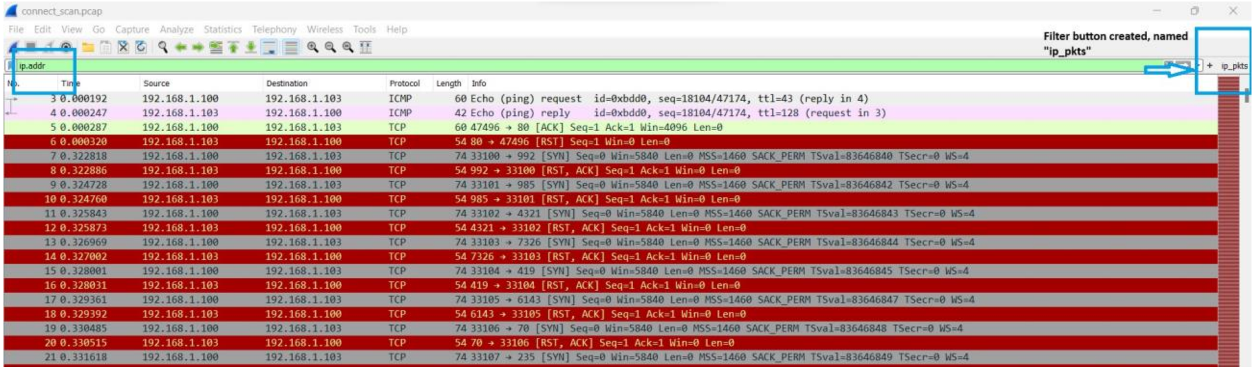
Dan Otieno
CPE 449-01
Wireshark Assignment.
11/11/23.

PART 1:

Connect_scan.pcap:

Write a filter to display all non-IP packets in the file.

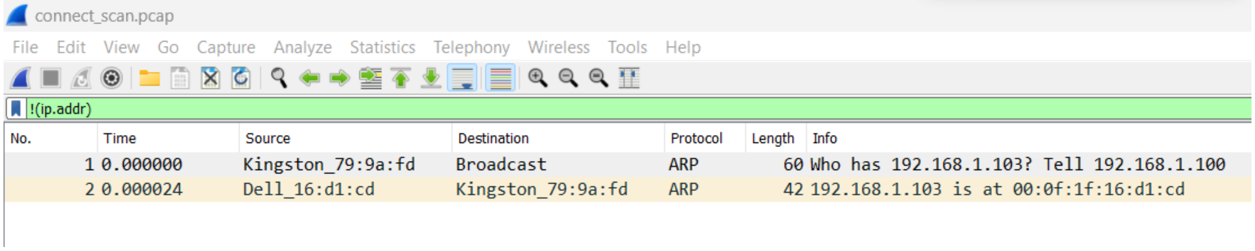
a. **Use the "Expression" button to find a filter that displays only IP packets.**
   - **I created a filter button by pressing the "+" symbol on the right hand side of the filter field.**
   - **Named the filter button "ip_pkts".**
   - **Set filter text to "ip.addr".**
   - **Saved the button and tested it to display only IP packets.**


connect_scan.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Filter button created, named "ip_pkts"
+ ip_pkts

ip.addr

```
No.  Time       Source         Destination    Protocol Length Info
   3 0.000192   192.168.1.100  192.168.1.103  ICMP     60 Echo (ping) request  id=0xbdd0, seq=18104/47174, ttl=43 (reply in 4)
   4 0.000247   192.168.1.103  192.168.1.100  ICMP     42 Echo (ping) reply    id=0xbdd0, seq=18104/47174, ttl=128 (request in 3)
   5 0.000287   192.168.1.100  192.168.1.103  TCP      60 47496 → 80 [ACK] Seq=1 Ack=1 Win=4096 Len=0
   6 0.000320   192.168.1.103  192.168.1.100  TCP      54 80 → 47496 [RST] Seq=1 Win=0 Len=0
   7 0.322818   192.168.1.100  192.168.1.103  TCP      74 33100 → 992 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM TSval=83646840 TSecr=0 WS=4
   8 0.322886   192.168.1.103  192.168.1.100  TCP      54 992 → 33100 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
   9 0.324728   192.168.1.100  192.168.1.103  TCP      74 33101 → 985 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM TSval=83646842 TSecr=0 WS=4
  10 0.324760   192.168.1.103  192.168.1.100  TCP      54 985 → 33101 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
  11 0.325843   192.168.1.100  192.168.1.103  TCP      74 33102 → 4321 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM TSval=83646843 TSecr=0 WS=4
  12 0.325873   192.168.1.103  192.168.1.100  TCP      54 4321 → 33102 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
  13 0.326969   192.168.1.100  192.168.1.103  TCP      74 33103 → 7326 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM TSval=83646844 TSecr=0 WS=4
  14 0.327002   192.168.1.103  192.168.1.100  TCP      54 7326 → 33103 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
  15 0.328001   192.168.1.100  192.168.1.103  TCP      74 33104 → 419 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM TSval=83646845 TSecr=0 WS=4
  16 0.328031   192.168.1.103  192.168.1.100  TCP      54 419 → 33104 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
  17 0.329361   192.168.1.100  192.168.1.103  TCP      74 33105 → 6143 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM TSval=83646847 TSecr=0 WS=4
  18 0.329392   192.168.1.103  192.168.1.100  TCP      54 6143 → 33105 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
  19 0.330485   192.168.1.100  192.168.1.103  TCP      74 33106 → 70 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM TSval=83646848 TSecr=0 WS=4
  20 0.330515   192.168.1.103  192.168.1.100  TCP      54 70 → 33106 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
  21 0.331618   192.168.1.100  192.168.1.103  TCP      74 33107 → 235 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM TSval=83646849 TSecr=0 WS=4
```

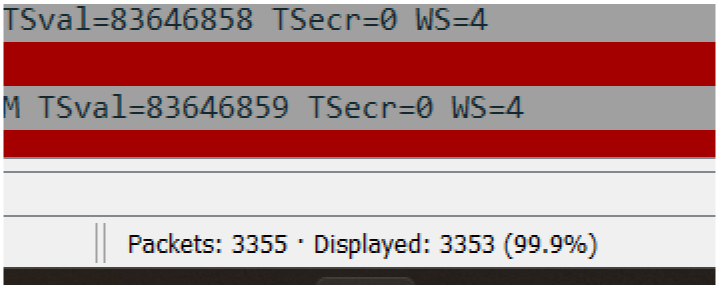
b. **Add a "not" operator in front of your IP filter.**


connect_scan.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

!(ip.addr)

```
No.  Time       Source             Destination        Protocol Length Info
   1 0.000000   Kingston_79:9a:fd  Broadcast          ARP      60 Who has 192.168.1.103? Tell 192.168.1.100
   2 0.000024   Dell_16:d1:cd      Kingston_79:9a:fd  ARP      42 192.168.1.103 is at 00:0f:1f:16:d1:cd
```

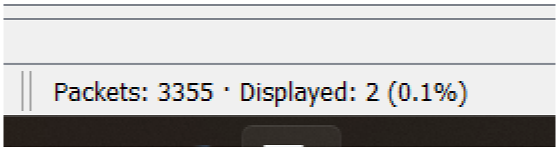
c. **How many packets are displayed using your completed filter?**
   - **When I apply the view IP packets only filter, 3353 packets are displayed.**



```
TSval=83646858 TSecr=0 WS=4

M TSval=83646859 TSecr=0 WS=4
```

Packets: 3355 · Displayed: 3353 (99.9%)

- When I filter out IP packets, only 2 are displayed.



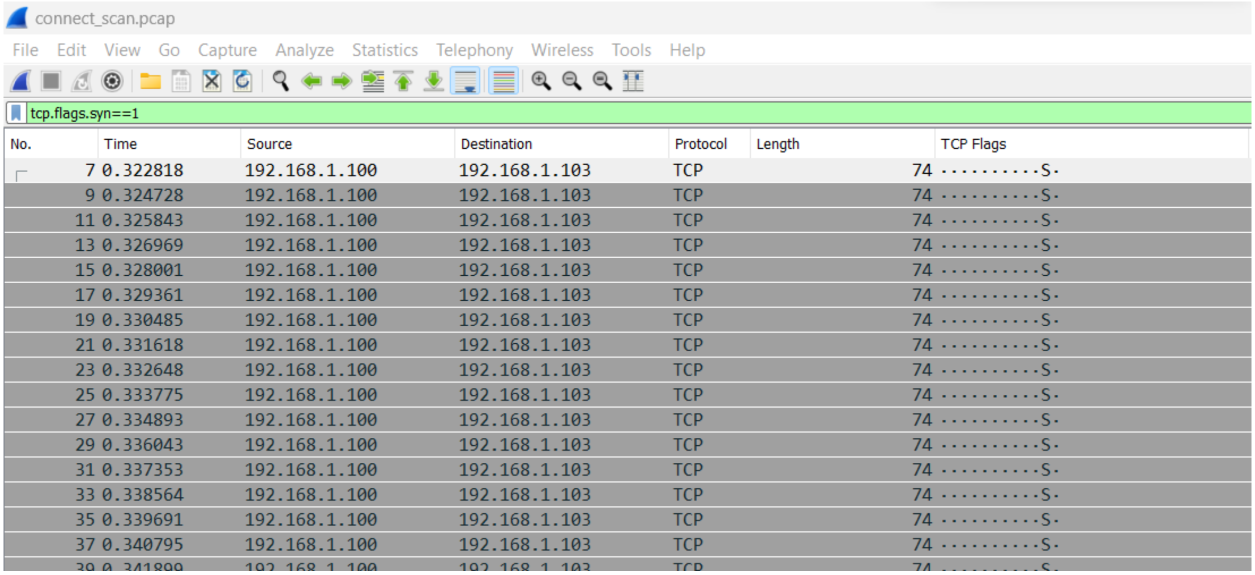
Packets: 3355 · Displayed: 2 (0.1%)

d. Include your final Wireshark filter in your answer. !(ip.addr)

Write a Wireshark filter to display all packets from the scanner.

a. Write a Wireshark filter that shows all TCP packets with the syn flag set.

- Expression used: tcp.flags.syn==1

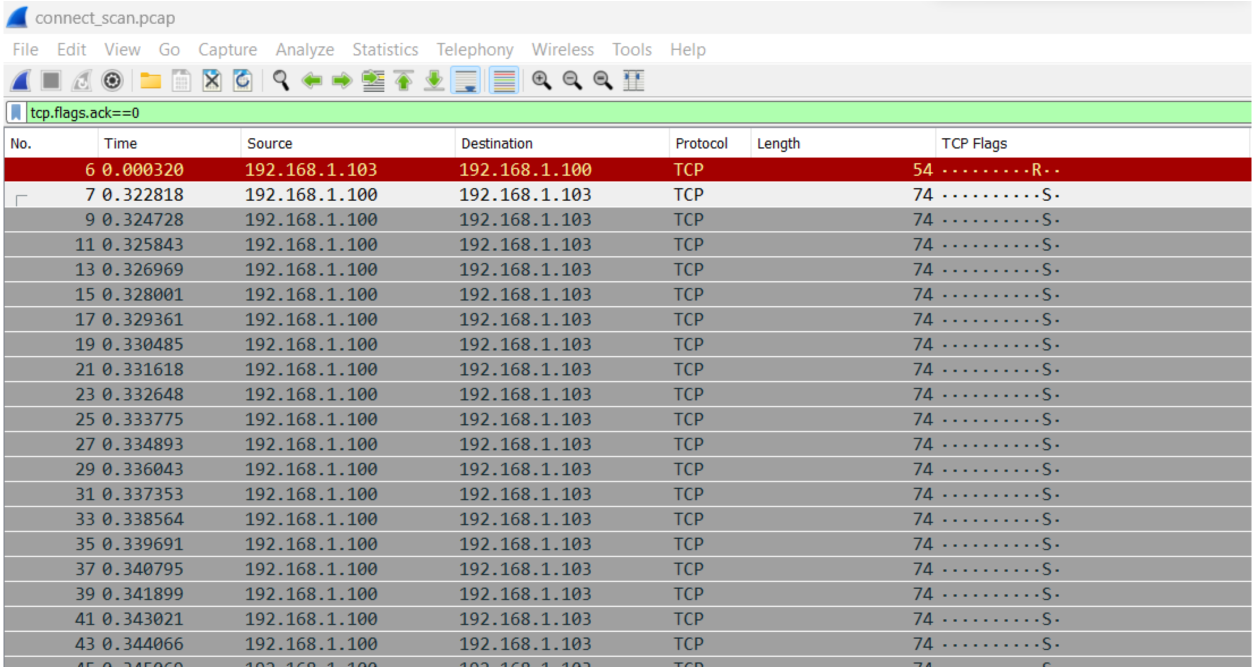


tcp.flags.syn==1

No.	Time	Source	Destination	Protocol	Length	TCP Flags
7	0.322818	192.168.1.100	192.168.1.103	TCP	74	··········S·
9	0.324728	192.168.1.100	192.168.1.103	TCP	74	··········S·
11	0.325843	192.168.1.100	192.168.1.103	TCP	74	··········S·
13	0.326969	192.168.1.100	192.168.1.103	TCP	74	··········S·
15	0.328001	192.168.1.100	192.168.1.103	TCP	74	··········S·
17	0.329361	192.168.1.100	192.168.1.103	TCP	74	··········S·
19	0.330485	192.168.1.100	192.168.1.103	TCP	74	··········S·
21	0.331618	192.168.1.100	192.168.1.103	TCP	74	··········S·
23	0.332648	192.168.1.100	192.168.1.103	TCP	74	··········S·
25	0.333775	192.168.1.100	192.168.1.103	TCP	74	··········S·
27	0.334893	192.168.1.100	192.168.1.103	TCP	74	··········S·
29	0.336043	192.168.1.100	192.168.1.103	TCP	74	··········S·
31	0.337353	192.168.1.100	192.168.1.103	TCP	74	··········S·
33	0.338564	192.168.1.100	192.168.1.103	TCP	74	··········S·
35	0.339691	192.168.1.100	192.168.1.103	TCP	74	··········S·
37	0.340795	192.168.1.100	192.168.1.103	TCP	74	··········S·
39	0.341899	192.168.1.100	192.168.1.103	TCP	74	··········S·

b. Write a second Wireshark filter that shows all TCP packets with the ack flag not set.

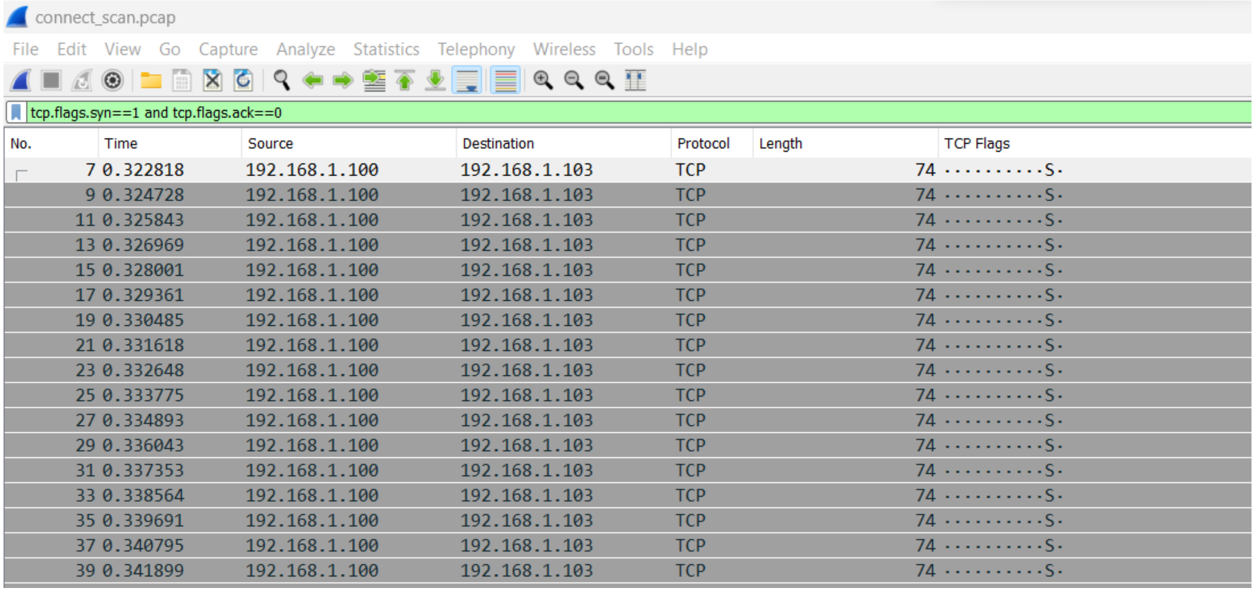
- Expression used: tcp.flags.ack==0.


connect_scan.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp.flags.ack==0

No.	Time	Source	Destination	Protocol	Length	TCP Flags
6	0.000320	192.168.1.103	192.168.1.100	TCP	54	·········R··
7	0.322818	192.168.1.100	192.168.1.103	TCP	74	··········S·
9	0.324728	192.168.1.100	192.168.1.103	TCP	74	··········S·
11	0.325843	192.168.1.100	192.168.1.103	TCP	74	··········S·
13	0.326969	192.168.1.100	192.168.1.103	TCP	74	··········S·
15	0.328001	192.168.1.100	192.168.1.103	TCP	74	··········S·
17	0.329361	192.168.1.100	192.168.1.103	TCP	74	··········S·
19	0.330485	192.168.1.100	192.168.1.103	TCP	74	··········S·
21	0.331618	192.168.1.100	192.168.1.103	TCP	74	··········S·
23	0.332648	192.168.1.100	192.168.1.103	TCP	74	··········S·
25	0.333775	192.168.1.100	192.168.1.103	TCP	74	··········S·
27	0.334893	192.168.1.100	192.168.1.103	TCP	74	··········S·
29	0.336043	192.168.1.100	192.168.1.103	TCP	74	··········S·
31	0.337353	192.168.1.100	192.168.1.103	TCP	74	··········S·
33	0.338564	192.168.1.100	192.168.1.103	TCP	74	··········S·
35	0.339691	192.168.1.100	192.168.1.103	TCP	74	··········S·
37	0.340795	192.168.1.100	192.168.1.103	TCP	74	··········S·
39	0.341899	192.168.1.100	192.168.1.103	TCP	74	··········S·
41	0.343021	192.168.1.100	192.168.1.103	TCP	74	··········S·
43	0.344066	192.168.1.100	192.168.1.103	TCP	74	··········S·

c. **Combine the above two filters with an AND statement to see the port scan as a set of SYN packets originating from the source IP.**


connect_scan.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp.flags.syn==1 and tcp.flags.ack==0

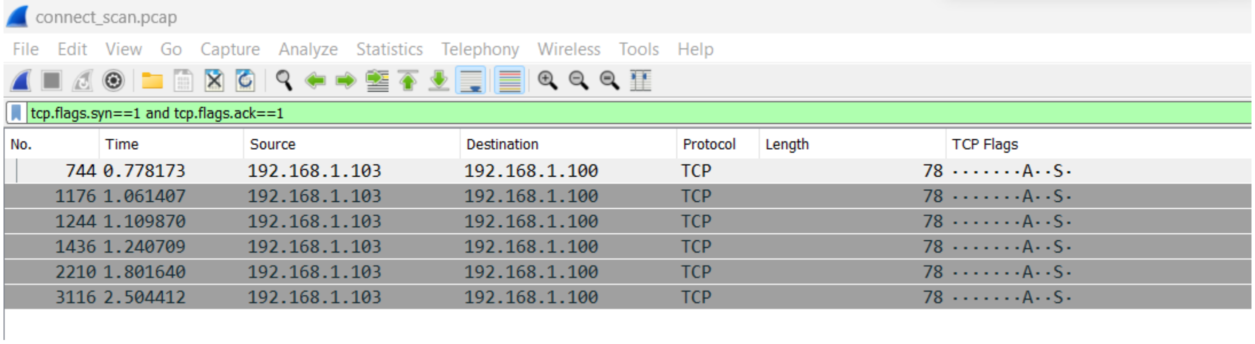
No.	Time	Source	Destination	Protocol	Length	TCP Flags
7	0.322818	192.168.1.100	192.168.1.103	TCP	74	··········S·
9	0.324728	192.168.1.100	192.168.1.103	TCP	74	··········S·
11	0.325843	192.168.1.100	192.168.1.103	TCP	74	··········S·
13	0.326969	192.168.1.100	192.168.1.103	TCP	74	··········S·
15	0.328001	192.168.1.100	192.168.1.103	TCP	74	··········S·
17	0.329361	192.168.1.100	192.168.1.103	TCP	74	··········S·
19	0.330485	192.168.1.100	192.168.1.103	TCP	74	··········S·
21	0.331618	192.168.1.100	192.168.1.103	TCP	74	··········S·
23	0.332648	192.168.1.100	192.168.1.103	TCP	74	··········S·
25	0.333775	192.168.1.100	192.168.1.103	TCP	74	··········S·
27	0.334893	192.168.1.100	192.168.1.103	TCP	74	··········S·
29	0.336043	192.168.1.100	192.168.1.103	TCP	74	··········S·
31	0.337353	192.168.1.100	192.168.1.103	TCP	74	··········S·
33	0.338564	192.168.1.100	192.168.1.103	TCP	74	··········S·
35	0.339691	192.168.1.100	192.168.1.103	TCP	74	··········S·
37	0.340795	192.168.1.100	192.168.1.103	TCP	74	··········S·
39	0.341899	192.168.1.100	192.168.1.103	TCP	74	··········S·

d. **Include the combined Wireshark filter in your answer.**
   - **tcp.flags.syn==1 and tcp.flags.ack==0.**

e. **What is the IP address of the scanner? In the screenshot above, 192.168.1.100 listed as the source IP address, with the syn flag is attributed to it.**

Write a Wireshark filter to display packets from successful TCP connection (established connections) requests.

a. Write a Wireshark filter to display [SYN, ACK] packets returning to the scanner from the victim. This filter should identify packets sent to the scanner IP address with SYN and ACK flags set.


connect_scan.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp.flags.syn==1 and tcp.flags.ack==1

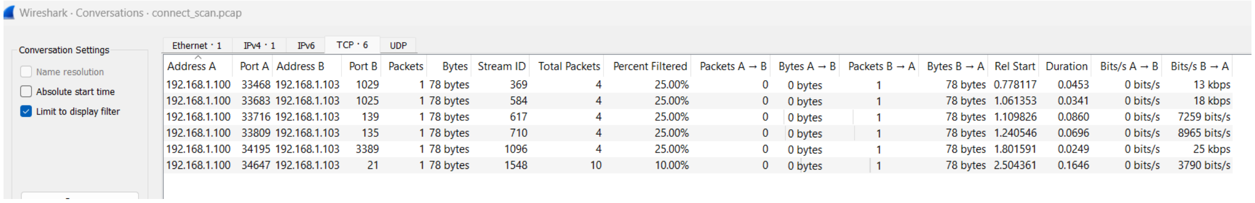
No.	Time	Source	Destination	Protocol	Length	TCP Flags
744	0.778173	192.168.1.103	192.168.1.100	TCP	78	·······A··S·
1176	1.061407	192.168.1.103	192.168.1.100	TCP	78	·······A··S·
1244	1.109870	192.168.1.103	192.168.1.100	TCP	78	·······A··S·
1436	1.240709	192.168.1.103	192.168.1.100	TCP	78	·······A··S·
2210	1.801640	192.168.1.103	192.168.1.100	TCP	78	·······A··S·
3116	2.504412	192.168.1.103	192.168.1.100	TCP	78	·······A··S·

b. Include this Wireshark filter in your answer.

- tcp.flags.syn==1 and tcp.flags.ack==1

c. How many open ports were found by the scanner?

- 6 ports are open.



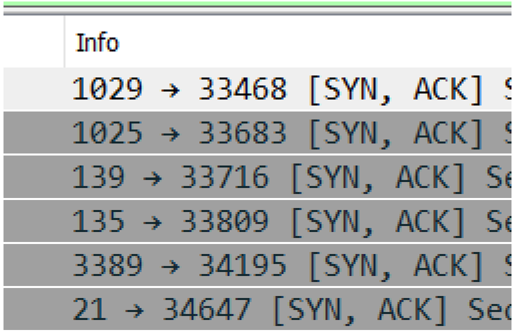
Conversation Settings

☐ Name resolution

☐ Absolute start time

☑ Limit to display filter

Address A	Port A	Address B	Port B	Packets	Bytes	Stream ID	Total Packets	Percent Filtered	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
192.168.1.100	33468	192.168.1.103	1029	1	78 bytes	369	4	25.00%	0	0 bytes	1	78 bytes	0.778117	0.0453	0 bits/s	13 kbps
192.168.1.100	33683	192.168.1.103	1025	1	78 bytes	584	4	25.00%	0	0 bytes	1	78 bytes	1.061353	0.0341	0 bits/s	18 kbps
192.168.1.100	33716	192.168.1.103	139	1	78 bytes	617	4	25.00%	0	0 bytes	1	78 bytes	1.109826	0.0860	0 bits/s	7259 bits/s
192.168.1.100	33809	192.168.1.103	135	1	78 bytes	710	4	25.00%	0	0 bytes	1	78 bytes	1.240546	0.0696	0 bits/s	8965 bits/s
192.168.1.100	34195	192.168.1.103	3389	1	78 bytes	1096	4	25.00%	0	0 bytes	1	78 bytes	1.801591	0.0249	0 bits/s	25 kbps
192.168.1.100	34647	192.168.1.103	21	1	78 bytes	1548	10	10.00%	0	0 bytes	1	78 bytes	2.504361	0.1646	0 bits/s	3790 bits/s



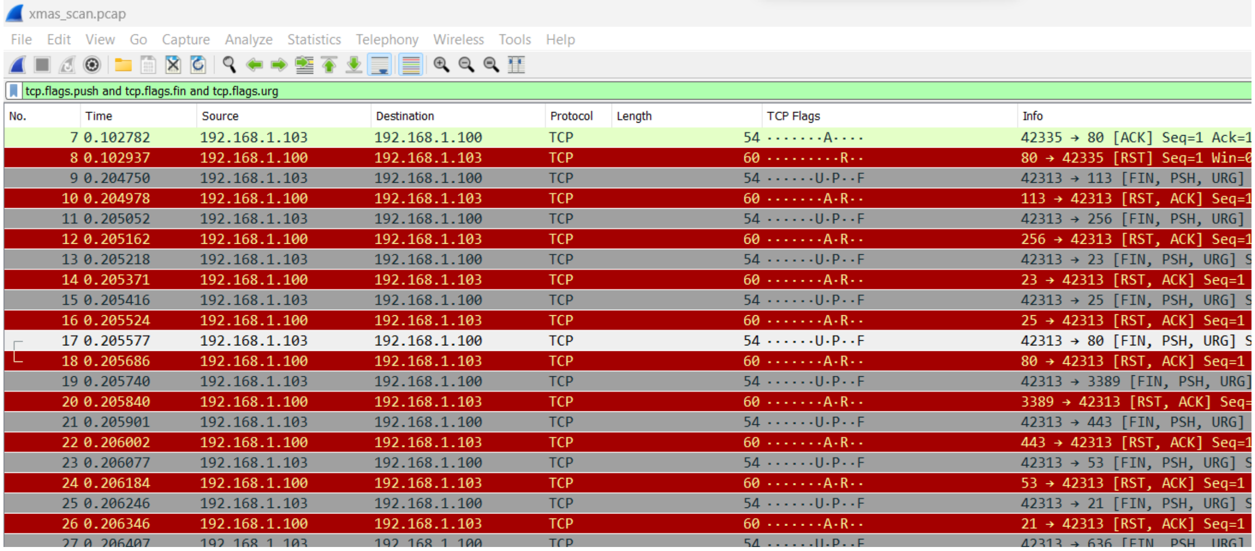
Info
1029 → 33468 [SYN, ACK] S
1025 → 33683 [SYN, ACK] S
139 → 33716 [SYN, ACK] Se
135 → 33809 [SYN, ACK] Se
3389 → 34195 [SYN, ACK] S
21 → 34647 [SYN, ACK] Se

Xmas_scan.pcp:

The xmas_scan file includes a network capture that observed one node on a network in the 192.168.1.xxx domain scanning other nodes using a Christmas scan. Answer the following questions. For a Christmas scan, the port scanner sends TCP packets with the FIN, PSH, and URG flags asserted. Write a filter to display only the aforementioned packets from the scanner. Use the tcp.flags filter category for this rule.

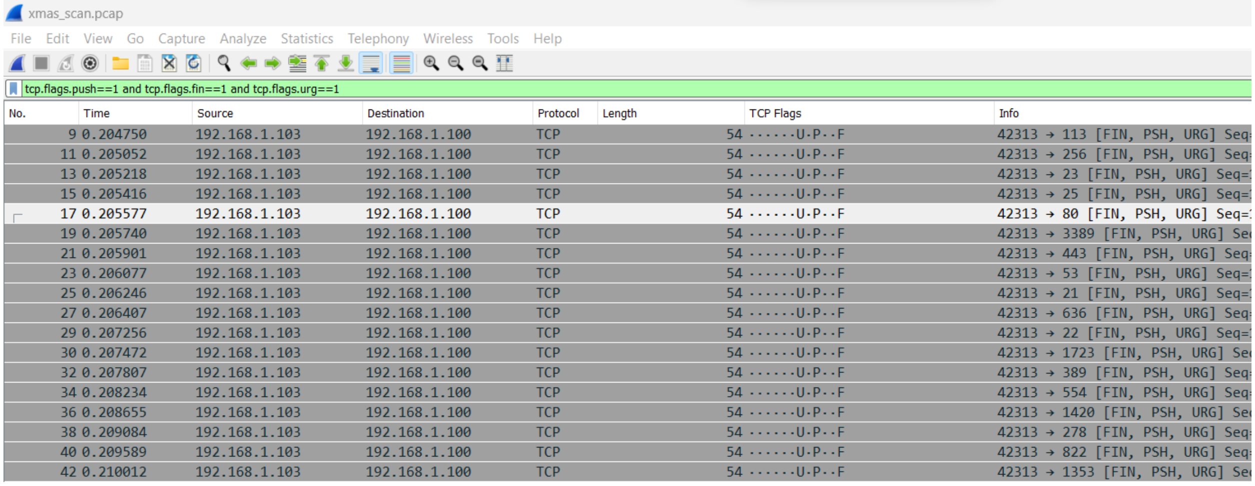
a. Try this filter "tcp.flags.push and tcp.flags.fin and tcp.flags.urg." Does this work? Why or why not?

- The command doesn’t work, all TCP packets are displayed because each packet has psh, fin and urg bits. The second screenshot shows a better filter expression, where those specific flag bits are set to 1.


xmas_scan.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp.flags.push and tcp.flags.fin and tcp.flags.urg

No.	Time	Source	Destination	Protocol	Length	TCP Flags	Info
7	0.102782	192.168.1.103	192.168.1.100	TCP	54	·······A····	42335 → 80 [ACK] Seq=1 Ack=1
8	0.102937	192.168.1.100	192.168.1.103	TCP	60	·········R··	80 → 42335 [RST] Seq=1 Win=0
9	0.204750	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 113 [FIN, PSH, URG]
10	0.204978	192.168.1.100	192.168.1.103	TCP	60	·······A·R··	113 → 42313 [RST, ACK] Seq=1
11	0.205052	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 256 [FIN, PSH, URG]
12	0.205162	192.168.1.100	192.168.1.103	TCP	60	·······A·R··	256 → 42313 [RST, ACK] Seq=1
13	0.205218	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 23 [FIN, PSH, URG] S
14	0.205371	192.168.1.100	192.168.1.103	TCP	60	·······A·R··	23 → 42313 [RST, ACK] Seq=1
15	0.205416	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 25 [FIN, PSH, URG] S
16	0.205524	192.168.1.100	192.168.1.103	TCP	60	·······A·R··	25 → 42313 [RST, ACK] Seq=1
17	0.205577	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 80 [FIN, PSH, URG] S
18	0.205686	192.168.1.100	192.168.1.103	TCP	60	·······A·R··	80 → 42313 [RST, ACK] Seq=1
19	0.205740	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 3389 [FIN, PSH, URG]
20	0.205840	192.168.1.100	192.168.1.103	TCP	60	·······A·R··	3389 → 42313 [RST, ACK] Seq=
21	0.205901	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 443 [FIN, PSH, URG]
22	0.206002	192.168.1.100	192.168.1.103	TCP	60	·······A·R··	443 → 42313 [RST, ACK] Seq=1
23	0.206077	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 53 [FIN, PSH, URG] S
24	0.206184	192.168.1.100	192.168.1.103	TCP	60	·······A·R··	53 → 42313 [RST, ACK] Seq=1
25	0.206246	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 21 [FIN, PSH, URG] S
26	0.206346	192.168.1.100	192.168.1.103	TCP	60	·······A·R··	21 → 42313 [RST, ACK] Seq=1
27	0.206407	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 636 [FIN, PSH, URG]

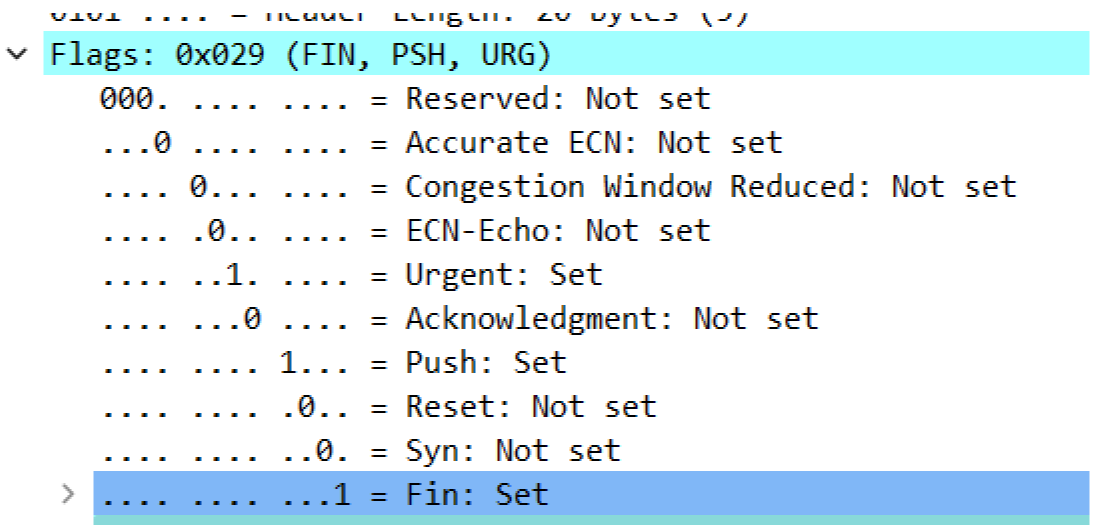

xmas_scan.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp.flags.push==1 and tcp.flags.fin==1 and tcp.flags.urg==1

No.	Time	Source	Destination	Protocol	Length	TCP Flags	Info
9	0.204750	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 113 [FIN, PSH, URG] Seq
11	0.205052	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 256 [FIN, PSH, URG] Seq
13	0.205218	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 23 [FIN, PSH, URG] Seq=
15	0.205416	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 25 [FIN, PSH, URG] Seq=
17	0.205577	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 80 [FIN, PSH, URG] Seq=
19	0.205740	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 3389 [FIN, PSH, URG] Se
21	0.205901	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 443 [FIN, PSH, URG] Seq
23	0.206077	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 53 [FIN, PSH, URG] Seq=
25	0.206246	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 21 [FIN, PSH, URG] Seq=
27	0.206407	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 636 [FIN, PSH, URG] Seq
29	0.207256	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 22 [FIN, PSH, URG] Seq=
30	0.207472	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 1723 [FIN, PSH, URG] Se
32	0.207807	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 389 [FIN, PSH, URG] Seq
34	0.208234	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 554 [FIN, PSH, URG] Seq
36	0.208655	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 1420 [FIN, PSH, URG] Se
38	0.209084	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 278 [FIN, PSH, URG] Seq
40	0.209589	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 822 [FIN, PSH, URG] Seq
42	0.210012	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 1353 [FIN, PSH, URG] Se

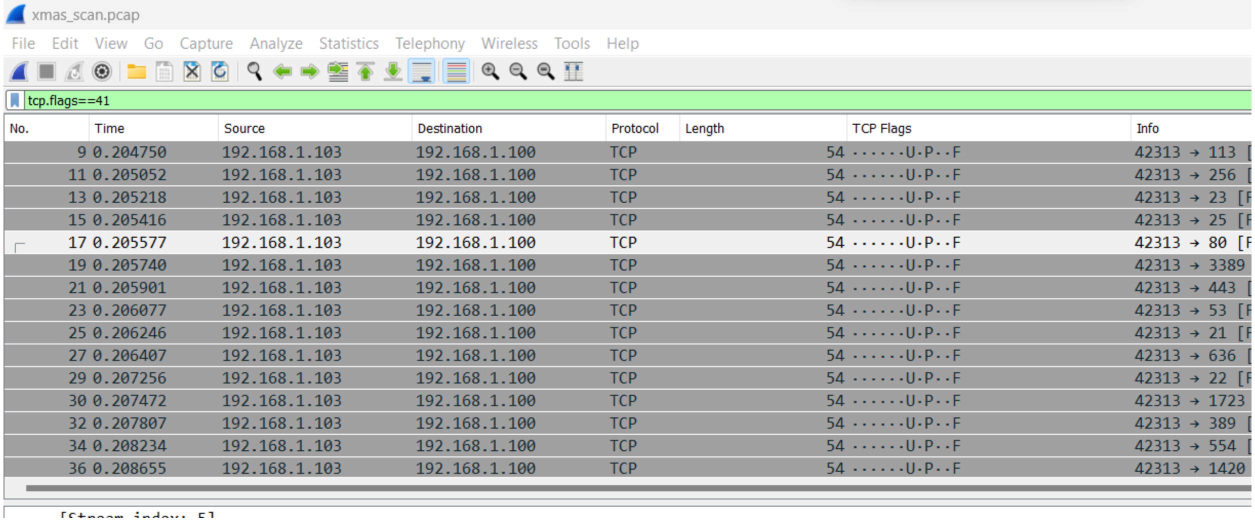
b. **Calculate an unsigned integer that equals the expected value of the flag bytes in the TCP header when the FIN, PSH, and URG flags asserted. Tip: Type out the 8 flag bits in order in binary and then convert to decimal on a calculator. Try the filter tcp.flags == <your unsigned integer>. Does this work? Why or why not? Include this Wireshark filter in your answer.**

- Frame 17 Flag details:



```
0101 .... = Header Length: 20 bytes (5)
Flags: 0x029 (FIN, PSH, URG)
    000. .... .... = Reserved: Not set
    ...0 .... .... = Accurate ECN: Not set
    .... 0... .... = Congestion Window Reduced: Not set
    .... .0.. .... = ECN-Echo: Not set
    .... ..1. .... = Urgent: Set
    .... ...0 .... = Acknowledgment: Not set
    .... .... 1... = Push: Set
    .... .... .0.. = Reset: Not set
    .... .... ..0. = Syn: Not set
    .... .... ...1 = Fin: Set
```

From the above screenshot, the hexadecimal value is 0x029, which is equivalent to 0010_1001 (we can also determine that binary equivalent by observing the flag bit values). The integer equivalent is 41, so I typed the expression tcp.flags==41 in the filter field, as shown in the screenshot below, that command worked, because we have specified in the filter expression that we want to see the packet with flag bits set such that that all 8 bits add up to a binary value equivalent to unsigned decimal 41.

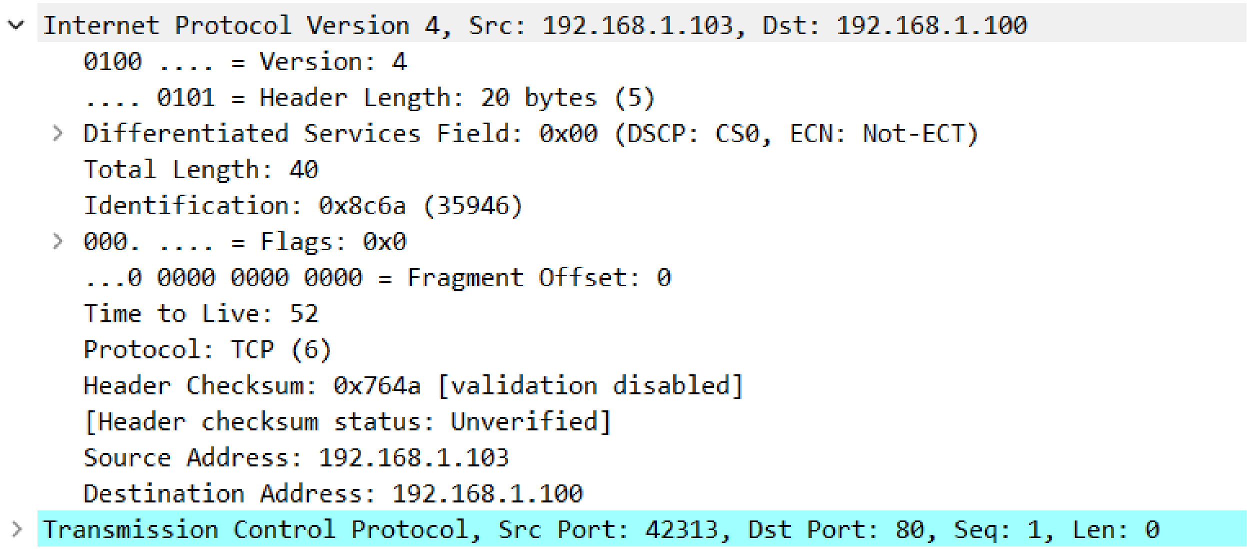


tcp.flags==41

No.	Time	Source	Destination	Protocol	Length	TCP Flags	Info
9	0.204750	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 113 [
11	0.205052	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 256 [
13	0.205218	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 23 [F
15	0.205416	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 25 [F
17	0.205577	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 80 [F
19	0.205740	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 3389
21	0.205901	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 443 [
23	0.206077	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 53 [F
25	0.206246	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 21 [F
27	0.206407	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 636 [
29	0.207256	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 22 [F
30	0.207472	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 1723
32	0.207807	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 389 [
34	0.208234	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 554 [
36	0.208655	192.168.1.103	192.168.1.100	TCP	54	······U·P··F	42313 → 1420

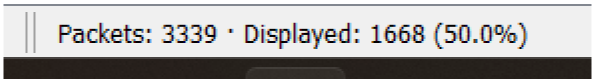
```
[Stream index: 5]
```

c. What is the IP address of the scanner? 192.168.1.103**.**



```
⌄ Internet Protocol Version 4, Src: 192.168.1.103, Dst: 192.168.1.100
     0100 .... = Version: 4
     .... 0101 = Header Length: 20 bytes (5)
   › Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
     Total Length: 40
     Identification: 0x8c6a (35946)
   › 000. .... = Flags: 0x0
     ...0 0000 0000 0000 = Fragment Offset: 0
     Time to Live: 52
     Protocol: TCP (6)
     Header Checksum: 0x764a [validation disabled]
     [Header checksum status: Unverified]
     Source Address: 192.168.1.103
     Destination Address: 192.168.1.100
› Transmission Control Protocol, Src Port: 42313, Dst Port: 80, Seq: 1, Len: 0
```

d. **How many packets have the FIN, PSH, and URG flags asserted?** With a filter expression of "tcp.flags.push==1 and tcp.flags.fin==1 and tcp.flags.urg==1" set, there are 1,668 packets displayed.

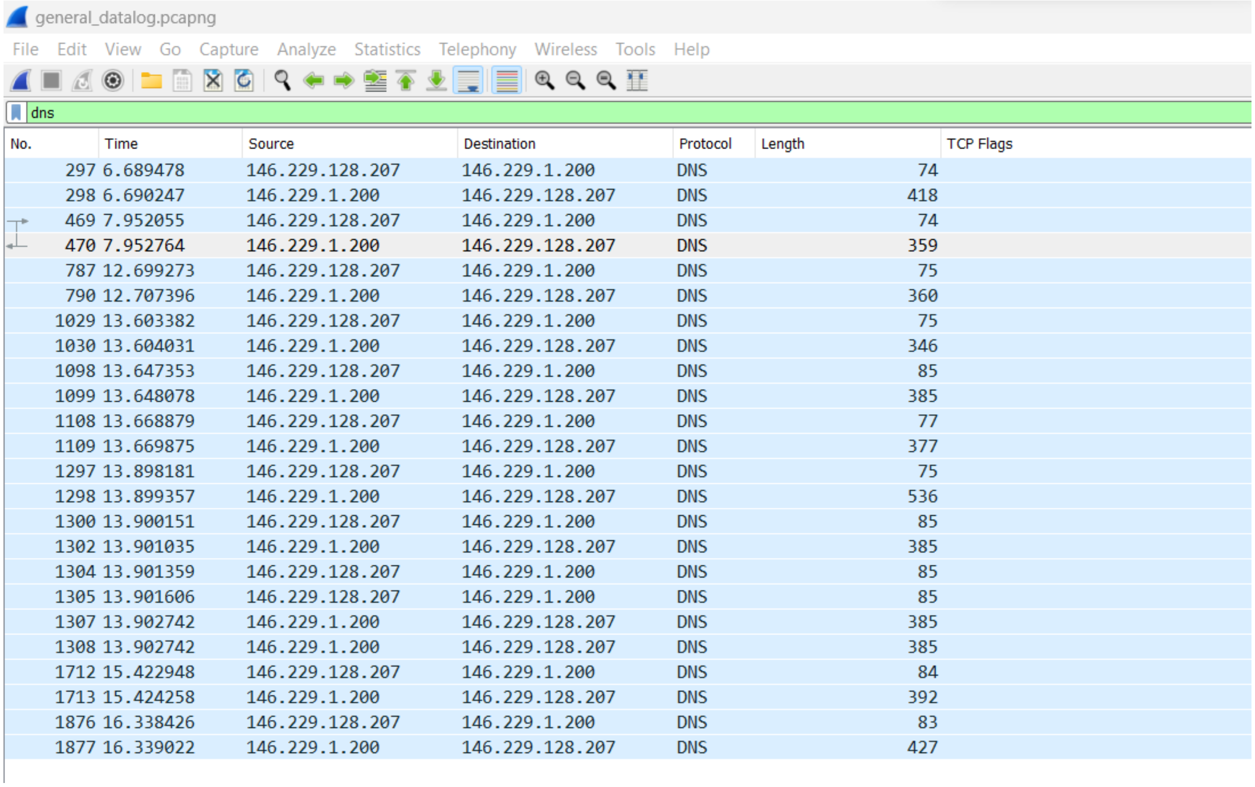


```
Packets: 3339 · Displayed: 1668 (50.0%)
```

General_datalog.pcapng:

The general_datalog file includes a network capture that observed general traffic on a computer. Answer the following questions. Write a filter to display all DNS packets. For most (or all protocols) typing the acronym for the protocol in the filter box will provide a filter to display only those packets.

a. **Write a filter to display all DNS packets. Include this Wireshark filter in your answer.**
   - Expression used: dns

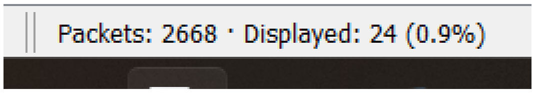

general_datalog.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

dns

No.	Time	Source	Destination	Protocol	Length	TCP Flags
297	6.689478	146.229.128.207	146.229.1.200	DNS	74	
298	6.690247	146.229.1.200	146.229.128.207	DNS	418	
469	7.952055	146.229.128.207	146.229.1.200	DNS	74	
470	7.952764	146.229.1.200	146.229.128.207	DNS	359	
787	12.699273	146.229.128.207	146.229.1.200	DNS	75	
790	12.707396	146.229.1.200	146.229.128.207	DNS	360	
1029	13.603382	146.229.128.207	146.229.1.200	DNS	75	
1030	13.604031	146.229.1.200	146.229.128.207	DNS	346	
1098	13.647353	146.229.128.207	146.229.1.200	DNS	85	
1099	13.648078	146.229.1.200	146.229.128.207	DNS	385	
1108	13.668879	146.229.128.207	146.229.1.200	DNS	77	
1109	13.669875	146.229.1.200	146.229.128.207	DNS	377	
1297	13.898181	146.229.128.207	146.229.1.200	DNS	75	
1298	13.899357	146.229.1.200	146.229.128.207	DNS	536	
1300	13.900151	146.229.128.207	146.229.1.200	DNS	85	
1302	13.901035	146.229.1.200	146.229.128.207	DNS	385	
1304	13.901359	146.229.128.207	146.229.1.200	DNS	85	
1305	13.901606	146.229.128.207	146.229.1.200	DNS	85	
1307	13.902742	146.229.1.200	146.229.128.207	DNS	385	
1308	13.902742	146.229.1.200	146.229.128.207	DNS	385	
1712	15.422948	146.229.128.207	146.229.1.200	DNS	84	
1713	15.424258	146.229.1.200	146.229.128.207	DNS	392	
1876	16.338426	146.229.128.207	146.229.1.200	DNS	83	
1877	16.339022	146.229.1.200	146.229.128.207	DNS	427	

b. **How many DNS packets are present?**

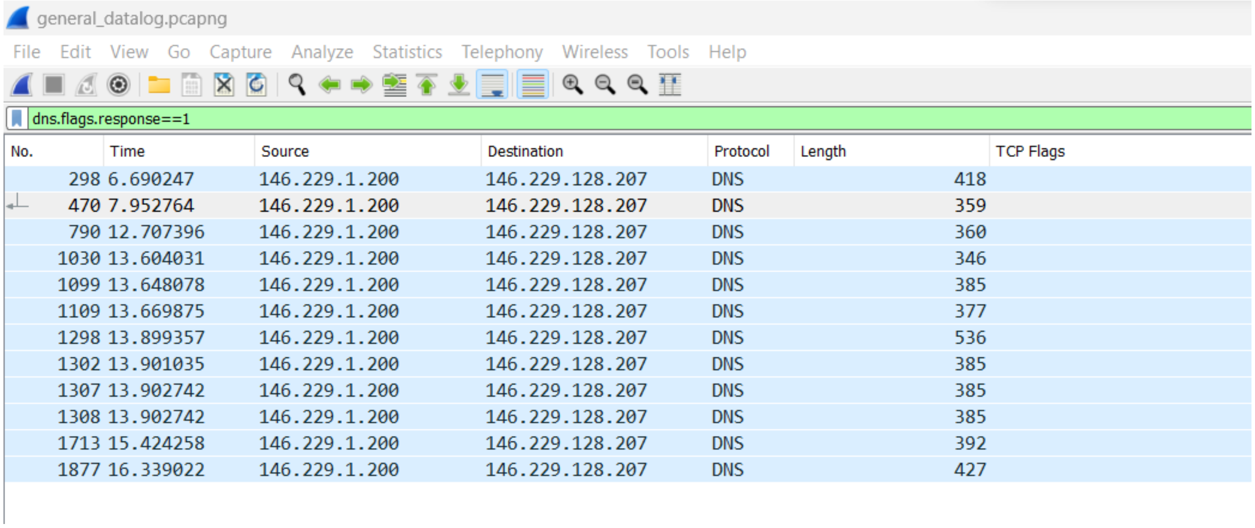
- **There are 24 DNS packets present.**



Packets: 2668 · Displayed: 24 (0.9%)

c. **Write a filter to display only DNS responses. Include this Wireshark filter in your answer.**

- **Expression used: dns.flags.response==1**

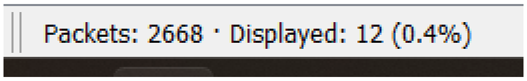

general_datalog.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

dns.flags.response==1

No.	Time	Source	Destination	Protocol	Length	TCP Flags
298	6.690247	146.229.1.200	146.229.128.207	DNS	418	
470	7.952764	146.229.1.200	146.229.128.207	DNS	359	
790	12.707396	146.229.1.200	146.229.128.207	DNS	360	
1030	13.604031	146.229.1.200	146.229.128.207	DNS	346	
1099	13.648078	146.229.1.200	146.229.128.207	DNS	385	
1109	13.669875	146.229.1.200	146.229.128.207	DNS	377	
1298	13.899357	146.229.1.200	146.229.128.207	DNS	536	
1302	13.901035	146.229.1.200	146.229.128.207	DNS	385	
1307	13.902742	146.229.1.200	146.229.128.207	DNS	385	
1308	13.902742	146.229.1.200	146.229.128.207	DNS	385	
1713	15.424258	146.229.1.200	146.229.128.207	DNS	392	
1877	16.339022	146.229.1.200	146.229.128.207	DNS	427	

d. **How many responses are present?**

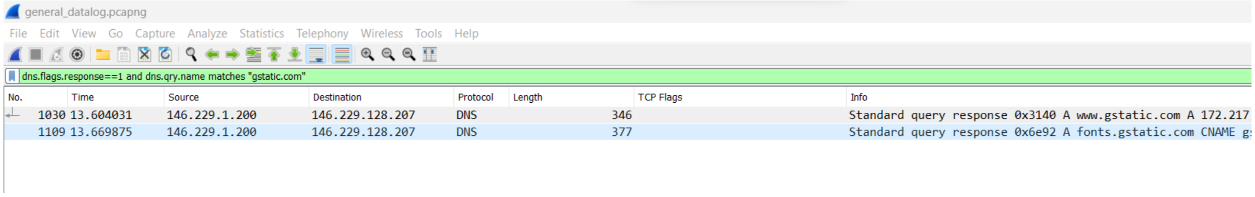
- 12 Responses are present.



Packets: 2668 · Displayed: 12 (0.4%)

e. **Write a filter to display only the DNS response for the gstatic.com query. Include this Wireshark filter in your answer.**

- Expression: dns.flags.response==1 and dns.qry.name matches “gstatic.com”

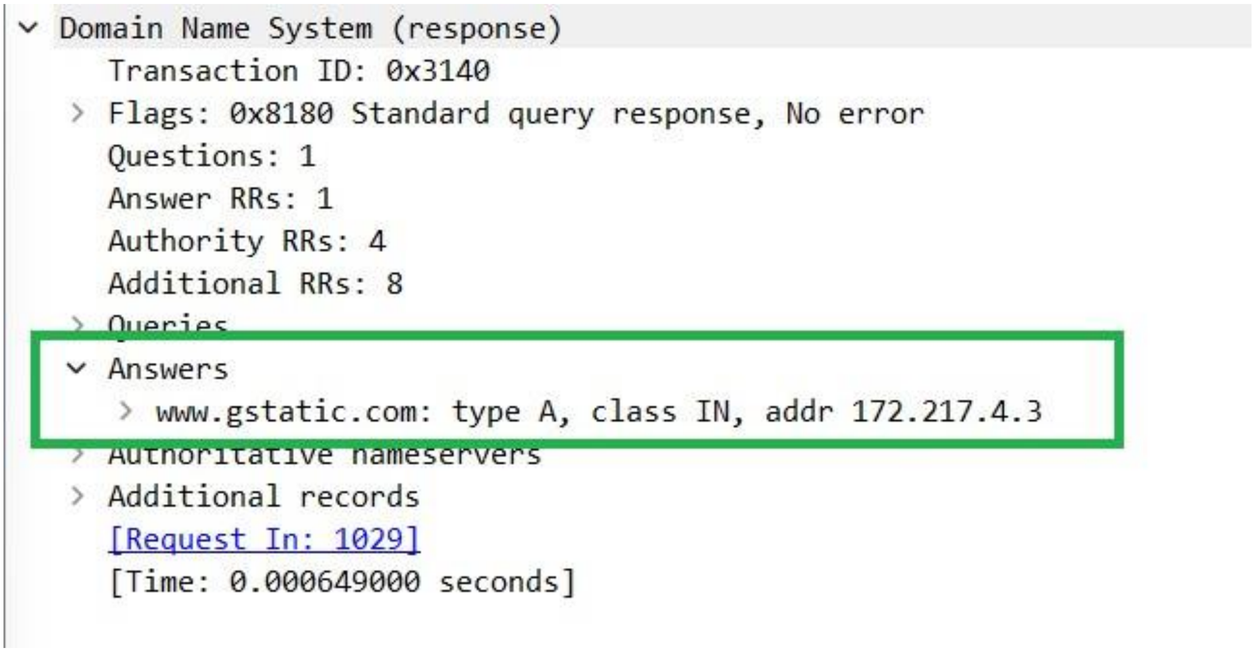


dns.flags.response==1 and dns.qry.name matches "gstatic.com"

No.	Time	Source	Destination	Protocol	Length	TCP Flags	Info
1030	13.604031	146.229.1.200	146.229.128.207	DNS	346		Standard query response 0x3140 A www.gstatic.com A 172.217
1109	13.669875	146.229.1.200	146.229.128.207	DNS	377		Standard query response 0x6e92 A fonts.gstatic.com CNAME g:

f. **What is the IP address of the gstatic.com server? You need to look in the packet manually to find the IP address. Look for the “Answers” field.**

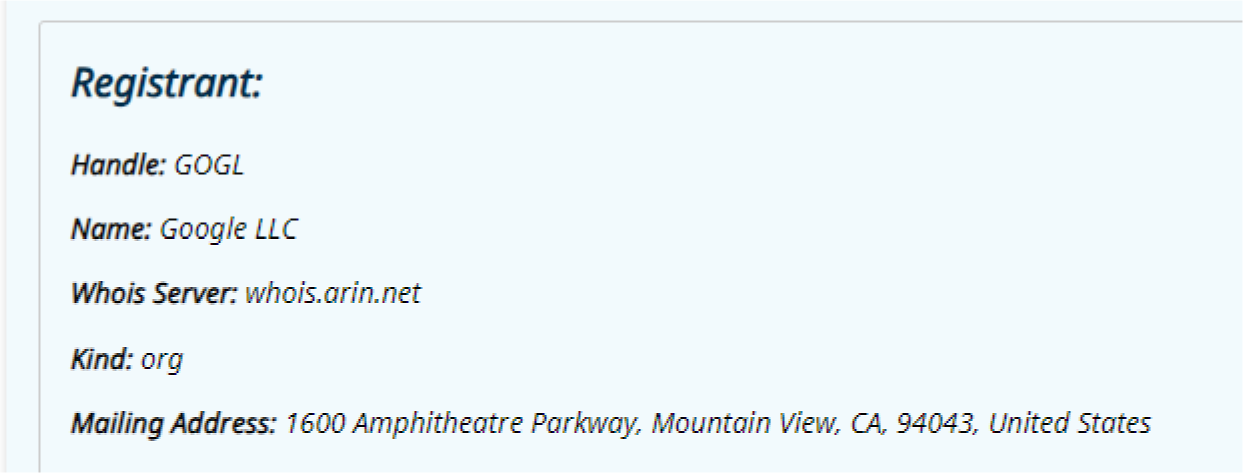
- The IP address for gstatic.com is 172.217.4.3



```
v Domain Name System (response)
    Transaction ID: 0x3140
  > Flags: 0x8180 Standard query response, No error
    Questions: 1
    Answer RRs: 1
    Authority RRs: 4
    Additional RRs: 8
  > Queries
  v Answers
      > www.gstatic.com: type A, class IN, addr 172.217.4.3
  > Authoritative nameservers
  > Additional records
    [Request In: 1029]
    [Time: 0.000649000 seconds]
```

g. **Use a Who IS registry on the internet to find the street address associated with the above IP address.**

- Lookup website used: https://lookup.icann.org/en/lookup
- Street Address: 1600 Amphitheatre Parkway, Mountain View, CA, 94043, United States.



Registrant:

Handle: GOGL

Name: Google LLC

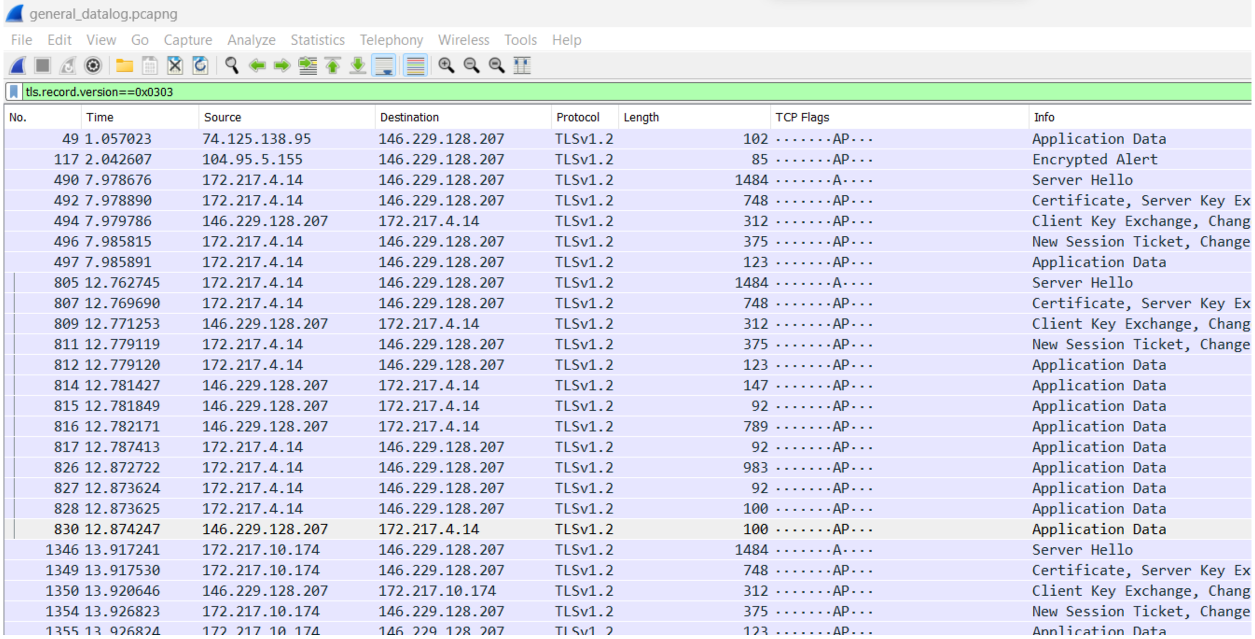
Whois Server: whois.arin.net

Kind: org

Mailing Address: 1600 Amphitheatre Parkway, Mountain View, CA, 94043, United States

The ssl filter type can display many versions of SSL and TLS packets.

a. **Write a filter to display only TLS v1.2 packets. Use Google to figure this one out. Include this Wireshark filter in your answer.**
   - **Expression: tls.record.version==0x0303**

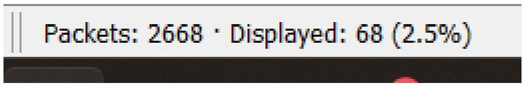


general_datalog.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tls.record.version==0x0303

No.	Time	Source	Destination	Protocol	Length	TCP Flags	Info
49	1.057023	74.125.138.95	146.229.128.207	TLSv1.2	102	·······AP···	Application Data
117	2.042607	104.95.5.155	146.229.128.207	TLSv1.2	85	·······AP···	Encrypted Alert
490	7.978676	172.217.4.14	146.229.128.207	TLSv1.2	1484	·······A····	Server Hello
492	7.978890	172.217.4.14	146.229.128.207	TLSv1.2	748	·······AP···	Certificate, Server Key Ex
494	7.979786	146.229.128.207	172.217.4.14	TLSv1.2	312	·······AP···	Client Key Exchange, Chang
496	7.985815	172.217.4.14	146.229.128.207	TLSv1.2	375	·······AP···	New Session Ticket, Change
497	7.985891	172.217.4.14	146.229.128.207	TLSv1.2	123	·······AP···	Application Data
805	12.762745	172.217.4.14	146.229.128.207	TLSv1.2	1484	·······A····	Server Hello
807	12.769690	172.217.4.14	146.229.128.207	TLSv1.2	748	·······AP···	Certificate, Server Key Ex
809	12.771253	146.229.128.207	172.217.4.14	TLSv1.2	312	·······AP···	Client Key Exchange, Chang
811	12.779119	172.217.4.14	146.229.128.207	TLSv1.2	375	·······AP···	New Session Ticket, Change
812	12.779120	172.217.4.14	146.229.128.207	TLSv1.2	123	·······AP···	Application Data
814	12.781427	146.229.128.207	172.217.4.14	TLSv1.2	147	·······AP···	Application Data
815	12.781849	146.229.128.207	172.217.4.14	TLSv1.2	92	·······AP···	Application Data
816	12.782171	146.229.128.207	172.217.4.14	TLSv1.2	789	·······AP···	Application Data
817	12.787413	172.217.4.14	146.229.128.207	TLSv1.2	92	·······AP···	Application Data
826	12.872722	172.217.4.14	146.229.128.207	TLSv1.2	983	·······AP···	Application Data
827	12.873624	172.217.4.14	146.229.128.207	TLSv1.2	92	·······AP···	Application Data
828	12.873625	172.217.4.14	146.229.128.207	TLSv1.2	100	·······AP···	Application Data
830	12.874247	146.229.128.207	172.217.4.14	TLSv1.2	100	·······AP···	Application Data
1346	13.917241	172.217.10.174	146.229.128.207	TLSv1.2	1484	·······A····	Server Hello
1349	13.917530	172.217.10.174	146.229.128.207	TLSv1.2	748	·······AP···	Certificate, Server Key Ex
1350	13.920646	146.229.128.207	172.217.10.174	TLSv1.2	312	·······AP···	Client Key Exchange, Chang
1354	13.926823	172.217.10.174	146.229.128.207	TLSv1.2	375	·······AP···	New Session Ticket, Change
1355	13.926824	172.217.10.174	146.229.128.207	TLSv1.2	123	·······AP···	Application Data

b. **How many TLSv1.2 packets are in the general_datalog pcap file?**
   - **There are 68 TLSv1.2 packets in this file.**



Packets: 2668 · Displayed: 68 (2.5%)

PART 2:

I opened the Security-Desk VM and analyzed each pcap file, starting with 10.7.

10.7.pcap:

- **After opening the file, I clicked on Analyze -->Expert Information. Next, I clicked Statistics-->Protocol Hierarchy, Statistics-->Conversations and Statistics-->Endpoints. I noted that IP 172.16.30.109 was sending mass ARP requests to a wide range of IPs across a network. The first ARP request was at packet 41. IP 172.16.30.109 initiated the TCP-3-way handshake with 172.16.10.7 at packet 3088 by sending a SYN connection establish request over port 80. I scrolled to the very bottom of the packet capture file and last saw the suspect IP in packet #6670. The rogue IP is 172.16.30.109 and the range is packets 41-6670.**

20.0.pcap:

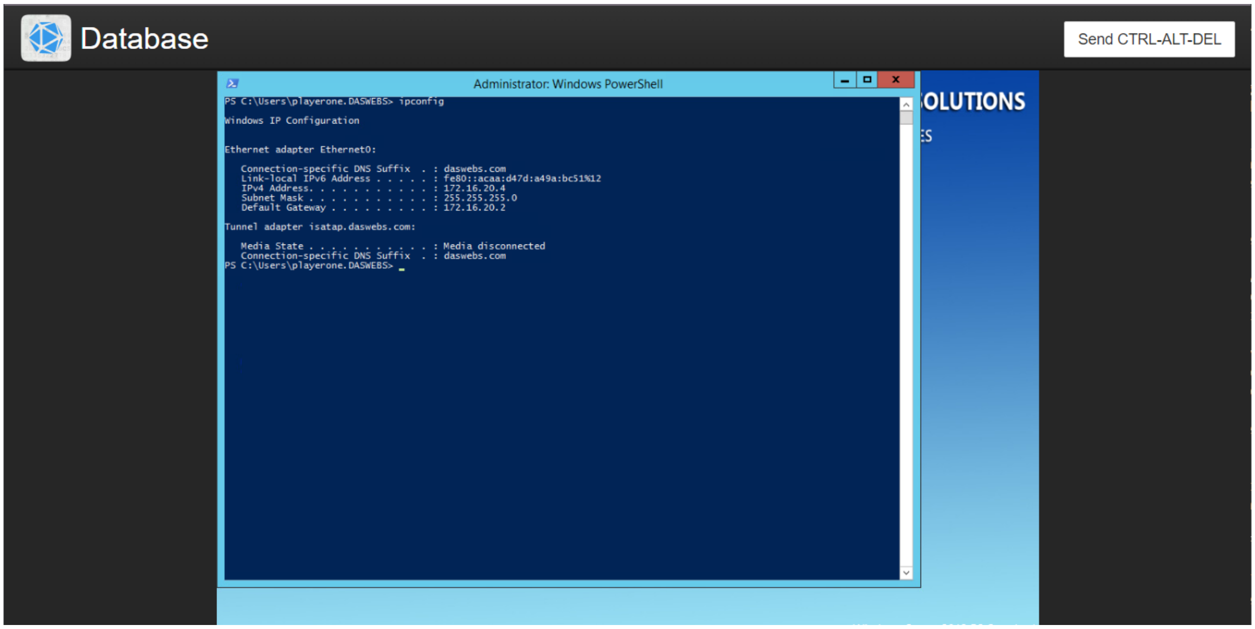
- **For this file the Expert Information details showed more than 2000 SYN connection requests and more than 2,000 RST/ACK connection reset packets. I noticed that IP 172.16.30.109 initiated SYN connection request with 172.16.20.2. Then 172.16.20.4 replied with SYN/ACK. IP 172.16.30.109 proceeded to spam 172.16.20.2 with SYN requests and 172.16.20.4 kept sending RST/ACK. I flagged IP 172.16.30.109 as malicious, with a packet range of 77-8801.**

30.21.pcap:

- **Using the Expert Information menu. I noticed that IP 172.16.30.109 was sending a lot of ARP requests. Then I noticed 172.16.30.109 initiated communication with 172.6.30.21 over SSH and at packet 8186, the connection between them was terminated. I flagged IP 172.16.30.109 as malicious, packet range of 963-8186.**

Canvas Questions.

IP address for database server: 172.16.20.4

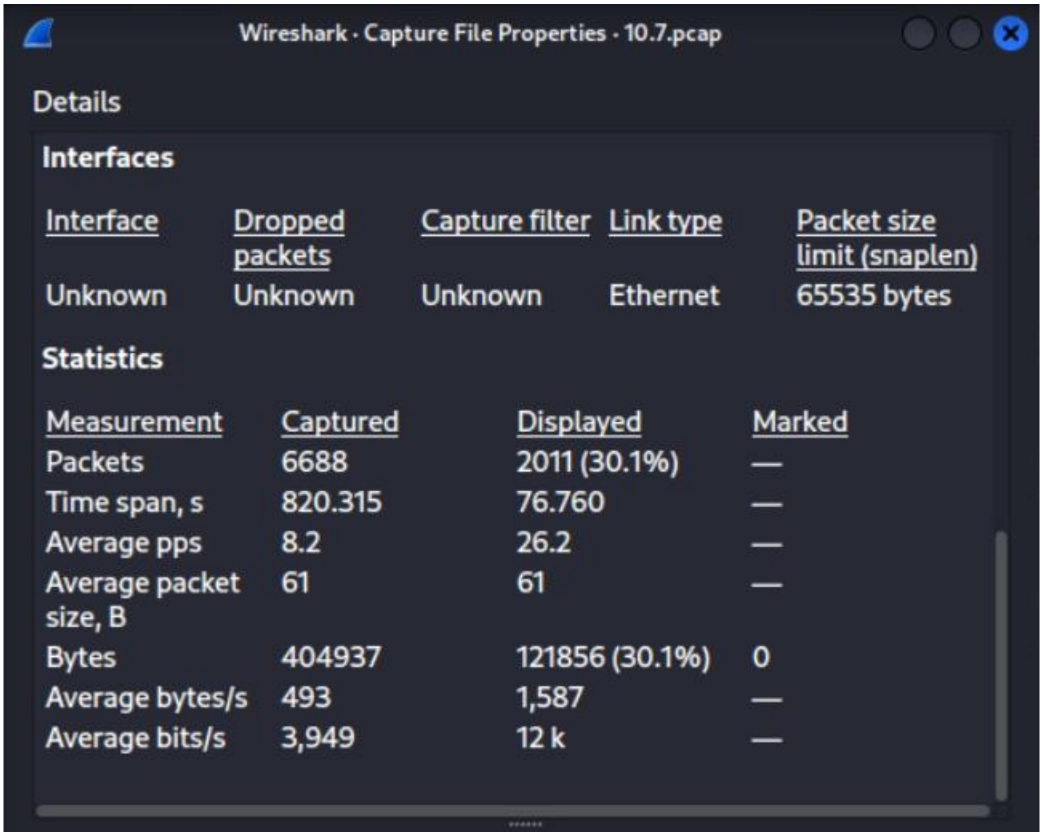


```
PS C:\Users\playerone.DASWEBS> ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . : daswebs.com
   Link-local IPv6 Address . . . . . : fe80::acaa:d47d:a49a:bc51%12
   IPv4 Address. . . . . . . . . . . : 172.16.20.4
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 172.16.20.2

Tunnel adapter isatap.daswebs.com:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . : daswebs.com
PS C:\Users\playerone.DASWEBS> _
```

2. For each pcap file analyzed, provide a 1 sentence description of each major action performed by the attacker.

- **10.7.pcap:**
  - **The attacker used IP 172.16.30.109 to initiate mass ARP requests and a TCP-3-way handshake with 172.16.10.7 on port 80, between packets 41 and 6670.**
- **20.0.pcap:**
  - **The attacker used IP 172.16.30.109 to spam-flood 172.16.20.2 and 172.16.20.4 with SYN requests in attempts to crash the server.**
- **30.21.pcap:**
  - **Once again, 172.16.30.109 connected with 172.6.30.21 at packet 7601, initiated communication over SSH, and completed connection at packet 8186.**

3. How many total packets were sent by the attacker?



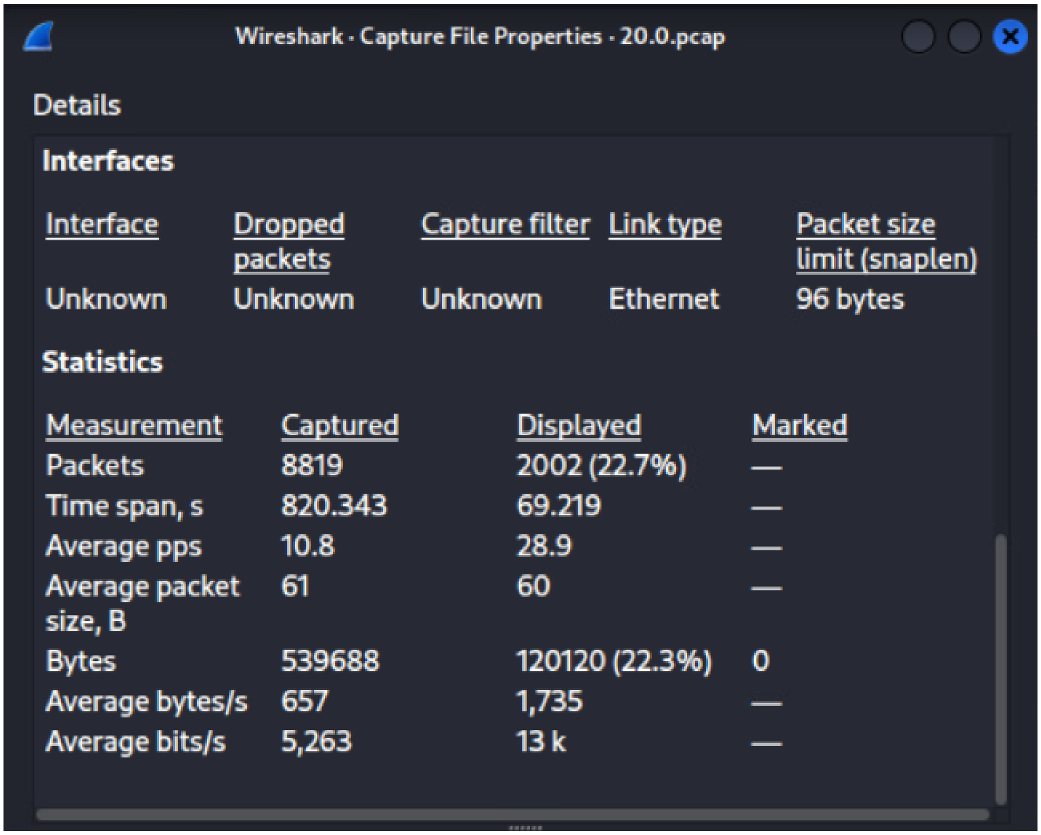
Details

Interfaces

Interface	Dropped packets	Capture filter	Link type	Packet size limit (snaplen)
Unknown	Unknown	Unknown	Ethernet	65535 bytes

Statistics

Measurement	Captured	Displayed	Marked
Packets	6688	2011 (30.1%)	—
Time span, s	820.315	76.760	—
Average pps	8.2	26.2	—
Average packet size, B	61	61	—
Bytes	404937	121856 (30.1%)	0
Average bytes/s	493	1,587	—
Average bits/s	3,949	12 k	—



Wireshark · Capture File Properties · 20.0.pcap

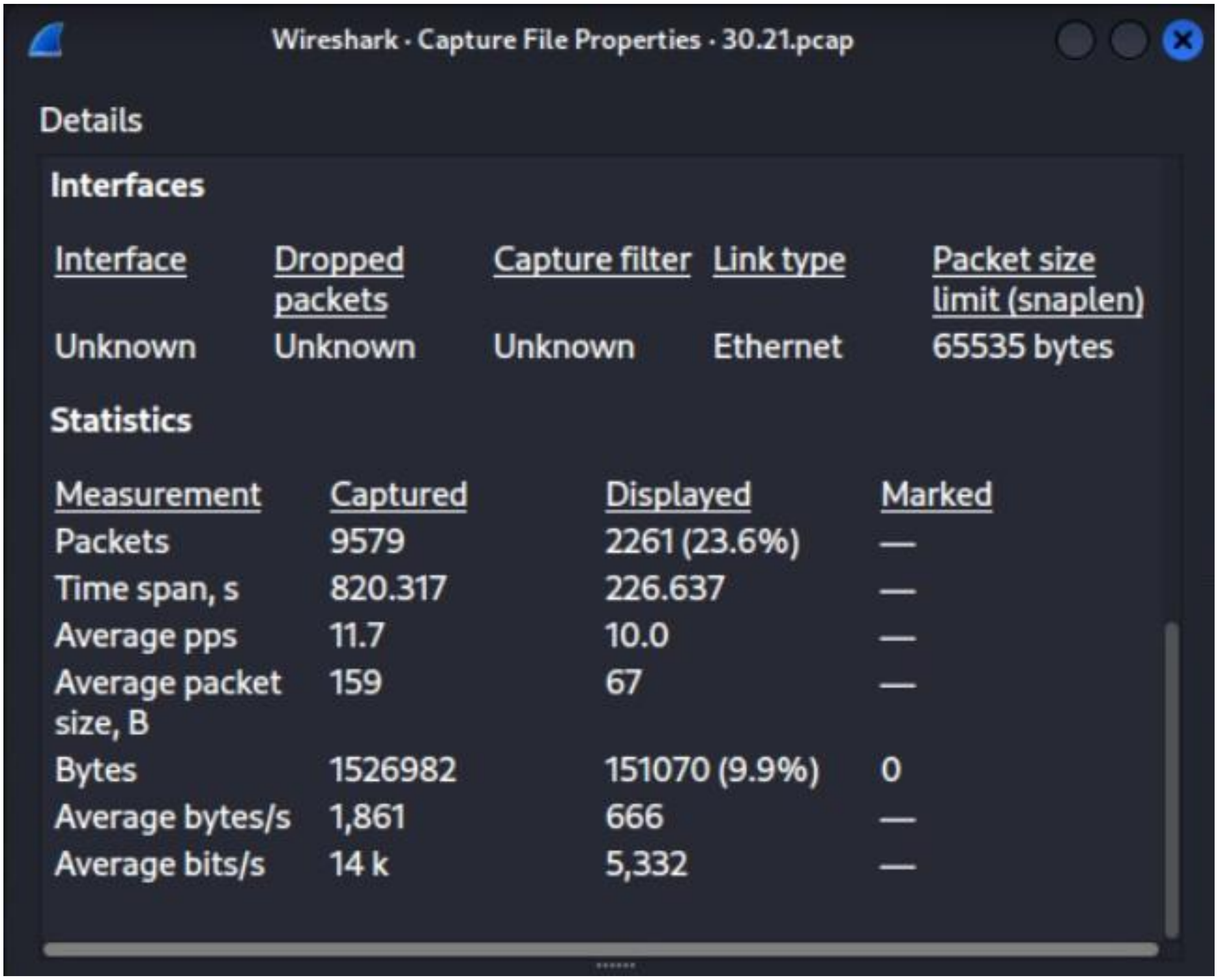
Details

Interfaces

Interface	Dropped packets	Capture filter	Link type	Packet size limit (snaplen)
Unknown	Unknown	Unknown	Ethernet	96 bytes

Statistics

Measurement	Captured	Displayed	Marked
Packets	8819	2002 (22.7%)	—
Time span, s	820.343	69.219	—
Average pps	10.8	28.9	—
Average packet size, B	61	60	—
Bytes	539688	120120 (22.3%)	0
Average bytes/s	657	1,735	—
Average bits/s	5,263	13 k	—



Wireshark · Capture File Properties · 30.21.pcap

Details

Interfaces

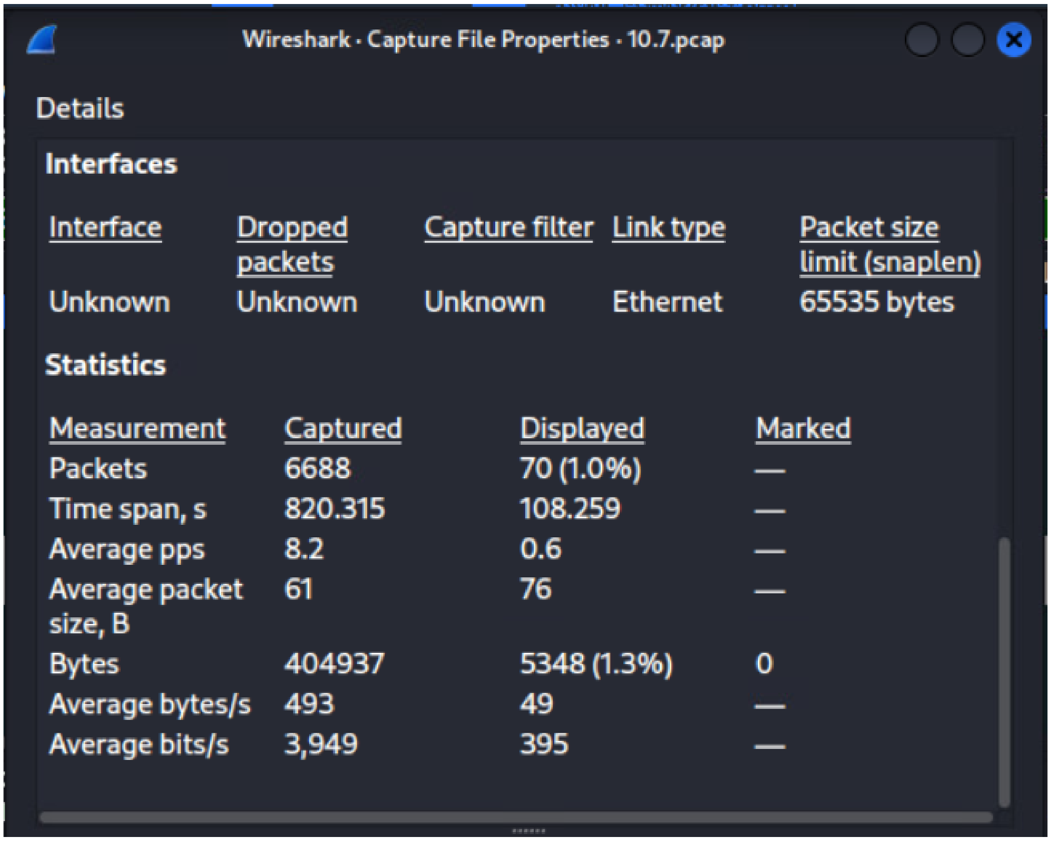
Interface	Dropped packets	Capture filter	Link type	Packet size limit (snaplen)
Unknown	Unknown	Unknown	Ethernet	65535 bytes

Statistics

Measurement	Captured	Displayed	Marked
Packets	9579	2261 (23.6%)	—
Time span, s	820.317	226.637	—
Average pps	11.7	10.0	—
Average packet size, B	159	67	—
Bytes	1526982	151070 (9.9%)	0
Average bytes/s	1,861	666	—
Average bits/s	14 k	5,332	—

Total packets = 2011 + 2002 + 2261 = 6,274.

4. How many total packets were sent to the attacker?



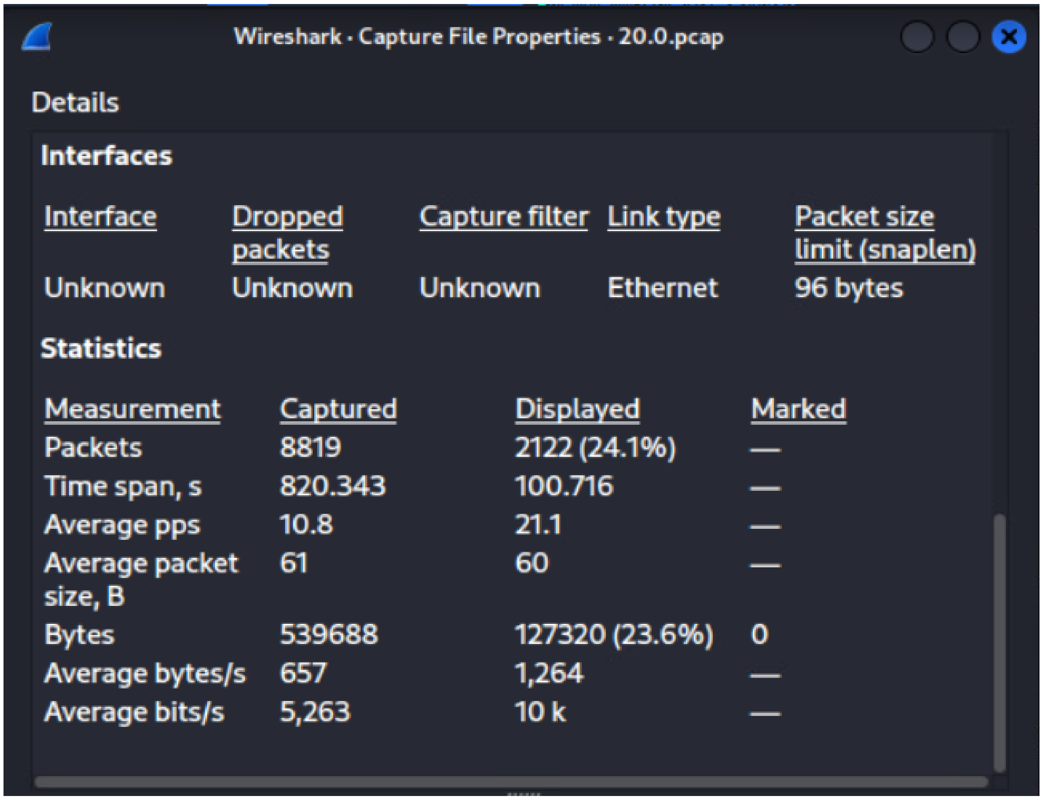
Details

Interfaces

Interface	Dropped packets	Capture filter	Link type	Packet size limit (snaplen)
Unknown	Unknown	Unknown	Ethernet	65535 bytes

Statistics

Measurement	Captured	Displayed	Marked
Packets	6688	70 (1.0%)	—
Time span, s	820.315	108.259	—
Average pps	8.2	0.6	—
Average packet size, B	61	76	—
Bytes	404937	5348 (1.3%)	0
Average bytes/s	493	49	—
Average bits/s	3,949	395	—



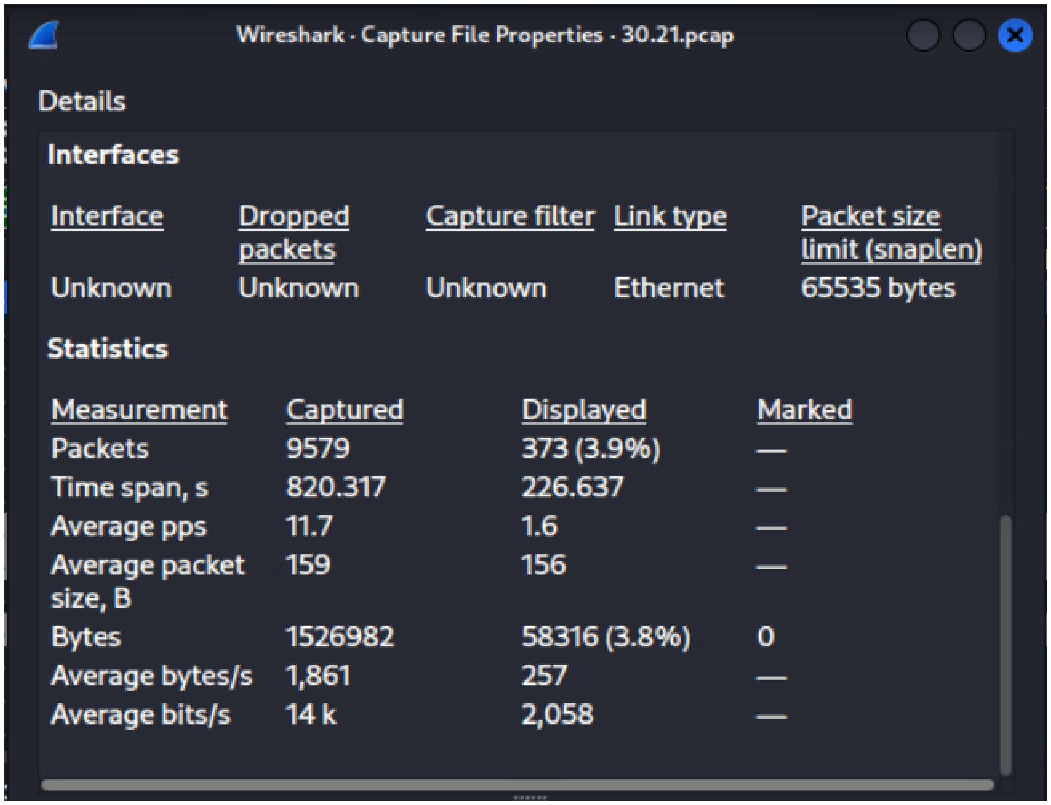
Details

Interfaces

Interface	Dropped packets	Capture filter	Link type	Packet size limit (snaplen)
Unknown	Unknown	Unknown	Ethernet	96 bytes

Statistics

Measurement	Captured	Displayed	Marked
Packets	8819	2122 (24.1%)	—
Time span, s	820.343	100.716	—
Average pps	10.8	21.1	—
Average packet size, B	61	60	—
Bytes	539688	127320 (23.6%)	0
Average bytes/s	657	1,264	—
Average bits/s	5,263	10 k	—



Wireshark · Capture File Properties · 30.21.pcap

Details

Interfaces

Interface	Dropped packets	Capture filter	Link type	Packet size limit (snaplen)
Unknown	Unknown	Unknown	Ethernet	65535 bytes

Statistics

Measurement	Captured	Displayed	Marked
Packets	9579	373 (3.9%)	—
Time span, s	820.317	226.637	—
Average pps	11.7	1.6	—
Average packet size, B	159	156	—
Bytes	1526982	58316 (3.8%)	0
Average bytes/s	1,861	257	—
Average bits/s	14 k	2,058	—

Total packets = 70 + 2,122 + 373 = 2,565.