NETWORK ARCHITECTURE 1

HOME WORK -3

SUBMITTED BY

JASWANTH KRISHNA AAVULA

16133295

1. Compare GBN, SR, and TCP (no delayed ACK). Assume that the timeout values for all three protocols are sufficiently long such that 5 consecutive data segments and their corresponding ACKs can be received (if not lost in the channel) by the receiving host (host B) and the sending host (host A) respectively. Suppose Host A sends 5 data segments to Host B, and the 2nd segment (sent from A) is lost. In the end, all 5 data segments have been correctly received by Host B.

(a) How many segments has Host A sent in total and how many ACKs has Host B sent in total? What are their sequence numbers? Answer this question for all three protocols?

(b) If the timeout values for all three protocol are much longer than 5 RTT, then which protocol successfully delivers all five data segments in shortest time interval?

Sol:

(a)

**GO BACK N:**

By data 5 data segments are supposed to be transmitted between Hosts A and B, but the second segment among the five segments is lost. It has also been mentioned that a sufficient timeout is maintained.

In go back N protocol if second segment is lost then all the following segments will be discarded and retransmission of segments takes place. So a total of **nine** segments will be transmitted and **eight** acknowledgements would be received.

Number of segments transmitted = 9

Number of acknowledgements received =8

Order of Sequence numbers Transmitted: Seq-1,Seq-2,Seq-3,Seq-4,Seq-5,Seq-2,Seq-3,Seq- 4,Seq-5

Order of acknowledgements received: Ack-1,Ack-1,Ack-1,Ack-1,Ack-2,Ack-3,Ack-4,Ack-5.

**SELECTIVE REPEAT**:

In selective repeat protocol if second segment is lost the host B receives other segments and stores them. After timeout the second segment is retransmitted again. So a total of **Six** segments are transmitted and **five** acknowledgements are received.

Number of segments transmitted = 6

Number of acknowledgements received = 5

Order of sequence numbers transmitted = Seq-1,Seq-2,Seq-3,Seq-4, Seq-5,Seq-2

Order of acknowledgements received= Ack-1,Ack-3,Ack-4,Ack-5,Ack-2.

**TCP:**

Implementation of TCP with a buffer can be explained as described below. For instance let us consider that each segment consists of 4 bits of data.

In this case even though the second segment is lost all the other segments transmitted are stored at receiver buffer but the acknowledgements sent by the host B contains the value of expected sequence number.

Number of segments transmitted = 6

Number of acknowledgements received = 5

Order of sequence numbers transmitted =Seq-1,Seq-5,Seq-9,Seq-13,Seq-17,Seq-5

Order of acknowledgements received = Ack-5,Ack-5,Ack-5,Ack-5,Ack-21

(b)TCP transmits all the five data segments successfully in shortest time interval because it sends data segments immediately after receiving 3 duplicate ACKs.

2. Consider TCP procedure for estimating RTT. Suppose that α = 0.1. Let sampleRTT1 be the most recent sample RTT, let sampleRTT2be the next most recent sample RTT, and so on.

(a) For a given TCP connection, suppose four acknowledgements have been returned with corresponding sample RTTs, sampleRTT4, sampleRTT3, sampleRTT2 and sampleRTT1. Expressed Estimated RTT in terms of the four sample RTTs.

Sol:

By data

α = 0.1

The RTT’s are given as sample RTT1, sample RTT2, Sample RTT3 and sample RTT4.

Total estimated RTT is given by the following formula

Estimated RTT = (1- α)\*Estimated RTT + α\*Sample RTT

Now we have four RTT’s. So there will be four estimated RTT’s

For RTT1, Estimated RTT1 = Sample RTT1

For RTT2, Estimated RTT2 = (1- α) Estimated RTT1 + αSample RTT2

=(1-0.1)Estimated RTT1 + 0.1Sample RTT2

= 0.9sample RTT1 + 0.1Sample RTT2

For RTT3, Estimated RTT3 = (1- α)[ α Sample RTT2 + (1- α)Estimated RTT1] + αSampleRTT3

=(1-0.1)[0.1Sample RTT2 + (1-0.1)sample RTT1] + 0.1sample RTT3

= 0.9[0.1Sample RTT2 + 0.9sample RTT1] + 0.1 sample RTT3

=0.1SampleRTT3+ 0.09SampleRTT2 + (0.9)2SampleRTT1

For RTT4, Estimated RTT4 = (1-α) Estimated RTT3 +αSample RTT4.

=0.9(0.1SampleRTT3+ 0.09SampleRTT2 + (0.9)2SampleRTT1) +0.1sample RTT4

=0.09sample RTT3+0.081sample RTT2+0.729sample RTT1+0.1sample RTT4

(b) Generalize the formula for n sample RTTs.

Sol:

The generalised formula for n sample RTT’s would be the summation of individual RTT’s.

Generalized Estimated RTT(n) = αΣn-1 (1- α)i-1Sample RTT(i) + (1- α)n-1SampleRTT(n)

(Note: The i value in the above equation starts from i=1)

(c) For the formula in part (b) let n approach infinity. Comment on why this averaging procedure is called an exponential moving average.

Sol:

In the generalized estimated RTT equation if n is approaching infinity then the second term , (1- α)n-1SampleRTT(n) approaches zero and the equation becomes

Generalized Estimated RTT(n) = αΣ∞-1 (1- α)i-1Sample RTT(i) + 0

For α= 0.1 Generalized Estimated RTT(n) = 0.1Σ∞-1 (0.9)i-1Sample RTT(i)

So this shows that estimated RTT decreases exponentially and hence it is called as exponential moving average.

3. Consider sending a large file from a host to another over a TCP connection that has no loss.

a. Suppose TCP uses AIMD for its congestion control without slow start. Assuming cwnd increases by 1 MSS every time a batch of ACKs is received and assuming approximately constant RTTs, how long does it take for cwnd increase from 5 MSS to 11 MSS?

Sol: By data

Cwnd increases by 1 MSS every time and RTT’s are assumed to be constant.

So total time taken for cwnd to increase from 5 MSS to 11 MSS = (11-5)MSS\*1RTT/1MSS

= (6)MSS\*1RTT/1MSS

= 6RTT.

b. What is the average throughput in terms of MSS and RTT for this connection up through time = 6 RTT?

Sol: Average throughput = (Total message sent) ÷ ( Total time)

Total messages sent = (5+6+7+8+9+10)MSS

=45 MSS

Total time = 6 RTT

Therefore average throughput = (45 MSS )÷( 6 RTT)

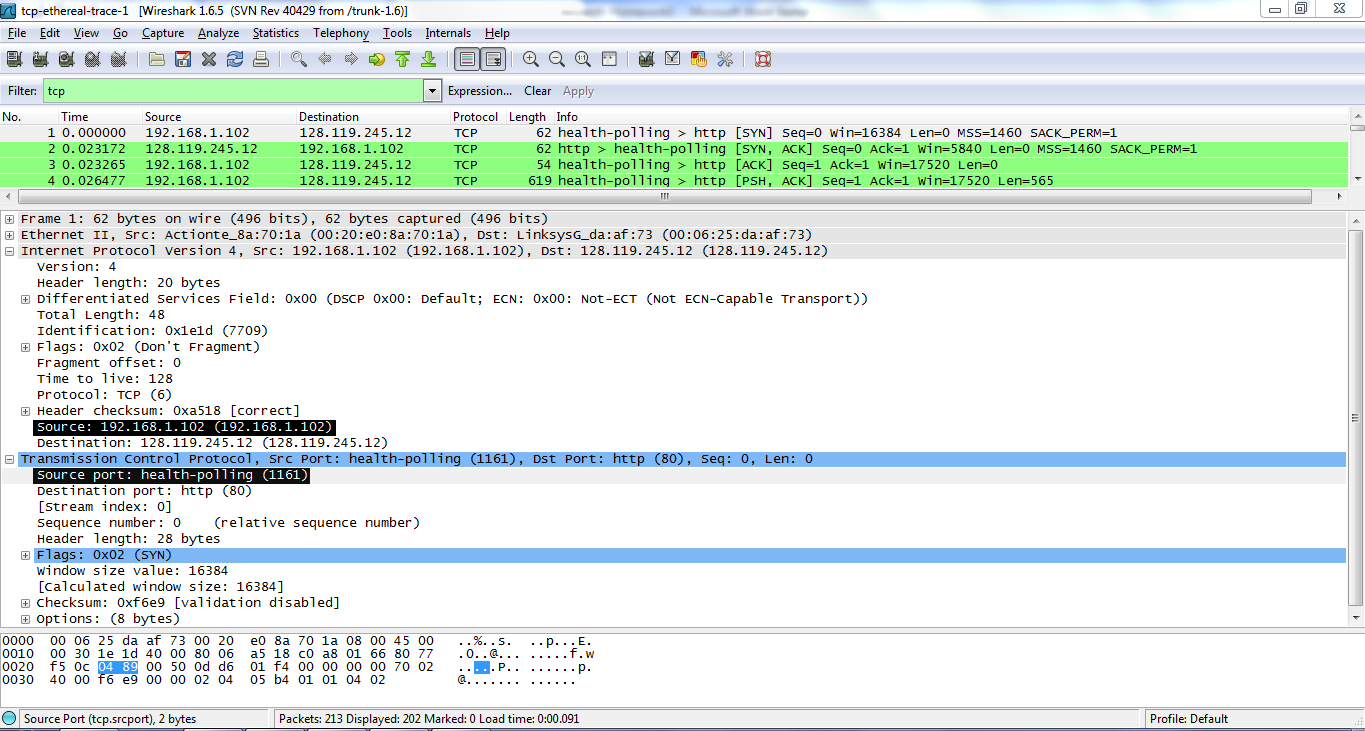
= 7.5MSS/RTT.

**Laboratory Homework: Wireshark**

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:

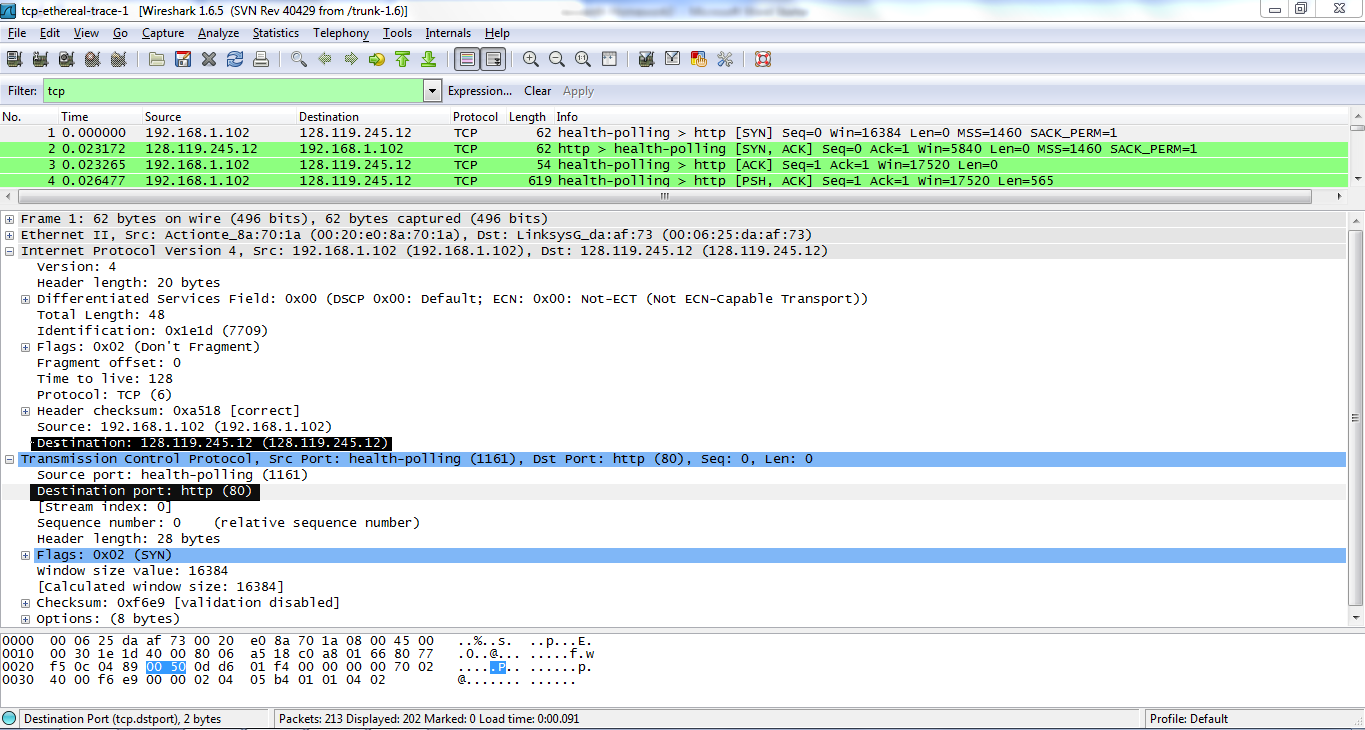


Source IP address : 192.168.1.102

Port number :1161

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

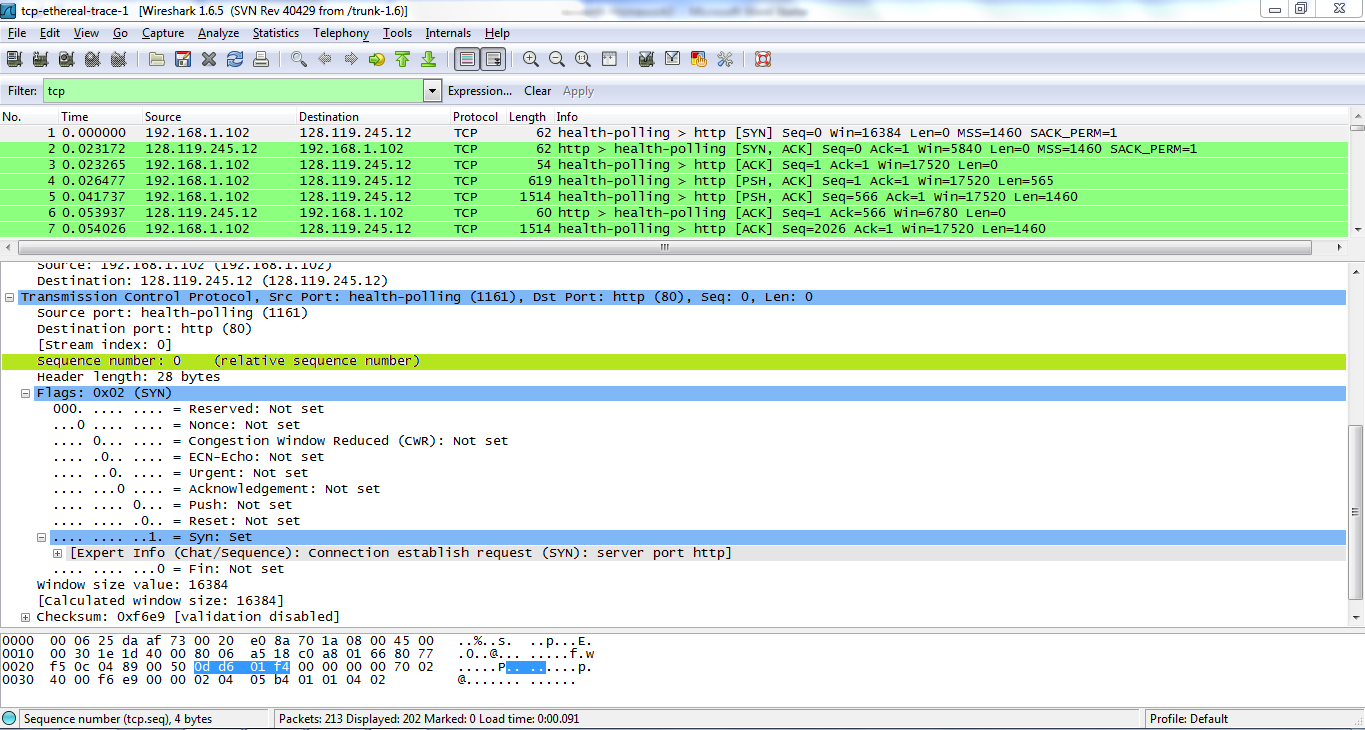
Ans:



The IP address of gaia.cs.umass.edu is **128.119.245.12** and the port number used by it to send and receive TCP segments is **80**

3. 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?

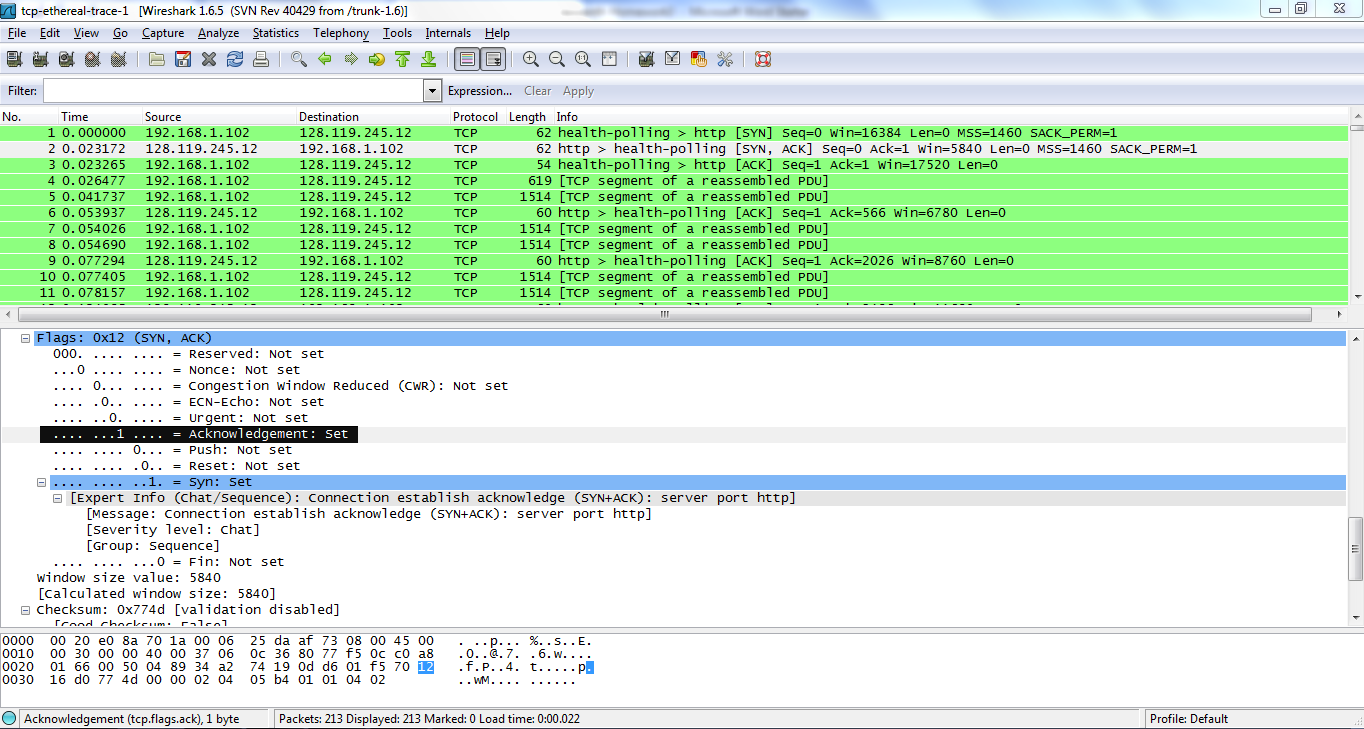
Ans:



* 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 is Zero (0).
* Flag identifies the segment as a SYN segment, ‘Syn :set’ in Flags : 0\*02 is set as 1

4. 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:

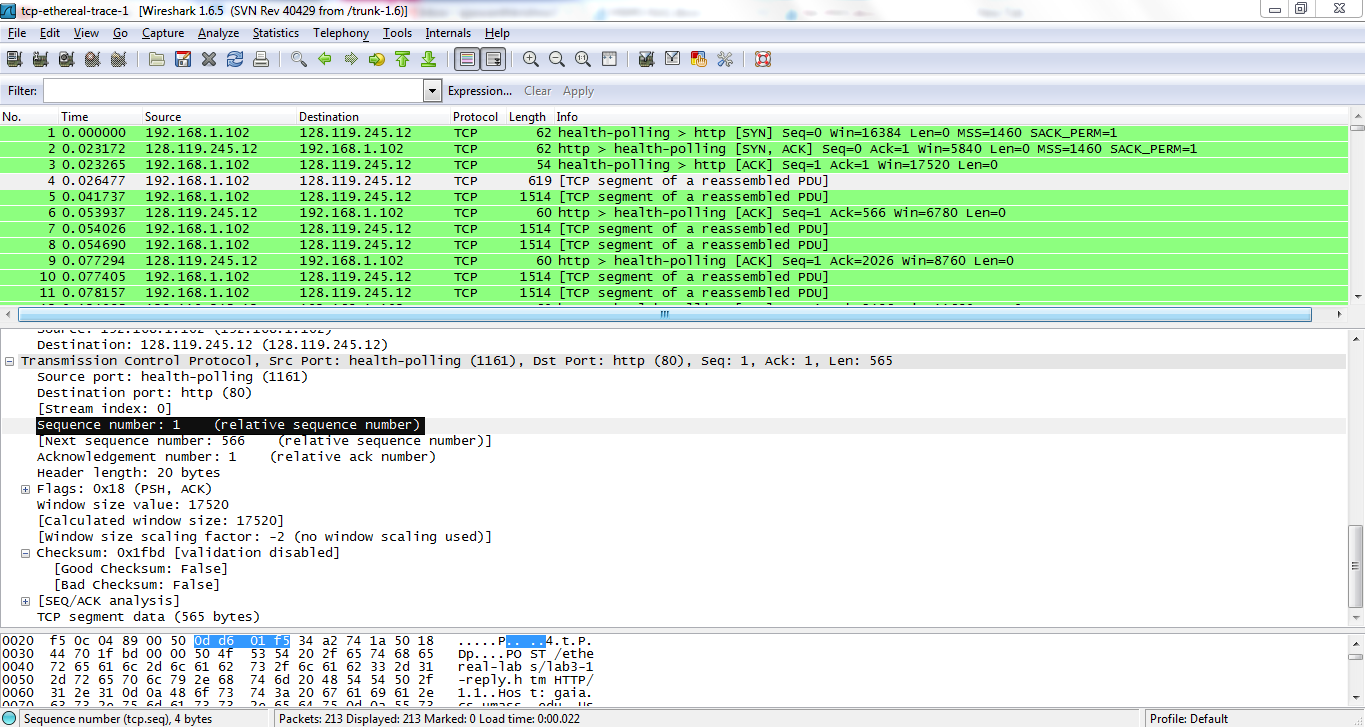


The sequence number of the SYNACK segment sent by gaia.cs.umass.edu to the client computer in reply to the SYN is Zero (0) and acknowledgement is one (1).

The segment that identifies the segment as a SYNACK segment is Flags: 0\*12(SYN, ACK) where in syn is set to 1. ACK i.e., relative Acknowledgement number is set to 1.

5. 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 Wire shark window, looking for a segment with a “POST” within its DATA field.

Ans

:

The sequence number of TCP segment containing the HTTP POST command is 1. It is found in frame 4.

6. 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 Estimated RTT value after the receipt of each ACK? Assume that the value of the Estimated RTT is equal to the measured RTT for the first segment, and then is computed using the Estimated RTT equation all subsequent segments. Note: Wire shark 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.

Ans:

Frame 4 is considered as the first segment as this contains the HTTP post. Therefore the sequence numbers of first six segments for frames 4,5,7,8,9,11 are as follows.

* Frame 4 sequence number:1 segment was sent at 0.026477
* Frame 5 sequence number:566 segment was sent at 0.041737
* Frame 7 sequence number:2026 segment was sent at 0.054026
* Frame 8 sequence number:3486 segment was sent at 0.054690
* Frame 10 sequence number:4946 segment was sent at 0.077405
* Frame 11 sequence number:6406 segment was sent at 0.078157

The segments of acknowledgements are received at time stated as follows.

* Frame 6 acknowledgement was received at 0.053937
* Frame 9 acknowledgement was received at 0.077294
* Frame 12 acknowledgement was received at 0.124085
* Frame 14 acknowledgement was received at 0.169118
* Frame 15 acknowledgement was received at 0.217299
* Frame 16 acknowledgement was received at 0.267802

The difference between TCP segment sent and its ACK received are as follows:

0.053937-0.026477 =0.02746

0.077294-0.041737 =0.035557

0.124085-0.054026 =0.070059

0.169118-0.054690 =0.114428

0.217299-0.077405 =0.139894

0.267802-0.078157 =0.189645

The Estimated RTT can be calculated as follows:

Estimated RTT = α\*Sample RTT + (1-α)\*Estimated RTT

For α = 0.125, then Estimated RTT = 0.125\*Sample RTT + 0.875\*Estimated RTT

Estimated RTT’s are calculated as follows:

For first ACK Estimated RTT1= sample RTT =0.02746

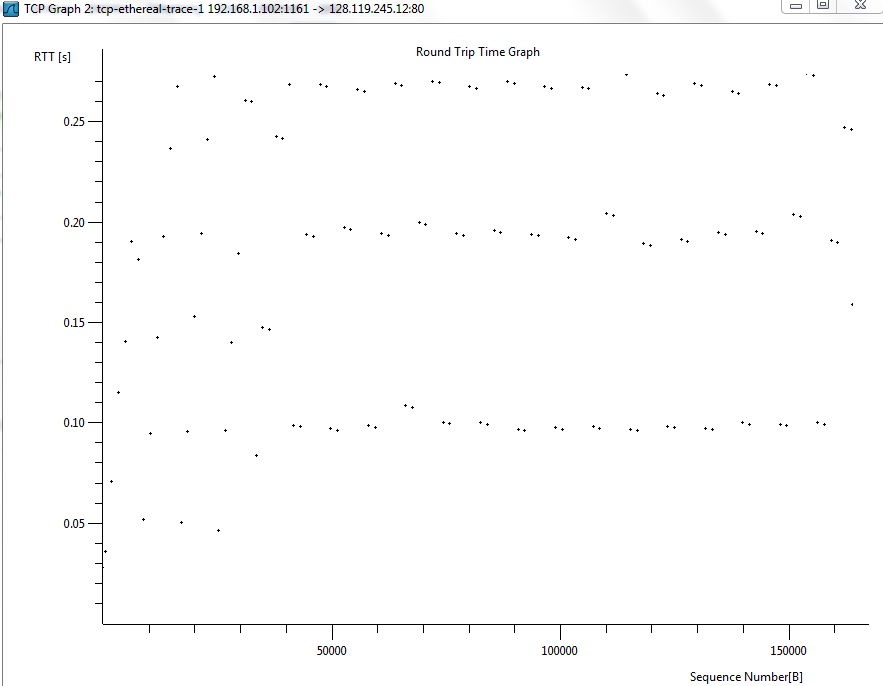
For second Ack Estimated RTT2= 0.125\*0.03557 + 0.875\*0.02746 =0.0285

For third Ack Estimated RTT3=0.125\*0.070059 + 0.875\*0.0285=0.0337

For fourth Ack Estimated RTT4= 0.125\*0.11443 + 0.875\*0.0337=0.0438

For the Fifth Ack Estimated RTT5= 0.125\*0.13989+ 0.875\*0.0438=0.0558

For the sixth ACK Estimated RTT6= 0.125\*0.18964+ 0.875\*0.0558=0.0725

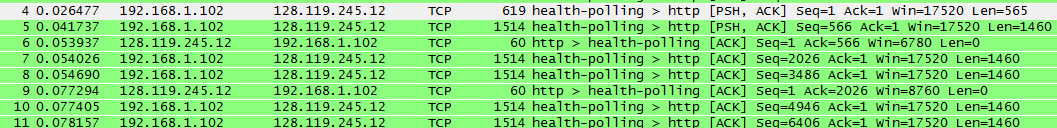


7. What is the length of each of the first six TCP segments?

Ans:

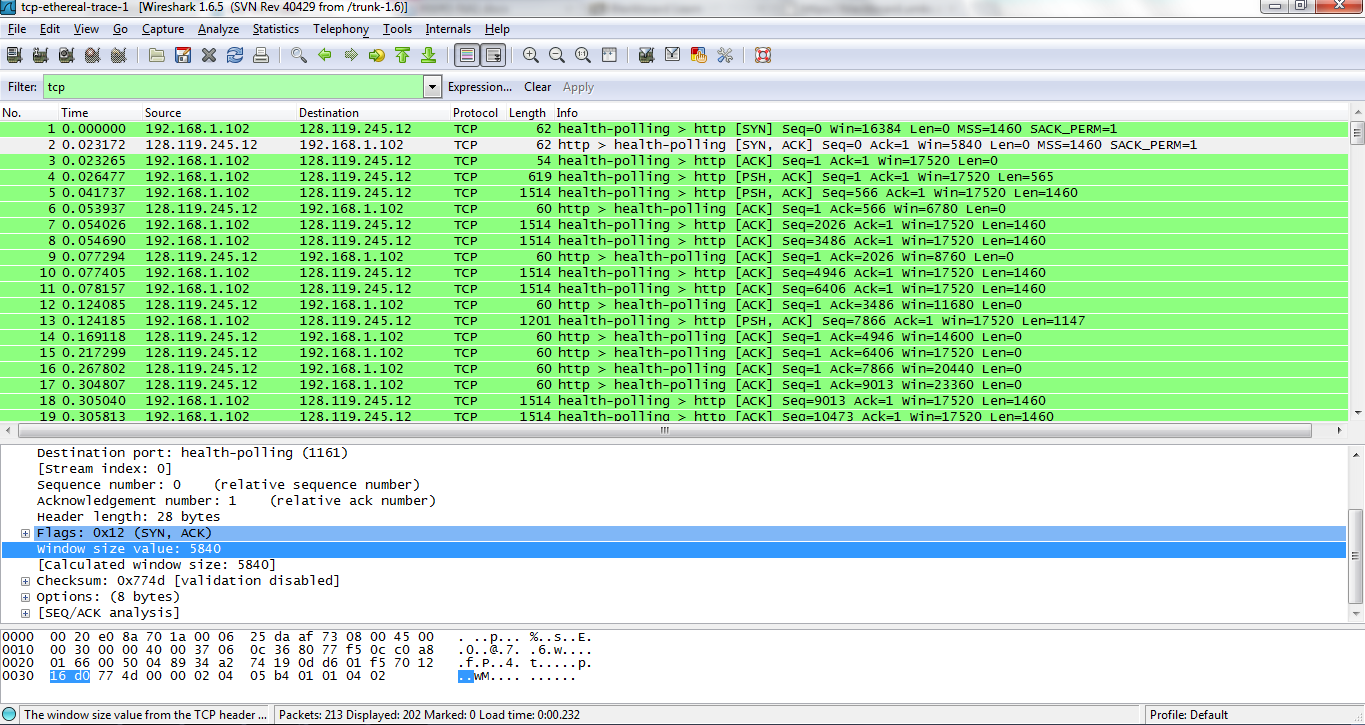
Length of first segment= 565

Length of remaining five segments= 1460 each



8. 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?

Ans:

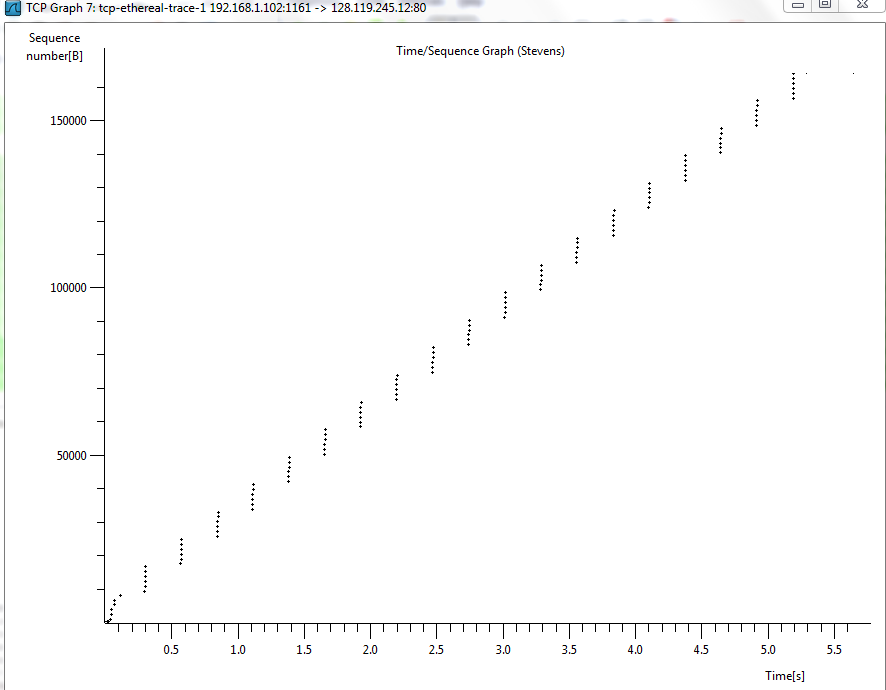


The minimum amount of available buffer space advertised is 5840 bytes. No, the lack of receiver buffer space did not throttle the sender.

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

Ans:

No, there are no retransmitted segments in the trace file. To answer this question I have checked the TCP stream Graph. If there were any retransmitted segments then there would be a change in size of dots in stevens graph. Moreover no sequence number is repeated. This also tells that there are no retransmissions.



10. How much data does the receiver typically acknowledge in an ACK? Can you identify cases where the receiver is ACKing every other received segment.

Ans:

The receiver acknowledges the data in an ACK which is equal to difference between the two consecutively acknowledged sequence numbers

Following are the cases in which receiver has acknowledged every other received segment

For the first ACK segment receiver acknowledges 565 bytes

From 2nd to 6th segment it acknowledges 1460 bytes.

11. What is the throughput (bytes transferred over time taken) for the TCP connection?

Explain how you calculated this value.

Throughput = (data transmitted) / (time taken for transmission)

Data transmitted = (164091 – 1) =164090 Bytes

Time taken for transmission= (5.45583 – 0.026477) = 5.429353 Sec

Therefore Throughput = 164090 / 5.429353

= 30.22 kbps