

# CS305 PA 2

## Introduction

The ability to analyze TCP traffic is a crucial skill in computer networking, as it allows us to understand how applications communicate over the network, identify performance issues and security threats, and optimize network configurations. The TCP stream analyzer that we will be introducing in this article is a key tool for achieving these objectives, as it will enable you to extract important information from TCP packets, such as sequence and acknowledgement numbers, payload data, and statistics. By the end of this assignment, you should have a solid understanding of how TCP stream analysis works. In this time of assignment, you should implement a TCP stream analyzer using Python and Scapy.

## Environment Setup

In this assignment, you are suggested to use Python 3.11, as we will test your code in this environment. And you should install few libraries. With anaconda, you can do this by this command in shell

```
conda create -y -n PA2 python=3.11 && conda activate PA2 && pip install -r requirements.txt
```

**MAKE SURE** that when running this command, the PWD must contain all files needed by test. To check this, run command

```
ls
```

and make sure the output contains these files:

```
RENAME_TO_SID.GRADER  
pkt_analyzer.py  
TCP_PKTS.pcap  
HTTP_.pcap  
PACKET_INFO.txt  
requirements.txt
```

Otherwise you can manually install each library by `pip`.

# Requirements

## Task 1

In this task, you should implement a simple TCP packet analyzer, which can be used to do:

- (5pts)  
Extract all TCP connections in a given `pcap` file, print them and save them into a `txt` file in the format of

```
/*Src. IP*/:/*Src. Port*/ -> /*Dst. IP*/:/*Dst. Port*/
```

Note that you only need to implement IPv4/TCP in this part. Make sure to output connection in time order. If you use the code skeleton, it should be easy to do.

- (45pts)
  1. Tell which packet in the `pcap` file belongs to which TCP stream, save them into a `txt` file, each packet occupied each line. You should tell the number sequence of the packet. You should also tell which packet belongs to the client (sent by the client), and which belongs to the server.
  2. And you should calculate relative ACK and SEQ nums, correctly. You must have learnt the definitions of relative ACK and SEQ, so detailed information about them are omitted for brevity.
  3. You should also label flags in TCP packets.

and the line is in the format of, if the sender is the server and receiver is client:

```
Server -> Client; Num: /*Packet Number*/ SEQ: /*SEQ NUM*/ ACK: /*ACK NUM*/ /*FLAGS*/
```

otherwise if the sender is the client and receiver is server:

```
Client -> Server; Num: /*Packet Number*/ SEQ: /*SEQ NUM*/ ACK: /*ACK NUM*/ /*FLAGS*/
```

Besides each packet, you are also requested to save these properties into the txt file, in the first line of the file, in the format of

```
Server : /*Server. IP*/:/*Serevr. Port*/ <--> Client : /*Client  
IP*/:/*Client Port*/
```

- (10pts) Handle IPv6 packets. You should implement your packet analyzer to handle TCP carried by IPv6 packets. After implementation, your pkt analyzer should be able to read out IPv6 packets and analyze TCP streams containing IPv6 packets.
- (20pts) Decode HTTP/1.1 stream.

The first line of the file is

```
Server : /*Server. IP*/:/*Serevr. Port*/ <--> Client : /*Client  
IP*/:/*Client Port*/
```

analyze http stream, read out the host of the stream, and identify each packet: response or request? Status code? Payload length? Please decode the HTTP in the format of

```
METHOD URL PROTOCAL_VERSION
```

```
PROTOCAL_VERSION STATUS_CODE STATUS
```

You only need to implement HTTP/1.1. You do not need to implement HTTPS. You are guarenteed that when testing this part, a .pcap that contains only HTTP packets will be provided. For those packets which does not have a HTTP header, you should use

```
..NO HEADER..
```

to indicate.

**There are three example txt files to demonstrate correct format of output of your program.** If you are confused, carefully check them to make sure you have understand the idea.

## Task 2

(20pts) Use the packet analyzer programmed by you to demonstrate some behaviors of TCP stream, including:

1. RENO-TCP Fast Recovery
2. Sliding of Congestion Window(how to check congestion windows from packets?)
3. Slow Start

and so on. The demonstration methods should include at least one figure automatically plotted by python. You are not allowed to submit a figure plotted by hand-making data. You have to **specificate** the code clip you used to plot. You must clearly indicate which part of the figure tells which TCP behavior. **Note** that you do not need to demonstrate all behaviours.

## Grading Rules

For Task 1, a script file will be provided. To use this script, you have to use Python 3.11 environment. You can use

```
python *your_sid*.GRADER
```

to run grader script. Make sure to submit the screenshot of it in your report. Do not try to modify answer.txt. BIG BROTHER IS WATCHING YOU.

For Task 2, you have to demonstrate behaviors of TCP streams, using a code clip to plot arguments of a TCP packet. You must clearly indicate which behavior is presented, explain reasons and give your code to do plotting.

## Step by step Tutorial

In this assignment you are required to use `scapy` library to decode packets saved in a `.pcap` file. In task 1, you can only use `pcap` file provided by us. Usage of `scapy` is very easy.

```
def packet_info(pcap_file, save_file):
    # Open the pcap file
    packets = rdpcap(pcap_file)

    # Loop through all packets in the pcap file
    with open(save_file, 'w') as f:
        for packet in packets:
            # Check if the packet is an IPv4 packet
            f.write("{}: {} -> {}: {} \n".format(packet['IP'].src,
            packet['TCP'].sport, packet['IP'].dst, packet['TCP'].dport))
```

You can simply use `packet['TCP']` to access the TCP layer of the packet.

## Tips

1. Carefully handle `LAYER_NAME`.
2. Carefully handle packets of different protocols, since different protocols have different members.
3. For application layer, i.e., the payload of TCP layer, you should note that directly use `packet['HTTP']` may not be a proper way. Use `packet['TCP'].payload` instead.
4. If you are confused, you can try breakpoint and use debugger to observe members and status of your code, as well as packet object. Or you can turn to official documents of course.
5. Note the difference of stream and connection. In this assignment you may assume that streams are specified by (server address, client address, server port, client port) and connections are specified by (dst address, src address, dst port, src port). This may be slightly different with precise definition of stream and connection in textbook.
6. You may encounter a lot of exceptions, like `KeyError`, `UnicodeDecodeError` and so on. Try to use `try...except` block.