Assignment 5
Wireshark assignment for Transport & Link layer

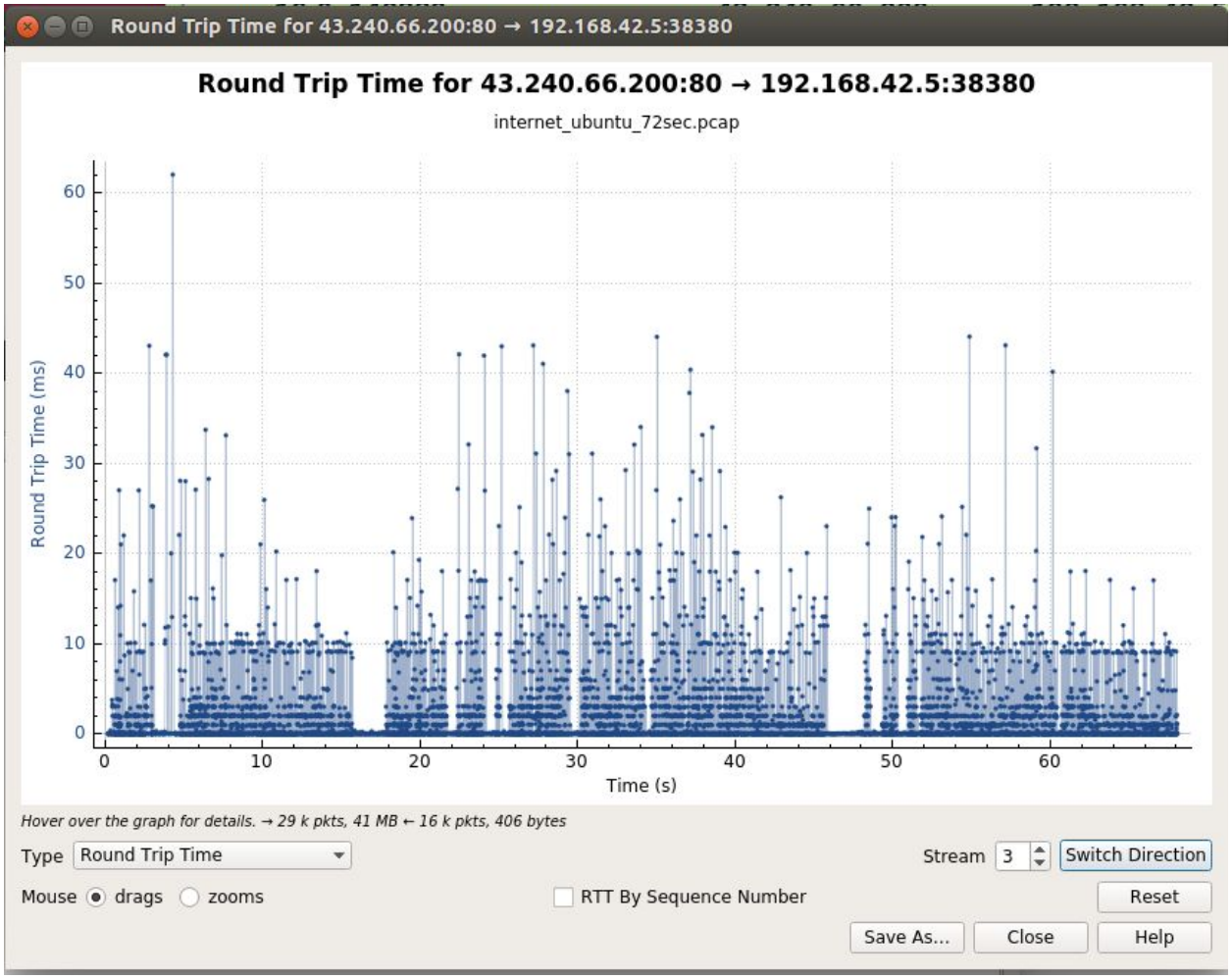
Harsh Agarwal
CS15BTECH11019

Transport Layer

Analysis of downloading ubuntu-16.04.3-desktop-amd64.iso from internet
http://mirrors.piconets.webwerks.in/ubuntu-mirror/ubuntu-releases/16.04.3/ubuntu-16.04.3-desktop-amd64.iso
Server IP - 43.240.66.200:80
My IP - 192.168.42.5:38380
Capture for 72 seconds
Command to filter this transaction :: **tcp.stream eq 3**

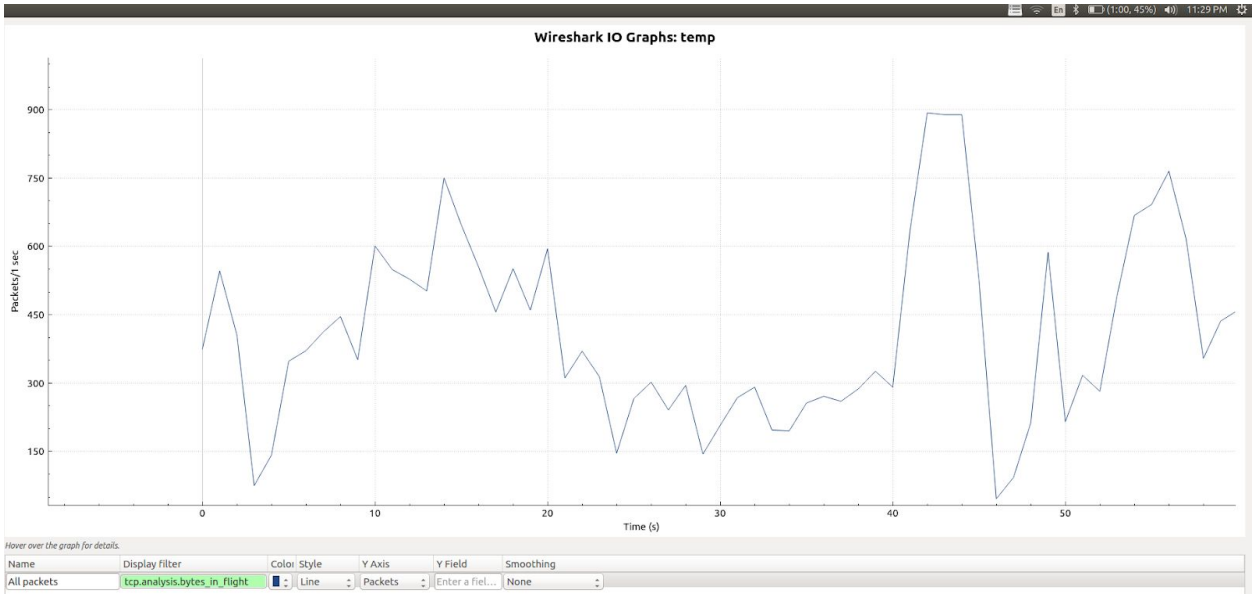
1)

a) RTT values on an average are same. There are fluctuations in the values but most value were close to 10ms

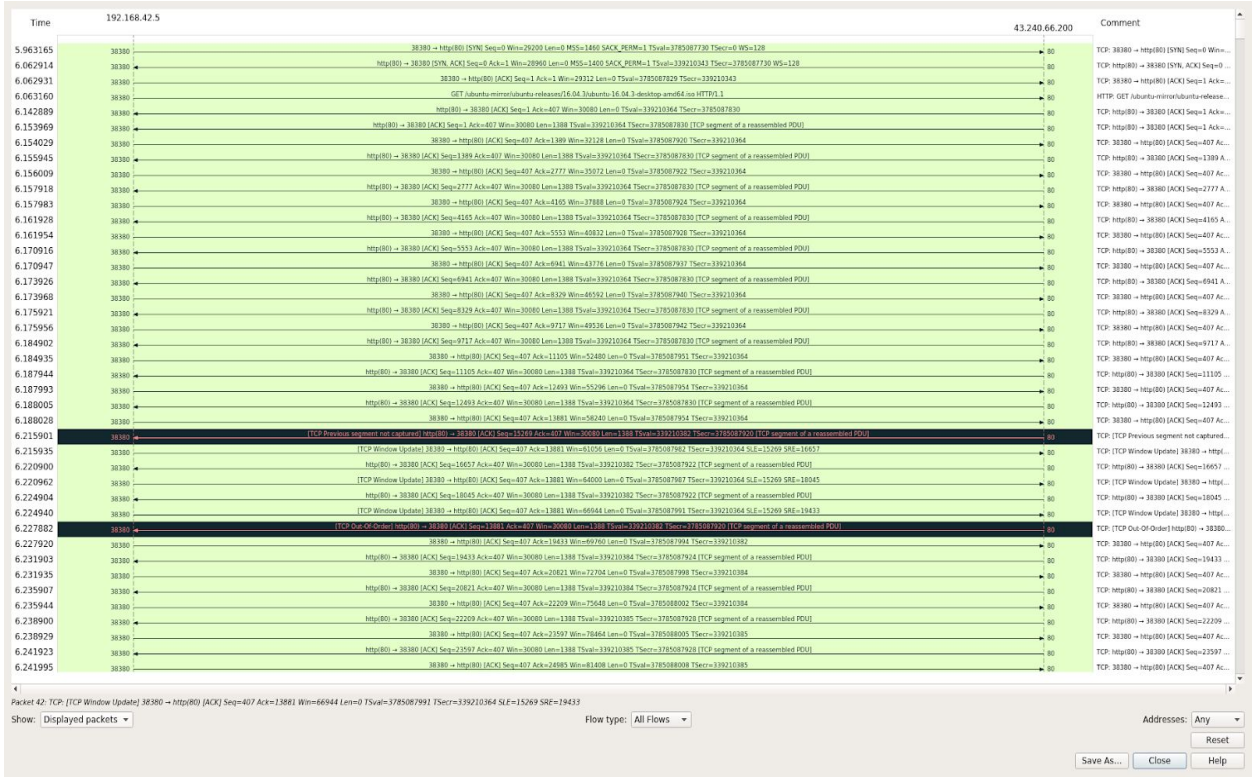


b) The IO Graph below is for tcp.analysis.bytes_in_flight vs time

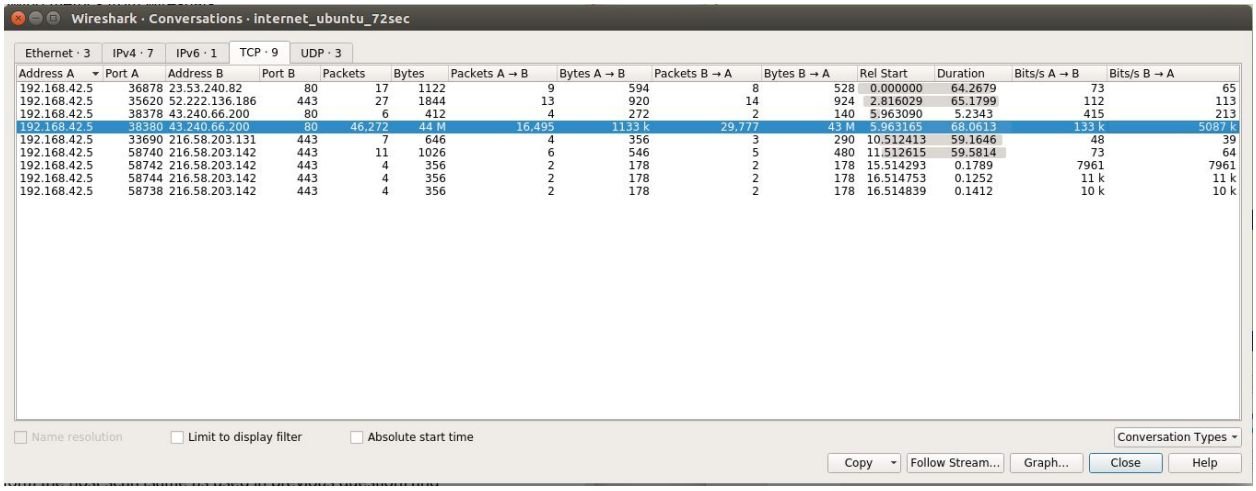
bytes_in_flight tells TCP Congestion window because at any instant this quantity tells exactly how many bytes were in the packet. Since this cannot be more than congestion , hence this is graph of congestion window. There is a lot of fluctuation in this meaning lot of variation in traffic(this is referred from RTT graph also).



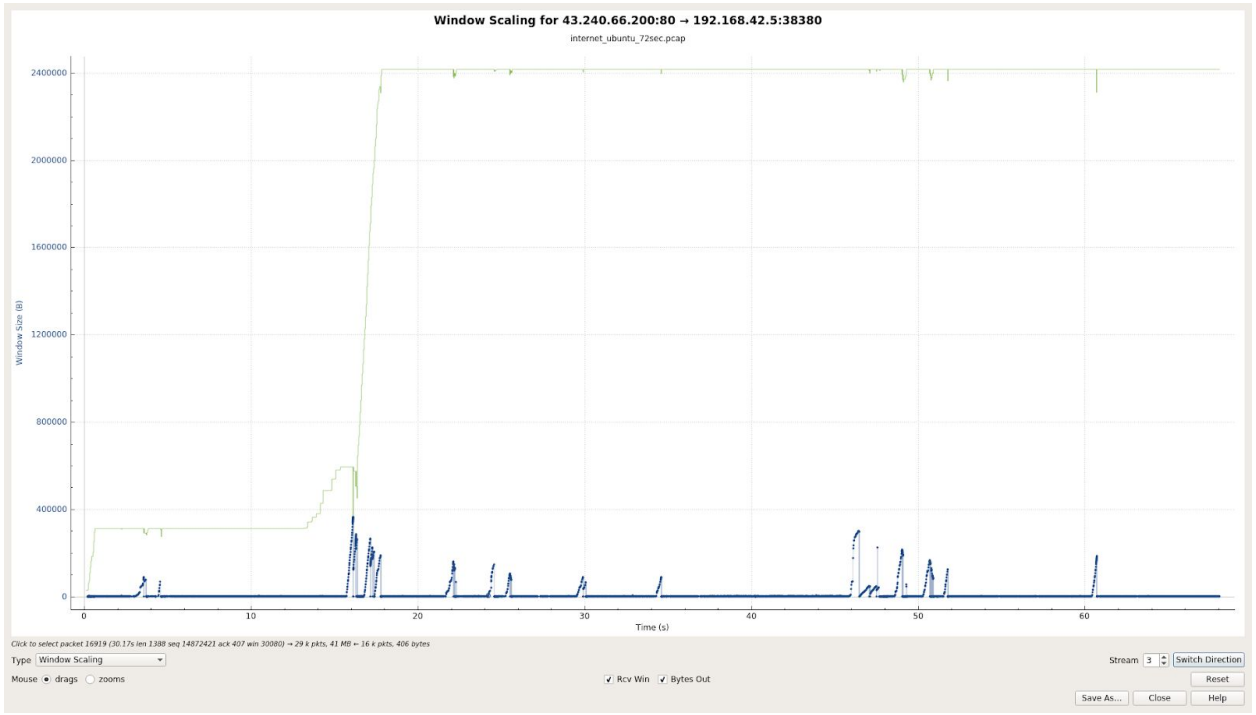
c) The flow graph feature shows a sender and a receiver view of the packet flow. (listing all seq-nos. & ack-nos. , etc). This is a really big graph as it tells of entire conversation



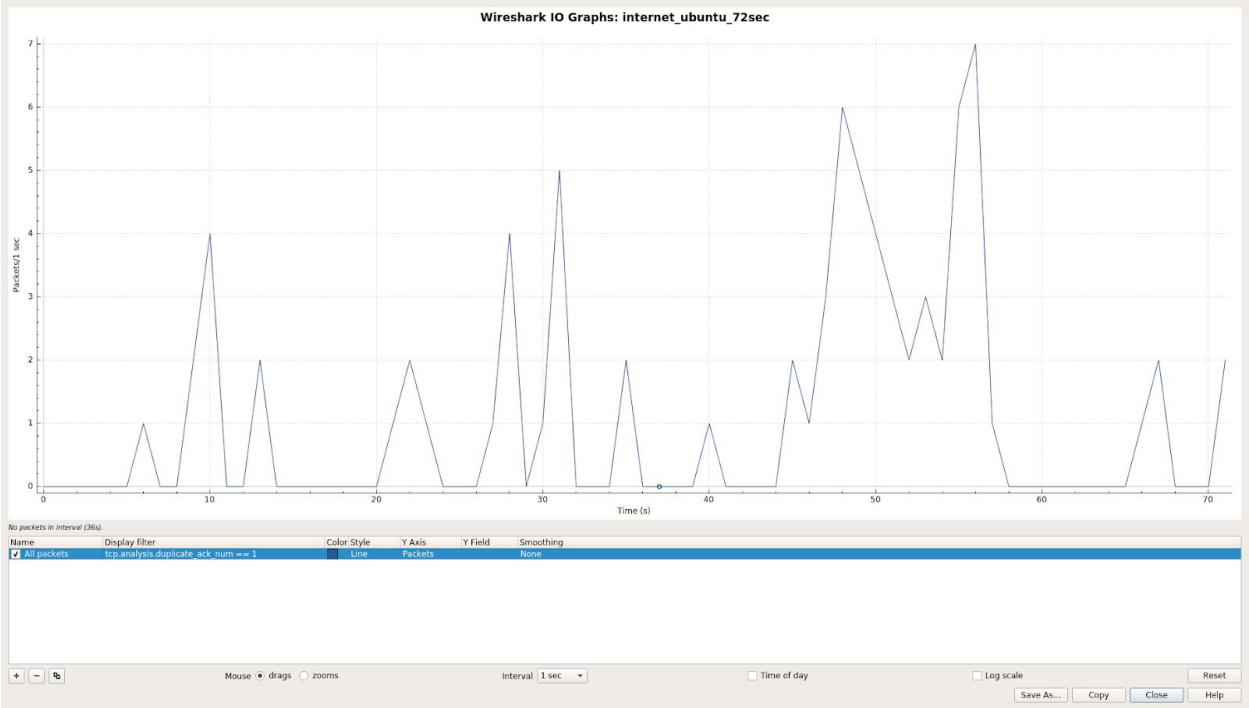
d) Average Throughput from sender -> receiver :: 5087kb/sec
Average Throughput from receiver -> sender :: 133kb/sec



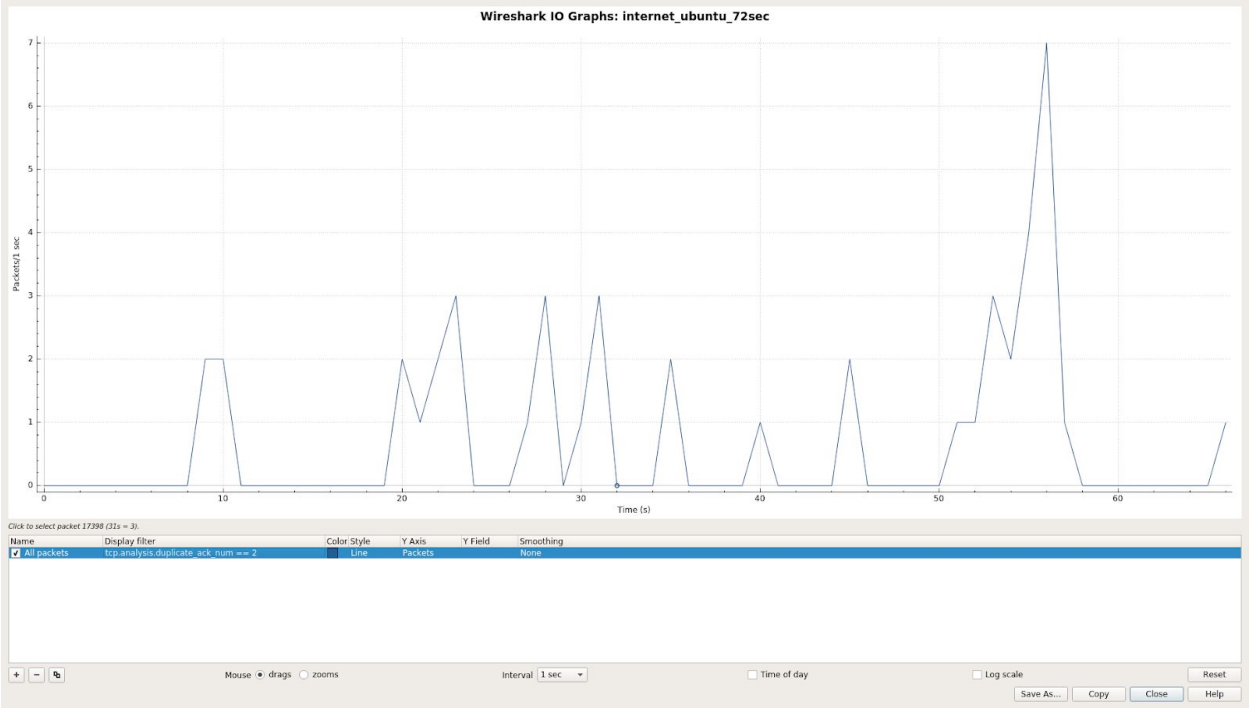
e)
The graph shows The size of the advertised receiver window incl. scaling factor.
The green plot shows receiver_window
The blue plot shows bytes_out. We can see from the green plot that receiver_window size increases (with fluctuations in the middle) and it gets stabilized at 2400000.



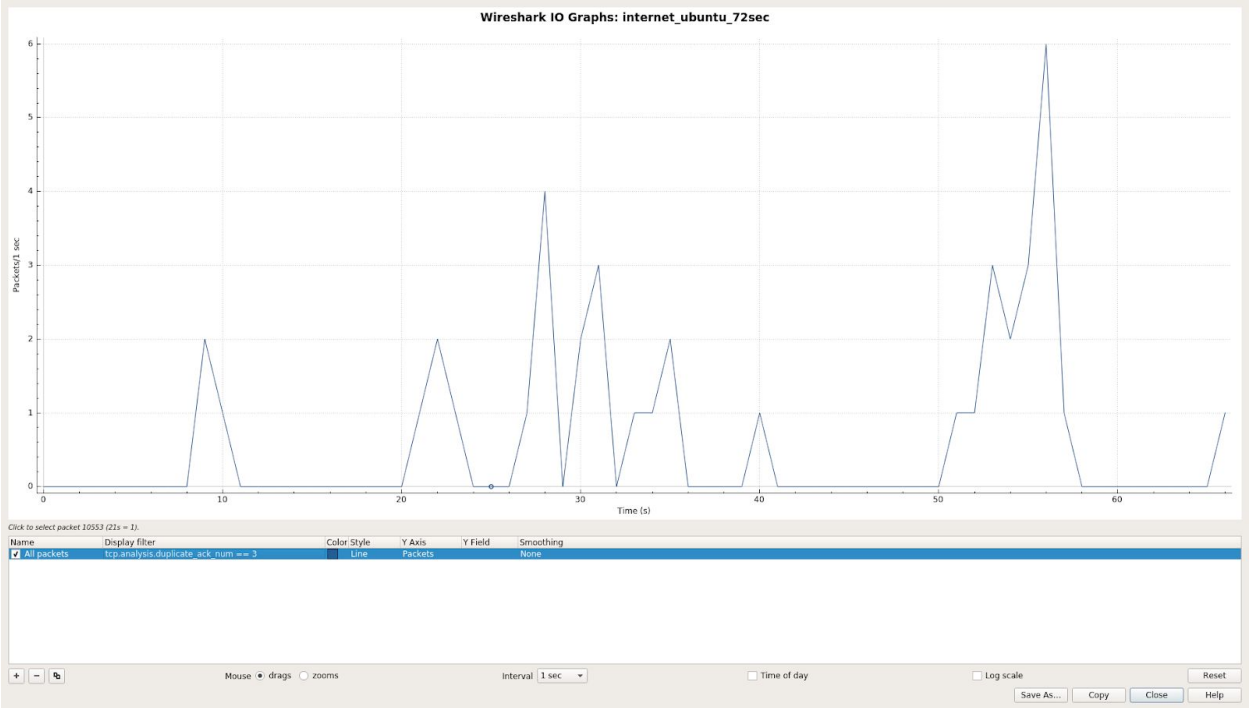
f) 1-duplicate Ack
Fraction :: 77 / 46365 (0.2%)
Command used :: **tcp.analysis.duplicate_ack_num == 1**



2-duplicate Ack
Fraction :: 45 / 46365 (0.1%)
Command used :: **tcp.analysis.duplicate_ack_num == 2**



3-duplicate Ack
Fraction :: 40 / 46365 (0.1%)
Command used :: **tcp.analysis.duplicate_ack_num == 3**

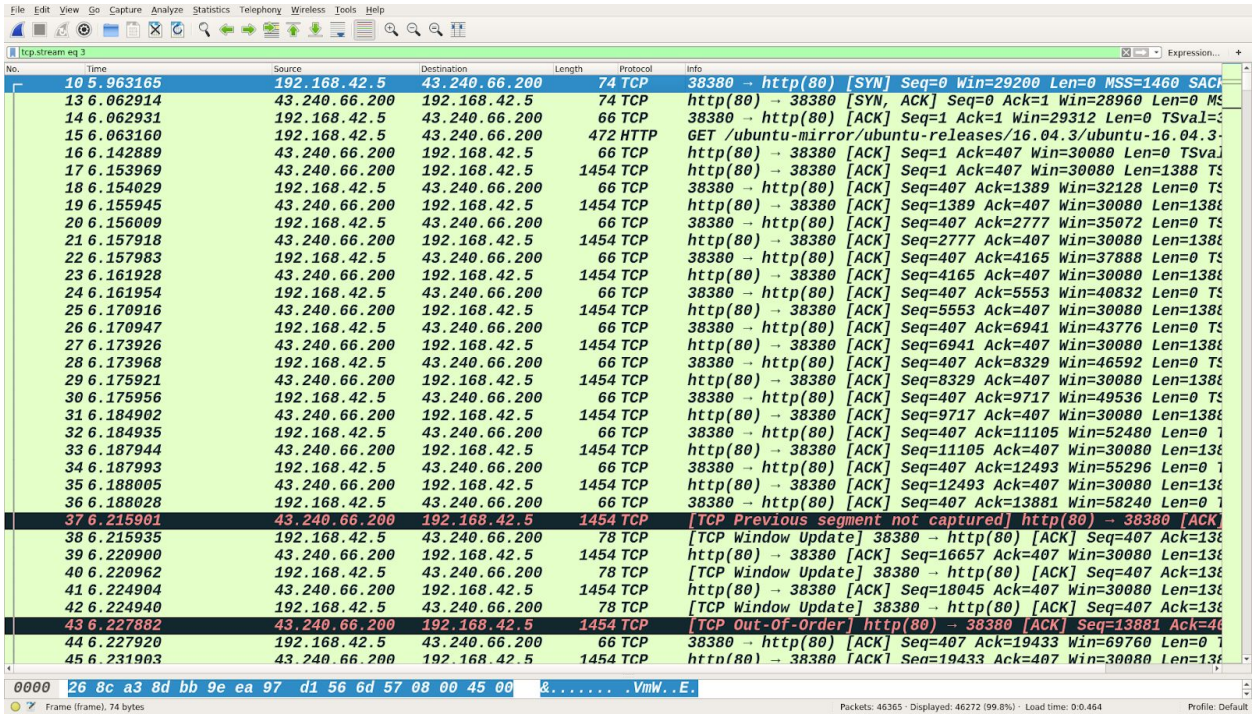


Number of dup-Acks is less compared to other packets. There is difference in the above 3 graphs because for some lost packets it is not needed to go upto 3-dup ACK. 3-dup ACK is mostly used in congestion control (via fast retransmit).

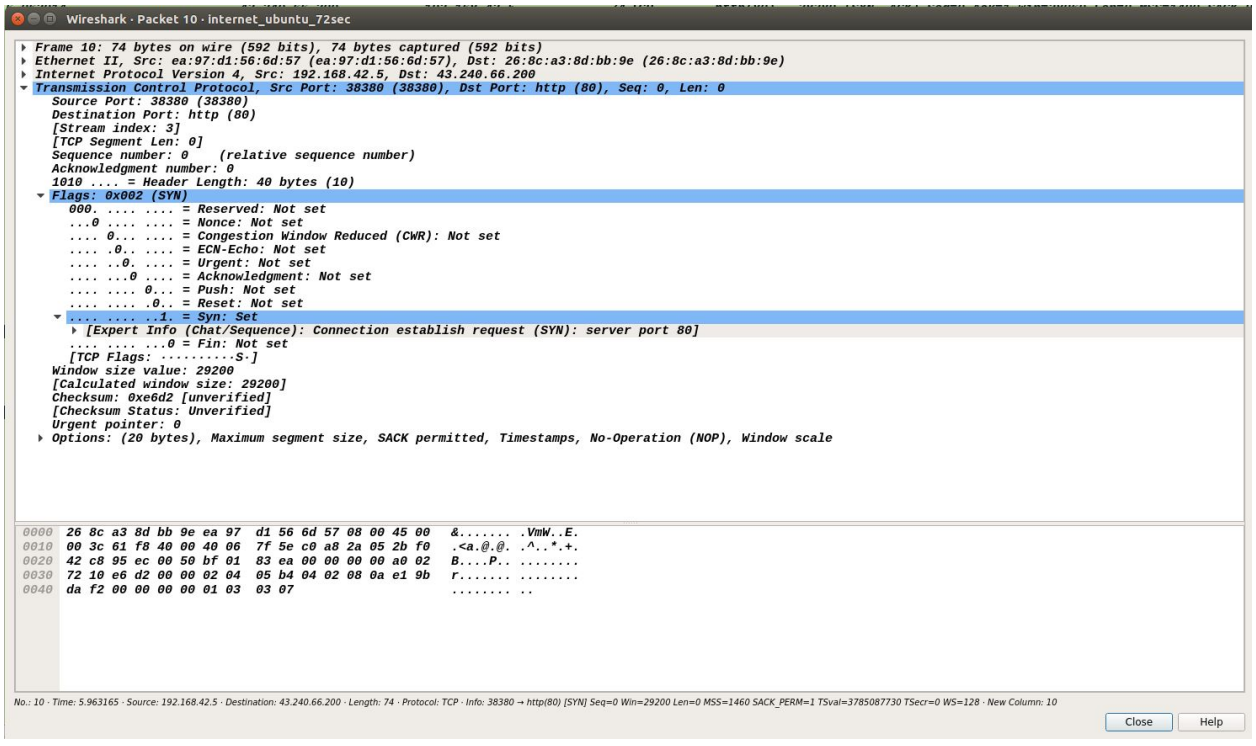
2)

I identified TCP 3-way handshake for the packet captured above.

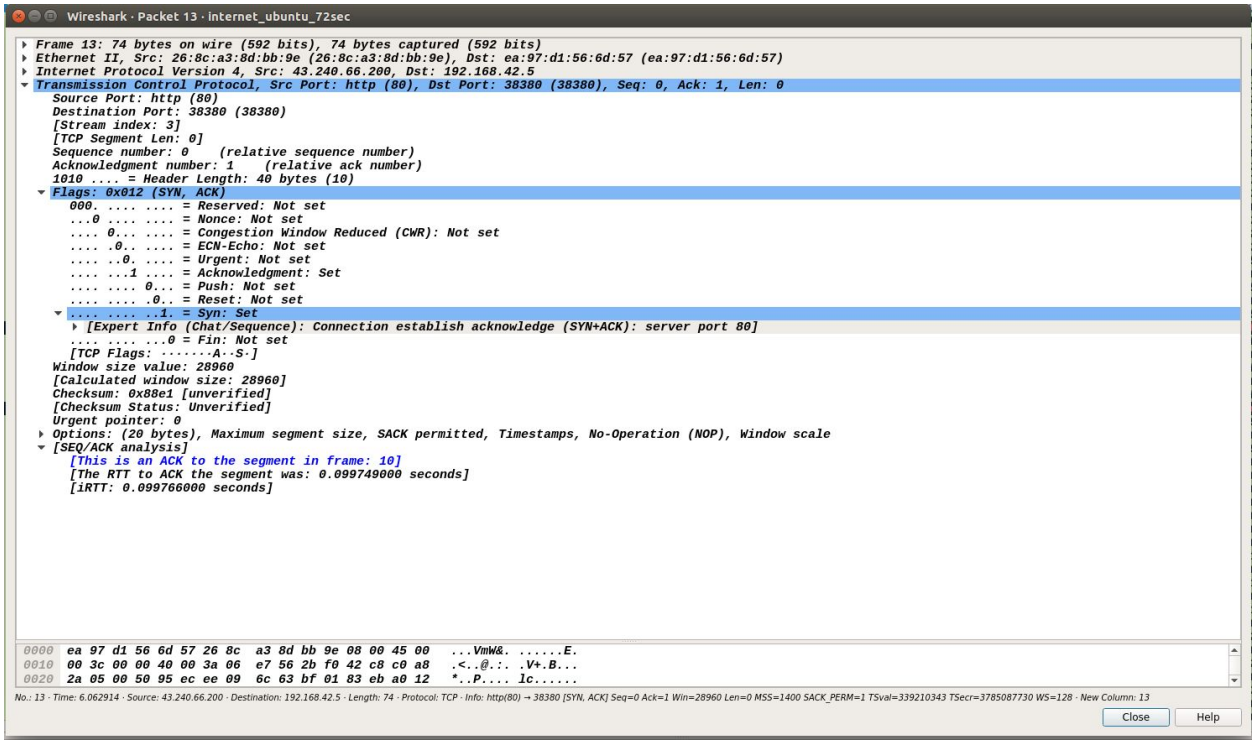
The first 3 packets in screenshot below illustrate 3-way handshake



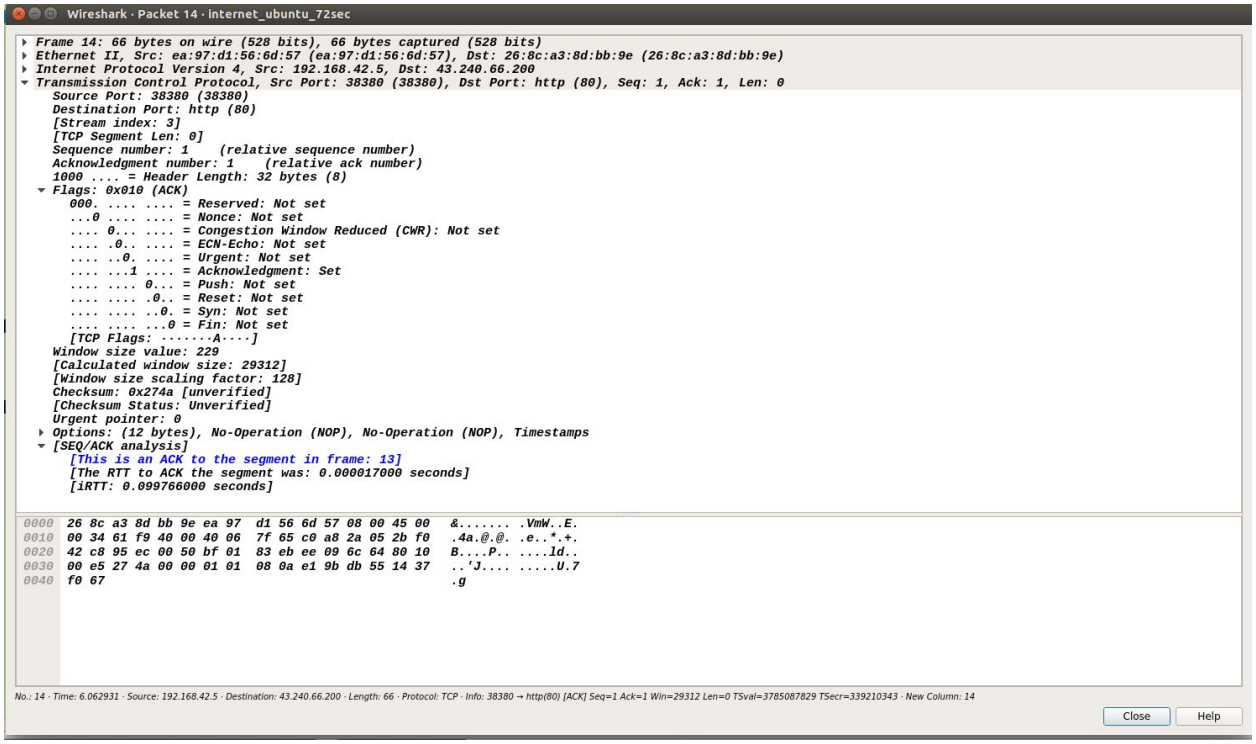
First packet is from receiver to sender with SYN bit as 1. There is no payload (len = 0) and initial sequence number 0 is exchanged (this is relative sequence number given by wireshark)



Second packet is from sender to receiver with SYN bit as 1 & ACK bit as 1. There is no payload (len = 0) and sequence number 0 , ack number 1(+1 than 1st sequence number) is exchanged (this is relative sequence number given by wireshark).



Third packet is from receiver to sender with SYN bit as 0 & ACK bit as 1. There is no payload (len = 0) and sequence number 1 , ack number 1 is exchanged (this is relative sequence number given by wireshark).

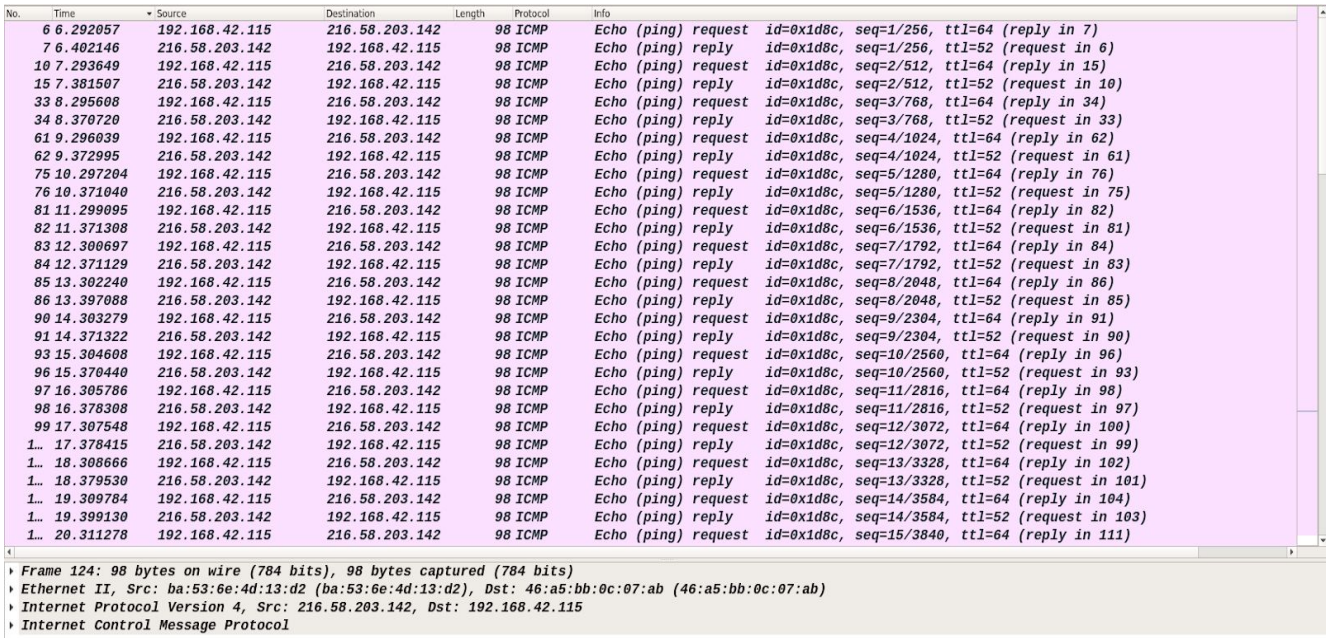


Finally a 4th packet is sent as a GET request for the iso file.

3) For pinging I pinged google.com (216.58.203.142) from my host IP(192.168.42.115)
First a DNS request was made from host followed by a DNS response with the IP address of google.com. The DNS server is at 192.168.42.129.

4	6.290205	192.168.42.115	192.168.42.129	70 DNS	Standard query 0xa310 A google.com
5	6.291516	192.168.42.129	192.168.42.115	86 DNS	Standard query response 0xa310 A google.com A 216.58.203.142

Then repeated ping requests were sent and ping replies were received. The protocol is ICMP , packet length is 98 bytes.
Every request and reply packet has a ttl and a (reply in xx / request in xx). There is also sequence number in the format
<sequence-number-in-big-endian>/<sequence-number-in-little-endian>



No.	Time	Source	Destination	Length	Protocol	Info
6	6.292057	192.168.42.115	216.58.203.142	98	ICMP	Echo (ping) request id=0x1d8c, seq=1/256, ttl=64 (reply in 7)
7	6.402146	216.58.203.142	192.168.42.115	98	ICMP	Echo (ping) reply id=0x1d8c, seq=1/256, ttl=52 (request in 6)
10	7.293649	192.168.42.115	216.58.203.142	98	ICMP	Echo (ping) request id=0x1d8c, seq=2/512, ttl=64 (reply in 15)
15	7.381507	216.58.203.142	192.168.42.115	98	ICMP	Echo (ping) reply id=0x1d8c, seq=2/512, ttl=52 (request in 10)
33	8.295608	192.168.42.115	216.58.203.142	98	ICMP	Echo (ping) request id=0x1d8c, seq=3/768, ttl=64 (reply in 34)
34	8.370720	216.58.203.142	192.168.42.115	98	ICMP	Echo (ping) reply id=0x1d8c, seq=3/768, ttl=52 (request in 33)
61	9.296039	192.168.42.115	216.58.203.142	98	ICMP	Echo (ping) request id=0x1d8c, seq=4/1024, ttl=64 (reply in 62)
62	9.372995	216.58.203.142	192.168.42.115	98	ICMP	Echo (ping) reply id=0x1d8c, seq=4/1024, ttl=52 (request in 61)
75	10.297204	192.168.42.115	216.58.203.142	98	ICMP	Echo (ping) request id=0x1d8c, seq=5/1280, ttl=64 (reply in 76)
76	10.371040	216.58.203.142	192.168.42.115	98	ICMP	Echo (ping) reply id=0x1d8c, seq=5/1280, ttl=52 (request in 75)
81	11.299095	192.168.42.115	216.58.203.142	98	ICMP	Echo (ping) request id=0x1d8c, seq=6/1536, ttl=64 (reply in 82)
82	11.371308	216.58.203.142	192.168.42.115	98	ICMP	Echo (ping) reply id=0x1d8c, seq=6/1536, ttl=52 (request in 81)
83	12.300697	192.168.42.115	216.58.203.142	98	ICMP	Echo (ping) request id=0x1d8c, seq=7/1792, ttl=64 (reply in 84)
84	12.371129	216.58.203.142	192.168.42.115	98	ICMP	Echo (ping) reply id=0x1d8c, seq=7/1792, ttl=52 (request in 83)
85	13.302240	192.168.42.115	216.58.203.142	98	ICMP	Echo (ping) request id=0x1d8c, seq=8/2048, ttl=64 (reply in 86)
86	13.397088	216.58.203.142	192.168.42.115	98	ICMP	Echo (ping) reply id=0x1d8c, seq=8/2048, ttl=52 (request in 85)
90	14.303279	192.168.42.115	216.58.203.142	98	ICMP	Echo (ping) request id=0x1d8c, seq=9/2304, ttl=64 (reply in 91)
91	14.371322	216.58.203.142	192.168.42.115	98	ICMP	Echo (ping) reply id=0x1d8c, seq=9/2304, ttl=52 (request in 90)
93	15.304608	192.168.42.115	216.58.203.142	98	ICMP	Echo (ping) request id=0x1d8c, seq=10/2560, ttl=64 (reply in 96)
96	15.370440	216.58.203.142	192.168.42.115	98	ICMP	Echo (ping) reply id=0x1d8c, seq=10/2560, ttl=52 (request in 93)
97	16.305786	192.168.42.115	216.58.203.142	98	ICMP	Echo (ping) request id=0x1d8c, seq=11/2816, ttl=64 (reply in 98)
98	16.378308	216.58.203.142	192.168.42.115	98	ICMP	Echo (ping) reply id=0x1d8c, seq=11/2816, ttl=52 (request in 97)
99	17.307548	192.168.42.115	216.58.203.142	98	ICMP	Echo (ping) request id=0x1d8c, seq=12/3072, ttl=64 (reply in 100)
1…	17.378415	216.58.203.142	192.168.42.115	98	ICMP	Echo (ping) reply id=0x1d8c, seq=12/3072, ttl=52 (request in 99)
1…	18.308666	192.168.42.115	216.58.203.142	98	ICMP	Echo (ping) request id=0x1d8c, seq=13/3328, ttl=64 (reply in 102)
1…	18.379530	216.58.203.142	192.168.42.115	98	ICMP	Echo (ping) reply id=0x1d8c, seq=13/3328, ttl=52 (request in 101)
1…	19.309784	192.168.42.115	216.58.203.142	98	ICMP	Echo (ping) request id=0x1d8c, seq=14/3584, ttl=64 (reply in 104)
1…	19.399130	216.58.203.142	192.168.42.115	98	ICMP	Echo (ping) reply id=0x1d8c, seq=14/3584, ttl=52 (request in 103)
1…	20.311278	192.168.42.115	216.58.203.142	98	ICMP	Echo (ping) request id=0x1d8c, seq=15/3840, ttl=64 (reply in 111)

```
▸ Frame 124: 98 bytes on wire (784 bits), 98 bytes captured (784 bits)
▸ Ethernet II, Src: ba:53:6e:4d:13:d2 (ba:53:6e:4d:13:d2), Dst: 46:a5:bb:0c:07:ab (46:a5:bb:0c:07:ab)
▸ Internet Protocol Version 4, Src: 216.58.203.142, Dst: 192.168.42.115
▸ Internet Control Message Protocol
```

4) For nmap I created a VM with IP 192.168.244.128. The VM has 4 apps running.

APP	PORT
SSH	22
HTTP APACHE SERVER	80
PROMETHEUS SERVER	9090
NODE-EXPORTER SERVER	9100

Output produced on running **nmap 192.168.244.128**

```
harsh@harsh-Inspiron-5558:~$ nmap 192.168.244.128

Starting Nmap 7.01 ( https://nmap.org ) at 2018-03-01 22:12 IST
Nmap scan report for 192.168.244.128
Host is up (0.0017s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
9090/tcp open  zeus-admin
9100/tcp open  jetdirect

Nmap done: 1 IP address (1 host up) scanned in 13.16 seconds
```

Analysis on wireshark

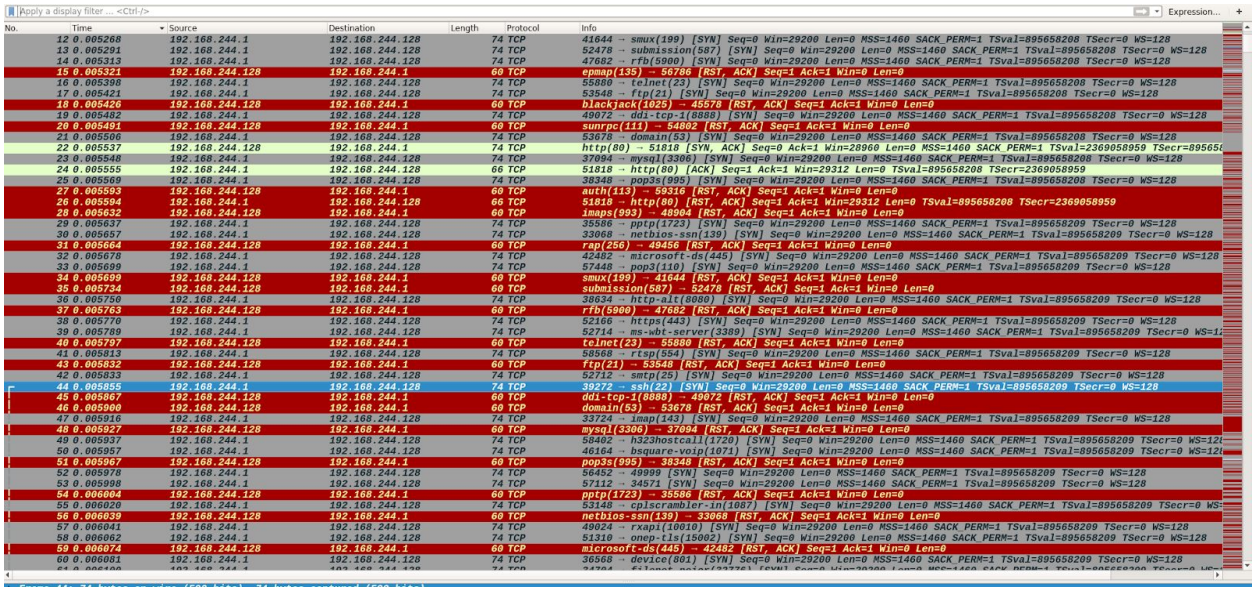
On running nmap 2012 TCP packets were shared between VM IP and vmware interface IP(192.168.244.1)

4 ARP packets between 2 VMware MAC addresses.

(to find MAC address of 192.168.244.128 & 192.168.244.1)

Nmap scans for open ports by sending TCP packets(with SYN bit as 1) with different destination ports. If it receives an ACK back , then it means port is active.

The screenshot shows a subset of the diff ports nmap has scanned.



No.	Time	Source	Destination	Length	Protocol	Info
12	0.005268	192.168.244.1	192.168.244.128	74	TCP	41644 → smux(199) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658208 TSecr=0 WS=128
13	0.005291	192.168.244.1	192.168.244.128	74	TCP	52478 → submission(587) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658208 TSecr=0 WS=128
14	0.005313	192.168.244.1	192.168.244.128	74	TCP	47682 → rfb(5900) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658208 TSecr=0 WS=128
15	0.005321	192.168.244.128	192.168.244.1	60	TCP	epmap(135) → 56706 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
16	0.005398	192.168.244.1	192.168.244.128	74	TCP	55880 → telnet(23) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658208 TSecr=0 WS=128
17	0.005421	192.168.244.1	192.168.244.128	74	TCP	53548 → ftp(21) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658208 TSecr=0 WS=128
18	0.005426	192.168.244.128	192.168.244.1	60	TCP	blackjack(1025) → 45572 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
19	0.005482	192.168.244.1	192.168.244.128	74	TCP	49072 → ddi-tcp-1(8888) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658208 TSecr=0 WS=128
20	0.005491	192.168.244.128	192.168.244.1	60	TCP	sunrpc(111) → 54802 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
21	0.005506	192.168.244.1	192.168.244.128	74	TCP	53678 → domain(53) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658208 TSecr=0 WS=128
22	0.005537	192.168.244.128	192.168.244.1	74	TCP	http(80) → 51818 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=2369058959 TSecr=895658
23	0.005548	192.168.244.1	192.168.244.128	74	TCP	37094 → mysql(3306) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658208 TSecr=0 WS=128
24	0.005555	192.168.244.1	192.168.244.128	66	TCP	51818 → http(80) [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=895658208 TSecr=2369058959
25	0.005569	192.168.244.1	192.168.244.128	74	TCP	38348 → pop3s(995) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658208 TSecr=0 WS=128
27	0.005593	192.168.244.128	192.168.244.1	60	TCP	auth(113) → 59316 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
26	0.005594	192.168.244.1	192.168.244.128	66	TCP	51818 → http(80) [RST, ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=895658208 TSecr=2369058959
28	0.005632	192.168.244.128	192.168.244.1	60	TCP	imaps(993) → 48904 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
29	0.005637	192.168.244.1	192.168.244.128	74	TCP	35586 → pptp(1723) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS=128
30	0.005657	192.168.244.1	192.168.244.128	74	TCP	33068 → netbios-ssn(139) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS=128
31	0.005664	192.168.244.128	192.168.244.1	60	TCP	rap(256) → 49456 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
32	0.005678	192.168.244.1	192.168.244.128	74	TCP	42482 → microsoft-ds(445) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS=128
33	0.005699	192.168.244.1	192.168.244.128	74	TCP	57448 → pop3(110) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS=128
34	0.005699	192.168.244.128	192.168.244.1	60	TCP	smux(199) → 41644 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
35	0.005734	192.168.244.128	192.168.244.1	60	TCP	submission(587) → 52478 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
36	0.005750	192.168.244.1	192.168.244.128	74	TCP	36684 → http-alt(8080) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS=128
37	0.005763	192.168.244.128	192.168.244.1	60	TCP	rfb(5900) → 47682 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
38	0.005770	192.168.244.1	192.168.244.128	74	TCP	52166 → https(443) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS=128
39	0.005789	192.168.244.1	192.168.244.128	74	TCP	52714 → ms-wbt-server(3389) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS=12
40	0.005797	192.168.244.128	192.168.244.1	60	TCP	telnet(23) → 55880 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
41	0.005813	192.168.244.1	192.168.244.128	74	TCP	58568 → rtsp(554) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS=128
43	0.005832	192.168.244.128	192.168.244.1	60	TCP	ftp(21) → 53548 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
42	0.005833	192.168.244.1	192.168.244.128	74	TCP	52712 → smtp(25) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS=128
44	0.005855	192.168.244.1	192.168.244.128	74	TCP	39272 → ssh(22) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS=128
45	0.005867	192.168.244.128	192.168.244.1	60	TCP	ddi-tcp-1(8888) → 49072 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
46	0.005900	192.168.244.128	192.168.244.1	60	TCP	domain(53) → 53678 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
47	0.005916	192.168.244.1	192.168.244.128	74	TCP	33724 → imap(143) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS=128
48	0.005927	192.168.244.128	192.168.244.1	60	TCP	mysql(3306) → 37094 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
49	0.005937	192.168.244.1	192.168.244.128	74	TCP	58402 → h323hostcall(1720) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS=12
50	0.005957	192.168.244.1	192.168.244.128	74	TCP	46164 → bsquare-voip(1071) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS=12
51	0.005967	192.168.244.128	192.168.244.1	60	TCP	pop3s(995) → 38348 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
52	0.005978	192.168.244.1	192.168.244.128	74	TCP	56452 → 49999 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS=128
53	0.005998	192.168.244.1	192.168.244.128	74	TCP	57112 → 34571 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS=128
54	0.006004	192.168.244.128	192.168.244.1	60	TCP	pptp(1723) → 35586 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
55	0.006020	192.168.244.1	192.168.244.128	74	TCP	53148 → cplscrambler-in(1087) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS
56	0.006039	192.168.244.128	192.168.244.1	60	TCP	netbios-ssn(139) → 33068 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
57	0.006041	192.168.244.1	192.168.244.128	74	TCP	49024 → rxapi(10010) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS=128
58	0.006062	192.168.244.1	192.168.244.128	74	TCP	51310 → onep-tls(15002) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS=128
59	0.006074	192.168.244.128	192.168.244.1	60	TCP	microsoft-ds(445) → 42482 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
60	0.006081	192.168.244.1	192.168.244.128	74	TCP	36568 → device(801) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658209 TSecr=0 WS=128

If a port is not up the VM send a TCP packet with RST & ACK field set as 1 , meaning port is active.

tcp.port == 3306

No.	Time	Source	Destination	Length	Protocol	Info
23	0.005548	192.168.244.1	192.168.244.128	74	TCP	37094 → mysql(3306) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658208 TSecr=0 WS=128
48	0.005927	192.168.244.128	192.168.244.1	60	TCP	mysql(3306) → 37094 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

If port is up , nmap tries to find whether the app is listening at the port or transmitting.

On sending TCP SYN packet to 9090 , it got an ACK back.

On sending TCP ACK packet to 9090 , it got an RST back, meaning port is not transmitting anything.

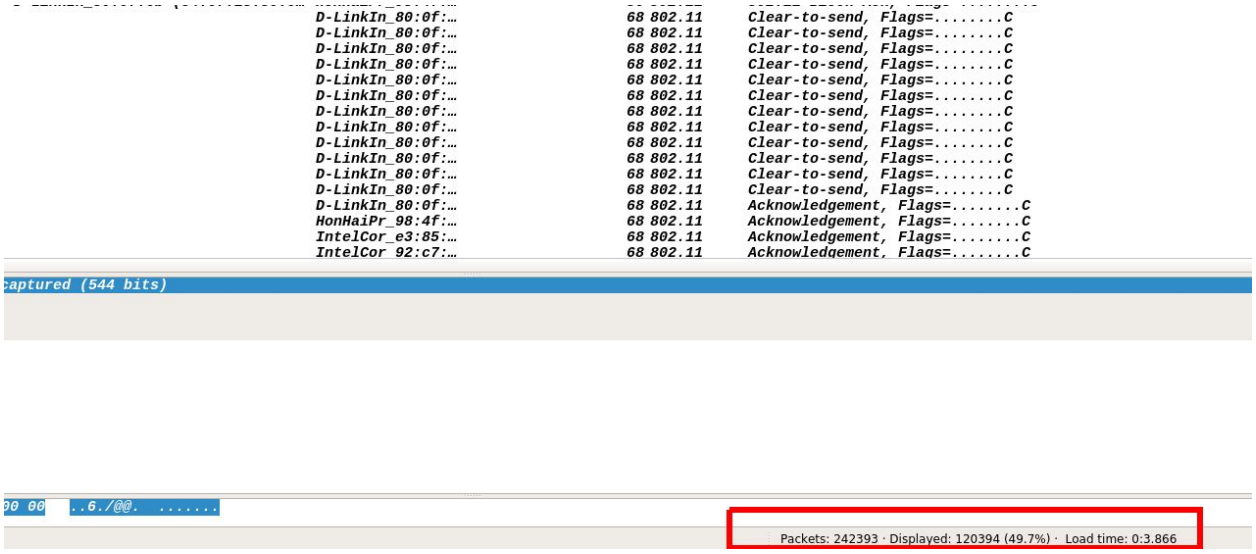
No.	Time	Source	Destination	Length	Protocol	Info
884	0.024705	192.168.244.1	192.168.244.128	74	TCP	49068 → websm(9090) [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=895658228 TSecr=0 WS=128
1024	0.026487	192.168.244.128	192.168.244.1	74	TCP	websm(9090) → 49068 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=2369058964 TSecr=8956582
1025	0.026500	192.168.244.1	192.168.244.128	66	TCP	49068 → websm(9090) [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=895658229 TSecr=2369058964
1040	0.026649	192.168.244.1	192.168.244.128	66	TCP	49068 → websm(9090) [RST, ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=895658230 TSecr=2369058964

Link Layer

1)

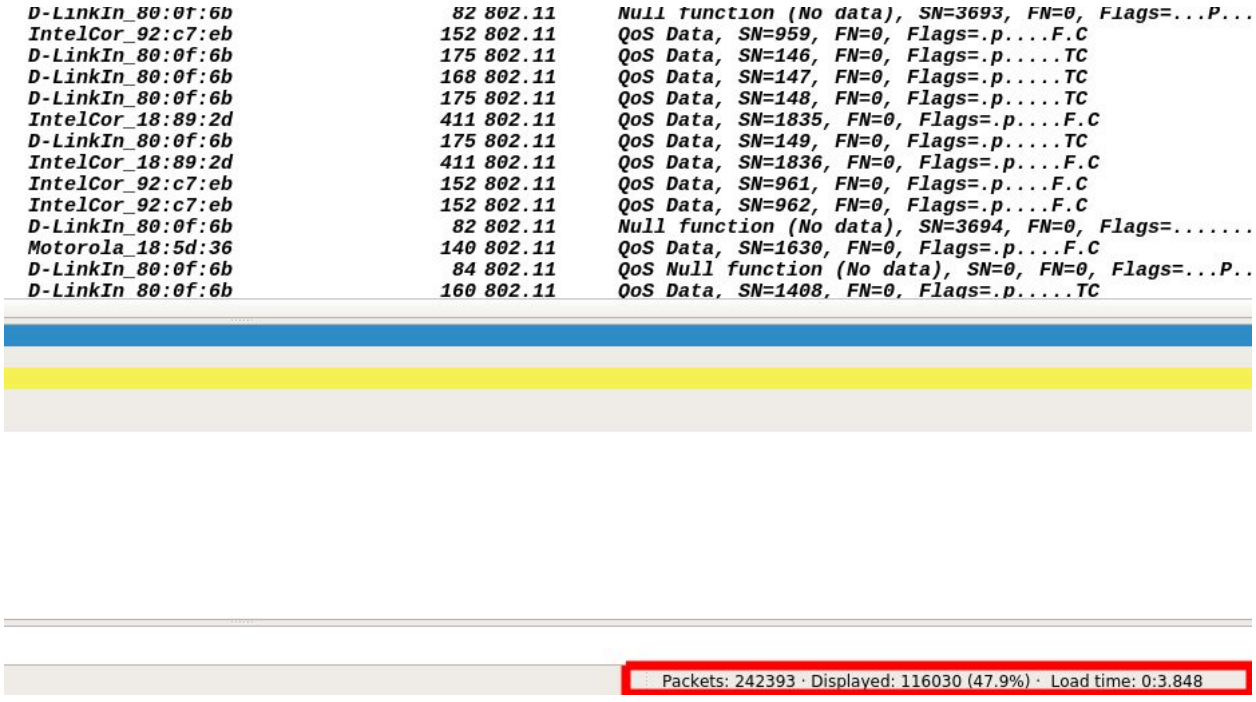
Fraction of Control Packets :: 120394 / 242393 = 49.7%

The command to filter control packets is **wlan.fc.type == 1**



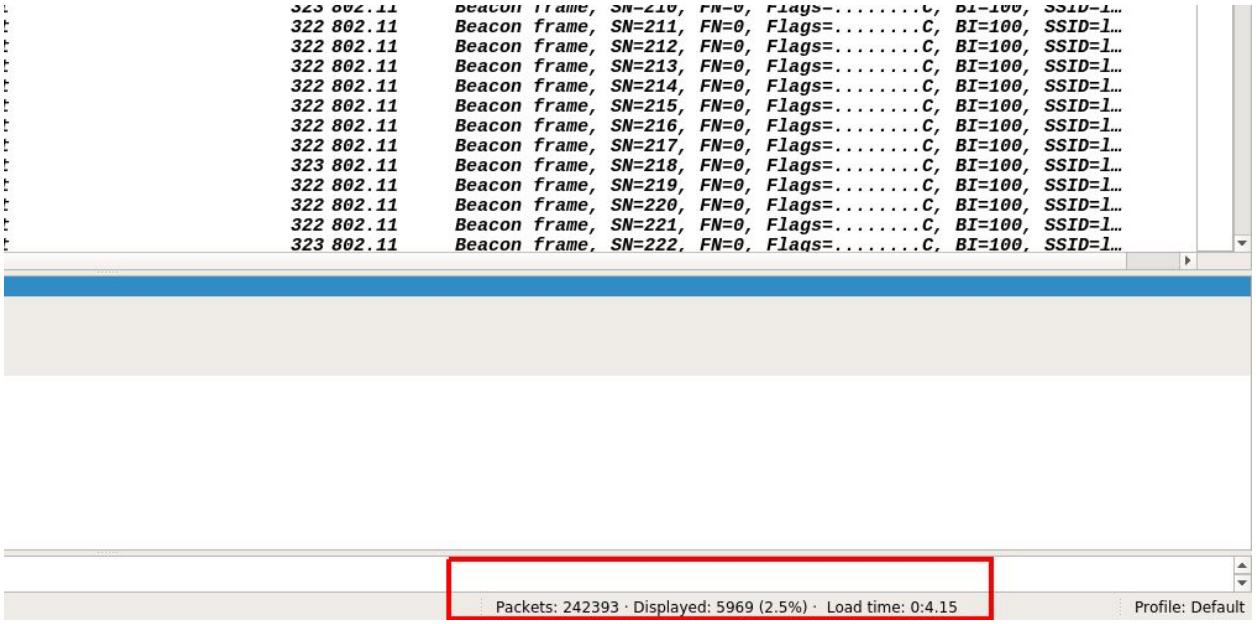
Fraction of Data Packets :: 116030 / 242393 = 47.9%

The command to filter data packets is **wlan.fc.type == 2**



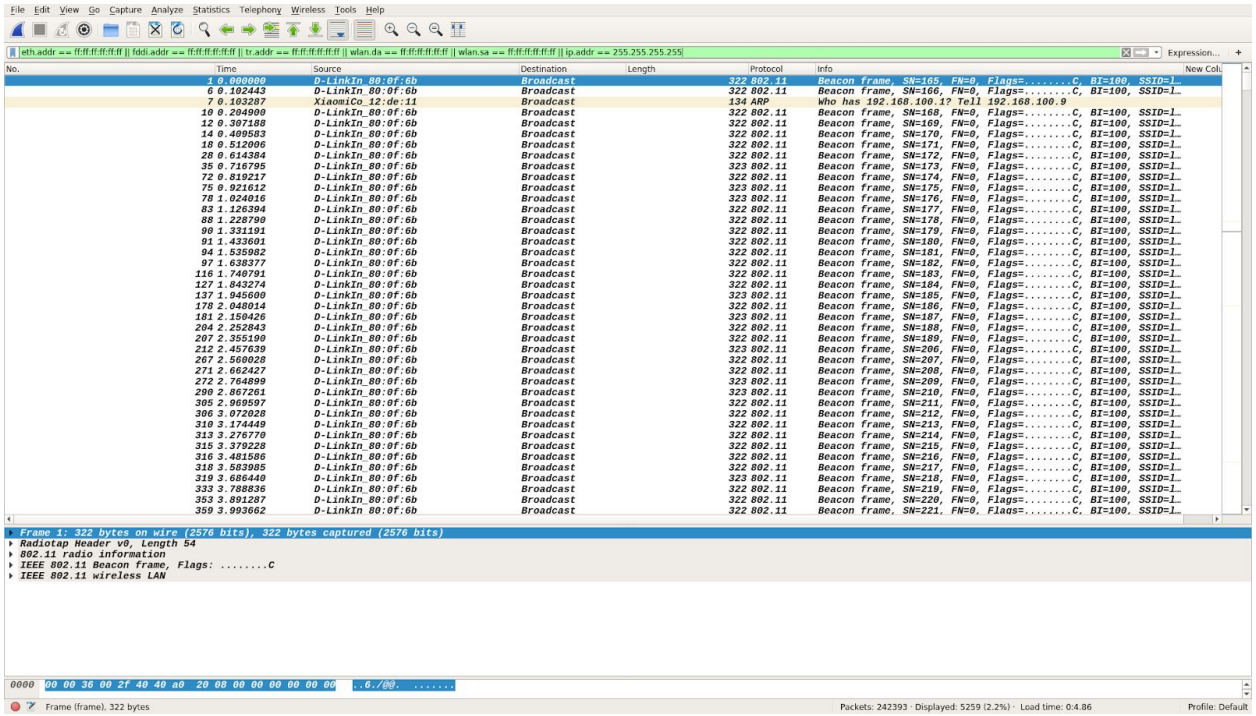
Fraction of Management Packets :: 5969 / 242393 = 2.5%

The command to filter management packets is **wlan.fc.type == 0**



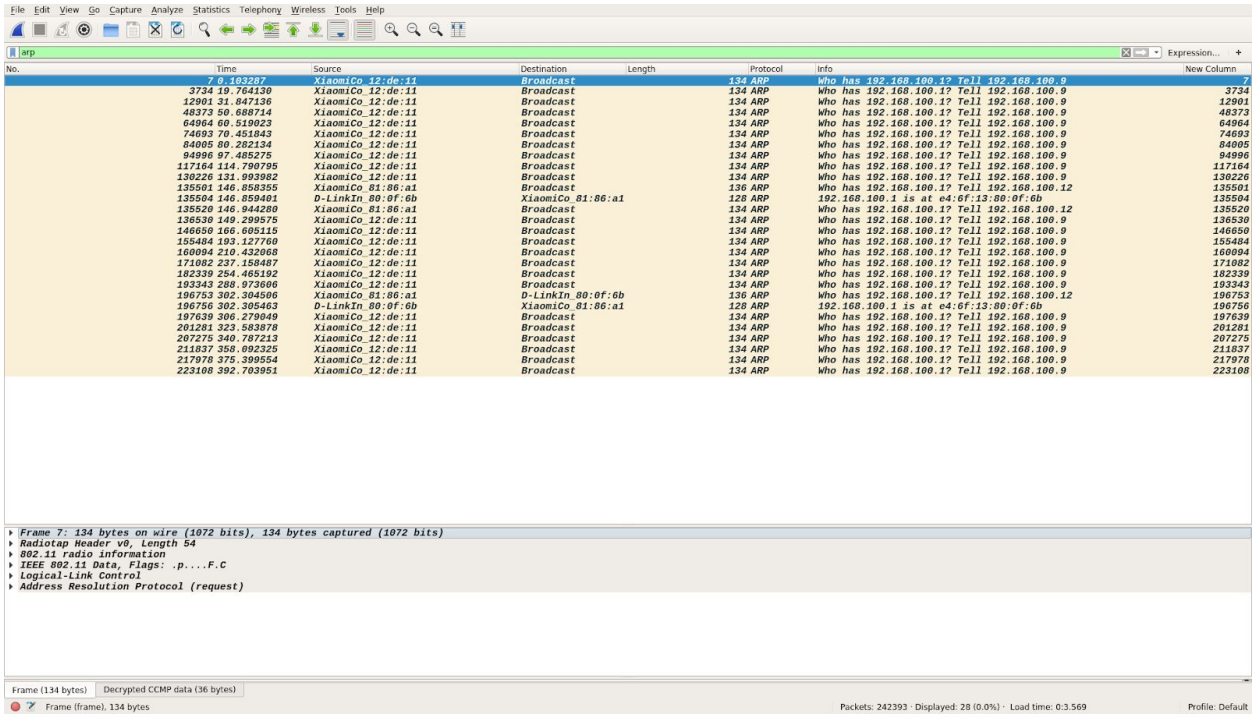
2)

Fraction of ARP :: 28 / 242393 = 0.0%

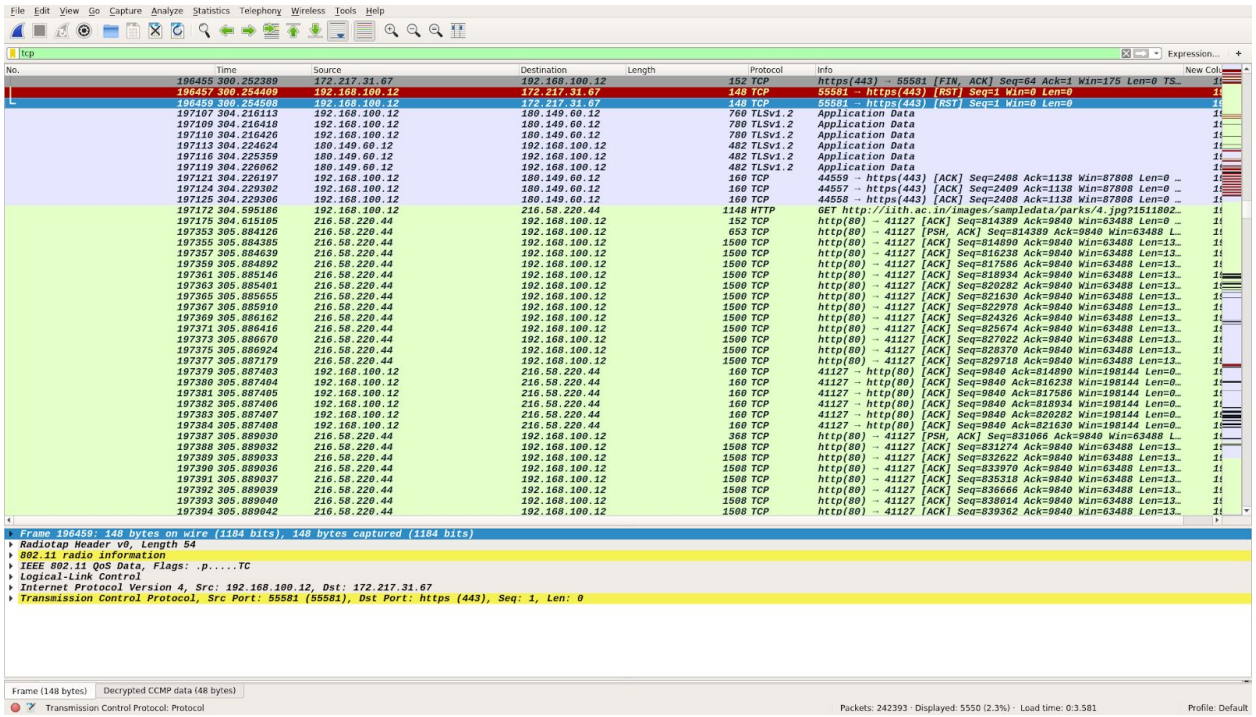


Fraction of Broadcast :: 5259 / 242393 = 2.2%

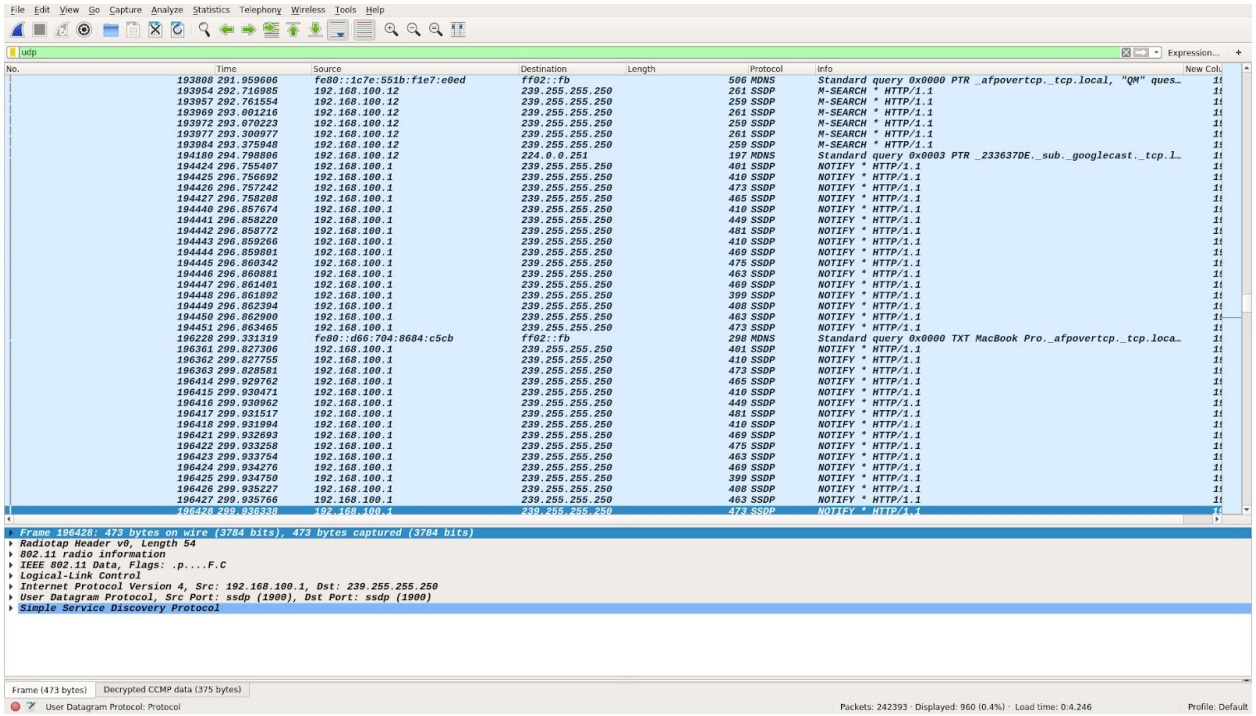
Wireshark Filter :: **eth.addr == ff:ff:ff:ff:ff:ff || fddi.addr == ff:ff:ff:ff:ff:ff || tr.addr == ff:ff:ff:ff:ff:ff || wlan.da == ff:ff:ff:ff:ff:ff || wlan.sa == ff:ff:ff:ff:ff:ff || ip.addr == 255.255.255.255**



Fraction of TCP :: 5550 / 242393 = 2.3%(TCP packets are maximum)

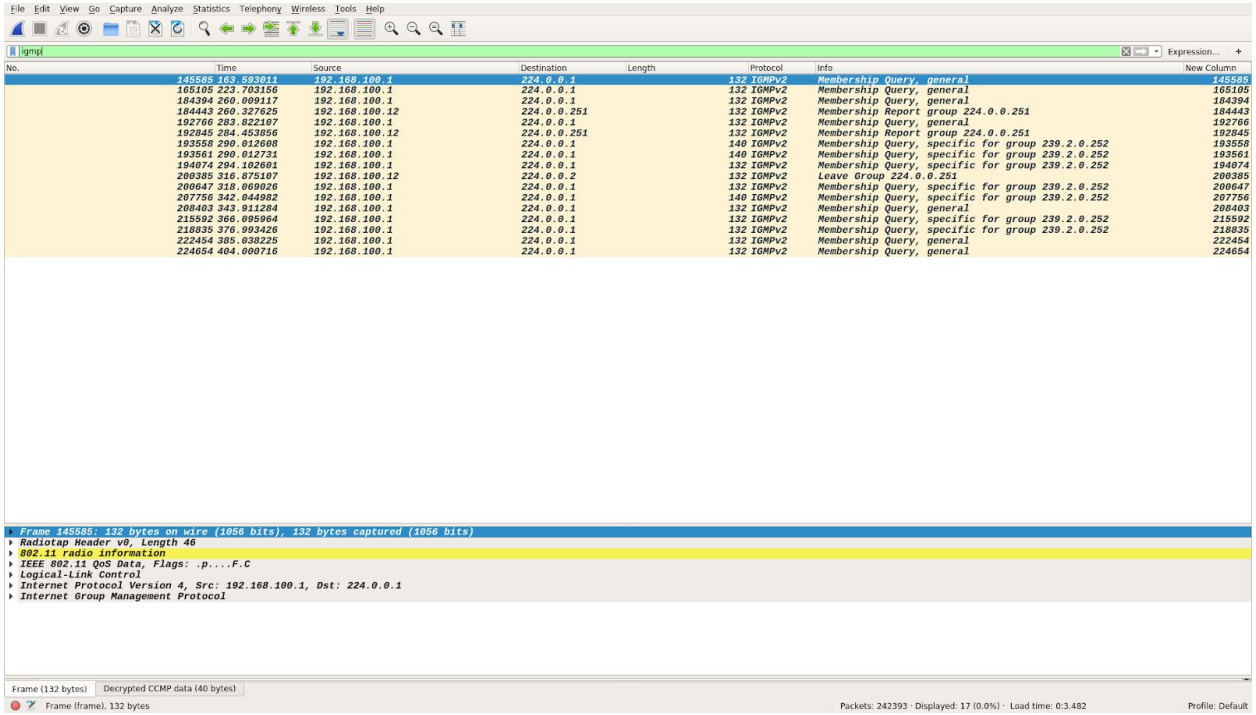


Fraction of UDP :: 960 / 242393 = 0.4% (UDP is lesser than TCP as expected)



Fraction of ICMP :: 0 / 242393 = 0.0% (no ping operation during capture)

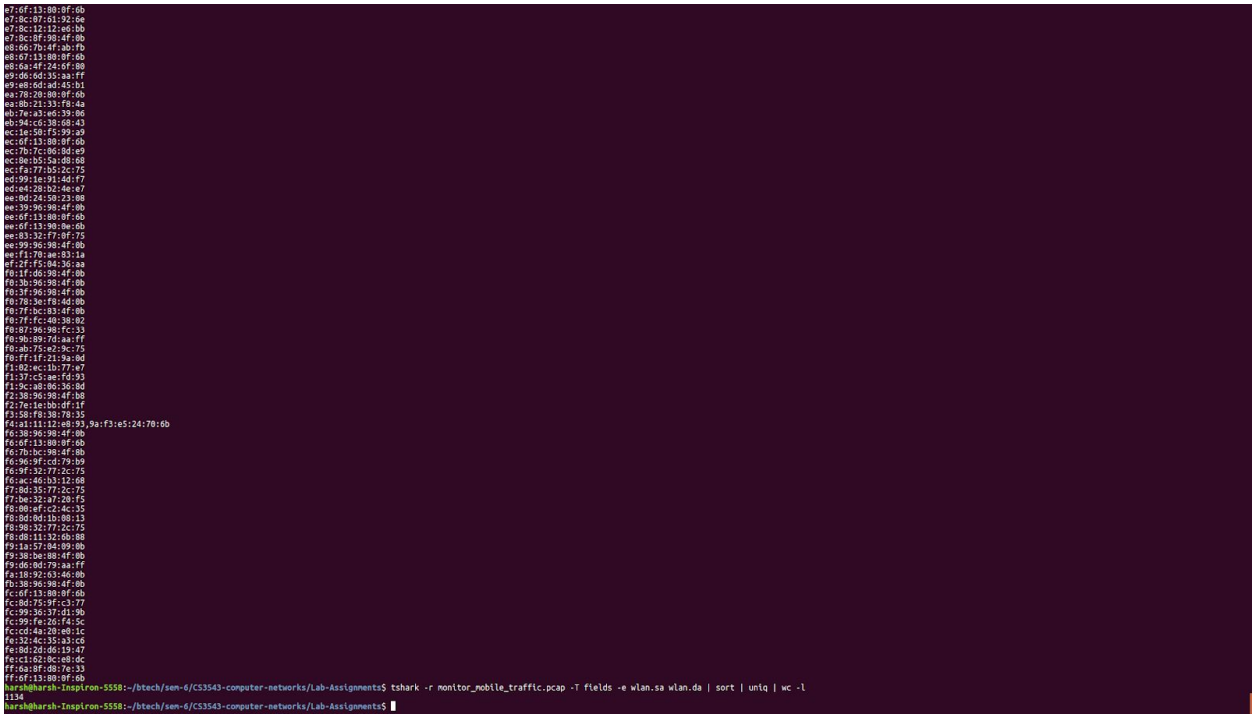
Fraction of IGMP :: 17 / 242393 = 0.0%



3)

Command used :: **tshark -r monitor_mobile_traffic.pcap -T fields -e wlan.sa wlan.da | sort | uniq | wc -l**

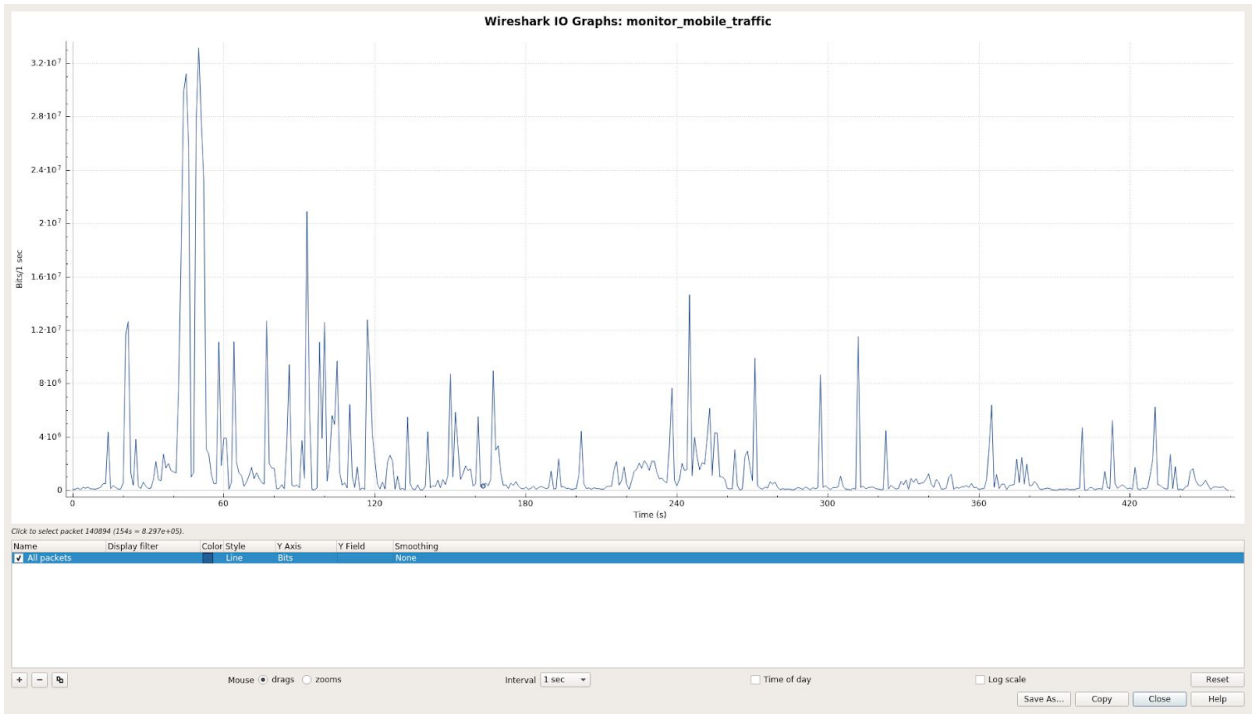
No of unique users (MACs addresses) :: 1134



4)

The traffic pattern from IO graph shows Bits/sec on Y-Axis and time on X-Axis

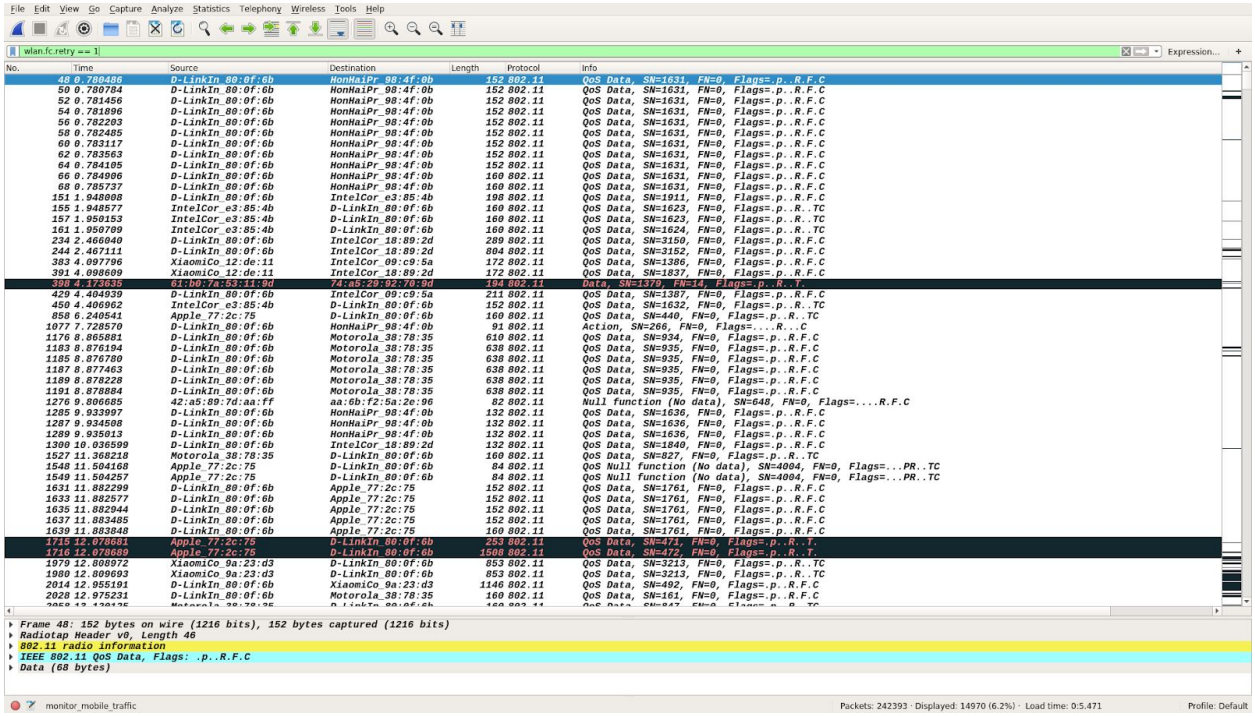
Total packets were captured for 460 seconds . In the beginning seconds there was a sudden burst of traffic. After that traffic was relatively lesser. But small bursts of packets are still there, showing typical internet behaviour.



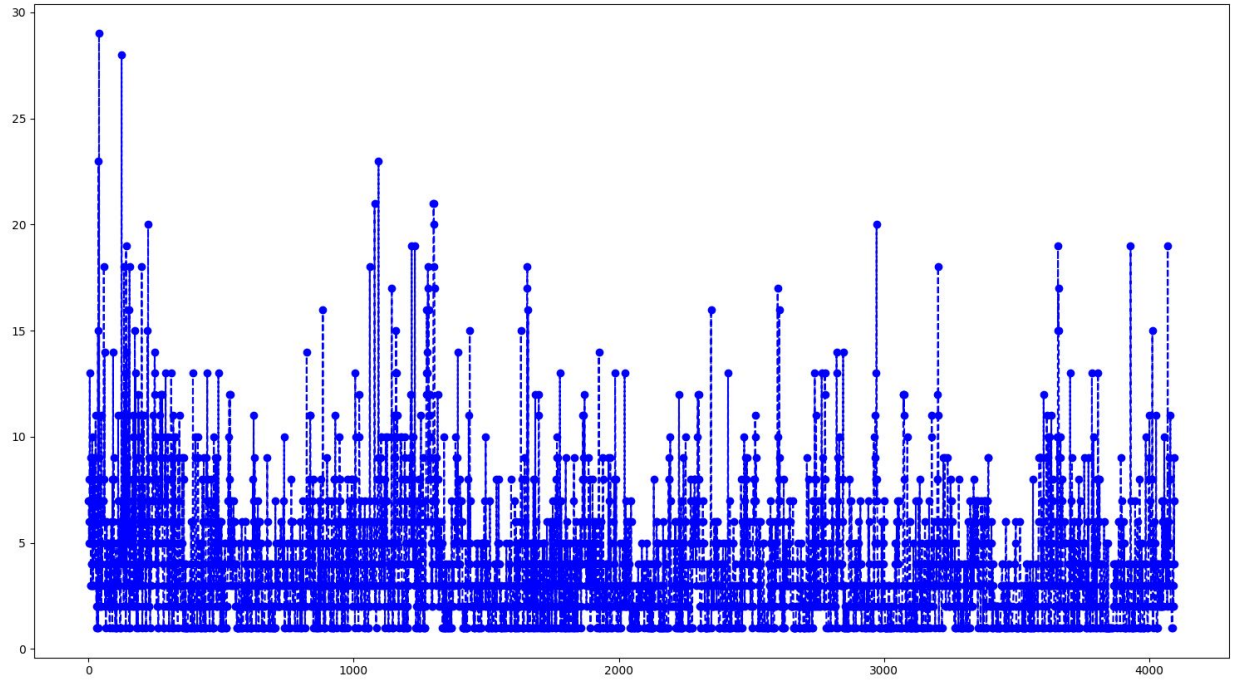
5)

Command used :: **wlan.fc.retry == 1**

Fraction of retransmission :: 14970 / 242393 (6.2%)



6)
This is a plot of **maximum number of retransmission vs packet number.**
The X axis denotes sequence number & the Y axis denotes no. of retransmissions for that sequence number. The sequence numbers that are not in the graph mean that the no. of retransmissions for them is 0. For the sequence number in graph , they have at least 1 retransmission.
The number of retransmissions vary between 1 and 29 and 6.2% of all packets are link-layer retransmission.
This graph was plotted using numpy and pylab.



Drive Links for PCAP files

Pinging_google_com.pcap (for Transport Layer Q3)
https://drive.google.com/file/d/1jtfSLCmuF9VkU_XSk-m-FMeASjRqoAce/view?usp=sharing

Nmap-packets.pcap (for Transport Layer Q4)
https://drive.google.com/file/d/1LIWDpoEliGoWyxkyEck-61AHG232Vtbp/view?usp=sharing

Monitor_mobile_traffic.pcap : containing the packet capture in monitor mode of WiFi for Link-Layer
https://drive.google.com/open?id=1p6CYD_zKkbJ8rDPe3Ztf-ZXnWk4xkN8A

Internet_ubuntu_72sec.pcap :: download ubuntu iso
https://drive.google.com/file/d/1JbWS2w0p1Fx_r1tqMjIzdXkNgyEbhzaC/view?usp=sharing