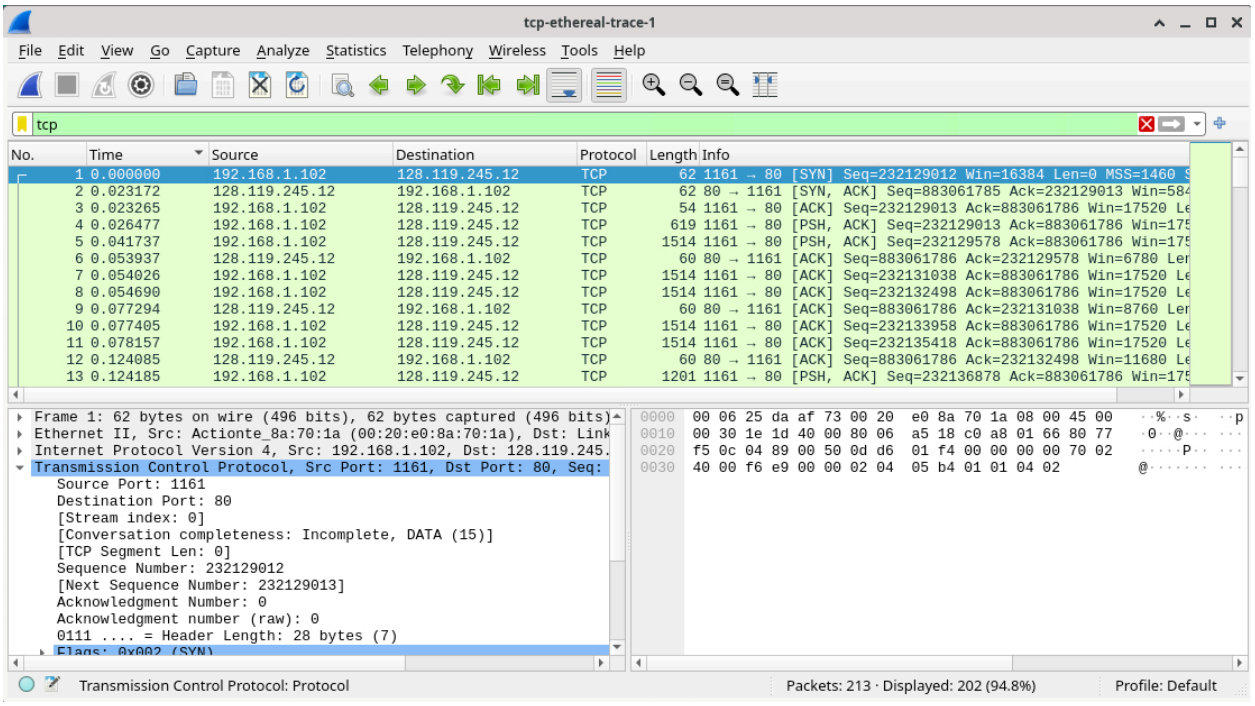
Aarnav z5477523 Lab 4

Exercise 1: Understanding TCP using Wireshark

Question 1 . What is the IP address of gaia.cs.umass.edu? On what port number is it sending and receiving TCP segments for this connection? What are the IP address and TCP port numbers used by the client computer (source) that is transferring the file to gaia.cs.umass.edu?

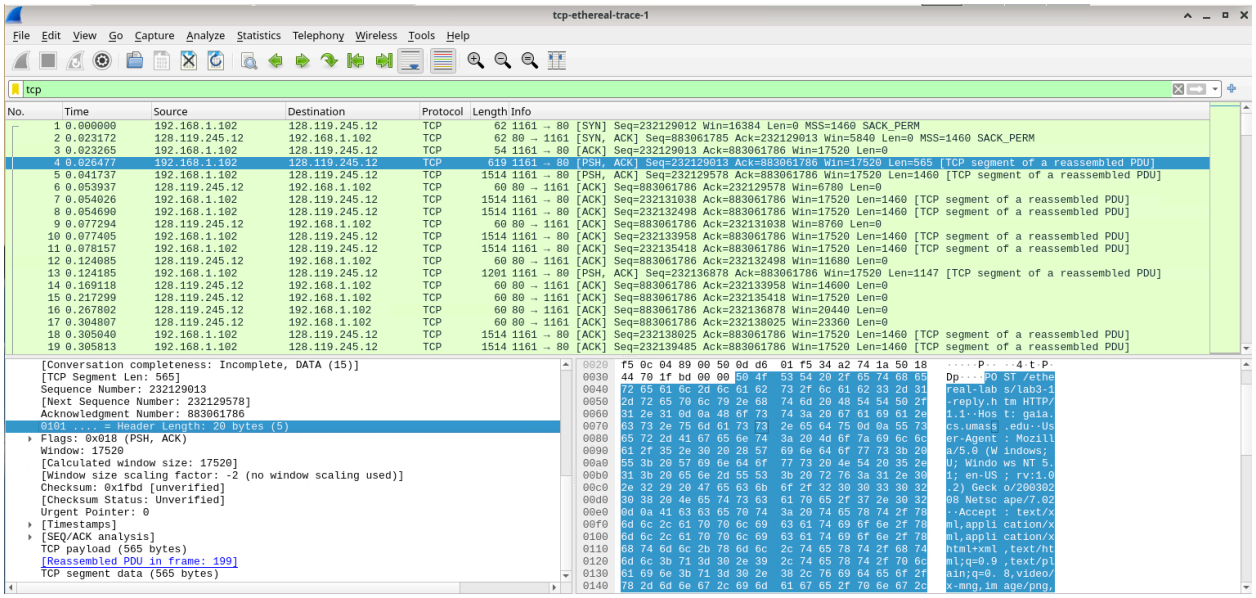


IP address of gaia.cs.umass.edu is 128.119.245.12
The port for the connection with source gaia.cs.umass is 80 for sending and receiving .

IP address of the source is 198.168.1.102, and the TCP Port for this port is 1161.

Question 2 . What is the sequence number of the TCP segment containing the HTTP POST command? Note that to find the POST command, you'll need to dig into the packet content field at the bottom of the Wireshark window, looking for a segment with a " POST " within its DATA field.

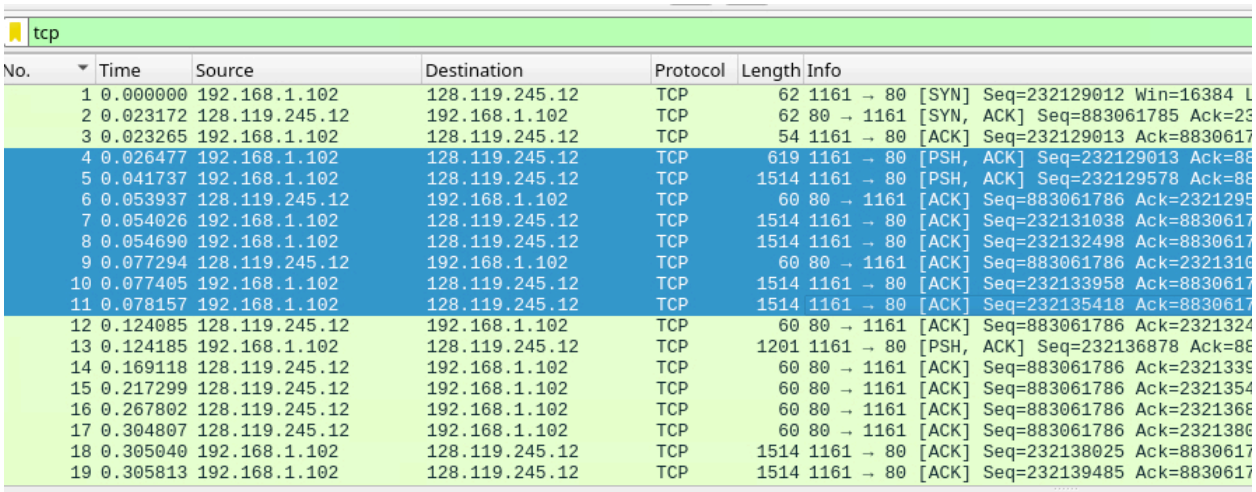
Aarnav z5477523 Lab 4



The sequence number is 232129013 for the TCP containing POST /ethereal-labslab3-1-reply.h HTTP

Question 3. Consider the TCP segment containing the HTTP POST as the first segment in the TCP connection.

(a) What are the sequence numbers of the first six segments in the TCP connection (including the segment containing the HTTP POST) sent from the client to the webserver (Do not consider the ACKs received from the server as part of these six segments)?

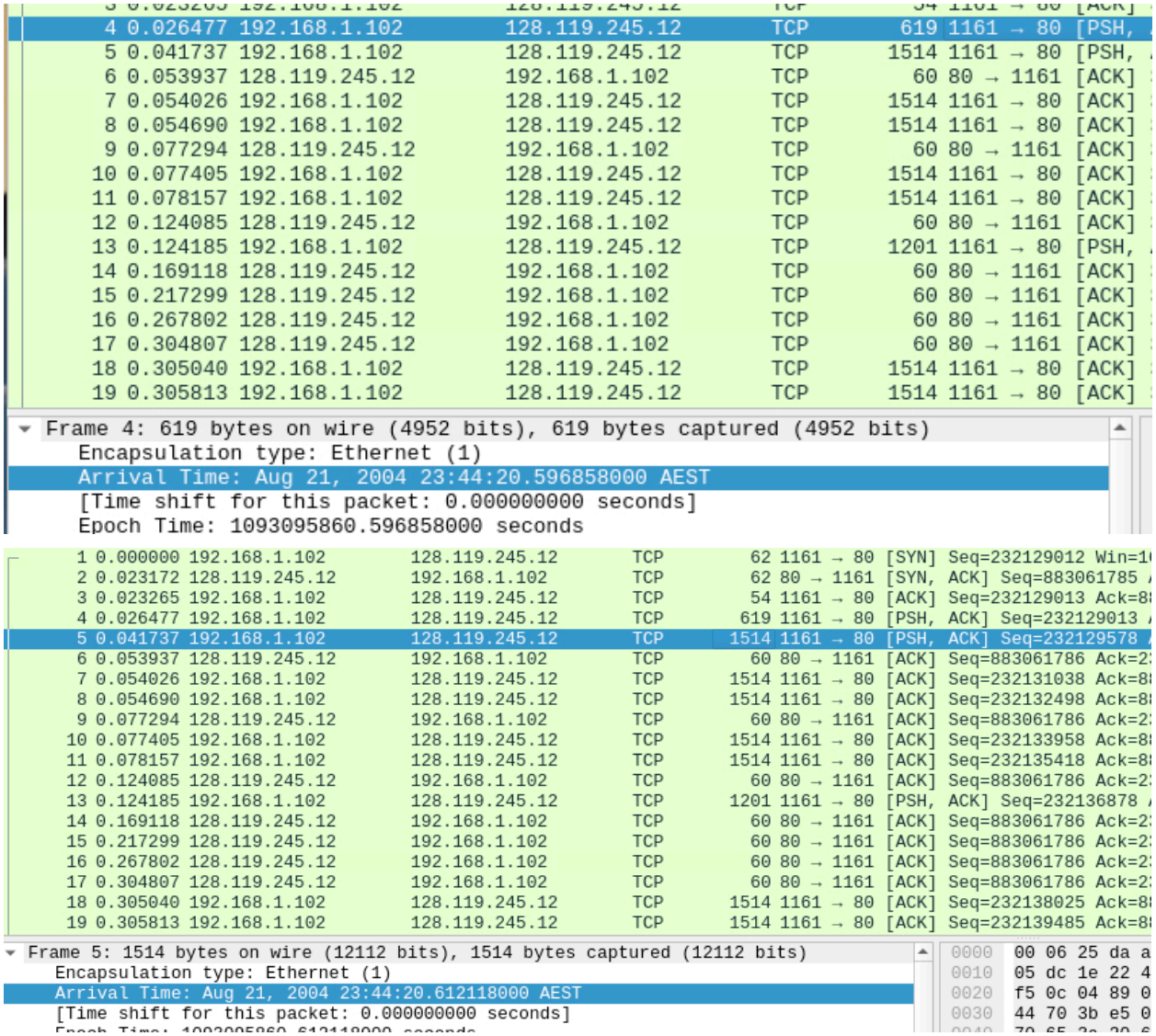


1. 232129013
2. 232129578
3. 232131038
4. 232132498
5. 232133958

Aarnav z5477523 Lab 4

6. 232135418

(b) At what time was each segment sent? When was the ACK for each segment received? 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 six segments?



1. 232129013 Sent at 23:44:20.596858, the ack was received at 23:44:20.624318 which gives us a RTT value of 27.460ms
2. 232129578 Sent at 23:44:20.612118, the ack was received at 23:44:20.624318 which gives us a RTT value of 12.200ms
3. 232131038 Sent at 23:44:20.624407, the ack was received at 23:44:20.647675 which gives us a RTT value of 23.268ms
4. 232132498 Sent at 23:44:20.625071, the ack was received at 23:44:20.694466 which gives us a RTT value of 69.395ms

5. 232133958 Sent at 23:44:20.647786, the ack was received at 23:44:20.739499 which gives us a RTT value of 91.713ms
6. 232135418 Sent at 23:44:20.648538, the ack was received at 23:44:20.787768 which gives us a RTT value of 139.142ms

(c) What is the EstimatedRTT value (see relevant parts of Section 3.5 or lecture slides) after receiving each ACK? Assume that the initial value of EstimatedRTT is equal to the measured RTT (SampleRTT) for the first segment and then is computed using the EstimatedRTT equation for all subsequent segments. Set alpha to 0.125.

If alpha is 0.125, then we can take the 1-alpha value as 0.875.

Hence we have: Estimated RTT = 0.875 x RTvalue + 0.125 x SampleRTT
1. 27.460 (same as sample)
2. (0.875 × 27.460) + (0.125 × 12.200) = 25.5525 ms
3. (0.875 × 25.5525) + (0.125 × 23.268) = 25.2670 ms
4. (0.875 × 25.2670) + (0.125 × 69.395) = 30.7839 ms
5. (0.875 × 30.7839) + (0.125 × 91.713) = 38.3975 ms
6. (0.875 × 38.3975) + (0.125 × 139.142) = 50.4906 ms

(d) What is the length of each of the first six TCP segments?

Question 4. What is the minimum amount of available buffer space advertised at the receiver for the entire trace? Does the lack of receiver buffer space ever throttle the sender?

The “win” is a representation of the minimum buffer space, so the minimum “win” is 5840 in line 2. This suggests that the receiver's buffer never limits the sender, since the buffer space stays at a minimum of 5840 and never reaches 0. Instead, the window size grows.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.102	128.119.245.12	TCP	62	1161 → 80 [SYN] Seq=232129012 Win=16384 Len=0 MSS=1460 SACK_PERM
2	0.023172	128.119.245.12	192.168.1.102	TCP	62	80 → 1161 [SYN, ACK] Seq=883061785 Ack=232129013 Win=5840 Len=0 MSS=1460 SACK_PERM
3	0.023265	192.168.1.102	128.119.245.12	TCP	54	1161 → 80 [ACK] Seq=232129013 Ack=883061786 Win=17520 Len=0
4	0.026477	192.168.1.102	128.119.245.12	TCP	619	1161 → 80 [PSH, ACK] Seq=232129013 Ack=883061786 Win=17520 Len=565 [TCP segment of a reassembled PDU]
5	0.041737	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [PSH, ACK] Seq=232129578 Ack=883061786 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
6	0.053937	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=883061786 Ack=232129578 Win=6780 Len=0
7	0.054026	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=232131038 Ack=883061786 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
8	0.054690	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=232132498 Ack=883061786 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
9	0.077294	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=883061786 Ack=232131038 Win=8760 Len=0
10	0.077405	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=232133958 Ack=883061786 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
11	0.078157	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=232135418 Ack=883061786 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
12	0.124085	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=883061786 Ack=232132498 Win=11680 Len=0
13	0.124185	192.168.1.102	128.119.245.12	TCP	1201	1161 → 80 [PSH, ACK] Seq=232136878 Ack=883061786 Win=17520 Len=1147 [TCP segment of a reassembled PDU]
14	0.169118	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=883061786 Ack=232133958 Win=14600 Len=0
15	0.217299	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=883061786 Ack=232135418 Win=17520 Len=0
16	0.267802	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=883061786 Ack=232136878 Win=20440 Len=0
17	0.304807	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=883061786 Ack=232138025 Win=23360 Len=0
18	0.305040	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=232138025 Ack=883061786 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
19	0.305813	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=232139485 Ack=883061786 Win=17520 Len=1460 [TCP segment of a reassembled PDU]

Question 5. Are there any retransmitted segments in the trace file? To answer this question, what did you check for (in the trace)?

Aarnav z5477523 Lab 4

The trace file's absence of duplicate sequence numbers indicates that no segments were retransmitted. We check for duplicate sequence numbers.

Question 6. How much data does the receiver typically acknowledge in an ACK? Can you identify cases where the receiver is ACKing every other received segment (recall the discussion about delayed acks from the lecture notes or Section 3.5 of the text)?

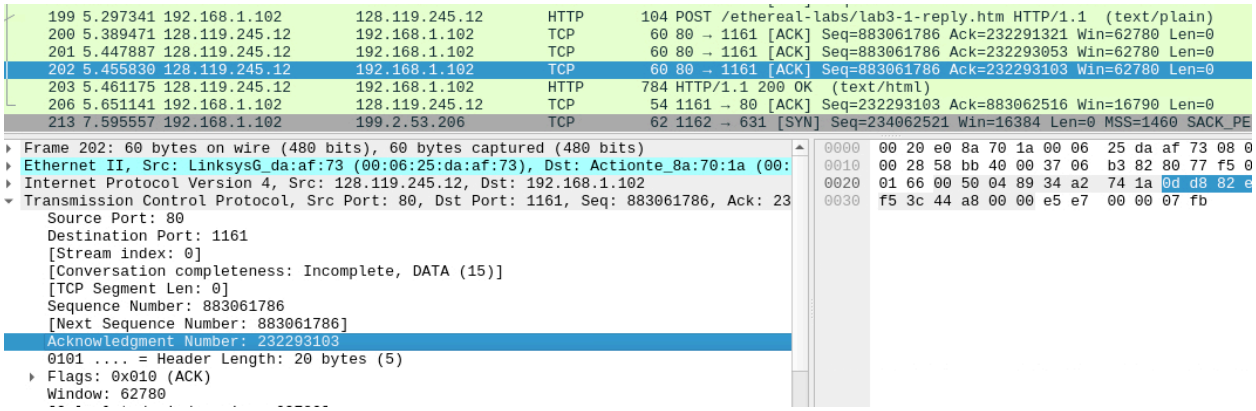
Typically, the receiver issues ACKs after receiving the Maximum Segment Size (MSS). The first segment, at 565 bytes, doesn't prompt an immediate ACK, resulting in a delayed ACK on line 6 for segments 1 and 2. In contrast, the following segments are 1460 bytes—equal to the MSS—leading the receiver to send an ACK after every alternate segment in 3-6.

Question 7. What is the TCP connection's throughput (bytes transferred per unit of time during the connection)?

Explain how you calculated this value.

We must find total time and total bytes sent.

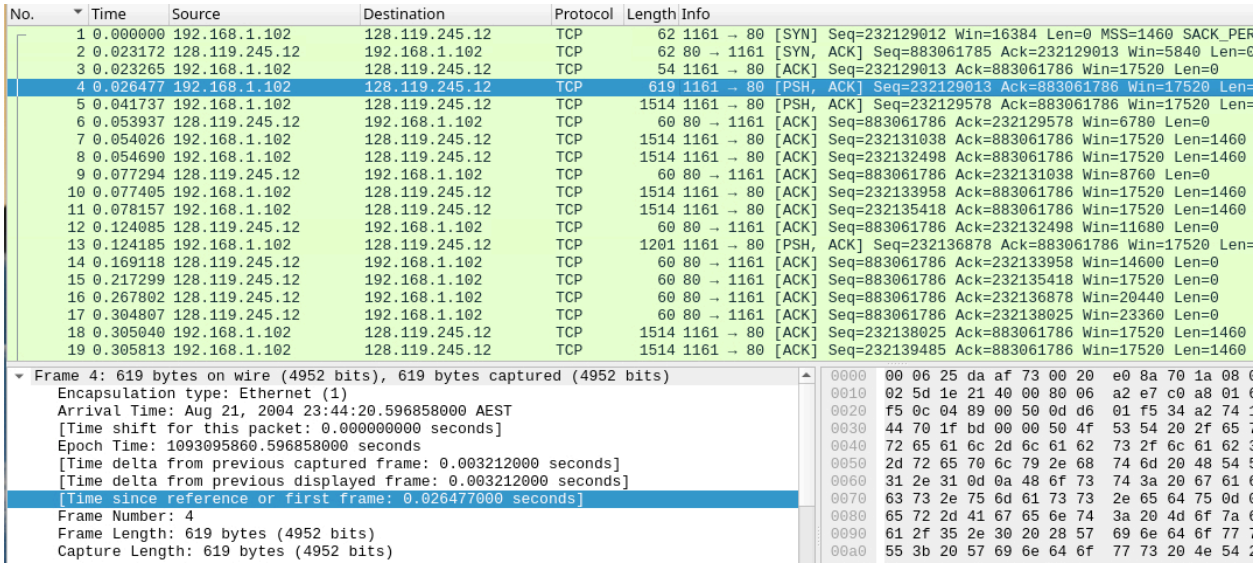
Total bytes is calculated by finding the difference of the numbers of the last ACK received on line 202 to the ISN (initial sequence number) on line 4. Last ACK is 232293103 and ISN is 232129013, meaning total bytes is 164090



On the same TCP above, in row 202, we can look at frame 202 to see the time since reference or first frame: 5.45583 seconds. But since we only want throughput, we must take time since

Aarnav z5477523 Lab 4

first frame from frame 4 which has a time of 0.026477000 s.



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.102	128.119.245.12	TCP	62	1161 → 80 [SYN] Seq=232129012 Win=16384 Len=0 MSS=1460 SACK_PER
2	0.023172	128.119.245.12	192.168.1.102	TCP	62	80 → 1161 [SYN, ACK] Seq=883061785 Ack=232129013 Win=5840 Len=0
3	0.023265	192.168.1.102	128.119.245.12	TCP	54	1161 → 80 [ACK] Seq=232129013 Ack=883061786 Win=17520 Len=0
4	0.026477	192.168.1.102	128.119.245.12	TCP	619	1161 → 80 [PSH, ACK] Seq=232129013 Ack=883061786 Win=17520 Len=
5	0.041737	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [PSH, ACK] Seq=232129578 Ack=883061786 Win=17520 Len=
6	0.053937	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=883061786 Ack=232129578 Win=6780 Len=0
7	0.054026	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=232131038 Ack=883061786 Win=17520 Len=1460
8	0.054690	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=232132498 Ack=883061786 Win=17520 Len=1460
9	0.077294	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=883061786 Ack=232131038 Win=8760 Len=0
10	0.077405	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=232133958 Ack=883061786 Win=17520 Len=1460
11	0.078157	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=232135418 Ack=883061786 Win=17520 Len=1460
12	0.124085	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=883061786 Ack=232132498 Win=11680 Len=0
13	0.124185	192.168.1.102	128.119.245.12	TCP	1201	1161 → 80 [PSH, ACK] Seq=232136878 Ack=883061786 Win=17520 Len=
14	0.169118	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=883061786 Ack=232133958 Win=14600 Len=0
15	0.217299	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=883061786 Ack=232135418 Win=17520 Len=0
16	0.267802	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=883061786 Ack=232136878 Win=20440 Len=0
17	0.304807	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=883061786 Ack=232138025 Win=23360 Len=0
18	0.305040	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=232138025 Ack=883061786 Win=17520 Len=1460
19	0.305813	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=232139485 Ack=883061786 Win=17520 Len=1460

```
Frame 4: 619 bytes on wire (4952 bits), 619 bytes captured (4952 bits)
    Encapsulation type: Ethernet (1)
    Arrival Time: Aug 21, 2004 23:44:20.596858000 AEST
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1093095860.596858000 seconds
    [Time delta from previous captured frame: 0.003212000 seconds]
    [Time delta from previous displayed frame: 0.003212000 seconds]
    [Time since reference or first frame: 0.026477000 seconds]
    Frame Number: 4
    Frame Length: 619 bytes (4952 bits)
    Capture Length: 619 bytes (4952 bits)
```

```
0000  00 06 25 da af 73 00 20  e0 8a 70 1a 08 0
0010  02 5d 1e 21 40 00 80 06  a2 e7 c0 a8 01 0
0020  f5 0c 04 89 00 50 0d d6  01 f5 34 a2 74 1
0030  44 70 1f bd 00 00 50 4f  53 54 20 2f 65 7
0040  72 65 61 6c 2d 6c 61 62  73 2f 6c 61 62 3
0050  2d 72 65 70 6c 79 2e 68  74 6d 20 48 54 5
0060  31 2e 31 0d 0a 48 6f 73  74 3a 20 67 61 6
0070  63 73 2e 75 6d 61 73 73  2e 65 64 75 0d 0
0080  65 72 2d 41 67 65 6e 74  3a 20 4d 6f 7a 6
0090  61 2f 35 2e 30 20 28 57  69 6e 64 6f 77 7
00a0  55 3b 20 57 69 6e 64 6f  77 73 20 4e 54 2
```

We subtract the two above values 5.45583 - 0.026477 = 5.429353 seconds.

Hence byte per unit of time we can divide as such:

164090 / 5.429353 = 30223.83 bytes per second.

Exercise 2: TCP Connection Management

Consider the following TCP transaction between a client (10.9.16.201) and a server (10.99.6.175).

No	Source IP	Destination IP	Protocol	Info
295	10.9.16.201	10.99.6.175	TCP	50045 > 5000 [SYN] Seq=2818463618 win=8192 MSS=1460
296	10.99.6.175	10.9.16.201	TCP	5000 > 50045 [SYN, ACK] Seq=1247095790 Ack=2818463619 win=262144 MSS=1460
297	10.9.16.201	10.99.6.175	TCP	50045 > 5000 [ACK] Seq=2818463619 Ack=1247095791 win=65535
298	10.9.16.201	10.99.6.175	TCP	50045 > 5000 [PSH, ACK] Seq=2818463619 Ack=1247095791 win=65535
301	10.99.6.175	10.9.16.201	TCP	5000 > 50045 [ACK] Seq=1247095791 Ack=2818463652 win=262096
302	10.99.6.175	10.9.16.201	TCP	5000 > 50045 [PSH, ACK] Seq=1247095791 Ack=2818463652 win=262144
303	10.9.16.201	10.99.6.175	TCP	50045 > 5000 [ACK] Seq=2818463652 Ack=1247095831 win=65535
304	10.9.16.201	10.99.6.175	TCP	50045 > 5000 [FIN, ACK] Seq=2818463652 Ack=1247095831 win=65535
305	10.99.6.175	10.9.16.201	TCP	5000 > 50045 [FIN, ACK] Seq=1247095831 Ack=2818463652 win=262144
306	10.9.16.201	10.99.6.175	TCP	50045 > 5000 [ACK] Seq=2818463652 Ack=1247095832 win=65535
308	10.99.6.175	10.9.16.201	TCP	5000 > 50045 [ACK] Seq=1247095831 Ack=2818463653 win=262144

Question 1 . What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and server?

In the first row, which is Seq=2818463618

Question 2. What is the sequence number of the SYNACK segment sent by the server to the client computer in reply to the SYN? What is the value of the Acknowledgement field in the SYNACK segment? How did the server determine that value?

In the second row, which is seq=1247095790. Value of Ack=2818463619. The server determined the acknowledgment number by taking the client's initial sequence number (2818463618) and adding 1.

Question 3. What is the sequence number of the ACK segment sent by the client computer in response to the SYNACK? What is the value of the Acknowledgment field in this ACK segment? Does this segment contain any data?

In the third row, Sequence number = 2818463619. ACK value is 1247095790 which is one plus the sequence number of the SYNACK response segment. No data is in this segment

Question 4. Who has done the active close? Is it the client or the server? How you have determined this? What type of closure has been performed? 3 Segment (FIN/FINACK/ACK), 4 Segment (FIN/ACK/FIN/ACK) or Simultaneous close?

In row with No. 304 we can see that the client did the active close since it sent the first FIN (i.e. the source IP was 10.9.16.201 - the client's ip). The server (10.99.6.175) replies with FIN, ACK which is a passive close in packet No. 305. Packets no 306 and 308 are ACK from the client and then the server, respectively making it a 4-segment close:

1. Client to Server: FIN, ACK (packet 304)
2. Server to Client: ACK (packet 305 is FIN, ACK; ACK part is part 2)
3. Server to Client: FIN (same packet 305)
4. Client to Server: ACK (packet 306)

So, it is a proper 4-segment close (FIN / ACK / FIN / ACK)

Question 5. How many data bytes have been transferred from the client to the server and from the server to the client during the whole duration of the connection? What relationship does this have with the Initial Sequence Number and the final ACK received from the other side?

The total data sent = Final ACK received from the other side – Initial Sequence Number (ISN) – 1 (for SYN)

which gives us:

From the client to the server:
Initial Seq: 2818463618 (from SYN in packet 295)
Final ACK received from server is 2818463653 (in packets 305 & 308)
Bytes transferred = 2818463653 - 2818463618 - 1 = 34 bytes

From the server to the client:
Initial Seq: 1247095790 (from SYN-ACK in packet 296)
Final Ack from client: 1247095832 (seen in packet 306)
Bytes transferred = 1247095832 - 1247095790 - 1 = 41 bytes