Inge Becht 6096906

NS Lab Assignment
TCP Protocol

Chariklis Pittaras (c.pittaras@uva.nl)

Karel van der Veldt (k.vd.veldt@uva.nl)

Lab date: Nov. 20, 2012
Hand-in time (return via email) by Nov. 22, 2012 23:59CEST

Total points: 10 pts

Abstract

In this assignment we are going to investigate the TCP protocol using the Wireshark.

Preparation

We suggest you to review the sections 3.5 and 3.7 from the book.

Tasks

In this exercise we are going to analyze some TCP traffic, which is generated when you send a file to a web server. We are going to use the already Wireshark captured file *wireshark_trace-lab6*. In this trace file we captured the traffic, while a client computer was sending the file *rfc813.txt* to a server.

Task 1 – TCP Handshake and Statistics (2 pts)

Questions:

1. What are the IP addresses of the client computer and the server? What are the TCP source and destination port numbers, which were used by the client computer to send the file rfc813.txt?

   a. *Client IP:* 145.18.214.201

   b. Server IP: 195.169.124.10

   c. Source Port: 56653

d. *Destination Port:* 80

2. How many TCP segments and bytes the client sent and received to/from the server?

   a. Number of sent TCP segments: 32

   b. Number of sent Bytes: 40998

   c. Number of received TCP segments: 28

   d. *Number of received bytes:* 2533

3. **(a)** What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and the server? **(b)** What is it in the segment that identifies the segment as a SYN segment?

   a. SEQ = 0

   b. The SYN flag Is set to 1

4. **(a)** What is the sequence number of the SYNACK segment that was sent by the server to the client computer in reply to the SYN? **(b)** What is the value of the ACKnowledgement field in the SYNACK segment? **(c)** How did the server determine that value? **(d)** What is it in the segment that identifies the segment as a SYNACK segment?

   a. SEQ = 0

   b. ACK = 1

   c. previous SEQ of client was 0 and SYN segment is empy so ACK = 1

   d. Both SYN and ACK flags are 1

Task 2 – TCP Sequence Numbers (2 pts)

Questions:

5. **(a)** Find the first TCP segment (sent from the client to the server) that is after the first three handshake TCP segments. For this segment, and the next four

TCP segments that are sent from the client to the serve, fill the following table. **(b)** How the sequence number is calculated each time? (Tip: to see only the interesting TCP flow, select one of the three handshake TCP segments and next select *`Analyze->Follow TCP Stream`*. You may also find useful to use the tool *`Statistics->Flow Graph`*)

	Sequence Number	TCP segment Data (bytes)
Segment 1	1	1351
Segment 2	1352	1351
Segment 3	2703	1351
Segment 4	4054	1351
Segment 5	5405	1351

b. SEQ represents the first byte of a sent segment in view of the total bytes stream that is already sent. SEQ = previous SEQ + previous segment data

6. **(a)** Are there any retransmitted segments in the trace file? **(b)** What values did you check for (in the trace) in order to answer this question? (Tip: if you want to sort the displayed packets by a specific attribute, i.e. source or time, then just click on the specific column)

   a. There are no retransmitted segments

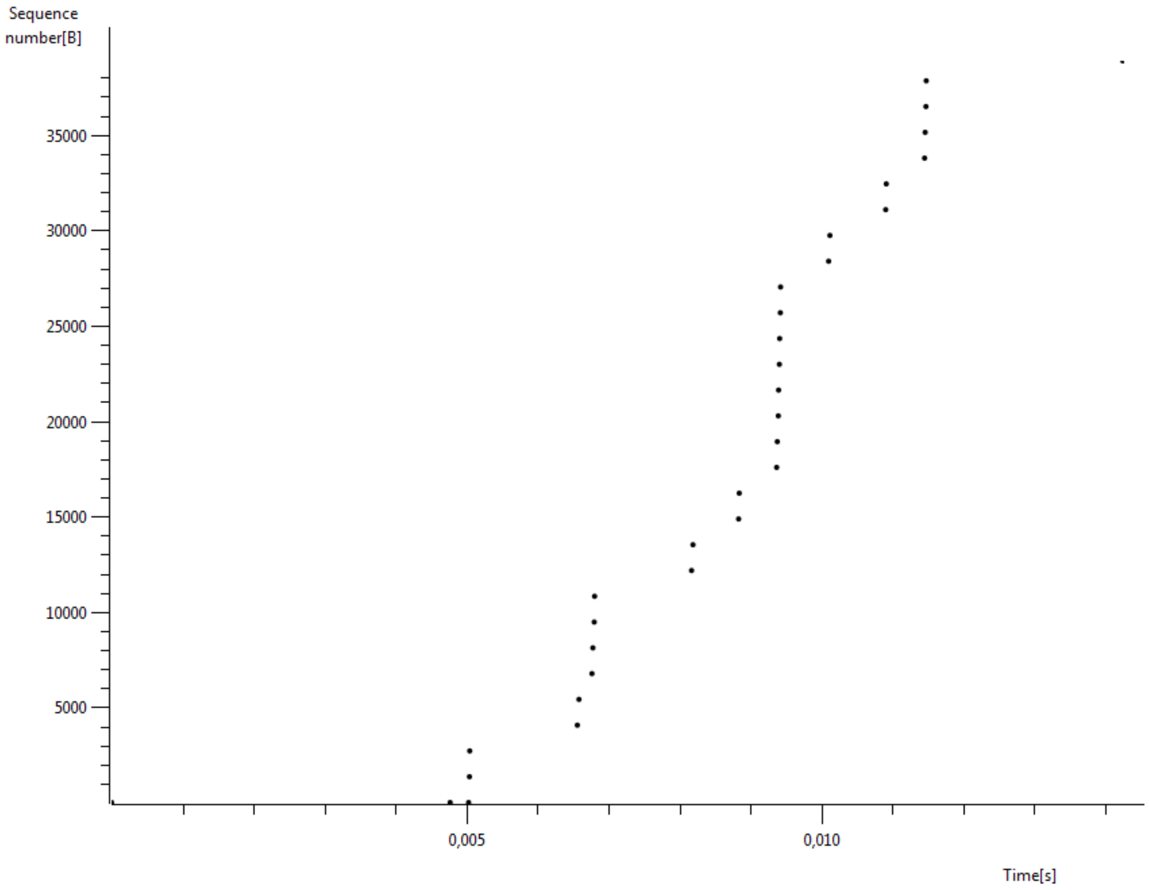
   b. I filtered the trace on tcp.analysis.retransmission and tcp.analysis.duplicate_ack, none of which gave a result.

      Also looking in the FlowGraph did't show any duplicate ACKS or resend segments.

7. Choose a TCP segment that was sent from the client to the server. Draw the Time-Sequence-Graph (Stevens) (*`Statistics-> TCP StreamGraph-> Time-`*

`Sequence-Graph (Stevens))` for this stream. Is the graph consistent with your answer in the previous question? Explain your answer.

Yes this is consistent. It is clear from the graph that segments are sent only ones, because each new segment increases the sequence number and never goes down to a previous sequence number. Only in the beginning when no data is yet sent the sequence number stays the same, and in the end when all data is sent.



Task 3 – TCP RTT (2 pts)

Questions:

8. Choose the first four segments from the question 5, at which time was each segment sent? When was the ACK for each segment received? Given the time when each TCP segment was sent, and when its acknowledgement was received, calculate the sample RTT value for each of the four segments. Fill this information in the following table. (Note: for time use *View->Time Display Format-> Seconds Since Beginning of Capture.* You can also use the Flow Graph tool but you should provide the time with all the available decimal digits and not rounded values)

	Sent time	ACK received time	SampleRTT (seconds)
Segment 1	8.625738	8.627205	0.001467
Segment 2	8.62575	8.627411	0.0016610
Segment 3	8.625754	8.627453	0.001699
Segment 4	8.62727	8.628823	0.0015530000000

9. What is the EstimatedRTT value (see page 265 in the book) after the receipt of each ACK? Assume that the value of the EstimatedRTT is equal to the measured SampleRTT for the first segment, and then is computed using the EstimatedRTT equation on page 265 for all subsequent segments. (Note: use α=0.125 as it is in the book). Fill the following table with your answers.

	EstimatedRTT	Calculation
Segment 1	0.001467	---
Segment 2	0.00149125	0.875 * 0.001467 + 0.125 * 0.0016610
Segment 3	0.00151721875	0.875 * 0.00149125 + 0.125 * 0.001699
Segment 4	0.00152169140625	0.875 * 0.00151721875+ 0.125 * 0.001553

Task 4 – TCP Congestion Control (4 pts)

Load the Wireshark trace file *wireshark_trace_lab6_congestion.* In this file, we captured the traffic while the client computer was downloading the *files install-tl.zip* from a server 1 in Russia and the file *koma-script.tar.xz* from a server 2 in Japan.

Questions:

10. Give the IP of the server #1 and #2, and in each case the protocol that the client used in order to download the file.

    a. *IP of server #1*: 195.128.64.25

    b. *Used protocol for server #1*: TCP

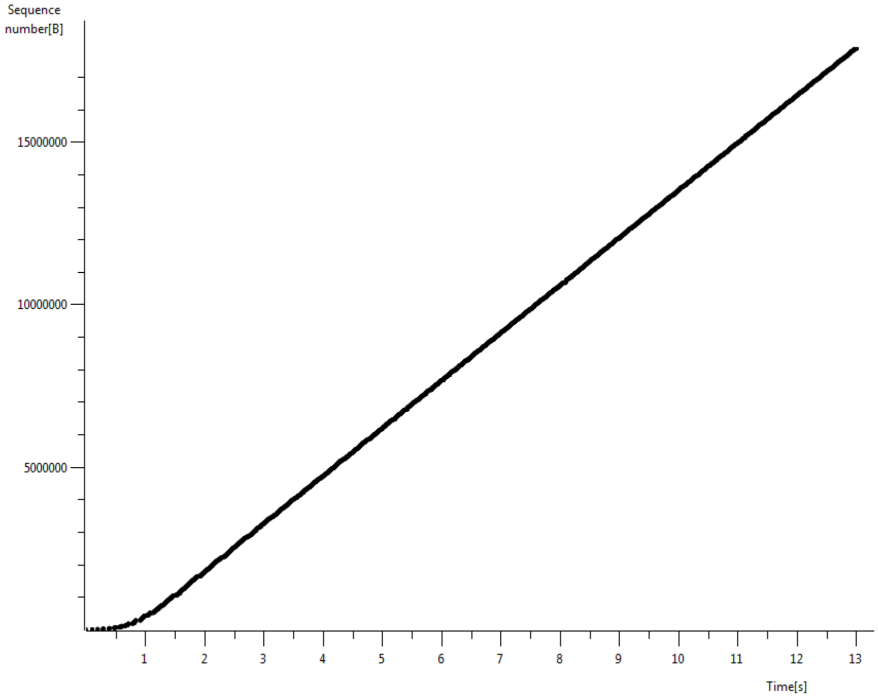
    c. *IP of server #2*: 150.65.7.130

    d. *Used protocol for server #2*: FTP

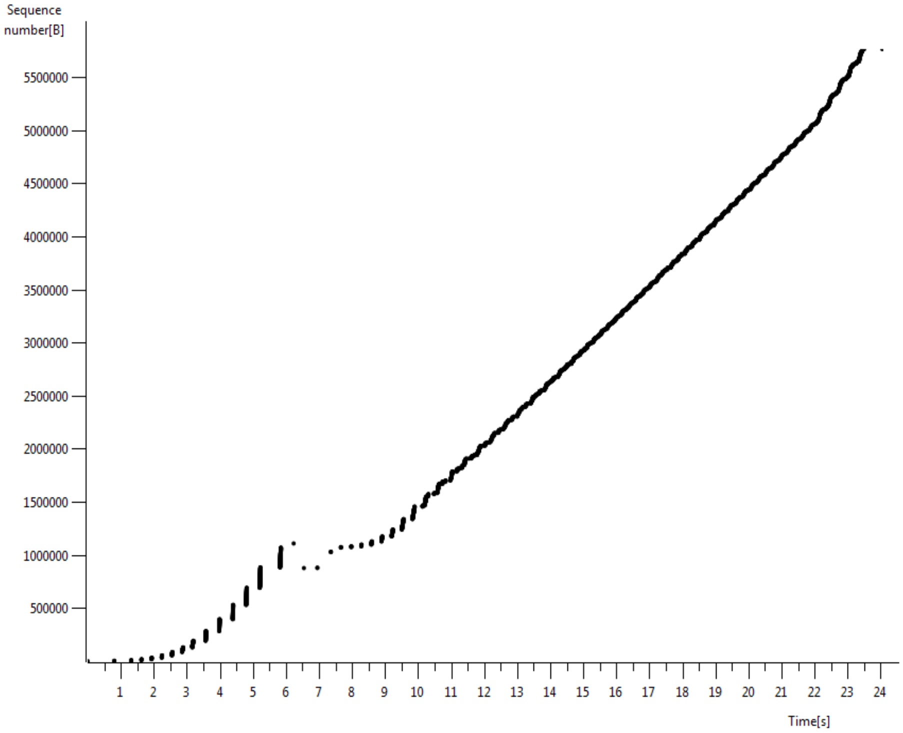
11. Plot the two *Time-Sequence Graphs (Stevens)* (*Statistics -> TCP StreamGraph*) of the data streams (used for downloading the files) from server #1 and server #2 to the client. **(a)** Explain what you see in the two cases, relating this to the theory of congestion control. **(b)** Why is there a difference

    a. We see that in case of server two a certain amount of segments gets dropped because the network is too congested to be able to deliver everything. The first server does not have this problem

    b. This difference could be because the congestion window of server 2 got too big,causing more segments being send than could be handled by the receiver.



server#1:195.128.64.25 → client



#2 server: 150.65.7.130 → client

12. Based on the previous graph for server #2, **(a)** how much data did the server send (approximately), up to the time of 10s, **(b)** and how much data did it send in the next 10s (Between 10s – 20s)? **(c)** Explain how you obtained these values.

    a. Around 1484201 bytes

    b. 4517761 - 1484201 = 3033560 bytes

    c. By clicking on the data point in the graph, which showed the segments that belonged to these points in the trace and reading out the sequence number (both points are far after all retransmissions so no missing data segments were used in these calculations)

13. Based on the graph of server #2 in question 11, and the trace file, explain what happened between approximately 6 – 8 second. Give all the necessary sequence numbers to justify your answer.

    Between 6 and 8 seconds some segments did not arrive at the client. This is seen from the many duplicate Acks that were send from the client to the server. An acknowledgement for 876041 was send over a hundred times to the server. The server responded with resending 3 segments with SEQ 876041, 877489, 878937 after these retransmissions another segment went missing, as seen by the client segment containing an duplicative ACK = 1026633. The server now directly sends segment with SEQ 1026633 and 1028081. The client now sends one more duplicative ACK with ACK = 1086825. The server now resends segments with SEQ=1029529 to SEQ = 1107721 and continuing as normal.

These answers were found by using the filter tcp.analysis.retransmission || tcp.analysis.duplicate_ack that shows only the duplicate acks and the retransmissions.

Submission

You have to submit:

- Your answers to all of the questions. Use this document for you answers and provide your answers in the appropriate answer field for each question.

- The Time-Sequence-Graph in question 7.

- The two Time-Sequence-Graphs in question 11.

Attention: You have to submit **one PDF** file that contains all the answers and graphs; the name of the file should be ***lab6-<lastnamefirstletter>.pdf*** *(example: lab6-vanderveldtk.pdf, or lab6-pittarasc.pdf)*.

Any other kind of submission will not be taken into account. You must also put your full name and your student number at the top of the file.