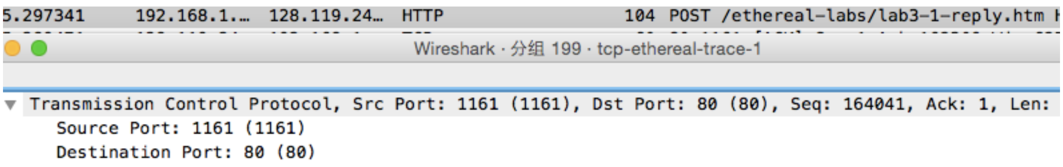
实验报告 Lab 2

姓名：王钦　学号：13349112

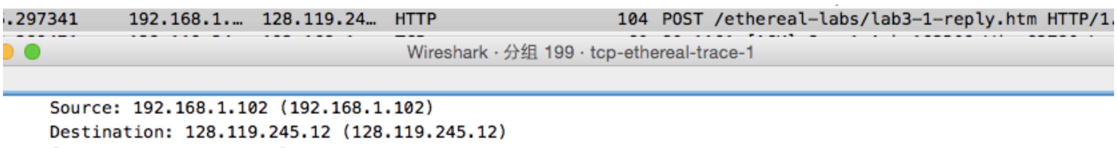
A first look at the captured trace

1. Client IP: `192.168.1.102`,Client Port: `1161`



```
5.297341    192.168.1.…  128.119.24…  HTTP    104  POST /ethereal-labs/lab3-1-reply.htm H
```

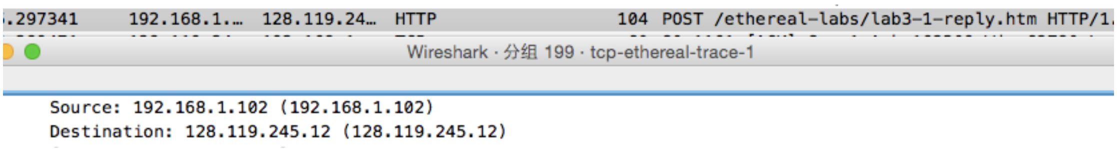
```
▼ Transmission Control Protocol, Src Port: 1161 (1161), Dst Port: 80 (80), Seq: 164041, Ack: 1, Len:
    Source Port: 1161 (1161)
    Destination Port: 80 (80)
```



```
.297341    192.168.1.…  128.119.24…  HTTP    104  POST /ethereal-labs/lab3-1-reply.htm HTTP/1.
```

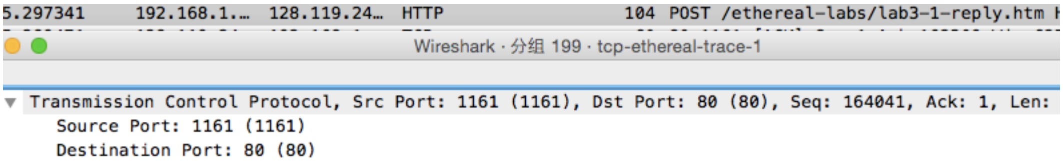
```
    Source: 192.168.1.102 (192.168.1.102)
    Destination: 128.119.245.12 (128.119.245.12)
```

2. IP address of gaia.cs.umass.edu: `128.119.245.12` ,Port: `80`



```
.297341    192.168.1.…  128.119.24…  HTTP    104  POST /ethereal-labs/lab3-1-reply.htm HTTP/1.
```

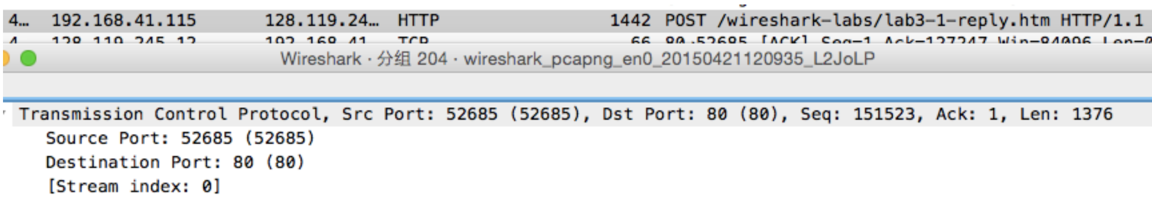
```
    Source: 192.168.1.102 (192.168.1.102)
    Destination: 128.119.245.12 (128.119.245.12)
```



```
5.297341    192.168.1.…  128.119.24…  HTTP    104  POST /ethereal-labs/lab3-1-reply.htm H
```

```
▼ Transmission Control Protocol, Src Port: 1161 (1161), Dst Port: 80 (80), Seq: 164041, Ack: 1, Len:
    Source Port: 1161 (1161)
    Destination Port: 80 (80)
```

3. I use `wget` to download alice.txt and upload.And my computer client IP: `192.168.41.115`,Souce Port: `52685`



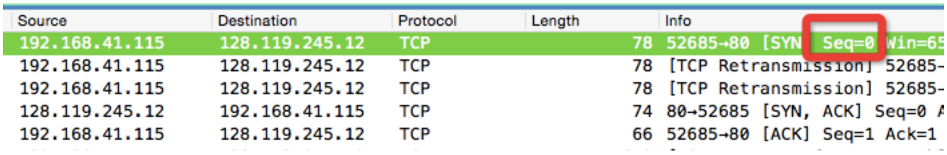
```
4…  192.168.41.115      128.119.24…  HTTP      1442  POST /wireshark-labs/lab3-1-reply.htm HTTP/1.1
```

Wireshark · 分组 204 · wireshark_pcapng_en0_20150421120935_L2JoLP

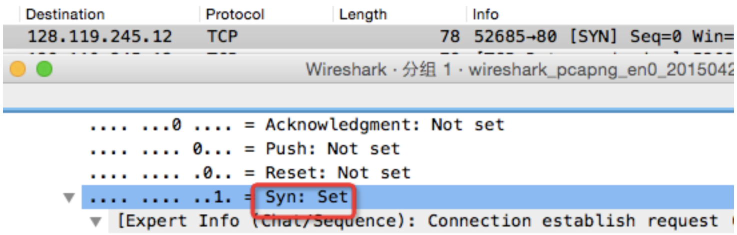
```
 Transmission Control Protocol, Src Port: 52685 (52685), Dst Port: 80 (80), Seq: 151523, Ack: 1, Len: 1376
    Source Port: 52685 (52685)
    Destination Port: 80 (80)
    [Stream index: 0]
```

Tcp Basics

4. Sequence Number is： 0,In flags of identifies note the Syn is set,ackledgment is not-set (note:I am not use the tcp-ethereal-trace-1 in wireshark- traces.zip)



```
Source           Destination      Protocol     Length      Info
192.168.41.115   128.119.245.12   TCP                  78  52685→80 [SYN] Seq=0 Win=65
192.168.41.115   128.119.245.12   TCP                  78  [TCP Retransmission] 52685-
192.168.41.115   128.119.245.12   TCP                  78  [TCP Retransmission] 52685-
128.119.245.12   192.168.41.115   TCP                  74  80→52685 [SYN, ACK] Seq=0 A
192.168.41.115   128.119.245.12   TCP                  66  52685→80 [ACK] Seq=1 Ack=1
```

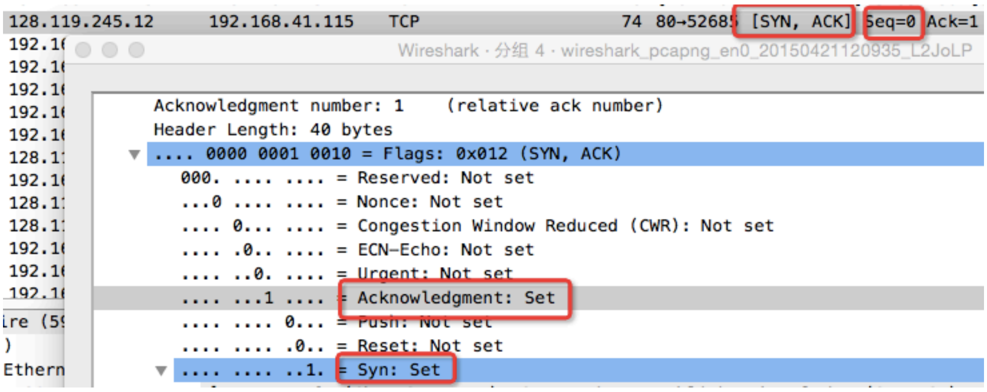


```
Destination      Protocol     Length      Info
128.119.245.12   TCP                  78  52685→80 [SYN] Seq=0 Win=
```

Wireshark · 分组 1 · wireshark_pcapng_en0_2015042

```
    .... ...0 .... = Acknowledgment: Not set
    .... .... 0... = Push: Not set
    .... .... .0.. = Reset: Not set
  ▼ .... .... ..1. = Syn: Set
    ▼ [Expert Info (Chat/Sequence): Connection establish request
```

5. Sequence Number is : 0.ACKnowledgement number is 1.Determine this value by last TCP SYN segment: `ACKnowledgement value= initiate sequence number of the TCP SYN segment+1` .In flags of identifies note the Syn is set,ackledgment is set

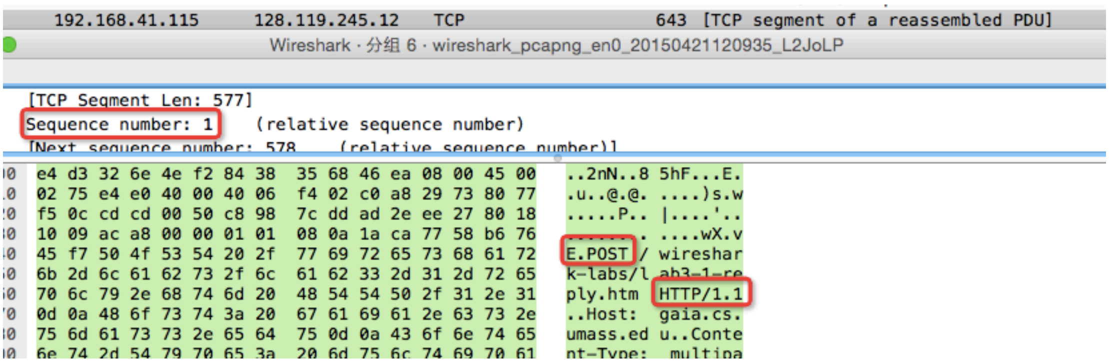


```
128.119.245.12   192.168.41.115   TCP                74  80→52685 [SYN, ACK] Seq=0 Ack=1
```

Wireshark · 分组 4 · wireshark_pcapng_en0_20150421120935_L2JoLP

```
192.1(      Acknowledgment number: 1    (relative ack number)
192.1(      Header Length: 40 bytes
192.1(    ▼ .... 0000 0001 0010 = Flags: 0x012 (SYN, ACK)
192.1(        000. .... .... = Reserved: Not set
192.1(        ...0 .... .... = Nonce: Not set
128.1:        .... 0... .... = Congestion Window Reduced (CWR): Not set
192.1(        .... .0.. .... = ECN-Echo: Not set
128.1:        .... ..0. .... = Urgent: Not set
128.1:        .... ...1 .... = Acknowledgment: Set
192.1(        .... .... 0... = Push: Not set
192.1(        .... .... .0.. = Reset: Not set
192.1(      ▼ .... .... ..1. = Syn: Set
ire (5!
)
Ethern
```

6. The sequence number of the TCP segment containing the HTTP POST command : 1.



```
    192.168.41.115   128.119.245.12   TCP             643  [TCP segment of a reassembled PDU]
```

Wireshark · 分组 6 · wireshark_pcapng_en0_20150421120935_L2JoLP

```
  [TCP Segment Len: 577]
  Sequence number: 1     (relative sequence number)
  [Next sequence number: 578    (relative sequence number)]
```

```
)0  e4 d3 32 6e 4e f2 84 38  35 68 46 ea 08 00 45 00   ..2nN..8 5hF...E.
.0  02 75 e4 e0 40 00 40 06  f4 02 c0 a8 29 73 80 77   .u..@.@. ....)s.w
!0  f5 0c cd cd 00 50 c8 98  7c dd ad 2e ee 27 80 18   .....P.. |....'..
}0  10 09 ac a8 00 00 01 01  08 0a 1a ca 77 58 b6 76   ........ ....wX.v
.0  45 f7 50 4f 53 54 20 2f  77 69 72 65 73 68 61 72   E.POST / wireshar
;0  6b 2d 6c 61 62 73 2f 6c  61 62 33 2d 31 2d 72 65   k-labs/l ab3-1-re
;0  70 6c 79 2e 68 74 6d 20  48 54 54 50 2f 31 2e 31   ply.htm  HTTP/1.1
'0  0d 0a 48 6f 73 74 3a 20  67 61 69 61 2e 63 73 2e   ..Host:  gaia.cs.
}0  75 6d 61 73 73 2e 65 64  75 0d 0a 43 6f 6e 74 65   umass.ed u..Conte
)0  6e 74 2d 54 79 70 65 3a  20 6d 75 6c 74 69 70 61   nt-Type:  multipa
```

7. The first six `[No,sequence number]` see below chart

No.	Sequence number
6	1
7	578
8	715
9	2143
11	3571
14	4999

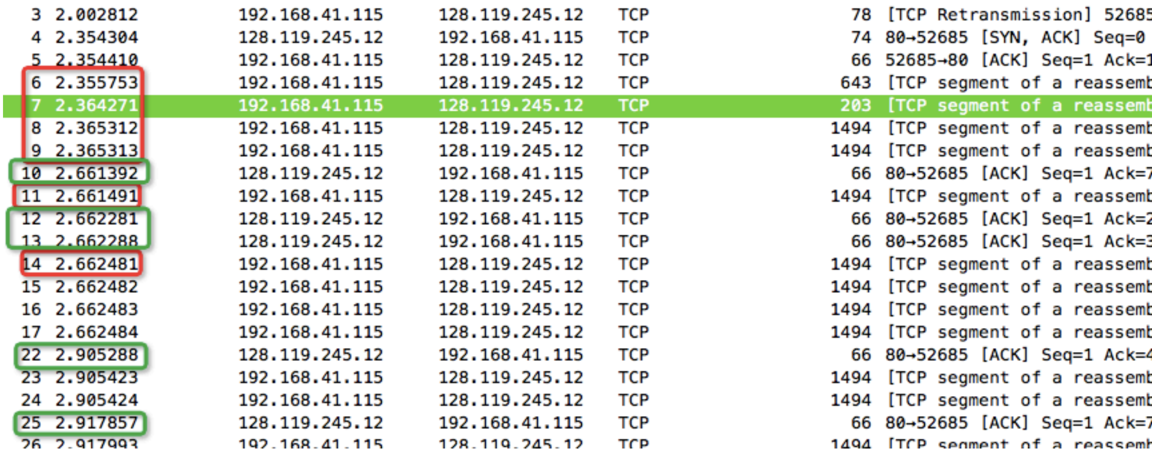
The first six `[No,time segment send,time ACK segment received,RTT]` see below chart

No.	Time send.	Time ACK received	RTT(seconds)
6	2.355753000	2.661392000	0.3056390
7	2.364271000	2.661392000	0.2971210
8	2.365312000	2.662281000	0.2969690
9	2.365313000	2.662288000	0.296975
11	2.661491000	2.905288000	0.243797
14	2.662481000	2.917857000	0.255376

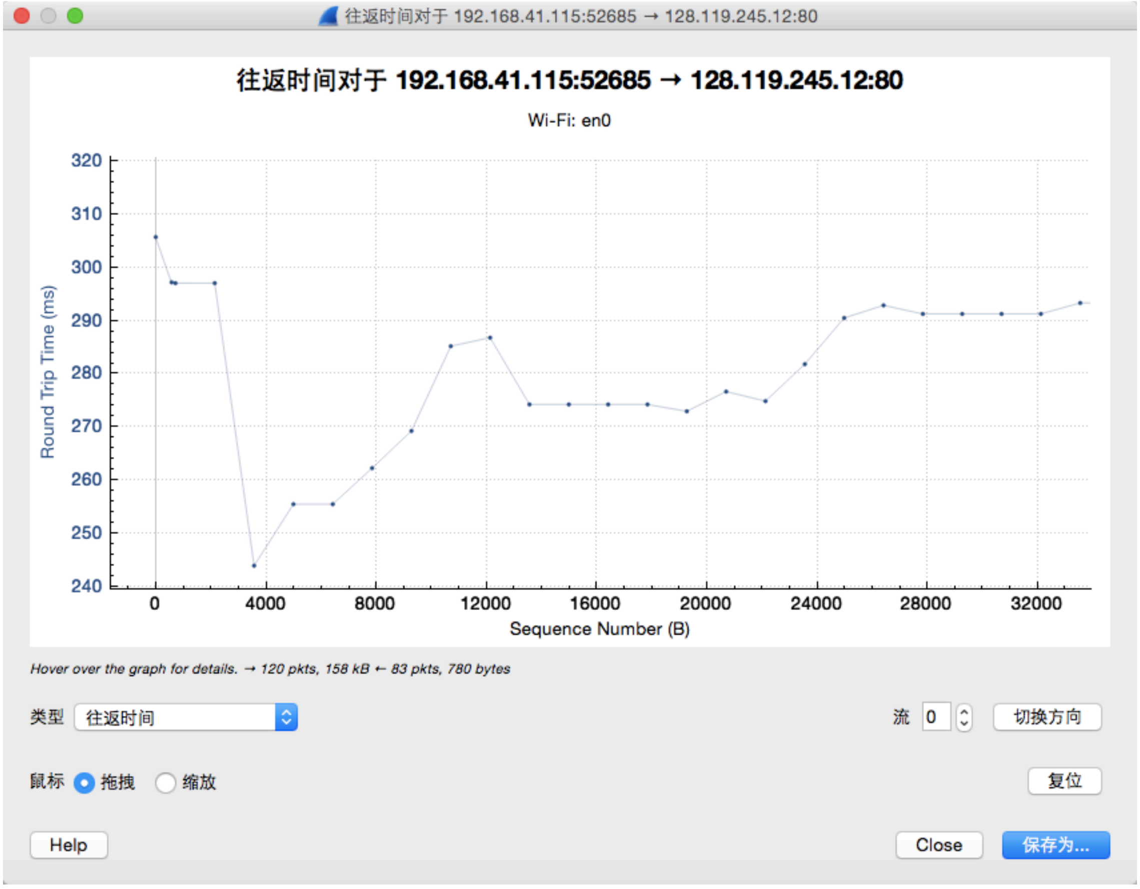
Determine the formula :
`EstimatedRTT = 0.875 * EstimatedRTT + 0.125 * SampleRTT`.
The first six `[No, EstimatedRTT]` see below chart

No.	EstimatedRTT(seconds)
6	0.3056390
7	0.30457425
8	0.297102
9	0.29696975
11	0.29032775
14	0.245244375

Snapshot:



```
 3 2.002812   192.168.41.115  128.119.245.12  TCP    78 [TCP Retransmission] 52685
 4 2.354304   128.119.245.12  192.168.41.115  TCP    74 80→52685 [SYN, ACK] Seq=0
 5 2.354410   192.168.41.115  128.119.245.12  TCP    66 52685→80 [ACK] Seq=1 Ack=1
 6 2.355753   192.168.41.115  128.119.245.12  TCP   643 [TCP segment of a reassemb
 7 2.364271   192.168.41.115  128.119.245.12  TCP   203 [TCP segment of a reassemb
 8 2.365312   192.168.41.115  128.119.245.12  TCP  1494 [TCP segment of a reassemb
 9 2.365313   192.168.41.115  128.119.245.12  TCP  1494 [TCP segment of a reassemb
10 2.661392   128.119.245.12  192.168.41.115  TCP    66 80→52685 [ACK] Seq=1 Ack=7
11 2.661491   192.168.41.115  128.119.245.12  TCP  1494 [TCP segment of a reassemb
12 2.662281   128.119.245.12  192.168.41.115  TCP    66 80→52685 [ACK] Seq=1 Ack=2
13 2.662288   128.119.245.12  192.168.41.115  TCP    66 80→52685 [ACK] Seq=1 Ack=3
14 2.662481   192.168.41.115  128.119.245.12  TCP  1494 [TCP segment of a reassemb
15 2.662482   192.168.41.115  128.119.245.12  TCP  1494 [TCP segment of a reassemb
16 2.662483   192.168.41.115  128.119.245.12  TCP  1494 [TCP segment of a reassemb
17 2.662484   192.168.41.115  128.119.245.12  TCP  1494 [TCP segment of a reassemb
22 2.905288   128.119.245.12  192.168.41.115  TCP    66 80→52685 [ACK] Seq=1 Ack=4
23 2.905423   192.168.41.115  128.119.245.12  TCP  1494 [TCP segment of a reassemb
24 2.905424   192.168.41.115  128.119.245.12  TCP  1494 [TCP segment of a reassemb
25 2.917857   128.119.245.12  192.168.41.115  TCP    66 80→52685 [ACK] Seq=1 Ack=7
26 2.917993   192.168.41.115  128.119.245.12  TCP  1494 [TCP segment of a reassemb
```



往返时间对于 192.168.41.115:52685 → 128.119.245.12:80

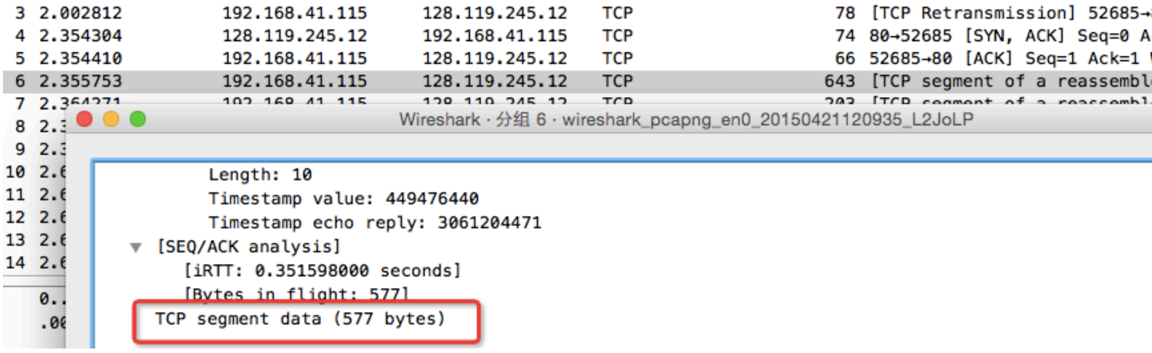
Hover over the graph for details. → 120 pkts, 158 kB ← 83 pkts, 780 bytes

类型 往返时间

流 0

鼠标 ● 拖拽 ○ 缩放

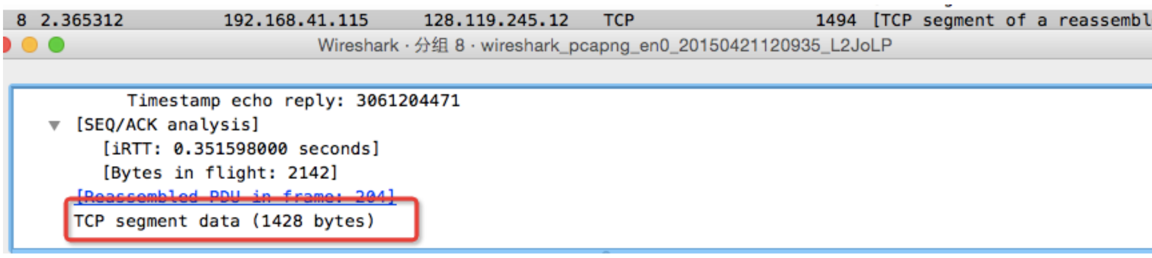
8. The length of first TCP package is `577 bytes`,The 5 rest is `1428 bytes`



```
 3 2.002812      192.168.41.115   128.119.245.12   TCP        78 [TCP Retransmission] 52685→
 4 2.354304      128.119.245.12   192.168.41.115   TCP        74 80→52685 [SYN, ACK] Seq=0 A
 5 2.354410      192.168.41.115   128.119.245.12   TCP        66 52685→80 [ACK] Seq=1 Ack=1
 6 2.355753      192.168.41.115   128.119.245.12   TCP       643 [TCP segment of a reassembl
 7 2.364271      192.168.41.115   128.119.245.12   TCP       203 [TCP segment of a reassembl
 8 2.3
 9 2.3
10 2.6
11 2.6
12 2.6
13 2.6
14 2.6
```

Wireshark · 分组 6 · wireshark_pcapng_en0_20150421120935_L2JoLP

```
        Length: 10
        Timestamp value: 449476440
        Timestamp echo reply: 3061204471
  ▼ [SEQ/ACK analysis]
      [iRTT: 0.351598000 seconds]
      [Bytes in flight: 577]
    TCP segment data (577 bytes)
```

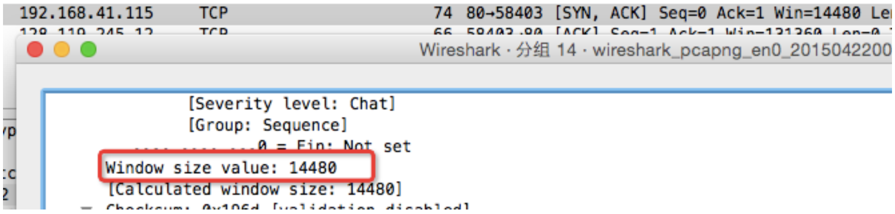


```
 8 2.365312      192.168.41.115   128.119.245.12   TCP      1494 [TCP segment of a reassembl
```

Wireshark · 分组 8 · wireshark_pcapng_en0_20150421120935_L2JoLP

```
        Timestamp echo reply: 3061204471
  ▼ [SEQ/ACK analysis]
      [iRTT: 0.351598000 seconds]
      [Bytes in flight: 2142]
    [Reassembled PDU in frame: 204]
    TCP segment data (1428 bytes)
```

9. The minimum amount of available buffer space: `14480`.No, sender is never throttled due to lacking of receiver buffer

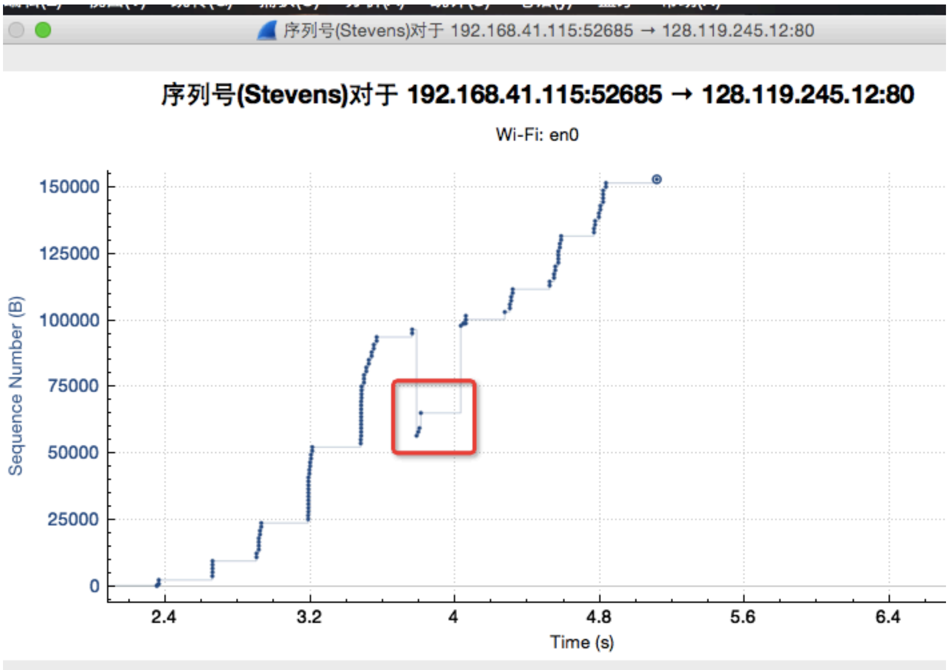


```
192.168.41.115   TCP        74 80→58403 [SYN, ACK] Seq=0 Ack=1 Win=14480 Le
128.119.245.12   TCP        66 58403→80 [ACK] Seq=1 Ack=1 Win=131360 Len=0
```

Wireshark · 分组 14 · wireshark_pcapng_en0_2015042200

```
            [Severity level: Chat]
            [Group: Sequence]
      .... .... ...0 = Fin: Not set
    Window size value: 14480
    [Calculated window size: 14480]
  ▼ Checksum: 0x106d [validation disabled]
```

10. Yes,there has some retransmitted segment.I check it by `Statistics->TCP Stream Graph- >Time/Sequene Graph`. And I find some decrease.



11. Amount data doesn't first is 137,subsequence are 1428,There cases in below table.

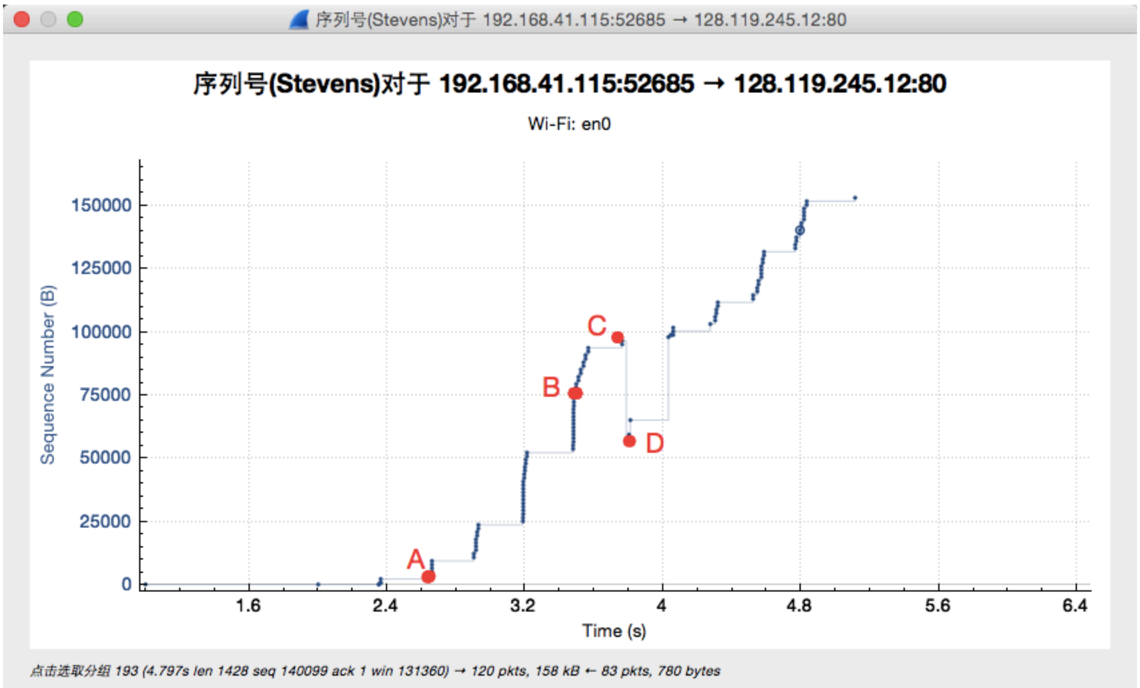
No.	Acknowledged sequence number	Acknowledged data
8	578	137
10	715	1428
12	2143	1428
13	3571	1428
22	4999	1428
25	7855	1428
30	9283	1428

12. The first segment size when start transmit data, sequence is `1` mean 1 byte.And the last sequence is `|164091|`,we conclude follow figures

```
amount data = 164091 - 1 = 16490 bytes
amount time = 4.836206 - 2.355753 = 2.480453 sec
```

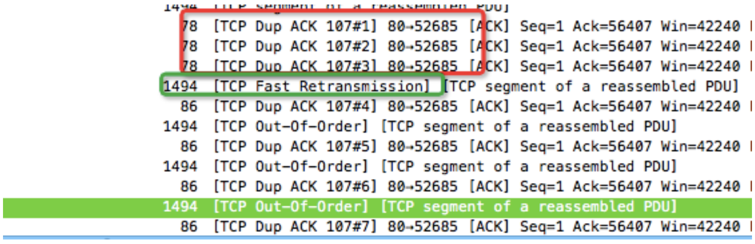
$$throughput = \frac{amount\ data}{amount\ time} = \frac{164090}{2.480453} = 66153.23894466052\, bytes/s$$

13. At point A the begin of slow start ,point B is end of slow start then TCP use additive-Increase approach because the `CongWin` reaches `Threshold`.So point B is the begin of congestion avoidance take over.Until at point C sender had received three duplicate ACKs,TCP use multiplicative decrease,`Threshold` is set to one half of the current `ConWin`.So point D is the end of congestion avoidance phase,then `CongWin` will ramps up linearly.



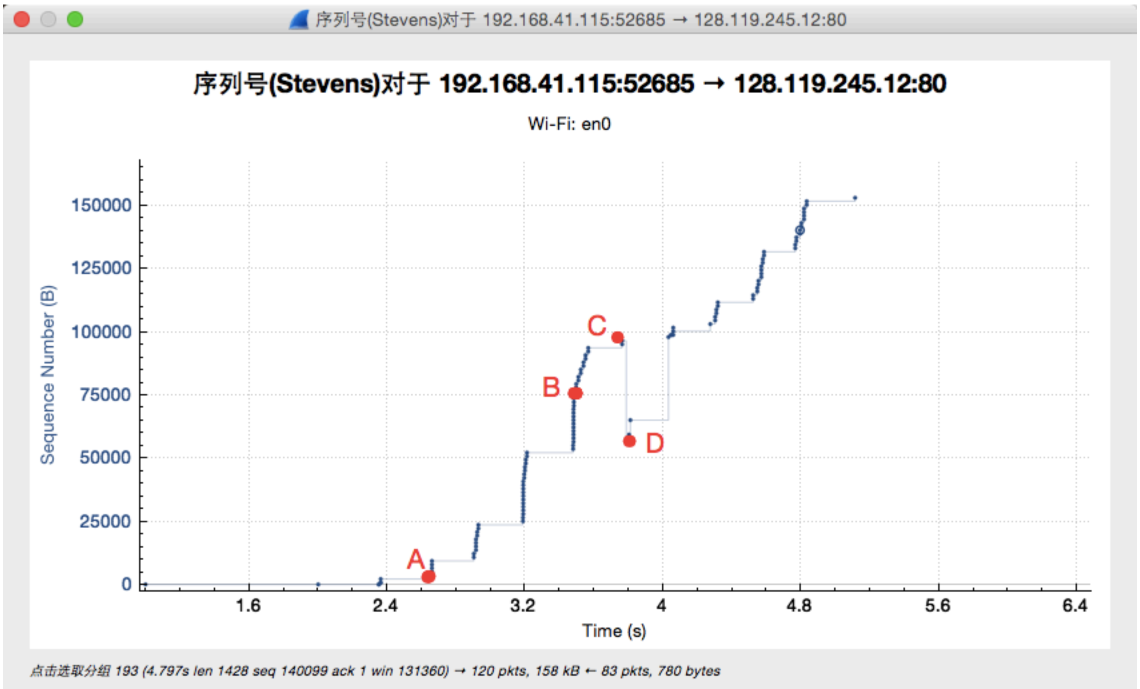
点击选取分组 193 (4.797s len 1428 seq 140099 ack 1 win 131360) → 120 pkts, 158 kB ← 83 pkts, 780 bytes

From below figure we can see after three duplicate ACKs,maybe the seq 56407 is lost,then client send a fast retransmission.And TCP use multiplicative decrease approach.



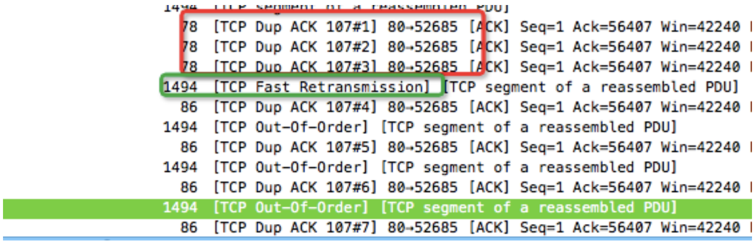
```
1494  [TCP segment of a reassembled PDU]
  78  [TCP Dup ACK 107#1] 80→52685 [ACK] Seq=1 Ack=56407 Win=42240 I
  78  [TCP Dup ACK 107#2] 80→52685 [ACK] Seq=1 Ack=56407 Win=42240 I
  78  [TCP Dup ACK 107#3] 80→52685 [ACK] Seq=1 Ack=56407 Win=42240 I
1494  [TCP Fast Retransmission] [TCP segment of a reassembled PDU]
  86  [TCP Dup ACK 107#4] 80→52685 [ACK] Seq=1 Ack=56407 Win=42240 I
1494  [TCP Out-Of-Order] [TCP segment of a reassembled PDU]
  86  [TCP Dup ACK 107#5] 80→52685 [ACK] Seq=1 Ack=56407 Win=42240 I
1494  [TCP Out-Of-Order] [TCP segment of a reassembled PDU]
  86  [TCP Dup ACK 107#6] 80→52685 [ACK] Seq=1 Ack=56407 Win=42240 I
1494  [TCP Out-Of-Order] [TCP segment of a reassembled PDU]
  86  [TCP Dup ACK 107#7] 80→52685 [ACK] Seq=1 Ack=56407 Win=42240 I
```

14. At point A the begin of slow start ,point B is end of slow start then TCP use additive-Increase approach because the `CongWin` reaches `Threshold`.So point B is the begin of congestion avoidance take over.Until at point C sender had received three duplicate ACKs,TCP use multiplicative decrease,`Threshold` is set to one half of the current `ConWin`.So point D is the end of congestion avoidance phase,then `CongWin` will ramps up linearly.



点击选取分组 193 (4.797s len 1428 seq 140099 ack 1 win 131360) → 120 pkts, 158 kB ← 83 pkts, 780 bytes

From below figure we can see after three duplicate ACKs,maybe the seq 56407 is lost,then client send a fast retransmission.And TCP use multiplicative decrease approach.



```
1494  [TCP segment of a reassembled PDU]
  78  [TCP Dup ACK 107#1] 80→52685 [ACK] Seq=1 Ack=56407 Win=42240
  78  [TCP Dup ACK 107#2] 80→52685 [ACK] Seq=1 Ack=56407 Win=42240
  78  [TCP Dup ACK 107#3] 80→52685 [ACK] Seq=1 Ack=56407 Win=42240
1494  [TCP Fast Retransmission] [TCP segment of a reassembled PDU]
  86  [TCP Dup ACK 107#4] 80→52685 [ACK] Seq=1 Ack=56407 Win=42240
1494  [TCP Out-Of-Order] [TCP segment of a reassembled PDU]
  86  [TCP Dup ACK 107#5] 80→52685 [ACK] Seq=1 Ack=56407 Win=42240
1494  [TCP Out-Of-Order] [TCP segment of a reassembled PDU]
  86  [TCP Dup ACK 107#6] 80→52685 [ACK] Seq=1 Ack=56407 Win=42240
1494  [TCP Out-Of-Order] [TCP segment of a reassembled PDU]
  86  [TCP Dup ACK 107#7] 80→52685 [ACK] Seq=1 Ack=56407 Win=42240
```