Trabalho Prático 1
Engenharia de Serviços em Rede
PL34

Mariana Rodrigues[pg50622], Jorge Melo[pg50507], and Inês Vicente[pg50436]

Universidade do Minho

ETAPA 1 - Streaming HTTP simples sem adaptação dinâmica de débito

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 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).

Inicialmente, começamos por fazer a captura do **videoA.mp4** e do **videoB.mp4**.

Figura 1. Vídeos A e B

Prosseguindo para a construção da topologia base no CORE.

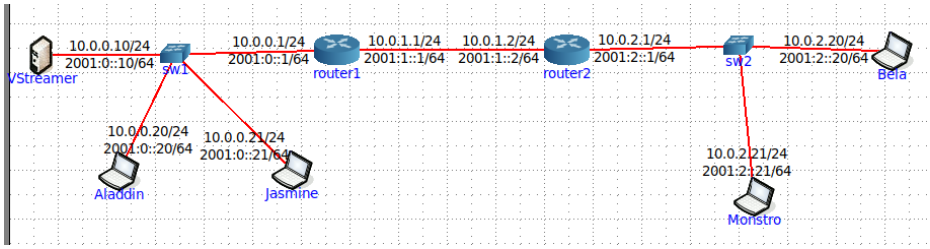

Figura 2. Topologia Core

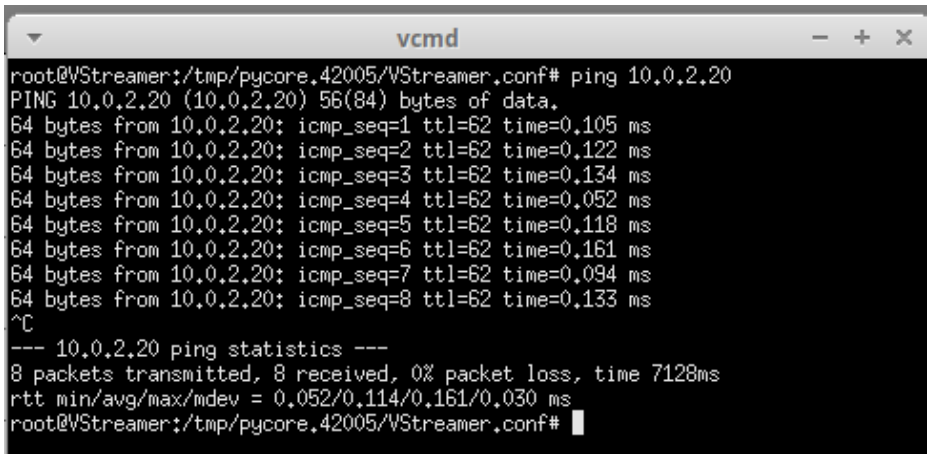

Figura 3. Teste à conectividade da rede

Com a topologia a funcionar e os vídeos capturados anteriormente, avançamos para fazer o streaming por **HTTP**, no **VStreamer** com o *VLC* do ficheiro *videoA.mp4* com transcoding para "**Video – Theora + Vorbis (Ogg)**". Posteriormente, colocamos o *VLC* no portátil **Jasmine**, mas desta vez a funcionar como cliente.

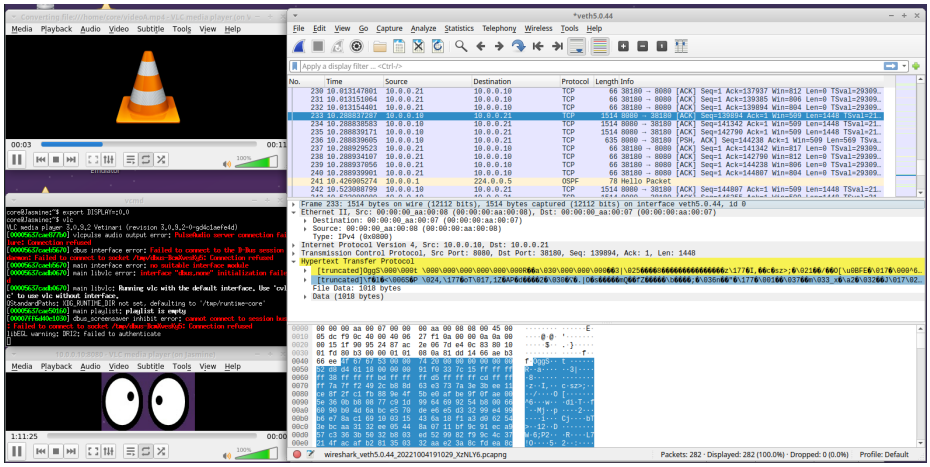

Figura 4. Streaming por HTTP do ficheiro videoA.mp4 através do VLC

Tendo isto, prosseguimos para a transmissão do videoA.mp4 através de uma página HTML.

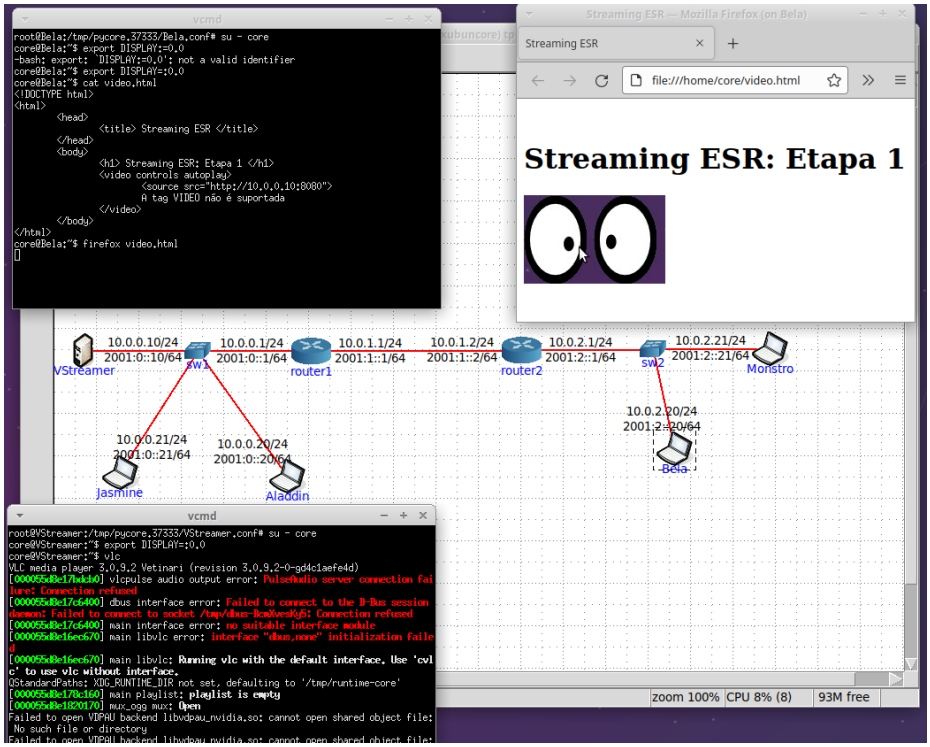

Figura 5. Verificação do funcionamento do vídeo na página do Firefox

Verificando que estava tudo a correr como previsto, efetuamos a recolha de uma amostra de tráfego.

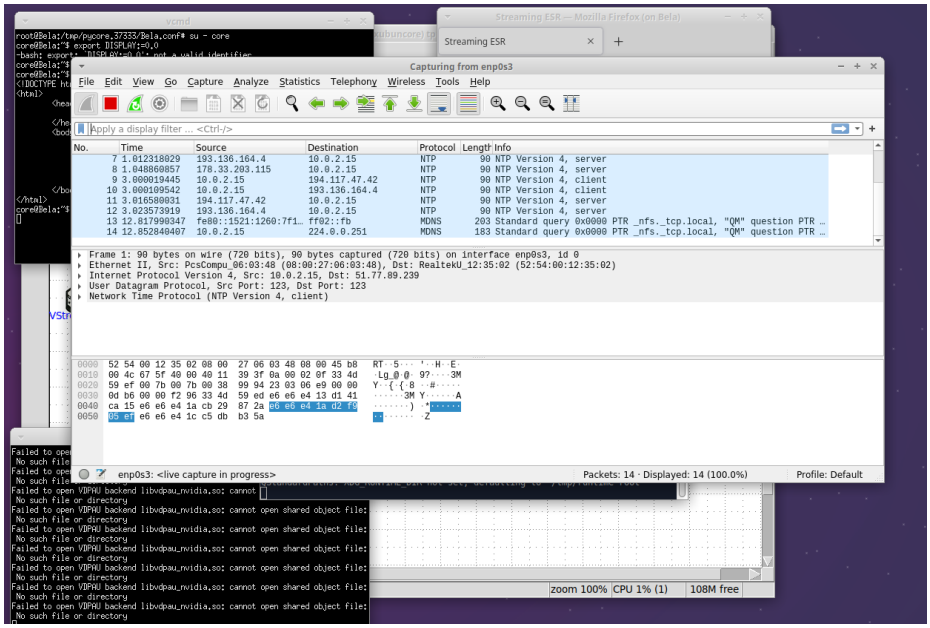

Figura 6. Recolha de uma amostra de tráfego no wireshark

Posto isto, tendo os 2 clientes anteriores a funcionar separadamente, seguimos para o teste de ambos os clientes em conjunto.

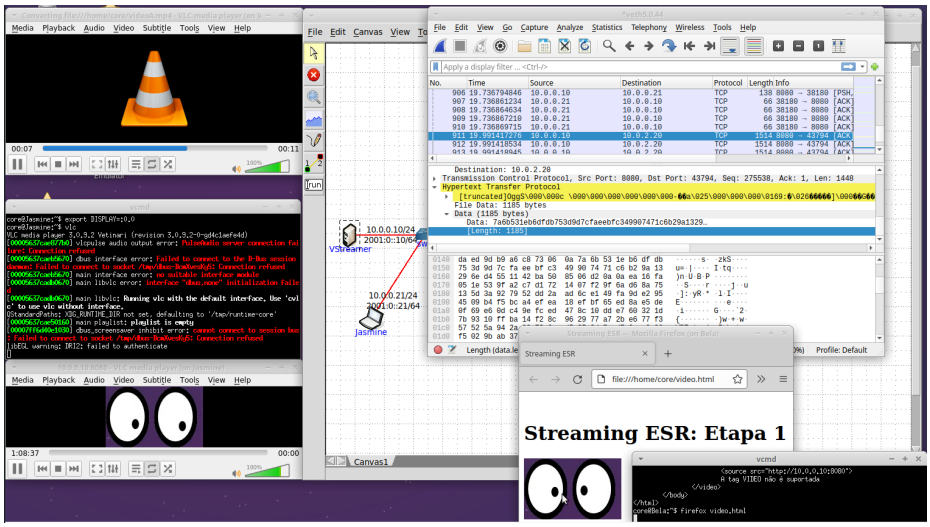

Figura 7. 2 clientes (VLC e Firefox)

Por último, efetuamos a stream de vídeo usando o comando **ffplay** como um terceiro cliente no portátil **Monstro**.

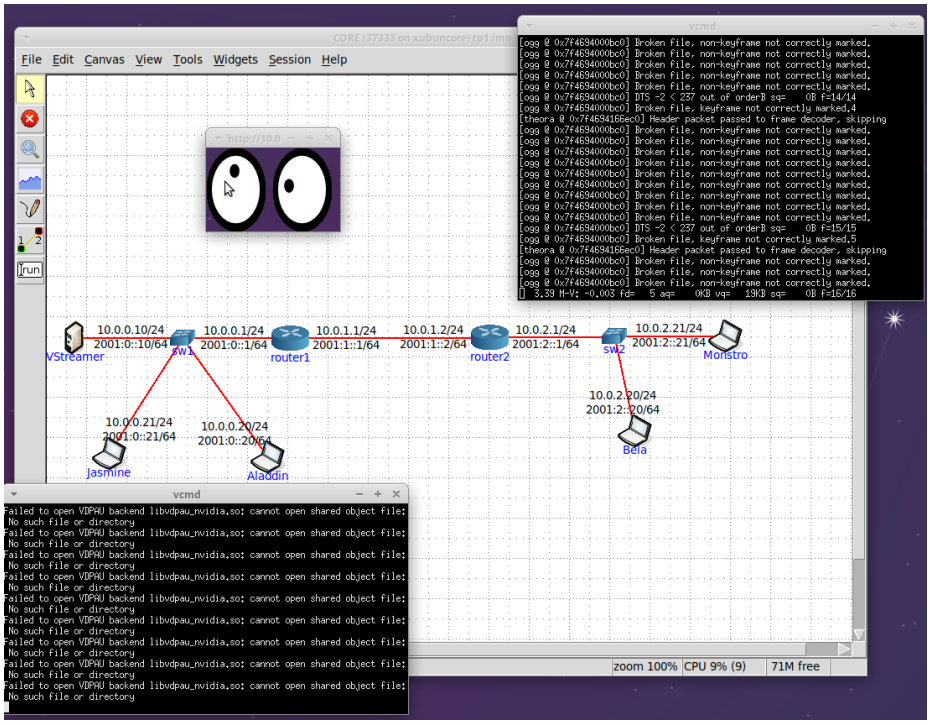

Figura 8. Três clientes (VLC, Firefox e ffplay)

Com os três clientes a correr, as imagens anteriores e com recolhas de informação a partir da ferramenta "*Wireshark*", verificámos que:

– **encapsulamento usado**:
  - **VLC :** Ethernet II, IPv4 e TCP
  - **Firefox :** Ethernet II, IPv4, TCP e Hypertex Transfer Protocol
  - **FFPlay :** Ethernet II, IPv4, TCP e Hypertex Transfer Protocol
– **total de fluxos gerados**:

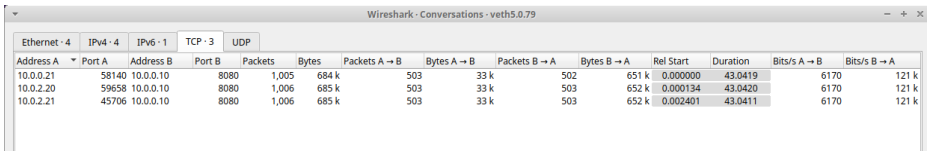

Figura 9. Informacao relativa as amostras de tragefo no link de saıda do servidor

  - **VLC:** 503
  - **Firefox:** 503
  - **ffplay:** 503
– **taxa em bps necessárias**:

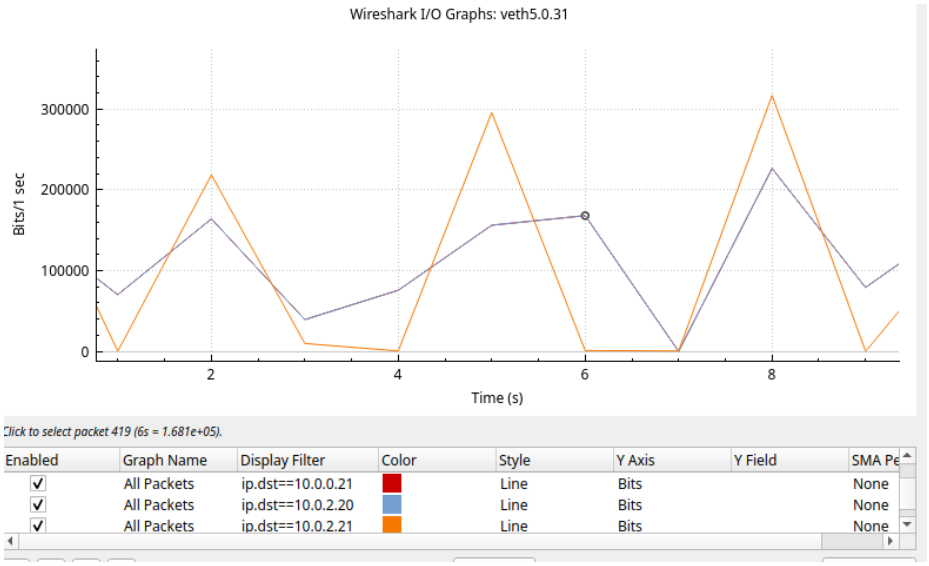

Figura 10. Wireshark I/O Graph dos bps dos 3 clientes

2

Etapa 2 - Streaming adaptativo sobre HTTP (MPEG-DASH)

Procedendo, agora, para a criação de três variantes do **vídeoB.mp4**, com diferentes resoluções

Começamos por colocar os vídeos a serem visualizados no firefox, em simultâneo, nos portáteis **Bela** e **Aladdin**.

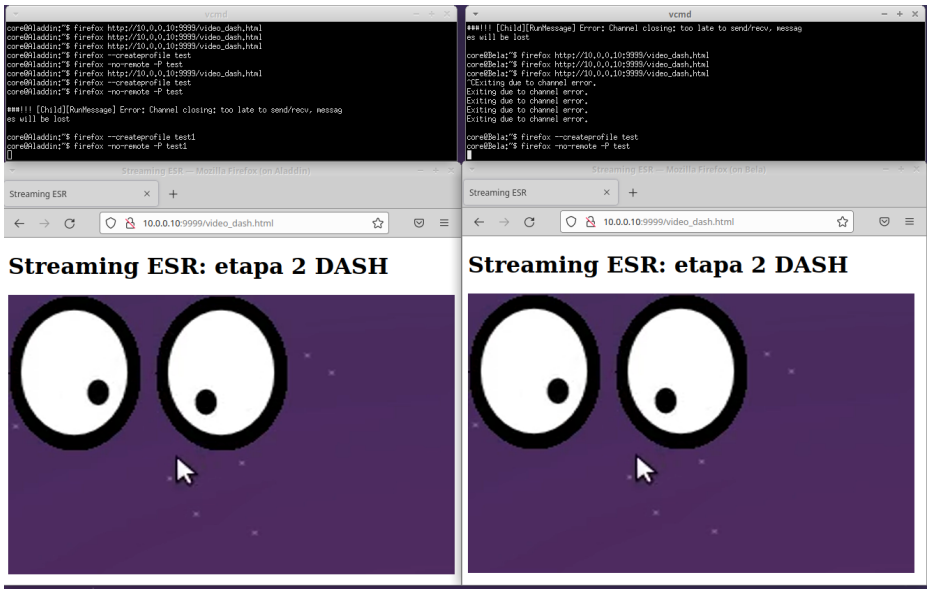

Figura 11. Visualização com o firefox nos portáteis Bela e Aladdin

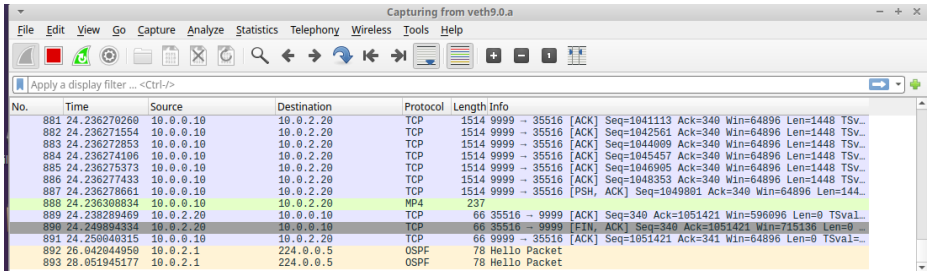

Figura 12. Captura de tráfego no portátil Bela

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.

Figura 13. video_manifest.mpd

Observando o ficheiro *video_manifest.mpd*, podemos observar que o vídeo de menor resolução tem uma *bandwidth* de **121717** bps.

A pilha protocolar necessária para que o cliente conseguisse receber o vídeo no firefox foi com TCP, Internet Protocol e HTTP.

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.

No core, alterando o limite de *bps* do link entre o Bela e o *switch* para um valor entre a *bandwidth* do vídeo com menos resolução e do segundo vídeo com menor resolução, conseguimos forçar o portátil Bela a transmitir para o firefox o vídeo com a menor resolução.

Quanto ao Alladin, o link está definido como 0, que é o valor *default*, o que significa que não há limite de tráfego que pode passar naquele link. Portanto, o vídeo de maior resolução será transmitido por defeito.

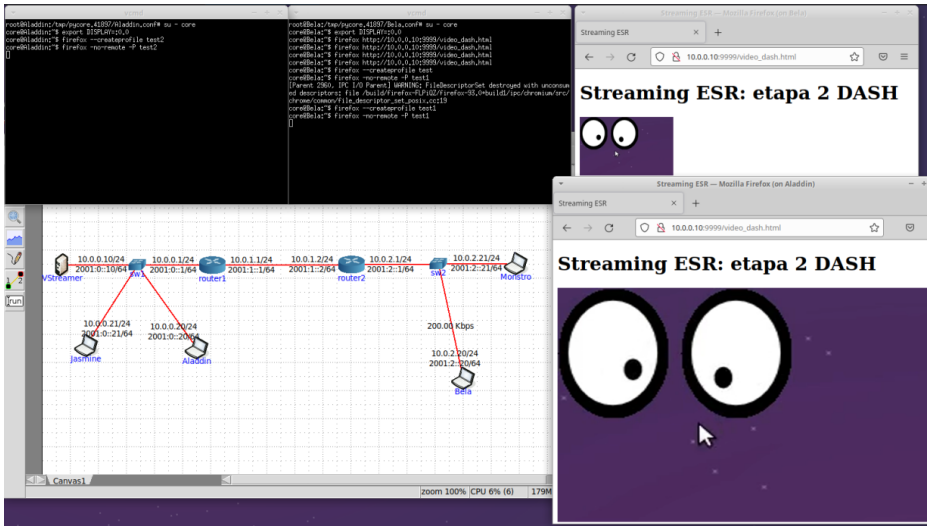

Figura 14. Visualização com o firefox no portátil Bela com o vídeo de menor resolução e do Alladin com o de maior resolução

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

O cliente tenta aceder à página. O servidor envia o ficheiro MPD, que o cliente consulta e, a partir daí, obtém a informação relativa à *bandwidth* mínima requerida pelos diferentes vídeos. Depois, tendo em conta a *bandwidth* de ligação entre o servidor e o cliente, escolhe a resolução que consegue suportar. Ou seja, ao diminuir a *bandwidth* da ligação, naturalmente o vídeo que o cliente vai obter vai ter uma menor resolução.

Etapa 3: Streaming RTP/RTCP unicast sobre UDP e multicast com anúnicos SAP

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.

Começando, então, com o *unicast*, criamos uma sessão de *streaming* com RTP com o *ffmpeg* no **VStreamer** e um cliente *ffplay* no portátil **Monstro**.

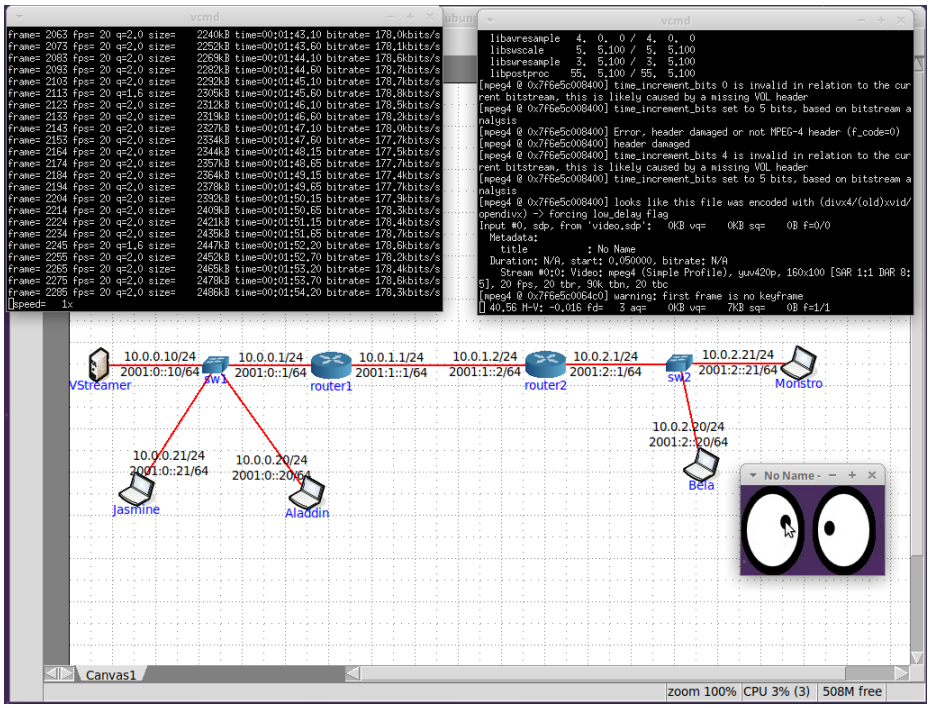

Figura 15. Sessão de *streaming unicast*

De seguida, capturámos o tráfego com o *wireshark* no link de saída do servidor.

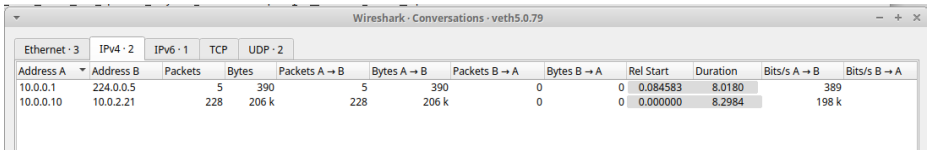

Figura 16. Captura de tráfego para *unicast*

Podemos notar que o débito para *Unicast* é de **198k bps**.
Para o *multicast*, criamos uma topologia com apenas um *switch*, com 4 portáteis e um servidor ligados ao mesmo.

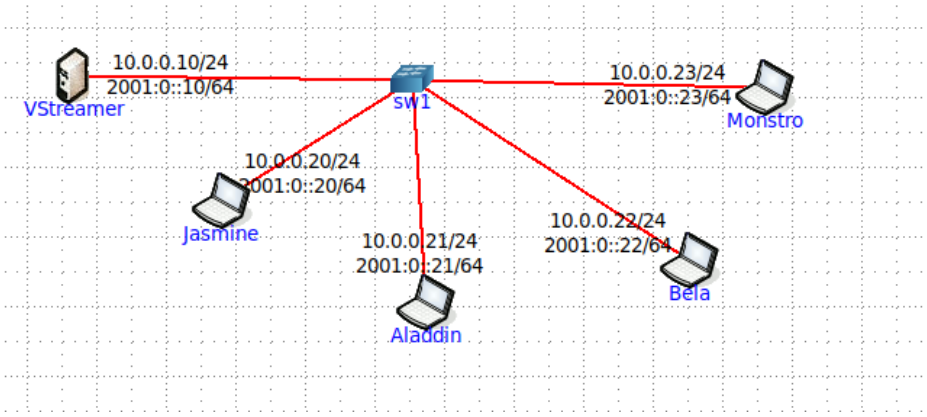

Figura 17. Topologia para *multicast*

Em seguida, iniciámos, para cada portátil, uma sessão de *streaming multicast* com o *ffmpeg* em sessões diferentes.

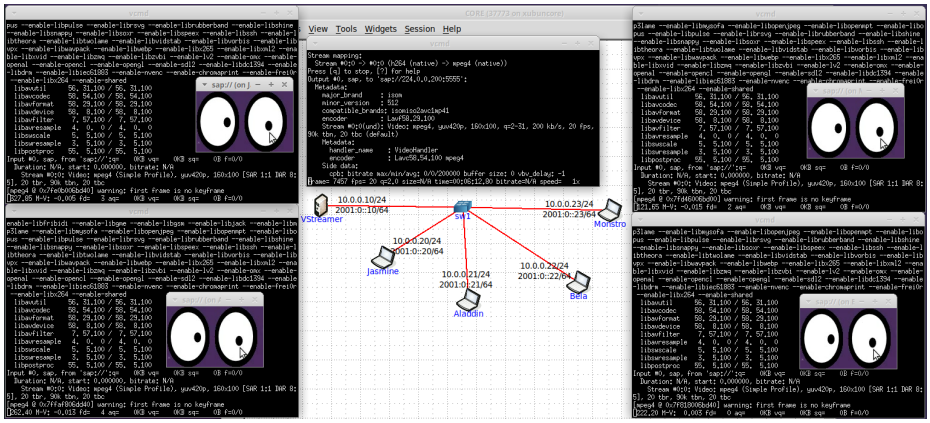

Figura 18. Vários portáteis em sessões diferentes em *multicast*

Finalmente, capturámos o tráfego no link de saída do servidor do wireshark

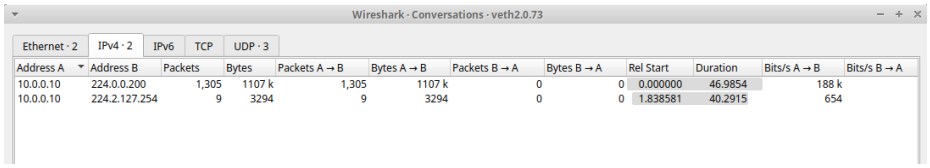

Figura 19. Captura de tráfego para *multicast*

Podemos então observar que o débito é de **188k** para *multicast*.

Para finalizar e para conseguirmos comparar ambos os cenários, fomos comparar os valores de débito:

- **unicast**: 198k
- **multicast**: 188k

Com isto, podemos concluir algumas vantagens que o *multicast* tem sobre o *unicast*. Em termos de escalabilidade, podemos observar que com um débito semelhante, o *multicast* consegue satisfazer um maior número de clientes.

Para além disso, em *unicast*, para calcular o débito total multiplica-se o número de clientes pelo débito de um cliente, enquanto que o *multicast*, que só envia o conteúdo uma vez por todas as redes, mantém um débito mais ou menos constante, pelo que aumentar o número de clientes para *unicast* é muito mais dispendioso.

Conclusão

Este projeto serviu de ferramenta para aprender vários temas abordados nesta cadeira, entre elas:

- Streaming HTTP com e sem adaptação dinâmica de débito
- funcionamento do protocolo DASH
- diferenças entre *unicast* e *multicast*, as suas vantagens e desvantagens