ECSE308 Lab 4

IP & TCP

Zhanyue Zhang:260944809

Zhiheng Zhou: 260955157

ECSE308

Professor Tho Le-Ngoc

2023-11-15

Lab 4: IP & TCP

Part 1: Internet Protocol (IP)

Questions:

Q1: How many ICMP packets are in the list plane?

Answer: There are 64 packets in the list plane.

Q2: How many probe packets are sent from the source to the destination for each TTL?

Answer: Three probe packets are sent from the source to the destination for each TTL.

Q3: The last few echo-request ICMP packets are followed by the echo-reply ICMP packets. Compare one of them with the corresponding reply. Determine which fields are similar and which fields are different? Explain the reason.

Answer:
We compared one of the echo-request ICMP packets and its corresponding echo-reply ICMP packets, there are several key fields to pay attention to:

1. Header length and total length: They have the same header length (20 bytes) and the same total length (92).
   The consistent header length is expected for ICMP packets.
2. Protocol: They both use ICMP protocol (1).
   This is consistent for echo-request and echo-reply as they are part of the ICMP protocol.
3. Code: 0 for both of them.
   This is expected for echo-request and echo-reply in ICMP.
4. Identifier: They both have 1 (0x0001) and 256 (0x0100) as their BE and LE identifiers.
5. Sequence: They have the same sequence number BE 31 (0x001f) and sequence number LE 7936 (0x1f00)
   The identifier and sequence numbers are used to match echo requests with their corresponding replies. Hence their identifiers and sequence numbers are the same.
6. Data: 0 for both of them (64-bit)
   In our case, the payload appears to be empty for both of them. Both the ICMP echo-request and echo-reply have a data field value of 0 indicating an empty payload.
7. Identification:
   - ICMP echo-request: 0x003A
   - ICMP echo-reply: 0x2EAD

The different Identification values between ICMP echo-request and echo-reply are designed to facilitate the proper correlation of requests and replies in network communication.

8. TTL:
    - ICMP echo-request: 11
    - ICMP echo-reply: 54

    The difference in TTL values suggests that the echo-reply has traveled through more routers than the echo-request.
9. Source address and destination address:
    - ICMP echo-request:
      Source Address: 10.10.84.226; Destination Address: 104.17.79.30
    - ICMP echo-reply:
      Source Address: 104.17.79.30; Destination Address: 10.10.84.226

    We noticed a reversal of source and destination addresses in the ICMP echo-reply compared to the echo-request. This reversal of addresses is a convention that helps establish a clear association between the echo request and its corresponding echo reply. It facilitates the identification of the sender and recipient in both messages and allows for straightforward tracking of the communication flow.
10. Type:
    - ICMP echo-request: 8 (Echo (ping) request)
    - ICMP echo-reply: 0 (Echo (ping) reply)

    The type field distinguishes between different ICMP message types.
11. Checksum:
    - ICMP echo-request: 0xf7df
    - ICMP echo-reply: 0xffdf

    The checksum is used for error-checking. Checksums are different for the echo-request and echo-replay due to changes in the packet during transmission.

Q4: What are the TTL values for these last few packets? Determine the number of routers between the source and destination based on these TTL values.

Answer:
TTL values for echo-request packets are 11, and for echo-reply packets are 54. Thus, the number of routers is 54-11=43. There are approximately 43 routers or hops between the source and destination for these ICMP packets

Q5: Examine the IP packet header of the last echo-request ICMP packet, what is the value in the "Protocol" field? What does this field indicate?

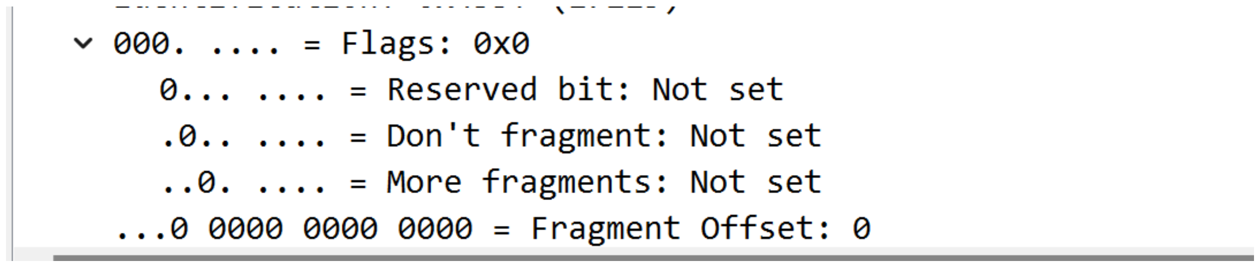
Answer:
In the last echo-request ICMP packet, its protocol value is 1. The "Protocol" field indicates the next level protocol in the data payload of the IP packet. In our case, it indicates that the payload of the IP packet contains ICMP data.

Q6: How many bytes are in this IP header? How many bytes are in the payload of this IP packet? Explain how you determined the number of payload bytes

Answer:
There are 5 * 4 bytes = 20 bytes in this IP header. The number of bytes in the payload can be calculated by subtracting the header length from the total length: 92-20=72 bytes.

Q7: Has this IP packet been fragmented? Explain how you determined whether or not the packet has been fragmented.

Answer:
This IP packet has not been fragmented. By checking its fragment offset (Fig. 1), we observed its fragment offset is 0. Moreover, we noticed the 'Don't Fragment' and 'More Fragments' are not set, which indicates that this IP packet has not been fragmented.

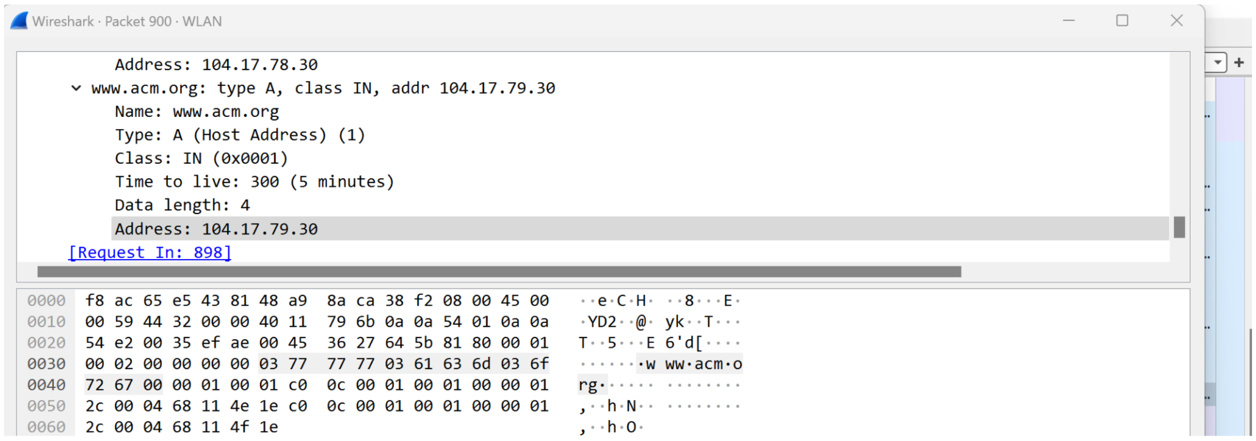


```
⌄ 000. .... = Flags: 0x0
    0... .... = Reserved bit: Not set
    .0.. .... = Don't fragment: Not set
    ..0. .... = More fragments: Not set
  ...0 0000 0000 0000 = Fragment Offset: 0
```

Fig. 1. Fragment Offset

Q8: How the IP address of www.acm.org can be found? Determine the packet and the filed in the packet that contains this information.

Answer:
We used the Domain Name System (DNS) to find the IP address of www.acm.org (Fig. 2), which is 104.17.79.30.



Wireshark · Packet 900 · WLAN

```
        Address: 104.17.78.30
    ⌄ www.acm.org: type A, class IN, addr 104.17.79.30
        Name: www.acm.org
        Type: A (Host Address) (1)
        Class: IN (0x0001)
        Time to live: 300 (5 minutes)
        Data length: 4
        Address: 104.17.79.30
    [Request In: 898]
```

```
0000  f8 ac 65 e5 43 81 48 a9  8a ca 38 f2 08 00 45 00   ··e·C·H· ··8···E·
0010  00 59 44 32 00 00 40 11  79 6b 0a 0a 54 01 0a 0a   ·YD2··@· yk··T···
0020  54 e2 00 35 ef ae 00 45  36 27 64 5b 81 80 00 01   T··5···E 6'd[····
0030  00 02 00 00 00 00 03 77  77 77 03 61 63 6d 03 6f   ·······w ww·acm·o
0040  72 67 00 00 01 00 01 c0  0c 00 01 00 01 00 00 01   rg······ ········
0050  2c 00 04 68 11 4e 1e c0  0c 00 01 00 01 00 00 01   ,··h·N·· ········
0060  2c 00 04 68 11 4f 1e                               ,··h·O·
```

Fig. 2. IP address of www.acm.org

Part 2: 802.11 frames

Questions:

Q1: What is the SSID of the access point that is issuing the beacon frame?

Answer: The SSID is “Coherer” (0x43 6f 68 65 72 65 72)

Q2: What are the time intervals between transmissions of beacon frames? Does the beacon frame contain this information?

Answer:
In the Beacon frame, we found the Beacon Interval field in the "Beacon Interval" field of the "Fixed Parameters" section. The time interval between transmissions of beacon frames is 0.102400 [Seconds].

Q3: What is the source MAC address in the beacon frame?

Answer: The source Mac address is Cisco-Li_82:b2:55 (00:0c:41:82:b2:55)

Q4: What is the destination MAC address in the beacon frame? What does this address mean?

Answer:
The destination MAC address is Broadcast (ff:ff:ff:ff:ff:ff). It indicates that the frame is being broadcasted to all devices within the Wi-Fi network's coverage area.

Q5: How many data rates can the access point support?

Answer:
This access point supports eight data rates. Its supported rates are 1(B), 2(B), 5.5(B), 11(B), 18, 24, 36, 54, [Mbit/sec]

Q6: Examine the beacon frame, what frequency does the advertised network use?

Answer: The advertised network uses 2412MHz frequency.

Q7: By looking at the list plane, indicate what type of packets have the smallest size? What type has the largest size?

Answer:
The control type of packets has the smallest size, and the data type of packets has the largest size.

Q8: Before sending data to Apple_82:36:3a, what frames are exchanged between this device and the access point?

Answer:

1. Probe Request and Probe Response frames are exchanged

   By applying filter *wlan.addr contains 82:36:3a && wlan.fc.type_subtype == 0x04 || (wlan.addr contains 82:36:3a && wlan.fc.type_subtype == 0x05)*, we noticed the frames exchanged before sending data include Probe Request and Probe Response frames.
2. Authentication Request and Authentication Response frames are exchanged

   By applying filter *wlan.addr contains 82:36:3a && wlan.fc.type_subtype == 0x0b || (wlan.addr contains 82:36:3a && wlan.fc.type_subtype == 0x0c)*, we noticed that the frames exchanged before sending data include Authentication Request and Authentication Response frames.
3. Association Request and Association Response frames are exchanged

   By applying filter *wlan.addr contains 82:36:3a && wlan.fc.type_subtype == 0x00 || (wlan.addr contains 82:36:3a && wlan.fc.type_subtype == 0x01)*, we noticed that the frames exchanged before sending data include Association Request and Association Response frames.
4. Reassociation frames are not exchanged

   By applying filter *wlan.addr contains 82:36:3a && wlan.fc.type_subtype == 0x02 || (wlan.addr contains 82:36:3a && wlan.fc.type_subtype == 0x03)*, we noticed that there are no Reassociation frames exchanged between the device with MAC address 82:36:3a and the access point in the captured data.

Q9: Examine the Authentication frame sent by Apple_82:36:3a, does the host want the authentication to require a key or be open?

Answer:
From the information: Authentication Algorithm: Open System (0), we know that the host wants the authentication to be open.

Q10: Examine the response Authentication frame sent by the AP to Apple_82:36:3a. What is the Association ID for this host? What is the usage of this ID?

Answer:
The association ID for this host is: ..00 0000 0000 0001 = Association ID: 0x0001
The usage of the Association ID is to uniquely identify the association of a specific station (in this case, the station with the MAC address "Apple_82:36:3a") with the Access Point or BSS

Q11: What transmission rates is the host willing to use? The AP? To answer this question, you will need to look into the parameters fields of the 802.11 wireless LAN management frame. Examine the beacon frame, what frequency does the advertised network use?

Answer:
Transmission rates that the host is willing to use are 1(B), 2(B), 5.5(B), 11(B), 18, 24, 36, 54, [Mbit/sec].
The AP is willing to use transmission rates 6, 9, 12, 48, [Mbit/sec].
The advertised network uses 2412MHz as its frequency.

Q12: Examine the Disassociation frame sent by Apple_82:36:3a to the AP. What is the reason that this user sent Disassociation frame?

Answer: Reason code: Disassociated because sending STA is leaving (or has left) BSS (0x0008)

Part 3: TCP

Questions:

Q1: How many TCP datagrams are exchanged between your computer and the server to establish the TCP connection? Why each of these segments is needed to set up the TCP connection?

Answer:
In our case, 6 datagrams are sent in total. The client sends a SYN to the server. In response, the server sends a SYN-ACK back when it receives the SYN. Then the client sends back an ACK. Our client and server send the SYN and SYN-ACK two times each before a HTTP response.

Q2: Which end point started the TCP Connection-Establishment phase?

Answer: The client (our computer) started the TCP connection-establishment.

Q3: What flags are set in each of these TCP datagrams?

Answer: Syn flag is set in first TCP datagram, Syn and Acknowledgement is set in [SYN,ACK] datagram, Acknowledgement is set in last [ACK] datagram.

Q4: What is the initial value of the sequence number on the client's side?

Answer: Initial value is 0.

Q5: What is the initial value of the sequence number on the server's side?

Answer: Initial value is 0.

Q6: What is the value of the Acknowledgement field in the SYN ACK datagram? How did the server determine that value?

Answer: Ack value is 1. The server checks the flag, 0 means no acknowledgement and 1 means there is acknowledgement.

)

Q7: For the TCP SYN datagram, determine the following a. the source port number b. the destination port number c. the size of the window d. the header length

Answer:
Source port: 55407;
Destination port:80
Window: 64240
Header length: 32 bytes.

Q8: For the TCP SYN ACK datagram, determine the following a. the source port number b. the destination port number c. the size of the window d. the header length

Answer:
Source port: 80;
Destination port: 55407;
Window: 26883;
Header length: 32 bytes.

Q9: What is the usage of the window field in the TCP segments?

Answer: The TCP window size field controls the flow of data and its value is between 2 and 65535 bytes.

Q10: Consider the TCP segment containing the HTTP GET as the first segment in the TCP connection. For the first three TCP segments, answer the following questions: a. When was each segment sent? b. At what time was the ACK for each segment received? c. Given the difference between when each TCP segment was sent, and when its acknowledgement was received, what is the RTT value for each of the three segments? d. What is the Estimated RTT value after the receipt of each ACK? e. What is the length of each of the first three TCP segments?

Answer:
First sent:8.908394s. ACK received: 9.031754s. RTT= 0.12336s
Second sent: 9.043619s ACK received: 9.169126s. RTT= 0.125507s
Third send: 9.256074s ACK received: 9.382859s. RTT =0.126785s

Estimated RTT1 = 0.12336s
Estimated RTT2 = 0.875*0.12336s + 0.125*0.125507 = 0.1236s
Estimated RTT3 = 0.875*0.1236 + 0.125*0.126785 = 0.12399s

Total length for first segment: 499
Total length for second segment: 500

Total length for second segment: 444

Q11: Are the client's port number and the server's port number the same in the entire trace? What is the usage of the port number?

Answer: Client is always in 80 ports while the server's port number has changed over the entire trace. Port numbers are part of the addressing information that helps identify senders and receivers of information and a particular application on the devices.

Q12: What is the minimum amount of available buffer space advertised at the received for the entire trace? Does the lack of receiver buffer space ever throttle the sender?

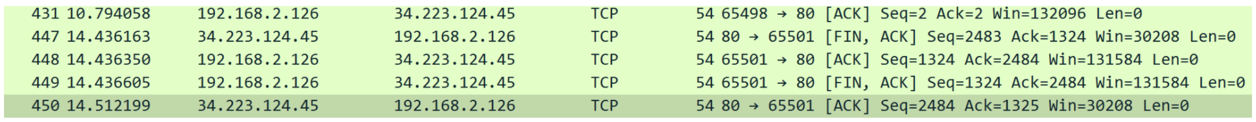
Answer: Minimum amount of buffer space advertised and neverssl.com is 26883 bytes by inspecting the window size of the first ACK from the server. Sender is never throttled by looking at the trace.

Q13: Are there any retransmitted segments in the trace file? What did you check for (in the trace) in order to answer this question?

Answer: There are no retransmitted segments since the sequence number from the source to the destination are increasing monotonically by looking at the trace file.

Q14: How much data does the receiver typically acknowledge in an ACK? Can you identify cases where the receiver is ACKing every other received segment.

Answer:
The difference between two acknowledged sequence numbers is the acknowledged data.
For the first few ACKs, the data size is 460 bytes.
We can find in the last ACK, ACK = 920 = 460*2.

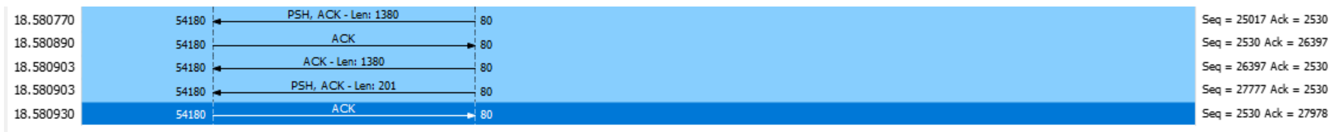


```
431 10.794058    192.168.2.126    34.223.124.45    TCP    54 65498 → 80 [ACK] Seq=2 Ack=2 Win=132096 Len=0
447 14.436163    34.223.124.45    192.168.2.126    TCP    54 80 → 65501 [FIN, ACK] Seq=2483 Ack=1324 Win=30208 Len=0
448 14.436350    192.168.2.126    34.223.124.45    TCP    54 65501 → 80 [ACK] Seq=1324 Ack=2484 Win=131584 Len=0
449 14.436605    192.168.2.126    34.223.124.45    TCP    54 65501 → 80 [FIN, ACK] Seq=1324 Ack=2484 Win=131584 Len=0
450 14.512199    34.223.124.45    192.168.2.126    TCP    54 80 → 65501 [ACK] Seq=2484 Ack=1325 Win=30208 Len=0
```

.

Q15: Calculate the throughput (bytes transferred per unit time) for the TCP connection? Explain how you obtained this value.

Answer: The time between the first HTTP get request and the last ACK is 18.580930 - 10.022 = 8.5589s. The total number of bytes transmitted are 25978 - 1 = 25977 bytes. Hence the total throughput is
25977 / 8.5589 = 3035 bytes/s. We can find the values by looking at the first and last TCP segment.



```
18.580770   54180 <-- PSH, ACK - Len: 1380 -- 80   Seq = 25017 Ack = 2530
18.580890   54180 -- ACK --------------------> 80   Seq = 2530 Ack = 26397
18.580903   54180 <-- ACK - Len: 1380 ------- 80   Seq = 26397 Ack = 2530
18.580903   54180 <-- PSH, ACK - Len: 201 --- 80   Seq = 27777 Ack = 2530
18.580930   54180 -- ACK --------------------> 80   Seq = 2530 Ack = 27978
```

Q16: How many TCP datagrams are exchanged for the termination phase?

Answer: During the termination, there are 4 datagrams exchanged. Firstly, the client sends the Fin, and the server replies with Ack. Then the server sends a Fin, and the client replies with a Ack.

Q17: Which end point started the Connection Termination phase?

Answer: Client.

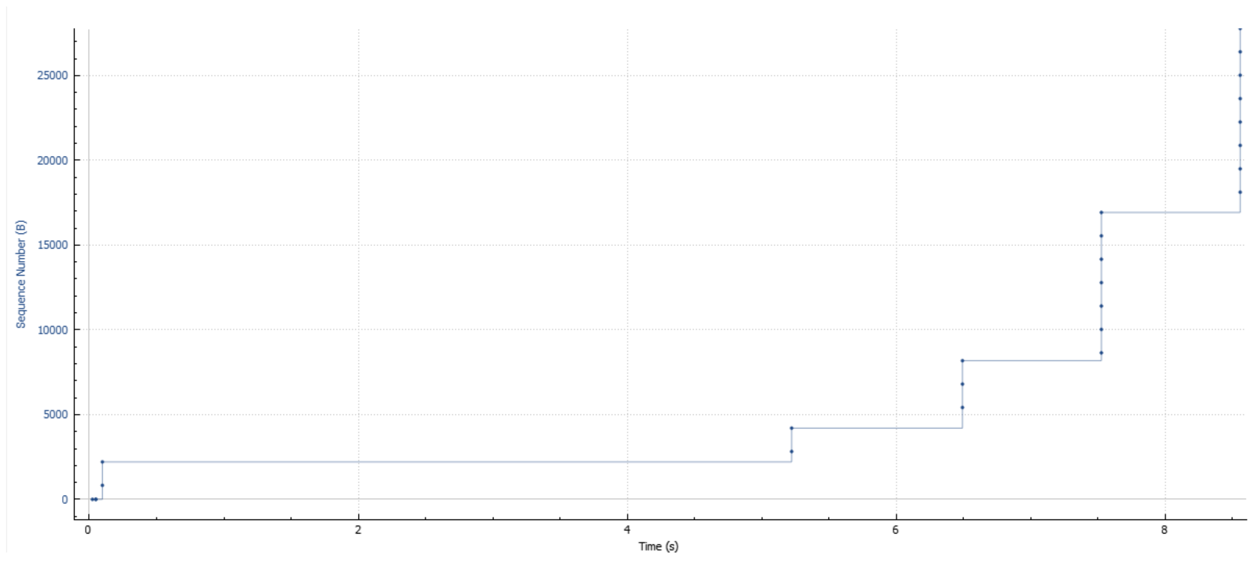
Q18: What flags are set in each of segments used for connection termination?

Answer: Fin flag is set for the first and third segments. Ack flag is set for the second and last segments.

Q19: Use the Time-Sequence-Graph (Stevens) plotting tool to view the sequence number versus time plot of segments being sent from the client to the server. Can you identify where TCP's slow start phase begins and ends, and where congestion avoidance takes over? Explain your answer.

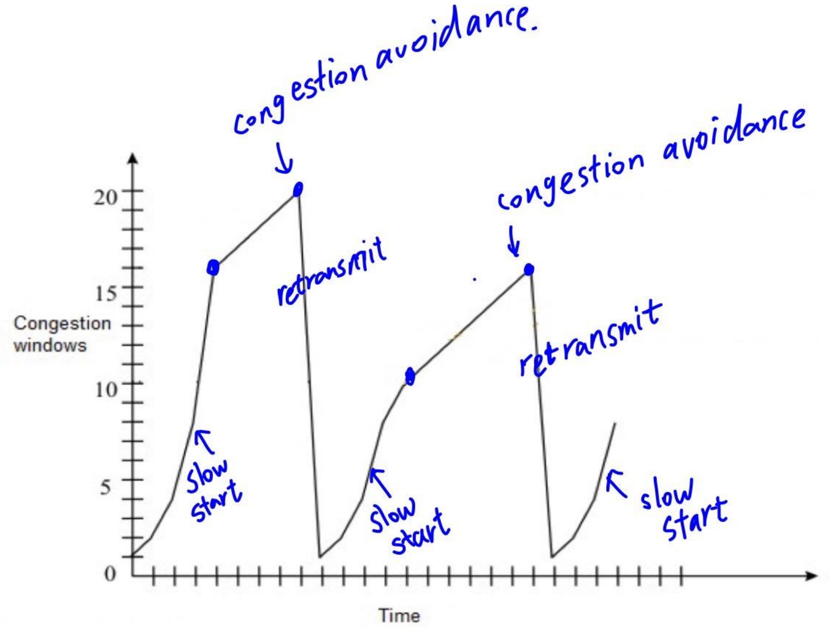
Answer:

Slow start begins at 10.05s and ends at 17.55s. Congestion avoidance starts at 18.58s since the sequence number doesn't increase exponentially from then.



Q20: Locate the different phases of the congestion control mechanism on the below graph. Also describe the congestion control algorithm.

Answer:
Congestion control algorithm includes slow start, congestion avoidance, and fast retransmit and fast recovery. TCP starts by sending a small number of segments and exponentially increases the transmission rate until it reaches a threshold. Once the threshold is reached,TCP only increases the congestion window by one MSS linearly for one RTT.


congestion avoidance.
congestion avoidance
retransmit
retransmit
slow start
slow start
slow start
Congestion windows
Time
20
15
10
5
0