CT2108 Lab - TCP Protocol Analysis

General

The objectives of this lab session are to investigate the behavior of the celebrated TCP protocol in detail. You will do so by analysing a trace of the TCP segments sent and received in downloading a large file from a server to your computer. Before starting this assignment, it is presumed that you know the foundations related to Wireshark, such as capturing and filtering packets. These were already covered in previous lab sessions.

Before beginning your exploration of TCP, you will need to use Wireshark to obtain a packet trace of the TCP transfer of a large file from a remote server to your computer. You will find some large test files here: http://www.thinkbroadband.com/download.html However, you can use any server you want for this lab work so long as the files are fairly big, 10MB is a reasonable size for testing. The actual content of the file does not matter at all for this lab work.

Now start up Wireshark and begin the packet capture as normal. Returning to your web browser, download a file to your computer. When the file has been fully downloaded stop Wireshark packet capture. Before analysing the behavior of the TCP connection in detail, let's take a high-level view of the trace. First, filter the packets displayed in the Wireshark window by entering "tcp" (lowercase, no quotes, and don't forget to press return after entering) into the display filter specification window towards the top of the Wireshark window. It may also help to use the Find Packet command from the Edit menu, as shown in the live class, to help search for and identify the HTTP GET Request for the actual download, you can then apply a display filter to only show packets related to that TCP stream using the Follow TCP Stream command from the Analyze menu.

You should see a series of TCP and HTTP messages between your computer and the server from where you downloaded the file. You should see the initial three-way handshake containing a SYN message. You should see an HTTP GET message. Depending on the version of Wireshark you are using, you might see a series of "HTTP Continuation" messages being sent from your computer to the server. In reality there is no such thing as an HTTP Continuation message, this is Wireshark's way of indicating that there are multiple TCP segments being used to carry a single HTTP message. In recent versions of Wireshark, you will see "[TCP segment of a reassembled PDU]" in the Info column of the Wireshark display to indicate that this TCP segment contained data that belonged to an upper layer protocol message (in our case here, HTTP). You should also see TCP ACK segments being exchanged between your computer and the server.

Where required, when answering a question or performing analysis, you can include a screen shot or print out of the packet(s) within the trace that you used to answer the question asked. To print a packet, use *File->Print*, choose *Selected packet only*, choose *Packet summary line,* and select the minimum amount of packet detail that you need to answer the question. Also choose *Output to File* to save the selected packet into a text file for pasting into a document if this is needed. You should see a series of TCP segments sent between your computer and the server. Now follow the instructions and try to answer each of the following questions:

1. What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between your computer and the server? What is it in the segment that identifies the segment as a SYN segment? Wireshark will display sequence numbers and ack numbers in both raw and relative format, make sure you understand the difference between these two formats, as explained in the live class.

2. What is the sequence number of the SYN, ACK segment sent by the server to your computer in reply to the SYN? What is the value of the Acknowledgement field in the SYN, ACK segment? How did the server determine that value? What is it in the segment that identifies the segment as a SYN, ACK segment?

3. What is the sequence number of the TCP segment containing the initial HTTP GET command? Note that in order to find the GET command, you'll need to dig into the packet content field at the bottom of the Wireshark window, looking for a segment with "HTTP" or something similar within its DATA field.

4. Consider the TCP segment containing the HTTP GET as the first segment in the TCP connection. What are the sequence numbers of the first four segments in the TCP connection (including the segment containing the HTTP GET)? At what time was each segment sent? When was the ACK for each segment received?

5. What is the length of each of the first four TCP segments received from the server? What is the typical amount of available buffer space advertised by your computer for the entire trace? Give some examples of this value. Does the lack of receiver buffer space ever seem to throttle the sender?

6. You should now examine the amount of data sent per unit time from the server to your computer. Rather than calculating this from the raw data in the Wireshark window, we'll use one of Wireshark's TCP graphing utilities - Time-Sequence-Graph(Stevens) to plot out data. Select a TCP segment from the server and then select the menu: Statistics->TCP Stream Graph-> Time-Sequence-Graph(Stevens). Each dot in the resulting graph represents a TCP segment sent, plotting the sequence number of the segment versus the time at which it was sent. Note that a set of dots stacked above each other represents a series of packets that were sent back-to-back by the sender. Can you identify where TCP's slow-start phase begins and ends, and where congestion avoidance takes over if at all? If necessary, you can select a portion of the displayed graph to zoom in for better detail.