COMPUTER NETWORKS LAB - 1

NAME : KOKILA K N
SRN : PES2201800625
Week no : 1
Date : 31/Aug

STUDY AND UNDERSTAND THE BASIC NETWORKING TOOLS - WIRESHARK, TCPDUMP, PING, TRACEROUTE AND NETCAT.

Objectives

Learn and Understand Network Tools

1. Wireshark

• Perform and analyse Ping PDU capture

• Examine HTTP packet capture

• capture using filter

Analyze HTTP packet

2. Netcat

• Establish communication between client and server

• Transfer files

3. Tcpdump

• Capture packets

4. Ping

• Test the connectivity between 2 systems

5. Traceroute

• Perform traceroute checks

6. Nmap

• Explore an entire network

TASK 1: LINUX INTERFACE CONFIGURATION (IFCONFIG / IP COMMAND)

Step 1: To display status of all active network interfaces.

```
kokilareddy@kokilas-MacBook-Air ~ % ip addr show
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
        inet 127.0.0.1/8 lo0
        inet6 ::1/128
        inet6 fe80::1/64 scopeid 0x1
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        ether 14:c2:13:04:44:a4
        inet6 fe80::e3:6ab1:a36d:8558/64 secured scopeid 0x4
        inet 192.168.43.27/24 brd 192.168.43.255 en0
        inet6 2409:4071:238c:380d:1428:ec11:fe3c:c857/64 autoconf secured
        inet6 2409:4071:238c:380d:185e:ff98:8722:305d/64 autoconf temporary
en1: flags=8963<UP,BROADCAST,SMART,RUNNING,PROMISC,SIMPLEX,MULTICAST> mtu 1500
        ether 82:18:5b:c4:78:40
        inet 10.0.4.50/8 brd 10.255.255.255 en1
awdl0: flags=8943<UP,BROADCAST,RUNNING,PROMISC,SIMPLEX,MULTICAST> mtu 1484
        ether 86:9e:b7:b1:34:0c
        inet6 fe80::849e:b7ff:feb1:340c/64 scopeid 0x8
llw0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        ether 86:9e:b7:b1:34:0c
        inet6 fe80::849e:b7ff:feb1:340c/64 scopeid 0x9
utun0: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1380
        inet6 fe80::67e1:9c68:405b:c504/64 scopeid 0xa
utun1: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 2000
        inet6 fe80::6455:9b0c:8679:a2e5/64 scopeid 0xb
```

```
kokilareddy@kokilas-MacBook-Air ~ % arp -a
? (192.168.43.1) at 20:34:fb:55:1c:fd on en0 ifscope [ethernet]
? (192.168.43.27) at 14:c2:13:4:44:a4 on en0 ifscope permanent [ethernet]
? (224.0.0.251) at 1:0:5e:0:0:fb on en0 ifscope permanent [ethernet]
kokilareddy@kokilas-MacBook-Air ~ %
```

Interface Name	IP address IP4	MAC address	Scope
eno	192.168.43.1	20:34:fb:55:1c:fd	[ethernet]
eno	192.168.43.27	14:c2:13:4:44:a4	permanent[ethernet]
eno	224.0.0.251	1:0:5e:0:0:fb	permanent[ethernet]

Step 2: To assign an IP address to an interface.

Step 3: To activate / deactivate a network interface.

```
kokilareddy@kokilas-MacBook-Air ~ % sudo ifconfig en1 10.0.4.50
kokilareddy@kokilas-MacBook-Air ~ % sudo ifconfig en1 up
```

Step 4: To show the current neighbour table in kernel

```
kokilareddy@kokilas-MacBook-Air ~ % ip neigh
2409:4071:238c:380d::2 dev en0 lladdr 20:34:fb:55:1c:fd REACHABLE
2409:4071:238c:380d:1428:ec11:fe3c:c857 dev en0 lladdr 14:c2:13:4:44:a4 REACHABLE
2409:4071:238c:380d:185e:ff98:8722:305d dev en0 lladdr 14:c2:13:4:44:a4 REACHABLE
fe80::1 dev lo0 lladdr (incomplete) REACHABLE
fe80::e3:6ab1:a36d:8558 dev en0 lladdr 14:c2:13:4:44:a4 REACHABLE
fe80::2234:fbff:fe55:1cfd dev en0 lladdr 20:34:fb:55:1c:fd REACHABLE
fe80::849e:b7ff:feb1:340c dev awdl0 lladdr 86:9e:b7:b1:34:c REACHABLE
fe80::849e:b7ff:feb1:340c dev llw0 lladdr 86:9e:b7:b1:34:c REACHABLE
fe80::67e1:9c68:405b:c504 dev utun0 lladdr (incomplete) REACHABLE
fe80::6455:9b0c:8679:a2e5 dev utun1 lladdr (incomplete) REACHABLE
192.168.43.1 dev en0 lladdr 20:34:fb:55:1c:fd REACHABLE
192.168.43.27 dev en0 lladdr 14:c2:13:4:44:a4 REACHABLE
224.0.0.251 dev en0 lladdr 1:0:5e:0:0:fb REACHABLE
```

TASK 2: PING PDU (PACKET DATA UNITS OR PACKETS) CAPTURE

Step 1: Assign an IP address to the system (Host).

Step 2: Launch Wireshark and select 'any' interface

Step 3: In terminal, type ping 10.0.4.50

Step 4: Analyze the following in Terminal

```
64 bytes from 10.0.4.50: icmp_seq=95 ttl=64 time=0.125 ms
64 bytes from 10.0.4.50: icmp_seq=96 ttl=64 time=0.128 ms
64 bytes from 10.0.4.50: icmp_seq=97 ttl=64 time=0.078 ms
64 bytes from 10.0.4.50: icmp_seq=98 ttl=64 time=0.173 ms
```

- TTL
- Protocol used by ping **ICMP**
- Time

Step 5: Analyze the following in Wireshark

Detals	First echo request	First echo response
Frame Number	1	1
Source IP address	10.0.4.50	10.0.4.50
Destination IP address	10.0.4.50	10.0.4.50
ICMP Type Value	8	8
ICMP Code Value	0	0
Source Ethernet Address	14:c2:13:4:44:a4	14:c2:13:4:44:a4
Destination Ethernet Address	14:c2:13:4:44:a4	14:c2:13:4:44:a4
Internet Protocol Version	4	4
Time To Live (TTL) Value	64	64

TASK 3: HTTP PDU CAPTURE USING WIRESHARK'S FILTER FEATURE

Step 1: Launch Wireshark and select 'any' interface. On the Filter toolbar, type-in 'http' and press enter

Step 2: Open Firefox browser, and browse www.flipkart.com

Step 3: Analyze the first (interaction of host to the web server) and second frame (response of server to the client). By analyzing the filtered frames, complete the table below:

Detals	First echo request	First echo response
Frame Number	712	713
Source port	54204	80
Destination port	80	54204
Source Ip adress	192.168.43.27	17.253.83.201
Destination IP address	17.253.83.201	192.168.43.27
Source Ethernet Address	14:c2:13:04:44:a4	20:34:fb:55:1c:fd
Destination Ethernet Address	20:34:fb:55:1c:fd	14:c2:13:04:44:a4

Step 4: Analyze the HTTP request and response and complete the table below.

HTTP REQUEST		HTTP RESPONSE	
Get	[GET /img/background1.jpg HTTP/1.1\]	Server	Microsoft-IIS/7.5\
Host	www.vcfsc.in	Content-type	image/jpeg
User-agent	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_5) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/13.1.1 Safari/605.1.15	Date	Sat, 05 Sep 2020 12:16:49 GMT
Accept-language	en-us	Location	
Accept-encoding	gzip, deflate	Content-length	20686
Connection	keep-alive	Connection	keep-alive

TASK 4: CAPTURING PACKETS WITH TCPDUMP

Step 1: Use the command tcpdump -D to see which interfaces are available for capture.

```
sudo tcpdump -D
```

```
kokilareddy@kokilas-MacBook-Air ~ % sudo tcpdump -D

1.en0 [Up, Running]
2.p2p0 [Up, Running]
3.awdl0 [Up, Running]
4.llw0 [Up, Running]
5.utun0 [Up, Running]
6.utun1 [Up, Running]
7.lo0 [Up, Running, Loopback]
8.bridge0 [Up, Running]
9.en1 [Up, Running]
10.gif0 [none]
11.stf0 [none]
```

Step 2: Capture all packets in any interface by running this command:

```
sudo tcpdump -i any
```

Step 3: Understand the output format.

Step 4: To filter packets based on protocol, specifying the protocol in the command line. For example, capture ICMP packets only by using this command:

```
sudo tcpdump -i any -c5 icmp
```

Step 5: Check the packet content. For example, inspect the HTTP content of a web request like this:

```
sudo tcpdump -i any -c10 -nn -A port 80
```

Step 6: To save packets to a file instead of displaying them on screen, use the option -w:

```
sudo tcpdump -i any -c10 -nn -w webserver.pcap port 80
```

TASK 5: PERFORM TRACEROUTE CHECKS

Step 1: Run the traceroute using the following command.

sudo traceroute www.google.com

```
kokilareddy@kokilas-MacBook-Air ~ %  sudo traceroute www.google.com

traceroute to www.google.com (172.217.160.132), 64 hops max, 52 byte packets
 1  192.168.43.1 (192.168.43.1)  2.904 ms  3.306 ms  2.388 ms
 2  * * *
 3  10.72.194.35 (10.72.194.35)  115.110 ms  34.685 ms  39.553 ms
 4  192.168.61.44 (192.168.61.44)  39.552 ms
    192.168.61.46 (192.168.61.46)  37.604 ms
    192.168.61.44 (192.168.61.44)  38.845 ms
 5  192.168.61.43 (192.168.61.43)  38.787 ms  37.277 ms
    192.168.61.47 (192.168.61.47)  40.370 ms
 6  172.26.74.86 (172.26.74.86)  43.720 ms  37.379 ms  40.536 ms
 7  172.26.74.99 (172.26.74.99)  38.901 ms
    172.26.74.98 (172.26.74.98)  35.239 ms  37.956 ms
 8  192.168.61.37 (192.168.61.37)  29.972 ms
    192.168.61.33 (192.168.61.33)  34.024 ms
    192.168.61.31 (192.168.61.31)  37.604 ms
 9  192.168.61.36 (192.168.61.36)  29.597 ms
    192.168.61.34 (192.168.61.34)  30.763 ms
    192.168.61.32 (192.168.61.32)  36.419 ms
10  172.25.118.3 (172.25.118.3)  36.421 ms
    172.25.118.5 (172.25.118.5)  38.221 ms
    172.25.118.7 (172.25.118.7)  42.598 ms
11  172.25.106.40 (172.25.106.40)  33.286 ms  45.318 ms  63.281 ms
12  172.16.21.8 (172.16.21.8)  40.160 ms
    172.16.21.10 (172.16.21.10)  42.807 ms
    172.16.21.104 (172.16.21.104)  42.356 ms
13  172.16.2.9 (172.16.2.9)  79.141 ms
    172.16.2.65 (172.16.2.65)  39.672 ms  44.159 ms
14  172.16.20.28 (172.16.20.28)  36.547 ms
    172.25.106.39 (172.25.106.39)  37.845 ms
    172.25.41.166 (172.25.41.166)  43.020 ms
15  10.70.80.197 (10.70.80.197)  47.823 ms  34.288 ms  39.499 ms
16  10.70.80.225 (10.70.80.225)  43.414 ms  41.121 ms  40.007 ms
17  74.125.48.26 (74.125.48.26)  43.509 ms  56.684 ms  43.909 ms
18  108.170.253.97 (108.170.253.97)  52.878 ms  39.207 ms
    108.170.253.113 (108.170.253.113)  28.935 ms
19  216.239.59.171 (216.239.59.171)  35.086 ms  44.630 ms  51.697 ms
20  maa03s29-in-f4.1e100.net (172.217.160.132)  39.124 ms  41.508 ms  37.947 ms
```

Step 2: Analyze destination address of google.com and no. of hops

Destination address : 172.217.160.132

Number of hops : 20

Step 3: To speed up the process, you can disable the mapping of IP addresses with hostnames

by using the -n option

```
sudo traceroute -n www.google.com
```

Step 4: The -I option is necessary so that the traceroute uses ICMP.

```
sudo traceroute -I www.google.com
```

Step 5: By default, traceroute uses icmp (ping) packets. If you'd rather test a TCP connection to gather data more relevant to web server, you can use the -T flag.

```
sudo traceroute -T www.google.com
```

TASK 6: EXPLORE AN ENTIRE NETWORK FOR INFORMATION (NMAP)

Step 1: You can scan a host using its host name or IP address, for instance.

nmap www.pes.edu

```
kokilareddy@kokilas-MacBook-Air ~ % nmap www.pes.edu

Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-05 15:15 IST
Nmap scan report for www.pes.edu (13.71.123.138)
Host is up (0.046s latency).
Not shown: 998 filtered ports
PORT    STATE SERVICE
80/tcp  open  http
443/tcp open  https

Nmap done: 1 IP address (1 host up) scanned in 5.53 seconds
```

Step 2: Alternatively, use an IP address to scan.

nmap 163.53.78.128

```
kokilareddy@kokilas-MacBook-Air ~ % nmap 172.217.160.132
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-05 15:21 IST
Nmap scan report for maa03s29-in-f4.1e100.net (172.217.160.132)
Host is up (0.056s latency).
Not shown: 998 filtered ports
PORT    STATE SERVICE
80/tcp  open  http
443/tcp open  https

Nmap done: 1 IP address (1 host up) scanned in 6.83 seconds
```

Step 3: Scan multiple IP address or subnet (IPv4)

nmap 192.168.1.1 192.168.1.2 192.168.1.3

```
kokilareddy@kokilas-MacBook-Air ~ % nmap 192.168.1.1 192.168.1.2 192.168.1.3
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-05 15:20 IST
Nmap done: 3 IP addresses (0 hosts up) scanned in 3.09 seconds
```

TASK 7 A): NETCAT AS CHAT TOOL

a) Intra system communication (Using 2 terminals in the same system)

Step 1: Open a terminal (Ctrl+Alt+T). This will act as a Server.

Step 2: Type nc -l any_portnum (For eg., nc -l 1234)

Note: It will goto listening mode

Step 3: Open another terminal and this will act as a client.

Step 4: Type nc <your-system-ip-address> portnum

Note: portnum should be common in both the terminals (for eg., nc 10.0.2.8 1234)

Step 5: Type anything in client will appear in server

b) Inter system communication

Setup a simple switched network of 2 PCs with one acting as Web server. Assign IP addresses for both PCs. Set the capture option as described above.

Step 1: Open terminal on Server machine (Machine 1).

Step 2: Type nc -l any_portnum

Step 3: Open terminal on the Client machine (Machine 2)

Step 4: Type nc <server-ip-address> portnum

Step 5: Type anything in client will appear in the server terminal

```
kokilareddy@kokilas-MacBook-Air ~ % nc -l 9876
This is inter system communication
```

```
navyareddy@kokilas-MacBook-Air ~ % nc 192.168.43.27 9876
This is inter system communication
```

TASK 7 B): USE NETCAT TO TRANSFER FILES

The netcat utility can also be used to transfer files.

Step 1: At the server side, create an empty file named 'test.txt'

sudo nc -l 555 > test.txt

Note: 2 students can combine for the following tasks (switch and cables can be taken from Lab technicians)

Step 2: At the client side, we have a file 'test.txt'. Add some contents to it.

Step 3: Run the client as:

sudo nc 10.0.2.8 555 < test.txt

here, 10.0.2.8 is the IP address of server and 555 is the port number.

Step 4: At server side, verify the file transfer using the command

cat test.txt

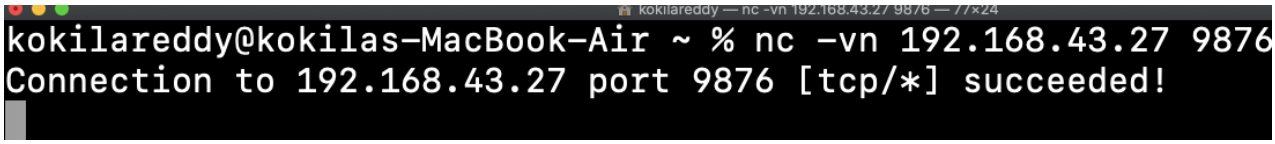
```
kokilareddy@kokilas-MacBook-Air ~ % sudo nc -l 555 > test.txt
This is a file
kokilareddy@kokilas-MacBook-Air ~ %
```

```
kokilareddy@kokilas-MacBook-Air ~ % nc 192.168.43.27 555 <test.txt
This is a file
kokilareddy@kokilas-MacBook-Air ~ %
```

TASK 7 C): OTHER COMMANDS

1) To test if a particular TCP port of a remote host is open.

nc -vn 10.0.2.8 555



2) Run a web server with a static web page.

Step 1: Run the command below on local host (e.g. 10.0.2.8) to start a web server that

serves test.html on port 80.

while true; do sudo nc -lp 80 < test.html;

done

Step 2: Now open http://10.0.2.8/test.html from another host to access it.

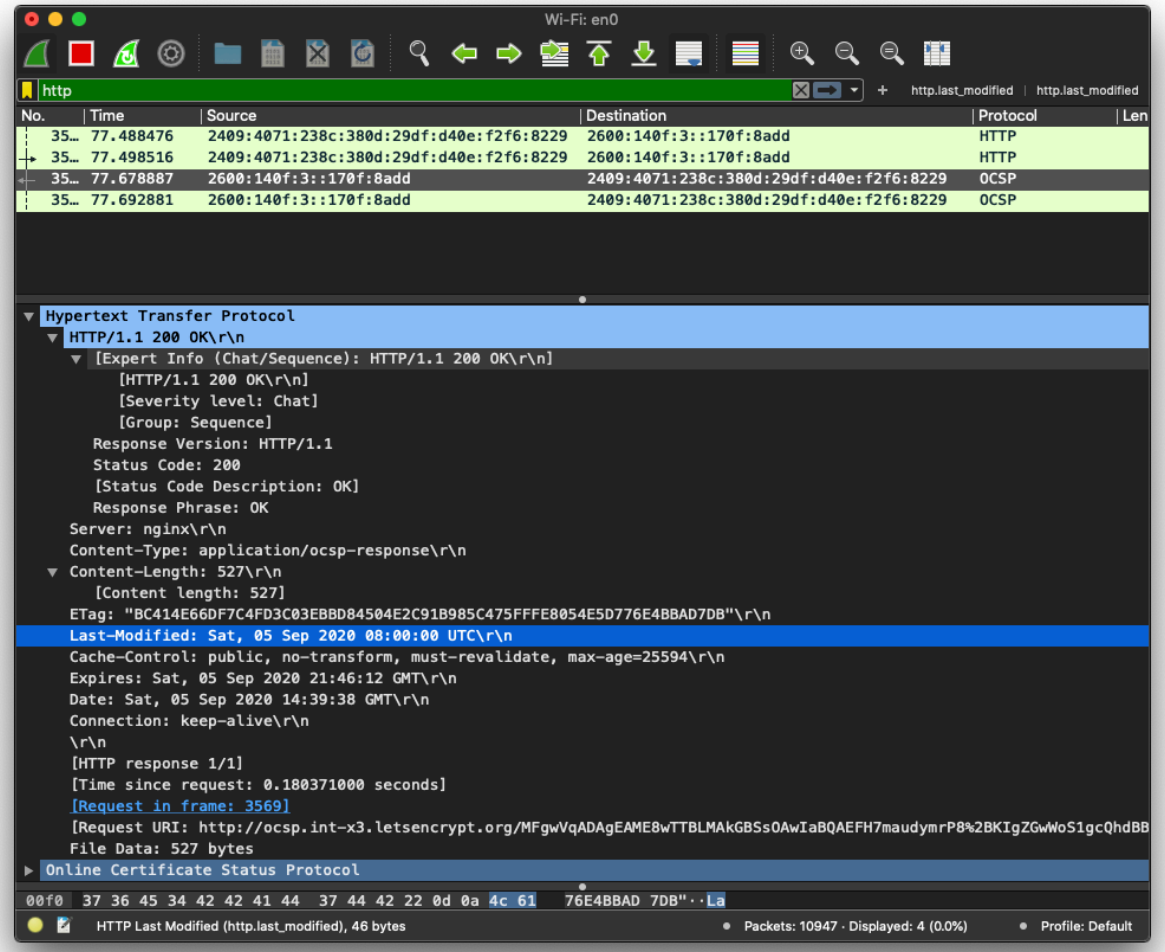
Step 3: Observe the details on the terminal

QUESTIONS ON ABOVE OBSERVATIONS:

1) Is your browser running HTTP version 1.0 or 1.1? What version of HTTP is the server?

HTTP 1.1

2) When was the HTML file that you are retrieving last modified at the server?



Last-Modified: Sat, 05 Sep 2020 08:00:00 UTC\r\n

3) How to tell ping to exit after a specified number of ECHO_REQUEST packets?

ping google.com -c 1

Ping website -c number_of_ECHO_REQUESTS required.

4) How will you identify remote host apps and OS?

We can use the nmap command under UNIX, OS X, BSD or Linux operating systems to detect remote operating systems and running apps.

Commands

$ sudo nmap -O -v localhost
$ sudo nmap -O -v server.ip.address

```
kokilareddy@kokilas-MacBook-Air ~ % sudo nmap -O -v localhost
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-05 20:24 IST
Initiating SYN Stealth Scan at 20:24
Scanning localhost (127.0.0.1) [1000 ports]
Discovered open port 22/tcp on 127.0.0.1
Discovered open port 3306/tcp on 127.0.0.1
Discovered open port 5900/tcp on 127.0.0.1
Discovered open port 5432/tcp on 127.0.0.1
Discovered open port 3283/tcp on 127.0.0.1
Discovered open port 88/tcp on 127.0.0.1
Completed SYN Stealth Scan at 20:24, 3.26s elapsed (1000 total ports)
Initiating OS detection (try #1) against localhost (127.0.0.1)
Retrying OS detection (try #2) against localhost (127.0.0.1)
Retrying OS detection (try #3) against localhost (127.0.0.1)
Retrying OS detection (try #4) against localhost (127.0.0.1)
Retrying OS detection (try #5) against localhost (127.0.0.1)
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00020s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 498 filtered ports, 496 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
88/tcp   open  kerberos-sec
3283/tcp open  netassistant
3306/tcp open  mysql
5432/tcp open  postgresql
5900/tcp open  vnc
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.80%E=4%D=9/5%OT=22%CT=1%CU=30089%PV=N%DS=0%DC=L%G=Y%TM=5F53A6C4
OS:%P=x86_64-apple-darwin17.7.0)SEQ(SP=106%GCD=1%ISR=10B%TI=Z%CI=RD%II=RI%T
OS:S=A)OPS(O1=M3FD8NW6NNT11SLL%O2=M3FD8NW6NNT11SLL%O3=M3FD8NW6NNT11%O4=M3FD
OS:8NW6NNT11SLL%O5=M3FD8NW6NNT11SLL%O6=M3FD8NNT11SLL)WIN(W1=FFFF%W2=FFFF%W3
OS:=FFFF%W4=FFFF%W5=FFFF%W6=FFFF)ECN(R=Y%DF=Y%T=40%W=FFFF%O=M3FD8NW6SLL%CC=
OS:N%Q=)T1(R=Y%DF=Y%T=40%S=O%A=S+%F=AS%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=
OS:40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T5(R=Y%DF=N%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0
OS:%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T7(R=Y%DF=N%T=40%W=0%S=Z
OS:%A=S%F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%T=40%IPL=38%UN=0%RIPL=G%RID=G%RIPCK=Z%R
OS:UCK=0%RUD=G)IE(R=Y%DFI=S%T=40%CD=S)

Uptime guess: 4.030 days (since Tue Sep  1 19:41:13 2020)
Network Distance: 0 hops
TCP Sequence Prediction: Difficulty=258 (Good luck!)
IP ID Sequence Generation: All zeros

Read data files from: /usr/local/bin/../share/nmap
OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 17.27 seconds
           Raw packets sent: 1609 (74.846KB) | Rcvd: 2228 (100.396KB)
```

```
kokilareddy@kokilas-MacBook-Air ~ % sudo nmap -O -v server.192.168.43.27
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-05 20:25 IST
Failed to resolve "server.192.168.43.27".
Read data files from: /usr/local/bin/../share/nmap
WARNING: No targets were specified, so 0 hosts scanned.
Nmap done: 0 IP addresses (0 hosts up) scanned in 1.55 seconds
           Raw packets sent: 0 (0B) | Rcvd: 0 (0B)
kokilareddy@kokilas-MacBook-Air ~ %
```

EXERCISES:

1) Capture and Analyze IPv4 / IPv6 packets

IPv4 / IPv6 packet header

Get	MFgwVqADAgEAME8wTTBLMAkGBSsOAwIaBQAEFH7maudymrP8%2BKIgZGwWoS1gcQhdBBSoSmpjBH3duubRObemRWXv86jsoQISBEuWmr%2FxMD1%2B1qak8H2ULYIr HTTP/1.1
Host	ocsp.int-x3.letsencrypt.org
User-Agent	com.apple.trustd/2.0
Accept-Language	En-us
Cache-control	public, no-transform, must-revalidate, max-age=25594\r\n
Pragma	
Connection	keep-alive

2) Explore various other network configuration, troubleshooting and debugging tools such as Route, Netstat, etc.

Netstat : The netstat command generates displays that show network status and protocol statistics. we can display the status of TCP and UDP endpoints in table format, routing table information, and interface information. Netstat displays various types of network data depending on the command line option selected. These displays are the most useful for system administration.

Netstat -i for getting the details of number transmission units, packets received ,input errors,ackets transmitted, output errors and collisions

Netstat -s command displays statistics for TCP ,UDP , SCTP , IP ,IPv6 ,IGMP, ICMP ,ICMPv6.

Netstat -r command used to display the routing tables

```
kokilareddy@kokilas-MacBook-Air ~ % netstat -i
Name       Mtu   Network       Address            Ipkts Ierrs    Opkts Oerrs  Coll
lo0   16384 <Link#1>                         15696     0    15696     0     0
lo0   16384 127           localhost          15696     -    15696     -     -
lo0   16384 localhost   ::1                  15696     -    15696     -     -
lo0   16384 kokilas-mac fe80:1::1            15696     -    15696     -     -
gif0* 1280  <Link#2>                             0     0        0     0     0
stf0* 1280  <Link#3>                             0     0        0     0     0
en0   1500  <Link#4>    14:c2:13:04:44:a4   238411     0   254625     0     0
en0   1500  kokilas-mac fe80:4::48a:202d:   238411     -   254625     -     -
en0   1500  192.168.43    192.168.43.27     238411     -   254625     -     -
en0   1500  2409:4071:2 2409:4071:238c:38   238411     -   254625     -     -
en0   1500  2409:4071:2 2409:4071:238c:38   238411     -   254625     -     -
en1   1500  <Link#5>    82:18:5b:c4:78:40        0     0        0     0     0
bridg 1500  <Link#6>    82:18:5b:c4:78:40        0     0        0     0     0
p2p0  2304  <Link#7>    06:c2:13:04:44:a4        0     0        0     0     0
awdl0 1484  <Link#8>    3e:f4:d0:31:0b:f0        0     0      220     0     0
awdl0 1484  fe80::3cf4: fe80:8::3cf4:d0ff        0     -      220     -     -
llw0  1500  <Link#9>    3e:f4:d0:31:0b:f0        0     0        0     0     0
llw0  1500  fe80::3cf4: fe80:9::3cf4:d0ff        0     -        0     -     -
utun0 1380  <Link#10>                            0     0      194     0     0
utun0 1380  kokilas-mac fe80:a::1a7e:b09:        0     -      194     -     -
utun1 2000  <Link#11>                            0     0      194     0     0
utun1 2000  kokilas-mac fe80:b::1ca4:cc12        0     -      194     -     -
utun2 1380  <Link#13>                            0     0        2     0     0
utun2 1380  kokilas-mac fe80:d::df7c:9a75        0     -        2     -     -
utun3 2000  <Link#14>                            0     0        2     0     0
utun3 2000  kokilas-mac fe80:e::b4e8:26d1        0     -        2     -     -
kokilareddy@kokilas-MacBook-Air ~ %
```

```
kokilareddy@kokilas-MacBook-Air ~ % netstat -r
Routing tables

Internet:
Destination        Gateway            Flags        Netif Expire
default            192.168.43.1       UGSc           en0
127                localhost          UCS            lo0
localhost          localhost          UH             lo0
169.254            link#4             UCS            en0      !
192.168.43         link#4             UCS            en0      !
192.168.43.1/32    link#4             UCS            en0      !
192.168.43.1       20:34:fb:55:1c:fd  UHLWIir        en0   1191
192.168.43.27/32   link#4             UCS            en0      !
192.168.43.27      14:c2:13:4:44:a4   UHLWI          lo0
224.0.0/4          link#4             UmCS           en0      !
224.0.0.251        1:0:5e:0:0:fb      UHmLWI         en0
239.255.255.250    1:0:5e:7f:ff:fa    UHmLWI         en0
255.255.255.255/32 link#4             UCS            en0      !
```