Trabalho de Redes de Comunicação Multimídia

Aluno: Marcelo Corni Alves

Data: 13/11/2024

Etapa 01: Instalação e Configuração do FFmpeg

O FFMpeg DASH foi instalado em ambiente Windos com o comando de prompt a seguir:

```
winget install ffmpeg
```

Após a instalação, foi verificada a versão e seus componentes instalados com o commando a seguir:

```
ffmpeg -version
```

Saída do comando:

ffmpeg version 7.1-full_build-www.gyan.dev Copyright (c) 2000-2024 the FFmpeg developers built with gcc 14.2.0 (Rev1, Built by MSYS2 project) configuration: --enable-gpl --enable-version3 --enable-static --disable-w32threads --disable-autodetect --enable-fontconfig --enable-iconv --enable-gnutls --enable-libxml2 --enable-gmp --enable-bzlib --enable-lzma --enable-libsnappy --enable-zlib --enable-librist --enable-libsrt --enable-libssh --enable-libzmq --enable-avisynth --enable-libbluray --enable-libcaca --enable-sdl2 --enable-libaribb24 --enable-libaribcaption --enable-libdav1d --enable-libdavs2 --enable-libopenjpeg --enable-libquirc --enable-libuavs3d --enable-libxevd --enable-libzvbi --enable-libqrencode --enable-librav1e --enable-libsvtav1 --enable-libvvenc --enable-libwebp --enable-libx264 --enable-libx265 --enable-libxavs2 --enable-libxeve --enable-libxvid --enable-libaom --enable-libjxl --enable-libvpx --enable-mediafoundation --enable-libass --enable-frei0r --enable-libfreetype --enable-libfribidi --enable-libharfbuzz --enable-liblensfun --enable-libvidstab --enable-libvmaf --enable-libzimg --enable-amf --enable-cuda-llvm --enable-cuvid --enable-dxva2 --enable-d3d11va --enable-d3d12va --enable-ffnvcodec --enable-libvpl --enable-nvdec --enable-nvenc --enable-vaapi --enable-libshaderc --enable-vulkan --enable-libplacebo --enable-opencl --enable-libcdio --enable-libgme --enable-libmodplug --enable-libopenmpt --enable-libopencore-amrwb --enable-libmp3lame --enable-libshine --enable-libtheora --enable-libtwolame --enable-libvo-amrwbenc --enable-libcodec2 --enable-libilbc --enable-libgsm --enable-liblc3 --enable-libopencore-amrnb --enable-libopus --enable-libspeex --enable-libvorbis --enable-ladspa --enable-libbs2b --enable-libflite --enable-libmysofa --enable-librubberband --enable-libsoxr --enable-chromaprint

libavutil 59. 39.100 / 59. 39.100

libavcodec 61. 19.100 / 61. 19.100

libavformat 61. 7.100 / 61. 7.100

libavdevice 61. 3.100 / 61. 3.100

libavfilter 10. 4.100 / 10. 4.100

libswscale 8. 3.100 / 8. 3.100

libswresample 5. 3.100 / 5. 3.100

libpostproc 58. 3.100 / 58. 3.100

Etapa 02: Captura e Codificação de Áudio e Vídeo em Tempo Real

Para a captura de vídeo, foi desenvolvida uma aplicação em Python. Para listar os dispositivos de vídeo e áudio que seriam necessários para a codificação da aplicação, foi utilizado o comando abaixo:

```
ffmpeg -list_devices true -f dshow -i dummy
```

Para a captura e codificação foi utilizado o comando abaixo:

```
ffmpeg.exe -f dshow -video_size 1920x1080 -i video=Streaming Webcam:audio=Microfone (Streaming Webcam MIC) -c:v libx265 -c:a aac -t
```

Etapa 03: Transcodificação de Vídeo

Para a transcodificação do vídeo, foi utilizado o comando abaixo:

```
ffmpeg.exe -i "E:\Mestrado\Disciplinas\Comunicação Multimídia\Trabalhos\Trab01\captura_video.mp4" -c:v libx264 "E:\Mestrado\Discipli
```

Etapa 04: Multiplexação de Conteúdo em DASH com Alternativas de Qualidade de Vídeo e

Áudio

Para a multiplexação de conteúdo foi utilizado o comando abaixo:

```
ffmpeg.exe -i "E:\Mestrado\Disciplinas\Comunicação Multimídia\Trabalhos\Trab01\transcodificado_h264.mp4" -map 0 -b:v:0 3000k -s:v:0
```

Etapa 05: Multiplexação de Conteúdo em MPEG-2 TS

Para a multiplexação de conteúdo foi utilizado o comando abaixo:

```
ffmpeg.exe -i "E:\Mestrado\Disciplinas\Comunicação Multimídia\Trabalhos\Trab01\transcodificado_h264.mp4" -c:v copy -c:a aac -strict
```

Etapa 06: Streaming de Conteúdo

Para fazer o stream utilizando o DASH foi utilizado o comando abaixo:

```
ffmpeg.exe -f dash -i "E:\Mestrado\Disciplinas\Comunicação Multimídia\Trabalhos\Trab01\dash_output\manifest.mpd" -c copy -f dash -li
```

Para fazer o stream em RTP foi utilizado o comando abaixo:

```
ffmpeg.exe -re -i fixed_multiplexado.ts -map 0:v -c:v copy -f rtp rtp://localhost:5004 -map 0:a -c:a aac -b:a 128k -flags +global_he
```

Etapa 07: Teste do Conteúdo e do Streaming

Para testar a stream DASH foi utilizado o comando abaixo:

```
ffplay.exe dash_output\manifest.mpd
```

Para testar a stream RDP foi utilizado o comando abaixo:

```
ffplay.exe  -protocol_whitelist "file,udp,rtp" stream.sdp
```

Repositório no Github com código da aplicação em Python

https://github.com/marcelocorni/mpeg_dash_01