Week #1

Study and understand the basic networking tools - Wireshark, Tcpdump, Ping, Traceroute.

Learn and Understand Network Tools

1. Wireshark

- Perform and analyze Ping PDU capture
- Examine HTTP packet capture
- Analyze HTTP packet capture using filter

2. Tcpdump

- Capture packets

3. Ping

- Test the connectivity between 2 systems

4. Traceroute

- Perform traceroute checks

5. Nmap

- Explore an entire network

IMPORTANT INSTRUCTIONS:

- This manual is written for Ubuntu Linux OS only. You can also execute these experiments on VirtualBox or VMWare platform.
- For few tasks, you may need to create 2 VMs for experimental setup.
- Perform **sudo apt-get update** before installing any tool or utility.
- Install any tool or utility using the command **sudo apt-get install name_of_the_tool**
- Take screenshots wherever necessary and upload it to Edmodo as a single PDF file. (Refer general guidelines for submission requirements).
- To define an IP address for your machine (e.g., Section – 'a' & Serial number is 1, then your IP address should be 10.0.1.1. Section – 'h' & & Serial number is 23, then your IP address should be 10.0.8.23) – applicable only for relevant tasks (which doesn't requires internet connectivity to execute the tasks).

Task 1: Linux Interface Configuration (ifconfig / IP command)

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

ifconfig (or) **ip addr show**

Analyze and fill the following table:

ip address table:

Interface name	IP address (IPv4 / IPv6)	MAC address	
en0	10.1.0.108	26:8f:f9:47:e4:6f	
lo0	127.0.0.1	-	
anpi0	-	32:de:1a:59:fc:45	

```
Last login: Wed Jan 15 15:37:56 on console
[abhishekp@Abhisheks-MacBook-Air-3 ~ % ifconfig
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
        options=1203<RXCSUM,TXCSUM,TXSTATUS,SW_TIMESTAMP>
        inet 127.0.0.1 netmask 0xff000000
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
        nd6 options=201<PERFORMNUD,DAD>
gif0: flags=8010<POINTOPOINT,MULTICAST> mtu 1280
stf0: flags=0<> mtu 1280
anpi0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=400<CHANNEL_IO>
        ether 32:de:1a:59:fc:45
        media: none
        status: inactive
anpi1: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=400<CHANNEL_IO>
        ether 32:de:1a:59:fc:46
        media: none
        status: inactive
en3: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=400<CHANNEL_IO>
        ether 32:de:1a:59:fc:25
        nd6 options=201<PERFORMNUD,DAD>
        media: none
        status: inactive
en4: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=400<CHANNEL_IO>
        ether 32:de:1a:59:fc:26
        nd6 options=201<PERFORMNUD,DAD>
        media: none
        status: inactive
en1: flags=8963<UP,BROADCAST,SMART,RUNNING,PROMISC,SIMPLEX,MULTICAST> mtu 1500
        options=460<TSO4,TSO6,CHANNEL_IO>
        ether 36:80:dd:28:41:00
        media: autoselect <full-duplex>
        status: inactive
en2: flags=8963<UP,BROADCAST,SMART,RUNNING,PROMISC,SIMPLEX,MULTICAST> mtu 1500
        options=460<TSO4,TSO6,CHANNEL_IO>
        ether 36:80:dd:28:41:04
        media: autoselect <full-duplex>
        status: inactive
bridge0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=63<RXCSUM,TXCSUM,TSO4,TSO6>
        ether 36:80:dd:28:41:00
        Configuration:
                id 0:0:0:0:0:0 priority 0 hellotime 0 fwddelay 0
                maxage 0 holdcnt 0 proto stp maxaddr 100 timeout 1200
                root id 0:0:0:0:0:0 priority 0 ifcost 0 port 0
                ipfilter disabled flags 0x0
        member: en1 flags=3<LEARNING,DISCOVER>
                ifmaxaddr 0 port 8 priority 0 path cost 0
        member: en2 flags=3<LEARNING,DISCOVER>
                ifmaxaddr 0 port 9 priority 0 path cost 0
        nd6 options=201<PERFORMNUD,DAD>
        media: <unknown type>
        status: inactive
ap1: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=6460<TSO4,TSO6,CHANNEL_IO,PARTIAL_CSUM,ZEROINVERT_CSUM>
        ether 96:66:3f:52:e1:ee
        nd6 options=201<PERFORMNUD,DAD>
        media: autoselect (none)
        status: inactive
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=6460<TSO4,TSO6,CHANNEL_IO,PARTIAL_CSUM,ZEROINVERT_CSUM>
        ether 26:8f:f9:47:e4:6f
        inet6 fe80::1863:d811:994b:6597%en0 prefixlen 64 secured scopeid 0xb
        inet 10.1.0.108 netmask 0xfffff800 broadcast 10.1.7.255
        nd6 options=201<PERFORMNUD,DAD>
        media: autoselect
        status: active
utun0: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1380
        inet6 fe80::d490:b55b:97a1:2f2b%utun0 prefixlen 64 scopeid 0xd
        nd6 options=201<PERFORMNUD,DAD>
awdl0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=6460<TSO4,TSO6,CHANNEL_IO,PARTIAL_CSUM,ZEROINVERT_CSUM>
        ether ae:de:8c:33:7e:f4
        inet6 fe80::acde:8cff:fe33:7ef4%awdl0 prefixlen 64 scopeid 0xe
        nd6 options=201<PERFORMNUD,DAD>
        media: autoselect
        status: active
llw0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=400<CHANNEL_IO>
        ether ae:de:8c:33:7e:f4
        inet6 fe80::acde:8cff:fe33:7ef4%llw0 prefixlen 64 scopeid 0xf
        nd6 options=201<PERFORMNUD,DAD>
        media: autoselect (none)
utun1: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1500
        inet6 fe80::40e6:7a26:1a27:1964%utun1 prefixlen 64 scopeid 0x10
        nd6 options=201<PERFORMNUD,DAD>
utun2: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 2000
        inet6 fe80::3251:55b9:f1b2:21e1%utun2 prefixlen 64 scopeid 0x11
        nd6 options=201<PERFORMNUD,DAD>
utun3: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1000
        inet6 fe80::ce81:b1c:bd2c:69e8%utun3 prefixlen 64 scopeid 0x12
        nd6 options=201<PERFORMNUD,DAD>
abhishekp@Abhisheks-MacBook-Air-3 ~ %
```

Step 2: To assign an IP address to an interface, use the following command.

sudo ifconfig interface_name 10.0.your_section.your_sno netmask 255.255.255.0 (or)

sudo ip addr add 10.0.your_section.your_sno /24 dev interface_name

```
abhishekp@Abhisheks-MacBook-Air-3 ~ % sudo ifconfig lo0 10.0.4.002 netmask 255.255.255.0
abhishekp@Abhisheks-MacBook-Air-3 ~ % ifconfig
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
        options=1203<RXCSUM,TXCSUM,TXSTATUS,SW_TIMESTAMP>
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
        inet 10.0.4.2 netmask 0xffffff00
        nd6 options=201<PERFORMNUD,DAD>
```

Step 3: To activate / deactivate a network interface, type.

sudo ifconfig interface_name down

sudo ifconfig interface_name up

```
abhishekp@Abhisheks-MacBook-Air-3 ~ % sudo ifconfig lo0 down
abhishekp@Abhisheks-MacBook-Air-3 ~ % ifconfig
lo0: flags=8048<LOOPBACK,RUNNING,MULTICAST> mtu 16384
        options=1203<RXCSUM,TXCSUM,TXSTATUS,SW_TIMESTAMP>
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
        inet 10.0.4.2 netmask 0xffffff00
        nd6 options=201<PERFORMNUD,DAD>
```

Step 4: To show the current neighbor table in kernel, type

ip neigh

```
abhishekp@Abhisheks-MacBook-Air-3 ~ % if neigh
if>
```

Task 2: Ping PDU (Packet Data Units or Packets) Capture

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

Note: IP address of your system should be 10.0.your_section.your_sno.

Step 2: Launch Wireshark and select ‘any’ interface

Step 3: In terminal, type **ping 10.0.your_section.your_sno**

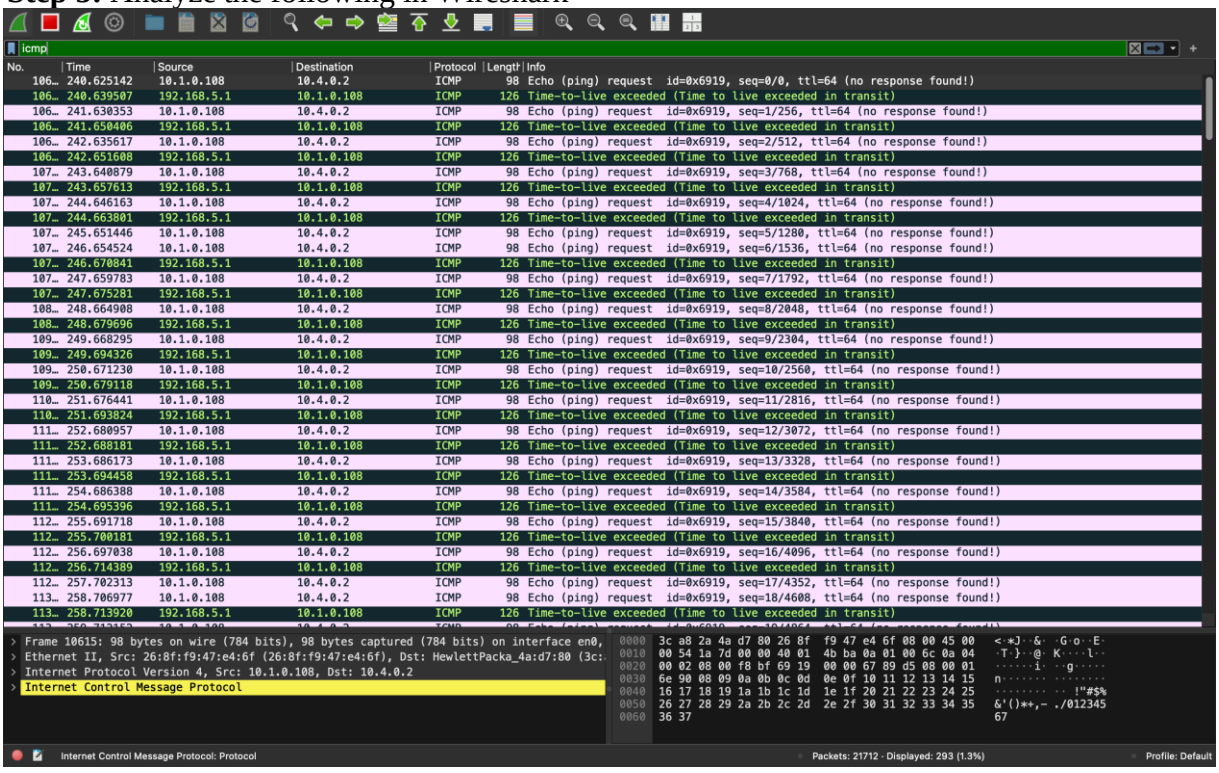
Observations to be made

Step 4: Analyze the following in Terminal

- TTL
- Protocol used by ping
- Time

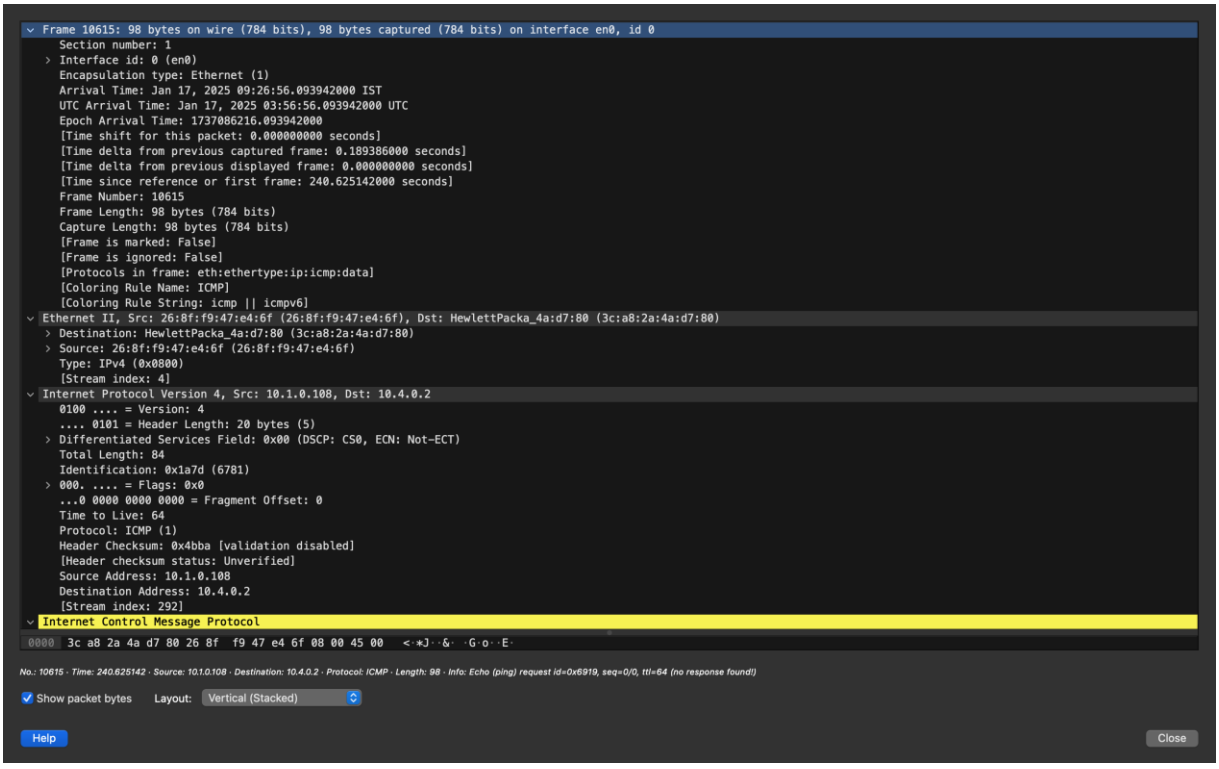
```
abhishekp@Abhisheks-MacBook-Air-3 ~ % ping 10.4.002
PING 10.4.002 (10.4.0.2): 56 data bytes
92 bytes from 192.168.5.1: Time to live exceeded
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src      Dst
 4  5  00 5400 1a7d   0 0000  01  01 8aba 10.1.0.108  10.4.0.2

Request timeout for icmp_seq 0
92 bytes from 192.168.5.1: Time to live exceeded
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src      Dst
 4  5  00 5400 7b48   0 0000  01  01 29ef 10.1.0.108  10.4.0.2

Request timeout for icmp_seq 1
92 bytes from 192.168.5.1: Time to live exceeded
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src      Dst
 4  5  00 5400 bf01   0 0000  01  01 e635 10.1.0.108  10.4.0.2

Request timeout for icmp_seq 2
92 bytes from 192.168.5.1: Time to live exceeded
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src      Dst
 4  5  00 5400 4fc6   0 0000  01  01 5571 10.1.0.108  10.4.0.2

Request timeout for icmp_seq 3
92 bytes from 192.168.5.1: Time to live exceeded
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src      Dst
 4  5  00 5400 12a1   0 0000  01  01 9296 10.1.0.108  10.4.0.2
```

Step 5: Analyze the following in Wireshark



On Packet List Pane, select the first echo packet on the list. On Packet Details Pane, click on each of the four "+" to expand the information. Analyze the frames with the first echo request and echo reply and complete the table below.

Details	First Echo Request	First Echo Reply
Frame Number	10615	10616
Source IP address	10.1.0.108	192.168.5.1
Destination IP address	10.4.0.2	10.1.0.108
ICMP Type Value	8	11
ICMP Code Value	0	0
Source Ethernet Address	28:8f:f9:47:e4:6f	3c:a8:2a:4a:d7:80
Destination Ethernet Address	3c:a8:2a:4a:d7:80	26:8f:f9:47:e4:6f
Internet Protocol Version	4	4
Time To Live (TTL) Value	64	63

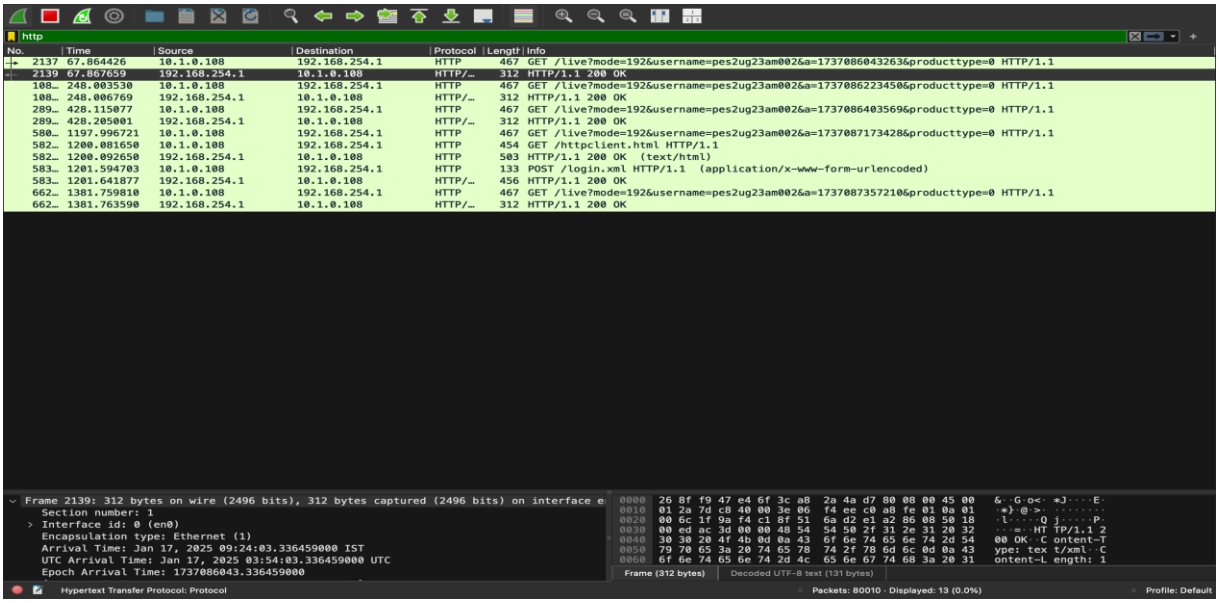


Task 3: HTTP PDU Capture

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



Observations to be made

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:

Details	First Echo Request	First Echo Reply
Frame Number	2137	2139
Source Port	62657	8090
Destination Port	8090	62657
Source IP address	10.1.0.108	192.168.254.1
Destination IP address	192.168.254.1	10.1.0.108
Source Ethernet Address	26:8f:f9:47:e4:6f	3c:a8:2a:4a:d7:80
Destination Ethernet Address	3c:a8:2a:4a:d7:80	26:8f:f9:47:e4:6f

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

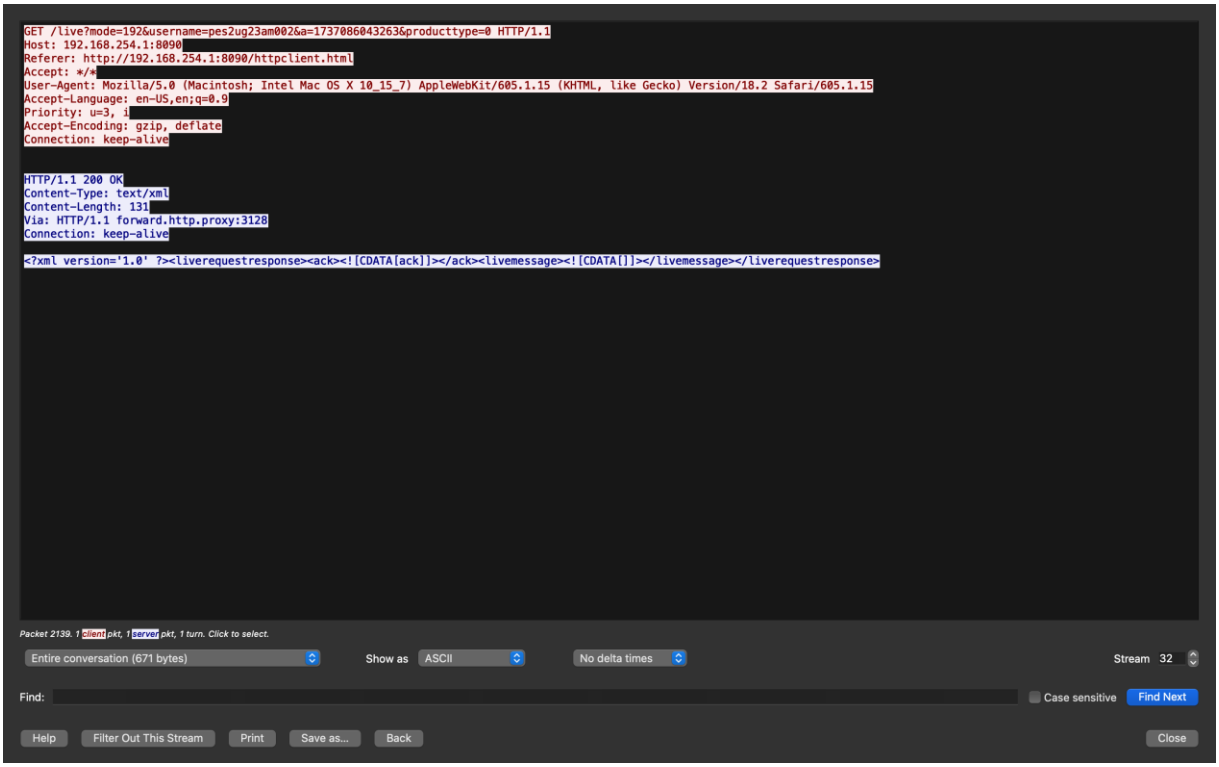
HTTP Request		HTTP Response	
Get	GET /live?mode=192&username=pes2ug23am002&a=1737086043263&producttype=0 HTTP/1.1	Server	-
Host	192.168.254.1:8090	Content-Type	text/xml
User-Agent	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/18.2	Date	-

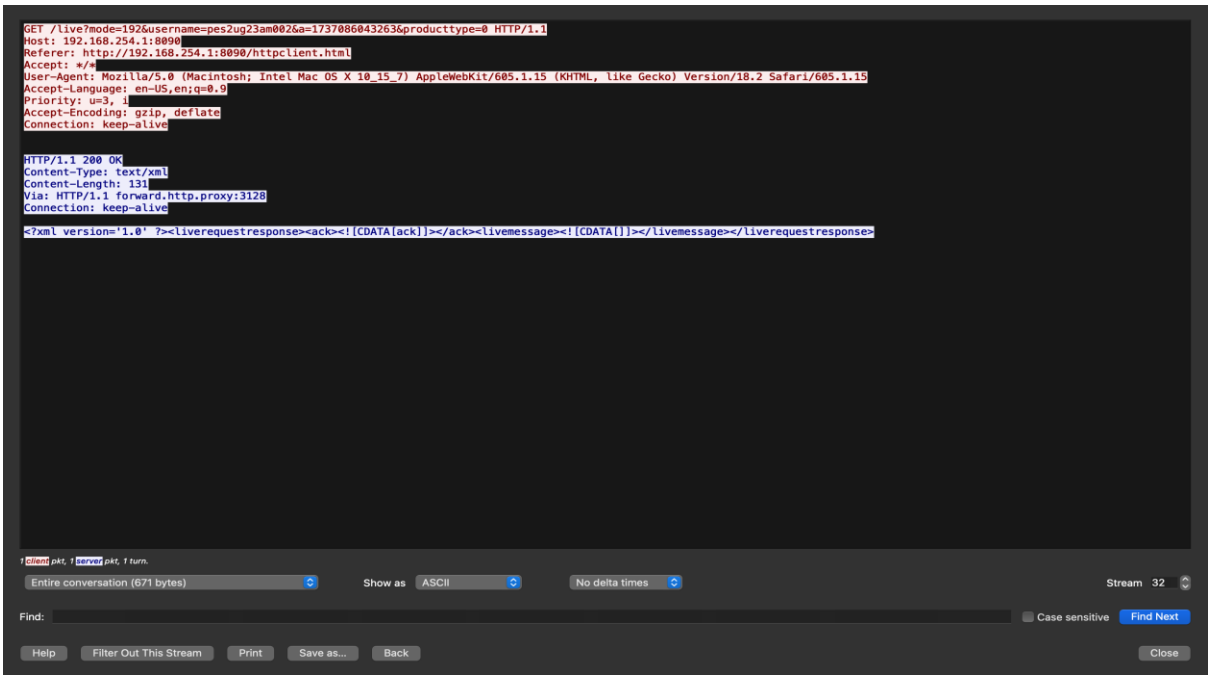
	Safari/605.1.15		
Accept-Language	en-US,en;q=0.9	Location	HTTP/1.1 forward.http.proxy:3128
Accept-Encoding	gzip, deflate	Content-Length	131
Connection	keep-alive	Connection	keep-alive

Using Wireshark's Follow TCP Stream

Step 1: Make sure the filter is blank. Right-click any packet inside the Packet List Pane, then select 'Follow TCP Stream'. For demo purpose, a packet containing the HTTP GET request "GET / HTTP / 1.1" can be selected.

Step 2: Upon following a TCP stream, screenshot the whole window.





Task 4: Capturing packets with tcpdump

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

sudo tcpdump –D

```
Last login: Fri Jan 17 09:26:38 on ttys001
abhishekp@Abhisheks-MacBook-Air-3 ~ % sudo tcpdump -D
Password:
1.ap1 [Up, Running, Wireless, Not associated]
2.en0 [Up, Running, Wireless, Associated]
3.utun0 [Up, Running]
4.awdl0 [Up, Running, Wireless, Associated]
5.llw0 [Up, Running, Connection status unknown]
6.utun1 [Up, Running]
7.utun2 [Up, Running]
8.utun3 [Up, Running]
9.anpi0 [Up, Running, Disconnected]
10.anpi1 [Up, Running, Disconnected]
11.en3 [Up, Running, Disconnected]
12.en4 [Up, Running, Disconnected]
13.en1 [Up, Running, Disconnected]
14.en2 [Up, Running, Disconnected]
15.bridge0 [Up, Running, Disconnected]
16.lo0 [Running, Loopback]
17.gif0 [none]
18.stf0 [none]
```

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

sudo tcpdump -i any

```
abhishekp@Abhisheks-MacBook-Air-3 ~ % sudo tcpdump -i any
tcpdump: data link type PKTAP
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on any, link-type PKTAP (Apple DLT_PKTAP), snapshot length 524288 bytes
11:28:51.090742 IP 10.1.0.108 > 10.4.0.2: ICMP echo request, id 26905, seq 3799, length 64
11:28:51.104531 IP 192.168.5.1 > 10.1.0.108: ICMP time exceeded in-transit, length 92
11:28:51.151876 IP 10.1.0.108.63303 > dns.google.domain-s: Flags [P.], seq 1584405307:158440545
9, ack 2758477489, win 2048, options [nop,nop,TS val 3880442704 ecr 2144451785], length 152
11:28:51.165816 IP dns.google.domain-s > 10.1.0.108.63303: Flags [P.], seq 1:493, ack 152, win
1040, options [nop,nop,TS val 2144457789 ecr 3880442704], length 492
11:28:51.166192 IP 10.1.0.108.63303 > dns.google.domain-s: Flags [.], ack 493, win 2040, option
s [nop,nop,TS val 3880442718 ecr 2144457789], length 0
11:28:51.168037 IP 10.1.0.108.63303 > dns.google.domain-s: Flags [P.], seq 152:304, ack 493, wi
n 2048, options [nop,nop,TS val 3880442720 ecr 2144457789], length 152
11:28:51.182273 IP dns.google.domain-s > 10.1.0.108.63303: Flags [P.], seq 493:985, ack 304, wi
n 1040, options [nop,nop,TS val 2144457805 ecr 3880442720], length 492
11:28:51.182501 IP 10.1.0.108.63303 > dns.google.domain-s: Flags [.], ack 985, win 2040, option
s [nop,nop,TS val 3880442734 ecr 2144457805], length 0
11:28:51.286854 IP 10.1.0.108.63303 > dns.google.domain-s: Flags [P.], seq 304:456, ack 985, wi
n 2048, options [nop,nop,TS val 3880442839 ecr 2144457805], length 152
11:28:51.302053 IP dns.google.domain-s > 10.1.0.108.63303: Flags [P.], seq 985:1477, ack 456, w
in 1040, options [nop,nop,TS val 2144457924 ecr 3880442839], length 492
11:28:51.302249 IP 10.1.0.108.63303 > dns.google.domain-s: Flags [.], ack 1477, win 2040, optio
ns [nop,nop,TS val 3880442854 ecr 2144457924], length 0
11:28:51.306957 IP 10.1.0.27.mdns > mdns.mcast.net.mdns: 3 [2q] PTR (QM)? _233637DE._sub._googl
ecast._tcp.local. PTR (QM)? _googlecast._tcp.local. (61)
11:28:51.306961 IP6 fe80::bc96:d0ff:fe6f:e0a4.mdns > ff02::fb.mdns: 1742 PTR (QM)? _quickshare.
_tcp.local. (40)
```

Note: Perform some pinging operation while giving above command. Also type

www.google.com in browser.

Observation

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

```
abhishekp@Abhisheks-MacBook-Air-3 ~ % sudo tcpdump -i any -c5 icmp
Password:
tcpdump: data link type PKTAP
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on any, link-type PKTAP (Apple DLT_PKTAP), snapshot length 524288 bytes
11:51:53.688161 IP 10.1.0.108 > pesuec.pdc.local: ICMP 10.1.0.108 udp port 45301 unreachable, l
ength 36
11:51:53.709318 IP 10.1.0.108 > pesuec.pdc.local: ICMP 10.1.0.108 udp port 46704 unreachable, l
ength 36
12:16:07.937714 IP 10.1.0.108 > pesuec.pdc.local: ICMP 10.1.0.108 udp port 58679 unreachable, l
ength 36
12:16:07.937767 IP 10.1.0.108 > pesuec.pdc.local: ICMP 10.1.0.108 udp port 58679 unreachable, l
ength 36
12:16:07.950546 IP 10.1.0.108 > pesuec.pdc.local: ICMP 10.1.0.108 udp port 58354 unreachable, l
ength 36
5 packets captured
19049 packets received by filter
0 packets dropped by kernel
```

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

```
abhishekp@Abhisheks-MacBook-Air-3 ~ % sudo tcpdump -i any -c10 -nn -A port 80
Password:
tcpdump: data link type PKTAP
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on any, link-type PKTAP (Apple DLT_PKTAP), snapshot length 524288 bytes
12:23:31.802790 IP 10.1.0.108.63573 > 23.38.59.250.80: Flags [S], seq 146499883, win 65535, opt
ions [mss 1460,nop,wscale 6,nop,nop,TS val 2556268078 ecr 0,sackOK,eol], length 0
E..@..@.@..+
..l.&;..U.P..i+........HW.............
.]..........
12:23:31.820928 IP 23.38.59.250.80 > 10.1.0.108.63573: Flags [S.], seq 1621249776, ack 14649988
4, win 65160, options [mss 1460,sackOK,TS val 1973583120 ecr 2556268078,nop,wscale 7], length 0
E..<..@.:../.&;.
..l.P.U`.N...i,.....{.........
u.}..]......
12:23:31.821216 IP 10.1.0.108.63573 > 23.38.59.250.80: Flags [.], ack 1, win 2058, options [nop
,nop,TS val 2556268097 ecr 1973583120], length 0
E..4..@.@..7
..l.&;..U.P..i,`.N....
.......
.].Au.}.
12:23:31.821487 IP 10.1.0.108.63573 > 23.38.59.250.80: Flags [P.], seq 1:354, ack 1, win 2058,
options [nop,nop,TS val 2556268097 ecr 1973583120], length 353: HTTP: GET /ME8wTTBLMEkwRzAHBgUr
DgMCGgQU36oS4yixCUGT4p9Cgs5HQEKVWKMEFLE%2Bw2kD%2BL9HAdSYJhoIAu9jZCvDAhAHF3kRAF0iZ%2FaIkvaPi1BY
HTTP/1.1
E.....@.@...
..l.&;..U.P..i,`.N....
.h.....
```

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

```
abhishekp@Abhisheks-MacBook-Air-3 ~ % sudo tcpdump -i any -c10 -nn -w webserver.pcap port 80
Password:
tcpdump: data link type PKTAP
tcpdump: listening on any, link-type PKTAP (Apple DLT_PKTAP), snapshot length 524288 bytes
10 packets captured
48885 packets received by filter
0 packets dropped by kernel
```

Task 5: Perform Traceroute checks

Step 1: Run the traceroute using the following command.

```
abhishekp@Abhisheks-MacBook-Air-3 ~ % sudo traceroute www.google.com
Password:
traceroute to www.google.com (142.250.192.132), 64 hops max, 40 byte packets
 1  10.1.0.1 (10.1.0.1)  15.474 ms  4.302 ms  4.133 ms
 2  192.168.5.1 (192.168.5.1)  5.780 ms  5.237 ms  4.132 ms
 3  192.168.254.1 (192.168.254.1)  3.552 ms  4.272 ms  4.360 ms
 4  1.6.222.153 (1.6.222.153)  5.217 ms
    static-161.83.12.61-tataidc.co.in (61.12.83.161)  9.023 ms
    1.6.222.153 (1.6.222.153)  4.399 ms
 5  100.70.137.132 (100.70.137.132)  13.519 ms  13.259 ms  13.658 ms
 6  100.70.136.115 (100.70.136.115)  13.475 ms  13.388 ms  13.959 ms
 7  100.70.136.109 (100.70.136.109)  13.662 ms  13.579 ms  23.378 ms
 8  100.70.138.77 (100.70.138.77)  13.242 ms  13.919 ms  13.572 ms
 9  100.70.136.28 (100.70.136.28)  13.564 ms  13.478 ms  13.652 ms
10  * * *
11  * * *
12  72.14.219.169 (72.14.219.169)  25.790 ms  14.732 ms  15.075 ms
13  * * *
14  142.251.55.66 (142.251.55.66)  17.735 ms
    216.239.43.172 (216.239.43.172)  17.228 ms
    142.251.55.242 (142.251.55.242)  15.209 ms
15  142.251.229.250 (142.251.229.250)  21.762 ms
    142.250.208.230 (142.250.208.230)  14.180 ms
    142.250.62.66 (142.250.62.66)  14.000 ms
16  bom12s18-in-f4.1e100.net (142.250.192.132)  25.727 ms
    142.251.49.232 (142.251.49.232)  23.814 ms
    142.250.212.0 (142.250.212.0)  25.397 ms
```

sudo traceroute www.google.com

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

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

```
abhishekp@Abhisheks-MacBook-Air-3 ~ % sudo traceroute -n www.google.com
traceroute to www.google.com (142.250.192.132), 64 hops max, 40 byte packets
 1  10.1.0.1  3.981 ms  4.140 ms  4.198 ms
 2  192.168.5.1  5.214 ms  3.687 ms  3.903 ms
 3  192.168.254.1  3.586 ms  3.506 ms  3.549 ms
 4  1.6.222.153  8.222 ms
    61.12.83.161  7.076 ms  8.842 ms
 5  100.70.137.132  14.087 ms  13.746 ms  13.664 ms
 6  100.70.136.115  13.920 ms  14.258 ms  19.792 ms
 7  115.111.221.61  7.291 ms
    100.70.136.109  13.855 ms  14.333 ms
 8  100.70.138.77  16.277 ms  18.212 ms  13.628 ms
 9  100.70.136.28  14.063 ms
    115.112.15.114  11.994 ms  11.801 ms
10  * * *
11  * * *
12  72.14.219.169  26.345 ms  15.478 ms  15.684 ms
13  * * 142.250.238.206  37.479 ms
14  216.239.56.64  15.874 ms
    142.251.55.238  16.053 ms
    142.251.55.120  14.824 ms
15  142.251.50.58  14.300 ms
    142.251.229.250  14.639 ms
    172.253.71.132  14.018 ms
16  142.251.49.232  22.586 ms
    172.253.72.137  15.769 ms
    72.14.232.34  31.619 ms
17  192.178.110.105  24.361 ms
    142.250.192.132  26.329 ms
    192.178.110.199  22.960 ms
```

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

sudo traceroute -I www.google.com

```
abhishekp@Abhisheks-MacBook-Air-3 ~ % sudo traceroute -I www.google.com
traceroute to www.google.com (142.250.192.132), 64 hops max, 48 byte packets
 1  10.1.0.1 (10.1.0.1)  4.281 ms  4.508 ms  3.959 ms
 2  192.168.5.1 (192.168.5.1)  3.883 ms  3.981 ms  5.475 ms
 3  192.168.254.1 (192.168.254.1)  3.344 ms  3.790 ms  3.479 ms
 4  1.6.222.153 (1.6.222.153)  4.390 ms  4.320 ms  4.584 ms
 5  100.70.137.132 (100.70.137.132)  13.532 ms  13.892 ms  13.210 ms
 6  100.70.136.115 (100.70.136.115)  13.190 ms  14.180 ms  13.257 ms
 7  100.70.136.109 (100.70.136.109)  13.047 ms  13.016 ms  13.067 ms
 8  100.70.138.77 (100.70.138.77)  16.048 ms  13.428 ms  13.340 ms
 9  100.70.136.28 (100.70.136.28)  14.149 ms  13.236 ms  13.378 ms
10  100.70.136.136 (100.70.136.136)  13.388 ms * *
11  100.70.136.25 (100.70.136.25)  13.140 ms * *
12  72.14.219.169 (72.14.219.169)  23.288 ms  15.540 ms  14.818 ms
13  216.239.43.133 (216.239.43.133)  14.287 ms  13.816 ms  13.127 ms
14  142.250.208.230 (142.250.208.230)  13.562 ms  13.015 ms  13.059 ms
15  142.251.49.232 (142.251.49.232)  22.434 ms  22.558 ms  22.499 ms
16  192.178.110.109 (192.178.110.109)  23.134 ms  23.141 ms  23.258 ms
17  172.253.50.147 (172.253.50.147)  23.466 ms  23.574 ms  22.636 ms
18  bom12s18-in-f4.1e100.net (142.250.192.132)  22.556 ms  22.822 ms  22.634 ms
```

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

```
abhishekp@Abhisheks-MacBook-Air-3 ~ % sudo traceroute -T www.google.com
Version 1.4a12+Darwin
Usage: traceroute [-adDeFInrSvx] [-A as_server] [-f first_ttl] [-g gateway] [-i iface]
        [-M first_ttl] [-m max_ttl] [-p port] [-P proto] [-q nqueries] [-s src_addr]
        [-t tos] [-w waittime] [-z pausemsecs] host [packetlen]
```

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

```
abhishekp@Abhisheks-MacBook-Air-3 ~ % nmap www.pes.edu
Starting Nmap 7.95 ( https://nmap.org ) at 2025-01-17 12:55 IST
Nmap scan report for www.pes.edu (98.70.112.52)
Host is up (0.029s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT    STATE SERVICE
22/tcp  open  ssh
80/tcp  open  http
443/tcp open  https

Nmap done: 1 IP address (1 host up) scanned in 42.72 seconds
```

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

nmap 163.53.78.128

```
abhishekp@Abhisheks-MacBook-Air-3 ~ % nmap 163.53.78.128
Starting Nmap 7.95 ( https://nmap.org ) at 2025-01-17 12:56 IST
Note: Host seems down. If it is really up, but blocking our ping probes, try -Pn
Nmap done: 1 IP address (0 hosts up) scanned in 3.05 seconds
```

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

nmap 192.168.1.1 192.168.1.2 192.168.1.3

```
abhishekp@Abhisheks-MacBook-Air-3 ~ % nmap 192.168.1.1 192.168.1.2 192.168.1.3
Starting Nmap 7.95 ( https://nmap.org ) at 2025-01-17 12:57 IST
Nmap done: 3 IP addresses (0 hosts up) scanned in 0.06 seconds
```

Questions on above observations:

1) Is your browser running HTTP version 1.0 or 1.1? What version of HTTP is the server? 1.1
2) When was the HTML file that you are retrieving last modified at the server? 20th dec 24
3) How to tell ping to exit after a specified number of ECHO_REQUEST packets? -c
4) How will you identify remote host apps and OS? nmap