COMP3234B Computer and Communication Networks

Lab 3: TCP Trace Analysis

Introduction

In this lab, we will investigate the behavior of the important TCP protocol in detail. We'll do so by analyzing a trace of the TCP segments sent and received in transferring a 406KB file (Agatha.pdf) from your computer to a remote server. We will study TCP's use of sequence and acknowledgement numbers for providing reliable data transfer; we'll look at TCP's receiver-advertised flow control mechanism; we'll also briefly consider TCP connection setup and we'll investigate the performance (throughput and round-trip time) of the TCP connection between your computer and the server.

Set up Wireshark Packet Sniffer

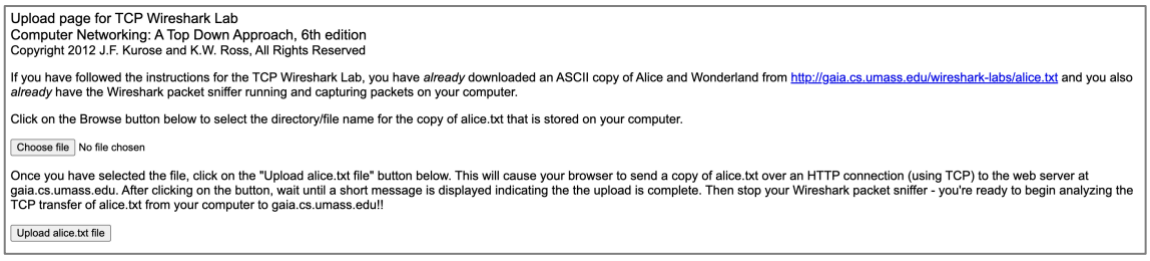
Download wireshark from https://www.wireshark.org/download.html and install the latest version for your operating system*. **Install ChmodBFP as well** (included in the package you have downloaded), in order to be able to capture packets.

Capture a bulk TCP transfer from your computer to a server

We will use Wireshark to obtain a packet trace of the TCP transfer of the file from your computer to a remote server. You'll do so by accessing a webpage that will allow you to enter the name of the file stored on your computer, and then transfer the file to a Web server using the HTTP POST method. We'll be running Wireshark during this time to obtain the trace of the TCP segments sent and received from your computer.

Do the following:

- Go to http://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html.
- You should see a web page that looks like:


Upload page for TCP Wireshark Lab
Computer Networking: A Top Down Approach, 6th edition
Copyright 2012 J.F. Kurose and K.W. Ross, All Rights Reserved

If you have followed the instructions for the TCP Wireshark Lab, you have *already* downloaded an ASCII copy of Alice and Wonderland from http://gaia.cs.umass.edu/wireshark-labs/alice.txt and you also *already* have the Wireshark packet sniffer running and capturing packets on your computer.

Click on the Browse button below to select the directory/file name for the copy of alice.txt that is stored on your computer.

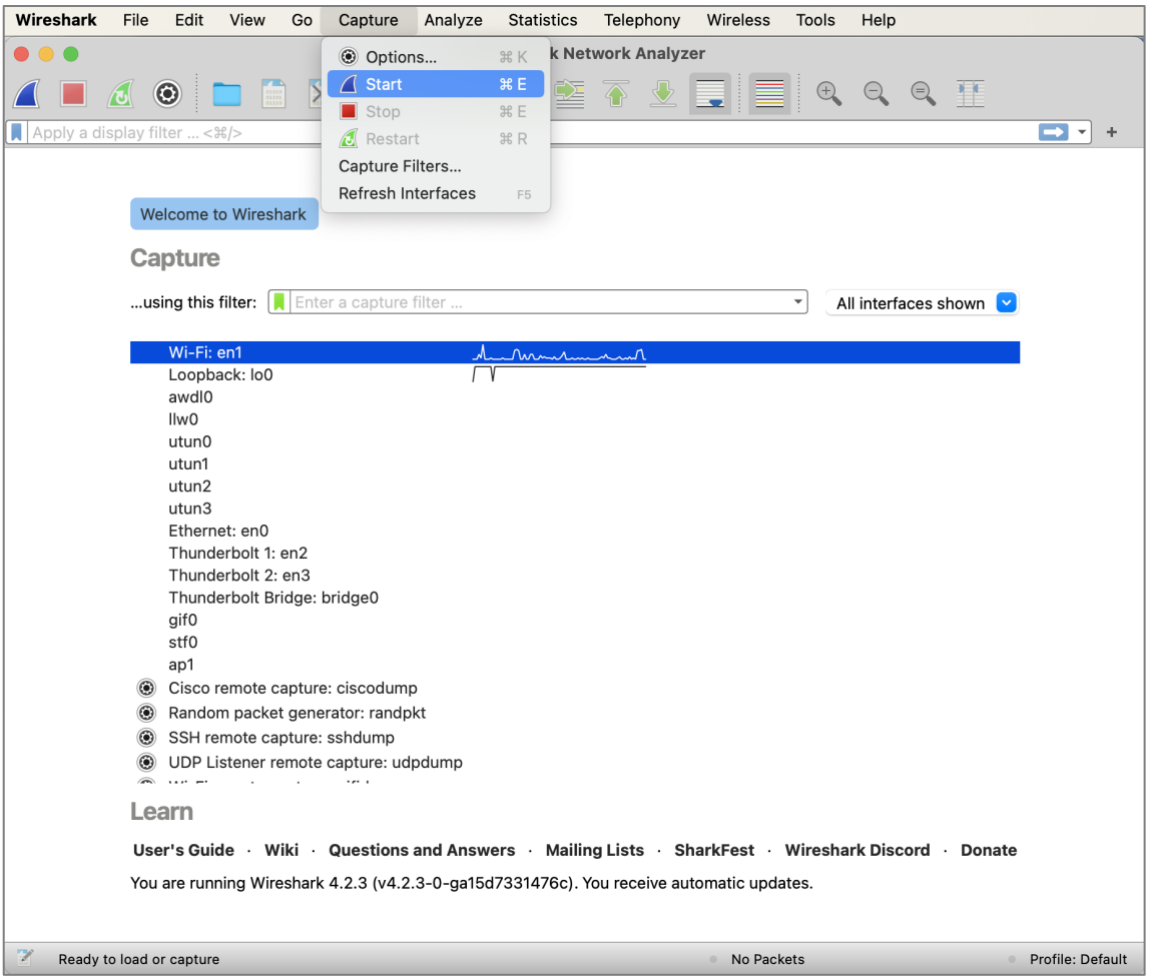
Choose file | No file chosen

Once you have selected the file, click on the "Upload alice.txt file" button below. This will cause your browser to send a copy of alice.txt over an HTTP connection (using TCP) to the web server at gaia.cs.umass.edu. After clicking on the button, wait until a short message is displayed indicating the the upload is complete. Then stop your Wireshark packet sniffer - you're ready to begin analyzing the TCP transfer of alice.txt from your computer to gaia.cs.umass.edu!!

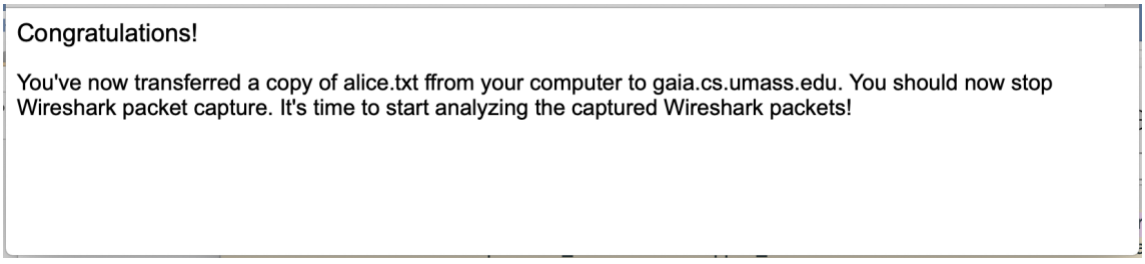
Upload alice.txt file

- Use the *"*Choose file*"* button in this form to select the file **Agatha.pdf** on your computer (we have provided Agatha.pdf on Moodle). Don't yet press the "Upload alice.txt file" button.

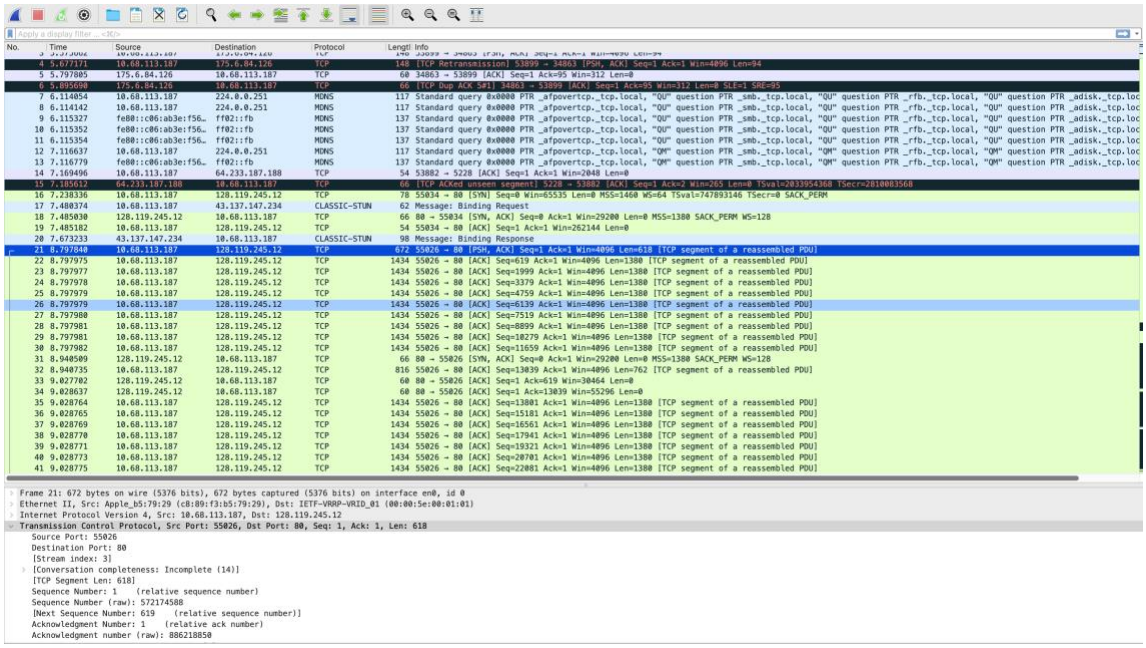
- Now start up Wireshark and begin packet capture *(in top menu bar: Capture->Start)*



- Returning to your browser, press the "Upload alice.txt file" button to upload the file to the gaia.cs.umass.edu server (though the button says uploading "alice.txt", we are uploading "Agatha.pdf" instead). Once the file has been uploaded, a congratulations page will be displayed in your browser window, as below:



- Stop Wireshark packet capture by clicking the red button as in the following screenshot. Your Wireshark window should look similar to the window shown below.



Before analyzing the behavior of the TCP connection in detail, let's take a high level view of the trace.

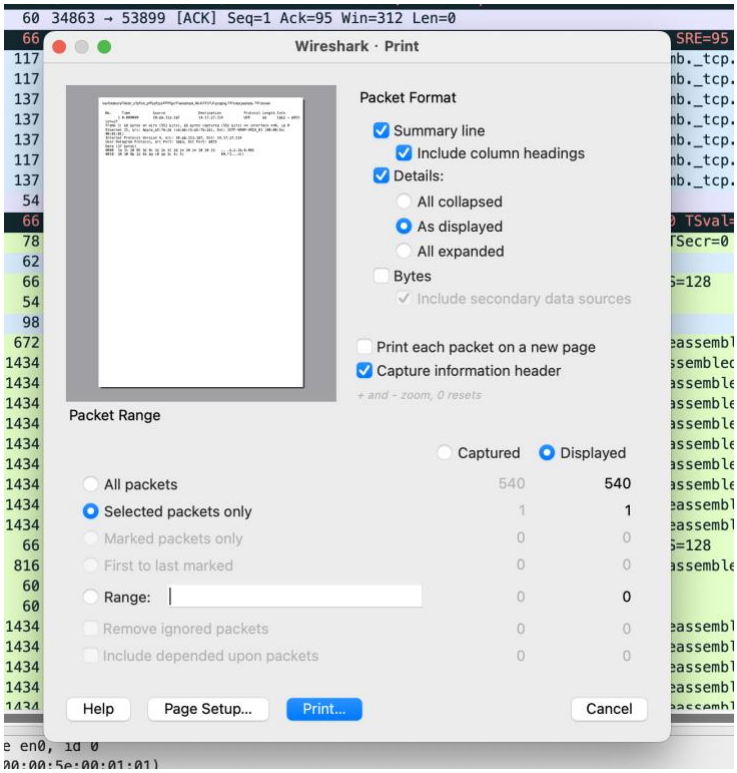
- Filter the packets displayed in the Wireshark window by entering "tcp" (lowercase, no quotes, and don't forget to press return after entering) into the "Apply a display filter filter" field towards the top of the Wireshark window.

- What you should see is a series of TCP messages between your computer and gaia.cs.umass.edu (128.119.245.12) plus some other TCP messages between your computer and other IP addresses (depending on what apps you are running on your computer). You can ignore the TCP messages with other hosts but focus on ones with 128.119.245.12. You should see the initial three-way handshake messages (SYN, ACKs).

- Now remove the filter of "tcp" from the "Apply a display filter" field (by clicking the "x" at the right end of the input field) and enter a new filter "http" there. You should be able to see a few HTTP messages (recall HTTP uses TCP), among which there is the HTTP POST message from your computer to the server, and its response from the server to your computer.

Lab exercises

Please submit screenshots or printout of the packet(s) within the Wireshark trace that you use to answer the questions below. Screenshots of the Wireshark window are usually easier to capture. If you prefer packet printout, you can 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, as in the following screenshot:



Note: for sequence numbers asked in the following questions, give both the relative and raw sequence numbers if you see both in the trace.

1. What are the IP address and TCP port number used by your client computer (source) to transfer the file to gaia.cs.umass.edu? To answer this question, select an HTTP message and explore the details of the TCP segment used to carry this HTTP message.

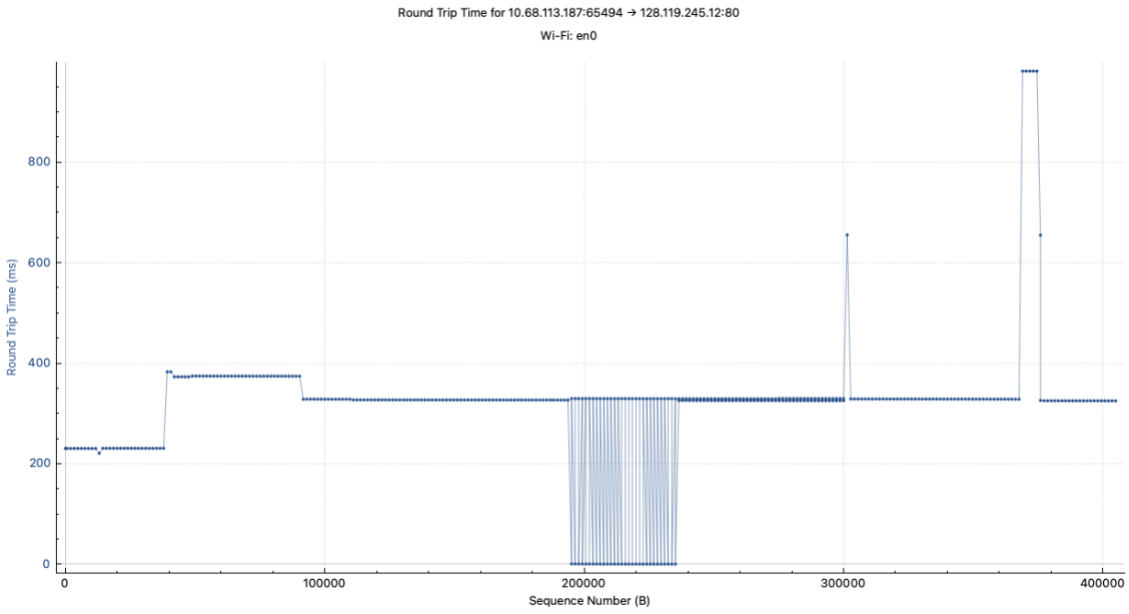
2. What is the IP address of gaia.cs.umass.edu? On what port number is it sending and receiving TCP segments for this connection?

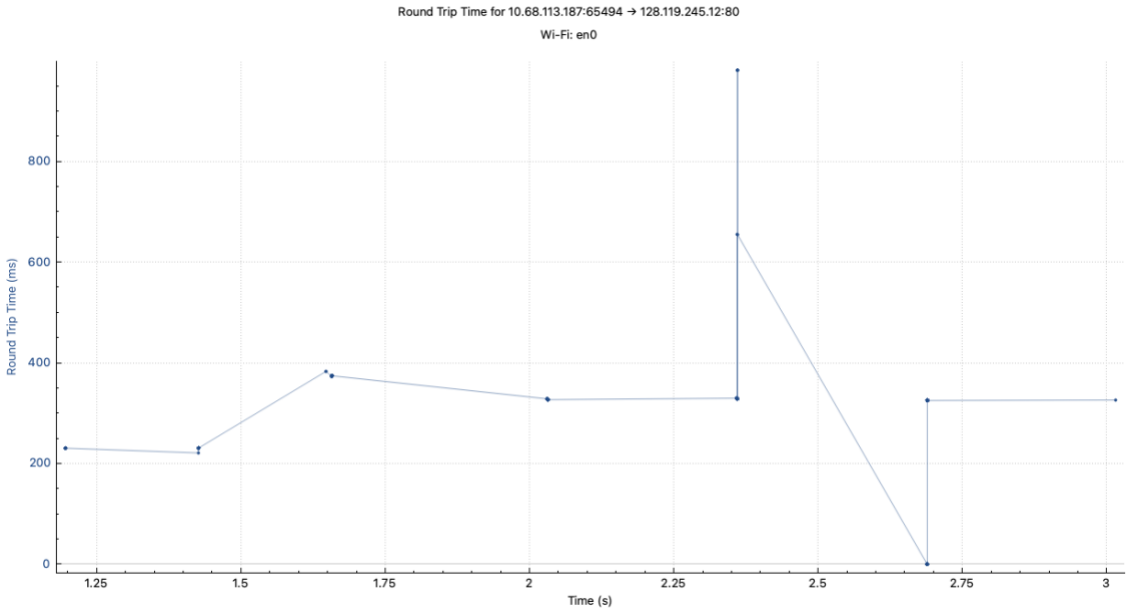
3. What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between your client computer and gaia.cs.umass.edu? How did you identify a segment as a SYN segment?

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 number field in the SYNACK segment? How did gaia.cs.umass.edu determine that value? How did you identify the segment as a SYNACK segment? One functionality of the SYNACK segment is for the server to notify the client the MSS that can be accepted on this connection. What is the value of the MSS indicated in the SYNACK segment? In which field of the SYNACK segment did you find it?

5. Since the MSS is smaller than the file size (406K Bytes), the HTTP POST message is partitioned and carried in multiple TCP segments. Note that the TCP segment you see when checking out the HTTP POST packet to answer question 1 above, is in fact the last TCP segment carrying the last partition of the HTTP POST message. Find out the first TCP segment carrying the first partition of the HTTP POST message. Starting from this first TCP segment containing the first partition of the HTTP POST message, what are the sequence numbers of the first six segments carrying the HTTP POST message? At what time was each segment sent? When was the ACK for each segment received? Give the difference between when each TCP segment was sent and when its acknowledgement was received (i.e., the RTT value) for each of the six segments. What is the EstimatedRTT value 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 21 of 8_TCP_II_COMP3234B_s2024.pdf 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 that is sent from the client to the gaia.cs.umass.edu server in the Wireshark window. Then select: *Statistics->TCP Stream Graph->Round Trip Time, you may need to click the "Switch Direction" for displaying*. See below as an example: (RTT by Sequence Number & RTT by Time)





6. What is the size of the payload in each of the first six TCP segments (those segments you found in question 5 above)?

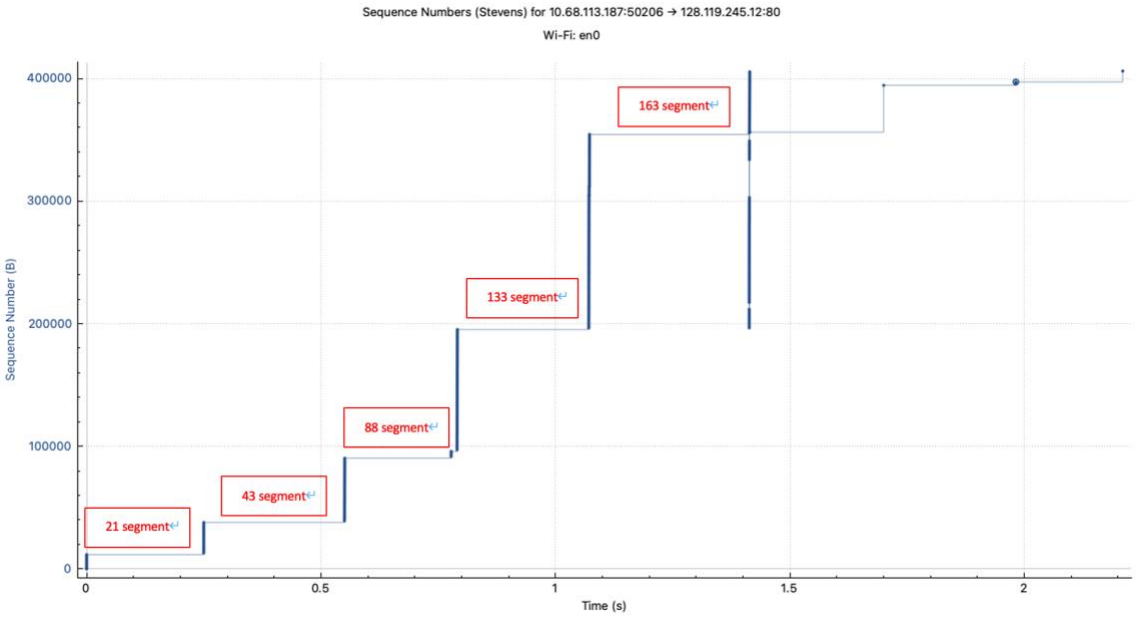
7. Read more about TCP's receive window advertisement at https://datatracker.ietf.org/doc/html/draft-scharf-tcpm-flow-control-quick-start-00 (Sec. 3.1 and 4.1), to understand more about the receive window size advertised by the gaia.cs.umass.edu server (receiver of the file) in the ACK segments it sends to your client computer (i.e., Window size value, Window size scaling factor, and Calculated window size which equals Window size value * Window size scaling factor). Then find out the minimum and maximum sizes of available receiver buffer space advertised by the server in the entire trace.

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

9. Give the amount of data acknowledged by the server in the first 6 ACKs that it sends to acknowledge receipt of partitions of HTTP POST message.

10. What is the throughput (application-layer data transferred per unit time) of the TCP connection, from the time when the client sends the first TCP segment carrying the HTTP POST message to the time when the ACK for the last TCP segment carrying the HTTP POST message is received from the server? Explain how you calculated this value. Note the application-layer data refers to the payload of the TCP segments, rather than payload of the HTTP POST message.

11. TCP congestion control in action
Select a TCP segment in the Wireshark window. Then select the menu : *Statistics->TCP Stream Graph-> Time-Sequence-Graph(Stevens)*. You should see a plot that looks similar to the following plot (without the # of segments marked):



Here, each dot 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. Try to plot a similar graph as the above in your wireshark. If you cannot obtain a similar graph as the above, you can answer the following questions based on the above given plot instead.

Can you estimate where TCP's slowstart phase begins and ends, and where congestion avoidance takes over? Think about why the measured data differs from the idealized congestion control behavior of TCP that we've studied in the text and briefly give the reason.

Submission:

Please put all your answers to the above questions in a word document and insert screenshots of your Wireshark window (or printout of packet information) where needed. Please convert the word document to a ***lab3-yourstudentid.pdf*** file and submit the PDF file on Moodle by 23:59 Monday March 11, 2024:

(1) Login Moodle.
(2) Find "Labs" in the left column and click "Lab 3".
(3) Click "Add submission", browse your .pdf file and save it. Done.

(4) You will receive an automatic confirmation email, if the submission was successful.
(5) You can “Edit submission” to your already submitted file, but ONLY before the deadline.