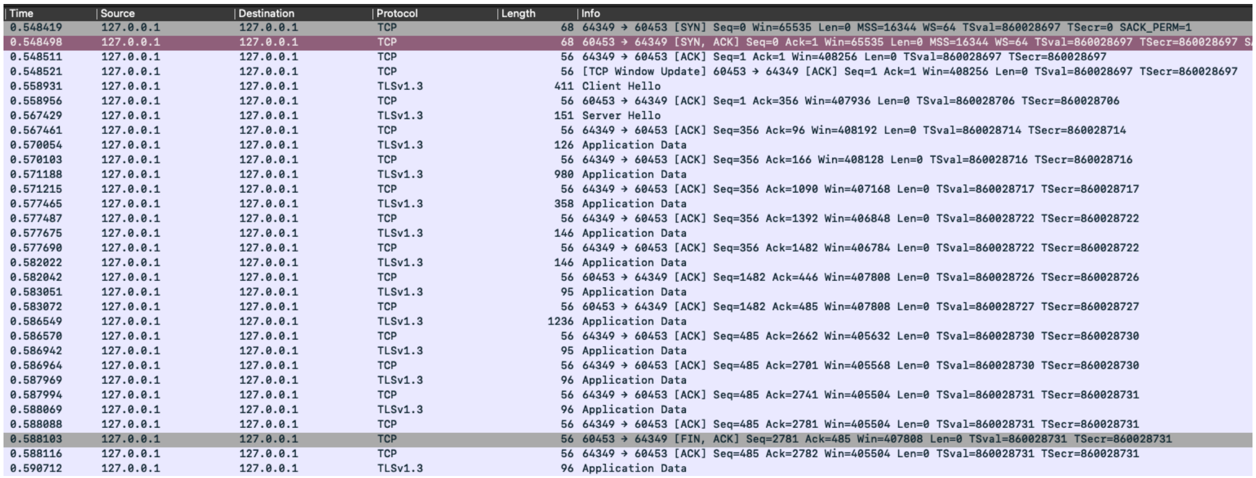
COMP 416: Computer Networks
Project 2

Hasan Can Aslan – 60453

Part 1A. SSL Implementation and Experiments

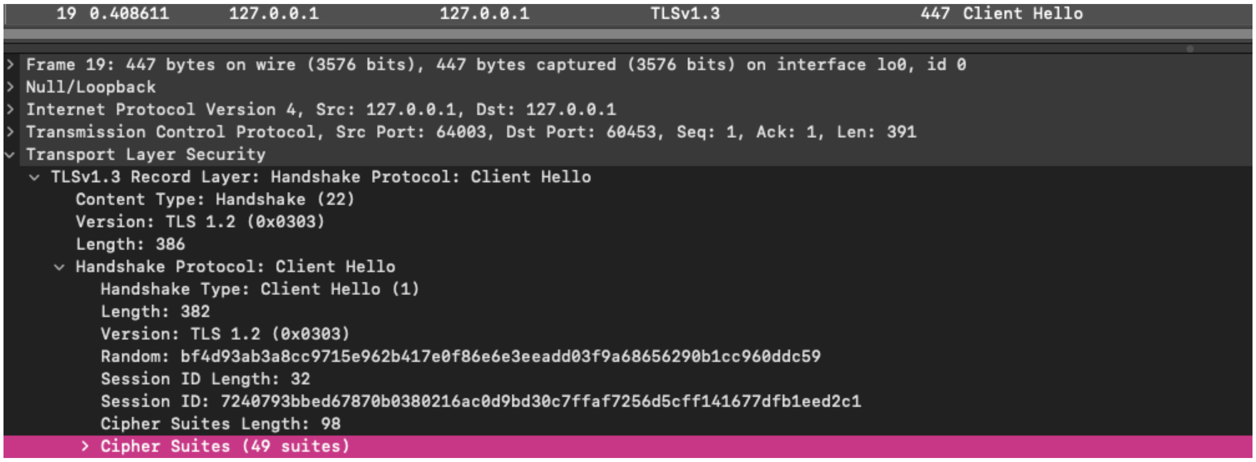
Question 1. For each number, transmitted packets as follows:



Time	Source	Destination	Protocol	Length	Info
0.548419	127.0.0.1	127.0.0.1	TCP	68	64349 → 60453 [SYN] Seq=0 Win=65535 Len=0 MSS=16344 WS=64 TSval=860028697 TSecr=0 SACK_PERM=1
0.548498	127.0.0.1	127.0.0.1	TCP	68	60453 → 64349 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=16344 WS=64 TSval=860028697 TSecr=860028697 S
0.548511	127.0.0.1	127.0.0.1	TCP	56	64349 → 60453 [ACK] Seq=1 Ack=1 Win=408256 Len=0 TSval=860028697 TSecr=860028697
0.548521	127.0.0.1	127.0.0.1	TCP	56	[TCP Window Update] 60453 → 64349 [ACK] Seq=1 Ack=1 Win=408256 Len=0 TSval=860028697 TSecr=860028697
0.558931	127.0.0.1	127.0.0.1	TLSv1.3	411	Client Hello
0.558956	127.0.0.1	127.0.0.1	TCP	56	60453 → 64349 [ACK] Seq=1 Ack=356 Win=407936 Len=0 TSval=860028706 TSecr=860028706
0.567429	127.0.0.1	127.0.0.1	TLSv1.3	151	Server Hello
0.567461	127.0.0.1	127.0.0.1	TCP	56	64349 → 60453 [ACK] Seq=356 Ack=96 Win=408192 Len=0 TSval=860028714 TSecr=860028714
0.570054	127.0.0.1	127.0.0.1	TLSv1.3	126	Application Data
0.570103	127.0.0.1	127.0.0.1	TCP	56	64349 → 60453 [ACK] Seq=356 Ack=166 Win=408128 Len=0 TSval=860028716 TSecr=860028716
0.571188	127.0.0.1	127.0.0.1	TLSv1.3	980	Application Data
0.571215	127.0.0.1	127.0.0.1	TCP	56	64349 → 60453 [ACK] Seq=356 Ack=1090 Win=407168 Len=0 TSval=860028717 TSecr=860028717
0.577465	127.0.0.1	127.0.0.1	TLSv1.3	358	Application Data
0.577487	127.0.0.1	127.0.0.1	TCP	56	64349 → 60453 [ACK] Seq=356 Ack=1392 Win=406848 Len=0 TSval=860028722 TSecr=860028722
0.577675	127.0.0.1	127.0.0.1	TLSv1.3	146	Application Data
0.577690	127.0.0.1	127.0.0.1	TCP	56	64349 → 60453 [ACK] Seq=356 Ack=1482 Win=406784 Len=0 TSval=860028722 TSecr=860028722
0.582022	127.0.0.1	127.0.0.1	TLSv1.3	146	Application Data
0.582042	127.0.0.1	127.0.0.1	TCP	56	60453 → 64349 [ACK] Seq=1482 Ack=446 Win=407808 Len=0 TSval=860028726 TSecr=860028726
0.583051	127.0.0.1	127.0.0.1	TLSv1.3	95	Application Data
0.583072	127.0.0.1	127.0.0.1	TCP	56	60453 → 64349 [ACK] Seq=1482 Ack=485 Win=407808 Len=0 TSval=860028727 TSecr=860028727
0.586549	127.0.0.1	127.0.0.1	TLSv1.3	1236	Application Data
0.586570	127.0.0.1	127.0.0.1	TCP	56	64349 → 60453 [ACK] Seq=485 Ack=2662 Win=405632 Len=0 TSval=860028730 TSecr=860028730
0.586942	127.0.0.1	127.0.0.1	TLSv1.3	95	Application Data
0.586964	127.0.0.1	127.0.0.1	TCP	56	64349 → 60453 [ACK] Seq=485 Ack=2701 Win=405568 Len=0 TSval=860028730 TSecr=860028730
0.587969	127.0.0.1	127.0.0.1	TLSv1.3	96	Application Data
0.587994	127.0.0.1	127.0.0.1	TCP	56	64349 → 60453 [ACK] Seq=485 Ack=2741 Win=405504 Len=0 TSval=860028731 TSecr=860028731
0.588069	127.0.0.1	127.0.0.1	TLSv1.3	96	Application Data
0.588088	127.0.0.1	127.0.0.1	TCP	56	64349 → 60453 [ACK] Seq=485 Ack=2781 Win=405504 Len=0 TSval=860028731 TSecr=860028731
0.588103	127.0.0.1	127.0.0.1	TCP	56	60453 → 64349 [FIN, ACK] Seq=2781 Ack=485 Win=407808 Len=0 TSval=860028731 TSecr=860028731
0.588116	127.0.0.1	127.0.0.1	TCP	56	64349 → 60453 [ACK] Seq=485 Ack=2782 Win=405504 Len=0 TSval=860028731 TSecr=860028731
0.590712	127.0.0.1	127.0.0.1	TLSv1.3	96	Application Data

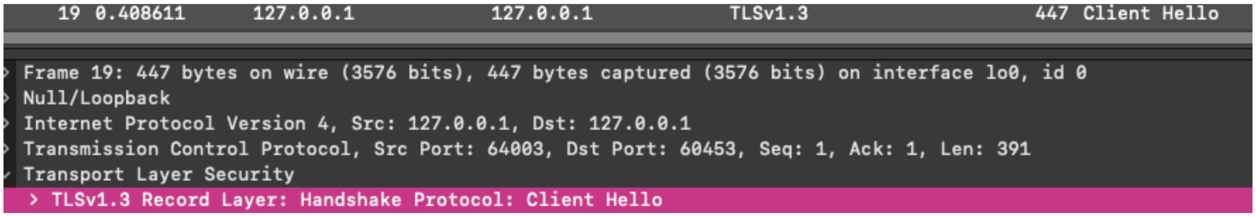
18 TCP, 13 TLSv1.3 packets transmitted for each number. TLS requires TCP, so in total 31 TCP packets are transmitted for each number. Therefore, 155 TCP packets are transmitted in total while my KUSIS ID number is exchanged.

Question 2. My client supports 49 cipher suites. It can be found in the "Client Hello" message.



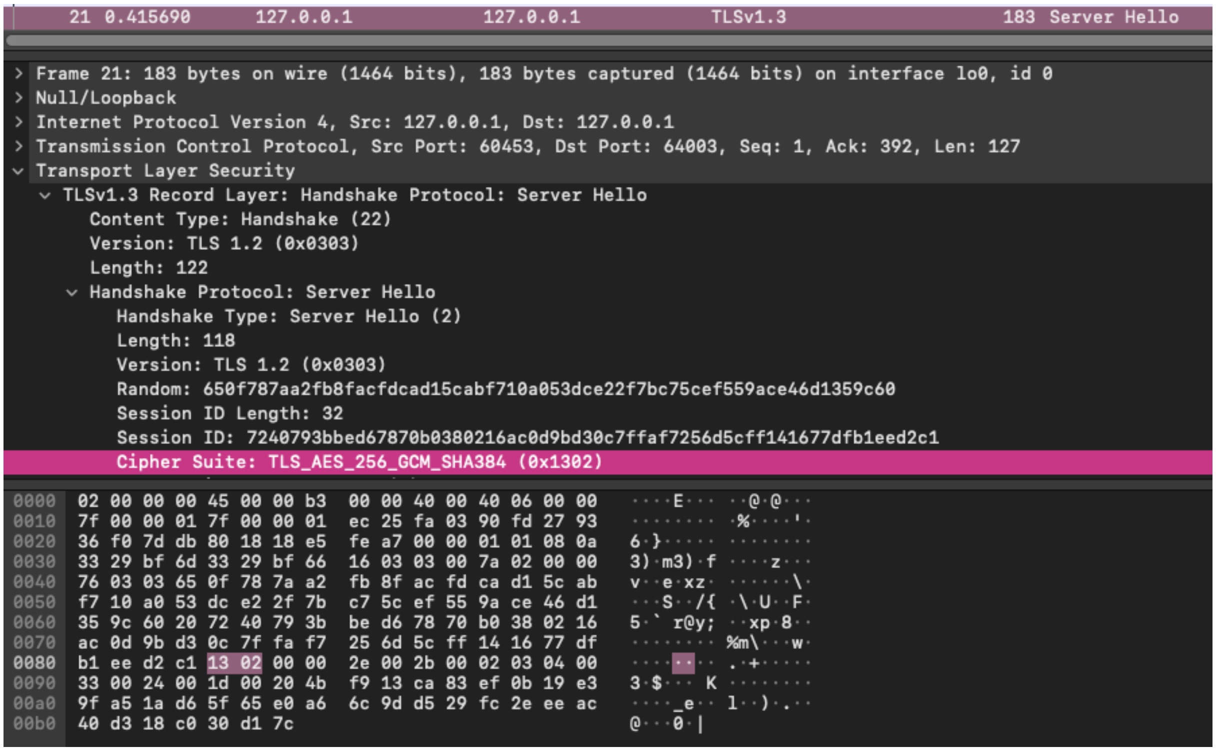
```
   19 0.408611      127.0.0.1          127.0.0.1          TLSv1.3           447 Client Hello

> Frame 19: 447 bytes on wire (3576 bits), 447 bytes captured (3576 bits) on interface lo0, id 0
> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 64003, Dst Port: 60453, Seq: 1, Ack: 1, Len: 391
v Transport Layer Security
  v TLSv1.3 Record Layer: Handshake Protocol: Client Hello
      Content Type: Handshake (22)
      Version: TLS 1.2 (0x0303)
      Length: 386
    v Handshake Protocol: Client Hello
        Handshake Type: Client Hello (1)
        Length: 382
        Version: TLS 1.2 (0x0303)
        Random: bf4d93ab3a8cc9715e962b417e0f86e6e3eeadd03f9a68656290b1cc960ddc59
        Session ID Length: 32
        Session ID: 7240793bbed67870b0380216ac0d9bd30c7ffaf7256d5cff141677dfb1eed2c1
        Cipher Suites Length: 98
      > Cipher Suites (49 suites)
```

Question 3. My client's first message "Client Hello" indicates the last supported version, which is TLSv1.3. So, my client supports all TLS versions up to v1.3.



```
   19 0.408611      127.0.0.1          127.0.0.1          TLSv1.3           447 Client Hello

> Frame 19: 447 bytes on wire (3576 bits), 447 bytes captured (3576 bits) on interface lo0, id 0
> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 64003, Dst Port: 60453, Seq: 1, Ack: 1, Len: 391
v Transport Layer Security
  > TLSv1.3 Record Layer: Handshake Protocol: Client Hello
```

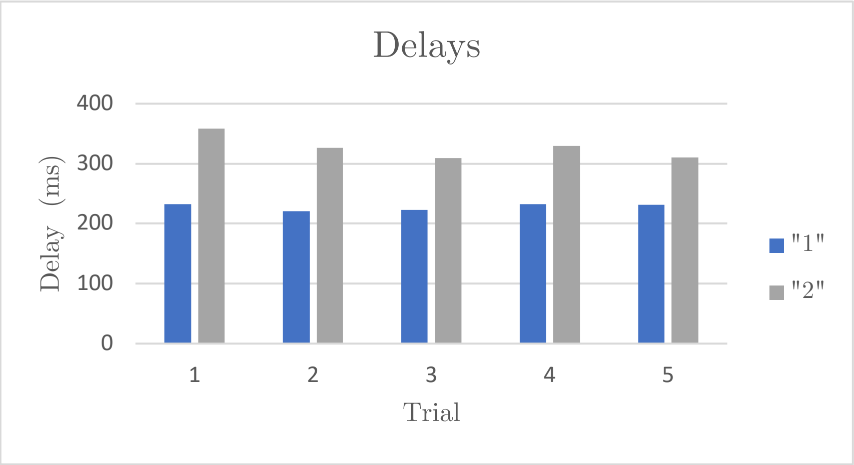
Question 4. Server will be using TLS_AES_256_GCM_SHA384 cipher suite. Hex dump of key is 0x1302.



```
   21 0.415690       127.0.0.1             127.0.0.1             TLSv1.3                    183 Server Hello

> Frame 21: 183 bytes on wire (1464 bits), 183 bytes captured (1464 bits) on interface lo0, id 0
> Null/Loopback
> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 60453, Dst Port: 64003, Seq: 1, Ack: 392, Len: 127
v Transport Layer Security
  v TLSv1.3 Record Layer: Handshake Protocol: Server Hello
      Content Type: Handshake (22)
      Version: TLS 1.2 (0x0303)
      Length: 122
    v Handshake Protocol: Server Hello
        Handshake Type: Server Hello (2)
        Length: 118
        Version: TLS 1.2 (0x0303)
        Random: 650f787aa2fb8facfdcad15cabf710a053dce22f7bc75cef559ace46d1359c60
        Session ID Length: 32
        Session ID: 7240793bbed67870b0380216ac0d9bd30c7ffaf7256d5cff141677dfb1eed2c1
        Cipher Suite: TLS_AES_256_GCM_SHA384 (0x1302)

0000  02 00 00 00 45 00 00 b3  00 00 40 00 40 06 00 00   ····E···  ··@·@···
0010  7f 00 00 01 7f 00 00 01  ec 25 fa 03 90 fd 27 93   ········  ·%····'·
0020  36 f0 7d db 80 18 18 e5  fe a7 00 00 01 01 08 0a   6·}·····  ········
0030  33 29 bf 6d 33 29 bf 66  16 03 03 00 7a 02 00 00   3)·m3)·f  ····z···
0040  76 03 03 65 0f 78 7a a2  fb 8f ac fd ca d1 5c ab   v··e·xz·  ······\·
0050  f7 10 a0 53 dc e2 2f 7b  c7 5c ef 55 9a ce 46 d1   ···S··/{  ·\·U··F·
0060  35 9c 60 20 72 40 79 3b  be d6 78 70 b0 38 02 16   5·` r@y;  ··xp·8··
0070  ac 0d 9b d3 0c 7f fa f7  25 6d 5c ff 14 16 77 df   ········  %m\···w·
0080  b1 ee d2 c1 13 02 00 00  2e 00 2b 00 02 03 04 00   ········  .·+·····
0090  33 00 24 00 1d 00 20 4b  f9 13 ca 83 ef 0b 19 e3   3·$··· K  ········
00a0  9f a5 1a d6 5f 65 e0 a6  6c 9d d5 29 fc 2e ee ac   ····_e··  l··)·.··
00b0  40 d3 18 c0 30 d1 7c                               @···0·|
```

Part 1B. SSL vs TCP: Delay Measurements

Question 5. I get timestamp both sending the request and after receiving the server response. This graph shows the time difference of these timestamps for each message "1" and "2" in 5 trials.

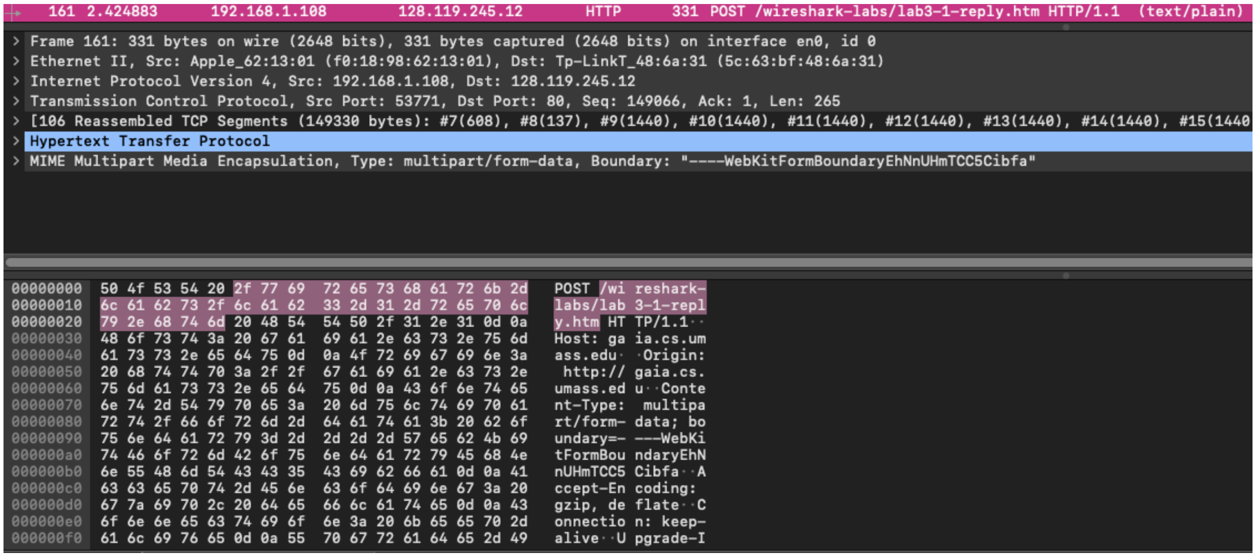


	"1"	"2"
1	233ms	358ms
2	221ms	326ms
3	223ms	309ms
4	232ms	330ms
5	231ms	310ms

Part 2. TCP Experiments

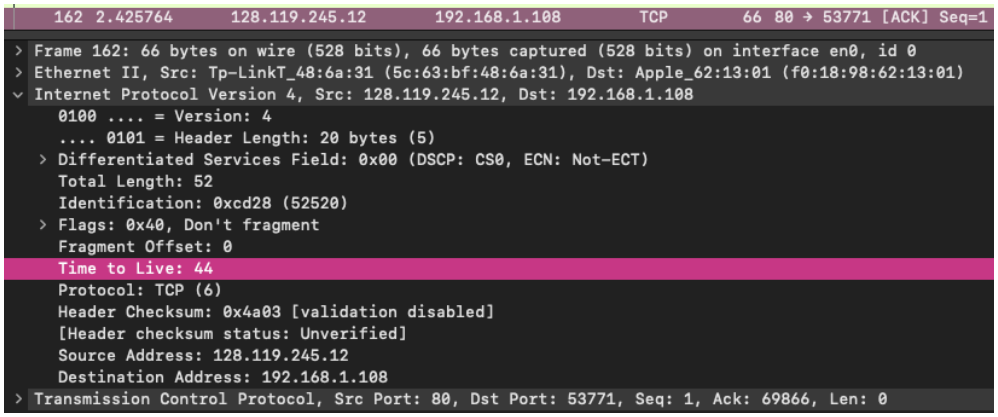
Question 6. When a server starts a TCP connection, it assigns a random initial sequence number in range 0 and 2^{32}. Nonetheless, Wireshark displays relative sequence number instead of number assigned from host. Relative ACK Number is the number that Wireshark displays that relative to initial sequence number. In this way, we can keep track of sequence numbers easily.

Question 7. 149066.



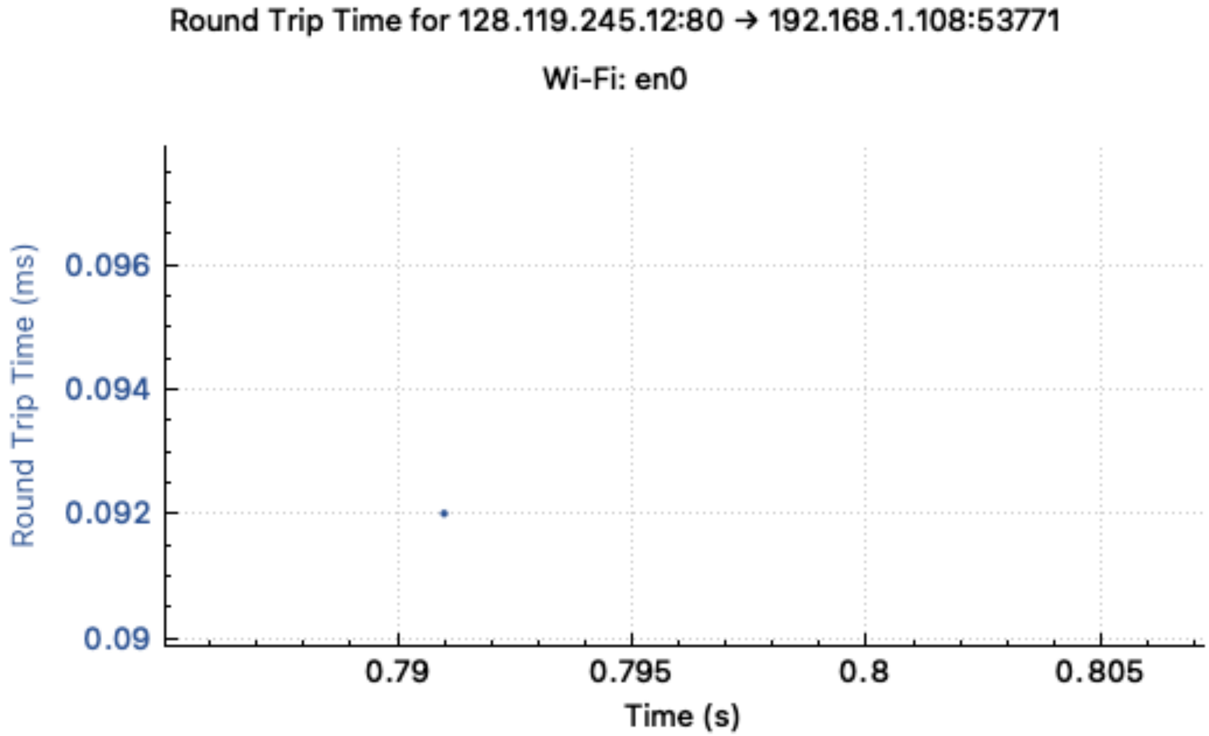
```
  161  2.424883       192.168.1.108         128.119.245.12        HTTP       331  POST /wireshark-labs/lab3-1-reply.htm HTTP/1.1  (text/plain)
> Frame 161: 331 bytes on wire (2648 bits), 331 bytes captured (2648 bits) on interface en0, id 0
> Ethernet II, Src: Apple_62:13:01 (f0:18:98:62:13:01), Dst: Tp-LinkT_48:6a:31 (5c:63:bf:48:6a:31)
> Internet Protocol Version 4, Src: 192.168.1.108, Dst: 128.119.245.12
> Transmission Control Protocol, Src Port: 53771, Dst Port: 80, Seq: 149066, Ack: 1, Len: 265
> [106 Reassembled TCP Segments (149330 bytes): #7(608), #8(137), #9(1440), #10(1440), #11(1440), #12(1440), #13(1440), #14(1440), #15(1440
> Hypertext Transfer Protocol
> MIME Multipart Media Encapsulation, Type: multipart/form-data, Boundary: "----WebKitFormBoundaryEhNnUHmTCC5Cibfa"

00000000  50 4f 53 54 20 2f 77 69  72 65 73 68 61 72 6b 2d   POST /wi reshark-
00000010  6c 61 62 73 2f 6c 61 62  33 2d 31 2d 72 65 70 6c   labs/lab 3-1-repl
00000020  79 2e 68 74 6d 20 48 54  54 50 2f 31 2e 31 0d 0a   y.htm HT TP/1.1··
00000030  48 6f 73 74 3a 20 67 61  69 61 2e 63 73 2e 75 6d   Host: ga ia.cs.um
00000040  61 73 73 2e 65 64 75 0d  0a 4f 72 69 67 69 6e 3a   ass.edu· ·Origin:
00000050  20 68 74 74 70 3a 2f 2f  67 61 69 61 2e 63 73 2e    http:// gaia.cs.
00000060  75 6d 61 73 73 2e 65 64  75 0d 0a 43 6f 6e 74 65   umass.ed u··Conte
00000070  6e 74 2d 54 79 70 65 3a  20 6d 75 6c 74 69 70 61   nt-Type:  multipa
00000080  72 74 2f 66 6f 72 6d 2d  64 61 74 61 3b 20 62 6f   rt/form- data; bo
00000090  75 6e 64 61 72 79 3d 2d  2d 2d 2d 57 65 62 4b 69   undary=- ---WebKi
000000a0  74 46 6f 72 6d 42 6f 75  6e 64 61 72 79 45 68 4e   tFormBou ndaryEhN
000000b0  6e 55 48 6d 54 43 43 35  43 69 62 66 61 0d 0a 41   nUHmTCC5 Cibfa··A
000000c0  63 63 65 70 74 2d 45 6e  63 6f 64 69 6e 67 3a 20   ccept-En coding: 
000000d0  67 7a 69 70 2c 20 64 65  66 6c 61 74 65 0d 0a 43   gzip, de flate··C
000000e0  6f 6e 6e 65 63 74 69 6f  6e 3a 20 6b 65 65 70 2d   onnectio n: keep-
000000f0  61 6c 69 76 65 0d 0a 55  70 67 72 61 64 65 2d 49   alive··U pgrade-I
```

Question 8. 44.



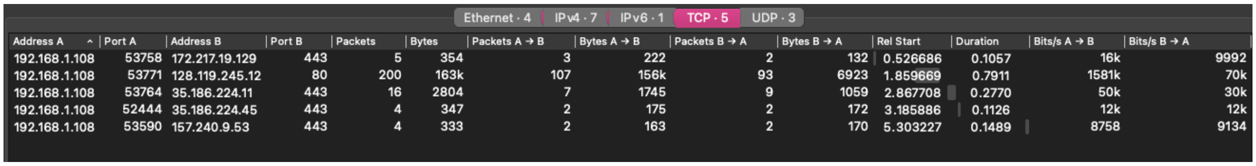
```
  162  2.425764       128.119.245.12        192.168.1.108         TCP        66  80 → 53771 [ACK] Seq=1
> Frame 162: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface en0, id 0
> Ethernet II, Src: Tp-LinkT_48:6a:31 (5c:63:bf:48:6a:31), Dst: Apple_62:13:01 (f0:18:98:62:13:01)
v Internet Protocol Version 4, Src: 128.119.245.12, Dst: 192.168.1.108
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 52
    Identification: 0xcd28 (52520)
  > Flags: 0x40, Don't fragment
    Fragment Offset: 0
    Time to Live: 44
    Protocol: TCP (6)
    Header Checksum: 0x4a03 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 128.119.245.12
    Destination Address: 192.168.1.108
> Transmission Control Protocol, Src Port: 80, Dst Port: 53771, Seq: 1, Ack: 69866, Len: 0
```

Question 9. Same for all 6, 0.092 ms.



$$\text{EstimatedRTT} = (1 - \alpha) \bullet \text{EstimatedRTT} + \alpha \bullet \text{SampleRTT}$$

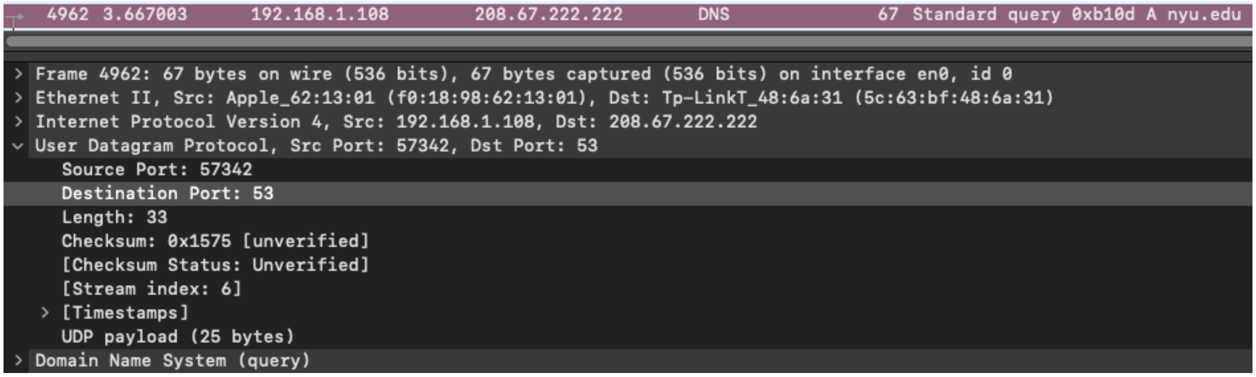
Question 10. Stream index in the TCP header identifies unique TCP stream. It is an internal mapping in Wireshark.



Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
192.168.1.108	53758	172.217.19.129	443	5	354	3	222	2	132	0.526686	0.1057	16k	9992
192.168.1.108	53771	128.119.245.12	80	200	163k	107	156k	93	6923	1.859669	0.7911	1581k	70k
192.168.1.108	53764	35.186.224.11	443	16	2804	7	1745	9	1059	2.867708	0.2770	50k	30k
192.168.1.108	52444	35.186.224.45	443	4	347	2	175	2	172	3.185886	0.1126	12k	12k
192.168.1.108	53590	157.240.9.53	443	4	333	2	163	2	170	5.303227	0.1489	8758	9134

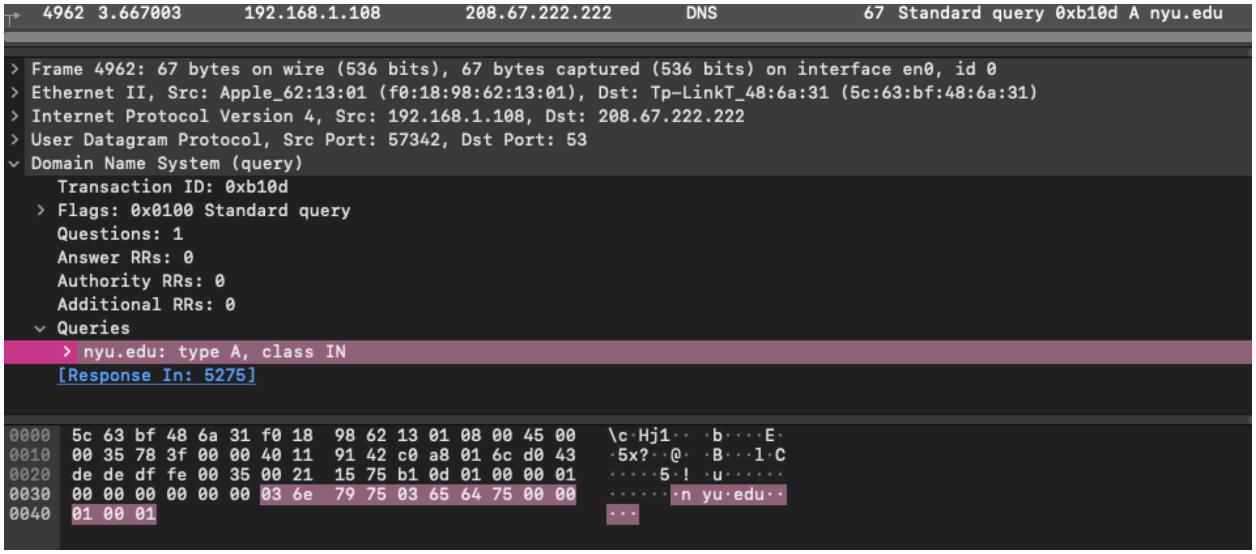
Part 2. UDP Experiments

Question 11. Source socket address is my IP address and port number which is 192.168.1.108:57342 and destination is OpenDNS server IP address and port number which is 208.67.222.222:53. We can find this information under DNS header.



```
  4962  3.667003      192.168.1.108         208.67.222.222        DNS            67  Standard query 0xb10d A nyu.edu

> Frame 4962: 67 bytes on wire (536 bits), 67 bytes captured (536 bits) on interface en0, id 0
> Ethernet II, Src: Apple_62:13:01 (f0:18:98:62:13:01), Dst: Tp-LinkT_48:6a:31 (5c:63:bf:48:6a:31)
> Internet Protocol Version 4, Src: 192.168.1.108, Dst: 208.67.222.222
v User Datagram Protocol, Src Port: 57342, Dst Port: 53
    Source Port: 57342
    Destination Port: 53
    Length: 33
    Checksum: 0x1575 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 6]
  > [Timestamps]
    UDP payload (25 bytes)
> Domain Name System (query)
```

Question 12.

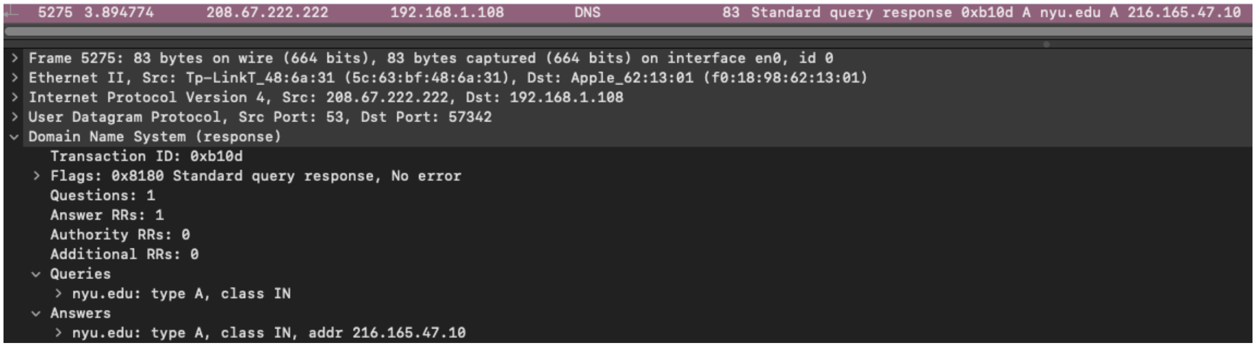


```
  4962  3.667003      192.168.1.108         208.67.222.222        DNS            67  Standard query 0xb10d A nyu.edu

> Frame 4962: 67 bytes on wire (536 bits), 67 bytes captured (536 bits) on interface en0, id 0
> Ethernet II, Src: Apple_62:13:01 (f0:18:98:62:13:01), Dst: Tp-LinkT_48:6a:31 (5c:63:bf:48:6a:31)
> Internet Protocol Version 4, Src: 192.168.1.108, Dst: 208.67.222.222
> User Datagram Protocol, Src Port: 57342, Dst Port: 53
v Domain Name System (query)
    Transaction ID: 0xb10d
  > Flags: 0x0100 Standard query
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  v Queries
    > nyu.edu: type A, class IN
    [Response In: 5275]

0000  5c 63 bf 48 6a 31 f0 18  98 62 13 01 08 00 45 00   \c·Hj1··  ·b····E·
0010  00 35 78 3f 00 00 40 11  91 42 c0 a8 01 6c d0 43   ·5x?··@·  ·B···l·C
0020  de de df fe 00 35 00 21  15 75 b1 0d 01 00 00 01   ·····5·!  ·u······
0030  00 00 00 00 00 00 03 6e  79 75 03 65 64 75 00 00   ·······n  yu·edu··
0040  01 00 01                                           ···
```

Question 13. I connected to local DNS server works in recursive manner because my client asks to local DNS server for IP address corresponding to nyu.edu. If it works in iterative manner, it should connect to "." Which is root name server then, .edu server, contacts next name server up to the find requested IP address.

Recursive DNS is much faster, but it is vulnerable to attacks and unauthorized usage. In recursive, client only send query to 1st server.

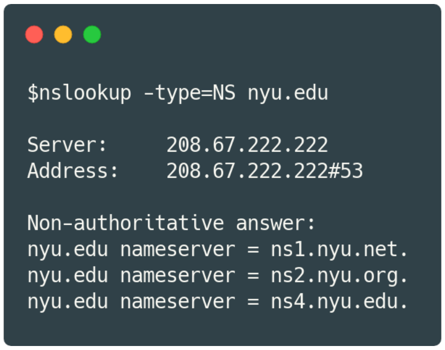
Question 14. I get 1 response to my request. I get A Record for IP. It is public IP address and uses IPv4. It points to domain of IP address.



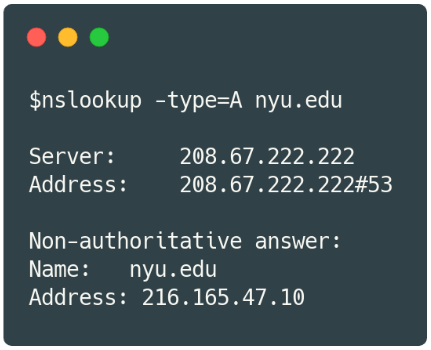
```
  5275 3.894774      208.67.222.222        192.168.1.108         DNS               83 Standard query response 0xb10d A nyu.edu A 216.165.47.10

> Frame 5275: 83 bytes on wire (664 bits), 83 bytes captured (664 bits) on interface en0, id 0
> Ethernet II, Src: Tp-LinkT_48:6a:31 (5c:63:bf:48:6a:31), Dst: Apple_62:13:01 (f0:18:98:62:13:01)
> Internet Protocol Version 4, Src: 208.67.222.222, Dst: 192.168.1.108
> User Datagram Protocol, Src Port: 53, Dst Port: 57342
v Domain Name System (response)
    Transaction ID: 0xb10d
  > Flags: 0x8180 Standard query response, No error
    Questions: 1
    Answer RRs: 1
    Authority RRs: 0
    Additional RRs: 0
  v Queries
    > nyu.edu: type A, class IN
  v Answers
    > nyu.edu: type A, class IN, addr 216.165.47.10
```

Question 15. Most common DNS types are: Mail exchanger record (MX Record), Canonical Name record (CNAME Record), Address Mapping record (A Record), Mail exchanger record (MX Record), Text Record (TXT Record), Name Server records (NS Record).

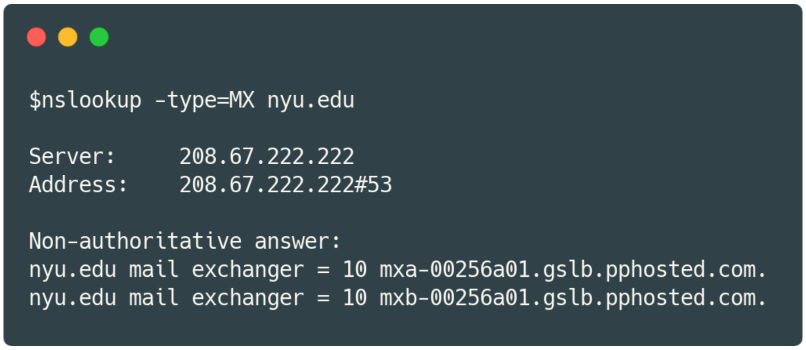
When using `nslookup` we can specify the type of DNS record using `-type` flag. For example, `nslookup -type=NS www.nyu.edu`.



```
$nslookup -type=NS nyu.edu

Server:      208.67.222.222
Address:     208.67.222.222#53

Non-authoritative answer:
nyu.edu nameserver = ns1.nyu.net.
nyu.edu nameserver = ns2.nyu.org.
nyu.edu nameserver = ns4.nyu.edu.
```



```
$nslookup -type=A nyu.edu

Server:      208.67.222.222
Address:     208.67.222.222#53

Non-authoritative answer:
Name:    nyu.edu
Address: 216.165.47.10
```



```
$nslookup -type=MX nyu.edu

Server:      208.67.222.222
Address:     208.67.222.222#53

Non-authoritative answer:
nyu.edu mail exchanger = 10 mxa-00256a01.gslb.pphosted.com.
nyu.edu mail exchanger = 10 mxb-00256a01.gslb.pphosted.com.
```