

**VENNELA G**

**20BDS0146**

**INFORMATION  
SECURITY  
MANAGEMENT**

**LAB ASSIGNMENT 1**

**SLOT L37+L38**

## Analysis of Network using Wireshark:

**Q. Analyse the layered structure of network protocols using a web browsing example.**

**Examine the header structure of the PDUs at the data link, IP, transport, and application layers. In particular, how addresses and port numbers work together to enable end-to-end applications.**

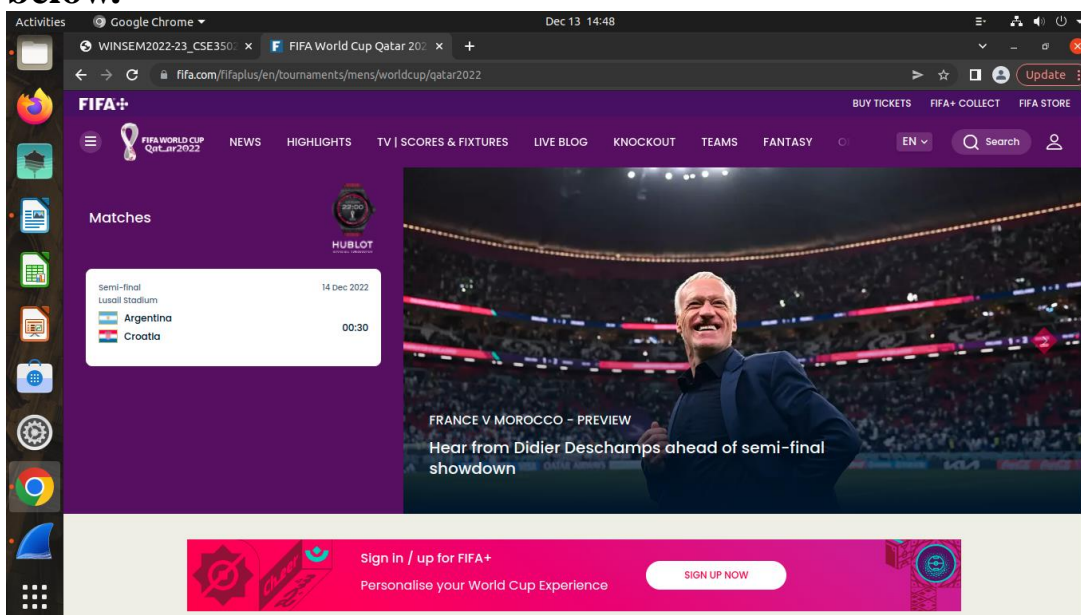
**Protocols to Examine:**

- Ethernet and IP addressing
- DNS Query and Response
- TCP three-way handshake, sequence, and ACK numbering
- HTTP GET and Response messages

## Protocol Analysis Questions

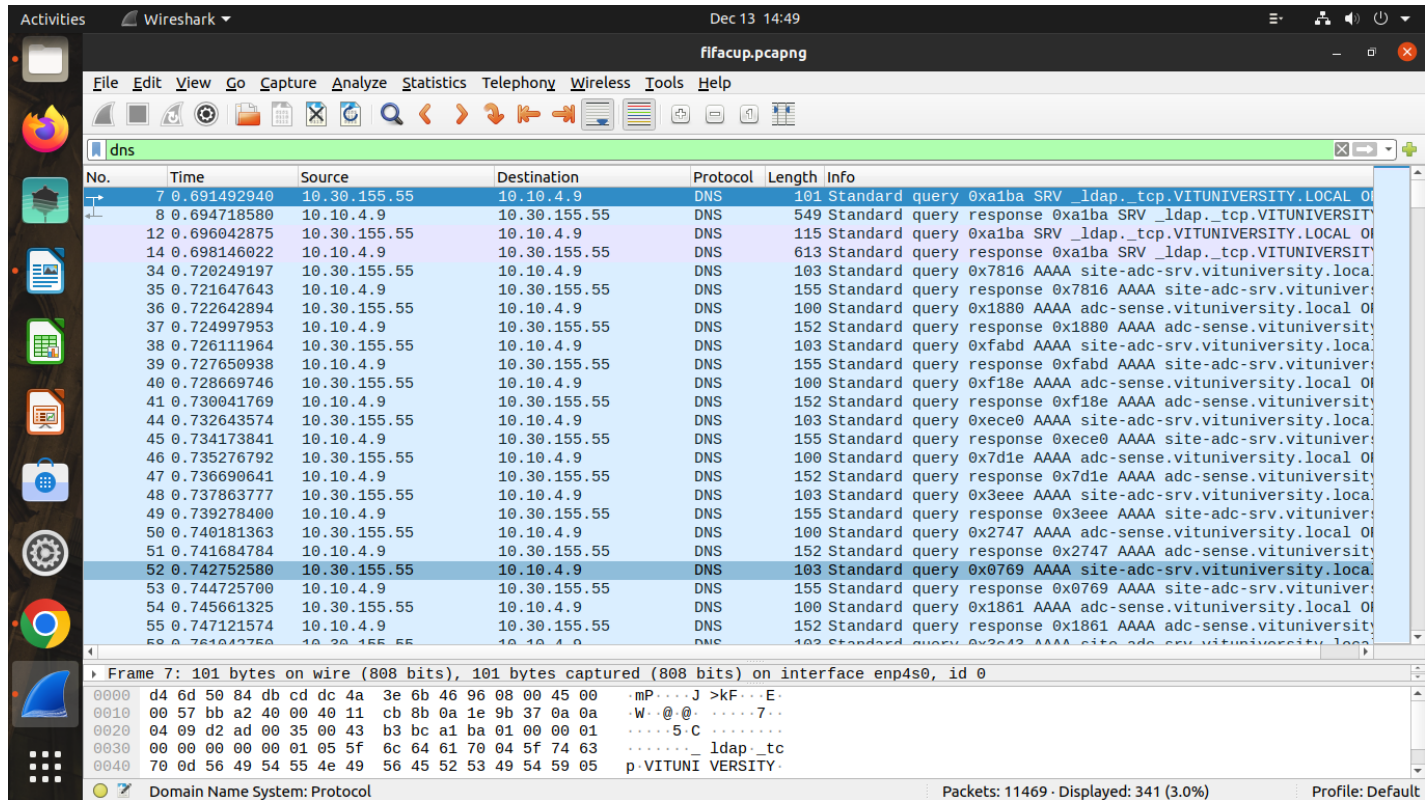
**To begin with this, we have captured the packets using Wireshark by visiting the website: [fifa.com](https://fifa.com) and the captured file was saved.**

**1. Protocols Capture: To ensure if we have captured DNS, TCP, and HTTP packets, we use the filter options from Wireshark as shown below.**



## a. DNS packets:

### *Using DNS filter*



The screenshot shows the Wireshark network protocol analyzer interface. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. The toolbar contains various icons for file operations, capture control, and analysis. The 'dns' filter is applied to the packet list. The packet list table shows the following data:

No.	Time	Source	Destination	Protocol	Length	Info
7	0.691492940	10.30.155.55	10.10.4.9	DNS	101	Standard query 0xa1ba SRV _ldap._tcp.VITUNIVERSITY.LOCAL
8	0.694718580	10.10.4.9	10.30.155.55	DNS	549	Standard query response 0xa1ba SRV _ldap._tcp.VITUNIVERSITY
12	0.696042875	10.30.155.55	10.10.4.9	DNS	115	Standard query 0xa1ba SRV _ldap._tcp.VITUNIVERSITY.LOCAL
14	0.698146022	10.10.4.9	10.30.155.55	DNS	613	Standard query response 0xa1ba SRV _ldap._tcp.VITUNIVERSITY
34	0.720249197	10.30.155.55	10.10.4.9	DNS	103	Standard query 0x7816 AAAA site-adc-srv.vituniversity.local
35	0.721647643	10.10.4.9	10.30.155.55	DNS	155	Standard query response 0x7816 AAAA site-adc-srv.vituniver
36	0.722642894	10.30.155.55	10.10.4.9	DNS	100	Standard query 0x1880 AAAA adc-sense.vituniversity.local
37	0.724997953	10.10.4.9	10.30.155.55	DNS	152	Standard query response 0x1880 AAAA adc-sense.vituniversity
38	0.726111964	10.30.155.55	10.10.4.9	DNS	103	Standard query 0xfabd AAAA site-adc-srv.vituniversity.local
39	0.727650938	10.10.4.9	10.30.155.55	DNS	155	Standard query response 0xfabd AAAA site-adc-srv.vituniver
40	0.728669746	10.30.155.55	10.10.4.9	DNS	100	Standard query 0xf18e AAAA adc-sense.vituniversity.local
41	0.730041769	10.10.4.9	10.30.155.55	DNS	152	Standard query response 0xf18e AAAA adc-sense.vituniversity
44	0.732643574	10.30.155.55	10.10.4.9	DNS	103	Standard query 0xece0 AAAA site-adc-srv.vituniversity.local
45	0.734173841	10.10.4.9	10.30.155.55	DNS	155	Standard query response 0xece0 AAAA site-adc-srv.vituniver
46	0.735276792	10.30.155.55	10.10.4.9	DNS	100	Standard query 0x7d1e AAAA adc-sense.vituniversity.local
47	0.736690641	10.10.4.9	10.30.155.55	DNS	152	Standard query response 0x7d1e AAAA adc-sense.vituniversity
48	0.737863777	10.30.155.55	10.10.4.9	DNS	103	Standard query 0x3eee AAAA site-adc-srv.vituniversity.local
49	0.739278400	10.10.4.9	10.30.155.55	DNS	155	Standard query response 0x3eee AAAA site-adc-srv.vituniver
50	0.740181363	10.30.155.55	10.10.4.9	DNS	100	Standard query 0x2747 AAAA adc-sense.vituniversity.local
51	0.741684784	10.10.4.9	10.30.155.55	DNS	152	Standard query response 0x2747 AAAA adc-sense.vituniversity
52	0.742752580	10.30.155.55	10.10.4.9	DNS	103	Standard query 0x0769 AAAA site-adc-srv.vituniversity.local
53	0.744725700	10.10.4.9	10.30.155.55	DNS	155	Standard query response 0x0769 AAAA site-adc-srv.vituniver
54	0.745661325	10.30.155.55	10.10.4.9	DNS	100	Standard query 0x1861 AAAA adc-sense.vituniversity.local
55	0.747121574	10.10.4.9	10.30.155.55	DNS	152	Standard query response 0x1861 AAAA adc-sense.vituniversity

The packet details pane shows the selected packet (No. 7) with the following structure:

- Frame 7: 101 bytes on wire (808 bits), 101 bytes captured (808 bits) on interface enp4s0, id 0
- 0000 d4 6d 50 84 db cd dc 4a 3e 6b 46 96 08 00 45 00 .mP....J>kF...E.
- 0010 00 57 bb a2 40 00 40 11 cb 8b 0a 1e 9b 37 0a 0a .W..@.@.....7..
- 0020 04 09 d2 ad 00 35 00 43 b3 bc a1 ba 01 00 00 01 .....5.C.....
- 0030 00 00 00 00 01 05 5f 6c 64 61 70 04 5f 74 63 .....\_ldap.\_tc
- 0040 70 0d 56 49 54 55 4e 49 56 45 52 53 49 54 59 05 p.VITUNI.VERSITY.

The packet bytes pane shows the raw data in hexadecimal and ASCII. The status bar at the bottom indicates: Packets: 11469 · Displayed: 341 (3.0%) Profile: Default

## b. TCP packets:

### *Using TCP filter*

Activities Wireshark Dec 13 14:52 fifacup.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.30.155.55	142.250.196.78	TCP	66	51516 → 443 [ACK] Seq=1 Ack=1 Win=501 Len=0 TSval=23602737
2	0.052460870	142.250.196.78	10.30.155.55	TCP	66	[TCP ACKed unseen segment] 443 → 51516 [ACK] Seq=1 Ack=2 W
4	0.435254440	185.125.188.61	10.30.155.34	TLSv1.2	90	Application Data
9	0.695088736	10.30.155.55	10.10.4.9	TCP	78	42918 → 53 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=
10	0.695888452	10.10.4.9	10.30.155.55	TCP	66	53 → 42918 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
11	0.695948247	10.30.155.55	10.10.4.9	TCP	54	42918 → 53 [ACK] Seq=1 Ack=1 Win=64256 Len=0
12	0.696042875	10.30.155.55	10.10.4.9	DNS	115	Standard query 0xa1ba SRV _ldap._tcp.VITUNIVERSITY.LOCAL 0
13	0.696501294	10.10.4.9	10.30.155.55	TCP	60	53 → 42918 [ACK] Seq=1 Ack=62 Win=64256 Len=0
14	0.698146022	10.10.4.9	10.30.155.55	DNS	613	Standard query response 0xa1ba SRV _ldap._tcp.VITUNIVERSIT
15	0.698184588	10.30.155.55	10.10.4.9	TCP	54	42918 → 53 [ACK] Seq=62 Ack=560 Win=64128 Len=0
16	0.699715702	10.30.155.55	10.50.2.217	TCP	74	43346 → 389 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=
17	0.700234979	10.50.2.217	10.30.155.55	TCP	66	389 → 43346 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=146
18	0.700307784	10.30.155.55	10.50.2.217	TCP	54	43346 → 389 [ACK] Seq=1 Ack=1 Win=64256 Len=0
19	0.700564168	10.30.155.55	10.50.2.217	LDAP	141	searchRequest(1) "<R00T>" baseObject
20	0.701506003	10.50.2.217	10.30.155.55	LDAP	228	searchResEntry(1) "<R00T>"   searchResDone(1) success [1
21	0.701586094	10.30.155.55	10.50.2.217	TCP	54	43346 → 389 [ACK] Seq=88 Ack=175 Win=64128 Len=0
22	0.701768111	10.30.155.55	10.50.2.217	LDAP	61	unbindRequest(2)
23	0.701793184	10.30.155.55	10.50.2.217	TCP	54	43346 → 389 [FIN, ACK] Seq=95 Ack=175 Win=64128 Len=0
24	0.702228752	10.50.2.217	10.30.155.55	TCP	60	389 → 43346 [ACK] Seq=175 Ack=96 Win=2102272 Len=0
25	0.702272250	10.50.2.217	10.30.155.55	TCP	60	389 → 43346 [RST, ACK] Seq=175 Ack=96 Win=0 Len=0
26	0.703836970	10.30.155.55	10.30.2.214	TCP	74	58422 → 389 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=
27	0.704191178	10.30.2.214	10.30.155.55	TCP	66	389 → 58422 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=146
28	0.704238281	10.30.155.55	10.30.2.214	TCP	54	58422 → 389 [ACK] Seq=1 Ack=1 Win=64256 Len=0
29	0.704406355	10.30.155.55	10.30.2.214	LDAP	314	searchRequest(1) "<R00T>" baseObject
30	0.705007096	10.30.2.214	10.30.155.55	TCP	1514	389 → 58422 [ACK] Seq=1 Ack=261 Win=2102272 Len=1460 [TCP

Frame 1: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface enp4s0, id 0

```

0000  d4 6d 50 84 db cd dc 4a 3e 6b 46 96 08 00 45 00  .mP...J>kF...E.
0010  00 34 55 4a 40 00 40 06 ec db 0a 1e 9b 37 8e fa  .4UJ@.@?....7..
0020  c4 4e c9 3c 01 bb 35 92 11 dc 00 1d 0b 7c 80 10  .N<...5.....|...
0030  01 f5 f8 c4 00 00 01 01 08 0a 8c ae eb 4e 1d 77  .P.....:....&
0040  b6 1a                                     .N.w

```

Transmission Control Protocol: Protocol Packets: 11469 · Displayed: 10173 (88.7%) Profile: Default

## c. HTTP packets:

Activities Wireshark Dec 13 14:51 fifacup.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

No.	Time	Source	Destination	Protocol	Length	Info
1560	31.511524386	10.30.155.55	192.229.232.174	HTTP	504	GET / HTTP/1.1

Frame 1560: 504 bytes on wire (4032 bits), 504 bytes captured (4032 bits) on interface enp4s0, id 0

```

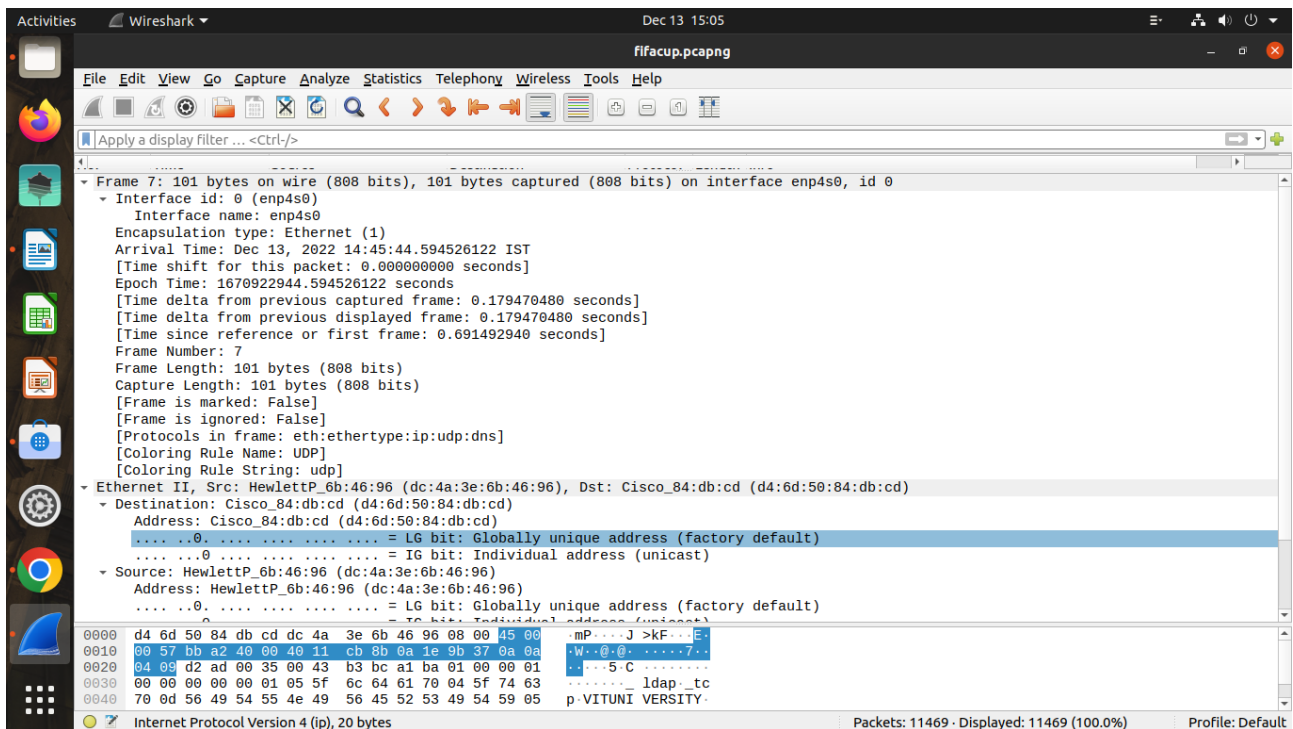
0000  d4 6d 50 84 db cd dc 4a 3e 6b 46 96 08 00 45 00  .mP...J>kF...E.
0010  01 ea aa 8c 40 00 40 06 3f 98 0a 1e 9b 37 c0 e5  .@.@?....7...
0020  e8 ae 8e a0 00 00 c1 b4 f0 74 46 9b 7c b5 80 18  .P.....tF0|...
0030  01 f6 50 c6 00 00 01 01 08 0a 3b ad 0f a4 9f 26  .P.....:....&
0040  8b 21 47 45 54 20 2f 20 48 54 54 50 2f 31 2e 31  .!GET / HTTP/1.1

```

Hypertext Transfer Protocol: Protocol Packets: 11469 · Displayed: 1 (0.0%) Profile: Default

## 2. Ethernet frame, IP packet, and UDP datagram:

i. Examine the frame for the first DNS packet sent by the client



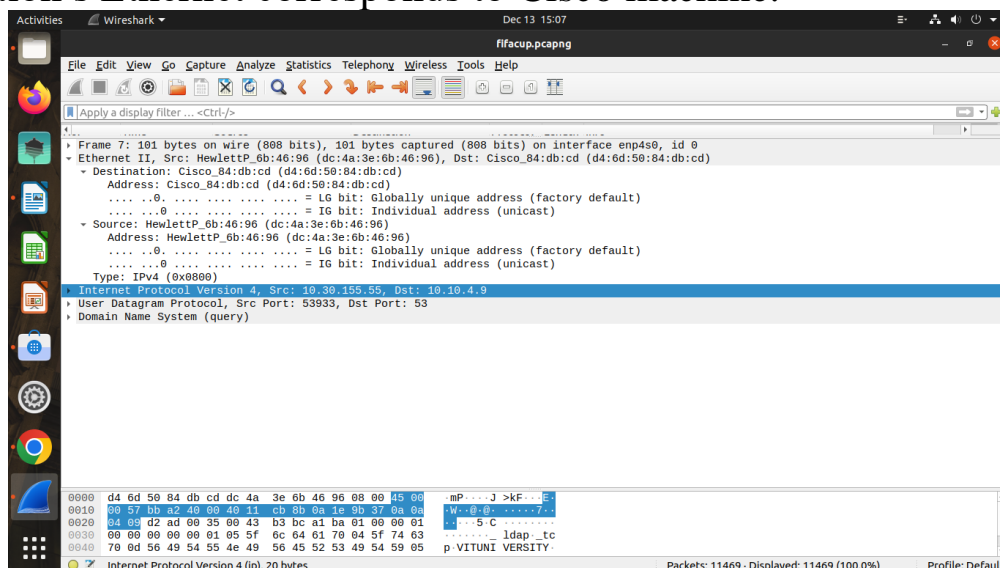
**b. By examining protocols, we found the client's Ethernet: HewlettP\_6b:46:96 (dc:4a:3e:6b:46:96)**

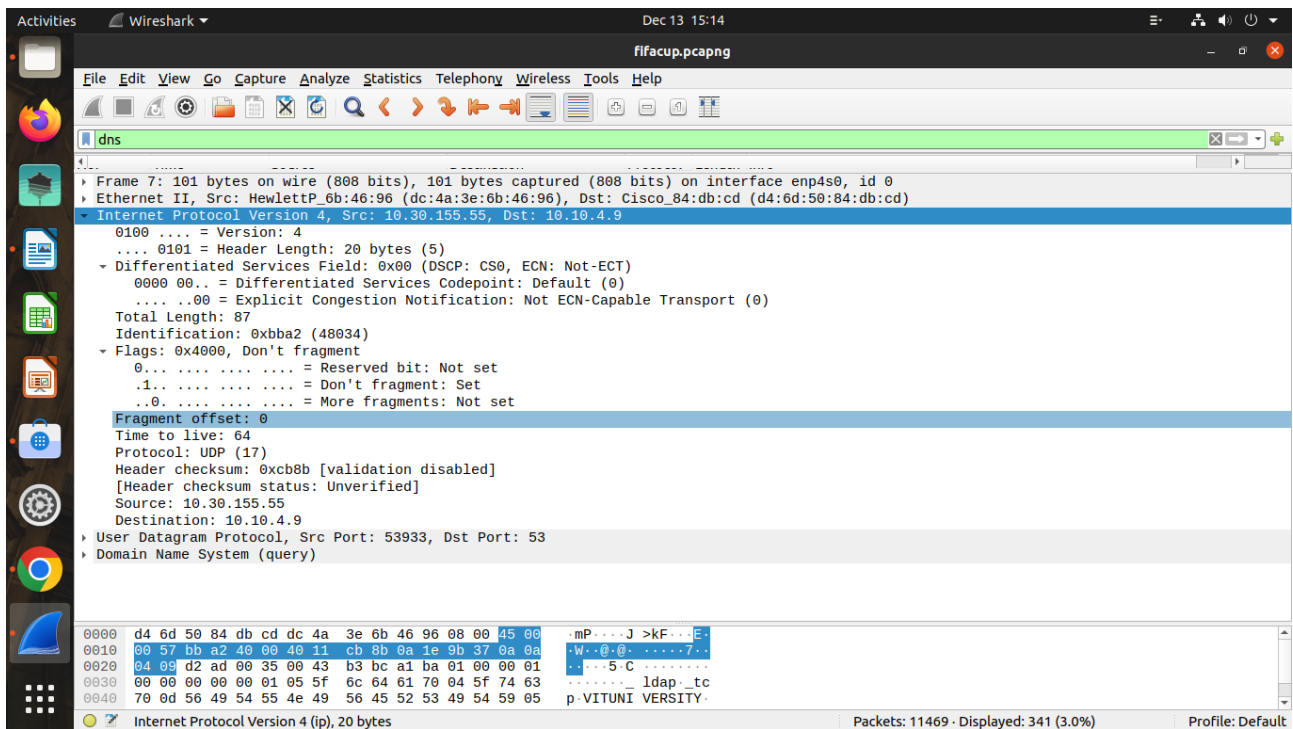
**c. Content of type field in Ethernet Frame: IPv4**

**d. Destination:**

- Ethernet: Cisco\_84:db:cd (d4:6d:50:84:db:cd)
- IP: 10.10.4.9

Destination's Ethernet corresponds to Cisco machine.



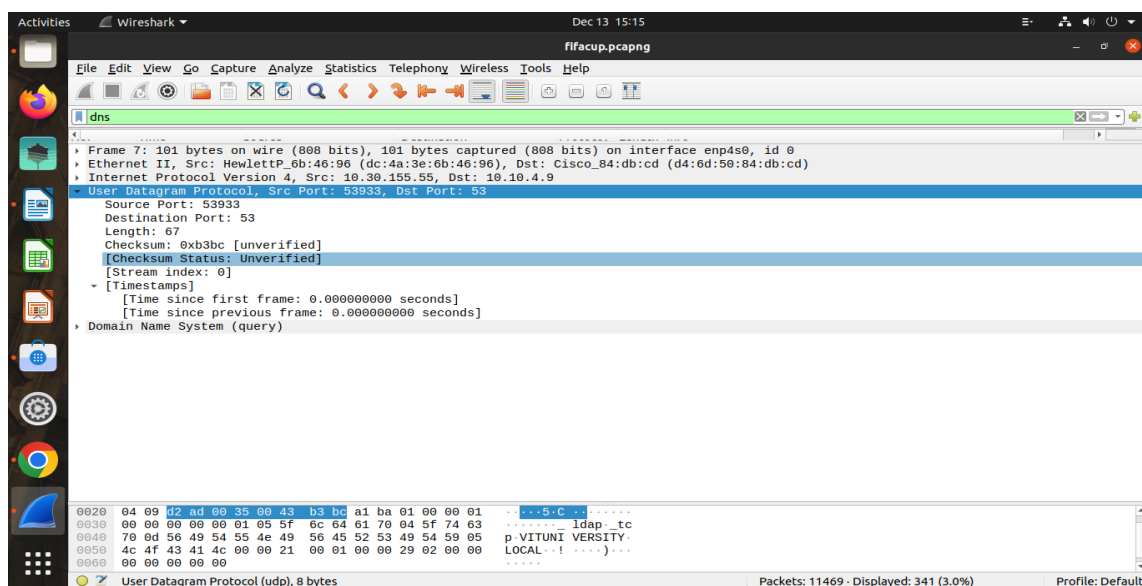


## ii. Examine the IP header for the first DNS packet sent by the client

- Header Length = 20 bytes and Total Length = 87
- Protocol Type field = UDP (User Datagram Protocol) (17)  
Type of protocol in payload = UD

## iii. Examine the UDP header of the first DNS packet sent by the client

a.

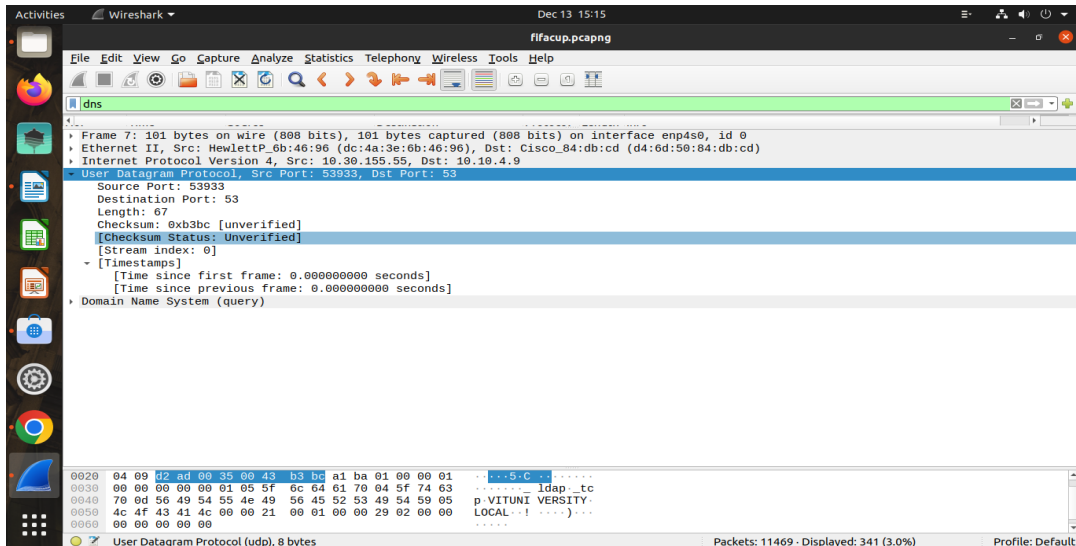


Client Port = 53933  
 Server Port = 53



HTTP as an application layer protocol in payload.

b. If we see the DNS packets, we can conclude that Header Length of IP remains constant for any of the DNS packet sent. Below image is the second DNS packet sent by the client.



### 3. DNS

**Examine the DNS query message in the DNS packet sent by the client.**

**a. What field indicates whether the message is query or a response?**

If Queries field is only present in DNS packet, then we can say the message is query and if there is Queries along with Answers field then we can say that message is a response message.

**b. What information is carried in the body of the query?**

Information such as Domain Name, Name Length, Type and Class is present in the body of the query.

**c. What is the query transaction ID?**

The query transaction ID is a 16-bits random value chosen by the client. When a client sends a question to a DNS server, it remembers the question and its identifier. When a server returns an answer, it returns in the Transaction ID field the identifier chosen by the client. With this client can match the received answer with the question that it sent.

#### d. Identify the fields that carry the type and class of the query.

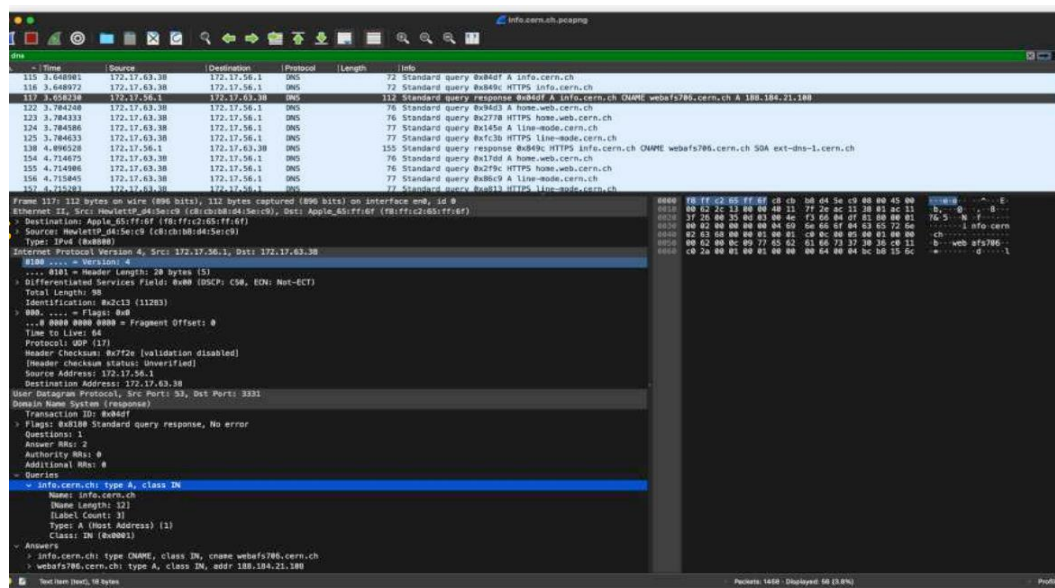
Ans: The Queries field in DNS carries the type and class of the query.

Consider the packet that carries the DNS response to the above query.

a. What should the Ethernet and IP addresses for this packet be?

Verify that these addresses are as expected.

In case of source, the Ethernet and IP addresses for this packet should be the server's system addresses. And in case of destination, client's system Ethernet and IP addresses should be there. The below shown screenshot verifies that:



*Ethernet:*

Source: Cisco\_84:db:cd (d4:6d:84:db:cd)

Destination: HewlettP\_6b:46:96 (dc:4a:3e:6b:46:96)

*IP:*

Source: 10.10.4.9

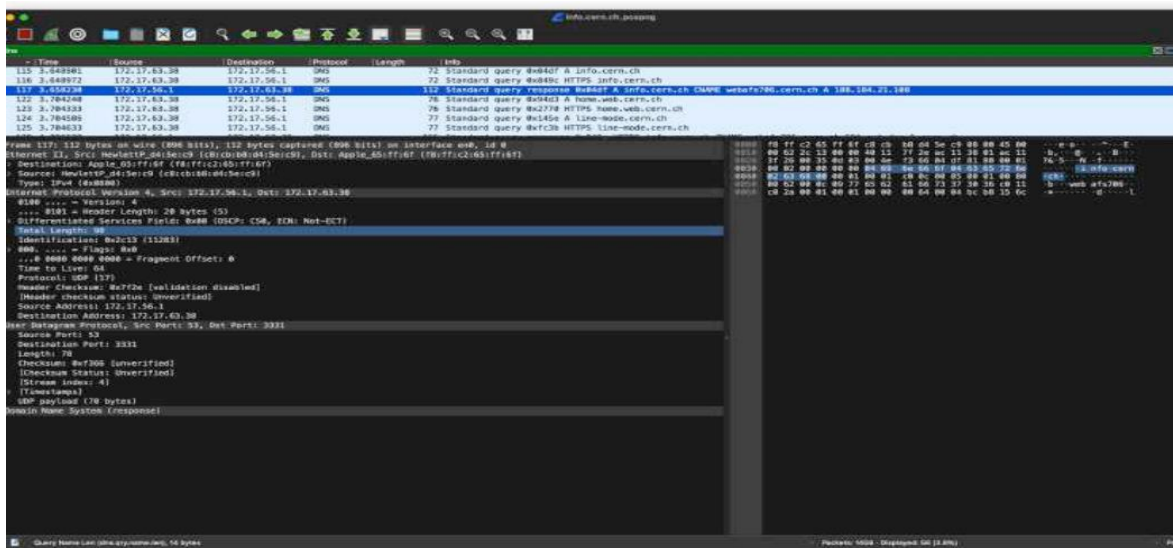
Destination: 10.30.155.55

b. What is the size of the IP packet and UDP datagram that carry the response? Is it longer than the query?

The size of the IP packet and UDP datagram that carry the response varies from the query.

Response:





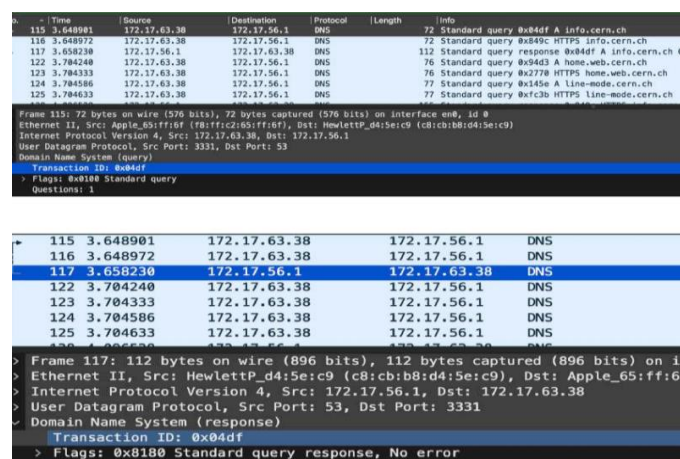
IP packet length: 87 bytes

UDP datagram packet length: 67 bytes.

Yes, the IP packet and UDP datagram for the response has longer length than that of query.

c. Confirm that the transaction ID in the response message is correct.

The transaction ID for the response is 0x04df and that of query is also same. Hence, it is correct.



d. How many answers are provided in the response message? Compare the answers and their time-to-live values.

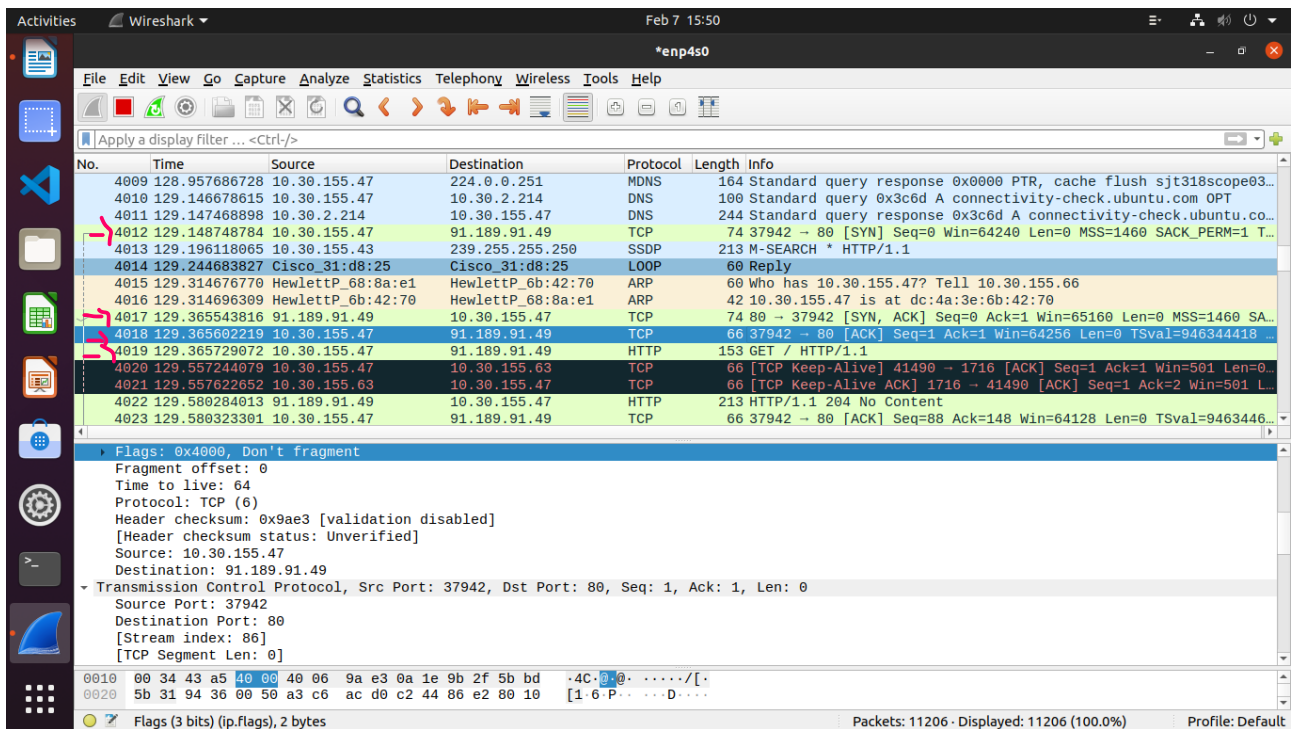
There are two answers in the response message.

Fifa.com- 98 sec (1min, 38 sec)

Fifaplustournaments.com- 100 sec (1min, 40 sec)

## 4. TCP three-way handshake

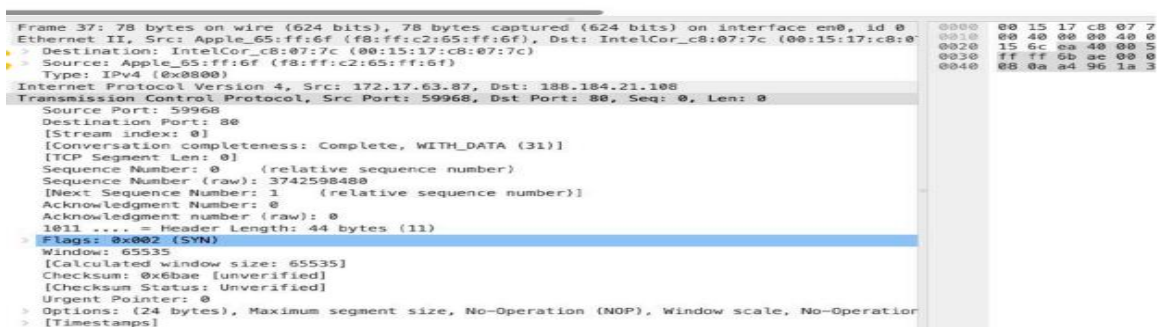
- Identify the frame that carries the first TCP segment in the three-way handshake that sets up the connection between the http client and server.



a. What source Ethernet and IP addresses do you expect in this segment? What protocol and type fields do you expect in the first segment? Confirm that these addresses are as expected.

The client's ethernet and IP address would be there. There would be IP and TCP protocol present in segment. The following image verifies what we have expected.

No.	Time	Source	Destination	Protocol	Length	Info
37	3.315084	172.17.63.87	188.184.21.108	TCP	78	59968 → 80 [SYN] Seq=0 Win=65535 Len=0
45	3.486056	188.184.21.108	172.17.63.87	TCP	74	80 → 59968 [SYN, ACK] Seq=0 Ack=1 Win=0
47	3.486206	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [ACK] Seq=1 Ack=1 Win=13171
335	21.532291	172.17.63.87	188.184.21.108	HTTP	584	GET /hypertext/WWW/Status.html HTTP/1.1
353	21.695870	188.184.21.108	172.17.63.87	TCP	66	80 → 59968 [ACK] Seq=1 Ack=519 Win=3000
354	21.698276	188.184.21.108	172.17.63.87	TCP	1514	80 → 59968 [ACK] Seq=1 Ack=519 Win=3000
355	21.698277	188.184.21.108	172.17.63.87	HTTP	1445	HTTP/1.1 200 OK (text/html)
356	21.698427	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [ACK] Seq=519 Ack=2828 Win=
357	21.698715	188.184.21.108	172.17.63.87	TCP	66	80 → 59968 [FIN, ACK] Seq=2828 Ack=519
358	21.698823	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [ACK] Seq=519 Ack=2829 Win=
359	21.698913	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [FIN, ACK] Seq=519 Ack=2829
360	21.864797	188.184.21.108	172.17.63.87	TCP	66	80 → 59968 [ACK] Seq=2829 Ack=520 Win=



b. Explain the values in the destination Ethernet and IP addresses in the first segment? To what machines these addresses correspond?

The destination system has the name (IntelCor\_c8:07:7c) with mac address as (00:15:17:c8:07:7c) and IP address as 188.184.21.108

(ip.addr eq 188.184.21.108 and ip.addr eq 172.17.63.87) and (tcp.port eq 80 and tcp.port eq 59968)

No.	Time	Source	Destination	Protocol	Length	Info
37	3.315004	172.17.63.87	188.184.21.108	TCP	78	59968 → 80 [SYN] Seq=0 Win=65535 Len=0
45	3.486056	188.184.21.108	172.17.63.87	TCP	74	80 → 59968 [SYN, ACK] Seq=0 Ack=1 Win=
47	3.486206	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [ACK] Seq=1 Ack=1 Win=13171
335	21.532291	172.17.63.87	188.184.21.108	HTTP	584	GET /hypertext/www/Status.html HTTP/1.
353	21.695870	188.184.21.108	172.17.63.87	TCP	66	80 → 59968 [ACK] Seq=1 Ack=519 Win=300
354	21.698276	188.184.21.108	172.17.63.87	TCP	1514	80 → 59968 [ACK] Seq=1 Ack=519 Win=300
355	21.698277	188.184.21.108	172.17.63.87	HTTP	1445	HTTP/1.1 200 OK (text/html)
356	21.698427	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [ACK] Seq=519 Ack=2828 Win=
357	21.698715	188.184.21.108	172.17.63.87	TCP	66	80 → 59968 [FIN, ACK] Seq=2828 Ack=519
358	21.698823	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [ACK] Seq=519 Ack=2829 Win=
359	21.699313	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [FIN, ACK] Seq=519 Ack=2829
360	21.864797	188.184.21.108	172.17.63.87	TCP	66	80 → 59968 [ACK] Seq=2829 Ack=520 Win=

Frame 37: 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on interface en0, id 0  
Ethernet II, Src: Apple\_G5:ff:6f (f8:ff:c2:65:ff:6f), Dst: IntelCor\_c8:07:7c (00:15:17:c8:07:7c)  
Source: Apple\_G5:ff:6f (f8:ff:c2:65:ff:6f)  
Type: IPv4 (0x0800)  
Internet Protocol Version 4, Src: 172.17.63.87, Dst: 188.184.21.108  
0100 ... = Version: 4  
... 0101 = Header Length: 20 bytes (5)  
Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)  
Total Length: 64  
Identification: 0x0000 (0)  
010 ... = Flags: 0x2, Don't fragment  
... 0000 0000 0000 = Fragment Offset: 0  
Time to Live: 64  
Protocol: TCP (6)  
Header Checksum: 0x7d2b [validation disabled]  
[Header checksum status: Unverified]  
Source Address: 172.17.63.87  
Destination Address: 188.184.21.108  
Transmission Control Protocol, Src Port: 59968, Dst Port: 80, Seq: 0, Len: 0

c. Identify the ephemeral port number used by the client and confirm that the well-known port number is the correct value for HTTP.

The ephemeral port number used by client is 59968. The well-known port number for HTTP is from 0 – 1023.

d. What is the length of the TCP segment?

The TCP segment length is zero.

(ip.addr eq 188.184.21.108 and ip.addr eq 172.17.63.87) and (tcp.port eq 80 and tcp.port eq 59968)

No.	Time	Source	Destination	Protocol	Length	Info
37	3.315004	172.17.63.87	188.184.21.108	TCP	78	59968 → 80 [SYN] Seq=0 Win=65535 Len=0
45	3.486056	188.184.21.108	172.17.63.87	TCP	74	80 → 59968 [SYN, ACK] Seq=0 Ack=1 Win=
47	3.486206	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [ACK] Seq=1 Ack=1 Win=13171
335	21.532291	172.17.63.87	188.184.21.108	HTTP	584	GET /hypertext/www/Status.html HTTP/1.
353	21.695870	188.184.21.108	172.17.63.87	TCP	66	80 → 59968 [ACK] Seq=1 Ack=519 Win=300
354	21.698276	188.184.21.108	172.17.63.87	TCP	1514	80 → 59968 [ACK] Seq=1 Ack=519 Win=300
355	21.698277	188.184.21.108	172.17.63.87	HTTP	1445	HTTP/1.1 200 OK (text/html)
356	21.698427	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [ACK] Seq=519 Ack=2828 Win=
357	21.698715	188.184.21.108	172.17.63.87	TCP	66	80 → 59968 [FIN, ACK] Seq=2828 Ack=519
358	21.698823	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [ACK] Seq=519 Ack=2829 Win=
359	21.699313	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [FIN, ACK] Seq=519 Ack=2829
360	21.864797	188.184.21.108	172.17.63.87	TCP	66	80 → 59968 [ACK] Seq=2829 Ack=520 Win=

Source Port: 59968  
Destination Port: 80  
[Stream index: 0]  
[Conversation completeness: Complete, WITH\_DATA (31)]  
[Length: 0]  
Sequence Number: 0 (relative sequence number)  
Sequence Number (raw): 3742598480  
[Next Sequence Number: 1 (relative sequence number)]  
Acknowledgment Number: 0  
Acknowledgment Number (raw): 0  
1011 ... = Header Length: 44 bytes (11)  
Flags: 0x002 (SYN)  
Window: 65535  
[Calculated window size: 65535]  
Checksum: 0x6bae [unverified]  
[Checksum Status: Unverified]  
Urgent Pointer: 0  
Options: (24 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation  
TCP Option - Maximum segment size: 1460 bytes  
TCP Option - No-Operation (NOP)  
TCP Option - Window scale: 6 (multiply by 64)  
TCP Option - No-Operation (NOP)  
TCP Option - No-Operation (NOP)  
TCP Option - Timestamp  
TCP Option - SACK permitted  
TCP Option - End of Option List (EOL)



e. What is the initial sequence number for the segments from the client to the server? What is the initial window size? What is the maximum segment size?

The initial sequence number for the segments from the client to the server is 0. Initial window size = 65535 Maximum segment size = 1460 bytes

f. Find the hex character that contains the SYN flag bit.

Hex character = 0x002 (SYN)

ip.addr eq 172.17.63.87 and ip.addr eq 188.184.21.108 and (tcp.port eq 59968 and tcp.port eq 80)

Time	Source	Destination	Protocol	Length	Info
37 3.315004	172.17.63.87	188.184.21.108	TCP	78	59968 → 80 [SYN] Seq=0 Win=65535 Len=0
45 3.486056	188.184.21.108	172.17.63.87	TCP	74	80 → 59968 [SYN, ACK] Seq=0 Ack=1 Win=0
47 3.486206	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [ACK] Seq=1 Ack=1 Win=13171
335 21.532291	172.17.63.87	188.184.21.108	HTTP	584	GET /hypertext/World/Status.html HTTP/1.1
353 21.695870	188.184.21.108	172.17.63.87	TCP	66	80 → 59968 [ACK] Seq=1 Ack=519 Win=3000
354 21.698276	188.184.21.108	172.17.63.87	TCP	1514	80 → 59968 [ACK] Seq=1 Ack=519 Win=3000
355 21.698277	188.184.21.108	172.17.63.87	HTTP	1445	HTTP/1.1 200 OK (text/html)
356 21.698427	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [ACK] Seq=519 Ack=2828 Win=
357 21.698715	188.184.21.108	172.17.63.87	TCP	66	80 → 59968 [FIN, ACK] Seq=2828 Ack=519
358 21.698823	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [ACK] Seq=519 Ack=2829 Win=
359 21.699313	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [FIN, ACK] Seq=519 Ack=2829
360 21.864797	188.184.21.108	172.17.63.87	TCP	66	80 → 59968 [ACK] Seq=2829 Ack=520 Win=

Source Port: 59968		0000 00 15 17 c8 07 7
Destination Port: 80		0010 00 40 00 00 40 0
[Stream index: 0]		0020 15 6c ea 40 00 5
[Conversation completeness: Complete, WITH_DATA (31)]		0030 ff ff 6b ae 00 0
[TCP Segment Len: 0]		0040 00 0a a4 96 1a 3
Sequence Number: 0 (relative sequence number)		
Sequence Number (raw): 3742598480		
[Next Sequence Number: 1 (relative sequence number)]		
Acknowledgment Number: 0		
1011 .... = Header Length: 44 bytes (11)		
Flags: 0x002 (SYN)		
000. .... = Reserved: Not set		
...0 .... = Accurate ECN: Not set		
....0... = Congestion Window Reduced: Not set		
....0... = ECN-Echo: Not set		
....0... = Urgent: Not set		
....0... = Acknowledgment: Not set		
....0... = Push: Not set		
....0... = Reset: Not set		
→ ....0... = SYN: Set		
....0... = Fin: Not set		
[TCP Flags: .....S.]		
Window: 65535		
[Calculated window size: 65535]		
Checksum: 0x6bae [unverified]		

• **Identify the frame that carries the second segment in the three-way handshake.**

a. How much time elapsed between the capture of the first and second segments?

The total of 18 sec (approx.) is elapsed between the first and second segment.

b. Before examining the captured packets, specify the values for the following fields in this frame:

• Source and destination addresses and type field in Ethernet frame.

Source: Apple\_65:ff:6f (f8:ff:c2:65:ff:6f) Destination: IntelCor\_c8:07:7c (00:15:17:c8:07:7c) Type: IPv4 (0x0800)

• Source and destination IP addresses and port numbers in IP packet.

Source Address: 172.17.63.87 Destination Address: 188.184.21.108

Source Port number = 59968 Destination Port number = 80

• Acknowledgement number in TCP segment.

Acknowledgment Number: 2828 (relative ack number)

• Values of flag bits.

In flag bits, Acknowledgment bit is only set

- *Confirm that the frame contains the expected values.*

No.	Time	Source	Destination	Protocol	Length	Info
37	3.315004	172.17.63.87	188.184.21.108	TCP	78	59968 → 80 [SYN] Seq=0 Win=65535 Len=
45	3.486056	188.184.21.108	172.17.63.87	TCP	74	80 → 59968 [SYN, ACK] Seq=0 Ack=1 Wi
47	3.486206	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [ACK] Seq=1 Ack=1 Win=131
335	21.532291	172.17.63.87	188.184.21.108	HTTP	584	GET /hypertext/www/Status.html HTTP/
353	21.698870	188.184.21.108	172.17.63.87	TCP	66	80 → 59968 [ACK] Seq=1 Ack=519 Win=3
354	21.698276	188.184.21.108	172.17.63.87	TCP	1514	80 → 59968 [ACK] Seq=1 Ack=519 Win=3
355	21.698277	188.184.21.108	172.17.63.87	HTTP	1445	HTTP/1.1 200 OK (text/html)
356	21.698427	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [ACK] Seq=519 Ack=2828 Wi
357	21.698715	188.184.21.108	172.17.63.87	TCP	66	80 → 59968 [FIN, ACK] Seq=2828 Ack=5
358	21.698823	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [ACK] Seq=519 Ack=2829 Wi
359	21.699313	172.17.63.87	188.184.21.108	TCP	66	59968 → 80 [FIN, ACK] Seq=519 Ack=28
360	21.864797	188.184.21.108	172.17.63.87	TCP	66	80 → 59968 [ACK] Seq=2829 Ack=520 Wi

Ethernet II, Src: Apple_G5:ff:6f (f8:ff:c2:65:ff:6f), Dst: IntelCor_c8:07:7c (00:15:17:c8:07:7c)	
Destination: IntelCor_c8:07:7c (00:15:17:c8:07:7c)	0000 00 15 17 c8 07
Address: IntelCor_c8:07:7c (00:15:17:c8:07:7c)	0010 00 34 00 00 40
..... = LG bit: Globally unique address (factory default)	0020 15 6c ea 40 00
..... = IG bit: Individual address (unicast)	0030 07 de 6a 3f 00
Source: Apple_G5:ff:6f (f8:ff:c2:65:ff:6f)	0040 6f 12
Address: Apple_G5:ff:6f (f8:ff:c2:65:ff:6f)	
..... = LG bit: Globally unique address (factory default)	
..... = IG bit: Individual address (unicast)	
Type: IPv4 (0x0800)	

Internet Protocol Version 4, Src: 172.17.63.87, Dst: 188.184.21.108	
Version: 4	0100 .... = Version: 4
Header Length: 20 bytes (5)	.... 0101 = Header Length: 20 bytes (5)
Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)	.... 0100 = Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 52	.... 0100 = Total Length: 52
Identification: 0x0000 (0)	.... 0100 = Identification: 0x0000 (0)
Flags: 0x2, Don't fragment	.... 0100 = Flags: 0x2, Don't fragment
Fragment Offset: 0	.... 0100 = Fragment Offset: 0
Time to Live: 64	.... 0100 = Time to Live: 64
Protocol: TCP (6)	.... 0100 = Protocol: TCP (6)
Header Checksum: 0x7d37 [validation disabled]	.... 0100 = Header Checksum: 0x7d37 [validation disabled]
[Header checksum status: Unverified]	.... 0100 = [Header checksum status: Unverified]
Source Address: 172.17.63.87	.... 0100 = Source Address: 172.17.63.87
Destination Address: 188.184.21.108	.... 0100 = Destination Address: 188.184.21.108

Transmission Control Protocol, Src Port: 59968, Dst Port: 80, Seq: 519, Ack: 2828, Len: 0	
Source Port: 59968	.... 0100 = Source Port: 59968
Destination Port: 80	.... 0100 = Destination Port: 80
[Stream index: 0]	.... 0100 = [Stream index: 0]
[Conversation completeness: Complete, WITH_DATA (31)]	.... 0100 = [Conversation completeness: Complete, WITH_DATA (31)]
[TCP Segment Len: 0]	.... 0100 = [TCP Segment Len: 0]
Sequence Number: 519 (relative sequence number)	.... 0100 = Sequence Number: 519 (relative sequence number)
Sequence Number (raw): 3742598999	.... 0100 = Sequence Number (raw): 3742598999
[Next Sequence Number: 519 (relative sequence number)]	.... 0100 = [Next Sequence Number: 519 (relative sequence number)]
Acknowledgment Number: 2828 (relative ack number)	.... 0100 = Acknowledgment Number: 2828 (relative ack number)
Acknowledgment number (raw): 2823574157	.... 0100 = Acknowledgment number (raw): 2823574157
1000 .... = Header Length: 32 bytes (8)	.... 0100 = 1000 .... = Header Length: 32 bytes (8)
Flags: 0x010 (ACK)	.... 0100 = Flags: 0x010 (ACK)
0000 .... = Reserved: Not set	.... 0100 = 0000 .... = Reserved: Not set
.... 0000 .... = Accurate ECN: Not set	.... 0100 = .... 0000 .... = Accurate ECN: Not set
.... 0000 .... = Congestion Window Reduced: Not set	.... 0100 = .... 0000 .... = Congestion Window Reduced: Not set
.... 0000 .... = ECN-Echo: Not set	.... 0100 = .... 0000 .... = ECN-Echo: Not set
.... 0000 .... = Urgent: Not set	.... 0100 = .... 0000 .... = Urgent: Not set

c. What is the length of the TCP segment?

The length of TCP segment is zero

d. What is the initial sequence number for the connection from the server to the client? What is the maximum segment size?

Initial sequence number from server to client: 2828 (relative sequence number) Maximum segment size = 1460 bytes

## 5. HTTP GET

- **Identify the frame that carries the HTTP “GET” message.**

a. *Confirm that the sequence and acknowledgement values in the TCP header are as Expected*

Sequence Number = 1 Acknowledgment Number = 1

b. *Examine the flag bits in the TCP header. Can you explain why the two flag bits are set?*

The Acknowledgment (ACK) and Push (PSH) in flags bits are set. When ACK bit is set, it contains the value of the next sequence number the sender of the packet is expecting to receive. When PSH bit is set, it is

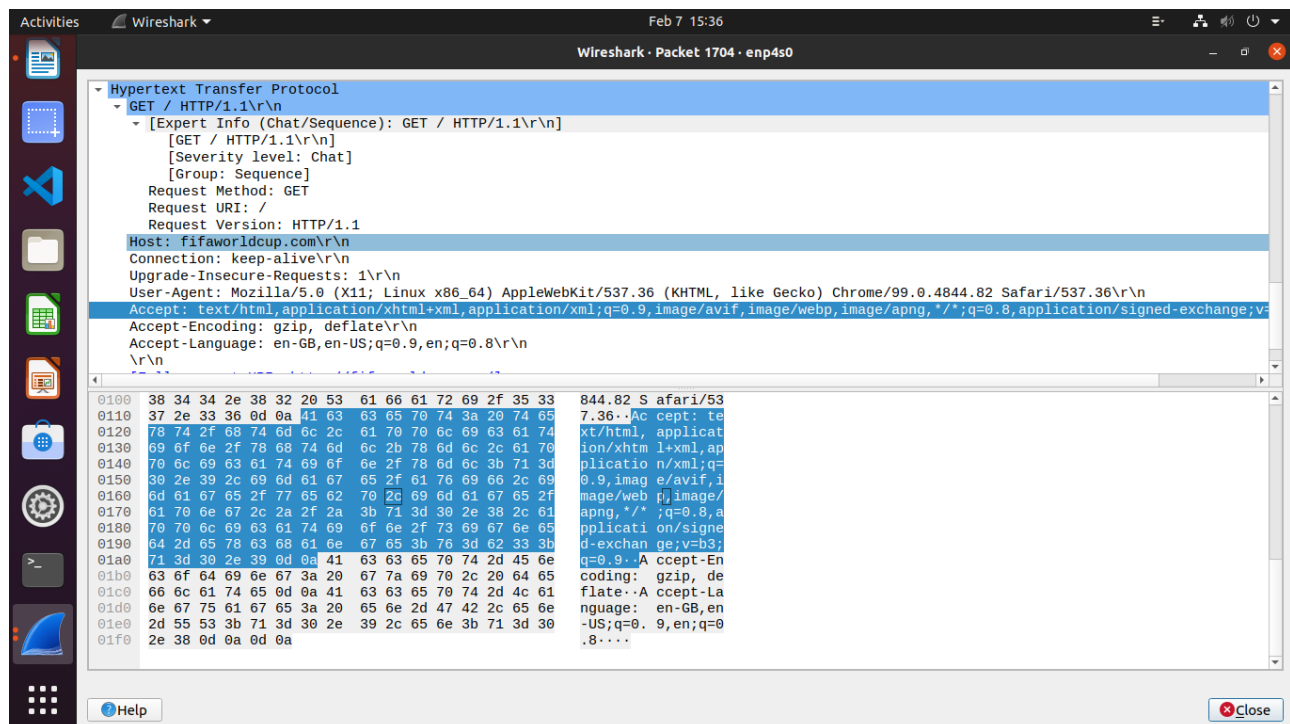
notification from the sender to the receiver that the receiver should pass all the data to the application quickly.

c. What are the lengths of the TCP segment and payload?

The TCP segment length: 87

- Consider the contents of the “GET” message.

a. Scroll down the third pane in the Wireshark window and compare the decoded text with the contents of the HTTP message in the second window.



b. Count the number of octets in the message and verify that this number is consistent with the length information in the TCP header.

Total number of octets can be calculated by subtracting the header length of IP and header length of TCP from total length of IP. i.e.,

IP → Total length = (IP header length + TCP Header length + application)

Therefore, Total number of octets = IP\_Length – IP\_header\_length – TCP\_header\_length

Total number of octets = 139 – 20 – 32 = 87



Wireshark interface showing network traffic capture. The packet list displays several HTTP requests and responses. The packet details pane shows the structure of an Internet Protocol Version 4 packet and a Transmission Control Protocol segment.

No.	Time	Source	Destination	Protocol	Length	Info
1704	41.753011521	10.30.155.47	192.229.232.174	HTTP	502	GET / HTTP/1.1
1756	42.760512477	192.229.232.174	10.30.155.47	HTTP	1283	HTTP/1.1 503 Service Unavailable (text/html)
1767	42.846511030	10.30.155.47	192.229.232.174	HTTP	447	GET /favicon.ico HTTP/1.1
1768	42.848210285	192.229.232.174	10.30.155.47	HTTP	1294	HTTP/1.1 503 Service Unavailable (text/html)
5711	194.039464887	10.30.155.47	185.125.190.49	HTTP	153	GET / HTTP/1.1
5712	194.179466947	185.125.190.49	10.30.155.47	HTTP	213	HTTP/1.1 204 No Content
13210	494.177867810	10.30.155.47	35.224.170.84	HTTP	153	GET / HTTP/1.1
13216	494.453191025	35.224.170.84	10.30.155.47	HTTP	214	HTTP/1.1 204 No Content

Internet Protocol Version 4, Src: 10.30.155.47, Dst: 185.125.190.49

- 0100 ... = Version: 4
- ... 0101 = Header Length: 20 bytes (5)
- Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
- Total Length: 139
- Identification: 0xfb69 (64361)
- Flags: 0x4000, Don't fragment
- Fragment offset: 0
- Time to live: 64
- Protocol: TCP (6)
- Header checksum: 0x2207 [validation disabled]
- [Header checksum status: Unverified]
- Source: 10.30.155.47
- Destination: 185.125.190.49

Transmission Control Protocol, Src Port: 38734, Dst Port: 80, Seq: 1, Ack: 1, Len: 87

- Source Port: 38734
- Destination Port: 80
- [Stream index: 88]
- [TCP Segment Len: 87]

0010 00 8b fb 69 40 00 40 06 22 07 0a 1e 9b 2f b9 7d ... i@. .{

0020 be 31 97 4e 00 50 52 a5 4e 17 d3 98 82 c5 80 18 ... 1.N.PR. N.....

Total Length (ip.len), 2 bytes

Packets: 18831 · Displayed: 8 (0.0%)

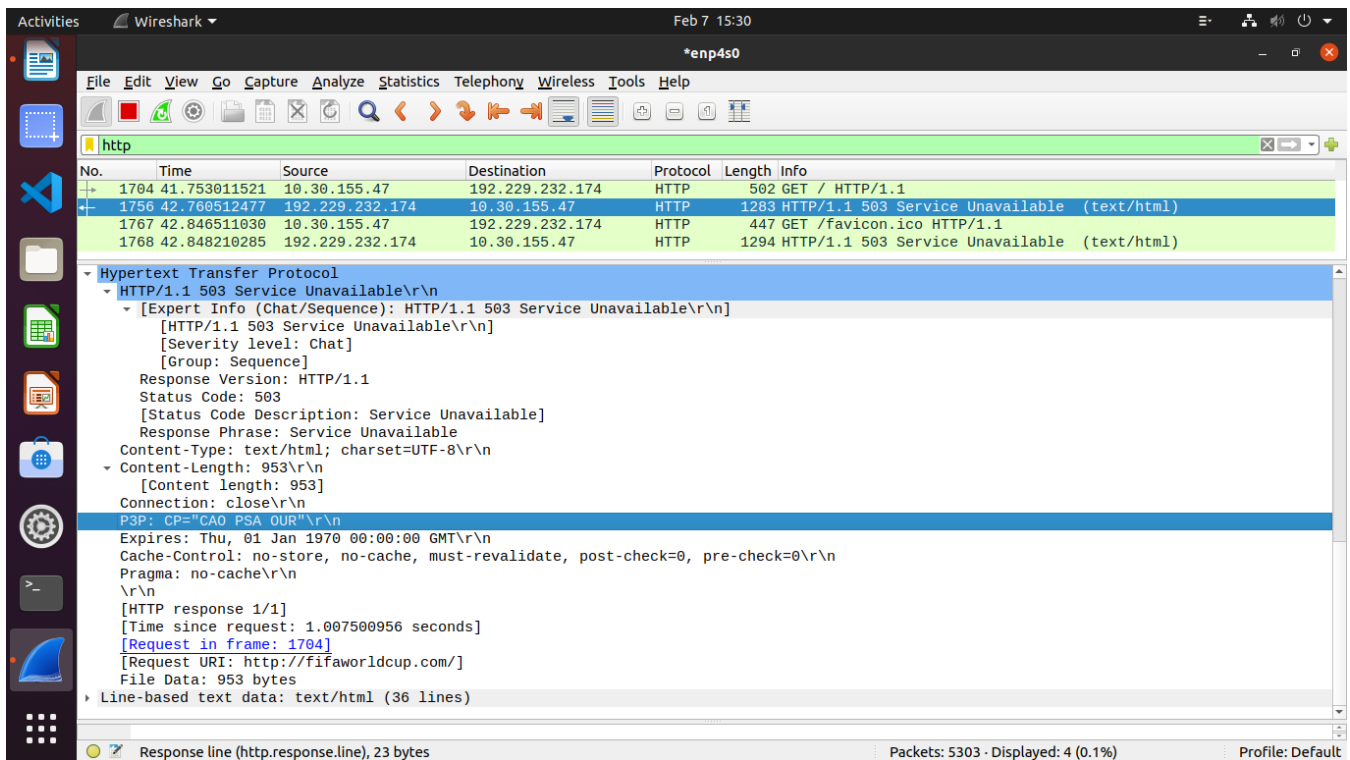
Profile: Default

c. What is the next sequence number that is expected in the next segment from the server?

The next sequence number that is expected in the next segment is same. Since, the server acknowledge with sequence number that it has received and ask it for next segment.

## • HTTP Response

• How much time elapses between the capture of the GET message and the capture of the corresponding Response message?



- Determine whether the server responds with an HTTP response message or simply with a TCP ACK segment. Verify that the sequence number in the segment from the server is as expected

Server response with a message. In this case, server response with a text/html file.

- Consider the segment that contains the HTTP response message a. What is the length of the payload in the TCP segment?

The length of the TCP segment is found to be 256 bytes

No.	Time	Source	Destination	Protocol	Length	Info
641	40.229542	172.17.63.87	188.184.21.108	HTTP	591	GET /hypertext/www/Project/Codin
665	40.396136	188.184.21.108	172.17.63.87	HTTP	322	HTTP/1.1 200 OK (text/html)
534	33.184950	172.17.63.87	188.184.21.108	TCP	78	59971 → 80 [SYN] Seq=0 Win=65535
551	33.344133	188.184.21.108	172.17.63.87	TCP	74	80 → 59971 [SYN, ACK] Seq=0 Ack=
552	33.344246	172.17.63.87	188.184.21.108	TCP	66	59971 → 80 [ACK] Seq=1 Ack=1 Win
660	40.393080	188.184.21.108	172.17.63.87	TCP	66	80 → 59971 [ACK] Seq=1 Ack=526 W
662	40.395054	188.184.21.108	172.17.63.87	TCP	1514	80 → 59971 [ACK] Seq=1 Ack=526 W
664	40.395312	172.17.63.87	188.184.21.108	TCP	66	59971 → 80 [ACK] Seq=526 Ack=144
666	40.396138	188.184.21.108	172.17.63.87	TCP	66	80 → 59971 [FIN, ACK] Seq=1705 A
667	40.396254	172.17.63.87	188.184.21.108	TCP	66	59971 → 80 [ACK] Seq=526 Ack=170
668	40.396297	172.17.63.87	188.184.21.108	TCP	66	59971 → 80 [ACK] Seq=526 Ack=170
669	40.397030	172.17.63.87	188.184.21.108	TCP	66	59971 → 80 [FIN, ACK] Seq=526 Ac
676	40.555383	188.184.21.108	172.17.63.87	TCP	66	80 → 59971 [ACK] Seq=1705 Ack=52

<p>Frame 665: 322 bytes on wire (2576 bits), 322 bytes captured (2576 bits) on interface</p> <p>Ethernet II, Src: IntelCor_c8:07:7c (08:15:17:c8:07:7c), Dst: Apple_65:ff:6f (f8:00:00:00:00:00)</p> <p>Internet Protocol Version 4, Src: 188.184.21.108, Dst: 172.17.63.87</p> <p>Transmission Control Protocol, Src Port: 80, Dst Port: 59971, Seq: 1449, Ack: 526</p> <p>Source Port: 80</p> <p>Destination Port: 59971</p> <p>[Stream index: 14]</p> <p>[Conversation completeness: Complete, MITH_DATA (31)]</p> <p>[TCP Segment Len: 256]</p> <p>Sequence Number: 1449 (relative sequence number)</p> <p>Sequence Number (raw): 3464020730</p> <p>Next Sequence Number: 1705 (relative sequence number)</p> <p>Acknowledgment Number: 526 (relative ack number)</p> <p>Acknowledgment number (raw): 3473845228</p> <p>1000 .... = Header Length: 32 bytes (8)</p> <p>Flags: 0x01B (PSH, ACK)</p> <p>Window: 235</p> <p>[Calculated window size: 30000]</p> <p>[Window size scaling factor: 128]</p> <p>Checksum: 0x4e38 [unverified]</p> <p>[Checksum Status: Unverified]</p> <p>Urgent Pointer: 0</p> <p>Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps</p> <p>[Timestamps]</p> <p>[Time since first frame in this TCP stream: 7.211186000 seconds]</p> <p>[Time since previous frame in this TCP stream: 0.000240000 seconds]</p> <p>[SEQ/ACK analysis]</p> <p>[RRTT: 0.159296000 seconds]</p> <p>[Bytes in flight: 256]</p> <p>[Bytes sent since last PSH flag: 1704]</p> <p>TCP payload (256 bytes)</p> <p>TCP segment data (256 bytes)</p> <p>[2 Reassembled TCP Segments (1704 bytes): #662(1448), #663(256)]</p> <p>Hypertext Transfer Protocol</p> <p>128 bytes of data (128 bytes)</p>	<p>0000 f8 ff c2 65 ff 6f 00</p> <p>0010 01 34 85 4a 40 00 2e</p> <p>0020 3f 57 00 50 ea 43 ce</p> <p>0030 00 eb 4a 38 00 00 01</p> <p>0040 b9 91 3c 50 3e 0e 28</p> <p>0050 70 6f 69 6c 74 65 72</p> <p>0060 20 3c 41 20 4e 41 4d</p> <p>0070 2e 2e 2f 43 6f 64 69</p> <p>0080 64 73 2e 68 74 6d 6c</p> <p>0090 6f 6d 61 69 6e 20 73</p> <p>00a0 20 29 2e 20 5f 5f 5f</p> <p>00b0 5f 5f 5f 5f 5f 5f 5f</p> <p>00c0 5f 5f 5f 5f 5f 5f 5f</p> <p>00d0 5f 5f 5f 5f 5f 5f 5f</p> <p>00e0 5f 5f 5f 5f 5f 5f 5f</p> <p>00f0 41 20 4e 41 4d 45 3d</p> <p>0100 74 70 3a 2f 69 6e</p> <p>0110 68 2e 2f 68 79 70 65</p> <p>0120 ff 44 69 73 63 6c 61</p> <p>0130 3e 54 69 6d 20 42 4c</p> <p>0140 50 3e</p>
--	--

b. Examine whether any of the flags are set and explain why they are set  
ACK flags are set because server is acknowledging the request received by it and PSH is set because it shows the end of the data.

c. What acknowledgement number is expected in the next segment from the client?

Acknowledgement number would be increased for the next segment upon the request made by client.

• Consider the HTTP response message

a. What is the result code in the response message?

The result code in the response message is 200 which means OK.

b. Highlight the “data” section of the HTTP response message. Scroll down the third pane in the Wireshark window and compare the decoded text with the contents of the web page that was displayed on your screen.

Activities Wireshark Feb 7 15:36

Wireshark · Packet 1704 · enp4s0

Hypertext Transfer Protocol

GET / HTTP/1.1\r\n

[Expert Info (Chat/Sequence): GET / HTTP/1.1\r\n]

[GET / HTTP/1.1\r\n]

[Severity level: Chat]

[Group: Sequence]

Request Method: GET

Request URI: /

Request Version: HTTP/1.1

Host: fifaworldcup.com\r\n

Connection: keep-alive\r\n

Upgrade-Insecure-Requests: 1\r\n

User-Agent: Mozilla/5.0 (X11; Linux x86\_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/99.0.4844.82 Safari/537.36\r\n

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,\*/\*;q=0.8,application/signed-exchange;v=b3\r\n

Accept-Encoding: gzip, deflate\r\n

Accept-Language: en-GB,en-US;q=0.9,en;q=0.8\r\n

\r\n

0109 38 34 34 2e 38 32 20 53 61 66 61 72 69 2f 35 33 844.82 S afari/53

0110 37 2e 33 36 0d 0a 41 63 63 65 70 74 3a 20 74 65 7.36..Ac cept: te

0111 78 74 2f 68 74 6d 6c 2c 61 70 70 6c 69 63 61 74 xt/html, applicat

0112 69 6f 6e 2f 78 68 74 6d 6c 2b 78 6d 6c 2c 61 70 ion/xhtml+xml,ap

0113 70 6c 69 63 61 74 69 6f 6e 2f 78 6d 6c 3b 71 3d plicatio n/xml;q=

0114 30 2e 39 2c 69 6d 61 67 65 2f 61 76 69 66 2c 69 0.9,imag e/avif,i

0115 6d 61 67 65 2f 77 65 62 70 2c 69 6d 61 67 65 2f image/web p,image/

0116 61 70 6e 67 2c 2a 2f 2a 3b 71 3d 30 2e 38 2c 61 apng,\*/\* ;q=0.8,a

0117 70 70 6c 69 63 61 74 69 6f 6e 2f 73 69 67 6e 65 pplicati on/signe

0118 64 2d 65 78 63 68 61 6e 67 65 3b 76 3d 62 33 3b d-exchan ge;v=b3;

0119 71 3d 30 2e 39 0d 0a 41 63 63 65 70 74 2d 45 6e q=0.9..A ccept-En

0120 63 6f 64 69 6e 67 3a 20 67 7a 69 70 2c 20 64 65 coding: gzip, de

0121 66 6c 61 74 65 0d 0a 41 63 63 65 70 74 2d 4c 61 flate..A ccept-La

0122 6e 67 75 61 67 65 3a 20 65 6e 2d 47 42 2c 65 6e nguage: en-GB,La

0123 2d 55 53 3b 71 3d 30 2e 39 2c 65 6e 3b 71 3d 30 -US;q=0. 9,en;q=0

0124 2e 38 0d 0a 0d 0a .8....

Help Close