Engenharia de Serviços em Rede

TP2

Trabalho realizado por:

Carlos Miguel Luzia de Carvalho

Paulo Silva Sousa

Rui Emanuel Gomes Vieira

PG47092
Carlos Carvalho

PG47556
Paulo Sousa

PG47635
Rui Vieira

Grupo 25
2021/2022
Universidade do Minho

Conteúdo

I Questões e Respostas

1 Questão 1

Capture três pequenas amostras de trágefo no link de saída do servidor, respetivamente com 1 cliente (VLC), com 2 clientes (VLC e Firefox) e com 3 clientes (VLC, Firefox e ffmeg). 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)

1.1 Topologia

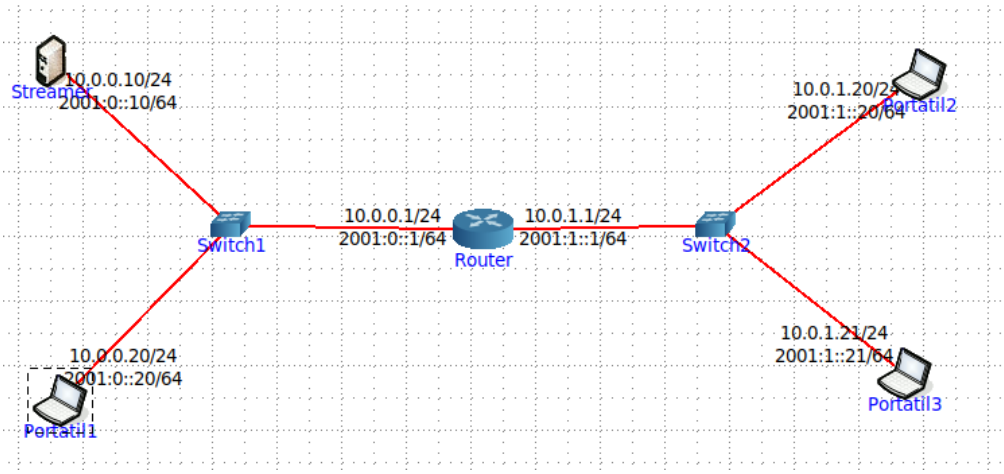


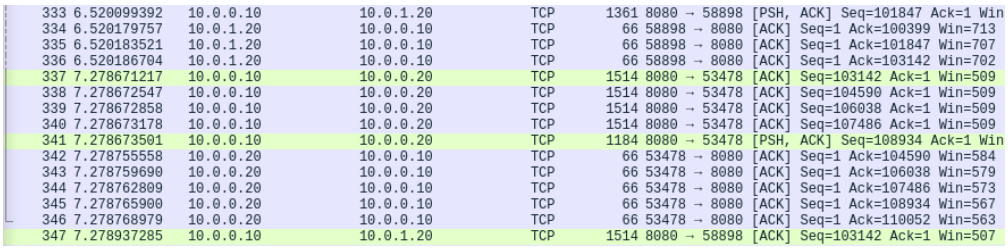
Figura 1: Topologia de Rede

1.2 Amostras de Tráfego

Nas figuras abaixo estão expostas as amostras de tráfego, no link de saída do servidor, com vários números de clientes, é de notar que em cada uma destas amostras é possível identificar o tráfego entre os diferentes clientes e o *streamer*.

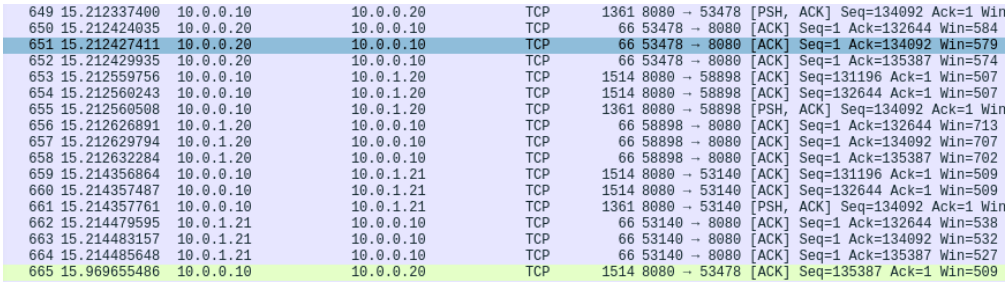
488	40.218411822	10.0.0.1	224.0.0.5	OSPF	78 Hello Packet
489	40.337172356	10.0.0.10	10.0.0.20	TCP	1514 8080 → 54442 [ACK] Seq=
490	40.337173498	10.0.0.10	10.0.0.20	TCP	1514 8080 → 54442 [ACK] Seq=
491	40.337173839	10.0.0.10	10.0.0.20	TCP	1514 8080 → 54442 [ACK] Seq=
492	40.337174162	10.0.0.10	10.0.0.20	TCP	220 8080 → 54442 [PSH, ACK]
493	40.337242713	10.0.0.20	10.0.0.10	TCP	66 54442 → 8080 [ACK] Seq=
494	40.337246275	10.0.0.20	10.0.0.10	TCP	66 54442 → 8080 [ACK] Seq=
495	40.337248431	10.0.0.20	10.0.0.10	TCP	66 54442 → 8080 [ACK] Seq=
496	40.337250459	10.0.0.20	10.0.0.10	TCP	66 54442 → 8080 [ACK] Seq=
497	41.079059178	10.0.0.10	10.0.0.20	HTTP	66 HTTP/1.0 200 OK
498	41.122053330	10.0.0.20	10.0.0.10	TCP	66 54442 → 8080 [ACK] Seq=

Figura 2: Amostras de tráfego no link de saída do servidor, com 1 cliente (VLC)



333	6.520099392	10.0.0.10	10.0.1.20	TCP	1361 8080 → 58898 [PSH, ACK] Seq=101847 Ack=1 Win
334	6.520179757	10.0.1.20	10.0.0.10	TCP	66 58898 → 8080 [ACK] Seq=1 Ack=100399 Win=713
335	6.520183521	10.0.1.20	10.0.0.10	TCP	66 58898 → 8080 [ACK] Seq=1 Ack=101847 Win=707
336	6.520186704	10.0.1.20	10.0.0.10	TCP	66 58898 → 8080 [ACK] Seq=1 Ack=103142 Win=702
337	7.278671217	10.0.0.10	10.0.0.20	TCP	1514 8080 → 53478 [ACK] Seq=103142 Ack=1 Win=509
338	7.278672547	10.0.0.10	10.0.0.20	TCP	1514 8080 → 53478 [ACK] Seq=104590 Ack=1 Win=509
339	7.278672858	10.0.0.10	10.0.0.20	TCP	1514 8080 → 53478 [ACK] Seq=106038 Ack=1 Win=509
340	7.278673178	10.0.0.10	10.0.0.20	TCP	1514 8080 → 53478 [ACK] Seq=107486 Ack=1 Win=509
341	7.278673501	10.0.0.10	10.0.0.20	TCP	1184 8080 → 53478 [PSH, ACK] Seq=108934 Ack=1 Win
342	7.278755558	10.0.0.20	10.0.0.10	TCP	66 53478 → 8080 [ACK] Seq=1 Ack=104590 Win=584
343	7.278759690	10.0.0.20	10.0.0.10	TCP	66 53478 → 8080 [ACK] Seq=1 Ack=106038 Win=579
344	7.278762809	10.0.0.20	10.0.0.10	TCP	66 53478 → 8080 [ACK] Seq=1 Ack=107486 Win=573
345	7.278765900	10.0.0.20	10.0.0.10	TCP	66 53478 → 8080 [ACK] Seq=1 Ack=108934 Win=567
346	7.278768979	10.0.0.20	10.0.0.10	TCP	66 53478 → 8080 [ACK] Seq=1 Ack=110052 Win=563
347	7.278937285	10.0.0.10	10.0.1.20	TCP	1514 8080 → 58898 [ACK] Seq=103142 Ack=1 Win=507

Figura 3: Amostras de tráfego no link de saída do servidor,com 2 clientes (VLC e Firefox)



649	15.212337400	10.0.0.10	10.0.0.20	TCP	1361 8080 → 53478 [PSH, ACK] Seq=134092 Ack=1 Win
650	15.212424035	10.0.0.20	10.0.0.10	TCP	66 53478 → 8080 [ACK] Seq=1 Ack=132644 Win=584
651	15.212427411	10.0.0.20	10.0.0.10	TCP	66 53478 → 8080 [ACK] Seq=1 Ack=134092 Win=579
652	15.212429935	10.0.0.20	10.0.0.10	TCP	66 53478 → 8080 [ACK] Seq=1 Ack=135387 Win=574
653	15.212559756	10.0.0.10	10.0.1.20	TCP	1514 8080 → 58898 [ACK] Seq=131196 Ack=1 Win=507
654	15.212560243	10.0.0.10	10.0.1.20	TCP	1514 8080 → 58898 [ACK] Seq=132644 Ack=1 Win=507
655	15.212560508	10.0.0.10	10.0.1.20	TCP	1361 8080 → 58898 [PSH, ACK] Seq=134092 Ack=1 Win
656	15.212626891	10.0.1.20	10.0.0.10	TCP	66 58898 → 8080 [ACK] Seq=1 Ack=132644 Win=713
657	15.212629794	10.0.1.20	10.0.0.10	TCP	66 58898 → 8080 [ACK] Seq=1 Ack=134092 Win=707
658	15.212632284	10.0.1.20	10.0.0.10	TCP	66 58898 → 8080 [ACK] Seq=1 Ack=135387 Win=702
659	15.214356864	10.0.0.10	10.0.1.21	TCP	1514 8080 → 53140 [ACK] Seq=131196 Ack=1 Win=509
660	15.214357487	10.0.0.10	10.0.1.21	TCP	1514 8080 → 53140 [ACK] Seq=132644 Ack=1 Win=509
661	15.214357761	10.0.0.10	10.0.1.21	TCP	1361 8080 → 53140 [PSH, ACK] Seq=134092 Ack=1 Win
662	15.214479595	10.0.1.21	10.0.0.10	TCP	66 53140 → 8080 [ACK] Seq=1 Ack=132644 Win=538
663	15.214483157	10.0.1.21	10.0.0.10	TCP	66 53140 → 8080 [ACK] Seq=1 Ack=134092 Win=532
664	15.214485648	10.0.1.21	10.0.0.10	TCP	66 53140 → 8080 [ACK] Seq=1 Ack=135387 Win=527
665	15.969655486	10.0.0.10	10.0.0.20	TCP	1514 8080 → 53478 [ACK] Seq=135387 Ack=1 Win=509

Figura 4: Amostras de tráfego no link de saída do servidor, com 3 clientes (VLC, Firefox e ffmeg)

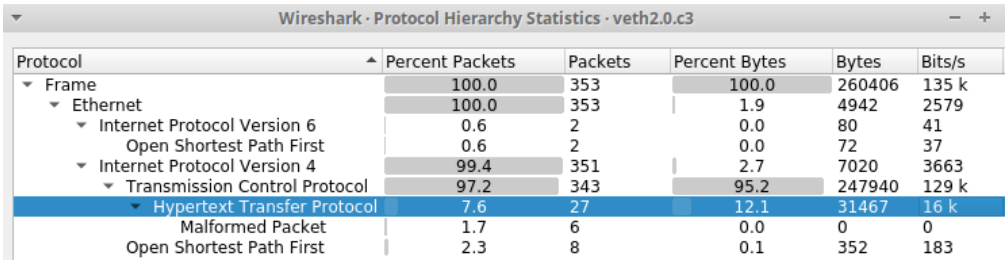
1.3 Taxa em *bps* necessária

Para identificar a taxa utilizamos o *ffmpeg -i video1.mp4* e o wireshark.

Através do *ffmpeg* conseguimos obter a taxa teórica necessária, 11000 bps. Já através da captura do wireshark obtemos uma taxa real de 16000 bps.

```
Input #0, mov,mp4,m4a,3gp,3g2,mj2, from 'video1.mp4':
  Metadata:
    major_brand     : isom
    minor_version   : 512
    compatible_brands: isomiso2avc1mp41
    encoder         : Lavf58.29.100
  Duration: 00:00:20.70, 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
```

Figura 5: Taxa Teórica utilizando o comando *ffmpeg -i video1.mp4*



Wireshark · Protocol Hierarchy Statistics · veth2.0.c3

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s
Frame	100.0	353	100.0	260406	135 k
Ethernet	100.0	353	1.9	4942	2579
Internet Protocol Version 6	0.6	2	0.0	80	41
Open Shortest Path First	0.6	2	0.0	72	37
Internet Protocol Version 4	99.4	351	2.7	7020	3663
Transmission Control Protocol	97.2	343	95.2	247940	129 k
Hypertext Transfer Protocol	7.6	27	12.1	31467	16 k
Malformed Packet	1.7	6	0.0	0	0
Open Shortest Path First	2.3	8	0.1	352	183

Figura 6: Taxa Real utililzando uma captura de Wireshark

A diferença entre as duas taxas deve-se ao facto de existir picos na débito na captura, o que faz com que a média da taxa capturada é maior no início e converge para o valor teórico à medida que o tempo de captura aumenta.

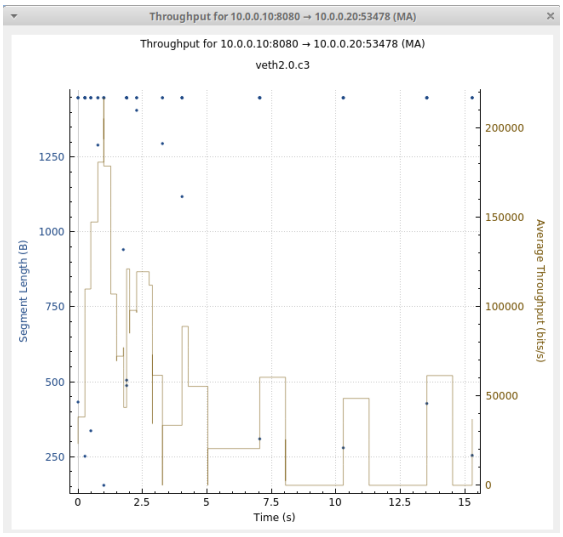


Figura 7: Gráfico com o débito utilizado ao longo do tempo

1.4 Encapsulamento usado e o número total de fluxos gerados

Na captura de wireshark que obtivemos conseguimos observar que na camada física da pilha protocolar é utilizado o protocolo Ethernet, na camada Rede é utilizado o Internet Protocol Version 4 (IPV4), na camada de Transporte o Transmission Control Protocol (TCP) e, por último, na camada de Aplicação, usamos o Hypertext Transfer Protocol (HTTP).

Através da figura 8 e 9 observamos que o protocolo HTTP apenas é identificado no primeiro pacote enviado. Isto deve-se ao facto de ser este pacote que leva o cabeçalho deste protocolo, os restantes apenas transportam o payload, pois foi fragmentado.

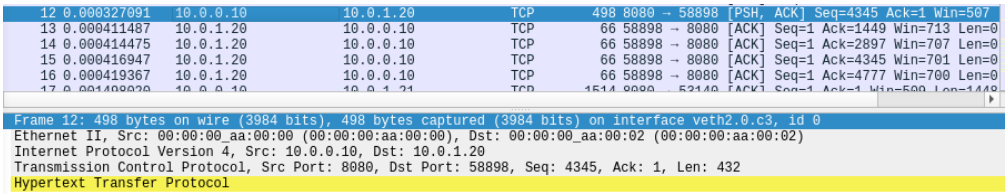


Figura 8: Primeiro pacote de uma transmissão HTTP

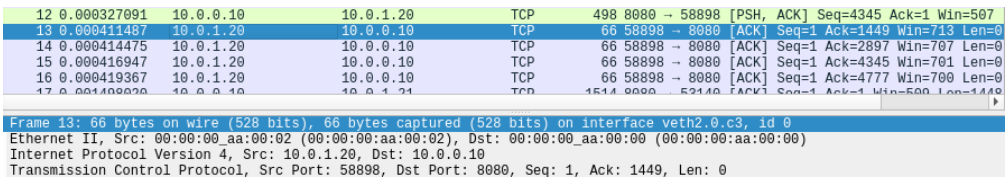


Figura 9: Segundo pacote de uma transmissão HTTP

As 3 figuras a baixo traduzem o tráfego no link de saída do servidor, com 3 clientes (VLC, Firefox e ffmeg), com vários filtros de stream, tratando-se ordenadamente dos vários fluxos por cliente, uma vez que para cada filtro só aparece o tráfego entre um determinado cliente e o *streamer*,pelas mesmas portas. Já na quarta figura, aplicando o filtro de novo, verificamos que não existem outros fluxos, ou seja, não existe nenhum cliente a realizar tráfego por mais que uma porta encontrando apenas de 3 tráfegos correspondendo a cada um dos clientes.

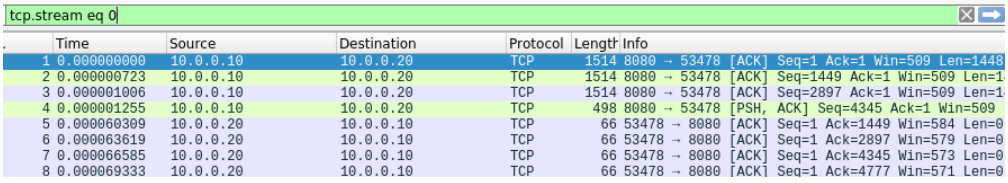


Figura 10: Tráfego de rede com filtro de fluxo 0

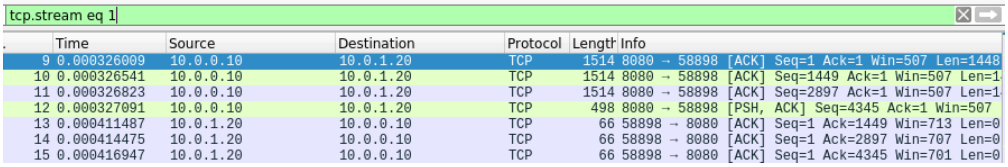


Figura 11: Tráfego de rede com filtro de fluxo 1

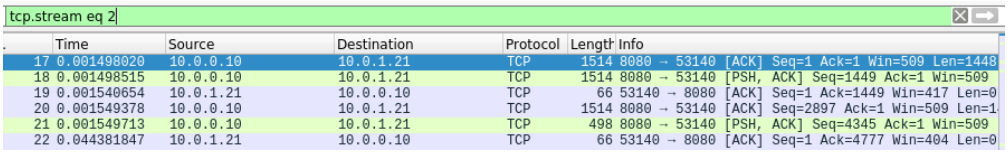


Figura 12: Tráfego de rede com filtro de fluxo 2

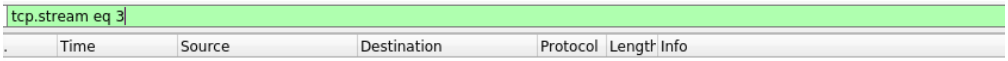


Figura 13: Tráfego de rede com filtro de fluxo 3

1.5 Escalabilidade da Solução

Relativamente a escalabilidade tratando-se de uma transmissão unicast, podemos observar que não é escalável ou seja, há um aumento linear de acordo com o número de clientes a que o streamer tem de transmitir.

2 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.

```
Input #0, mov,mp4,m4a,3gp,3g2,mj2, from 'video2.mp4':
  Metadata:
    major_brand     : isom
    minor_version   : 512
    compatible_brands: isomiso2avc1mp41
    encoder         : Lavf58.29.100
  Duration: 00:00:09.87, start: 0.000000, bitrate: 26 kb/s
    Stream #0:0(und): Video: h264 (High) (avc1 / 0x31637661), yuv420p, 200x200,
22 kb/s, 30 fps, 30 tbr, 15360 tbn, 60 tbc (default)
    Metadata:
      handler name    : VideoHandler
```

Figura 14: Largura de banda para streaming

Como podemos observar na figura acima, a largura de banda necessária para que o cliente de streaming consiga receber o vídeo no firefox é de 22 kb/s.

```
Frame 640: 401 bytes on wire (3208 bits), 401 bytes captured (3208 bits) on interface veth2.0.66, id 0
Ethernet II, Src: 00:00:00_aa:00:01 (00:00:00:aa:00:01), Dst: 00:00:00_aa:00:00 (00:00:00:aa:00:00)
Internet Protocol Version 4, Src: 10.0.0.20, Dst: 10.0.0.10
Transmission Control Protocol, Src Port: 36386, Dst Port: 9999, Seq: 1, Ack: 1, Len: 335
Hypertext Transfer Protocol
```

Figura 15: Captura de Wireshark

Na captura de wireshark acima observamos que na camada física da pilha protocolar é utilizado o protocolo Ethernet, na camada Rede é utilizado o Internet Protocol Version 4 (IPV4), na camada de Transporte o Transmission Control Protocol (TCP) e, por último, na camada de Aplicação, usamos o Hypertext Transfer Protocol (HTTP).

3 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.

De modo a obter os débitos necessários para o envio de cada vídeo, fomos verificar os seus valores ao ficheiro *video_ manifest.mpd*

```
</SegmentList>
<Representation id="1" mimeType="video/mp4" codecs="avc3.64000c" width="180" height="120" frameRate="30" sar="1:1" startWithSAP="0" bandwidth="109264">
 <BaseURL>video2_180_120_200k_dash.mp4</BaseURL>
 <SegmentList timescale="15360" duration="7680">
   <SegmentURL mediaRange="928-5606" indexRange="928-971"/>
   <SegmentURL mediaRange="5607-8721" indexRange="5607-5650"/>
   <SegmentURL mediaRange="8722-12372" indexRange="8722-8765"/>
   <SegmentURL mediaRange="12373-24729" indexRange="12373-12416"/>
   <SegmentURL mediaRange="24730-31236" indexRange="24730-24773"/>
   <SegmentURL mediaRange="31237-38284" indexRange="31237-31280"/>
   <SegmentURL mediaRange="38285-52573" indexRange="38285-38328"/>
   <SegmentURL mediaRange="52574-59149" indexRange="52574-52617"/>
   <SegmentURL mediaRange="59150-64495" indexRange="59150-59193"/>
   <SegmentURL mediaRange="64496-69829" indexRange="64496-64539"/>
   <SegmentURL mediaRange="69830-81163" indexRange="69830-69873"/>
   <SegmentURL mediaRange="81164-83507" indexRange="81164-81207"/>
   <SegmentURL mediaRange="83508-90028" indexRange="83508-83551"/>
   <SegmentURL mediaRange="90029-101605" indexRange="90029-90072"/>
   <SegmentURL mediaRange="101606-107784" indexRange="101606-101649"/>
   <SegmentURL mediaRange="107785-114129" indexRange="107785-107828"/>
   <SegmentURL mediaRange="114130-126184" indexRange="114130-114173"/>
   <SegmentURL mediaRange="126185-132318" indexRange="126185-126228"/>
   <SegmentURL mediaRange="132319-134301" indexRange="132319-132362"/>
  <SegmentURL mediaRange="134302-134758" indexRange="134302-134345"/>
 </SegmentList>
```

Figura 16: Video com resolução 180x120

```
<Representation id="2" mimeType="video/mp4" codecs="avc3.640014" width="360" height="240" frameRate="30" sar="1:1" startWithSAP="0" bandwidth="266245">
 <BaseURL>video2_360_240_500k_dash.mp4</BaseURL>
 <SegmentList timescale="15360" duration="7680">
   <SegmentURL mediaRange="928-11582" indexRange="928-971"/>
   <SegmentURL mediaRange="11583-19519" indexRange="11583-11626"/>
   <SegmentURL mediaRange="19520-28767" indexRange="19520-19563"/>
   <SegmentURL mediaRange="28768-58810" indexRange="28768-28811"/>
   <SegmentURL mediaRange="58811-76239" indexRange="58811-58854"/>
   <SegmentURL mediaRange="76240-94873" indexRange="76240-76283"/>
   <SegmentURL mediaRange="94874-131514" indexRange="94874-94917"/>
   <SegmentURL mediaRange="131515-149104" indexRange="131515-131558"/>
   <SegmentURL mediaRange="149105-161487" indexRange="149105-149148"/>
   <SegmentURL mediaRange="161488-174103" indexRange="161488-161531"/>
   <SegmentURL mediaRange="174104-201249" indexRange="174104-174147"/>
   <SegmentURL mediaRange="201250-207399" indexRange="201250-201293"/>
   <SegmentURL mediaRange="207400-221131" indexRange="207400-207443"/>
   <SegmentURL mediaRange="221132-248284" indexRange="221132-221175"/>
   <SegmentURL mediaRange="248285-263410" indexRange="248285-248328"/>
   <SegmentURL mediaRange="263411-278963" indexRange="263411-263454"/>
   <SegmentURL mediaRange="278964-308769" indexRange="278964-279007"/>
   <SegmentURL mediaRange="308770-323369" indexRange="308770-308813"/>
   <SegmentURL mediaRange="323370-327750" indexRange="323370-323413"/>
  <SegmentURL mediaRange="327751-328369" indexRange="327751-327794"/>
 </SegmentList>
```

Figura 17: Video com resolução 360x240

```
<Representation id="3" mimeType="video/mp4" codecs="avc3.64001e" width="540" height="360" frameRate="30" sar="1:1" startWithSAP="0" bandwidth="476314">
 <BaseURL>video2_540_360_1000k_dash.mp4</BaseURL>
 <SegmentList timescale="15360" duration="7680">
   <SegmentURL mediaRange="928-21421" indexRange="928-971"/>
   <SegmentURL mediaRange="21422-35629" indexRange="21422-21465"/>
   <SegmentURL mediaRange="35630-49575" indexRange="35630-35673"/>
   <SegmentURL mediaRange="49576-107411" indexRange="49576-49619"/>
   <SegmentURL mediaRange="107412-140562" indexRange="107412-107455"/>
   <SegmentURL mediaRange="140563-177479" indexRange="140563-140606"/>
   <SegmentURL mediaRange="177480-246353" indexRange="177480-177523"/>
   <SegmentURL mediaRange="246354-280372" indexRange="246354-246397"/>
   <SegmentURL mediaRange="280373-304928" indexRange="280373-280416"/>
   <SegmentURL mediaRange="304929-326142" indexRange="304929-304972"/>
   <SegmentURL mediaRange="326143-372119" indexRange="326143-326186"/>
   <SegmentURL mediaRange="372120-382727" indexRange="372120-372163"/>
   <SegmentURL mediaRange="382728-406950" indexRange="382728-382771"/>
   <SegmentURL mediaRange="406951-452892" indexRange="406951-406994"/>
   <SegmentURL mediaRange="452893-478375" indexRange="452893-452936"/>
   <SegmentURL mediaRange="478376-504060" indexRange="478376-478419"/>
   <SegmentURL mediaRange="504061-554253" indexRange="504061-504104"/>
   <SegmentURL mediaRange="554254-578907" indexRange="554254-554297"/>
   <SegmentURL mediaRange="578908-586512" indexRange="578908-578951"/>
  <SegmentURL mediaRange="586513-587454" indexRange="586513-586556"/>
 </SegmentList>
```

Figura 18: Video com resolução 546x360

Como podemos observar através das figuras acima, o vídeo com resolução 180x120 requere de uma largura de banda de 109264 bps, o vídeo com resolução 360x240 requere 266245 bps e o vídeo com resolução 540x360 requere 476314 bps.

Por defeito, na topologia criada na questão 1, quando era feito o pedido de envio de um vídeo, era enviado o vídeo com maior resolução.

Assim, tivemos de alterar a topologia para que fosse possivel enviar o vídeo de maior resolução para o portátil 1 e o vídeo com menor resolução para o portátil 2, como podemos ver na figura abaixo.

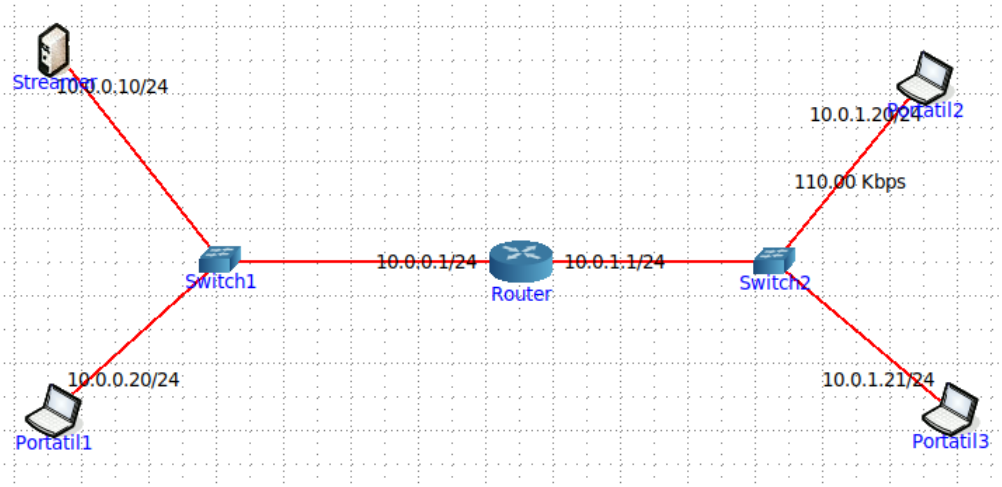


Figura 19: Topologia alterada para suportar limites de largura de banda

Nesta nova topologia adicionamos um limite de largura de banda na ligação entre o Switch 2 e o portátil 2 de 110kb/s. Assim, o único vídeo que é possível enviar é o vídeo de menor resolução e é garantido que o portátil 2 recebe esse vídeo.

Estes resultados podem ser comprovados com as figuras abaixo.

Na primeira podemos verificar que o portátil 1 com IP 10.0.0.20 fez um pedido *GET /video2_540_360_1000k_dash.mp4 HTTP/1.1*, que corresponde ao pedido do vídeo com resolução 540x360.

Na segunda podemos verificar que o portátil 2 com IP 10.0.1.20 fez um pedido *GET /video2_180_120_1000k_dash.mp4 HTTP/1.1*, que corresponde ao pedido do vídeo com menor resolução.

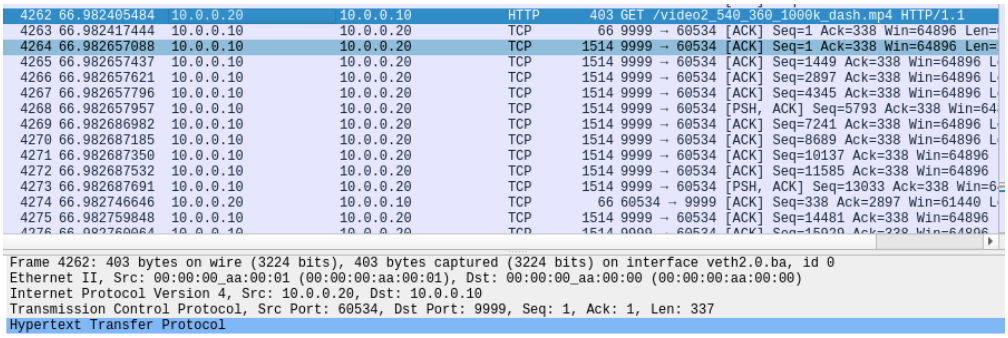


Figura 20: Pedido do vídeo do portátil 1

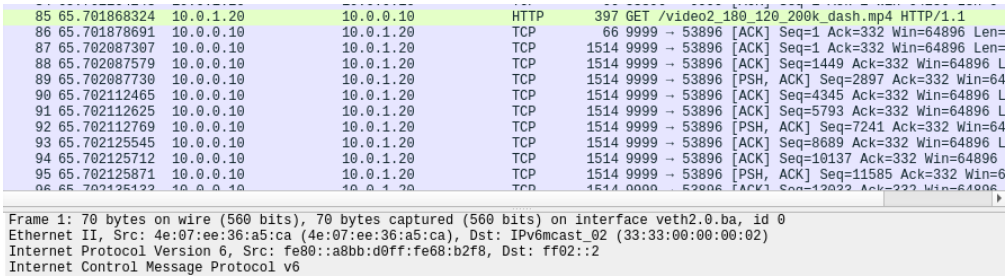


Figura 21: Pedido do vídeo do portátil 2

4 Questão 4

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

O conteúdo multimédia é capturado e armazenado num servidor HTTP e entregue usando HTTP, este conteúdo existe no servidor em duas partes: Media Presentation Description (MPD), que descreve um **manifest file** do conteúdo disponível, as suas várias alternativas, os seus endereços URL e outras características, como os segmentos, que contêm os fluxos de bits de multimédia reais na forma de chunks, em arquivos únicos ou múltiplos.

Para reproduzir o conteúdo, o cliente DASH primeiro obtém o MPD. Ao analisar o MPD, o cliente DASH aprende sobre o tempo do programa, disponibilidade de conteúdo de media, tipos de media, resoluções, larguras de banda mínimas e máximas e a existência de várias alternativas codificadas de componentes de multimédia, recursos de acessibilidade e gestão de direitos digitais necessários (DRM) , localizações de componentes de media na rede e outras características de conteúdo. Usando essas informações, o cliente DASH seleciona a alternativa codificada apropriada e começa a transmitir o conteúdo buscando os segmentos usando solicitações HTTP GET.

Após o armazenamento em buffer apropriado para permitir variações na taxa de transferência da rede, o cliente continua a buscar os segmentos subsequentes e também monitoriza as flutuações da largura de banda da rede. Dependendo das suas medidas, o cliente decide como se adaptar à largura de banda disponível, buscando segmentos de diferentes alternativas (com taxas de bits mais baixas ou mais altas) para manter um buffer adequado.

A especificação MPEG-DASH define apenas o MPD e os formatos de segmento. A entrega do MPD e os formatos de codificação de media contendo os segmentos, bem como o comportamento do cliente para busca, heurística de adaptação e reprodução de conteúdo, estão fora do alcance do MPEG-DASH.

```
</SegmentList>
<Representation id="1" mimeType="video/mp4" codecs="avc3.64000c" width="180" height="120" frameRate="30" sar="1:1" startWithSAP="0" bandwidth="109264">
 <BaseURL>video2_180_120_200k_dash.mp4</BaseURL>
 <SegmentList timescale="15360" duration="7680">
   <SegmentURL mediaRange="928-5606" indexRange="928-971"/>
   <SegmentURL mediaRange="5607-8721" indexRange="5607-5650"/>
   <SegmentURL mediaRange="8722-12372" indexRange="8722-8765"/>
   <SegmentURL mediaRange="12373-24729" indexRange="12373-12416"/>
   <SegmentURL mediaRange="24730-31236" indexRange="24730-24773"/>
   <SegmentURL mediaRange="31237-38284" indexRange="31237-31280"/>
   <SegmentURL mediaRange="38285-52573" indexRange="38285-38328"/>
   <SegmentURL mediaRange="52574-59149" indexRange="52574-52617"/>
   <SegmentURL mediaRange="59150-64495" indexRange="59150-59193"/>
   <SegmentURL mediaRange="64496-69829" indexRange="64496-64539"/>
   <SegmentURL mediaRange="69830-81163" indexRange="69830-69873"/>
   <SegmentURL mediaRange="81164-83507" indexRange="81164-81207"/>
   <SegmentURL mediaRange="83508-90028" indexRange="83508-83551"/>
   <SegmentURL mediaRange="90029-101605" indexRange="90029-90072"/>
   <SegmentURL mediaRange="101606-107784" indexRange="101606-101649"/>
   <SegmentURL mediaRange="107785-114129" indexRange="107785-107828"/>
   <SegmentURL mediaRange="114130-126184" indexRange="114130-114173"/>
   <SegmentURL mediaRange="126185-132318" indexRange="126185-126228"/>
   <SegmentURL mediaRange="132319-134301" indexRange="132319-132362"/>
  <SegmentURL mediaRange="134302-134758" indexRange="134302-134345"/>
 </SegmentList>
```

Figura 22: Parte do Ficheiro MPD

5 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.

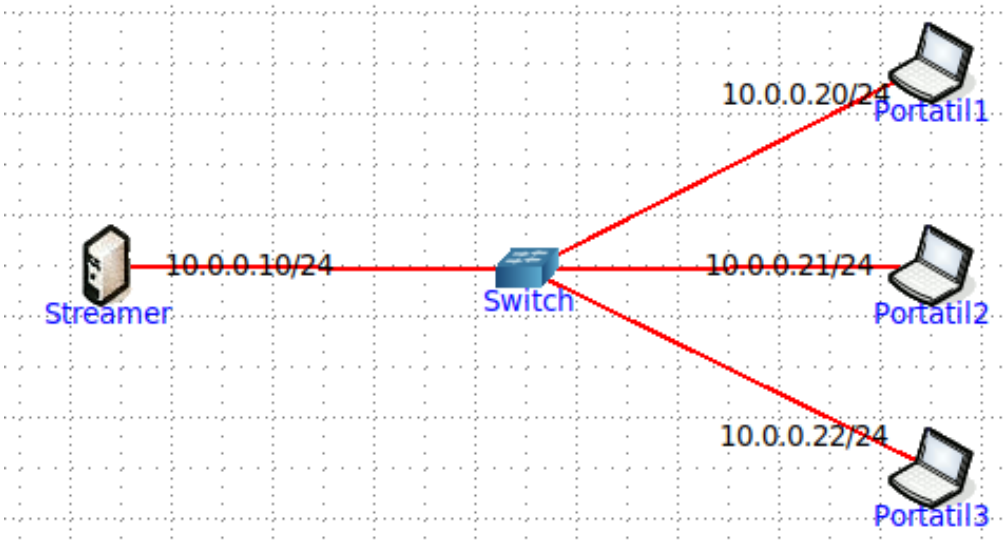


Figura 23: Topologia Multicast

1	0.000000000	10.0.0.10	10.0.1.21	UDP	96 49890 → 5555 Len=54
2	0.053579738	10.0.0.10	10.0.1.21	UDP	96 49890 → 5555 Len=54
3	0.104702568	10.0.0.10	10.0.1.21	UDP	96 49890 → 5555 Len=54
4	0.157737948	10.0.0.10	10.0.1.21	UDP	96 49890 → 5555 Len=54
5	0.200324881	10.0.0.10	10.0.1.21	UDP	96 49890 → 5555 Len=54
6	0.253018830	10.0.0.10	10.0.1.21	UDP	96 49890 → 5555 Len=54
7	0.305454554	10.0.0.10	10.0.1.21	UDP	96 49890 → 5555 Len=54
8	0.357948275	10.0.0.10	10.0.1.21	UDP	96 49890 → 5555 Len=54
9	0.400450748	10.0.0.10	10.0.1.21	UDP	96 49890 → 5555 Len=54
10	0.409733291	10.0.0.1	224.0.0.5	OSPF	78 Hello Packet
11	0.456551818	10.0.0.10	10.0.1.21	UDP	1514 49890 → 5555 Len=1472
12	0.456747963	10.0.0.10	10.0.1.21	UDP	1514 49890 → 5555 Len=1472
13	0.456811529	10.0.0.10	10.0.1.21	UDP	1514 49890 → 5555 Len=1472
14	0.456863170	10.0.0.10	10.0.1.21	UDP	1514 49890 → 5555 Len=1472
15	0.456913272	10.0.0.10	10.0.1.21	UDP	222 49890 → 5555 Len=180
16	0.508992600	10.0.0.10	10.0.1.21	UDP	98 49890 → 5555 Len=56

Figura 24: Trafego Unicast

1	0.000000000	10.0.0.10	224.0.0.100	UDP	129 54472 → 5555 Len=87
2	0.042342384	10.0.0.10	224.0.0.100	UDP	824 54472 → 5555 Len=782
3	0.094660999	10.0.0.10	224.0.0.100	UDP	1491 54472 → 5555 Len=1449
4	0.147422052	10.0.0.10	224.0.0.100	UDP	651 54472 → 5555 Len=609
5	0.201490190	10.0.0.10	224.0.0.100	UDP	1242 54472 → 5555 Len=1200
6	0.245970320	10.0.0.10	224.0.0.100	UDP	942 54472 → 5555 Len=900
7	0.301765166	10.0.0.10	224.0.0.100	UDP	1514 54472 → 5555 Len=1472
8	0.304012078	10.0.0.10	224.0.0.100	UDP	1514 54472 → 5555 Len=1472
9	0.304280084	10.0.0.10	224.0.0.100	UDP	1514 54472 → 5555 Len=1472
10	0.304550674	10.0.0.10	224.0.0.100	UDP	1514 54472 → 5555 Len=1472
11	0.304748917	10.0.0.10	224.0.0.100	UDP	241 54472 → 5555 Len=199
12	0.346851551	10.0.0.10	224.0.0.100	UDP	1109 54472 → 5555 Len=1067
13	0.391181107	10.0.0.10	224.0.0.100	UDP	1188 54472 → 5555 Len=1146
14	0.447234867	10.0.0.10	224.0.0.100	UDP	591 54472 → 5555 Len=549
15	0.496123087	10.0.0.10	224.0.0.100	UDP	154 54472 → 5555 Len=112
16	0.550089242	10.0.0.10	224.0.0.100	UDP	1217 54472 → 5555 Len=1175

Figura 25: Trafego Multicast

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s
Frame	100.0	463	100.0	310968	149 k
Ethernet	100.0	463	2.1	6482	3123
Internet Protocol Version 6	0.2	1	0.0	40	19
Internet Control Message Protocol v6	0.2	1	0.0	16	7
Internet Protocol Version 4	99.8	462	3.0	9240	4452
User Datagram Protocol	99.8	462	1.2	3696	1781
Session Announcement Protocol	0.6	3	0.3	972	468
Session Description Protocol	0.6	3	0.3	900	433
Real-Time Transport Protocol	69.5	322	50.6	157307	75 k
MP4V-ES	69.5	322	49.3	153443	73 k
Real-time Transport Control Protocol	0.6	3	0.0	84	40
Data	28.9	134	42.8	133131	64 k

Figura 26: Multicast com 1 cliente

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s
Frame	100.0	297	100.0	178618	133 k
Ethernet	100.0	297	2.3	4158	3108
Internet Protocol Version 4	100.0	297	3.3	5940	4440
User Datagram Protocol	100.0	297	1.3	2376	1776
Session Announcement Protocol	1.0	3	0.5	972	726
Session Description Protocol	1.0	3	0.5	900	672
Real-Time Transport Protocol	94.3	280	88.9	158743	118 k
MP4V-ES	94.3	280	87.0	155383	116 k
Real-time Transport Control Protocol	1.0	3	0.0	84	62
Data	3.7	11	3.6	6345	4743

Figura 27: Multicast com 2 clientes

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s
Frame	100.0	614	100.0	393966	143 k
Ethernet	100.0	614	2.2	8596	3125
Internet Protocol Version 6	0.3	2	0.0	80	29
Internet Control Message Protocol v6	0.3	2	0.0	32	11
Internet Protocol Version 4	99.7	612	3.1	12240	4450
User Datagram Protocol	99.7	612	1.2	4896	1780
Session Announcement Protocol	0.7	4	0.3	1296	471
Session Description Protocol	0.7	4	0.3	1200	436
Real-Time Transport Protocol	77.7	477	79.4	312728	113 k
MP4V-ES	77.7	477	77.9	307004	111 k
Real-time Transport Control Protocol	0.7	4	0.0	112	40
Data	20.7	127	13.7	53986	19 k

Figura 28: Multicast com 3 clientes

O tipo de roteamento Unicast é um tipo de comunicação na qual um quadro é enviado de um host e enviado a um destino específico, por exemplo um ping que é realizado no terminal de comando, enviará uma quantidade "X" de pacotes de dados para um endereço especifico, como podemos observar na figura 22, onde o streamer envia um pacote especificamente para o Portátil 3.

Na transmissão Unicast, há apenas um remetente e um receptor. Esta é a forma predominante de transmissão em redes locais e na Internet, onde ocorre a transmissão *peer-to-peer*.

Já um tipo de roteamento Multicast é um tipo de comunicação na qual um quadro ou pacote é enviado para um grupo específico de dispositivos ou clientes, no caso especifico escolheu-se um endereço de envio especial, que é um endereço de grupo, sendo este o 224.0.0.100 porta 5555, como podemos observar na figura 23 onde o streamer envia pacotes para o endereço 224.0.0.100.

Os clientes da transmissão multicast devem ser membros de um grupo multicast lógico para receber as informações, ou seja, invés de ser enviado para um único destino (endereço IP específico), o tráfego de multicast, permite o envio de informações para um determinado grupo de clientes, cada um com um endereço IP diferente, ao mesmo tempo.

Durante uma transmissão Multicast, o transmissor envia os pacotes de dados somente uma vez, ficando a cargo dos receptores captarem esta transmissão e reproduzi-la. Esta técnica diminui consideravelmente o tráfego em diversas situações, como por exemplo, quando vários clientes estão a assistir a uma transmissão de um jogo de futebol, propagado por um servidor.

Para além disso a transmissão via Multicast é vantajosa em relação à escalabilidade uma vez que Multicast permitem que aplicações escalem, ou seja, que sirvam a um grande número de usuários sem sobrecarregar a rede, mais facilmente que uma transmissão em Unicast, uma vez que como visto no ponto 1.5, transmissões Unicast não são escaláveis uma vez que aumentam linearmente com o aumento do número de clientes, pelo contrário as transmissões multicast mantém-se ao longo do tempo uma vez que só enviam um pacote para o endereço de grupo no caso da figura 224.0.0.100 e só nesse IP é feito o tratamento do pacote,não dependendo assim do número de clientes. Isto pode ser comprovado com a observação das figuras 25, 26 e 27 onde apesar de existirem alterações de tamanho no protocol não há necessáriamente um aumento ao longo do tempo uma vez que na imagem 25 temos um uso de 75kbits/s já na 26 temos 118 kbits/s o que pode levar a crer que houve um aumento, porém na figura 27 o tamanho volta a diminuir para 113kbits/s.

Relativamente a desvantagens em transmissões Multicast o encaminhamento eficiente da pacotes aos múltiplos destinatários não garante entretanto a entrega confiável dos dados, mais precisamente, serviços importantes como o controle de erro, controle de fluxos e de congestionamento que são tratados na camada superior, de transporte.

II Conclusão

Neste projeto foi-nos possível expandir os nossos conhecimentos sobre streaming de aúdio e vídeo a pedido e em tempo real, percebendo as opções disponíveis a nível da pilha protocolar e conhecendo melhor os formatos multimédia utilizados.

Concluindo, este trabalho ajudou-nos bastante a consolidar os conhecimentos sobre streaming de aúdio e vídeo da unidade curricular de Engenharia de Serviços em Rede.