Communication Networks – final project

Background

Everyday apps, such as emailing, web surfing, and streaming, highly vary in their traffic characteristics – e.g.,

A. IP header fields.
B. TCP header fields.
C. TLS header fields.
D. packet sizes.
E. Packets inter-arrivals (the time between every two packets).
F. Flow size (namely, the overall number of packets).
G. Flow volume (namely, the overall number of bytes transmitted).

Learning traffic characteristics has several important applications. For instance, the telecom/internet provider can use it to prioritize traffic and improve user experience. An attacker can use it to learn the user's activity, thus mitigating privacy. Even if the traffic is encrypted, an attacker who can view the encrypted packets can learn from them which apps the victim used at each moment.

In this project, we'll perform an introductory study of the traffic characteristics of the packets that we use.

Part I: Answer the following Questions:

1. A user reports that their file transfer is slow, and you need to analyze the transport layer to identify the potential reasons. What factors could contribute to the slow transfer, and how would you troubleshoot it?
2. Analyze the effects of TCP's flow control mechanism on data transmission. How would it impact performance when the sender has significantly higher processing power than the receiver?
3. Analyze the role of routing in a network where multiple paths exist between the source and destination. How does the path choice affect network performance, and what factors should be considered in routing decisions?
4. How does MPTCP improve network performance?
5. You are monitoring network traffic and notice high packet loss between two routers. Analyze the potential causes for packet loss at the Network and Transport Layers and recommend steps to resolve the issue.

Part II: Read the following papers (included in the .zip file of this project)

- FlowPic_Encrypted_Internet_Traffic_Classification_is_as_Easy_as_Image Recognition
- Early_Traffic_Classification_With_Encrypted_ClientHello_A_Multi-Country Study
- Analyzing HTTPS Encrypted Traffic to Identify User's Operating System, Browser and Application

For each paper, write:

- What is the main contribution of the paper?
- What traffic features does the paper use, and which are novel?
- What are the main results (you may copy the figures from the paper), and what are the insights from their results?

You may use AI tools, but **you must** check the reliability of the answer. In particular, you should **write the prompts that you used.**

Part III

1. Use Wireshark (or another software) to record your PC's traffic. Please open only one app at a time and minimize "noise" traffic (e.g., SW updates, notifications, etc.). Consider the following apps: (Due to TLS, you may need to save the keys to decrypt the traffic)

● Web-surfing 1: browse web pages using a browser (e.g., Google Chrome).
● Web-surfing 2: browse web pages using another browser (e.g., MS Edge).
● Audio streaming: e.g., Spotify, where you can choose voice-only (e.g., podcast, or song without video clip).
● Video streaming: e.g., Youtube.
● Video conferencing: e.g., Zoom.

2. Present plots comparing those apps' traffic characteristics (e.g., the characteristics A, B, C, and D mentioned above). You may use Wireshark plots. Your code must correctly run on Linux, and will be checked on Linux prompt, using Python's clean installation. If you use Python packages, you should state that in the .pdf.

3. Explain the plots literally. E.g., say: "app X typically uses much larger packets than app Y".

4. Record your PC's traffic again. **Suppose you're an attacker** who tries to learn which apps the user accessed. Consider 2 options:

- The attacker knows each packet's size, time stamp, and hash of the 4-tuple (source IP, dest IP, source port, dest port) flow ID.
- The attacker knows only the size and time stamp for each packet. Check to what extent the attacker can identify which site/apps the user visited, **even if the traffic is fully encrypted (or at least anonymized).** Explain why the attacker can or cannot identify the site and how we can mitigate this attack

Bonuses (10 pt):

Repeat the test when one main app is open but another is used occasionally. E.g., the user listens to music using Spotify and occasionally sends emails. If you have an idea for a slightly different aspect of this problem, you are welcome to mail one of the lecturers and get permission to implement it.

<u>Submission instructions</u>

The submission is in groups of up to 4 students.

1. **GitHub:** Open a GitHub repository. The repo should include:
    a. A README file that briefly explains the project and provides compilation/running instructions.
    b. Your final report, as a .pdf file.
    c. A directory named /src/, containing all your code.
    d. A directory named /res/ containing all your result files that are not too large (e.g., figures).
    e. Do NOT upload large files (e.g., .pcap files), as GitHub's free account does not support this.
2. **LinkedIn:** Each student in the group should open a LinkedIn profile (if he doesn't have one yet), and link the GitHub's repo page to his profile as a post and/or under "projects".
3. **Moodle:** Submit a .zip file. The filename should include the IDs of all the students in the group, e.g. 012345678_9876543_77711111.zip
   The .zip file should include:
    a. A link to your GitHub repo. Including readme that explain what you did and the summary of the papers.
    b. The .pcap files you recorded. If the .pcap files are large, upload them to the cloud and insert a link to them in the .PDF file. Please don't submit the .pcap as it is; consider filtering it.
    c. The code for parsing the .pcap files should run on every computer. Hence, every decision should be general and not specific to your computer. E.g., use a relative path and not the absolute path of your machine. **Your code must correctly run on Linux, and will be checked on Linux.**
    d. Write clean and clear code, including identifiers, function names, and parameters (instead of arbitrary numbers within the code).
    e. Handle edge cases and bugs, such as trying to open a file that does not exist and division by zero.
    f. You may use code you find on the Internet and even AI tools, but please explicitly write the sources and promotes you used.
    g. A private teacher or guide may help you, but they should **not** write your code (or parts from it) themselves.

Good luck!