Lab04 by Katrina

Exercise 1

Q1

IP address of `gaia.cs.umass.edu` (server) is `128.119.245.12`.
Port: `80`.
Client (the host)'s IP address: `192.168.1.102`, port: `1161`.

Q2.

Corresponding sequence#: `232129013`.
Segment data (next page):

```
0000   00 06 25 da af 73 00 20 e0 8a 70 1a 08 00 45 00   ..%..s. ..p...E.
0010   02 5d 1e 21 40 00 80 06 a2 e7 c0 a8 01 66 80 77   .].!@........f.w
0020   f5 0c 04 89 00 50 0d d6 01 f5 34 a2 74 1a 50 18   .....P....4.t.P.
0030   44 70 1f bd 00 00 50 4f 53 54 20 2f 65 74 68 65   Dp....POST /ethe
0040   72 65 61 6c 2d 6c 61 62 73 2f 6c 61 62 33 2d 31   real-labs/lab3-1
0050   2d 72 65 70 6c 79 2e 68 74 6d 20 48 54 54 50 2f   -reply.htm HTTP/
0060   31 2e 31 0d 0a 48 6f 73 74 3a 20 67 61 69 61 2e   1.1..Host: gaia.
0070   63 73 2e 75 6d 61 73 73 2e 65 64 75 0d 0a 55 73   cs.umass.edu..Us
0080   65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 6c 6c   er-Agent: Mozill
0090   61 2f 35 2e 30 20 28 57 69 6e 64 6f 77 73 3b 20   a/5.0 (Windows;
00a0   55 3b 20 57 69 6e 64 6f 77 73 20 4e 54 20 35 2e   U; Windows NT 5.
00b0   31 3b 20 65 6e 2d 55 53 3b 20 72 76 3a 31 2e 30   1; en-US; rv:1.0
00c0   2e 32 29 20 47 65 63 6b 6f 2f 32 30 30 33 30 32   .2) Gecko/200302
00d0   30 38 20 4e 65 74 73 63 61 70 65 2f 37 2e 30 32   08 Netscape/7.02
00e0   0d 0a 41 63 63 65 70 74 3a 20 74 65 78 74 2f 78   ..Accept: text/x
00f0   6d 6c 2c 61 70 70 6c 69 63 61 74 69 6f 6e 2f 78   ml,application/x
0100   6d 6c 2c 61 70 70 6c 69 63 61 74 69 6f 6e 2f 78   ml,application/x
0110   68 74 6d 6c 2b 78 6d 6c 2c 74 65 78 74 2f 68 74   html+xml,text/ht
0120   6d 6c 3b 71 3d 30 2e 39 2c 74 65 78 74 2f 70 6c   ml;q=0.9,text/pl
0130   61 69 6e 3b 71 3d 30 2e 38 2c 76 69 64 65 6f 2f   ain;q=0.8,video/
0140   78 2d 6d 6e 67 2c 69 6d 61 67 65 2f 70 6e 67 2c   x-mng,image/png,
0150   69 6d 61 67 65 2f 6a 70 65 67 2c 69 6d 61 67 65   image/jpeg,image
0160   2f 67 69 66 3b 71 3d 30 2e 32 2c 74 65 78 74 2f   /gif;q=0.2,text/
0170   63 73 73 2c 2a 2f 2a 3b 71 3d 30 2e 31 0d 0a 41   css,*/*;q=0.1..A
0180   63 63 65 70 74 2d 4c 61 6e 67 75 61 67 65 3a 20   ccept-Language:
0190   65 6e 2d 75 73 2c 20 65 6e 3b 71 3d 30 2e 35 30   en-us, en;q=0.50
01a0   0d 0a 41 63 63 65 70 74 2d 45 6e 63 6f 64 69 6e   ..Accept-Encodin
01b0   67 3a 20 67 7a 69 70 2c 20 64 65 66 6c 61 74 65   g: gzip, deflate
01c0   2c 20 63 6f 6d 70 72 65 73 73 3b 71 3d 30 2e 39   , compress;q=0.9
01d0   0d 0a 41 63 63 65 70 74 2d 43 68 61 72 73 65 74   ..Accept-Charset
01e0   3a 20 49 53 4f 2d 38 38 35 39 2d 31 2c 20 75 74   : ISO-8859-1, ut
01f0   66 2d 38 3b 71 3d 30 2e 36 36 2c 20 2a 3b 71 3d   f-8;q=0.66, *;q=
0200   30 2e 36 36 0d 0a 4b 65 65 70 2d 41 6c 69 76 65   0.66..Keep-Alive
0210   3a 20 33 30 30 0d 0a 43 6f 6e 6e 65 63 74 69 6f   : 300..Connectio
0220   6e 3a 20 6b 65 65 70 2d 61 6c 69 76 65 0d 0a 52   n: keep-alive..R
0230   65 66 65 72 65 72 3a 20 68 74 74 70 3a 2f 2f 67   eferer: http://g
0240   61 69 61 2e 63 73 2e 75 6d 61 73 73 2e 65 64 75   aia.cs.umass.edu
0250   2f 65 74 68 65 72 65 61 6c 2d 6c 61 62 73 2f 6c   /ethereal-labs/l
0260   61 62 33 2d 31 2e 68 74 6d 0d 0a                  ab3-1.htm..
```

Q3 & Q4. Question for EstRTT

1. `Sequence number: 232129013`
   `[Time since reference or first frame: 0.026477000 seconds]`
   `TCP segment data (565 bytes)`

Expected ACK: 232129013 + 565 = 232129578

ACK for the 1st segments from server:

```
Acknowledgment number: 232129578
[Time since reference or first frame: 0.053937000 seconds]
```

Time difference: .053937 - .026477 = .02746s
RTT value: 0.02746s

$$EstimatedRTT = 1 * sampleRTT = 0.02746s$$

2.
```
Sequence number: 232129578
[Time since reference or first frame: 0.041737000 seconds]
TCP segment data (1460 bytes)
```

Expected ACK: 232129578 + 1460 = 232131038

ACK for the 2nd segments from server:

```
Acknowledgment number: 232131038
[Time since reference or first frame: 0.077294000 seconds]
```

Time difference: .077294 - .041737 = 0.035557s
RTT value: 0.035557s

$$\begin{aligned} EstimatedRTT &= 0.875 * estimatedRTT + 0.125 * sampleRTT \\ &= 0.875 * 0.02746 + 0.125 * 0.035557 \\ &= 0.028472125s \end{aligned}$$

3.
```
Sequence number: 232131038
[Time since reference or first frame: 0.054026000 seconds]
TCP segment data (1460 bytes)
```

Expected ACK: 232131038 + 1460 = 232132498

ACK for the 3rd segment:

```
Sequence number: 883061786
[Time since reference or first frame: 0.124085000 seconds]
```

Time difference: .124085 - .054026 = 0.070059s
RTT value: 0.070059s

$$\begin{aligned} EstimatedRTT &= 0.875 * estimatedRTT + 0.125 * sampleRTT \\ &= 0.875 * 0.028472125 + 0.125 * 0.070059 \\ &= 0.033670484s \end{aligned}$$

4.
```
Sequence number: 232132498
```

`[Time since reference or first frame: 0.054690000 seconds]`
`TCP segment data (1460 bytes)`

Expected ACK: 232132498 + 1460 = 232133958

ACK for the 4th segment:
`Acknowledgment number: 232133958`
`[Time since reference or first frame: 0.169118000 seconds]`

Time difference: .169118 - .05469 = 0.114428s
RTT value: 0.114428s

$$\begin{aligned} EstimatedRTT &= 0.875 * estimatedRTT + 0.125 * sampleRTT \\ &= 0.875 * 0.033670484 + 0.125 * 0.114428 \\ &= 0.043765174s \end{aligned}$$

5. `Sequence number: 232133958`
`[Time since reference or first frame: 0.077405000 seconds]`
`TCP segment data (1460 bytes)`

Expected ACK: 232133958 + 1460 = 232135418

ACK for the 5th segment:
`Acknowledgment number: 232135418`
`[Time since reference or first frame: 0.217299000 seconds]`

Time differnece: .217299 - .077405 = 0.139894s
RTT value: 0.139894s

$$\begin{aligned} EstimatedRTT &= 0.875 * estimatedRTT + 0.125 * sampleRTT \\ &= 0.875 * 0.043765174 + 0.125 * 0.139894 \\ &= 0.055781277s \end{aligned}$$

6. `Sequence number: 232135418`
`[Time since reference or first frame: 0.078157000 seconds]`
`TCP segment data (1460 bytes)`

Expected ACK: 232135418 + 1460 = 232136878

ACK for the 6th segment:
`Acknowledgment number: 232136878`
`[Time since reference or first frame: 0.267802000 seconds]`

Time differnece: .267802 - .078157 = 0.189645s
RTT value: 0.189645s

$$
\begin{aligned}
EstimatedRTT &= 0.875 * estimatedRTT + 0.125 * sampleRTT \\
&= 0.875 * 0.055781277 + 0.125 * 0.189645 \\
&= 0.072514242s
\end{aligned}
$$

Q5.

The minimum buffer space related to the minimum window size during the connection, which is `16384`. Yes, if there is lack of receiver buffer space, the window size sent to sender will be restrict and throttle the sender.

Q6.

There are no retransmitted segments because there isn't any repeating ACK number and all of the sequence number sent by sender are in increasing order, which means the sender never (re)sent an "old" segment.

Q7.

By observing the trace, finding that initially an ACK acknoledged 1460 bytes of data (from TCP segment length of corresponding segment). However, from segment No. 69, TCP started to acknowledge every two segments from sender, which contains 2 * 1460 = 2920 bytes data.

Q8.

Brain storming:
Throughput = data transmitted/time.
In this trace, consider "time" as the total time of connection, which is the difference between when the first POST HTTP segment was sent (0.026477s), and when the last ACK is received (5.45583s).
`[Time since reference or first frame: 5.455830000 seconds]` Also, as the ACK is actually the received seq# + segment size, and this ACK will be the next seq# sent from receiver. That means the data transmitted in packet = ACK# - corresponding seq#. Hence, the total data transmitted can be calculator by the last ACK# received (232293103) - first seq# (232129013).

Hence, the answer is:

$$
\begin{aligned}
\text{throughput} &= \text{data / time} \\
&= (232293103 - 232129013)/(5.45583 - 0.026477) \\
&= 30222.753982 \ \text{bytes/sec}
\end{aligned}
$$

Exercise 2

Q1.

sequence number of TCP SYN: `2818463618`.

Q2.

SYNACK sequence number: `1247095790`.
Acknowledgement number: `2818463619`. This is the next bit (expected seq to received next) of the sequence number of the received pkt. To compute the value, simply add 1 to the seq# (the initial seq# in this case) in received pkt.

Q3.

sequence nubmer: `2818463619`.
ACK nubmer: `1247095791`.
No, ACK doesn't contain data, it only contain ACKbit in header.

Q4.

Both client and server have done the active close as they send FIN to each other. As the client and the server sent the FIN at the same time, it's simultaneous close.

Q5.

No.301 & No.298 (client -> server):
2818463652 - 2818463619 = 33 bytes

No.302 & No.303 (server -> client):
1247095831 - 1247095791 = 40 bytes

Totally `33 + 40 = 73 bytes` were sent during the connection.

They are equal (both `2818463618`).