Mihai Dan

10/28/15

CS 372

Lab 3

**1.** Answer the following questions:

a) 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.

Source IP Address: 10.248.46.132

TCP Source Port: 54592

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

Destination IP Address: 128.119.245.12

TCP Destination Port: 80

c) What is the IP address and TCP port number used by your client computer (source) to transfer the file to gaia.cs.umass.edu?

I used my own trace for parts a) and b).

**2.** 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?

TCP SYN segment sequence number = 0

The Flag is set to 0x002, which depicts that this is a SYN segment.

**3.** 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?

SYNACK segment sequence number = 0

SYNACK Acknowledgement field value = 1

This value was determined based on the fact that the acknowledgement number is the ` next anticipated number from the sender. The Flag 0x012, meaning (SYN, ACK), is what identifies the segment as a SYNACK.

**4.** 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.

The sequence number of the TCP segment connecting the HTTP POST command is 1.

**5.** 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 239 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 239 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.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Sequence Number | Time Sent (sec) | ACK return time (sec) | RTT (sec) |
| 1 | 1 | 0.103215000 | 0.161002000 | 0.057787 |
| 2 | 576 | 0.105847000 | 0.211659000 | 0.105812 |
| 3 | 1836 | 0.105867000 | 0.212734000 | 0.106867 |
| 4 | 3096 | 0.105875000 | 0.212735000 | 0.106860 |
| 5 | 4356 | 0.205828000 | 0.240660000 | 0.034832 |
| 6 | 5616 | 0.208288000 | 0.290820000 | 0.082532 |

EstimatedRTT = ( 1 – 0.125 ) \* EstimatedRTT + 0.125 \* SampleRTT

EstimatedRTT of segment 1:

EstimatedRTT = 0.875\*(0.057787) + 0.125\*(0.057787) = 0.057787 seconds

EstimatedRTT of segment 2:

EstimatedRTT = 0.875\*(0.057787) + 0.125\*(0.105812) = 0.063790 seconds

EstimatedRTT of segment 3:

EstimatedRTT = 0.875\*(0.063790) + 0.125\*(0.106867) = 0.069175 seconds

EstimatedRTT of segment 4:

EstimatedRTT = 0.875\*(0.069175) + 0.125\*(0.106860) = 0.073885 seconds

EstimatedRTT of segment 5:

EstimatedRTT = 0.875\*(0.073885) + 0.125\*(0.034832) = 0.069004 seconds

EstimatedRTT of segment 6:

EstimatedRTT = 0.875\*(0.069004) + 0.125\*(0.082532) = 0.070695 seconds

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

1.) 629 bytes

2.) 1314 bytes

3.) 1314 bytes

4.) 1314 bytes

5.) 1314 bytes

6.) 1314 bytes

**7.** 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?

The minimum amount of available space buffer is synonymous to the Calculated window size field, which in this instance is calculated to be 262144 bytes.

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

There are no files that have been retransmitted. This can be inferred from the fact that each sequence number is sequential and there are no repeats.

**9.** 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 247 in the text).

The receiver typically acknowledges 1260 bytes of data in every ACK. This can be determined by looking at the ACK sequence numbers, which correspond with which byte of data was most currently transmitted. A example of the case where the receiver ACK’s every other received segment is shown by the acknowledgement number for packet number 27. The sequence number was 33336, which is 2520 bytes after the previous ACK, meaning it confirmed both of the 1260-byte segment in one ACK.

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

In order to calculate the throughput, the size of the file and time of transfer is needed.

We can calculate the size from the sequence number of the first TCP message to the last ACK message received. That is 152,901 – 1 = 152,900 bytes.

The time of the last ACK received was 0.62041000 seconds. The time of the initial message was 0.081907000 seconds, yielding a difference of 0.538503 seconds.

**11.** 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.

Below is the Time-Sequence-Graph(Stevens) as plotted with the Wireshark tool. The graph shows that TCP’s slowstart phase begins around 14 milliseconds and lasted until around 56 milliseconds into the data transfer. This graph is different from what you would expect from the ideal TCP slow start graph we have been studying in class for a couple reasons. In the ideal TCP graph, there are no errors or any delays in package delivery. An ideal connection would also have little to no congestion, which allows for smoother running and clearer results.

