

计算机网络 TCP 实验报告

姓名：陈鹤影

学号：PB21061287

日期：2022.10.23

一、实验目的：

1. 熟悉并了解 TCP 报文的具体格式，掌握 Seq、ACK、WIN 等。
2. 掌握 TCP 的建立过程——三次握手。
3. 通过对 TCP 流的分析，掌握 TCP 的拥塞控制机制。
4. 学习利用 Wireshark 绘制 TCP 流图形。
5. 掌握 TCP 连接性能的相关计算。

二、实验流程及问题回答：

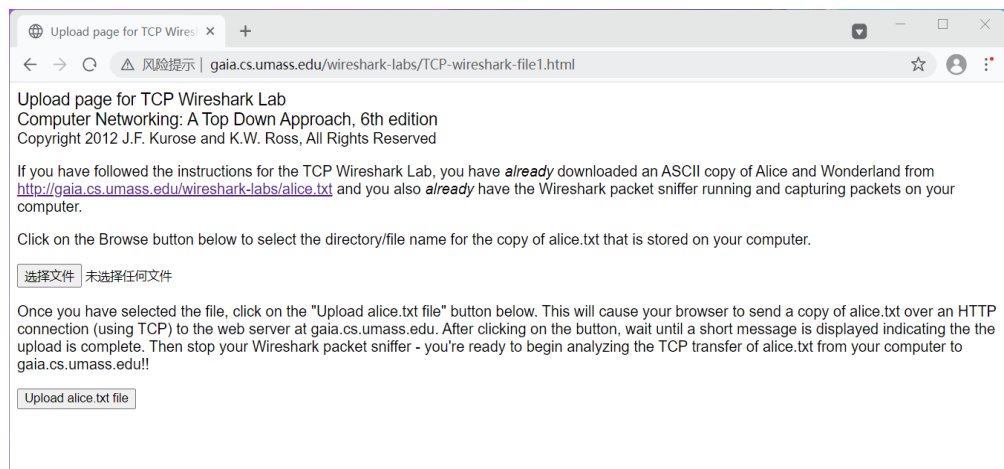
Part 1: Capturing a bulk TCP transfer from your computer to a remote server

1. 实验内容：

使用 Wireshark 跟踪文件从主机到远程服务器的 TCP 传输，利用 HTTP POST 方法将文件发送到 Web 服务器（传输较大的数据量）。

2. 实验流程：

- 1) 获取需要传输的文件（获取文件储存在 alice.txt 中）
- 2) 登入 <http://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html> 网页。界面如图所示：



(图 1-1 文件上传登入界面)

- 3) 浏览并选择对应的文件（alice.txt），启动 Wireshark 抓包，上传文件。
- 4) 上传成功，结束 Wireshark 抓包。抓包结果如图所示：

No.	Time	Source	Destination	Frame	Protocol	Length	Info
19	2022-10-24 00:17:25.382112	192.168.43.252	128.119.245.12	✓	TCP	66	58307 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
20	2022-10-24 00:17:25.798663	128.119.245.12	192.168.43.252	✓	TCP	66	80 → 58307 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1360 SACK_PERM=1 WS=128
21	2022-10-24 00:17:25.798663	128.119.245.12	192.168.43.252	✓	TCP	66	80 → 58399 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1360 SACK_PERM=1 WS=128
22	2022-10-24 00:17:25.798663	128.119.245.12	192.168.43.252	✓	TCP	66	80 → 58307 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1360 SACK_PERM=1 WS=128
23	2022-10-24 00:17:25.798872	192.168.43.252	128.119.245.12	✓	TCP	54	64982 → 80 [ACK] Seq=1 Ack=1 Win=66560 Len=0
24	2022-10-24 00:17:25.798957	192.168.43.252	128.119.245.12	✓	TCP	54	58399 → 80 [ACK] Seq=1 Ack=1 Win=66560 Len=0
25	2022-10-24 00:17:25.798979	192.168.43.252	128.119.245.12	✓	TCP	54	58307 → 80 [ACK] Seq=1 Ack=1 Win=66560 Len=0
26	2022-10-24 00:17:25.799772	192.168.43.252	128.119.245.12	✓	TCP	781	58307 → 80 [PSH, ACK] Seq=1 Ack=1 Win=66560 Len=727 [TCP segment of a reassembled f
27	2022-10-24 00:17:25.800095	192.168.43.252	128.119.245.12	✓	TCP	12294	58307 → 80 [ACK] Seq=728 Ack=1 Win=66560 Len=12240 [TCP segment of a reassembled P
28	2022-10-24 00:17:26.132925	192.168.43.252	183.2.143.108	✓	TCP	55	[TCP Keep-Alive] 61603 → 443 [ACK] Seq=1 Ack=1 Win=258 Len=1
29	2022-10-24 00:17:26.227174	128.119.245.12	192.168.43.252	✓	TCP	66	[TCP Window Update] 80 → 58307 [ACK] Seq=1 Ack=1 Win=32128 Len=0 SLE=728 SRE=2088
30	2022-10-24 00:17:26.227174	128.119.245.12	192.168.43.252	✓	TCP	54	80 → 58307 [ACK] Seq=1 Ack=2088 Win=33664 Len=0
31	2022-10-24 00:17:26.227174	128.119.245.12	192.168.43.252	✓	TCP	54	80 → 58307 [ACK] Seq=1 Ack=6168 Win=42368 Len=0
32	2022-10-24 00:17:26.227174	128.119.245.12	192.168.43.252	✓	TCP	66	80 → 58307 [ACK] Seq=1 Ack=2088 Win=36608 Len=0 SLE=4808 SRE=6168
33	2022-10-24 00:17:26.227174	128.119.245.12	192.168.43.252	✓	TCP	66	[TCP Window Update] 80 → 58307 [ACK] Seq=1 Ack=2088 Win=39424 Len=0 SLE=3448 SRE=61
34	2022-10-24 00:17:26.227304	192.168.43.252	128.119.245.12	✓	TCP	12294	58307 → 80 [PSH, ACK] Seq=12968 Ack=1 Win=66560 Len=12240 [TCP segment of a reasem
35	2022-10-24 00:17:26.227502	128.119.245.12	192.168.43.252	✓	TCP	54	80 → 58307 [ACK] Seq=1 Ack=7528 Win=45312 Len=0
36	2022-10-24 00:17:26.227502	192.168.43.252	128.119.245.12	✓	TCP	2774	58307 → 80 [ACK] Seq=25208 Ack=1 Win=66560 Len=2720 [TCP segment of a reassembled f
37	2022-10-24 00:17:26.280955	183.2.143.108	192.168.43.252	✓	TCP	66	[TCP Keep-Alive ACK] 443 → 61603 [ACK] Seq=1 Ack=2 Win=260 Len=0 SLE=1 SRE=2
38	2022-10-24 00:17:26.286458	128.119.245.12	192.168.43.252	✓	TCP	54	80 → 58307 [ACK] Seq=1 Ack=8888 Win=48256 Len=0
39	2022-10-24 00:17:26.286521	192.168.43.252	128.119.245.12	✓	TCP	2774	58307 → 80 [ACK] Seq=27928 Ack=1 Win=66560 Len=2720 [TCP segment of a reassembled f
40	2022-10-24 00:17:26.363692	192.168.43.252	128.119.245.12	✓	TCP	54	80 → 58307 [ACK] Seq=1 Ack=10248 Win=51200 Len=0
41	2022-10-24 00:17:26.363802	128.119.245.12	192.168.43.252	✓	TCP	2774	58307 → 80 [ACK] Seq=30648 Ack=1 Win=66560 Len=2720 [TCP segment of a reassembled f
42	2022-10-24 00:17:26.363887	192.168.43.252	128.119.245.12	✓	TCP	54	80 → 58307 [ACK] Seq=1 Ack=11608 Win=54016 Len=0
43	2022-10-24 00:17:26.367400	128.119.245.12	192.168.43.252	✓	TCP	2774	58307 → 80 [PSH, ACK] Seq=33368 Ack=1 Win=66560 Len=2720 [TCP segment of a reasem
44	2022-10-24 00:17:26.367484	192.168.43.252	128.119.245.12	✓	TCP	54	80 → 58307 [ACK] Seq=1 Ack=12968 Win=56960 Len=0
45	2022-10-24 00:17:26.417498	128.119.245.12	192.168.43.252	✓	TCP	54	80 → 58307 [ACK] Seq=1 Ack=12968 Win=56960 Len=0

(图 1-2 Wireshark 对大文件上传过程的抓包)

Part 2: A first look at the captured trace

1. 实验内容：

初步分析获得的 trace。

2. 实验流程：

1) 过滤出“tcp”数据包

2) 分析数据可以观察到 TCP 的三次握手过程：

- 客户端 TCP 首先向服务器端 TCP 发送 SYN 报文段 (SYN = 1)。客户随机选择初始序列号 (Seq = 0)。
- 服务器提取 TCP SYN 报文段，向客户 TCP 发送允许连接的 SYNACK 报文段 (SYN = 1, ACK = 客户 TCP Seq+1,)。服务器随机选择初始序列号 (Seq = 0)。
- 接收到 SYNACK 报文段后，客户给该 TCP 连接分配缓存和变量。客户主机向服务器主机发送报文段 (SYN = 0, ACK = 服务器 TCP Seq+1)。

观察到过程如下，其中我们可以观察到客户 TCP 向服务器发送了多个报文段，同时收到了多个回复。

我们观察到在完成第三次握手后，TCP 连接进入 ESTABLISHED 状态，随后客户和服务器开始传送数据。可以观察到一个携带应用数据的 PSH 包。

No.	Time	Source	Destination	Frame	Protocol	Length	Info
7	2022-10-24 00:17:25.123251	192.168.43.252	128.119.245.12	✓	TCP	66	58399 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
8	2022-10-24 00:17:25.123592	192.168.43.252	128.119.245.12	✓	TCP	66	64982 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
9	2022-10-24 00:17:25.280560	124.71.151.201	192.168.43.252	✓	TLSv1.2	106	Application Data
11	2022-10-24 00:17:25.280569	124.71.151.201	192.168.43.252	✓	TCP	54	443 → 52183 [ACK] Seq=1 Ack=119 Win=95 Len=0
12	2022-10-24 00:17:25.280569	124.71.151.201	192.168.43.252	✓	TCP	54	443 → 52183 [ACK] Seq=1 Ack=158 Win=95 Len=0
13	2022-10-24 00:17:25.280569	124.71.151.201	192.168.43.252	✓	TCP	54	443 → 52183 [ACK] Seq=1 Ack=560 Win=97 Len=0
14	2022-10-24 00:17:25.280569	124.71.151.201	192.168.43.252	✓	TLSv1.2	429	Application Data
15	2022-10-24 00:17:25.286705	192.168.43.252	124.71.151.201	✓	TCP	54	52183 → 443 [ACK] Seq=560 Ack=53 Win=255 Len=0
16	2022-10-24 00:17:25.286796	192.168.43.252	124.71.151.201	✓	TCP	54	[TCP Dup ACK 1581] 52183 → 443 [ACK] Seq=560 Ack=53 Win=255 Len=0
17	2022-10-24 00:17:25.286820	192.168.43.252	124.71.151.201	✓	TCP	54	[TCP Dup ACK 1582] 52183 → 443 [ACK] Seq=560 Ack=53 Win=255 Len=0
18	2022-10-24 00:17:25.286960	192.168.43.252	124.71.151.201	✓	TCP	54	52183 → 443 [ACK] Seq=560 Ack=53 Win=255 Len=0
19	2022-10-24 00:17:25.382112	192.168.43.252	128.119.245.12	✓	TCP	66	58307 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
20	2022-10-24 00:17:25.798663	128.119.245.12	192.168.43.252	✓	TCP	66	80 → 64982 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1360 SACK_PERM=1 WS=128
21	2022-10-24 00:17:25.798663	128.119.245.12	192.168.43.252	✓	TCP	66	80 → 58399 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1360 SACK_PERM=1 WS=128
22	2022-10-24 00:17:25.798663	128.119.245.12	192.168.43.252	✓	TCP	66	80 → 58307 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1360 SACK_PERM=1 WS=128
23	2022-10-24 00:17:25.798872	192.168.43.252	128.119.245.12	✓	TCP	54	64982 → 80 [ACK] Seq=1 Ack=1 Win=66560 Len=0
24	2022-10-24 00:17:25.798957	192.168.43.252	128.119.245.12	✓	TCP	54	58399 → 80 [ACK] Seq=1 Ack=1 Win=66560 Len=0
25	2022-10-24 00:17:25.798979	192.168.43.252	128.119.245.12	✓	TCP	54	58307 → 80 [ACK] Seq=1 Ack=1 Win=66560 Len=0
26	2022-10-24 00:17:25.799772	192.168.43.252	128.119.245.12	✓	TCP	781	58307 → 80 [PSH, ACK] Seq=1 Ack=1 Win=66560 Len=727 [TCP segment of a reassembled PDU]
27	2022-10-24 00:17:26.132925	192.168.43.252	183.2.143.108	✓	TCP	55	[TCP Keep-Alive] 61603 → 443 [ACK] Seq=1 Ack=1 Win=258 Len=1
28	2022-10-24 00:17:26.227174	128.119.245.12	192.168.43.252	✓	TCP	66	[TCP Window Update] 80 → 58307 [ACK] Seq=1 Ack=1 Win=32128 Len=0 SLE=728 SRE=2088
29	2022-10-24 00:17:26.227174	128.119.245.12	192.168.43.252	✓	TCP	54	80 → 58307 [ACK] Seq=1 Ack=2088 Win=33664 Len=0
30	2022-10-24 00:17:26.227174	128.119.245.12	192.168.43.252	✓	TCP	66	80 → 58307 [ACK] Seq=1 Ack=2088 Win=36608 Len=0 SLE=4808 SRE=6168
31	2022-10-24 00:17:26.227174	128.119.245.12	192.168.43.252	✓	TCP	66	[TCP Window Update] 80 → 58307 [ACK] Seq=1 Ack=2088 Win=39424 Len=0 SLE=3448 SRE=6168
32	2022-10-24 00:17:26.227174	128.119.245.12	192.168.43.252	✓	TCP	54	80 → 58307 [ACK] Seq=1 Ack=6168 Win=42368 Len=0
33	2022-10-24 00:17:26.227174	128.119.245.12	192.168.43.252	✓	TCP	66	[TCP Window Update] 80 → 58307 [ACK] Seq=1 Ack=2088 Win=36608 Len=0 SLE=4808 SRE=6168

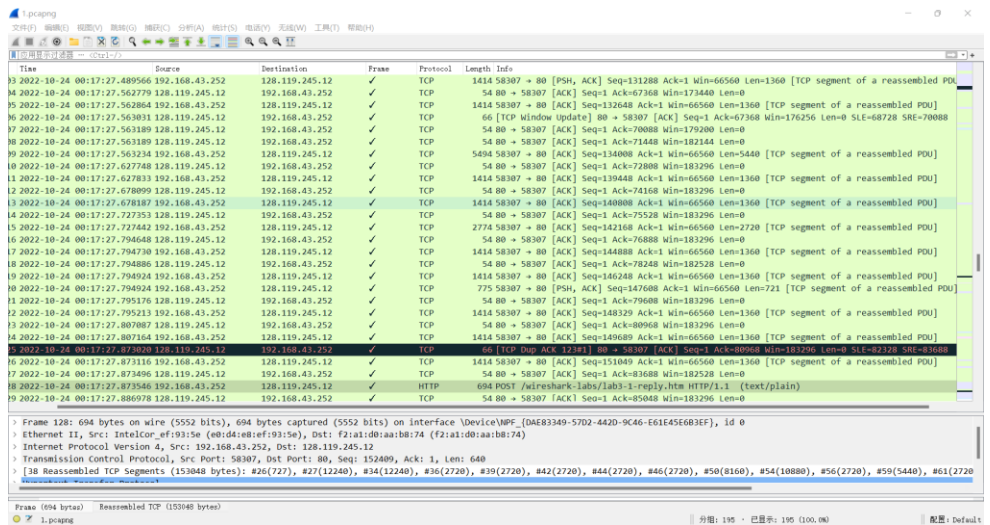
(图 1-3 三次握手过程)

3) 可以观察到 HTTP POST 报文:

127	2022-10-24 00:17:27.873496	128.119.245.12	192.168.43.252	✓	TCP	54 80 → 58307 [ACK] Seq=1 Ack=83688 Win=182528 Len=0
128	2022-10-24 00:17:27.873546	192.168.43.252	128.119.245.12	✓	HTTP	694 POST /wtfreshark-labs/lab3-1-reply.htm HTTP/1.1 (text/plain)
129	2022-10-24 00:17:27.886978	128.119.245.12	192.168.43.252	✓	TCP	54 80 → 58307 [ACK] Seq=1 Ack=85848 Win=182296 Len=0
130	2022-10-24 00:17:27.952701	128.119.245.12	192.168.43.252	✓	TCP	54 80 → 58307 [ACK] Seq=1 Ack=87768 Win=182528 Len=0
131	2022-10-24 00:17:27.967164	128.119.245.12	192.168.43.252	✓	TCP	54 80 → 58307 [ACK] Seq=1 Ack=89196 Win=182296 Len=0

(图 1-4 HTTP POST 报文)

4) 同样也可以观察到大量带有“[TCP segment of a reassembled PDU]”信息的报文段。
(在基于 TCP 的传输中, 如果应用层消息过大 (如超过 MSS) TCP Segment 不能一次包含全部的应用层 PDU, 而要把一个完整消息分成多个段, 就会将除了最后一个报文段 (segment) 的所有其他报文段都打上“TCP segment of a reassembled PDU”。) 同时也可以看到服务器返回的大量确认报文。

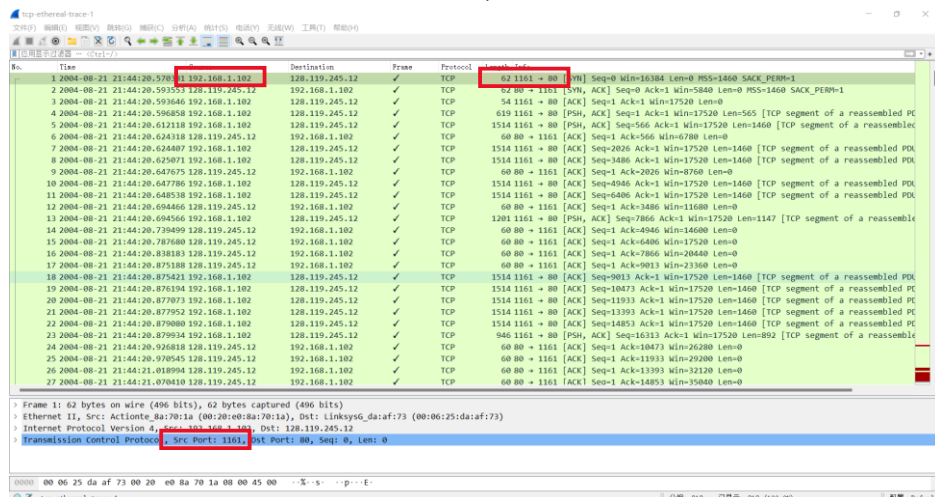


(图 1-5 [TCP segment of a reassembled PDU]and[ACK])

3. Q&A(以下内容利用所给 trace 和本机 trace 分别分析):

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?

Ans: IP address: 192.168.1.102; TCP port number: 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?

Ans: IP address: 128.119.245.12; TCP port number: 80;

No.	Time	Source	Destination	Frame	Protocol	Length	Info
1	2004-08-21 21:44:20.570381	192.168.1.102	128.119.245.12	✓	TCP	60	1161 → 80 [SYN] Seq=0 Win=0 Len=0
2	2004-08-21 21:44:20.593553	128.119.245.12	192.168.1.102	✓	TCP	62	80 → 1161 [SYN, ACK] Seq=0 Ack=1 Win=0 Len=0
3	2004-08-21 21:44:20.593646	192.168.1.102	128.119.245.12	✓	TCP	54	1161 → 80 [ACK] Seq=1 Ack=1 Win=0 Len=0
4	2004-08-21 21:44:20.596858	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	2004-08-21 21:44:20.612118	192.168.1.102	128.119.245.12	✓	TCP	514	1161 → 80 [PSH, ACK] Seq=566 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
6	2004-08-21 21:44:20.624318	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=2026 Ack=1 Win=0 Len=0
7	2004-08-21 21:44:20.624807	192.168.1.102	128.119.245.12	✓	TCP	514	1161 → 80 [ACK] Seq=2026 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
8	2004-08-21 21:44:20.625071	192.168.1.102	128.119.245.12	✓	TCP	514	1161 → 80 [ACK] Seq=3486 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
9	2004-08-21 21:44:20.647675	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=2026 Win=0 Len=0
10	2004-08-21 21:44:20.647796	192.168.1.102	128.119.245.12	✓	TCP	514	1161 → 80 [ACK] Seq=4046 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
11	2004-08-21 21:44:20.648538	192.168.1.102	128.119.245.12	✓	TCP	514	1161 → 80 [ACK] Seq=4046 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
12	2004-08-21 21:44:20.694466	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=3486 Win=0 Len=0
13	2004-08-21 21:44:20.694566	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	2004-08-21 21:44:20.739499	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=4046 Win=0 Len=0
15	2004-08-21 21:44:20.787680	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=4046 Win=0 Len=0
16	2004-08-21 21:44:20.838183	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=7866 Win=0 Len=0
17	2004-08-21 21:44:20.875188	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=9013 Win=0 Len=0
18	2004-08-21 21:44:20.875421	192.168.1.102	128.119.245.12	✓	TCP	514	1161 → 80 [ACK] Seq=9013 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
19	2004-08-21 21:44:20.876194	192.168.1.102	128.119.245.12	✓	TCP	514	1161 → 80 [ACK] Seq=10473 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
20	2004-08-21 21:44:20.877073	192.168.1.102	128.119.245.12	✓	TCP	514	1161 → 80 [ACK] Seq=11933 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
21	2004-08-21 21:44:20.877952	192.168.1.102	128.119.245.12	✓	TCP	514	1161 → 80 [ACK] Seq=13393 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
22	2004-08-21 21:44:20.879080	192.168.1.102	128.119.245.12	✓	TCP	514	1161 → 80 [ACK] Seq=14853 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
23	2004-08-21 21:44:20.879934	192.168.1.102	128.119.245.12	✓	TCP	946	1161 → 80 [PSH, ACK] Seq=16313 Ack=1 Win=17520 Len=892 [TCP segment of a reassembled PDU]
24	2004-08-21 21:44:20.926818	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=10473 Win=0 Len=0
25	2004-08-21 21:44:20.976455	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=11933 Win=0 Len=0
26	2004-08-21 21:44:21.018994	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=13393 Win=0 Len=0
27	2004-08-21 21:44:21.070410	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=14853 Win=0 Len=0

Frame 1: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on interface [Device]VMPF [0x83349-5702-4420-9C46-E61E5E683EF], id 0
 Ethernet II, Src: Actionte_Ra70:1a (00:20:e0:ba:70:1a), Dst: Linksys_daf: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

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?

Ans: IP address: 192.168.43.252; TCP port number: 58307;

No.	Time	Source	Destination	Frame	Protocol	Length	Info
13	2022-10-24 00:17:25.280569	124.71.151.201	192.168.43.252	✓	TCP	54	443 → 52183 [ACK] Seq=1 Ack=560 Win=97 Len=0
14	2022-10-24 00:17:25.280569	124.71.151.201	192.168.43.252	✓	TLSv1.2	429	Application Data
15	2022-10-24 00:17:25.280795	192.168.43.252	124.71.151.201	✓	TCP	54	52183 → 443 [ACK] Seq=560 Ack=53 Win=255 Len=0
16	2022-10-24 00:17:25.280826	192.168.43.252	124.71.151.201	✓	TCP	54	[TCP Dup ACK 15#1] 52183 → 443 [ACK] Seq=560 Ack=53 Win=255 Len=0
17	2022-10-24 00:17:25.280826	192.168.43.252	124.71.151.201	✓	TCP	54	[TCP Dup ACK 15#2] 52183 → 443 [ACK] Seq=560 Ack=53 Win=255 Len=0
18	2022-10-24 00:17:25.280826	192.168.43.252	124.71.151.201	✓	TCP	60	58307 → 80 [RST] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
19	2022-10-24 00:17:25.798653	128.119.245.12	192.168.43.252	✓	TCP	60	80 → 58307 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1360 SACK_PERM=1 WS=128
20	2022-10-24 00:17:25.798653	128.119.245.12	192.168.43.252	✓	TCP	60	80 → 58309 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1360 SACK_PERM=1 WS=128
21	2022-10-24 00:17:25.798653	128.119.245.12	192.168.43.252	✓	TCP	60	80 → 58307 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1360 SACK_PERM=1 WS=128
22	2022-10-24 00:17:25.798872	192.168.43.252	128.119.245.12	✓	TCP	54	64982 → 80 [ACK] Seq=1 Ack=1 Win=66560 Len=0
23	2022-10-24 00:17:25.798957	192.168.43.252	128.119.245.12	✓	TCP	54	58399 → 80 [ACK] Seq=1 Ack=1 Win=66560 Len=0
24	2022-10-24 00:17:25.798957	192.168.43.252	128.119.245.12	✓	TCP	54	58307 → 80 [ACK] Seq=1 Ack=1 Win=66560 Len=0
25	2022-10-24 00:17:25.799772	192.168.43.252	128.119.245.12	✓	TCP	781	58307 → 80 [PSH, ACK] Seq=1 Ack=1 Win=66560 Len=727 [TCP segment of a reassembled PDU]
26	2022-10-24 00:17:25.800095	192.168.43.252	128.119.245.12	✓	TCP	12294	58307 → 80 [ACK] Seq=728 Ack=1 Win=66560 Len=12240 [TCP segment of a reassembled PDU]
27	2022-10-24 00:17:26.132925	192.168.43.252	183.2.143.108	✓	TCP	55	[TCP Keep-Alive] 61603 → 443 [ACK] Seq=1 Ack=1 Win=250 Len=1
28	2022-10-24 00:17:26.227174	128.119.245.12	192.168.43.252	✓	TCP	66	[TCP Window Update] 80 → 58307 [ACK] Seq=1 Ack=2088 Win=3608 Len=0 SLE=728 SRE=2088
29	2022-10-24 00:17:26.227174	128.119.245.12	192.168.43.252	✓	TCP	54	80 → 58307 [ACK] Seq=1 Ack=2088 Win=3608 Len=0
30	2022-10-24 00:17:26.227174	128.119.245.12	192.168.43.252	✓	TCP	54	80 → 58307 [ACK] Seq=1 Ack=6168 Win=42368 Len=0
31	2022-10-24 00:17:26.227174	128.119.245.12	192.168.43.252	✓	TCP	66	80 → 58307 [ACK] Seq=1 Ack=3608 Win=3608 Len=0 SLE=4888 SRE=6168
32	2022-10-24 00:17:26.227174	128.119.245.12	192.168.43.252	✓	TCP	66	[TCP Window Update] 80 → 58307 [ACK] Seq=1 Ack=2088 Win=39424 Len=0 SLE=3448 SRE=6168
33	2022-10-24 00:17:26.227304	192.168.43.252	128.119.245.12	✓	TCP	12294	58307 → 80 [PSH, ACK] Seq=12968 Ack=1 Win=66560 Len=12240 [TCP segment of a reassembled PDU]
34	2022-10-24 00:17:26.227502	128.119.245.12	192.168.43.252	✓	TCP	54	80 → 58307 [ACK] Seq=1 Ack=7528 Win=45312 Len=0
35	2022-10-24 00:17:26.227560	192.168.43.252	128.119.245.12	✓	TCP	2774	58307 → 80 [ACK] Seq=25208 Ack=1 Win=66560 Len=2720 [TCP segment of a reassembled PDU]
36	2022-10-24 00:17:26.280955	183.2.143.108	192.168.43.252	✓	TCP	66	[TCP Keep-Alive] 443 → 61603 [ACK] Seq=1 Ack=2 Win=260 Len=0 SLE=1 SRE=2
37	2022-10-24 00:17:26.280955	192.168.43.252	128.119.245.12	✓	TCP	54	80 → 58307 [ACK] Seq=1 Ack=8888 Win=48256 Len=0
38	2022-10-24 00:17:26.280521	192.168.43.252	128.119.245.12	✓	TCP	2774	58307 → 80 [ACK] Seq=27928 Ack=1 Win=66560 Len=2720 [TCP segment of a reassembled PDU]

Frame 128: 604 bytes on wire (5552 bits), 604 bytes captured (5552 bits) on interface [Device]VMPF [0x83349-5702-4420-9C46-E61E5E683EF], id 0
 Ethernet II, Src: IntelCor_ef:93:5e (e0:d4:e0:ef:93:5e), Dst: f2:a1:d0:aa:bb:74 (f2:a1:d0:aa:bb:74)
 Internet Protocol Version 4, Src: 192.168.43.252, Dst: 128.119.245.12
 Transmission Control Protocol, Src Port: 58307, Dst Port: 80, Seq: 153409, Ack: 1, Len: 640
 138 Reassembled TCP Segments (153048 bytes): #26(727), #27(12240), #34(12240), #36(2720), #39(2720), #42(2720), #44(2720), #46(2720), #50(8160), #54(10880), #56(2720), #59(5440), #61(2720)

Part 3: TCP Basics

1. 实验内容:

分析 TCP 数据流。

2. 实验流程:

选择分析-关闭 HTTP 协议，关闭后 Wireshark 界面如图所示：

No.	Time	Source	Destination	Frame	Protocol	Length	Info
1	2004-08-21 21:44:20.570381	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	2004-08-21 21:44:20.593553	128.119.245.12	192.168.1.102	✓	TCP	62	80 → 1161 [ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460 SACK_PERM=1
3	2004-08-21 21:44:20.593646	192.168.1.102	128.119.245.12	✓	TCP	54	1161 → 80 [ACK] Seq=1 Ack=1 Win=17520 Len=0
4	2004-08-21 21:44:20.596858	192.168.1.102	128.119.245.12	✓	TCP	619	1161 → 80 [PSH, ACK] Seq=1 Ack=1 Win=17520 Len=565
5	2004-08-21 21:44:20.612118	192.168.1.102	128.119.245.12	✓	TCP	1514	1161 → 80 [PSH, ACK] Seq=566 Ack=1 Win=17520 Len=1460
6	2004-08-21 21:44:20.624407	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=566 Win=6780 Len=0
7	2004-08-21 21:44:20.624407	192.168.1.102	128.119.245.12	✓	TCP	1514	1161 → 80 [ACK] Seq=2026 Ack=1 Win=17520 Len=1460
8	2004-08-21 21:44:20.625071	192.168.1.102	128.119.245.12	✓	TCP	1514	1161 → 80 [ACK] Seq=3486 Ack=1 Win=17520 Len=1460
9	2004-08-21 21:44:20.647675	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=2026 Win=8760 Len=0
10	2004-08-21 21:44:20.647786	192.168.1.102	128.119.245.12	✓	TCP	1514	1161 → 80 [ACK] Seq=4946 Ack=1 Win=17520 Len=1460
11	2004-08-21 21:44:20.648538	192.168.1.102	128.119.245.12	✓	TCP	1514	1161 → 80 [ACK] Seq=6406 Ack=1 Win=17520 Len=1460
12	2004-08-21 21:44:20.694466	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=3486 Win=11680 Len=0
13	2004-08-21 21:44:20.694566	192.168.1.102	128.119.245.12	✓	TCP	1201	1161 → 80 [PSH, ACK] Seq=7866 Ack=1 Win=17520 Len=1147
14	2004-08-21 21:44:20.739499	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=4946 Win=14600 Len=0
15	2004-08-21 21:44:20.787688	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=6406 Win=17520 Len=0
16	2004-08-21 21:44:20.838183	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=7866 Win=20480 Len=0
17	2004-08-21 21:44:20.875188	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=9013 Win=23360 Len=0
18	2004-08-21 21:44:20.875421	192.168.1.102	128.119.245.12	✓	TCP	1514	1161 → 80 [ACK] Seq=9013 Ack=1 Win=17520 Len=1460
19	2004-08-21 21:44:20.876194	192.168.1.102	128.119.245.12	✓	TCP	1514	1161 → 80 [ACK] Seq=10473 Ack=1 Win=17520 Len=1460
20	2004-08-21 21:44:20.877073	192.168.1.102	128.119.245.12	✓	TCP	1514	1161 → 80 [ACK] Seq=11933 Ack=1 Win=17520 Len=1460
21	2004-08-21 21:44:20.877952	192.168.1.102	128.119.245.12	✓	TCP	1514	1161 → 80 [ACK] Seq=13393 Ack=1 Win=17520 Len=1460
22	2004-08-21 21:44:20.879080	192.168.1.102	128.119.245.12	✓	TCP	1514	1161 → 80 [ACK] Seq=14853 Ack=1 Win=17520 Len=1460
23	2004-08-21 21:44:20.879349	192.168.1.102	128.119.245.12	✓	TCP	946	1161 → 80 [PSH, ACK] Seq=16113 Ack=1 Win=17520 Len=892
24	2004-08-21 21:44:20.926818	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=10473 Win=26280 Len=0
25	2004-08-21 21:44:20.970545	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=11933 Win=29280 Len=0
26	2004-08-21 21:44:21.018994	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=13393 Win=32120 Len=0
27	2004-08-21 21:44:21.070410	128.119.245.12	192.168.1.102	✓	TCP	60	80 → 1161 [ACK] Seq=1 Ack=14853 Win=35040 Len=0

(图 1-6 在分析中关闭 TCP 协议后界面)

3. Q&A(以下内容利用所给 trace 分析):

- 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?

Ans: Sequence number: 0; 报文段的 SYN 标志被置 1 及“Connection establish request(SYN):server port 80”;

```

Frame 1: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on interface 0
Ethernet II, Src: Actione_Ba70:1a (00:20:e0:ba:70:1a), Dst: Linksys0_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]
  [Conversation completeness: Incomplete, DATA (15)]
  [TCP segment length: 61]
  Sequence Number: 0 (relative sequence number)
  [Next Sequence Number: 1 (relative sequence number)]
  Acknowledgment Number: 0
  Acknowledgment number (raw): 0
  0111 .... = Header Length: 20 bytes (7)
  Flags: 0x002 (SYN)
    SYN .... = Reserved: Not set
    ...0 .... = Nonce: Not set
    ...0 .... = Congestion Window Reduced (CWR): Not set
    ...0 .... = ECH Echo: Not set
    ...0 .... = Urgent: Not set
    ...0 .... = Acknowledgment: Not set
    ...0 .... = Push: Not set
    ...0 .... = Reset: Not set
    SYN .... = SYN: Set
    [Expert Info (Chat/Sequence): Connection establish request (SYN): server port 80]
    [TCP Flags: .....S.]
    Window: 16384
    [Calculated window size: 16384]
    Checksum: 0xf5e0 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
    Options: (8 bytes), Maximum segment size, No-Operation (NOP), No-Operation (NOP), SACK permitted
    [Timestamps]
  
```

- 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?

Ans: Sequence number: 0; Value of the Acknowledgement field: 1; 服务器通过接收到的 SYN 报文中 Seq + 1 得到 ACK 的值; 通过 ACK 和 SYN 均被置 1 证明这是一个 SYNACK 报文段。

Send times: 2004-08-21 21:44:20.596858,
2004-08-21 21:44:20.612118,
2004-08-21 21:44:20.624407.

2004-08-21 21:44:20.625071,
2004-08-21 21:44:20.647786,
2004-08-21 21:44:20.648538;

Receive times: 第 1 个包: 2004-08-21 21:44:20.624318,
第 2 个包: 2004-08-21 21:44:20.647675,
第 3 个包: 2004-08-21 21:44:20.694466,
第 4 个包: 2004-08-21 21:44:20.739499,
第 5 个包: 2004-08-21 21:44:20.787680,
第 6 个包: 2004-08-21 21:44:20.838183;

RTT value for each: 第 1 个包: RTT=0.02746,
第 2 个包: RTT= 0.035557,
第 3 个包: RTT=0.070059,
第 4 个包: RTT=0.114428,
第 5 个包: RTT=0.139894,
第 6 个包: RTT=0.189645;

EstimatedRTT value: 计算公式为

EstimatedRTT = 0.875*EstimatedRTT+0.125 SampleRTT

(这里假设初始 EstimatedRTT 与第一次得到的 SampleRTT 相等)

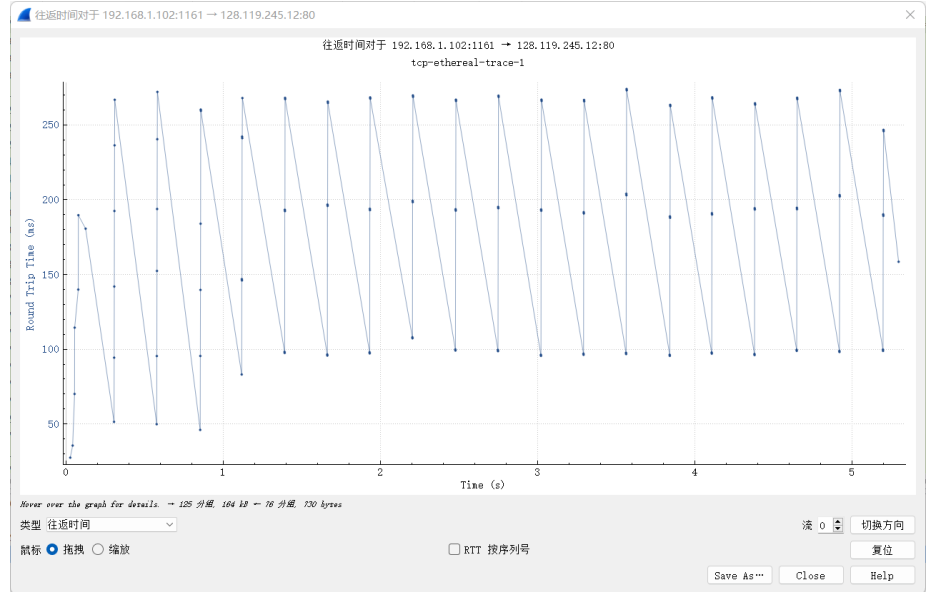
第 1 个包: EstimatedRTT =0.02746,
第 2 个包: EstimatedRTT = 0.028472125,
第 3 个包: EstimatedRTT =0.0336704844,
第 4 个包: EstimatedRTT =0.0437651738,
第 5 个包: EstimatedRTT =0.0557812771,
第 6 个包: EstimatedRTT =0.0725142425;

4	2004-08-21	21:44:20.596858	192.168.1.102	128.119.245.12	✓	TCP	619 1161 → 80 [PSH, ACK] Seq=1 Ack=1 Win=17520 Len=565
5	2004-08-21	21:44:20.612118	192.168.1.102	128.119.245.12	✓	TCP	1514 1161 → 80 [PSH, ACK] Seq=566 Ack=1 Win=17520 Len=1460
6	2004-08-21	21:44:20.624318	128.119.245.12	192.168.1.102	✓	TCP	60 80 → 1161 [ACK] Seq=1 Ack=566 Win=0 Len=0
7	2004-08-21	21:44:20.624487	192.168.1.102	128.119.245.12	✓	TCP	1514 1161 → 80 [ACK] Seq=2026 Ack=1 Win=1720 Len=1460
8	2004-08-21	21:44:20.625071	192.168.1.102	128.119.245.12	✓	TCP	1514 1161 → 80 [ACK] Seq=3486 Ack=1 Win=1720 Len=1460
9	2004-08-21	21:44:20.647675	128.119.245.12	192.168.1.102	✓	TCP	60 80 → 1161 [ACK] Seq=1 Ack=2026 Win=0 Len=0
9	2004-08-21	21:44:20.647786	192.168.1.102	128.119.245.12	✓	TCP	1514 1161 → 80 [ACK] Seq=4846 Ack=1 Win=1720 Len=1460
10	2004-08-21	21:44:20.648538	192.168.1.102	128.119.245.12	✓	TCP	1514 1161 → 80 [ACK] Seq=6086 Ack=1 Win=1720 Len=1460
11	2004-08-21	21:44:20.694466	128.119.245.12	192.168.1.102	✓	TCP	60 80 → 1161 [ACK] Seq=1 Ack=3486 Win=0 Len=0
12	2004-08-21	21:44:20.694566	192.168.1.102	128.119.245.12	✓	TCP	1201 1161 → 80 [PSH, ACK] Seq=7886 Ack=1 Win=17520 Len=1147
13	2004-08-21	21:44:20.739499	128.119.245.12	192.168.1.102	✓	TCP	60 80 → 1161 [ACK] Seq=1 Ack=4846 Win=0 Len=0
14	2004-08-21	21:44:20.787680	128.119.245.12	192.168.1.102	✓	TCP	60 80 → 1161 [ACK] Seq=1 Ack=6086 Win=1720 Len=0
15	2004-08-21	21:44:20.838183	128.119.245.12	192.168.1.102	✓	TCP	60 80 → 1161 [ACK] Seq=1 Ack=7886 Win=20440 Len=0

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

Ans: Length: 565, 1460, 1460, 1460, 1460, 1460; 58307;

4	2004-08-21	21:44:20.596858	192.168.1.102	128.119.245.12	✓	TCP	619 1161 → 80 [PSH, ACK] Seq=1 Ack=1 Win=17520 Len=565
5	2004-08-21	21:44:20.612118	192.168.1.102	128.119.245.12	✓	TCP	1514 1161 → 80 [PSH, ACK] Seq=566 Ack=1 Win=17520 Len=1460
6	2004-08-21	21:44:20.624318	128.119.245.12	192.168.1.102	✓	TCP	60 80 → 1161 [ACK] Seq=1 Ack=566 Win=0 Len=0
7	2004-08-21	21:44:20.624487	192.168.1.102	128.119.245.12	✓	TCP	1514 1161 → 80 [ACK] Seq=2026 Ack=1 Win=1720 Len=1460
8	2004-08-21	21:44:20.625071	192.168.1.102	128.119.245.12	✓	TCP	1514 1161 → 80 [ACK] Seq=3486 Ack=1 Win=1720 Len=1460
9	2004-08-21	21:44:20.647675	128.119.245.12	192.168.1.102	✓	TCP	60 80 → 1161 [ACK] Seq=1 Ack=2026 Win=0 Len=0
10	2004-08-21	21:44:20.647786	192.168.1.102	128.119.245.12	✓	TCP	1514 1161 → 80 [ACK] Seq=4846 Ack=1 Win=1720 Len=1460
11	2004-08-21	21:44:20.648538	192.168.1.102	128.119.245.12	✓	TCP	1514 1161 → 80 [ACK] Seq=6086 Ack=1 Win=1720 Len=1460
12	2004-08-21	21:44:20.694466	128.119.245.12	192.168.1.102	✓	TCP	60 80 → 1161 [ACK] Seq=1 Ack=3486 Win=0 Len=0
13	2004-08-21	21:44:20.694566	192.168.1.102	128.119.245.12	✓	TCP	1201 1161 → 80 [PSH, ACK] Seq=7886 Ack=1 Win=17520 Len=1147
14	2004-08-21	21:44:20.739499	128.119.245.12	192.168.1.102	✓	TCP	60 80 → 1161 [ACK] Seq=1 Ack=4846 Win=0 Len=0
15	2004-08-21	21:44:20.787680	128.119.245.12	192.168.1.102	✓	TCP	60 80 → 1161 [ACK] Seq=1 Ack=6086 Win=17520 Len=0
16	2004-08-21	21:44:20.838183	128.119.245.12	192.168.1.102	✓	TCP	60 80 → 1161 [ACK] Seq=1 Ack=7886 Win=20440 Len=0



(图 1-7 TCP RTT 流图形)

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

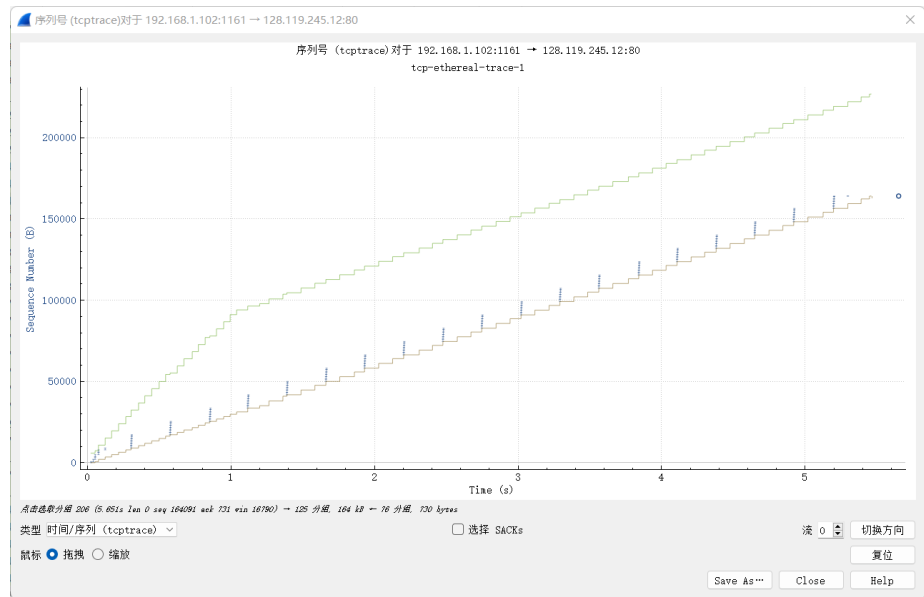
Ans: Minimum amount of available buffer: 5840; 由于 TCP 的流量控制机制，当接收窗口大小较小时发送方的传输速率会下降，但在上述过程中，接收窗口始终大于 MSS，故不会限制发送方传输速率。

1	2004-08-21 21:44:20.570381	192.168.1.102	128.119.245.12	✓	TCP	62 1161 → 80 [SYN] Seq=0 Min=16384 Len=0 MSS=1460 SACK_PERM=1
2	2004-08-21 21:44:20.593553	128.119.245.12	192.168.1.102	✓	TCP	62 80 → 1161 [SYN, ACK] Seq=0 Ack=1 Min=16384 Len=0 MSS=1460 SACK_PERM=1
3	2004-08-21 21:44:20.593646	192.168.1.102	128.119.245.12	✓	TCP	54 1161 → 80 [ACK] Seq=1 Ack=1 Min=17520 Len=0
4	2004-08-21 21:44:20.596858	192.168.1.102	128.119.245.12	✓	TCP	619 1161 → 80 [PSH, ACK] Seq=1 Ack=1 Min=17520 Len=565
5	2004-08-21 21:44:20.612118	192.168.1.102	128.119.245.12	✓	TCP	1514 1161 → 80 [PSH, ACK] Seq=566 Ack=1 Min=17520 Len=1460
6	2004-08-21 21:44:20.624318	128.119.245.12	192.168.1.102	✓	TCP	60 80 → 1161 [ACK] Seq=1 Ack=566 Min=6780 Len=0
7	2004-08-21 21:44:20.624407	192.168.1.102	128.119.245.12	✓	TCP	1514 1161 → 80 [ACK] Seq=2026 Ack=1 Min=17520 Len=1460
8	2004-08-21 21:44:20.625071	192.168.1.102	128.119.245.12	✓	TCP	1514 1161 → 80 [ACK] Seq=3486 Ack=1 Min=17520 Len=1460
9	2004-08-21 21:44:20.647675	128.119.245.12	192.168.1.102	✓	TCP	60 80 → 1161 [ACK] Seq=1 Ack=2026 Min=8760 Len=0
10	2004-08-21 21:44:20.647786	192.168.1.102	128.119.245.12	✓	TCP	1514 1161 → 80 [ACK] Seq=4946 Ack=1 Min=17520 Len=1460
11	2004-08-21 21:44:20.648538	192.168.1.102	128.119.245.12	✓	TCP	1514 1161 → 80 [ACK] Seq=6486 Ack=1 Min=17520 Len=1460
12	2004-08-21 21:44:20.694466	128.119.245.12	192.168.1.102	✓	TCP	60 80 → 1161 [ACK] Seq=1 Ack=3486 Min=11680 Len=0

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

Ans: 可以观察到 TCP 流中相应的序列号一直增大，并未出现重复的情况，因此可知并没有出现重传。同样也可以利用 TCP 序列号流图形得到上述结论。

No.	Time	Source	Destination	Frame	Protocol	Length	Info
1	2004-08-21 21:44:20.570381	192.168.1.102	128.119.245.12	✓	TCP	62	62 1161 → 80 [SYN] Seq=0 Min=16384 Len=0 MSS=1460 SACK_PERM=1
2	2004-08-21 21:44:20.593553	128.119.245.12	192.168.1.102	✓	TCP	62	62 80 → 1161 [SYN, ACK] Seq=0 Ack=1 Min=16384 Len=0 MSS=1460 SACK_PERM=1
3	2004-08-21 21:44:20.593646	192.168.1.102	128.119.245.12	✓	TCP	54	54 1161 → 80 [ACK] Seq=1 Ack=1 Min=17520 Len=0
4	2004-08-21 21:44:20.596858	192.168.1.102	128.119.245.12	✓	TCP	619	619 1161 → 80 [PSH, ACK] Seq=1 Ack=1 Min=17520 Len=565
5	2004-08-21 21:44:20.612118	192.168.1.102	128.119.245.12	✓	TCP	1514	1514 1161 → 80 [PSH, ACK] Seq=566 Ack=1 Min=17520 Len=1460
6	2004-08-21 21:44:20.624318	128.119.245.12	192.168.1.102	✓	TCP	60	60 80 → 1161 [ACK] Seq=1 Ack=566 Min=6780 Len=0
7	2004-08-21 21:44:20.624407	192.168.1.102	128.119.245.12	✓	TCP	1514	1514 1161 → 80 [ACK] Seq=2026 Ack=1 Min=17520 Len=1460
8	2004-08-21 21:44:20.625071	192.168.1.102	128.119.245.12	✓	TCP	1514	1514 1161 → 80 [ACK] Seq=3486 Ack=1 Min=17520 Len=1460
9	2004-08-21 21:44:20.647675	128.119.245.12	192.168.1.102	✓	TCP	60	60 80 → 1161 [ACK] Seq=1 Ack=2026 Min=8760 Len=0
10	2004-08-21 21:44:20.647786	192.168.1.102	128.119.245.12	✓	TCP	1514	1514 1161 → 80 [ACK] Seq=4946 Ack=1 Min=17520 Len=1460
11	2004-08-21 21:44:20.648538	192.168.1.102	128.119.245.12	✓	TCP	1514	1514 1161 → 80 [ACK] Seq=6486 Ack=1 Min=17520 Len=1460
12	2004-08-21 21:44:20.694466	128.119.245.12	192.168.1.102	✓	TCP	60	60 80 → 1161 [ACK] Seq=1 Ack=3486 Min=11680 Len=0
13	2004-08-21 21:44:20.694566	192.168.1.102	128.119.245.12	✓	TCP	1201	1201 1161 → 80 [PSH, ACK] Seq=7866 Ack=1 Min=17520 Len=1147
14	2004-08-21 21:44:20.730499	128.119.245.12	192.168.1.102	✓	TCP	60	60 80 → 1161 [ACK] Seq=1 Ack=4946 Min=14600 Len=0
15	2004-08-21 21:44:20.787080	128.119.245.12	192.168.1.102	✓	TCP	60	60 80 → 1161 [ACK] Seq=1 Ack=6486 Min=17520 Len=0
16	2004-08-21 21:44:20.838183	128.119.245.12	192.168.1.102	✓	TCP	60	60 80 → 1161 [ACK] Seq=1 Ack=7866 Min=20440 Len=0
17	2004-08-21 21:44:20.875188	128.119.245.12	192.168.1.102	✓	TCP	60	60 80 → 1161 [ACK] Seq=1 Ack=9013 Min=23360 Len=0
18	2004-08-21 21:44:20.875421	192.168.1.102	128.119.245.12	✓	TCP	1514	1514 1161 → 80 [ACK] Seq=9013 Ack=1 Min=17520 Len=1460
19	2004-08-21 21:44:20.876184	192.168.1.102	128.119.245.12	✓	TCP	1514	1514 1161 → 80 [ACK] Seq=10873 Ack=1 Min=17520 Len=1460
20	2004-08-21 21:44:20.877073	192.168.1.102	128.119.245.12	✓	TCP	1514	1514 1161 → 80 [ACK] Seq=11933 Ack=1 Min=17520 Len=1460
21	2004-08-21 21:44:20.877952	192.168.1.102	128.119.245.12	✓	TCP	1514	1514 1161 → 80 [ACK] Seq=13393 Ack=1 Min=17520 Len=1460
22	2004-08-21 21:44:20.879080	192.168.1.102	128.119.245.12	✓	TCP	1514	1514 1161 → 80 [ACK] Seq=14853 Ack=1 Min=17520 Len=1460
23	2004-08-21 21:44:20.879934	192.168.1.102	128.119.245.12	✓	TCP	946	946 1161 → 80 [PSH, ACK] Seq=16313 Ack=1 Min=17520 Len=892
24	2004-08-21 21:44:20.926818	128.119.245.12	192.168.1.102	✓	TCP	60	60 80 → 1161 [ACK] Seq=1 Ack=18043 Min=26280 Len=0
25	2004-08-21 21:44:20.970545	128.119.245.12	192.168.1.102	✓	TCP	60	60 80 → 1161 [ACK] Seq=1 Ack=11193 Min=29200 Len=0
26	2004-08-21 21:44:21.010994	128.119.245.12	192.168.1.102	✓	TCP	60	60 80 → 1161 [ACK] Seq=1 Ack=12333 Min=32120 Len=0
27	2004-08-21 21:44:21.070410	128.119.245.12	192.168.1.102	✓	TCP	60	60 80 → 1161 [ACK] Seq=1 Ack=13443 Min=35040 Len=0
28	2004-08-21 21:44:21.115433	128.119.245.12	192.168.1.102	✓	TCP	60	60 80 → 1161 [ACK] Seq=1 Ack=16523 Min=37960 Len=0
29	2004-08-21 21:44:21.146798	128.119.245.12	192.168.1.102	✓	TCP	60	60 80 → 1161 [ACK] Seq=1 Ack=17725 Min=37960 Len=0
30	2004-08-21 21:44:21.147052	192.168.1.102	128.119.245.12	✓	TCP	1514	1514 1161 → 80 [ACK] Seq=17205 Ack=1 Min=17520 Len=1460
31	2004-08-21 21:44:21.147766	192.168.1.102	128.119.245.12	✓	TCP	1514	1514 1161 → 80 [ACK] Seq=18665 Ack=1 Min=17520 Len=1460
32	2004-08-21 21:44:21.148710	192.168.1.102	128.119.245.12	✓	TCP	1514	1514 1161 → 80 [ACK] Seq=20125 Ack=1 Min=17520 Len=1460
33	2004-08-21 21:44:21.149576	192.168.1.102	128.119.245.12	✓	TCP	1514	1514 1161 → 80 [ACK] Seq=21585 Ack=1 Min=17520 Len=1460



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)

Ans: 大部分确认为 1460 bytes; 如第 52 个报文段就是间隔确认的。

46	2004-08-21	21:44:21.427183	192.168.1.102	128.119.245.12	✓	TCP	1514	1161	→ 80	[ACK Seq=31237 Ack=1 Win=17520 Len=1460
47	2004-08-21	21:44:21.428064	192.168.1.102	128.119.245.12	✓	TCP	946	1161	→ 80	[PSH Seq=31260 Ack=1 Win=17520 Len=892
48	2004-08-21	21:44:21.469804	128.119.245.12	192.168.1.102	✓	TCP	60	80	→ 1161	[ACK Seq=1 Ack=26857 Win=54880 Len=0
49	2004-08-21	21:44:21.519926	128.119.245.12	192.168.1.102	✓	TCP	60	80	→ 1161	[ACK Seq=1 Ack=26817 Win=54880 Len=0
50	2004-08-21	21:44:21.565096	128.119.245.12	192.168.1.102	✓	TCP	60	80	→ 1161	[ACK Seq=1 Ack=29277 Win=13320 Len=0
51	2004-08-21	21:44:21.618201	128.119.245.12	192.168.1.102	✓	TCP	60	80	→ 1161	[ACK Seq=1 Ack=31237 Win=62780 Len=0
52	2004-08-21	21:44:21.687478	128.119.245.12	192.168.1.102	✓	TCP	60	80	→ 1161	[ACK Seq=1 Ack=33589 Win=62780 Len=0

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

Ans: 传输数据总量为 164091 bytes, 传输总时间为 T= 26.026211-20.596858=5.429353s。Throughput = $\frac{164091 \text{ bytes}}{5.429353 \text{ s}} \approx 30222.94 \text{ Byte/s}$ 。也可以利用图表直接得出。

1

2004-08-21

21:44:20.570381

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

2004-08-21

21:44:20.593553

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

2004-08-21

21:44:20.593646

192.168.1.102

128.119.245.12

✓

TCP

54

1161

→ 80

[ACK Seq=1 Ack=1 Win=17520 Len=0

4

2004-08-21

21:44:20.596858

192.168.1.102

128.119.245.12

✓

TCP

619

1161

→ 80

[PSH, ACK Seq=1 Ack=1 Win=17520 Len=565

5

2004-08-21

21:44:20.612418

192.168.1.102

128.119.245.12

✓

TCP

1514

1161

→ 80

[PSH, ACK Seq=1 Ack=566 Win=17520 Len=1460

6

2004-08-21

21:44:20.624318

128.119.245.12

192.168.1.102

✓

TCP

60

80

→ 1161

[ACK Seq=1 Ack=566 Win=6780 Len=0

⚙

⏮

⏪

⏩

⏭

⏴

⏵

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

🔍

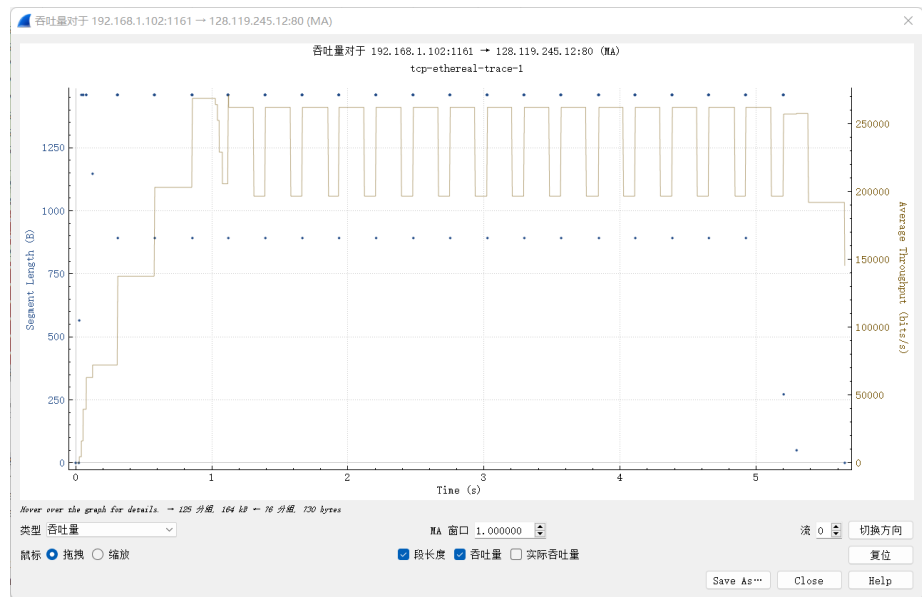
🔍

🔍

🔍

🔍

🔍



(图 1-8 TCP 吞吐量流图形)

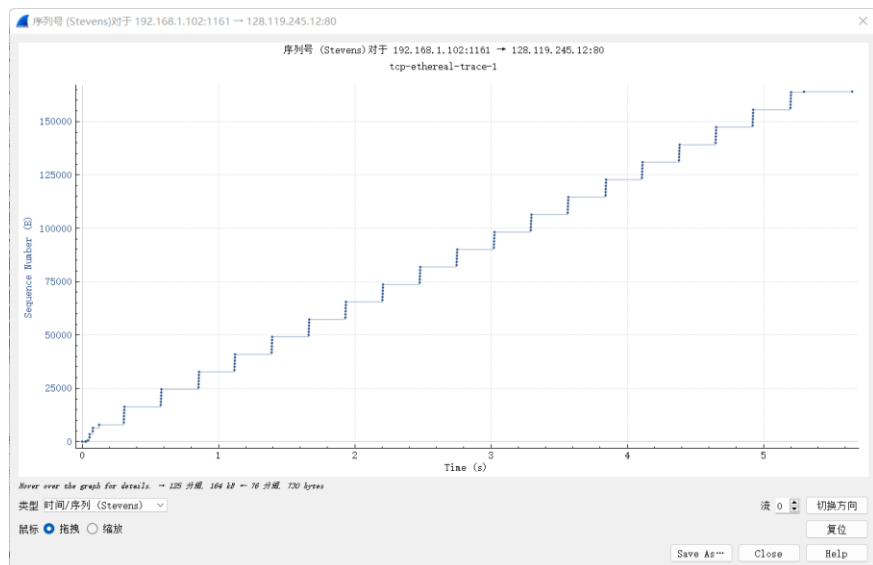
Part 4: TCP congestion control in action

1. 实验内容:

分析 TCP 数据包, 观察 TCP 拥塞控制机制。

2. 实验流程:

利用 Wireshark 制图, 得到如图所示时间序列:

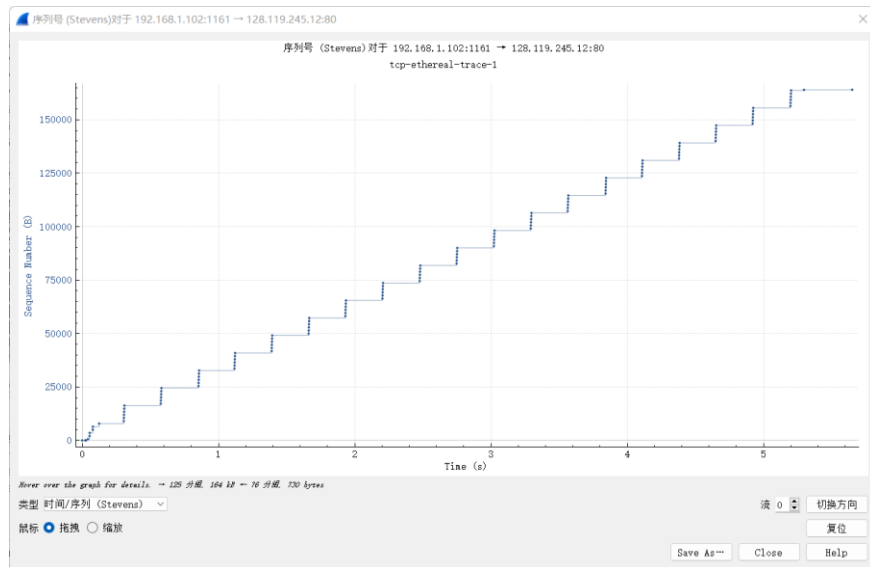


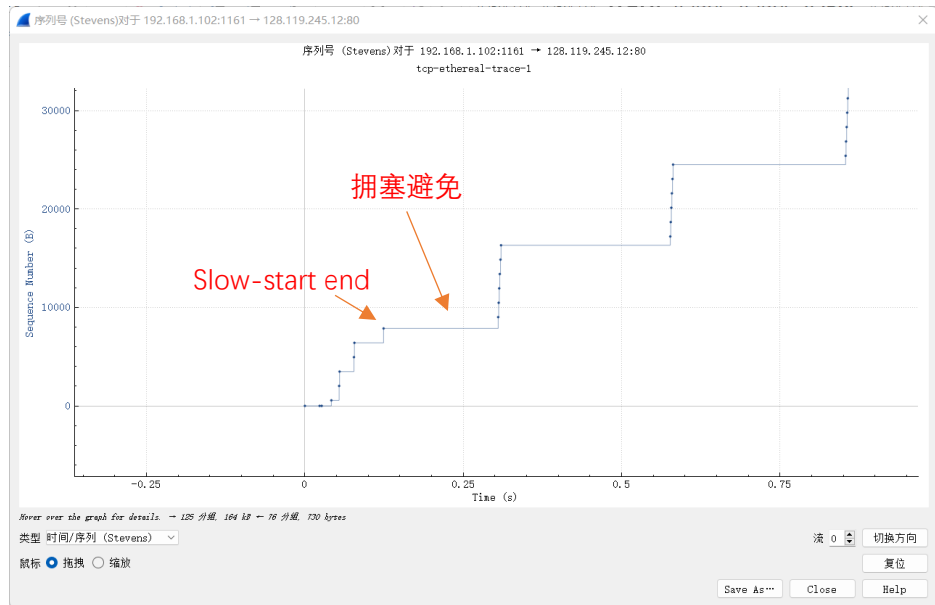
(图 1-9 Stevens TCP 流图形)

3. Q&A(以下内容利用所给 trace 分析):

- 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?

Ans: 慢启动从 TCP 连接建立完成后发送第一个分组（分组 5）开始，在第 13 个分组之后不再呈指数增长，慢启动结束，拥塞避免开始。观察可知在后续传输过程中每次均发送六个报文段，说明 cwnd 遇到了 ssthresh，进入 congestion avoidance 阶段，cwnd 不再增加，但是书上写得理想机制是会线性增加直到遇到超时，可能这里存在不同的控制机制。此外，在慢启动阶段 cwnd 并不是严格按照指数增长的，与课本上的理想情况存在一定的差别。





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

Ans: 传输数据总量为164091bytes, 传输总时间为 $T = 29.536794 - 25.799772 = 3.737022s$ 。Throughput = $\frac{153049bytes}{3.737022s} \approx 40954.80Byte/s$ 。也可以利用图表直接得出。

23	2022-10-24	00:17:25.798872	192.168.43.252	128.119.245.12	✓	TCP	54 64982 → 80 [ACK] Seq=1 Ack=1 Win=66560 Len=0
24	2022-10-24	00:17:25.798957	192.168.43.252	128.119.245.12	✓	TCP	54 58399 → 80 [ACK] Seq=1 Ack=1 Win=66560 Len=0
25	2022-10-24	00:17:25.798920	192.168.43.252	128.119.245.12	✓	TCP	54 58307 → 80 [ACK] Seq=1 Ack=1 Win=66560 Len=0
26	2022-10-24	00:17:25.799772	192.168.43.252	128.119.245.12	✓	TCP	781 58307 → 80 [PSH, ACK] Seq=1 Ack=1 Win=66560 Len=727 [TCP segment of a reassembled P
27	2022-10-24	00:17:25.800095	192.168.43.252	128.119.245.12	✓	TCP	12294 58307 → 80 [ACK] Seq=728 Ack=1 Win=66560 Len=12240 [TCP segment of a reassembled PC
28	2022-10-24	00:17:26.132925	192.168.43.252	183.2.143.108	✓	TCP	55 [TCP Keep-Alive] 61603 → 443 [ACK] Seq=1 Ack=1 Win=258 Len=1
29	2022-10-24	00:17:26.227174	128.119.245.12	192.168.43.252	✓	TCP	66 [TCP Window update] 80 → 58307 [ACK] Seq=1 Ack=1 Win=32128 Len=0 SLE=728 SRE=2088
30	2022-10-24	00:17:26.227174	128.119.245.12	192.168.43.252	✓	TCP	54 800 → 58307 [ACK] Seq=1 Ack=2088 Win=3664 Len=0

No.	Time	Source	Destination	Frame	Protocol	Length	Info
169	2022-10-24 00:17:28.987585	128.119.245.12	192.168.43.252	✓	TCP	54	800 → 58307 [ACK] Seq=1 Ack=130088 Win=269824 Len=0
170	2022-10-24 00:17:29.807838	128.119.245.12	192.168.43.252	✓	TCP	54	800 → 58307 [ACK] Seq=1 Ack=130448 Win=269824 Len=0
171	2022-10-24 00:17:29.879843	128.119.245.12	192.168.43.252	✓	TCP	54	800 → 58307 [ACK] Seq=1 Ack=140080 Win=269824 Len=0
172	2022-10-24 00:17:29.879596	128.119.245.12	192.168.43.252	✓	TCP	54	800 → 58307 [ACK] Seq=1 Ack=142168 Win=269824 Len=0
173	2022-10-24 00:17:29.110795	192.168.43.252	183.2.143.108	✓	TCP	55	[TCP Keep-Alive] 61603 → 443 [ACK] Seq=1 Ack=1 Win=258 Len=1
174	2022-10-24 00:17:29.138846	128.119.245.12	192.168.43.252	✓	TCP	54	800 → 58307 [ACK] Seq=1 Ack=145296 Win=269824 Len=0
175	2022-10-24 00:17:29.139864	128.119.245.12	192.168.43.252	✓	TCP	54	800 → 58307 [ACK] Seq=1 Ack=146880 Win=269824 Len=0
176	2022-10-24 00:17:29.139426	128.119.245.12	192.168.43.252	✓	TCP	54	800 → 58307 [ACK] Seq=1 Ack=146248 Win=269824 Len=0
177	2022-10-24 00:17:29.162674	183.2.143.108	192.168.43.252	✓	TCP	66	[TCP Keep-Alive ACK] 443 → 61603 [ACK] Seq=1 Ack=2 Win=260 Len=0 SLE=1 SRE=2
178	2022-10-24 00:17:29.166099	128.119.245.12	192.168.43.252	✓	TCP	54	800 → 58307 [ACK] Seq=1 Ack=147680 Win=269824 Len=0
179	2022-10-24 00:17:29.234730	128.119.245.12	192.168.43.252	✓	TCP	54	800 → 58307 [ACK] Seq=1 Ack=148320 Win=269824 Len=0
180	2022-10-24 00:17:29.234927	128.119.245.12	192.168.43.252	✓	TCP	54	800 → 58307 [ACK] Seq=1 Ack=149680 Win=269824 Len=0
181	2022-10-24 00:17:29.536794	128.119.245.12	192.168.43.252	✓	TCP	54	800 → 58307 [ACK] Seq=1 Ack=152480 Win=269824 Len=0
182	2022-10-24 00:17:29.536794	128.119.245.12	192.168.43.252	✓	TCP	66	800 → 58307 [ACK] Seq=1 Ack=146680 Win=269824 Len=0 SLE=151049 SRE=152489
183	2022-10-24 00:17:29.536794	128.119.245.12	192.168.43.252	✓	HTTP	831	HTTP/1.1 200 OK (text/html)
184	2022-10-24 00:17:29.536794	128.119.245.12	192.168.43.252	✓	TCP	54	800 → 58307 [ACK] Seq=1 Ack=153049 Win=269824 Len=0
185	2022-10-24 00:17:29.536862	192.168.43.252	128.119.245.12	✓	TCP	54	58307 → 80 [ACK] Seq=130488 Ack=778 Win=65792 Len=0
186	2022-10-24 00:17:30.166395	192.168.43.252	183.2.143.108	✓	TCP	55	[TCP Keep-Alive] 61603 → 443 [ACK] Seq=1 Ack=1 Win=258 Len=1
187	2022-10-24 00:17:30.232284	183.2.143.108	192.168.43.252	✓	TCP	66	[TCP Keep-Alive ACK] 443 → 61603 [ACK] Seq=1 Ack=2 Win=260 Len=0 SLE=1 SRE=2
188	2022-10-24 00:17:30.594721	192.168.43.252	36.155.201.42	✓	TCP	108	61437 → 8080 [PSH, ACK] Seq=1 Ack=1 Win=257 Len=54
189	2022-10-24 00:17:30.815839	36.155.201.42	192.168.43.252	✓	TCP	124	8080 → 61437 [PSH, ACK] Seq=1 Ack=55 Win=126 Len=70
190	2022-10-24 00:17:30.816133	117.18.237.29	192.168.43.252	✓	TCP	54	800 → 61591 [ACK] Seq=1 Ack=1 Win=133 Len=0
191	2022-10-24 00:17:30.816133	117.18.237.29	192.168.43.252	✓	TCP	54	[TCP ACK] Seq=1 Ack=1 Win=257 Len=0
192	2022-10-24 00:17:30.864176	192.168.43.252	36.155.201.42	✓	TCP	54	61437 → 8080 [ACK] Seq=55 Ack=71 Win=257 Len=0
193	2022-10-24 00:17:31.242990	192.168.43.252	183.2.143.108	✓	TCP	55	[TCP Keep-Alive] 61603 → 443 [ACK] Seq=1 Ack=1 Win=258 Len=1
194	2022-10-24 00:17:31.302237	183.2.143.108	192.168.43.252	✓	TCP	66	[TCP Keep-Alive ACK] 443 → 61603 [ACK] Seq=1 Ack=2 Win=260 Len=0 SLE=1 SRE=2

Frame 184: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 'Device\NPF_{0AE83349-5702-4420-9C46-E61E4E683EF}', id 0

Ethernet II, Src: f2:a1:d0:aa:bb:74 (f2:a1:d0:aa:bb:74), Dst: IntelCor_ef:93:5e (e0:d4:e0:ef:93:5e)

Internet Protocol Version 4, Src: 128.119.245.12, Dst: 192.168.43.252

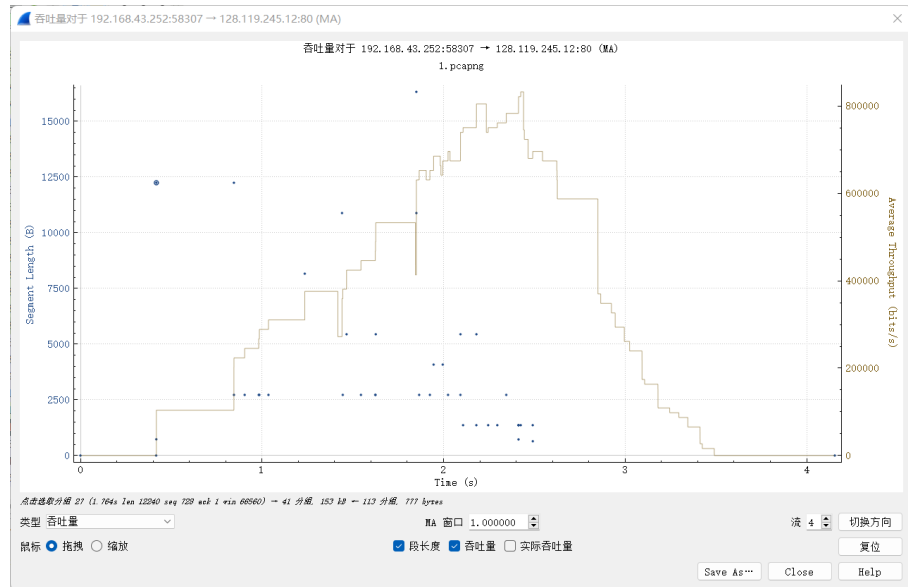
Transmission Control Protocol, Src Port: 80, Dst Port: 58307, Seq: 1, Ack: 153049, Len: 0

0000 e0 d4 e0 ef 93 5e f2 a1 d0 aa b0 74 00 00 45 48t...EH

Transmission Control Protocol: Reset

开始: 195 / 已显示: 191 (97.90)

配置: Default



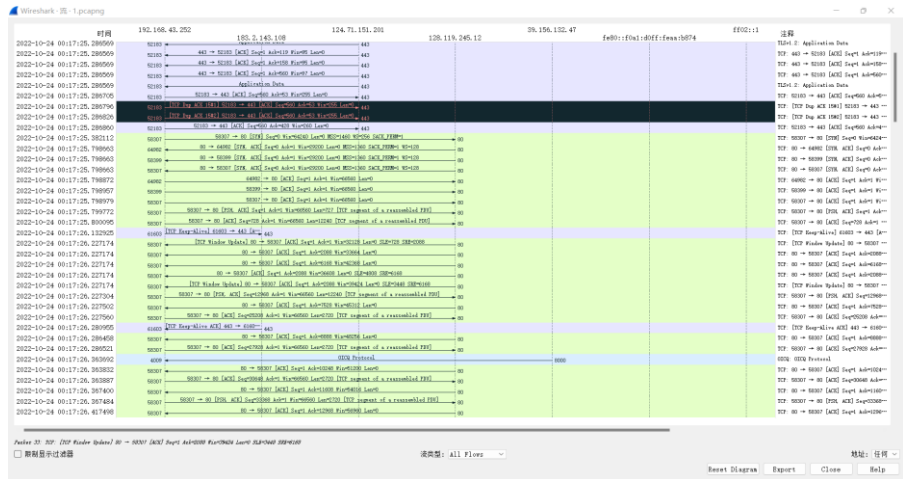
(图 1-10 TCP 吞吐量流图形)

Ans: 慢启动从 TCP 连接建立完成后发送第一个分组 (分组 27) 开始, 在第 36 个分组之后不再呈指数增长, 慢启动结束, 拥塞避免开始。可以比所给 trace 看到更为明显的拥塞控制过程。



补充内容:

利用 Wireshark 我们也可以看到 TCP 流量图, 以实验中获得数据为例, 可以得到如下流量图:



(图 1-11 TCP 流量图)

除了三次握手，在结束连接时 TCP 也具有四次挥手过程：

由于 TCP 连接是全双工的，因此，每个方向都必须单独进行关闭。在一方完成数据发送任务后，将会发送一个 FIN 来终止本方连接，但是在这个 TCP 连接上仍然能够发送数据，直到接收方也发送了 FIN。首先进行关闭的一方将执行主动关闭，而另一方则执行被动关闭。

(1) 第一次挥手：客户端发送一个 FIN，用来关闭客户端到服务器的数据传送，客户端进入 FIN_WAIT_1 状态。

(2) 第二次挥手：服务器收到 FIN 后，发送一个 ACK 给客户端，确认序号为收到序号+1，服务器进入 CLOSE_WAIT 状态。

(3) 第三次挥手：服务器发送一个 FIN，用来关闭服务器到客户端的数据传送，服务器进入 LAST_ACK 状态。

(4) 第四次挥手：客户端收入 FIN 后，客户端进入 TIME_WAIT 状态，接着发送一个 ACK 给服务器，确认序号为收到序号+ 1，服务器进入 CLOSED 状态，完成四次挥手。