计算机网络第五次实验实验报告

舒文炫

2021 年 10 月 25 日

目录

Chapter 1

实验内容

本次实验主要详细的了解 TCP 协议的内容，通过传输一个 150KB 的文件，分析 TCP 报文的收发，学习 TCP 的一些行为：

- 通过序号和确认号保证可靠信息传输
- 拥塞控制算法如何实现
- 流量控制机制
- TCP 连接的建立，以及该连接的一些表现 (吞吐量，RTT)

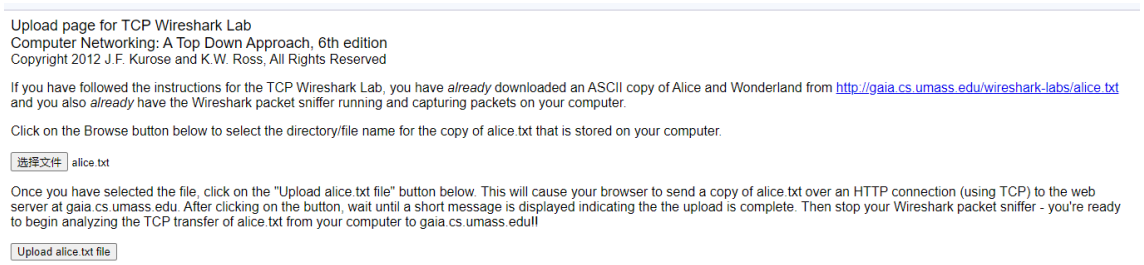
Chapter 2

实验过程

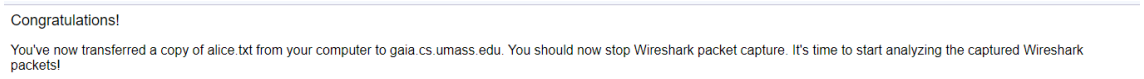
2.1 Capturing a bulk TCP transfer from your computer to a remote server

打开浏览器，输入网址 http://gaia.cs.umass.edu/wiresharklabs/alice.txt 会打开一个包含爱丽丝梦游仙境的文本，复制下来，保存到电脑，命名 alice.txt.

再打开网址 http://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html. 选择文件 alice.txt, 现在情况如下



打开 wireshark，开始捕获，并点击网页的 upload file 按钮，上传完毕后会弹出这样的网页

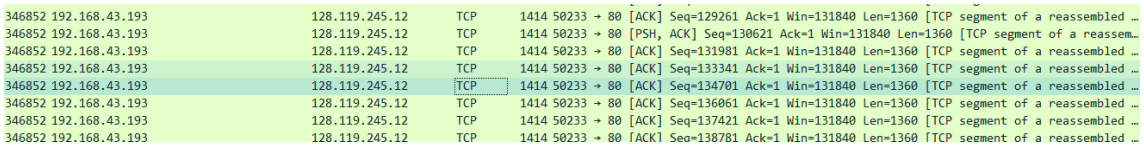


这时可以停止捕获。

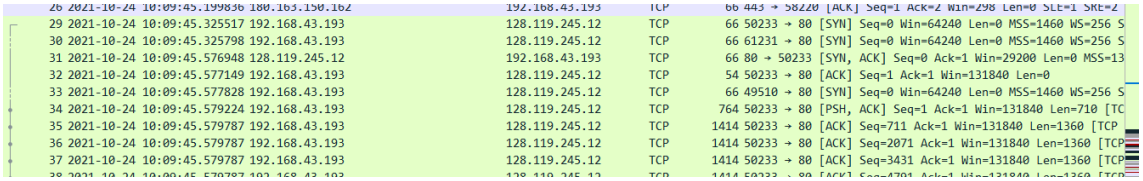
2.2 A first look at the captured trace

2.2.1 结果截图

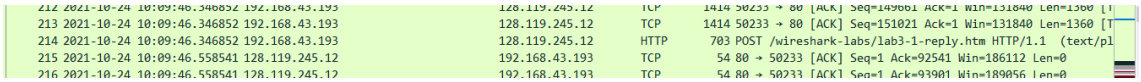
在之前捕获的包列表里面可以看到这一系列的 TCP 报文段，且后面有 [TCP segmentof a reassembled PDU] 表示这个报文段包含上层协议也就是 HTTP 协议报文数据。



这张截图包含几个 SYN 报文，这是建立 TCP 连接三次握手，同时也可以看到几个 ACK 的消息



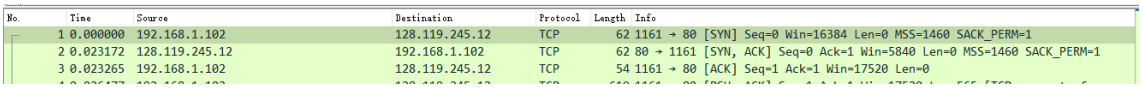
这里是一个 HTTP POST 报文，这里包含了我传输的最后一个分段，只在这是 wireshark 才认为这是 HTTP 协议



2.2.2 小节思考题

注：除了第 3,14 题是要用自己捕获的包回答，其他的直接使用了作者提供的 (实验指导书这么要求的)

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”(refer to Figure 2 in the “Getting Started with Wireshark” Lab if you’re uncertain about the Wireshark windows.



见图，序号为 1 的包，传文件的源 IP 地址为 192.168.1.102，TCP 端口号为 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?

见图，序号为 1 的包，目的 IP 地址即为所求，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?

这里用到我自己捕获的包，图在前面，可以看到包序号 29，源 IP 地址为 192.168.43.193，端口号为 50233

2.3 TCP Basics

2.3.1 需要的截图

这里将需要的截图贴在这里，后面思考题就直接写了
将第一个 SYN 包的具体信息打印到文件中截图如下:

```
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
Frame 1: 62 bytes on wire (496 bits), 62 bytes captured (496 bits)
Ethernet II, Src: Actionte_8a:70:1a (00:20:e0:8a:70:1a), Dst: LinksysG_da:af:73 (00:06:25:da:af:73)
Internet Protocol Version 4, Src: 192.168.1.102, Dst: 128.119.245.12
Transmission Control Protocol, Src Port: 1161, Dst Port: 80, Seq: 0, Len: 0
    Source Port: 1161
    Destination Port: 80
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence Number: 0    (relative sequence number)
    Sequence Number (raw): 232129012
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 0
    Acknowledgment number (raw): 0
    0111 .... = Header Length: 28 bytes (7)
    Flags: 0x002 (SYN)
    Window: 16384
    [Calculated window size: 16384]
    Checksum: 0xf6e9 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
    Options: (8 bytes), Maximum segment size, No-Operation (NOP), No-Operation (NOP), SACK permitted
    [Timestamps]
```

其对应的 ACK 包具体信息如下：

```
No.     Time           Source                Destination           Protocol Length Info
      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
Frame 2: 62 bytes on wire (496 bits), 62 bytes captured (496 bits)
Ethernet II, Src: LinksysG_da:af:73 (00:06:25:da:af:73), Dst: Actionte_8a:70:1a (00:20:e0:8a:70:1a)
Internet Protocol Version 4, Src: 128.119.245.12, Dst: 192.168.1.102
Transmission Control Protocol, Src Port: 80, Dst Port: 1161, Seq: 0, Ack: 1, Len: 0
    Source Port: 80
    Destination Port: 1161
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence Number: 0    (relative sequence number)
    Sequence Number (raw): 883061785
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 232129013
    0111 .... = Header Length: 28 bytes (7)
    Flags: 0x012 (SYN, ACK)
    Window: 5840
    [Calculated window size: 5840]
    Checksum: 0x774d [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
    Options: (8 bytes), Maximum segment size, No-Operation (NOP), No-Operation (NOP), SACK permitted
    [SEQ/ACK analysis]
    [Timestamps]
```

在第 4 个包的 data 域里面看到了如下的 POST 命令，从而可以知道这是包含了 POST 命令的 TCP 报名段

```
00 06 25 da af 73 00 20  e0 8a 70 1a 08 00 45 00   ··%··s· ··p···E·
02 5d 1e 21 40 00 80 06  a2 e7 c0 a8 01 66 80 77   ·]·!@··· ·····f·w
f5 0c 04 89 00 50 0d d6  01 f5 34 a2 74 1a 50 18   ·····P·· ··4·t·P·
44 70 1f bd 00 00 50 4f  53 54 20 2f 65 74 68 65   Dp····PO ST /ethe
72 65 61 6c 2d 6c 61 62  73 2f 6c 61 62 33 2d 31   real-lab s/lab3-1
2d 72 65 70 6c 79 2e 68  74 6d 20 48 54 54 50 2f   -reply.h tm HTTP/
31 2e 31 0d 0a 48 6f 73  74 3a 20 67 61 69 61 2e   1.1··Hos t: gaia.
63 73 2e 75 6d 61 73 73  2e 65 64 75 0d 0a 55 73   cs.umass .edu··Us
65 72 2d 41 67 65 6e 74  3a 20 4d 6f 7a 69 6c 6c   er-Agent : Mozill
```

该报文段的具体信息如下：

```
No.     Time           Source                Destination           Protocol Length Info
      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]
Frame 4: 619 bytes on wire (4952 bits), 619 bytes captured (4952 bits)
Ethernet II, Src: Actionte_8a:70:1a (00:20:e0:8a:70:1a), Dst: LinksysG_da:af:73 (00:06:25:da:af:73)
Internet Protocol Version 4, Src: 192.168.1.102, Dst: 128.119.245.12
Transmission Control Protocol, Src Port: 1161, Dst Port: 80, Seq: 1, Ack: 1, Len: 565
    Source Port: 1161
    Destination Port: 80
    [Stream index: 0]
    [TCP Segment Len: 565]
    Sequence Number: 1    (relative sequence number)
    Sequence Number (raw): 232129013
    [Next Sequence Number: 566    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 883061786
    0101 .... = Header Length: 20 bytes (5)
    Flags: 0x018 (PSH, ACK)
    Window: 17520
    [Calculated window size: 17520]
    [Window size scaling factor: -2 (no window scaling used)]
    Checksum: 0x1fbd [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
    [SEQ/ACK analysis]
    [Timestamps]
    TCP payload (565 bytes)
    [Reassembled PDU in frame: 199]
    TCP segment data (565 bytes)
```

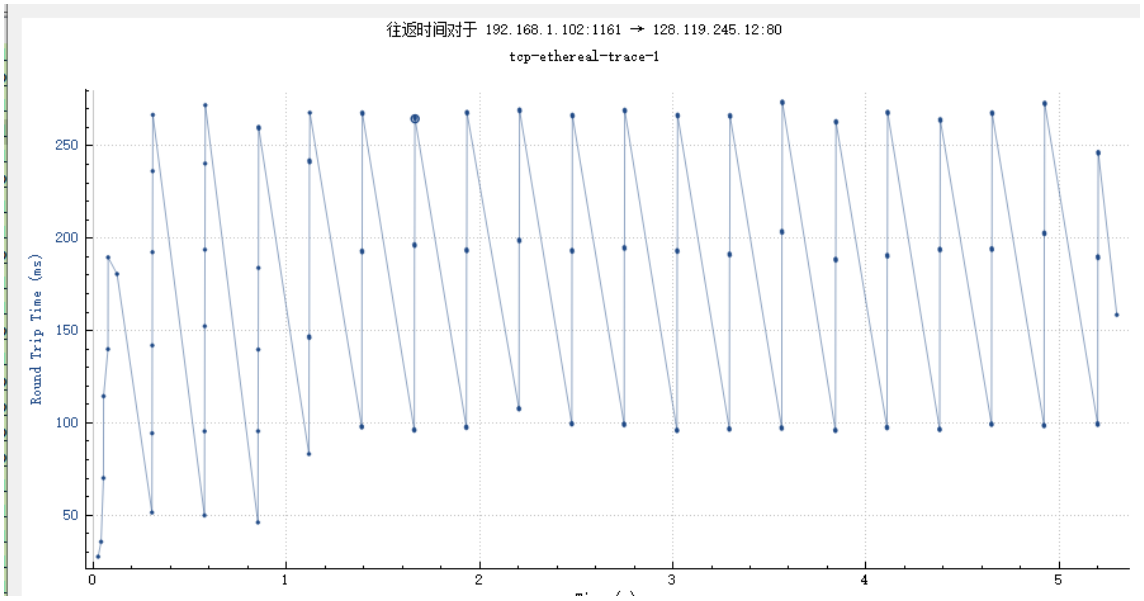
从包含 POST 命令的报文开始往后的 6 个报文：

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 reas
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 r
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 reass
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 reass
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 reass
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 reass

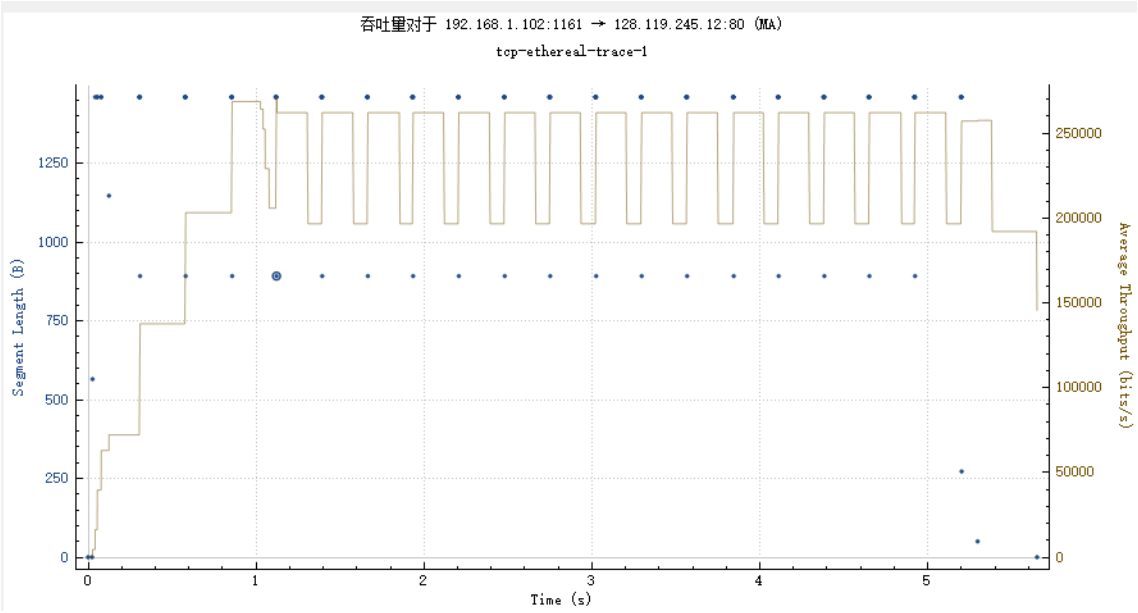
对应的 ACK 报文：

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 reass
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 reass
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 reass
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 reass
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
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

他们的周转时间 RTT 的截图



吞吐量的截图



2.3.2 小节思考题

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?

该小节第一个截图，可以看到有一行 Sequence Number: 0 (relative sequence number)，说明序号是 0，这里是相对序号，方便人看的，后面一行是原始的序号。还有一行 Flags: 0x002 (SYN)。Flags 标志位就是来表示传输过程中连接状态的，从低位到高位依次是 FIN,SYN,RST,PSH,ACK,URG,ECE,CWR,NS。0x002 表示其为 SYN 报文段。

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?

该小节第二个截图，可以看到一行 Sequence Number: 0 (relative sequence number)，说明序号是 0，后面有一行 Acknowledgment Number: 1 (relative ack number) 说明 ACK 域的值为 1，这个值是当前接收端所期待的序号，也就是当前接收到的 TCP 报文段序号 +1. 后面还有一行 Flags: 0x012 (SYN, ACK)，通过这个确认是 SYNACK 报文段

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.

该小节第四个截图，可以看到序号为 1。

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 theTCP 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 Section 3.5.3, page 242 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 242 for all subsequent segments.

综合第 5,6 个截图，为了使结果清晰，这里我做一个表其中 EstimatedRTT 使用公式

$$EstimatedRTT = 0.875EstimatedRTT + 0.125SampleRTT$$

, 结果保留六位有效数字

编号	报文段序号	发送时间	ACK 时间	RTT	EstimatedRTT
1	1	0.026477	0.053937	0.02746	0.027460
2	566	0.041737	0.077294	0.035557	0.028472
3	2026	0.054026	0.124085	0.070059	0.033670
4	3486	0.054690	0.169118	0.114428	0.043765
5	4946	0.077405	0.217299	0.139894	0.055781
6	6406	0.078157	0.267802	0.189645	0.072514

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

感觉这里面前六个也是以含 HTTP POST 的 TCP 报文开始的，所以为长度分别为 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?

观察所有的包，接收方最小的缓冲空间 (或者叫接收窗口) 大小是 5840 字节，如果接收窗口不够了，会限制发送方发送。

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

没有，可以观察这些发送出去的包的序号，这些序号严格单调递增，没有重复的，从而没有重发

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).

典型的，每个 ACK 会确认 1460 字节的数据。这个 every other 是每隔一个还是任意其他的？如果根据书上内容，比较接近的含义是累积确认，这种情况我们可以注意收到的 ACK 号，基本上每次增加不会超过 1460，这表示是相邻的两个报文段，如果增加的超过了这个数，就有可能前面 ACK 的发生了丢包，这里就能看出累积确认了。

```
1514 1161 → 80 [ACK] Seq=33589 Ack=1 Win=17520 Len=1460 [TCP segment of a reas
1514 1161 → 80 [ACK] Seq=35049 Ack=1 Win=17520 Len=1460 [TCP segment of a reas
1514 1161 → 80 [ACK] Seq=36509 Ack=1 Win=17520 Len=1460 [TCP segment of a reas
1514 1161 → 80 [ACK] Seq=37969 Ack=1 Win=17520 Len=1460 [TCP segment of a reas
1514 1161 → 80 [ACK] Seq=39429 Ack=1 Win=17520 Len=1460 [TCP segment of a reas
 946 1161 → 80 [PSH, ACK] Seq=40889 Ack=1 Win=17520 Len=892 [TCP segment of a
  60 80 → 1161 [ACK] Seq=1 Ack=35049 Win=62780 Len=0
  60 80 → 1161 [ACK] Seq=1 Ack=37969 Win=62780 Len=0
```

注意我选出来的这几项，发出去的包序号依次是 35049，36509，37969，但是后面的 ACK 号只有 35049，37969 中间的那个丢掉了，这里就是累积确认发生了

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

吞吐量的截图已经在前面贴出来了，表中的吞吐量大致在 260000bps 和 200000bps 之间变化，粗糙的估计，可以从抓到的包看到，一次大概会连着发 6 个报文段，5 个报文段大致 1460bytes，一个 892bytes，然后一个来回的时间大致 0.264s，从而可以估计吞吐量为 $\frac{(1460\times5+892)\times8}{0.264} = 248242.4242bps$ 这是很粗糙的估计了不过大致能用。

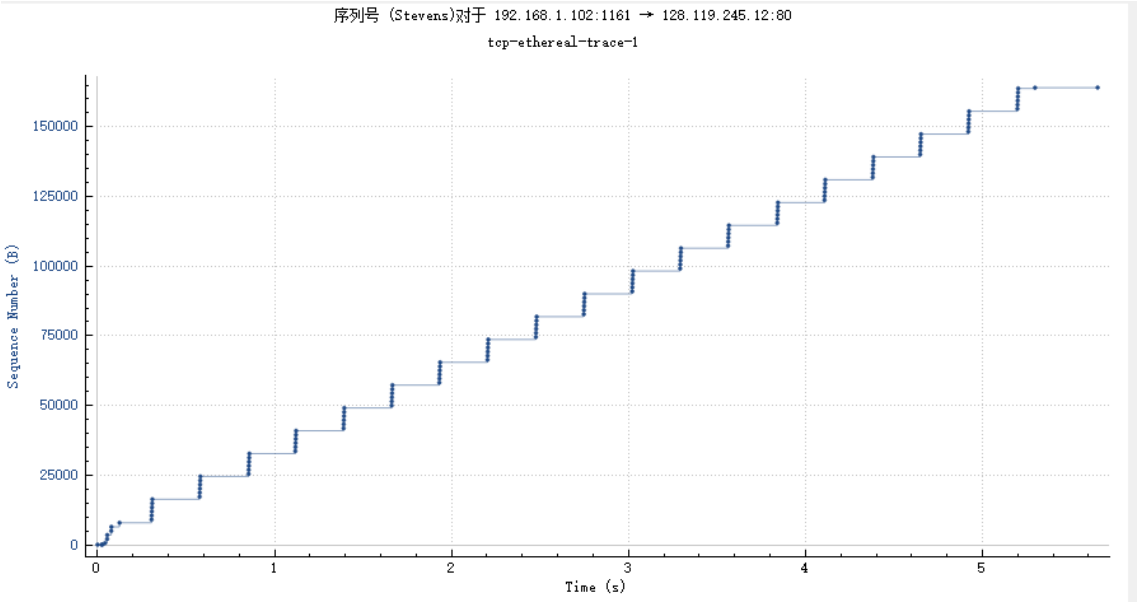
202	5.455830	128.119.245.12	192.168.1.102	TCP	60 80 → 1161 [ACK] Seq=1 Ack=164091 Win=62780 Len=0
203	5.461175	128.119.245.12	192.168.1.102	HTTP	784 HTTP/1.1 200 OK (text/html)

或者还可以取全局的一个平均，HTTP POST 到 HTTP OK 从上图可以看到，这一共花了 5.461175s，总共传输数据 164091B，这是从上面的 ACK 号看出从而两者相除，得到平均吞吐量 240346.644651bps，也在吞吐量图的范围内，可以接受

2.4 TCP congestion control in action

2.4.1 过程截图

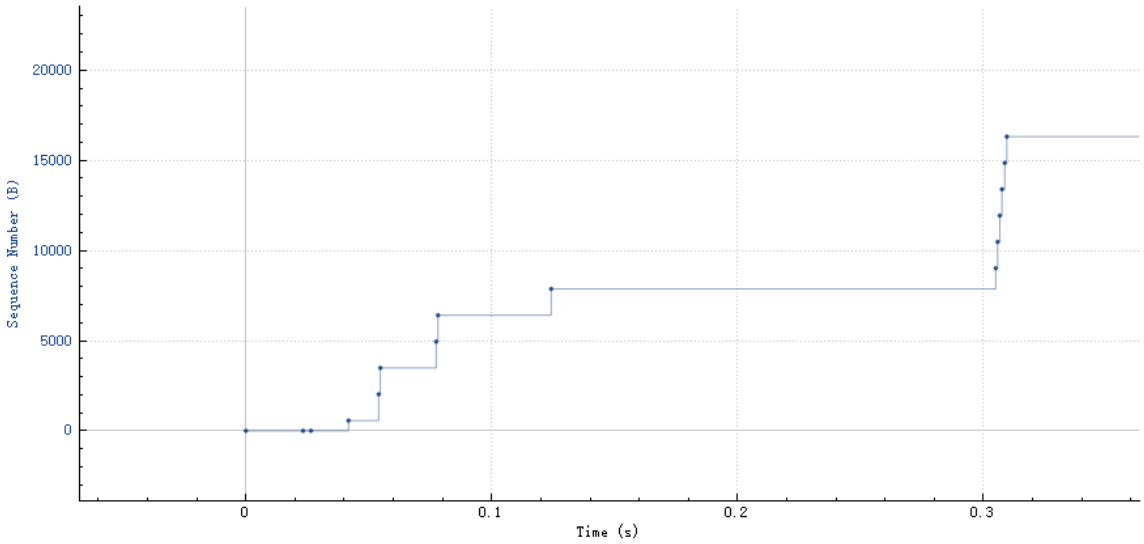
选择一个 TCP 报文段，调出时序图



2.4.2 小节思考题

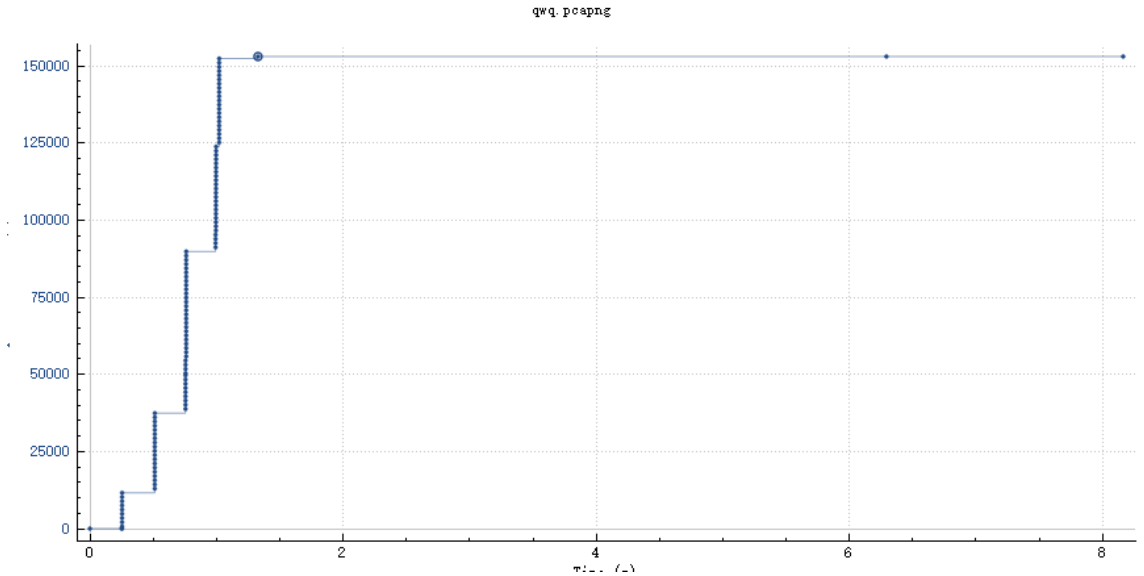
13. Use the Time-Sequence-Graph(Stevens) plotting tool to view the sequence number versus time plot of segments being sent from the client to the gaia.cs.umass.edu server. Can you identify where TCP's slowstart phase begins and ends, and where congestion avoidance takes over? Comment on ways in which the measured data differs from the idealized behavior of TCP that we've studied in the text.

从图中可以看到，一开始的一段同一时间发出的报文段还比较少，并且在逐渐增大，这一段就是慢启动阶段，大致从 0s 到 0.31s 我将这一块放大



可以很清晰的看到最开始一次只能发一个报文段，后面变成两个，最终在 6 个。后面每次都最多发 6 个，所以后面就进入了拥塞避免阶段，但是这里拥塞窗口一直保持，并没有按照书上说的线性增加，这里应该是实现的时候有不一样的机制，和书上理想情况不一样。

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



这是我的时序图，可以看到这里每次同时可以发出的报文段的数量一直在增加，慢启动阶段从 0s 到 1s，而且慢启动结束 (可能还没有结束) 文件就传完了，没有拥塞避免阶段，只能说现在的链路确实吞吐量很大可以达到 10^6 量级了，对于不怎么大的文件很快就能传完。

Chapter 3

实验总结

本次实验我更详细的了解了 TCP 报文的形式，以及在实际情况下 TCP 的拥塞控制机制如何的表现，这和书本上所说的理想情况不大一样，在看累积确认那一块的时候，着实锻炼了我的眼力和耐心 orz