

Universidade do Minho

Departamento de Informática

UC: Engenharia de Serviços em Rede

Trabalho Prático 1

Alunos:

Eduardo Cunha – PG55939

Guilherme Gonçalves – PG57546

José Pacheco – PG55972

PL7-Grupo N.^o 71

Braga, 2024



ÍNDICE

1.	Introdução.....	6
2.	Procedimentos da etapa 1	7
3.	Questões e Respostas da etapa 1	12
	Questão 1: Após capturar 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).....	12
	A. Protocolos utilizados na transferência, bem como a experiência que o utilizador terá caso o link utilizado tenha perdas.....	13
	B. Identifique o número total de fluxo gerados e elabore um gráfico que demonstre a evolução do débito dependendo do número de clientes	14
	C. Comente a escalabilidade da solução para 1000 utilizadores, assim como 10000 utilizadores.....	15
4.	Procedimentos da etapa 2.....	16
5.	Questões e Respostas da etapa 2	24
	Questão 2: Utilize o wireshark para determinar 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. Explique como obteve esta informação.....	24
	Questão 3: Compare a largura de banda medida na questão anterior com a que é disponibilizada pelo ffplay. Qual é a razão para a diferença entre as duas?	25
	Questão 4: 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 e justifique a largura de banda necessária para que o stream de vídeo sofra alterações.....	27
	Questão 5: Descreva o funcionamento do DASH neste caso concreto, referindo o papel do ficheiro MPD criado e comparando o modelo de streaming com o que foi utilizado na Questão 1	28
6.	Procedimentos da Etapa 3.....	29
7.	Questões e Respostas da etapa 3	34



Questão 6: 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 também para os cenários de 1000 e 10000 clientes.	34
8. Conclusão.....	36



ÍNDICE DE IMAGEM

Figura 1 - Captura de vídeo	7
Figura 2 – Construção e teste conectividade	7
Figura 3 – Traceroute.....	8
Figura 4 - Ping	8
Figura 5 – Upload do vídeo	8
Figura 6 - Inicialização do VLC, VSTREAMER.....	9
Figura 7 - Jasmine a assistir a stream.....	10
Figura 8 - Amostra de tráfego com o Wireshark na interface de saída do servidor..	10
Figura 9 – Streaming Firefox	11
Figura 10 - Amostra de tráfego	11
Figura 11 – Configuração do DISPLAY no Monstro	11
Figura 12 - Taxa em bps necessária para transmitir o video1.mp4.....	12
Figura 13 - 3 clientes (vlc + firefox + ffplay monstro):.....	12
Figura 14 - 2 clientes (vlc + firefox bela)	12
Figura 15 - 1 cliente (vlc jasmine):.....	13
Figura 16 - Fluxo dos 3 clientes	14
Figura 17 - Gráfico exemplo	14
Figura 18 - Primeira versão do videoB.mp4	16
Figura 19 - Segunda versão do videoB.mp4	16
Figura 20 - Terceira versão do videoB.mp4.....	17
Figura 21 - Produção do ficheiro MPD	17
Figura 22 - Ficheiro MPD com a descrição das alternativas.....	18
Figura 23 - Preparação da página HTML5 video_dash.html para visualizar o vídeo	18
Figura 24 – Ficheiro HTML	19
Figura 25 - Testar da conectividade no Core	19
Figura 26 - Serviço do conteúdo com um servidor HTTP.....	20
Figura 27 - Firefox no portátil Bela.....	20
Figura 28 - Firefox no portátil Alladin.....	21
Figura 29 - Captura de amostras de tráfego com Wireshark.....	21
Figura 30 - Alteração da capacidade dos links no portátil Bela	22
Figura 31 - Demonstração da alteração dos links	22
Figura 32 – Conexão Aladdin e Bela (com limitação de rede)	23



Figura 33 - Analise do ficheiro video_manifest.mpd	24
Figura 34 - Computador Jasmine	24
Figura 35 - Analise da largura de banda usando Wireshark	24
Figura 36 - Pilha protocolar	25
Figura 37 - Transmissão do "vídeoB"	25
Figura 38 - Valores Obtidos	26
Figura 39 - Débitos no wireshark	27
Figura 41 - Ping na Jasmine	29
Figura 42 - Inicio uma sessão de streaming com RTP com ffmpeg	29
Figura 43 - Inicio de um cliente ffplay no Monstro:	30
Figura 44 - Captuae o tráfego com o Wireshark no link de saída do servidor	31
Figura 45 - Nova topologia	31
Figura 46 - Teste a conectividade entre os sistemas	32
Figura 47 - Inicio de diferentes sessões	33
Figura 48 - Captura do tráfego no link de saída do servidor com o Wireshark	33
Figura 49 - Cenário Unicast	34
Figura 50 - Cenário Multicast	34



1. INTRODUÇÃO

Este trabalho foi desenvolvido no âmbito da Unidade Curricular de Engenharia de Serviços em Rede, com o objetivo de explorar diferentes soluções de streaming de vídeo.

Através do emulador CORE, pretende-se analisar os diversos cenários e protocolos aplicados na transmissão de multimédia, bem como compreender como ocorre a adaptação da taxa de transmissão em condições de rede variáveis.

Para tal, o trabalho foi dividido em três etapas distintas. Na primeira fase, será estudada a implementação prática de streaming em HTTP, sem adaptação de débito. A segunda fase abordará técnicas mais avançadas de streaming adaptativo, como o Dynamic Adaptive Streaming over HTTP (DASH), focando em como as mudanças nas condições de rede afetam a qualidade da transmissão. Por fim, na terceira fase, será explorado o streaming sobre UDP, tanto em cenários de unicast como de multicast. Para suportar as análises e experimentos, serão utilizadas ferramentas open-source amplamente reconhecidas no setor, como FFmpeg, VLC e Wireshark.

2. PROCEDIMENTOS DA ETAPA 1

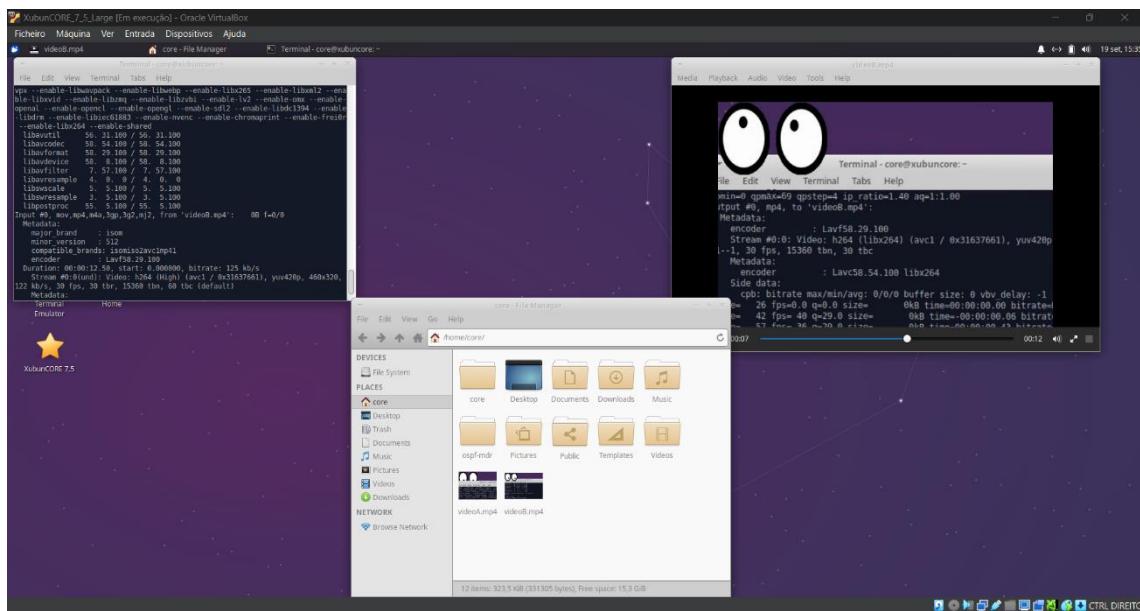


Figura 1 - Captura de vídeo

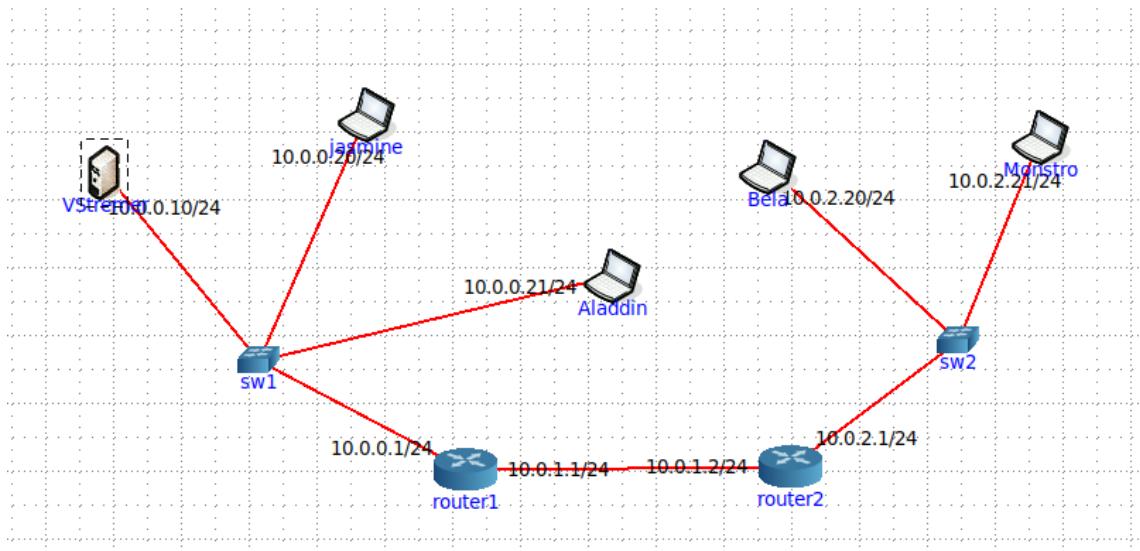


Figura 2 – Construção e teste conectividade

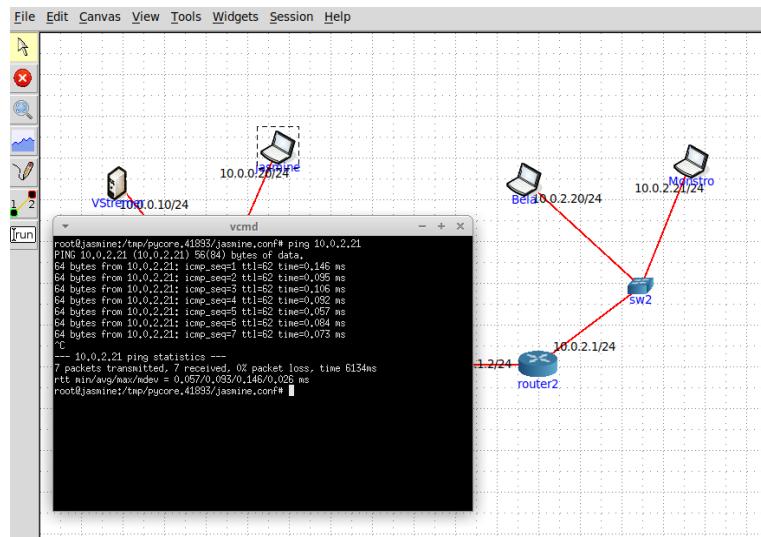


Figura 4 - Ping

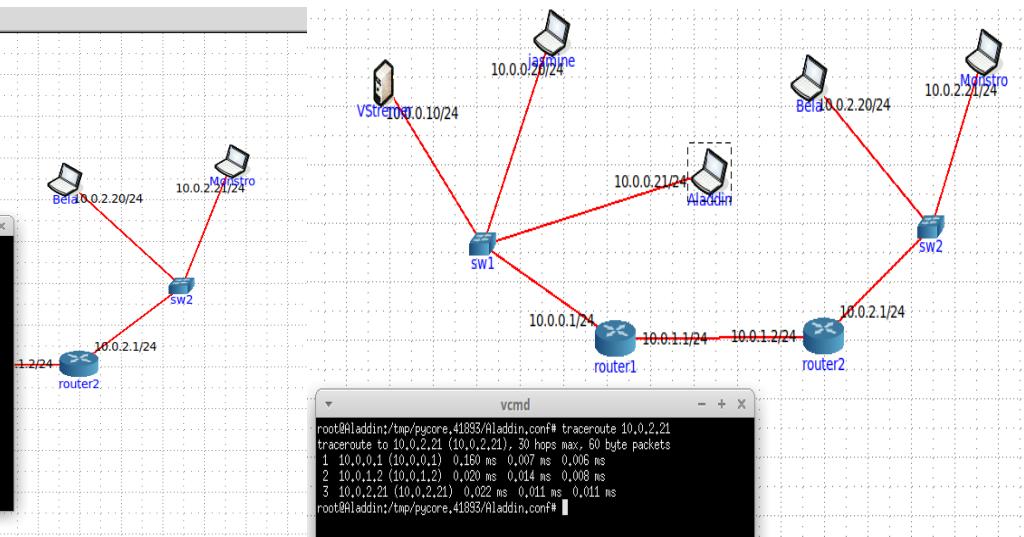


Figura 3 – Traceroute

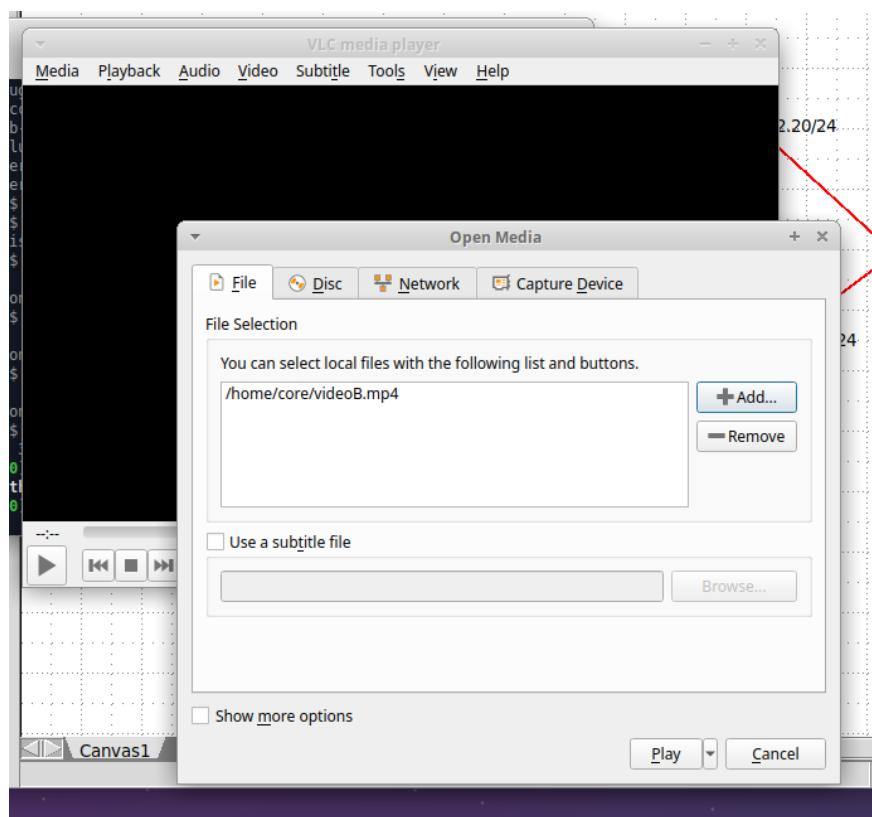


Figura 5 – Upload do vídeo

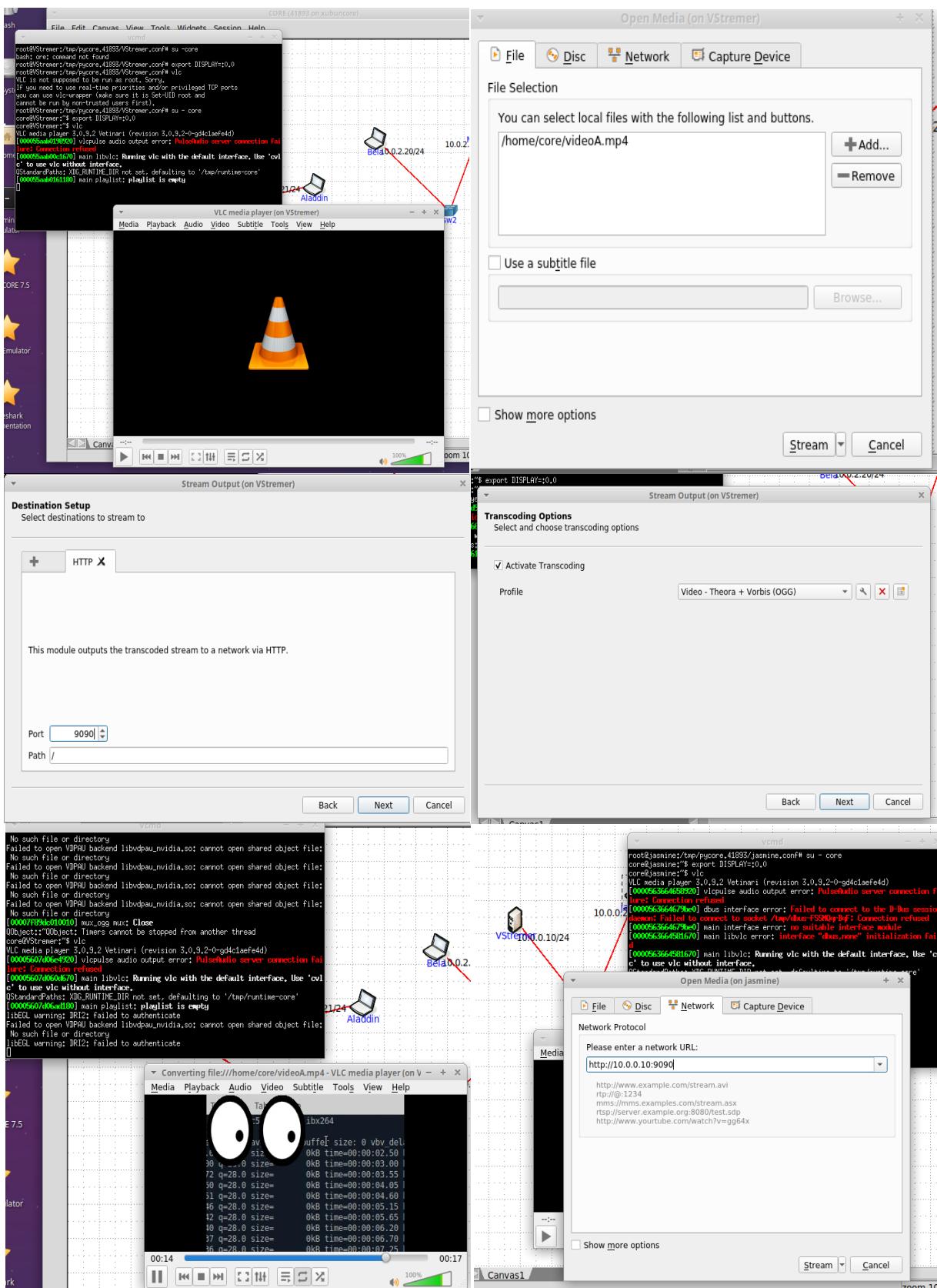


Figura 6 - Inicialização do VLC, VSTREAMER

*Alteração para a porta 9090 em vez de 8080.

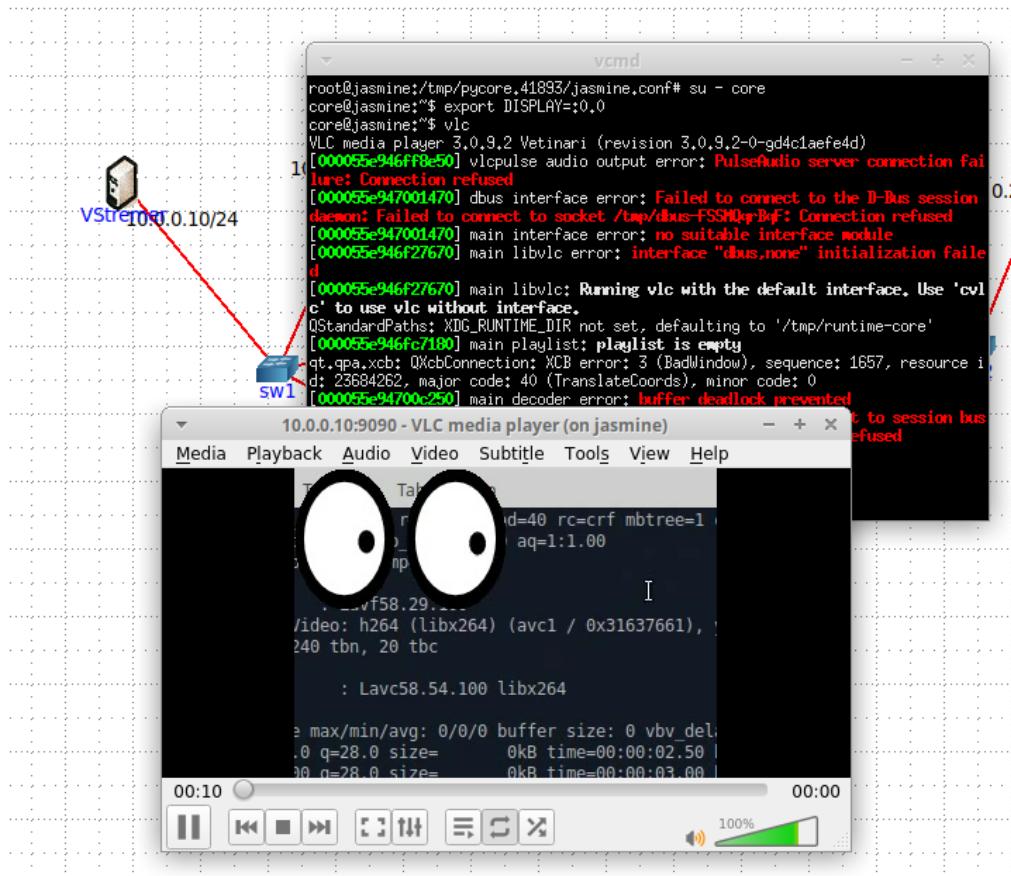


Figura 7 - Jasmine a assistir a stream

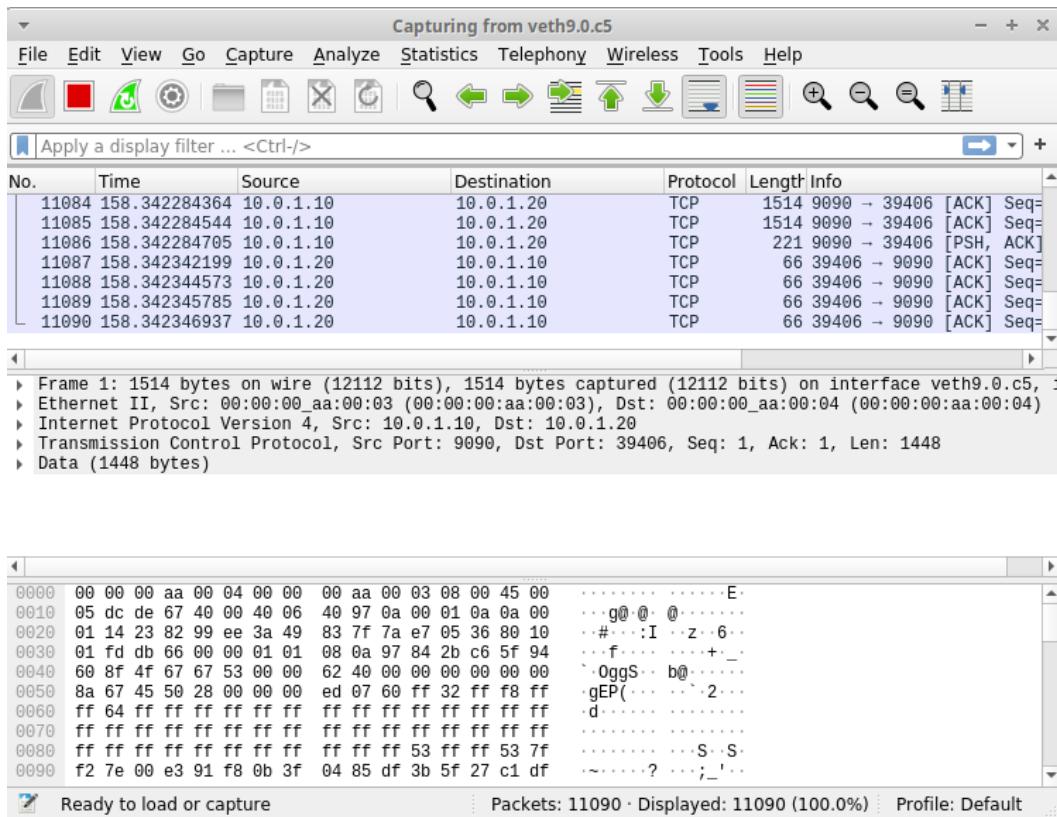
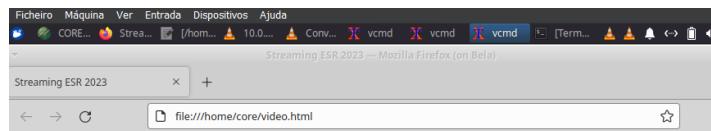


Figura 8 - Amostra de tráfego com o Wireshark na interface de saída do servidor



Streaming ESR: Etapa 1

```
Terminal Tabs Help
12 q=28.0 size= 0KB time=0:00:03.55
10 q=28.0 size= 0KB time=0:00:04.05
14 q=28.0 size= 0KB time=0:00:04.60
16 q=28.0 size= 0KB time=0:00:05.15
12 q=28.0 size= 0KB time=0:00:05.65
10 q=28.0 size= 0KB time=0:00:06.20
27 q=28.0 size= 0KB time=0:00:06.70
26 q=28.0 size= 0KB time=0:00:07.25
44 q=28.0 size= 0KB time=0:00:07.80
13 q=28.0 size= 0KB time=0:00:08.35
20 q=28.0 size= 0KB time=0:00:08.90
11 q=28.0 size= 0KB time=0:00:09.45
10 q=28.0 size= 0KB time=0:00:10.00

[...]
root@Bela:/tmp/pycore_41893/Bela.conf# su - core
core@Bela:~$ export DISPLAY=:0.0
core@Bela:~$ cat video.html
<!DOCTYPE HTML>
<html>
  <head>
    <title> Streaming ESR 2023 </title>
  </head>
  <body>
    <h1> Streaming ESR: Etapa 1 </h1>
    <video controls autoplay>
      <source href="http://10.0.0.10:9090">
    A tag VIDEO não é suportada
    </video>
  </body>
</html>
core@Bela:~$ firefox video.html
[GFX1--]; Unrecognized feature VIDEO_OVERLAY
core@Bela:~$ firefox video.html
[GFX1--]; Unrecognized feature VIDEO_OVERLAY
```

Figura 9 – Streaming Firefox

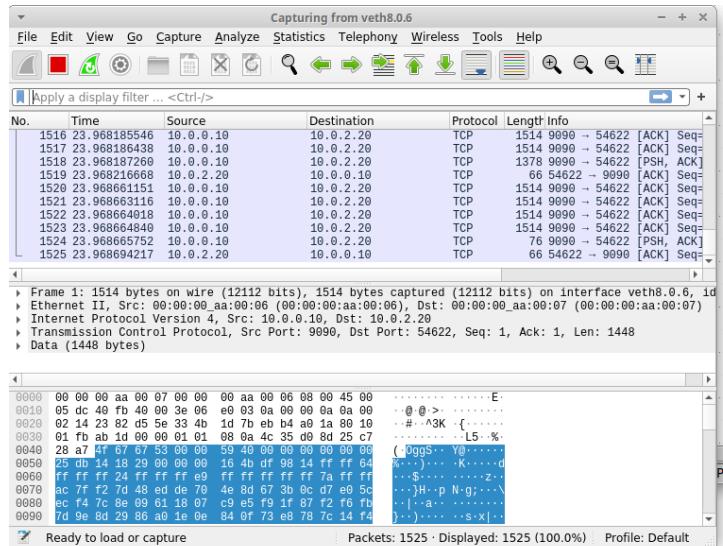


Figura 10 - Amostra de tráfego

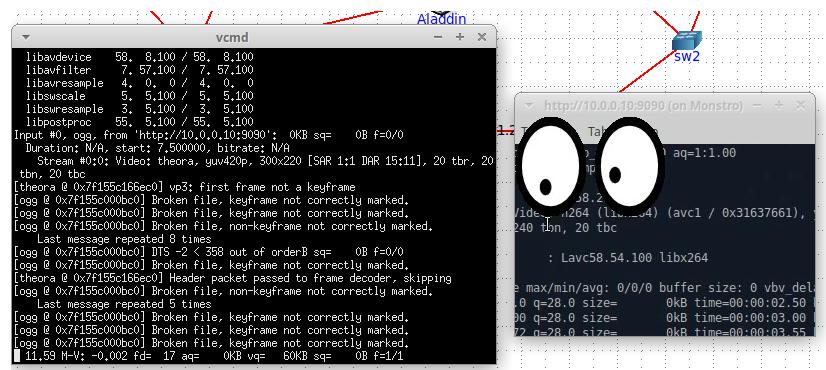


Figura 11 – Configuração do DISPLAY no Monstro

3. QUESTÕES E RESPOSTAS DA ETAPA 1

QUESTÃO 1: APÓS CAPTURAR 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).

```
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: 00:00:17.95, start: 0.000000, bitrate: 69 kb/s
  Stream #0:0(und): Video: h264 (High) (avc1 / 0x31637661), yuv420p, 300x220,
  66 kb/s, 20 fps, 20 tbr, 10240 tbn, 40 tbc (default)
  Metadata:
    handler_name : VideoHandler
```

Figura 12 - Taxa em bps necessária para transmitir o video1.mp4

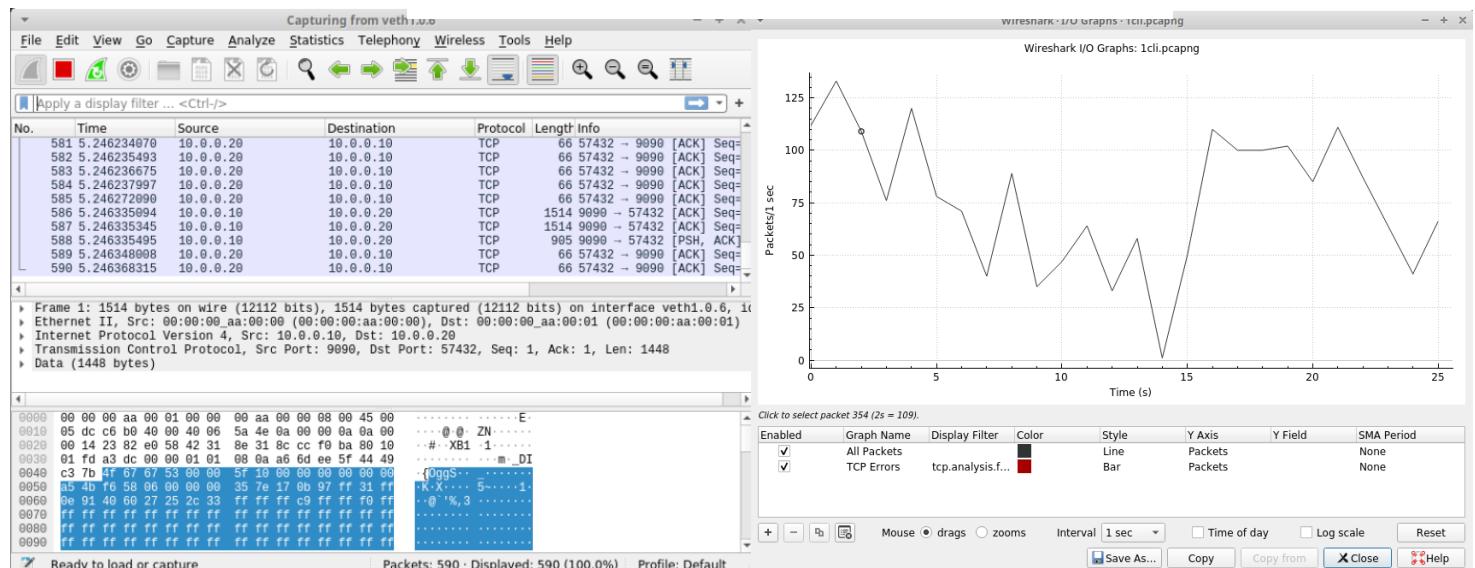


Figura 13 - 3 clientes (vlc + firefox + ffplay monstro):

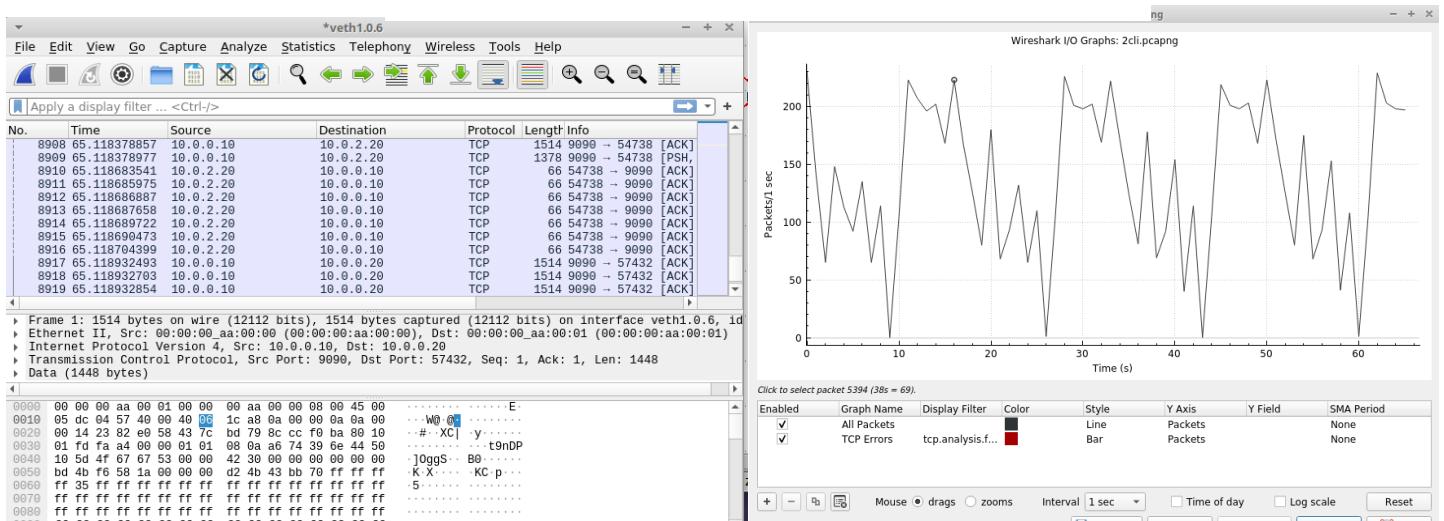


Figura 14 - 2 clientes (vlc + firefox bela)

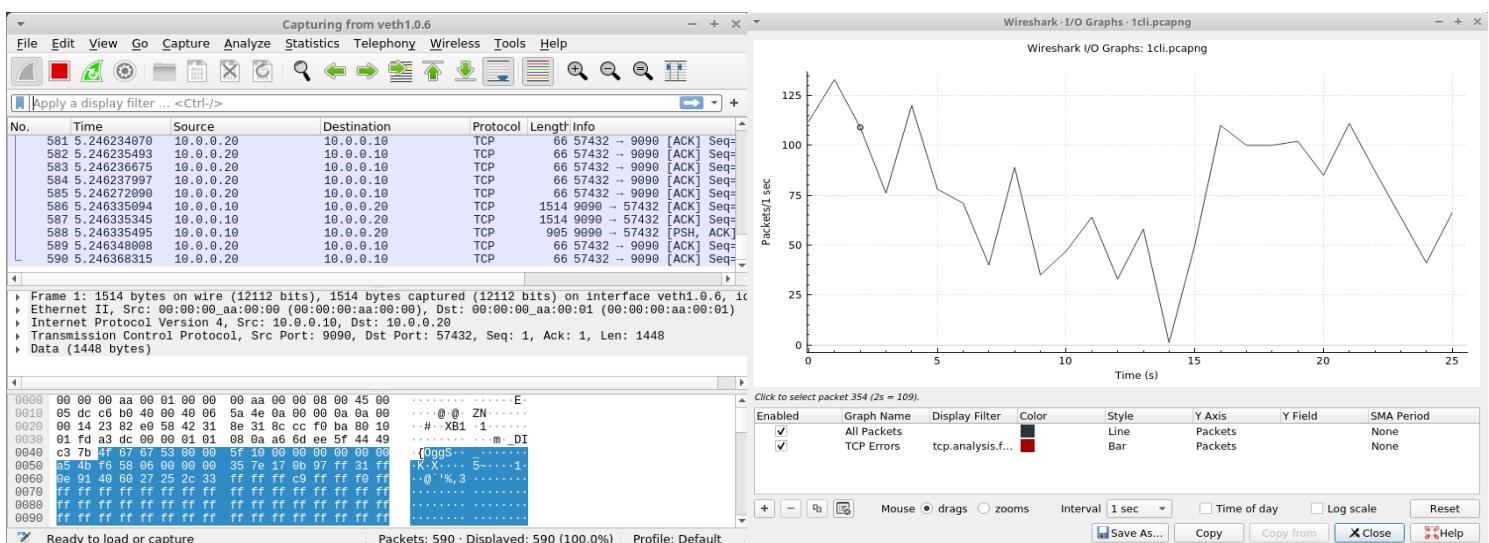


Figura 15 - 1 cliente (vlc jasmine):

Através da análise do comando "ffmpeg -i videoA.mp4", determinamos que o "bitrate" do vídeo é de 69 kbps como está demonstrado na **figura 12**, logo, teoricamente, a taxa em bps necessária para transmitir o vídeo para 3 clientes seria de $69 \times 3 = 207$ kbps = 207000 bps.

A. PROTOCOLOS UTILIZADOS NA TRANSFERÊNCIA, BEM COMO A EXPERIÊNCIA QUE O UTILIZADOR TERÁ CASO O LINK UTILIZADO TENHA PERDAS

Como podemos observar nas **figuras 13, 14 e 15**, o principal protocolo utilizado na transferência de dados é o TCP. O TCP é um protocolo de transporte na Internet que garante a entrega fiável de dados entre dois utilizadores. Este protocolo assegura que o receptor esteja "atento" à conexão através do processo de "three-way handshake". Além disso, o TCP realiza a retransmissão de pacotes que possam ser perdidos, assegurando assim a integridade dos dados. Outra característica importante do TCP é a sequencialização dos dados, garantindo que os pacotes sejam entregues na ordem correta. Posto isto, a ocorrência de perdas de pacotes pode resultar em atrasos ou até mesmo, interrupções na reprodução do vídeo.

B. IDENTIFIQUE O NÚMERO TOTAL DE FLUXO GERADOS E ELABORE UM GRÁFICO QUE DEMONSTRE A EVOLUÇÃO DO DÉBITO DEPENDENDO DO NÚMERO DE CLIENTES

Considerando a captura com 3 clientes, é clara a existência de 3 fluxos distintos, como podemos confirmar na imagem seguinte:

TCP · 3						
Address A	Port A	Address B	Port B	Packets	Bytes	I
10.0.0.20	57432	10.0.0.10	9090	5,616	4620 k	
10.0.2.20	54738	10.0.0.10	9090	5,555	4618 k	
10.0.2.21	36168	10.0.0.10	9090	5,496	4627 k	

Figura 16 - Fluxo dos 3 clientes

Através da análise dos gráficos presentes nas **figuras 13, 14 e 15**, podemos observar facilmente que o **débito aumenta com o número de clientes**. Elaborámos o gráfico seguinte, onde o eixo dos X corresponde ao número de utilizadores e o eixo dos Y ao valor do débito.

Importa salientar que o gráfico serve apenas como uma exemplificação e aproximação, não correspondendo exatamente aos valores reais obtidos.

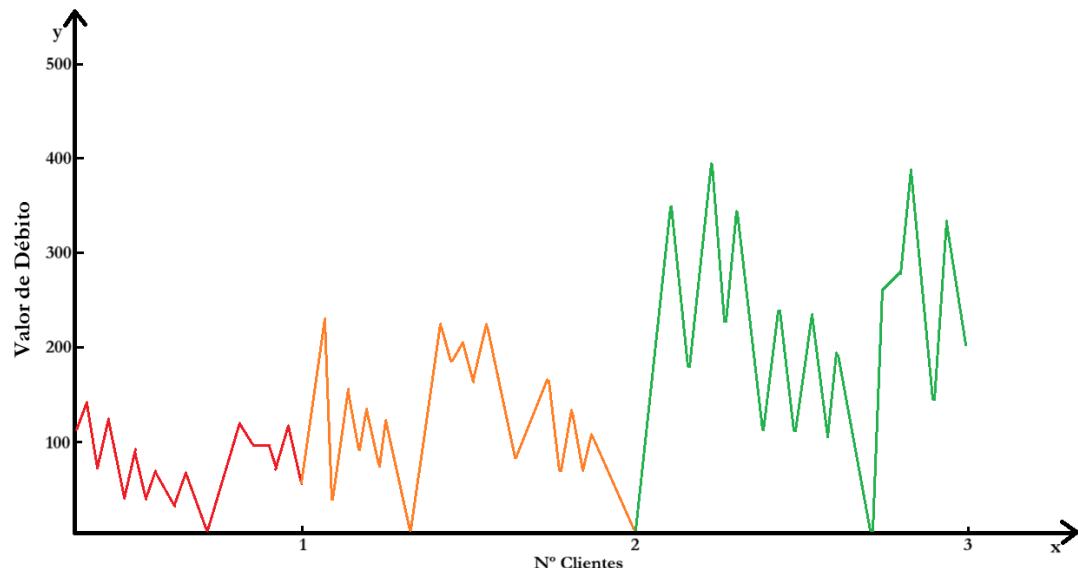


Figura 17 - Gráfico exemplo

C. COMENTE A ESCALABILIDADE DA SOLUÇÃO PARA 1000 UTILIZADORES, ASSIM COMO 10000 UTILIZADORES.

Considerando um débito de 125 kbps por utilizador (valor aproximado com base na observação do gráfico da **figura 15**), que por si só já é muito superior ao bitrate do vídeo, é possível estimar o débito necessário para 1000 e 10 000 utilizadores.

O débito necessário pode ser expresso pela seguinte fórmula:

$$D = N \cdot B$$

Onde:

- **D** é o débito total necessário (em bps);
- **N** é o número de clientes;
- **B** é a taxa de bits média do vídeo (em bps).

Para 1000 utilizadores, o cálculo é o seguinte:

$$D = 125 \text{ kbps} \times 1000 = 125\,000 \text{ kbps}.$$

Para 10 000 utilizadores, o cálculo seria:

$$D = 125 \text{ kbps} \times 10\,000 = 1\,250\,000 \text{ kbps}.$$

Constata-se, portanto, que à medida que o número de utilizadores aumenta, a quantidade de débito do servidor cresce de forma linear (de acordo com a função definida), exigindo uma largura de banda excessivamente elevada. A solução atual revela-se insustentável para cenários com 1000 ou 10 000 utilizadores.

4. PROCEDIMENTOS DA ETAPA 2

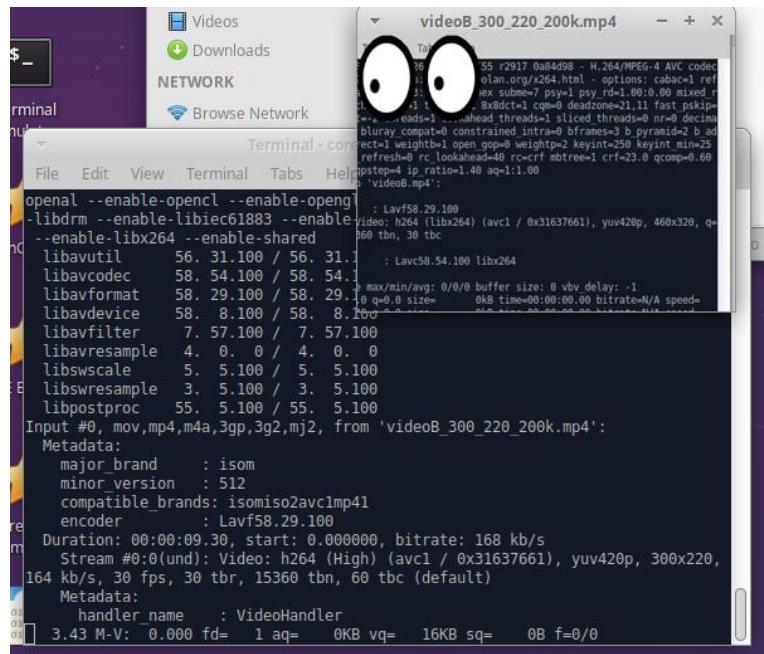


Figura 18 - Primeira versão do videoB.mp4

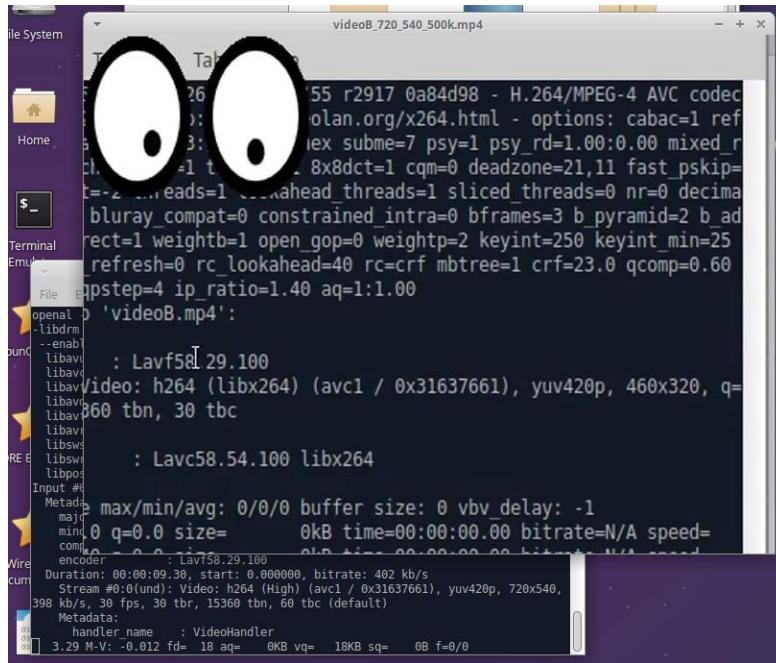


Figura 19 - Segunda versão do videoB.mp4

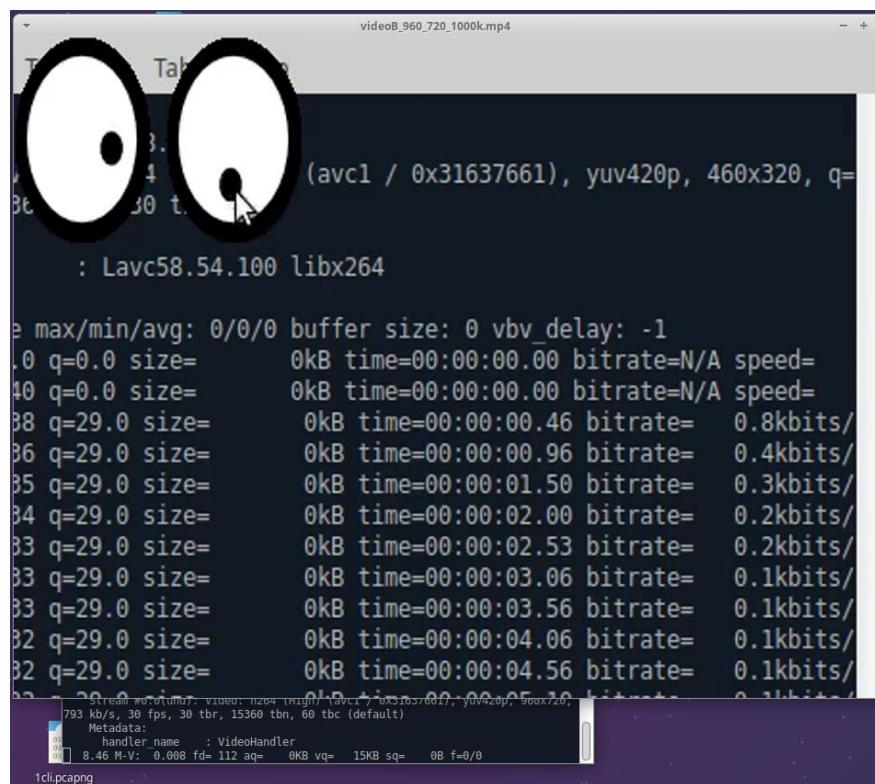


Figura 20 - Terceira versão do videoB.mp4

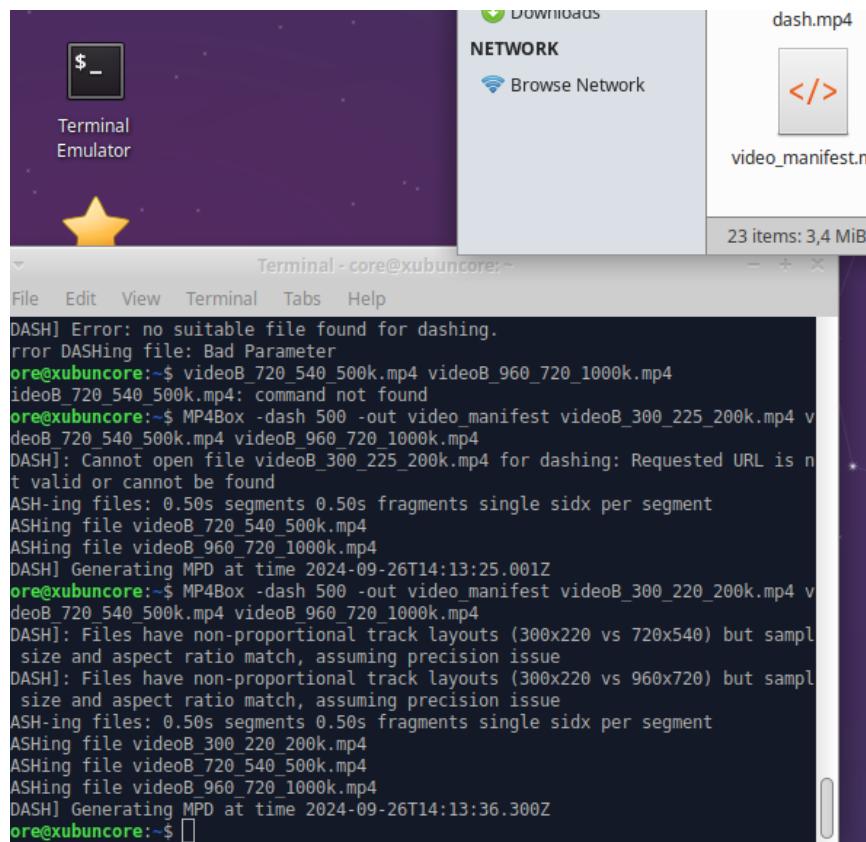


Figura 21 - Produção do ficheiro MPD

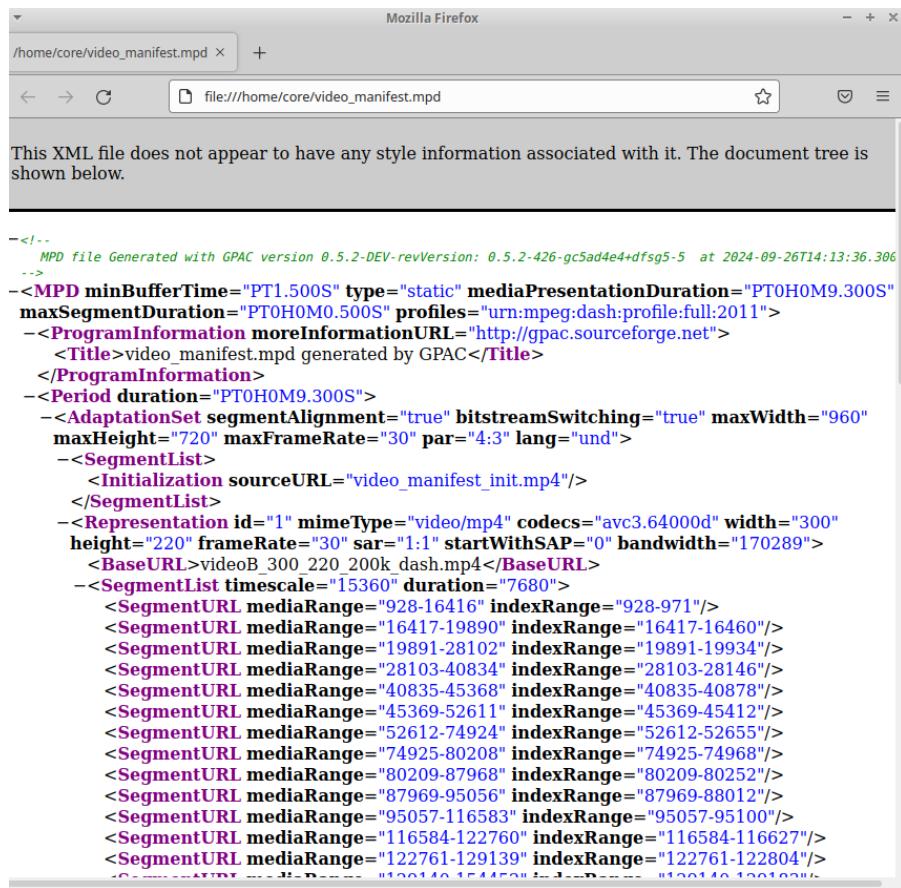


Figura 22 - Ficheiro MPD com a descrição das alternativas

```

File Edit View Terminal Tabs Help
DASHing file videoB_720_540_500k.mp4
DASHing file videoB_960_720_1000k.mp4
[DASH] Generating MPD at time 2024-09-26T14:13:36.300Z
core@xubuncore:~$ wget http://cdn.dashjs.org/latest/dash.all.min.js
--2024-09-26 15:20:13-- http://cdn.dashjs.org/latest/dash.all.min.js
Resolving cdn.dashjs.org (cdn.dashjs.org)... 194.210.238.75, 194.210.238.72, 200
1:690:c01:1::c2d2:ee4b, ...
Connecting to cdn.dashjs.org (cdn.dashjs.org)|194.210.238.75|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/x-javascript]
Saving to: 'dash.all.min.js'

dash.all.min.js      [ => ] 775,25K 3,91MB/s   in 0,2s

2024-09-26 15:20:13 (3,91 MB/s) - 'dash.all.min.js' saved [793860]

core@xubuncore:~$ wget http://cdn.dashjs.org/latest/dash.all.debug.js
--2024-09-26 15:20:33-- http://cdn.dashjs.org/latest/dash.all.debug.js
Resolving cdn.dashjs.org (cdn.dashjs.org)... 194.210.238.72, 194.210.238.75, 200
1:690:c01:1::c2d2:ee48, ...
Connecting to cdn.dashjs.org (cdn.dashjs.org)|194.210.238.72|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/x-javascript]
Saving to: 'dash.all.debug.js'

dash.all.debug.js     [ => ] 2,92M 1,16MB/s   in 2,5s

2024-09-26 15:20:36 (1,16 MB/s) - 'dash.all.debug.js' saved [3062440]
core@xubuncore:~$ 

```

Figura 23 - Preparação da página HTML5 video_dash.html para visualizar o vídeo



```
Terminal - core@xubuncore:~  
File Edit View Terminal Tabs Help  
GNU nano 4.8 video dash.html Modified  
<!DOCTYPE html>  
<html>  
    <head>  
        <title> Streaming ESR 2024 </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>
```

Figura 24 – Ficheiro HTML

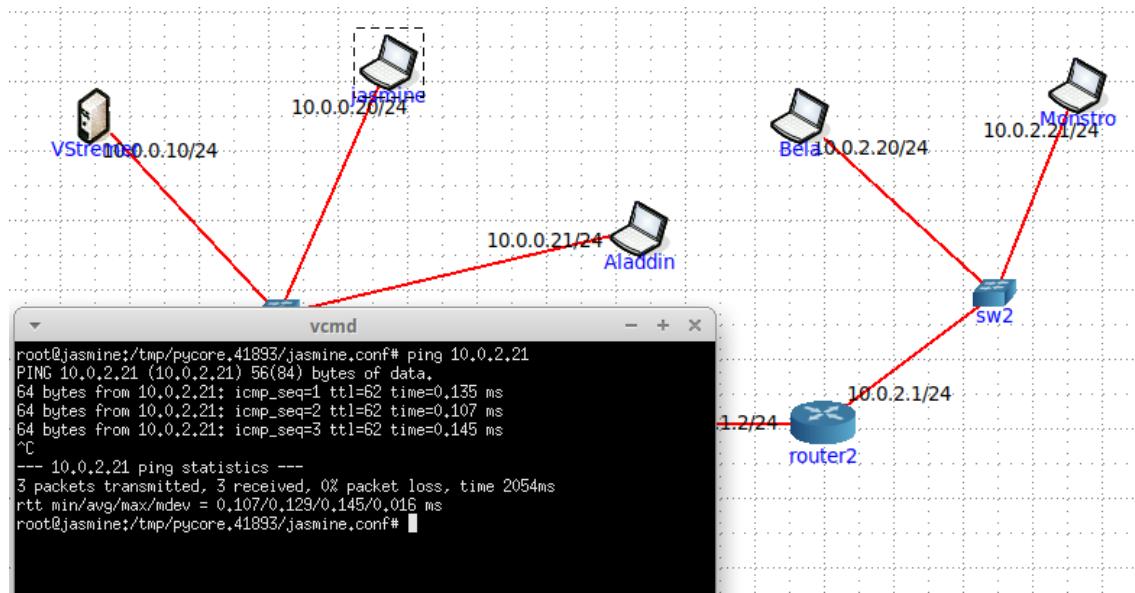


Figura 25 - Testar da conectividade no Core

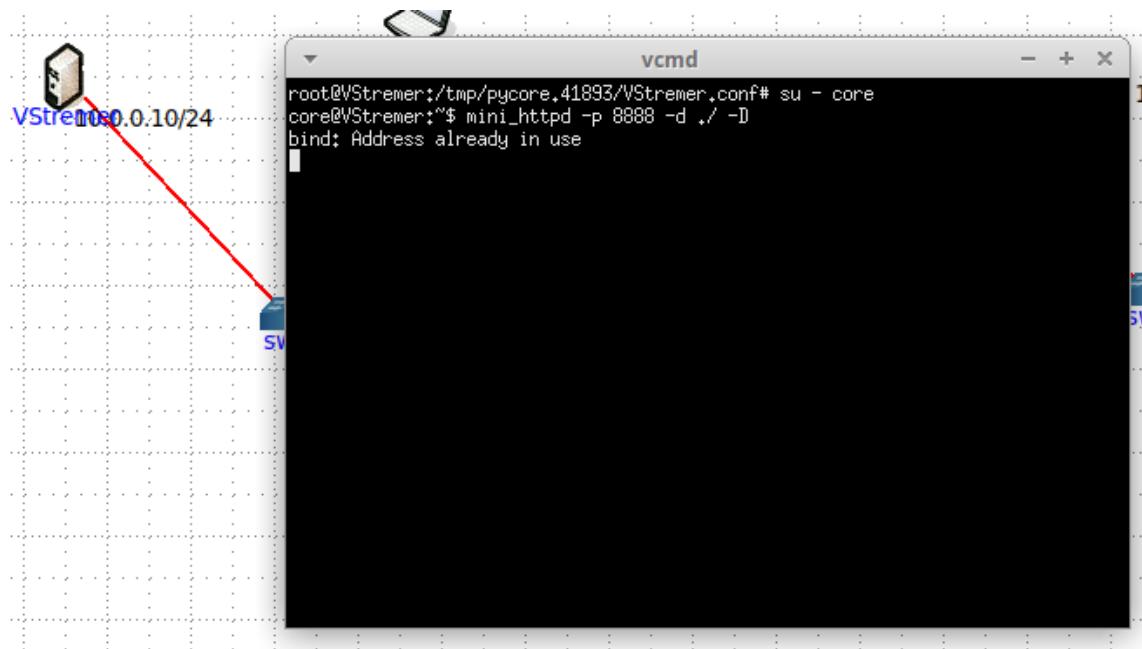


Figura 26 - Serviço do conteúdo com um servidor HTTP

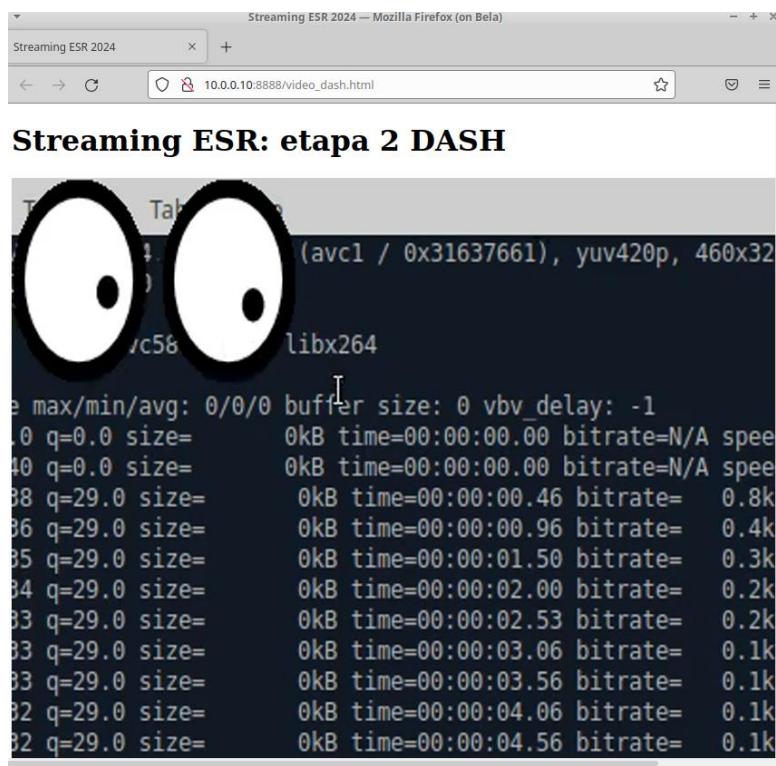


Figura 27 - Firefox no portátil Bela

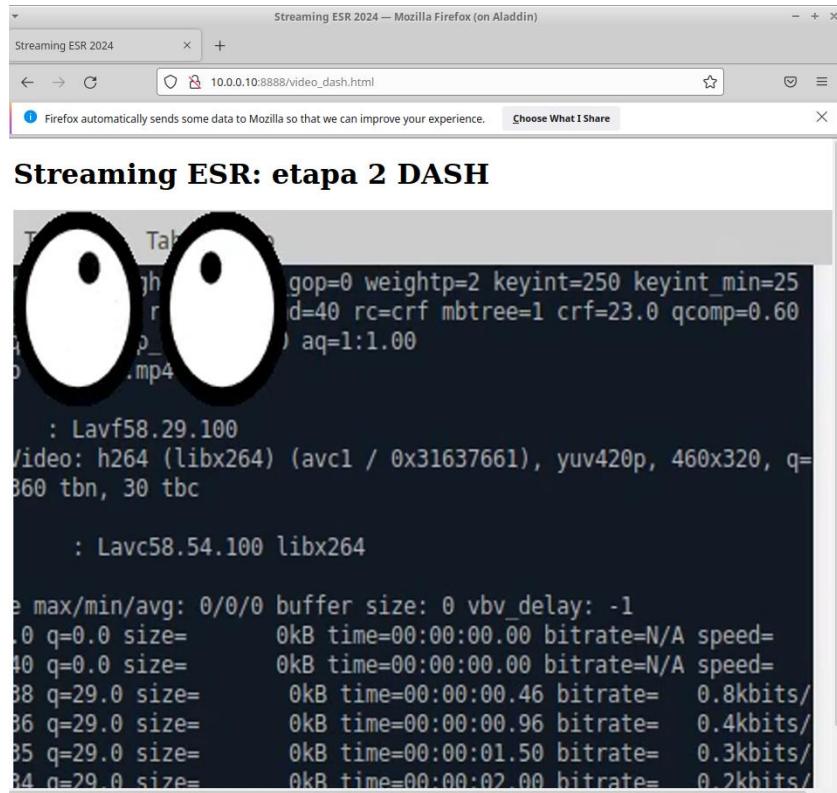


Figura 28 - Firefox no portátil Alladin

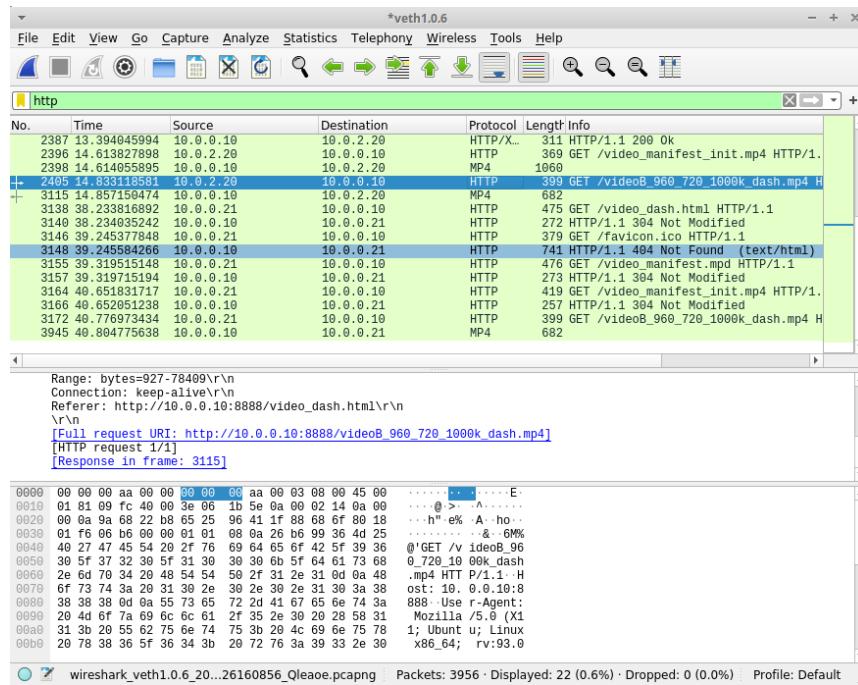


Figura 29 - Captura de amostras de tráfego com Wireshark

Neste passo, após discussão com a professora, verificámos que os nossos valores não correspondiam ao esperado. Na nossa máquina, não foram capturados os pedidos de todas as resoluções do vídeo conforme previsto. A docente sugeriu que prosseguíssemos, mas que investigássemos as diferenças no comportamento da nossa máquina em comparação com as

restantes, uma vez que isso não representava um obstáculo significativo para a compreensão do protocolo.

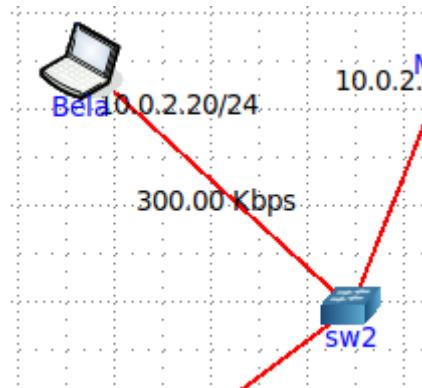


Figura 30 - Alteração da capacidade dos links no portátil Bela

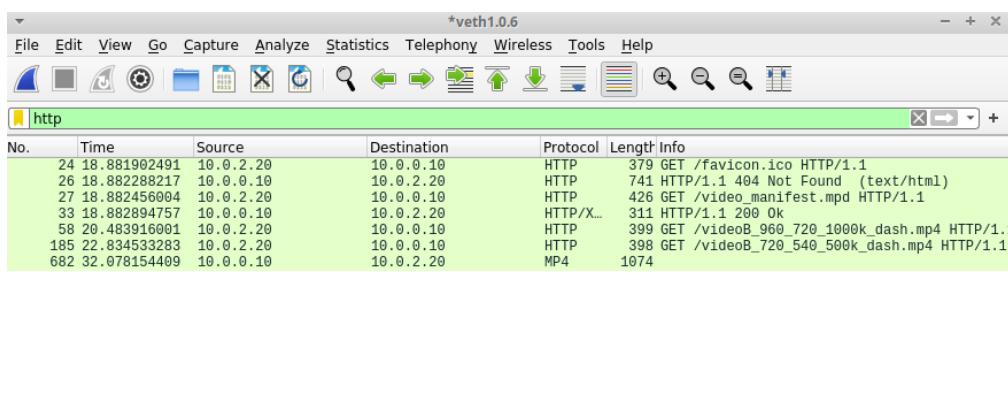


Figura 31 - Demonstração da alteração dos links

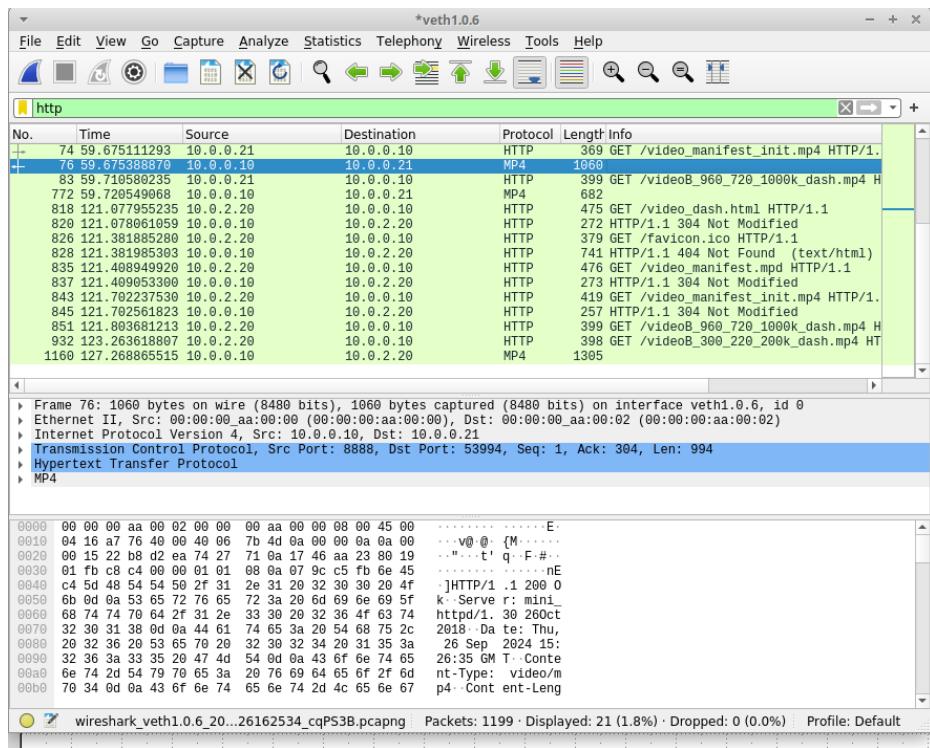


Figura 32 – Conexão Aladdin e Bela (com limitação de rede)

5. QUESTÕES E RESPOSTAS DA ETAPA 2

QUESTÃO 2: UTILIZE O WIRESHARK PARA DETERMINAR 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. EXPLIQUE COMO OBTEVE ESTA INFORMAÇÃO.

Através da análise do ficheiro “vídeo_manifeste.mpd”, conseguimos observar que a largura de banda mínima para o vídeo com melhor resolução é **798777 bps** (figura 33). Através de uma limitação na rede da Jasmine (figura 34) conseguimos produzir no Wireshark resultados relativamente próximos, como é possível observar na figura 35.

-<**Representation id="3" mimeType="video/mp4" codecs="avc3.64001f" width="960" height="720" frameRate="30" sar="1:1" startWithSAP="0" bandwidth="798777"**>

Figura 33 - Analise do ficheiro video_manifeste.mpd

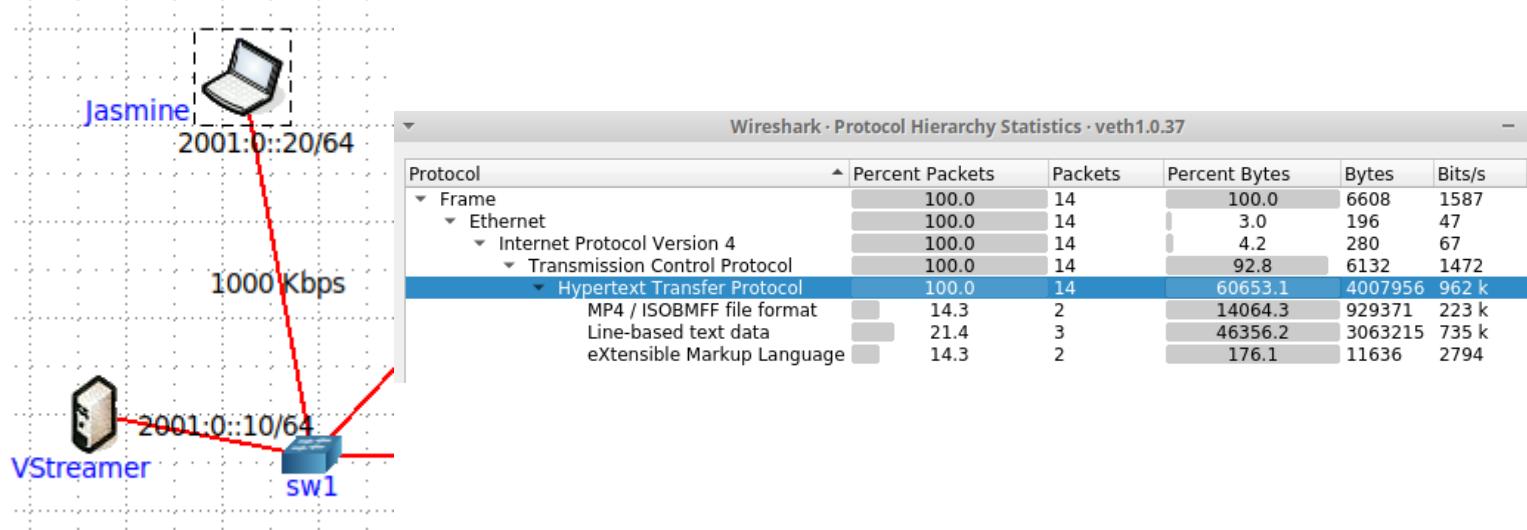


Figura 35 - Analise da largura de banda usando Wireshark

Figura 34 - Computador Jasmine

O valor obtido é superior por vários motivos como: cabeçalhos dos protocolos, overhead, retransmissões, entre outros...

A pilha protocolar usada neste cenário é **Ethernet/IP/TCP/HTTP**, como é possível observar na **figura 36** onde estão listados todos os protocolos presentes num dos pacotes HTTP capturados.

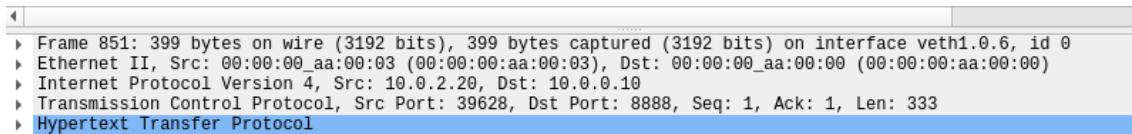


Figura 36 - Pilha protocolar

QUESTÃO 3: COMPARE A LARGURA DE BANDA MEDIDA NA QUESTÃO ANTERIOR COM A QUE É DISPONIBILIZADA PELO FFPLAY. QUAL É A RAZÃO PARA A DIFERENÇA ENTRE AS DUAS?

Para comparar a largura de banda fizemos a transmissão do “vídeoB” na maior resolução através do ffplay (**figura 37**) e observamos que os valores obtidos foram relativamente semelhantes (**figura 38**), mas não os esperados.

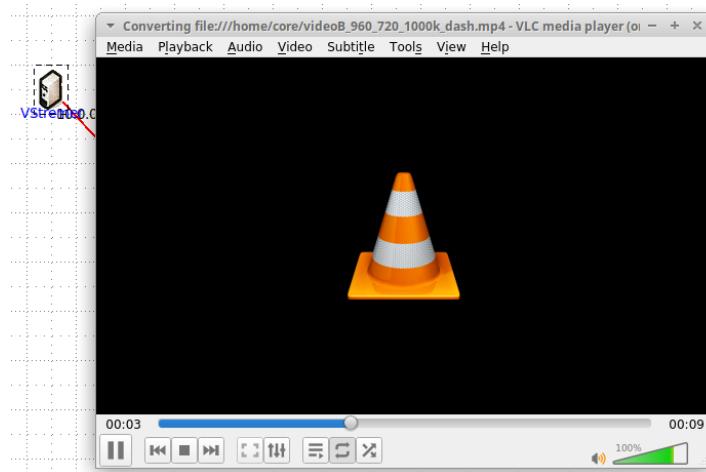


Figura 37 - Transmissão do "vídeoB"

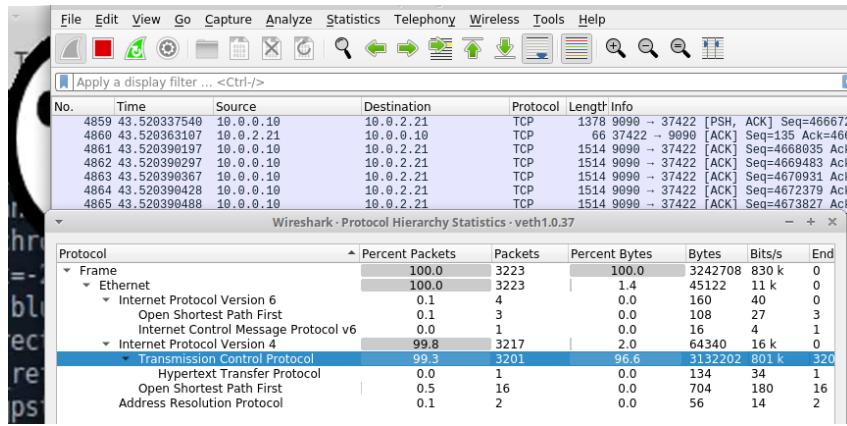


Figura 38 - Valores Obtidos

O ffplay tenderá a usar mais largura de banda, já que transmite com o bitrate original do vídeo, sem adaptação dinâmica para condições variáveis de rede, porém como foi usado o mesmo vídeo numa rede ilimitada sem variações o que aconteceu é justificável. Caso fosse imposta alguma limitação na rede o DASH adaptar-se-ia e usaria menos largura de banda (ajustando a qualidade do vídeo). Nesse caso o ffplay poderia nem sequer conseguir realizar conexão.

QUESTÃO 4: 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 E JUSTIFIQUE A LARGURA DE BANDA NECESSÁRIA PARA QUE O STREAM DE VÍDEO SOFRA ALTERAÇÕES.

Como podemos observar na **figura 30**, o débito da rede no portátil Bela foi limitado. Na figura seguinte (**figura 39**), vemos que o cliente Alladin solicitou o vídeo na maior resolução e conseguiu obtê-lo. Em seguida, a cliente Bela fez o mesmo pedido, mas, devido à sua capacidade de rede, não foi possível. Assim, logo a seguir, foi efetuado um pedido de uma versão do vídeo com menor resolução.

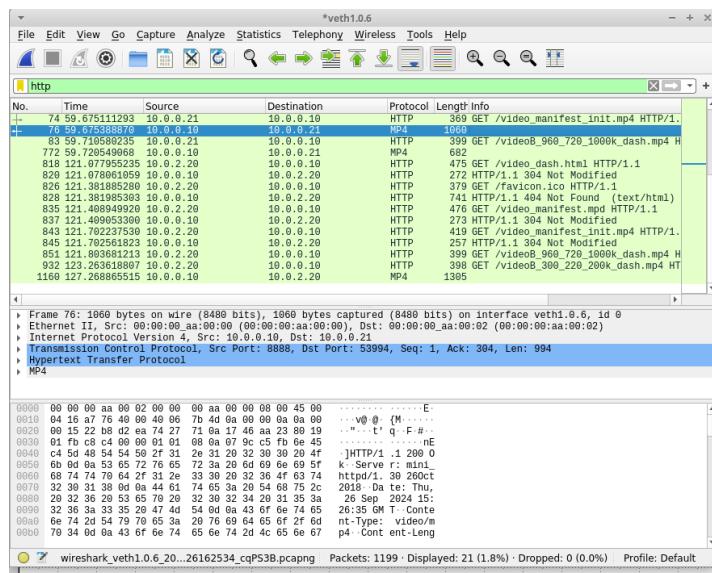


Figura 39 - Débitos no wireshark

Após discutir os resultados obtidos com a professora da aula prática, chegámos à conclusão de que a nossa máquina (possivelmente devido a uma atualização antes do início da ficha) não realiza o pedido por segmentos (chunks) e consegue receber todos os segmentos do vídeo de uma só vez. Isso explica a diferença em relação aos resultados esperados.

Com base na análise da **figura 33**, é possível inferir que uma largura de banda inferior a 798.777 bps provocaria alterações na transmissão do vídeo, sendo esse o valor mínimo teórico necessário para exibir o vídeo na sua maior resolução.



QUESTÃO 5: DESCREVA O FUNCIONAMENTO DO DASH NESTE CASO CONCRETO, REFERINDO O PAPEL DO FICHEIRO MPD CRIADO E COMPARANDO O MODELO DE STREAMING COM O QUE FOI UTILIZADO NA QUESTÃO 1

O Streaming Adaptativo Dinâmico sobre HTTP (DASH) é um protocolo que possibilita a transmissão de conteúdo de vídeo em tempo real de forma adaptativa.

Estrutura do DASH

- Segmentação de Vídeo: O vídeo é dividido em pequenos segmentos (geralmente com alguns segundos de duração). Cada segmento pode ter várias versões codificadas em diferentes qualidades (bitrate).
- Ficheiro MPD: É um ficheiro XML que descreve os segmentos disponíveis, as suas qualidades e a forma de aceder a eles.

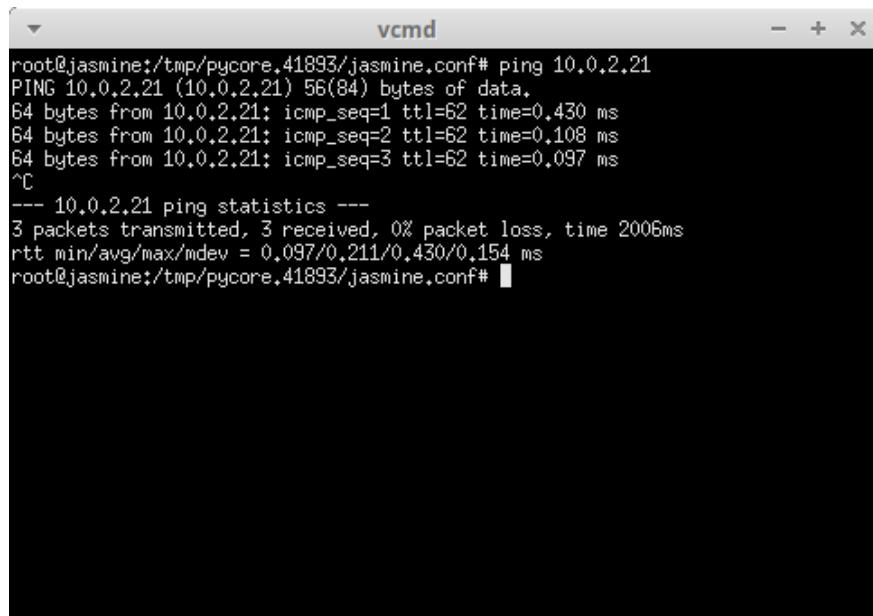
Comparação com o Modelo de Streaming Utilizado na Etapa 1:

- Qualidade Fixa: A qualidade do vídeo é definida no início da transmissão, sem possibilidade de ajuste dinâmico.
- Sem Segmentação: O vídeo pode ser transmitido como um fluxo contínuo ou em segmentos não adaptativos, o que pode resultar em interrupções se a largura de banda variar.

Vantagens do DASH em Comparaçāo

- Adaptação Dinâmica: O DASH permite que o cliente ajuste a qualidade do vídeo em tempo real.
- Experiência do Utilizador: Com a capacidade de adaptar a qualidade, o DASH proporciona uma experiência de visualização mais fluida, mesmo em condições de rede variáveis.

6. PROCEDIMENTOS DA ETAPA 3

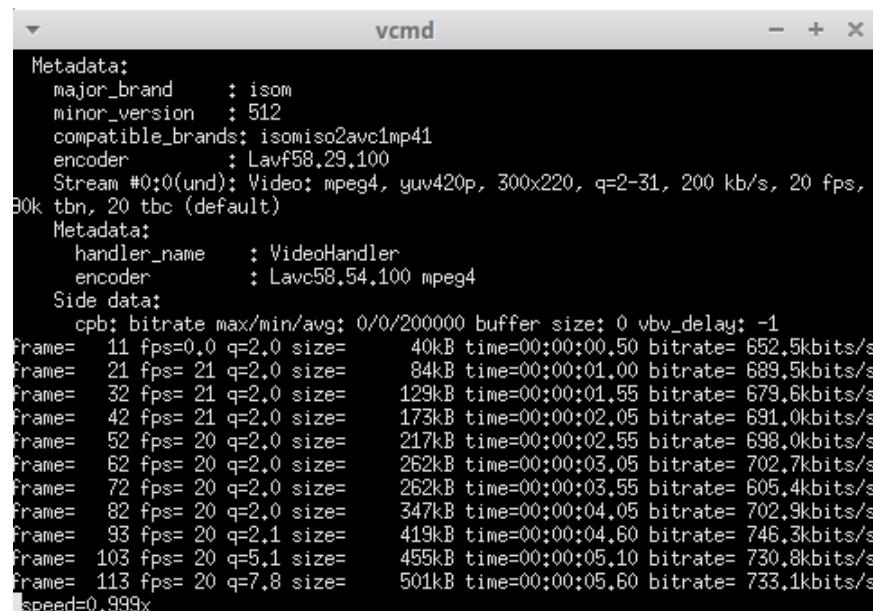


```

root@jasmine:/tmp/pycore.41893/jasmine.conf# ping 10.0.2.21
PING 10.0.2.21 (10.0.2.21) 56(84) bytes of data.
64 bytes from 10.0.2.21: icmp_seq=1 ttl=62 time=0.430 ms
64 bytes from 10.0.2.21: icmp_seq=2 ttl=62 time=0.108 ms
64 bytes from 10.0.2.21: icmp_seq=3 ttl=62 time=0.097 ms
^C
--- 10.0.2.21 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2006ms
rtt min/avg/max/mdev = 0.097/0.211/0.430/0.154 ms
root@jasmine:/tmp/pycore.41893/jasmine.conf#

```

Figura 40 - Ping na Jasmine



```

Metadata:
    major_brand     : isom
    minor_version   : 512
    compatible_brands: isomiso2avc1mp41
    encoder         : Lavf58.29.100
Stream #0:0(und): Video: mpeg4, yuv420p, 300x220, q=2-31, 200 kb/s, 20 fps,
90k tbn, 20 tbc (default)
Metadata:
    handler_name   : VideoHandler
    encoder        : Lavc58.54.100 mpeg4
Side data:
    cpb: bitrate max/min/avg: 0/0/200000 buffer size: 0 vbv_delay: -1
frame= 11 fps=0.0 q=2.0 size=      40kB time=00:00:00.50 bitrate= 652.5kbits/s
frame= 21 fps= 21 q=2.0 size=      84kB time=00:00:01.00 bitrate= 689.5kbits/s
frame= 32 fps= 21 q=2.0 size=     129kB time=00:00:01.55 bitrate= 679.6kbits/s
frame= 42 fps= 21 q=2.0 size=     173kB time=00:00:02.05 bitrate= 691.0kbits/s
frame= 52 fps= 20 q=2.0 size=     217kB time=00:00:02.55 bitrate= 698.0kbits/s
frame= 62 fps= 20 q=2.0 size=     262kB time=00:00:03.05 bitrate= 702.7kbits/s
frame= 72 fps= 20 q=2.0 size=     262kB time=00:00:03.55 bitrate= 605.4kbits/s
frame= 82 fps= 20 q=2.0 size=     347kB time=00:00:04.05 bitrate= 702.9kbits/s
frame= 93 fps= 20 q=2.1 size=     419kB time=00:00:04.60 bitrate= 746.3kbits/s
frame= 103 fps= 20 q=5.1 size=     455kB time=00:00:05.10 bitrate= 730.8kbits/s
frame= 113 fps= 20 q=7.8 size=     501kB time=00:00:05.60 bitrate= 733.1kbits/s
speed=0.999x

```

Figura 41 - Início uma sessão de streaming com RTP com ffmpeg

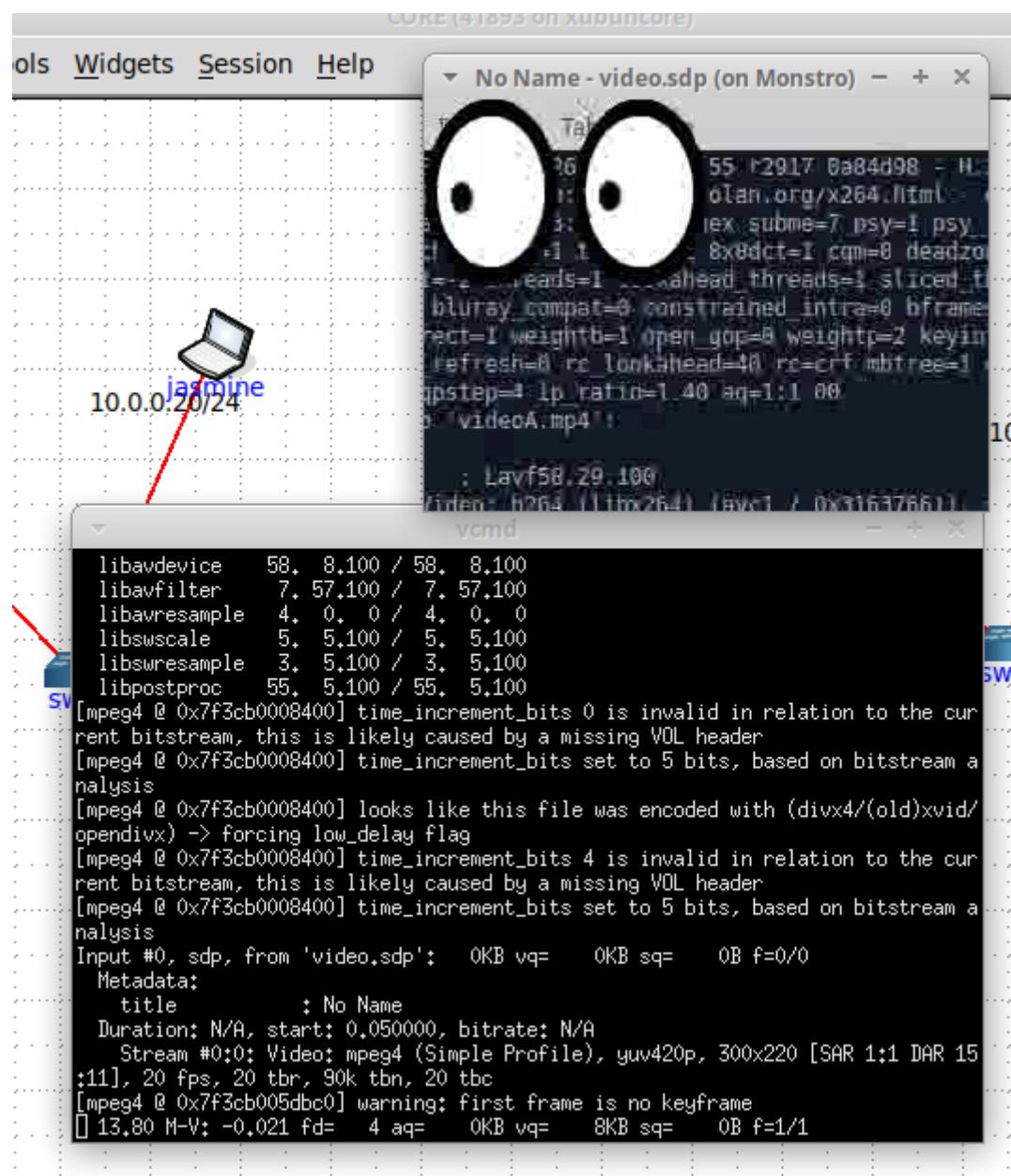


Figura 42 - Inicio de um cliente ffplay no Monstro:

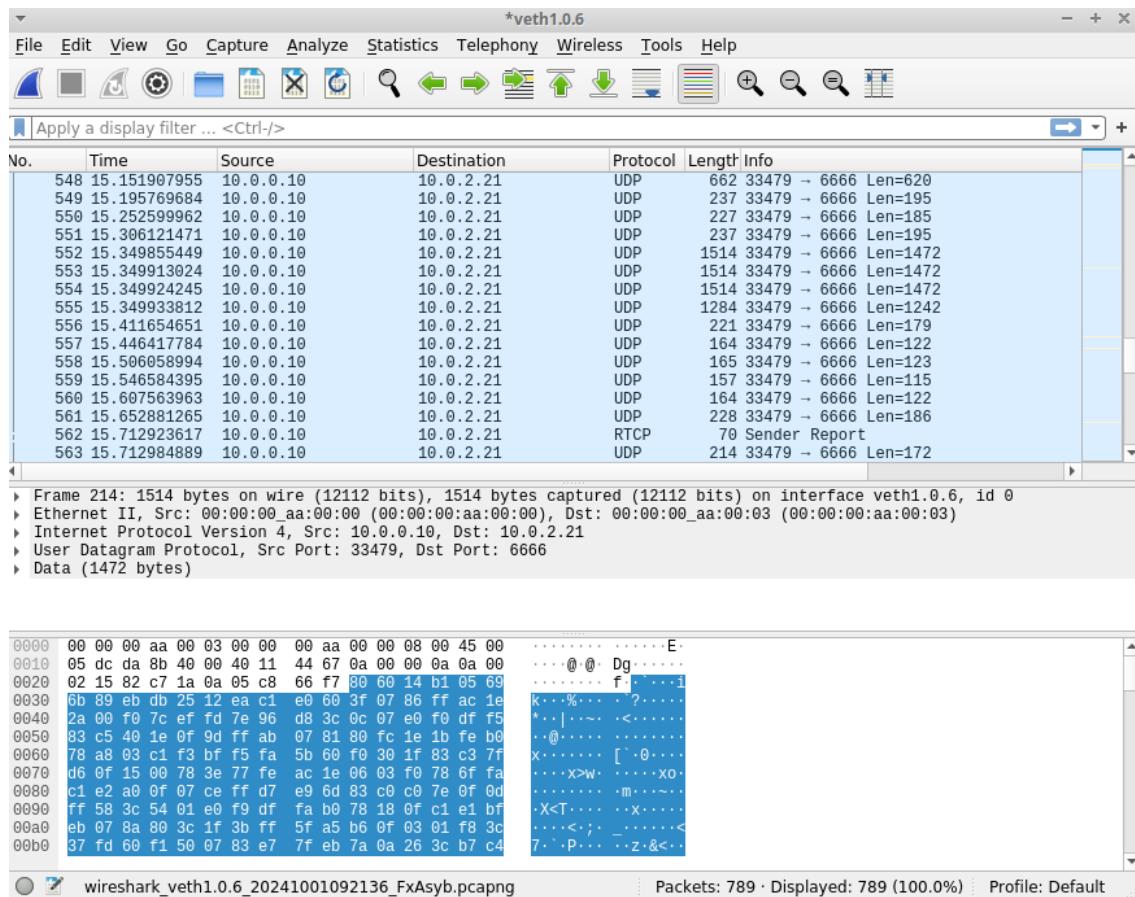


Figura 43 - Captura e tráfego com o Wireshark no link de saída do servidor

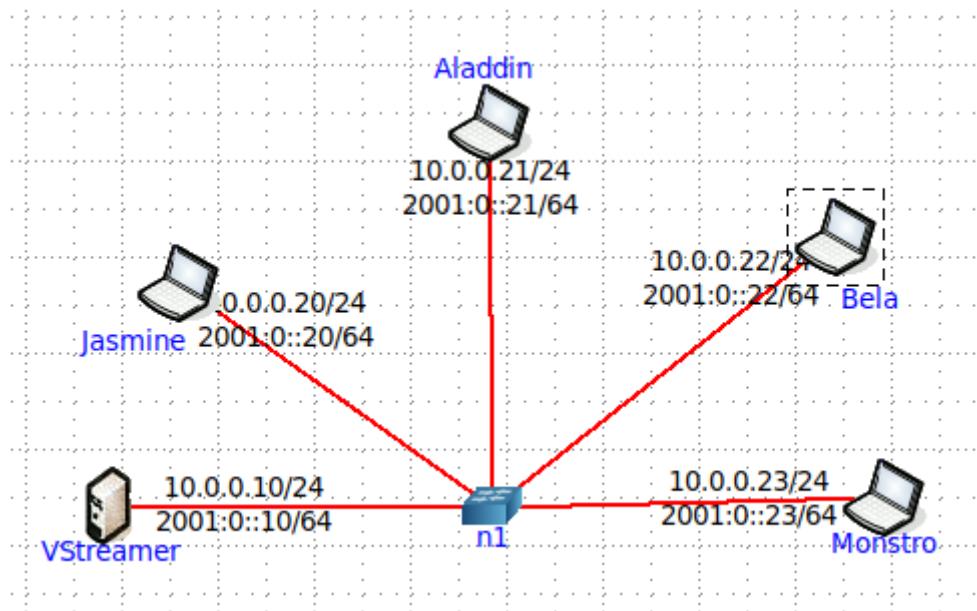


Figura 44 - Nova topologia



The figure consists of two vertically stacked terminal windows titled "vcmd".

The top window shows the command "ping 10.0.0.23" being run from a root shell. The output indicates successful ping results with TTL values of 64 and times ranging from 0.061 ms to 0.338 ms. It also displays ping statistics for 3 transmitted packets, 3 received, 0% loss, and a round-trip time (rtt) of 0.124 ms.

```
root@Jasmine:/tmp/pycore.40775/Jasmine.conf# ping 10.0.0.23
PING 10.0.0.23 (10.0.0.23) 56(84) bytes of data.
64 bytes from 10.0.0.23: icmp_seq=1 ttl=64 time=0.338 ms
64 bytes from 10.0.0.23: icmp_seq=2 ttl=64 time=0.091 ms
64 bytes from 10.0.0.23: icmp_seq=3 ttl=64 time=0.061 ms
^C
--- 10.0.0.23 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2042ms
rtt min/avg/max/mdev = 0.061/0.163/0.338/0.124 ms
root@Jasmine:/tmp/pycore.40775/Jasmine.conf#
```

The bottom window shows a continuous stream of video frame statistics. Each line starts with "frame=" followed by a frame number, then "fps=" and a value (e.g., 20), then "q=" and a quality value (e.g., 20.9), then "size=N/A", then "time=" and a timestamp (e.g., 00:00:07.55), then "bitrate=N/A", and finally "speed=0.996x". The frame numbers range from 152 to 456, and the speed remains constant at 0.996x.

```
frame= 152 fps= 20 q=20.9 size=N/A time=00:00:07.55 bitrate=N/A speed=0.996x
frame= 162 fps= 20 q=19.7 size=N/A time=00:00:08.05 bitrate=N/A speed=0.996x
frame= 172 fps= 20 q=16.3 size=N/A time=00:00:08.55 bitrate=N/A speed=0.995x
frame= 183 fps= 20 q=18.2 size=N/A time=00:00:09.10 bitrate=N/A speed=0.999x
frame= 193 fps= 20 q=19.4 size=N/A time=00:00:09.60 bitrate=N/A speed=0.999x
frame= 203 fps= 20 q=21.1 size=N/A time=00:00:10.10 bitrate=N/A speed=0.999x
frame= 213 fps= 20 q=20.8 size=N/A time=00:00:10.60 bitrate=N/A speed=0.998x
frame= 223 fps= 20 q=15.8 size=N/A time=00:00:11.10 bitrate=N/A speed=0.997x
frame= 233 fps= 20 q=16.3 size=N/A time=00:00:11.60 bitrate=N/A speed=0.997x
frame= 243 fps= 20 q=13.4 size=N/A time=00:00:12.10 bitrate=N/A speed=0.996x
frame= 264 fps= 20 q=12.4 size=N/A time=00:00:13.15 bitrate=N/A speed=0.999x
frame= 274 fps= 20 q=12.4 size=N/A time=00:00:13.65 bitrate=N/A speed=0.999x
frame= 284 fps= 20 q=17.1 size=N/A time=00:00:14.15 bitrate=N/A speed=0.999x
frame= 294 fps= 20 q=18.5 size=N/A time=00:00:14.65 bitrate=N/A speed=0.998x
frame= 304 fps= 20 q=22.8 size=N/A time=00:00:15.15 bitrate=N/A speed=0.998x
frame= 314 fps= 20 q=17.8 size=N/A time=00:00:15.65 bitrate=N/A speed=0.997x
frame= 324 fps= 20 q=16.3 size=N/A time=00:00:16.15 bitrate=N/A speed=0.997x
frame= 335 fps= 20 q=14.9 size=N/A time=00:00:17.70 bitrate=N/A speed=0.999x
frame= 365 fps= 20 q=19.4 size=N/A time=00:00:18.20 bitrate=N/A speed=0.999x
frame= 375 fps= 20 q=11.9 size=N/A time=00:00:18.70 bitrate=N/A speed=0.999x
frame= 436 fps= 20 q=22.0 size=N/A time=00:00:21.75 bitrate=N/A speed=0.999x
frame= 446 fps= 20 q=31.0 size=N/A time=00:00:22.25 bitrate=N/A speed=0.999x
frame= 456 fps= 20 q=27.1 size=N/A time=00:00:22.75 bitrate=N/A speed=0.999x
```

Figura 45 - Teste a conectividade entre os sistemas

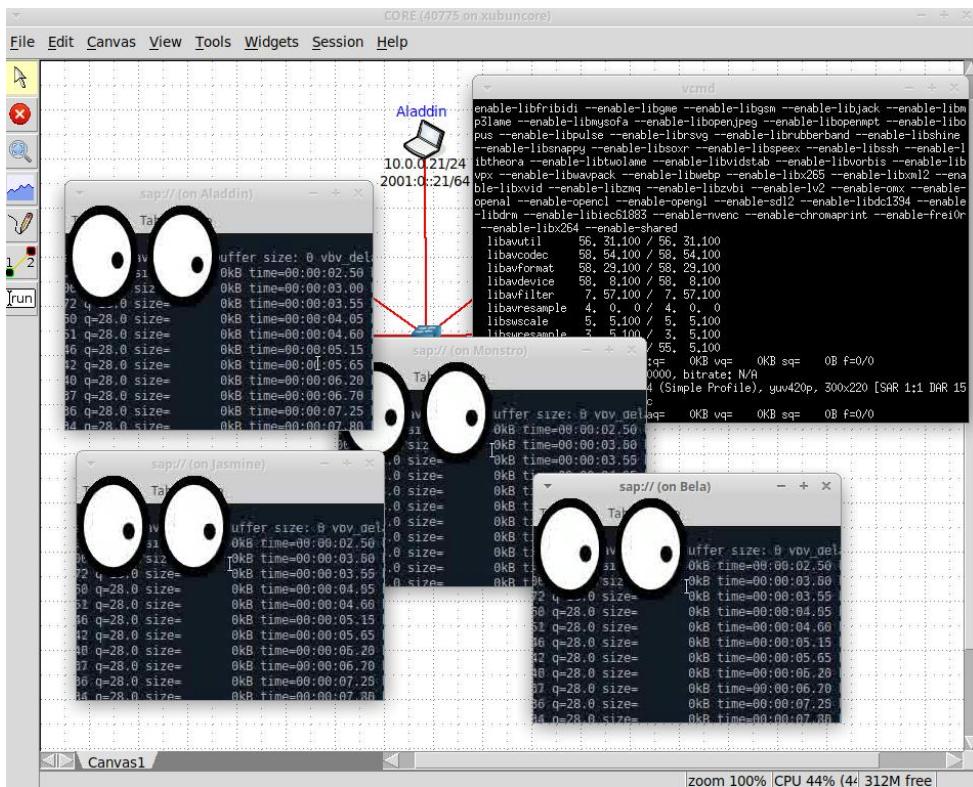


Figura 46 - Início de diferentes sessões

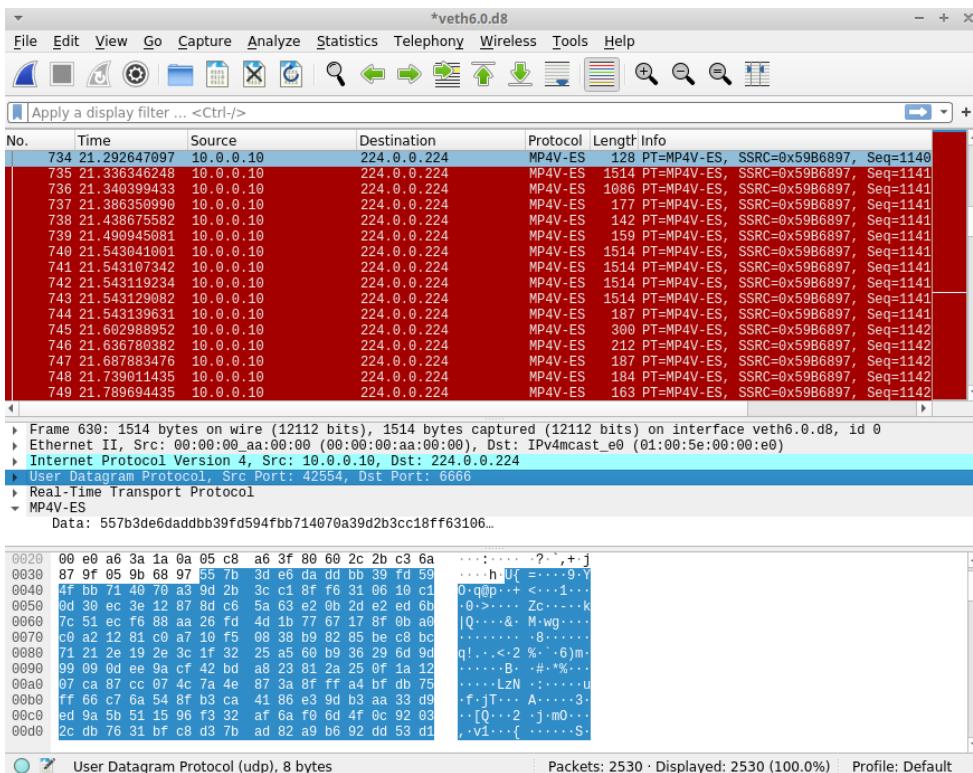


Figura 47 - Captura do tráfego no link de saída do servidor com o Wireshark

7. QUESTÕES E RESPOSTAS DA ETAPA 3

QUESTÃO 6: 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 TAMBÉM PARA OS CENÁRIOS DE 1000 E 10000 CLIENTES.

Comparando os dois cenários, no cenário **Unicast** a comunicação é feita de um único remetente para um único receptor. Cada transmissão gera uma cópia separada dos dados para cada cliente.

Ethernet - 5 IPv4 - 4 IPv6 - 1 TCP UDP - 7													
Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.0.0.10	40837	10.0.0.23	6666	3.268	2562 k	3.268	2562 k	0	0	0.000000	98.0879	209 k	0
10.0.0.10	54389	10.0.0.22	6666	3.370	2694 k	3.370	2694 k	0	0	0.035918	98.0515	219 k	0

Figura 48 - Cenário Unicast

O cenário **Multicast** permite que um único fluxo de dados seja enviado a múltiplos destinatários simultaneamente. Este cenário apresenta várias vantagens, entre elas temos a escalabilidade e a eficiência de rede.

Ethernet - 3 IPv4 - 2 IPv6 - 1 TCP UDP - 3													
Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.0.0.10	42554	224.0.0.224	6666	2.501	1983 k	2.501	1983 k	0	0	0.000000	74.0390	214 k	0

Figura 49 - Cenário Multicast

Utilizando uma expressão matemática semelhante à utilizada na **pergunta c) da Etapa 1**, juntamente com os valores obtidos nas **figuras 49 e 50** relativos ao fluxo de dados, podemos analisar um cenário com 1000 clientes da seguinte forma:

Para **Unicast**:

$$D \approx 215 \text{ kbps} \times 1000 = 215 \text{ 000 kbps.}$$

Para **Multicast**:

Como não é criado um novo fluxo de dados para cada cliente:

$$D \approx 219 \text{ kbps} \times 1 = 2190 \text{ kbps.}$$



Deste modo, verificamos que, no Unicast, o tráfego gerado é bastante elevado, o que pode causar congestionamento na rede. Em contrapartida, no Multicast, o tráfego é otimizado, tendo um impacto muito reduzido na largura de banda.

Considerando um cenário com 10.000 clientes, a demanda aumentaria de forma proporcional, resultando em aproximadamente 10 vezes mais tráfego no modelo Unicast. Isso poderia levar a uma situação crítica, com lentidão ou até falhas no serviço, tornando essa abordagem muito pouco escalável. Em contrapartida, no modelo Multicast, o tráfego é mantido sob controle, pois os valores permanecem constantes, garantindo a eficiência da rede e demonstrando, assim, ser uma solução muito mais escalável.



8. CONCLUSÃO

Neste trabalho, foram explorados os principais conceitos relacionados com o streaming de vídeo. Ao longo das três etapas do projeto, foi possível avaliar a eficácia de protocolos como HTTP, DASH, unicast e multicast.

Consideramos que todos os objetivos propostos foram atingidos, embora a nossa máquina tenha apresentado resultados inesperados na etapa 2, aquando da captura de pacotes do protocolo DASH. Acreditamos que alguns valores de questões relacionadas possam ter sido afetados devido a essa situação, mas não consideramos que tenha sido um obstáculo à nossa aprendizagem e entendimento do tema.

Em suma, este trabalho proporcionou uma visão prática e teórica sobre os desafios e soluções relacionados com o streaming em redes modernas, contribuindo significativamente para a nossa compreensão das tecnologias de transmissão de conteúdos multimédia na Internet.