



# Universidade do Minho

Licenciatura em Engenharia Informática

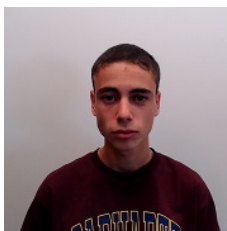
## Redes de Computadores

Trabalho Prático 3

Grupo 34



Diogo Rebelo  
(A93278)



Hugo Brandão  
(A93287)



Gonçalo Freitas  
(A93297)

20 de junho de 2023

## Conteúdo

<b>1</b>	<b>Questões e Respostas</b>	<b>3</b>
1	Questão nº 3: Captura e análise de Tramas Ethernet . . . . .	3
1.1	Alínea 1 . . . . .	4
1.2	Alínea 2 . . . . .	4
1.3	Alínea 3 . . . . .	5
1.4	Alínea 4 . . . . .	5
1.5	Alínea 5 . . . . .	7
1.6	Alínea 6 . . . . .	7
1.7	Alínea 7 . . . . .	8
2	Questão nº 4: Protocolo ARP . . . . .	9
2.1	Alínea 8 . . . . .	9
2.2	Alínea 9 . . . . .	9
2.3	Alínea 10 . . . . .	10
2.4	Alínea 11 . . . . .	10
2.5	Alínea 12 . . . . .	10
2.6	Alínea 13 . . . . .	11
2.7	Alínea 14 . . . . .	12
3	Questão 5 . . . . .	14
3.1	Alínea 15 . . . . .	14
3.2	Alínea 16 . . . . .	16
<b>2</b>	<b>Conclusão</b>	<b>20</b>

## Lista de Figuras

1	Trecho da captura de tráfego no acesso ao url especificado. . . . .	4
2	Informações Ethernet sobre a mensagem HTTP GET. . . . .	4
3	Informações mais detalhadas sobre a mensagem HTTP GET. . . . .	5
4	Informação alusiva aos tamanhos necessários da mensagem HTTP GET. . .	6
5	Informação alusiva à mensagem de resposta. . . . .	7
6	Tabela ARP . . . . .	9
7	Tabela ARP . . . . .	9
8	Tabela ARP . . . . .	10
9	Tabela ARP . . . . .	11
10	Topologia . . . . .	12
11	Tcpdump em LAN comutada . . . . .	14

12	Tcpdump em LAN partilhada . . . . .	15
13	Departamento B com interfaces identificadas . . . . .	16
14	Comando <code>ping 192.168.34.164</code> . . . . .	18
15	Comando <code>ping 192.168.34.163</code> . . . . .	18
16	Captura de tráfego no Departamento B, na Jasmine. . . . .	18
17	Captura de tráfego no Departamento B, no Servidor B. . . . .	19

## Lista de Tabelas

1	Tabela de Mensagens com formato $\langle ordem \rangle - \langle tipo\_mensagem \rangle$ . . . .	12
2	Tabela final de endereços MAC do Departamento B. . . . .	19

# 1. Questões e Respostas

## Questão nº 3: Captura e análise de Tramas Ethernet

Assegure-se que a cache do seu browser está vazia. Ative o Wireshark na sua máquina nativa. No seu browser, acesse ao URL <https://elearning.uminho.pt>. Pare a captura do Wireshark e proceda da seguinte forma:

1. Localize o estabelecimento da conexão entre o cliente e o servidor HTTP (sequência de tramas com as TCP flags TCP SYN, SYNACK, ACK ativas);
2. Após a fase de estabelecimento seguro da conexão, obtenha o número de ordem da sequência de bytes capturada (coluna da esquerda na janela do Wireshark) correspondente à trama que transporta os primeiros dados aplicativos enviados do cliente para o servidor (Application Data);
3. Identifique também o número de ordem da trama com a resposta proveniente do servidor que contém os dados correspondentes ao acesso web realizado pelo cliente (browser);
4. [...] Expanda a informação do nível da ligação de dados e observe o conteúdo da trama Ethernet (cabeçalho e dados (payload)).

Responda às perguntas seguintes com base no conteúdo da trama Ethernet que contém a mensagem de acesso ao servidor (HTTP GET encriptada).

Como sugerido, iniciamos a captura no Wireshark, abrimos o Browser e acedemos ao link solicitado. Aquando da captura, não conseguimos identificar a mensagem de acesso ao servidor (HTTP GET) diretamente, já que esta se encontrava encriptada, através da utilização do protocolo TLS. Poderíamos utilizar equivalentemente uma mensagem que mostrasse este acesso, como a de “Client Hello”, todavia, optamos por aceder ao um url diferente: <https://cbslocal.com/>, sugerido pelo docente.

Então, voltamos a iniciar a captura, acedemos ao site em questão, paramos a captura e agora conseguimos identificar de imediato a mensagem que pretendíamos.

Seguindo a metodologia, identificamos a sequência de tramas que evidencia o início do estabelecimento da conexão HTTP e encontramos a mensagem procurada. Uma forma de o encontrar de imediato é através da aplicação de um filtro ao protocolo HTTP.

Abaixo surge selecionada a mensagem HTTP GET, sendo através dela que se respondem às questões seguintes. As informações dessa mensagem surgem de seguida, na respetiva alínea.

No.	Time	Source	Destination	Protocol	Length	Info
559	15.092431896	193.137.16.65	172.26.33.158	DNS	99	Standard query response 0xb77e A cbslocal.co
560	15.093471662	172.26.33.158	192.0.66.136	TCP	74	39714 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1
561	15.094022171	172.26.33.158	192.0.66.136	TCP	74	39716 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1
562	15.111605462	192.0.66.136	172.26.33.158	TCP	66	80 → 39714 [SYN, ACK] Seq=0 Ack=1 Win=29200
563	15.111712948	172.26.33.158	192.0.66.136	TCP	54	39714 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0
564	15.111857916	192.0.66.136	172.26.33.158	TCP	66	80 → 39716 [SYN, ACK] Seq=0 Ack=1 Win=29200
565	15.111895528	172.26.33.158	192.0.66.136	TCP	54	39716 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0
566	15.112205996	172.26.33.158	192.0.66.136	HTTP	496	GET / HTTP/1.1
567	15.129225143	192.0.66.136	172.26.33.158	TCP	54	80 → 39714 [ACK] Seq=1 Ack=443 Win=30720 Len
568	15.129477347	192.0.66.136	172.26.33.158	HTTP	405	HTTP/1.1 301 Moved Permanently (text/html)
569	15.129502096	172.26.33.158	192.0.66.136	TCP	54	39714 → 80 [ACK] Seq=443 Ack=352 Win=64128 L
570	15.133710401	172.26.33.158	142.250.200.138	UDP	151	36591 → 443 Len=109
571	15.135216307	172.26.33.158	192.0.66.136	TCP	74	38260 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=

Figura 1: Trecho da captura de tráfego no acesso ao url especificado.

## Alínea 1

| Anote os endereços MAC de origem e de destino da trama capturada.

Com a mensagem selecionada, analisamos os detalhes na janela abaixo e verificamos os endereços MAC associados:

- Endereço MAC de Origem: e8:d0:fc:ce:46:37
- Endereço MAC de Destino: 00:d0:03:ff:94:00

Segue-se a informação relativa à mensagem que o comprova:

```

No.      Time      Source      Destination      Protocol Length Info
 566 15.112205996 172.26.33.158 192.0.66.136      HTTP      496      GET / HTTP/1.1
Frame 566: 496 bytes on wire (3968 bits), 496 bytes captured (3968 bits) on interface wlp2s0, id 0
Ethernet II, Src: LiteonTe_ce:46:37 (e8:d0:fc:ce:46:37), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
  Destination: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
  Source: LiteonTe_ce:46:37 (e8:d0:fc:ce:46:37)
  Type: IPv4 (0x0800)
Internet Protocol Version 4, Src: 172.26.33.158, Dst: 192.0.66.136
Transmission Control Protocol, Src Port: 39714, Dst Port: 80, Seq: 1, Ack: 1, Len: 442
Hypertext Transfer Protocol

```

Figura 2: Informações Ethernet sobre a mensagem HTTP GET.

## Alínea 2

| Identifique a que sistemas se referem. Justifique.

Tendo em conta o que é um endereço MAC (endereço/identificador único associado à interface de comunicação que conecta um dispositivo à respetiva rede), o endereço MAC de Origem é o endereço da NIC (Network Interface Controller) associada ao dispositivo de origem, que é, neste caso, o nosso computador pessoal. Já o endereço MAC de Destino é o endereço da NIC associada ao dispositivo de destino, que é, neste caso, o servidor da rede local [1].

### Alínea 3

| Qual o valor hexadecimal do campo Type da trama Ethernet? O que significa?

Como é visível na figura 2, o valor do campo *Type* é 0x0800 e este valor corresponde ao protocolo IP, mais propriamente, indica que a trama em questão transporta datagramas IPv4. Este valor indica a camada para a qual o payload do pacote da camada Ethernet será passado (IP)[2].

### Alínea 4

| Quantos bytes são usados no encapsulamento protocolar, i.e. desde o início da trama até ao início dos dados do nível aplicacional (Application Data Protocol: http-over-tls)? Calcule e indique, em percentagem, a sobrecarga (overhead) introduzida pela pilha protocolar.

```
No.      Time          Source          Destination      Protocol Length Info
566 15.112205996 172.26.33.158 192.0.66.136    HTTP      496    GET / HTTP/1.1
Frame 566: 496 bytes on wire (3968 bits), 496 bytes captured (3968 bits) on interface wlp2s0, id 0
Ethernet II, Src: LiteonTe_ce:46:37 (e8:d0:fc:ce:46:37), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
Internet Protocol Version 4, Src: 172.26.33.158, Dst: 192.0.66.136
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 482
  Identification: 0x07cd (1997)
  Flags: 0x4000, Don't fragment
  Fragment offset: 0
  Time to live: 64
  Protocol: TCP (6)
  Header checksum: 0x6108 [validation disabled]
  [Header checksum status: Unverified]
  Source: 172.26.33.158
  Destination: 192.0.66.136
  Transmission Control Protocol, Src Port: 39714, Dst Port: 80, Seq: 1, Ack: 1, Len: 442
  Hypertext Transfer Protocol
```

Figura 3: Informações mais detalhadas sobre a mensagem HTTP GET.

Sabemos que, em termos de camadas, temos a seguinte estrutura, analisando a figura 3:

1. Layer 7 – Application Layer : HTTP
2. Layer 5 – Transport Layer : TCP
3. Layer 3 – Network Layer : IP
4. Layer 2 – Data Link Layer : Ethernet
5. Layer 1 – Physical Layer : Frame

```

▶ Frame 566: 496 bytes on wire (3968 bits), 496 bytes captured (3968 bits) on interface wlp2s0, id 0
▶ Ethernet II, Src: LiteonTe_ce:46:37 (e8:d0:fc:ce:46:37), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
▼ Internet Protocol Version 4, Src: 172.26.33.158, Dst: 192.0.66.136
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    ▶ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
      Total Length: 482
      Identification: 0x07cd (1997)
    ▶ Flags: 0x4000, Don't fragment
      Fragment offset: 0
      Time to live: 64
      Protocol: TCP (6)
      Header checksum: 0x6108 [validation disabled]
      [Header checksum status: Unverified]
      Source: 172.26.33.158
      Destination: 192.0.66.136
▼ Transmission Control Protocol, Src Port: 39714, Dst Port: 80, Seq: 1, Ack: 1, Len: 442
    Source Port: 39714
    Destination Port: 80
    [Stream index: 1]
    [TCP Segment Len: 442]
    Sequence number: 1 (relative sequence number)
    Sequence number (raw): 1420962871
    [Next sequence number: 443 (relative sequence number)]
    Acknowledgment number: 1 (relative ack number)
    Acknowledgment number (raw): 337687178
    0101 .... = Header Length: 20 bytes (5)
    ▶ Flags: 0x018 (PSH, ACK)
      Window size value: 502
      [Calculated window size: 64256]
      [Window size scaling factor: 128]
      Checksum: 0xd215 [unverified]
      [Checksum Status: Unverified]
      Urgent pointer: 0
    ▶ [SEQ/ACK analysis]
    ▶ [Timestamps]
    TCP payload (442 bytes)
▶ Hypertext Transfer Protocol

```

Figura 4: Informação alusiva aos tamanhos necessários da mensagem HTTP GET.

Através da figura anterior, conseguimos extrair vários valores:

- Frame Length: 496 Bytes;
- IP Total Length: 482 Bytes;

- IP Total Length = IP Header Length + TCP Header Length + Application = 482 Bytes;
- TCP Payload = 442 Bytes

Assim, sabendo o TCP payload, basta ir ao tamanho total da trama e subtrair esse valor e o dos cabeçalhos:

$$FrameLength - TCPPayload - IPHeader - TCPHeader$$

496 - 442 - 20 - 20 = 14 Bytes. O que nos dá um overhead de  $14/496 = 0,028225806$  (aproximadamente 2,8 % (1cd))

## Alínea 5

Qual é o endereço Ethernet da fonte? A que sistema de rede corresponde? Justifique.

Analisando a informação da mensagem de resposta, temos:

```
No.      Time          Source          Destination      Protocol Length Info
568 15.129477347 192.0.66.136    172.26.33.158    HTTP      405      HTTP/1.1 301 Moved Permanently (text/html)
Frame 568: 405 bytes on wire (3240 bits), 405 bytes captured (3240 bits) on interface wlp2s0, id 0
Ethernet II, Src: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00), Dst: LiteonTe_ce:46:37 (e8:d0:fc:ce:46:37)
Destination: LiteonTe_ce:46:37 (e8:d0:fc:ce:46:37)
Address: LiteonTe_ce:46:37 (e8:d0:fc:ce:46:37)
.... 0. .... = LG bit: Globally unique address (factory default)
.... 0. .... = IG bit: Individual address (unicast)
Source: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
Address: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
.... 0. .... = LG bit: Globally unique address (factory default)
.... 0. .... = IG bit: Individual address (unicast)
Type: IPv4 (0x0800)
Internet Protocol Version 4, Src: 192.0.66.136, Dst: 172.26.33.158
Transmission Control Protocol, Src Port: 80, Dst Port: 39714, Seq: 1, Ack: 443, Len: 351
Hypertext Transfer Protocol
Line-based text data: text/html (7 lines)
```

Figura 5: Informação alusiva à mensagem de resposta.

O endereço da fonte é 00:d0:03:ff:94:00 e corresponde ao servidor, o qual envia a resposta ao pedido feito pelo computador pessoal, o endereço coincide inclusive com o endereço MAC de destino na trama HTTP GET.

## Alínea 6

Qual é o endereço MAC do destino? A que sistema corresponde?

De modo análogo, olhando para a figura 4, o endereço MAC do destino é e8:d0:fc:ce:46:37, correspondendo ao nosso computador pessoal.



**Alínea 7**

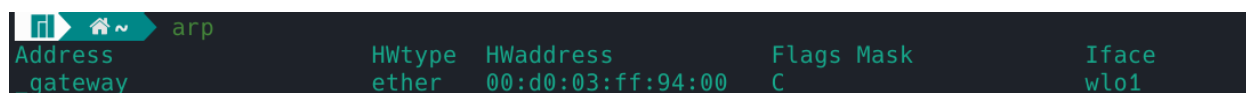
Atendendo ao conceito de desencapsulamento protocolar, identifique os vários protocolos contidos na trama recebida.

Observando a figura 4, identificamos 4 protocolos: Ethernet, IPv4, TCP e HTTP.

## Questão nº 4: Protocolo ARP

### Alínea 8

Observe o conteúdo da tabela ARP. Diga o que significa cada uma das colunas.



Address	HWtype	HWaddress	Flags	Mask	Iface
gateway	ether	00:d0:03:ff:94:00	C		wlo1

Figura 6: Tabela ARP

A coluna *Adress* é a coluna referente aos endereços IP ou alternativamente o nome do *host*, neste caso a entrada "\_gateway" significa o IP da máquina. A coluna *HWtype* indica o tipo da conexão, que neste caso é *ethernet*. A seguinte (*HWaddress*) indica o endereço MAC. A coluda *Flags*, "C" (Complete) significa que a entrada foi obtida dinamicamente através do protocolo ARP, ou seja, a conexão foi obtida com sucesso. A última coluna *Iface* significa "Interface" e representa a porta do sistema pela qual a conexão é feita.

### Alínea 9

Qual é o valor hexadecimal dos endereços origem e destino na trama Ethernet que contém a mensagem com o pedido ARP (ARP Request)? Como interpreta e justifica o endereço destino usado?

De forma a conseguir obter pedidos ARP mais facilmente utilizamos a seguinte topologia no *core* dentro da máquina virtual:

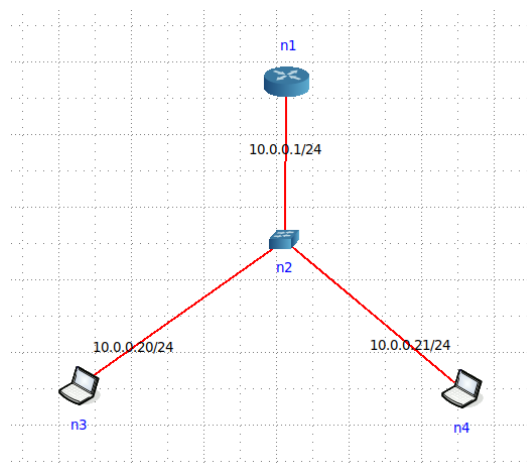
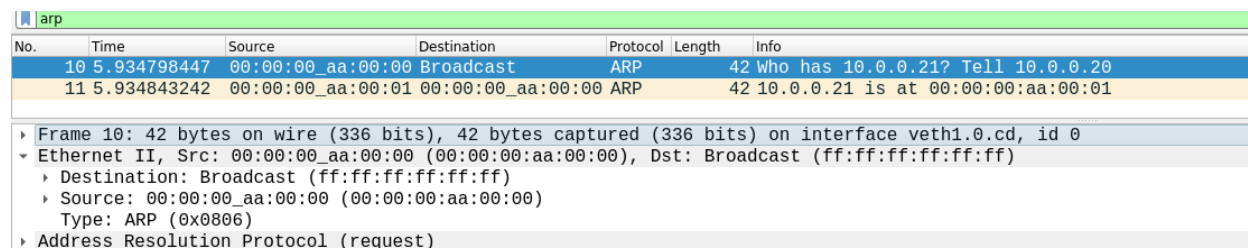


Figura 7: Tabela ARP

De forma a observar o pedido ARP efetuamos um ping a partir do Portátil *n3* para o Portátil *n4* (10.0.0.21). Obtendo no *Wireshark* as seguintes entradas.



The image shows a Wireshark packet capture window with the filter 'arp'. It displays two packets. Packet 10 is an ARP request (Type: 0x0806) from source 00:00:00:aa:00:00 to destination Broadcast (ff:ff:ff:ff:ff:ff). Packet 11 is an ARP response (Type: 0x0800) from source 00:00:00:aa:00:01 to destination 00:00:00:aa:00:00. The details pane for packet 10 is expanded, showing Ethernet II, Destination: Broadcast, Source: 00:00:00:aa:00:00, Type: ARP (0x0806), and Address Resolution Protocol (request).

No.	Time	Source	Destination	Protocol	Length	Info
10	5.934798447	00:00:00_aa:00:00	Broadcast	ARP	42	Who has 10.0.0.21? Tell 10.0.0.20
11	5.934843242	00:00:00_aa:00:01	00:00:00_aa:00:00	ARP	42	10.0.0.21 is at 00:00:00:aa:00:01

Frame 10: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface veth1.0.cd, id 0  
 Ethernet II, Src: 00:00:00\_aa:00:00 (00:00:00:aa:00:00), Dst: Broadcast (ff:ff:ff:ff:ff:ff)  
 Destination: Broadcast (ff:ff:ff:ff:ff:ff)  
 Source: 00:00:00\_aa:00:00 (00:00:00:aa:00:00)  
 Type: ARP (0x0806)  
 Address Resolution Protocol (request)

Figura 8: Tabela ARP

Tal como conseguimos verificar acima, os endereços são:

- Origem: 00:00:00:aa:00:00
- Destino: ff:ff:ff:ff:ff:ff

O endereço de destino é este, pois como o a tabela ARP estava vazia, o Portátil *n3* não conhecia o IP fornecido, ou seja, necessitava de enviar a todos os nós da rede local a quem pertencia o IP dado.

### Alínea 10

| Qual o valor hexadecimal do campo tipo da trama Ethernet? O que indica?

Tal como é possível verificar na Figura 8, o tipo da trama é 0x0806, que indica que esta é uma mensagem do protocolo ARP

### Alínea 11

| Como pode confirmar que se trata efetivamente de um pedido ARP? Identifique que tipo de endereços estão contidos na mensagem ARP? Que conclui?

Trata-se de um pedido ARP, já que o valor o Opcode é 1 (request) e os endereços contidos na mensagem ARP são endereços MAC.

### Alínea 12

| Explícite que tipo de pedido ou pergunta é feita pelo host de origem.

O host de origem pergunta a todos os nós da sua rede se o seu IP é igual ao que está à procura, em caso afirmativo quer saber o seu endereço MAC.

### Alínea 13

Localize a mensagem ARP que é a resposta ao pedido ARP efetuado.

11 5.934843242	00:00:00_aa:00:01 00:00:00_aa:00:00 ARP	42 10.0.0.21 is at 00:00:00_aa:00:01
▶ Frame 11: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface veth1.0.cd, id 0 ▶ Ethernet II, Src: 00:00:00_aa:00:01 (00:00:00_aa:00:01), Dst: 00:00:00_aa:00:00 (00:00:00_aa:00:00) ▼ Address Resolution Protocol (reply) Hardware type: Ethernet (1) Protocol type: IPv4 (0x0800) Hardware size: 6 Protocol size: 4 Opcode: reply (2) Sender MAC address: 00:00:00_aa:00:01 (00:00:00_aa:00:01) Sender IP address: 10.0.0.21 Target MAC address: 00:00:00_aa:00:00 (00:00:00_aa:00:00) Target IP address: 10.0.0.20		

Figura 9: Tabela ARP

| Qual o valor do campo ARP opcode? O que especifica?

O valor do campo ARP *opcode* é 2, e especifica que é uma mensagem do tipo *reply*.

| Em que campo da mensagem ARP está a resposta ao pedido ARP?

A resposta ao pedido ARP está no campo *Sender MAC address*, sendo neste caso "00:00:00:aa:00:01".

### Alínea 14

Na situação em que efetua um ping a outro host, assuma que este está diretamente ligado ao mesmo router, mas noutra subrede, e que todas as tabelas ARP se encontram inicialmente vazias. Esboce um diagrama em que indique claramente, e de forma cronológica, todas as mensagens ARP e ICMP trocadas, até à recepção da resposta ICMP do host destino.

Tendo em conta a seguinte topologia como orientação:

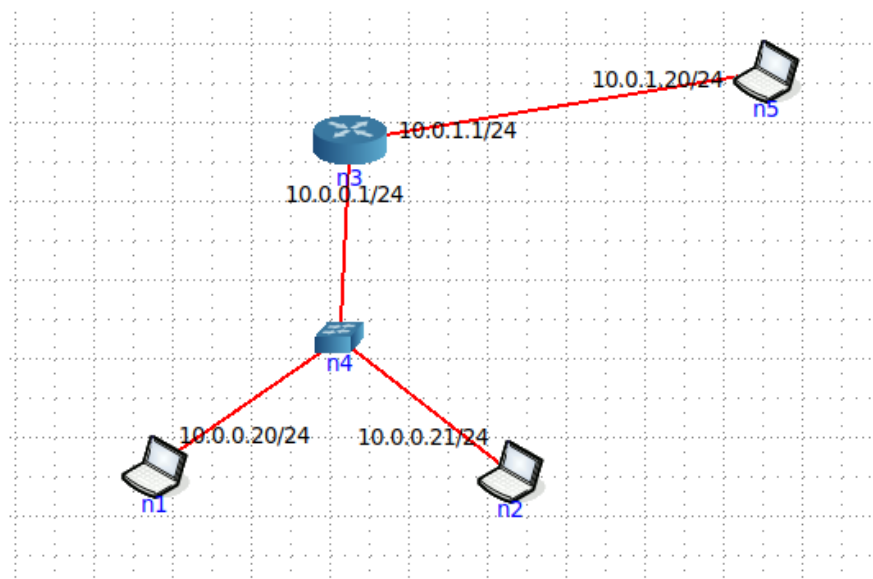


Figura 10: Topologia

E tendo em conta um *ping* do Portátil *n1* para 10.0.1.20 (Portátil *n5*), nas condições referidas no enunciado, chegamos à seguinte tabela cronológica de mensagens:

De \ Para	n1	n2	n3	n4	n5
n1	_____			1 - ARP REQUEST 5 - PING REQUEST	
n2		_____			
n3			_____	3 - ARP REPLY 11 - PING REPLY	7 - ARP REQUEST 9 - PING REQUEST
n4	4 - ARP REPLY 12 - PING REPLY	2 - ARP REQUEST	2 - ARP REQUEST 6 - PING REQUEST	_____	
n5			8 - ARP REPLY 10 - PING REPLY		_____

Tabela 1: Tabela de Mensagens com formato *< ordem > – < tipo\_mensagem >*

Sendo assim, inicialmente *n2* consulta a sua tabela de encaminhamento, a qual revela que o IP dado é exterior à sua rede, ou seja, o *ping* terá de ser encaminhado ao router (pois é

este que tem contacto com a restante subrede), desta forma é enviado um pedido ARP para todos os nós da sua subrede de forma a conhecer o endereço MAC do nó com endereço IP 10.0.0.1. Quando o pedido chega ao router este envia uma resposta ARP de volta ao Portátil *n1* informando o seu endereço MAC. A partir do momento que a tabela ARP foi preenchida, é enviado o *ping* para o *router*. Ao chegar ao *router* este envia um pedido ARP para todos os nós da outra subrede (neste caso só existe um) de forma a conhecer o endereço MAC do nó com endereço IP 10.0.1.20. O pedido ao chegar a *n5* é respondido, e após isso o *router* envia o *ping* a *n5*. Resta agora fazer o caminho inverso.

## Questão 5

### Alínea 15

Através da opção tcpdump verifique e compare como flui o tráfego nas diversas interfaces do dispositivo de interligação no departamento A (LAN partilhada) e no departamento B (LAN comutada) quando se gera tráfego intra-departamento (por exemplo, fazendo ping IPaddr da Bela para Monstro, da Jasmine para o Alladin, etc.). Que conclui?

```

root@Jasmine:/tmp/pycore.33599/Jasmine.conf# tcpdump
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
^C11:19:01.789519 IP6 fe80::200:ff:feaa:9 > ff02::5: OSPFv3, Hello, length 36
11:19:01.891051 IP 192.168.34.161 > 224.0.0.5: OSPFv2, Hello, length 44
11:19:02.069756 IP6 fe80::200:ff:feaa:f > ip6-allrouters: ICMP6, router solicitation, length 16
11:19:02.070073 IP6 fe80::3ced:81ff:fe3a:1b07 > ip6-allrouters: ICMP6, router solicitation, length 16
11:19:02.280642 IP6 fe80::3ced:81ff:fe3a:1b07.wdns > ff02::fb.wdns: 0 [2q] PTR (QM)? _ipps._tcp.local. PTR (QM)? _ipp._tcp.local. (45)
11:19:03.812419 ARP, Request who-has 192.168.34.162 tell 192.168.34.164, length 28
11:19:03.812474 ARP, Reply 192.168.34.162 is-at 00:00:00:aa:00:0f (oui Ethernet), length 28
11:19:03.813163 IP 192.168.34.164 > 192.168.34.162: ICMP echo request, id 27, seq 1, length 64
11:19:03.813179 IP 192.168.34.162 > 192.168.34.164: ICMP echo reply, id 27, seq 1, length 64
11:19:03.890869 IP 192.168.34.161 > 224.0.0.5: OSPFv2, Hello, length 44
11:19:04.813699 IP 192.168.34.164 > 192.168.34.162: ICMP echo request, id 27, seq 2, length 64
11:19:04.813724 IP 192.168.34.162 > 192.168.34.164: ICMP echo reply, id 27, seq 2, length 64
11:19:05.890932 IP 192.168.34.161 > 224.0.0.5: OSPFv2, Hello, length 44

13 packets captured
13 packets received by filter
0 packets dropped by kernel
root@Jasmine:/tmp/pycore.33599/Jasmine.conf#

root@SB:/tmp/pycore.33599/SB.conf# tcpdump
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
^C11:19:02.941042 IP6 fe80::e43c:b5ff:fe6e:c61e.wdns > ff02::fb.wdns: 0 [2q] PTR (QM)? _ipps._tcp.local. PTR (QM)? _ipp._tcp.local. (45)
11:19:03.812421 ARP, Request who-has 192.168.34.162 tell 192.168.34.164, length 28
11:19:03.890867 IP 192.168.34.161 > 224.0.0.5: OSPFv2, Hello, length 44
11:19:05.890930 IP 192.168.34.161 > 224.0.0.5: OSPFv2, Hello, length 44

4 packets captured
4 packets received by filter
0 packets dropped by kernel
root@SB:/tmp/pycore.33599/SB.conf#

root@Alladin:/tmp/pycore.33599/Alladin.conf# ping 192.168.34.162
PING 192.168.34.162 (192.168.34.162) 56(84) bytes of data:
64 bytes from 192.168.34.162: icmp_seq=1 ttl=64 time=1.69 ms
64 bytes from 192.168.34.162: icmp_seq=2 ttl=64 time=0.211 ms
^C
--- 192.168.34.162 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/ndev = 0.211/0.348/1.686/0.737 ms
root@Alladin:/tmp/pycore.33599/Alladin.conf#

```

Figura 11: Tcpdump em LAN comutada

```

root@SA:/tmp/pycore.33599/SA.conf# tcpdump
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
^C11:16:58.398143 IP 10.0.4.1 > 224.0.0.5: OSPFv2, Hello, length 44
11:16:58.496261 IP6 fe80::200:ff:feaa:18 > ff02::5: OSPFv3, Hello, length 36
11:16:59.714452 ARP, Request who-has 10.0.4.20 tell 10.0.4.21, length 28
11:16:59.714498 ARP, Reply 10.0.4.20 is-at 00:00:00:aa:00:17 (oui Ethernet), length 28
11:16:59.714509 IP 10.0.4.21 > 10.0.4.20: ICMP echo request, id 27, seq 1, length 64
11:16:59.714527 IP 10.0.4.20 > 10.0.4.21: ICMP echo reply, id 27, seq 1, length 64
11:17:00.398445 IP 10.0.4.1 > 224.0.0.5: OSPFv2, Hello, length 44
11:17:00.715619 IP 10.0.4.21 > 10.0.4.20: ICMP echo request, id 27, seq 2, length 64
11:17:00.715657 IP 10.0.4.20 > 10.0.4.21: ICMP echo reply, id 27, seq 2, length 64
9 packets captured
9 packets received by filter
0 packets dropped by kernel
root@SA:/tmp/pycore.33599/SA.conf#

root@Monstro:/tmp/pycore.33599/Monstro.conf# tcpdump
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
^C11:16:58.398141 IP 10.0.4.1 > 224.0.0.5: OSPFv2, Hello, length 44
11:16:58.496259 IP6 fe80::200:ff:feaa:18 > ff02::5: OSPFv3, Hello, length 36
11:16:59.714450 ARP, Request who-has 10.0.4.20 tell 10.0.4.21, length 28
11:16:59.714485 ARP, Reply 10.0.4.20 is-at 00:00:00:aa:00:17 (oui Ethernet), length 28
11:16:59.714508 IP 10.0.4.21 > 10.0.4.20: ICMP echo request, id 27, seq 1, length 64
11:16:59.714520 IP 10.0.4.20 > 10.0.4.21: ICMP echo reply, id 27, seq 1, length 64
11:17:00.398443 IP 10.0.4.1 > 224.0.0.5: OSPFv2, Hello, length 44
11:17:00.715618 IP 10.0.4.21 > 10.0.4.20: ICMP echo request, id 27, seq 2, length 64
11:17:00.715644 IP 10.0.4.20 > 10.0.4.21: ICMP echo reply, id 27, seq 2, length 64
11:17:02.398499 IP 10.0.4.1 > 224.0.0.5: OSPFv2, Hello, length 44
10 packets captured
10 packets received by filter
0 packets dropped by kernel
root@Monstro:/tmp/pycore.33599/Monstro.conf#

root@Bela:/tmp/pycore.33599/Bela.conf# ping 10.0.4.20
PING 10.0.4.20 (10.0.4.20) 56(84) bytes of data:
64 bytes from 10.0.4.20: icmp_seq=1 ttl=64 time=0.112 ms
64 bytes from 10.0.4.20: icmp_seq=2 ttl=64 time=0.073 ms
^C
--- 10.0.4.20 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.073/0.095/0.112/0.016 ms
root@Bela:/tmp/pycore.33599/Bela.conf#

```

Figura 12: Tcpcmd em LAN partilhada

Tal como conseguimos comprovar nas figuras acima, em redes com LAN partilhada os pacotes ICMP são repetidos para todas as interfaces, e sendo assim no caso acima em que é feito um *ping* de *Bela* para *Monstro* os pacotes não são trocados apenas entre origem e destino pois *SA* também os recebe. Este comportamento não é verificado em LAN comutada, sendo estes pacotes trocados apenas entre origem e destino. Sendo assim, *hubs* resolvem o problema dos domínios de colisão repetindo as mensagens para todos os nós, enquanto os *switches* constroem a própria tabela de comutação de forma a saber para quem devem redirecionar os pacotes.

Para além disso, conseguimos perceber que assim que a tabela de endereços MAC está construída, o switch consegue estabelecer apenas a comunicação entre as entidades em questão, redirecionando o pacote que recebe apenas para o respetivo destino e não para todos os outros dispositivos. Isto permite um aumento de performance na respetiva rede. De um modo geral, conseguimos identificar diferenças principais entre estas estruturas: (1) o hub tem apenas um domínio de colisão, enquanto no switch diferentes portas têm dife-



rentes domínios de colisão. (2) O hub não consegue guardar endereços MAC, enquanto o switch consegue. (3) os dispositivos em questão operam em camadas diferentes: o primeiro na camada física e o segundo na camada de ligação de rede.

### Alínea 16

Construa manualmente a tabela de comutação do switch do Departamento B, atribuindo números de porta à sua escolha.

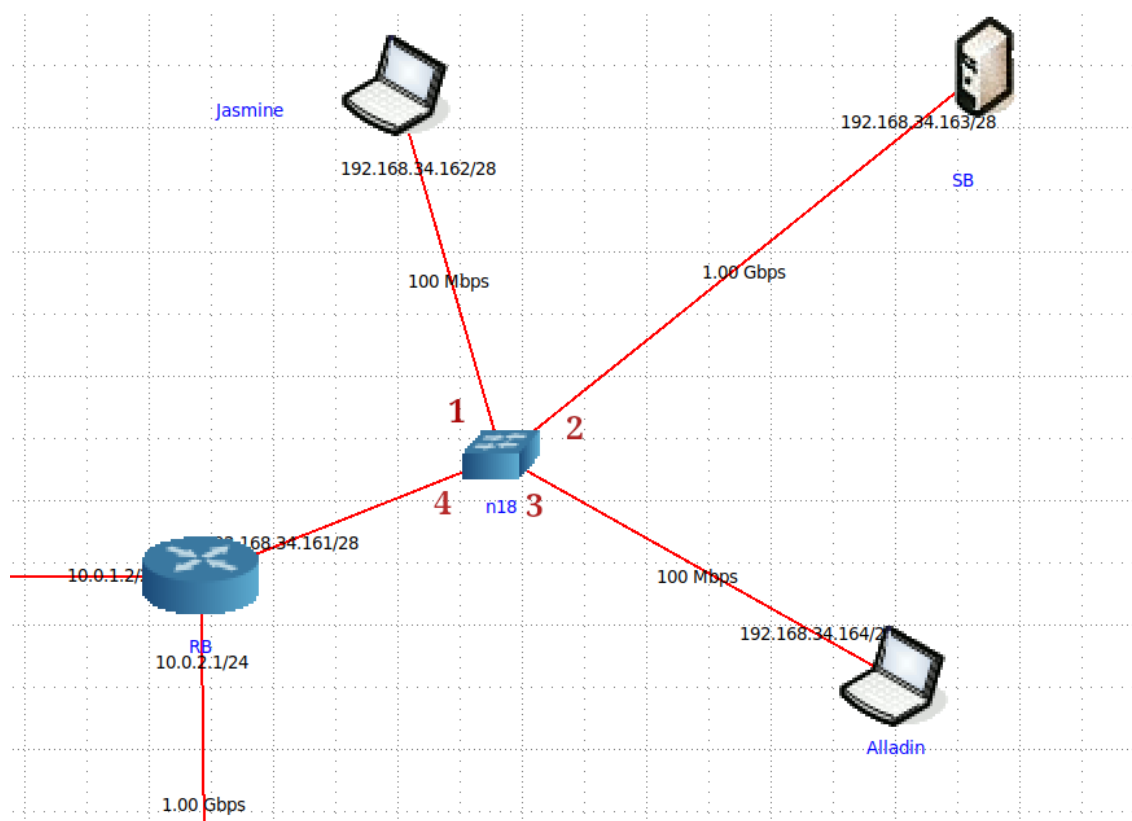


Figura 13: Departamento B com interfaces identificadas

Por definição, um switch tem a capacidade de transmitir um pacote apenas para um dispositivo capaz de o receber. Para isso, o switch usa sua tabela de endereços MAC para tomar uma decisão em relação a este encaminhamento. Sempre que um switch inicia a sua atividade começa por ter a sua tabela de endereços MAC vazia. Quando o switch recebe um pacote numa das interfaces, coloca o endereço MAC de origem do dispositivo de envio na sua tabela de endereços MAC. Contudo, o switch ainda não tem ideia do endereço MAC

de destino, portanto, ele envia o pacote para todas as suas interfaces ativas, exceto para a interface de onde recebeu esse pacote (*Flooding*).

Tendo em conta a ideia anterior, para construir a tabela de endereços MAC do departamento B temos de determinar uma maneira estabelecer uma troca de pacotes entre os constituintes deste departamento. Podemos, então, efetuar um ping da Jasmine para o Alladin e posteriormente do Router B para o Servidor B. Isto faz com que, seguindo o raciocínio anterior, se tenha o fluxo seguinte:

1. (PING 1) Com o primeiro ping, a Jasmine envia um pacote para o Alladin, iniciando o processo de comunicação;
2. O switch recebe o pacote na interface 1 e coloca o endereço MAC de origem (Jasmine) na sua tabela de endereços MAC;
3. Como o endereço MAC do Alladin não está disponível na tabela de endereços MAC do switch, o pacote é encaminhado para todas as interfaces ativas à exceção da 1;
4. O Alladin recebe o pacote e responde à Jasmine. O switch recebe este pacote na interface 3 e coloca o endereço MAC de origem (Alladin) na tabela de endereços MAC;
5. Agora, Jasmine e Alladin podem fazer uma comunicação entre si. No entanto, o Servidor B não poderá ver os pacotes transmitidos entre estes. Quando um dispositivo comunicar com o Servidor B, o seu endereço MAC também será atualizado na tabela de endereços MAC do switch. Então, ao fazer o segundo ping, inicia-se esta comunicação;
6. (PING 2) O switch recebe o pacote do Router B pela interface 4 e coloca na sua tabela o endereço deste. Novamente, o endereço do Servidor B não está na tabela do switch, pelo que este envia este pacote para todos os outros dispositivos à exceção do router B. O Servidor B recebe este pacote e responde pela interface 2 ao router, então, o switch guarda o endereço MAC do servidor na sua tabela. Neste momento, podem comunicar entre si.

Começamos por identificar cada uma das interfaces, atribuindo-lhes um número. Iniciamos a captura no Wireshark, a partir da Jasmine, enviando o primeiro ping para o Alladin. De seguida, enviamos um ping do Router B para o Servidor B. Estas duas operações [PING 1] e [PING 2], estão ilustradas nas capturas abaixo:

```

root@Jasmine:/tmp/pycore.36977/Jasmine.conf# ping 192.168.34.164
PING 192.168.34.164 (192.168.34.164) 56(84) bytes of data.
64 bytes from 192.168.34.164: icmp_seq=1 ttl=64 time=1.55 ms
64 bytes from 192.168.34.164: icmp_seq=2 ttl=64 time=0.335 ms
64 bytes from 192.168.34.164: icmp_seq=3 ttl=64 time=0.319 ms
64 bytes from 192.168.34.164: icmp_seq=4 ttl=64 time=2.47 ms

```

Figura 14: Comando ping 192.168.34.164

```

root@RB:/tmp/pycore.36977/RB.conf# ping 192.168.34.163
PING 192.168.34.163 (192.168.34.163) 56(84) bytes of data.
64 bytes from 192.168.34.163: icmp_seq=1 ttl=64 time=1.57 ms
64 bytes from 192.168.34.163: icmp_seq=2 ttl=64 time=0.109 ms
64 bytes from 192.168.34.163: icmp_seq=3 ttl=64 time=0.107 ms

```

Figura 15: Comando ping 192.168.34.163

Em relação ao tráfego e aos detalhes dos pacotes, temos:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.34.161	224.0.0.5	OSPF	78	Hello Packet
2	1.999952025	192.168.34.161	224.0.0.5	OSPF	78	Hello Packet
3	3.621599468	fe80::200:ff:feaa:9	ff02::5	OSPF	90	Hello Packet
4	4.006637854	192.168.34.162	192.168.34.164	ICMP	98	Echo (ping) request id=0x001f, seq=1/256, ttl=
5	4.007817663	192.168.34.161	224.0.0.5	OSPF	78	Hello Packet
6	4.008137096	192.168.34.164	192.168.34.162	ICMP	98	Echo (ping) reply id=0x001f, seq=1/256, ttl=
7	5.007865722	192.168.34.162	192.168.34.164	ICMP	98	Echo (ping) request id=0x001f, seq=2/512, ttl=
8	5.008153486	192.168.34.164	192.168.34.162	ICMP	98	Echo (ping) reply id=0x001f, seq=2/512, ttl=
9	6.008296001	192.168.34.161	224.0.0.5	OSPF	78	Hello Packet
10	6.014763668	192.168.34.162	192.168.34.164	ICMP	98	Echo (ping) request id=0x001f, seq=3/768, ttl=
11	6.015037007	192.168.34.164	192.168.34.162	ICMP	98	Echo (ping) reply id=0x001f, seq=3/768, ttl=
12	7.038976301	192.168.34.162	192.168.34.164	ICMP	98	Echo (ping) request id=0x001f, seq=4/1024, ttl=
13	7.041389589	192.168.34.164	192.168.34.162	ICMP	98	Echo (ping) reply id=0x001f, seq=4/1024, ttl=
14	8.008814383	192.168.34.161	224.0.0.5	OSPF	78	Hello Packet
15	9.057251621	00:00:00_aa:00:0f	00:00:00_aa:00:10	ARP	42	Who has 192.168.34.164? Tell 192.168.34.162
16	9.057897665	00:00:00_aa:00:10	00:00:00_aa:00:0f	ARP	42	192.168.34.164 is at 00:00:00_aa:00:10

Frame 4: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface veth10.0.e1, id 0

Ethernet II, Src: 00:00:00\_aa:00:0f (00:00:00\_aa:00:0f), Dst: 00:00:00\_aa:00:10 (00:00:00\_aa:00:10)

- Destination: 00:00:00\_aa:00:10 (00:00:00\_aa:00:10)
- Source: 00:00:00\_aa:00:0f (00:00:00\_aa:00:0f)
- Type: IPv4 (0x0800)

Internet Protocol Version 4, Src: 192.168.34.162, Dst: 192.168.34.164

Internet Control Message Protocol

Figura 16: Captura de tráfego no Departamento B, na Jasmine.

No.	Time	Source	Destination	Protocol	Length	Info
7	10.014197502	192.168.34.161	224.0.0.5	OSPF	78	Hello Packet
8	12.014288208	192.168.34.161	224.0.0.5	OSPF	78	Hello Packet
9	13.435630598	fe80::200:ff:feaa:9	ff02::5	OSPF	90	Hello Packet
10	14.031386722	192.168.34.161	224.0.0.5	OSPF	78	Hello Packet
11	16.023084570	192.168.34.161	224.0.0.5	OSPF	78	Hello Packet
12	18.023629687	192.168.34.161	224.0.0.5	OSPF	78	Hello Packet
13	20.023987738	192.168.34.161	224.0.0.5	OSPF	78	Hello Packet
14	20.636022511	00:00:00_aa:00:09	Broadcast	ARP	42	Who has 192.168.34.163? Tell 192.168.34.161
15	20.636153320	00:00:00_aa:00:0e	00:00:00_aa:00:09	ARP	42	192.168.34.163 is at 00:00:00_aa:00:0e
16	20.636288000	192.168.34.161	192.168.34.163	ICMP	98	Echo (ping) request id=0x0058, seq=1/256, ttl=
17	20.636345932	192.168.34.163	192.168.34.161	ICMP	98	Echo (ping) reply id=0x0058, seq=1/256, ttl=
18	21.636510128	192.168.34.161	192.168.34.163	ICMP	98	Echo (ping) request id=0x0058, seq=2/512, ttl=
19	21.636545319	192.168.34.163	192.168.34.161	ICMP	98	Echo (ping) reply id=0x0058, seq=2/512, ttl=
20	22.027518184	192.168.34.161	224.0.0.5	OSPF	78	Hello Packet
21	22.644208209	192.168.34.161	192.168.34.163	ICMP	98	Echo (ping) request id=0x0058, seq=3/768, ttl=
22	22.644242953	192.168.34.163	192.168.34.161	ICMP	98	Echo (ping) reply id=0x0058, seq=3/768, ttl=
23	23.447972155	fe80::200:ff:feaa:9	ff02::5	OSPF	90	Hello Packet
24	24.032301471	192.168.34.161	224.0.0.5	OSPF	78	Hello Packet

▶	Frame 16: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface vethf.0.e1, id 0
▼	Ethernet II, Src: 00:00:00_aa:00:09 (00:00:00_aa:00:09), Dst: 00:00:00_aa:00:0e (00:00:00_aa:00:0e)
▶	Destination: 00:00:00_aa:00:0e (00:00:00_aa:00:0e)
▶	Source: 00:00:00_aa:00:09 (00:00:00_aa:00:09)
▶	Type: IPv4 (0x0800)
▶	Internet Protocol Version 4, Src: 192.168.34.161, Dst: 192.168.34.163
▶	Internet Control Message Protocol

Figura 17: Captura de tráfego no Departamento B, no Servidor B.

Assim sendo, olhando para os campos de “Source” e “Destination”, de cada imagem acima, obtemos a tabela resultante de endereços MAC formada pelo switch:

Device	Interface	MAC Address
Jasmine	1	00:00:00:aa:00:0f
SB	2	00:00:00:aa:00:0e
Alladin	3	00:00:00:aa:00:10
RB	4	00:00:00:aa:00:09

Tabela 2: Tabela final de endereços MAC do Departamento B.

## 2. Conclusão

O presente trabalho assentou em três partes estruturais. A primeira parte prendeu-se principalmente com a análise de tráfego de tramas Ethernet, onde ficamos a compreender a relação entre a identificação dos respetivos endereços MAC e os sistemas físicos, sendo-se abordado o conceito de encapsulamento protocolar e o respetivo tamanho, em bytes, de cada porção na pilha protocolar. Já em relação à segunda parte, aprofunda-se o protocolo ARP e procura-se compreender o significado de alguns conceitos consigo relacionados, nomeadamente, analisando mensagens de pedido/resposta. Na última parte, estuda-se a questão dos domínios de colisão, já que existiria, na topologia utilizada, a possibilidade de vários hosts poderem coincidir temporalmente no envio de uma trama, deteriorizando as tramas originalmente enviadas. Esta parte permitiu compreender o modo de funcionamento de switches e a sua importância no solucionamento destas colisões. A construção de tabelas de endereços MAC é, então, fulcral, neste sentido.

Concluindo, este trabalho correu bastante bem, conseguimos responder a todas as perguntas, o que nos permitiu aprofundar e consolidar os conceitos teóricos que já aprendemos.

## Referências

- [1] C. N. Academy, “Ethernet switching.” [Online]. Available: <https://www.ciscopress.com/articles/printerfriendly/3089352>
- [2] J. F. Kurose and K. W. Ross, *Computer networking : a top-down approach*.