Lab Assignment 1a

Application Layer (A Video Streaming Application)

Due Date: Friday, February 10th by 11:59PM

Instructor: Dr. Abdelhak Bentaleb

Submission Deadline

10th February 2023 (Friday) 11:59 pm sharp. 2 marks penalty will be imposed on late submission (Late submission refers to submission or re-submission after the deadline). The submission folder will be closed on **13th February 2023 (Friday) 11:59 pm sharp**, and no late submissions will be accepted afterward.

Objectives

In this assignment, you will inspect one multimedia streaming applications that run on `Dynamic Adaptive Streaming over HTTP (DASH)` and discuss some important aspects that characterize this popular streaming standard. After completing this assignment, you should **(1)** have a good understanding of how DASH work in real-life applications and **(2)** know how to use common network inspection tools for simple network debugging tasks.

This assignment is worth **4 marks** in total. There are two exercises, and each exercise is worth 3 marks. All the work in this assignment shall be completed **individually**.

Setup

You should use your local machine for all tasks in this assignment. The tools you need (on your local machine or lab machine) are:

1. Google Chrome or any other modern browser
2. Wireshark

Submission

Create a zip file that contains the submission files (details below) and submit it to the corresponding assignment in Moodle. Name your zip file as ≪Student number≫.zip, where ≪Student number≫ refers to your student ID number. Your zip file should only contain these two submission files:

1. `submission-≪Student number≫.txt`: A single plain text file to record all your answers to the discussion questions in Exercises 1 and 2.
2. `dash-wireshark-<Student number>.pcapng`: See Exercise 1, Task 3.

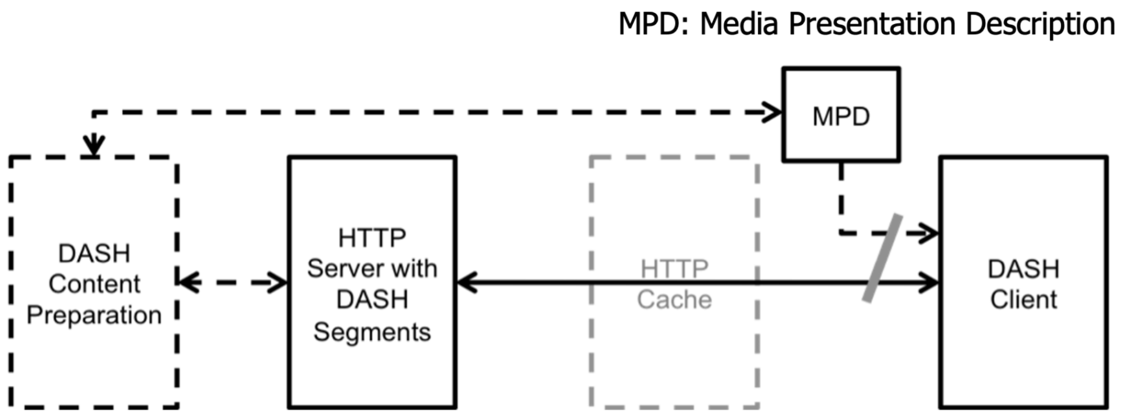
MPD: Media Presentation Description
MPD
DASH Content Preparation
HTTP Server with DASH Segments
HTTP Cache
DASH Client

Figure 1: A streaming application built on DASH.

Plagiarism Warning

You are free to discuss this assignment with your friends. **However, you should refrain from sharing your answers or network logs**. We highly recommend that you attempt this assignment on your own and figure things out along the way as many resources are available online.

We employ a zero-tolerance policy against plagiarism. If a suspicious case is found, the student would be asked to explain his/her answers to the evaluator in person. The confirmed breach may result in a zero mark for the assignment and further disciplinary action from the school.

Question & Answer

If you have any doubts about this assignment, please post your questions on Moodle or consult the TA. However, the TA will NOT debug issues for students. The intention of Q&A is to help clarify misconceptions or give you necessary directions.

Exercise 1 - DASH

In the lecture, we learned about `DASH (Dynamic Adaptive Streaming over HTTP)`, which is a pull-based streaming standard over HTTP. This exercise helps us understand how DASH works in real-life streaming applications. There are three tasks in this exercise and each task is worth 1 mark. **We recommend completing the tasks in order**.

Recall the architecture of a streaming application built on DASH is shown in Figure 1.

The HTTP server provides the playlist and media segments (sometimes referred to as streamlets) for the DASH client to retrieve (*i.e.*, "pull") on-demand.

For our DASH client in this exercise, we will use a browser-based video player provided by the DASH Industry Forum at:

`https://reference.dashif.org/dash.js/v4.5.0/samples/dash-if-reference-player/index.html`

Please access the player using any modern browser (we recommend using Google Chrome).

Task 1: MPD Playlist (1 mark)

The player provides several MPD playlists for users to choose from (see the list under "Stream"). In this assignment, we will use our playlist at

`https://nustreaming.github.io/streaming/bbb.mpd`
Download the playlist by pasting the MPD URL in your browser's address bar. Inspect the file with your favorite text editor and answer the following discussion questions:

1. How many different **video quality levels** are provided by this playlist?
2. What are the highest and lowest quality levels?
3. Why do we need different quality levels? Please give an example where the player would retrieve the video at different quality levels.

Record your answers clearly in `submission-≪Student number≫.txt`.

Task 2: Video Segments (1 mark)

Next, play the video provided by our playlist by replacing the MPD URL in the DASH client and clicking "Load". Now let's inspect the video segments retrieved by the client. On your browser, open up the `network inspection tool` [1] to view the browser's network log.

While the player is playing (*i.e.*, streaming) the video, you should see the network log being updated dynamically. This log contains the list of HTTP responses received by the client, including the video segments retrieved.

Pick one video segment from the list, download it, and answer the following discussion questions:

1. What is the URL of your selected video segment?
2. What is the file format of this video segment?
3. What is the duration of this video segment? (Note that all video segments from the same playlist should have the same duration.)
4. Different applications may use different video segment durations (typically between 2 and 10 seconds). From a networking perspective, describe one advantage and one disadvantage of using longer video segments (*e.g.*, 10 seconds per segment).

Record your answers clearly in `submission-≪Student number≫.txt`.

Task 3: HTTP-based Transfer (2 marks)

From Task 1 and 2, we can see that for DASH, the MPD playlist and video segments are retrieved over HTTP (*i.e.*, the HTTP URLs). Now let's inspect the HTTP request and response in greater detail using Wireshark.

Open Wireshark on the machine running the browser. Using the **same MPD URL** as in Task 1, try retrieving the MPD file again on your browser and observe the capture log being updated on Wireshark.

[1] For Google Chrome, right-click "Inspect" and click on the "Network" tab. For more details, refer to `https://developers.google.com/web/tools/chrome-devtools/network`

Save Wireshark's capture log for this HTTP request and response in the default pcapng file format. Use `Wireshark capture filters` [2] **to capture and save only the relevant packets**.

Save your file as `dash-wireshark-≪Student number≫.pcapng`.

[2] Check `https://wiki.wireshark.org/CaptureFilters`