Lab Assignment 2

Application/Transport Layers (A Video Streaming Application)
Due Date: Monday, April 10th by 11:59PM

Instructors: Dr. Abdelhak Bentaleb & Dr. Sandra Cespedes

Introduction and Submission Deadline

In this assignment, you have to develop a web or mobile application based on your preference that does video recording and uploading (to a server) at the same time. This lab assignment is worth 15 marks. The deadline for submission is **April 10th, 2023, 11:59 pm (Monday)** 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 **April 12th, 2023**, and a 0 mark will be awarded.

Group Work

This assignment can be solved **in groups 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 the web or mobile application, server-side media content preparation scripts, and your demonstration. Please make sure that your programs are well organized and structured. Moreover, you have to book an appointment with TA to demonstrate your solution. You can use Python, Javascript, HTML, CSS, PHP, Andriod, C/C++ or a combination of programming languages to develop your solution. **We will grade your code and the functionality of each program**.

Materials Submission

For individual submission, please submit a zip file that contains the source code of your programs and a presentation for the demo (details below), 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.

Your zip file should only contain these submission files:

1. Your web or mobile app source code and installable package (.apk file in case of a mobile app);
2. Your PHP, Python, or other framework's server-side code for content preparation (*i.e.*, real-time storing in our server).
3. A detailed README.txt file, that includes: Your name(s), student ID, and your username(s) on the host server, and A brief description of each source file, and how it works.

Before you submit the file. Make sure that your code compiles without ANY errors!

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).

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 application (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).

You will be given several utilities to help you complete the lab assignment. The lab assignment is mainly **to create a web or mobile application that captures a live video and uploads its streamlets to a server as they become available.**

Detailed Tasks (13 marks)

The web/mobile application, running on a smartphone or laptop/desktop, is required to provide the following sub-tasks:

1. Capture a live video feed of 720p resolution at 30 fps from the device camera. You can use `WebCodecs API`[1] for web or `MediaCodec API`[2] from Android[3] to encode frames with h.264 encoding with 5 Mbps bitrate levels.

2. Upload the captured frames to a web server **reliably on-the-fly**.

[1] Check: `https://www.w3.org/TR/webcodecs/`

[2] Check: `https://developer.android.com/reference/android/media/MediaCodec`

[3] iOS is not recommended for 2 main reasons: 1) privacy issues, 2) Lack of open-source APIs that help to complete the tasks.

- Segment the original live feed into a number of self-contained 3-second-long MP4 segments before uploading to the server. Segmentation can either be done first, before the upload, or in parallel, together with the upload. We recommend to do everything in paralell using `MediaCodec` API.
- Use the HTTP POST method to deliver the segmented MP4 video feed to the server. The server has to store the video segments in a video repository location (*i.e.*, directory), which you define. To keep track of the uploading status, you may use a PHP and MySQL database. For segmentation, you can use any segmentation tool such as `ffmpeg`[4] or `MP4Box`[5].

For you to understand the internal structure of the MP4 format, we intentionally ask you to segment the video at the mobile or web client (*i.e.*, at your mobile or your laptop/desktop), not at the server. To segment the video, you may use third-party libraries such as `MP4Parser`[6]

To upload the video segments reliably, you are required to **design a simple protocol on top of HTTP for guaranteed delivery, such as checking the current upload status or providing segments with a sequence number**.

The following are additional functionalities that the web or mobile app can provide, which will receive extra credits:

3. Provide a resumed upload when the network connection is interrupted.
4. Retrieve the list of the uploaded videos available from the web server.
5. Videos should be playable live onto the client device during a session as well as stored on the server for on-demand playback.

Demo Presentation (2 marks)

You will present your lab assignment 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 (run your programs) to the lab demonstrator (1 mark) and (2) an explanation of how you developed all three tasks (1 mark).

Reference and Software Information

All the students taking the COMP 445 module will receive a user account on one of our servers (**Section M**: `labs445-1.encs.concorida.ca` — **Section W**: `labs445-2.encs.concorida.ca`). Each student can use the account to access the server. The server is running the Ubuntu LTS distribution of Linux. You will need to use `ssh`[7] to connect to the server machine. If you use the default LAMP setup we provided, your content preparation scripts will be executed by the web server from your `home_dir/public_html` directory. For example, if group1 wants to store the file `info.php` it will be placed at `/home/team1/public_html/info.php` and will be available over the web at `labs445-x.encs.concordia.ca/~team1/info.php`. The server also has `SSL`[8] enabled. Therefore, teams can optionally use https in their GET and POST request URLs. It is, however, recommended to avoid SSL for requests involving file transfers such as the streamlets.

When segmenting the video with the web or mobile application, you may use the `MP4Parser` package. When developing server utilities, you are allowed to use any useful MPEG-4 parsing and transcoding

[4] Check: `https://ffmpeg.org/`

[5] Check: `https://github.com/gpac/gpac/wiki/MP4Box`

[6] check: `https://github.com/sannies/mp4parser`

[7] Check: `https://www.ssh.com/academy/ssh`

[8] https://www.ssl.com/faqs/faq-what-is-ssl/

utilities such as MP4Box or ffmpeg. For your convenience, you can use ffmpeg, MP4Box, and mp4info[9], *etc.*, commands from /usr/local/bin on the server. If you need any further enhancements/additions to the provided default setup or prefer a custom server-side setup other than the default LAMP stack, please contact the TA.

[9] Check: http://www.bento4.com/