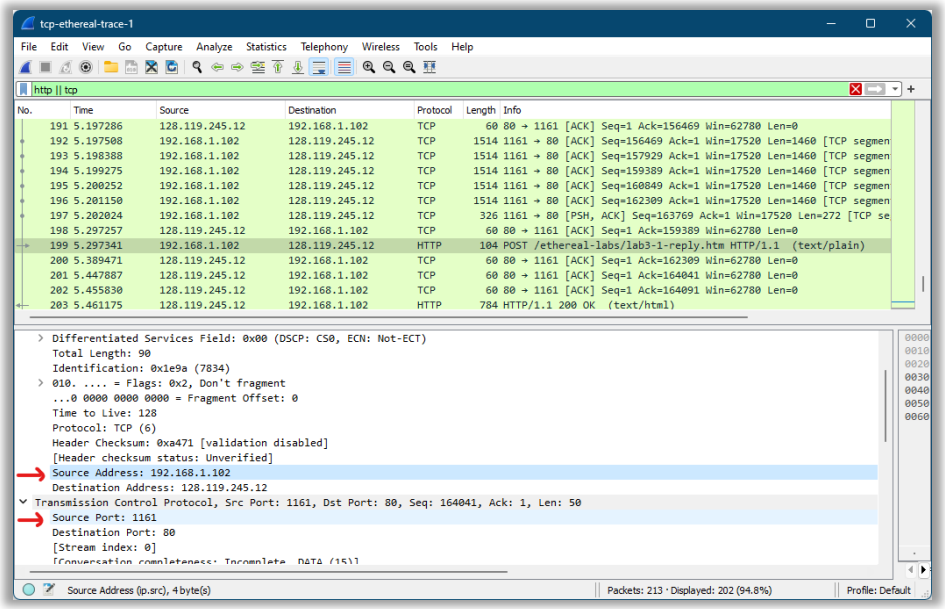
WireShark Lab 05 - TCP v7.0

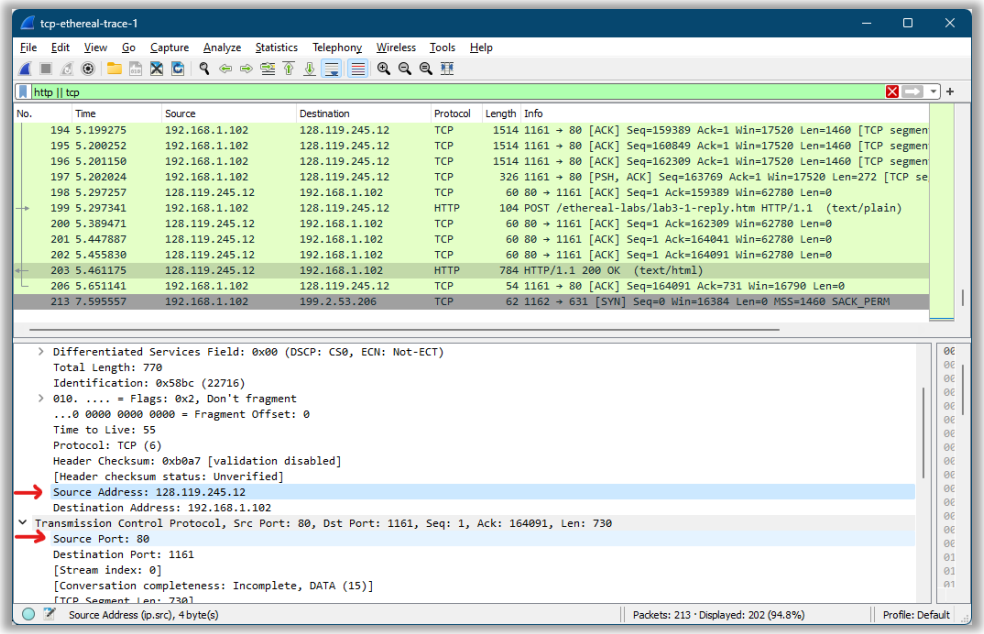
Igor Augusto Reis Gomes – 12011BSI290 – igor.augusto@ufu.br

Heitor Guimarães Da Fonseca Filho – 12011BSI203 – heitor.filho@ufu.br

1) **What is the IP address and TCP port number used by the client computer (source) that is transferring the file to gaia.cs.umass.edu?**
   - **a)** Endereço IP: 192.168.1.102
   - **b)** Porta TCP: 1161



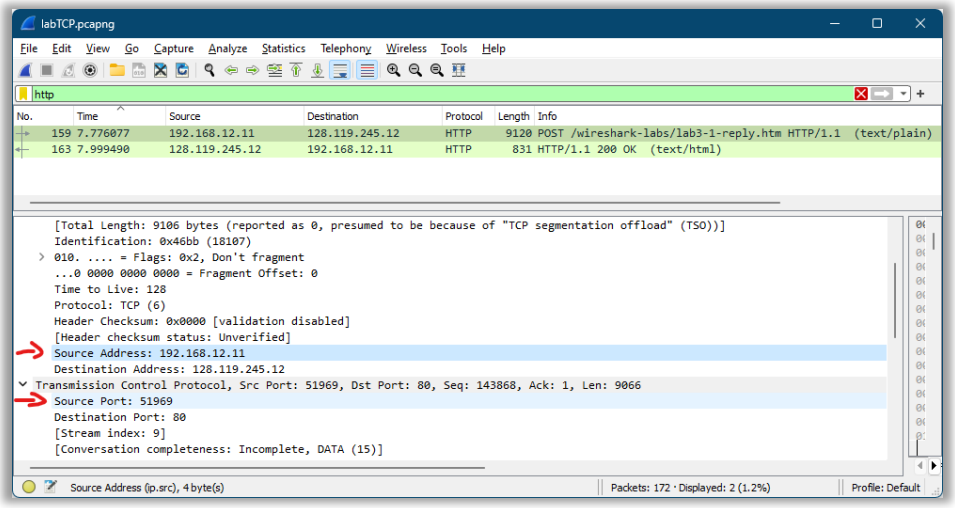
2) **What is the IP address of gaia.cs.umass.edu? On what port number is it sending and receiving TCP segments for this connection?**
   - **a)** Endereço IP: 128.119.245.12
   - **b)** Porta TCP: 80



3) What is the IP address and TCP port number used by your client computer (source) to transfer the file to gaia.cs.umass.edu?

a) Os prints anteriores foram feitos utilizando a captura já pronta fornecida, abaixo segue aquela na qual nós fizemos, fazendo upload do arquivo “alice.txt” no site indicado.

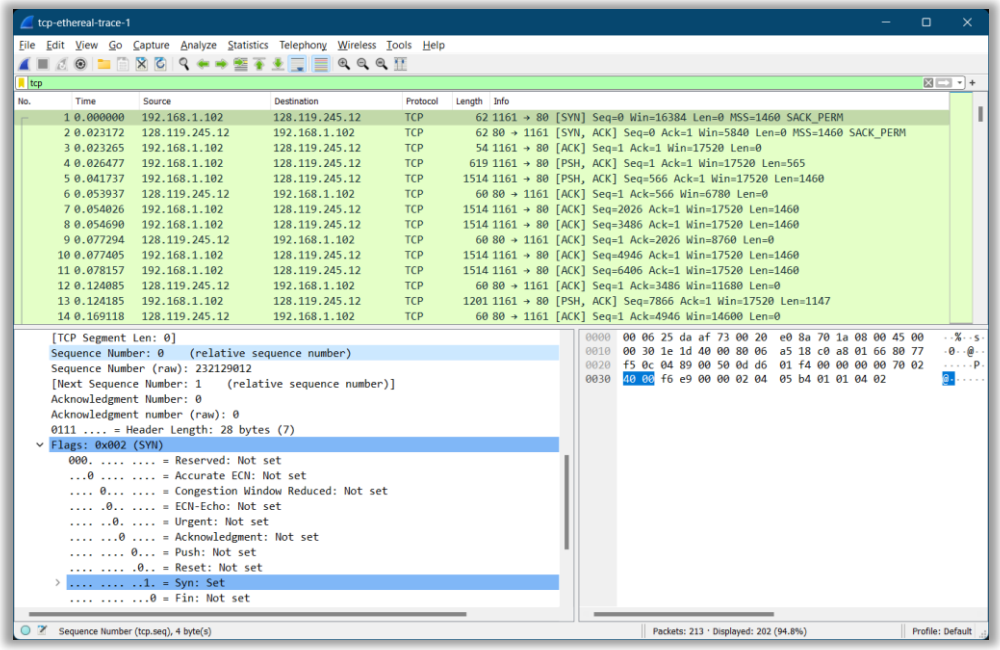
b) Endereço IP de origem: 192.168.12.11 / de destino: 128.119.245.12.
Porta TCP de origem: 51969 / de destino: 80.



4) What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and gaia.cs.umass.edu? What is it in the segment that identifies the segment as a SYN segment?

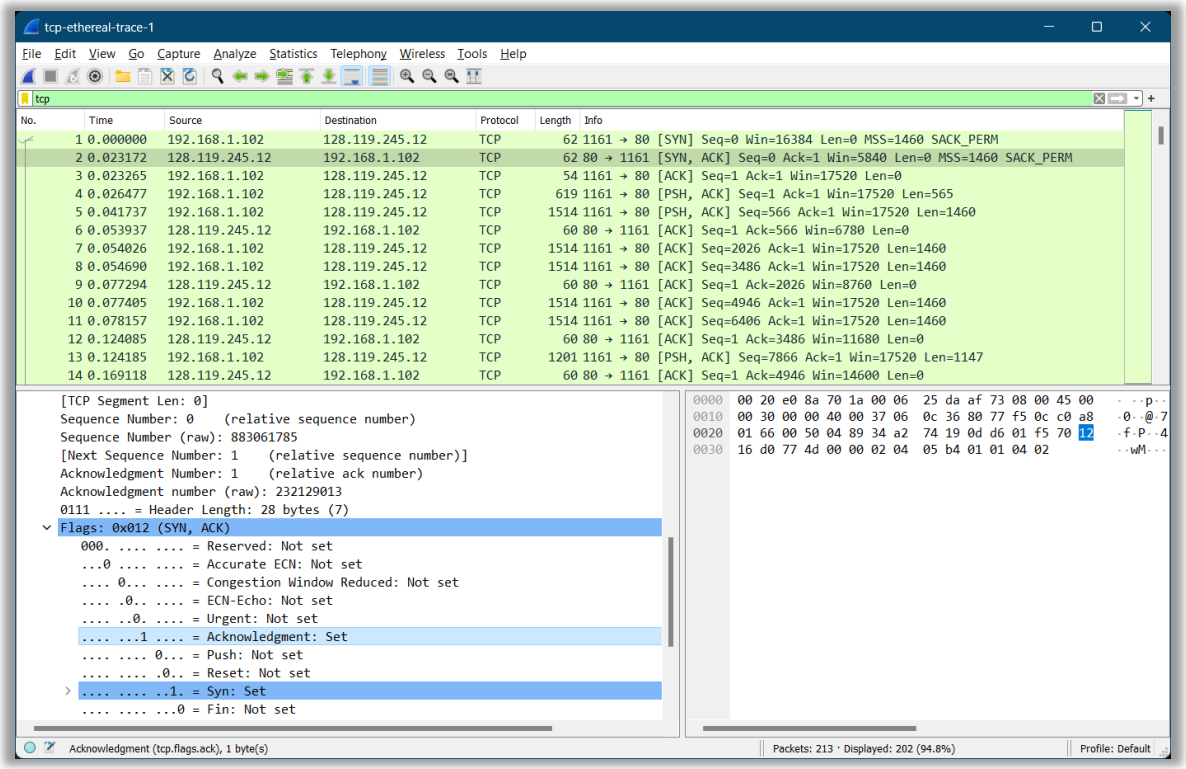
a) Número de sequência: 0 (zero).

b) No campo de flags, é possível identificar a flag SYN com valor “setado” (definido) para 1 (bit), ou seja, indicando que o segmento é de fato um segmento SYN.



5) What is the sequence number of the SYNACK segment sent by gaia.cs.umass.edu to the client computer in reply to the SYN? What is the value of the Acknowledgement field in the SYNACK segment? How did gaia.cs.umass.edu determine that value? What is it in the segment that identifies the segment as a SYNACK segment?

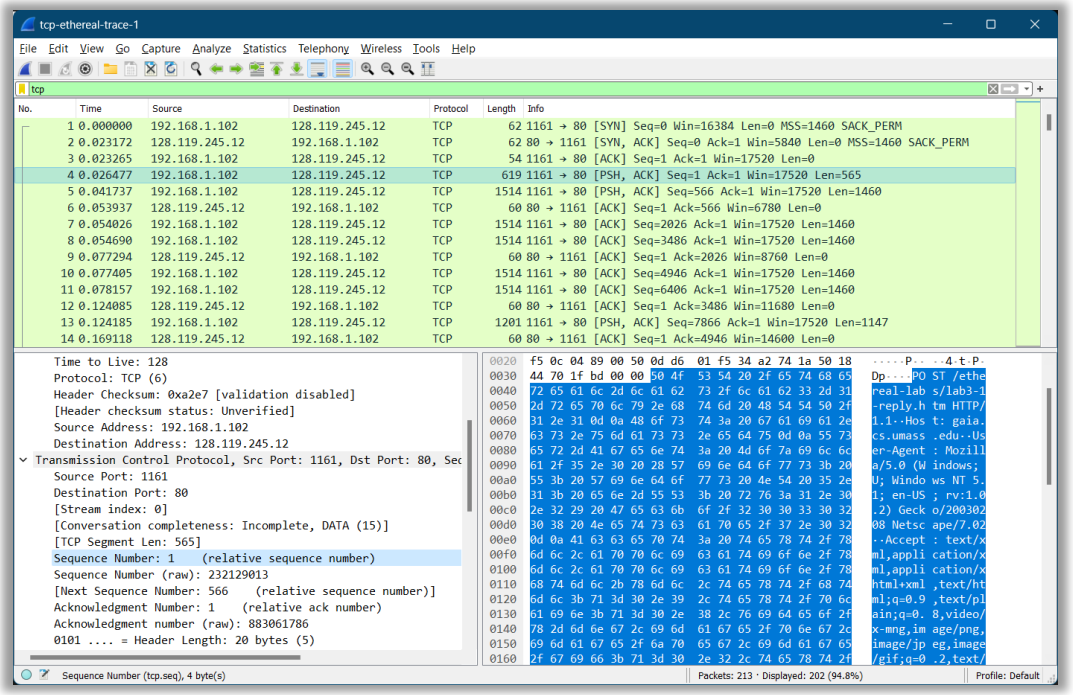
a) Número de sequência: 0 (zero).
b) Número de Acknowledgment (ACK): 1.
c) O valor do campo de Acknowledgment é determinado pelo servidor gaia.cs.umass.edu adicionando 1 ao número de sequência original do segmento SYN do cliente.
d) Um segmento vai ser identificado como SYNACK se ambas as flags SYN e Acknowledgment no segmento estiverem setadas para o valor 1.



6) **What is the sequence number of the TCP segment containing the HTTP POST command? Note that in order 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.**

a) Número de sequência do segmento TCP que contêm o comando HTTP Post: 1 (um).

Indicado abaixo no campo de data, há também o comando POST na janela da direita.



7) **Consider the TCP segment containing the HTTP POST as the first segment in the TCP connection. What are the sequence numbers of the first six segments in the TCP connection (including the segment containing the HTTP POST)?**

Next Sequence Number (NSN) permite saber o próximo pacote da sua sequência.

- **i)** Pacote 4, seq: 1, nsn: 566
- **ii)** Pacote 5, seq: 566, nsn: 2026
- **iii)** Pacote 7, seq: 2026, nsn: 3486
- **iv)** Pacote 8, seq: 3486, nsn: 4946
- **v)** Pacote 10, seq: 4946, nsn: 6406
- **vi)** Pacote 11, seq: 6406

At what time was each segment sent?

- **i)** Pacote 4, tempo: 0.026477
- **ii)** Pacote 5, tempo: 0.041737
- **iii)** Pacote 7, tempo: 0.054026
- **iv)** Pacote 8, tempo: 0.054690
- **v)** Pacote 10, tempo: 0.077405
- **vi)** Pacote 11, tempo: 0.078157

When was the ACK for each segment received?

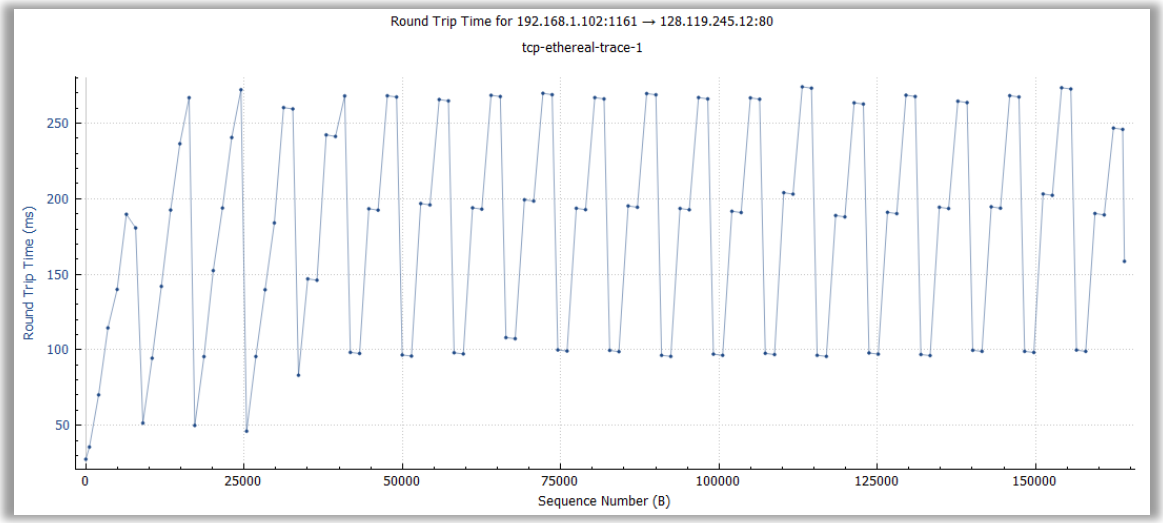
- **i)** Pacote 5, tempo: 0.041737
- **ii)** Pacote 6, tempo: 0.053937
- **iii)** Pacote 9, tempo: 0.077294
- **iv)** Pacote 12, tempo: 0.124085
- **v)** Pacote 14, tempo: 0.169118
- **vi)** Pacote 15, tempo: 0.217299

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
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
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]
11	0.078157	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=6406 Ack=1 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=1 Ack=3486 Win=11680 Len=0
13	0.124185	192.168.1.102	128.119.245.12	TCP	1201	1161 → 80 [PSH, ACK] Seq=7866 Ack=1 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=1 Ack=4946 Win=14600 Len=0
15	0.217299	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=6406 Win=17520 Len=0
16	0.267802	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=7866 Win=20440 Len=0

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?

RTT = ACK - SEQ

- **i)** RTT = 0.041737 - 0.026477 = 0.015260
- **ii)** RTT = 0.053937 - 0.041737 = 0.012200
- **iii)** RTT = 0.077294 - 0.054026 = 0.023268
- **iv)** RTT = 0.124085 - 0.054690 = 0.069395
- **v)** RTT = 0.169118 - 0.077405 = 0.091713
- **vi)** RTT = 0.217299 - 0.078157 = 0.139142



What is the EstimatedRTT value (see Section 3.5.3, page 242 in text) after the receipt of each ACK?

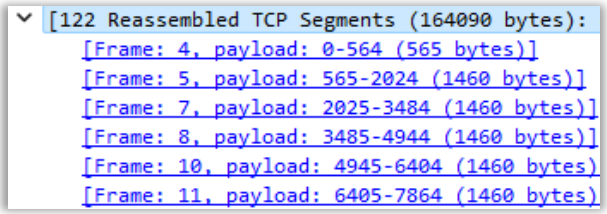
$$ERTT = (1 - \alpha) * ERTT + \alpha * SampleRTT$$

Assumindo ERTT0 = 0.1s e α = 0.125

i) ERTT1 = (1 - 0.125) * 0.1 + 0.125 * 0.015260 ≈ 0.103815
ii) ERTT2 = (1 - 0.125) * 0.103815 + 0.125 * 0.012200 ≈ 0.10107375
iii) ERTT3 = (1 - 0.125) * 0.10107375 + 0.125 * 0.023268 ≈ 0.09827046875
iv) ERTT4 = (1 - 0.125) * 0.09827046875 + 0.125 * 0.069395 ≈ 0.09724648828
v) ERTT5 = (1 - 0.125) * 0.09724648828 + 0.125 * 0.091713 ≈ 0.09669592203
vi) ERTT6 = (1 - 0.125) * 0.09669592203 + 0.125 * 0.139142 ≈ 0.09667026525

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

i) Pacote 4, tamanho: 565
ii) Pacote 5, tamanho: 1460
iii) Pacote 7, tamanho: 1460
iv) Pacote 8, tamanho: 1460
v) Pacote 10, tamanho: 1460
vi) Pacote 11, tamanho: 1460

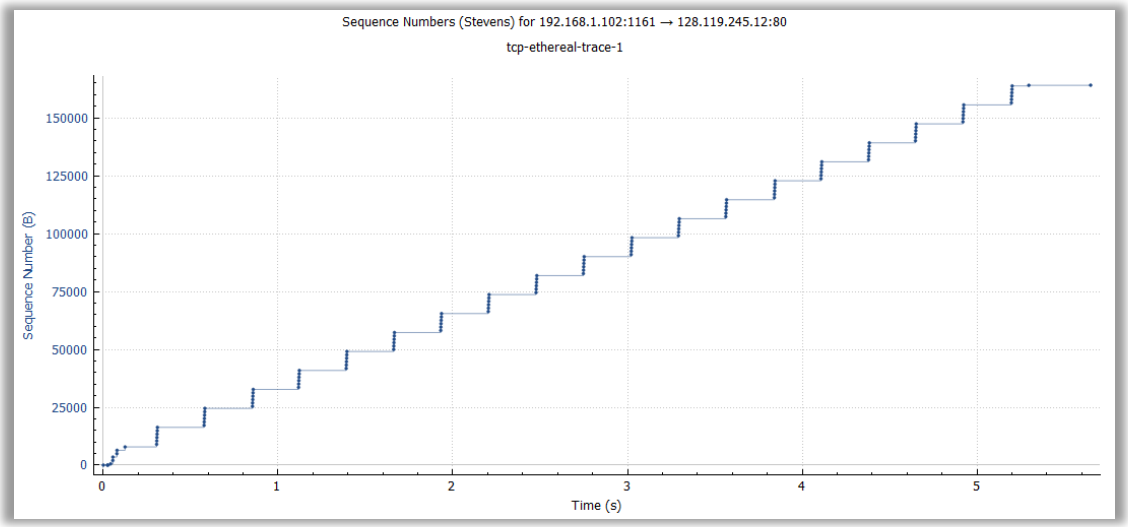


9) What is the minimum amount of available buffer space advertised at the received for the entire trace? Does the lack of receiver buffer space ever throttle the sender?

A quantidade mínima é definida pelo campo Window = 17520 (bytes).
Não, pois o buffer continuou a aumentar.

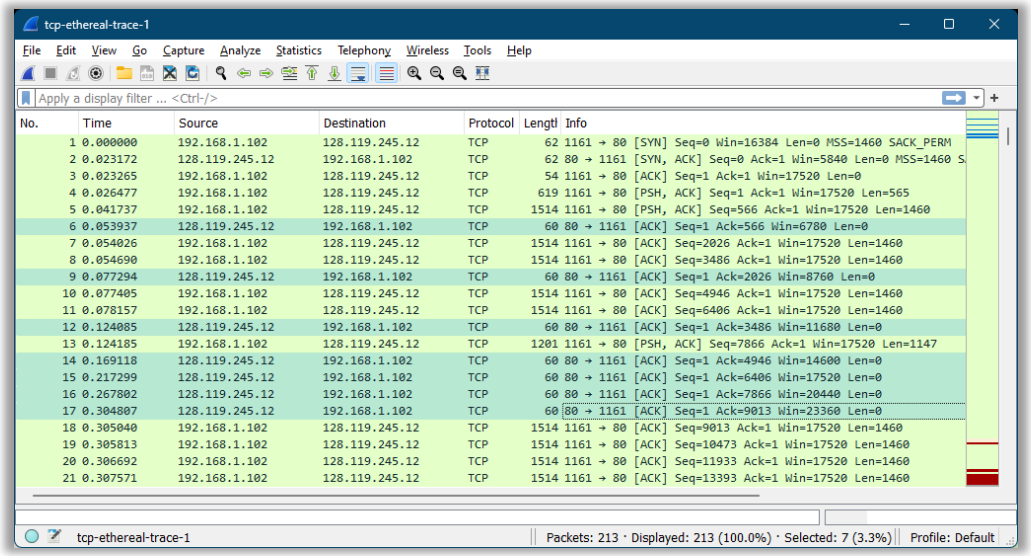
10) Are there any retransmitted segments in the trace file? What did you check for (in the trace) in order to answer this question?

Não há segmentos retransmitidos no arquivo. Utilizei o gráfico de RTT do WireShark (obtido por meio de Statistics -> TCP Stream Graphs -> Time Sequence (Stevens)), como é possível observar no print abaixo, todos os números de sequência da origem para o destino estão aumentando monotonicamente em relação ao tempo, não indicando retransmissão.



11) How much data does the receiver typically acknowledge in an ACK? Can you identify cases where the receiver is ACKing every other received segment (see Table 3.2 on page 250 in the text).

É possível observar abaixo que os números ACK aumentam na sequência de 566, 2026, 3486, 4946, 6406, 7866, 9013, etc. Eles estão aumentando em 1460 (40 header TCP/IP + 1460 payload) cada vez, indicando que o receptor está reconhecendo 1460 bytes por vez, sendo este valor é o mesmo do campo “length”.

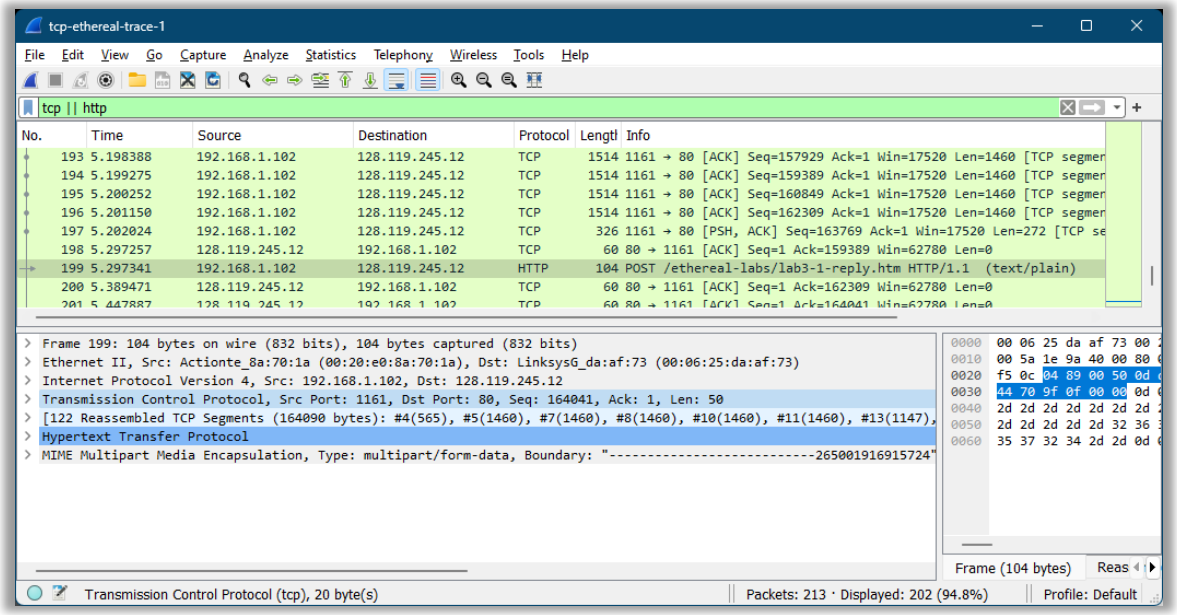


12) What is the throughput (bytes transferred per unit time) for the TCP connection? Explain how you calculated this value.

A taxa de transferência (throughput) pode ser obtida pela seguinte fórmula:

$$throughput = \frac{\text{total de bytes transferidos}}{\text{intervalo de tempo}} = \frac{164090}{5.297341 - 0.023265} \approx 31{,}180.36 \text{ bytes}$$

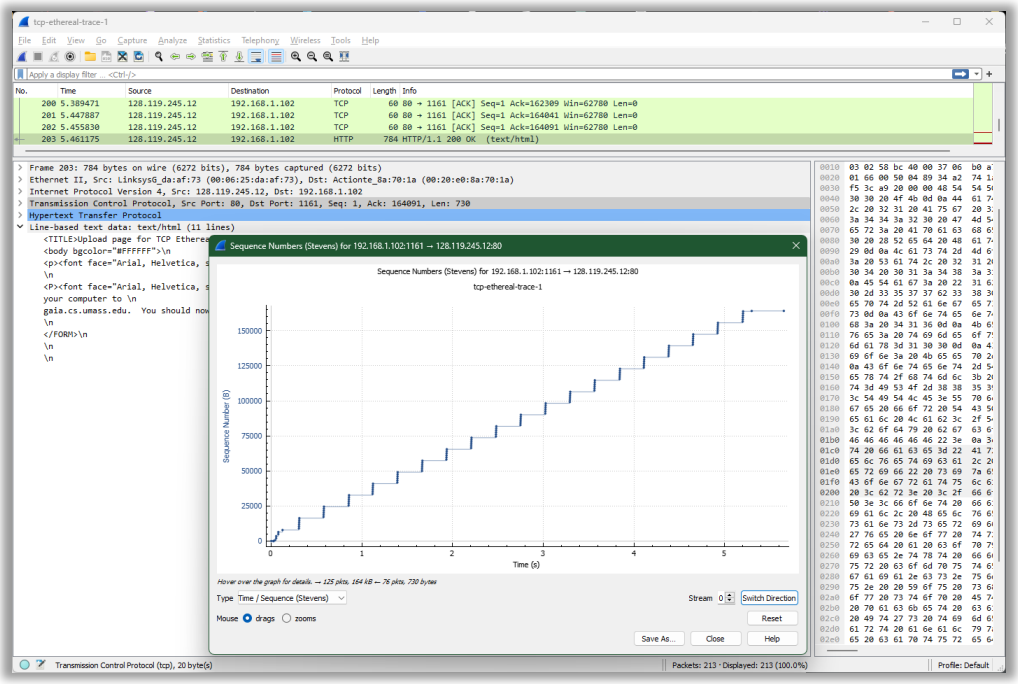
$$= \frac{31{,}180.36}{1024} \approx 30.42\ KB/s$$



13) 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 gaia.cs.umass.edu server. Can you identify where TCP's slowstart phase begins and ends, and where congestion avoidance takes over? Comment on ways in which the measured data differs from the idealized behavior of TCP that we've studied in the text.

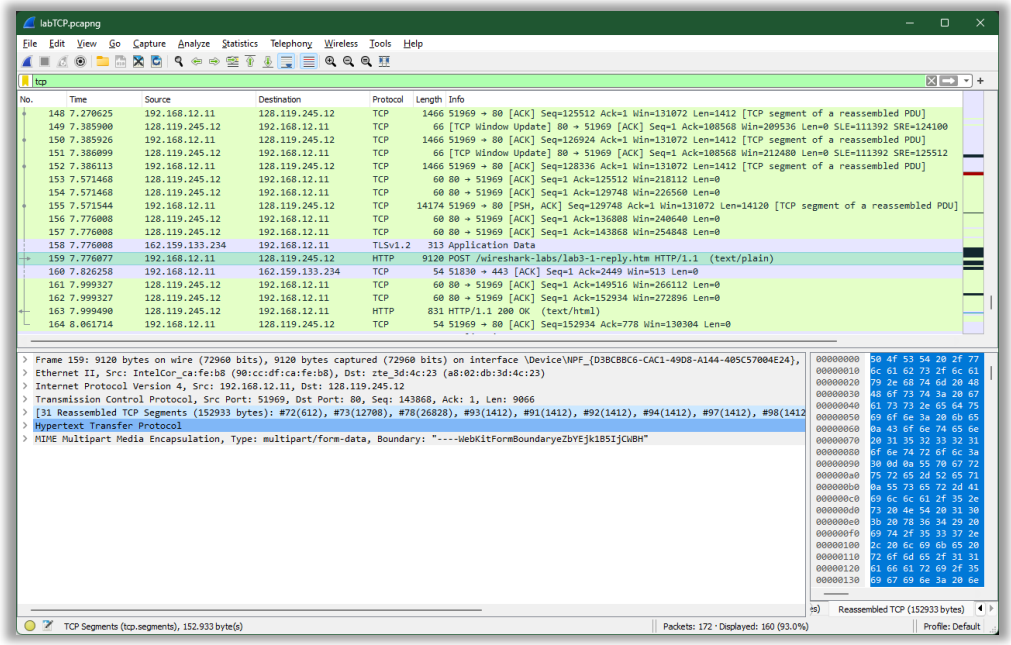
O início lento (slowstart) começa em 0,02327s e termina em 5,651s, assim:

$$slowstart = 5.651 - 0.02327 = 5{,}62773\ segundos$$



14) Answer each of two questions above for the trace that you have gathered when you transferred a file from your computer to gaia.cs.umass.edu

a) 12. $throughput = \frac{152933}{7.776077 - 5.371122} \approx 76{,}466.5 \text{ bytes} = \frac{31{,}180.36}{1024} \approx 74.75\ KB/s$



b) 13. O início lento (slowstart) começa em 5,577s e termina em 7.776s, assim:

$$slowstart = 7.776 - 5.577 = 2.199\ segundos$$