

Engenharia de Serviços em Rede

Trabalho Prático N°2

Streaming de áudio e vídeo a pedido e em tempo real

Ana Luísa Carneiro, Ana Rita Peixoto, and Luís Miguel Pinto

University of Minho, Department of Informatics, 4710-057 Braga, Portugal
e-mail: {pg46983,pg46988,pg47428}@alunos.uminho.pt

Questão 1. Capture três pequenas amostras de tráfego no link de saída do servidor, respetivamente com 1 cliente (VLC), com 2 clientes (VLC e Firefox) e com 3 clientes (VLC, Firefox e ffmpeg). Identifique a taxa em bps necessária (usando o `ffmpeg -i video1.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).

Relativamente à taxa em bps necessária, podemos observar qual seria a taxa esperada através do comando `ffmpeg -i video1.mp4`, tal como apresentado na figura 1. A partir da execução deste comando, concluímos que o valor teórico seria de 11 kbps. Contudo, ao analisar pelo *wireshark* (figura 2), podemos observar que a taxa real necessária para a transmissão do vídeo foi na verdade 17 kbps, para a captura com apenas 1 cliente (VLC). Este valor pode ser justificado devido à ocorrência de perdas na transmissão (*malformed packet*). Podemos ainda observar a taxa relativa à captura com 2 clientes (VLC e Firefox) e com 3 clientes (VLC, Firefox e ffmpeg), nas figuras 3 e 4, respetivamente.

```
core@xubuncore:~$ ffmpeg -i video1.mp4
ffmpeg version 4.2.4-ubuntu1 Copyright (c) 2000-2020 the FFmpeg developers
  built with gcc 9 (Ubuntu 9.3.0-10ubuntu2)
  configuration: --prefix=/usr --extra-version=ubuntu0.1 --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-libaom --enable-libass --enable-libbluray --enable-libbrotli --enable-libcaca --enable-libcdio --enable-libcodec2 --enable-libfontconfig --enable-libfreetype --enable-libfribidi --enable-libgsm --enable-libgss --enable-libjack --enable-libmp3lame --enable-libmysofa --enable-libopenjpeg --enable-libopenmpt --enable-libopus --enable-libpulse --enable-librav --enable-librub --enable-libshine --enable-libsnappy --enable-libsoxr --enable-libspeex --enable-libsrt --enable-libtheora --enable-libtwolame --enable-libvidstab --enable-libvorbis --enable-libvpx --enable-libwavpack --enable-libwebp --enable-libx264 --enable-libx265 --enable-libxml2 --enable-libxvid --enable-libzmq --enable-libzvbi --enable-lv2 --enable-omx --enable-opengl --enable-openssl --enable-opencl --enable-opencl --enable-sdl2 --enable-libdc1394 --enable-libdrm --enable-libiec1883 --enable-nvenc --enable-chromaprint --enable-frei0r --enable-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
  libbscale 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 'video1.mp4':
Metadata:
  major_brand      : isom
  minor_version    : 512
  compatible_brands: isoiso2avc1mp41
  encoder          : Lavf58.29.100
Duration: 00:00:11.15, start: 0.000000, bitrate: 13 kb/s
Stream #0:0(und): Video: h264 (High) (avc1 / 0x31637661), yuv420p, 180x120, 11 kb/s, 20 fps, 20 tbr, 10240 tbn, 40 tbc (default)
Metadata:
  handler_name     : VideoHandler
```

Fig. 1: Taxa bps esperada para transmissão

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	Ei
Frame	100.0	202	100.0	132640	79 k	0	0
Ethernet	100.0	202	2.1	2828	1694	0	0
Internet Protocol Version 6	0.5	1	0.0	40	23	0	0
Internet Protocol Version 4	99.5	201	3.0	4020	2408	0	0
Transmission Control Protocol	96.0	194	94.5	125408	75 k	172	9
Hypertext Transfer Protocol	10.9	22	21.7	28797	17 k	20	2
Malformed Packet	1.0	2	0.0	0	0	2	0
Open Shortest Path First	3.5	7	0.2	308	184	7	3

Fig. 2: Taxa bps real na transmissão com 1 cliente

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes
Frame	100.0	372	100.0	242026	189 k	0	0
Ethernet	100.0	372	2.2	5208	4078	0	0
Internet Protocol Version 6	0.3	1	0.0	40	31	0	0
Internet Protocol Version 4	99.2	369	3.0	7380	5778	0	0
Transmission Control Protocol	97.8	364	94.7	229086	179 k	342	2001
Hypertext Transfer Protocol	5.9	22	12.1	29198	22 k	20	2837
Malformed Packet	0.5	2	0.0	0	0	2	0
Open Shortest Path First	1.3	5	0.1	220	172	5	220
Address Resolution Protocol	0.5	2	0.0	56	43	2	56

Fig. 3: Taxa bps real na transmissão com 2 clientes

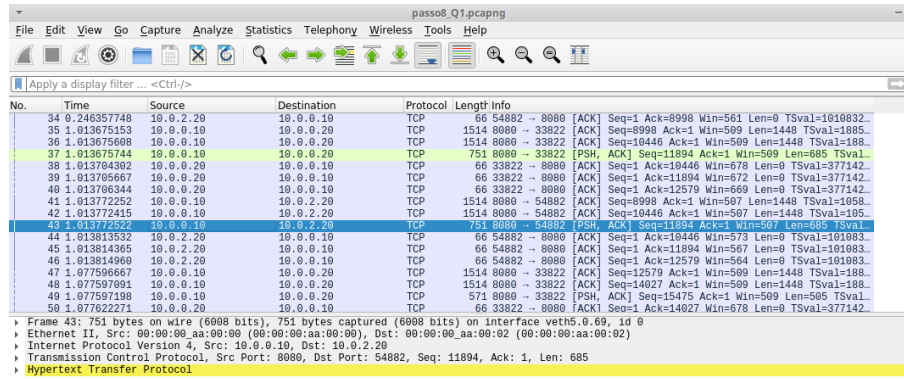
Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits
Frame	100.0	443	100.0	296494	395 k	0	0	0
Ethernet	100.0	443	2.1	6202	8266	0	0	0
Internet Protocol Version 6	0.2	1	0.0	40	53	0	0	0
Internet Protocol Version 4	99.8	442	3.0	8840	11 k	0	0	0
Transmission Control Protocol	98.9	438	94.8	281220	374 k	411	243561	324 k
Hypertext Transfer Protocol	6.1	27	12.9	35220	50 k	27	35220	50 k
Open Shortest Path First	0.9	4	0.1	176	234	4	176	234

Fig. 4: Taxa bps real na transmissão com 3 clientes

No que toca ao encapsulamento dos pacotes transmitidos, podemos observar que este processo ocorre em todos os diferentes níveis da pilha protocolar: camada de transporte (TCP), camada de rede (IPv4), camada de aplicação (HTTP) e na camada de ligação de dados que abrange o protocolo *Ethernet*. Podemos tirar estas conclusões a partir da análise de tramas TCP capturadas pelo *wireshark*, tal como apresentado na figura 5. A informação presente nesta figura é relativa à captura com apenas 1 cliente (VLC). Para as capturas com 2 (VLC e Firefox) e 3 (VLC, Firefox e ffmpeg) o encapsulamento que ocorreu foi o mesmo, tal como é possível observar nas figuras 6 e 7, respetivamente.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [ACK] Seq=1 Ack=1 Win=509 Len=1448 TSval=1884574...
2	0.000000	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [ACK] Seq=1449 Ack=1 Win=509 Len=1448 TSval=1884...
3	0.000000	10.0.0.10	10.0.0.20	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=1449 Win=678 Len=0 TSval=3770626...
4	0.000031	10.0.0.20	10.0.0.10	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=2897 Win=672 Len=0 TSval=3770626...
5	0.000032	10.0.0.20	10.0.0.10	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=3582 Win=669 Len=0 TSval=3770626...
6	0.000033	10.0.0.20	10.0.0.10	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=3582 Win=669 Len=0 TSval=3770626...
7	0.049541	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [ACK] Seq=3582 Ack=1 Win=509 Len=1448 TSval=1884...
8	0.049542	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [ACK] Seq=5030 Ack=1 Win=509 Len=1448 TSval=1884...
9	0.049543	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [ACK] Seq=6478 Ack=1 Win=509 Len=1448 TSval=1884...
10	0.049544	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [ACK] Seq=7926 Ack=1 Win=509 Len=1448 TSval=1884...
11	0.049545	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [PSH, ACK] Seq=9374 Ack=1 Win=509 Len=1448 TSval=...
12	0.049577	10.0.0.10	10.0.0.20	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=5030 Win=678 Len=0 TSval=3770626...
13	0.049578	10.0.0.10	10.0.0.20	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=6478 Win=672 Len=0 TSval=3770626...
14	0.049579	10.0.0.10	10.0.0.20	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=7926 Win=666 Len=0 TSval=3770626...
15	0.049580	10.0.0.10	10.0.0.20	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=9374 Win=661 Len=0 TSval=3770626...
16	0.049581	10.0.0.10	10.0.0.20	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=18822 Win=655 Len=0 TSval=3770626...
17	0.049582	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [ACK] Seq=18822 Ack=1 Win=509 Len=1448 TSval=188...
18	0.049583	10.0.0.10	10.0.0.20	TCP	1072	8080 → 33822 [PSH, ACK] Seq=12270 Ack=1 Win=509 Len=1896 TSva...
19	0.049584	10.0.0.10	10.0.0.20	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=12270 Win=649 Len=0 TSval=377062...
20	0.049585	10.0.0.10	10.0.0.20	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=13276 Win=645 Len=0 TSval=377062...
21	1.346821	10.0.0.1	224.0.0.5	OSPF	78	Hello Packet
22	2.198552	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [ACK] Seq=13276 Ack=1 Win=509 Len=1448 TSval=188...
23	2.198553	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [ACK] Seq=14724 Ack=1 Win=509 Len=1448 TSval=188...
24	2.198554	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [ACK] Seq=16172 Ack=1 Win=509 Len=1448 TSval=188...
25	2.198555	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [ACK] Seq=17620 Ack=1 Win=509 Len=1448 TSval=188...

Fig. 5: Encapsulamento da trama TCP com 1 cliente



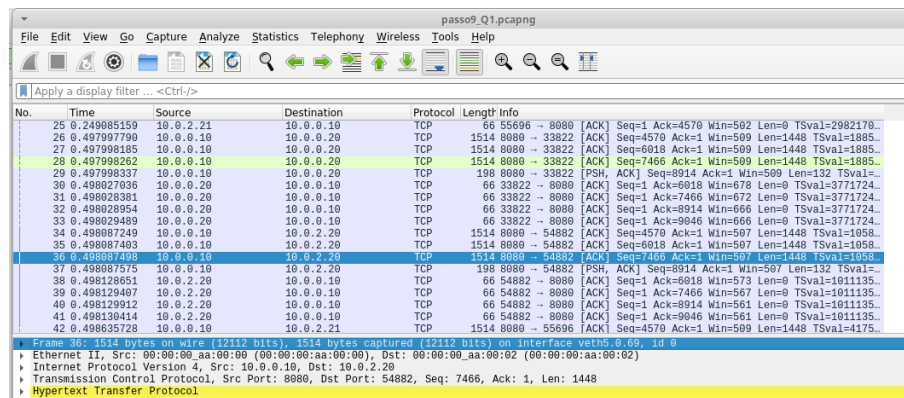
The image shows a Wireshark packet capture of a TCP connection. The interface includes a menu bar (File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, Help), a toolbar, and a display filter bar. The packet list table shows the following data:

No.	Time	Source	Destination	Protocol	Length	Info
34	0.246357748	10.0.2.20	10.0.0.10	TCP	66	54882 → 8080 [ACK] Seq=1 Ack=8998 Win=561 Len=0 TSval=1010832...
35	1.013675153	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [ACK] Seq=8998 Ack=1 Win=509 Len=1448 TSval=1885...
36	1.013675608	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [ACK] Seq=10446 Ack=1 Win=509 Len=1448 TSval=188...
37	1.013675744	10.0.0.10	10.0.0.20	TCP	751	8080 → 33822 [PSH, ACK] Seq=11894 Ack=1 Win=509 Len=685 TSval=...
38	1.013704302	10.0.0.20	10.0.0.10	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=10446 Win=678 Len=0 TSval=377142...
39	1.013705667	10.0.0.20	10.0.0.10	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=11894 Win=672 Len=0 TSval=377142...
40	1.013706344	10.0.0.20	10.0.0.10	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=12579 Win=669 Len=0 TSval=377142...
41	1.013772252	10.0.0.10	10.0.2.20	TCP	1514	8080 → 54882 [ACK] Seq=8998 Ack=1 Win=507 Len=1448 TSval=1058...
42	1.013772415	10.0.0.10	10.0.2.20	TCP	1514	8080 → 54882 [ACK] Seq=10446 Ack=1 Win=507 Len=1448 TSval=105...
43	1.013772522	10.0.0.10	10.0.2.20	TCP	751	8080 → 54882 [PSH, ACK] Seq=11894 Ack=1 Win=507 Len=685 TSval=...
44	1.013813532	10.0.2.20	10.0.0.10	TCP	66	54882 → 8080 [ACK] Seq=1 Ack=10446 Win=573 Len=0 TSval=101083...
45	1.013814365	10.0.2.20	10.0.0.10	TCP	66	54882 → 8080 [ACK] Seq=1 Ack=11894 Win=567 Len=0 TSval=101083...
46	1.013814960	10.0.2.20	10.0.0.10	TCP	66	54882 → 8080 [ACK] Seq=1 Ack=12579 Win=564 Len=0 TSval=101083...
47	1.077596667	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [ACK] Seq=12579 Ack=1 Win=509 Len=1448 TSval=188...
48	1.077597091	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [ACK] Seq=14027 Ack=1 Win=509 Len=1448 TSval=188...
49	1.077597198	10.0.0.10	10.0.0.20	TCP	571	8080 → 33822 [PSH, ACK] Seq=15475 Ack=1 Win=509 Len=505 TSval=...
50	1.077622271	10.0.0.20	10.0.0.10	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=14027 Win=678 Len=0 TSval=377142...

Below the packet list, the packet details pane shows the selected packet (No. 43) with the following information:

- Frame 43: 751 bytes on wire (6008 bits): 751 bytes captured (6008 bits) on interface veth5.0.69, id 0
- Ethernet II, Src: 00:00:00:aa:00:00 (00:00:00:aa:00:00), Dst: 00:00:00:aa:00:02 (00:00:00:aa:00:02)
- Internet Protocol Version 4, Src: 10.0.0.10, Dst: 10.0.2.20
- Transmission Control Protocol, Src Port: 8080, Dst Port: 54882, Seq: 11894, Ack: 1, Len: 685
- Hypertext Transfer Protocol

Fig. 6: Encapsulamento da trama TCP com 2 clientes



The image shows a Wireshark packet capture of a TCP connection. The interface includes a menu bar (File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, Help), a toolbar, and a display filter bar. The packet list table shows the following data:

No.	Time	Source	Destination	Protocol	Length	Info
25	0.249085159	10.0.2.21	10.0.0.10	TCP	66	55696 → 8080 [ACK] Seq=1 Ack=4570 Win=502 Len=0 TSval=2982170...
26	0.497997790	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [ACK] Seq=4570 Ack=1 Win=509 Len=1448 TSval=1885...
27	0.497998195	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [ACK] Seq=6018 Ack=1 Win=509 Len=1448 TSval=1885...
28	0.497998262	10.0.0.10	10.0.0.20	TCP	1514	8080 → 33822 [ACK] Seq=7466 Ack=1 Win=509 Len=1448 TSval=1885...
29	0.497998337	10.0.0.10	10.0.0.20	TCP	198	8080 → 33822 [PSH, ACK] Seq=8914 Ack=1 Win=509 Len=132 TSval=...
30	0.498027036	10.0.0.20	10.0.0.10	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=6018 Win=678 Len=0 TSval=3771724...
31	0.498028381	10.0.0.20	10.0.0.10	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=7466 Win=672 Len=0 TSval=3771724...
32	0.498028954	10.0.0.20	10.0.0.10	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=8914 Win=666 Len=0 TSval=3771724...
33	0.498029489	10.0.0.20	10.0.0.10	TCP	66	33822 → 8080 [ACK] Seq=1 Ack=9046 Win=666 Len=0 TSval=3771724...
34	0.498087249	10.0.0.10	10.0.2.20	TCP	1514	8080 → 54882 [ACK] Seq=4570 Ack=1 Win=507 Len=1448 TSval=1058...
35	0.498087403	10.0.0.10	10.0.2.20	TCP	1514	8080 → 54882 [ACK] Seq=6018 Ack=1 Win=507 Len=1448 TSval=1058...
36	0.498087498	10.0.0.10	10.0.2.20	TCP	1514	8080 → 54882 [ACK] Seq=7466 Ack=1 Win=507 Len=1448 TSval=1058...
37	0.498087575	10.0.0.10	10.0.2.20	TCP	198	8080 → 54882 [PSH, ACK] Seq=8914 Ack=1 Win=507 Len=132 TSval=...
38	0.498128651	10.0.2.20	10.0.0.10	TCP	66	54882 → 8080 [ACK] Seq=1 Ack=6018 Win=573 Len=0 TSval=1011135...
39	0.498129407	10.0.2.20	10.0.0.10	TCP	66	54882 → 8080 [ACK] Seq=1 Ack=7466 Win=567 Len=0 TSval=1011135...
40	0.498129912	10.0.2.20	10.0.0.10	TCP	66	54882 → 8080 [ACK] Seq=1 Ack=8914 Win=561 Len=0 TSval=1011135...
41	0.498130414	10.0.2.20	10.0.0.10	TCP	66	54882 → 8080 [ACK] Seq=1 Ack=9046 Win=561 Len=0 TSval=1011135...
42	0.498635728	10.0.0.10	10.0.0.21	TCP	1514	8080 → 55696 [ACK] Seq=4570 Ack=1 Win=509 Len=1448 TSval=4175...

Below the packet list, the packet details pane shows the selected packet (No. 36) with the following information:

- Frame 36: 1514 bytes on wire (12112 bits): 1514 bytes captured (12112 bits) on interface veth5.0.69, id 0
- Ethernet II, Src: 00:00:00:aa:00:00 (00:00:00:aa:00:00), Dst: 00:00:00:aa:00:02 (00:00:00:aa:00:02)
- Internet Protocol Version 4, Src: 10.0.0.10, Dst: 10.0.2.20
- Transmission Control Protocol, Src Port: 8080, Dst Port: 54882, Seq: 7466, Ack: 1, Len: 1448
- Hypertext Transfer Protocol

Fig. 7: Encapsulamento da trama TCP com 3 clientes

streams, tal como apresentado na figura 10. Podemos concluir que cada *stream* corresponde a cada serviço utilizado: VLC, Firefox e ffmpeg.

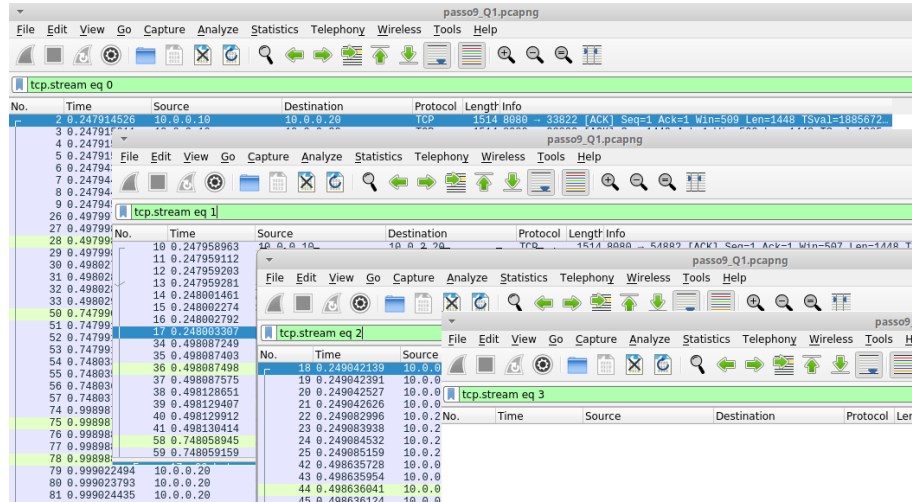


Fig. 10: Fluxos gerados com 3 clientes

Por último, podemos concluir que esta solução não escala muito bem. Através da observação das capturas efetuadas com diferentes clientes concluímos que o servidor envia a resposta de pedidos individualmente para cada cliente. Deste modo, caso o número de clientes aumente consideravelmente, o servidor terá que enviar as tramas para cada cliente, mesmo que o pedido de cada cliente seja o mesmo.

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 assim como a informação sobre as diversas resoluções do vídeo está descrito no ficheiro "video_manifest.mpd".

Para que o cliente consiga assistir ao vídeo de menor resolução, 180*120, é necessário uma largura de banda de 433870 bps, tal como podemos ver na figura 11. Para assistir ao vídeo de resolução intermédia, 360*240, é necessário um *link* com 1673630 bps de largura de banda, demonstrado na figura 12. Finalmente, para assistir ao vídeo de resolução mais alta, 540*360, é necessário uma largura de banda de 8297710 bps, tal como está na figura 13.

```

<Representation id="1" mimeType="video/mp4" codecs="avc3.64000c" width="180" height="120" frameRate="30" sar="1:1" startWithSAP="0" bandwidth="167363"/>
<BaseURL>video2_180_120_200k_dash.mp4</BaseURL>
<SegmentList timescale="15360" duration="7680">
  <SegmentURL mediaRange="928-5489" indexRange="928-971"/>
  <SegmentURL mediaRange="5490-11811" indexRange="5490-5533"/>
  <SegmentURL mediaRange="11812-15912" indexRange="11812-11855"/>
  <SegmentURL mediaRange="15913-27256" indexRange="15913-15956"/>
  <SegmentURL mediaRange="27257-30577" indexRange="27257-27300"/>
  <SegmentURL mediaRange="30578-33824" indexRange="30578-30621"/>
  <SegmentURL mediaRange="33825-55239" indexRange="33825-33868"/>
  <SegmentURL mediaRange="55240-64067" indexRange="55240-55282"/>
  <SegmentURL mediaRange="64068-76294" indexRange="64068-64111"/>
  <SegmentURL mediaRange="76295-88172" indexRange="76295-76338"/>
  <SegmentURL mediaRange="88173-111015" indexRange="88173-88216"/>
  <SegmentURL mediaRange="111016-117622" indexRange="111016-111059"/>
  <SegmentURL mediaRange="117623-126691" indexRange="117623-117666"/>
  <SegmentURL mediaRange="126692-144078" indexRange="126692-126645"/>
  <SegmentURL mediaRange="144079-151624" indexRange="144079-144122"/>
  <SegmentURL mediaRange="151625-158901" indexRange="151625-151668"/>
  <SegmentURL mediaRange="158902-178245" indexRange="158902-158945"/>
  <SegmentURL mediaRange="178246-185298" indexRange="178247-178290"/>
  <SegmentURL mediaRange="185299-192426" indexRange="185299-185342"/>
  <SegmentURL mediaRange="192427-199646" indexRange="192427-192470"/>
  <SegmentURL mediaRange="199647-219411" indexRange="199647-199690"/>
  <SegmentURL mediaRange="219412-229775" indexRange="219412-219455"/>
  <SegmentURL mediaRange="229776-238181" indexRange="229776-229819"/>
  <SegmentURL mediaRange="238182-257618" indexRange="238182-238225"/>
  <SegmentURL mediaRange="257619-260807" indexRange="257619-257662"/>
</SegmentList>
</Representation>

```

Fig. 11: Informação sobre a resolução 180*120 do vídeo

```

<Representation id="2" mimeType="video/mp4" codecs="avc3.640014" width="360" height="240" frameRate="30" sar="1:1" startWithSAP="0" bandwidth="433872"/>
<BaseURL>video2_360_240_500k_dash.mp4</BaseURL>
<SegmentList timescale="15360" duration="7680">
  <SegmentURL mediaRange="928-11542" indexRange="928-971"/>
  <SegmentURL mediaRange="11543-28707" indexRange="11543-11586"/>
  <SegmentURL mediaRange="28708-41810" indexRange="28708-28751"/>
  <SegmentURL mediaRange="41811-73257" indexRange="41811-41854"/>
  <SegmentURL mediaRange="73258-83135" indexRange="73258-73301"/>
  <SegmentURL mediaRange="83136-91258" indexRange="83136-83179"/>
  <SegmentURL mediaRange="91259-147503" indexRange="91259-91302"/>
  <SegmentURL mediaRange="147504-176284" indexRange="147504-147547"/>
  <SegmentURL mediaRange="176285-211259" indexRange="176285-176328"/>
  <SegmentURL mediaRange="211260-237201" indexRange="211260-211303"/>
  <SegmentURL mediaRange="237202-298630" indexRange="237202-237245"/>
  <SegmentURL mediaRange="298631-316289" indexRange="298631-298674"/>
  <SegmentURL mediaRange="316290-332321" indexRange="316290-316333"/>
  <SegmentURL mediaRange="332322-382048" indexRange="332322-332365"/>
  <SegmentURL mediaRange="382049-399526" indexRange="382049-382092"/>
  <SegmentURL mediaRange="399527-416036" indexRange="399527-399570"/>
  <SegmentURL mediaRange="416037-468904" indexRange="416037-416080"/>
  <SegmentURL mediaRange="468905-487790" indexRange="468905-468948"/>
  <SegmentURL mediaRange="487791-504292" indexRange="487791-487834"/>
  <SegmentURL mediaRange="504293-521457" indexRange="504293-504336"/>
  <SegmentURL mediaRange="521458-574423" indexRange="521458-521501"/>
  <SegmentURL mediaRange="574424-598376" indexRange="574424-574467"/>
  <SegmentURL mediaRange="598377-618721" indexRange="598377-598420"/>
  <SegmentURL mediaRange="618722-669630" indexRange="618722-618765"/>
  <SegmentURL mediaRange="669631-676117" indexRange="669631-669674"/>
</SegmentList>
</Representation>

```

Fig. 12: Informação sobre a resolução 360*240 do vídeo

```

<Representation id="3" mimeType="video/mp4" codecs="avc3.64001e" width="540" height="360" frameRate="30" sar="1:1" startWithSAP="0" bandwidth="829771"/>
<BaseURL>video2_540_360_1000k_dash.mp4</BaseURL>
<SegmentList timescale="15360" duration="7680">
  <SegmentURL mediaRange="928-20616" indexRange="928-971"/>
  <SegmentURL mediaRange="20617-52980" indexRange="20617-20660"/>
  <SegmentURL mediaRange="52981-78019" indexRange="52981-53024"/>
  <SegmentURL mediaRange="78020-137919" indexRange="78020-78063"/>
  <SegmentURL mediaRange="137920-158936" indexRange="137920-137963"/>
  <SegmentURL mediaRange="158937-171906" indexRange="158937-158980"/>
  <SegmentURL mediaRange="171907-270342" indexRange="171907-171950"/>
  <SegmentURL mediaRange="270343-351299" indexRange="270343-270386"/>
  <SegmentURL mediaRange="351300-389535" indexRange="351300-331343"/>
  <SegmentURL mediaRange="389536-448099" indexRange="389536-389579"/>
  <SegmentURL mediaRange="448100-565128" indexRange="448100-448143"/>
  <SegmentURL mediaRange="565129-597240" indexRange="565129-565172"/>
  <SegmentURL mediaRange="597241-634134" indexRange="597241-597284"/>
  <SegmentURL mediaRange="634135-725773" indexRange="634135-634178"/>
  <SegmentURL mediaRange="725774-760827" indexRange="725774-725817"/>
  <SegmentURL mediaRange="760828-794081" indexRange="760828-760871"/>
  <SegmentURL mediaRange="794082-886878" indexRange="794082-794125"/>
  <SegmentURL mediaRange="886879-927442" indexRange="886879-886922"/>
  <SegmentURL mediaRange="927443-963599" indexRange="927443-927486"/>
  <SegmentURL mediaRange="963600-999856" indexRange="963600-963643"/>
  <SegmentURL mediaRange="999857-1097616" indexRange="999857-999900"/>
  <SegmentURL mediaRange="1097617-1145426" indexRange="1097617-1097660"/>
  <SegmentURL mediaRange="1145427-1184040" indexRange="1145427-1145470"/>
  <SegmentURL mediaRange="1184041-1278383" indexRange="1184041-1184984"/>
  <SegmentURL mediaRange="1278384-1293059" indexRange="1278384-1278427"/>
</SegmentList>
</Representation>

```

Fig. 13: Informação sobre a resolução 520*360 do vídeo

Os protocolos utilizados para a transmissão do vídeo encontram-se na figura 14. Neste figura podemos ver o protocolo *Ethernet* a ser utilizado na camada de ligação de dados, o protocolo IP na camada de rede, o protocolo TCP para a camada de transporte e o HTTP na camada de aplicação.

Todos estes protocolos estão associados e dão suporte à rede global Internet, uma vez que o vídeo foi transmitido na *WEB* a partir do *firefox*.

No.	Time	Source	Destination	Protocol	Length	Info
14722	18.127544	10.0.0.20	10.0.0.10	TCP	66	34302 → 9999 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=41599490...
14723	18.127782	10.0.0.20	10.0.0.10	HTTP	402	GET /video2_560_240_500k_dash.mp4 HTTP/1.1
14724	18.127790	10.0.0.10	10.0.0.20	TCP	66	9999 → 34302 [ACK] Seq=1 Ack=337 Win=64896 Len=0 TSval=141513...
14725	18.127909	10.0.0.10	10.0.0.20	TCP	1514	9999 → 34302 [ACK] Seq=1 Ack=337 Win=64896 Len=1448 TSval=141...
14726	18.127910	10.0.0.10	10.0.0.20	TCP	1514	9999 → 34302 [ACK] Seq=1449 Ack=337 Win=64896 Len=1448 TSval=...
14727	18.127910	10.0.0.10	10.0.0.20	TCP	1514	9999 → 34302 [ACK] Seq=2897 Ack=337 Win=64896 Len=1448 TSval=...
14728	18.127911	10.0.0.10	10.0.0.20	TCP	1514	9999 → 34302 [ACK] Seq=4345 Ack=337 Win=64896 Len=1448 TSval=...
14729	18.127912	10.0.0.10	10.0.0.20	TCP	1514	9999 → 34302 [ACK] Seq=5793 Ack=337 Win=64896 Len=1448 T...
14730	18.127923	10.0.0.10	10.0.0.20	TCP	1514	9999 → 34302 [ACK] Seq=7241 Ack=337 Win=64896 Len=1448 TSval=...

▶ Frame 14723: 402 bytes on wire (3216 bits), 402 bytes captured (3216 bits)
 ▶ Ethernet II, Src: 00:00:00:aa:00:51 (00:00:00:aa:00:51), Dst: 00:00:00:aa:00:50 (00:00:00:aa:00:50)
 ▶ Internet Protocol Version 4, Src: 10.0.0.20, Dst: 10.0.0.10
 ▶ Transmission Control Protocol, Src Port: 34302, Dst Port: 9999, Seq: 1, Ack: 1, Len: 336
 ▶ Hypertext Transfer Protocol

Fig. 14: Camada protocolar

Questão 3. Ajuste o débito dos *links* da topologia de modo que o cliente no portátil 2 exiba o vídeo de menor resolução e o cliente no portátil 1 exiba o vídeo com mais resolução. Mostre evidências.

Para que portátil 1 exiba o vídeo de maior resolução é necessário restringir o *link* da topologia para que esta permita a transmissão do vídeo de resolução 540*360. Para isso, verificamos a largura de banda necessária para transmitir o vídeo com essa resolução no ficheiro "video_manifest.mpd" (figura 13).

Como a largura de banda necessário à transmissão do vídeo de resolução alta é 829771 bps basta que o *link* do portátil 1 seja restringido à largura de banda de 9000000 bps, isto é, 900 kbps (figura 15).

Na captura do *wireshark* para o portátil 1 (figura 16), podemos ver na linha 27 que há uma transmissão do vídeo de maior resolução (540*360), tal como era pretendido. Contudo, ao longo da transmissão, devido à funcionalidade do DASH, a resolução do vídeo vai diminuindo para a resolução intermédia (linha 1524) e para a resolução mais baixa (linha 3033).

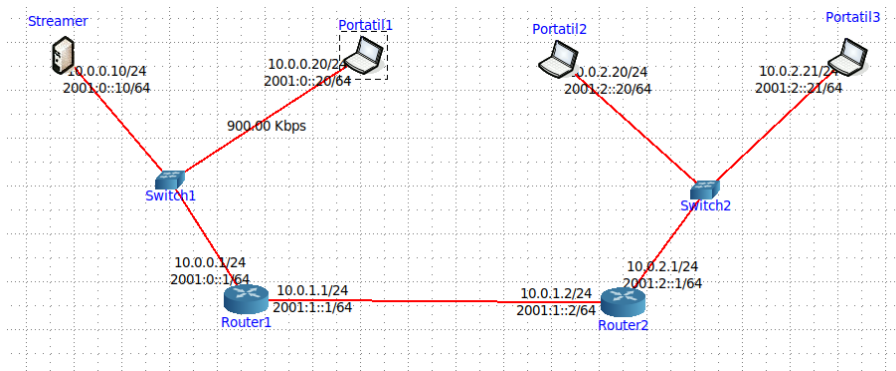


Fig. 15: Rede com restrição para maior resolução

No.	Time	Source	Destination	Protocol	Length	Info
12	4.823208855	10.0.0.20	10.0.0.10	HTTP	379	GET /favicon.ico HTTP/1.1
15	4.838816531	10.0.0.10	10.0.0.20	HTTP	741	HTTP/1.1 404 Not Found (text/html)
27	5.221653837	10.0.0.20	10.0.0.10	HTTP	399	GET /video2_540_360_1080k_dash.mp4 HTTP/1.1
1514	17.249881786	10.0.0.10	10.0.0.20	MP4	268	
1524	18.323768235	10.0.0.20	10.0.0.10	HTTP	400	GET /video2_360_240_500k_dash.mp4 HTTP/1.1
2275	24.810998281	10.0.0.10	10.0.0.20	MP4	173	
2282	24.633708178	10.0.0.20	10.0.0.10	HTTP	400	GET /video2_360_240_500k_dash.mp4 HTTP/1.1
3026	30.923517739	10.0.0.10	10.0.0.20	MP4	173	
3033	30.986481479	10.0.0.20	10.0.0.10	HTTP	400	GET /video2_180_120_200k_dash.mp4 HTTP/1.1

Fig. 16: Captura da transmissão para o Portátil 1

Para que portátil 2 exiba o vídeo de menor resolução é necessário restringir o *link* da topologia para que esta permita a transmissão do vídeo de resolução 180*120. Para isso é necessário verificar a largura de banda dessa resolução no ficheiro "video_manifest.mpd" (figura 11).

Como a largura de banda necessário à transmissão do vídeo de resolução baixa é 1673630 bps basta que o *link* do portátil 2 seja restringido à largura de banda de 2000000 bps, isto é, 200 kbps (figura 17).

Na captura do *wireshark* para o portátil 2 (figura 18), podemos ver na linha 78 que há uma transmissão do vídeo de menor resolução, tal como era pretendido.

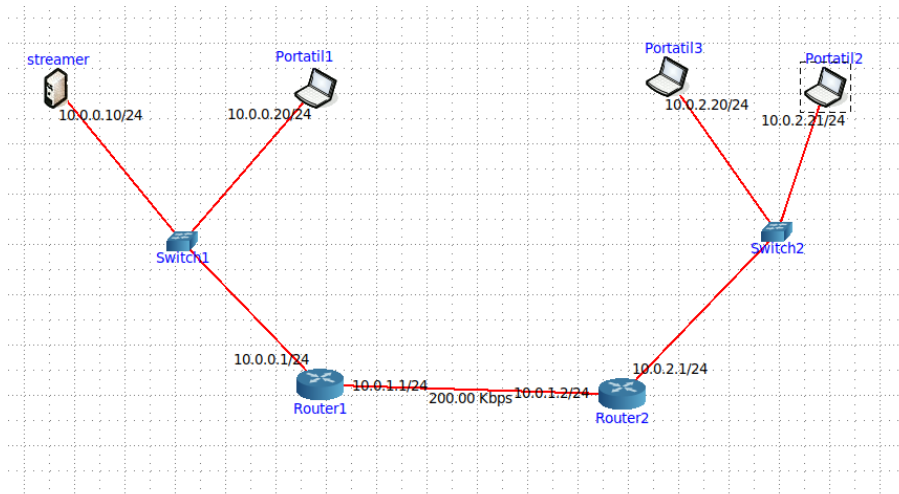


Fig. 17: Rede Core com restrição para menor resolução

No.	Time	Source	Destination	Protocol	Length	Info
8	4.499322	10.0.2.21	10.0.0.10	HTTP	426	GET /video_dash.html HTTP/1.1
10	4.562787	10.0.0.10	10.0.2.21	HTTP	666	HTTP/1.1 200 OK (text/html)
19	5.210322	10.0.2.21	10.0.0.10	HTTP	426	GET /video_manifest.mpd HTTP/1.1
35	5.570257	10.0.0.10	10.0.2.21	HTTP/X.	198	HTTP/1.1 200 OK
43	5.570368	10.0.2.21	10.0.0.10	HTTP	379	GET /favicon.ico HTTP/1.1
48	5.581356	10.0.2.21	10.0.0.10	HTTP	364	GET /video_manifest.mpd HTTP/1.1
50	5.639159	10.0.0.10	10.0.2.21	HTTP	741	HTTP/1.1 404 Not Found (text/html)
64	5.941684	10.0.0.10	10.0.2.21	HTTP/X.	198	HTTP/1.1 200 OK
69	6.051762	10.0.2.21	10.0.0.10	HTTP	369	GET /video_manifest_init.mpd HTTP/1.1
71	6.132516	10.0.0.10	10.0.2.21	MP4	1660	
78	6.190417	10.0.2.21	10.0.0.10	HTTP	397	GET /video2_180_120_200k_dash.mp4 HTTP/1.1
451	17.144409	10.0.0.10	10.0.2.21	MP4	439	
459	18.201089	10.0.2.21	10.0.0.10	HTTP	399	GET /video2_180_120_200k_dash.mp4 HTTP/1.1
829	29.168387	10.0.0.10	10.0.2.21	MP4	439	
835	29.209942	10.0.2.21	10.0.0.10	HTTP	400	GET /video2_180_120_200k_dash.mp4 HTTP/1.1

Fig. 18: Captura da transmissão para o Portátil 2

Questão 4. Descreva o funcionamento do DASH neste caso concreto, referindo o papel do ficheiro MPD criado.

O DASH (*Dynamic Adaptive Streaming over HTTP*) é uma técnica de *streaming* por HTTP que permite a transmissão dinâmica e adaptativa às condições da rede como por exemplo a largura de banda do *link*. Assim sendo, o DASH vai a verificar a largura de banda do *link* e avaliar qual a melhor resolução do vídeo a ser transmitido nesse *link*, a partir do ficheiro "video\manifest.mpd". É neste ficheiro onde está armazenado as informações sobre cada resolução disponível do vídeo tal como a largura de banda necessária do *link* para transmitir o vídeo nessa resolução específica. Desta forma, durante o *stream* do vídeo, o DASH vai avaliando a qualidade de transmissão e conectividade e alterando a resolução do vídeo conforme essas características.

Um exemplo do funcionamento da DASH ocorreu durante a *stream* do vídeo no *firefox* no portátil 1 e 2. Durante essa transmissão, foi realizada uma captura com o apoio do *Wireshark* de forma a ver os dados recebidos por ambos os clientes, tal como está representado nas figuras 19 e 20.

Em ambos os casos podemos ver que o *streaming* foi inicializado com um vídeo de resolução alta (540*360) tal como está linhas 60 e 52 nas figuras 19 e 20, respetivamente. Contudo, ao longo da *stream* podemos ver o vídeo a mudar a sua resolução para uma mais baixa (360*240) tal como está nas linhas 3036 e 4162 nas figuras 19 e 20, respetivamente, sendo esta a resolução que acaba por ser transmitida até ao fim da *stream*.

Isto aconteceu pois o DASH verificou que a conectividade e/ou a transmissão na rede não foram suficientes para manter a transmissão, na íntegra, da resolução mais alta (540*360) e por isso baixou a resolução do vídeo para que o utilizador conseguisse ver o vídeo completo sem que note perdas de frame ou pausas devido à falta de conexão.

No.	Time	Source	Destination	Protocol	Length Info
14	15.515434	10.0.0.20	10.0.0.10	HTTP	425 GET /video_dash.html HTTP/1.1
16	15.515562	10.0.0.10	10.0.0.20	HTTP	666 HTTP/1.1 200 OK (text/html)
26	16.166498	10.0.0.20	10.0.0.10	HTTP	379 GET /favicon.ico HTTP/1.1
28	16.166594	10.0.0.10	10.0.0.20	HTTP	741 HTTP/1.1 404 Not Found (text/html)
35	16.181031	10.0.0.20	10.0.0.10	HTTP	426 GET /video_manifest.mpd HTTP/1.1
42	16.181177	10.0.0.10	10.0.0.20	HTTP/XL	198 HTTP/1.1 200 OK
51	16.545664	10.0.0.20	10.0.0.10	HTTP	369 GET /video_manifest_init.mp4 HTTP/1.1
53	16.545792	10.0.0.10	10.0.0.20	MP4	1060
60	16.580601	10.0.0.20	10.0.0.10	HTTP	399 GET /video2_540_360_1000k_dash.mp4 HTTP/1.1
1022	16.589598	10.0.0.10	10.0.0.20	MP4	268
1029	16.647513	10.0.0.20	10.0.0.10	HTTP	401 GET /video2_540_360_1000k_dash.mp4 HTTP/1.1
2046	16.655247	10.0.0.10	10.0.0.20	MP4	268
2053	16.722394	10.0.0.20	10.0.0.10	HTTP	401 GET /video2_540_360_1000k_dash.mp4 HTTP/1.1
3029	16.746644	10.0.0.10	10.0.0.20	MP4	268
3036	17.343793	10.0.0.20	10.0.0.10	HTTP	400 GET /video2_360_240_500k_dash.mp4 HTTP/1.1
3611	17.344102	10.0.0.10	10.0.0.20	MP4	173
3623	17.390914	10.0.0.20	10.0.0.10	HTTP	400 GET /video2_360_240_500k_dash.mp4 HTTP/1.1
4129	17.395054	10.0.0.10	10.0.0.20	MP4	173
4136	17.413867	10.0.0.20	10.0.0.10	HTTP	400 GET /video2_360_240_500k_dash.mp4 HTTP/1.1
4640	17.417759	10.0.0.10	10.0.0.20	MP4	173
4655	17.432005	10.0.0.20	10.0.0.10	HTTP	401 GET /video2_360_240_500k_dash.mp4 HTTP/1.1
5174	17.436885	10.0.0.10	10.0.0.20	MP4	173
5181	17.465650	10.0.0.20	10.0.0.10	HTTP	402 GET /video2_360_240_500k_dash.mp4 HTTP/1.1
5694	17.469303	10.0.0.10	10.0.0.20	MP4	173
5701	17.496292	10.0.0.20	10.0.0.10	HTTP	402 GET /video2_360_240_500k_dash.mp4 HTTP/1.1
6214	17.500192	10.0.0.10	10.0.0.20	MP4	173
6221	17.522432	10.0.0.20	10.0.0.10	HTTP	402 GET /video2_360_240_500k_dash.mp4 HTTP/1.1
6719	17.525899	10.0.0.10	10.0.0.20	MP4	173
6726	17.554839	10.0.0.20	10.0.0.10	HTTP	402 GET /video2_360_240_500k_dash.mp4 HTTP/1.1
7247	17.558725	10.0.0.10	10.0.0.20	MP4	173
7254	17.577327	10.0.0.20	10.0.0.10	HTTP	402 GET /video2_360_240_500k_dash.mp4 HTTP/1.1
7777	17.581458	10.0.0.10	10.0.0.20	MP4	173
7784	17.594344	10.0.0.20	10.0.0.10	HTTP	402 GET /video2_360_240_500k_dash.mp4 HTTP/1.1
8359	17.598168	10.0.0.10	10.0.0.20	MP4	173
8368	17.616200	10.0.0.20	10.0.0.10	HTTP	402 GET /video2_360_240_500k_dash.mp4 HTTP/1.1
8940	17.620688	10.0.0.10	10.0.0.20	MP4	173
8952	17.646471	10.0.0.20	10.0.0.10	HTTP	402 GET /video2_360_240_500k_dash.mp4 HTTP/1.1
9527	17.650265	10.0.0.10	10.0.0.20	MP4	173
9536	17.667903	10.0.0.20	10.0.0.10	HTTP	402 GET /video2_360_240_500k_dash.mp4 HTTP/1.1
10128	17.670707	10.0.0.10	10.0.0.20	MP4	173
10135	17.692977	10.0.0.20	10.0.0.10	HTTP	402 GET /video2_360_240_500k_dash.mp4 HTTP/1.1
10640	17.695851	10.0.0.10	10.0.0.20	MP4	173
10655	17.723638	10.0.0.20	10.0.0.10	HTTP	402 GET /video2_360_240_500k_dash.mp4 HTTP/1.1
11306	17.728195	10.0.0.10	10.0.0.20	MP4	173
11330	17.757368	10.0.0.20	10.0.0.10	HTTP	402 GET /video2_360_240_500k_dash.mp4 HTTP/1.1
11866	17.761343	10.0.0.10	10.0.0.20	MP4	173
11873	17.777076	10.0.0.20	10.0.0.10	HTTP	402 GET /video2_360_240_500k_dash.mp4 HTTP/1.1

Fig. 19: Captura para o Portátil 1

No.	Time	Source	Destination	Protocol	Length	Info
10	6.592882	10.0.0.21	10.0.0.10	HTTP	423	GET /video_dash.html HTTP/1.1
12	6.593822	10.0.0.10	10.0.2.21	HTTP	666	HTTP/1.1 200 OK (text/html)
22	6.208920	10.0.2.21	10.0.0.10	HTTP	379	GET /favicon.ico HTTP/1.1
24	6.209035	10.0.0.10	10.0.2.21	HTTP	741	HTTP/1.1 404 Not Found (text/html)
31	6.232258	10.0.2.21	10.0.0.10	HTTP	426	GET /video_manifest.mpd HTTP/1.1
38	6.232427	10.0.0.10	10.0.2.21	HTTP/X..	198	HTTP/1.1 200 OK
47	6.595892	10.0.2.21	10.0.0.10	HTTP	369	GET /video_manifest_init.mpd HTTP/1.1
49	6.595241	10.0.0.10	10.0.2.21	MP4	1068	
56	6.675870	10.0.2.21	10.0.0.10	HTTP	399	GET /video2_540_360_1000k_dash.mpd HTTP/1.1
115	6.686846	10.0.0.10	10.0.2.21	MP4	268	
1122	6.749855	10.0.2.21	10.0.0.10	HTTP	401	GET /video2_540_360_1000k_dash.mpd HTTP/1.1
2110	6.760078	10.0.0.10	10.0.2.21	MP4	268	
2117	6.844661	10.0.2.21	10.0.0.10	HTTP	401	GET /video2_540_360_1000k_dash.mpd HTTP/1.1
3179	6.867608	10.0.0.10	10.0.2.21	MP4	268	
3177	7.448190	10.0.2.21	10.0.0.10	HTTP	402	GET /video2_540_360_1000k_dash.mpd HTTP/1.1
4147	7.461031	10.0.0.10	10.0.2.21	MP4	268	
4154	7.533588	10.0.2.21	10.0.0.10	HTTP	419	GET /video_manifest_init.mpd HTTP/1.1
4156	7.533731	10.0.0.10	10.0.2.21	HTTP	257	HTTP/1.1 304 Not Modified
4162	7.538982	10.0.2.21	10.0.0.10	HTTP	408	GET /video2_360_240_500k_dash.mpd HTTP/1.1
4690	7.543820	10.0.0.10	10.0.2.21	MP4	173	
4697	7.570351	10.0.2.21	10.0.0.10	HTTP	400	GET /video2_360_240_500k_dash.mpd HTTP/1.1
5236	7.575131	10.0.0.10	10.0.2.21	MP4	173	
5243	7.599523	10.0.2.21	10.0.0.10	HTTP	401	GET /video2_360_240_500k_dash.mpd HTTP/1.1
5779	7.595362	10.0.0.10	10.0.2.21	MP4	173	
5786	7.612774	10.0.2.21	10.0.0.10	HTTP	402	GET /video2_360_240_500k_dash.mpd HTTP/1.1
6322	7.621209	10.0.0.10	10.0.2.21	MP4	173	
6329	7.652131	10.0.2.21	10.0.0.10	HTTP	402	GET /video2_360_240_500k_dash.mpd HTTP/1.1
6904	7.657022	10.0.0.10	10.0.2.21	MP4	173	
6911	7.687289	10.0.2.21	10.0.0.10	HTTP	402	GET /video2_360_240_500k_dash.mpd HTTP/1.1
7481	7.691933	10.0.0.10	10.0.2.21	MP4	173	
7496	7.706212	10.0.2.21	10.0.0.10	HTTP	402	GET /video2_360_240_500k_dash.mpd HTTP/1.1
8031	7.724908	10.0.0.10	10.0.2.21	MP4	173	
8039	7.766065	10.0.2.21	10.0.0.10	HTTP	402	GET /video2_360_240_500k_dash.mpd HTTP/1.1
8609	7.771852	10.0.0.10	10.0.2.21	MP4	173	
8624	7.816631	10.0.2.21	10.0.0.10	HTTP	402	GET /video2_360_240_500k_dash.mpd HTTP/1.1
9187	7.821798	10.0.0.10	10.0.2.21	MP4	173	
9224	7.835324	10.0.2.21	10.0.0.10	HTTP	402	GET /video2_360_240_500k_dash.mpd HTTP/1.1
9732	7.839626	10.0.0.10	10.0.2.21	MP4	173	
9739	7.857100	10.0.2.21	10.0.0.10	HTTP	402	GET /video2_360_240_500k_dash.mpd HTTP/1.1
10248	7.862718	10.0.0.10	10.0.2.21	MP4	173	
10255	7.886947	10.0.2.21	10.0.0.10	HTTP	402	GET /video2_360_240_500k_dash.mpd HTTP/1.1
10792	7.892844	10.0.0.10	10.0.2.21	MP4	173	
10799	7.906610	10.0.2.21	10.0.0.10	HTTP	402	GET /video2_360_240_500k_dash.mpd HTTP/1.1
11328	7.927215	10.0.0.10	10.0.2.21	MP4	173	
11335	7.997132	10.0.2.21	10.0.0.10	HTTP	402	GET /video2_360_240_500k_dash.mpd HTTP/1.1
11908	8.001213	10.0.0.10	10.0.2.21	MP4	173	
11915	8.039718	10.0.2.21	10.0.0.10	HTTP	402	GET /video2_360_240_500k_dash.mpd HTTP/1.1
12489	8.043426	10.0.0.10	10.0.2.21	MP4	173	
12501	8.142399	10.0.2.21	10.0.0.10	HTTP	402	GET /video2_360_240_500k_dash.mpd HTTP/1.1
13075	8.164772	10.0.0.10	10.0.2.21	MP4	173	
13086	8.228191	10.0.2.21	10.0.0.10	HTTP	402	GET /video2_360_240_500k_dash.mpd HTTP/1.1
13652	8.232807	10.0.0.10	10.0.2.21	MP4	173	
13679	8.303515	10.0.2.21	10.0.0.10	HTTP	402	GET /video2_360_240_500k_dash.mpd HTTP/1.1

Fig. 20: Captura para o Portátil 2

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.

A experiência feita nesta etapa consistiu em testar cenários de rede distintos, tendo em conta *streaming* em *unicast* (transmissão entre apenas um *sender* e um *receiver*) e em *multicast* (envio de pacotes de uma entidade para um grupo de *hosts* na rede).

Para o serviço de *multicast* é possível efetuar uma análise das suas **vantagens e desvantagens** em termos de **escalabilidade e tráfego de rede**. Em termos de **escalabilidade** (aumento do número de clientes na rede), podemos verificar que o serviço *multicast* é mais escalável do que o *unicast* na medida em que responde de forma mais eficiente quando existem vários clientes na rede, o que conduz ao melhor aproveitamento da largura de banda existente e evitando o envio de pacotes redundantes. Deste modo, o serviço *multicast* é benéfico não só em termos de escalabilidade como também no âmbito do **tráfego na rede**.

A maior **desvantagem** do *streaming* em *multicast* é a sua elevada complexidade relativamente ao *streaming* em *unicast*, o que provoca maiores custos de controlo e *overhead*.

Por fim, através das capturas de tráfego no *wireshark* das transmissões *multicast* e *unicast* efetuadas (figura 21 e 22), é possível observar que a quantidade de dados enviada em cada uma das transmissões é muito semelhante, 131 k e 135 k, respetivamente. No entanto, os 135 k no *unicast* são relativos a 1 só cliente, ao passo que os 131 k do *multicast* são relativos a 3 clientes. Deste modo, tal como esperado, podemos concluir que aproveitamos melhor a largura de banda e que a solução *multicast* é muito mais **escalável**.

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets
Frame	100.0	520	100.0	342410	145 k	0
Ethernet	100.0	520	2.1	7280	3090	0
Internet Protocol Version 6	0.4	2	0.0	80	33	0
Open Shortest Path First	0.4	2	0.0	72	30	2
Internet Protocol Version 4	99.6	518	3.0	10360	4398	0
User Datagram Protocol	97.9	509	1.2	4072	1728	0
Data	97.3	506	93.5	319994	135 k	506
ADwin configuration protocol	0.6	3	0.0	156	66	3
Open Shortest Path First	1.7	9	0.1	396	168	9

Fig. 21: Quantidade de dados transmitida em *unicast*

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets
Frame	100.0	775	100.0	516265	144 k	0
Ethernet	100.0	775	2.1	10850	3039	0
Internet Protocol Version 6	0.4	3	0.0	120	33	0
Internet Control Message Protocol v6	0.4	3	0.0	48	13	3
Internet Protocol Version 4	99.6	772	3.0	15440	4325	0
User Datagram Protocol	99.6	772	1.2	6176	1730	0
Session Announcement Protocol	0.8	6	0.4	1944	544	0
Session Description Protocol	0.8	6	0.3	1800	504	6
Real-Time Transport Protocol	93.9	728	90.6	467617	131 k	0
MP4V-ES	93.9	728	88.9	458881	126 k	728
Real-time Transport Control Protocol	0.8	6	0.0	168	47	6
Data	4.1	32	2.7	13902	3894	32

Fig. 22: Quantidade de dados transmitida em *multicast*

Conclusão.

Com a realização deste guião conseguimos explorar um espectro variado dos vários protocolos associados ao *streaming* de áudio e vídeo. Neste guião conseguimos analisar o tráfego e o nível de escalabilidade na transmissão de vídeo em HTTP estático e verificar o funcionamento da técnica DASH (HTTP Dinâmico), ambos sobre o protocolo TCP. Finalmente avaliar e comparar os cenários de *unicast* e *multicast* a partir do protocolo UDP com RTP/RTCP