Amta Sulaiman

CSE-310

Prof. Balasubramanian

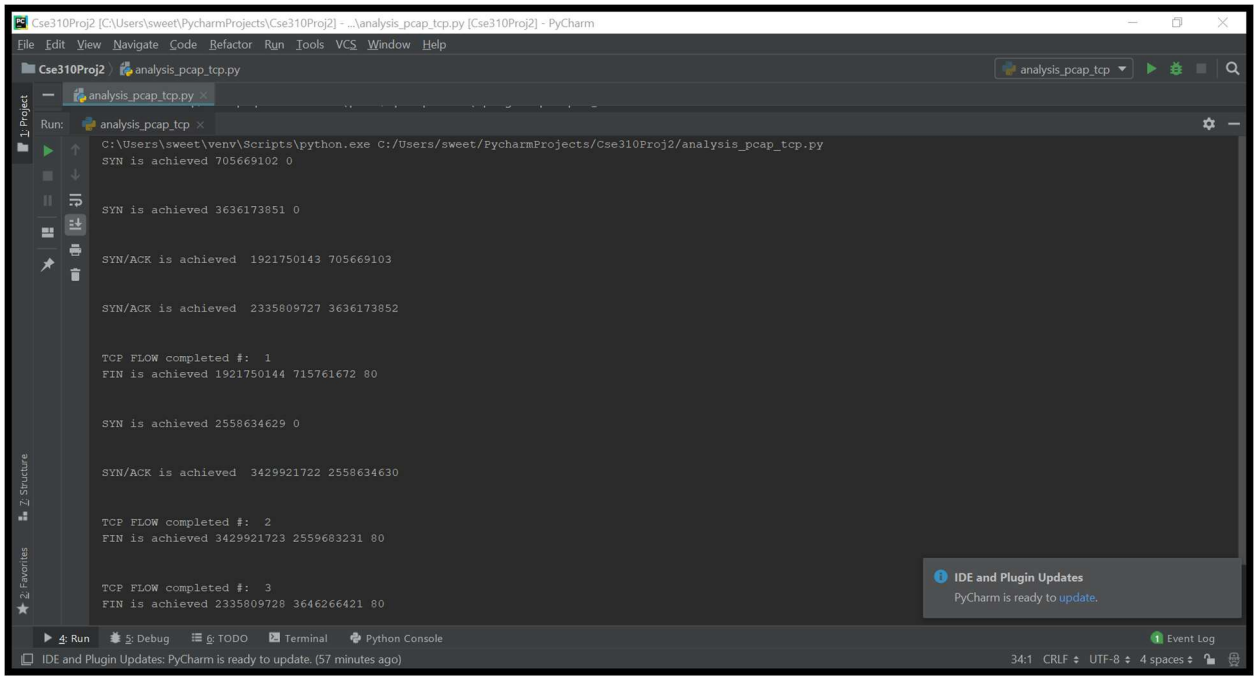
Assignment # 3 (Part A)

Summary:

The attached program deals with analyzing the *.pcap* file consisting of TCP dump trace to differentiate between TCP flows in the trace. The code is written in Python (on Pycharm). The program uses a python library "dpkt" to analyze the given file.

It first reads the pcap file using dpkt.pcap.Reader(obj) function. It consists of a count variable to keep record of TCP flow. Pcap file is then iterated through a for each loop. At first Ethernet frame is retrieved from the buffer inside the loop, then IP layer data packet is retrieved and lastly TCP packet is stored in *"tcp"* variable. It goes further in the code only if the traffic is TCP, otherwise it ignores the other traffic. It checks for 3 different flags in the tcp packet, whether the packet has SYN, SYN/ACK or FIN. It increments the count of TCP flow whenever a FIN is received that means a TCP flow is completed (from sender to received). TCP sequence number and ack number are displayed for the syn, syn/ack and fin.

1- The total number of TCP flows initiated from the sender is = 3 (Execute the analysis_pcap_tcp.py code to check the "TCP FLOW completed #: " on the output screen)



2(a)- There are total 3 TCP flows so, here are the two

For the first TCP flow: The ack number (from sender to receiver) is same as that of the Ack # sent in the last SYN of the 3-way handshake of that particular TCP flow. The receiving window size is 3 for that TCP flow.

```
***Seq Number:  705669127
Ack Number:  1921750144
Receive Window size:  3
***Seq Number:  705670575
Ack Number:  1921750144
Receive Window size:  3
```

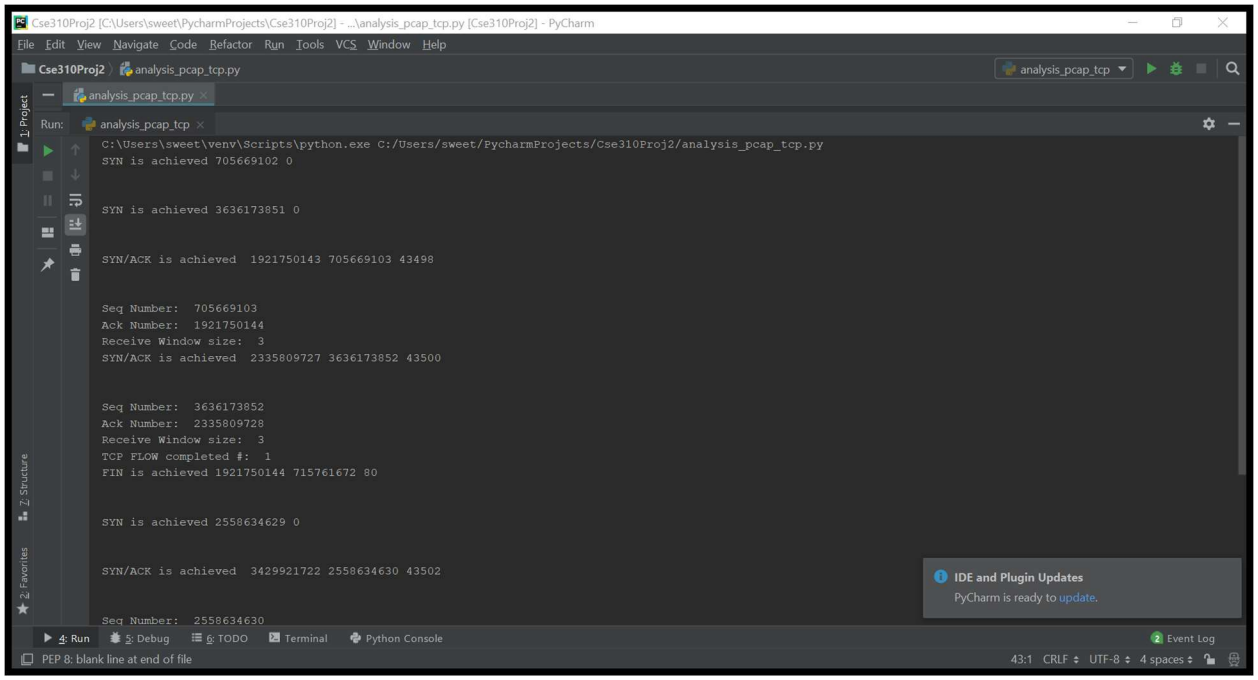
For the second TCP flow: The ack number (from sender to receiver) is same as that of the Ack # sent in the last SYN of the 3-way handshake of that particular TCP flow. The receiving window size is 3 for that TCP flow.

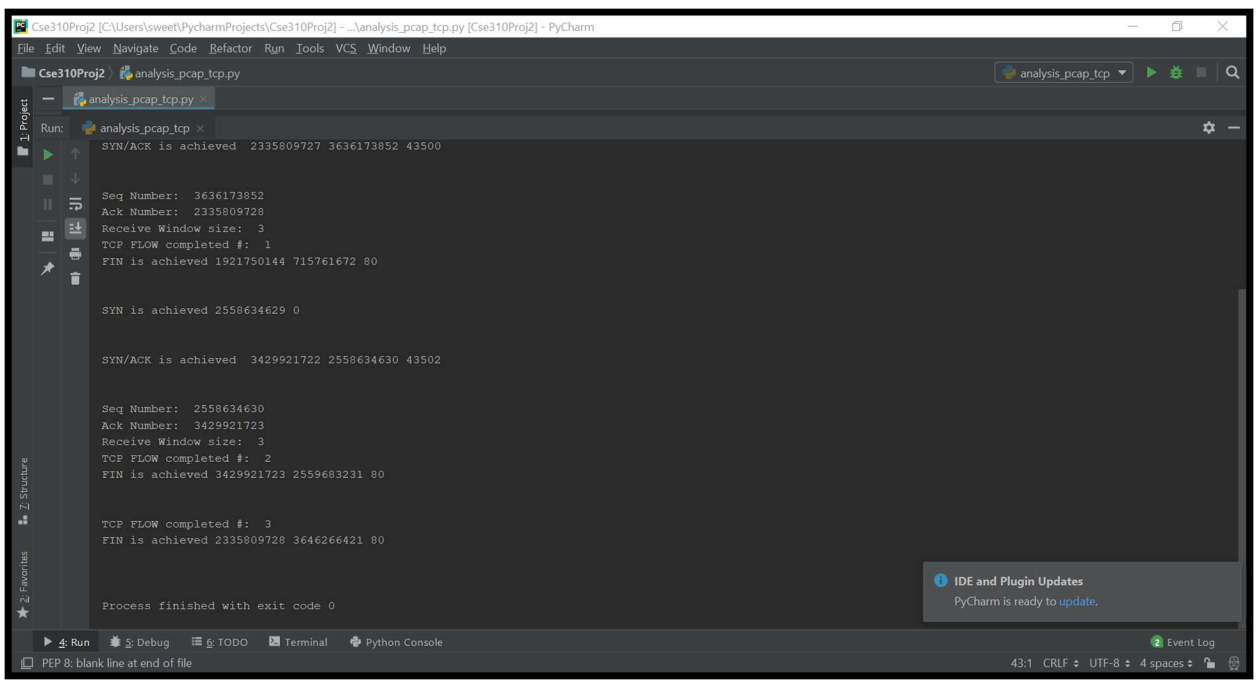
```
***Seq Number:  3636173876
Ack Number:  2335809728
Receive Window size:  3
***Seq Number:  3636175324
Ack Number:  2335809728
Receive Window size:  3
```

For the third TCP flow: The ack number (from sender to receiver) is same as that of the Ack # sent in the last SYN of the 3-way handshake of that particular TCP flow. The receiving window size is 3 for that TCP flow.

```
***Seq Number:  2558634654
Ack Number:  3429921723
Receive Window size:  3
***Seq Number:  2558636102
Ack Number:  3429921723
Receive Window size:  3
```

Or The full screenshot is attached below:





2(b)- The throughput in the given TCP dump trace is calculated by calculating the time from the first byte sent to the last ack received by the sender. I gathered three separate throughputs for this analyzes that are calculated after the 3-way handshake for each, respectively.

```
TCP FLOW completed #:  1
FIN is achieved 1921750144 715761672 80
Throughput(in seconds) for data sent from sender to receiver:  1157.971
```

```
TCP FLOW completed #:  2
FIN is achieved 3429921723 2559683231 80
Throughput(in seconds) for data sent from sender to receiver:  12646.598
```

```
TCP FLOW completed #:  3
FIN is achieved 2335809728 3646266421 80
Throughput(in seconds) for data sent from sender to receiver:  10646.355
```

2(c)- Loss of packets is estimated by calculating the packets that have been retransmitted due to either triple duplicate ack (TDA) or timeout, and then it is divided by the total no. of packets sent. A list can be used to check the repeated ones (packets or seq #).

```
loss = (len(my_list)-len(my_set))/len(my_list)*100
print("The loss rate is: ", loss)
```

```
The loss rate is:  0.6768647624204683
```

INSTRUCTIONS:

This is a code written in python, consisting of two imports i.e. dpkt and time. The detailed description is defined in the summary. You just need to execute the analysis_pcap_tcp.py file and it displays the most important information retrieved from the file via help of dpkt library. Tcp flags are used to catch the appropriate packets and their characteristics. A count is used to keep record of the completion of TCP flow whenever a FIN is received by the sender. Extra cases are included to obtain the last SYN packet from the sender to the receiver as the handshake or TCP flows are not in order inside the file. The Ack number is used from the second SYN (of the handshake) to obtain the first Seq number after the handshake for each TCP flow. These

cases help to display the Seq #, Ack # and Window size for certain transactions. Time that could be used to obtain the throughput is also initiated after the handshake, that means in 3 cases 2nd SYN.