COMP416: Computer Networks
Project 2

Analysis of Transport Layer Protocols with Wireshark

Due: December 2, 2022, 11:59pm (Late submissions will not be accepted).

Submission of the project deliverables is via Blackboard.

This is an individual project. You are not allowed to share your code and answers.

This project is about **transport layer** of the network protocol stack.

The focus is on the **SSL, TCP, and UDP protocols**. For this purpose, you are asked to modify the provided SSL client/server codes, implement TCP client/server codes for delay comparison as specified below, experiment with TCP, and explore UDP features. You should use the **WireShark network protocol analyzer** tool to answer transport layer-related questions.

WireShark is the world's foremost network protocol analyzer and is the **de facto standard** across many industries and educational institutions. It can be downloaded freely at **http://www.wireshark.org/download.html**. WireShark allows users to trace the network activity by capturing all the packets that hit your network interface. It tags the information of each layer by parsing the given byte stream according to the corresponding protocol.

You should read this document carefully before starting your tasks.

In your answers, you are required to provide Screenshots of Wireshark capture highlighted/filtered appropriately supporting your results.

Part 1. UDP Experiments

In this part, you are assigned a **unique** URL to work with. The list is provided in the file Individual_URL_List.pdf, and you must use the URL assigned to you. You should provide the appropriate

screenshots and work on the correct domain to get credit. *nslookup* command works as an IP address resolver. When you provide a domain name as an argument, it will return the IP address of that domain. Now, take the following steps provided then answer the questions accordingly.

- Start your Wireshark software and start capturing packets from the appropriate interface.
- Use nslookup command to resolve the IP address of the URL that is assigned to you.
- Stop packet capturing in the Wireshark and apply the appropriate **display filter**.

1. How many queries are shown in Wireshark in response to a single nslookup command? If more than one, what could be the reason?

2. What does the stream index for a UDP packet signify in Wireshark?

3. Observe the Flowgraph for the UDP messages. Which IP address is the domain name being resolved? Can you apply domain name resolution on that IP address and perform a quick internet search about the DNS server?

4. Observe the flags under the DNS header. What is the purpose of the Authoritative and Recursion flags?

5. What are the types of DNS Records (name them in the report, but you may be asked about their significance during the demo)? How can you specify the type of DNS Record when using the ‘nslookup’ command? Share the results using the nslookup with any ‘3’ DNS Record Types.

Part 2. TCP Experiments

Before beginning the exploration of TCP, you need to use Wireshark to obtain a packet trace of the TCP transfer of a file from your computer to a remote server. You are asked to do so by accessing a Web page that will allow you to enter the name of a file stored on your computer (which contains the ASCII text of *Alice in Wonderland*) http://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.htm then transfer the file to a Web server using the HTTP POST method. We are using the POST method rather than the GET method as we would like to transfer a large amount of data *from* your computer to the server. Of course, you need to run Wireshark during this time to obtain the trace of the TCP segments sent and received from your computer.

Perform the following, then answer the related question:

- Start up your web browser. Go to the http://gaia.cs.umass.edu/wireshark-labs/alice.txt and retrieve an ASCII copy of *Alice in Wonderland*. Store this file somewhere on your computer.
- Next, go to http://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html. You should see a screen like in Figure 1.
- Use the *Browse button* in this form to enter the file's name (full path name) on your computer containing *Alice in Wonderland* (or do so manually). Do not yet press the *"Upload alice.txt file"* button.
- Now start up Wireshark and begin packet capture *(Capture->Start)* and then press *OK* on the Wireshark Packet Capture Options screen (we will not need to select any options here). ▪ Returning to your browser, press the *"Upload alice.txt file"* button to upload the file to the gaia.cs.umass.edu.server. Once the file has been uploaded, a short congratulations message will be displayed in your browser window.
- Stop Wireshark packet capture.

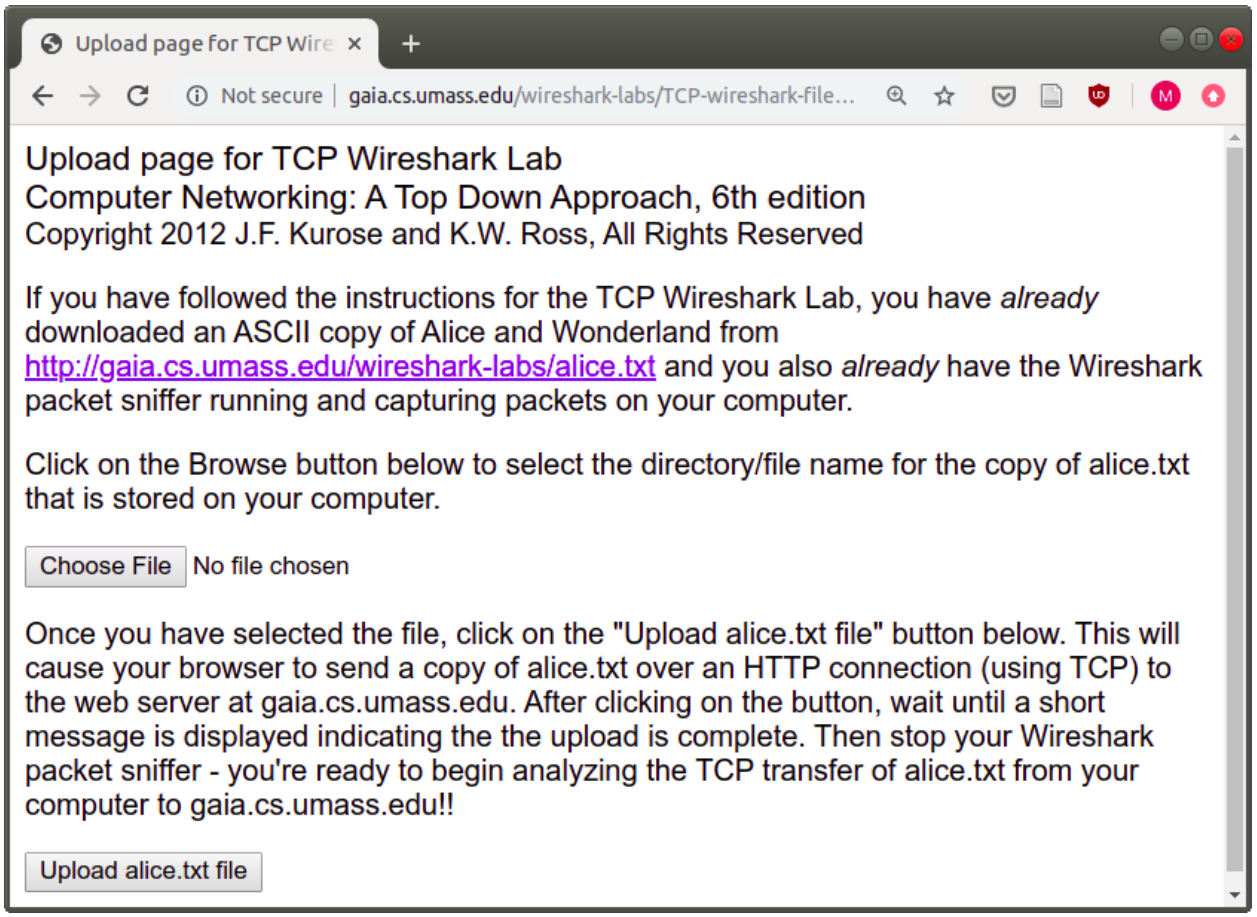

Figure 1. Text File upload Screen

Answer the following questions for the TCP segments.

6. Use the Flow graph options under Statistics in Wireshark and identify the major sequences (handshake, termination, data exchange, etc.) concerning packets involved in exchanging information. Use the following format

Start-End time	Source Socket	Destination Socket	Sequence Number	Number of Packets in the stage	Stream ID

7. What does the stream index in the TCP header signify? Are the packets being transmitted during the experiment all belonging to the same stream index? What does the same or different stream index mean in the context of this experiment?

8. Print the RTT graph for the entire communication. What is the average RTT value for the entire communication sequence?

Note: Wireshark has a nice feature that allows you to plot the RTT for each TCP segment sent. Select a TCP segment in the "listing of captured packets" window sent from the client to the gaia.cs.umass.edu server. Then select: *Statistics->TCP Stream Graph- >Round Trip Time Graph*.

9. Explain and provide the values for the following fields in the TCP packets:

   (a) TCP Segment Length

   (b) Sequence Number and relative Sequence Number

   (c) Acknowledgement Number and relative Acknowledgement number

   (d) Nonce Flag

   (e) Congestion Window Reduce (CWR) Flag

Part 3. A. SSL Implementation and Experiments

Recall that you are provided an SSL client/server code that performs echo on top of an SSL socket. The corresponding SSL practical content codes and slides are available through the course website.

The server sends the certificate to the user at the start of the session. The user adds this certificate to the local key store for authentication.

In this part, you are asked to modify the code for SSL demonstration:

- Establish an SSL socket connection between the client and the server.
- Second, your client should receive a randomly generated string of 16 characters one by one in separate messages in a non-persistent manner and print each character it receives in every new connection. Then, it should print the final merged string.

Important Notes:

- Your modified SSL codes must be submitted along with your project report. You should explain your answers in your project report and provide your WireShark outputs for each question to get credit.
- On Windows operating systems, you might not be able to capture the Loopback interface, which is the traffic inside your operating system. You need to capture the Loopback interface when you run your server and client software in a single operating system to capture incoming and outgoing packets.

Execute Wireshark and start packet capture before running the code to capture all the traffic. Making sure that no other data traffic occurs during this time might help filter out the corresponding packets. After running your code and stopping packet capture, answer the following questions based on the captured packets:

10. What cipher suite set does the client send? Which cipher does the server choose for the remainder of this communication?
11. Observe the packet showing the certificate transfer. Check under the SSL header. How many certificates does the server send during the Handshake, and why?
12. Find the message from the Server Hello group containing the session ID. What is the session ID in the corresponding message? What is the purpose of specifying a session ID?

Part 3.B. SSL vs. TCP: Delay measurements

In this part, you are asked to evaluate the performance of SSL and TCP using the time delay between the client generating a request and receiving the answer as a metric. This part can be performed using the sample TCP codes provided in the first practical content and the SSL code provided for this project and using Wireshark for packet capture. You will need to perform the following for both TCP and SSL.

First, the client will send the character “1” to the server as a query, and as a response, the server will send a random string of 16 characters in a persistent manner back to the client. Measure the time delay between sending the request and receiving the server response when communicating over a TCP socket vs. an SSL socket.

Second, the client will send the character “2” to the server as a query, and as a response, the server will send a randomly generated string of 16 characters back to the client in a non-persistent fashion. Measure the time delay between the request and receiving the server’s final character response when communicating over a TCP socket vs. an SSL socket.

13. Report both delays for 5 different executions and present the measurements as a single graph. Briefly describe the reasons for the results you have obtained.

14. Find the TCP Handshake and Terminate Messages. Do these messages have the stream index? Why or why not?

Project Deliverables:

Important Note: You are expected to submit a project report, in PDF format, that documents and explains all the steps you have performed to achieve the assigned project tasks. A full-grade report details and illustrates the execution of the project. Anyone who follows your report should be able to reproduce your performed tasks without effort. Use screenshots to explain the steps and provide clear and precise textual descriptions. All reports would be analyzed for plagiarism. Please be aware of the Koç University Statement on Academic Honesty.

The name of your project .zip file must be <surname>-<KUSIS-id>.zip
You should turn in a single .zip file including

- Source codes: Containing the source codes of client and server for both SSL and TCP.

- Project.pdf file (**Your report should include answers and the corresponding Wireshark screenshots for each answer.**)

- Saved capture files from the Wireshark.

Figures in your report should be scaled to be visible and clear enough. All figures should have captions, be numbered according to their order of appearance in the report, and be referenced clearly in your answer text. All pages should be numbered and have headers the same as your file naming criteria.

If you employ any (online) resources in this project, you must reference them in your report.

There is no page limit for your report.

Good Luck!