Lab Cycle: 3
Experiment No.: 2
Date: 7-06-2022

Aim : Illustrate the steps involved in installing the LAMP Stack on a Linux machine. Deploy phpMyadmin.

Steps to install Lamp Stack

A. Steps to installing apache2 server, mariadb and php.

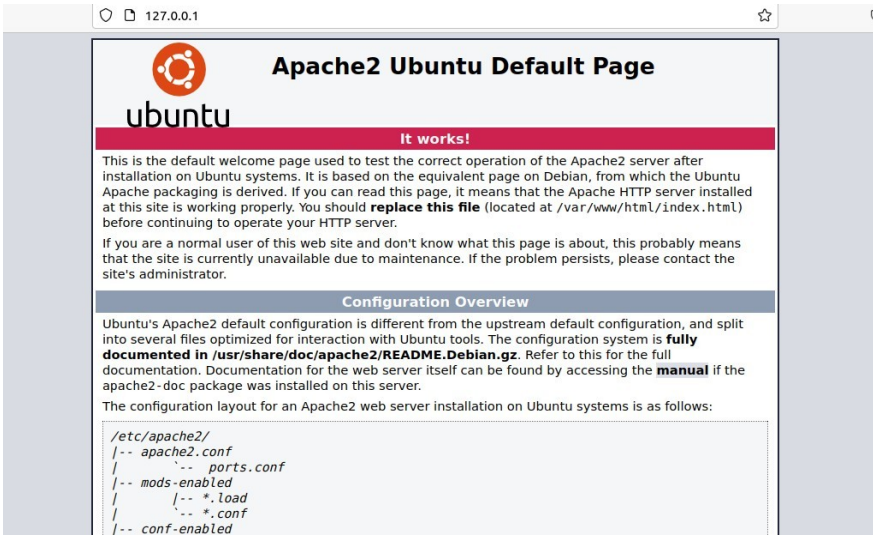
1. Initially update the repositories information
   sudo apt update

2. Install apache2 Web Server
   sudo apt install apache2

```
user@user-VirtualBox:~$ sudo apt install apache2
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libapr1 libaprutil1
  libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.2-0
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapr1 libaprutil1
  libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.2-0
0 upgraded, 9 newly installed, 0 to remove and 67 not upgraded.
Need to get 1,820 kB of archives.
After this operation, 7,945 kB of additional disk space will be used.
```

```
user@user-VirtualBox:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor prese>
     Active: active (running) since Thu 2022-06-09 13:58:36 IST; 1min 15s ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 3419 (apache2)
      Tasks: 55 (limit: 5854)
     Memory: 4.7M
     CGroup: /system.slice/apache2.service
             ├─3419 /usr/sbin/apache2 -k start
             ├─3421 /usr/sbin/apache2 -k start
             └─3422 /usr/sbin/apache2 -k start

Jun 09 13:58:36 user-VirtualBox systemd[1]: Starting The Apache HTTP Server...
Jun 09 13:58:36 user-VirtualBox apachectl[3418]: AH00558: apache2: Could not re>
Jun 09 13:58:36 user-VirtualBox systemd[1]: Started The Apache HTTP Server.
lines 1-15/15 (END)
```

3. Check 127.0.0.1 from the browser, we can see the start page of Apache.



4. Install Mariadb Server and Client

sudo apt install mariadb-server mariadb-client

```
user@user-VirtualBox:~$ sudo apt install mariadb-server mariadb-client
[sudo] password for user:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  galera-3 gawk libaio1 libcgi-fast-perl libcgi-pm-perl
  libconfig-inifiles-perl libdbd-mysql-perl libdbi-perl libfcgi-perl
  libhtml-template-perl libreadline5 libsigsegv2 libsnappy1v5
  libterm-readkey-perl mariadb-client-10.3 mariadb-client-core-10.3
  mariadb-common mariadb-server-10.3 mariadb-server-core-10.3 socat
Suggested packages:
  gawk-doc libclone-perl libmldbm-perl libnet-daemon-perl
  libsql-statement-perl libipc-sharedcache-perl mailx mariadb-test tinyca
The following NEW packages will be installed:
  galera-3 gawk libaio1 libcgi-fast-perl libcgi-pm-perl
  libconfig-inifiles-perl libdbd-mysql-perl libdbi-perl libfcgi-perl
  libhtml-template-perl libreadline5 libsigsegv2 libsnappy1v5
  libterm-readkey-perl mariadb-client mariadb-client-10.3
  mariadb-client-core-10.3 mariadb-common mariadb-server mariadb-server-10.3
  mariadb-server-core-10.3 socat
0 upgraded, 22 newly installed, 0 to remove and 67 not upgraded.
Need to get 20.2 MB of archives.
After this operation, 167 MB of additional disk space will be used.
```

5. Make Mariadb Secure by setting password, disabling anonymous users etc.

sudo mysql_secure_installation

```
user@user-VirtualBox:~$ sudo mysql_secure_installation
[sudo] password for user:

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user.  If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

You already have a root password set, so you can safely answer 'n'.

Change the root password? [Y/n] n
 ... skipping.
```

```
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

You already have a root password set, so you can safely answer 'n'.

Change the root password? [Y/n] n
 ... skipping.

By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] y
 ... Success!

Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] y
 ... Success!
```

6. Install PHP and commonly used packages

   sudo apt install php libapache2-mod-php php-opcache php-cli php-gd php-curl php-mysql

```
user@user-VirtualBox:~$ sudo apt install php libapache2-mod-php php-opcache php-cli php-gd php
-curl php-mysql
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'php7.4-opcache' instead of 'php-opcache'
The following additional packages will be installed:
```

B. Steps to install phpMyAdmin

1. First login to mysql.

```
user@user-VirtualBox:~$ sudo mysql -uroot
[sudo] password for user:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 36
Server version: 10.3.34-MariaDB-0ubuntu0.20.04.1 Ubuntu 20.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

2. Create user ‘phpmyadmin’ and grant privileges to phpmyadmin.

   CREATE USER 'phpmyadmin'@'localhost' IDENTIFIED BY 'phpmyadminpassword';
   GRANT ALL PRIVILEGES ON *.* TO 'phpmyadmin'@'localhost';
   FLUSH PRIVILEGES;

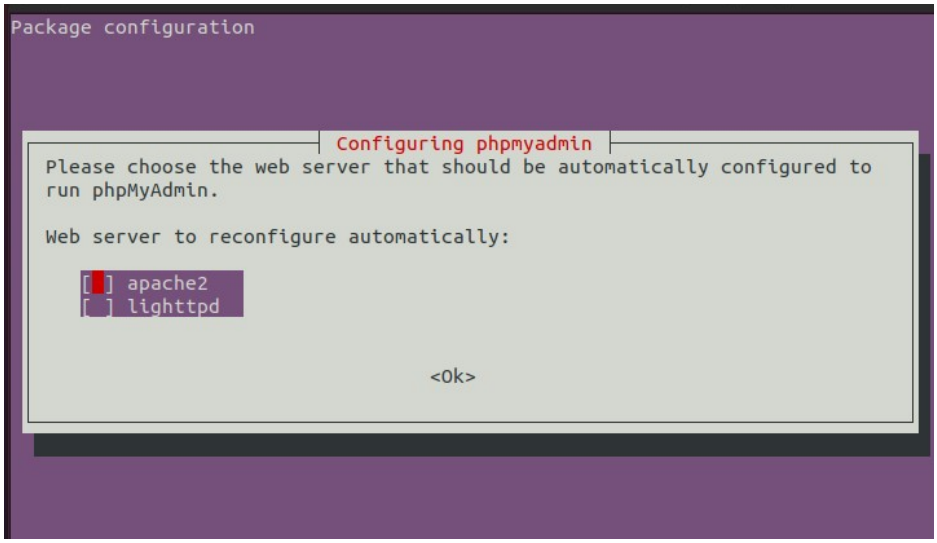
```
MariaDB [(none)]> CREATE USER 'phpmyadmin'@'localhost' IDENTIFIED BY 'phpmyadminpassword';
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON *.* TO 'phpmyadmin'@'localhost';
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> exit;
Bye
```

3. Install phpmyadmin package

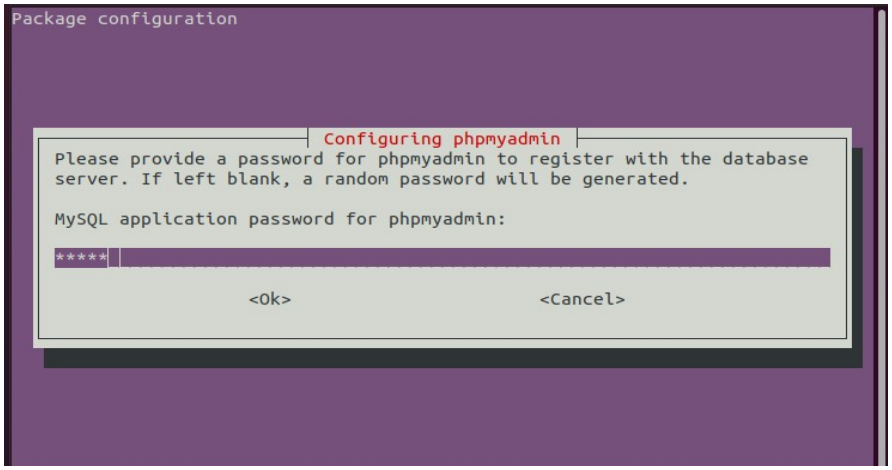
sudo apt install phpmyadmin php-mbstring php-zip php-json

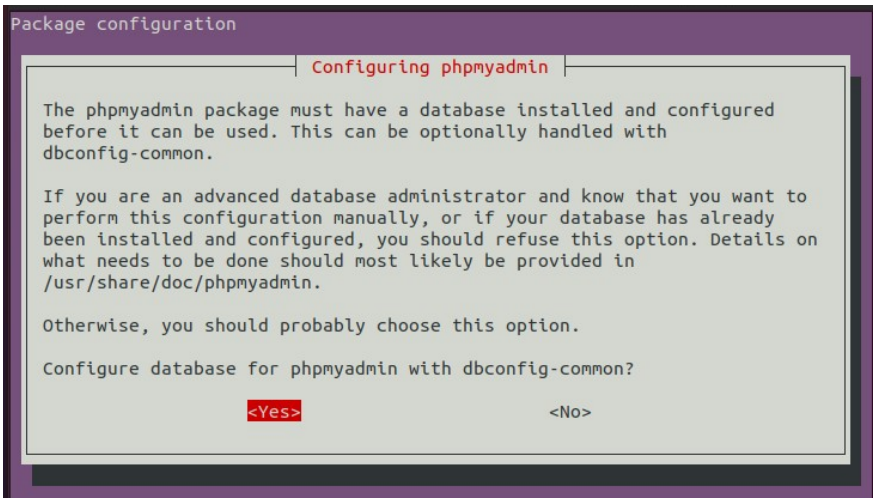
```
user@user-VirtualBox:~$ sudo apt install phpmyadmin php-mbstring php-zip php-json
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  dbconfig-common dbconfig-mysql icc-profiles-free javascript-common libjs-jquery
  libjs-openlayers libjs-sphinxdoc libjs-underscore libonig5 libzip5 php-bz2
  php-google-recaptcha php-phpmyadmin-motranslator php-phpmyadmin-shapefile
  php-phpmyadmin-sql-parser php-phpseclib php-psr-cache php-psr-container php-psr-log
  php-symfony-cache php-symfony-cache-contracts php-symfony-expression-language
  php-symfony-service-contracts php-symfony-var-exporter php-tcpdf php-twig
  php-twig-extensions php-xml php7.4-bz2 php7.4-mbstring php7.4-xml php7.4-zip
Suggested packages:
```

4. Next installation prompt to the automatic setup of the webserver to be used alongside phpMyAdmin. The choice is Apache2 .



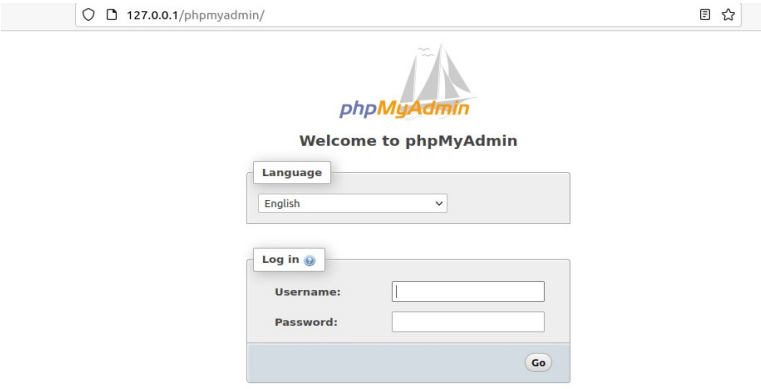
5. Next step is the configuration of a phpMyAdmin database. Select db-configuration and set password. Choose yes to enable the database configuration steps.





6. Reload the apache2 server.
   sudo systemctl restart apache2

7. Access phpmyadmin from browser using ‘127.0.0.1/phpmyadmin’.



8. Login to phpMyAdmin by using login credentials.

Lab Cycle: 3
Experiment No.: 3
Date: 14-06-2022

Aim : Create a Docker container of ubuntu:20.04. The container should be pre installed with nano. Use Dockerfile and build, create and start commands.

1. Install Docker Engine

```
ubuntu3@ubuntu3-VirtualBox:~$ sudo apt install docker.io
[sudo] password for ubuntu3:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd git git-man liberror-perl pigz runc ubuntu-fan
Suggested packages:
```

2.Create a Dockerfile with the following contents

```
  GNU nano 4.8
FROM ubuntu:20.04
RUN apt-get update
RUN apt -y install nano
```

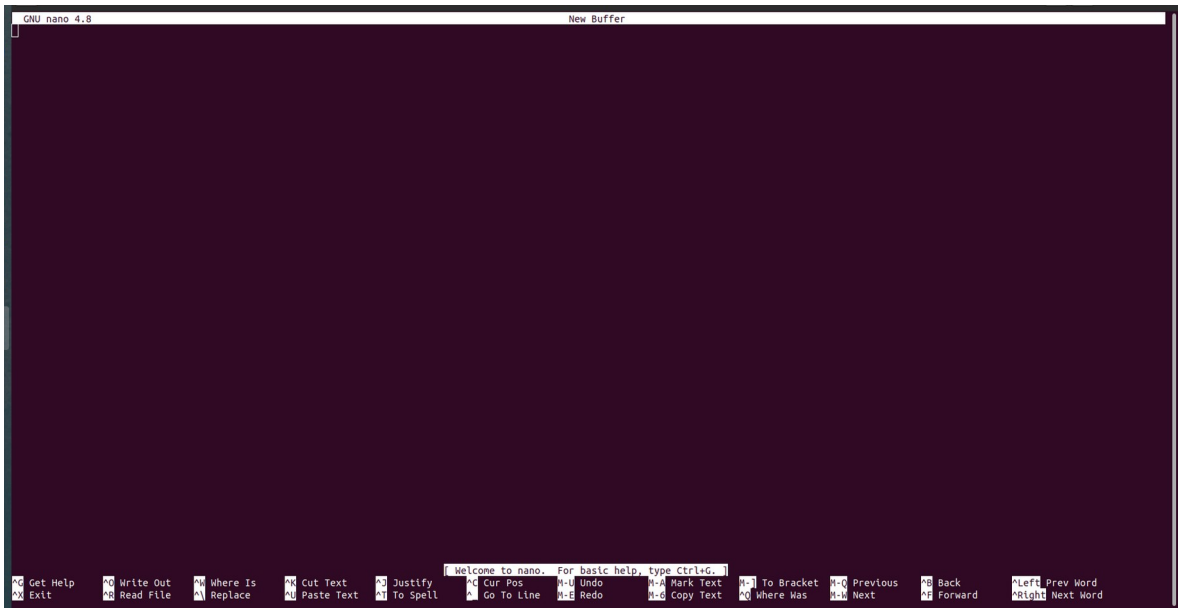
3. Build the image with a suitable name.

```
ubuntu3@ubuntu3-VirtualBox:~$ sudo docker build -t myubuntu .
Sending build context to Docker daemon  535.7MB
Step 1/3 : FROM ubuntu:20.04
20.04: Pulling from library/ubuntu
d7bfe07ed847: Pull complete
Digest: sha256:fd92c36d3cb9b1d027c4d2a72c6bf0125da82425fc2ca37c414d4f010180dc19
Status: Downloaded newer image for ubuntu:20.04
 ---> 20fffa419e3a
Step 2/3 : RUN apt-get update
 ---> Running in ba0624951a11
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:2 http://archive.ubuntu.com/ubuntu focal InRelease [265 kB]
Get:3 http://security.ubuntu.com/ubuntu focal-security/universe amd64 Packages [880 kB]
Get:4 http://security.ubuntu.com/ubuntu focal-security/restricted amd64 Packages [1286 kB]
Get:5 http://archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:6 http://security.ubuntu.com/ubuntu focal-security/multiverse amd64 Packages [27.5 kB]
Get:7 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [1931 kB]
Get:8 http://archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:9 http://archive.ubuntu.com/ubuntu focal/main amd64 Packages [1275 kB]
```

4. Create the container with a suitable name then start and attach to the container

```
Successfully tagged myubuntu:latest
ubuntu3@ubuntu3-VirtualBox:~$ sudo docker create --name container -i -t myubuntu
8baf59d8a3acc8de807fd33db707897d38e5d0977cbac5a6599385828f60e8aa
ubuntu3@ubuntu3-VirtualBox:~$ sudo docker start -i -a container
root@8baf59d8a3ac:/# nano
```

5. Check whether nano is installed in the container, by giving the command nano.



Lab Cycle 3
Experiment No.: 4
Date: 5-07-2022

Aim: Conduct a study of the important options and uses of the following commands:

1. ping
2. traceroute
3. tcpdump
4. ip
5. nc

1. ping

ping stands for Packet Internet Groper. Linux ping is one of the most used network troubleshooting commands. It basically checks for the network connectivity between two nodes.

a) Checking the connection to google.com using ping.

```
user1@user1-VirtualBox:~$ ping google.com
PING google.com (142.250.195.206) 56(84) bytes of data.
64 bytes from maa03s42-in-f14.1e100.net (142.250.195.206): icmp_seq=1 ttl=55 time=39.6 ms
64 bytes from maa03s42-in-f14.1e100.net (142.250.195.206): icmp_seq=2 ttl=55 time=42.6 ms
64 bytes from maa03s42-in-f14.1e100.net (142.250.195.206): icmp_seq=3 ttl=55 time=42.7 ms
64 bytes from maa03s42-in-f14.1e100.net (142.250.195.206): icmp_seq=4 ttl=55 time=38.3 ms
64 bytes from maa03s42-in-f14.1e100.net (142.250.195.206): icmp_seq=5 ttl=55 time=36.3 ms
64 bytes from maa03s42-in-f14.1e100.net (142.250.195.206): icmp_seq=6 ttl=55 time=23.5 ms
64 bytes from maa03s42-in-f14.1e100.net (142.250.195.206): icmp_seq=7 ttl=55 time=36.3 ms
64 bytes from maa03s42-in-f14.1e100.net (142.250.195.206): icmp_seq=8 ttl=55 time=30.6 ms
64 bytes from maa03s42-in-f14.1e100.net (142.250.195.206): icmp_seq=9 ttl=55 time=26.3 ms
^C
--- google.com ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8013ms
rtt min/avg/max/mdev = 23.458/35.135/42.660/6.491 ms
```

b) Specify the count and limit the response packets.

```
user1@user1-VirtualBox:~$ ping -c 5 google.com
PING google.com (142.250.195.78) 56(84) bytes of data.
64 bytes from maa03s38-in-f14.1e100.net (142.250.195.78): icmp_seq=1 ttl=55 time=33.7 ms
64 bytes from maa03s38-in-f14.1e100.net (142.250.195.78): icmp_seq=2 ttl=55 time=43.9 ms
64 bytes from maa03s38-in-f14.1e100.net (142.250.195.78): icmp_seq=3 ttl=55 time=16.8 ms
64 bytes from maa03s38-in-f14.1e100.net (142.250.195.78): icmp_seq=4 ttl=55 time=42.7 ms
64 bytes from maa03s38-in-f14.1e100.net (142.250.195.78): icmp_seq=5 ttl=55 time=16.1 ms

--- google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4006ms
rtt min/avg/max/mdev = 16.074/30.632/43.912/12.108 ms
```

c) Wait interval seconds between sending each packet.

```
user1@user1-VirtualBox:~$ ping -c 5 -i 3 google.com
PING google.com (216.58.200.174) 56(84) bytes of data.
64 bytes from nrt12s11-in-f174.1e100.net (216.58.200.174): icmp_seq=1 ttl=55 time=63.5 ms
64 bytes from nrt12s11-in-f174.1e100.net (216.58.200.174): icmp_seq=2 ttl=55 time=56.0 ms
64 bytes from nrt12s11-in-f174.1e100.net (216.58.200.174): icmp_seq=3 ttl=55 time=65.1 ms
64 bytes from nrt12s11-in-f174.1e100.net (216.58.200.174): icmp_seq=4 ttl=55 time=83.4 ms
64 bytes from nrt12s11-in-f174.1e100.net (216.58.200.174): icmp_seq=5 ttl=55 time=65.2 ms

--- google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 12009ms
rtt min/avg/max/mdev = 56.025/66.644/83.405/9.035 ms
```

d) Print timestamp before each line.

```
user1@user1-VirtualBox:~$ ping -c 5 -D gogle.com
PING gogle.com (142.250.195.164) 56(84) bytes of data.
[1656996893.768292] 64 bytes from maa03s41-in-f4.1e100.net (142.250.195.164): icmp_seq=1 ttl=55 time=18.2 ms
[1656996894.771165] 64 bytes from maa03s41-in-f4.1e100.net (142.250.195.164): icmp_seq=2 ttl=55 time=19.6 ms
[1656996895.769869] 64 bytes from maa03s41-in-f4.1e100.net (142.250.195.164): icmp_seq=3 ttl=55 time=16.3 ms
[1656996896.773452] 64 bytes from maa03s41-in-f4.1e100.net (142.250.195.164): icmp_seq=4 ttl=55 time=16.0 ms
[1656996897.779093] 64 bytes from maa03s41-in-f4.1e100.net (142.250.195.164): icmp_seq=5 ttl=55 time=20.2 ms

--- gogle.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4009ms
rtt min/avg/max/mdev = 15.996/18.070/20.237/1.711 ms
```

2. traceroute

It is used to troubleshoot the network. It displays the route that a packet takes to reach the requested target. It starts at the first router and uses a series of UDP probe packets with increasing IP time-to-live (TTL) or hop count values to determine the sequence of routers that must be traversed to reach the target host. Each host in the path is sent three “probes” and the return times are reported. If a host does not respond within five seconds, traceroute prints an asterisk. The traceroute command sends UDP packets. It can as well send TCP or ICMP packets.

a) traceroute to **“google.com”**.

```
user1@user1-VirtualBox:~$ traceroute google.com
traceroute to google.com (142.250.77.238), 30 hops max, 60 byte packets
 1  _gateway (10.9.0.1)  0.348 ms  0.347 ms  0.400 ms
 2  172.17.17.17 (172.17.17.17)  0.233 ms  0.216 ms  0.200 ms
 3  2.2.2.2 (2.2.2.2)  0.296 ms  0.280 ms  0.264 ms
 4  14.139.188.81 (14.139.188.81)  0.812 ms  0.965 ms  1.299 ms
 5  * * *
 6  * * *
 7  * * *
 8  10.119.73.122 (10.119.73.122)  19.924 ms  19.895 ms  19.881 ms
 9  72.14.195.128 (72.14.195.128)  19.863 ms 72.14.213.20 (72.14.213.20)  17.757 ms 72.14.195.12
8 (72.14.195.128)  18.168 ms
10  * * *
11  108.170.253.97 (108.170.253.97)  18.320 ms 209.85.247.250 (209.85.247.250)  16.970 ms 142.25
0.236.156 (142.250.236.156)  16.905 ms
12  74.125.242.131 (74.125.242.131)  18.117 ms 108.170.253.121 (108.170.253.121)  18.051 ms 74.1
25.242.130 (74.125.242.130)  17.986 ms
13  72.14.239.10 (72.14.239.10)  97.415 ms 72.14.239.58 (72.14.239.58)  88.925 ms 64.233.174.3 (
64.233.174.3)  18.405 ms
14  72.14.239.58 (72.14.239.58)  74.161 ms 74.125.244.193 (74.125.244.193)  78.775 ms 72.14.239.
58 (72.14.239.58)  87.670 ms
15  172.253.66.107 (172.253.66.107)  68.567 ms 74.125.244.193 (74.125.244.193)  78.560 ms 108.17
0.225.88 (108.170.225.88)  71.914 ms
16  74.125.244.193 (74.125.244.193)  51.239 ms  51.164 ms 142.251.54.75 (142.251.54.75)  48.156
ms
17  del11s09-in-f14.1e100.net (142.250.77.238)  52.054 ms  53.154 ms  53.508 ms
```

b) Applying Numeric mode: print IP addresses instead of hostnames.

```
user1@user1-VirtualBox:~$ traceroute -n google.com
traceroute to google.com (142.250.195.78), 30 hops max, 60 byte packets
 1  10.9.0.1  0.302 ms  0.300 ms  0.333 ms
 2  172.17.17.17  0.131 ms  0.116 ms  0.168 ms
 3  2.2.2.2  0.215 ms  0.211 ms  0.193 ms
 4  14.139.188.81  0.617 ms  1.111 ms  1.186 ms
 5  * * *
 6  * * *
 7  * * *
 8  10.119.73.122  16.669 ms  17.064 ms  17.516 ms
 9  72.14.195.128  17.491 ms 72.14.213.20  18.345 ms 72.14.195.128  17.809 ms
10  * * *
11  142.251.55.62  23.285 ms 142.251.55.240  20.384 ms 142.250.228.186  20.639 ms
12  142.250.195.78  20.202 ms  25.859 ms 74.125.242.131  25.744 ms
```

3. tcpdump

It captures the traffic that is passing through the network interface and displays it. It is the most commonly used tool among network administrators for troubleshooting network issues and security testing. Use ctrl+c to stop capturing.

a) Capture Packets from Specific Interface.

```
user1@user1-VirtualBox:~$ sudo tcpdump -i enp0s3
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
05:21:33.108317 IP6 fe80::20b:82ff:feb8:2043.dhcpv6-client > ff02::1:2.dhcpv6-server: dhcp6
 solicit
05:21:33.109166 IP user1-VirtualBox.36207 > dns.google.domain: 36176+ [1au] PTR? 2.0.0.0.1.
0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.2.0.f.f.ip6.arpa. (101)
05:21:33.139017 8c:3b:ad:65:98:12 (oui Unknown) > 8d:3b:ad:65:98:12 (oui Unknown), ethertyp
e Unknown (0x88b7), length 66:
        0x0000:  000a f700 0101 0000 0016 b0af 0018 0000  ................
        0x0010:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0020:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0030:  ec1b fafb                                ....
05:21:33.147982 IP dns.google.domain > user1-VirtualBox.36207: 36176 NXDomain 0/1/1 (165)
05:21:33.148125 IP user1-VirtualBox.36207 > dns.google.domain: 36176+ PTR? 2.0.0.0.1.0.0.0.
0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.2.0.f.f.ip6.arpa. (90)
05:21:33.162603 LLDP, length 126
05:21:33.187193 IP dns.google.domain > user1-VirtualBox.36207: 36176 NXDomain 0/1/0 (154)
05:21:33.188484 IP user1-VirtualBox.57361 > dns.google.domain: 26375+ [1au] PTR? 3.4.0.2.8.
b.e.f.f.f.2.8.b.0.2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f.ip6.arpa. (101)
05:21:33.237966 ARP, Request who-has 10.9.0.13 tell _gateway, length 46
05:21:33.621492 ARP, Request who-has 10.9.0.30 tell 10.9.0.131, length 46
05:21:34.238218 ARP, Request who-has 10.9.0.13 tell _gateway, length 46
05:21:34.621553 ARP, Request who-has 10.9.0.30 tell 10.9.0.131, length 46
05:21:35.238003 ARP, Request who-has 10.9.0.13 tell _gateway, length 46
```

```
05:21:43.603633 IP dns.google.domain > user1-VirtualBox.39478: 62386 NXDomain 0/0/1 (52)
05:21:43.603738 IP user1-VirtualBox.39478 > dns.google.domain: 62386+ PTR? 152.0.9.10.in-addr.arpa. (41)
05:21:43.617521 ARP, Request who-has 10.9.0.30 tell 10.9.0.131, length 46
05:21:43.619106 IP dns.google.domain > user1-VirtualBox.39478: 62386 NXDomain 0/0/0 (41)
05:21:43.619669 IP user1-VirtualBox.43118 > dns.google.domain: 64658+ [1au] PTR? 255.7.9.10.in-addr.arpa. (52
05:21:43.634670 IP dns.google.domain > user1-VirtualBox.43118: 64658 NXDomain 0/0/1 (52)
05:21:43.634778 IP user1-VirtualBox.43118 > dns.google.domain: 64658+ PTR? 255.7.9.10.in-addr.arpa. (41)
05:21:43.649489 IP dns.google.domain > user1-VirtualBox.43118: 64658 NXDomain 0/0/0 (41)
05:21:44.667935 ARP, Request who-has 10.9.0.30 tell 10.9.0.131, length 46
05:21:45.624692 ARP, Request who-has 10.9.0.30 tell 10.9.0.131, length 46
^C
58 packets captured
79 packets received by filter
21 packets dropped by kernel
```

b) Capture Only N Number of Packets.

```
user1@user1-VirtualBox:~$ sudo tcpdump -c 5 -i enp0s3
[sudo] password for user1:
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
06:11:55.100372 IP 10.9.0.176.52408 > 239.255.255.250.1900: UDP, length 125
06:11:55.101240 IP user1-VirtualBox.55680 > dns.google.domain: 9258+ [1au] PTR? 250.255.255.239.in-addr.arpa. (57)
06:11:55.141630 IP dns.google.domain > user1-VirtualBox.55680: 9258 NXDomain 0/1/1 (114)
06:11:55.141743 IP user1-VirtualBox.55680 > dns.google.domain: 9258+ PTR? 250.255.255.239.in-addr.arpa. (46)
06:11:55.212962 IP dns.google.domain > user1-VirtualBox.55680: 9258 NXDomain 0/1/0 (103)
5 packets captured
17 packets received by filter
0 packets dropped by kernel
```

c) Print the list of the network interfaces available on the system and on which tcpdump can capture packets.

```
user1@user1-VirtualBox:~$ sudo tcpdump -D
1.enp0s3 [Up, Running]
2.lo [Up, Running, Loopback]
3.any (Pseudo-device that captures on all interfaces) [Up, Running]
4.docker0 [Up]
5.bluetooth-monitor (Bluetooth Linux Monitor) [none]
6.nflog (Linux netfilter log (NFLOG) interface) [none]
7.nfqueue (Linux netfilter queue (NFQUEUE) interface) [none]
```

d) Write the raw packets to file rather than parsing and printing them out.

```
user1@user1-VirtualBox:~$ sudo tcpdump -i enp0s3 -w tcpdumpresult
tcpdump: listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
^C21 packets captured
31 packets received by filter
0 packets dropped by kernel
```

e) Read packets from file (which was created with the -w option)

```
user1@user1-VirtualBox:~$ sudo tcpdump -r tcpdumpresult
reading from file tcpdumpresult, link-type EN10MB (Ethernet)
01:28:07.333554 ARP, Request who-has 10.12.0.1 tell 10.12.0.187, length 46
01:28:07.403511 IP 10.9.1.7.netbios-dgm > 10.9.7.255.netbios-dgm: UDP, length 209
01:28:07.685742 IP 10.9.0.154.53006 > 239.255.255.250.1900: UDP, length 174
01:28:07.687350 IP 10.9.0.154.53006 > 239.255.255.250.1900: UDP, length 174
01:28:07.729204 ARP, Request who-has 192.168.10.100 tell 192.168.10.155, length 46
01:28:07.757782 8c:3b:ad:65:98:12 (oui Unknown) > 8d:3b:ad:65:98:12 (oui Unknown), ethertyp
e Unknown (0x88b7), length 66:
        0x0000:  000a f700 0101 0000 0012 85a6 0018 0000  ................
        0x0010:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0020:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0030:  8286 7b20                                ..{.
01:28:07.896345 ARP, Request who-has 10.12.0.1 tell 10.12.0.167, length 46
01:28:08.126241 IP 10.9.0.132.39482 > 239.255.255.250.1900: UDP, length 405
01:28:08.127314 IP 10.9.0.132.39482 > 239.255.255.250.1900: UDP, length 477
01:28:08.254001 ARP, Request who-has 192.168.10.100 tell 192.168.10.161, length 46
01:28:08.289470 IP 10.90.90.90.62992 > 255.255.255.255.62976: UDP, length 317
01:28:08.379321 IP 10.90.90.90.62992 > 255.255.255.255.62976: UDP, bad length 333 > 317
01:28:08.612662 STP 802.1w, Rapid STP, Flags [Learn, Forward, Agreement], bridge-id 8000.34
:a8:4e:04:22:aa.8048, length 47
01:28:08.689380 IP 10.9.0.154.53006 > 239.255.255.250.1900: UDP, length 174
01:28:08.691134 IP 10.9.0.154.53006 > 239.255.255.250.1900: UDP, length 174
01:28:08.753177 ARP, Request who-has 192.168.10.100 tell 192.168.10.155, length 46
01:28:08.920439 ARP, Request who-has 10.12.0.1 tell 10.12.0.167, length 46
01:28:09.033745 IP 10.9.1.1.57375 > 239.255.255.250.1900: UDP, length 172
01:28:09.037086 IP 10.9.0.147.55875 > 239.255.255.250.1900: UDP, length 174
01:28:09.085712 IP 10.9.0.147.55878 > 239.255.255.250.1900: UDP, length 175
01:28:09.277980 ARP, Request who-has 192.168.10.100 tell 192.168.10.161, length 46
```

4. ip

This is the latest and updated version of ifconfig command. It is a handy tool for configuring the network interfaces for linux administrators. It can be used to assign and remove addresses, take the interfaces up or down, and much more useful tasks.

a) Display the installed interface on our system.

```
user1@user1-VirtualBox:~$ ip -c link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default q
len 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT group
 default qlen 1000
    link/ether 08:00:27:34:00:ff brd ff:ff:ff:ff:ff:ff
```

b) show all IP addresses associated on all network devices.

```
user1@user1-VirtualBox:~$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:34:00:ff brd ff:ff:ff:ff:ff:ff
    inet 10.9.1.2/21 brd 10.9.7.255 scope global dynamic noprefixroute enp0s3
       valid_lft 27202sec preferred_lft 27202sec
    inet6 fe80::77c1:89cc:4cbd:5d55/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:92:bc:d9:db brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
       valid_lft forever preferred_lft forever
```

c) The ‘set’ option with up and down arguments is used to start and stop a network interface.

i. To start the interface, execute **sudo ip link set enp0s3 up**

```
user1@user1-VirtualBox:~$ sudo ip link set enp0s3 up
user1@user1-VirtualBox:~$ ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT group default qlen 1000
    link/ether 08:00:27:34:00:ff brd ff:ff:ff:ff:ff:ff
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN mode DEFAULT group default
    link/ether 02:42:02:48:de:09 brd ff:ff:ff:ff:ff:ff
```

ii. To stop the interface, execute **sudo ip link set enp0s3 down**

```
user1@user1-VirtualBox:~$ sudo ip link set enp0s3 down
user1@user1-VirtualBox:~$ ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp0s3: <BROADCAST,MULTICAST> mtu 1500 qdisc fq_codel state DOWN mode DEFAULT group default qlen 1000
    link/ether 08:00:27:34:00:ff brd ff:ff:ff:ff:ff:ff
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN mode DEFAULT group default
    link/ether 02:42:92:bc:d9:db brd ff:ff:ff:ff:ff:ff
```

5. nc

nc(Netcat) is one of the powerful networking tool, security tool or network monitoring tool. It acts like cat command over a network. It is generally used for the following purposes:

1.Operation related to TCP, UDP or UNIX-domain sockets

2. Port Scanning

3. Port listening

4. Port redirection

5. Open Remote connections

6. Read/Write data across network

7. Network debugging

8. Network daemon testing

9. Simple TCP proxies

10. A Socks or HTTP Proxy Command for ssh

It is designed to be a reliable "back-end" tool, used directly or driven by other programs and scripts.

Enter the following command on the terminal inorder to,
a) send an HTTP request

```
user1@user1-VirtualBox:~$ nc google.co.in 80
GET/HTTP/1.1/index.html
HTTP/1.0 400 Bad Request
Content-Type: text/html; charset=UTF-8
Referrer-Policy: no-referrer
Content-Length: 1555
Date: Fri, 08 Jul 2022 09:28:45 GMT

<!DOCTYPE html>
<html lang=en>
  <meta charset=utf-8>
  <meta name=viewport content="initial-scale=1, minimum-scale=1, width=device-width">
  <title>Error 400 (Bad Request)!!1</title>
  <style>
    *{margin:0;padding:0}html,code{font:15px/22px arial,sans-serif}html{background:#fff;col
or:#222;padding:15px}body{margin:7% auto 0;max-width:390px;min-height:180px;padding:30px 0
15px}* > body{background:url(//www.google.com/images/errors/robot.png) 100% 5px no-repeat;p
adding-right:205px}p{margin:11px 0 22px;overflow:hidden}ins{color:#777;text-decoration:none
}a img{border:0}@media screen and (max-width:772px){body{background:none;margin-top:0;max-w
idth:none;padding-right:0}}#logo{background:url(//www.google.com/images/branding/googlelogo
/1x/googlelogo_color_150x54dp.png) no-repeat;margin-left:-5px}@media only screen and (min-r
esolution:192dpi){#logo{background:url(//www.google.com/images/branding/googlelogo/2x/googl
elogo_color_150x54dp.png) no-repeat 0% 0%/100% 100%;-moz-border-image:url(//www.google.com/
images/branding/googlelogo/2x/googlelogo_color_150x54dp.png) 0}}@media only screen and (-we
bkit-min-device-pixel-ratio:2){#logo{background:url(//www.google.com/images/branding/google
logo/2x/googlelogo_color_150x54dp.png) no-repeat;-webkit-background-size:100% 100%}}#logo{d
isplay:inline-block;height:54px;width:150px}
  </style>
  <a href=//www.google.com/><span id=logo aria-label=Google></span></a>
  <p><b>400.</b> <ins>That’s an error.</ins>
  <p>Your client has issued a malformed or illegal request.  <ins>That’s all we know.</ins>
```

b) start listening on a port, open two terminal windows:

i. Terminal 1 for listening

```
user1@user1-VirtualBox:~$ nc -l 2000
hi
welcome
thank you
```

ii. Terminal 2 sending request

```
user1@user1-VirtualBox:~$ nc 127.0.0.1 2000
hi
welcome
thank you
```

c) perform port scanning,

i. scanning a single port

```
user1@user1-VirtualBox:~$ nc -v -w 2 -z 127.0.0.1 8000
Connection to 127.0.0.1 8000 port [tcp/*] succeeded!
user1@user1-VirtualBox:~$ nc -v -w 2 -z 127.0.0.1 2000
Connection to 127.0.0.1 2000 port [tcp/cisco-sccp] succeeded!
```

ii. scanning a multiple ports

```
user1@user1-VirtualBox:~$ nc -v -w 2 -z 127.0.0.1 2000 4000
Connection to 127.0.0.1 2000 port [tcp/cisco-sccp] succeeded!
nc: connect to 127.0.0.1 port 4000 (tcp) failed: Connection refused
```

iii. scanning a range of ports

```
user1@user1-VirtualBox:~$ nc -v -w 2 -z 127.0.0.1 8080-8100
nc: connect to 127.0.0.1 port 8080 (tcp) failed: Connection refused
nc: connect to 127.0.0.1 port 8081 (tcp) failed: Connection refused
nc: connect to 127.0.0.1 port 8082 (tcp) failed: Connection refused
nc: connect to 127.0.0.1 port 8083 (tcp) failed: Connection refused
nc: connect to 127.0.0.1 port 8084 (tcp) failed: Connection refused
nc: connect to 127.0.0.1 port 8085 (tcp) failed: Connection refused
nc: connect to 127.0.0.1 port 8086 (tcp) failed: Connection refused
nc: connect to 127.0.0.1 port 8087 (tcp) failed: Connection refused
Connection to 127.0.0.1 8088 port [tcp/omniorb] succeeded!
nc: connect to 127.0.0.1 port 8089 (tcp) failed: Connection refused
nc: connect to 127.0.0.1 port 8090 (tcp) failed: Connection refused
nc: connect to 127.0.0.1 port 8091 (tcp) failed: Connection refused
nc: connect to 127.0.0.1 port 8092 (tcp) failed: Connection refused
nc: connect to 127.0.0.1 port 8093 (tcp) failed: Connection refused
nc: connect to 127.0.0.1 port 8094 (tcp) failed: Connection refused
nc: connect to 127.0.0.1 port 8095 (tcp) failed: Connection refused
nc: connect to 127.0.0.1 port 8096 (tcp) failed: Connection refused
nc: connect to 127.0.0.1 port 8097 (tcp) failed: Connection refused
nc: connect to 127.0.0.1 port 8098 (tcp) failed: Connection refused
nc: connect to 127.0.0.1 port 8099 (tcp) failed: Connection refused
nc: connect to 127.0.0.1 port 8100 (tcp) failed: Connection refused
```