Homework #2

Mike Pendleton

John Hulett

West Texas A@M University

**CIDM 6340 Network Management & Information Security – Fall 2021**

**West Texas A&M University**

Homework #2

***[N.B: You can work with a team/group of MAX 5 students, but you need to submit your own Homework report. Also, please mention all group members name in the cover page]***

**Name : \_\_\_\_\_Mike Pendleton\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Id:\_** **1058201**

**Rubrics:**

*Full points:* Correct and complete answer.

*Partial points:* Correct but not complete answer.

*No points:* No answer or Incorrect AND incomplete answers.

**Q1 Chapter-3 [10 points] : True or false?**

1. Host A is sending Host B a large file over a TCP connection. Assume Host B has no data to send Host A. Host B will not send acknowledgments to Host A because Host B cannot piggyback the acknowledgments on data.
   1. **False**
2. The size of the TCP *rwnd* never changes throughout the duration of the connection.
   1. **False**
3. Suppose Host A is sending Host B a large file over a TCP connection. The number of unacknowledged bytes that A sends cannot exceed the size of the receive buffer.
   1. **True**
4. Suppose Host A is sending a large file to Host B over a TCP connection. If the sequence number for a segment of this connection is m, then the sequence number for the subsequent segment will necessarily be m + 1.
   1. **False**
5. The TCP segment has a field in its header for *rwnd*.
   1. **True**
6. Suppose that the last *SampleRTT* in a TCP connection is equal to 1 sec. The current value of *TimeoutInterval* for the connection will necessarily be ≥ 1 sec.
   1. **False**
7. Suppose Host A sends one segment with sequence number 38 and 4 bytes of data over a TCP connection to Host B. In this same segment the acknowledgment number is necessarily 42.
   1. **False**
8. While the network layer establishes logical communication between hosts, the transport layer establishes the logical communication between processes
   1. **True**
9. DDoS attacks works with bombard root servers with traffic, but we are not successful to date traffic filtering, local DNS servers cache IPs of TLD servers, and allowing root server bypass.
   1. **True**
10. UDP is connectionless and less reliable. If reliable transfer needed over UDP (e.g., HTTP/3), we need to add needed reliability at application layer, and add congestion control at application layer.
    1. **False**

**Q2. Chapter -3 [10 points] :** An application may choose UDP for a transport protocol because UDP offers finer application control (than TCP) of what data is sent in a segment and when.

a. Why does an application have more control of what data is sent in a segment?

b. Why does an application have more control on when the segment is sent?

1. With TCP, the application first sends the message to the socket. Then the data would go to the TCP buffer past the Socket. Finally, the TCP would grab the data from the TCP buffer. Thus, the TCP would have more control than the application about what data would be sent. In UDP data from the application are put in the segment directly. So the application can get more control over what data would be sent.
2. Due to the congestion and flow controls within TCP there would be much more of a delay putting the segment from the buffer to the network layer.  UDP does not have such delay, so an application has more control when the segment is sent over.

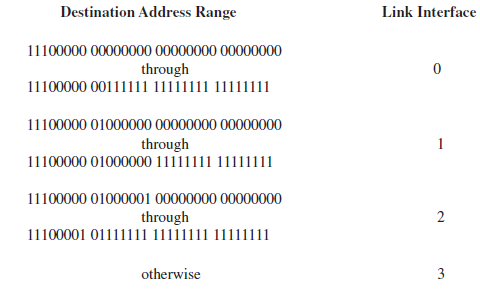
**Q3. Chapter -3 [10 points] :** Consider transferring an enormous file of L bytes from Host A to Host B. Assume an MSS of 536 bytes.

1. What is the maximum value of L such that TCP sequence numbers are not exhausted? Recall that the TCP sequence number field has 4 bytes.
   1. 2^32 = 4294967296 bytes and converted to Gb would be 4.19 maximum file size to be sent after accounting for the sequence number.
2. For the L you obtain in (a), find how long it takes to transmit the file. Assume that a total of 66 bytes of transport, network, and data-link header are added to each segment before the resulting packet is sent out over a 155 Mbps link. Ignore flow control and congestion control so A can pump out the segments back to back and continuously.
   1. 4294967296 / 536 = 8,012,999 \* 66 = 528,857,934 bytes of header.
   2. 4294967296 + 528,857,934 = 4,823,825,230 \* 10^9 bytes = 38590 Mb
      1. 38590 Mb \ 155 Mb/s = 248.96
         1. Answer is **249 seconds**

**Q4: Chapter – 4 [ 10 points]** Consider a datagram network using 32-bit host addresses. Suppose a router

has four links, numbered 0 through 3, and packets are to be forwarded to the

link interfaces as follows:

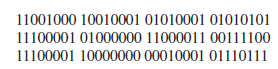


a. Provide a forwarding table that has five entries, uses longest prefix matching,

and forwards packets to the correct link interfaces. [ see book 4.2.2]

b. Describe how your forwarding table determines the appropriate link interface

for datagrams with destination addresses:



Answer: Forwarding Table A)

|  |  |  |
| --- | --- | --- |
| **Address Range (Destination** | **Range Cover** | **prefix match** |
| 11100000 00000000 00000000 00000000  through  11100000 00111111 11111111 11111111 | **11100000 00**000000 00000000 00000000  **11100000 00**111111 11111111 11111111 | 11100000 00  interface 0 |
| 11100000 01000000  00000000 00000000  through  11100000 01000000 11111111 11111111 | **11100000 01000000**  00000000 00000000  **11100000 01000000** 11111111 11111111 | 11100000 01000000  interface 1 |
| 11100000 01000001 00000000 00000000  through  11100001 01111111 11111111 11111111 | **1110000**0 01000001 00000000 00000000  **1110000**1 01111111 11111111 11111111 | 1110000  interface 2 |
| **11100000 01**000001 00000000 00000000  **11100001 01**111111 11111111 11111111 | 11100001 1  interface 3 |
| otherwise |  | interface 3 |

B) 11001000 10010001 01010001 01010101 - Matches to otherwise entry, link 3 - the address range in not in the forwarding table so it goes to otherwise

11100001 01000000 11000011 00111100 = Matches to 111000 entry, link 2

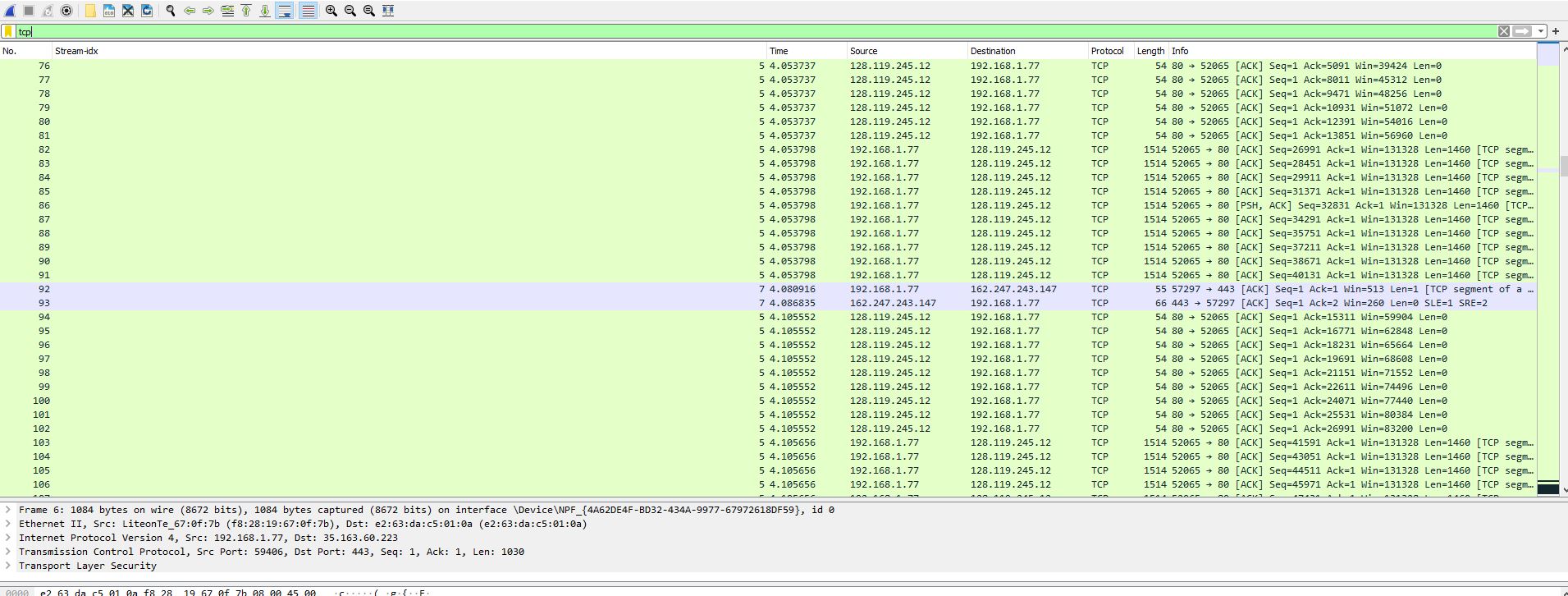
11100001 10000000 00010001 01110111 = matches to 11100001 1 entry link 3

**Q5: [ 10 points] Chapter -3 and 4.**

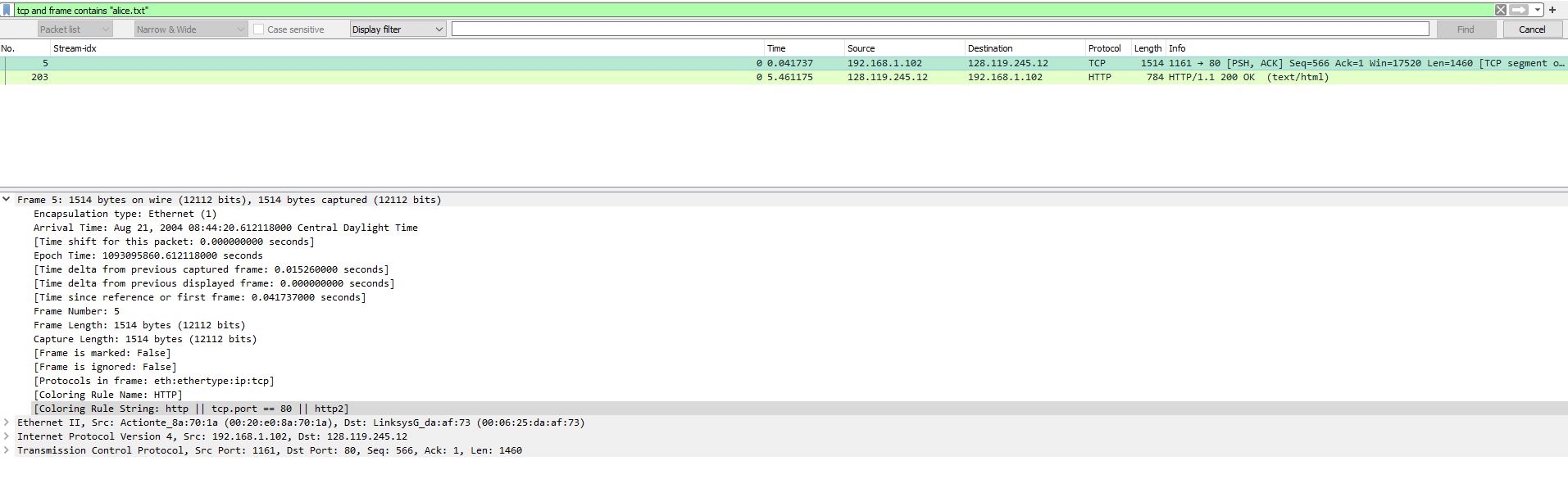
1. Consider that Alice sends e-mail to Bob, how will you explain the scenario through SMTP protocol [Hints see in week#3 slide].
   1. Email client sends message through the outgoing smtp connector of Alices email server which then opens a connection to bobs email server (This can be the same server, or different mail servers). The tcp connection for the syn request gets the ack response and the email server on bobs end then inputs the email in bobs email storage. Then bob can read the message through his email client.
2. Briefly explain a TCP segment structure and an IPv6 datagram format. [ see box like figures from lecture 3 and lecture 4]
   1. TCP Segment structure –32 bit , consists of the the source and destination port, sequence number, acknowledgement number, header, receive window(flow control only used in tcp), option and the application data.
   2. ipv6 datagram format – does has no checksum, no fragmentation/reassemble, no options when compared to ipv4. Also it is 32 bit with 128 bit ip addressing which is hexadecimal.

**Practical:**

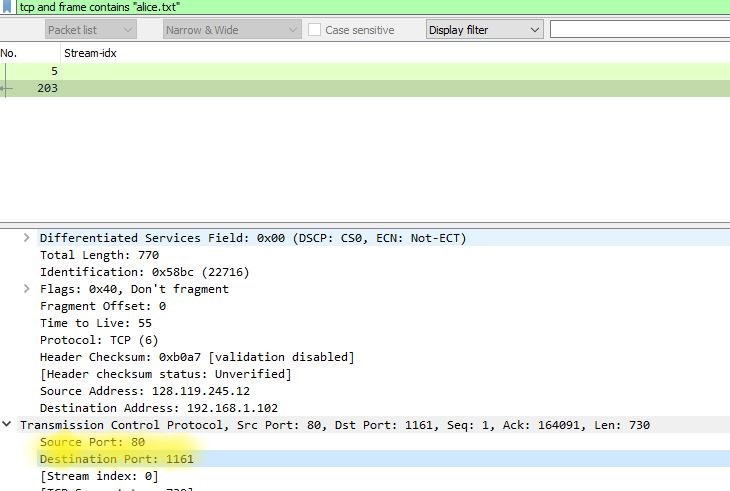
**Q5. [ 25 points] Wireshark Lab (See attachment#1 : TCP lab)**



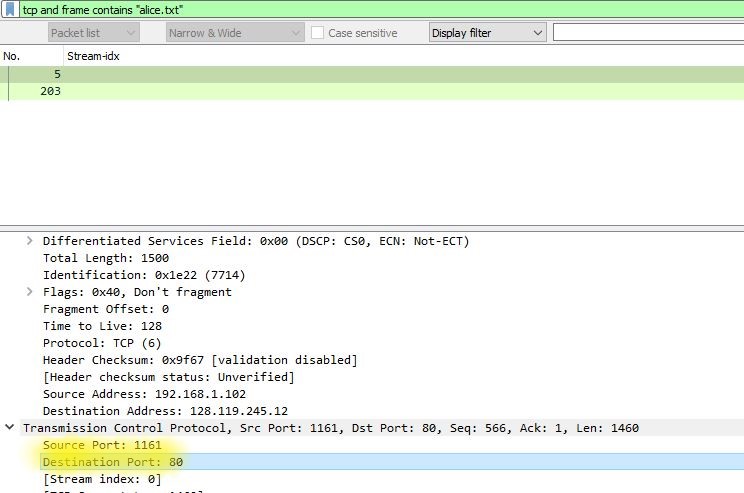
* 1. Client ip 192.168.1.102 and it is tcp port 80
  2. I used filter ‘tcp and frame contains "alice.txt"’ to get the exact frame in question



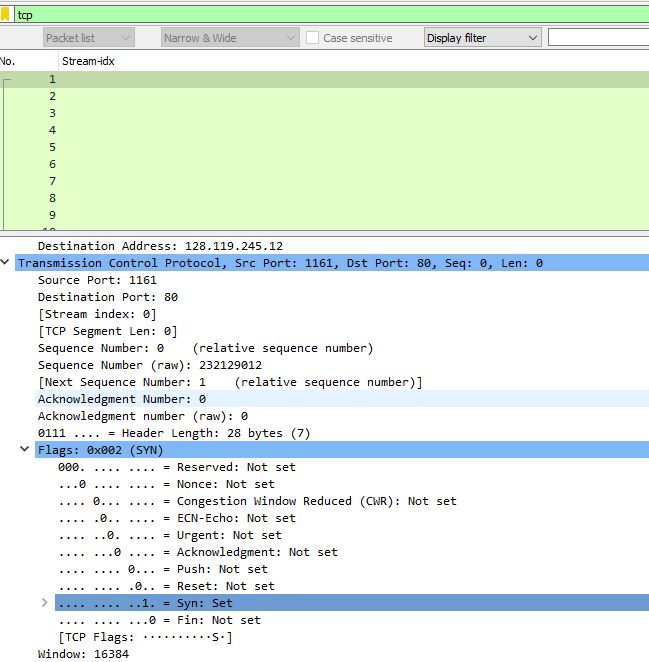
1. Question 2
   1. Ip address of gaia.cs.umass.edu is 128.119.245.12, the destination port is 1161 and the source port is 80.



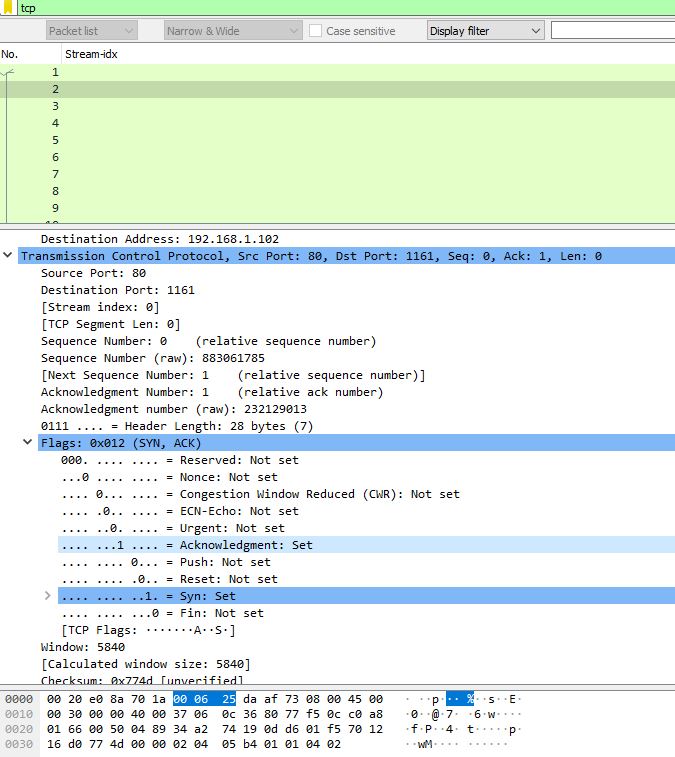
* 1. The source and destination ports for the client machine are reversed. Source port is 1161 and the destination port is 80



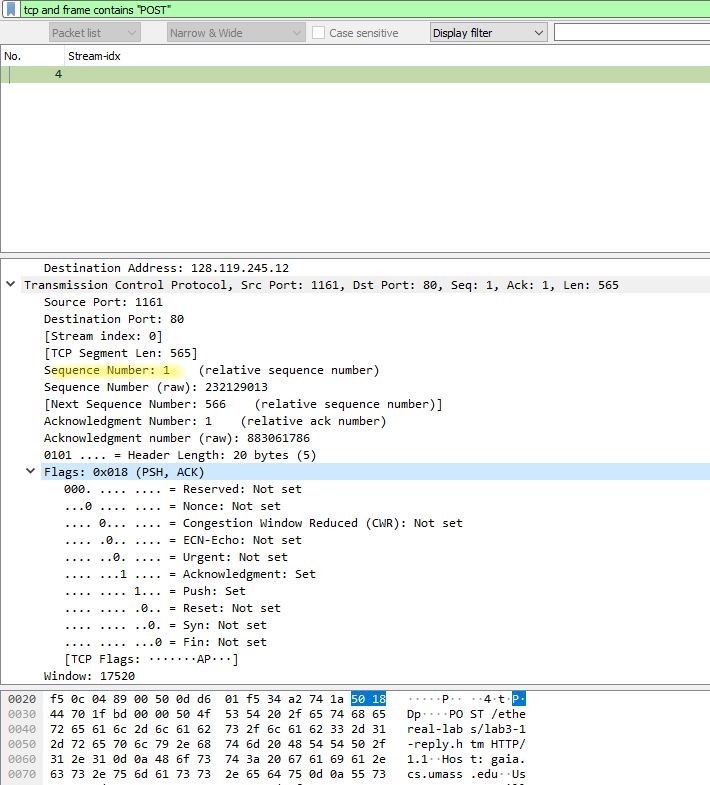
* 1. Sequence number: 0
  2. The syn segment is set under the flags



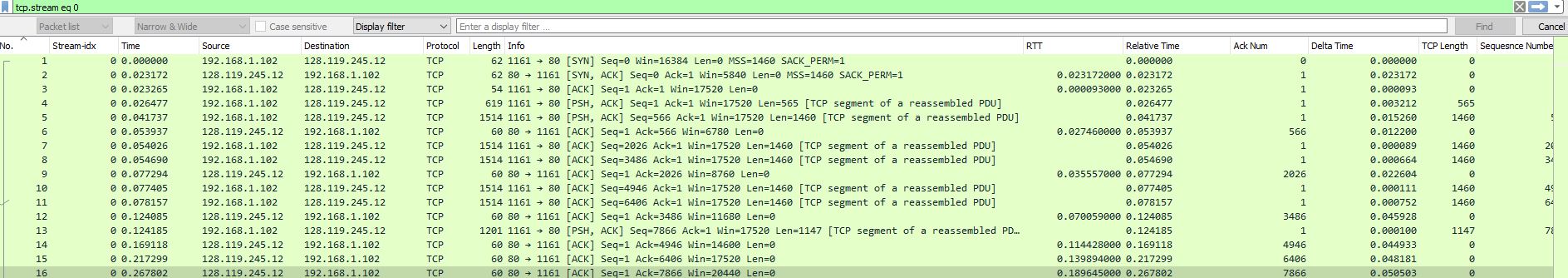
* 1. The sequence number of the synack is 0
  2. The value of the acknowledgement field is 1.
  3. The server adds a value of one plus the value of the sequence number. In this case since the sequence number was 0, the value of the ack is 1
  4. The flag selection determines this as stated prior in the request, so the response adds the ack.



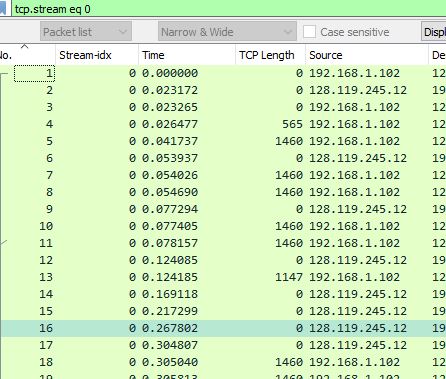
* 1. The sequence number of the post is Sequence Number: 1 (relative sequence number), I used the filter “tcp and frame contains "POST"”



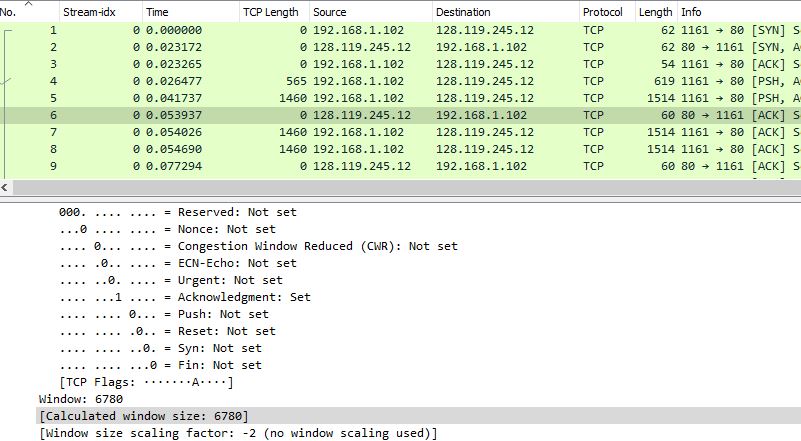
* 1. The first 6 segments the sequence numbers are
  2. first 6 segments are:
     1. 4 sequence number 1
     2. 5 sequence number 566
     3. 7 sequence number 2026
     4. 8 sequence number 3486
     5. 10 sequence number 4946
     6. 11 sequence number 6406
  3. I added RTT in the column by setting the column value to “tcp.analysis.ack\_rtt”
  4. Acknowledgments are: RTT:
     1. 6 ack number 566 0.027460000
     2. 9 ack number 2026 0.035557000
     3. 12 ack number 3486 0.070059000
     4. 14 ack number 4946 0.114428000
     5. 15 ack number 6406 0.139894000
     6. 16 ack number 7866 0.189645000



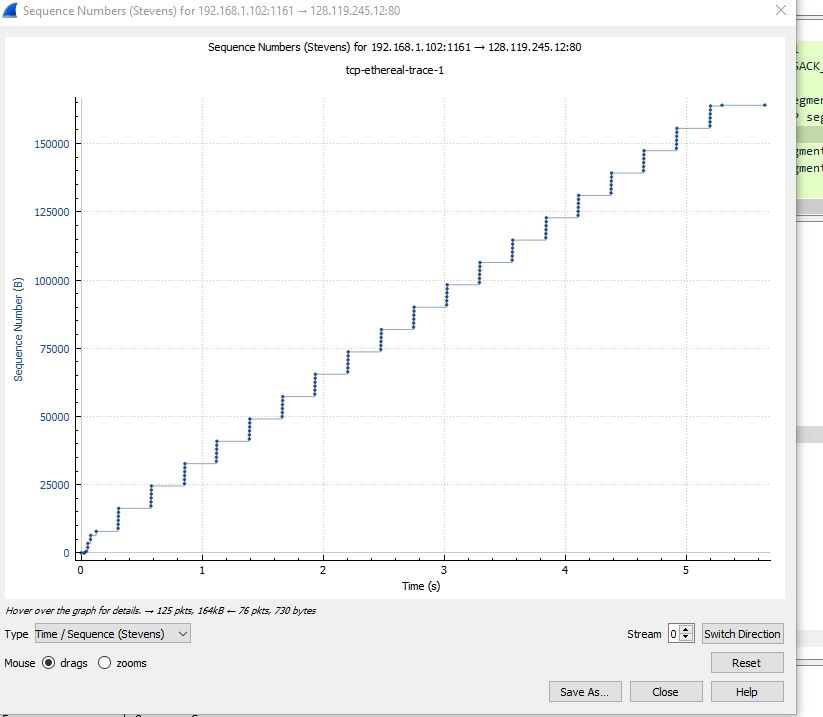
* 1. The first segment is 565 segments and the other 5 are 1460 in length.



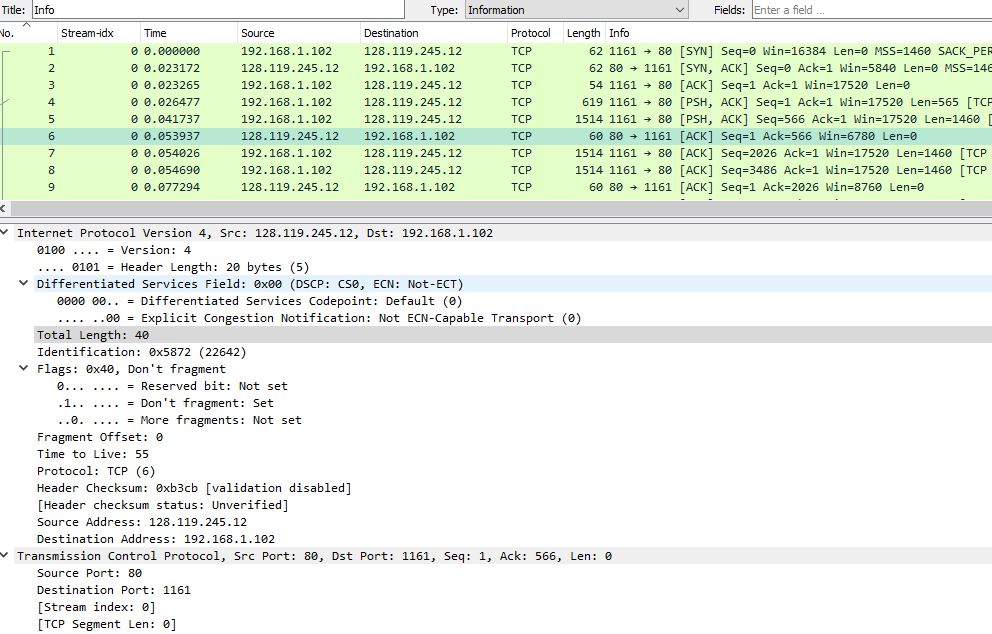
* 1. The minimum amount of available buffer space advertised is 6780



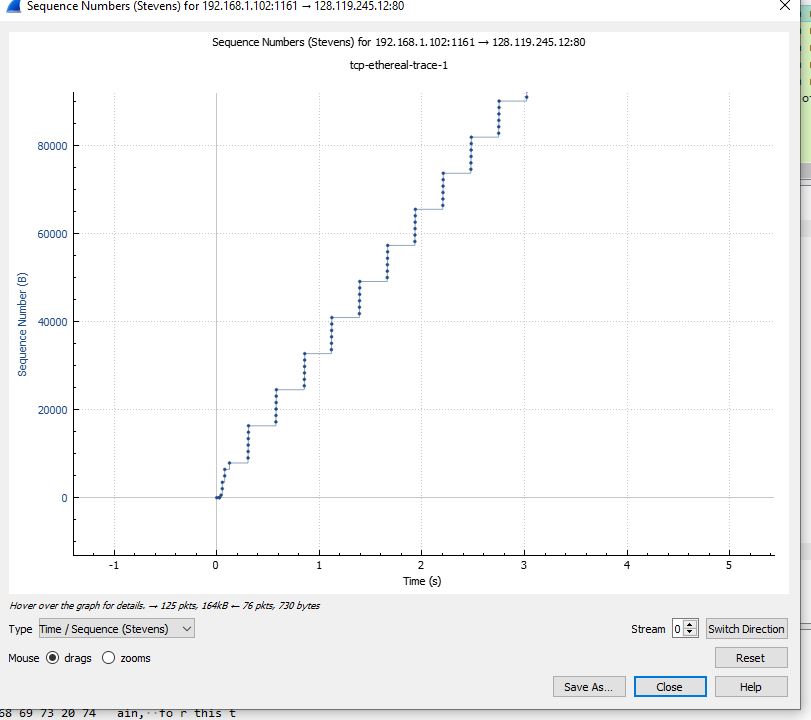
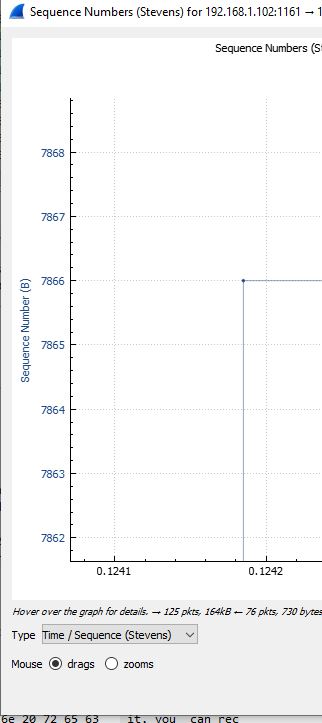
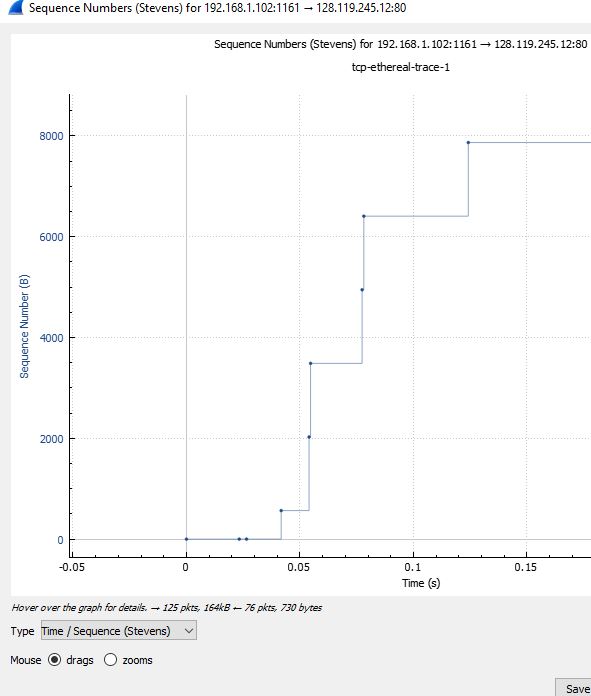
* 1. No, because the sequence numbers continually increase. The stevens graph in tcp streams graphs.



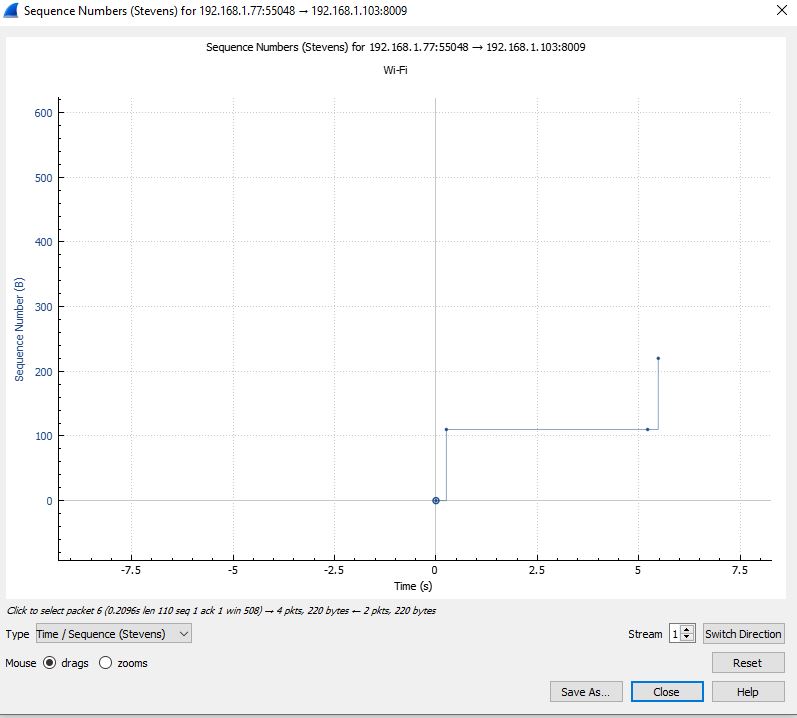
* 1. The acknowledgement are all 40, the payload of what tcp is sending does change.
  2. In this instance with the amount of data sent, I do see where the ack is responding several times with not many syn. This is because the packet transfer will complete once all data has been delivered to the client. There is not a need for the syn to match the amount of ack.



* 1. The first transmission time is 0.023172 and the last ack is 5.651141 these subtract to get total transmission time of 5.627969
  2. Then the size of the file is 152000 bytes / 5.627969 transmission time =
  3. **27,007.966** bytes per second (**throughput**)
  4. The slow start begins on sequence number 565 and ends on sequence number 7866. This would be due to the handshake, in order to deliver the file. Congestion avoidance then takes over as the rest of the sequence numbers are consistent by looking at the graph below.
  5. This is different than what we covered thus far because we can visually see by looking in the zoomed in graph provided exactly where it begins. You can also then cross reference this to the sequence number on 565 as this is when the post begins.



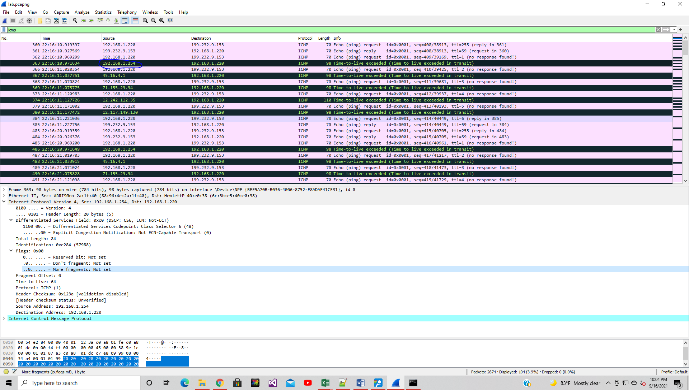
* 1. My wireshark trace does not show a slow start like the uploaded example does. But if I calculate the throughput this does make sense. I have very high throughput with my environment.
  2. 152000 bytes / 0.000382 = 397,905,759.162 bytes per second. It is possible there was high latency in the environment where this file was used for the lab.



**Q6. [ 25 points] Wireshark Lab: HTTP (see Attachemnt#2 : IP lab)**

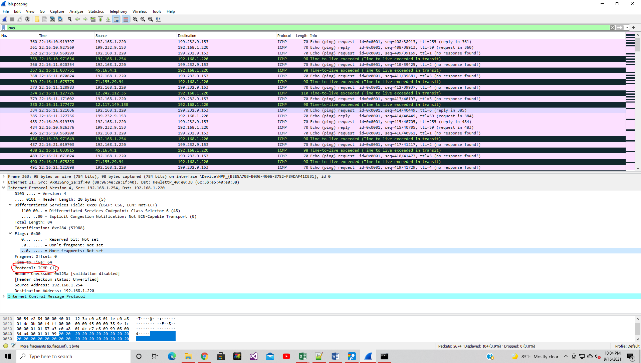
1. What is the IP address of your computer?

answer = 192.168.254

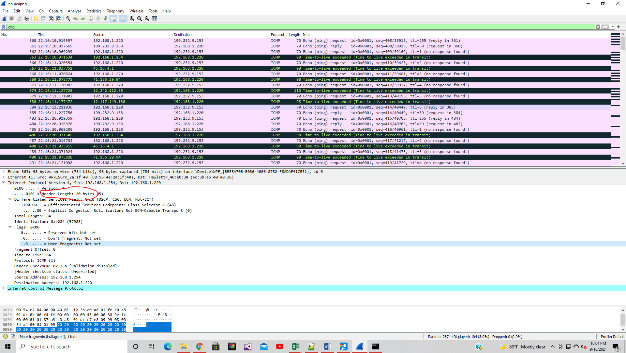


1. Within the IP packet header, what is the value in the upper layer protocol field?

Answer = ICMP (1)

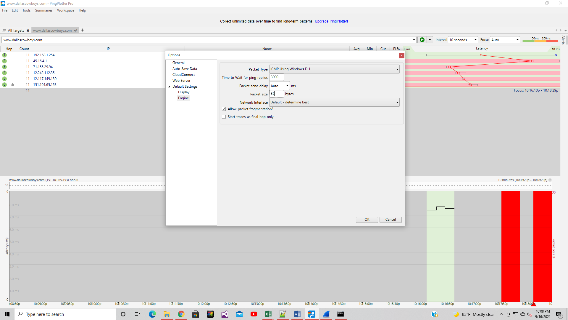


1. How many bytes are in the IP header?
2. 20 bytes



How many bytes are in the payload of the IP datagram?

1. 36



Explain how you determined the number of payload bytes.

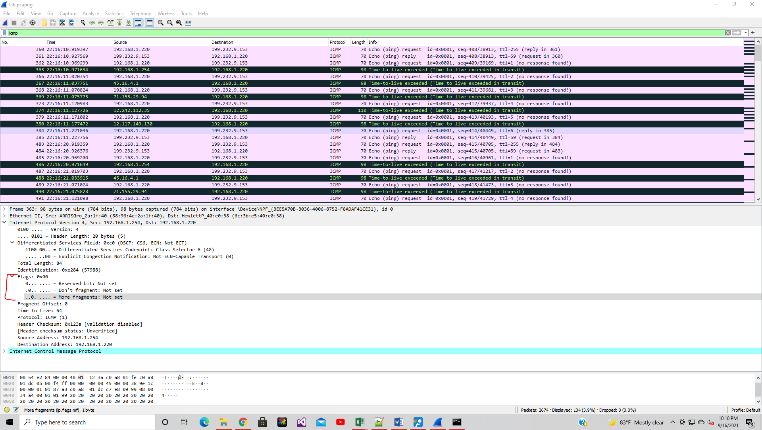
1. To get the payload bytes you take the packet size minus the header bytes

so 56-20= 36

4. Has this IP datagram been fragmented?  Explain how you determined whether or not the datagram has been fragmented.

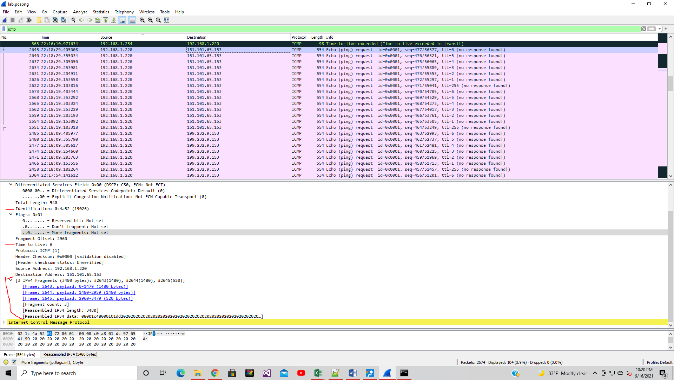
Answer:  No the IP datagram has not been fragmented.  This is due to the following

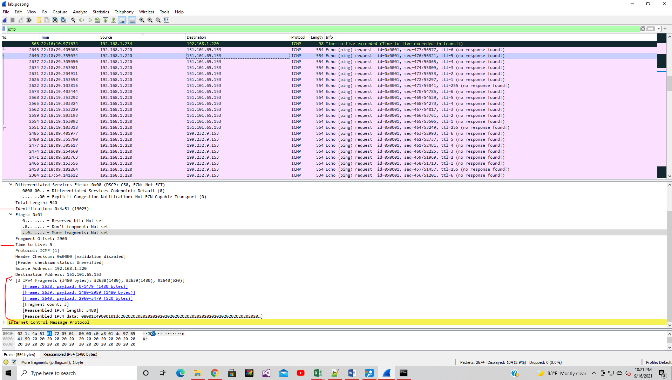
don't fragment is “Not Set”, the Reserved Bit is “Not Set” and the fragment offset is set to 0



5. Which fields in the IP datagram *always* change from one datagram to the next within this series of ICMP messages sent by your computer?

Answer: As you go down the list the Identification changes, the Time to Live changes and the IPv4 fragments change





6. Which fields stay constant?  Which of the fields *must* stay constant? Which fields must change?  Why?

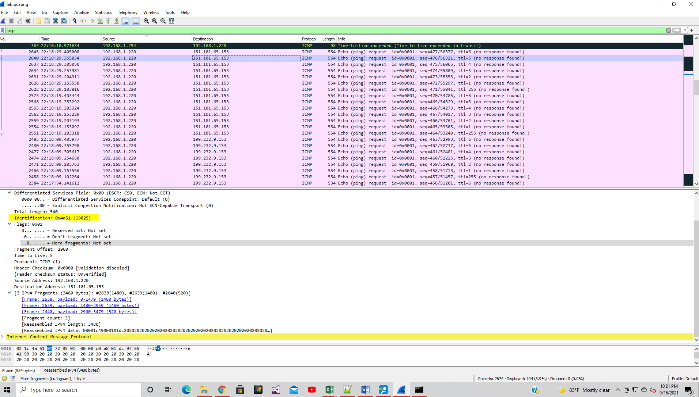
Answer: The fields that stay constant are 1) Version stays as IPv4, 2) Header length- length of TCP is 20 bytes, 3) Differentiated service field- all use ICMP protocol, and 4) Total length- stays the same because they all use the same ICMP protocol.

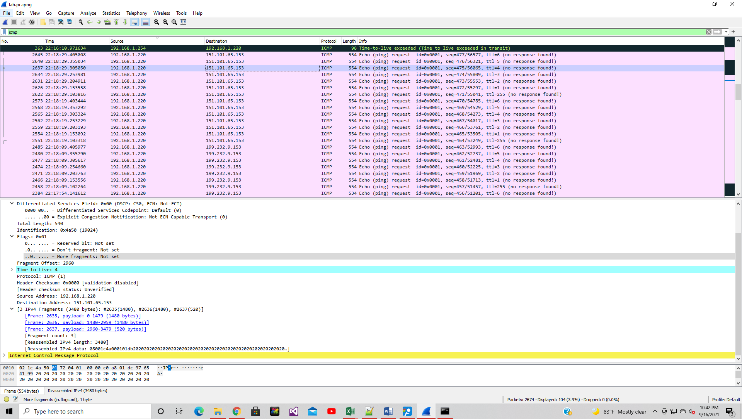
The fields that must stay constant are 1) Version- we see IPv4, 2) Header length- typical length of TCP is 20 bytes, 3) Differentiated service field- all use ICMP protocol, and 4)

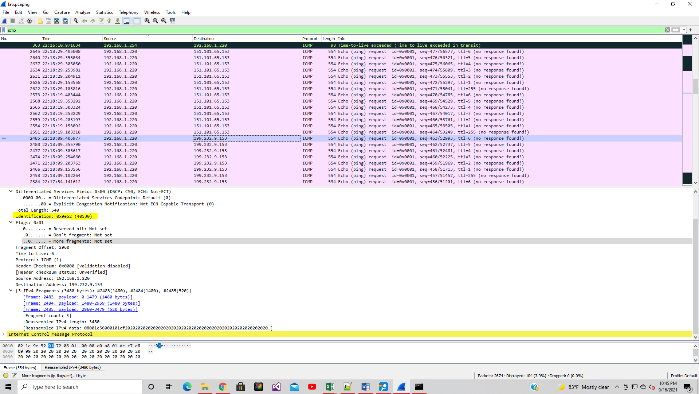
Total length stays constant due to the use of ICMP protocol.  The fields that must change are 1) Identification; as packets have their own unique value, 2) Time to Live as each one has different length of route, and 3) the Header checksum field as all packets have its own header

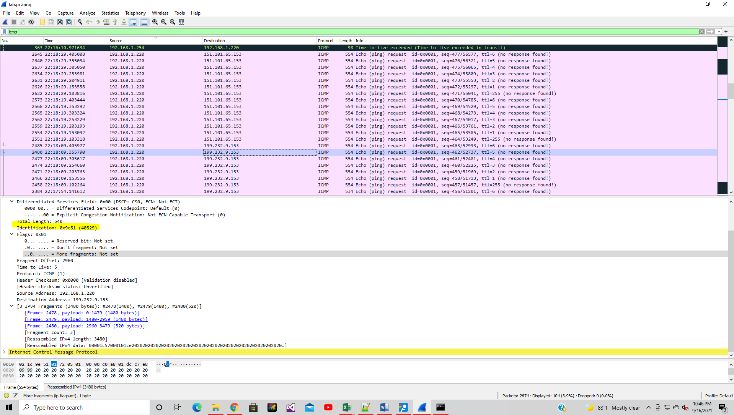
7. Describe the pattern you see in the values in the Identification field of the IP datagram

Answer:  As you navigate down the list the identification hex value and number change.  The number decreases each time.  When the ip address changes the sequence number changes.  The first two screenshots show 19025 and then 19024 for the 1st IP address, the 3rd and 4th screenshots show 40530 and then 40529



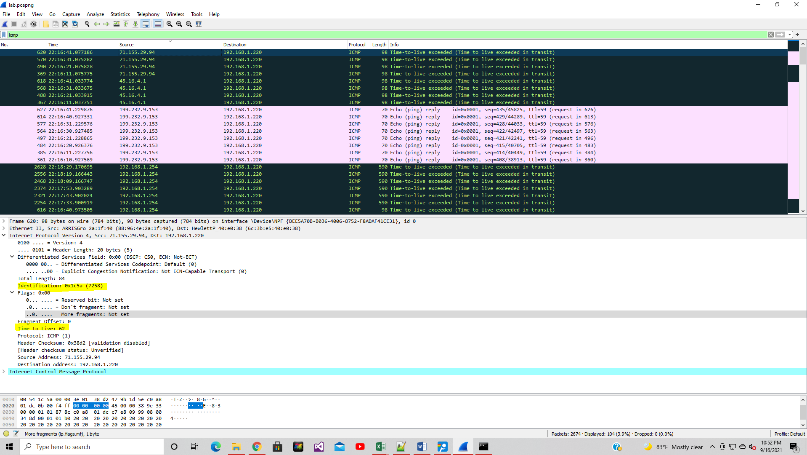






8. What is the value in the Identification field and the TTL field?

Answer:  The identification field has 7258, and the time to live is 62

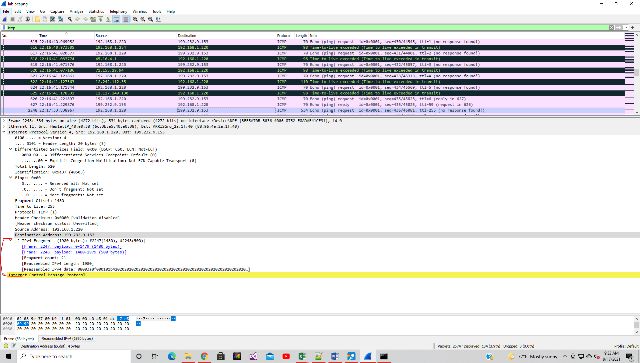


9. Do these values remain unchanged for all of the ICMP TTL-exceeded replies sent to your computer by the nearest (first hop) router?  Why?

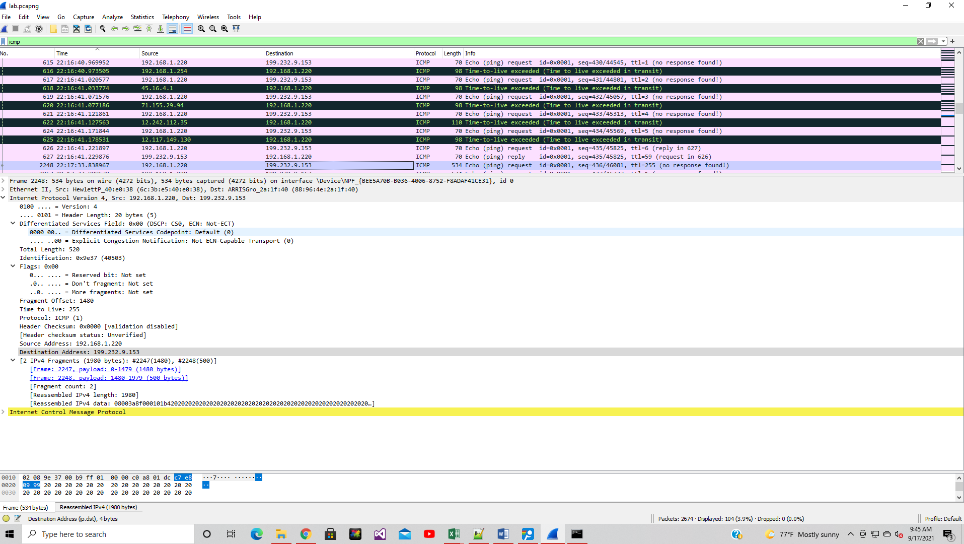
Answer: The identification field changes because they need to have a unique number so we know where it is coming from. The time to live (TTL) does not change because it is always the same in this case.

10. Find the first ICMP Echo Request message that was sent by your computer after you changed the *Packet Size* in *pingplotter* to be 2000. Has that message been fragmented across more than one IP datagram?

Answer:  Yes the message has been fragmented



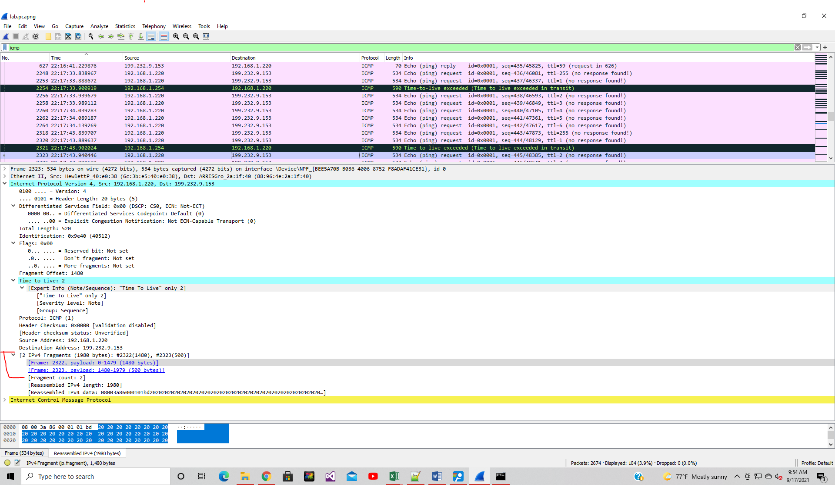
11. Print out the first fragment of the fragmented IP datagram. What information in the IP header indicates that the datagram been fragmented?  What information in the IP header indicates whether this is the first fragment versus a latter fragment?  How long is this IP datagram?



Answer: a) The more fragment has been set b) The Fragment offset has been set to 0 which means this is the 1st segment C) the datagram is a total of 1500

12. Print out the second fragment of the fragmented IP datagram. What information in the IP header indicates that this is not the first datagram fragment?  Are the more fragments?  How can you tell?

a) The Fragment offset has been set to 1480 which indicates its the 2nd fragment. b) yes there are more fragments.  c) you can tell there are 2 fragments by the IPV4 section of the datagram appearing and showing the fragments and showing a fragment count of 2



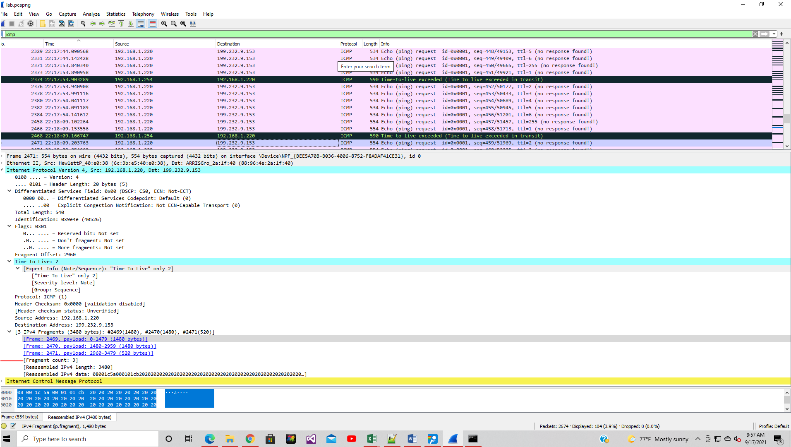
13. What fields change in the IP header between the first and second fragment?

Answer: The Fragment offset and header checksum fields changed between the 1st and 2nd fragments

Now find the first ICMP Echo Request message that was sent by your computer after you changed the *Packet Size* in *pingplotter* to be 3500.

14. How many fragments were created from the original datagram?

Answer:  The number of fragments is now 3



15. What fields change in the IP header among the fragments?

Answer:  The fragment and offset change among the fragments.  The time to live changes and the total length ranges from 150 to 540.

For each lab, please try to answer as many questions possible.

For each lab, please try to answer as many questions possible.