SE331: Introduction to Computer Networks	**Recitation 11**
Semester 1 5785	**22 Jan 2025**
Lecturer: Michael J. May	**Kinneret College**

TCP Threads and Analysis

1 Sentence Server

The first thing we will do is develop the sentence server application together. The server has the following properties:

- The client connects to the server based on a provided IP address and port
- The server listens on a provided IP address and port
- The server can accept and handle multiple clients at once
- The server uses a thread to handle each client.

We'll write the code together as part of the recitation.

2 TCP Trace Analysis

We will next analyze a few TCP download examples from Wireshark. The examples show how TCP behaves in a normal case and also some of the congestion control features that we saw in lecture today.

2.1 A First TCP Trace

Consider the first trace shown in `Simple-TCP-Conversation.pcapng.gz`. Using Wireshark, perform the following operations and answer the following questions:

1. Identify the three way handshake at the beginning of the conversation. Which packets are part of it?
2. Once you have found the first step of the three way handshake, use *Right Click* → *Conversation Filter* → *TCP* to show only the packets in the conversation. What are remote and local port numbers for the conversation?
3. Use *Right Click* → *Follow* → *TCP Stream* to show the whole TCP stream of the conversation. Can you tell what's going on in the HTTP part of the conversation?

Advertised Window

1. Find the advertised window values for the two sides of the conversation. What is the advertised window size for the server at the beginning of the conversation? What's the advertised window size for the recipient at the beginning of the conversation? Why are they not identical?

   Note: You'll notice the scaling factor value in the TCP header. You can find out more about the value in RFC 1323 (`https://tools.ietf.org/html/rfc1323#page-8`).
2. Find the header scaling factor field in the TCP options using the filter `tcp.options.wscale.multiplier`. In which packets are the values for the window scaling factor found?
3. Compare the values you found to the values of the advertised window size at the end of the conversation. Are they the same? Why?

4. Use *Statistics → TCP Stream Graphs → Window Scaling* to see a graphical representation of how the window scaling grows or shrinks over the course of the conversation. You can switch directions of the conversation using the “Switch Direction” button.

Congestion Control

1. Starting at packet 170 and continuing through 183, there is a series of packets which dropped. Find the indication that there were packets dropped from the server to the client (what field values show you that)?
2. Use the filter `tcp.options.sack` to identify packets with the SACK header options. What do the values in the SACK option header mean? Learn more about SACK in RFC 2018 (`https://tools.ietf.org/html/rfc2018`).
3. Use *Statistics → TCP Stream Graphs → Throughput* to see a graph of the throughput of the conversation. How do the packet drops during the conversation affect the throughput?

Closing

1. Identify the closing handshake for the conversation. Which side (client or server) decided to close first?

2.2 A Larger TCP Trace

Consider the TCP trace shown in the file “Recitation-TCP-Large-Download.pcapng”. Perform the steps above the new file. In this case, the conversation went on for much longer (larger file downloaded) and there’s no clean closing of the conversation at the end. The server can be found at 43.226.6.79, so you will want to use it as part of your conversation filter.

Produce the following graphs for the new trace:

1. *Statistics → TCP Stream Graphs → Throughput*
   - You can see the sawtooth pattern of the congestion control window very clearly on the graph.
2. *Statistics → TCP Stream Graphs → Window Scaling.*
   - In this graph, notice the steep rise in the advertised window in one direction, but the steady size in the other. What could be causing that?