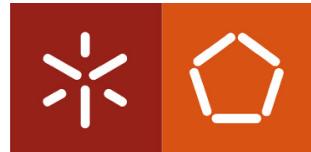


UNIVERSIDADE DO MINHO

ESCOLA DE ENGENHARIA



Engenharia de Serviços em Rede

Mestrado em Engenharia Informática

Streaming de áudio e vídeo a pedido e em tempo real

Trabalho Prático 1 - PL51

Francisco Novo - PG50374
Francisco Izquierdo - PG50384
Tiago Ribeiro - PG50779

Outubro, 2022

Conteúdo

Introdução	2
Etapa 1	3
Etapa 2	7
Etapa 3	11
Conclusão	13

Introdução

No âmbito da Unidade Curricular de Engenharia de Serviços em Rede, foi desenvolvido o primeiro trabalho prático proposto pelos docentes. Neste projeto são abordados conceitos acerca de streaming de áudio e vídeo a pedido e em tempo real.

Para a realização deste trabalho, o emulador CORE e a ferramenta de análise do tráfego de rede *Wireshark* foram essenciais para as soluções apresentadas.

O objetivo deste trabalho consiste em perceber as opções disponíveis em termos de pilha protocolar e as diferenças conceptuais entre elas e perceber como funcionam os vários formatos multimédia e como estes podem ser empacotados. O mesmo encontra-se dividido em três partes, cada uma correspondente a uma etapa do trabalho.

Etapa 1

O objetivo desta etapa é testar o streaming sobre HTTP, que apesar de pouco eficiente é muito popular na Internet.

A primeira tarefa proposta pelo enunciado exige a construção de uma topologia base no CORE, com pelo menos um servidor, 3 portáteis, 2 switches e um ou dois routers. Assim, o grupo desenvolveu a topologia que se encontra na figura 1.

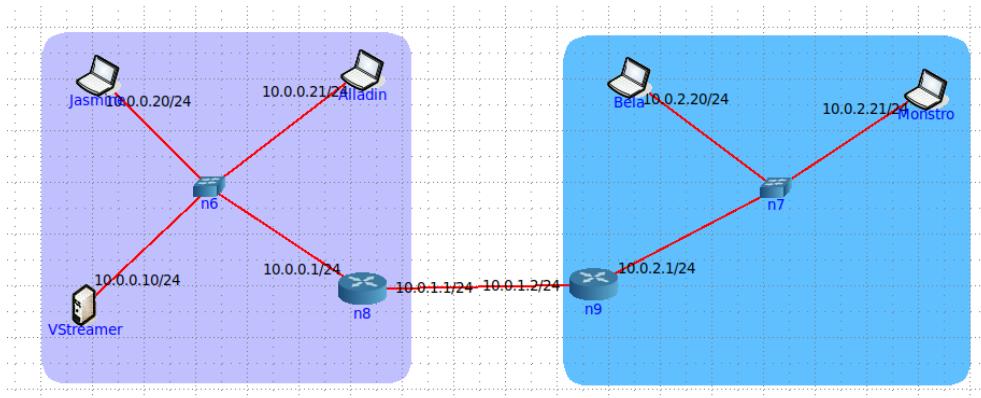


Figura 1: Topologia

Foi criado um ficheiro HTML com o propósito de apresentar o vídeo no Firefox.

```
core@xubuncore:~$ cat video.html
<!DOCTYPE html>
<html>
  <head>
    <title> Streaming ESR - Etapa 1 </title>
  </head>
  <body>
    <h1> Streaming ESR: Etapa 1 (Exercício 6) </h1>
    <video controls autoplay>
      <source src="http://10.0.0.10:8080">
    </video>
  </body>
</html>
```

Figura 2: Ficheiro HTML criado

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 ffplay). Identifique a taxa em bps necessária (usando o ffmpeg -i videoA.mp4 e/ou o próprio wireshark), o encapsulamento usado e o número total de fluxos gerados. Comente a escalabilidade da solução. Ilustre com evidências da realização prática do exercício (ex: capturas de ecrã).

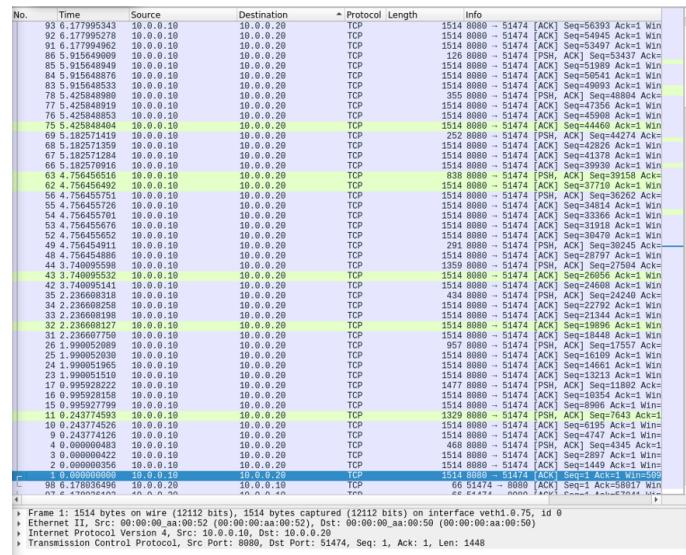


Figura 3: Amostra de tráfego do Wireshark no link de saída do servidor com 1 cliente (VLC)

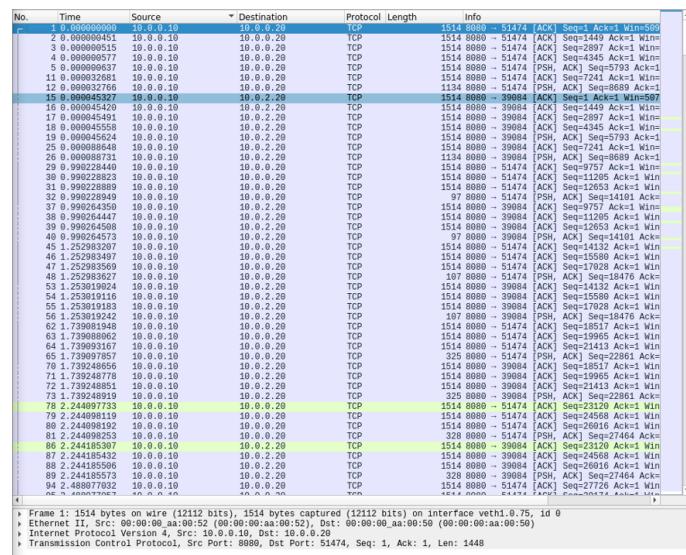


Figura 4: Amostra de tráfego do Wireshark no link de saída do servidor com 2 clientes (VLC e Firefox)

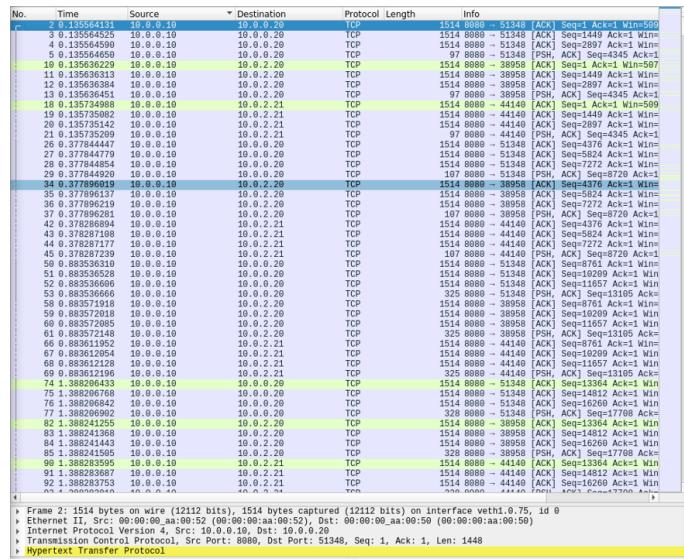


Figura 5: Amostra de tráfego do Wireshark no link de saída do servidor com 3 clientes (VLC, Firefox e ffplay)

Para obter a taxa em bps necessária para realizar o streaming do vídeo, foi utilizado o ffmpeg.

```

libavutil      56. 31.100 / 56. 31.100
libavcodec     58. 54.100 / 58. 54.100
libavformat    58. 29.100 / 58. 29.100
libavdevice    58.  8.100 / 58.  8.100
libavfilter     7. 57.100 / 7. 57.100
libavresample   4.  0.  0 / 4.  0.  0
libswscale       5. 5.100 / 5. 5.100
libswresample   3. 5.100 / 3. 5.100
libpostproc     55. 5.100 / 55. 5.100
Input #0, mov,mp4,m4a,3gp,3g2,mj2, from 'videoA.mp4':
Metadata:
  major_brand   : isom
  minor_version : 512
  compatible_brands: isomiso2avc1mp41
  encoder       : Lavf58.29.100
Duration: 00:00:15.25, start: 0.000000, bitrate: 14 kb/s
Stream #0:0(und): Video: h264 (High) (avc1 / 0x31637661), yuv420p, 160x100, 12 kb/s, 20 fps, 20 tbr, 10240 tbn, 40 tbc
(default)
Metadata:
  handler_name  : VideoHandler
At least one output file must be specified
core@xubuncore:~$ 

```

Figura 6: Output do comando ffmpeg -i videoA.mp4

Como é possível ver pela figura 6, a taxa de bps necessária é de 14 kb/s.

Através das capturas do Wireshark é visível no campo *Protocol* que os pacotes estão encapsulados no protocolo TCP/IP.

Quanto aos fluxos, estes dependem do número de clientes. Cada cliente irá corresponder a um fluxo, sendo que um fluxo é constituído por um ip de origem, um ip de destino, uma porta de origem, uma porta de destino e um protocolo. Na captura Wireshark da figura 3, é visível que existe apenas um fluxo, pois os pacotes viajam sempre da mesma origem (VStreamer 10.0.0.10) para o mesmo destino (Jasmine 10.0.0.20). Com dois clientes existem dois fluxos: do servidor VStreamer (10.0.0.10) para o cliente Jasmine (10.0.0.20) e do servidor VStreamer (10.0.0.10) para o cliente Bela (10.0.2.20), como se pode ver na figura 4. Quando estão três clientes ligados ao servidor existem três fluxos. O cliente Jasmine (10.0.0.20), que utiliza o VLC, o cliente Bela (10.0.2.20), que utiliza o Firefox, e o cliente Monstro (10.0.2.21), que utiliza o ffplay, estão todos individualmente ligados ao servidor VStreamer (10.0.0.10) a receber o vídeo que este está a fazer streaming.

Como para cada cliente existe um fluxo novo, a escalabilidade desta solução não irá ser a melhor, pois com um número gigante de clientes irá significar um número gigante de fluxos, o que irá originar uma sobrecarga de fluxos para o servidor, pois este tem de estar em contacto com todos os clientes para partilhar o vídeo.

Etapa 2

Inicialmente, esta etapa passa por pegar num vídeo em formato *.mp4* e gerar pelo menos três variantes do vídeo, com maior ou menor resolução, representadas na figura 7, respectivamente.

Figura 7: Compressões do vídeo realizadas

De seguida, foi necessário preparar uma página *HTML5* para visualizar o vídeo. O grupo desenvolveu o ficheiro dash.html (figura 8) que permite atribuir a melhor resolução de vídeo dada a largura de banda do link atribuído.

```
core@xubuncore:~$ cat video dash.html
<!DOCTYPE html>
<html>
  <head>
    <title> Streaming ESR - Etapa 2 </title>
    <script src="dash.all.debug.js"></script>
  </head>
  <body>
    <h1> Streaming ESR: Etapa 2 DASH </h1>
    <video data-dashjs-player autoplay src="video_manifest.mpd" controls="true">
    </video>
  </body>
</html>
core@xubuncore:~$
```

Figura 8: Ficheiro html

A pedido do passo 11 foi capturada uma amostra do tráfego com o Wireshark.

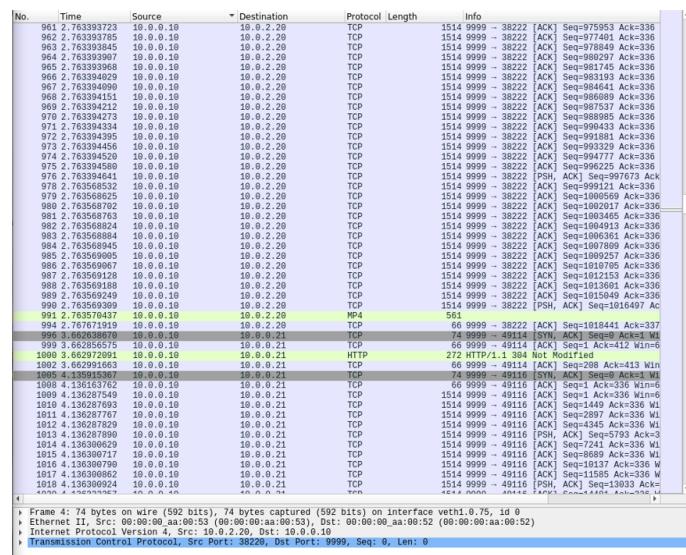


Figura 9: Captura Wireshark do passo 11

Questão 2: Diga qual a largura de banda necessária, em bits por segundo, para que o cliente de streaming consiga receber o vídeo no Firefox e qual a pilha protocolar usada neste cenário.

A largura de banda necessária para que o cliente de streaming receba o vídeo no Firefox é de 256 kb/s, uma vez que este consegue obter o vídeo de menor resolução, apesar das tentativas falhadas de carregar os vídeos com maior resolução, devendo-se ao facto de os vídeos de maior resolução terem um *bit rate* superior à largura de banda disponível. Isto pode ser comprovado, comparando o *bit rate* do vídeo B com diferentes resoluções (após as diversas conversões) face à largura de banda do link. Na figura 10 é possível confirmar o supracitado. A pilha protocolar utilizada neste cenário é TCP/IP, algo que é verificado através das capturas do Wireshark.

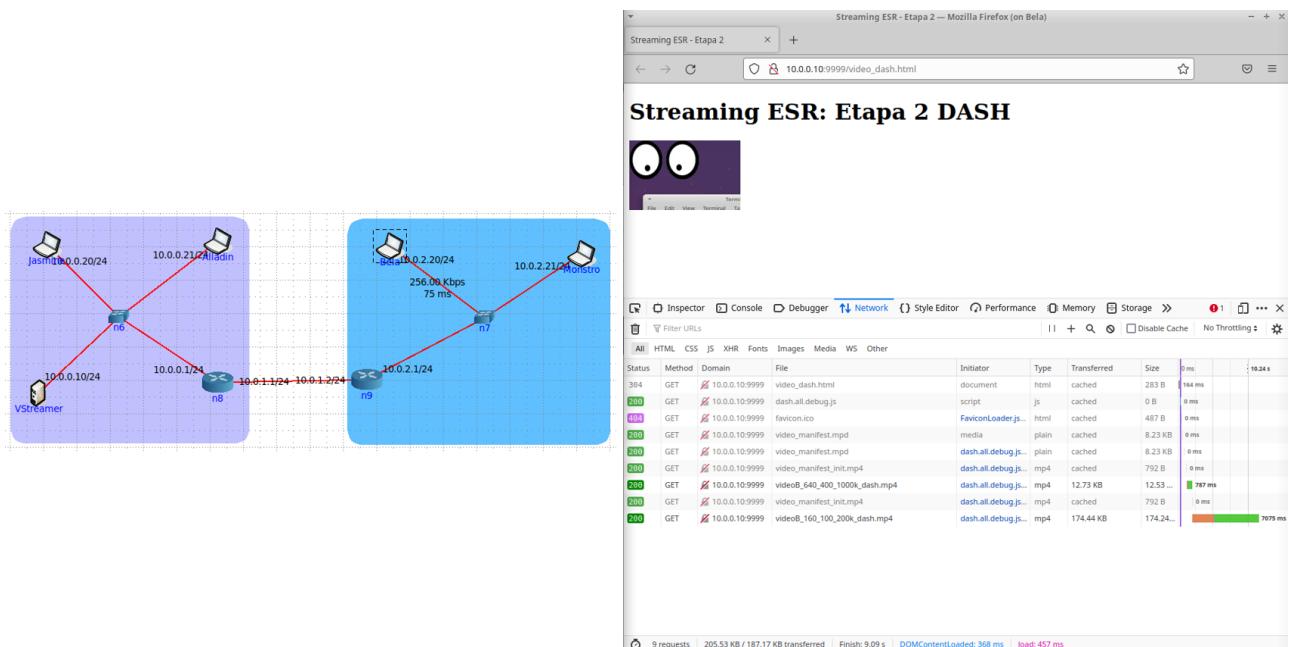


Figura 10: Monstro com largura de banda limitada a 256 kb/s

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.

Primeiramente, o grupo limitou a largura de banda do link de comunicação do cliente Bela a 64 kb/s. Neste caso, não é possível obter nenhuma resolução de vídeo uma vez que a largura de banda não é suficiente para carregar qualquer resolução, como pode ser visto na figura 11.

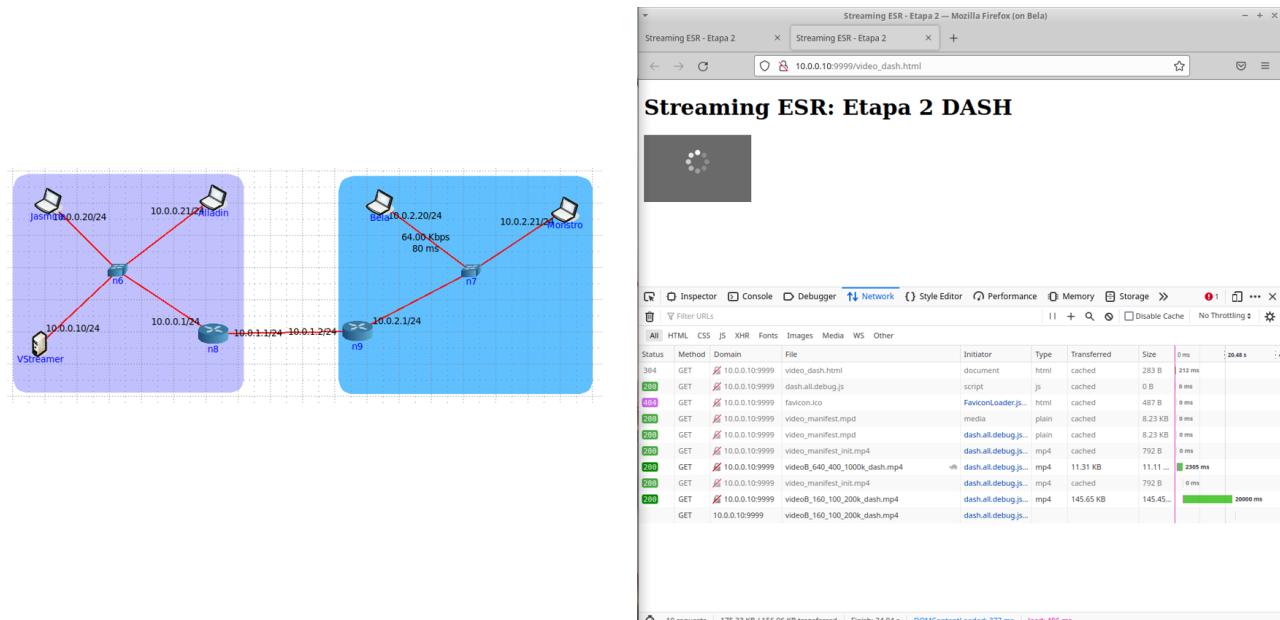


Figura 11: Bela com largura de banda limitada a 64 kb/s

Assim, foi testada uma largura de banda do link de comunicação do cliente Bela a 256 kb/s e concluímos, após vários testes, que neste portátil o vídeo que é atribuído nem sempre é o de pior resolução, uma vez que dependendo do tráfego na rede, é possível este carregar o vídeo com a resolução média, apesar de na maior parte dos testes ser destacado o vídeo com pior resolução.

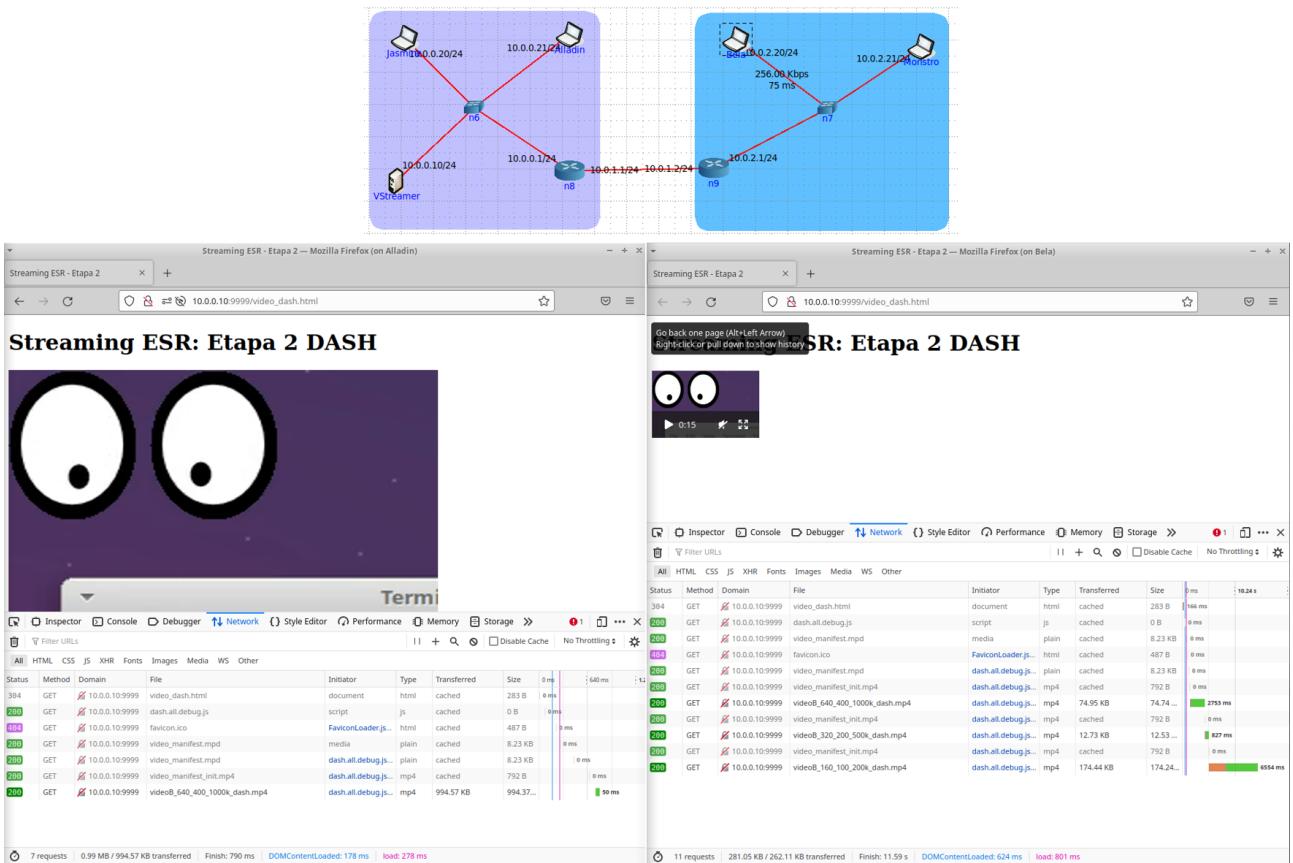


Figura 12: Alladin com largura de banda ilimitada e Bela com largura de banda limitada a 256 kb/s

Em relação ao cliente Alladin não foi definida nenhuma limitação da largura de banda, uma vez que neste caso carrega sempre o vídeo com a melhor resolução.

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

DASH (Dynamic Adaptive Streaming over HTTP), tem como objetivo a adaptação da taxa de bits de um serviço de streaming disponibilizado por um servidor HTTP. Isto é, dada a taxa de bits disponível no canal de transmissão de streaming permite sem intervenção humana haver uma adaptação à mesma, em que é entregue o serviço streaming com a qualidade do mesmo automaticamente ajustada. Este processo foi de facto comprovado ao longo desta etapa, na medida em que ao ajustar a largura de banda dos links, o DASH encarregou-se de atribuir o vídeo B de melhor resolução face à largura de banda. No que concerne ao ficheiro MPD, este é responsável pelo bom funcionamento do DASH, uma vez que tem as diretrizes necessárias para que o DASH consiga proceder à seleção do vídeo cuja resolução se adapta melhor face à largura de banda, tais como informações sobre os fluxos que o servidor utiliza e as larguras de banda associadas.

Etapa 3

Por fim, nesta última etapa o streaming será sempre realizado sobre UDP, quer em unicast quer em multicast. As diferenças entre estes dois tipos de comunicação é abordada na questão 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.

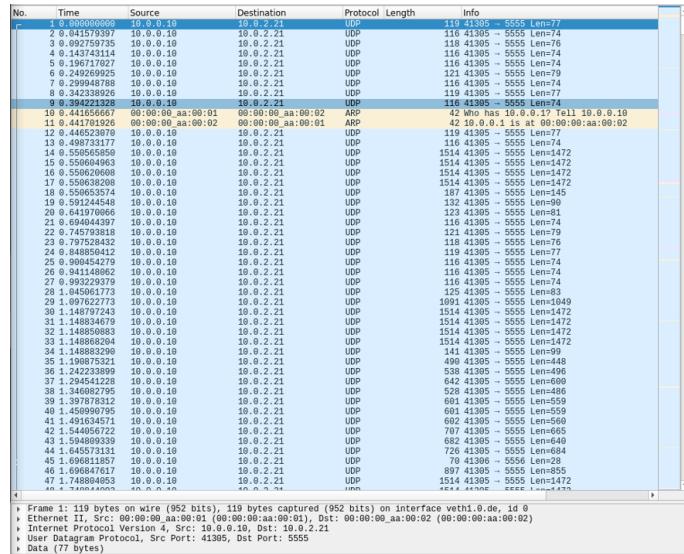


Figura 13: Captura Wireshark do cenário unicast

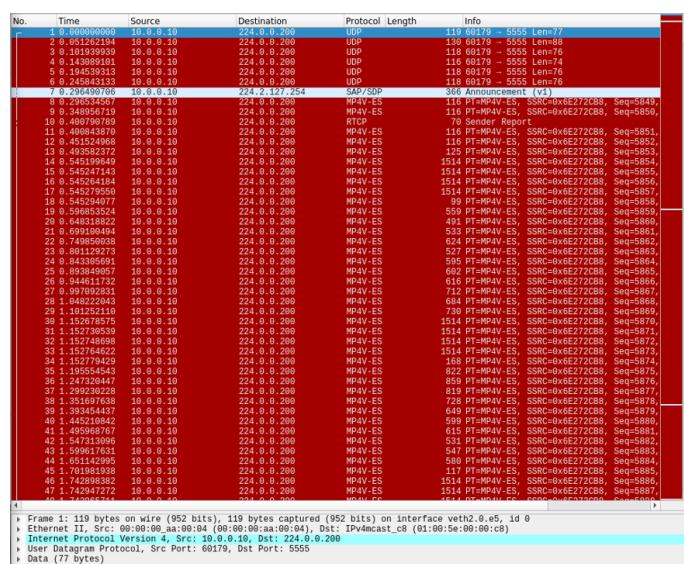


Figura 14: Captura Wireshark do cenário multicast

Em unicast o servidor envia o vídeo diretamente para cada cliente. Neste caso, apenas envia para o cliente Monstro (10.0.2.21) que se encontra noutra rede. Já em multicast o servidor

envia um descrição da sessão para um grupo multicast (224.0.0.200) e de seguida envia o vídeo para esse mesmo endereço, sendo que os switches estão encarregues de reencaminhar estas tramas para os clientes.

Para um maior número de clientes, dentro da mesma rede, o multicast é mais escalável que o unicast, pois apenas necessita de um switch que vai encaminhar as tramas para os vários clientes da rede, ultrapassando o problema dos fluxos, sendo que os clientes apenas têm de se sincronizar com o endereço da sessão. Com isto, não é feita uma sobrecarga de tráfego na rede sobre o servidor, permitindo assim evitar problemas como a falta de largura de banda ou incapacidade do servidor. Em unicast, se existir um número grande de clientes, o servidor terá de enviar o vídeo individualmente para cada um desses mesmos clientes, podendo haver problemas nomeadamente no tráfego da rede, já que mais fluxos sobrecarregam o servidor podendo a largura de banda não ser suficiente para a transmissão. Assim, quando existe um número grande de clientes o multicast apresenta um menor impacto na congestão da rede quando comparado com o unicast. Contudo convém mencionar que o multicast de forma a funcionar e ser escalável entre redes é necessário garantir o correto redirecionamento através de routers nas periferias das redes alvo.

Conclusão

A realização deste projeto permitiu consolidar noções acerca de streaming de áudio e vídeo a pedido em tempo real, bem como perceber quais as opções disponíveis a nível da pilha protocolar e as suas diferenças. É de realçar que este trabalho também moldou o conhecimento acerca de formatos multimédia e como tratar do seu empacotamento. Além disso, permitiu cimentar e aplicar os conhecimentos abordados nas aulas teóricas e também o uso de software de gestão e manipulação de redes, destacando o CORE e o Wireshark.

Por fim, como todas as questões foram respondidas de acordo com o enunciado, concluímos que podemos retirar um balanço positivo deste projeto já que foi essencial para o nosso conhecimento pois adquirimos boas bases para trabalhos futuros.