

中国科学技术大学计算机学院

计算机网络实验报告

实验三

利用 Wireshark 观察 TCP 报文

学 号：PE20060014

姓 名：王晨

专 业：计算机科学与技术

指导老师：张信明

中国科学技术大学计算机学院

2020 年 11 月 11 日

一、 实验目的

- 1、 熟悉并掌握 Wireshark 的使用方法。
- 2、 通过捕获观察并分析 TCP 报文，理解 TCP 的细节，包括：为了可靠传输的 SEQ、ACK 序号使用；TCP 的拥塞控制算法-慢启动和拥塞避免；TCP 的流量控制机制；TCP 连接的建立。

二、 实验原理

Wireshark 是非常流行的网络封包分析软件，功能十分强大。可以截取各种网络封包，显示网络封包的详细信息。Wireshark 使用 Npcap 作为接口，直接与网卡进行数据报文交换，监听共享网络上传送的数据包。Npcap 是替代 WinPcap 的新型 Windows 网络数据包截获软件。能够比原有的 WinPcap 数据包获得更好的抓包性能，并且稳定性更好。

三、 实验条件

- 1、 硬件条件：Mac
- 2、 软件条件：Mac OS Catalina

Chrome Web Browser

Wireshark 3.2.7

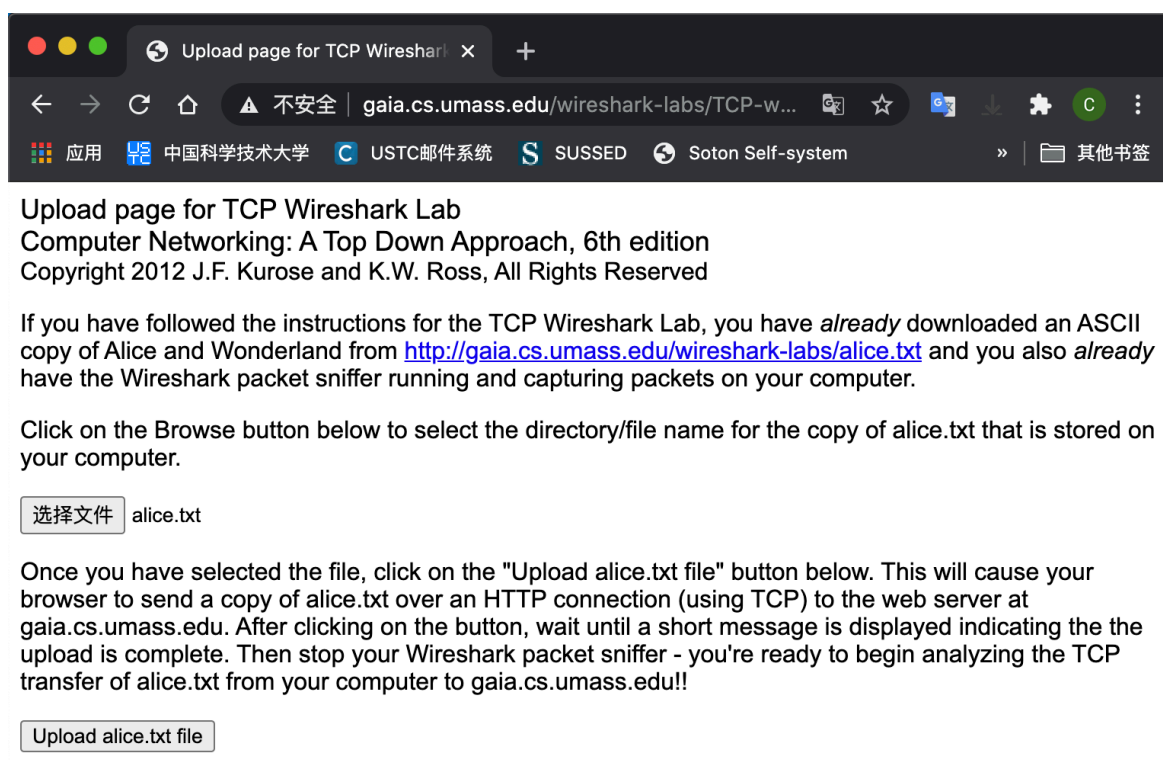
四、 实验过程

1. Capturing a bulk TCP transfer from your computer to a remote server

访问 <http://gaia.cs.umass.edu/wiresharklabs/alice.txt> , 下载 alice.txt, 保存在本地;

访问

<http://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html>, 选择文件 alice.txt, 但是先不要上传;



Upload page for TCP Wireshark Lab
Computer Networking: A Top Down Approach, 6th edition
Copyright 2012 J.F. Kurose and K.W. Ross, All Rights Reserved

If you have followed the instructions for the TCP Wireshark Lab, you have *already* downloaded an ASCII copy of Alice and Wonderland from <http://gaia.cs.umass.edu/wireshark-labs/alice.txt> and you also *already* have the Wireshark packet sniffer running and capturing packets on your computer.

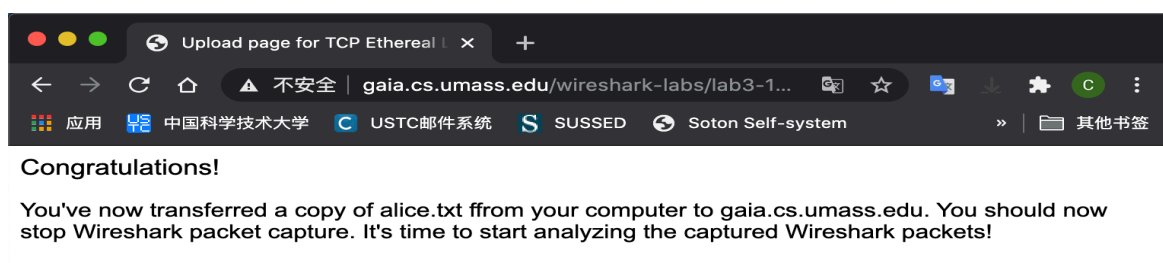
Click on the Browse button below to select the directory/file name for the copy of alice.txt that is stored on your computer.

alice.txt

Once you have selected the file, click on the "Upload alice.txt file" button below. This will cause your browser to send a copy of alice.txt over an HTTP connection (using TCP) to the web server at gaia.cs.umass.edu. After clicking on the button, wait until a short message is displayed indicating the upload is complete. Then stop your Wireshark packet sniffer - you're ready to begin analyzing the TCP transfer of alice.txt from your computer to gaia.cs.umass.edu!!

打开 wireshark, 开始捕获;

切回浏览器, 开始上传;

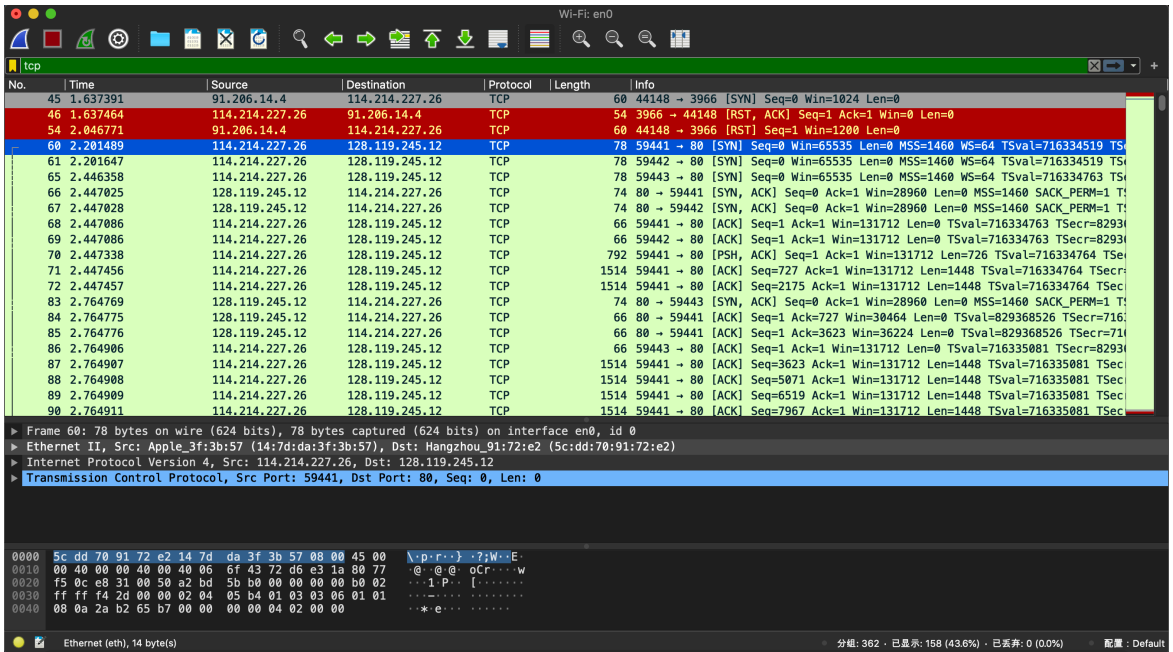


Upload page for TCP Ethernet Lab
Computer Networking: A Top Down Approach, 6th edition
Copyright 2012 J.F. Kurose and K.W. Ross, All Rights Reserved

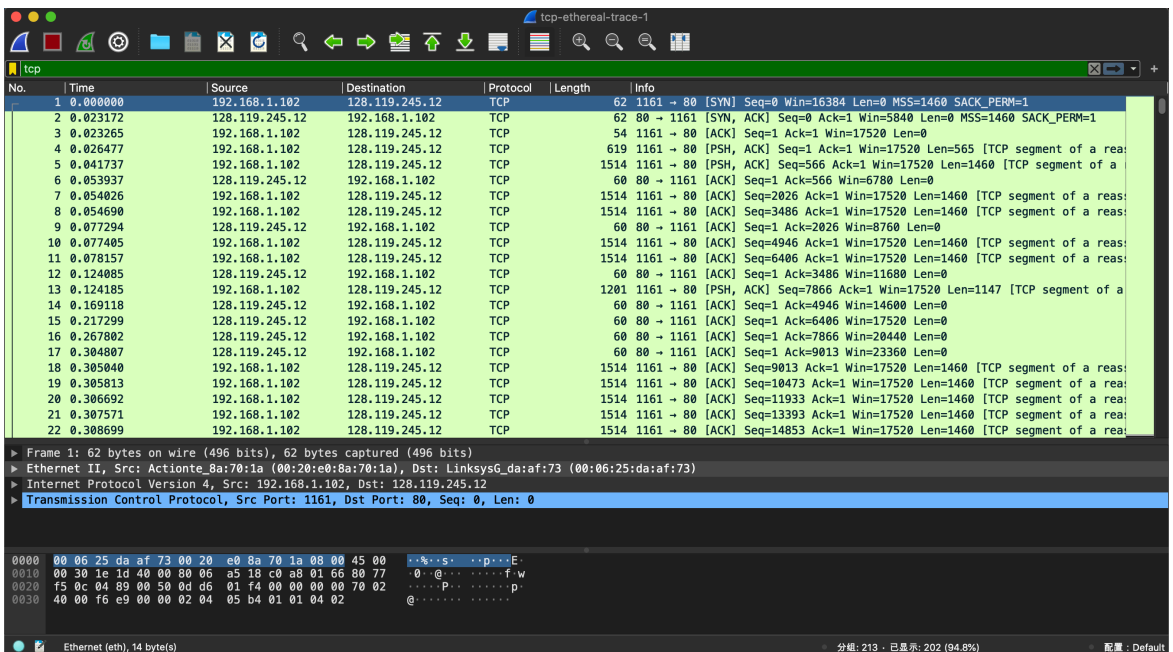
Congratulations!

You've now transferred a copy of alice.txt from your computer to gaia.cs.umass.edu. You should now stop Wireshark packet capture. It's time to start analyzing the captured Wireshark packets!

停止 wireshark 的捕获, 在 wireshark 界面可以看到:



2. A first look at the captured trace



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? To answer this question, it's

probably easiest to select an HTTP message and explore the details of the TCP packet used to carry this HTTP message, using the “details of the selected packet header window”

答：如图，客户端电脑的 IP 地址为 192.168.1.102；TCP 端口为 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?

答：gaia.cs.umass.edu 的 IP 地址为 128.119.245.12；端口号为 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?

60	2.201489	114.214.227.26	128.119.245.12	TCP	78	59441 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=716334519 T
61	2.201647	114.214.227.26	128.119.245.12	TCP	78	59442 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=716334519 TS
65	2.446358	114.214.227.26	128.119.245.12	TCP	78	59443 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=716334763 TS

答：我自己电脑的 IP 地址为：114.214.227.26，端口号：59441

3. TCP Basics

Answer the following questions for the TCP segments:

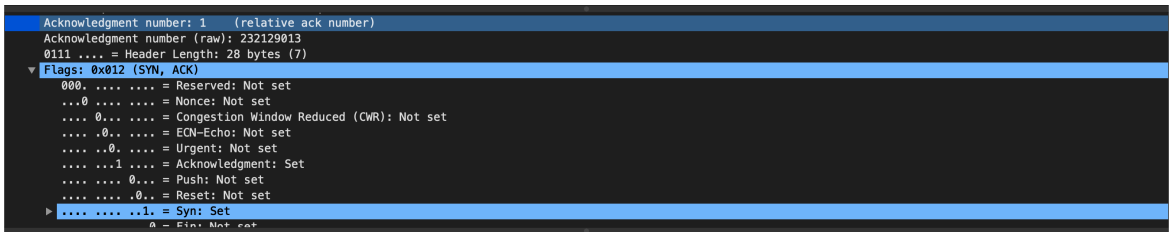
4. What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and gaia.cs.umass.edu? What is it in the segment that identifies the segment as a SYN segment?

```
Acknowledgment number: 0
Acknowledgment number (raw): 0
0111 .... = Header Length: 28 bytes (7)
▼ Flags: 0x002 (SYN)
000. .... = Reserved: Not set
...0 .... = Nonce: Not set
.... 0... = Congestion Window Reduced (CWR): Not set
.... 0... = ECN-Echo: Not set
.... 0... = Urgent: Not set
.... 0... = Acknowledgment: Not set
.... 0... = Push: Not set
.... 0... = Reset: Not set
▶ .... 1... = Syn: Set
.... 0... = Fin: Not set
```

答：SYN 的 Seq 序号为 0，确认报文为 SYN 报文的标志是报文的 TCP HEADER 中的 flag 域中的 SYN 被置为 1。

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?

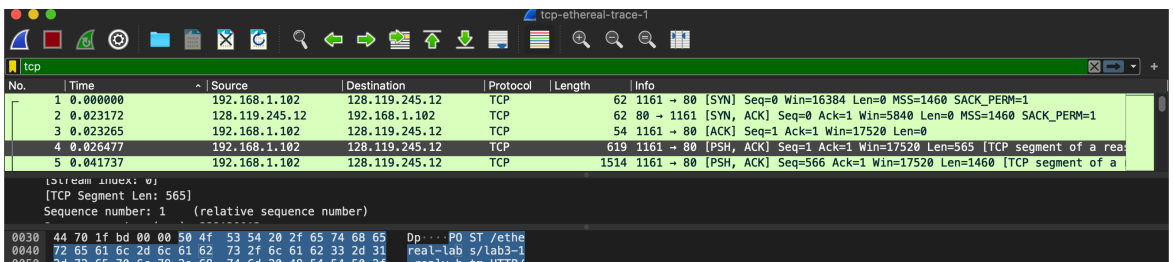


答: sequence number=0; Acknowledgement number=1.

gaia.cs.umass.edu 将 Acknowledgement number 值设置为所期望的下一个来自客户端的报文的 Sequence Number, 也就是之前收到的客户端序号+1。

Flag field 中的 ACK 位和 SYN 位被置为 1。

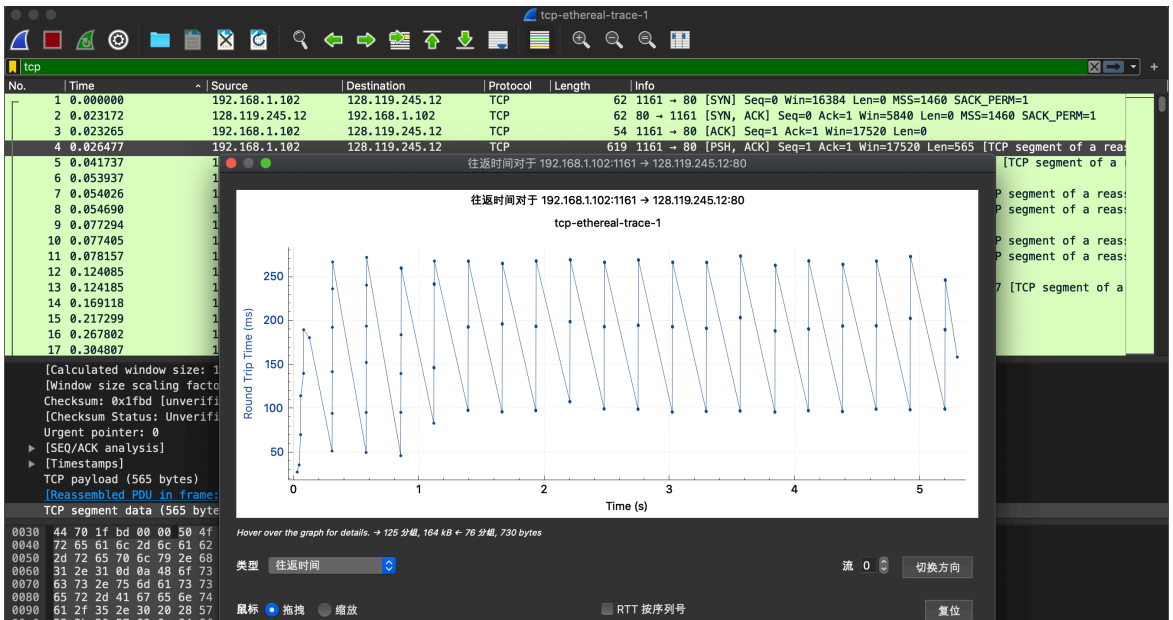
- What is the sequence number of the TCP segment containing the HTTP POST command? Note that in order to find the POST command, you'll need to dig into the packet content field at the bottom of the Wireshark window, looking for a segment with a "POST" within its DATA field.



答: sequence number=1

- 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)? 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? What is the EstimatedRTT value (see page 249 in text) after the receipt of each ACK? Assume that the value of the EstimatedRTT is equal to the measured RTT for the first segment, and then is computed using the EstimatedRTT equation on page 249 for all subsequent segments.

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 data segment]
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 data segment]
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 data segment]
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 data segment]
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 data segment]
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 data segment]



答: $\text{EstimatedRTT} = (1-a) * \text{EstimatedRTT} + a * \text{SampleRTT}$, 取 $a=0.125$, 计算可得

i	Seq	发送时间	收到 ACK 时间	RTT(ms)	EstimatedRTT(ms)
1	1	0.026477	0.053937	27.5	27.5
2	566	0.041737	0.077294	35.6	28.5
3	2026	0.054026	0.124085	70.1	33.7
4	3486	0.054690	0.169118	114.4	43.8
5	4946	0.077405	0.217299	139.9	55.8
6	6406	0.078157	0.267802	189.7	72.5

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

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 data segment]
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 data segment]
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 data segment]
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 data segment]
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 data segment]
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 data segment]

答: 如上图所示, 分别为 565, 1460, 1460, 1460, 1460, 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?

答：服务器最小的缓冲空间（Win）为 5840 字节。全程服务器端的 Win 在慢慢变大，最大变到 62780 字节，没有限制发送端。

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

答：没有重传的报文。检查发送端的 Seq 序号是否是单调递增的，有无重复的 Seq。

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.

50	0.994715	128.119.245.12	192.168.1.102	TCP	60	80 → 1161	[ACK] Seq=1 Ack=29777 Win=61320 Len=0
51	1.039820	128.119.245.12	192.168.1.102	TCP	60	80 → 1161	[ACK] Seq=1 Ack=31237 Win=62780 Len=0
52	1.117097	128.119.245.12	192.168.1.102	TCP	60	80 → 1161	[ACK] Seq=1 Ack=33589 Win=62780 Len=0
53	1.117333	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80	[ACK] Seq=33589 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled data stream]
54	1.118133	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80	[ACK] Seq=35849 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled data stream]
55	1.119029	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80	[ACK] Seq=36509 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled data stream]
56	1.119858	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80	[ACK] Seq=37969 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled data stream]
57	1.120902	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80	[ACK] Seq=39429 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled data stream]
58	1.121891	192.168.1.102	128.119.245.12	TCP	946	1161 → 80	[PSH, ACK] Seq=40889 Ack=1 Win=17520 Len=892 [TCP segment of a reassembled data stream]
59	1.208421	128.119.245.12	192.168.1.102	TCP	60	80 → 1161	[ACK] Seq=1 Ack=35049 Win=62780 Len=0
60	1.265026	128.119.245.12	192.168.1.102	TCP	60	80 → 1161	[ACK] Seq=1 Ack=37069 Win=62780 Len=0
61	1.362074	128.119.245.12	192.168.1.102	TCP	60	80 → 1161	[ACK] Seq=1 Ack=40889 Win=62780 Len=0
62	1.389886	128.119.245.12	192.168.1.102	TCP	60	80 → 1161	[ACK] Seq=1 Ack=41781 Win=62780 Len=0

答：典型的有 1460 和 892 字节。此外还有 565, 1147, 2352(892+1460), 2920(1460+1460) 等。一次 ACK 所 acknowledge 的字节数根据相邻 Ack 的差值算出。一次确认两个报文的有确认 2920 和 2352 字节的。如图中的第 52 个报文，就确认了 2352 字节；第 60 个报文，就确认了 2920 字节。

12. What is the throughput (bytes transferred per unit time) for the TCP connection?

Explain how you calculated this value.

3	0.026477	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 data stream]
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 data stream]
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 data stream]

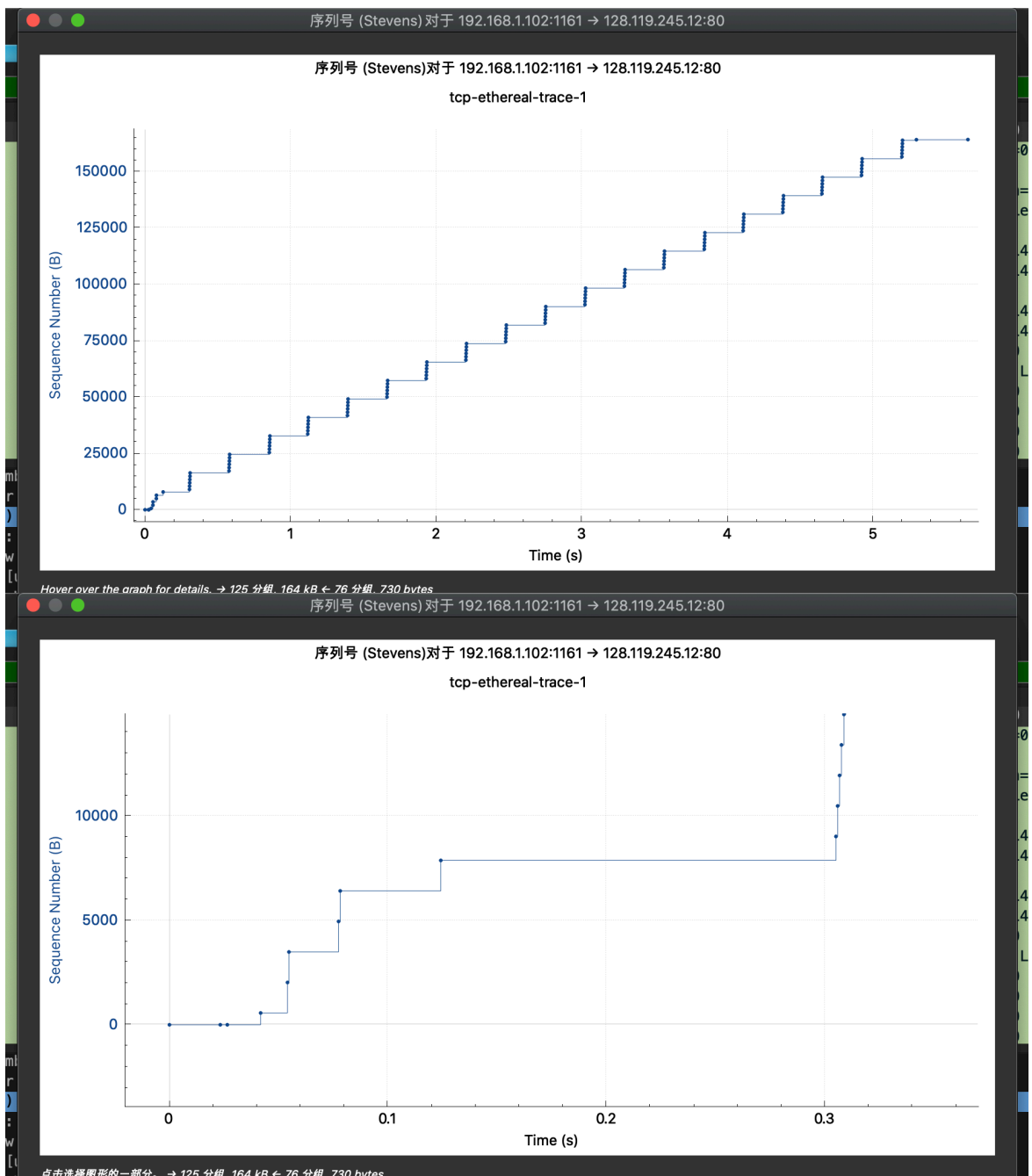
答：考虑第一次发送 post 到发送端收到最后一条 Ack 这一过程的吞吐量（注意这只是平均吞吐量，不一定是最大的瞬时吞吐量）。

总的时长 $t = 5.455830 - 0.026477 = 5.43$ (s)。

所传递的数据量 $w = 164091 - 1 = 164090$ (Bytes)。

吞吐量为 $\text{Throughput} = w/t = 164090/5.43 = 30.22\text{KBps}$

4. TCP congestion control in action



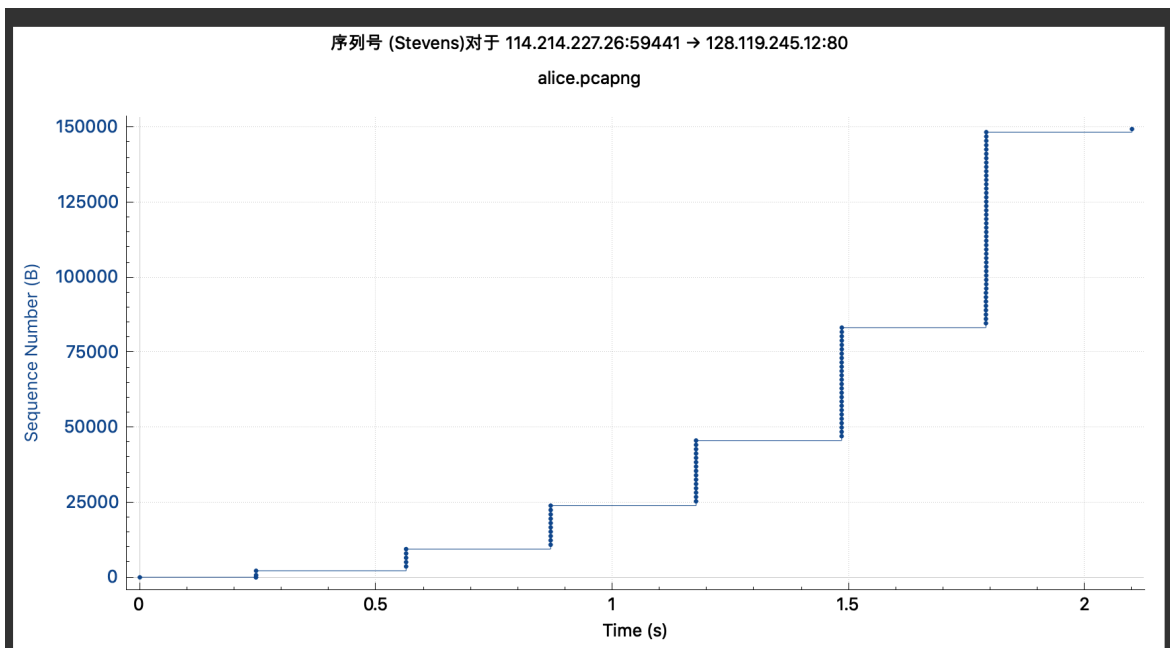
13. Use the *Time-Sequence-Graph(Stevens)* plotting tool to view the sequence number versus time plot of segments being sent from the client to the gaia.cs.umass.edu server. Can you identify where TCP's slowstart phase begins and ends, and where

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.

答：慢启动应该是在 post 发送第一次数据时开始，但是从这张图里看不出何时结束慢启动和开始拥塞避免。因为这个 trace 中没有出现拥塞导致的超时或者丢包的情况，而且接收方和发送方的窗口都远大于发送的数据量。

在确认过程中还可以看到，发送方连续发送 $1460 \times 5 + 892$ 字节 = 8192 字节的包就会暂停发送，直到这 8192 字节被全部确认。这一数据传输过程，应该是被应用程序所严格的控制着，而不仅仅依赖于 TCP 自身的拥塞控制机制。

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



答：同样，慢启动在 post 发送开始时开始，从上图中可以看出 cwnd 呈指数增长；同时，在我的 trace 中可以看出也没有拥塞情况，慢启动没有结束文件就已经全部传输完毕了，所以拥塞避免也没有开始，所以看不出何时结束。