Programming Assignment 2: Analyzing packet traces (PCAP)
CSE 310, Spring 2022
Instructor: Aruna Balasubramanian
Due date: March 10 2022, 9.00pm

The goal of this assignment is to dissect TCP packets. To do this, you should be familiar with the packet formats (PCAP files).

Specifically, your goal is to parse a PCAP file. PCAP is the file format used to store packets captured on the wire. PCAP files are in binary format and cannot be read directly. A PCAP library is used to parse the binary packet. Your goal is to write a parser that analyzes the packet with the help of the PCAP library.

TCPdump is the command-line tool that also analyzes the packets captured on the wire. Wireshark is the graphical version of TCPDump. You can check out these tools to see some examples of packet analysis if you want.

Part A PCAP Programming Task and flow-level information (70 points)

Your task is to write a program *analysis_pcap_tcp* that analyzes a PCAP file to characterize the TCP flows in the trace. A TCP flow starts with a TCP "SYN" and ends at a TCP "FIN" between two hosts. A TCP flow is uniquely identified by the tuple: (source port, source IP address, destination port, destination IP address). There can be multiple TCP flows at the same time between the two hosts, on different ports.

You can use a PCAP library to analyze this file. Example PCAP libraries are provided at the end of this assignment. A PCAP library helps convert a PCAP packet from binary to byte format. You need to then write code to analyze the bytes to get the information about the packet.

[**Important**: You can create your own packet structures and read the bytes into the structure. This will let you easily parse the bytes rather than doing byte operations. You can also use the ethernet and TCP modules in the PCAP library to get these packets. However, you cannot convert the PCAP file into text for analysis.]

Specifically, we have captured packets that are going on the wire---both packets from the computer and to the computer. This packet capture is in PCAP format and called *assignment2.pcap* in the resource section. In this file, we have captured packets sent between 130.245.145.12 and 128.208.2.198. Node 130.245.145.12 establishes the connection (let's call it sender) with 128.208.2.198 (let's call it receiver) and then sends data. The trace was captured at the sender.

Your `` analysis_pcap_tcp" code should take as input any pcap file (but specifically should work with *assignment2.pcap*). You can hardcode the sender and receiver IP addresses in your code. Your code should output the answers to these questions (Ignore non-TCP traffic):

- The number of TCP flows initiated from the sender. A TCP flow starts with a SYN and ends with a FIN, and a TCP flow is identified by a (source port, source IP address, destination port, destination IP address). A sender can initiate multiple TCP flows at the same time.

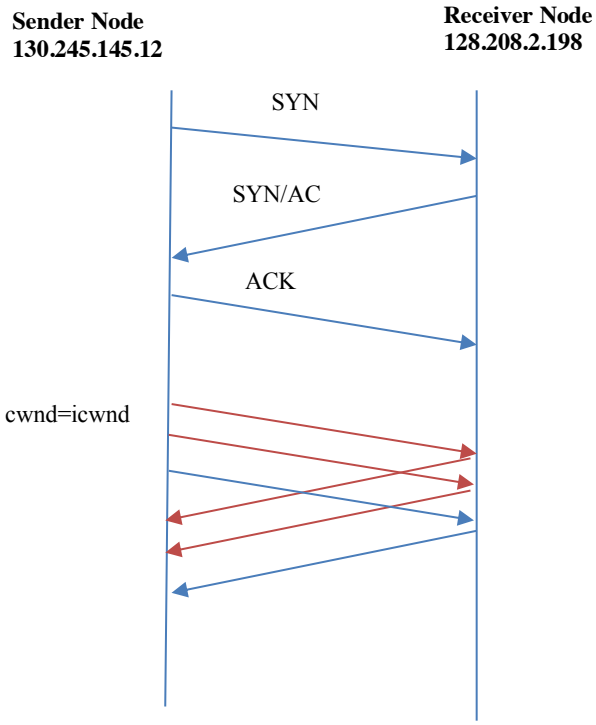
- For each TCP flow

(a) Write down the (source port, source IP address, destination port, destination IP address)

(b) For the first two transactions after the TCP connection is set up (from sender to receiver), the values of the Sequence number, Ack number, and Receive Window size. In the figure below, the first two transactions are marked in orange. If there is a packet loss, this illustration should still work. If the last ACK in the three-way handshake is piggy-backed with the first packet (in orange), then you should still start with this piggy-backed packet.

(c) The sender throughput. I am defining throughput as the total amount of bytes sent by the sender over a period. The period is the time between sending the first byte to receiving the last acknowledgement. For throughput, only consider the packets at the TCP level (including the header). You can ignore all other headers and acks.

This is only one example, your flow may look different



Part B Congestion control (30 points)

Now extend your program so that it will output the following answer to the questions. For each TCP flow:

(1) Print the first 3 congestion window sizes (or till the end of the flow, if there are less than 3 congestion windows). The congestion window is estimated at the sender. You need to estimate the congestion window size empirically since the information is not available in the packet. Comment on how the congestion window size grows. Remember that your estimation may not be perfect, but that is ok. Congestion window sizes change at roughly RTT-intervals.

(2) The number of times a retransmission occurred due to triple duplicate ack and the number of times a retransmission occurred due to timeout. Please note that you need to analyze the packet and use some logic to determine when a packet is a triple duplicate ack and timeout. **You cannot use a library (such as PyShark) to get this answer.** In rare cases, a packet may be retransmitted even if there is no triple duplicate ack (for that packet) or a timeout. Please note these as well if you see them.

Submission Instruction

As before, you may write your programs in **Python or C/C++**. If you want to write in any other language, please talk to me. Viewing these traces on Wireshark can be helpful.

You need to submit your homework in a single zip file as follows:

- The zip file and (the root folder inside) should be named using your last name, first name, and the assignment number, all separated by a dash ('-') e.g. lastname-firstname-assignment2.zip
- The zip file should contain (i) the high-level summary of the analysis_pcap_tcp code including how you estimated the answers to the questions in Part A and Part B, (ii) the analysis_pcap_tcp program, and (iii) instructions on how to run your code

Some example pcap libraries that you can use:
C/C++ - libpcap
Python - dpkt