Lab Assignment 1b

Application Layer (A Video Streaming Application)

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

Instructor: Dr. Abdelhak Bentaleb

Introduction and Submission Deadline

In this assignment, you will conduct a set of experiments to test how HTTP works in the case of `HTTP Adaptive Streaming (HAS)`. This experimental assignment is worth 8 marks. The deadline for submission is **February 10th, 2023, 11:59 pm (Friday)** sharp. Do not leave your submission to the last minute in case of an unforeseeable situation. You may submit many times and only the last submission will be graded. 2 marks penalty will be imposed on late submissions (*i.e.*, submissions or re-submissions made after the deadline). No submission will be accepted after **February 13th, 2023**, and a 0 mark will be awarded.

Group Work

This assignment should be solved **individually on in a group with a maximum of two students**. Under no circumstances should you solve it in a group and then submit it as an individual solution. This is considered plagiarism. Group work needs special care during submission (see below).

Grading

We will evaluate your submitted materials which include a report and presentation. Please make sure that your report and presentation are well organized and structured. Moreover, you have to book an appointment with TA to demonstrate your experimental results. We recommend automating the results collection by developing a script based on Python 3 or Javascript. **We will grade your report, presentation, a how you conduct results collection**.

Materials Submission

For individual submission, please submit a zip file that contains the report, presentation, and result collection scripts (if applicable) and submit it to the corresponding assignment in Moodle. Please name your zip file as ≪Student number≫.zip, where ≪Student number≫ refers to your ID number. Note that the first and last letters of your student number should be capital letters. Do not put your zip file under any subfolders.

For group work, please designate one person to submit the zip file, instead of submitting the same file twice by two different persons. The file name should contain the student ID of two members and be named as ≪Student number 1≫-≪Student number 2≫.zip.

We will deduct 1 mark for every type of failure to follow instructions (*e.g.*, wrong program name, wrong zip file name, folder structure).

Grading Rubric

We will grade your programs based on:

Report [3 marks]

Your write-up should be similar to a standard report (check Paper 1 and Paper 2 below). Please use the provided template. Please include (at least) the following sections:

1. Abstract (an overview of what the assignment includes)
2. Introduction and motivation for your work (your specific design and implementation choices)
3. Results, comparison, and discussion, i.e., what should we learn from this and what could possibly be further improved.
4. Conclusions
5. References

Most of the marks will be given for sections 2. and 3.

Presentation [4 marks]

You will present your lab assignment in class during a lab session after the submission deadline. You will have approximately 10 minutes to demo your assignment. The demo should include (1) a presentation to the lab demonstrator (2.5 marks), and (2) how you conduct the experiments and collect the results (1.5 marks).

Automated Scripts [1 mark]

Any developed scripts to automate the experiment and result collection.

Question and Answer

If you have any doubts about this assignment, please post your questions on Moodle forum or consult the TA. We will not debug programs for you. However, we may help to clarify misconceptions or give necessary directions if required.

Plagiarism Warning

You are free to discuss this assignment with your friends. However, ultimately, you should write your own code. We employ a zero-tolerance policy against plagiarism. If a suspicious case is found, we will award zero points and may take further disciplinary action.

A Word of Advice

This assignment can be time-consuming. We suggest that you start your assignment early. **Do not post your solution in any public domain on the Internet or share it with friends, even after this term.**

Assignment Description

In this lab assignment, your task is to build a DASH-compliant (Dynamic Adaptive Streaming over HTTP— that is, compatible with MPEG DASH standard) client-server-based video-on-demand streaming system on top of a LAMP stack (Linux, Apache, MySQL, PHP).

The DASH approach of streaming is becoming very popular. It is basically a replacement for the RTSP/RTP/RTCP-based approach to streaming, especially with Video on Demand (VoD). With DASH, the server is a simple HTTP web server (e.g., Apache). The media (video) is divided into small individually playable video segments which are, for example, 10 seconds long. These segments are also called streamlets. The client media player retrieves the streamlets from the web server, one at a time, and plays them without interruption. For the client to know the streamlet files that belong to a complete video, the server provides a playlist file (also called a Media Presentation Description (MPD)). The playlist file has a special format; it is basically an XML file for the MPEG DASH standard. To start streaming, the client player loads the playlist file and then starts to download the streamlets that are listed in this file. Your task is to generate such a playlist file onto the server from a media file and support VoD playback of those streamlets onto another client device (*e.g.*, your laptop).

You will be given several utilities that will help you to get the lab assignment done. The lab assignment work is to **(1)** create a media player application based on `dash.js` that plays a stored (VoD) DASH playlist, while adaptively switching and playing video streamlets, and **(2)** Conduct various set of experiments to test/compare the efficiency of built-in Adaptive BitRate (ABR) algorithms.

The web-based media player application, based on `dash.js` (**v4.5.0**), should retrieve the list of the uploaded videos available on the web server. Your player should allow on-demand playback of previously uploaded videos. It should support adaptive stream switching through ABR – that means it adaptively chooses the most suitable quality for the next video streamlet to be downloaded based on your network condition while playing the current streamlet. The minimum steps for this task are:

1. Retrieve a list of videos available on your server and download the MPEG-DASH XML-formatted playlist file for the selected video (on-demand).
2. Switch the rendering of one video streamlet to another based on the ABR algorithm.
3. Collect the results during the playback every 1 second. The metrics you should collect periodically are: `selected bitrate (Mbps)`, `buffer level (second)`, `measured throughput (Mbps)`, `segment download time (second)`, and `segment size (byte)`. **The result collection can be done either through an implemented customized script or by modifying the HTML of the video player (i.e., add a Javascript function).**
4. Repeat steps 1., 2., and 3. for every ABR (Throughput-based, BOLA, and Dynamic).
5. Compare the three ABRs in terms of metrics highlighted in step 3.
6. Analyze/Discuss the results and write the report.

Reference and Software Information

We provide a web server that stores the segmented video content with a link to the MPD, which is located at: `https://nustreaming.github.io/streaming/bbb.mpd`. You can access the player using your browser using the following URL:

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

Paper 1: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8424813

Paper 2: https://www.cc.gatech.edu/fac/Constantinos.Dovrolis/Papers/final-saamer-mmsys11.pdf