KOÇ
ÜNİVERSİTESİ
COMPUTER ENGINEERING

COMP416: Computer Networks
Project 2
Transport Layer Protocols' Experiments with Wireshark

Due: December 4, 2023, 11:59 pm (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 codes and answers with each other.

This project is about the **Transport Layer** of the network protocol stack.

The focus is on the **UDP, TCP, and SSL** protocols. You should use the **WireShark network protocol analyzer tool** to answer each question. The project has two parts, the first part is about UDP and TCP experimentation. The second part focuses on the comparison of TCP and SSL.

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 from here. 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 1.1. UDP Experiment

In this part, you are given a sample trace called udp_sample.

Start your Wireshark software and filter for all packets of UDP protocol. Your UDP segment for the following questions should be the one that has the same number as your student ID's last two digits.

Answer the following questions for your selected UDP segment.

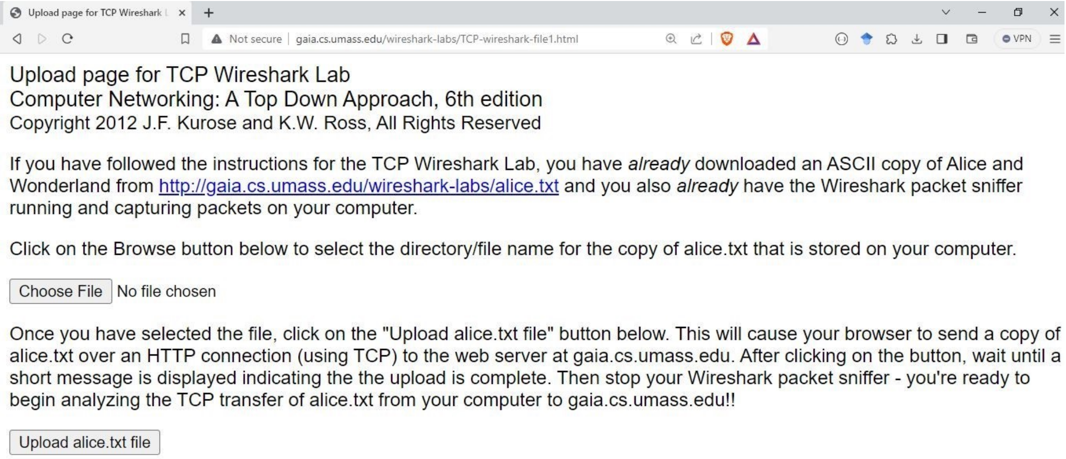
1. What is the segment's time to live value, numerically? What does that mean? What may happen if TTL reaches to zero before the segment arrives its destination?
2. What does the stream index for a UDP segment signify? What is your segment's stream index?
3. What is the checksum value of the segment? What would happen if the value was different than what you have observed? Is there a similar application for the same functionality in TCP? If so, what are the differences in between?
4. Observe the flags under the IP v4 header. What is the purpose of the Reserved bit flag? What is the value of your segment's reserved bit flag?
5. Explain the Length field. What does this value signify? Provide your packet's length.

Part 1.2. TCP Experiment

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 here that will allow you to enter the name of a file stored on your computer (which contains the ASCII text of Alice in Wonderland) 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 questions:

- Start up your web browser.
- Go to here and retrieve an ASCII copy of *Alice in Wonderland*. Store this file somewhere on your computer.
- Next, go to here. 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.


Upload page for TCP Wireshark L
Not secure
gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html
VPN

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

Figure 1. Text File Upload Screen

Answer the following questions for the TCP segments. For questions requiring a UDP segment, refer to your segment at Part 1.1. Provide screenshot for all questions.

1. Can you identify any segments marked as retransmissions? Are there indications of segment loss in the TCP flow? (Hint: Use post-filters.)
2. What is the round-trip time for a specific TCP connection? Are there variations in RTT over the course of the communication? (Hint: Use the "Statistics" menu.)
3. Identify the information that was present in the TCP but not in UDP. What do they signify? What is their purpose?
4. Observe the TCP and UDP segments' flow graphs from the Statistics menu. Explain what is being sent as data and their meanings. Underline the differences.

Part 2.1 TCP vs SSL Experiments

In this part of the project, you will analyze the packets of TCP and SSL messages and compare them to each other. You are asked to evaluate the performance of both protocols using the time delay between the client generating a request and receiving the answer as a metric. You will be using Wireshark for packet capture.

You are going to design a server that has two echo services, an insecure and secure one. The insecure service will be using a TCP socket and the secure one will be using an SSL socket. The server runs both services simultaneously and can serve more than one client at the same time. The client you are going to design can connect to the secure or the insecure service based on the choice of the user at the beginning of the connection to the server. Figure 2 shows an overview of the system.

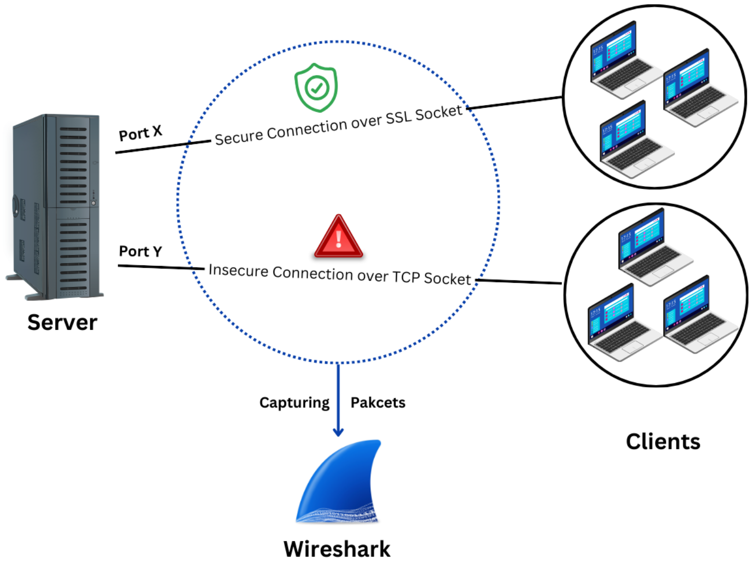

Port X
Secure Connection over SSL Socket
Port Y
Insecure Connection over TCP Socket
Server
Capturing
Pakcets
Clients
Wireshark

Figure 2: Overview of the components of TCP vs SSL Experiment

Run your server and two clients. The first client will be sending an echo request through the insecure channel, and the second client will be sending over the secure one. Make each client send your Student ID concatenated with COMP416, character by character. Capture the packets sent by each client and save the captured packets separately. After you save the captured packets for each client, answer the following questions as a comparison between the clients (i.e. a comparison between TCP and SSL) while providing brief explanations for the results.

1. What are the used protocol(s) in the captured packets?
2. Can you find the sent data (the characters) within the sent packets? Show that as screenshots when possible.
3. By looking at the received response in the packets, can you understand what is sent? Show that as screenshots when possible.
4. Report delays for 5 different executions and present the measurements as a single graph. Briefly describe the reasons for the results you have obtained. Measure the delays in code (Java).

   Answer the following questions considering only the secure channel (SSL) execution:

5. In the observed packets, you can see “Client Hello” and “Server Hello ...” packets. Briefly explain what those packets are and write your own observations.
6. How does the protocol avoid sending the certificate for each connection? Show the used solution from the captured packets.

Part 2.2. Creating Keystore and Trust Store

In part 2.1, you used a certificate for the server, with a keystore and trust store provided. In this part, you are asked to generate a keystore, a certificate, and a trust store.

Please note that you are **not** required to use the keystore and trust store you are going to generate in your code.

To create the keystore, the certificate, and the trust store, follow the steps provided in the following link: https://docs.oracle.com/cd/E19509-01/820-3503/ggsxx/index.html. Make sure that you put your name and surname as the certificate author, other fields can be filled according to you. In your report, put a screenshot of the information of the generated certificate where the screenshot has to show your name and surname clearly. In addition, put the passwords you used to generate the key store and trust store.

Project deliverables

Important Notes:

- 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.
- 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

- The server and client projects.
- Project.pdf file (**Your report should include answers and the corresponding Wireshark screenshots for each answer.**)
- Saved capture files from Wireshark.
- The generated certificate, keystore, and trust store.

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!