



Universidade do Minho
Escola de Engenharia
Mestrado em Engenharia Informática

Unidade Curricular de Engenharia de Serviços em Rede

Ano Letivo de 2022/2023

Relatório TP1 Streaming de áudio e vídeo a pedido e em tempo real

Grupo 6 - PL6

Sérgio Gonçalves pg41099
Rita Gomes pg50723

13 de Outubro de 2022

Índice

1	Introdução	1
2	Captura de um vídeo com gravação de ecrã	2
3	Etapa 1: Streaming HTTP simples sem adaptação dinâmica de débito	3
3.1	Questão 1	3
4	Etapa 2: Streaming adaptativo sobre HTTP (MPEG-DASH)	6
4.1	Questão 2	6
4.2	Questão 3	6
4.3	Questão 4	7
5	Etapa 3: Streaming RTP/RTCP unicast sobre UDP e multicast com anúncios SAP	8
5.1	Questão 5	9
6	Conclusões	10
	Lista de Siglas e Acrónimos	11

1 Introdução

O presente relatório apresenta uma breve descrição das tarefas realizadas no âmbito do trabalho prático TP1 da unidade curricular de Engenharia de Serviços em Rede, referente à experimentação de várias soluções de streaming a pedido e em tempo real usando o emulador CORE como bancada experimental.

A proposta de trabalho está estruturada em 3 etapas. Inicialmente, antes de qualquer etapa, vamos criar manualmente dois vídeos (videoA e videoB) a partir da captura de ecrã, usando o ffmpeg. Esses vídeos serão depois usados em todas as etapas.

Na etapa 1, pretende-se fazer streaming por HTTP apenas, sem adaptação do débito. Nesta etapa o VLC é usado simultaneamente como servidor de streaming e como cliente. Depois são acrescentados mais dois clientes (firefox e ffplay) e estuda-se o impacto no tráfego da rede.

Na etapa 2, o servidor de streaming é substituído pelo ffmpeg, que serve o mesmo conteúdo por HTTP mas agora com débito adaptativo. Para simplificar, usam-se apenas duas streams com diferentes exigências em termos de recursos. Fazem-se umas manipulações da capacidade dos links, para ver como o browser se tenta ajustar.

Posteriormente, na etapa 3, vamos usar os históricos protocolos de streaming sobre UDP. Experimenta-se em primeiro lugar o uso do RTP/RTCP em unicast e anúncios SAP, com 4 clientes, mudando de unicast para multicast (ao nível da rede) para perceber o impacto em termos de tráfego na rede.

Conforme sugestão do próprio enunciado, nas secções que se seguem apresentam-se, por ordem, cada uma das perguntas, seguida da respectiva resolução. Por vezes são utilizados printscreen como complemento às respostas.

2 Captura de um vídeo com gravação de ecrã

Esta parte do trabalho era apresentada na forma de um guião que, na sua grande maioria, era apenas para replicar e pouco exigia dos alunos. Todas as tarefas propostas no enunciado foram executadas sem problemas. Apresenta-se na figura a seguir o resultado obtido de gravar corretamente o videoA. Posteriormente, seguimos o mesmo raciocínio para o videoB.

```
core@xubuncore:~$ ffmpeg -i videoA.mp4
ffmpeg version 4.2.7-0ubuntu0.1 Copyright (c) 2000-2022 the FFmpeg developers
  built with gcc 9 (Ubuntu 9.4.0-1ubuntu1-20.04.1)
  configuration: --prefix=/usr --extra-version=0ubuntu0.1 --toolchain=hardened --libdir=/usr/lib/x86_64-linux-gnu --incdir=/usr/include/x86_64-linux-gnu
  u --arch=amd64 --enable-gpl --disable-stripping --enable-avresample --disable-filter=resample --enable-avisynth --enable-gnutls --enable-ladspa --enable-
  e-libaom --enable-libass --enable-libbluray --enable-libbs2b --enable-libcaca --enable-libcdio --enable-libcodec2 --enable-libflite --enable-libfontcon
  fig --enable-libfreetype --enable-libfribidi --enable-libgsm --enable-libjack --enable-liblame --enable-libmfx --enable-libmysofa --enable-libopenjp
  eg --enable-libopenmpt --enable-libopus --enable-libpulse --enable-librtmp --enable-librubberband --enable-libshine --enable-libsnappp --enable-libsoxr
  --enable-libspeex --enable-libsrt --enable-libtheora --enable-libtwolame --enable-libvidstab --enable-libvorbis --enable-libvpx --enable-libwavpack --
  enable-libwebp --enable-libx265 --enable-libxml2 --enable-libxvid --enable-libzmq --enable-libzvt --enable-lv2 --enable-omx --enable-opengl --enable-d
  pencl --enable-opengl --enable-sdl2 --enable-libdca1394 --enable-libdrm --enable-libiec61883 --enable-nvenc --enable-chromaprint --enable-frei0r --enab
  e-libx264 --enable-shared
  libavutil      56. 31.100 / 56. 31.100
  libavcodec     58. 54.100 / 58. 54.100
  libavformat    58. 29.100 / 58. 29.100
  libavdevice    58.  8.100 / 58.  8.100
  libavfilter    7. 57.100 / 7. 57.100
  libavresample  4.  0.  0 / 4.  0.  0
  libswscale     5.  5.100 / 5.  5.100
  libswresample  3.  5.100 / 3.  5.100
  libpostproc   55.  5.100 / 55.  5.100
Input #0, mov,mp4,m4a,3gp,3g2,mj2, from 'videoA.mp4':
  Metadata:
    major_brand      : isom
    minor_version    : 512
    compatible_brands: isomiso2avc1mp41
    encoder         : Lavf58.29.100
  Duration: 00:09:25.50, start: 0.000000, bitrate: 8 kb/s
  Stream #0:0(und): Video: h264 (High) (avc1 / 0x31637661), yuv420p, 160x100, 6 kb/s, 20 fps, 20 tbr, 10240 tbn, 40 tbc (default)
  Metadata:
    handler name     : VideoHandler
At least one output file must be specified
core@xubuncore:~$ ffplay videoA.mp4
ffplay version 4.2.7-0ubuntu0.1 Copyright (c) 2003-2022 the FFmpeg developers
  built with gcc 9 (Ubuntu 9.4.0-1ubuntu1-20.04.1)
  configuration: --prefix=/usr --extra-version=0ubuntu0.1 --toolchain=hardened --libdir=/usr/lib/x86_64-linux-gnu --incdir=/usr/include/x86_64-linux-gr
  u --arch=amd64 --enable-gpl --disable-stripping --enable-avresample --disable-filter=resample --enable-avisynth --enable-gnutls --enable-ladspa --enab
  e-libaom --enable-libass --enable-libbluray --enable-libbs2b --enable-libcaca --enable-libcdio --enable-libcodec2 --enable-libflite --enable-libfontcon
  fig --enable-libfreetype --enable-libfribidi --enable-libgsm --enable-libjack --enable-liblame --enable-libmfx --enable-libmysofa --enable-libopenjp
  eg --enable-libopenmpt --enable-libopus --enable-libpulse --enable-librtmp --enable-librubberband --enable-libshine --enable-libsnappp --enable-libsoxr
  --enable-libspeex --enable-libsrt --enable-libtheora --enable-libtwolame --enable-libvidstab --enable-libvorbis --enable-libvpx --enable-libwavpack --
  enable-libwebp --enable-libx265 --enable-libxml2 --enable-libxvid --enable-libzmq --enable-libzvt --enable-lv2 --enable-omx --enable-opengl --enable-d
  pencl --enable-opengl --enable-sdl2 --enable-libdca1394 --enable-libdrm --enable-libiec61883 --enable-nvenc --enable-chromaprint --enable-frei0r --enab
  e-libx264 --enable-shared
  libavutil      56. 31.100 / 56. 31.100
  libavcodec     58. 54.100 / 58. 54.100
  libavformat    58. 29.100 / 58. 29.100
  libavdevice    58.  8.100 / 58.  8.100
  libavfilter    7. 57.100 / 7. 57.100
  libavresample  4.  0.  0 / 4.  0.  0
  libswscale     5.  5.100 / 5.  5.100
  libswresample  3.  5.100 / 3.  5.100
  libpostproc   55.  5.100 / 55.  5.100
Input #0, mov,mp4,m4a,3gp,3g2,mj2, from 'videoA.mp4':  08 f=0/0
  Metadata:
    major_brand      : isom
    minor_version    : 512
    compatible_brands: isomiso2avc1mp41
    encoder         : Lavf58.29.100
```

Figura 2.1: VideoA gerado com sucesso.

3 Etapa 1: Streaming HTTP simples sem adaptação dinâmica de débito

3.1 Questão 1

Capture três pequenas amostras de tráfego no link de saída do servidor, respectivamente com 1 cliente (VLC), com 2 clientes (VLC e Firefox) e com 3 clientes (VLC, Firefox e ffplay). Identifique a taxa em bps necessária (usando o `ffmpeg -i videoA.mp4` e/ou o próprio wireshark), o encapsulamento usado e o número total de fluxos gerados. Comente a escalabilidade da solução. Ilustre com evidências da realização prática do exercício (ex: capturas de ecrã).

A topologia seguida nesta etapa foi a seguinte:

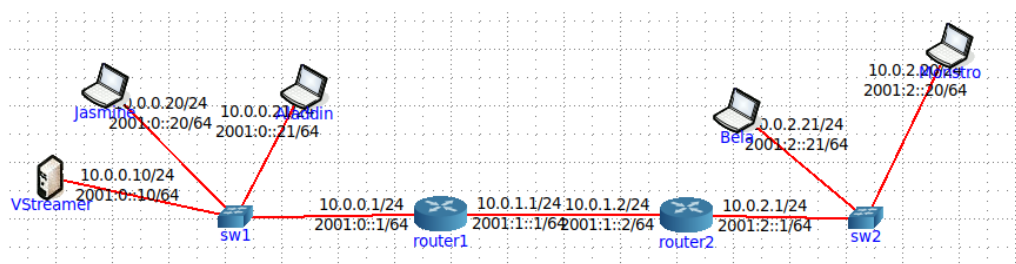


Figura 3.1: Topologia construída.

Nesta etapa foi feito streaming HTTP simples sem adaptação de débito. Verificou-se, usando o comando referido no enunciado, que a taxa necessária para a transmissão do vídeo é 6000 bps ou superior.

```

core@ubuntu:~$ ffmpeg -i videoA.mp4
ffmpeg version 4.2.7-0ubuntu0.1 Copyright (c) 2000-2022 the FFmpeg developers
  built with gcc 9 (Ubuntu 9.4.0-1ubuntu1-20.04.1)
  configuration: --prefix=/usr --extra-version=0ubuntu0.1 --toolchain=hardened --libdir=/usr/lib/x86_64-linux-gnu --incdir=/usr/include/x86_64-linux-gnu
  --arch=amd64 --enable-gpl --disable-stripping --enable-avresample --disable-filter-resample --enable-avisynth --enable-gnutls --enable-ladspa --enable-l
  --enable-libass --enable-libbluray --enable-libs26 --enable-libcaca --enable-libcdio --enable-libcodecs2 --enable-libflite --enable-libfontcon
  --enable-libfreetype --enable-libfribidi --enable-libgsm --enable-libjack --enable-libmp3lame --enable-libmysofa --enable-libopenjp
  --enable-libopenmpt --enable-libopus --enable-libpulse --enable-librsync --enable-librubberband --enable-libshine --enable-lsnp --enable-libsoxr
  --enable-lspspeex --enable-libssh --enable-libtheora --enable-libtwolame --enable-libvidstab --enable-libvorbis --enable-libvpx --enable-libwavpack --
  --enable-libwebp --enable-libx265 --enable-libxml2 --enable-libxvid --enable-libzmq --enable-libzbi --enable-lv2 --enable-omx --enable-opengl --enable-o
  --enable-opengl --enable-sdl2 --enable-libcd1394 --enable-libdrm --enable-libiec61883 --enable-nvenc --enable-chromaprint --enable-frei0r --enabl
  --enable-shared
  libavutil      56. 31.100 / 56. 31.100
  libavcodec     58. 54.100 / 58. 54.100
  libavformat    58. 29.100 / 58. 29.100
  libavdevice    58.  8.100 / 58.  8.100
  libavfilter     7. 57.100 / 7. 57.100
  libavresample   4.  0.  0 / 4.  0.  0
  libbswscale    5.  5.100 / 5.  5.100
  libswresample   3.  5.100 / 3.  5.100
  libpostproc    55.  5.100 / 55.  5.100
Input #0: mov,mp4,m4a,3gp,3g2,mj2, from 'videoA.mp4':
Metadata:
  major_brand      : isom
  minor_version    : 512
  compatible_brands: isomiso2avc1mp41
  encoder          : Lavf58.29.100
Duration: 00:00:25.50, start: 0.000000, bitrate: 8 kb/s
Stream #0:0(und): Video: h264 (High) (avc1 / 0x31637661), yuv420p, 160x100, 6 kb/s, 20 fps, 20 tbr, 10240 tbn, 40 tbc (default)
Metadata:
  handler name     : VideoHandler
at least one output file must be specified

```

Figura 3.2: Taxa necessária para a transmissão do vídeoA.

De seguida, são acrescentados mais dois clientes (firefox e fflay) e é estudado o impacto dos mesmos no tráfego da rede. Para tal, foi capturado tráfego com o Wireshark para cada uma das três situações. Utilizando estes dados, e fazendo uso da funcionalidade Statistics > Conversations do programa e filtrando por TCP, conseguimos ter acesso ao número de fluxos e também à largura de banda em bps transmitida do servidor para os clientes. Concluimos que é gerado um fluxo por cada cliente e é usado o TCP como encapsulamento. Obtemos os seguintes resultados:

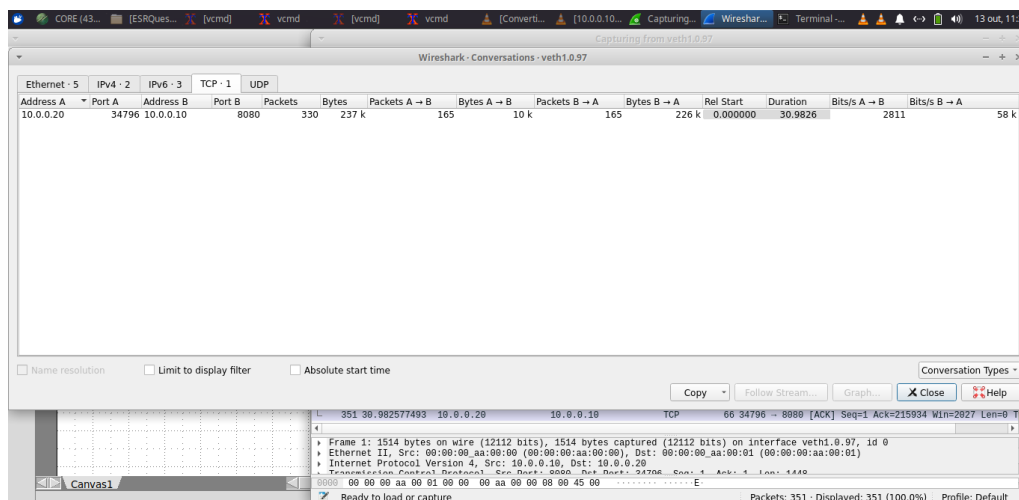


Figura 3.3: Dados referentes ao streaming do video 1 com um cliente.

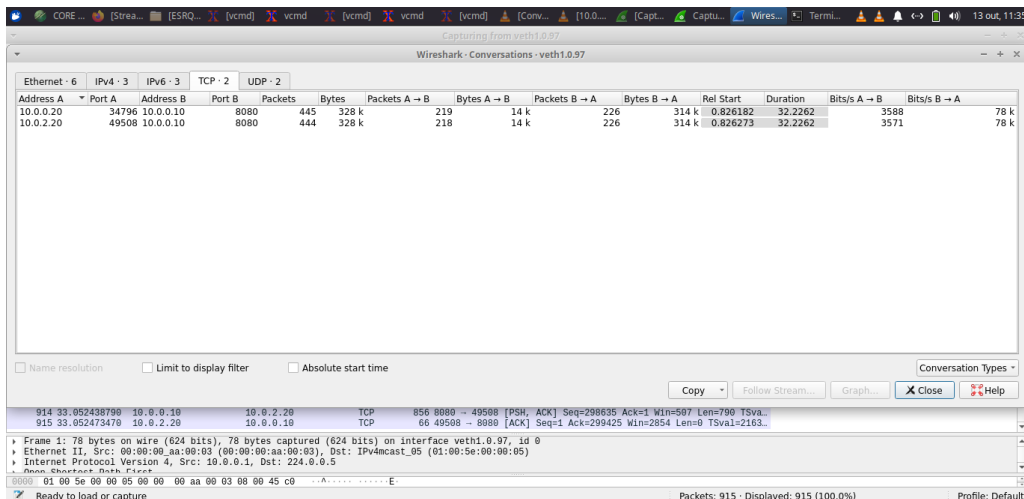


Figura 3.4: Dados referentes ao streaming do video 1 com dois clientes.

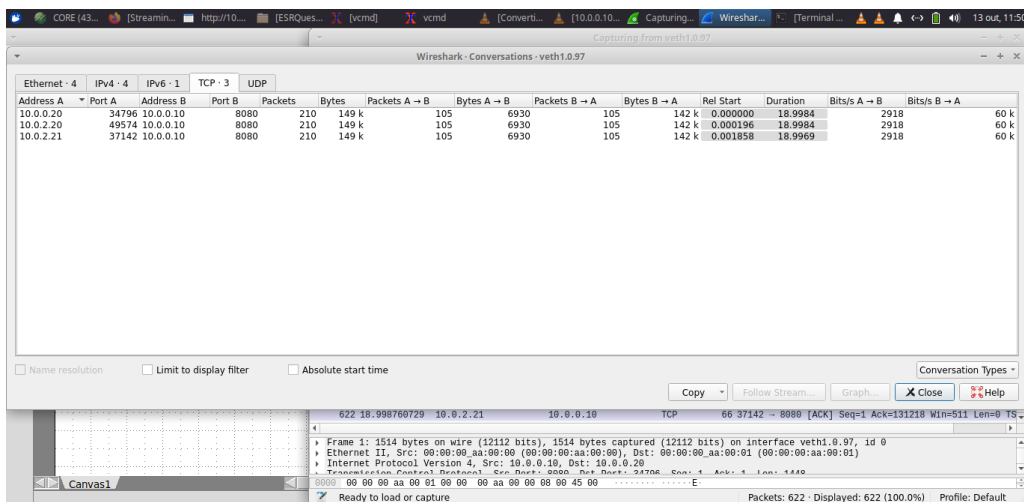


Figura 3.5: Dados referentes ao streaming do video 1 com três clientes.

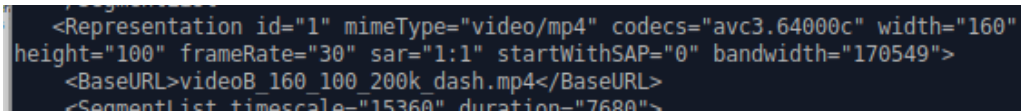
Através desta análise, concluímos que a solução não é escalável. As capturas do Wireshark permitiram verificar que o servidor responde aos pedidos dos clientes um de cada vez. Desta forma, se o número de clientes na rede aumentar de forma significativa, haverá perda de qualidade de serviço, sendo que este ficará mais lento. Se diminuirmos o bitrate conseguimos observar a diminuição da qualidade de serviço através dos dados estatísticos do fluxo (Wireshark).

4 Etapa 2: Streaming adaptativo sobre HTTP (MPEG-DASH)

4.1 Questão 2

Diga qual a largura de banda necessária, em bits por segundo, para que o cliente de streaming consiga receber o vídeo no firefox e qual a pilha protocolar usada neste cenário.

A largura de banda necessária para que o cliente consiga receber o vídeo terá de ser igual ou maior 170549 bps. Este valor é correspondente ao vídeo de menor resolução. A pilha protocolar utilizada neste cenário é TCP/IP.



```
<Representation id="1" mimeType="video/mp4" codecs="avc3.64000c" width="160"
height="100" frameRate="30" sar="1:1" startWithSAP="0" bandwidth="170549">
  <BaseURL>videoB_160_100_200k_dash.mp4</BaseURL>
  <SegmentList timescale="15360" duration="7680">
```

Figura 4.1: Visualização do ficheiro videomanifest.mpd

4.2 Questão 3

Ajuste o débito dos links da topologia de modo que o cliente no portátil Bela exiba o vídeo de menor resolução e o cliente no portátil Alladin exiba o vídeo com mais resolução. Mostre evidências.

Para o portátil Bela exibir o vídeo de menor resolução, limitou-se a largura de banda do link de acesso ao portátil para um valor ligeiramente superior à largura de banda mínima desse vídeo. Sendo que a largura de banda mínima é 170549bps decidimos colocar o portátil Bela com 200000bps. O valor da largura de banda do vídeo de resolução média é 442770bps, pelo que para definir o valor para o portátil Bela tivemos de garantir que este está no intervalo [170549bps,442770bps]. Quanto ao portátil Alladin, não é necessário limitar a largura de banda pois o DASH irá utilizar o de maior qualidade (caso a conexão o permita).

De seguida, apresentamos o resultado obtido:

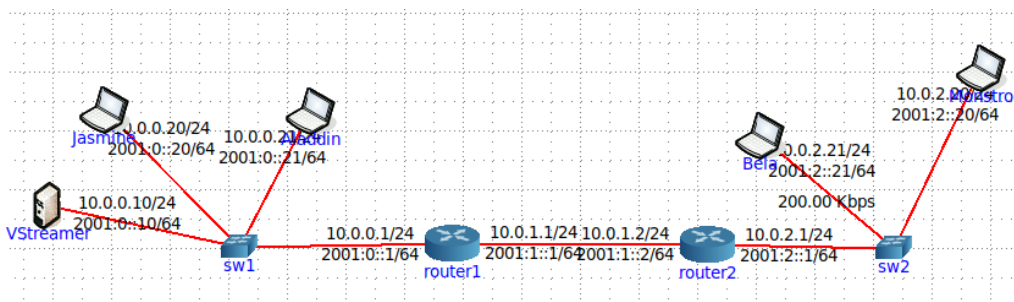


Figura 4.2: Ajuste do débito do portátil Bela.

No.	Time	Source	Destination	Protocol	Length	Info
4	0.002219722	10.0.0.21	10.0.0.10	HTTP	407	GET /videoB_640_400_1000k_dash.mp4 HTTP/1.1
1380	0.067961770	10.0.0.10	10.0.0.21	MP4	640	
1389	0.226193513	10.0.0.21	10.0.0.10	HTTP	407	GET /videoB_640_400_1000k_dash.mp4 HTTP/1.1
2856	0.397546745	10.0.0.21	10.0.0.10	HTTP	407	GET /videoB_640_400_1000k_dash.mp4 HTTP/1.1
4229	0.541324977	10.0.2.21	10.0.0.10	HTTP	404	GET /videoB_160_100_200k_dash.mp4 HTTP/1.1
4247	0.658786034	10.0.0.21	10.0.0.10	HTTP	407	GET /videoB_640_400_1000k_dash.mp4 HTTP/1.1
5641	0.707662000	10.0.0.10	10.0.0.21	MP4	640	
5663	1.049923961	10.0.0.21	10.0.0.10	HTTP	407	GET /videoB_640_400_1000k_dash.mp4 HTTP/1.1
7289	1.073234440	10.0.0.10	10.0.0.21	MP4	224	
7305	1.226379009	10.0.0.21	10.0.0.10	HTTP	407	GET /videoB_640_400_1000k_dash.mp4 HTTP/1.1
8798	1.278634516	10.0.0.10	10.0.0.21	MP4	640	
8817	1.485749972	10.0.0.21	10.0.0.10	HTTP	407	GET /videoB_640_400_1000k_dash.mp4 HTTP/1.1
10433	1.514658977	10.0.0.10	10.0.0.21	MP4	224	
10444	1.626990948	10.0.0.21	10.0.0.10	HTTP	407	GET /videoB_640_400_1000k_dash.mp4 HTTP/1.1
11839	1.714621188	10.0.0.10	10.0.0.21	MP4	640	
11848	1.818056723	10.0.0.21	10.0.0.10	HTTP	407	GET /videoB_640_400_1000k_dash.mp4 HTTP/1.1
13449	1.862332627	10.0.0.10	10.0.0.21	MP4	640	
13802	10.554280570	10.0.0.10	10.0.0.21	MP4	676	
13906	15.352132286	10.0.2.21	10.0.0.10	HTTP	404	GET /videoB_160_100_200k_dash.mp4 HTTP/1.1
13945	15.894981308	10.0.2.21	10.0.0.10	HTTP	404	GET /videoB_160_100_200k_dash.mp4 HTTP/1.1
14354	26.623278733	10.0.0.10	10.0.2.21	MP4	876	

Figura 4.3: Captura wireshark obtida.

4.3 Questão 4

Descreva o funcionamento do DASH neste caso concreto, referindo o papel do ficheiro MPD criado.

O DASH é uma ferramenta adaptativa de streaming de iniciativa da DASH Industry Forum para estabelecer um produto de qualidade para reprodutores de vídeo e áudio, no formato MPEG-DASH. Por outras palavras, isto significa que é permitida à stream trocar de bitrate tendo em conta a largura de banda disponível, de maneira a que o vídeo continue a reproduzir, ou seja, não fique em buffering. O ficheiro MPD é um ficheiro xml com informação sobre as várias streams e a largura de banda associada a cada uma delas, como também outros metadados como os codecs a usar e como o tipo MIME. Na página HTML é referenciado este ficheiro MPD e não os ficheiros multimédia, como acontece no caso de não usarmos técnicas adaptativas. O browser tendo em conta a largura de banda que tem disponível escolhe a melhor stream, dentro das possibilidades, que melhor consegue reproduzir.

5 Etapa 3: Streaming RTP/RTCP unicast sobre UDP e multicast com anúncios SAP

A topologia construída nesta etapa foi a seguinte:

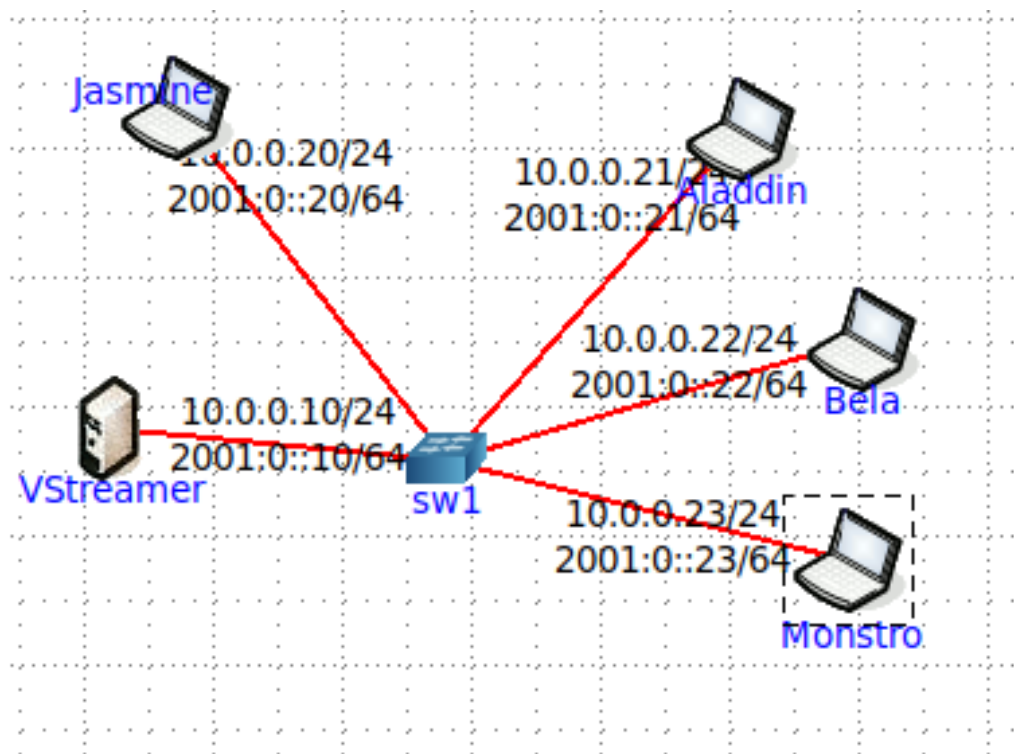


Figura 5.1: Topologia construída.

5.1 Questão 5

Compare o cenário unicast aplicado com o cenário multicast. Mostre vantagens e desvantagens na solução multicast ao nível da rede, no que diz respeito a escalabilidade (aumento do nº de clientes) e tráfego na rede. Tire as suas conclusões.

No cenário aplicado (Unicast), como estamos perante um servidor centralizado, é alocado um canal de transmissão de dados unicast para atender os pedidos de cada cliente. No entanto, como a banda do servidor é um recurso limitado, esta forma de implementação torna-se inviável quando muitos clientes precisam de obter respostas aos seus pedidos simultaneamente, pois a demanda por banda cresce linearmente com a taxa de chegada de pedidos.

Nas soluções multicast a transmissão de dados é feita para múltiplos clientes simultaneamente, isto é, os pacotes são transmitidos eficientemente para múltiplos pontos distintos ao mesmo tempo. Além disso, as mensagens só passam por um link uma única vez e somente são duplicadas quando o link para os destinatários se divide em duas direções. Isto permite o aumento da eficiência das comunicações de um para muitos, sendo, portanto, esta solução escalável. Além disso, como os switches vão mandar o vídeo apenas para os clientes que o peçam, a qualidade do mesmo irá ser preservada e o vídeo continuará a ser visualizável. Outra vantagem desta solução, é que permite o menor uso da largura de banda, uma vez que, os pacotes são enviados de acordo com as necessidades dos recetores, permitindo assim que a mesma não seja limitada pela largura de banda do access end do cliente. Além disso, com base no descrito, podemos concluir que permite a minimização do tráfego na rede. Por outro lado, existem também desvantagens no que toca ao cenário Multicast. Primeiramente, comparativamente com o cenário Unicast, não existe um mecanismo de correção de erros. Portanto, é uma solução que não é fiável, para contrariar isto, deve-se responsabilizar a camada aplicacional para a resolução deste problema. Além disso é de notar que como não existem mecanismos de controlo de congestão como no TCP, uma vez que o protocolo da camada de transporte utilizado neste tipo de cenário é o UDP, existe a possibilidade de haver congestão na rede.

Ethernet - 5		IPv4 - 2		IPv6 - 3		TCP		UDP - 2					
Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.0.0.10	35223	10.0.2.20	5555	488	303 k	488	303 k	0	0	0.000000	16.2499	149 k	
10.0.0.10	35224	10.0.2.20	5556	3	210	3	210	0	0	3.361568	10.0429	167	

Figura 5.2: Dados referentes ao streaming do vídeo com unicast.

Ethernet - 2		IPv4 - 2		IPv6		TCP		UDP - 3					
Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.0.0.10	43183	224.0.0.200	5555	878	548 k	878	548 k	0	0	0.000000	29.1493	150 k	
10.0.0.10	51245	224.2.127.254	9875	5	1830	5	1830	0	0	4.603650	20.0986	728	
10.0.0.10	43184	224.0.0.200	5556	5	350	5	350	0	0	4.604506	20.0979	139	

Figura 5.3: Dados referentes ao streaming do vídeo com multicast.

6 Conclusões

Com a realização deste trabalho foi possível aprofundar os conceitos abordados em sala de aula acerca de Streaming de áudio e vídeo.

Com o auxílio das ferramentas Core e Wireshark, conseguimos verificar que o uso do UDP para streaming é mais eficiente do que o uso do TCP.

Relativamente ao TCP, o uso de DASH em detrimento do HTTP simples sem adaptação dinâmica de débito permite que a stream mude a qualidade do vídeo de modo a adaptar-se à internet do cliente de forma a que este não tenha que esperar pelo buffering.

Por fim, foi feito Streaming sobre UDP, tanto unicast como multicast, para posterior comparação entre os dois.

Em conclusão, o grupo considera que realizou o trabalho com sucesso tendo aprofundado os seus conhecimentos em vídeo streaming.

Lista de Siglas e Acrónimos

HTTP HyperText Transfer Protocol

TCP Transmission Control Protocol

UDP User Datagram Protocol

SSH Secure SHell