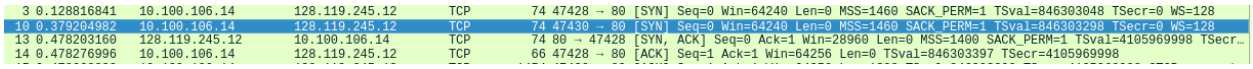
Name: Abhijit Paul

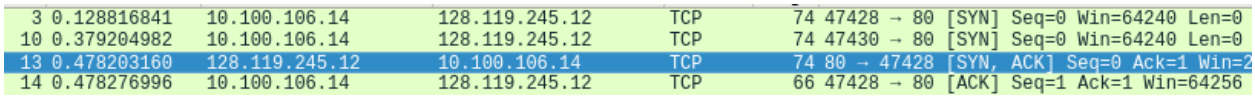
ID:1201

Lab: WireShark TCP

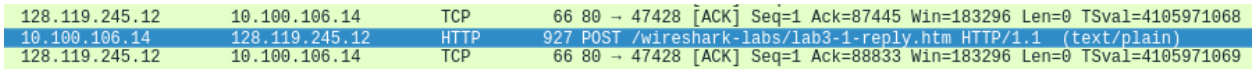
1. IP address and TCP port number used by the client computer:
   Source IP: 10.100.106.14
   Source Port: 47430, 47428
   We found two port numbers for the first two SYN segments. By looking at the code, we are able to determine that port 47428 communicates with port 80 later.

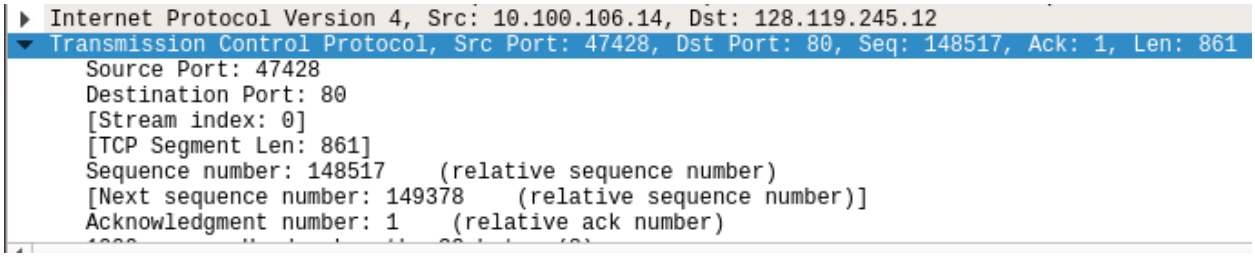


2. The IP address and port of gaia.cs.umass.edu:
   Destination IP:128.119.245.12
   Destination Port: 80



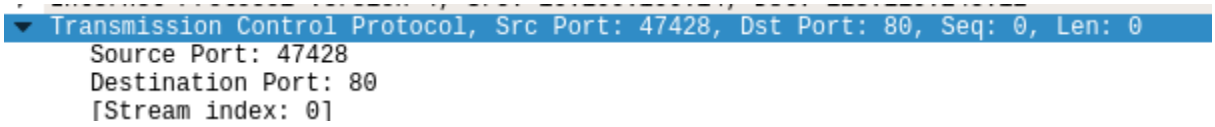
3. The IP address and TCP port number used by the client computer (source) to transfer the file to gaia.cs.umass.edu is:





   Client IP: 10.100.106.14
   Client Port: 47428

4. The sequence number of the TCP SYN segment:

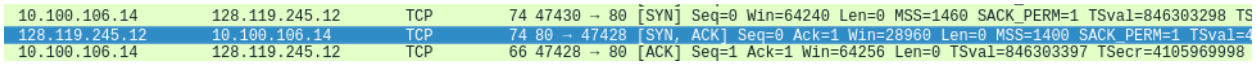


   SYN segment number = 0

```
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
```

SYN FLAG is set(1).

5. Sequence number of the SYNACK segment is = 0



The value of the Acknowledgement field in the SYNACK segment = 1

```
Flags: 0x012 (SYN, ACK)
    000. .... .... = Reserved: Not set
    ...0 .... .... = Nonce: Not set
    .... 0... .... = Congestion Window Reduced (CWR
    .... .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·]
```

How did gaia.cs.umass.edu determine that value of SYNACK, ACK Flag?

Answer: The server understood that from the previous SYN TCP segment. Under TCP protocol 3-way handshake, the destination returns SYNACK when it receives SYN and wishes to establish the connection.

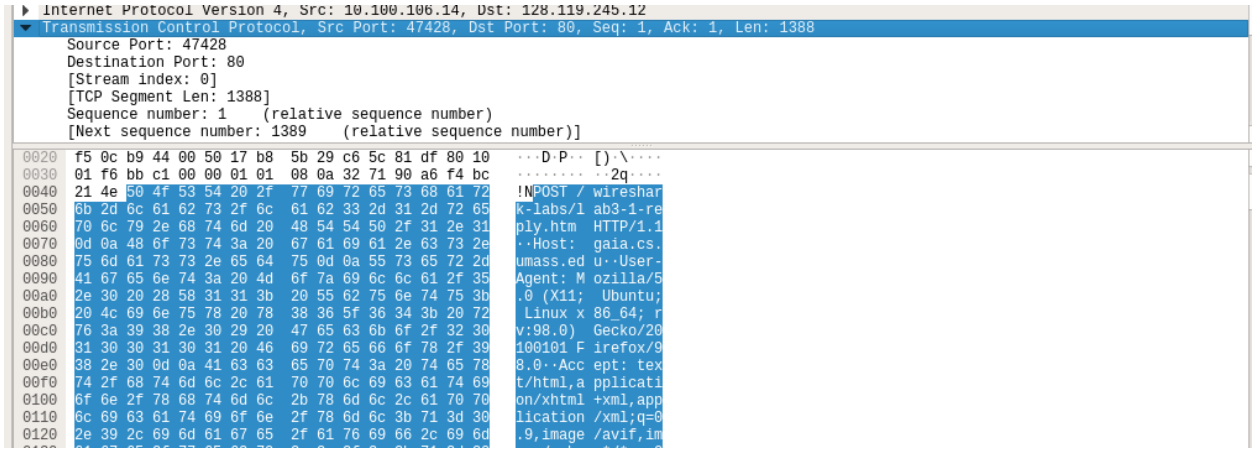
What is it in the segment that identifies the segment as a SYNACK segment?

SYN flag = 1

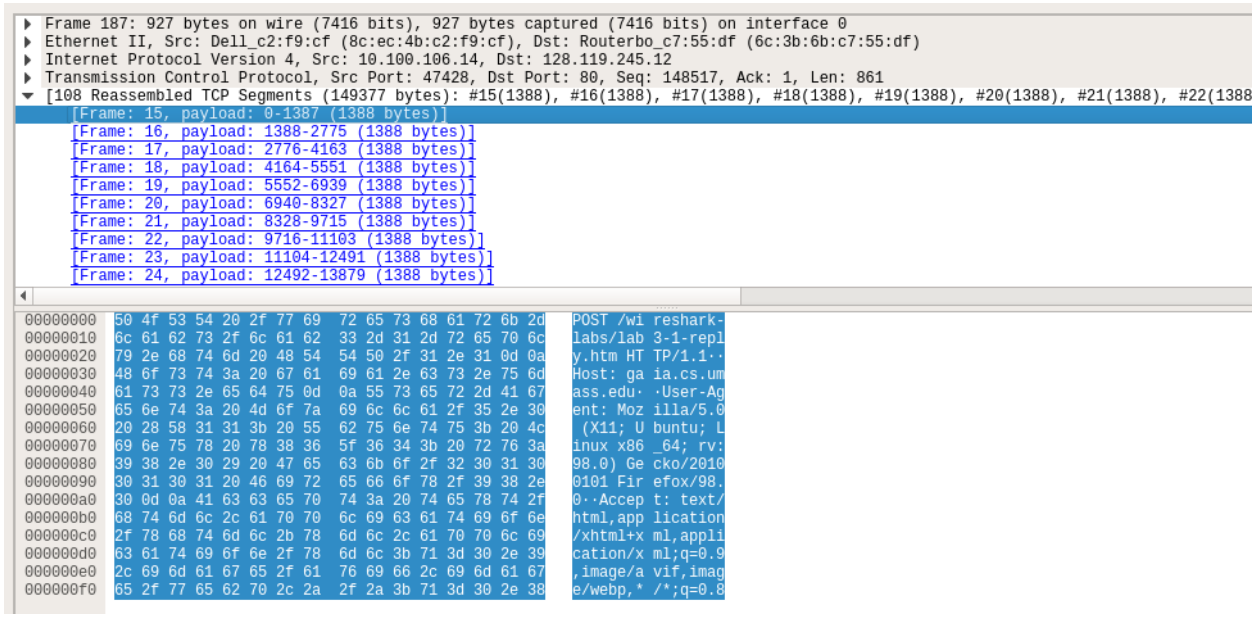
ACK flag = 1

6. **What is the sequence number of the TCP segment containing the HTTP POST command?**

   Sequence Number = 1



7. If TCP POST Seq = 0

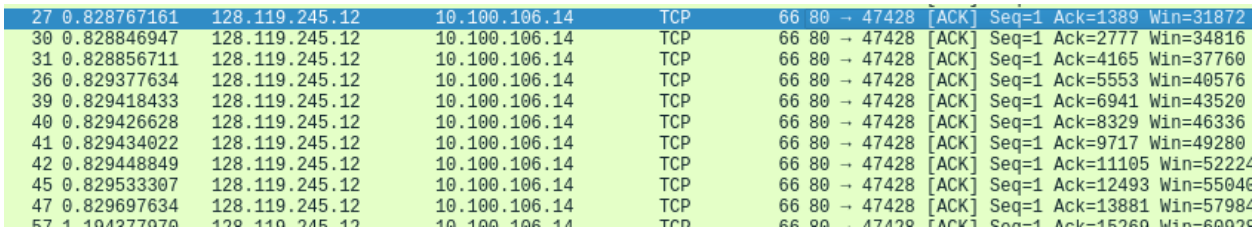


What are the sequence numbers of the first six segments in the TCP connection (including the segment containing the HTTP POST)?
tcp.ack == 2777

Sequence Number	Sent(seconds)	ACK Received	RTT
0	0.478668828	0.828767161	0.350098333
1389	0.478683740	0.828846947	0.350163207

2777	0.478788173	0.828856711	0.350068538
4165	0.478795434	0.829377634	0.3505822
5553	0.478973377	0.829418433	0.350445056
6941	0.478987045	0.829426628	0.350439583

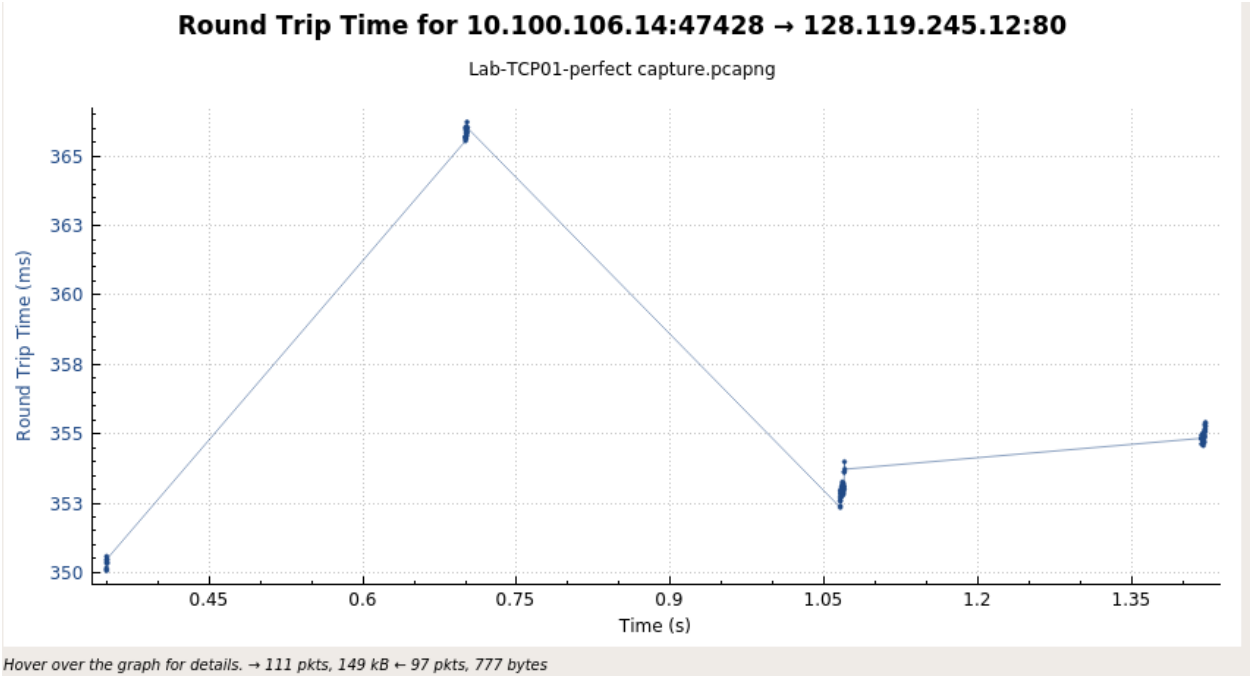
Tcp.ack == 1 is sent time
Tcp.seq == 1 is receive time



27	0.828767161	128.119.245.12	10.100.106.14	TCP	66 80 → 47428 [ACK] Seq=1 Ack=1389 Win=31872
30	0.828846947	128.119.245.12	10.100.106.14	TCP	66 80 → 47428 [ACK] Seq=1 Ack=2777 Win=34816
31	0.828856711	128.119.245.12	10.100.106.14	TCP	66 80 → 47428 [ACK] Seq=1 Ack=4165 Win=37760
36	0.829377634	128.119.245.12	10.100.106.14	TCP	66 80 → 47428 [ACK] Seq=1 Ack=5553 Win=40576
39	0.829418433	128.119.245.12	10.100.106.14	TCP	66 80 → 47428 [ACK] Seq=1 Ack=6941 Win=43520
40	0.829426628	128.119.245.12	10.100.106.14	TCP	66 80 → 47428 [ACK] Seq=1 Ack=8329 Win=46336
41	0.829434022	128.119.245.12	10.100.106.14	TCP	66 80 → 47428 [ACK] Seq=1 Ack=9717 Win=49280
42	0.829448849	128.119.245.12	10.100.106.14	TCP	66 80 → 47428 [ACK] Seq=1 Ack=11105 Win=5222
45	0.829533307	128.119.245.12	10.100.106.14	TCP	66 80 → 47428 [ACK] Seq=1 Ack=12493 Win=5504
47	0.829697634	128.119.245.12	10.100.106.14	TCP	66 80 → 47428 [ACK] Seq=1 Ack=13881 Win=5798

Estimated RTT:
Assuming, α=0.125 and EstimatedRTT for segment 1 = 0.350098333

EstimatedRTT for segment 2 = 0.875*0.350098333 +
0.125*0.350163207 = 0.35010644225
EstimatedRTT for segment 3 =
0.875*0.35010644225+0.125*0.350068538 = 0.35010170421
EstimatedRTT for segment 4 =
0.875*0.35010170421+0.125*0.3505822 = 0.35016176618
EstimatedRTT for segment 5 =
0.875*0.35016176618+0.125*0.350445056 = 0.3501971774
EstimatedRTT for segment 6 =
0.875*0.3501971774+0.125*0.350439583 = 0.3502274781

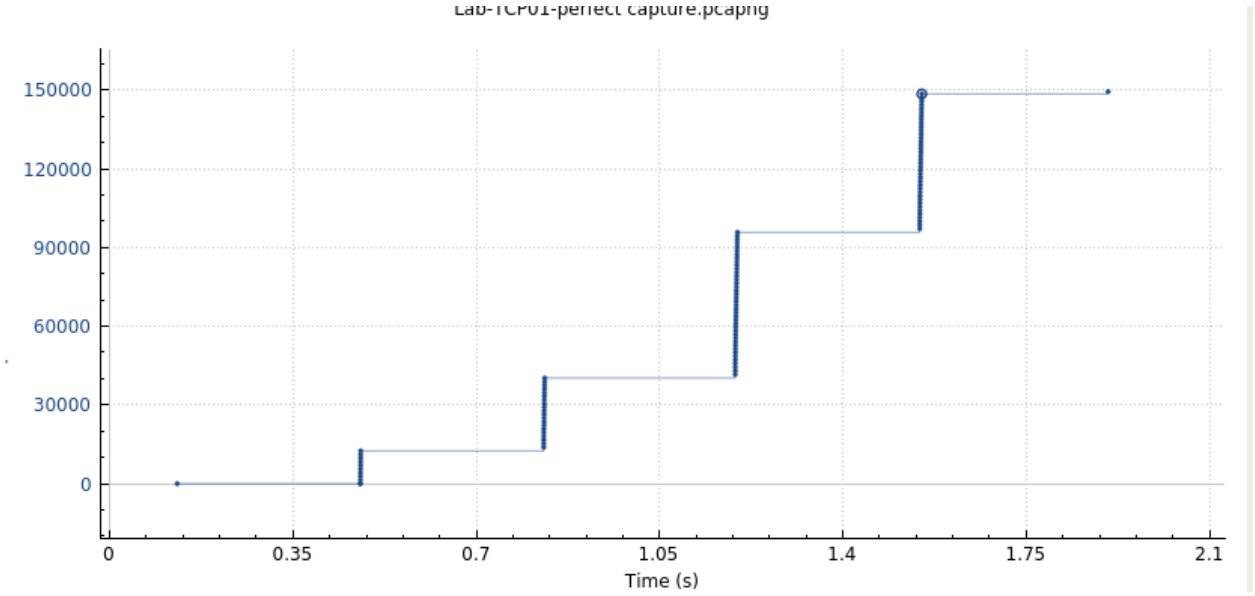


8. **What is the length of each of the first six TCP segments?**
   1388 bytes
9. **What is the minimum amount of available buffer space advertised at the received for the entire trace?**
   1499 bytes

```
[Next sequence number: 1    (relative sequence number)]
Acknowledgment number: 124921    (relative ack number)
1000 .... = Header Length: 32 bytes (8)
▸ Flags: 0x010 (ACK)
Window size value: 1499
[Calculated window size: 191872]
[Window size scaling factor: 128]
Checksum: 0x4762 [unverified]
[Checksum Status: Unverified]
Urgent pointer: 0
▸ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
▸ [SEQ/ACK analysis]
▸ [Timestamps]
```

Does the lack of receiver buffer space ever throttle the Sender?

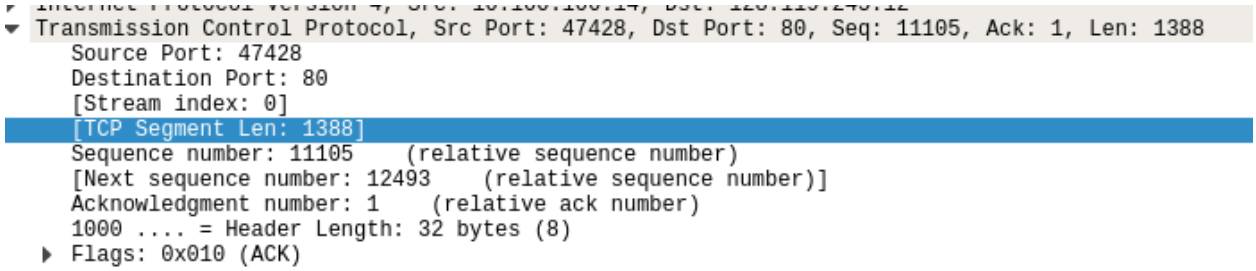
No, there was no packet loss as we can see in the graph. The congestion control algorithm has increased the congestion-window size over the whole time.



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

    No, no retransmission happened. From the above graph, we can be sure that no package loss occurred. I checked the graph for it.

11. **How much data does the receiver typically acknowledge in an ACK?**



    The segment size is 1388 bytes.

12. **What is the throughput (bytes transferred per unit time) for the TCP connection?**

    Total bytes sent = last ACK from server - 1 = 149378 -1 = 149377

    Total time = last ACK time - first Segment time = 1.906035922 - 0.128816841

So throughput = Total bytes sent / Total time = 84050.9769431 bytes/second

215	1.905331223	128.119.245.12	10.100.106.14	TCP	66 80 → 47428 [ACK] Seq=1 Ack=142965 Win=229632 Len=0 TSval=4105971421 TSecr=846304469
216	1.905338335	128.119.245.12	10.100.106.14	TCP	66 80 → 47428 [ACK] Seq=1 Ack=144353 Win=232448 Len=0 TSval=4105971421 TSecr=846304469
217	1.905458010	128.119.245.12	10.100.106.14	TCP	66 80 → 47428 [ACK] Seq=1 Ack=145741 Win=235392 Len=0 TSval=4105971421 TSecr=846304469
218	1.905881865	128.119.245.12	10.100.106.14	TCP	66 80 → 47428 [ACK] Seq=1 Ack=147129 Win=238208 Len=0 TSval=4105971421 TSecr=846304470
219	1.906035922	128.119.245.12	10.100.106.14	TCP	66 80 → 47428 [ACK] Seq=1 Ack=149378 Win=242816 Len=0 TSval=4105971421 TSecr=846304470
220	1.906480109	128.119.245.12	10.100.106.14	HTTP	843 HTTP/1.1 200 OK (text/html)
221	1.906512019	10.100.106.14	128.119.245.12	TCP	66 47428 → 80 [ACK] Seq=149378 Ack=778 Win=64128 Len=0 TSval=846304826 TSecr=4105971422

13-14. **Congestion - Slow Start and Avoidance state**

From the graph, we can see that it never had to go to congestion avoidance state. It remained in slow start state the whole time.