



UNIVERSIDADE DO MINHO

MESTRADO EM ENGENHARIA INFORMÁTICA

Engenharia de Sistemas de Redes  
Trabalho Prático 1

Bernardo Saraiva (PG50259)

José Gonçalves (PG50519)

Daniel Azevedo (PG50311)

29-09-2022

# Conteúdo

<b>1</b>	<b>Etapa 1</b>	<b>3</b>
1.1	Questão 1 . . . . .	3
<b>2</b>	<b>Etapa 2</b>	<b>6</b>
2.1	Questão 2 . . . . .	6
2.2	Questão 3 . . . . .	7
2.3	Questão 4 . . . . .	9
<b>3</b>	<b>Etapa 3</b>	<b>11</b>
3.1	Questão 5 . . . . .	11
<b>4</b>	<b>Conclusão</b>	<b>14</b>

# Capítulo 1

## Etapa 1

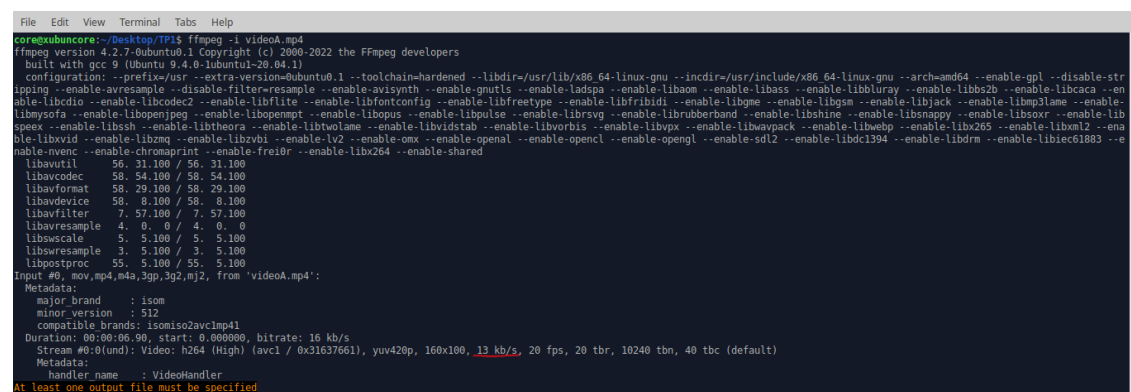
### 1.1 Questão 1

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 fplay). 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).

Para verificar a taxa em bps necessária, foi utilizado o comando `ffmpeg -i videoA.mp4`. Através do output do mesmo, podemos verificar que a taxa é de 13 Kb/s.

No entanto, podemos verificar que a taxa real é superior à necessária. No caso de só existir um cliente, temos uma taxa de 15 Kb/s. Quando temos dois clientes a aceder ao servidor, esta taxa passa para 16 Kb/s. Por fim quando são 3 cliente, a taxa é de 37 Kb/s.

Este aumento em relação à taxa teórica deve-se a perdas e as consecutivamente retransmissões.



```
File Edit View Terminal Tabs Help
core@ubuntu:~/Desktop/TP1$ 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-str
ipping --enable-avresample --disable-filter=resample --enable-avisynth --enable-gnutls --enable-ladspa --enable-libaom --enable-libbass --enable-libbluray --enable-libbs2b --enable-libcaca --en
able-libcdio --enable-libcodec2 --enable-libflite --enable-libfontconfig --enable-libfreetype --enable-libfribidi --enable-libgme --enable-libgsm --enable-libjack --enable-libmp3lame --enable-
libmysofa --enable-libopenjpeg --enable-libopenmpt --enable-libopus --enable-libpulse --enable-librav1e --enable-librubertand --enable-libshine --enable-libsnappy --enable-libsoxr --enable-lib
speex --enable-libssh --enable-libtheora --enable-libtwolame --enable-libvidstab --enable-libvorbis --enable-libvpx --enable-libwavpack --enable-libwebp --enable-libx264 --enable-libx265 --ena
ble-libxvid --enable-libzmq --enable-libzvbi --enable-lv2 --enable-omx --enable-opengl --enable-opencore --enable-opencl --enable-opengl --enable-sdl2 --enable-libcd1394 --enable-libdrm --enable-libiec61883 --e
nable-nvenc --enable-chromaprint --enable-frei0r --enable-libx264 --enable-shared
libavutil      56. 31.100 / 56. 31.100
libavcodec     58. 24.100 / 58. 24.100
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
    Duration: 00:00:06.90, start: 0.000000, bitrate: 16 kb/s
    Stream #0:0(und): Video: h264 (High) (avc1 / 0x31637661), yuv420p, 160x100, 13 kb/s, 20 fps, 20 tbr, 10240 tbn, 40 tbc (default)
  Metadata:
    handler_name     : VideoHandler
at least one output file must be specified
```

Figura 1.1: Taxa em bps

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
Frame	100.0	185	100.0	122342	116 k	0	0	0
Ethernet	100.0	185	2.1	2590	2469	0	0	0
Internet Protocol Version 6	0.5	1	0.0	40	38	0	0	0
Open Shortest Path First	0.5	1	0.0	36	34	1	36	34
Internet Protocol Version 4	99.5	184	3.0	3680	3509	0	0	0
Transmission Control Protocol	97.3	180	94.7	115820	110 k	169	99540	94 k
Hypertext Transfer Protocol	5.9	11	13.3	16324	15 k	9	13428	12 k
Malformed Packet	1.1	2	0.0	0	0	2	0	0
Open Shortest Path First	2.2	4	0.1	176	167	4	176	167

Figura 1.2: Taxa em bps real com um cliente(vlc)

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
Frame	100.0	389	100.0	255860	191 k	0	0	0
Ethernet	100.0	389	2.1	5446	4084	0	0	0
Internet Protocol Version 6	0.3	1	0.0	40	30	0	0	0
Open Shortest Path First	0.3	1	0.0	36	27	1	36	27
Internet Protocol Version 4	99.2	386	3.0	7720	5790	0	0	0
Transmission Control Protocol	97.9	381	94.7	242342	181 k	366	221296	165 k
Hypertext Transfer Protocol	3.9	15	8.3	21358	16 k	13	18462	13 k
Malformed Packet	0.5	2	0.0	0	0	2	0	0
Open Shortest Path First	1.3	5	0.1	220	165	5	220	165
Address Resolution Protocol	0.5	2	0.0	56	42	2	56	42

Figura 1.3: Taxa em bps real com dois cliente(vlc + firefox)

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
Frame	100.0	615	100.0	411853	360 k	0	0	0
Ethernet	100.0	615	2.1	8610	7528	0	0	0
Internet Protocol Version 4	100.0	615	3.0	12300	10 k	0	0	0
Transmission Control Protocol	99.3	611	94.9	390767	341 k	581	347490	303 k
Hypertext Transfer Protocol	4.9	30	10.5	43109	37 k	25	35869	31 k
Malformed Packet	0.8	5	0.0	0	0	5	0	0
Open Shortest Path First	0.7	4	0.0	176	153	4	176	153

Figura 1.4: Taxa em bps real com dois cliente(vlc + firefox + ffmpeg)

Relativamente ao encapsulamento usado, independente do número de clientes, estão presentes os 4 níveis da pilha protocolar, sendo eles: Ethernet (camada de ligação de dados), IPv4 (camada de rede), TCP (camada de transporte) e HTTP (camada de aplicação).

```

▶ Frame 3: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface veth1.0.3, id 0
▶ Ethernet II, Src: 00:00:00_aa:00:00 (00:00:00:aa:00:00), Dst: 00:00:00_aa:00:01 (00:00:00:aa:00:01)
▶ Internet Protocol Version 4, Src: 10.0.0.10, Dst: 10.0.0.20
▶ Transmission Control Protocol, Src Port: 8080, Dst Port: 43540, Seq: 2897, Ack: 1, Len: 1448
▶ Hypertext Transfer Protocol

```

Figura 1.5: Encapsulamento

No que diz respeito ao número de fluxo podemos verificar, através das figuras 1.6, 1.7 e 1.8, que quando temos apenas um cliente (vlc) é gerado um único fluxo. Passando para dois clientes (vlc + firefox) temos dois fluxos gerados. Por fim, existe três fluxos gerados quando passamos a ter três clientes (vlc + firefox + ffplay).

Ethernet · 2		IPv4 · 2		IPv6	TCP · 1		UDP						
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
10.0.0.20	43540	10.0.0.10	8080	110	73 k	55	3630	55	70 k	0.000000	5.4425	5335	102 k

Figura 1.6: Fluxo 1 cliente (vlc)

Ethernet · 4		IPv4 · 3		IPv6 · 1		TCP · 2		UDP						
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	
10.0.0.20	43540	10.0.0.10	8080	216	147 k	108	7128	108	139 k	0.000000	10.6664	5346	104 k	
10.0.2.20	33398	10.0.0.10	8080	165	108 k	83	5780	82	102 k	3.071857	7.5946	6088	107 k	

Figura 1.7: Fluxo 2 cliente (vlc + firefox)

Ethernet · 3		IPv4 · 4		IPv6		TCP · 3		UDP						
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	
10.0.0.20	60762	10.0.0.10	8080	94	63 k	47	3102	47	60 k	0.000000	3.9304	6313	123 k	
10.0.2.20	44810	10.0.0.10	8080	94	63 k	47	3102	47	60 k	0.000204	3.9304	6313	123 k	
10.0.2.21	34204	10.0.0.10	8080	94	63 k	47	3102	47	60 k	0.001470	3.9298	6314	123 k	

Figura 1.8: Fluxo 3 cliente (vlc + firefox + ffplay)

Através da análise das capturas do wireshark efetuadas, foi possível observar que o servidor responde a cada cliente individualmente, independentemente se o pedido é o mesmo. Assim, podemos concluir que a solução não tem grande escalabilidade, pois com o aumento do número de clientes iria existir perda de qualidade do serviço, e conseqüentemente iria ficar mais lento.

# Capítulo 2

## Etapa 2

### 2.1 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.

Para que seja possível o cliente de streaming conseguir receber o vídeo no firefox, é necessário que a conexão permita uma largura de banda maior do que o bit rate do vídeo, já que é necessário considerar o overhead associado à transmissão dos pacotes. Neste caso, através da análise ao ficheiro *videoManifest.mpd*, é possível inferir o bit rate de cada um dos vídeos nas diferentes resoluções.

```
<Representation id="1" mimeType="video/mp4" codecs="avc3.64000c" width="160" height="100" frameRate="30000/1001" sar="1:1" startWithSAP="0" bandwidth="78530">
  <BaseURL>videoB_160_100_200k_dash.mp4</BaseURL>
  <SegmentList timescale="30000" duration="15000">
    <SegmentURL mediaRange="926-5296" indexRange="926-969"/>
    <SegmentURL mediaRange="5297-8160" indexRange="5297-5340"/>
    <SegmentURL mediaRange="8161-11551" indexRange="8161-8204"/>
    <SegmentURL mediaRange="11552-18737" indexRange="11552-11595"/>
    <SegmentURL mediaRange="18738-23552" indexRange="18738-18781"/>
    <SegmentURL mediaRange="23553-29010" indexRange="23553-23596"/>
    <SegmentURL mediaRange="29011-38095" indexRange="29011-29054"/>
    <SegmentURL mediaRange="38096-42014" indexRange="38096-38139"/>
    <SegmentURL mediaRange="42015-46616" indexRange="42015-42058"/>
    <SegmentURL mediaRange="46617-49669" indexRange="46617-46660"/>
    <SegmentURL mediaRange="49670-58028" indexRange="49670-49713"/>
    <SegmentURL mediaRange="58029-58628" indexRange="58029-58072"/>
  </SegmentList>
</Representation>
```

Figura 2.1: Dados referentes ao vídeo na dimensão 160 x 100

Da figura 2.1 é possível verificar que é necessário uma largura de banda superior a 78530 bps, como descrito no campo *bandwidth* da primeira linha.

```
<Representation id="2" mimeType="video/mp4" codecs="avc3.640014" width="320" height="200" frameRate="30000/1001" sar="1:1" startWithSAP="0" bandwidth="181115">
<BaseURL>videoB 320 200 500k dash.mp4</BaseURL>
<SegmentList timescale="30000" duration="15000">
<SegmentURL mediaRange="926-10622" indexRange="926-969"/>
<SegmentURL mediaRange="10623-17767" indexRange="10623-10666"/>
<SegmentURL mediaRange="17768-24885" indexRange="17768-17811"/>
<SegmentURL mediaRange="24886-41729" indexRange="24886-24929"/>
<SegmentURL mediaRange="41730-53500" indexRange="41730-41773"/>
<SegmentURL mediaRange="53501-60105" indexRange="53501-53544"/>
<SegmentURL mediaRange="60106-87929" indexRange="60106-60149"/>
<SegmentURL mediaRange="87930-98278" indexRange="87930-87973"/>
<SegmentURL mediaRange="98279-109040" indexRange="98279-98322"/>
<SegmentURL mediaRange="109041-115347" indexRange="109041-109084"/>
<SegmentURL mediaRange="115348-134381" indexRange="115348-115391"/>
<SegmentURL mediaRange="134382-135216" indexRange="134382-134425"/>
</SegmentList>
</Representation>
```

Figura 2.2: Dados referentes ao vídeo na dimensão 320 x 200

De igual modo, através da figura 2.2 é possível verificar que a largura de banda mínima para transmitir este vídeo é de 181115 bps.

```
<Representation id="3" mimeType="video/mp4" codecs="avc3.64001e" width="640" height="400" frameRate="30000/1001" sar="1:1" startWithSAP="0" bandwidth="445335">
<BaseURL>videoB 640 400 1000k dash.mp4</BaseURL>
<SegmentList timescale="30000" duration="15000">
<SegmentURL mediaRange="926-21958" indexRange="926-969"/>
<SegmentURL mediaRange="21959-41018" indexRange="21959-22002"/>
<SegmentURL mediaRange="41019-58648" indexRange="41019-41062"/>
<SegmentURL mediaRange="58649-99910" indexRange="58649-58692"/>
<SegmentURL mediaRange="99911-126852" indexRange="99911-99954"/>
<SegmentURL mediaRange="126853-158267" indexRange="126853-126896"/>
<SegmentURL mediaRange="158268-215449" indexRange="158268-158311"/>
<SegmentURL mediaRange="215450-241435" indexRange="215450-215493"/>
<SegmentURL mediaRange="241436-268339" indexRange="241436-241479"/>
<SegmentURL mediaRange="268340-284634" indexRange="268340-268383"/>
<SegmentURL mediaRange="284635-331293" indexRange="284635-284678"/>
<SegmentURL mediaRange="331294-332477" indexRange="331294-331337"/>
</SegmentList>
</Representation>
```

Figura 2.3: Dados referentes ao vídeo na dimensão 640 x 400

Por fim, pela figura 2.3 verifica-se que o vídeo na dimensão 640 necessita de uma largura de banda superior a 445335 bps para que seja possível ser transmitido.

As camadas da pilha protocolar envolvidas neste processo são a camada de rede (através do protocolo IP), a camada de Transporte (com recurso ao TCP) e a de Aplicação (que trabalha com HTTP).

## 2.2 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.**

De modo a ajustar o débito dos links da topologia para que o portátil Bela exiba o vídeo com menor resolução, efetuou-se algumas tentativas de ajuste da largura de banda para a ligação ao portátil Bela. Deste modo, através de capturas Wireshark (ou análise através da aba Network do Firefox).

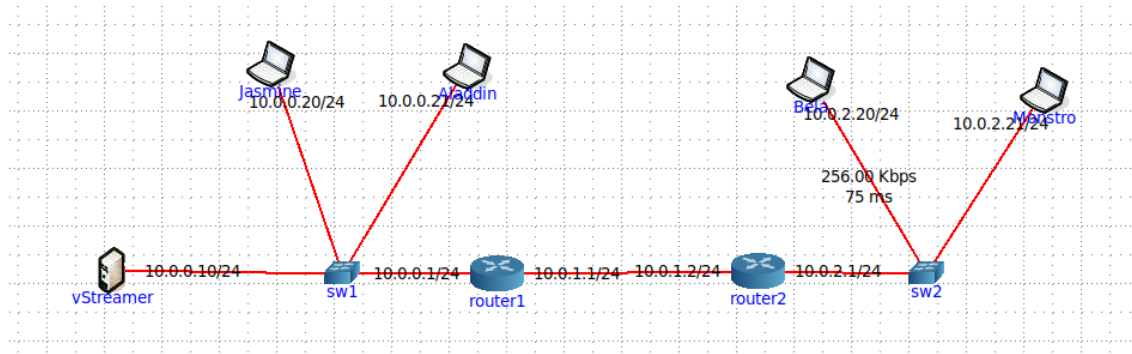


Figura 2.4: Topologia com restrição de largura de banda para o portátil Bela

Efetuada uma captura com limite da largura de banda a 256kbps no link que leva ao portátil Bela, verificou-se que o video apresentado é na resolução mais baixa (160 x 100).

No.	Time	Source	Destination	Protocol	Length	Info
42	32.511800097	10.0.2.20	10.0.0.10	HTTP	381	GET /favicon.ico HTTP/1.1
44	32.511971827	10.0.0.10	10.0.2.20	HTTP	741	HTTP/1.1 404 Not Found (text/html)
54	33.666090556	10.0.2.20	10.0.0.10	HTTP	401	GET /videoB_640_400_1000k_dash.mp4 HTTP/1.1
197	36.776680520	10.0.2.20	10.0.0.10	HTTP	402	GET /videoB_320_200_500k_dash.mp4 HTTP/1.1
252	37.661120836	10.0.2.20	10.0.0.10	HTTP	401	GET /videoB_160_100_200k_dash.mp4 HTTP/1.1
347	39.449774856	10.0.0.10	10.0.2.20	MP4	979	
359	39.801553980	10.0.2.20	10.0.0.10	HTTP	402	GET /videoB_160_100_200k_dash.mp4 HTTP/1.1
454	41.582073558	10.0.0.10	10.0.2.20	MP4	979	
466	41.916067570	10.0.2.20	10.0.0.10	HTTP	402	GET /videoB_160_100_200k_dash.mp4 HTTP/1.1
559	43.698029812	10.0.0.10	10.0.2.20	MP4	979	
572	44.038686762	10.0.2.20	10.0.0.10	HTTP	402	GET /videoB_160_100_200k_dash.mp4 HTTP/1.1
664	45.820090805	10.0.0.10	10.0.2.20	MP4	979	
677	46.327746755	10.0.2.20	10.0.0.10	HTTP	402	GET /videoB_160_100_200k_dash.mp4 HTTP/1.1

Figura 2.5: Captura com limite da ligação ao portátil Bela

Efetuada outros testes, testou-se o limite de largura de banda em 512kbps, verificando-se que o video apresentado foi na resolução intermédia (320 x 200).

No.	Time	Source	Destination	Protocol	Length	Info
73	72.293162917	10.0.2.20	10.0.0.10	HTTP	381	GET /favicon.ico HTTP/1.1
75	72.293268485	10.0.0.10	10.0.2.20	HTTP	741	HTTP/1.1 404 Not Found (text/html)
85	72.783410895	10.0.2.20	10.0.0.10	HTTP	401	GET /videoB_640_400_1000k_dash.mp4 HTTP/1.1
166	73.718252881	10.0.2.20	10.0.0.10	HTTP	402	GET /videoB_320_200_500k_dash.mp4 HTTP/1.1
350	75.896344334	10.0.0.10	10.0.2.20	MP4	824	
355	76.012385895	10.0.2.20	10.0.0.10	HTTP	402	GET /videoB_320_200_500k_dash.mp4 HTTP/1.1
527	78.332354026	10.0.0.10	10.0.2.20	MP4	824	
535	78.522443744	10.0.2.20	10.0.0.10	HTTP	402	GET /videoB_320_200_500k_dash.mp4 HTTP/1.1
708	80.663259337	10.0.0.10	10.0.2.20	MP4	824	
719	81.313001233	10.0.2.20	10.0.0.10	HTTP	402	GET /videoB_320_200_500k_dash.mp4 HTTP/1.1
888	83.500292568	10.0.2.20	10.0.0.10	HTTP	404	GET /videoB_320_200_500k_dash.mp4 HTTP/1.1
1037	85.225114170	10.0.2.20	10.0.0.10	HTTP	404	GET /videoB_320_200_500k_dash.mp4 HTTP/1.1
1201	87.890449806	10.0.0.10	10.0.2.20	MP4	824	

Figura 2.6: Captura com limitação a 512kbps do link ao portátil Bela

Para obter o video de maior resolução no portátil Alladin, simplesmente não se limitou a largura de banda, como se verifica em seguida.



http					
No.	Time	Source	Destination	Protocol	Length Info
20	21.151786278	10.0.0.21	10.0.0.10	HTTP	428 GET /video_manifest.mpd HTTP/1.1
25	21.152008395	10.0.0.10	10.0.0.21	HTTP/1.1	200 OK
34	21.246179033	10.0.0.21	10.0.0.10	HTTP	381 GET /favicon.ico HTTP/1.1
36	21.246384724	10.0.0.10	10.0.0.21	HTTP	741 HTTP/1.1 404 Not Found (text/html)
43	21.849914709	10.0.0.21	10.0.0.10	HTTP	371 GET /video_manifest_init.mpd HTTP/1.1
45	21.850118910	10.0.0.10	10.0.0.21	MP4	1060
52	21.952181244	10.0.0.21	10.0.0.10	HTTP	401 GET /videoB_640_400_1000k_dash.mpd HTTP/1.1
480	21.966545874	10.0.0.10	10.0.0.21	MP4	1157
501	22.034068468	10.0.0.21	10.0.0.10	HTTP	403 GET /videoB_640_400_1000k_dash.mpd HTTP/1.1
932	22.043886918	10.0.0.10	10.0.0.21	MP4	1157
950	22.106009773	10.0.0.21	10.0.0.10	HTTP	403 GET /videoB_640_400_1000k_dash.mpd HTTP/1.1
1348	22.117658450	10.0.0.10	10.0.0.21	MP4	1157
1370	22.263476654	10.0.0.21	10.0.0.10	HTTP	403 GET /videoB_640_400_1000k_dash.mpd HTTP/1.1
1766	22.280940601	10.0.0.10	10.0.0.21	MP4	1157
1779	22.351221192	10.0.0.21	10.0.0.10	HTTP	404 GET /videoB_640_400_1000k_dash.mpd HTTP/1.1
2147	22.359782783	10.0.0.10	10.0.0.21	MP4	1157
2166	22.455045297	10.0.0.21	10.0.0.10	HTTP	405 GET /videoB_640_400_1000k_dash.mpd HTTP/1.1
2583	22.479291158	10.0.0.10	10.0.0.21	MP4	1157
2596	22.579275933	10.0.0.21	10.0.0.10	HTTP	405 GET /videoB_640_400_1000k_dash.mpd HTTP/1.1
2995	22.591743417	10.0.0.10	10.0.0.21	MP4	1157
3015	22.649857233	10.0.0.21	10.0.0.10	HTTP	405 GET /videoB_640_400_1000k_dash.mpd HTTP/1.1
3404	22.660156447	10.0.0.10	10.0.0.21	MP4	1157
3419	22.685628622	10.0.0.21	10.0.0.10	HTTP	405 GET /videoB_640_400_1000k_dash.mpd HTTP/1.1
3796	22.698036742	10.0.0.10	10.0.0.21	MP4	1157
3803	22.731713220	10.0.0.21	10.0.0.10	HTTP	405 GET /videoB_640_400_1000k_dash.mpd HTTP/1.1
4186	22.744211925	10.0.0.10	10.0.0.21	MP4	1157
4205	22.767804058	10.0.0.21	10.0.0.10	HTTP	405 GET /videoB_640_400_1000k_dash.mpd HTTP/1.1
4615	22.780239107	10.0.0.10	10.0.0.21	MP4	1157
4636	22.806094187	10.0.0.21	10.0.0.10	HTTP	405 GET /videoB_640_400_1000k_dash.mpd HTTP/1.1
5034	22.818082570	10.0.0.10	10.0.0.21	MP4	1157

Figura 2.7: Captura sem limitação no Portátil Alladin

## 2.3 Questão 4

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

DASH (Dynamic Adaptive Streaming over HTTP) é uma técnica de streaming conhecida por permitir um bit rate adaptativo onde um ficheiro multimédia é particionado em um ou mais segmentos e entregue a um cliente através de HTTP. Este método permite que o conteúdo se adapte dinamicamente à largura de banda disponível, sendo bastante comum e ocorre quando por vezes se repara que um conteúdo multimédia alterna automaticamente para uma imagem de qualidade inferior ou superior de modo a ajustar às condições da rede. O Youtube e a Netflix são exemplos de plataformas que contam com esta técnica.

Neste caso, o ficheiro *.mpd* (media presentation description) é de extrema importância para o DASH, no sentido em que estes ficheiros contêm várias informações e parâmetros para o streaming do video em questão (nas várias resoluções), nomeadamente a largura de banda. A figura 2.8 é referente ao ficheiro mpd do exercício que se trata no presente trabalho prático.

```

<Representation id="1" mimeType="video/mp4" codecs="avc3.64000c" width="160" height="100" frameRate="30000/1001" sar="1:1" startWithSAP="0" bandwidth="78530">
<BaseURL>videoB_160_100_200k_dash.mp4</BaseURL>
<SegmentList timescale="30000" duration="15000">
<SegmentURL mediaRange="926-5296" indexRange="926-969"/>
<SegmentURL mediaRange="5297-8160" indexRange="5297-5340"/>
<SegmentURL mediaRange="8161-11551" indexRange="8161-8204"/>
<SegmentURL mediaRange="11552-18737" indexRange="11552-11595"/>
<SegmentURL mediaRange="18738-23552" indexRange="18738-18781"/>
<SegmentURL mediaRange="23553-29010" indexRange="23553-23596"/>
<SegmentURL mediaRange="29011-38095" indexRange="29011-29054"/>
<SegmentURL mediaRange="38096-42014" indexRange="38096-38139"/>
<SegmentURL mediaRange="42015-46616" indexRange="42015-42058"/>
<SegmentURL mediaRange="46617-49669" indexRange="46617-46660"/>
<SegmentURL mediaRange="49670-58028" indexRange="49670-49713"/>
<SegmentURL mediaRange="58029-58628" indexRange="58029-58072"/>
</SegmentList>
</Representation>
<Representation id="2" mimeType="video/mp4" codecs="avc3.640014" width="320" height="200" frameRate="30000/1001" sar="1:1" startWithSAP="0" bandwidth="181115">
<BaseURL>videoB_320_200_500k_dash.mp4</BaseURL>
<SegmentList timescale="30000" duration="15000">
<SegmentURL mediaRange="926-10622" indexRange="926-969"/>
<SegmentURL mediaRange="10623-17767" indexRange="10623-10666"/>
<SegmentURL mediaRange="17768-24885" indexRange="17768-17811"/>
<SegmentURL mediaRange="24886-41729" indexRange="24886-24929"/>
<SegmentURL mediaRange="41730-53500" indexRange="41730-41773"/>
<SegmentURL mediaRange="53501-66105" indexRange="53501-53544"/>
<SegmentURL mediaRange="66106-87929" indexRange="66106-66149"/>
<SegmentURL mediaRange="87930-98278" indexRange="87930-87973"/>
<SegmentURL mediaRange="98279-109040" indexRange="98279-98322"/>
<SegmentURL mediaRange="109041-115347" indexRange="109041-109084"/>
<SegmentURL mediaRange="115348-134381" indexRange="115348-115391"/>
<SegmentURL mediaRange="134382-135216" indexRange="134382-134425"/>
</SegmentList>
</Representation>
<Representation id="3" mimeType="video/mp4" codecs="avc3.64001e" width="640" height="400" frameRate="30000/1001" sar="1:1" startWithSAP="0" bandwidth="445335">
<BaseURL>videoB_640_400_1000k_dash.mp4</BaseURL>
<SegmentList timescale="30000" duration="15000">
<SegmentURL mediaRange="926-21958" indexRange="926-969"/>
<SegmentURL mediaRange="21959-41018" indexRange="21959-22002"/>
<SegmentURL mediaRange="41019-58648" indexRange="41019-41062"/>
<SegmentURL mediaRange="58649-99910" indexRange="58649-58692"/>
<SegmentURL mediaRange="99911-126852" indexRange="99911-99954"/>
<SegmentURL mediaRange="126853-158267" indexRange="126853-126896"/>
<SegmentURL mediaRange="158268-215449" indexRange="158268-158311"/>
<SegmentURL mediaRange="215450-241435" indexRange="215450-215493"/>
<SegmentURL mediaRange="241436-268339" indexRange="241436-241479"/>
<SegmentURL mediaRange="268340-284634" indexRange="268340-268383"/>
<SegmentURL mediaRange="284635-331293" indexRange="284635-284678"/>
<SegmentURL mediaRange="331294-332477" indexRange="331294-331337"/>
</SegmentList>
</Representation>

```

Figura 2.8: Ficheiro mpd referente ao trabalho prático

No caso da transmissão do exercício 2, foi possível verificar este mecanismo, sendo que esta técnica começa por tentar transmitir com a maior resolução disponível, mas logo se apercebe que não tem largura de banda suficiente para tal, reduzindo a capacidade sucessivamente até que esta se encaixe na largura de banda disponível.

Status	Method	Domain	File	Initiator	Type	Transferred	Size	Time	Other
200	GET	10.0.0.10:9999	video_dash.html	document	html	cached	331 B	0 ms	
200	GET	10.0.0.10:9999	dash.all.debug.js	script	js	cached	0 B	0 ms	
200	GET	10.0.0.10:9999	favicon.ico	FaviconLoader.get(186 (img))	html	cached	487 B	0 ms	
200	GET	10.0.0.10:9999	video_manifest.mpd	media	plain	cached	4.16 KB	0 ms	
200	GET	10.0.0.10:9999	video_manifest_init.mpd	dash.all.debug.js:55606 (xhr)	plain	cached	4.16 KB	0 ms	
200	GET	10.0.0.10:9999	video_manifest_init.mpd	dash.all.debug.js:55606 (xhr)	mp4	cached	792 B	0 ms	
200	GET	10.0.0.10:9999	videoB_640_400_1000k_dash.mp4	dash.all.debug.js:55606 (xhr)	mp4	324.89 KB	324.89 KB	1617 ms	
200	GET	10.0.0.10:9999	video_manifest_init.mpd	dash.all.debug.js:55606 (xhr)	mp4	cached	792 B	0 ms	
200	GET	10.0.0.10:9999	videoB_320_200_500k_dash.mp4	dash.all.debug.js:55606 (xhr)	mp4	132.25 KB	132.05 KB	2700 ms	
200	GET	10.0.0.10:9999	video_manifest_init.mpd	dash.all.debug.js:55606 (xhr)	mp4	cached	792 B	0 ms	
200	GET	10.0.0.10:9999	videoB_160_100_200k_dash.mp4	dash.all.debug.js:55606 (xhr)	mp4	57.45 KB	57.25 KB	1070 ms	
200	GET	10.0.0.10:9999	video_manifest_init.mpd	dash.all.debug.js:55606 (xhr)	mp4	cached	792 B	0 ms	
200	GET	10.0.0.10:9999	videoB_160_100_200k_dash.mp4	dash.all.debug.js:55606 (xhr)	mp4	57.45 KB	57.25 KB	1068 ms	
200	GET	10.0.0.10:9999	video_manifest_init.mpd	dash.all.debug.js:55606 (xhr)	mp4	cached	792 B	0 ms	
200	GET	10.0.0.10:9999	videoB_160_100_200k_dash.mp4	dash.all.debug.js:55606 (xhr)	mp4	57.45 KB	57.25 KB	1069 ms	

Figura 2.9: Exemplo em que o mecanismo DASH ocorre

Como é possível verificar pela figura 2.9, o mecanismo DASH interviu na transmissão, pelo que, por falta de largura de banda, foi recorrendo sucessivamente a um formato de qualidade inferior para que a transmissão pudesse ser efetuada.

## Capítulo 3

### Etapa 3

#### 3.1 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.

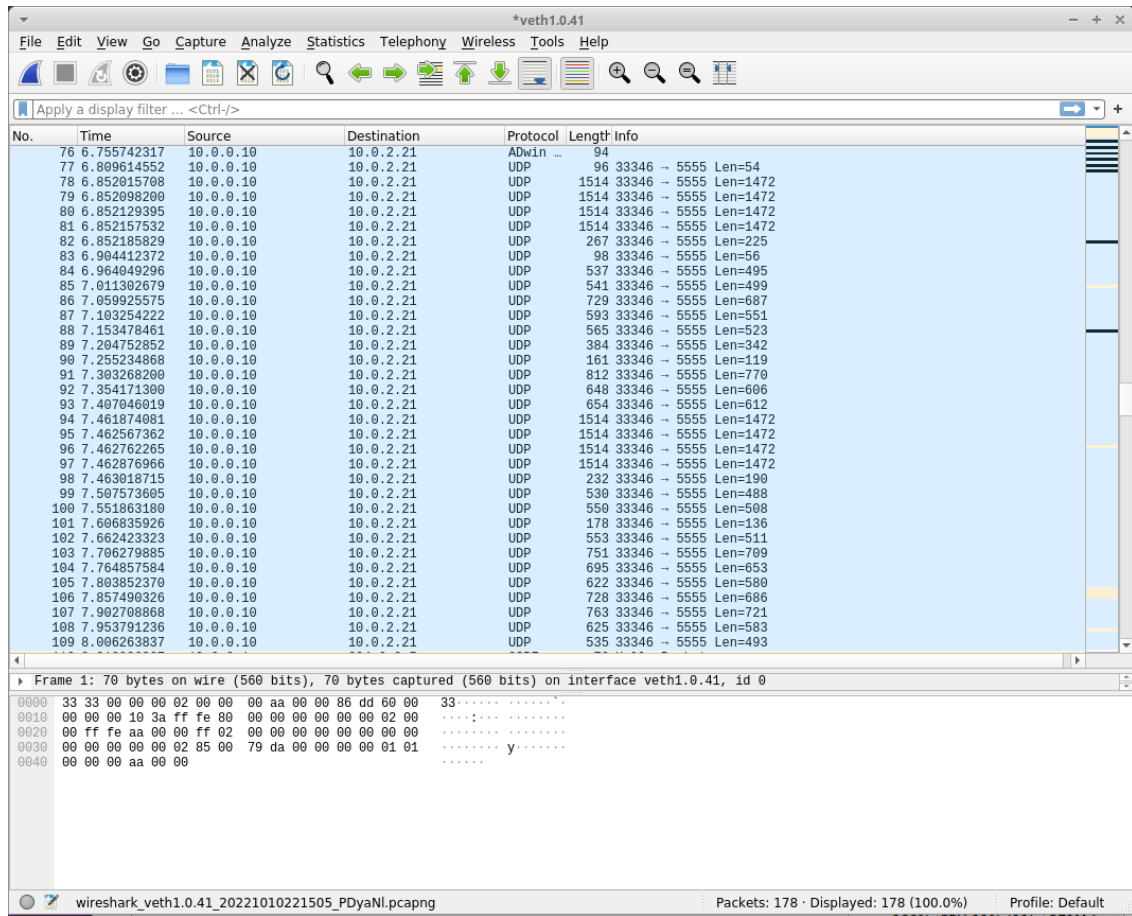


Figura 3.1: Captura Wireshark no link de saída do servidor com transmissão unicast

No.	Time	Source	Destination	Protocol	Length	Info
33	1.153239875	10.0.0.10	224.0.0.200	UDP	381	60541 → 5555 Len=339
34	1.199180627	10.0.0.10	224.0.0.200	UDP	1101	60541 → 5555 Len=1059
35	1.251719379	10.0.0.10	224.0.0.200	UDP	1151	60541 → 5555 Len=1109
36	1.308634298	10.0.0.10	224.0.0.200	UDP	105	60541 → 5555 Len=63
37	1.354140242	10.0.0.10	224.0.0.200	Adwin ...	94	
38	1.405134307	10.0.0.10	224.0.0.200	UDP	885	60541 → 5555 Len=843
39	1.447980708	10.0.0.10	224.0.0.200	UDP	1514	60541 → 5555 Len=1472
40	1.448068441	10.0.0.10	224.0.0.200	UDP	1514	60541 → 5555 Len=1472
41	1.448104203	10.0.0.10	224.0.0.200	UDP	1514	60541 → 5555 Len=1472
42	1.448135314	10.0.0.10	224.0.0.200	UDP	1514	60541 → 5555 Len=1472
43	1.448166072	10.0.0.10	224.0.0.200	UDP	200	60541 → 5555 Len=158
44	1.502411657	10.0.0.10	224.0.0.200	UDP	735	60541 → 5555 Len=693
45	1.553471020	10.0.0.10	224.0.0.200	UDP	497	60541 → 5555 Len=455
46	1.598250794	10.0.0.10	224.0.0.200	UDP	807	60541 → 5555 Len=765
47	1.660082612	10.0.0.10	224.0.0.200	UDP	608	60541 → 5555 Len=566
48	1.7052411995	10.0.0.10	224.0.0.200	UDP	116	60541 → 5555 Len=74
49	1.765778988	10.0.0.10	224.0.0.200	UDP	102	60541 → 5555 Len=60
50	1.797890164	10.0.0.10	224.0.0.200	UDP	100	60541 → 5555 Len=58
51	1.851196090	10.0.0.10	224.0.0.200	Adwin ...	94	
52	1.904113997	10.0.0.10	224.0.0.200	Adwin ...	94	
53	1.946013415	10.0.0.10	224.0.0.200	Adwin ...	94	
54	1.946450902	10.0.0.10	224.0.0.200	Adwin ...	94	
55	2.051528344	10.0.0.10	224.0.0.200	UDP	1514	60541 → 5555 Len=1472
56	2.052524133	10.0.0.10	224.0.0.200	UDP	1514	60541 → 5555 Len=1472
57	2.053170340	10.0.0.10	224.0.0.200	UDP	1514	60541 → 5555 Len=1472
58	2.053370185	10.0.0.10	224.0.0.200	UDP	1514	60541 → 5555 Len=1472
59	2.053945922	10.0.0.10	224.0.0.200	UDP	223	60541 → 5555 Len=181
60	2.099267739	10.0.0.10	224.0.0.200	UDP	188	60541 → 5555 Len=146
61	2.152646404	10.0.0.10	224.0.0.200	UDP	184	60541 → 5555 Len=142
62	2.291031728	10.0.0.10	224.0.0.200	UDP	150	60541 → 5555 Len=108
63	2.246176688	10.0.0.10	224.0.0.200	UDP	151	60541 → 5555 Len=109
64	2.303853443	10.0.0.10	224.0.0.200	UDP	139	60541 → 5555 Len=97
65	2.349101263	10.0.0.10	224.0.0.200	UDP	136	60541 → 5555 Len=94
66	2.402015361	10.0.0.10	224.0.0.200	UDP	140	60541 → 5555 Len=98
67	2.446101367	10.0.0.10	224.0.0.200	UDP	151	60541 → 5555 Len=109
68	2.501642818	10.0.0.10	224.0.0.200	UDP	135	60541 → 5555 Len=93
69	2.557113558	10.0.0.10	224.0.0.200	UDP	131	60541 → 5555 Len=89
70	2.601286976	10.0.0.10	224.0.0.200	UDP	131	60541 → 5555 Len=89
71	2.650586152	10.0.0.10	224.0.0.200	UDP	1514	60541 → 5555 Len=1472
72	2.657064924	10.0.0.10	224.0.0.200	UDP	1514	60541 → 5555 Len=1472
73	2.657150892	10.0.0.10	224.0.0.200	UDP	1514	60541 → 5555 Len=1472

Frame 1: 1154 bytes on wire (9232 bits), 1154 bytes captured (9232 bits) on interface veth1.0.9f, id 0

```

0000 01 00 5e 00 00 c8 00 00 00 aa 00 00 08 00 45 00  .A.....E
0010 04 74 cf 17 40 00 ff 11 bd 0e 0a 00 00 0a e0 00  .t. @.....
0020 00 c8 ec 7d 15 b3 04 60 af 73 80 e0 17 1e 4c e9  .}.....s...L
0030 22 63 c7 bb c0 25 00 00 01 b6 53 f0 23 ff ff f8  "c...%...S #...
0040 08 07 5b 6f d1 31 fc 21 0f a0 fc 10 42 04 e8 1d  c[o-1!...B...
0050 fd 00 ec f0 33 20 c0 87 e0 50 09 03 ea 0a 00 78  ...3...P....x

```

Ready to load or capture Packets: 266 · Displayed: 266 (100.0%) Profile: Default

Figura 3.2: Captura Wireshark no link de saída do servidor com transmissão unicast

Ao comparar o cenário unicast (transmissão baseada numa sessão entre o servidor e um cliente) com o cenário multicast (transmissão de um servidor para vários clientes), tendo como referência as figuras 3.1 e 3.2 é possível perceber vários fatores.

Cada cliente unicast ao conectar-se com o sevidor utiliza largura de banda adicional de forma a manter a conexão. Por exemplo, uma rede com 10 clientes a reproduzir uma stream de 10(kbps) geraria um tráfego de pelo menos 1000(kbps), já que o servidor terá de criar uma stream para cada cliente.

Em oposição, num cenário multicast o servidor de origem não encaminha pacotes para todos os clientes de forma direta, uma vez que confia em routers com suporte à tecnologia multicast ou switches como é apresentado na topologia em estudo para fazer o encaminhamento de pacotes. Por este motivo, cada cliente não gera overhead adicional tornando, na perspetiva do servidor, irrelevante o número de clientes, já que o servidor apenas transmite uma única stream de dados para o switch.

Por estes motivos, conclui-se que o cenário multicast é mais facilmente escalável e gere melhor o tráfego na rede, já que evita o envio de pacotes redundantes.

## Capítulo 4

# Conclusão

Durante a execução da primeira etapa, foi possível verificar que o streaming via HTTP simples, não apresenta uma solução muito viável pois tem dificuldades em lidar com a escalabilidade. Com a execução do cenário prático, foi possível compreender o mecanismo DASH, que tem extrema importância em cenários do dia-a-dia pelo seu ajuste dinâmico de qualidade de vídeo conforme a qualidade da ligação, nomeadamente a largura de banda. Por fim permitiu também perceber as diferenças entre o streaming unicast e multicast, bem como as características de ambas em relação a escalabilidade e gestão de tráfego na rede.