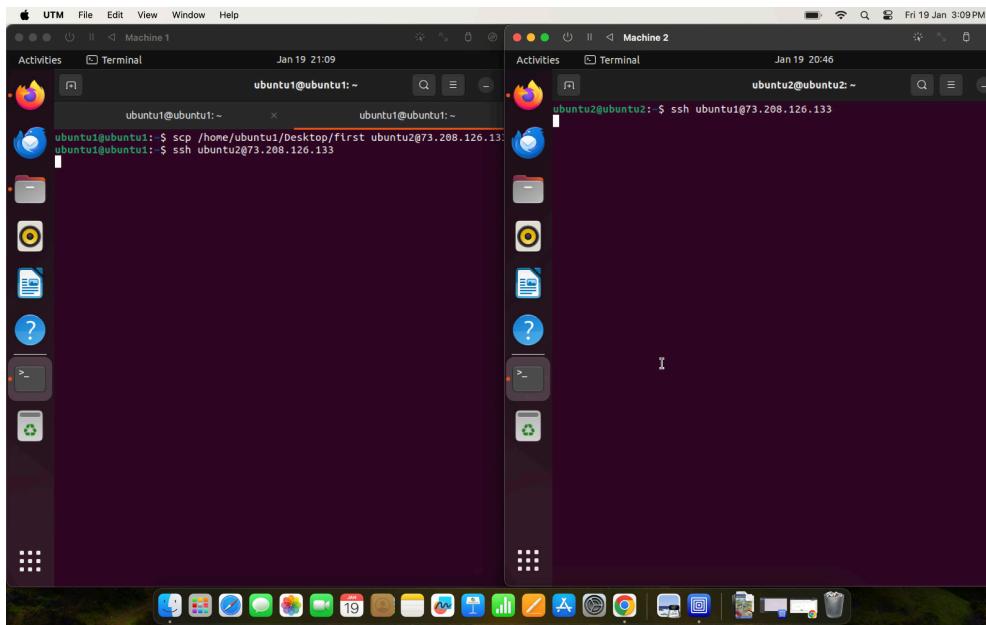




k)



I)



1. Download a virtual-box 7.0 in your machine if you are using windows other download UTM for mac uses. In my case, I'm using mac M1 chip which it is integrated with ARM, So i picked UTM.
2. Now download ubuntu 22.04 version from browser that fits your machine, I download a ARM related ubuntu image file. This iso file is used to install ubuntu on virtual box.
3. Now, click or start a new virtual machine, provide a readable name (let it be ubuntu1) , choose the location of iso file and click next.
4. Now allocate 4GB of RAM, 2 cores of processors, version as ubuntu 32 bit or 64 bit and change the network tab in advanced setting to bridged adaptor. Now click on proceed to build the virtual machine.
5. Till now you have configured everything for installation. Now you find ubuntu1 in the left navigation frame and double click on it to start. As soon as you double click, a popup is displayed and automatically starts the installation process by asking few questions.
6. Follow the on-screen instructions to complete the installation. During the installation process, it is mandatory to provide a username, strong password and re-type passwords fields .
7. Once ubuntu is properly installed, reboot your ubuntu. Now, block all ports by default and switch-on the firewall. Open only necessary port to access ssh, here the port is 22.
8. Now install openssh package and enable ssh firewall.
9. Repeat steps 4 to 8 to create another virtual machine and name the VM as ubuntu2.
10. generate ssh keys by using ssh-keygen and create public/private keys.
11. Now copy the public key of ubuntu1 to ubuntu2. Connect to the other VM using SSH and the private key from one of them. If successful, you will have created secure remote access between the VMs.

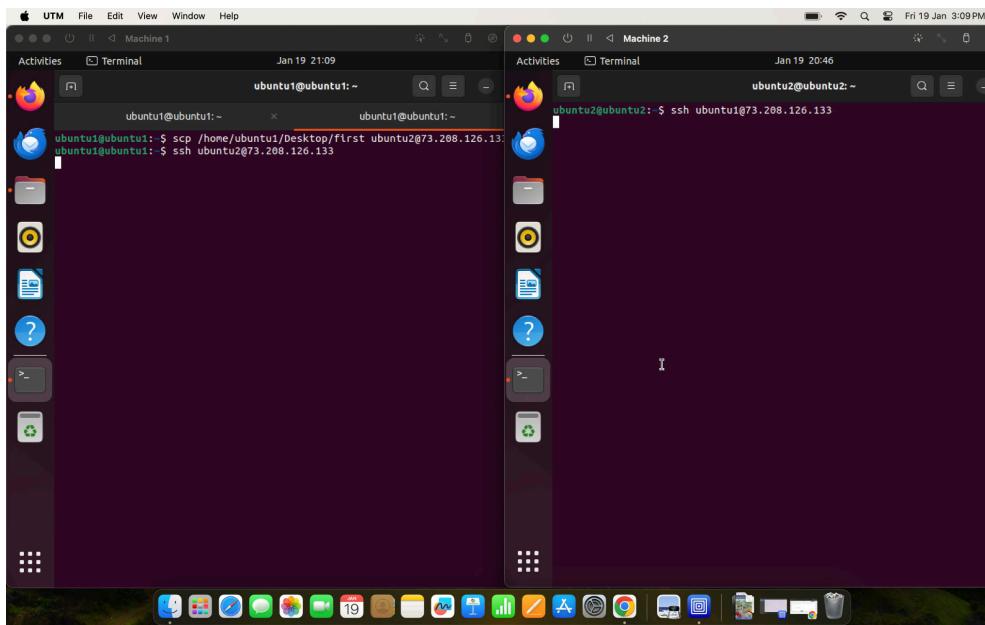
2)

a) ssh (Secure Shell)

It is used to communicate between 2 computers over a insecure network.

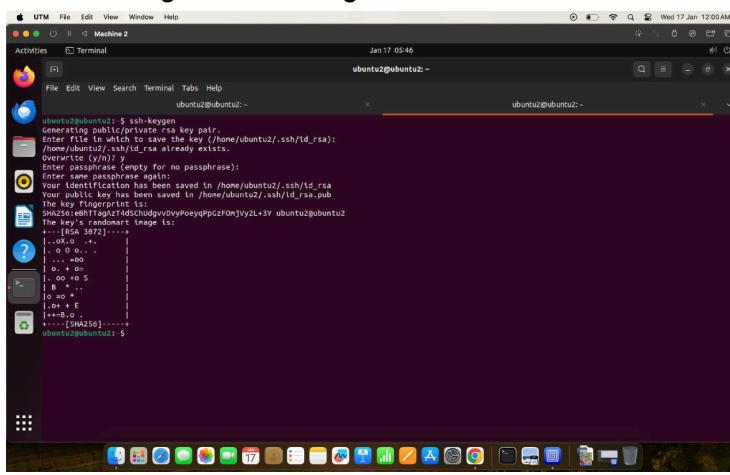
Syntax:

ssh username@host_ipAddress



b) ssh-keygen

It is used to generate, merge, and convert authentication keys for SSH connections.

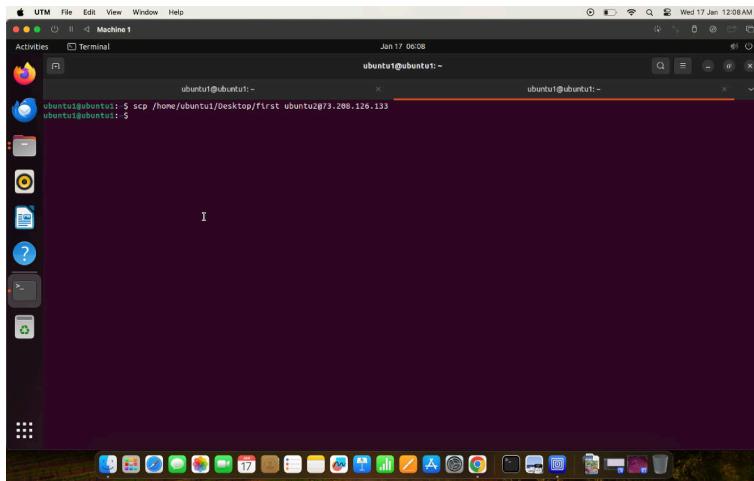


c) SCP (Secure Copy Protocol)

It is a secured transfer method used for copying files between hosts on the network.

Syntax:

Scp <local file path> username@remote_ipAddress:destination_directory



d) history

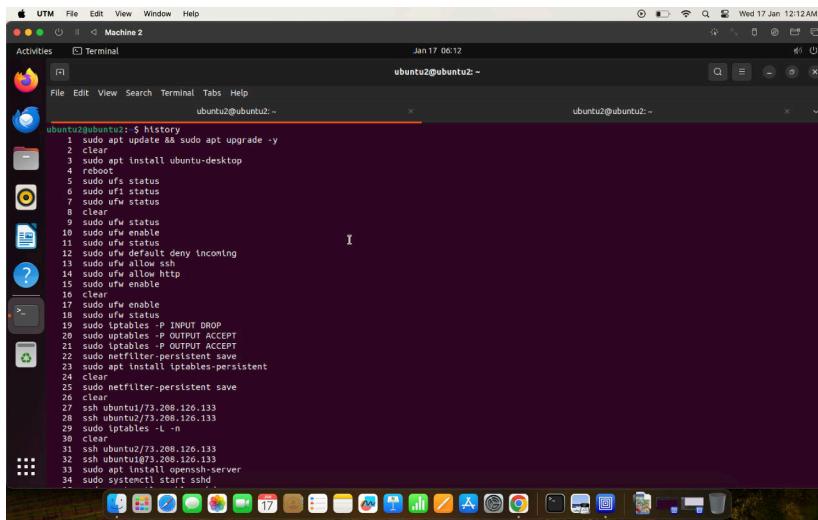
It provides all the previous commands that are executed in the current command prompt.

Syntax:

history

history <no of cmd>

history | grep <name of the command>

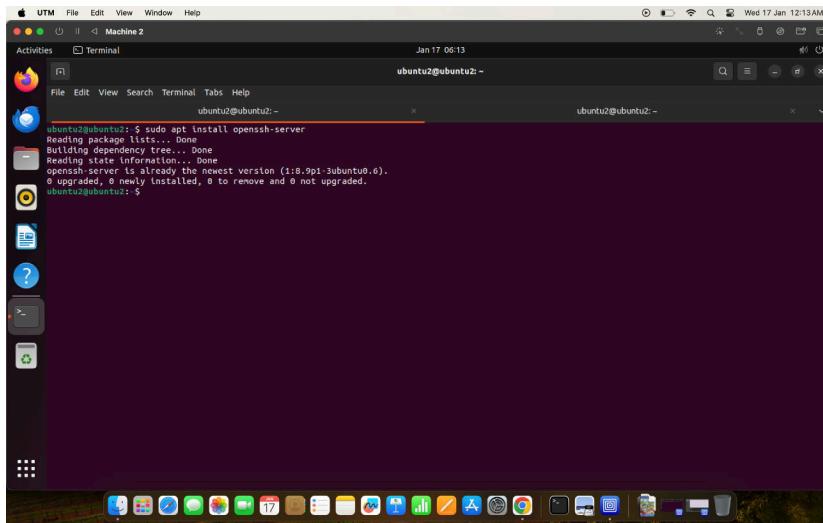


e) sudo

Sudo is a command-line utility which has special privileges temporarily and run commands in the root access. This is similar to Run as administrator in windows.\

Syntax:

sudo <command> [arguments]

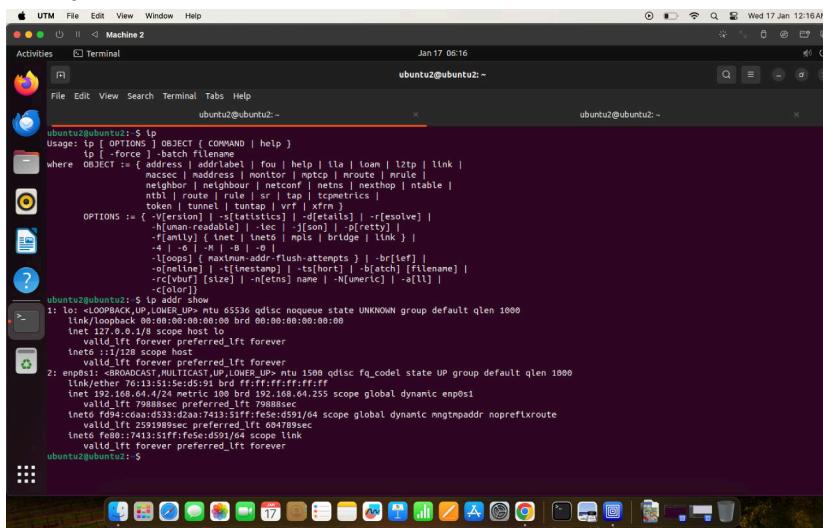


f) ip

It is used for configuration and managing network interfaces, routes and address in linux.

Syntax:

`ip [object] [command] [options]`



g) touch

This will create a new file at the current directory.

Syntax:

`touch <file name with proper extension>`

A screenshot of a Mac desktop environment. On the left is a dock with various application icons. In the center is a terminal window titled "Machine 2" with the command line interface "Terminal". The terminal shows the following session:

```
ubuntu2@ubuntu2:~/Desktop
ubuntu2@ubuntu2:~/Desktop$ touch myfile.txt
ubuntu2@ubuntu2:~/Desktop$ touch -m myfile.txt
ubuntu2@ubuntu2:~/Desktop$ ls
myfile.txt
ubuntu2@ubuntu2:~/Desktop$
```

h) ls

.ls is used to list all folders and files under a directory.

Syntax: ls

A screenshot of a Mac desktop environment. On the left is a dock with various application icons. In the center is a terminal window titled "Machine 2" with the command line interface "Terminal". The terminal shows the following session:

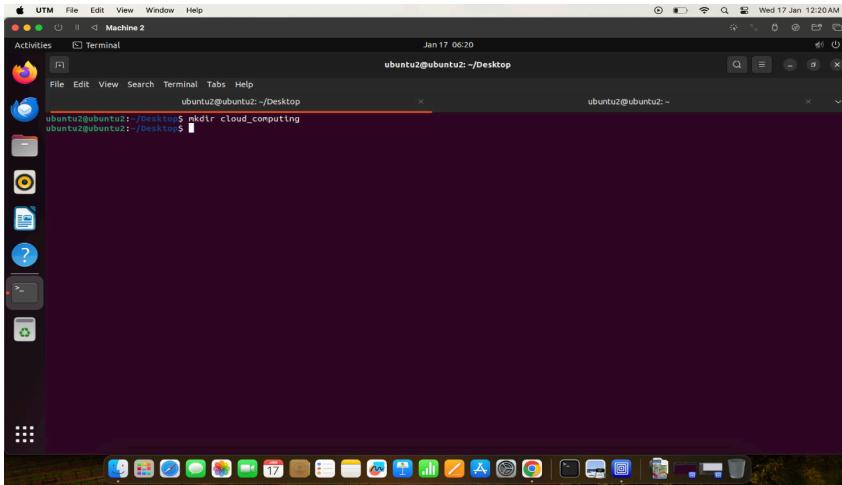
```
ubuntu2@ubuntu2:~/Desktop
ubuntu2@ubuntu2:~/Desktop$ mkdir cloud_computing
ubuntu2@ubuntu2:~/Desktop$ ls
cloud_computing myfile.txt
ubuntu2@ubuntu2:~/Desktop$
```

i) mkdir

This is used to create a new directory/ folder at a given path.

Syntax:

Mkdir <folder_name>

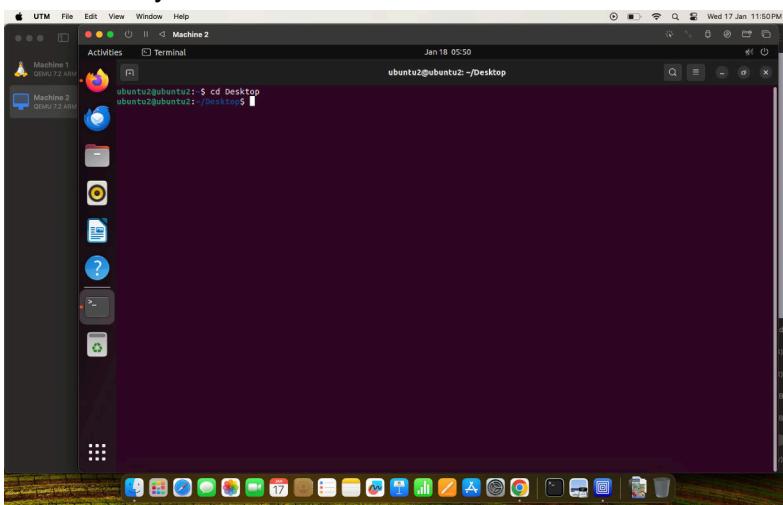


j) cd

The cd command in the linux is used to change directory. Here directory is nothing but a folder.

Syntax:

cd <directoryName>



k) dd

Convert, format and copy a file.

Syntax:

dd if=<inputFile> of=o<outputFile> [options]

inputFile : It specifies the source of the data which has to be read

outputFile: It specifies the destination where the data has to write.

Options : flag values like how the data need to be formatted or converted.

```

ubuntu2@ubuntu2:~/Desktop$ touch index1.txt
ubuntu2@ubuntu2:~/Desktop$ touch index2.txt
ubuntu2@ubuntu2:~/Desktop$ echo "HyperText Markup Language" > index1.txt
ubuntu2@ubuntu2:~/Desktop$ cat index1.txt index2.txt index.txt myfile.txt
ubuntu2@ubuntu2:~/Desktop$ dd if=index1.txt of=index2.txt conv=usec
0+1 records in
0+1 records out
20 bytes (20 B) copied, 0.00172815 s, 11.5 kB/s
ubuntu2@ubuntu2:~/Desktop$ more index1.txt
HyperText Markup Language
ubuntu2@ubuntu2:~/Desktop$ more index2.txt
HYPERTEXT MARKUP LANGUAGE
ubuntu2@ubuntu2:~/Desktop$
```

I) fdisk

Is used to manipulate the disk partition table. It has features like resize, delete, create, copy and move, change partition types and view partitions.

Syntax:

`fdisk [options] /dev/Disk`

```

ubuntu2@ubuntu2:~$ sudo fdisk -l
Disk /dev/loop0: 1.40 MB, 49160732 bytes, 97276 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/loop1: 16.49 MB, 16979664 bytes, 324972 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/loop2: 107.41 MB, 110978644 bytes, 224072 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/loop3: 4 KB, 4096 bytes, 8 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/loop4: 230.82 MB, 24202640 bytes, 472720 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/loop5: 99.28 MB, 62124032 bytes, 121336 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk /dev/loop6: 69.07 MB, 72629568 bytes, 141644 sectors
```

m) apt (Advanced Packaging Tool)

It is the core package management system in ubuntu which allows the features like install, update, upgrade and remove software packages through the command line.

Syntax:

`Sudo apt <option> <package-name>`

A screenshot of a UTM virtual machine interface. The terminal window shows the command `sudo apt update` being run, with a list of packages being downloaded from various Ubuntu repositories. The desktop environment includes a dock with various application icons.

```
ubuntu2@ubuntu2: ~$ sudo apt update
Hit:1 https://ports.ubuntu.com/ubuntu-ports jammy InRelease [119 kB]
Get:3 https://ports.ubuntu.com/ubuntu-ports jammy-updates InRelease [119 kB]
Get:4 https://ports.ubuntu.com/ubuntu-ports jammy-security InRelease [110 kB]
Get:6 https://ports.ubuntu.com/ubuntu-ports jammy-updates/universe arm64 Packages [559 kB]
Get:7 http://ports.ubuntu.com/ubuntu-ports jammy-updates/universe Translation-en [229 kB]
Get:9 https://ports.ubuntu.com/ubuntu-ports jammy-backports/universe arm64 Packages [64.4 kB]
Get:10 http://ports.ubuntu.com/ubuntu-ports jammy-backports/universe arm64 Packages [26.2 kB]
Get:12 http://ports.ubuntu.com/ubuntu-ports jammy-security/universe arm64 Packages [120 kB]
Get:13 http://ports.ubuntu.com/ubuntu-ports jammy-security/universe Translation-en [204 kB]
Get:15 http://ports.ubuntu.com/ubuntu-ports jammy-security/universe arm64 Packages [76.8 kB]
Get:16 http://ports.ubuntu.com/ubuntu-ports jammy-security/universe Translation-en [158 kB]
Fetched 5,844 kB in 4s (1,427 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
4 packages can be upgraded. Run 'apt list --upgradeable' to see them.
ubuntu2@ubuntu2: ~
```

n) vi

vi stands visual which is used as a text editor. You can edit, delete lines of code after using vi filename.extension. You can only write your code in command line.

Syntax:

Vi <fileName.extension>

Two screenshots of a UTM virtual machine showing the vi editor. The left screenshot shows the command `man vi` being run. The right screenshot shows the vi editor open with the file `index.c`, containing the code `#include <stdio.h>` and `int main() { printf("Hello World"); }`.

o) time

The time command is a built-in command to show the execution time of a command or a set of instructions. It can also show the details of resource usage like memory and CPU utilization.

Syntax:

```
ubuntu2@ubuntu2: ~ man time
ubuntu2@ubuntu2: ~ ls -l
total 52
-rwxr-xr-x 1 ubuntu2 ubuntu2 4896 Jan 18 06:35 a.out
drwxr-xr-x 3 ubuntu2 ubuntu2 4096 Jan 18 06:39 desktop
drwxr-xr-x 2 ubuntu2 ubuntu2 4096 Jan 17 04:15 Documents
drwxr-xr-x 2 ubuntu2 ubuntu2 4096 Jan 17 04:15 Downloads
-rw-rw-r- 1 ubuntu2 ubuntu2 59 Jan 18 06:45 index.c
drwxr-xr-x 2 ubuntu2 ubuntu2 4096 Jan 17 04:15 index.html
drwxr-xr-x 2 ubuntu2 ubuntu2 4096 Jan 17 04:15 Pictures
drwxr-xr-x 2 ubuntu2 ubuntu2 4096 Jan 17 04:15 Public
drwxr-xr-x 3 ubuntu2 ubuntu2 4096 Jan 17 04:23 snap
drwxr-xr-x 2 ubuntu2 ubuntu2 4096 Jan 17 04:15 Templates
drwxr-xr-x 2 ubuntu2 ubuntu2 4096 Jan 17 04:15 Videos
real    0m 0.012s
user    0m 0.009s
sys     0m 0.006s
ubuntu2@ubuntu2: ~
```

p) tar

It is a powerful tool to manage archive files. It can create, list, extract and manage the archive files.

Syntax:

`tar [option] [operation] [archiveFiles]`

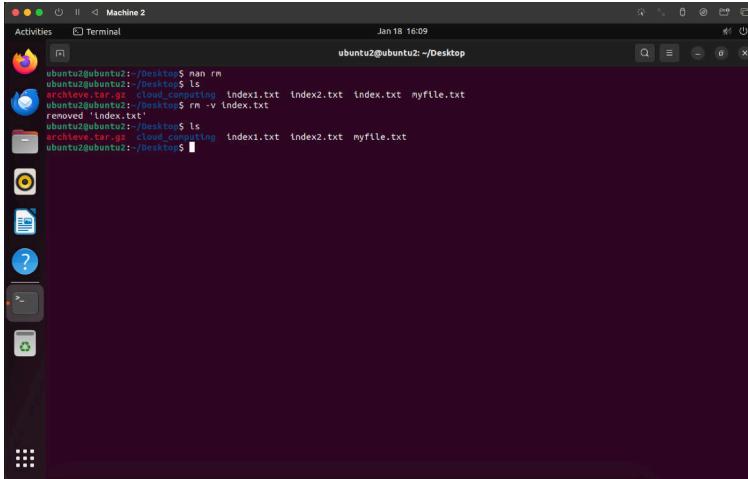
```
ubuntu2@ubuntu2:~/Desktop$ man tar
ubuntu2@ubuntu2:~/Desktop$ tar -czvf archive.tar.gz index1.txt
index1.txt
ubuntu2@ubuntu2:~/Desktop$ ls
archive.tar.gz  cloud_computing  index1.txt  index2.txt  index.txt  myfile.txt
ubuntu2@ubuntu2:~/Desktop$
```

q) rm

It is a command line keyword which is used to permanently remove a file or directory from a filesystem.

Syntax:

`rm [option] [file or directory name]`



A screenshot of an Ubuntu desktop environment. On the left is a dock with icons for Dash, Home, Applications, and Help. In the center is a terminal window titled "Machine 2" with the command line "ubuntu2@ubuntu2:~/Desktop\$". The terminal shows the following session:

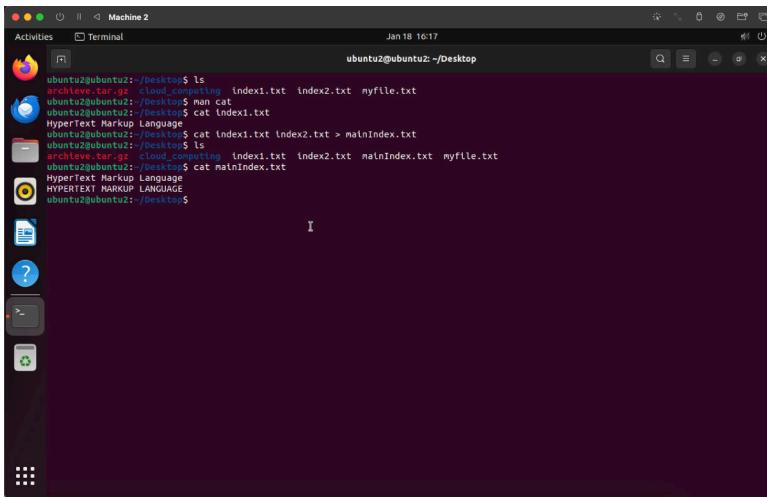
```
ubuntu2@ubuntu2:~/Desktop$ man rm
ubuntu2@ubuntu2:~/Desktop$ ls
archiveve.tar.gz  cloud_computing  index1.txt  index2.txt  index.txt  myfile.txt
ubuntu2@ubuntu2:~/Desktop$ rm -v index.txt
rm: removing index.txt
ubuntu2@ubuntu2:~/Desktop$ ls
archiveve.tar.gz  cloud_computing  index1.txt  index2.txt  myfile.txt
ubuntu2@ubuntu2:~/Desktop$
```

r) cat

It is a fundamental tool in ubuntu and linux commands which is used for 2 features. One, to print the content in the file and two, concatenates 2 or more files.

Syntax:

Cat [options => -n /-b /-v /-E] [file1] [file2]



A screenshot of an Ubuntu desktop environment. On the left is a dock with icons for Dash, Home, Applications, and Help. In the center is a terminal window titled "Machine 2" with the command line "ubuntu2@ubuntu2:~/Desktop\$". The terminal shows the following session:

```
ubuntu2@ubuntu2:~/Desktop$ ls
archiveve.tar.gz  cloud_computing  index1.txt  index2.txt  myfile.txt
ubuntu2@ubuntu2:~/Desktop$ man cat
ubuntu2@ubuntu2:~/Desktop$ cat index1.txt
HyperText Markup Language
ubuntu2@ubuntu2:~/Desktop$ cat index1.txt index2.txt > mainIndex.txt
ubuntu2@ubuntu2:~/Desktop$ ls
archiveve.tar.gz  cloud_computing  index1.txt  index2.txt  mainIndex.txt  myfile.txt
ubuntu2@ubuntu2:~/Desktop$ cat mainIndex.txt
HyperText Markup Language
HYPERTEXT MARKUP LANGUAGE
ubuntu2@ubuntu2:~/Desktop$
```

s) bash

It has 2 broad features. First, it understand and execute the instructions that are provided the user. Second, you can write a file that has sequence of commands and execute it like a bash scripting.

```
Assignment 1 $ network-test.sh
1 #!/bin/bash
2
3 # this is input file which contains few domain names line by line
4 inputFile="network-test-machineList.txt"
5
6 # this is output file which contains the domain name and avg RTT separated by a space
7 outputFile="network-test-latency.txt"
8
9 getPingAndCalculateRTT() {
10     # get the first argument by $1
11     # $1 is number of samples are 3
12     domainName=$1
13     sample=3
14
15     echo ${domainName}
16
17     # below code is used to send a specific number of ICMP(Internet Control Message Protocol) requests packets to the given DNS name.
18     # -c indicates a flag that indicates number of packets to send.
19     # grep is used to filter/ search for line at specific string : round-trip
20     # awk operates on per-line basis and respective value at $1 and prints
21     # cut command is used to extract only column $1 and gets 2nd value
22     avgRoundTripTime=$(ping -c $sample $domainName | grep "round-trip" | awk '{print $4}' | cut -d '/' -f 2)
23     echo $avgRoundTripTime
24
25     # writing domain and avg RTT values into the output file.
26     echo "$domainName $avgRoundTripTime" >> $outputFile
27 }
28
29 # checking whether the input file is present.
30 # If not present then a message is printed and program is terminated.
31 if [ ! -f $inputFile ]; then
32     echo "Error! The input file is not valid"
33     exit 1
34 fi
35
36 # Previously if any output file present, then we are removing it and creating a new file
37 if [ -f $outputFile ]; then
38     rm $outputFile
39     echo > $outputFile
40
41 main()
42 {
43     for domain in $(cat $inputFile)
44     do
45         getPingAndCalculateRTT $domain
46     done
47 }
48
49 main()
```

t) more

It allows us to view the content of the file in one page at a time. It is basically used for larger files.

Syntax:

```
more [options -n/-s/-u] fileName
```

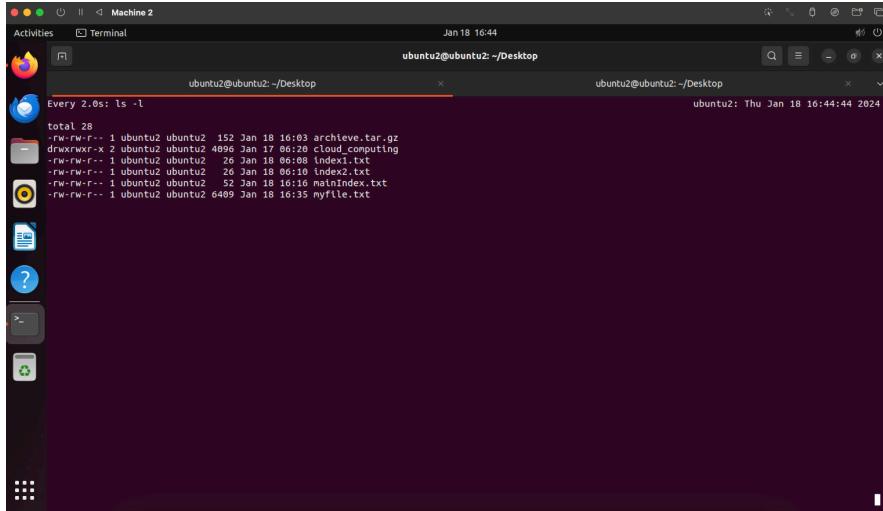
```
ubuntu2@ubuntu2:~/Desktop$ man more
ubuntu2@ubuntu2:~/Desktop$ ls
archieve.tar.gz  cloud_computing_index1.txt  index2.txt  mainIndex.txt  myfile.txt
ubuntu2@ubuntu2:~/Desktop$ more -24 myfile.txt
The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It defines the content and structure of web content. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.
Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for its appearance.
The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It defines the content and structure of web content. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.
Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for its appearance.
The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It defines the content and structure of web content. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.
Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for its appearance.
The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It defines the content and structure of web content. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.
Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for its appearance.
```

u) watch

It observes the changes in the output file of any command over time.

Syntax:

```
watch [options => -n/ -b/ -d/ -h/ -t] <command>
```

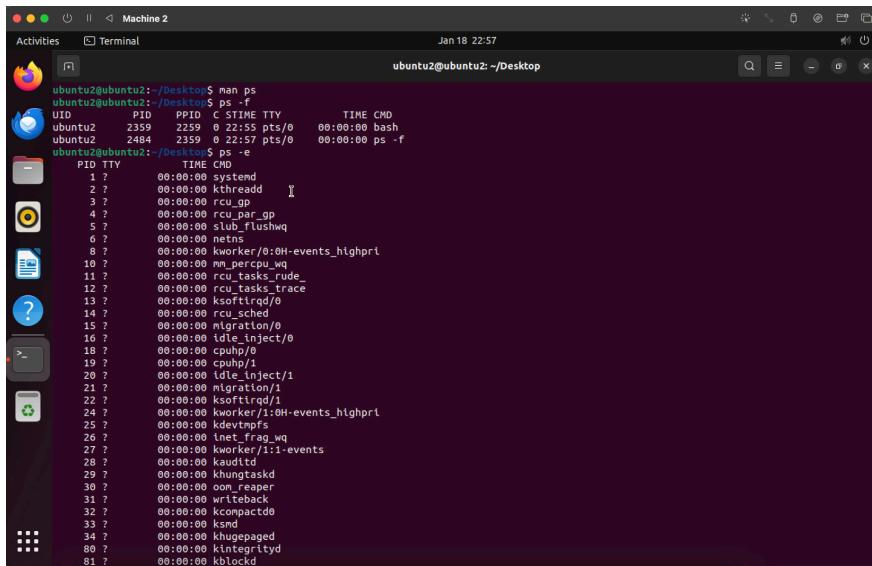


v) ps

ps refers to Process Status which gives us an idea or the snapshot of currently running processes. It displays few columns like PID, owner, CPU usage, memory usage and etc.,

Syntax:

`ps [options]`

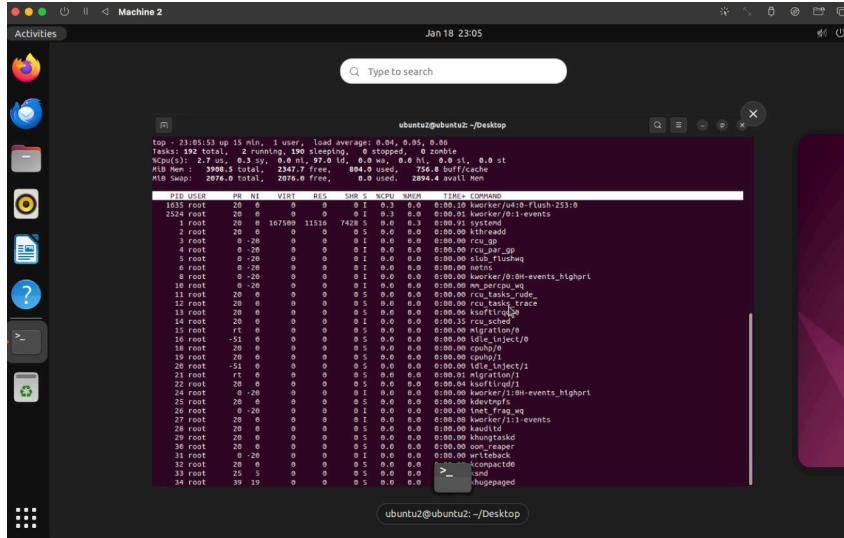


w) top

It displays dynamic and real-time processes and system resource usage. It provides information about CPUs, PIDs, memory usage, etc information while showing in the terminal.

Syntax:

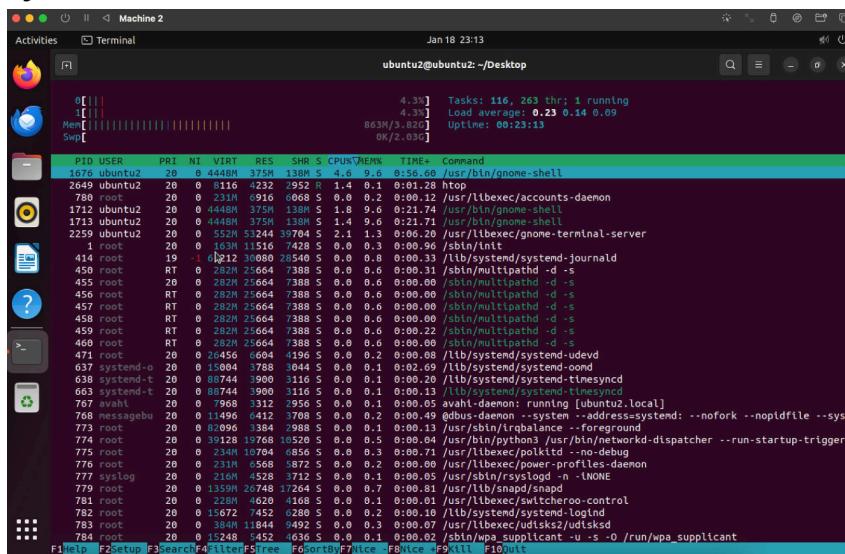
`top -u root -%CPU, %PID, USER, COMMAND, %MEM, TIME+`



x) htop

It is used to monitor and manage processes like it provides information about the core usage RAM memory usage. It is very much similar to top but it has additional features like mouse support, process tree view, virtual and horizontal scrolling.

Syntax:



htop

y) gcc

gcc stands for GNU Compiler Collection. It has a set of compilers and development related tools such as C, C++, Objective-C, objective-C++, fortran.

Syntax:

gcc [fileName.extension]

The screenshot shows a terminal window titled "Machine 2" with two tabs: "Activities" and "Terminal". The terminal window has a title bar "ubuntu2@ubuntu2: ~/Desktop" and a status bar "Jan 19 00:00". The terminal content displays the output of the "tail" command on a large file named "myfile.txt". The file contains several lines of assembly-like code, including comments like "GCC: (Ubuntu 11.4.0-1ubuntu1-22.04) 11.4.08Tx***P", and ends with "8TS6XX D***eN". The terminal window also shows other files in the directory: "archieve.tar.gz", "clangauge.c", "cloud_computing", "index1.txt", "index2.txt", "mainIndex.txt", and "myfile.txt".

z) tail

tail is a utility tool which is used to display last part/ chunks of a file or multiple files.

Syntax:

`tail [options] [file]`

The screenshot shows a terminal window titled "Machine 2" with two tabs: "Activities" and "Terminal". The terminal window has a title bar "ubuntu2@ubuntu2: ~/Desktop" and a status bar "Thu 18 Jan 09:09PM". The terminal content shows the command "tail myfile.txt" being run, followed by the content of the "myfile.txt" file. The file contains several lines of text, including "Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for its appearance.", "The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It defines the content and structure of web content. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.", and "Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for its appearance.". The terminal window also shows other files in the directory: "man", "ls", "archieve.tar.gz", "clangauge.c", "cloud_computing", "index1.txt", "index2.txt", and "mainIndex.txt".

aa) grep

grep stands for Global Regular Expression Print. It is used to search or filter a text and extract text patterns from a file.

Syntax:

`grep [options -i/ -c/ -n/ -v/ -r] pattern [file]`

```

ubuntu2@ubuntu2:~/Desktop$ man grep
ubuntu2@ubuntu2:~/Desktop$ ls
archive.tar.gz clangUAGE clangUAGE.c cloud_computing index1.txt index2.txt mainIndex.txt myfile.txt
ubuntu2@ubuntu2:~/Desktop$ grep -n HTML myfile.txt
1:1) The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It defines the content and structure of web content. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.
2:) Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for its appearance.
3:) The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It defines the content and structure of web content. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.
4:) Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for its appearance.
5:) The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It defines the content and structure of web content. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.
6:) Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for its appearance.
ubuntu2@ubuntu2:~/Desktop$ man html

```

bb) kill

It is a command-line utility which is used to terminate or stop a process.

Syntax:

`Kill [options] <PID>`

Code in javascript:

```
const http = require('http');
```

```
const server = http.createServer((req, res) => {
  // Handle incoming requests
});
```

```
server.listen(3000, () => { console.log('Server listening on port 3000'); });
```

```

ubuntu2@ubuntu2:~/Desktop$ ls
archive.tar.gz clangUAGE clangUAGE.c cloud_computing index1.txt index2.txt mainIndex.txt myfile.txt testPort.js
ubuntu2@ubuntu2:~/Desktop$ node testPort.js
Server listening on port 3000

```

The image shows two terminal windows side-by-side on a Mac OS X desktop. Both windows have the title bar "Machine 2" and show the date and time "Jan 19 00:37".

The left terminal window contains the following command and output:

```
ubuntu2@ubuntu2:~/Desktop$ man kill
ubuntu2@ubuntu2:~/Desktop$ lsof -i :3000
COMMAND PID USER FD TYPE DEVICE SIZE/OFF NODE NAME
node  7740 ubuntu2  18u  IPv6  61112      0t0  TCP *:3000 (LISTEN)
ubuntu2@ubuntu2:~/Desktop$ kill 7740
ubuntu2@ubuntu2:~/Desktop$
```

The right terminal window contains the following command and output:

```
ubuntu2@ubuntu2:~/Desktop$ ls
archive.tar.gz clangUAGE clangUAGE.c cloud_computing index1.txt index2.txt mainIndex.txt myfile.txt testPort.js
ubuntu2@ubuntu2:~/Desktop$ node testPort.js
Server listening on port 3000
Terminated
ubuntu2@ubuntu2:~/Desktop$
```

cc) killall

killall command terminates a set of signals or processes by name.

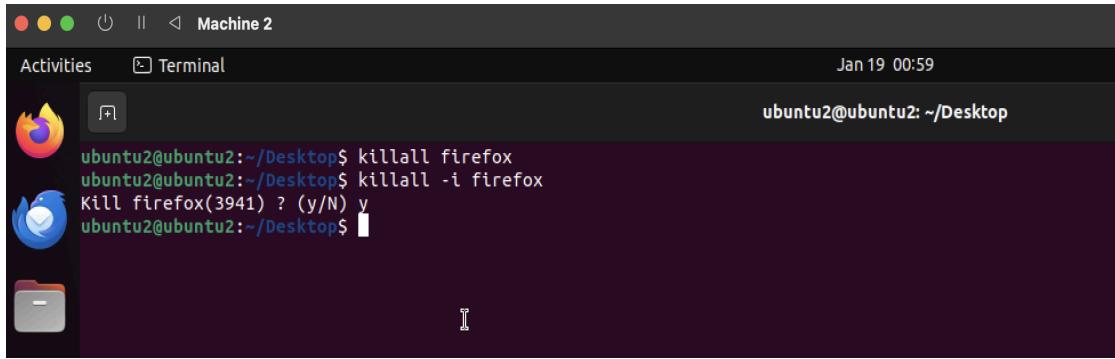
Syntax:

killall [options] <port name>

The image shows a single terminal window on a Mac OS X desktop. The title bar says "Machine 2" and the date and time are "Jan 19 00:58".

The terminal window contains the following command and interaction:

```
ubuntu2@ubuntu2:~/Desktop$ killall firefox
ubuntu2@ubuntu2:~/Desktop$ killall -i firefox
Kill firefox(3941) ? (y/N) y
ubuntu2@ubuntu2:~/Desktop$
```



A screenshot of a Ubuntu desktop environment. A terminal window titled "Machine 2" is open, showing the command line interface. The terminal window has a dark background with light-colored text. It displays the following commands and output:

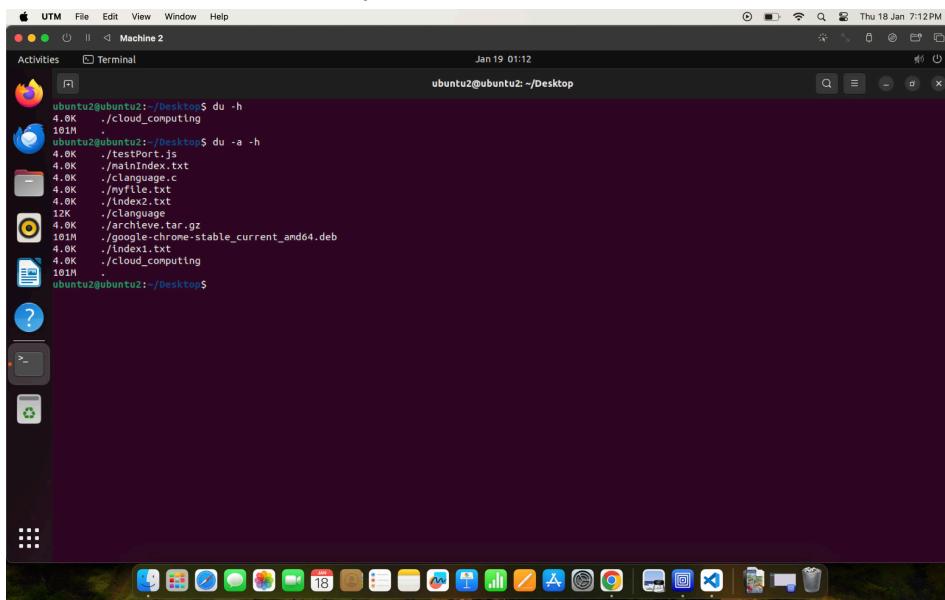
```
ubuntu2@ubuntu2:~/Desktop$ killall firefox
ubuntu2@ubuntu2:~/Desktop$ killall -l firefox
Kill firefox(3941) ? (y/N) y
ubuntu2@ubuntu2:~/Desktop$
```

dd) du

It is used to estimate the disk space usage of files and directories.

Syntax:

du [options] [files or directory]



A screenshot of a Ubuntu desktop environment. A terminal window titled "Machine 2" is open, showing the command line interface. The terminal window has a dark background with light-colored text. It displays the output of the du command:

```
ubuntu2@ubuntu2:~/Desktop$ du -h
4.0K ./.cloud_computing
101M .
ubuntu2@ubuntu2:~/Desktop$ du -a -h
4.0K ./testPort.js
4.0K ./mainIndex.txt
4.0K ./clangUAGE.c
4.0K ./myfile.txt
4.0K ./index2.txt
12K ./clangUAGE
101M ./myfile.tar.gz
101M ./google-chrome-stable_current_amd64.deb
4.0K ./index1.txt
4.0K ./cloud_computing
101M .
```

ee) df

df stands for disk free which tells us about the disk space usage on the mounted file system.

Syntax:

df [option] [file]

options such as -h (for displaying KB, MBs), -T (displays file type)

```

ubuntu2@ubuntu2:~/Desktop$ man df
ubuntu2@ubuntu2:~/Desktop$ df
Filesystem      1K-blocks   Used   Available Use% Mounted on
tmpfs            400232    1804    398428  1% /run
/dev/nvme0n1p1  11218472 9922216  704556 94% /
tmpfs             261140      0    2601140  0% /dev/shm
tmpfs              5138      4     5116  1% /run/lock
tmpfs            1992552  262208  1669104 15% /boot
/dev/vda1          1098628   6452   1092176  1% /boot/efi
tmpfs            400228     100    400128  1% /run/user/1000
Filesystem      Inodes IUsed IFree IUsed% Mounted on
tmpfs           500285  1025 499269  1% /run
tmpfs           500285  1025 499269  1% /run
tmpfs           500285  1025 499269  1% /dev/shm
tmpfs           500285  1025 499269  1% /dev/shm
tmpfs           500285  1025 499269  1% /run/lock
tmpfs           131972  260 130812  1% /boot
tmpfs             100057      0      0  1% /boot/efi
tmpfs           100057  145 99912  1% /run/user/1000
Filesystem      1K-blocks   Used   Available Use% Mounted on
tmpfs            400232    1836    398396  1% /run
/dev/nvme0n1p1  11218472 9922252  704556 94% /
tmpfs             2601140      0    2601140  0% /dev/shm
tmpfs              5138      4     5116  1% /run/lock
tmpfs            1992552  262208  1669104 15% /boot
/dev/vda1          1098628   6452   1092176  1% /boot/efi
tmpfs            400228     100    400128  1% /run/user/1000
ubuntu2@ubuntu2:~/Desktop$ df -k
Filesystem      Size   Used Avail Use% Mounted on
tmpfs           391M   1.8M 390M  1% /run
/dev/nvme0n1p1  11G  9.5G 689M 94% /
tmpfs           2.0G   2.0G  0  100% /dev/shm
tmpfs           513M   4.0M 509M  1% /run/lock
tmpfs           2.0G  257M  1.6G 15% /boot
tmpfs           1.1G   6.4M 1.1G  1% /boot/efi
tmpfs           391M  108K 391M  1% /run/user/1000
ubuntu2@ubuntu2:~/Desktop$ df -h

```

ff) screen

It allows you to create multiple virtual terminals within a single virtual terminal window.

Syntax:

Screen [options] [cmd [args]]

```

ubuntu2@ubuntu2:~/Desktop$ screen -ls
There are screens on:
  5118.pts-0          (01/19/2024 01:22:05 AM)          (Attached)
  4999.pts-0          (01/19/2024 01:22:07 AM)          (Attached)
  4998.pts-0          (01/19/2024 01:22:07 AM)          (Attached)
  3 Sockets in /run/screen/S-ubuntu2.
[remote detached from s118_file2]
ubuntu2@ubuntu2:~/Desktop$ 

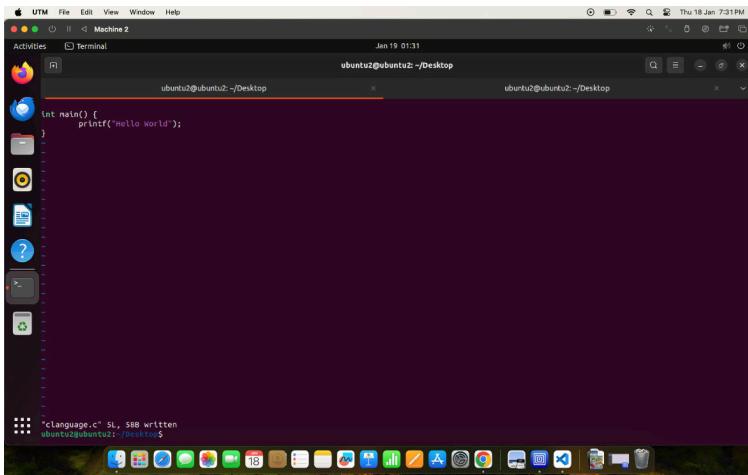
```

gg) vim

Vim (vi improved) is very similar to vi command. The

Syntax:

vim <fileName.extension>



```
int main() {
    printf("Hello World");
}

"clanguage.c" 5L, 58B written
ubuntu2@ubuntu2:~/Desktop$
```

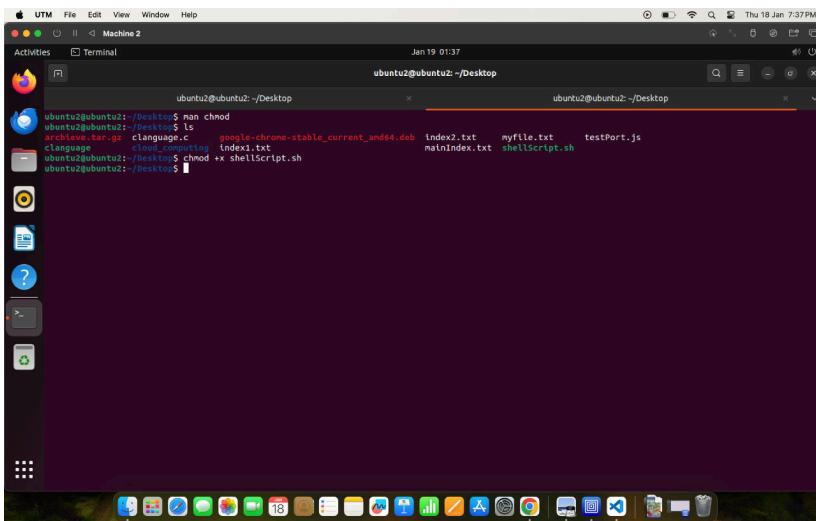
hh) chmod

chmod stands for change mode, it modifies the access permissions of a file or a directory and determine them with read, write and execute permissions.

Syntax:

chmod +x fileName.extension

r-> read, w->write, x-> execute



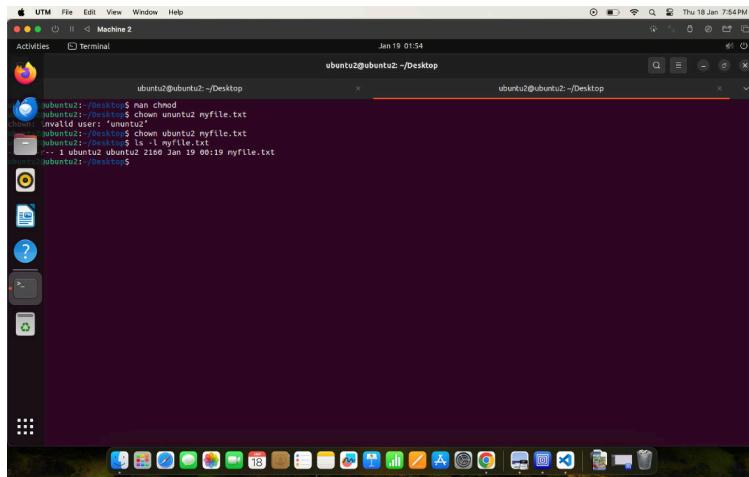
```
ubuntu2@ubuntu2:~/Desktop$ man chmod
ubuntu2@ubuntu2:~/Desktop$ ls
archive.tar.gz  clanguage.c  google-chrome-stable_current_amd64.deb  index2.txt  myfile.txt  testPort.js
clanguage  google-chrome-stable_current_amd64.deb  index1.txt  mainIndex.txt  shellScript.sh
ubuntu2@ubuntu2:~/Desktop$ chmod +x shellScript.sh
ubuntu2@ubuntu2:~/Desktop$
```

ii) chown

It is used to change ownership of a file or directory. It allows you to specify new owners or group for file and directories.

Syntax:

chown [OPTIONS] USER:[GROUP] fileName



```
ubuntu2@ubuntu2:~/Desktop
Activities Terminal Jan 19 01:54
ubuntu2@ubuntu2:~/Desktop
ubuntu2@ubuntu2:~/Desktop$ man chmod
ubuntu2@ubuntu2:~/Desktop$ chown ununtu2 myfile.txt
ubuntu2@ubuntu2:~/Desktop$ ls -l myfile.txt
ubuntu2@ubuntu2:~/Desktop$ ls -l myfile.txt
-- 1 ubuntu2 ubuntu 2160 Jan 19 00:19 myfile.txt
ubuntu2@ubuntu2:~/Desktop$
```

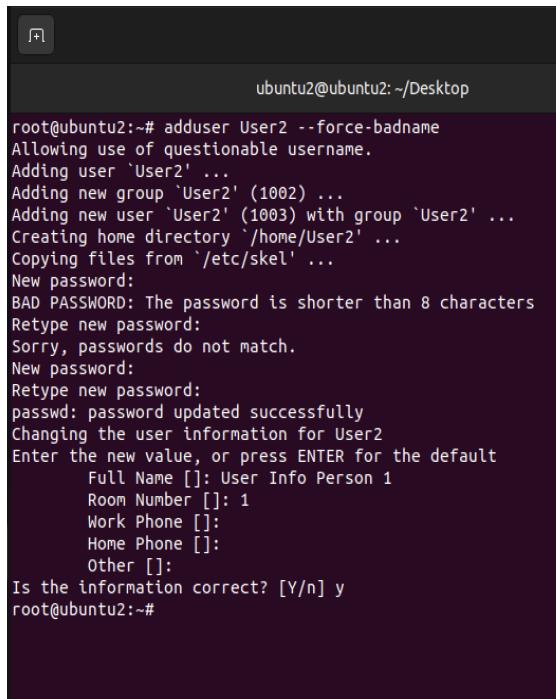
jj) **useradd**

It is used to add a new user account in ubuntu.

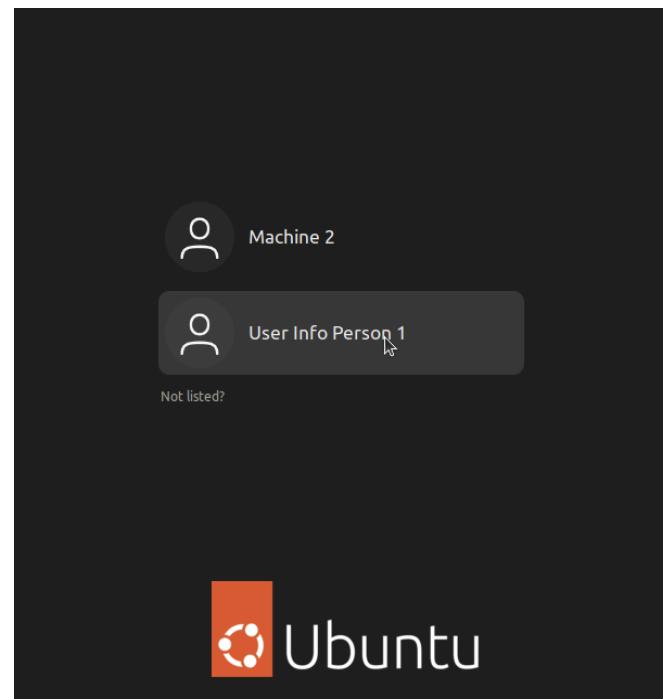
Syntax:

useradd [options] USERNAME

Options such as -d for specifies a custom home directory for the user, -s for set a users login. It is only done in root mode of shell, -u creates a unique ID number for a user. It is only done in root mode of ubuntu.



```
root@ubuntu2:~# adduser User2 --force-badname
Allowing use of questionable username.
Adding user `User2' ...
Adding new group `User2' (1002) ...
Adding new user `User2' (1003) with group `User2' ...
Creating home directory `/home/User2' ...
Copying files from `/etc/skel' ...
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
Sorry, passwords do not match.
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for User2
Enter the new value, or press ENTER for the default
      Full Name []: User Info Person 1
      Room Number []: 1
      Work Phone []:
      Home Phone []:
      Other []:
Is the information correct? [Y/n] y
root@ubuntu2:~#
```



kk) **mv**

It is a command which moves and renames a file or a directory.

Syntax:

`mv [options] source destination`

Options such as `-i` => prompt confirmation before overriding the existing files, `-f` => overwrites the existing files with any permissions, `-n` => Prevents overriding the existing files.

```
ubuntu2@ubuntu2:~/Desktop$ ls
archive.tar.gz clangage.c google-chrome-stable_current_amd64.deb index2.txt myfile.txt testPort.js
clangage cloud_computing index1.txt mainIndex.txt shellScript.sh
ubuntu2@ubuntu2:~/Desktop$ mv testPort.js ..//Downloads/testing.js
ubuntu2@ubuntu2:~/Desktop$ cd ..
ubuntu2@ubuntu2:~$ cd Downloads
ubuntu2@ubuntu2:~/Downloads$ ls
testing.js
ubuntu2@ubuntu2:~/Downloads$
```

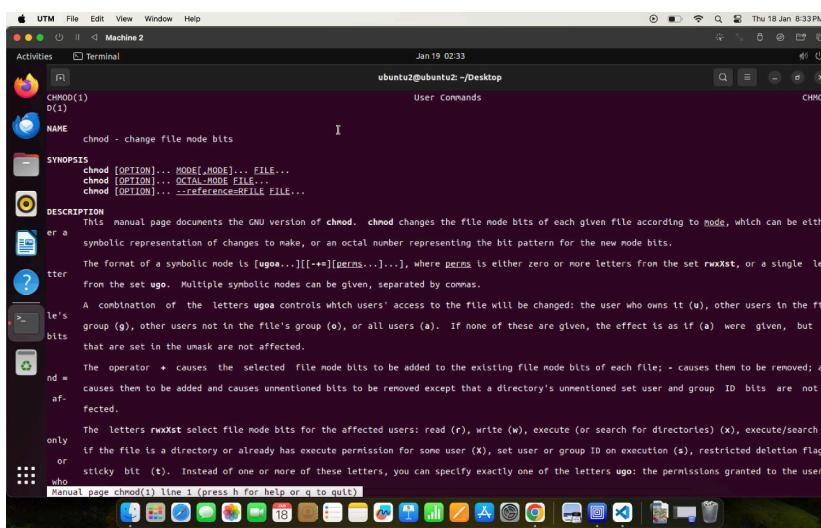
II) man

Man refers to manual.

It is used to describe command names with definition, syntax and its usage.

Syntax:

`man <commandName>`



mm) locate

locate command is the fast and efficient way of finding files and directories in the ubuntu system.

Syntax:

`locate [options] pattern`

Here pattern is nothing but a part file name.

It has options like `-i` which indices case sensitive search, `-c` => count the number of entries which match.

```
ubuntu2@ubuntu2:~$ man locate
man(1)                                locate(1)                              手册页

Ubuntu 22.04 LTS - 2023-01-19: 0.9.18.2-0.1

Activities Terminal Jan 19 03:58 ubuntu2@ubuntu2:~$ ubuntu2@ubuntu2:~$ locate index.c
index.c
ubuntu2@ubuntu2:~$
```

nn) find

find command is used to locate files and directories in the file system based on specific criteria

Syntax:

Find [start-directory] expression

```
ubuntu2@ubuntu2:~$ man find
ubuntu2@ubuntu2:~$ find . -name 'clang'
./Desktop/clang
ubuntu2@ubuntu2:~$
```

oo) sed

sed stands for stream editor which usually modifies the text files non-interactively, making all the changes in a single go.

Syntax:

sed [options] command file

Options here include

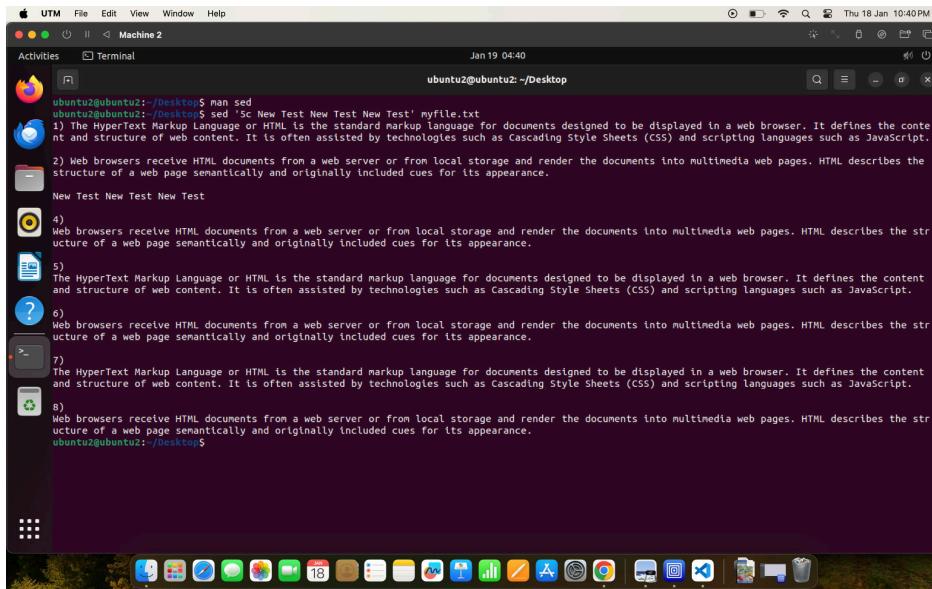
s => substitute text, d for deleting lines pattern matching,

a => appending text after a line is matching a pattern,

d => deleting a line after a line is matching a pattern,

i => inserting a text before a line is matched.

c => change the entire matching row.



A screenshot of a Mac OS X desktop environment. At the top is a dock with various application icons. Below the dock is a desktop background with a green and blue abstract pattern. In the center is a terminal window titled "Machine 2". The terminal shows the following command and its output:

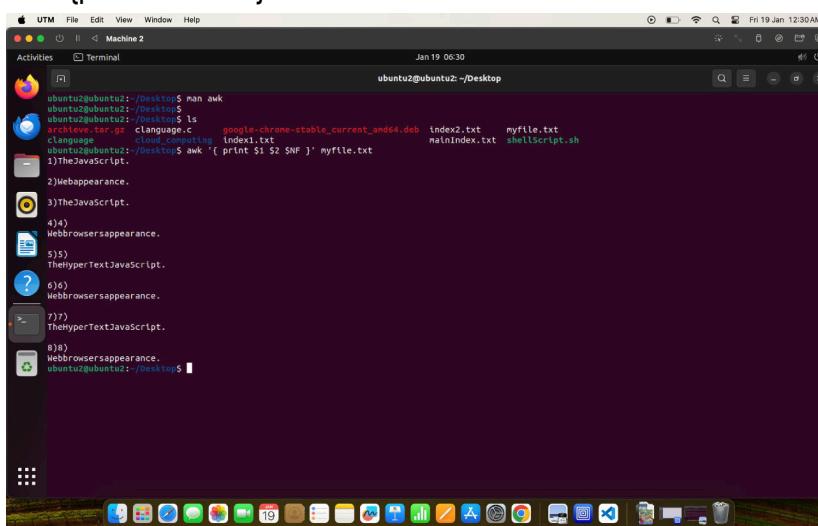
```
ubuntu2@ubuntu2:~/Desktop$ man sed
ubuntu2@ubuntu2:~/Desktop$ sed '5c New Test New Test New Test' myfile.txt
1) The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It defines the content and structure of web content. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.
2) Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for its appearance.
New Test New Test New Test
4)
5) Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for its appearance.
6)
7) The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It defines the content and structure of web content. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.
8)
Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for its appearance.
ubuntu2@ubuntu2:~/Desktop$
```

pp) awk

awk is a command line tool which is used for scanning patterns and processing text files.

Syntax:

awk '{pattern action}' file



A screenshot of a Mac OS X desktop environment. At the top is a dock with various application icons. Below the dock is a desktop background with a green and blue abstract pattern. In the center is a terminal window titled "Machine 2". The terminal shows the following command and its output:

```
ubuntu2@ubuntu2:~/Desktop$ man awk
ubuntu2@ubuntu2:~/Desktop$ ls
archive.tar.gz clangUAGE.c google-chrome-stable_current_amd64.deb index2.txt myfile.txt
clangUAGE.cloud_computing index1.txt nattyIndex.txt shellScript.sh
1)TheJavaScript.
2)Webappearance.
3)TheJavaScript.
4)
5)Webbrowsersappearance.
6)ThehyperTextJavaScript.
7)Webbrowsersappearance.
8)ThehyperTextJavaScript.
ubuntu2@ubuntu2:~/Desktop$ awk '{ print $1 $2 $NF }' myfile.txt
1)TheJavaScript.
2)Webappearance.
3)TheJavaScript.
4)
5)Webbrowsersappearance.
6)ThehyperTextJavaScript.
7)Webbrowsersappearance.
8)ThehyperTextJavaScript.
ubuntu2@ubuntu2:~/Desktop$
```

qq) diff

diff is a command line utility which is used to compare the contents of 2 files or directories.

Syntax:

diff [options] FILE1 FILE2

```

ubuntu2@ubuntu2:~/Desktop$ man diff
ubuntu2@ubuntu2:~/Desktop$ ls
archieve.tar.gz clanguage.c google-chrome-stable_current_amd64.deb index2.txt myfile.txt
clanguage   cloud_computing index1.txt mainIndex.txt shellScript.sh
ubuntu2@ubuntu2:~/Desktop$ diff index1.txt index2.txt
1c1
< HyperText Markup Language
> HYPERTEXT MARKUP LANGUAGE
ubuntu2@ubuntu2:~/Desktop$ diff index1.txt mainIndex.txt
1a2
> HYPERTEXT MARKUP LANGUAGE
ubuntu2@ubuntu2:~/Desktop$
```

The terminal shows the output of the 'diff' command comparing 'index1.txt' and 'index2.txt'. The output indicates that line 1 has been changed from 'HyperText Markup Language' to 'HYPERTEXT MARKUP LANGUAGE'. The file 'mainIndex.txt' is also listed in the directory.

rr) sort

sort command line utility which is used to sort lines of a specified file.

Syntax:

Sort [options] fileName

Here options are -n for numeric string and -r for reverse string

```

ubuntu2@ubuntu2:~/Desktop$ man sort
ubuntu2@ubuntu2:~/Desktop$ ls
archieve.tar.gz clanguage.c google-chrome-stable_current_amd64.deb index2.txt myfile.txt
clanguage   cloud_computing index1.txt mainIndex.txt shellScript.sh
ubuntu2@ubuntu2:~/Desktop$ sort -r myfile.txt
Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for its appearance.
Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for its appearance.
The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It defines the content and structure of web content. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.
HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It defines the content and structure of web content. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.
8)
6)
5)
3) The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It defines the content and structure of web content. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.
Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for its appearance.
1) The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It defines the content and structure of web content. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.

ubuntu2@ubuntu2:~/Desktop$
```

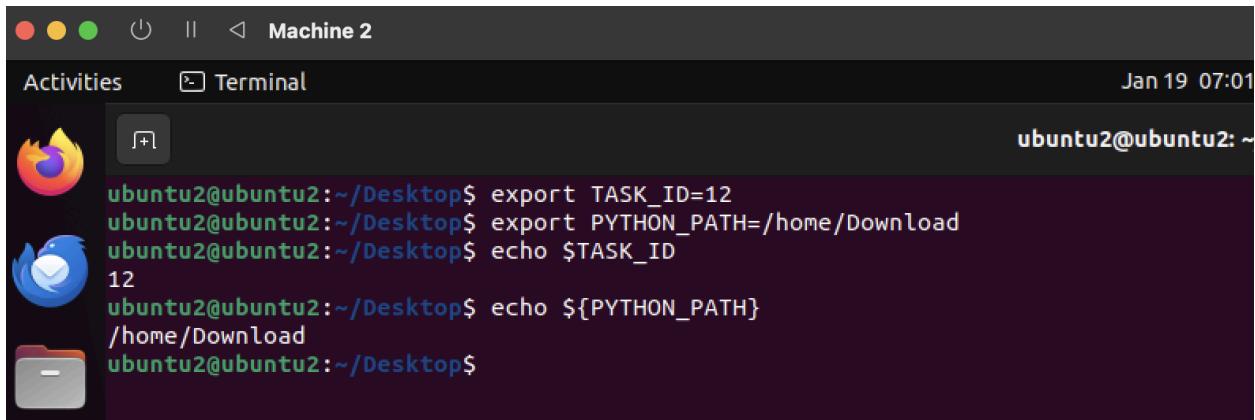
The terminal shows the output of the 'sort -r' command on 'myfile.txt'. The output lists the contents of 'myfile.txt' in reverse order. The file 'shellScript.sh' is also listed in the directory.

ss) export

The in-built command export promotes a local variable to global environment variable where this variable is accessible to all programs inside a current shell session.

Syntax:

export [name=[value]]



Activities Terminal Jan 19 07:01
ubuntu2@ubuntu2: ~

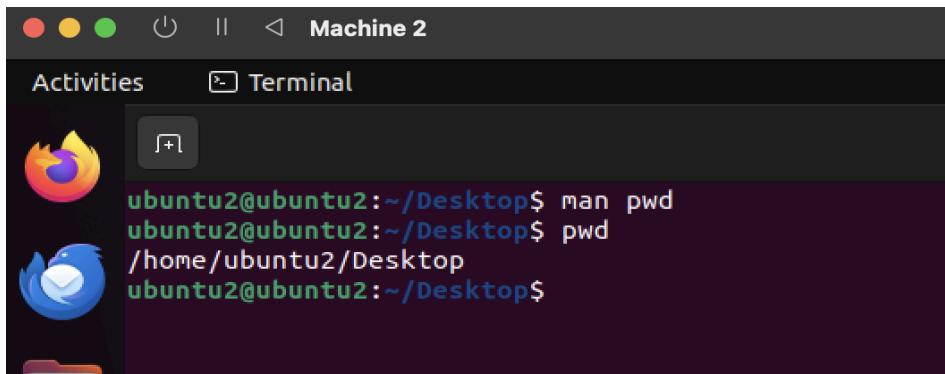
```
ubuntu2@ubuntu2:~/Desktop$ export TASK_ID=12
ubuntu2@ubuntu2:~/Desktop$ export PYTHON_PATH=/home/Download
ubuntu2@ubuntu2:~/Desktop$ echo $TASK_ID
12
ubuntu2@ubuntu2:~/Desktop$ echo ${PYTHON_PATH}
/home/Download
ubuntu2@ubuntu2:~/Desktop$
```

tt) **pwd**

The command line utility **pwd** stands for print name of current/working directory

Syntax:

pwd



Activities Terminal

```
ubuntu2@ubuntu2:~/Desktop$ man pwd
ubuntu2@ubuntu2:~/Desktop$ pwd
/home/ubuntu2/Desktop
ubuntu2@ubuntu2:~/Desktop$
```

uu) **crontab**

crontab is a command line utility which is used to execute a particular task called as cron job, at a scheduled time and intervals.

Syntax:

Min hours dayOfMonth month dayOfWeek

Min -> 0 to 59

Hours -> 0 to 23

dayOfMonth -> 1 to 31

Month -> 1 to 12

dayOfWeek -> 0 to 6

vv) mount

The mount command is used attaches a file system from a remote server or storage device. This makes the files to be accessible through the directories. This process is called mount point.

Syntax:

mount [options] device mountpoint

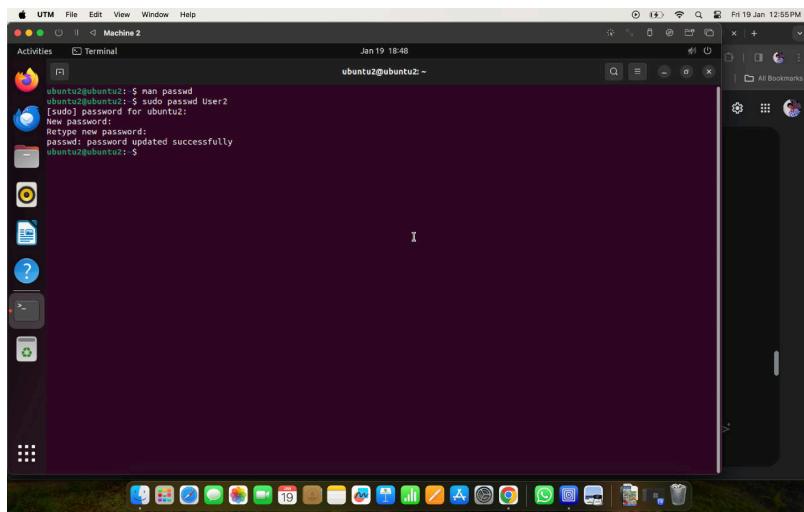
It has options like -t which explicitly indicates file system type and ro => read-only, rw => read-write.

ww) passwd

The passwd command line utility is used to manage user passwords. It is used to change, view status, set a new password and lock and unlock user accounts.

Syntax:

passwd [option] [username]



xx) uname

uname command gives the information about the operating system and hardware of the system.

Syntax:

uname [option]

It has options like

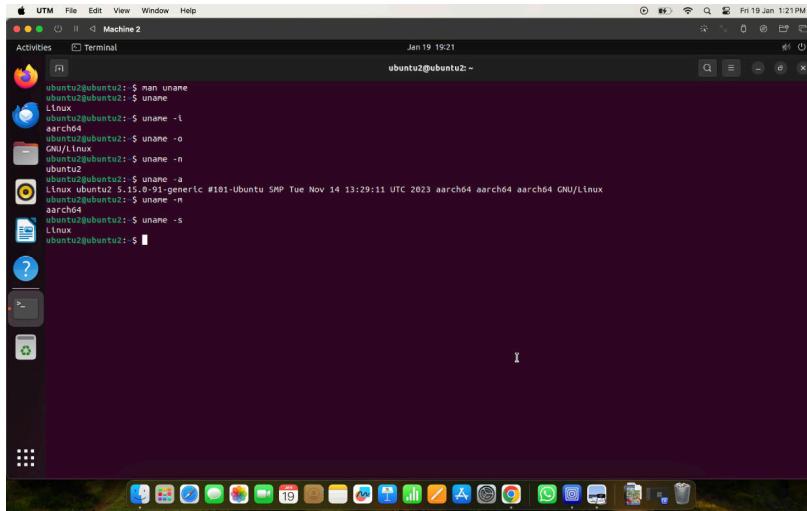
-a => print all available information of the system,

-n => print network node hostname,

-o => prints OS name,

-i => prints hardware platform,

-s => prints kernel name



```
ubuntu2@ubuntu2:~$ man uname
ubuntu2@ubuntu2:~$ uname
Linux
aarch64
Ubuntu
Ubuntu
aarch64
Ubuntu
aarch64
Ubuntu
aarch64
Ubuntu
aarch64
Ubuntu
aarch64
```

yy) whereis

It quickly identifies the binary, source code and manual pages files in a program or a command
Syntax:

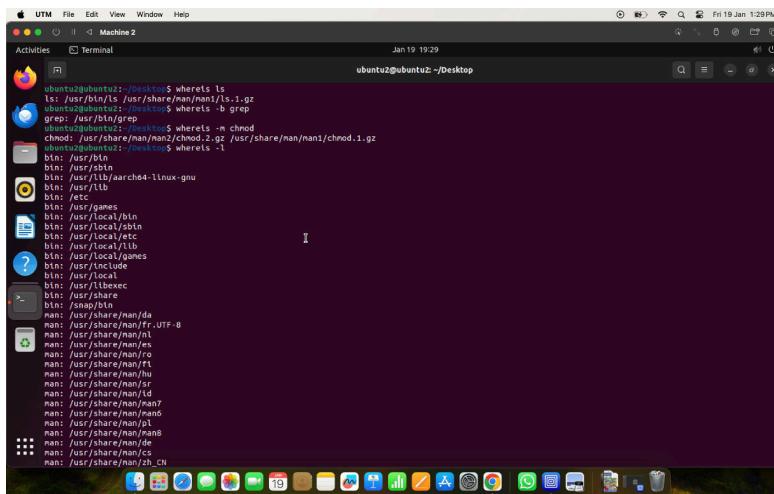
whereis [options] commandName

It has options such as

-b -> Search for only binary file

-s -> Search for only source code file

-m -> Search for only manual page file



```
ubuntu2@ubuntu2:~/Desktop$ whereis ls
ls: /bin/ls
ubuntu2@ubuntu2:~/Desktop$ whereis -b grep
grep: /usr/bin/grep
ubuntu2@ubuntu2:~/Desktop$ whereis -b grep
grep: /usr/bin/grep
chroot: /usr/share/man/man2/chroot.2.gz /usr/share/man/man1/chroot.1.gz
ubuntu2@ubuntu2:~/Desktop$ whereis -l
bin: /usr/bin
bin: /usr/bin/man
bin: /usr/lib/aarch64-linux-gnu
bin: /usr/lib
bin: /usr/libexec
bin: /usr/games
bin: /usr/local/bin
bin: /usr/local/lib
bin: /usr/local/games
bin: /usr/local/include
bin: /usr/local/libexec
bin: /usr/libexec
bin: /snap/bin
man: /usr/share/man/d
man: /usr/share/man/da
man: /usr/share/man/zh
man: /usr/share/man/zh
man: /usr/share/man/es
man: /usr/share/man/ro
man: /usr/share/man/it
man: /usr/share/man/hi
man: /usr/share/man/sr
man: /usr/share/man/ur
man: /usr/share/man/ru
man: /usr/share/man/zh
man: /usr/share/man/zh
man: /usr/share/man/zh
man: /usr/share/man/zh
man: /usr/share/man/de
man: /usr/share/man/cs
man: /usr/share/man/zh_CN
```

zz) whatis

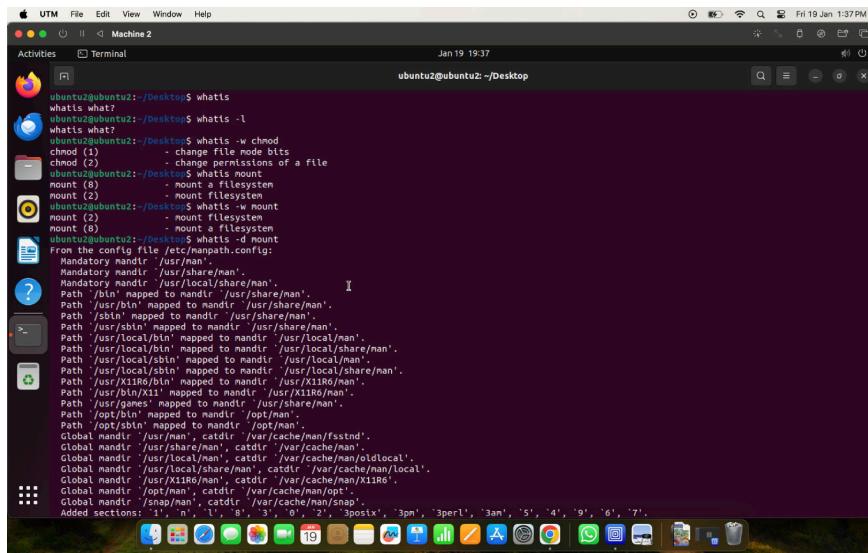
It gives a single-line brief information of a command or a function from the manual pages.

Syntax:

`whatis [options] commandName`

`-w` => This flag is used to search for complete words.

`-l` => get list all occurrences of the command or function



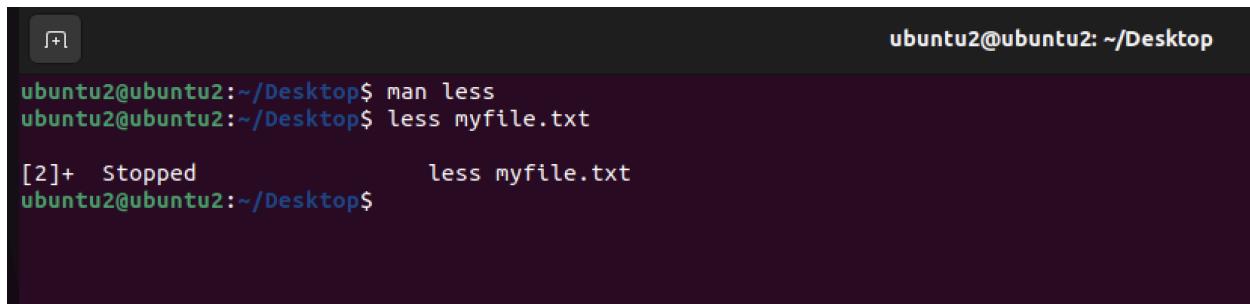
```
ubuntu2@ubuntu2:~/Desktop$ whatis
whatis
whatis what
ubuntu2@ubuntu2:~/Desktop$ whatis -l
whatis what?
ubuntu2@ubuntu2:~/Desktop$ whatis -w chmod
chmod (1)           - change file mode bits
chmod (2)           - change permissions of a file
mount (8)           - mount a filesystem
mount (2)           - mount filesystem
mount (5)           - unmount filesystem
mount (2)           - mount a filesystem
mount (8)           - mount a filesystem
ubuntu2@ubuntu2:~/Desktop$ whatis -d mount
From manpage file /etc/manpath.config:
Mandatory mandir '/usr/share/man'.
Mandatory mandir '/usr/local/share/man'.
Path '/bin' mapped to mandir '/usr/share/man'.
Path '/sbin' mapped to mandir '/usr/share/man'.
Path '/usr/sbin' mapped to mandir '/usr/share/man'.
Path '/usr/local/bin' mapped to mandir '/usr/local/share/man'.
Path '/usr/local/sbin' mapped to mandir '/usr/local/share/man'.
Path '/usr/local/usrbin' mapped to mandir '/usr/local/share/man'.
Path '/usr/X11R6/bin' mapped to mandir '/usr/X11R6/man'.
Path '/usr/bin/X11' mapped to mandir '/usr/X11R6/man'.
Path '/usr/games' mapped to mandir '/usr/share/man'.
Path '/opt/bin' mapped to mandir '/opt/man'.
Path '/opt/sbin' mapped to mandir '/opt/man'.
Global mandir '/usr/man', catdir '/var/cache/man/ssistd'.
Global mandir '/usr/local/man', catdir '/var/cache/man/local'.
Global mandir '/usr/local/share/man', catdir '/var/cache/man/local'.
Global mandir '/usr/X11R6/man', catdir '/var/cache/man/X11R6'.
Global mandir '/opt/man', catdir '/var/cache/man/opt'.
Global mandir '/snap/man', catdir '/var/cache/man/snap'.
Added sections: '1', 'n', 'l', '8', '3', '9', '2', '3posix', '3pm', '3perl', '3am', '5', '4', '9', '6', '7'.
```

aaa) less

It is a text file viewer, which helps in viewing the content of a file in one screen or page.

Syntax:

`less [options] fileName`



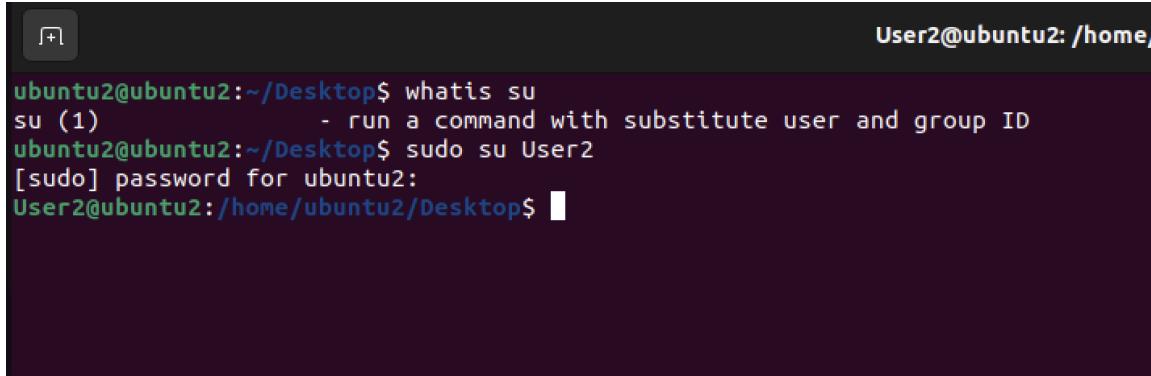
```
ubuntu2@ubuntu2:~/Desktop$ man less
ubuntu2@ubuntu2:~/Desktop$ less myfile.txt
[2]+  Stopped                  less myfile.txt
ubuntu2@ubuntu2:~/Desktop$
```

bbb) su

`su` stands for switch user, which allows you to change the current user to another user with in single shell session.

Syntax:

`su [options] username`



```
User2@ubuntu2: /home/  
ubuntu2@ubuntu2:~/Desktop$ whatis su  
su (1) - run a command with substitute user and group ID  
ubuntu2@ubuntu2:~/Desktop$ sudo su User2  
[sudo] password for ubuntu2:  
User2@ubuntu2:/home/ubuntu2/Desktop$
```

ccc) ping

Ping tells whether a host of IP is reached. It is handled by Internet Control Message Protocol (ICMP).

Syntax:

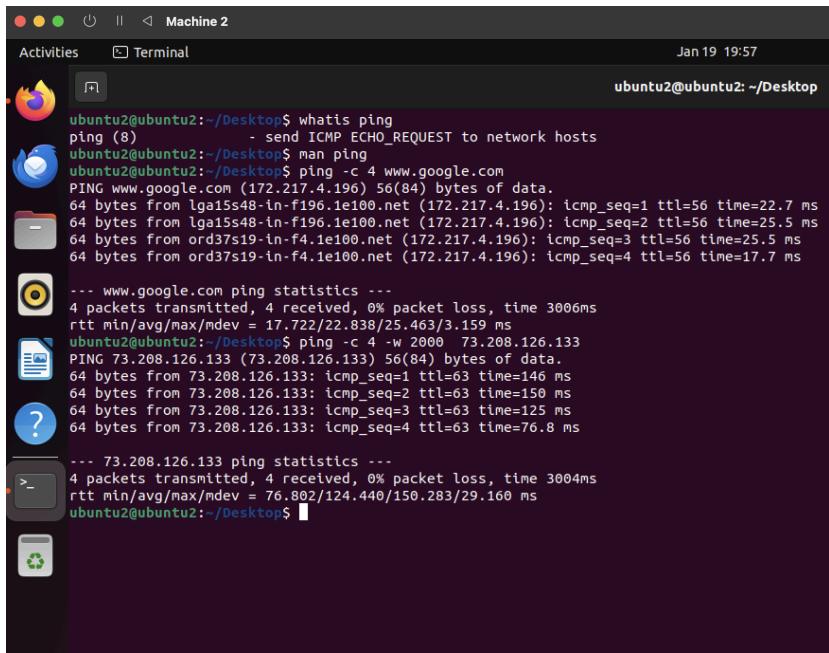
ping [options] hostName_or_IP_Address

It has options like

-c number => which servers ping for number times.

-t => It continuously ping the host until it is interrupted.

-w => wait or timeout



```
Activities Terminal Jan 19 19:57  
ubuntu2@ubuntu2:~/Desktop$ whatis ping  
ping (8) - send ICMP ECHO_REQUEST to network hosts  
ubuntu2@ubuntu2:~/Desktop$ man ping  
ubuntu2@ubuntu2:~/Desktop$ ping -c 4 www.google.com  
PING www.google.com (172.217.4.196) 56(84) bytes of data.  
64 bytes from lga15s48-in-f196.1e100.net (172.217.4.196): icmp_seq=1 ttl=56 time=22.7 ms  
64 bytes from lga15s48-in-f196.1e100.net (172.217.4.196): icmp_seq=2 ttl=56 time=25.5 ms  
64 bytes from ord37s19-in-f4.1e100.net (172.217.4.196): icmp_seq=3 ttl=56 time=25.5 ms  
64 bytes from ord37s19-in-f4.1e100.net (172.217.4.196): icmp_seq=4 ttl=56 time=17.7 ms  
  
--- www.google.com ping statistics ---  
4 packets transmitted, 4 received, 0% packet loss, time 3006ms  
rtt min/avg/max/mdev = 17.722/22.838/25.463/3.159 ms  
ubuntu2@ubuntu2:~/Desktop$ ping -c 4 -w 2000 73.208.126.133  
PING 73.208.126.133 (73.208.126.133) 56(84) bytes of data.  
64 bytes from 73.208.126.133: icmp_seq=1 ttl=63 time=146 ms  
64 bytes from 73.208.126.133: icmp_seq=2 ttl=63 time=150 ms  
64 bytes from 73.208.126.133: icmp_seq=3 ttl=63 time=125 ms  
64 bytes from 73.208.126.133: icmp_seq=4 ttl=63 time=76.8 ms  
  
--- 73.208.126.133 ping statistics ---  
4 packets transmitted, 4 received, 0% packet loss, time 3004ms  
rtt min/avg/max/mdev = 76.802/124.440/150.283/29.160 ms  
ubuntu2@ubuntu2:~/Desktop$
```

ddd) traceroute

It is a network diagnostic tool which tracks the location or path of the packet from where the data is moving towards i.e from one computer to a host of the network.

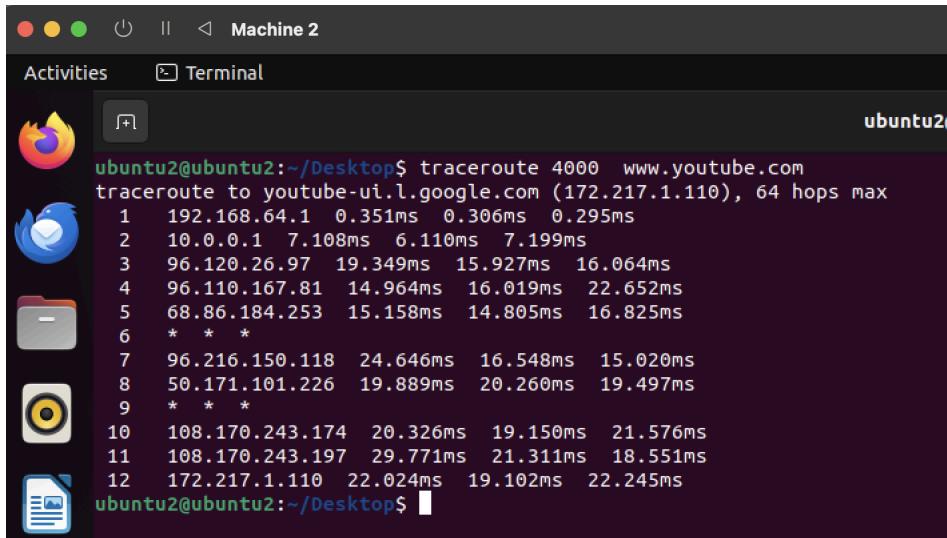
Syntax:

traceroute [options] hostName_or_IP_Address

The options include:

-c => trace a set a maximum number of hops

-w => wait or timeout



A screenshot of an Ubuntu desktop environment. The terminal window shows the command `traceroute 4000 www.youtube.com` being run. The output of the traceroute command is displayed, showing the path from the user's machine to youtube-ui.l.google.com through various routers and switches.

```
ubuntu2@ubuntu2:~/Desktop$ traceroute 4000 www.youtube.com
traceroute to youtube-ui.l.google.com (172.217.1.110), 64 hops max
 1  192.168.64.1  0.351ms  0.306ms  0.295ms
 2  10.0.0.1  7.108ms  6.110ms  7.199ms
 3  96.120.26.97  19.349ms  15.927ms  16.064ms
 4  96.110.167.81  14.964ms  16.019ms  22.652ms
 5  68.86.184.253  15.158ms  14.805ms  16.825ms
 6  * * *
 7  96.216.150.118  24.646ms  16.548ms  15.020ms
 8  50.171.101.226  19.889ms  20.260ms  19.497ms
 9  * * *
10  108.170.243.174  20.326ms  19.150ms  21.576ms
11  108.170.243.197  29.771ms  21.311ms  18.551ms
12  172.217.1.110  22.024ms  19.102ms  22.245ms
```

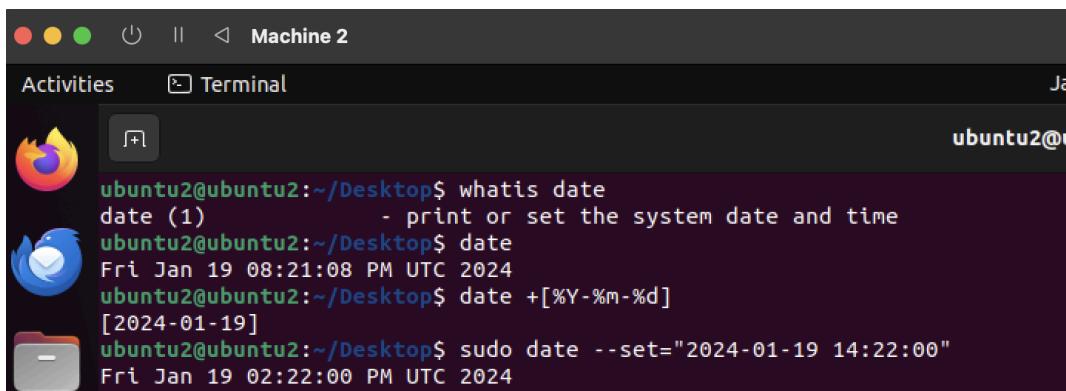
eee) date

It is used to display current date and time. It can also be formatted according to the user.

Syntax:

`date [options] [+format]`

Formatting patterns are -> %Y : for year, %m: for month, %d for Day, %H for hours, %M for minutes and %S for seconds.



A screenshot of an Ubuntu desktop environment. The terminal window shows examples of the `date` command. It first displays the help information for `date`, then shows the current date and time, and finally demonstrates how to change the system date to a specific value using the `--set` option.

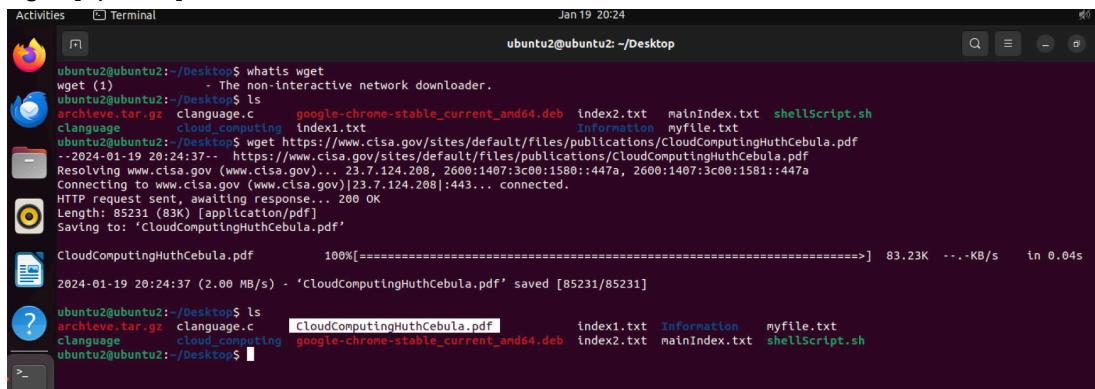
```
ubuntu2@ubuntu2:~/Desktop$ whatis date
date (1)          - print or set the system date and time
ubuntu2@ubuntu2:~/Desktop$ date
Fri Jan 19 08:21:08 PM UTC 2024
ubuntu2@ubuntu2:~/Desktop$ date +[%Y-%m-%d]
[2024-01-19]
ubuntu2@ubuntu2:~/Desktop$ sudo date --set="2024-01-19 14:22:00"
Fri Jan 19 02:22:00 PM UTC 2024
```

fff) wget

It is used to download files from internet and supports different protocols like HTTP, HTTPS and FTP, etc

Syntax:

wget [options] URL



```
Activities Terminal Jan 19 20:24
ubuntu2@ubuntu2:~/Desktop$ whatis wget
wget (1)           - The non-interactive network downloader.
ubuntu2@ubuntu2:~/Desktop$ ls
archive.tar.gz  clangage.c   CloudComputingHuthCebula.pdf  index1.txt  Information  myfile.txt
clanguage       cloud_computing  google-chrome-stable_current_amd64.deb  index2.txt  mainIndex.txt  shellScript.sh
ubuntu2@ubuntu2:~/Desktop$ wget https://www.cisa.gov/sites/default/files/publications/CloudComputingHuthCebula.pdf
--2024-01-19 20:24:37-- https://www.cisa.gov...
Resolving www.cisa.gov (www.cisa.gov)... 23.7.124.208, 2600:1407:3c00:1580::447a, 2600:1407:3c00:1581::447a
Connecting to www.cisa.gov (www.cisa.gov)|23.7.124.208|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 85231 (83K) [application/pdf]
Saving to: 'CloudComputingHuthCebula.pdf'

CloudComputingHuthCebula.pdf      100%[=====] 83.23K  ---KB/s   in 0.04s
2024-01-19 20:24:37 (2.00 MB/s) - 'CloudComputingHuthCebula.pdf' saved [85231/85231]

ubuntu2@ubuntu2:~/Desktop$ ls
archive.tar.gz  clangage.c   CloudComputingHuthCebula.pdf  index1.txt  Information  myfile.txt
clanguage       cloud_computing  google-chrome-stable_current_amd64.deb  index2.txt  mainIndex.txt  shellScript.sh
ubuntu2@ubuntu2:~/Desktop$
```

ggg) wc

The wc stands for “word count”. It gets the count of the words, characters, bytes and number of lines from a given file.

Syntax:

wc [options] [file(s)]

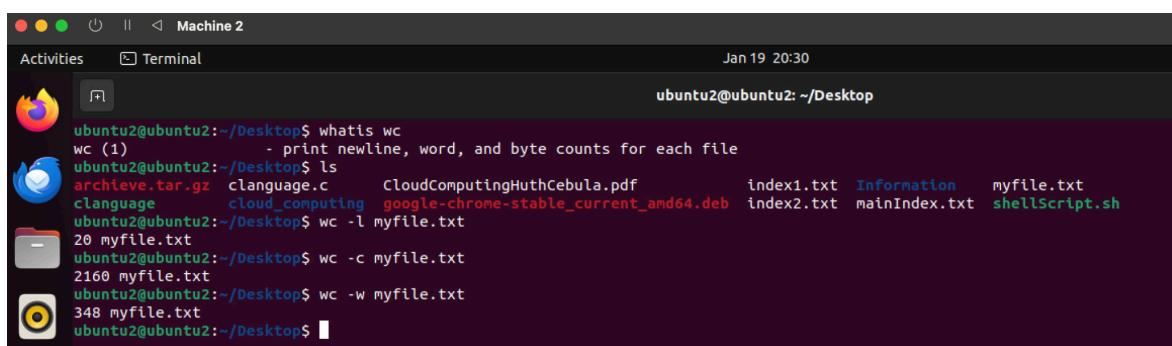
It has options such as

-w => count of number of words

-l => count of number of lines

-m => count of number of characters

-c => count of number of bytes



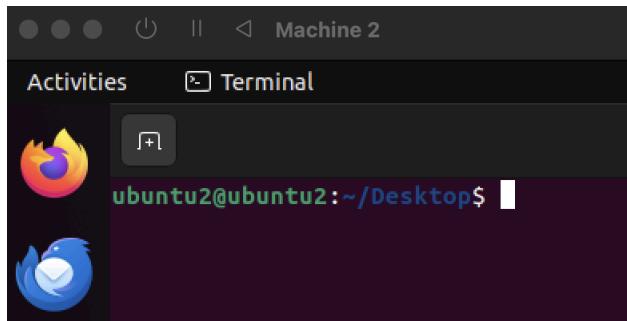
```
Activities Terminal Jan 19 20:30
ubuntu2@ubuntu2:~/Desktop$ whatis wc
wc (1)           - print newline, word, and byte counts for each file
ubuntu2@ubuntu2:~/Desktop$ ls
archive.tar.gz  clangage.c   CloudComputingHuthCebula.pdf  index1.txt  Information  myfile.txt
clanguage       cloud_computing  google-chrome-stable_current_amd64.deb  index2.txt  mainIndex.txt  shellScript.sh
ubuntu2@ubuntu2:~/Desktop$ wc -l myfile.txt
20 myfile.txt
ubuntu2@ubuntu2:~/Desktop$ wc -c myfile.txt
2160 myfile.txt
ubuntu2@ubuntu2:~/Desktop$ wc -w myfile.txt
348 myfile.txt
ubuntu2@ubuntu2:~/Desktop$
```

hhh) clear

This command is used to clear all the pre-executed commands in a terminal.

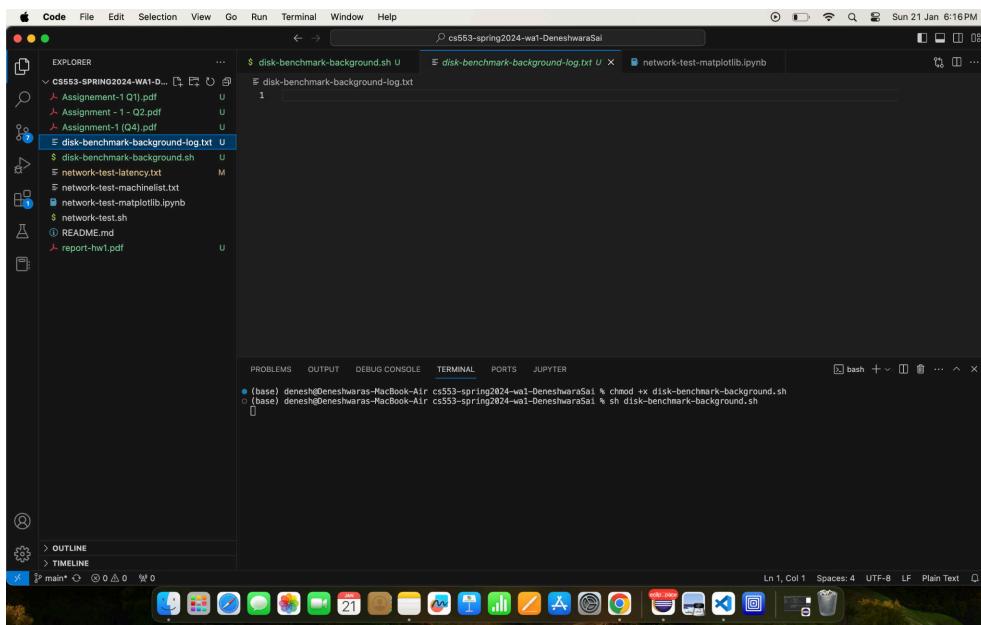
Syntax:

clear



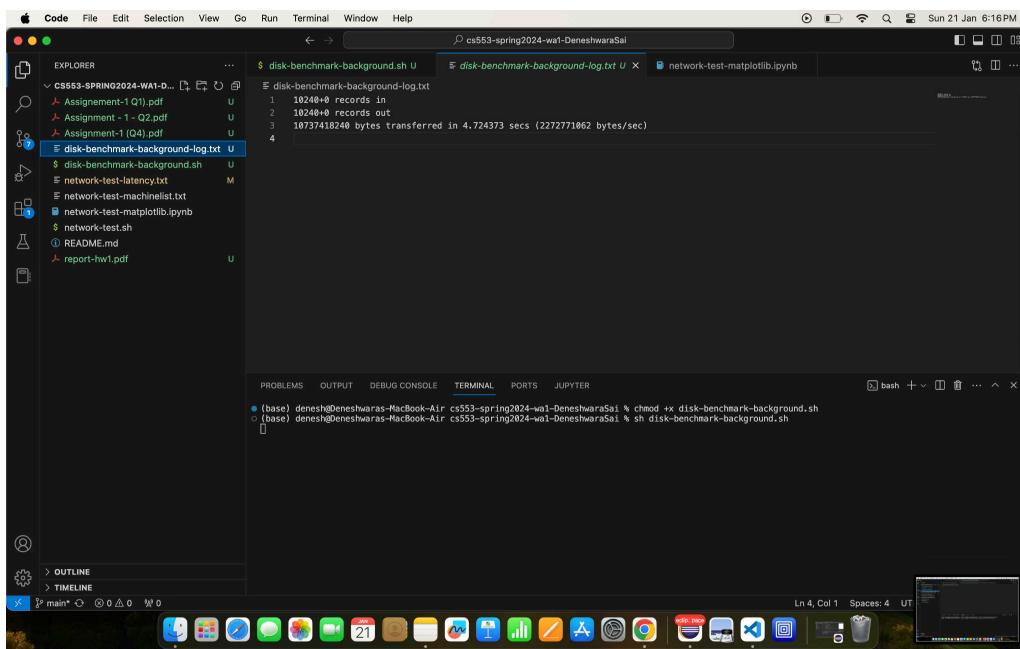
3)

a)



```
(base) denesh@Deneshwaras-MacBook-Air cs553-spring2024-wa1-DeneshwaraSai % chmod +x disk-benchmark-background.sh
(base) denesh@Deneshwaras-MacBook-Air cs553-spring2024-wa1-DeneshwaraSai % sh disk-benchmark-background.sh
```

A empty output file (**disk-benchmark-background-log.txt**) is created.



```
(base) denesh@Deneshwaras-MacBook-Air cs553-spring2024-wa1-DeneshwaraSai % chmod +x disk-benchmark-background.sh
(base) denesh@Deneshwaras-MacBook-Air cs553-spring2024-wa1-DeneshwaraSai % sh disk-benchmark-background.sh
```

The script is in the process of executing and it runs for 10 seconds. The output is printed even though the executing is continuing.

```

$ chmod +x disk-benchmark-background.sh
$ ./disk-benchmark-background.sh
1 10240+0 records in
2 10240+0 records out
3 10737418240 bytes transferred in 4.724373 secs (2272771062 bytes/sec)
4

```

The terminal output shows the command being run, the file sizes, the transfer rate, and the completion message "The execution is completed".

Now, execution is completed successfully and output is stored in **(disk-benchmark-background-log.txt)** file

3)
b)

```

$ ./network-test.sh
# This is input file which contains few domain names line by line

# This is output file which contains the domain name and avg RTT separated by a space

getPingAndCalculateRTT() {
    # get the first argument by $1
    # Given number of samples are 3
    dnsName=$1
    samples=3
    echo $dnsName
}

# Below code is used to send a specific number of ICMP (Internet Control Message Protocol) requests packets to the given IP
# -c indicates a flag that indicates number of packets to send.
# grep is used to filter/search for line at specific string : round-trip
# awk operates on per-line bases and respective value at $0 and prints
# cut is used to split and the code splits at / and gets 2nd value
awkRoundTripTime=$(nc -c $samples $dnsName | grep "round-trip" | awk '{print $0}' | cut -d '/' -f 2)

```

The terminal output shows the command being run, the input file, the output file, and the logic for calculating average round-trip time.

Lets give executable permission to network-test.sh file by **chmod +x network-test.sh** and get output by using **sh network-test.sh**.

Input File :

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows files in the "CS553-SPRING2024-WA1-DENESHWARASI" folder, including "Assignment-1-Q1.pdf", "Assignment - 1 - Q2.pdf", "Assignment-1-Q4.pdf", "disk-benchmark-background-log.txt", "disk-benchmark-background.sh", "network-test-latency.txt", "network-test-machineList.txt", "network-test-matplotlib.ipynb", "network-test.sh", and "README.md".
- Terminal:** Displays the command `chmod +x network-test.sh` being run in the terminal tab.
- Content Editor:** Shows the contents of the "network-test-machineList.txt" file, which lists 15 URLs with their response times:
 - 1 www.google.com 18.839
 - 2 www.youtube.com 23.565
 - 3 www.lit.edu 37.717
 - 4 www.oracle.com 21.525
 - 5 www.walmart.com 160.444
 - 6 www.microsoft.com 24.126
 - 7 www.apple.com 16.164
 - 8 www.primevideo.com 19.000
 - 9 www.amazon.com 17.521
 - 10 www.facebook.com 18.137
 - 11 www.linkedin.com 15.512
 - 12 www.wikipedia.org 88.558
 - 13 www.github.com 39.222
 - 14 www.udemy.com 16.655
 - 15 www.coursera.org

Output File :

The screenshot shows the VS Code interface with the following details:

- File Explorer:** Shows files in the "CS553-SPRING2024-WA1-DENESHWARASI" folder, including "Assignment-1-Q1.pdf", "Assignment - 1 - Q2.pdf", "Assignment-1-Q4.pdf", "disk-benchmark-background-log.txt", "disk-benchmark-background.sh", "network-test-latency.txt", "network-test-machineList.txt", "network-test-matplotlib.ipynb", "network-test.sh", and "README.md".
- Terminal:** Displays the command `sh network-test.sh` being run in the terminal tab.
- Content Editor:** Shows the contents of the "network-test-latency.txt" file, which lists 16 URLs with their response times:
 - 1 www.google.com 18.839
 - 2 www.youtube.com 23.565
 - 3 www.lit.edu 37.717
 - 4 www.oracle.com 21.525
 - 5 www.walmart.com 160.444
 - 6 www.microsoft.com 24.126
 - 7 www.apple.com 16.164
 - 8 www.primevideo.com 19.000
 - 9 www.amazon.com 17.521
 - 10 www.facebook.com 18.137
 - 11 www.linkedin.com 15.512
 - 12 www.wikipedia.org 88.558
 - 13 www.github.com 39.222
 - 14 www.udemy.com 16.655
 - 15 www.coursera.org
 - 16 www.coursera.org

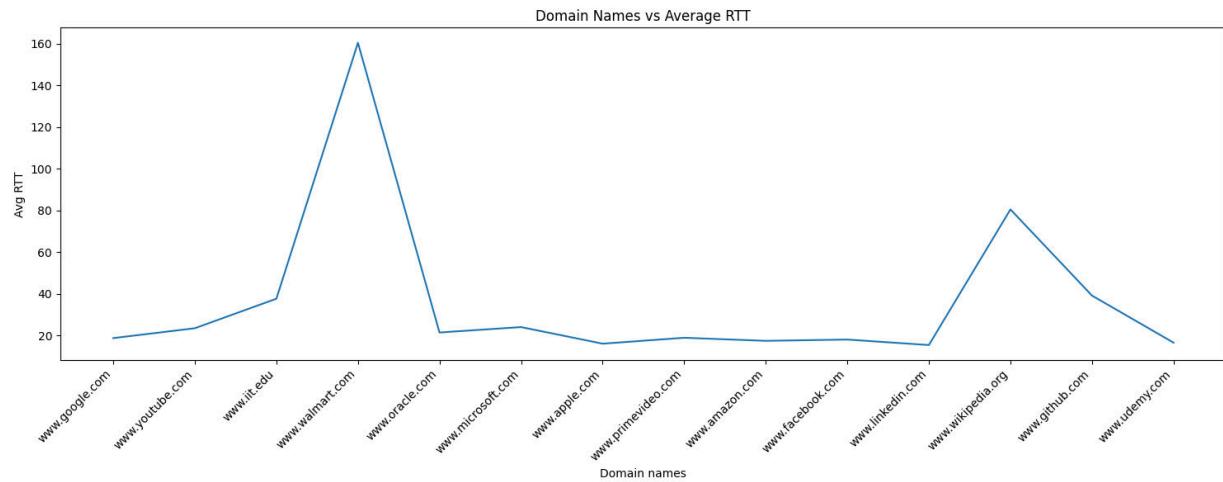
3)
c)

Use the file **network-test-matplotlib.ipynb** for running the program. For every cell use shift + enter to execute.

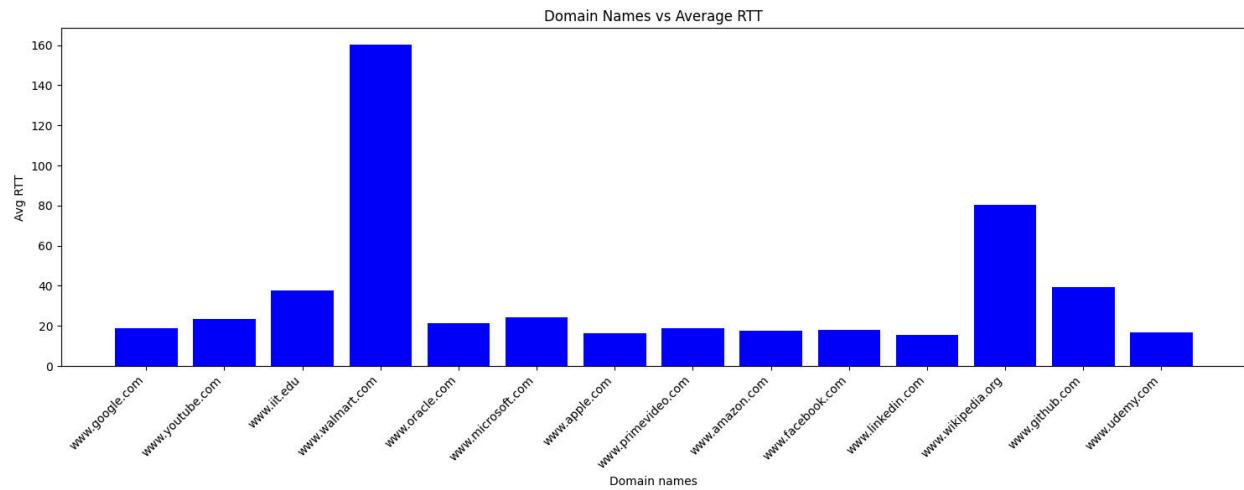
This program takes **network-test-latency.txt** as input and generates the below graphs.

The output screen indicates the domain name and its average RTTs.

Line Graph :



Bar Graph:



4)

a)

In the system configuration of the virtual machines, changing the number of processors may significantly impact their behavior. The change may directly affect performance, resource allocation, resource contention, and parallel processing.

Changing the number of processors:

- **Increasing Processors:**

This creates a virtual machine with high computing power, allowing the tasks to distribute their workload and perform parallel jobs. It leads to improved performance and latency and makes use of cores efficiently for parallel jobs.

- **Decreasing processors:**

Allocating less number of processors reduces the computational power of virtual machines slows the performance and cannot handle parallel tasks.

Scenario for Maximum Processors:

Scenario: Considering a high-traffic web microservice system that is responsible for handling incoming HTTP requests for a web application and performing various tasks like calculation, billing, authentication, and CRUD operations.

To handle such a high volume of incoming requests and ensure consistency in performance, it is mandatory to maximize the allocation of processors to the microservice.

By maximizing the allocation of processors, the microservice can distribute its workload across all threads and cores to handle the high traffic efficiently, ensuring responsiveness, transaction management, and scalability.

Scenario for Minimum Processors:

Scenario: Background Task Processor

Considering a microservice responsible for executing background tasks, such as generating reports, transmitting emails, or processing data asynchronously. These tasks are not scheduled and can be processed in a deferred manner.

Here, we can restrict or minimize the usage of processors as it is an asynchronous job and executes the jobs one after the other. Doing this helps optimize the resource utilization, and allows for more virtual machines to coexist on the host without causing resource contention.

Potential issues with setting a maximum processor:

- **Resource Contention:**

Allocating a maximum number of processors to multiple virtual machines on the same host and if physical cores are insufficient then it leads to resource contention. It results in degradation of performance and dependency on the core to free up.

- **Resource wastage:**
Allocating more processors for fewer computational jobs may lead to resource wastage, may not utilize the hardware up to the mark, and may cost high.
 - **Management complexities :**
Managing virtual machines with a high volume of processors is a complex task for cloud administrators. Troubleshooting, monitoring, and resource allocation become more difficult, which leads to an imbalance and unstable virtual environment.
 - **Cost/ budget:**
In cloud environments, when dealing with or allocating a maximum number of resources may result in higher expenses than expected.
-

4)

b)

Virtualization: Virtualization is a process of creating a virtual environment or instance of something such as operating systems, cloud servers, storage devices and network resources.

Paravirtualization:

A paravirtualization is a technique that allows the guest operating system running inside a virtual machine to communicate more efficiently with the hypervisor by using hypercalls.

Hypervisor:

A hypervisor is otherwise called as VMM(Virtual Machine Monitor), is a firmware or software layer that enables the virtualization of physical hardware, allowing multiple OS to run on multiple virtual machines. This hypervisor is at the middle of hardware and OS or VMs, managing and allocation resources to the host system to the individual virtual machines.

None, Legacy, Minimal, Hyper-V, and KVM refer to different types of paravirtualization interfaces.

Feature	None	Legacy	Minimal	Hyper-V	KVM
Hypervisor	There is no hypervisor	It uses Type-1 Hypervisor.	It uses Type-1 Hypervisor.	It uses Type-1 Hypervisor.	It uses Type-1 Hypervisor.
Purpose	There is no virtualization.	Server virtualization.	Lightweight virtualization.	Server virtualization	Server virtualization
Host OS	It directly communicates with the Hardware.	N/A (Bare-metal)	N/A (Bare-metal)	windows server.	Linux
Guest OS	N/A	Various OS	Various OS	Windows/ Linux	Various OS

Supported OS	None doesn't support any specific OS because it is not a virtualization platform.	Legacy virtualization platforms may have very limited options of support for modern operating systems.	These platforms supports a quite range of operating systems but with minimal features.	Hyper-V is primarily supports microsoft and windows based operating systems.	KVM supports wide range of operating systems including windows, linux.
Open source	These are not open source.	These may or may not be a open source platforms.	Minimal virtualization platforms may or may not be open source and it depends on the specific implementation.	Hyper-V is not open source as it is owned by microsoft.	KVM is open source and a part of linux kernel. It has strong community development and support.
Integration	None doesnot integrate with any other virtualization technologies.	Legacy may not integrate with modern technologies and cloud platforms.	Minimal virtualization have limited integration features, capabilities with other systems.	Hyper- V can integrate well with microsoft related technologies and often used in windows centric environments	KVM integrates well in any linux ecosystems.
Performance	N/A	High	Moderate	High	High
Management	N/A	Using GUI	Minmal (CLI)	GUI (Hyper-V manager)	CLI (virsh)
Live migration	No live migration	Yes, it has live migration.	Yes, it has live migration.	Yes, it has live migration.	Yes, it has live migration.
Resource	N/A	Dedicated resources	Shared resources	Shared resources	Shared resources
Networking	N/A	Networking through virtual switches.	Networking through Bridged.	Networking through virtual switches.	Bridged networking.
Examples	Basic	VMware	Xen.	Microsoft	KVM/ QEMU

	General purpose PC.	ESXi (Elastic Sky X integrated)		Hyper-V on windows server machine.	(Quick emulator) on a linux server.
--	---------------------	---------------------------------	--	------------------------------------	-------------------------------------

KVM (Kernel-based Virtual machine) are best recommended paravirtualization option for Ubuntu Linux.

Reasons :

The Linux kernels are integrated with KVM ensuring seamless compatibility and performance optimization that provides a scalable and robust virtualization solution to the Linux environments. KVM is an open-source virtualization technology, which is benefitted from regular updates, patches, maintenance, and strong community support, ensuring reliability, scalability, and stability. It offers both command line and Graphical User Interface(GUI) tools for flexibility management, includes security features from the Linux kernel, and is broadly inherited with the Linux virtualization ecosystem.

4)

c) The differences between IDE, SATA and NVMe :

Features	IDE Controller	SATA Controller	NVMe Controller
Full Forms and Compatibility	IDE stands for Integrated Drive Electronics controllers are widely used in older computer systems.	SATA stands for Serial Advanced Technology Attachment controllers are widely used in modern systems.	NVMe stands for Non-Volatile Memory Express are widely designed for high-performance solid state drives(SSDs).
Data Transfer	They provide a slower data transfer rates and limitations on drive size and operations. Rate is upto 133MB/s (IDE)	They provide a faster data transfer rates, larger drive size and better performance. Rates are upto 600MB/s in SATA-II and 6Gb/s in SATA-III	They provide ultra fast data transfer rates, high performance storage controller for SSDs and low latency. Rates are upto 32Gb/s and beyond
Cable Length	Limited by cable interference and signal degradation	Longer cable length and minimal interference	Shorter cable length and less susceptible to interference.
Hot-swapping	IDE does not support.	SATA supports hot-swapping.	NVMe supports hot-swapping.

Power Consumption	Consumes more power.	Consumes moderate power.	Power efficient and consumes less power.
Latency	Higher latency compared to SATA and NVMe.	Lower latency than IDE.	Lowest latency, gives high performance.
Error Handling	Limited error-handling techniques.	Improved error-checking and correction	Advanced error-handling and has a mechanism of reliability.
Form Factor	Larger and bulkier	More streamlined and smaller.	Compact, Suitable for space-constrained systems.
Scenario/Examples	Using an older version of desktop computer with IDE hard drive. Like a PC in early 2000s.	Using a computer system such as consumer-graded systems which is equipped with SATA SSD for reliable and fast storage.	Systems demanding high-speed and high-performance oriented data transfer machines such as gaming PCs, Play /work stations and high-end laptops.

4)

d) The difference between NAT, Bridged Adaptor, Internal Network and Host-only Network:

Features	NAT (Network Address translation)	Bridged Adaptor	Internal Network	Host-only network
Network Type	It is shared with host's IP	It is directly connected with physical network.	It creates closed network for virtual machines.	It allows network between hosts and virtual machines.
IP Address	Virtual machines use the IP Address for external Communication.	It gets its own IP Address on the physical network.	Virtual machines have their own IP address within the internal network.	In the host-only network, Virtual machines and host communicate using IP addresses.

Internet Access	Virtual machines can access the internet from host's connection.	Virtual machines get direct internet access from physical network.	No direct internet access	No direct internet access
Isolation	Virtual machines can communicate with each other.	Virtual machines are part of physical network.	Virtual machines can communicate within internet network.	Virtual machines and host can communicate within host-only network.
Communication Scope	Virtual machines can communicate with external networks.	VMs can communicate externally and locally.	VMs are limited to communicate within network.	VMs are limited to communicate within network.
DHCP usage	It uses hosts DHCP	Can use either DHCP or static IP assignment.	VMs uses internal DHCP	VMs either use DHCP or static IP assignment.
Security	Adds security by hiding VMs behind the host's IP.	The VMs are exposed to local network and they do require security measures.	Provides isolation with minimized external threads.	They are more secure than Bridged mode.
Examples	Running multiple virtual machines with Internet access.	VMs as web services on the local network.	Testing environments with isolated VMs.	Secure network for host and VMs.

4)
e)

The difference between USB(Universal Serial Bus) 1.1, 2.0 and 3.0 are :

Features	USB 1.1 Controller	USB 2.0 Controller	USB 3.0 Controller
Introduced Year	It was introduced in 1998	It was introduced in 2000	It was introduced in 2008
Data Transfer	The data transfer rate is upto 12Mbps.	The data transfer rate is upto 480Mbps.	The data transfer rate is upto 5Gbps.

Backward Compatability	USB 1.1 has a backward compatibility with USB 1.0.	USB 2.0 has a backward compatibility with all previous version like USB 1.1, 1.0	USB 3.0 has a backward compatibility with all previous version like USB 2.0, 1.1, 1.0.
Maximum Power output	Low-power devices with 2.5mA	Standard power devices with 500mA	Standard with 900mA and battery charging with 1.5A.
Connection Types	Standard-A and standard-B.	Standard-A and standard-B.	Standard-A, standard-B, Micro-B, Type-A and Type-B Superspeed.