

# A first look at the captured trace

实验步骤：用wireshark打开压缩包里的tcp-ethereal-trace-1即可。

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).

IP地址：192.168.1.102 端口号：1161

8	2004-08-21	21:44:20.625071	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] S
9	2004-08-21	21:44:20.647675	128.119.245.12	192.168.1.102	TCP	60 80 → 1161 [ACK] S
10	2004-08-21	21:44:20.647786	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] S
11	2004-08-21	21:44:20.648538	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] S
12	2004-08-21	21:44:20.694466	128.119.245.12	192.168.1.102	TCP	60 80 → 1161 [ACK] S

> 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

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?

IP地址：128.119.245.12 端口号：80

10	2004-08-21	21:44:20.647786	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] Seq
11	2004-08-21	21:44:20.648538	192.168.1.102	128.119.245.12	TCP	1514 1161 → 80 [ACK] Seq
12	2004-08-21	21:44:20.694466	128.119.245.12	192.168.1.102	TCP	60 80 → 1161 [ACK] Seq

> 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

If you have been able to create your own trace, answer the following question:

实验步骤：在网站<http://gaia.cs.umass.edu/wireshark-labs/alice.txt>上下载alice.txt，开始wireshark的捕获，再在<http://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html>上传文件alice.txt，停止捕获。

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?

IP地址：172.20.10.5 端口号：55462

20	2021-10-26 18:54:00.506317	172.20.10.5	112.34.111.235	TCP	54 55426 → 443 [RST, ACK]
21	2021-10-26 18:54:00.506798	172.20.10.5	128.119.245.12	TCP	66 55462 → 80 [SYN] Seq=0
22	2021-10-26 18:54:00.507177	172.20.10.5	128.119.245.12	TCP	66 55463 → 80 [SYN] Seq=0
29	2021-10-26 18:54:00.633092	172.20.10.5	137.116.139.120	TCP	66 55464 → 443 [SYN] Seq=0
30	2021-10-26 18:54:00.740616	137.116.139.120	172.20.10.5	TCP	66 443 → 55464 [SYN, ACK]

<

> Frame 21: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF\_{9FFF19E3-A853-41EB...}

> Ethernet II, Src: LiteonTe\_da:20:a1 (74:4c:a1:da:20:a1), Dst: b6:85:e1:05:57:64 (b6:85:e1:05:57:64)

> Internet Protocol Version 4, Src: 172.20.10.5, Dst: 128.119.245.12

> Transmission Control Protocol, Src Port: 55462, Dst Port: 80, Seq: 0, Len: 0

## TCP Basics

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?

序列号: 0      SYN段被设为1

<
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)
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
[TCP Flags: .....S.]
Window: 16384

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?

序列号: 0      确认号: 1, 此处的确认号就是客户的初始序号加1。

SYN段被设为1, 这将该报文段标识为SYNACK报文段

```

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)
v Flags: 0x012 (SYN, ACK)
  000. .... = Reserved: Not set
  ...0 .... = Nonce: Not set
  .... 0... = Congestion Window Reduced (CWR): Not set
  .... .0.. = ECN-Echo: Not set
  .... ..0. = Urgent: Not set
  .... ...1 = Acknowledgment: Set
  .... .... 0... = Push: Not set
  .... .... .0.. = Reset: Not set
  > .... .... ..1. = Syn: Set
  .... .... ...0 = Fin: Not set
  [TCP Flags: .....A..S.]
Window: 5840

```

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.

序列号: 1

```

[Stream index: 0]
[Conversation completeness: Incomplete, DATA (15)]
[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)

```

0000	00 06 25 da af 73 00 20	e0 8a 70 1a 08 00 45 00	..%.s. .p..E.
0010	02 5d 1e 21 40 00 80 06	a2 e7 c0 a8 01 66 80 77	.]!@... .f.w
0020	f5 0c 04 89 00 50 0d d6	01 f5 34 a2 74 1a 50 18	....P.. .4.t.P.
0030	44 70 1f bd 00 00 50 4f	53 54 20 2f 65 74 68 65	Dp...PO ST/ethe
0040	72 65 61 6c 2d 6c 61 62	73 2f 6c 61 62 33 2d 31	real-lab s/lab3-1
0050	2d 72 65 70 6c 79 2e 68	74 6d 20 48 54 54 50 2f	-reply.h tm HTTP/
0060	31 2e 31 0d 0a 48 6f 73	74 3a 20 67 61 69 61 2e	1.1.Hos t: gaia.
0070	63 73 2e 75 6d 61 73 73	2e 65 64 75 0d 0a 55 73	cs.umass .edu..Us

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)? 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.

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*.

segment	sequence numbers	sent time	received time	RTT	EstimatedRTT
segment1	1	0.596858	0.624318	0.027460	0.027460
segment2	566	0.612118	0.647675	0.035557	0.028472125
segment3	2026	0.624407	0.694466	0.070059	0.033670484
segment4	3486	0.625071	0.739499	0.114428	0.043765174
segment5	4946	0.647786	0.787680	0.139894	0.055781277
segment6	6406	0.648538	0.838183	0.189645	0.072514242

EstimatedRTT1 = 0.027460

EstimatedRTT2 =  $0.875 * 0.027460 + 0.125 * 0.035557 = 0.028472125$

EstimatedRTT2 =  $0.875 * 0.028472125 + 0.125 * 0.070059 = 0.033670484$

EstimatedRTT2 =  $0.875 * 0.033670484 + 0.125 * 0.114428 = 0.043765174$

EstimatedRTT2 =  $0.875 * 0.043765174 + 0.125 * 0.139894 = 0.055781277$

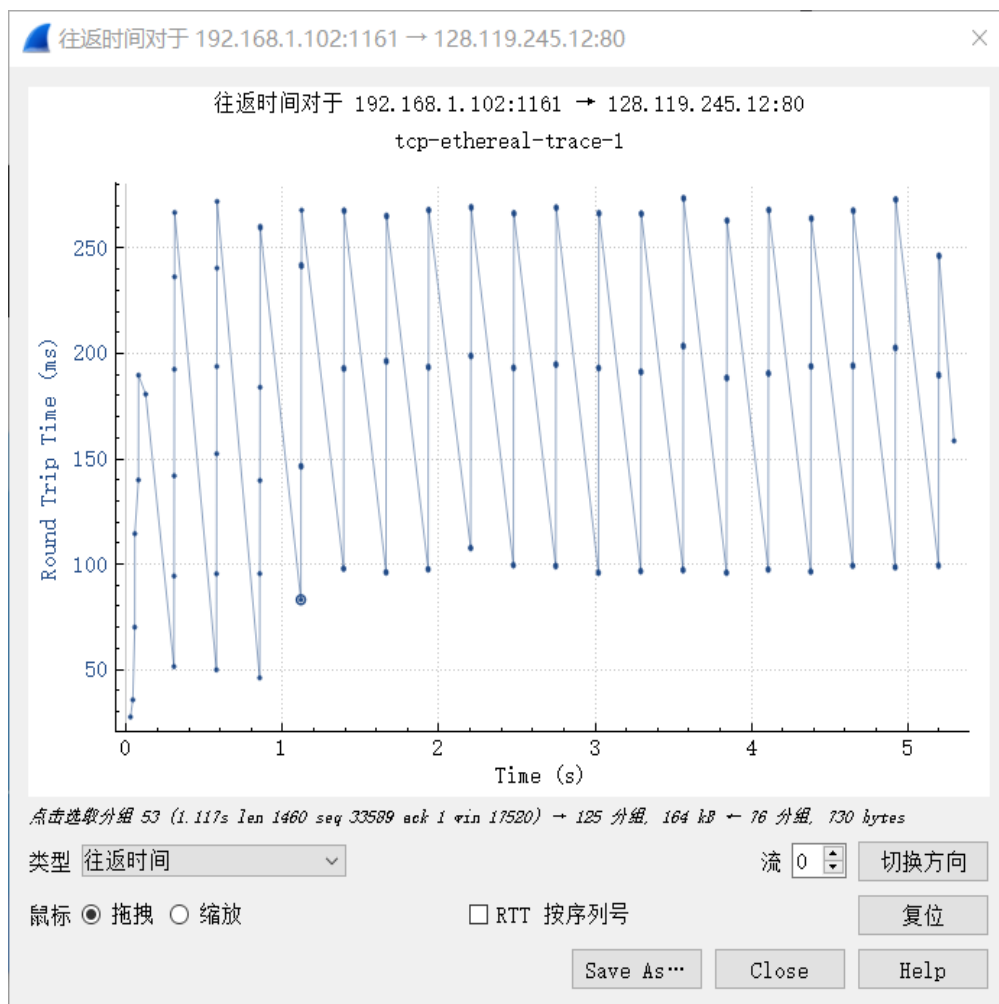
EstimatedRTT2 =  $0.875 * 0.055781277 + 0.125 * 0.189645 = 0.072514242$

从客户发向服务器的报文段：

Time	Source	Destination	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=
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
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 [TCP s
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 [TCP
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
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 [TCP se
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 [TCP se
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 [TCP se
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 [TCP se
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 [T
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 21:44:20.787680	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=6406 Win=17520 Len=0

从服务器发向客户的报文段：

No.	Time	Source	Destination	Protocol	Length	Info
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
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=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 [TCP
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 [TCP
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 [TCP
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 [TCP
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 21:44:20.787680	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=20440 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 [TCP
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 [TCP
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 [TCP



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

分别为: 565、1460、1460、1460、1460、1460

Protocol	Length	Info
TCP	62	1161 → 80 [SYN] Seq=0 Win=16384 Len=0 MSS=1460 SACK_PERM=1
TCP	62	80 → 1161 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460 SACK_PERM=1
TCP	54	1161 → 80 [ACK] Seq=1 Ack=1 Win=17520 Len=0
TCP	619	1161 → 80 [PSH, ACK] Seq=1 Ack=1 Win=17520 Len=565 [TCP segment of a reassembled P...
TCP	1514	1161 → 80 [PSH, ACK] Seq=566 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled P...
TCP	60	80 → 1161 [ACK] Seq=1 Ack=566 Win=6780 Len=0
TCP	1514	1161 → 80 [ACK] Seq=2026 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled P...
TCP	1514	1161 → 80 [ACK] Seq=3486 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled P...
TCP	60	80 → 1161 [ACK] Seq=1 Ack=2026 Win=8760 Len=0
TCP	1514	1161 → 80 [ACK] Seq=4946 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled P...
TCP	1514	1161 → 80 [ACK] Seq=6406 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled P...
TCP	60	80 → 1161 [ACK] Seq=1 Ack=3486 Win=11680 Len=0
TCP	1201	1161 → 80 [PSH, ACK] Seq=7866 Ack=1 Win=17520 Len=1147 [TCP segment of a reassembled P...
TCP	60	80 → 1161 [ACK] Seq=1 Ack=4946 Win=14600 Len=0
TCP	60	80 → 1161 [ACK] Seq=1 Ack=6406 Win=17520 Len=0
TCP	60	80 → 1161 [ACK] Seq=1 Ack=7866 Win=20440 Len=0
TCP	60	80 → 1161 [ACK] Seq=1 Ack=9013 Win=23360 Len=0

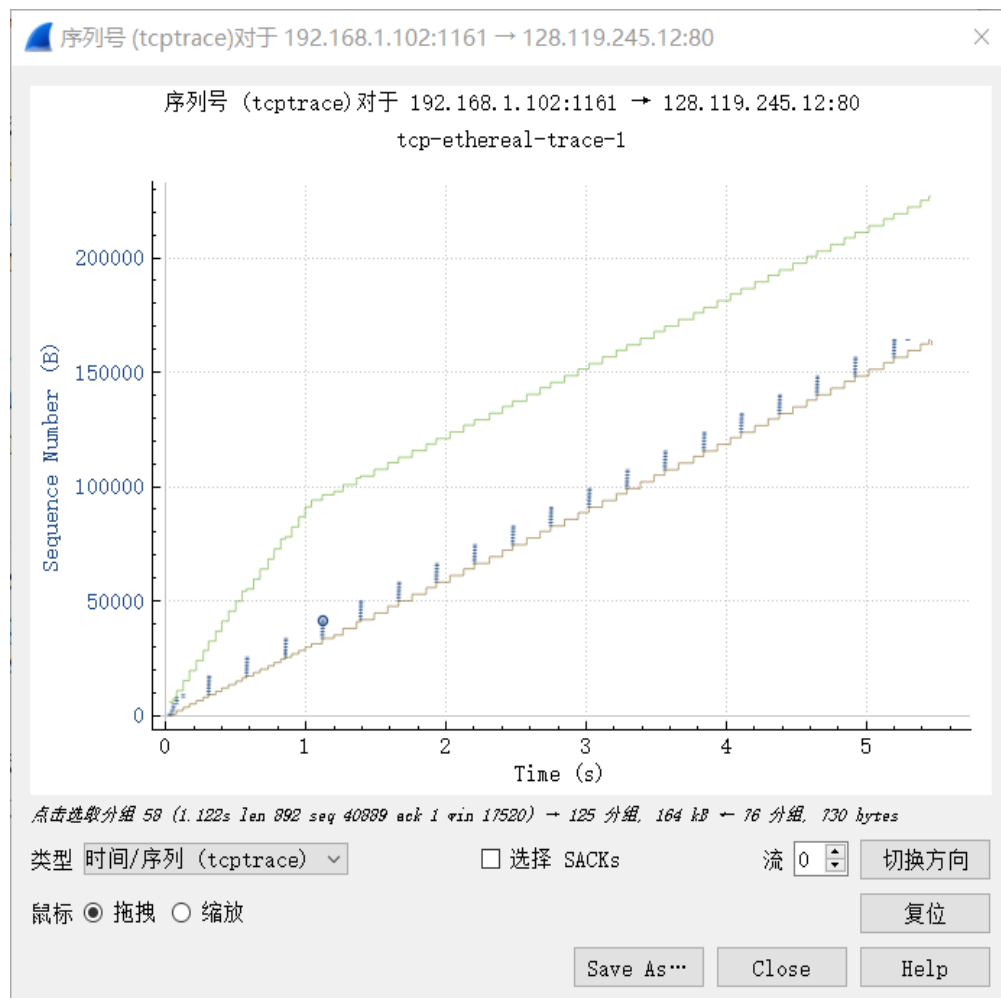
## 9. What is the minimum amount of available buffer space advertised at the received for the entire trace? Does the lack of receiver buffer space ever throttle the sender?

最小为5840，在整个窗口中并未因缓冲区不足限制发送。

th	Info
62	1161 → 80 [SYN] Seq=0 Win=16384 Len=0 MSS=1460 SACK_PERM=1
62	80 → 1161 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460 SACK_PERM=1
54	1161 → 80 [ACK] Seq=1 Ack=1 Win=17520 Len=0
19	1161 → 80 [PSH, ACK] Seq=1 Ack=1 Win=17520 Len=565 [TCP segment of
14	1161 → 80 [PSH, ACK] Seq=566 Ack=1 Win=17520 Len=1460 [TCP segment of
60	80 → 1161 [ACK] Seq=1 Ack=566 Win=6780 Len=0
14	1161 → 80 [ACK] Seq=2026 Ack=1 Win=17520 Len=1460 [TCP segment of
14	1161 → 80 [ACK] Seq=3486 Ack=1 Win=17520 Len=1460 [TCP segment of
60	80 → 1161 [ACK] Seq=1 Ack=2026 Win=8760 Len=0
14	1161 → 80 [ACK] Seq=4946 Ack=1 Win=17520 Len=1460 [TCP segment of
14	1161 → 80 [ACK] Seq=6406 Ack=1 Win=17520 Len=1460 [TCP segment of
60	80 → 1161 [ACK] Seq=1 Ack=3486 Win=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?

没有，通过不断增加的序列号可知



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).

大多数的ACK长度为1460 byte。

第80个报文段是对第76、77条报文段的累积确认



第87个报文段是对第81、82条报文段的累积确认

第88个报文段是对第83、84条报文段的累积确认

60 80 → 1161 [ACK] Seq=1 Ack=7866 Win=20440 Len=0									
60 80 → 1161 [ACK] Seq=1 Ack=9013 Win=23360 Len=0									
1514 1161 → 80 [ACK] Seq=9013 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]									
1514 1161 → 80 [ACK] Seq=10473 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]									
1514 1161 → 80 [ACK] Seq=11933 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]									
1514 1161 → 80 [ACK] Seq=13393 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]									
1514 1161 → 80 [ACK] Seq=14853 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]									
946 1161 → 80 [PSH, ACK] Seq=16313 Ack=1 Win=17520 Len=892 [TCP segment of a reassembled PDU]									
60 80 → 1161 [ACK] Seq=1 Ack=10473 Win=26280 Len=0									
75	2004-08-21	21:44:22.234579	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80	[ACK] Seq=54353 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]	
76	2004-08-21	21:44:22.235635	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80	[ACK] Seq=55813 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]	
77	2004-08-21	21:44:22.236532	192.168.1.102	128.119.245.12	TCP	946	1161 → 80	[PSH, ACK] Seq=57273 Ack=1 Win=17520 Len=892 [TCP segment of a reassembled PDU]	
78	2004-08-21	21:44:22.328608	128.119.245.12	192.168.1.102	TCP	60	80 → 1161	[ACK] Seq=1 Ack=52893 Win=62780 Len=0	
79	2004-08-21	21:44:22.430444	128.119.245.12	192.168.1.102	TCP	60	80 → 1161	[ACK] Seq=1 Ack=55813 Win=62780 Len=0	
80	2004-08-21	21:44:22.501261	128.119.245.12	192.168.1.102	TCP	60	80 → 1161	[ACK] Seq=1 Ack=58165 Win=62780 Len=0	
81	2004-08-21	21:44:22.501480	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80	[ACK] Seq=58165 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]	
82	2004-08-21	21:44:22.502260	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80	[ACK] Seq=59625 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]	
83	2004-08-21	21:44:22.503138	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80	[ACK] Seq=61085 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]	
84	2004-08-21	21:44:22.504017	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80	[ACK] Seq=62545 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]	
85	2004-08-21	21:44:22.505151	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80	[ACK] Seq=64005 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]	
86	2004-08-21	21:44:22.505967	192.168.1.102	128.119.245.12	TCP	946	1161 → 80	[PSH, ACK] Seq=65465 Ack=1 Win=17520 Len=892 [TCP segment of a reassembled PDU]	
87	2004-08-21	21:44:22.599450	128.119.245.12	192.168.1.102	TCP	60	80 → 1161	[ACK] Seq=1 Ack=61085 Win=62780 Len=0	
88	2004-08-21	21:44:22.697063	128.119.245.12	192.168.1.102	TCP	60	80 → 1161	[ACK] Seq=1 Ack=64005 Win=62780 Len=0	
89	2004-08-21	21:44:22.773576	128.119.245.12	192.168.1.102	TCP	60	80 → 1161	[ACK] Seq=1 Ack=66357 Win=62780 Len=0	
90	2004-08-21	21:44:22.773792	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80	[ACK] Seq=66357 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]	
91	2004-08-21	21:44:22.774506	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80	[ACK] Seq=67817 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]	

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

最后一个序列号减去第一个序列号再除以总时间

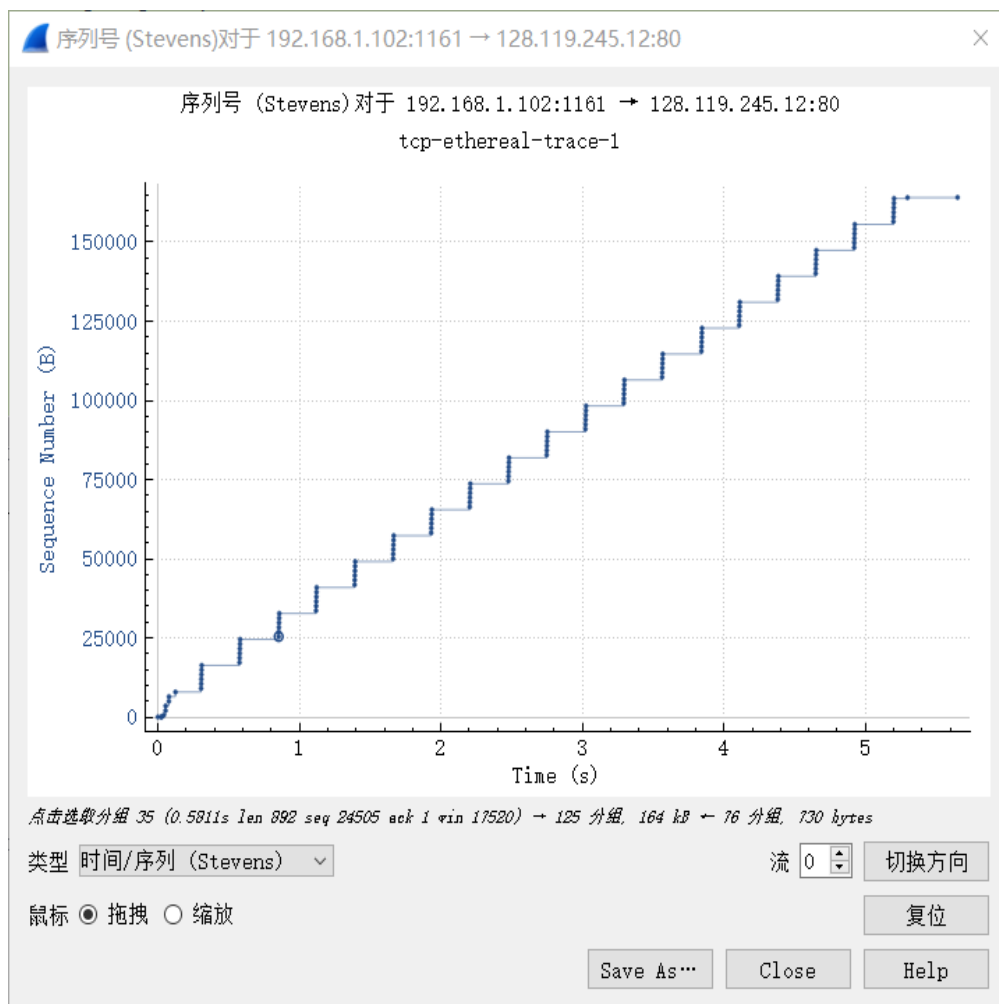
$$\text{吞吐量} = \frac{164091-1}{26.221522-20.596858} = 29173.29817 \text{ byte/sec} = 29.173 \text{ kbyte/sec}$$

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 [TCP segment of a reassembled PDU]
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 [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=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 [TCP segment of a reassembled PDU]
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 [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=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 [TCP segment of a reassembled PDU]
199	2004-08-21	21:44:25.867722	192.168.1.102	128.119.245.12	HTTP	104	POST /etherreal-labs/lab3-1-reply.htm HTTP/1.1 (text/plain)	
200	2004-08-21	21:44:25.959852	128.119.245.12	192.168.1.102	TCP	60	80 → 1161	[ACK] Seq=1 Ack=162309 Win=62780 Len=0
201	2004-08-21	21:44:26.018268	128.119.245.12	192.168.1.102	TCP	60	80 → 1161	[ACK] Seq=1 Ack=164041 Win=62780 Len=0
202	2004-08-21	21:44:26.026211	128.119.245.12	192.168.1.102	TCP	60	80 → 1161	[ACK] Seq=1 Ack=164091 Win=62780 Len=0
203	2004-08-21	21:44:26.031556	128.119.245.12	192.168.1.102	HTTP	784	HTTP/1.1 200 OK (text/html)	
204	2004-08-21	21:44:26.168471	192.168.1.100	192.168.1.1	SSDP	174	M-SEARCH * HTTP/1.1	
205	2004-08-21	21:44:26.169463	192.168.1.100	192.168.1.1	SSDP	175	M-SEARCH * HTTP/1.1	
206	2004-08-21	21:44:26.221522	192.168.1.102	128.119.245.12	TCP	54	1161 → 80	[ACK] Seq=164091 Ack=731 Win=16790 Len=0
207	2004-08-21	21:44:26.671425	192.168.1.100	192.168.1.1	SSDP	174	M-SEARCH * HTTP/1.1	
208	2004-08-21	21:44:26.672450	192.168.1.100	192.168.1.1	SSDP	175	M-SEARCH * HTTP/1.1	
209	2004-08-21	21:44:27.170533	192.168.1.100	192.168.1.1	SSDP	174	M-SEARCH * HTTP/1.1	

## 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.

在发出了HTTP POST报文段后，慢启动开始，但由于拥塞窗口大小无法直接从时间—序列图直接得到，故无法知道什么时候慢启动结束，什么时候拥塞避免开始。



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](http://gaia.cs.umass.edu)

如下为自己抓的包的时间—序列图，同样也不能看出什么时候慢启动结束，什么时候拥塞避免开始，因为拥塞窗口大小无法直接从时间—序列图直接得到。



