中国科学技术大学计算机学院

《计算机网络》实验报告

实验题目：lab3_WireShark Labs(TCP)

学生姓名：胡毅翔

学生学号：PB18000290

专业：计算机科学与技术

指导老师：张信明

完成日期：2020 年 11 月 15 日

计算机实验教学中心制

2019 年 09 月

实验目的

1.进一步掌握 Wireshark 网络分析工具。

2.捕获观察并分析 TCP 报文段结构。

3.回答本次实验指导书中的问题。

实验原理

本次实验使用 WireShark 工具。其中，用来观察执行协议实体之间交换的报文的基本工具称为分组嗅探器(packet sniffer)。分组嗅探器被动地拷贝(嗅探)由计算机发送和接收的报文；它也能显示出这些被捕获报文的各个协议字段的内容。分组嗅探器从不发送报文，同时接收到的报文也不会显式地发送到分组嗅探器。它接受的是发送/接收的报文的复制。

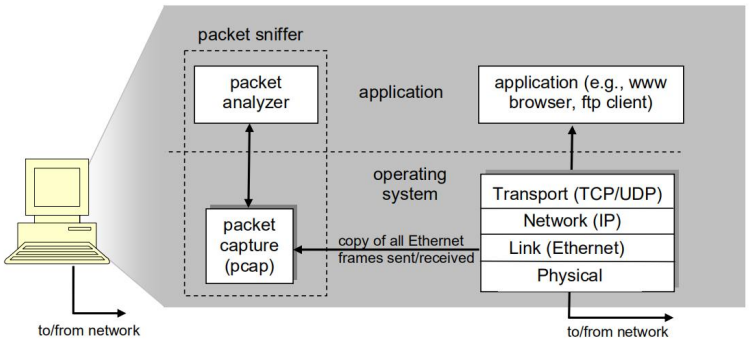

图 1

分组嗅探器的结构如图 1 所示。在图 1 的右边是运行在计算机上的协议和应用。分组嗅探器(图中画虚线框部分)是计算机中的附加软件(区别于上述协议和应用)，它包含两个部分。分组捕获库获取每一个链路层接收/发送的帧。第二部分是分组分析器，其中显示了协议所有字段的内容。为了实现这一目的，分组分析器必须理解所有协议所交换的信息的结构。比如，我们对图 1 中 HTTP 协议的各个字段信息感兴趣。分组分析器理解以太网帧的格式，所以可以从以太网帧中区分出 IP 数据报。同时，它还理解 IP 数据报格式，所以它能从 IP 数据报分离出 TCP 报文段。最后，它还理解 TCP 报文段格式，从中分离出 HTTP 报文。又因它理解 HTTP 协议，所以能在实现 WireShark 中显示 HTTP 协议各字段信息的功能。

实验环境

1.PC 一台

2.Windows 系统

3.WireShark 网络分析工具(版本 3.2.7)

4.Edge 浏览器(版本 86.0.622.56)

实验过程

WireShark Lab: TCP

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

实验步骤

1.启动浏览器。前往 http://gaia.cs.umass.edu/wireshark-labs/alice.txt，得到《爱丽丝梦游仙境》的 ASCII 码副本。保存到计算机上。

2.前往 http://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html。

3.进入到如下界面。

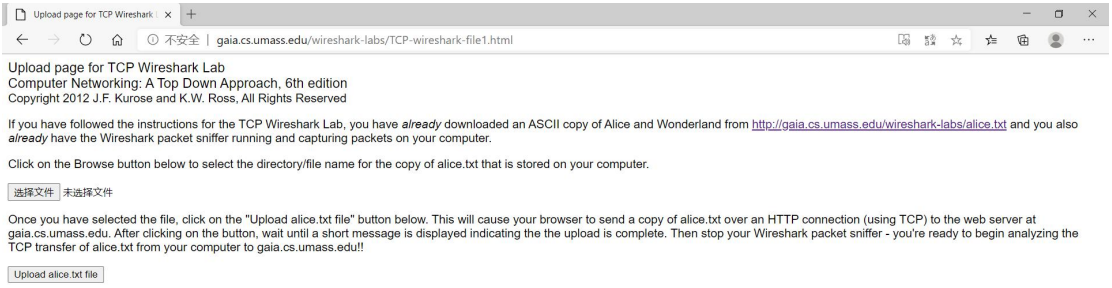

图 2

4.点击“选择文件”，选择刚刚保存的 alice.txt。

5.启动 WireShark 分组嗅探器。

6.上传 alice.txt。

7.得到上传成功的信息后，停止捕获分组。

A first look at the captured trace

实验步骤

1.在显示过滤器中输入”tcp”，观察捕获到的数据。

2.根据实验指导书要求，下载 wireshark-traces.zip，观察 tcp-ethereal-trace-1，并回答对应问题。

问题

Q1: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).

A1:客户端的 IP 地址是 192.168.1.102，TCP 端口号是 1161，如图 3。

Q2:What is the IP address of gaia.cs.umass.edu? On what port number is it sending and receiving TCP segments for this connection?

A2:gaia.cs.umass.edu 的 IP 地址是 128.119.245.12。本次链接使用的端口号是 80，如图 3。

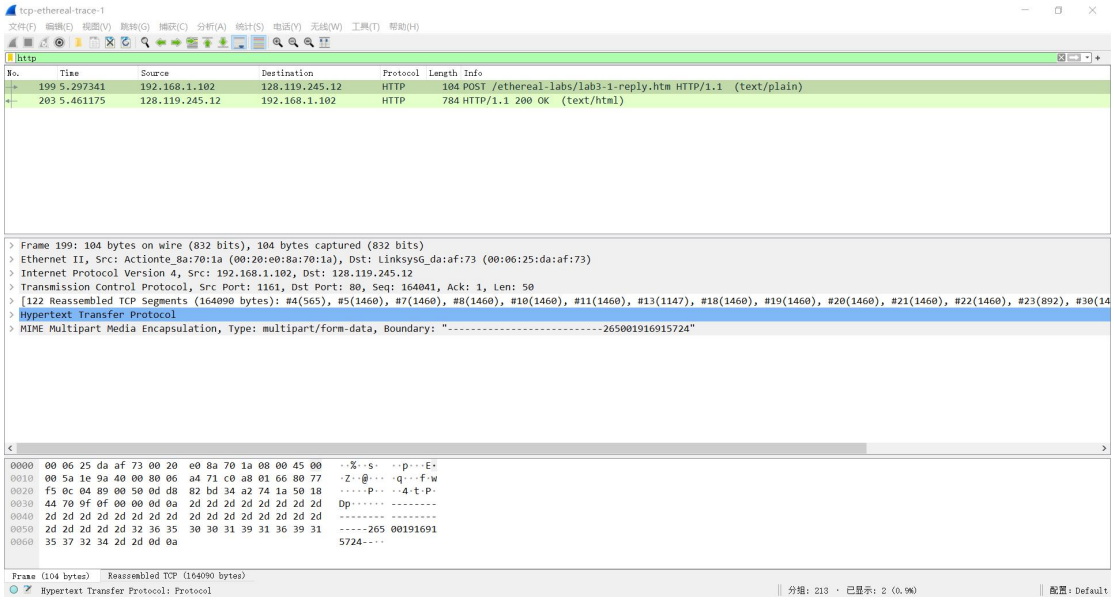

图 3

Q3:What is the IP address and TCP port number used by your client computer (source) to transfer the file to gaia.cs.umass.edu?

A3:我的计算机的 IP 地址是 192.168.43.19，使用的 TCP 端口号是 49669，如图 4。

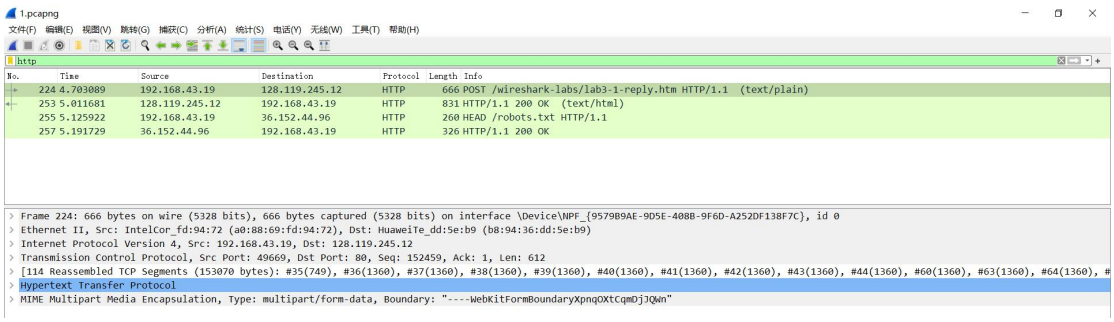

图 4

TCP Basics

实验步骤

1.在分析->启动的协议中，反选 HTTP，作进一步分析。

问题

Q4: 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?

A4:客户端的 SYN 报文段的序列号是 0。从图 5 中可见，在 Flags 中 SYN 被置为 1，表示该报文段是 SYN 报文段。

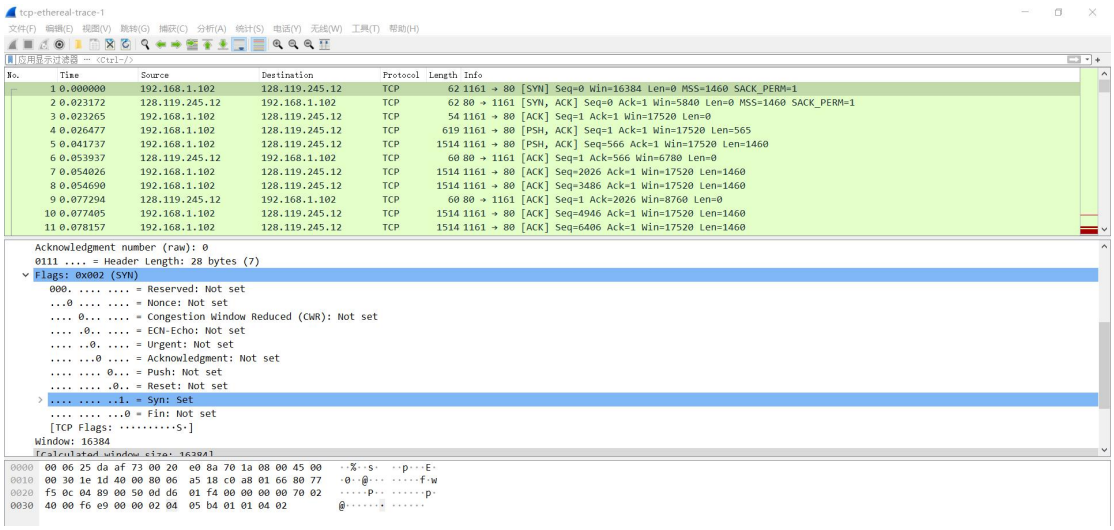

图 5

Q5: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?

A5:服务器的 SYNACK 报文段的序列号是 0。该报文段中 ACKnowledgment 域的值为 1，该值为 1 是因为：已收到序列号为 0 的报文段，期望收到序列号为 1 的报文段。Flags 中 SYN，ACK 置为 1，表明这是 SYNACK 报文段，如图 6。

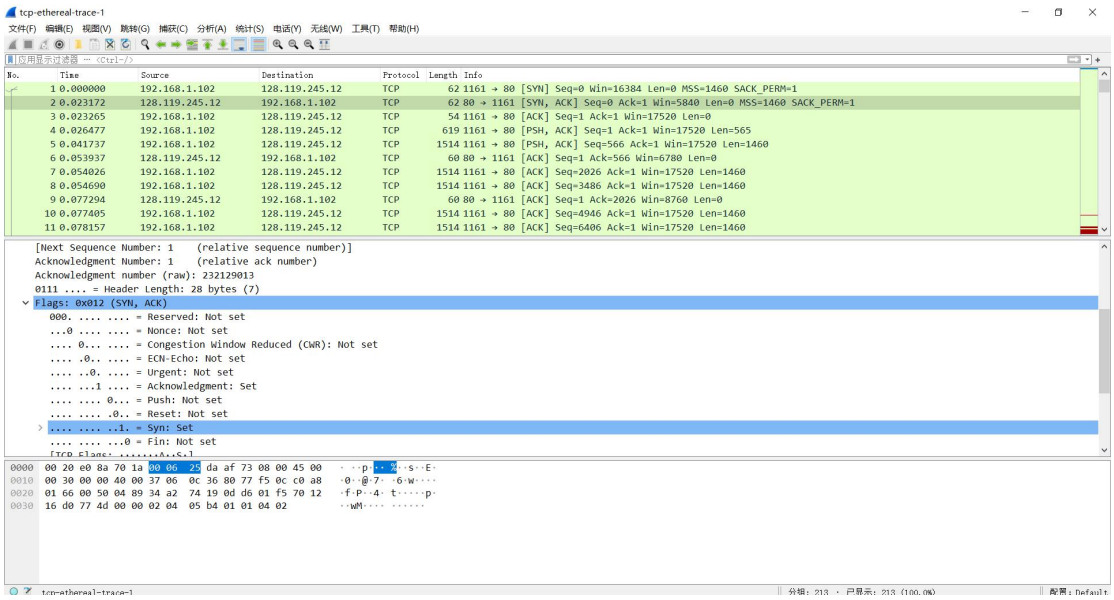

图 6

Q6: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.

A6:包含 HTTP POST 命令的 TCP 报文段的序列号是 1，如图 7。

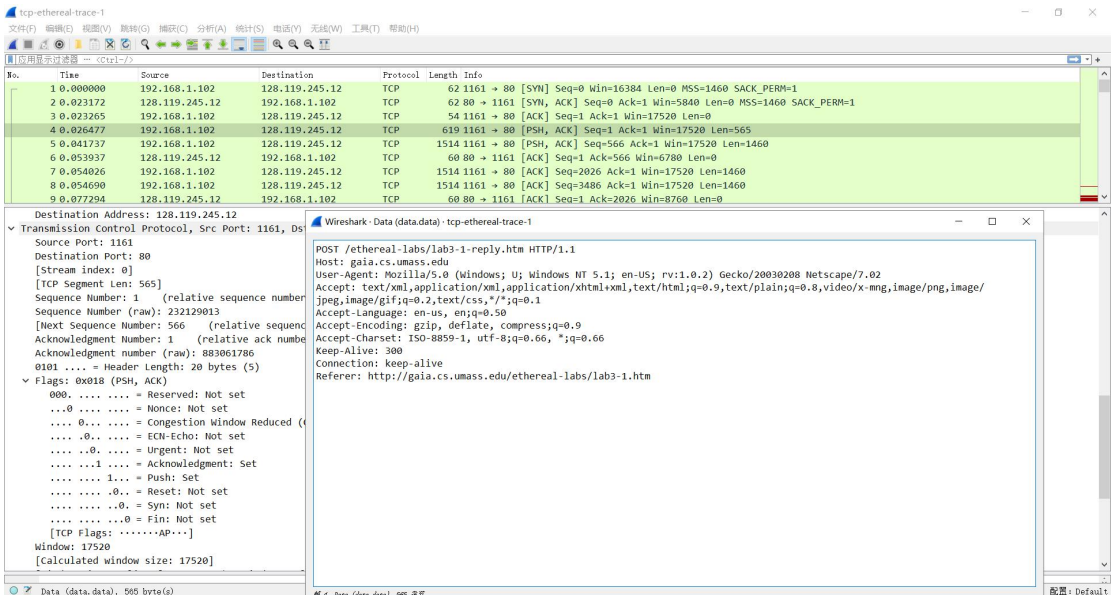

图 7

Q7: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.

Note: Wireshark has a nice feature that allows you to plot the RTT for each of the TCP segments sent. Select a TCP segment in the “listing of captured packets” window that is being sent from the client to the gaia.cs.umass.edu server. Then select: Statistics->TCP Stream Graph- >Round Trip Time Graph.

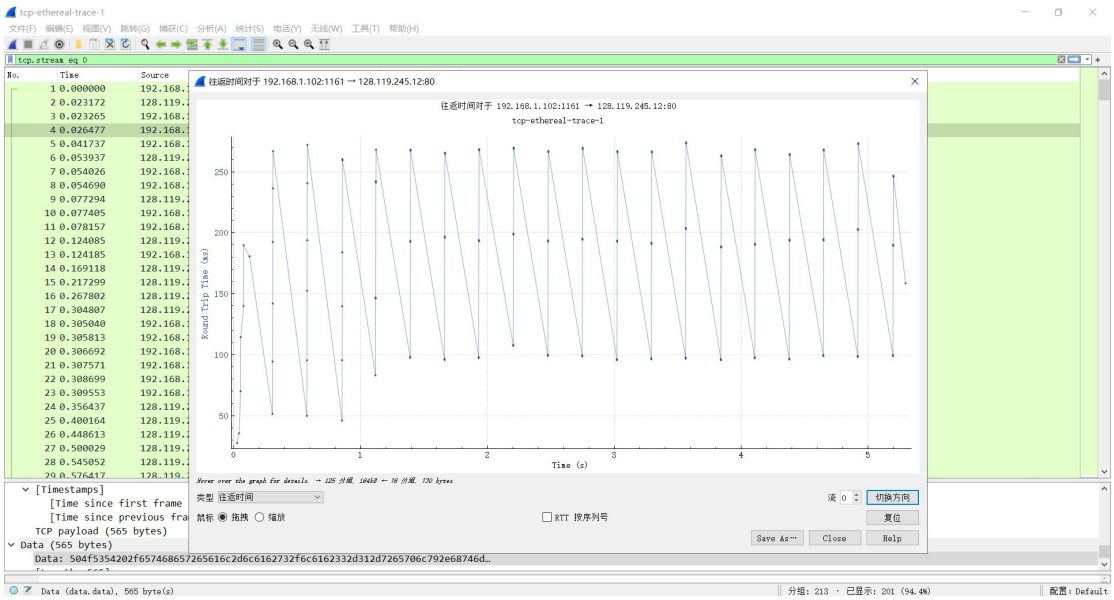

图 8

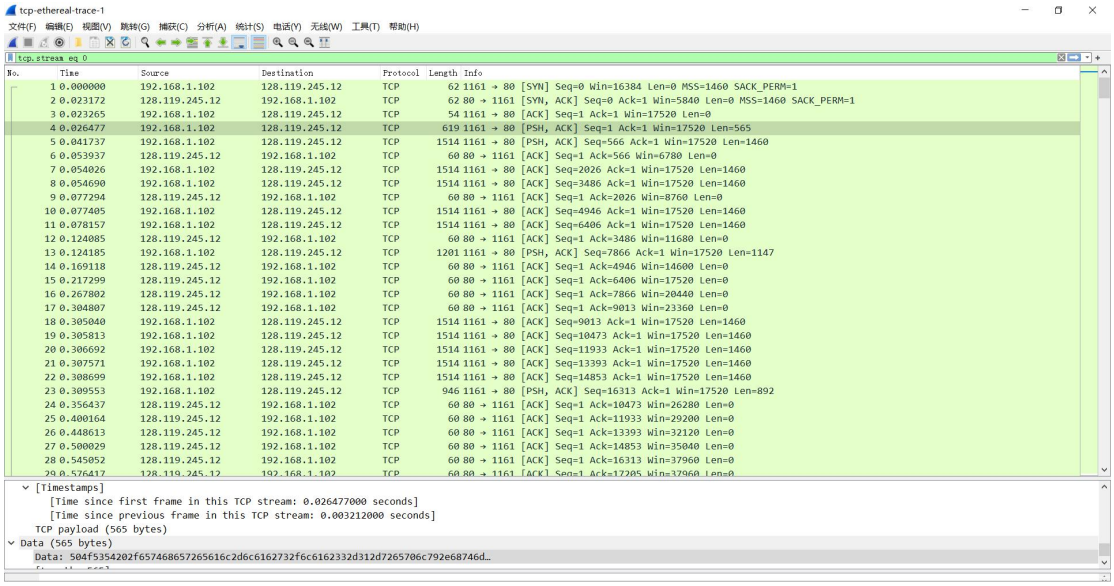

图 9

A7:根据图 8，图 9 的信息得到表 1。

序列号	发送时间(s)	收到 ACK 的时间(s)	RTT(s)	EstimatedRTT(s)	Length(byte)
1	0.026477	0.053937	0.02746	0.02746	565
566	0.041737	0.077294	0.035557	0.028472125	1460
2046	0.054026	0.124085	0.070059	0.033670484375	1460
3486	0.054690	0.169118	0.114428	0.0437651738281	1460
4946	0.077405	0.217299	0.139894	0.0557812770996	1460
6406	0.078157	0.267802	0.189645	0.0725142424622	1460

表 1

Q8: What is the length of each of the first six TCP segments?

A8:见表 1。

Q9: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?

A9:最小的窗口尺寸是 5840B，没有出现因接收窗口不足导致发送方被限制发送的情况，如图 10。

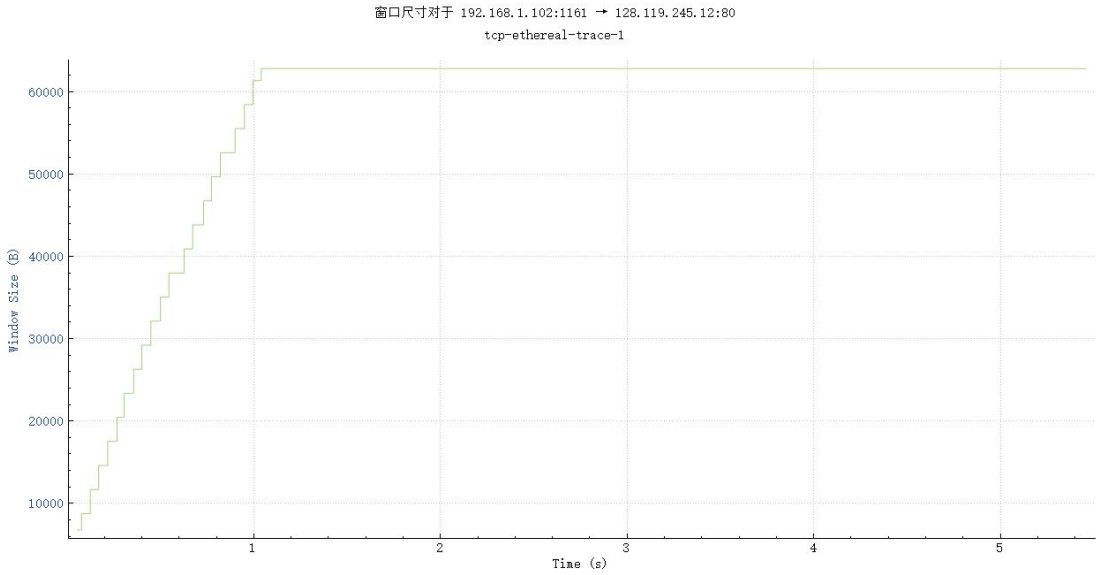

图 10

Q10: Are there any retransmitted segments in the trace file? What did you check for (in the trace) in order to answer this question?

A10:如图 11 所示，序列号一直在增大，没有出现重传。

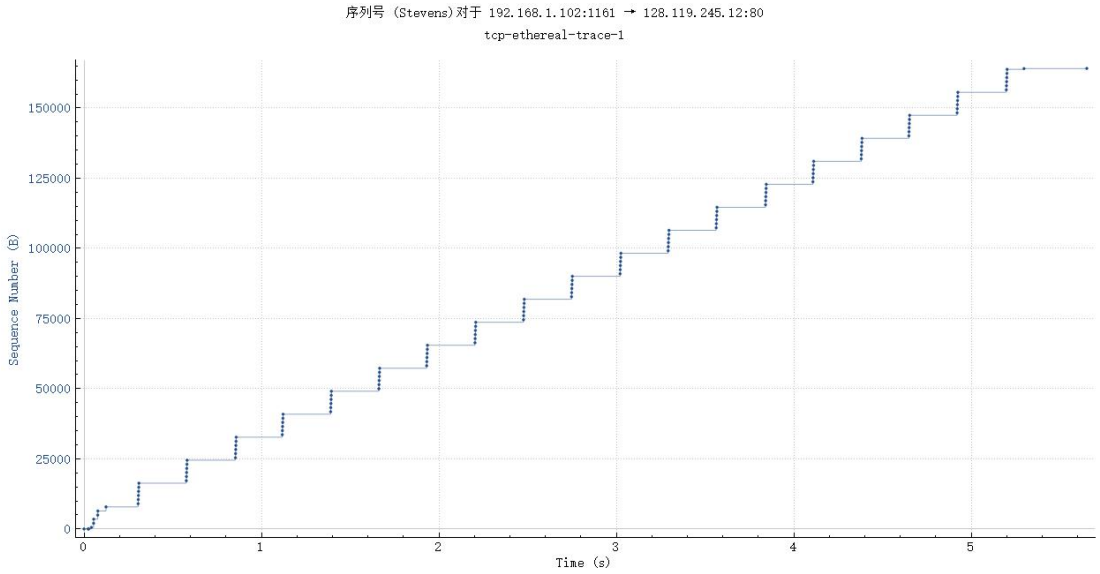

图 11

Q11: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 257 in the text).

A11:接收方通常一次 ACK1460 字节，从图 12 中可以看出，接收方在收到两个 TCP 报文段后，才发送一个 ACK 报文段。

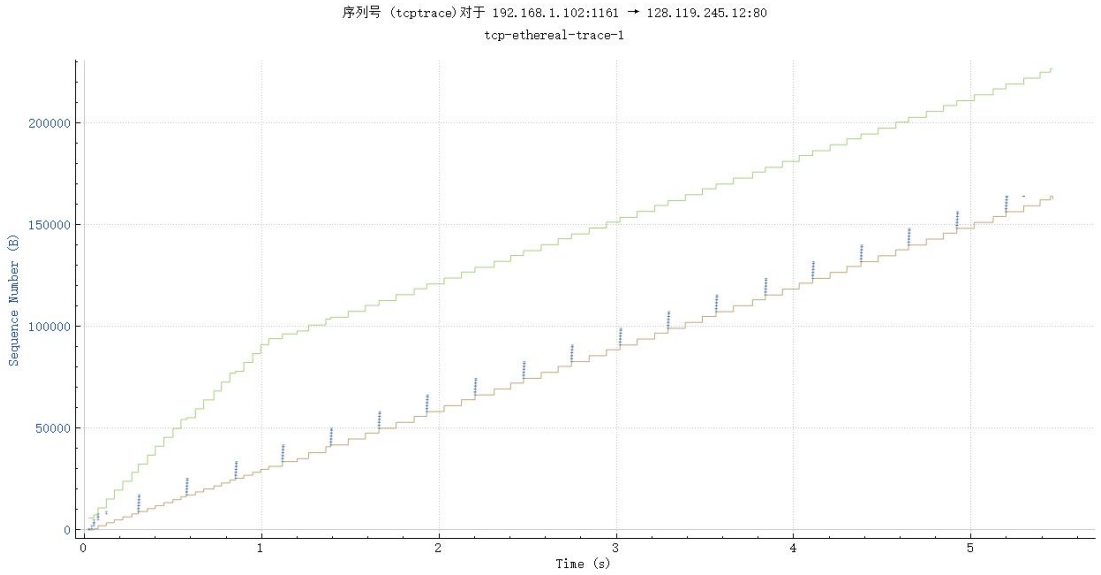

图 12

Q12:What is the throughput (bytes transferred per unit time) for the TCP connection? Explain how you calculated this value.

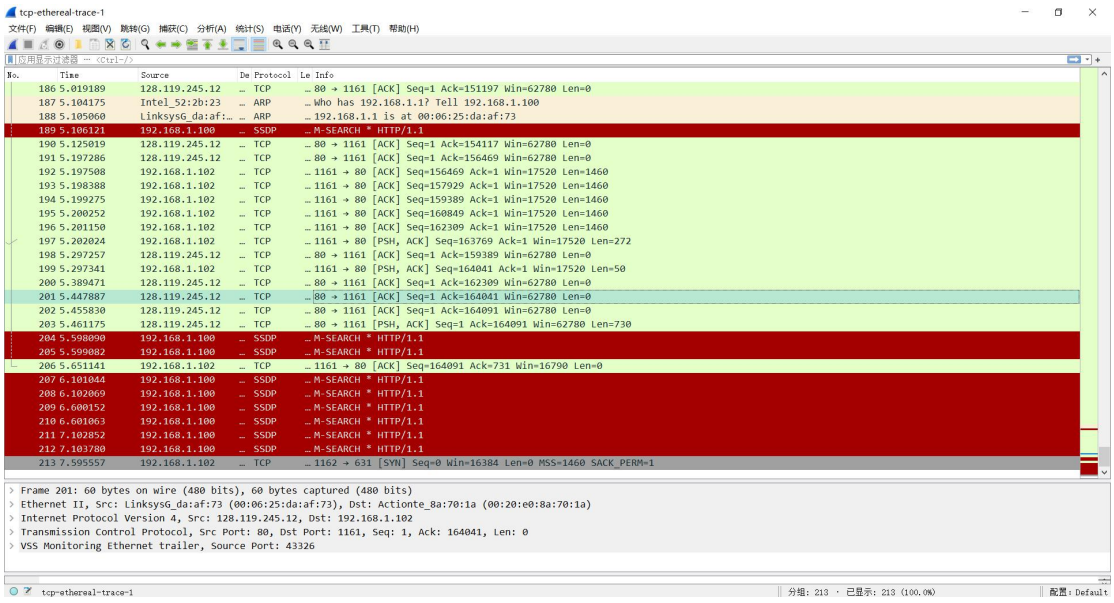

图 13

A12:从吞吐量-时间图(图 14)中，可见平均吞吐量在 200000-250000bits/s 之间。计算可用图 13 中序号为 201 的封包信息：ACK=164041，时间 5.447887s，求得：

$$\overline{Throughout} = \frac{164041*8}{5.447887} = 240888bps$$

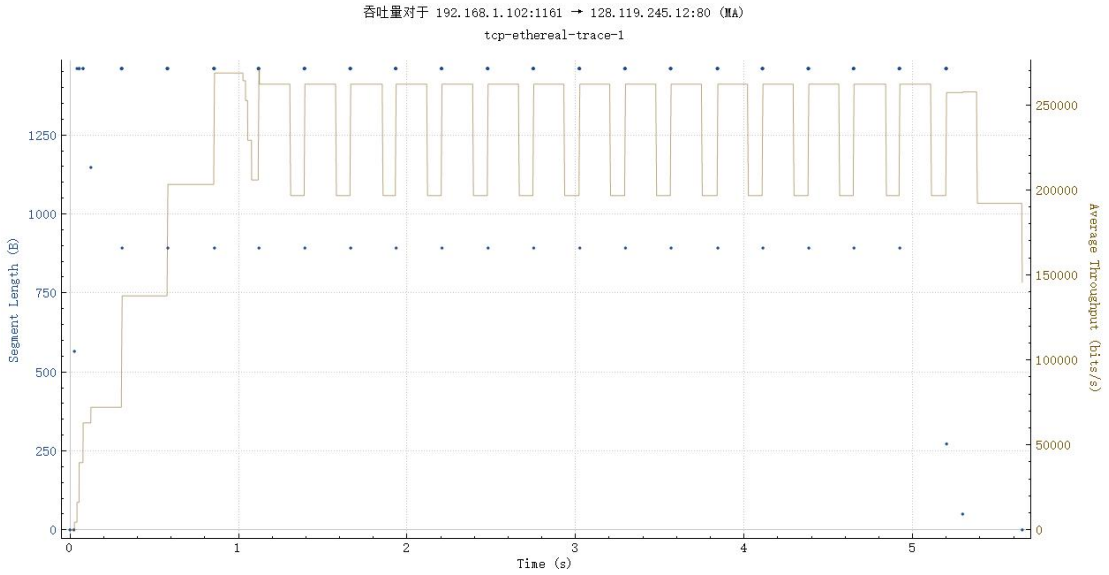

图 14

TCP congestion control in action

实验步骤

1.选择封包列表中的 TCP 报文段，然后选择菜单中：统计->TCP 流图形->时间序列(Stevens)。

问题

Q13: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.

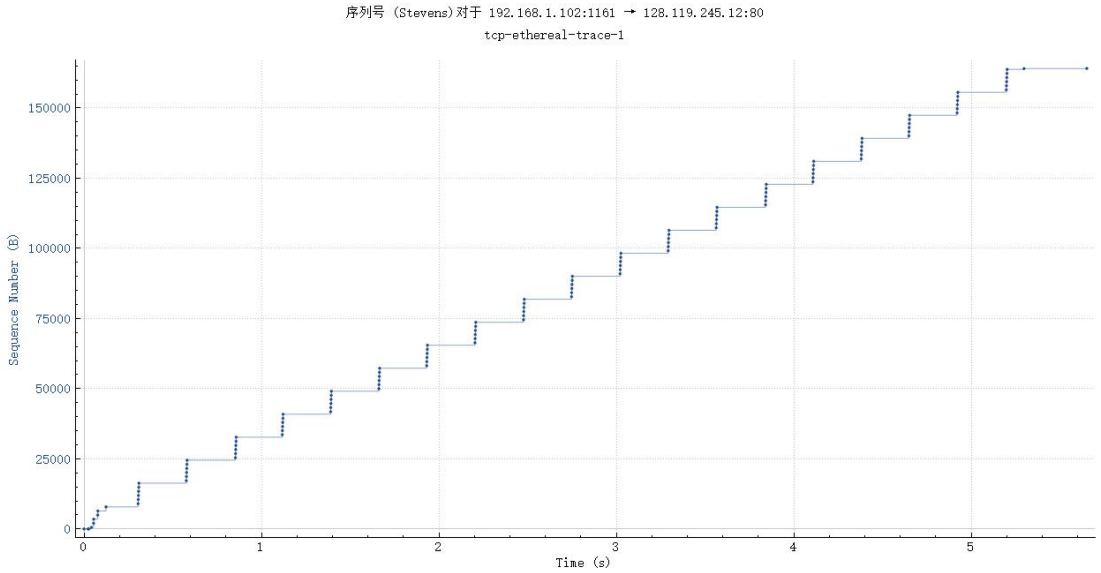

图 15

A13:由图 15 可见，发送方每隔一段时间发送相同数量的报文段，无法区分出慢启动阶段和拥塞避免阶段，与课本上所学并不一致。

Q14: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.

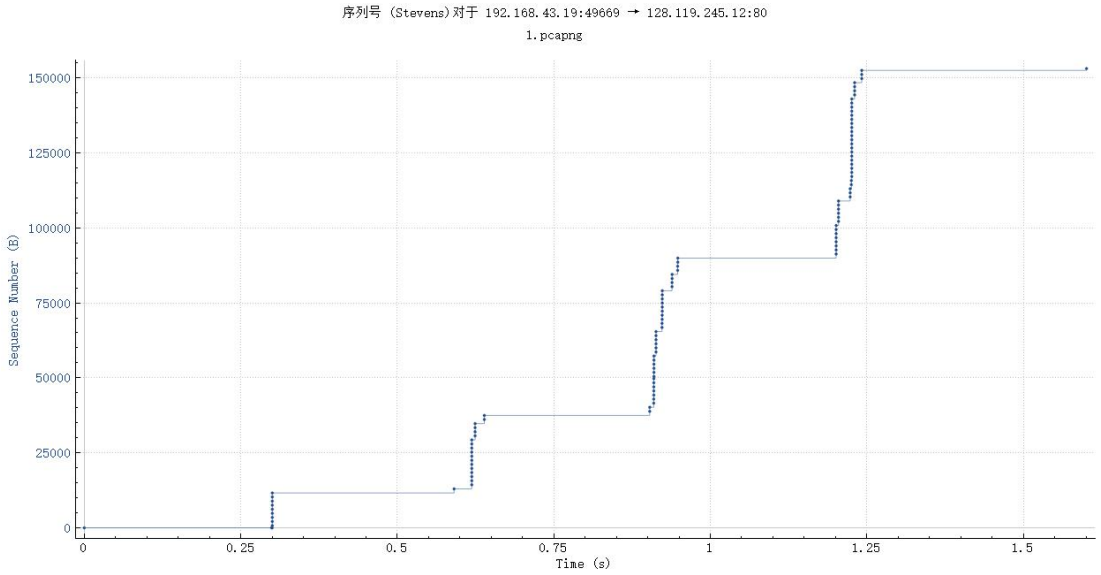

图 16

A14:图 16 为我的计算机发送 alice.txt 至 gaia.cs.umass.edu 的时间序列图，可以看出每隔一段时间发送的报文段数指数倍增长，故仍在慢启动阶段，还未进入拥塞避免阶段；与课本中所学相同。

总结

本次实验通过 WireShark 分组嗅探器捕获并分析 TCP 报文段，深入了解了 TCP 报文段的结构，对报文中的内容有了深刻的理解，在回顾教材内容的同时又有所提升。在比较下载所得的报文与自己捕获的报文的过程中，进一步加深了对传输层的理解。

附：本次实验使用 WireShark 分组嗅探器获得的数据部分以截图形式体现于实验报告中，完整数据在 Github 仓库中。