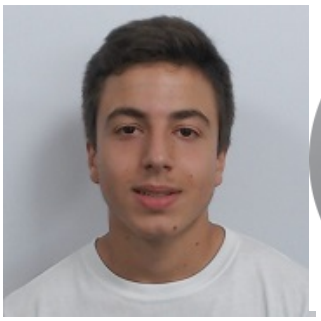
Universidade do Minho
Escola de Engenharia

Engenharia de Serviços em Rede

Streaming de áudio e vídeo a pedido e em tempo real

PL3 - Grupo 5
Luis Sousa a89597
Maria Barros pg47488
Pedro Barbosa pg47577



a89597

pg47488

pg47577

16 de novembro de 2021

Conteúdo

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)

Correndo o comando "ffmpeg -i video1.mp4 -hide_banner"obtemos informação relativa ao ficheiro "video1.mp4". Observando o *output* do comando, concluímos que a taxa em bps necessária (ou *bitrate*) é de 8 kb/s.

```
core@xubuncore:~/ESR/TP2$ ffmpeg -i video1.mp4 -hide_banner
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: 00:00:18.95, start: 0.000000, bitrate: 10 kb/s
    Stream #0:0(und): Video: h264 (High) (avc1 / 0x31637661), yuv420p, 180x120, 8 kb/s, 20 fps, 20 tbr, 10240 tbn, 40 tbc (default)
    Metadata:
      handler_name    : VideoHandler
```

Figura 1: Detalhes video1.mp4

Os 3 clientes de streaming usados neste exercício (VLC, Firefox e ffmpeg) usam HTTP Streaming. Desta forma os clientes estabelecem uma conexão TCP com o servidor que contém o vídeo guardado e enviam um pedido HTTP GET para o URL da stream. Depois, todos os pacotes são transmitidos via a conexão estabelecida, logo, o encapsulamento usado é TCP. [1]

Na figura seguinte podemos ver o estabelecimento da ligação entre o Portátil 1 e o Streamer nos frames 19 a 21, o pedido HTTP GET no frame 22 e a transmissão dos pacotes relativos ao conteúdo do vídeo nos restantes frames.

No.	Time	Source	Destination	Protocol	Length	Info
19	25.260690377	Portatil1	Streamer	TCP	78	32818 → 8080 [SYN] Seq=0 Win=64240
20	25.260699127	Streamer	Portatil1	TCP	74	8080 → 32818 [SYN, ACK] Seq=0 Ack=1
21	25.260708604	Portatil1	Streamer	TCP	66	32818 → 8080 [ACK] Seq=1 Ack=1 Win=
22	25.260754086	Portatil1	Streamer	HTTP	200	GET / HTTP/1.1
23	25.260758543	Streamer	Portatil1	TCP	66	8080 → 32818 [ACK] Seq=1 Ack=135 Wi
24	25.281782644	Streamer	Portatil1	TCP	169	8080 → 32818 [PSH, ACK] Seq=1 Ack=1
25	25.281807428	Portatil1	Streamer	TCP	66	32818 → 8080 [ACK] Seq=135 Ack=104
26	25.281821765	Streamer	Portatil1	TCP	1514	8080 → 32818 [ACK] Seq=104 Ack=135

Figura 2: Início da transmissão do Streamer para o Portátil 1

Para o cálculo do total de fluxos gerados no link de saída do servidor, decidimos filtrar os pacotes HTTP GET, pois, no HTTP Streaming, são estes que abrem a porta para o fluxo de pacotes TCP, do streamer para o cliente. Neste cenário, foram gerados 3 fluxos, um para cada portátil conectado à stream.

http

No.	Time	Source	Destination	Protocol	Length	Info
22	25.260754086	Portatil1	Streamer	HTTP	200	GET / HTTP/1.1
182	43.235692444	Portatil2	Streamer	HTTP	360	GET / HTTP/1.1
471	61.194692034	Portatil3	Streamer	HTTP	200	GET / HTTP/1.1

Figura 3: Pacotes HTTP GET recebidos pelo Streamer

De forma a perceber a escalabilidade desta solução de streaming, decidimos analisar o tráfego, com origem no Streamer, para as 3 fases (1 cliente, 2 clientes e 3 clientes).

Assim, usamos o filtro "ip.src == 10.0.0.10" para apenas serem apresentados os pacotes com origem no Streamer.

Para analisar o fluxo nas diferentes fases, usamos o filtro "frame.number >= x && frame.number < y" para captar apenas os pacotes relativos a cada uma delas. Depois, usamos a funcionalidade *Statistics -> Protocol Hierarchy* do Wireshark que nos apresenta informação relativa aos pacotes filtrados, permitindo-nos calcular o tráfego na rede.

Pela análise das três figuras seguintes, mais especificamente da coluna *Bits/s*, podemos observar que o tráfego TCP no link de saída do servidor é 44kbps, 84kbps e 121kbps para 1, 2 e 3 clientes conectados, respetivamente. Desta forma, podemos concluir que a taxa de pacotes enviados pelo servidor tem tendência a aumentar linearmente com o número de clientes conectados, neste caso, cerca de 40kbps por cliente.

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
Frame	100.0	73	100.0	98643	44k	0	0	0
Ethernet	100.0	73	1.0	1022	457	0	0	0
Internet Protocol Version 4	100.0	73	1.5	1460	653	0	0	0
Transmission Control Protocol	100.0	73	97.5	96161	43k	73	96161	43k

Figura 4: *Protocol Hierarchy* com 1 cliente

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
Frame	100.0	139	100.0	189088	84k	0	0	0
Ethernet	100.0	139	1.0	1946	871	0	0	0
Internet Protocol Version 4	100.0	139	1.5	2780	1244	0	0	0
Transmission Control Protocol	100.0	139	97.5	184362	82k	139	184362	82k

Figura 5: *Protocol Hierarchy* com 2 clientes

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
Frame	100.0	579	100.0	807995	121k	0	0	0
Ethernet	100.0	579	1.0	8106	1222	0	0	0
Internet Protocol Version 4	100.0	579	1.4	11580	1746	0	0	0
Transmission Control Protocol	100.0	579	97.6	788309	118k	579	788309	118k

Figura 6: *Protocol Hierarchy* com 3 clientes

Em baixo podemos ver o Streamer e os 3 clientes em execução simultânea.

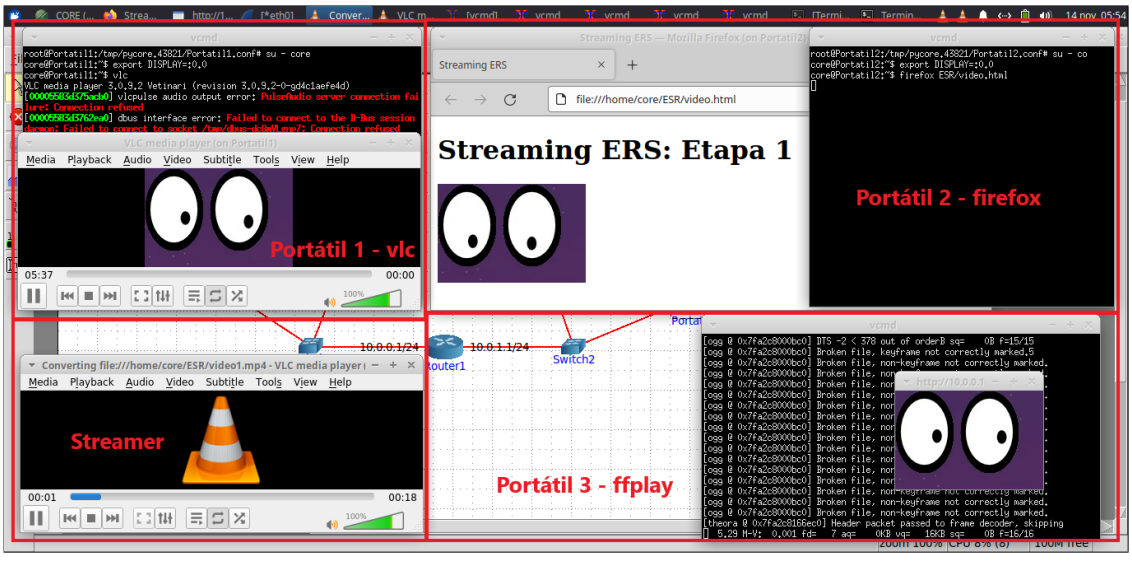


Figura 7: Streamer e 3 clientes em execução

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.

Observando os detalhes do ficheiro "video2.mp4", presentes na figura seguinte, podemos concluir que a largura de banda necessária para que o cliente de streaming consiga receber o vídeo no Firefox é 14 kbps.

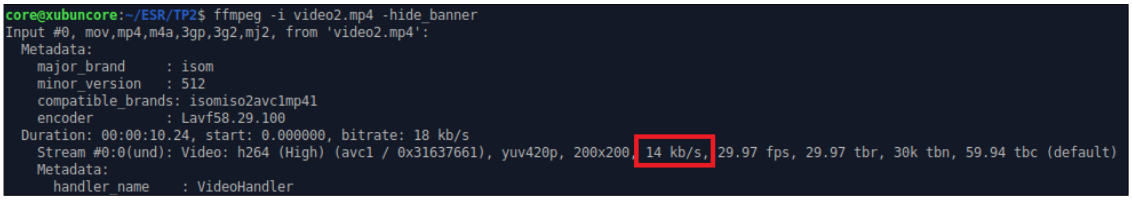


Figura 8: Detalhes video2.mp4

Em relação à pilha protocolar usada neste cenário (TCP/IP), podemos referir os protocolos mais relevantes, para cada uma das camadas.

- **Camada de aplicação:** nesta camada destaca-se o protocolo HTTP, pois é este que é usado para estabelecer a ligação entre o cliente e o servidor, através de um pedido HTTP GET.
- **Camada de transporte:** nesta camada destaca-se o protocolo TCP, maioritariamente usado para a transmissão dos dados do ficheiro em streaming. O pedido HTTP GET também está emcapsulado em TCP.
- **Camada de rede:** nesta camada destaca-se o protocolo IPv4.
- **Camada de ligação:** nesta camada destaca-se o protocolo Ethernet II.

Na figura seguinte, podemos ver os diferentes protocolos referidos em cima, num pacote de um pedido HTTP GET.

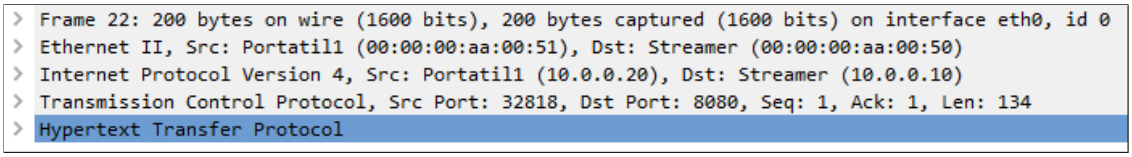


Figura 9: Frame de um pedido HTTP GET

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.

Na figura seguinte podemos observar os Portáteis 1 e 2 conectados à stream via DASH, fornecida pelo Streamer. Neste caso, o débito do link para o Portátil 2 foi diminuído, forçando-o a optar pelo vídeo com menor resolução, já o Portátil 1 consegue receber dados a uma taxa que lhe permite apresentar o vídeo de maior resolução.

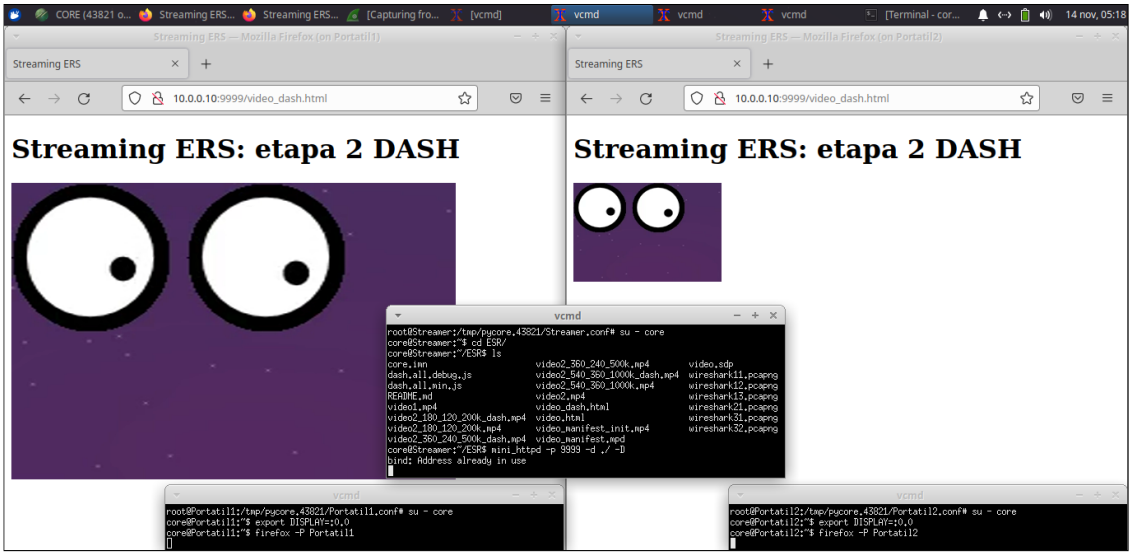


Figura 10: Streaming via DASH para dois portáteis

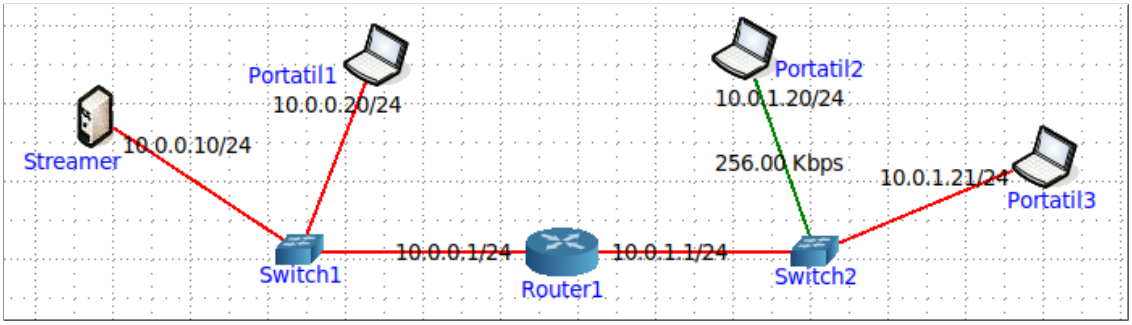


Figura 11: Topologia com débito para o Portátil 2 alterado

4 Questão 4

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

Para o DASH funcionar corretamente, é necessário criar várias versões do vídeo, com bitrates diferentes (quer seja alterando a sua resolução ou a taxa de frames), atribuindo um URL único para cada uma. Também é preciso criar um ficheiro de manifesto, onde são apresentadas várias informações em relação aos ficheiros de streaming. Destas informações destaca-se a largura de banda necessária para transmitir cada uma das versões do vídeo.

Estando estas pré-condições asseguradas, o cliente começa por requisitar o ficheiro de manifesto ao servidor de streaming e analisa as diferentes versões do vídeo. Depois, escolhe uma das versões tendo em conta a largura de banda disponível, enviando um pedido HTTP GET ao servidor. No entanto, o vídeo não é transferido todo de uma vez, sendo transferido em pedaços, permitindo ao cliente escolher a versão em tempo real, consoante a largura de banda disponível (esta pode variar ao longo da transmissão) e a quantidade de vídeo no buffer. Naturalmente, se o cliente tiver bastante vídeo em buffer ou se a largura de banda for elevada, este vai optar por uma versão com maior bitrate. De forma natural, também, se o cliente tiver pouco vídeo em buffer ou se a largura de banda for limitada, este vai escolher um pedaço de uma versão com menor bitrate. [1]

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.

Enquanto que no cenário *unicast* a informação é enviada de um ponto para outro existindo apenas um remetente e um recetor, no cenário *multicast* esta é enviada de um ou mais pontos para outros tantos havendo um ou mais remetentes e vários recetores, o que resulta num só fluxo, independentemente do número de clientes. [2]

Através da análise das figuras seguintes, podemos observar que a taxa de tráfego de dados em pacotes UDP (dados relativos ao streaming), no cenário *unicast*, varia consoante o número de clientes conectados à stream. Neste caso, o tráfego é de 79kbps e 163kbps para 1 e 3 clientes, respetivamente. Assim, conclui-se que cada host adicional aumento o tráfego na rede.

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes
Frame	100.0	254	100.0	128016	90 k	0	0
Ethernet	100.0	254	2.8	3556	2506	0	0
Internet Protocol Version 6	0.4	1	0.0	40	28	0	0
Open Shortest Path First	0.4	1	0.0	36	25	1	36
Internet Protocol Version 4	98.0	249	3.9	4980	3510	0	0
User Datagram Protocol	91.7	233	1.5	1864	1314	0	0
Data	87.0	221	88.6	113361	79 k	221	113361
ADwin configuration protocol	4.7	12	0.5	704	496	12	704
Open Shortest Path First	2.4	6	0.2	264	186	6	264
Internet Control Message Protocol	3.9	10	2.4	3099	2184	10	3099
Address Resolution Protocol	1.6	4	0.1	112	78	4	112

Figura 12: Cenário *unicast* com 1 cliente

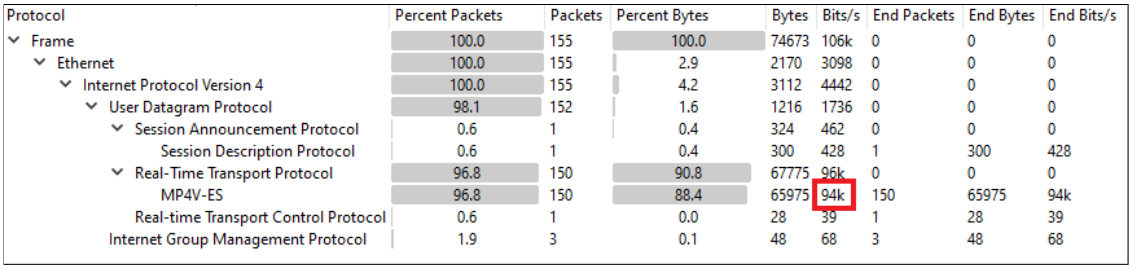
Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes
Frame	100.0	8648	100.0	4728897	179 k	0	0
Ethernet	100.0	8648	2.6	121072	4598	0	0
Internet Protocol Version 6	0.3	23	0.0	920	34	0	0
Open Shortest Path First	0.2	21	0.0	756	28	21	756
Internet Control Message Protocol v6	0.0	2	0.0	32	1	2	32
Internet Protocol Version 4	99.5	8601	3.6	172020	6533	0	0
User Datagram Protocol	97.1	8397	1.4	67176	2551	0	0
Data	92.8	8023	91.3	4317977	163 k	8023	4317977
ADwin configuration protocol	4.3	374	0.5	22580	857	374	22580
Open Shortest Path First	1.2	106	0.1	4664	177	106	4664
Internet Control Message Protocol	1.1	98	0.4	21028	798	98	21028
Address Resolution Protocol	0.3	24	0.0	672	25	24	672

Figura 13: Cenário *unicast* com 3 clientes

Já no cenário *multicast*, o tráfego mantém-se, independentemente do número de clientes, permitindo às aplicações escalar, isto é, trabalhar com grandes números de clientes. Isto confirma-se nas figuras seguintes, onde se observa que o tráfego de pacotes relativos ao vídeo em streaming (MP4V-ES) mantém-se perto dos 100kbps, tanto sem clientes como com 3 clientes.

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
Frame	100.0	191	100.0	97539	111k	0	0	0
Ethernet	100.0	191	2.7	2674	3058	0	0	0
Internet Protocol Version 4	100.0	191	3.9	3820	4369	0	0	0
User Datagram Protocol	100.0	191	1.6	1528	1747	0	0	0
Session Announcement Protocol	1.0	2	0.7	648	741	0	0	0
Session Description Protocol	1.0	2	0.6	600	686	2	600	686
Real-Time Transport Protocol	97.9	187	91.1	88813	101k	0	0	0
MP4V-ES	97.9	187	88.8	86569	99k	187	86569	99k
Real-time Transport Control Protocol	1.0	2	0.1	56	64	2	56	64

Figura 14: Cenário *multicast* sem clientes



Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
Frame	100.0	155	100.0	74673	106k	0	0	0
Ethernet	100.0	155	2.9	2170	3098	0	0	0
Internet Protocol Version 4	100.0	155	4.2	3112	4442	0	0	0
User Datagram Protocol	98.1	152	1.6	1216	1736	0	0	0
Session Announcement Protocol	0.6	1	0.4	324	462	0	0	0
Session Description Protocol	0.6	1	0.4	300	428	1	300	428
Real-Time Transport Protocol	96.8	150	90.8	67775	96k	0	0	0
MP4V-ES	96.8	150	88.4	65975	94k	150	65975	94k
Real-time Transport Control Protocol	0.6	1	0.0	28	39	1	28	39
Internet Group Management Protocol	1.9	3	0.1	48	68	3	48	68

Figura 15: Cenário *multicast* com 3 clientes

Assim, concluímos que o cenário *multicast* é vantajoso para streaming com vários clientes, pois o tráfego na rede é constante, independentemente do número de clientes. Por esta mesma razão, este cenário pode ser desfavorável, caso estejam poucos ou nenhuns clientes conectados à stream.

6 Conclusões

No âmbito deste trabalho prático abordamos o tema *Streaming de áudio e vídeo a pedido e em tempo real* com o qual acreditamos ter cumprido todos os objetivos propostos pelos docentes da unidade curricular.

Consideramos que este projeto teve especial relevo, na medida em que enriqueceu o nosso conhecimento na perceção da quantidade de largura de banda necessária para um *streamer* receber um vídeo, correlacionar o ajuste de débitos de links com a resolução dos vídeos em *streaming*, compreender as principais diferenças entre os cenários *unicast* e *multicast* focando, posteriormente, no cenário *multicast* ao nível da rede, analisar, em termos de escalabilidade, o tráfego no *link* de saída de um servidor com 1, 2 ou 3 clientes e aperfeiçoar as bases relativamente à forma como o *DASH* funciona através de um ficheiro *MPD*.

Em suma, o grupo considera que o propósito deste trabalho foi alcançado com sucesso, visto que intensificamos o nosso conhecimento sobre os temas em cima referidos.

Referências

[1] Kurose, J., & Ross, K. (2017). Computer Networking: A Top-Down Approach (7th ed.). Pearson.

[2] https://www.erg.abdn.ac.uk/users/gorry/course/intro-pages/uni-b-mcast.html