Assignment 6

This is the intellectual work of **171860611, 王麦迪**.

Question 1

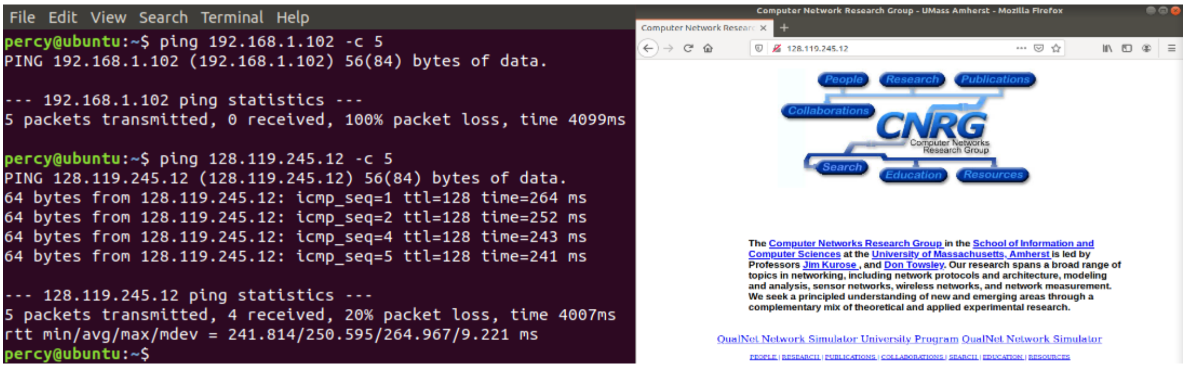
What is the IP address and TCP port numbers used by the client and the server?

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.102	128.119.245.12	TCP	62	1161 → 80
2	0.023172	128.119.245.12	192.168.1.102	TCP	62	80 → 1161
3	0.023265	192.168.1.102	128.119.245.12	TCP	54	1161 → 80

	Client	Server
IP address	192.168.1.102	128.119.245.12
Port number	1161	80

Question 2

Can you ping or visit above addresses? Why?



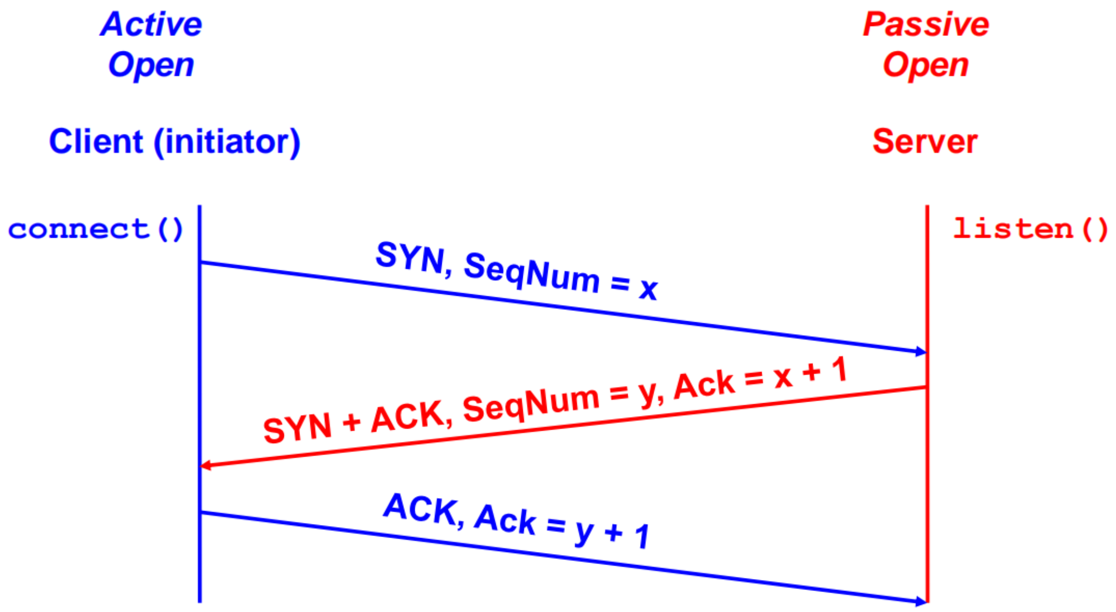
`192.168.1.102` is not ping reachable. `128.119.245.12` is ping reachable. `128.119.245.12` is actually the server IP for **University of Massachusetts's Computer Networks Research Group**.

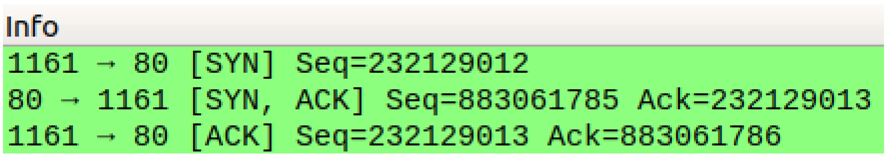
`192.168.1.102` is a private IPv4 address. All private addresses are used inside private networks and can't be routed on the Internet. Personally, I would guess that this host is behind an NAT and was assigned `192.168.1.102` by the DHCP server.

`128.119.245.12` is a public IPv4 address, so it should be reachable unless "magical" borders say otherwise.

Question 3

You were shown the following TCP handshake process in class, can you find the the progress sequence of x and y?





Info
1161 → 80 [SYN] Seq=232129012
80 → 1161 [SYN, ACK] Seq=883061785 Ack=232129013
1161 → 80 [ACK] Seq=232129013 Ack=883061786

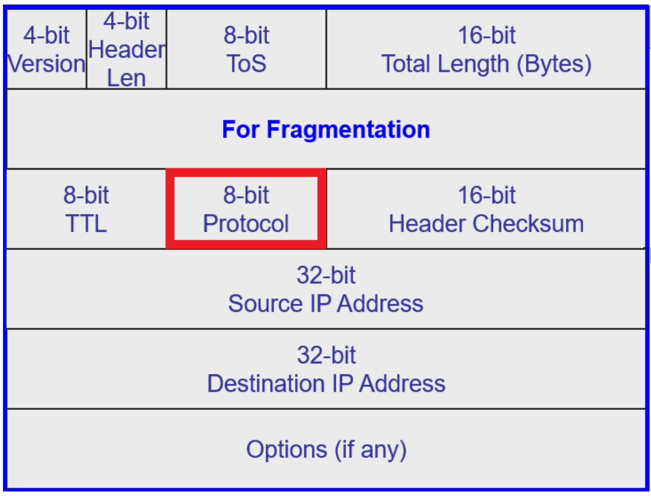
I have disabled the `relative sequence number` for TCP in Wireshark to get the actual sequence numbers, and as you can see from above:

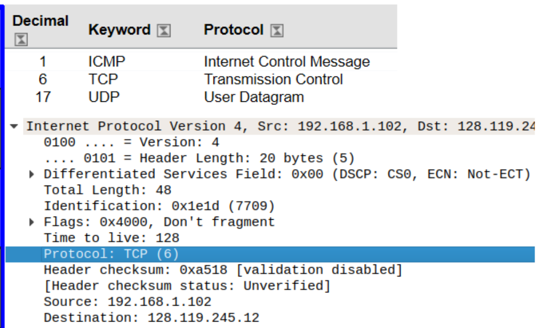
$$x = 232129012$$
$$y = 883061785$$

Question 4

How do we know if a packet is using TCP or UDP by only looking at a IP header?

What is the checksum for packet NO.1 TCP packet and how many bits can it check?





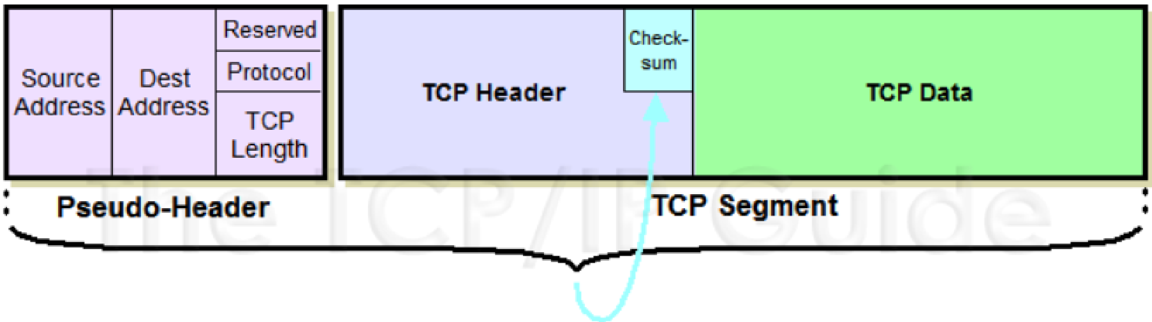
Decimal	Keyword	Protocol
1	ICMP	Internet Control Message
6	TCP	Transmission Control
17	UDP	User Datagram

```
▾ Internet Protocol Version 4, Src: 192.168.1.102, Dst: 128.119.24
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 48
    Identification: 0x1e1d (7709)
  ▸ Flags: 0x4000, Don't fragment
    Time to live: 128
    Protocol: TCP (6)
    Header checksum: 0xa518 [validation disabled]
    [Header checksum status: Unverified]
    Source: 192.168.1.102
    Destination: 128.119.245.12
```

We can obtain this information via the `Protocol` field in the IP header. 6 stands for `TCP` and 17 stands for `UDP`.

```
▾ Transmission Control Protocol, Src Port: 1161, Dst Port: 80, Seq: 232129012, Len: 0
    Source Port: 1161
    Destination Port: 80
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 232129012
    [Next sequence number: 232129012]
    Acknowledgment number: 0
    0111 .... = Header Length: 28 bytes (7)
  ▸ Flags: 0x002 (SYN)
    Window size value: 16384
    [Calculated window size: 16384]
    Checksum: 0xf6e9 [correct]
    [Checksum Status: Good]
    [Calculated Checksum: 0xf6e9]
    Urgent pointer: 0
  ▸ Options: (8 bytes), Maximum segment size, No-Operation (NOP), No-Operation (NOP),
  ▸ [Timestamps]
```

The checksum is `0xf6e9`.



Checksum Calculated Over Pseudo Header and TCP Segment

The `Checksum` field in TCP is computed over a pseudo header plus the original TCP segment. But there is a "chicken and egg" situation here since the `Checksum` field is part of the original TCP segment. The normal approach is to set the `Checksum` field to zero for the calculation. In other words, the checksum field field itself doesn't count as the part that is covered in the `Checksum` field. So the `Checksum` field can check:

$$\begin{aligned} & Pseudo\ Header + TCP\ Segment - Checksum \\ &= 12\ bytes + 28\ bytes - 2\ bytes \\ &= 38\ bytes \\ &= 304\ bits \end{aligned}$$

Question 5

Which link layer protocol is used?

What is the value of MTU and MSS?

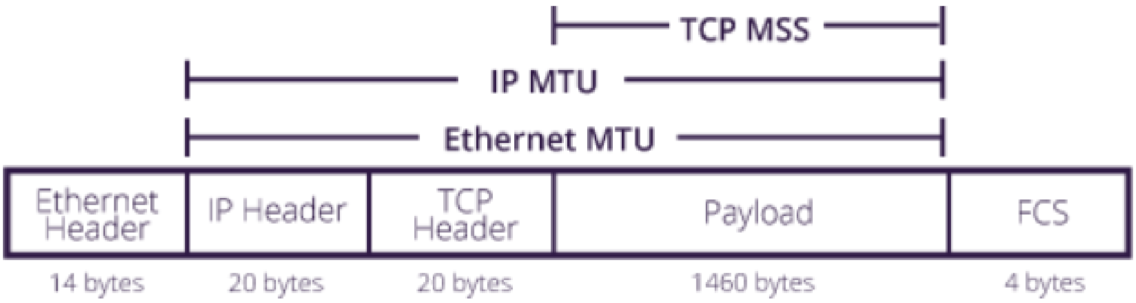
Why MSS != MTU or in other words, what's the value of MTU - MSS?

```
▸ Frame 1: 62 bytes
▸ Ethernet II
▸ Internet Protocol Version 4
▸ Transmission Control Protocol
```

The link layer protocol is `Ethernet II`.

```
▸ TCP Option - Maximum segment size: 1460 bytes
```

Since it is using Ethernet, MTU should be 1500 bytes. From the screenshot above, MSS is 1460 bytes.

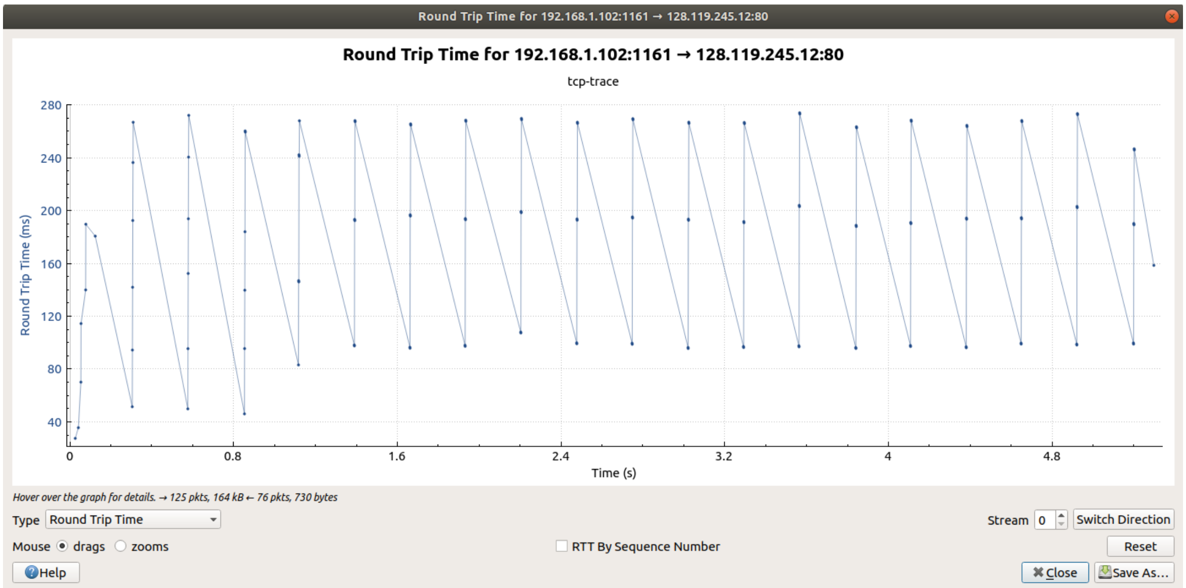


MTU - MSS is 40 bytes, exactly the combined header length of a TCP header(without options) and an IP header(without options). They are different because MSS represents the maximum size of TCP data payload while MTU represents the maximum size of Ethernet data payload.

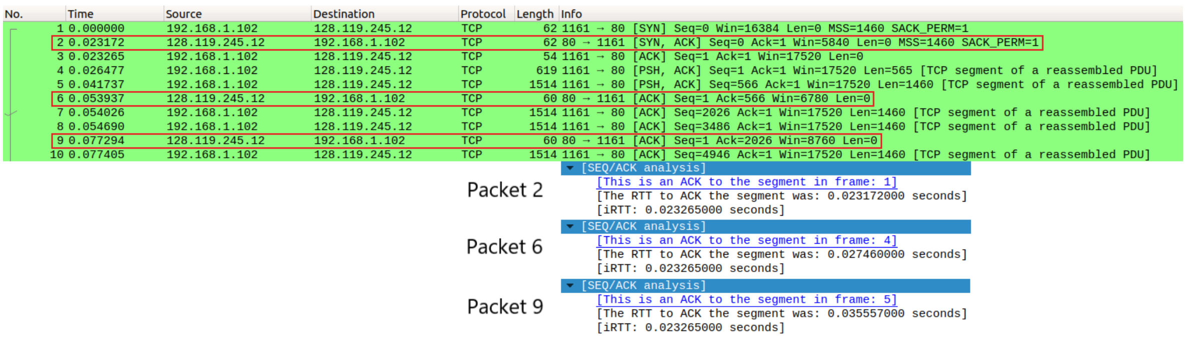
Question 6

Draw the RTT and RTT estimate graph like in slides42 with the first 10 packets(NO.1~10) and with `alpha = 0.8`.

Wireshark itself has an RTT calculation function in `Statistics -> TCP Stream Graphs -> Round Trip Time`, and it looks like this:



After figuring out how Wireshark calculated this graph, I found that there are only three `ACKs` coming back from the server in the first 10 packets(I activated the `Analyze TCP sequence numbers` in `Preferences` to make things easier):

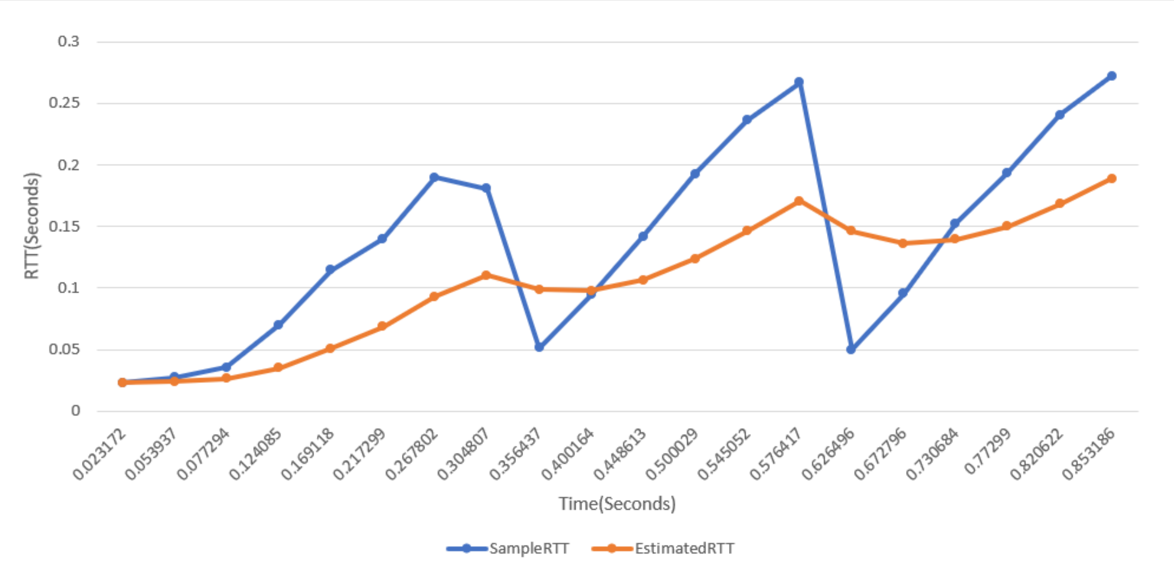


No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.102	128.119.245.12	TCP	62	1161 → 80 [SYN] Seq=0 Win=16384 Len=0 MSS=1460 SACK_PERM=1
2	0.023172	128.119.245.12	192.168.1.102	TCP	62	80 → 1161 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460 SACK_PERM=1
3	0.023265	192.168.1.102	128.119.245.12	TCP	54	1161 → 80 [ACK] Seq=1 Ack=1 Win=17520 Len=0
4	0.026477	192.168.1.102	128.119.245.12	TCP	619	1161 → 80 [PSH, ACK] Seq=1 Ack=1 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=566 Ack=1 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=1 Ack=566 Win=6780 Len=0
7	0.054026	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=2026 Ack=1 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=3486 Ack=1 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=1 Ack=2026 Win=8760 Len=0
10	0.077405	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=4946 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]

According to the `SEQ/ACK analysis` provided by Wireshark, also rechecked by my observation of sequence numbers and window sizes, these 3 `ACKs` correspond in this way:

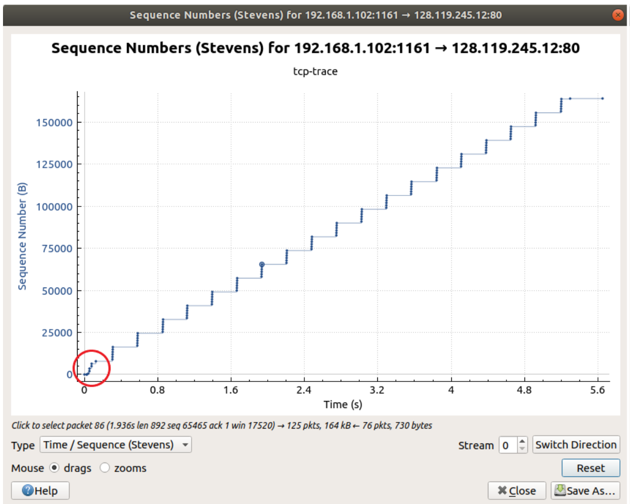
Packet	Acknowledging Packet Sent by Host
2	1
6	4
9	5

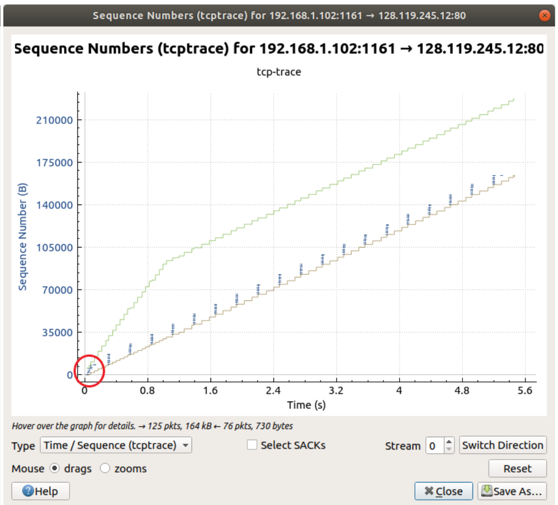
To be frank, I know TA wants to make things easier for us, but I don't see the point in drawing a graph with only 3 points, so I enlarged the range to plot. My initial `Estimated RTT` is the first RTT calculated:



Question 7

Have you observed the "slow start" and "congestion avoid" states? Is that same to what we learned("slow start RTT" "linear increase window size")? Why?





The part in the red circle should be the slow start phase. The rest should be the congestion avoid phase.

But this doesn't seem to be same as what we learned in class. When we enter the congestion avoid phase, the TCP window doesn't increase anymore, but instead sends 6 packets every RTT. In other words, there is no linear increase window size, the window size is fixed.