Assignment 1: Data Collection for Application Performance Monitoring

COL867: NetML

Deadline: 19 Feb 2025

Goal: The objective of this assignment is to familiarize you with the process of data collection and preprocessing in the context of networking. We will focus on the learning task of application performance monitoring. You will build an automation framework that streams DASH videos and collects the application-level performance metrics, network traffic, and HTTP ARchive (HAR) logs. Using this automation framework, you will collect data under different network conditions and finally analyze/preprocess the data.

1 Automation Framework (50 marks)

You will write a script (in Python or another language of their choice) to automate the process of browsing YouTube videos using Selenium and collecting HAR files and packet captures.

1.1 Task Details

The script should perform the following tasks:

- Open a browser (e.g., Chrome or Firefox).
- Navigate to a specified YouTube video URL.
- Play the video for a predefined duration of 3 minutes. Attempt to skip ads if they appear.
- Collect HAR files and packet captures during the session.
- Collect the following application performance metrics every second: video resolution and buffer occupancy. Additionally, log the startup latency (can be estimated at a granularity of 1 second). **Hint**: You can use `Stats for Nerds` along with JavaScript code to capture these metrics.
- Save the collected data for later analysis.

1.2 Submission Requirements

Your script should be executable using the following command:

```
python main.py --url <url> --output_pref <output_pref> --shaping true <default false>
```

The program must take the following inputs:

- `--url`: The YouTube video URL.
- `--output_pref`: The prefix for output filenames.

- `--shaping`: Whether network shaping is enabled (default is `false`).

The script should output four files with the specified prefix:

- `.pcap`: Packet capture file.
- `.har`: HAR log file.
- `.csv`: A CSV file with each line containing the timestamp, video resolution, and buffer occupancy (comma-separated).
- `.log`: A log file containing the following data as comma-separated values:
  - Video URL
  - Startup time
  - Re-buffering ratio
  - Average resolution
  - Average network bandwidth
  - Variance of network bandwidth

  If `--shaping` is `false`, enter `-1` for average network bandwidth and variance.

Example Output: If the output prefix is *out*, the program should generate `out.pcap`, `out.har`, `out.csv`, and `out.log`.

2 Data Collection (25 marks)

Using the automation framework created in Part 1, you need to collect data for a minimum of 50 video sessions. To do this you should create a list of 100 video URLs such that each video is at least 5 minutes long. This is to avoid finishing playback before the end of your experiment. For each video session, randomly pick one video url from this list as the input. The output prefix should be the *x_timestamp* where *x* is a 6-digit ID and concatenation of last-3 digits of you and your partner's entry number. The *timestamp* is the time of the beginning of the experiment, i.e., right before you open the video URL. You should emulate the network bandwidth during each session as described below.

Emulating network conditions: This is how you should vary the network bandwidth: start emulating bandwidth as soon as you begin the session. Vary the network bandwidth at 1-second intervals by picking a random bandwidth between 50 kbps and 5 Mbps.

You should emulate network conditions using traffic controller (`tc`). You might need a Linux-based OS to run `tc`. If you do not have access to a linux machine, feel free to request a BaadalVM. I will share a helper script to emulate the network conditions.

Bonus: You will get a bonus of 5 marks for each additional 50 sessions you collect data, for up to 150 additional sessions.

3 Data analysis (25 Marks)

You should analyze the data you collected from the perspective of application performance monitoring. In particular, you should analyze the following:

- **Dataset characterization** Plot CDF of the following distributions in your dataset: average video resolution, re-buffering ratio, startup time, average network bandwidth, variance in network bandwidth.

- **HTTP logs**: The HAR logs contain the HTTP data. Identify the transactions corresponding to video and audio data. Do you observe separate transactions? Can you identify the requested video resolution from the data? Plot the average resolution as observed in the HTTP logs and compare it with the average resolution obtained from the player logs. What is the error rate? Explain your observations.
- **Network logs**: Can you identify the audio/video transactions from the network logs? Using the logic we discussed in class, identify the HTTP transactions (both duration and sizes) from within the packet logs. Are the number of HTTP transactions and their sizes the same as in the HAR logs?

The analysis should be in a single report titled `report.pdf`. You should also submit the code you write for analysis. I would highly encourage you to use Python for the data analysis.

Note: You are highly encouraged to ask questions on Piazza. You can use GenAI tools for the coding part. Of course, you should understand the code.