



TP1- Engenharia de Serviços em Rede (ESR)
Redes sem Fios Streaming de áudio e vídeo a pedido e em tempo real

Grupo 31

João Castro (PG53929)
Rodrigo Freitas (PG54197)

October 12, 2023

1 Etapa 1

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).

Resposta:

1.1 Cenário 1

No primeiro cenário em que temos o servidor a fazer streaming de vídeo por HTTP através do VLC, e um único cliente (Jasmine) foram obtidos as seguintes taxas de débito a partir da captura do tráfego no Wireshark (Figure 1). Nessa captura podemos ver que no sentido VStreamer(10.0.0.10) → Jasmine(10.0.0.20) o débito foi de 59kbps (última coluna), enquanto que no sentido oposto foi de 3060 bps. Logo, foi necessária uma taxa de aproximadamente 62kbps (Figure 1). Um outro método adotado de averiguar a largura de banda era através da visualização gráfica da variação do débito, para isso foi necessário filtrar os pacotes da stream TCP na opção I/O Graphs do menu Statistics (Figure 2).

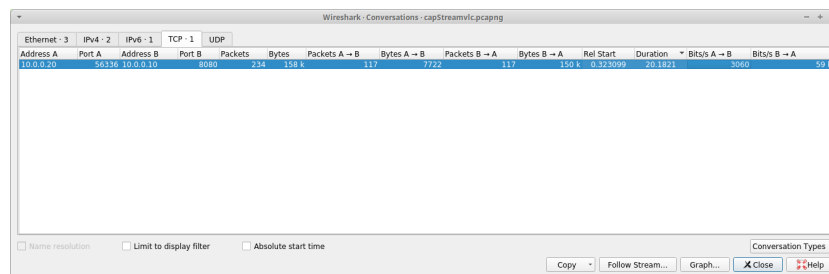


Figure 1: Estatísticas do tráfego capturado entre VStreamer e Jasmine

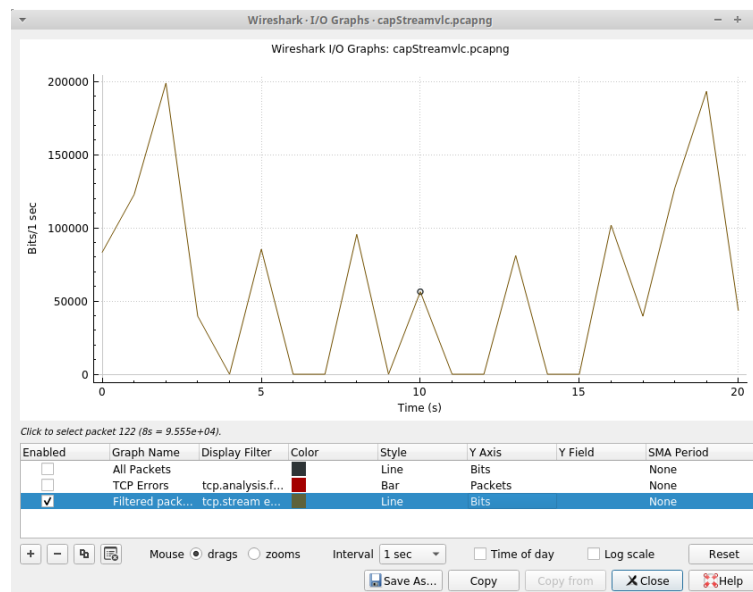


Figure 2: Gráfico da variação da largura de banda.

Neste cenário existe apenas 1 fluxo (VStreamer < - > Jasmine) como é possível ver através do gráfico de fluxos, onde está presente os handshakes e a troca de sinais ACK entre estes dois endpoints. Ainda no que toca ao encapsulamento, estes foram os protocolos usados ao longo da captura de tráfego, Ethernet > IP > TCP > HTTP (ordem de encapsulamento - Figure 4).

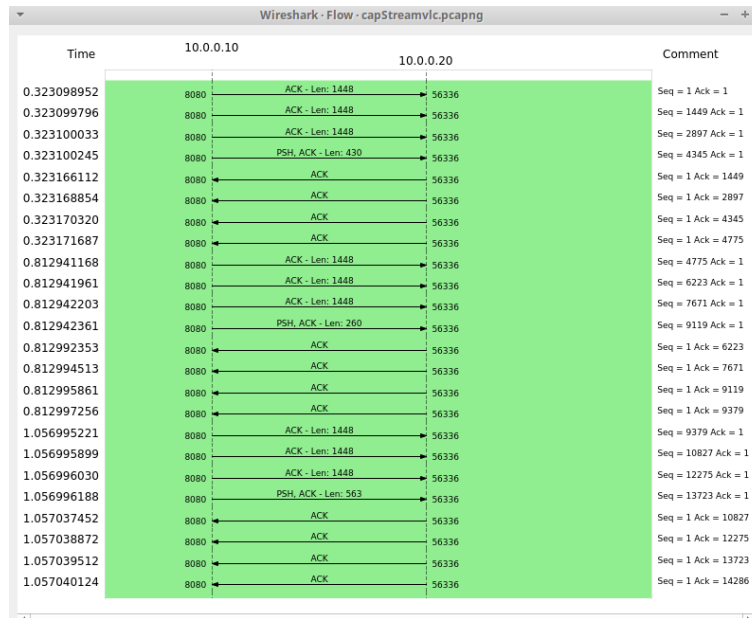


Figure 3: Gráfico de fluxos.

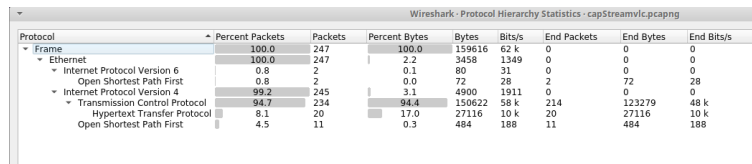


Figure 4: Encapsulamento (Protocol Hierarchy).

1.2 Cenário 2

No cenário 2 temos mais um cliente, sendo que este irá assistir ao streaming de vídeo através do Firefox. Como iremos observar de seguida, a largura de banda diminui com o aumento do número de utilizadores nesta infraestrutura. Na stream TCP 0, a taxa de débito do VStreamer para a Jasmine caiu dos 59 para os 47 kbps. Já na stream TCP 1 temos uma taxa de 54 kbps.

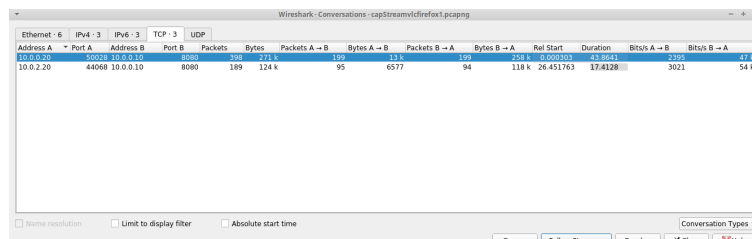


Figure 5: Estatísticas do tráfego capturado entre VStreamer - Jasmine, e VStreamer - Bela.

Neste cenário existem dois fluxos, um TCP stream correspondente ao fluxo VStreamer (10.0.0.10) - Jasmine (10.0.0.20), e um outro TCP stream correspondente ao fluxo VStreamer (10.0.0.10) -Bela (10.0.2.20). Por fim, estes foram os protocolos usados ao das comunicações entre os endpoints(Figure 7).

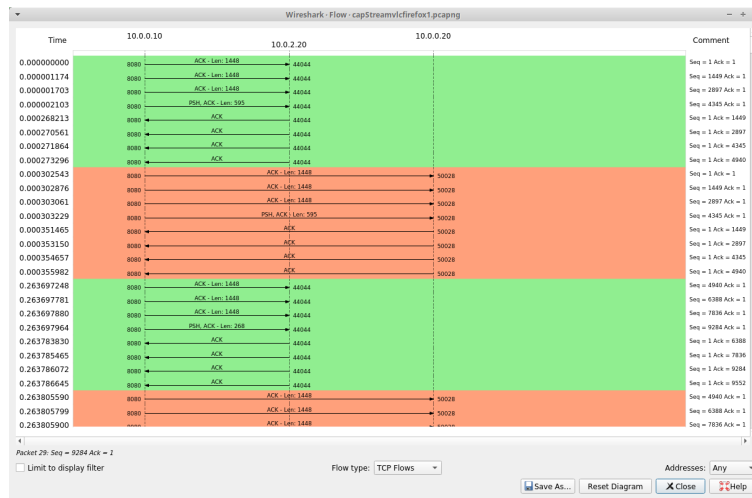


Figure 6: Gráfico de fluxos.

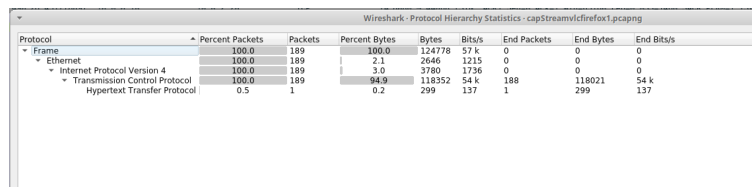


Figure 7: Encapsulamento (Protocol Hierarchy).

1.3 Cenário 3

No cenário 3 é adicionado mais um cliente, sendo que este irá executar o comando ffplay de modo a assistir o video a que o servidor está a fazer o streaming. Nesta situação é introduzida uma maior latência na taxa de débito, já que quer a Jasmine quer a Bela estão a receber dados a uma taxa de 46 kbps (menor que no cenário 2).

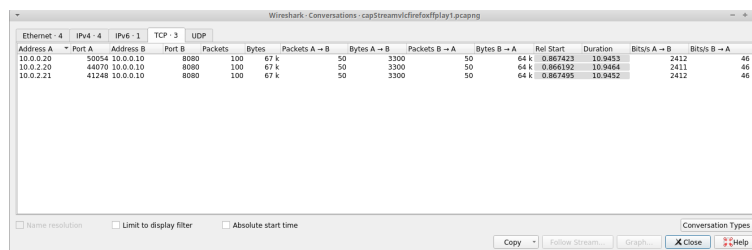


Figure 8: Estatísticas do tráfego capturado entre VStreamer - Jasmine, VStreamer - Bela, e VStreamer - Monstro

Neste cenário existem três fluxos, um TCP stream correspondente ao fluxo VStreamer (10.0.0.10) - Jasmine (10.0.0.20), um outro TCP stream correspondente ao fluxo VStreamer (10.0.0.10) -Bela (10.0.2.20), e um TCP stream correspondente ao fluxo VStreamer - Monstro(10.0.2.21).

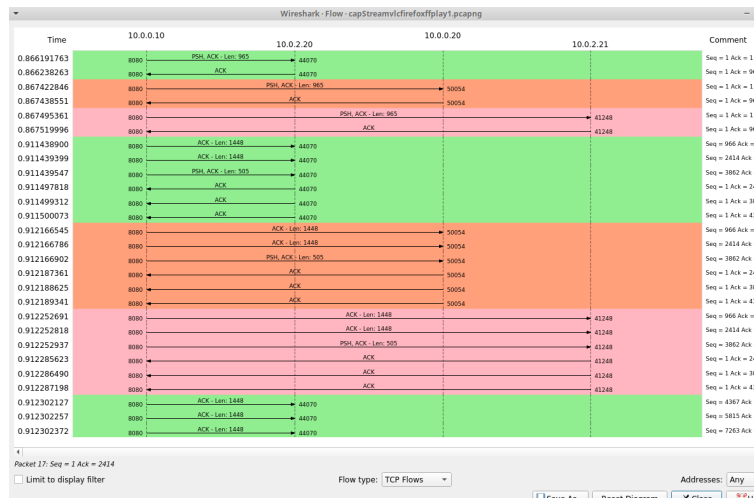


Figure 9: Gráfico de fluxos

Por fim, estes foram os protocolos usados ao das comunicações entre os endpoints(Figure 7).

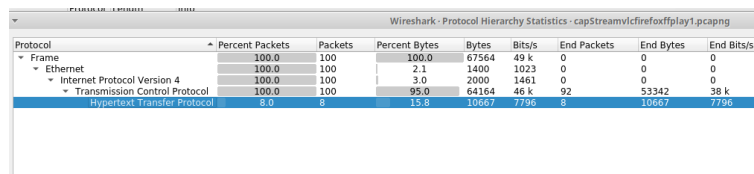
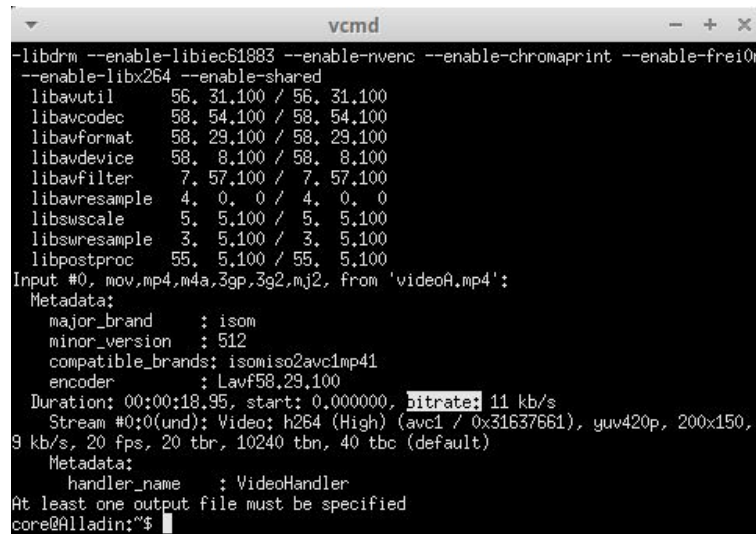


Figure 10: Encapsulamento (Protocol Hierarchy).

Assim, por um lado, podemos concluir que o aumento do número de utilizadores em concorrência nesta infraestrutura conduziu a uma redução da largura de banda. Esta redução de desempenho é resultante de uma arquitetura que tem como um ponto de único de falha o servidor VStreamer, o que gera um maior overhead a este componente (reduzido balanceamento de carga). Para além disso, não existe uma solução de streaming com bitrate adaptativo para as diversas condições da rede, de forma a que não se gerasse uma sobrecarga de rede.

De modo a complementarmos a nossa resposta, a taxa necessária teórica para a transmissão do vídeo A seria de 11 kb/s. Se executarmos o comando `ffmpeg -i videoA.mp4`, podemos verificar que a componente bitrate na figura 11.



```
vcmd
-ldrm --enable-libiec61883 --enable-nvenc --enable-chromaprint --enable-frei0r
--enable-libx264 --enable-shared
libavutil 56. 31.100 / 56. 31.100
libavcodec 58. 54.100 / 58. 54.100
libavformat 58. 29.100 / 58. 29.100
libavdevice 58.  8.100 / 58.  8.100
libavfilter 7. 57.100 / 7. 57.100
libavresample 4.  0.  0 / 4.  0.  0
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: 00:00:18.95, start: 0.000000, bitrate: 11 kb/s
Stream #0:0(und): Video: h264 (High) (avc1 / 0x31637661), yuv420p, 200x150,
9 kb/s, 20 fps, 20 tbr, 10240 tbn, 40 tbc (default)
Metadata:
  handler_name     : VideoHandler
At least one output file must be specified
core@Alladin:~$
```

Figure 11: Encapsulamento (Protocol Hierarchy).

2 Etapa 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.

Resposta: Em streaming com bitrate adaptativo existe uma tecnologia designada DASH(dynamic adaptive streaming over http) que consiste no representação de diferentes versões de bitrate do conteúdo do ficheiro de multimédia. Cada uma dessas representações está particionada em diferentes segmentos do vídeo. O cliente periodicamente vai verificando a largura de banda, sendo que escolhe segmentos com um bitrate máximo suportado pelo link. Esses URL's estão presentes no ficheiro de manifesto do servidor pelo que o cliente terá de fazer esse pedido ao servidor para obter essa informação. Logo, nesta etapa teremos de consultar nesse ficheiro a largura de banda necessária para receber o vídeo. Como podemos ver na figura 12, a bandwidth mínima para receber o vídeo com a menor resolução(200x150) possível é de 104352bps. Já para receber o vídeo com a maior resolução possível(640x480) a largura de banda mínima do link teria de ser igual ou superior a 475564bps. A pilha protocolar usada corresponde à camada da aplicação por HTTP.

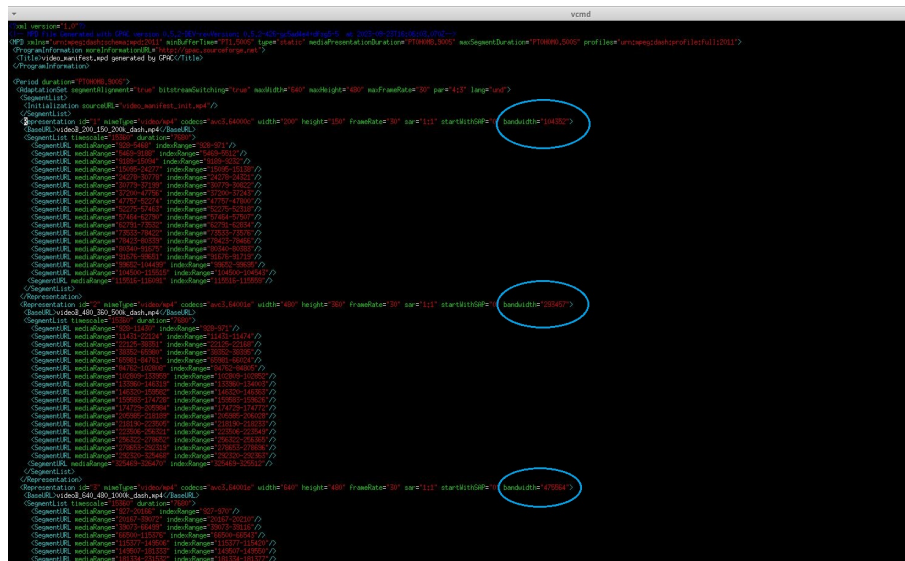


Figure 12: Ficheiro de manifesto

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.

Resposta: Após consultarmos o ficheiro de manifesto MPD do servidor, procedemos à verificação das larguras de banda necessárias para a transmissão de cada uma das versões dos vídeos. Como a Bela tinha de receber o vídeo com menor resolução alterámos a largura de banda do seu link para 100Kbps(o mínimo suficiente seria 104352bps), enquanto que para o Alladin este teria de receber o video com resolução máxima pelo que o seu link teria de suportar essa transmissão, logo alterámos a bandwith para 480Kbps (o mínimo suficiente seria 4775642bps).



Figure 13: Alterações na largura de banda dos links

Após efetuarmos essa alteração servimos o conteúdo com o servidor HTTP VStreamer, e colocámos cada um dos clientes a visualizar esse mesmo conteúdo. Com o auxílio do Wireshark à escuta na saída do servidor, filtrámos os pedidos GET que chegavam ao mesmo, pelo que foi possível identificar as versões dos vídeos solicitadas por cada um dos clientes de acordo com as suas condições de rede.

No.	Time	Source	Destination	Protocol	Length	Info
49	50.844293229	10.0.0.21	10.0.0.10	HTTP	425	GET /video_dash.html HTTP/1.1
50	52.476695381	10.0.0.21	10.0.0.10	HTTP	363	GET /dash.all.debug.js HTTP/1.1
3151	59.739414559	10.0.0.21	10.0.0.10	HTTP	379	GET /favicon.ico HTTP/1.1
3160	59.851995204	10.0.0.21	10.0.0.10	HTTP	426	GET /video_manifest.mpd HTTP/1.1
3177	62.455372567	10.0.0.21	10.0.0.10	HTTP	369	GET /video_manifest_init.mp4 HTTP/1.1
3186	63.144677445	10.0.0.21	10.0.0.10	HTTP	399	GET /videoB_640_480_1000k_dash.mp4 HTTP/1.1
3765	68.795958162	10.0.2.20	10.0.0.10	HTTP	379	GET /favicon.ico HTTP/1.1
3777	73.317478925	10.0.2.20	10.0.0.10	HTTP	399	GET /videoB_640_480_1000k_dash.mp4 HTTP/1.1
3878	76.113403584	10.0.2.20	10.0.0.10	HTTP	398	GET /videoB_200_150_200k_dash.mp4 HTTP/1.1

Frame 3777: 399 bytes on wire (3192 bits), 399 bytes captured (3192 bits) on interface veth9.0.c3, id 0	
Ethernet II, Src: 00:00:00:aa:00:03 (00:00:00:aa:00:03), Dst: 00:00:00:aa:00:00 (00:00:00:aa:00:00)	
Internet Protocol Version 4, Src: 10.0.2.20, Dst: 10.0.0.10	
Transmission Control Protocol, Src Port: 48980, Dst Port: 9999, Seq: 1, Ack: 1, Len: 333	
Hypertext Transfer Protocol	
GET /videoB_640_480_1000k_dash.mp4 HTTP/1.1\r\n	

Figure 14: Captura de tráfego no Wireshark

Como podemos observar, numa primeira instância o cliente Bela(10.0.2.20) fez um pedido GET do video com maior resolução (640*480), mas como o seu link não suportava essa transmissão, procedeu mais tarde a um pedido GET do video com resolução 200*150(entrada nº3878). Já o Alladin procedeu apenas a um pedido GET correspondente ao vídeo com maior resolução (entrada nº3186 da captura em cima apresentada).

Questão 4

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

Resposta: Como foi anteriormente referido DASH(dynamic adaptative streaming over http) consiste na representação de diferentes versões de bitrate de um ficheiro de multimédia(ex:vídeo). Cada uma dessas representações está dividida em diferentes segmentos/chunks de video. Toda essa informação encontra-se armazenada num ficheiro de manifesto que fornece os URL's para chunks diferentes. Assim, inicialmente, os clientes fazem o pedido ao servidor desse ficheiro de manifesto e, periodicamente, avaliam a largura de banda disponível. Deste modo, o cliente irá solicitando ao servidor o chunk que tenha um bitrate máximo para as condições de rede disponíveis do seu lado. Logo, a qualquer momento a representação do video que o cliente recebe poderá mudar, pelo que a qualidade do vídeo irá se ajustar/adaptar em conformidade com a largura de banda no momento.

3 Etapa 3

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.

Resposta: Para a realização desta etapa, alteramos a topologia usada de modo a ligar todos os hosts através de um switch.

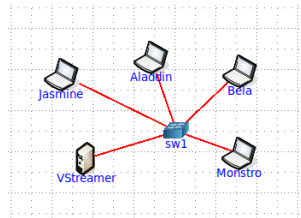


Figure 15: Topologia da etapa 3

Os cenários unicast têm como objetivo a transmissão de de informação de apenas uma fonte, para apenas um destino (1:1), enquanto no multicast, ainda que com apenas uma fonte, a informação pode ser recebida por múltiplos destinos. No nível de rede, o unicast acaba por ser um tipo de transmissão mais desvantajosa, uma vez que, caso seja necessário transmitir para mais do que um destino, haverá a necessidade de existirem N fluxos, uma vez que a ligação é de 1:1, ao contrário do multicast que apenas envia uma vez os dados, e posteriormente, os switches e routers replicam para os destinatários. Podemos comprovar estes factos, com a observação das seguintes duas capturas do wireshark em cenários de unicast e multicast.

7	0.245970631	10.0.0.10	10.0.2.21	UDP	1387	49647 → 5555	Len=1345
8	0.290262468	10.0.0.10	10.0.2.21	UDP	865	49647 → 5555	Len=823
9	0.341809920	10.0.0.10	10.0.2.21	UDP	149	49647 → 5555	Len=107
10	0.395634277	10.0.0.10	10.0.2.21	UDP	130	49647 → 5555	Len=88
11	0.437223271	10.0.0.10	10.0.2.21	UDP	1375	49647 → 5555	Len=1333
12	0.498280663	10.0.0.10	10.0.2.21	UDP	1514	49647 → 5555	Len=1472
13	0.498314970	10.0.0.10	10.0.2.21	UDP	1514	49647 → 5555	Len=1472
14	0.498322036	10.0.0.10	10.0.2.21	UDP	1514	49647 → 5555	Len=1472
15	0.498328115	10.0.0.10	10.0.2.21	UDP	1514	49647 → 5555	Len=1472
16	0.498333663	10.0.0.10	10.0.2.21	UDP	689	49647 → 5555	Len=647
17	0.539814838	10.0.0.10	10.0.2.21	UDP	659	49647 → 5555	Len=617
18	0.592965714	10.0.0.10	10.0.2.21	UDP	1289	49647 → 5555	Len=1247
19	0.636916412	10.0.0.10	10.0.2.21	UDP	991	49647 → 5555	Len=859
20	0.691306760	10.0.0.10	10.0.2.21	UDP	147	49647 → 5555	Len=105
21	0.746125154	10.0.0.10	10.0.2.21	UDP	137	49647 → 5555	Len=95
22	0.788512696	10.0.0.10	10.0.2.21	UDP	1353	49647 → 5555	Len=1311
23	0.841100090	10.0.0.10	10.0.2.21	UDP	684	49647 → 5555	Len=642
24	0.893498799	10.0.0.10	10.0.2.21	UDP	491	49647 → 5555	Len=449
25	0.936886587	10.0.0.10	10.0.2.21	UDP	166	49647 → 5555	Len=124
26	0.988928157	10.0.0.10	10.0.2.21	UDP	1004	49647 → 5555	Len=962
27	1.037034275	10.0.0.10	10.0.2.21	UDP	70	49648 → 5556	Len=28
28	1.037080338	10.0.0.10	10.0.2.21	UDP	674	49647 → 5555	Len=632
29	1.090141582	10.0.0.10	10.0.2.21	UDP	1514	49647 → 5555	Len=1472
30	1.090241249	10.0.0.10	10.0.2.21	UDP	1514	49647 → 5555	Len=1472

Figure 16: Captura unicast

No cenário em unicast o vídeo é enviado diretamente do VStreamer para o host Monstro, esta transmissão é feita em protocolo UDP (connectionless). Em ambos os cenários de transmissão é da responsabilidade da camada de aplicação fazer o controle de fluxos, de erros e de congestionamento.

6	0.143987493	10.0.0.10	224.0.0.200	UDP	1514	52620	→	5555	Len=1472
7	0.144067519	10.0.0.10	224.0.0.200	UDP	1514	52620	→	5555	Len=1472
8	0.144144063	10.0.0.10	224.0.0.200	UDP	400	52620	→	5555	Len=358
9	0.189464430	10.0.0.10	224.0.0.200	UDP	126	52620	→	5555	Len=84
10	0.259904192	10.0.0.10	224.0.0.200	UDP	122	52620	→	5555	Len=80
11	0.295995359	10.0.0.10	224.0.0.200	UDP	122	52620	→	5555	Len=80
12	0.348189014	10.0.0.10	224.0.0.200	UDP	122	52620	→	5555	Len=80
13	0.390281682	10.0.0.10	224.0.0.200	UDP	122	52620	→	5555	Len=80
14	0.440919787	10.0.0.10	224.0.0.200	UDP	122	52620	→	5555	Len=80
15	0.492774178	10.0.0.10	224.0.0.200	UDP	122	52620	→	5555	Len=80
16	0.541661488	10.0.0.10	224.0.0.200	UDP	122	52620	→	5555	Len=80
17	0.597249079	10.0.0.10	224.0.0.200	UDP	122	52620	→	5555	Len=80
18	0.642743585	10.0.0.10	224.0.0.200	UDP	122	52620	→	5555	Len=80
19	0.696156867	10.0.0.10	224.0.0.200	UDP	122	52620	→	5555	Len=80
20	0.740597077	10.0.0.10	224.0.0.200	UDP	1514	52620	→	5555	Len=1472
21	0.740738014	10.0.0.10	224.0.0.200	UDP	1514	52620	→	5555	Len=1472
22	0.740757081	10.0.0.10	224.0.0.200	UDP	1514	52620	→	5555	Len=1472
23	0.740771111	10.0.0.10	224.0.0.200	UDP	1514	52620	→	5555	Len=1472
24	0.740784492	10.0.0.10	224.0.0.200	UDP	400	52620	→	5555	Len=358
25	0.793882053	10.0.0.10	224.0.0.200	UDP	126	52620	→	5555	Len=84
26	0.839722817	10.0.0.10	224.0.0.200	UDP	122	52620	→	5555	Len=80
27	0.895777659	10.0.0.10	224.0.0.200	UDP	122	52620	→	5555	Len=80
28	0.939824594	10.0.0.10	224.0.0.200	UDP	122	52620	→	5555	Len=80
29	0.992379469	10.0.0.10	224.0.0.200	UDP	122	52620	→	5555	Len=80

Figure 17: Captura multicast

Por outro lado, no multicast, este apenas envia para um grupo multicast (neste caso o 224.0.0.200 porta 5555) em UDP, onde qualquer host consegue aceder desde que se encontre à escuta para este grupo. Com podemos observar com na figura 18, existe uma diferença significativa nos tempos de transmissão entre um cenário unicast e multicast, esta acontece devido ao facto de o multicast ser mais complexo que o unicast e servir um maior número de clientes e daí esta diferença de tempo. Por outro lado, se unicast for usado numa rede com um grande número de clientes, serão transmitidos cópias duplicadas de pacotes, consumindo um maior número de recursos e maior largura de banda. Como podemos visualizar na figura abaixo em unicast atinge-se um bitrate de 174kbps enquanto que em multicast atinge-se valores inferiores, 170kbps.

Ethernet - 2	IPv4 - 2	IPv6	TCP	UDP - 2									
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	49648	10.0.2.21	5555	17	732	17	732	0	0	0.000000	0.0000	174.1	174.1
10.0.0.10	49648	10.0.2.21	5556	1	70	1	70	0	0	1.037034	0.0000	—	—
Wireshark - Conversations - TCP-2-pcapng													

Ethernet - 2	IPv4 - 2	IPv6	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	52621	224.0.0.200	5555	272	2111	272	2111	0	0	0.000000	0.0000	170.1	170.1
10.0.0.10	38572	224.127.254.5	9875	2	732	2	732	0	0	4.045462	0.0486	1159	1159
10.0.0.10	52621	224.0.0.200	5556	2	140	2	140	0	0	4.045521	0.0486	221	221

Figure 18: Diferenças entre os métodos unicast e multicast

Deste modo, podemos concluir que o multicasat é mais eficiente em termos de escalabilidade, devido à sua menor utilização de recursos já que não há envio desnecessário de pacotes (os pacotes são encaminhados exactamente ao conjunto de destinatários pretendidos). Ainda que o unicast seja mais fácil de implementar existe um desperdício de largura de banda, logo a escolha de um ou outro, tudo irá depender do número de clientes e das necessidades do sistema.

4 Conclusões

Com a realização deste trabalho, foi possível perceber o funcionamento de um modo simplificado dos sistemas de streaming, bem como a compreensão da utilização dos mesmos, formatos de vídeo e ferramentas open source.