CSc 361: Computer Communications and Networks (Fall 2024)

Assignment 2: TCP Traffic Analysis

Final Due: 11:59 pm, Nov. 1, 2024

1 Goal

The purpose of this project is to understand the details of state management in Transmission Control Protocol (TCP). You are required to write a python program to analyze the TCP protocol behavior.

2 Requirements

You will be given a *sample* TCP trace file (sample-capture-file.cap). During the period traced, a single web client accesses different web sites on the Internet. This trace is to be used for your own test. TA might use a different trace file to test your code.

You need to write a python program for parsing and processing the trace file, and tracking TCP state information. In this assignment, your code will be tested on the server *linux.csc.uvic.ca.* As such, you are allowed to use **only the Python packages of python3 currently installed** on *linux.csc.uvic.ca.* You are not allowed to install/use other third-party python packages.

Your program should process the trace file and compute summary information about TCP connections. Note that a TCP connection is identified by a 4-tuple (IP source address, source port, IP destination address, destination port), and packets can flow in both directions on a connection (i.e., duplex). Also note that the packets from different connections can be arbitrarily interleaved with each other in time, so your program will need to extract packets and associate them with the correct connection.

The summary information to be computed for each TCP connection includes:

- the state of the connection. Possible states are: S0F0 (no SYN and no FIN), S1F0 (one SYN and no FIN), S2F0 (two SYN and no FIN), S1F1 (one SYN and one FIN), S2F1 (two SYN and one FIN), S2F2 (two SYN and two FIN), S0F1 (no SYN and one FIN), S0F2 (no SYN and two FIN), and so on, as well as R (connection reset due to protocol error). For consistence, we count a SYN+ACK segment (i.e., a segment with both SYN bit and ACK bit set to 1) as a SYN message. (Of course, a SYN segment is also counted as a SYN segment). Getting this state information correct is the most important part of your program. We are especially interested in the complete TCP connections for which **we see at least one SYN and at least one FIN**. For these complete connections, you can report additional information, as indicated in the following.

- the starting time, ending time, and duration of each complete connection

- the number of packets sent in each direction on each complete connection, as well as the total packets
- the number of data bytes sent in each direction on each complete connection, as well as the total bytes. This byte count is for data bytes (i.e., excluding the TCP and IP protocol headers).

Besides the above information for each TCP connection, your program needs to provide the following statistical results for the whole trace data:

- the number of reset TCP connections observed in the trace
- the number of TCP connections that were still open when the trace capture ended. (If a TCP connect has no **data** segment after FIN, **we consider** it closed. Otherwise, **we consider** it open.)
- the number of TCP connections established before the capture started. (If a TCP connection's first segment is not SYN, **we consider** it established before the trace capture.)
- the number of complete TCP connections observed in the trace
- Regarding the complete TCP connections you observed:
  - the minimum, mean, and maximum time durations of the complete TCP connections
  - the minimum, mean, and maximum RTT (Round Trip Time) values of the complete TCP connections
  - the minimum, mean, and maximum number of packets (both directions) sent on the complete TCP connections
  - the minimum, mean, and maximum receive window sizes (both sides) of the complete TCP connections.

As a guideline for output format, please follow the output format of this project shown in outputformat.pdf.

3 Deliverables and Marking Scheme

For your final submission of your assignment, you are required to submit your source code. You should include a readme file to tell TA how to run your code.

Zip your assignments (code) as one tar file using `%tar -czvf` on linux.csc.uvic.ca.

The marking scheme is as follows (refer to outputformat.pdf as well):

Components	Weight
Total number of connections	25
Connections' details	30
General Statistics	20
Complete TCP connections:	20
Readme.txt, code style	5
Total Weight	100

4 Plagiarism

This assignment is to be done individually. You are encouraged to discuss the design of your solution with your classmates, but each person must implement their own assignment.

The End