

1. Qual é o endereço MAC da interface ativa do seu computador?

```
[-] Ethernet II, Src: AsustekC_08:85:76 (10:bf:48:08:85:76), Dst: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
[-] Destination: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
[-] Source: AsustekC_08:85:76 (10:bf:48:08:85:76)
[-] Type: IP (0x0800)
```

R: O endereço Mac do nosso pc é 10:bf:48:08:85:76

2. Qual é o endereço MAC destino da trama? Em sua opinião, a que sistema é destinada essa trama, ou dito de outra forma, será destinada ao endereço Ethernet do servidor http para miei.di.uminho.pt? Justifique.

```
[-] Ethernet II, Src: AsustekC_08:85:76 (10:bf:48:08:85:76), Dst: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
[-] Destination: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
[-] Source: AsustekC_08:85:76 (10:bf:48:08:85:76)
[-] Type: IP (0x0800)
```

R: O endereço Mac destino é 00:0c:29:d2:19:f0. Não, a trama vai ser destinada ao router que vai estar ligado à rede do departamento, depois este irá aceder ao http server e irá enviar a resposta.

3. Qual o valor hexadecimal presente no campo tipo (Type) da trama Ethernet? O que significa?

```
[-] Ethernet II, Src: AsustekC_08:85:76 (10:bf:48:08:85:76), Dst: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
[-] Destination: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
[-] Source: AsustekC_08:85:76 (10:bf:48:08:85:76)
[-] Type: IP (0x0800)
```

R: 0x0800 tipo de dados que vai ser encapsulado (tipo IP).

4. Quantos bytes são usados desde o início da trama até ao caractere ASCII “G” do método HTTP GET? Calcule e indique, em percentagem, a sobrecarga (overhead) introduzida pela pilha protocolar no envio do HTTP GET.

```
[-] Frame 60: 416 bytes on wire (3328 bits), 416 bytes captured (3328 bits) on interface 0
[-] Ethernet II, Src: AsustekC_08:85:76 (10:bf:48:08:85:76), Dst: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
[-] Destination: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
[-] Address: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
[-] .... 00. .... = LG bit: Globally unique address (factory default)
[-] .... 00. .... = IG bit: Individual address (unicast)
[-] Source: AsustekC_08:85:76 (10:bf:48:08:85:76)
[-] Type: IP (0x0800)
[-] Internet Protocol Version 4, Src: 192.168.100.178 (192.168.100.178), Dst: 193.136.19.20 (193.136.19.20)
0000 00 0c 29 d2 19 f0 10 bf 48 08 85 76 08 00 45 00 ..).H.V..E.
0010 01 92 30 17 40 00 80 06 00 00 c0 a8 64 b2 c1 88 ..0.8...d...
0020 13 14 f3 83 00 50 ba 58 a9 c7 13 7e 59 6d 50 18 ...P.X...Ymp.
0030 01 00 fb 7b 00 00 47 45 54 20 2f 20 48 54 54 50 ...GE T / HTTP
0040 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 6d 69 65 69 /1.1..Ho st: miei
0050 7a 64 60 7a 75 64 60 6a 68 66 7a 70 7a 0d 0a 65 d4 umio ho st: u
```

R: São 54 bytes até chegarmos ao carácter G da instrução HTTP GET. Sendo tamanho da trama 416 então teremos um Overhead de  $54/416 = 0.1298 = 12,98\%$  aproximadamente.

5. Em ligações com fios pouco susceptíveis a erros, nem sempre as NICs geram o código de detecção de erros. Verifique se o campo FCS está a ser utilizado, justifique.

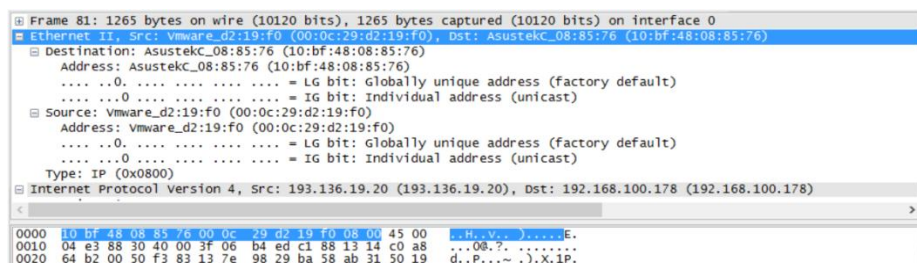
R: Não utiliza FCS pois se utilizasse a informação relativa a este iria aparecer a seguir à informação do tipo no nível 2.

6. Aceda à opção Edit/Preferences/Protocols/Ethernet e indique que é assumido o uso do campo FCS. Verifique qual o valor hexadecimal desse campo na trama capturada. Que conclui?

❌ Frame check sequence: 0x0d0a0d0a [incorrect, should be 0xe2354d38]

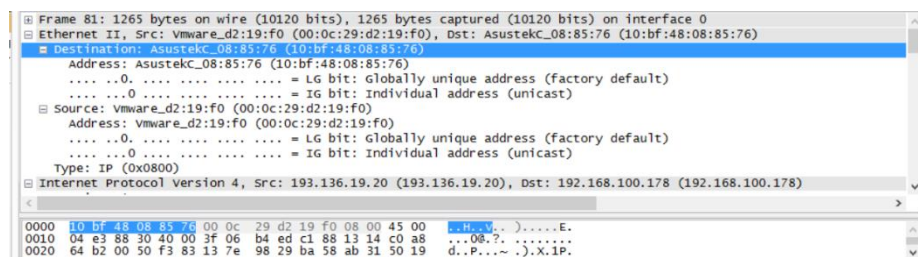
R: o valor capturado corresponde aos últimos 4 bytes da trama. Ao fazer a mudança vai dar como incorreto pois como não foi utilizado fcs não tem os bytes de verificação de erro que deveria ter, como mostrado na imagem.

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



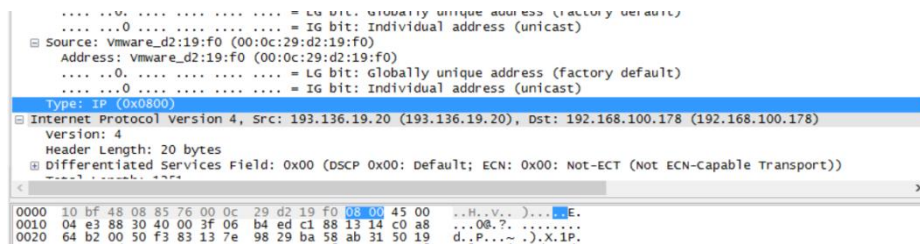
R: O endereço da fonte é 00:0c:29:d2:19:f0 que corresponde à máquina virtual responsável pelo endereçamento da página. Sendo solicitada a página miei.uminho.pt, o pedido terá de ser enviado para a rede que suporta o site, e a resposta virá deste mesmo.

8. Qual é o endereço MAC do destino? Reconhece-o?



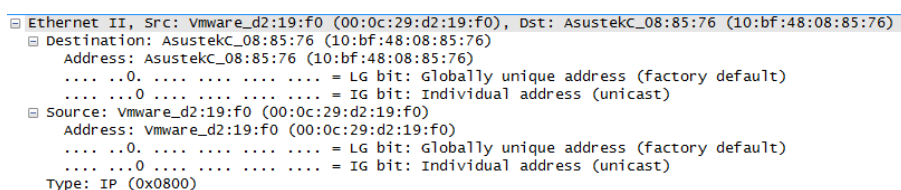
R: Como podemos ver na imagem o endereço mac de destino corresponde ao endereço mac do computador que efetuou o pedido anteriormente (10:bf:48:08:85:76).

9. Qual é o valor hexadecimal do campo tipo (Type)?



lor é 0x0800.

10. Que tipo de resposta foi enviada pelo servidor?



R: Foi enviada uma resposta de confirmação, um “ok”.

11. Observe o conteúdo da tabela ARP. O que significa cada uma das colunas?

```
Interface: 192.168.56.1 --- 0x2
Internet Address      Physical Address      Type
192.168.56.255        ff-ff-ff-ff-ff-ff    static
224.0.0.22            01-00-5e-00-00-16    static
224.0.0.252           01-00-5e-00-00-fc    static
239.192.152.143       01-00-5e-40-98-8f    static
239.255.255.250       01-00-5e-7f-ff-fa    static

Interface: 192.168.100.187 --- 0x3
Internet Address      Physical Address      Type
192.168.100.188       f0-79-59-33-25-f0    dynamic
192.168.100.224       1c-b7-2c-9f-14-e1    dynamic
192.168.100.254       00-0c-29-d2-19-f0    dynamic
192.168.100.255       ff-ff-ff-ff-ff-ff    static
224.0.0.22            01-00-5e-00-00-16    static
224.0.0.251           01-00-5e-00-00-fb    static
224.0.0.252           01-00-5e-00-00-fc    static
239.192.152.143       01-00-5e-40-98-8f    static
239.255.255.250       01-00-5e-7f-ff-fa    static
255.255.255.255       ff-ff-ff-ff-ff-ff    static
```

```
Interface: 192.168.56.1 --- 0x2
Internet Address      Physical Address      Type
192.168.56.255        ff-ff-ff-ff-ff-ff    static
224.0.0.22            01-00-5e-00-00-16    static
224.0.0.252           01-00-5e-00-00-fc    static
239.255.255.250       01-00-5e-7f-ff-fa    static

Interface: 192.168.100.187 --- 0x3
Internet Address      Physical Address      Type
192.168.100.172       00-1b-63-1e-b3-43    dynamic
192.168.100.254       00-0c-29-d2-19-f0    dynamic
192.168.100.255       ff-ff-ff-ff-ff-ff    static
224.0.0.22            01-00-5e-00-00-16    static
224.0.0.251           01-00-5e-00-00-fb    static
224.0.0.252           01-00-5e-00-00-fc    static
239.255.255.250       01-00-5e-7f-ff-fa    static
```

R:Na tabela de arp aparecem duas interface, uma para a rede local e outra para a rede wireless, estando na primeira coluna o IP, na segunda o MAC adress e na terceira o tipo, podendo ser estático ou dinâmico.

- ```

Ethernet II, Src: Vmware_d2:19:f0 (00:0c:29:d2:19:f0), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Destination: Broadcast (ff:ff:ff:ff:ff:ff)
    Address: broadcast (ff:ff:ff:ff:ff:ff)
        ....1.1. .... = LG bit: Locally administered address (this is NOT the factory default)
        ....1.1. .... = IG bit: Group address (multicast/broadcast)
Source: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
Address: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
    ....0. .... = LG bit: Globally unique address (factory default)
    ....0. .... = IG bit: Individual address (unicast)
Type: ARP (0x0806)
Padding: 00

```

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

```

# Ethernet II, Src: Vmware_d2:19:f0 (00:0c:29:d2:19:f0), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
# Destination: Broadcast (ff:ff:ff:ff:ff:ff)
#   Address: Broadcast (ff:ff:ff:ff:ff:ff)
#     .... 1. .... = LG bit: Locally administered address (this is NOT the factory default)
#     .... 1. .... = IG bit: Group address (multicast/broadcast)
# Source: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
#   Address: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
#     .... 0. .... = LG bit: Globally unique address (factory default)
#     .... 0. .... = IG bit: Individual address (unicast)
# Type: ARP (0x0806)
# Padding: 00
# Address Resolution Protocol (request)
0000 ff ff ff ff ff ff ff ff 00 0c 29 d2 19 f0 00 01 .....).
0010 08 00 06 04 00 01 00 0c 29 d2 19 f0 c0 a8 64 fe .....d.
0020 00 00 00 00 00 00 00 c0 a8 64 9e 00 00 00 00 00 .....d....
0030 00 00 00 00 00 00 00 00 00 00 00 00 .....

```

R: A Ethernet type corresponde a ARP (0x0806), que indica que o Address Resolution Protocol do tipo de Ethernet corresponde a esse valor.

14. Qual o valor do campo ARP opcode? O que especifica? Se necessário, consulte a RFC do protocolo ARP <http://tools.ietf.org/html/rfc826.html>.

| Address Resolution Protocol (request)                     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
|-----------------------------------------------------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-------|
| Hardware type: Ethernet (1)                               |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
| Protocol type: IP (0x0800)                                |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
| Hardware size: 6                                          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
| Protocol size: 4                                          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
| Opcode: request (1)                                       |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
| Sender MAC address: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
| Sender IP address: 192.168.100.254 (192.168.100.254)      |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
| Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00) |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
| Target IP address: 192.168.100.158 (192.168.100.158)      |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
| 0000                                                      | ff | ff | ff | ff | ff | ff | 00 | 0c | 29 | d2 | 19 | f0 | 08 | 06 | 00 01 |
| 0010                                                      | 08 | 00 | 06 | 04 | 00 | 01 | 00 | 0c | 29 | d2 | 19 | f0 | c0 | a8 | 64 fe |
| 0020                                                      | 00 | 00 | 00 | 00 | 00 | 00 | c0 | a8 | 64 | 9e | 00 | 00 | 00 | 00 | 00    |
| 0030                                                      | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00    |

R: O valor do campo ARP opcode é 00 01. É um valor de 16 bits que indica se é relativo a uma request ou a uma reply., que representa então uma flag, sendo ele neste caso um request como vemos na imagem acima indicado a azul Op-code: request (1).

15. A mensagem ARP contém o endereço IP de origem? Que tipo de pergunta é feita?

| No. | Time        | Source            | Destination       | Protocol | Length | Info                                          |
|-----|-------------|-------------------|-------------------|----------|--------|-----------------------------------------------|
| 3   | 0.41365400  | Vmware_d2:19:f0   | Broadcast         | ARP      | 60     | who has 192.168.100.158? Tell 192.168.100.254 |
| 575 | 3.08015200  | Vmware_d2:19:f0   | Broadcast         | ARP      | 60     | who has 192.168.100.158? Tell 192.168.100.254 |
| 579 | 4.08010400  | Vmware_d2:19:f0   | Broadcast         | ARP      | 60     | who has 192.168.100.158? Tell 192.168.100.254 |
| 584 | 5.07900400  | Vmware_d2:19:f0   | Broadcast         | ARP      | 60     | who has 192.168.100.158? Tell 192.168.100.254 |
| 590 | 6.43326000  | AsustekC_1c:37:79 | Broadcast         | ARP      | 60     | who has 192.168.100.254? Tell 192.168.100.182 |
| 591 | 6.49322000  | AsustekC_08:85:76 | Broadcast         | ARP      | 42     | who has 192.168.100.172? Tell 192.168.100.187 |
| 592 | 6.49380700  | Apple_1e:b3:43    | AsustekC_08:85:76 | ARP      | 60     | 192.168.100.172 is at 00:1b:63:1e:b3:43       |
| 595 | 6.68837600  | AsustekC_1c:37:79 | Broadcast         | ARP      | 60     | who has 192.168.100.182? Tell 0.0.0.0         |
| 639 | 9.59674000  | Vmware_d2:19:f0   | Broadcast         | ARP      | 60     | who has 192.168.100.158? Tell 192.168.100.254 |
| 640 | 10.18826500 | AsustekC_1c:37:79 | Broadcast         | ARP      | 60     | who has 192.168.100.182? Tell 0.0.0.0         |
| 650 | 10.59694200 | Vmware_d2:19:f0   | Broadcast         | ARP      | 60     | who has 192.168.100.158? Tell 192.168.100.254 |
| 651 | 10.68839300 | AsustekC_1c:37:79 | Broadcast         | ARP      | 60     | who has 192.168.100.248? Tell 192.168.100.182 |
| 655 | 11.18822600 | AsustekC_1c:37:79 | Broadcast         | ARP      | 60     | who has 192.168.100.182? Tell 0.0.0.0         |
| 659 | 11.59682900 | Vmware_d2:19:f0   | Broadcast         | ARP      | 60     | who has 192.168.100.158? Tell 192.168.100.254 |
| 662 | 12.05073300 | AsustekC_1c:37:79 | Broadcast         | ARP      | 60     | who has 192.168.100.254? Tell 192.168.100.182 |
| 663 | 12.05485100 | Vmware_d2:19:f0   | Broadcast         | ARP      | 60     | who has 192.168.100.248? Tell 192.168.100.254 |
| 667 | 12.18840100 | AsustekC_1c:37:79 | Broadcast         | ARP      | 60     | who has 192.168.100.182? Tell 0.0.0.0         |
| 674 | 13.24770100 | AsustekC_1c:37:79 | Broadcast         | ARP      | 60     | who has 192.168.100.254? Tell 192.168.100.182 |
| 678 | 13.49744600 | AsustekC_1c:37:79 | Broadcast         | ARP      | 60     | who has 192.168.100.254? Tell 192.168.100.182 |

R: O endereço IP de origem está a seguir à palavra Tell, que corresponde ao endereço de IP da nossa máquina. Temos como objetivo conectar diretamente com alguma máquina. Para isso enviamos um pedido (broadcast) para todos os dispositivos na rede ethernet a que estamos ligados, perguntando quem tem o IP 192.168.100.158, de forma a obtermos o seu endereço MAC para o poder conectar diretamente a este. A pergunta que é feita é “Who has 192.168.100.158?”, ou seja, quem tem este endereço IP.

16. Localize a mensagem ARP que é a resposta ao pedido ARP efectuado.

| No. | Time        | Source            | Destination       | Protocol | Length | Info                                          |
|-----|-------------|-------------------|-------------------|----------|--------|-----------------------------------------------|
| 3   | 0.41365400  | Vmware_d2:19:f0   | Broadcast         | ARP      | 60     | who has 192.168.100.158? Tell 192.168.100.254 |
| 575 | 3.08015200  | Vmware_d2:19:f0   | Broadcast         | ARP      | 60     | who has 192.168.100.158? Tell 192.168.100.254 |
| 579 | 4.08010400  | Vmware_d2:19:f0   | Broadcast         | ARP      | 60     | who has 192.168.100.158? Tell 192.168.100.254 |
| 584 | 5.07990400  | Vmware_d2:19:f0   | Broadcast         | ARP      | 60     | who has 192.168.100.158? Tell 192.168.100.254 |
| 590 | 6.43332600  | AsustekC_1c:37:79 | Broadcast         | ARP      | 60     | who has 192.168.100.254? Tell 192.168.100.182 |
| 591 | 6.49332200  | AsustekC_08:85:76 | Broadcast         | ARP      | 42     | who has 192.168.100.172? Tell 192.168.100.187 |
| 592 | 6.53101000  | Apple_1e:b3:43    | AsustekC_08:85:76 | ARP      | 60     | 192.168.100.172 has 192.168.100.187           |
| 595 | 6.68837600  | AsustekC_1c:37:79 | Broadcast         | ARP      | 60     | who has 192.168.100.182? Tell 0.0.0.0         |
| 639 | 9.59674000  | Vmware_d2:19:f0   | Broadcast         | ARP      | 60     | who has 192.168.100.158? Tell 192.168.100.254 |
| 640 | 10.18826500 | AsustekC_1c:37:79 | Broadcast         | ARP      | 60     | who has 192.168.100.182? Tell 0.0.0.0         |
| 650 | 10.59694200 | Vmware_d2:19:f0   | Broadcast         | ARP      | 60     | who has 192.168.100.158? Tell 192.168.100.254 |
| 651 | 10.68839300 | AsustekC_1c:37:79 | Broadcast         | ARP      | 60     | who has 192.168.100.248? Tell 192.168.100.182 |
| 655 | 11.18822600 | AsustekC_1c:37:79 | Broadcast         | ARP      | 60     | who has 192.168.100.182? Tell 0.0.0.0         |
| 659 | 11.59682900 | Vmware_d2:19:f0   | Broadcast         | ARP      | 60     | who has 192.168.100.158? Tell 192.168.100.254 |
| 662 | 12.05073300 | AsustekC_1c:37:79 | Broadcast         | ARP      | 60     | who has 192.168.100.254? Tell 192.168.100.182 |
| 663 | 12.05485100 | Vmware_d2:19:f0   | Broadcast         | ARP      | 60     | who has 192.168.100.248? Tell 192.168.100.254 |
| 667 | 12.18840100 | AsustekC_1c:37:79 | Broadcast         | ARP      | 60     | who has 192.168.100.182? Tell 0.0.0.0         |
| 674 | 13.24770100 | AsustekC_1c:37:79 | Broadcast         | ARP      | 60     | who has 192.168.100.254? Tell 192.168.100.182 |
| 678 | 13.49744600 | AsustekC_1c:37:79 | Broadcast         | ARP      | 60     | who has 192.168.100.254? Tell 192.168.100.182 |

R: A resposta dada é o endereço MAC correspondente ao endereço IP procurado, 00:1b:63:1e:b3:43.

a. Qual o valor do campo ARP opcode? O que especifica?

Padding: 00000000000000000000000000000000

Address Resolution Protocol (reply)

Hardware type: Ethernet (1)

Protocol type: IP (0x0800)

Hardware size: 6

Protocol size: 4

opcode: reply (2)

Sender MAC address: Apple\_1e:b3:43 (00:1b:63:1e:b3:43)

Sender IP address: 192.168.100.172 (192.168.100.172)

Target MAC address: AsustekC\_08:85:76 (10:bf:48:08:85:76)

Target IP address: 192.168.100.187 (192.168.100.187)

|      |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |                   |
|------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-------------------|
| 0000 | 10 | bf | 48 | 08 | 85 | 76 | 00 | 1b | 63 | 1e | b3 | 43 | 08 | 06 | 00 | 01 | ..H..V.. C..C...  |
| 0010 | 08 | 00 | 06 | 04 | 00 | 02 | 00 | 1b | 63 | 1e | b3 | 43 | c0 | a8 | 64 | ac | ....[.]. C..C..d. |
| 0020 | 10 | bf | 48 | 08 | 85 | 76 | c0 | a8 | 64 | bb | 00 | 00 | 00 | 00 | 00 | 00 | ..H..V.. d.....   |
| 0030 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | .....             |

R: O valor do campos Opcode é 00 02. O reply(2) especifica que é o valor que indicamos, basicamente o valor a resposta do “broadcast” que fizemos.



- b. Em que posição da mensagem ARP está a informação que responde ao pedido ARP?

Address Resolution Protocol (reply)

Hardware type: Ethernet (1)

Protocol type: IP (0x0800)

Hardware size: 6

Protocol size: 4

opcode: reply (2)

Sender MAC address: Apple\_1e:b3:43 (00:1b:63:1e:b3:43)

Sender IP address: 192.168.100.172 (192.168.100.172)

Target MAC address: AsustekC\_08:85:76 (10:bf:48:08:85:76)

Target IP address: 192.168.100.187 (192.168.100.187)

R: A resposta esta onde temos o MAC que pretendemos encontrar, como podemos ver na imagem acima a azul.

17. Quais são os valores hexadecimais para os endereços origem e destino da trama que contém a resposta ARP? Que conclui?

[- Address Resolution Protocol (reply)

Hardware type: Ethernet (1)

Protocol type: IP (0x0800)

Hardware size: 6

Protocol size: 4

Opcode: reply (2)

Sender MAC address: Apple\_1e:b3:43 (00:1b:63:1e:b3:43)

Sender IP address: 192.168.100.172 (192.168.100.172)

Target MAC address: AsustekC\_08:85:76 (10:bf:48:08:85:76)

Target IP address: 192.168.100.187 (192.168.100.187)

.....

0000 10 bf 48 08 85 76 00 1b 63 1e b3 43 08 06 00 01 ..H..v.. C..C...

0010 08 00 06 04 00 02 00 1b 63 1e b3 43 c0 a8 64 ac ..... C..C..d.

0020 10 bf 48 08 85 76 c0 a8 64 bb 00 00 00 00 00 00 ..H..V.. d.....

0030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....

R: Os Valores do endereço destino é c0 a8 64 ac.



| Address Resolution Protocol (reply)                       |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
|-----------------------------------------------------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-------|
| Hardware type: Ethernet (1)                               |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
| Protocol type: IP (0x0800)                                |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
| Hardware size: 6                                          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
| Protocol size: 4                                          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
| opcode: reply (2)                                         |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
| Sender MAC address: Apple_1e:b3:43 (00:1b:63:1e:b3:43)    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
| Sender IP address: 192.168.100.172 (192.168.100.172)      |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
| Target MAC address: AsustekC_08:85:76 (10:bf:48:08:85:76) |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
| Target IP address: 192.168.100.187 (192.168.100.187)      |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |
| 0000                                                      | 10 | bf | 48 | 08 | 85 | 76 | 00 | 1b | 63 | 1e | b3 | 43 | 08 | 06 | 00 01 |
| 0010                                                      | 08 | 00 | 06 | 04 | 00 | 02 | 00 | 1b | 63 | 1e | b3 | 43 | c0 | a8 | 64 ac |
| 0020                                                      | 10 | bf | 48 | 08 | 85 | 76 | c0 | a8 | 64 | bb | 00 | 00 | 00 | 00 | 00 00 |
| 0030                                                      | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 00 |

R: Os Valores para o endereço destino são c0 a8 64 bb.

Concluímos que podemos ligar diretamente a máquina através do seu endereço MAC.

18. Com auxílio do comando ifconfig obtenha os endereços Ethernet das interfaces dos diversos routers.

```
n1.eth0.65 Link encap:Ethernet HWaddr 66:9b:15:b4:c2:ca
        inet6 addr: fe80::649b:15ff:feb4:c2ca/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
        RX packets:81 errors:0 dropped:0 overruns:0 frame:0
        TX packets:104 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:7158 (7.1 KB) TX bytes:14276 (14.2 KB)

n2.eth0.65 Link encap:Ethernet HWaddr 66:f1:20:83:6e:12
        inet6 addr: fe80::64f1:20ff:fe83:6e12/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
        RX packets:75 errors:0 dropped:0 overruns:0 frame:0
        TX packets:111 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:7126 (7.1 KB) TX bytes:14418 (14.4 KB)

n2.eth1.65 Link encap:Ethernet HWaddr 4e:8b:09:01:16:f2
        inet6 addr: fe80::4c8b:9ff:fe01:16f2/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
        RX packets:85 errors:0 dropped:0 overruns:0 frame:0
        TX packets:100 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:7734 (7.7 KB) TX bytes:13764 (13.7 KB)

n3.eth0.65 Link encap:Ethernet HWaddr a6:b1:d6:51:90:75
        inet6 addr: fe80::a4b1:d6ff:fe51:9075/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
        RX packets:70 errors:0 dropped:0 overruns:0 frame:0
        TX packets:116 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:6536 (6.5 KB) TX bytes:15072 (15.0 KB)
```

R:  
Os

endereços Ethernet das interfaces dos diversos routers:

n1: 66:9b:15:b4:c2:ca

n2: 66:f1:20:83:6e:12 e 4e:8b:09:01:16:f2

n3: a6:b1:d6:51:90:75

19. Usando o comando arp obtenha o conteúdo das caches arp dos diversos sistemas.

```
Command line
arp -a
Command results
> arp -a
> arp -a
> n1 > arp -a:
A0 (10.0.0.1) em 00:00:00:aa:00:00 [ether] em eth0

> arp -a
> n2 > arp -a:
? (10.0.1.1) em 00:00:00:aa:00:02 [ether] em eth1
A1 (10.0.0.2) em 00:00:00:aa:00:01 [ether] em eth0

> n3 > arp -a:
? (10.0.1.2) em 00:00:00:aa:00:03 [ether] em eth0
```

20. Faça ping de n1 para n2. Que modificações observa nas caches ARP dos sistemas envolvidos.

```
root@n1: /tmp/pycore.41954/n1.conf
root@n1:/tmp/pycore.41954/n1.conf# ping 10.0.0.1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data:
64 bytes from 10.0.0.1: icmp_req=1 ttl=64 time=0.027 ms
64 bytes from 10.0.0.1: icmp_req=2 ttl=64 time=0.023 ms
64 bytes from 10.0.0.1: icmp_req=3 ttl=64 time=0.025 ms
64 bytes from 10.0.0.1: icmp_req=4 ttl=64 time=0.024 ms
64 bytes from 10.0.0.1: icmp_req=5 ttl=64 time=0.026 ms
64 bytes from 10.0.0.1: icmp_req=6 ttl=64 time=0.025 ms
64 bytes from 10.0.0.1: icmp_req=7 ttl=64 time=0.036 ms
64 bytes from 10.0.0.1: icmp_req=8 ttl=64 time=0.024 ms
^C
--- 10.0.0.1 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 6999ms
rtt min/avg/max/mdev = 0.023/0.026/0.036/0.005 ms
root@n1:/tmp/pycore.41954/n1.conf# arp -a
A0 (10.0.0.1) em 00:00:00:aa:00:00 [ether] em eth0
root@n1:/tmp/pycore.41954/n1.conf#

root@n2: /tmp/pycore.41954/n2.conf
root@n2:/tmp/pycore.41954/n2.conf# arp -a
? (10.0.1.1) em 00:00:00:aa:00:02 [ether] em eth1
A1 (10.0.0.2) em 00:00:00:aa:00:01 [ether] em eth0
root@n2:/tmp/pycore.41954/n2.conf#
```

R: Observando a cache do sistema n1 reparamos que o endereço 10.0.0.1 está ligado a 00:00:00:aa:00:00. A cache do sistema n2 reparamos que o endereço 10.0.0.2 está ligado a 00:00:00:aa:00:01.

21. Faça ping de n1 para n3. Consulte as caches ARP. Que conclui?

```

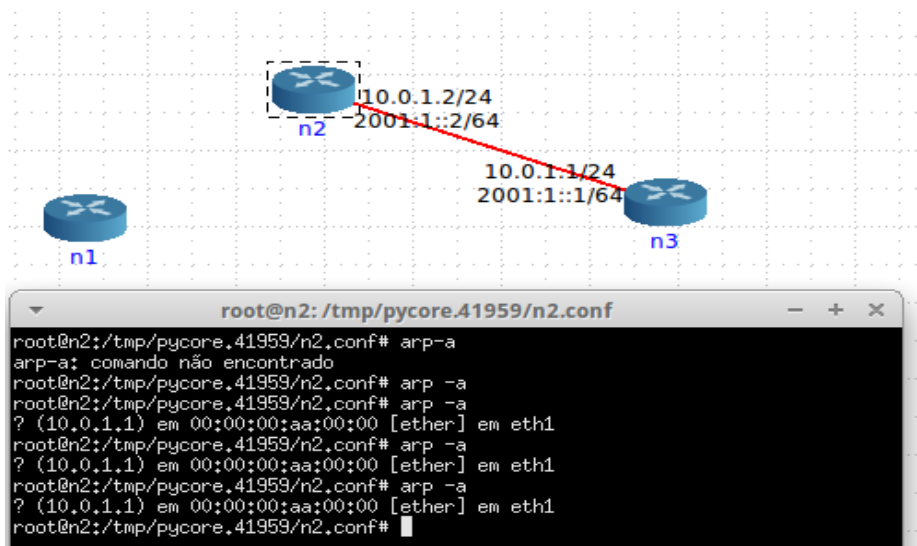
root@n1:/tmp/pycore.41954/n1.conf# ping 10.0.1.1
PING 10.0.1.1 (10.0.1.1) 56(84) bytes of data:
64 bytes from 10.0.1.1: icmp_req=1 ttl=63 time=0.031 ms
64 bytes from 10.0.1.1: icmp_req=2 ttl=63 time=0.029 ms
64 bytes from 10.0.1.1: icmp_req=3 ttl=63 time=0.029 ms
64 bytes from 10.0.1.1: icmp_req=4 ttl=63 time=0.031 ms
64 bytes from 10.0.1.1: icmp_req=5 ttl=63 time=0.032 ms
64 bytes from 10.0.1.1: icmp_req=6 ttl=63 time=0.034 ms
64 bytes from 10.0.1.1: icmp_req=7 ttl=63 time=0.030 ms
64 bytes from 10.0.1.1: icmp_req=8 ttl=63 time=0.033 ms
64 bytes from 10.0.1.1: icmp_req=9 ttl=63 time=0.030 ms
^C
--- 10.0.1.1 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 7996ms
rtt min/avg/max/mdev = 0.029/0.031/0.034/0.001 ms
root@n1:/tmp/pycore.41954/n1.conf# arp -a
A0 (10.0.0.1) em 00:00:00:aa:00:00 [ether] em eth0
root@n1:/tmp/pycore.41954/n1.conf#

root@n3:/tmp/pycore.41954/n3.conf# arp -a
? (10.0.1.2) em 00:00:00:aa:00:03 [ether] em eth0
root@n3:/tmp/pycore.41954/n3.conf#

```

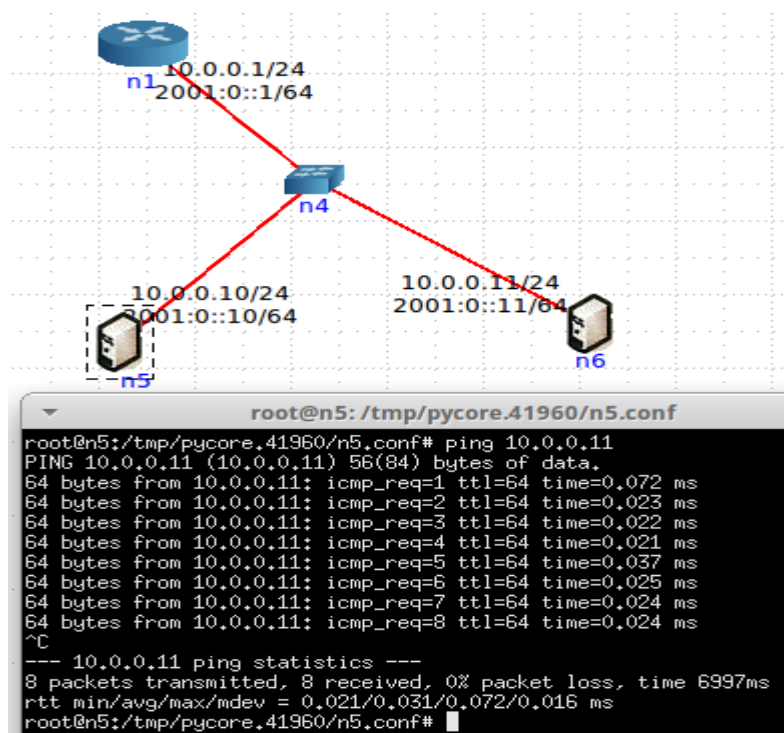
R: Através da imagem concluímos que não existe ligação de n1 para n3 como podemos comprovar através do arp das caches.

22. Em n1 remova a entrada correspondente a n2. Coloque uma nova entrada para n2 com endereço Ethernet inexistente. O que acontece?



R: Como podemos ver, apenas a interface para n3 aparece, sendo o endereço que colocamos ff:ff:ff:ff:ff ignorado.

23. Faça ping de n5 para n6. Sem consultar a tabela ARP anote a entrada que, em sua opinião, é criada na tabela ARP de n5. Verifique se a sua interpretação sobre a operação da rede Ethernet e protocolo ARP estava correto.



R: No ARP da bash de n5 deverá aparecer agora o IP de n6 e à frente o endereço mac de n6.

```
root@n5:/tmp/pycore.41960/n5.conf# arp -a
A10 (10.0.0.11) em 00:00:00:aa:00:03 [ether] em eth0
root@n5:/tmp/pycore.41960/n5.conf#
```

```
root@n6: /tmp/pycore.41960/n6.conf
root@n6:/tmp/pycore.41960/n6.conf# arp -a
A8 (10.0.0.10) em 00:00:00:aa:00:02 [ether] em eth0
root@n6:/tmp/pycore.41960/n6.conf#
```

R: Como podemos ver na imagem, aquilo que dissemos em cima confirmar-se, e em n6 também aparece uma nova entrada com o IP de n5.

## Parte 2

1. Identifique um pacote de pedido ARP gratuito originado pelo seu sistema. Verifique quantos pacotes ARP gratuito foram enviados e com que intervalo temporal?

|     |             |                   |                   |     |                                                  |
|-----|-------------|-------------------|-------------------|-----|--------------------------------------------------|
| 41  | 1.263539000 | vmware_d2:19:f0   | Broadcast         | ARP | 60 who has 192.168.100.193? Tell 192.168.100.254 |
| 42  | 1.274445000 | AsustekC_08:85:76 | Broadcast         | ARP | 42 Gratuitous ARP for 192.168.100.187 (Request)  |
| 68  | 1.405377000 | AsustekC_08:85:76 | Broadcast         | ARP | 42 who has 192.168.100.222? Tell 192.168.100.187 |
| 69  | 1.405749000 | SmcNetwo_8b:44:d7 | AsustekC_08:85:76 | ARP | 60 192.168.100.222 is at 00:22:2d:8b:44:d7       |
| 92  | 2.020527000 | vmware_d2:19:f0   | Broadcast         | ARP | 60 who has 192.168.100.201? Tell 192.168.100.254 |
| 111 | 2.685518000 | vmware_d2:19:f0   | Broadcast         | ARP | 60 who has 192.168.100.193? Tell 192.168.100.254 |
| 126 | 3.020505000 | vmware_d2:19:f0   | Broadcast         | ARP | 60 who has 192.168.100.201? Tell 192.168.100.254 |
| 130 | 6.600909000 | AsustekC_08:85:76 | Broadcast         | ARP | 42 who has 192.168.100.254? Tell 192.168.100.187 |
| 132 | 6.775060000 | AsustekC_08:85:76 | Broadcast         | ARP | 42 who has 192.168.100.187? Tell 0.0.0.0         |
| 137 | 6.999723000 | HewlettP_e3:0f:85 | AsustekC_08:85:76 | ARP | 60 who has 192.168.100.187? Tell 192.168.100.182 |
| 138 | 6.999786000 | AsustekC_08:85:76 | HewlettP_e3:0f:85 | ARP | 42 192.168.100.187 is at 10:bf:48:08:85:76       |
| 144 | 7.269323000 | vmware_d2:19:f0   | Broadcast         | ARP | 60 who has 192.168.100.201? Tell 192.168.100.254 |
| 145 | 7.275041000 | AsustekC_08:85:76 | Broadcast         | ARP | 42 who has 192.168.100.254? Tell 192.168.100.187 |
| 147 | 7.275506000 | vmware_d2:19:f0   | AsustekC_08:85:76 | ARP | 60 192.168.100.254 is at 00:0c:29:d2:19:f0       |
| 169 | 7.744199000 | vmware_d2:19:f0   | Broadcast         | ARP | 60 who has 192.168.100.193? Tell 192.168.100.254 |
| 174 | 7.775156000 | AsustekC_08:85:76 | Broadcast         | ARP | 42 who has 192.168.100.187? Tell 0.0.0.0         |
| 265 | 8.431300000 | vmware_d2:19:f0   | Broadcast         | ARP | 60 who has 192.168.100.201? Tell 192.168.100.254 |
| 271 | 8.539715000 | AsustekC_36:e6:0d | Broadcast         | ARP | 60 who has 192.168.100.190? Tell 192.168.100.228 |
| 298 | 8.764115000 | SamsungE_a8:b0:d2 | Broadcast         | ARP | 60 who has 192.168.100.254? Tell 192.168.100.186 |
| 299 | 8.775199000 | AsustekC_08:85:76 | Broadcast         | ARP | 42 who has 192.168.100.187? Tell 0.0.0.0         |
| 305 | 8.802141000 | vmware_d2:19:f0   | Broadcast         | ARP | 60 who has 192.168.100.193? Tell 192.168.100.254 |
| 308 | 8.816356000 | SamsungE_a8:b0:d2 | Broadcast         | ARP | 60 who has 192.168.100.254? Tell 192.168.100.186 |
| 321 | 9.110057000 | SamsungE_a8:b0:d2 | Broadcast         | ARP | 60 who has 192.168.100.186? Tell 0.0.0.0         |
| 350 | 9.431262000 | vmware_d2:19:f0   | Broadcast         | ARP | 60 who has 192.168.100.201? Tell 192.168.100.254 |
| 373 | 9.774453000 | AsustekC_08:85:76 | Broadcast         | ARP | 42 Gratuitous ARP for 192.168.100.187 (Request)  |

R: Foram enviados 2 pacotes com um espaço entre eles de 8,5.

2. Analise o conteúdo de um pedido ARP gratuito e identifique em que se distingue dos restantes pedidos ARP. Registe a trama Ethernet correspondente. Qual o resultado esperado face ao pedido ARP gratuito enviado?

### Arp normal:

Sender MAC address: AsustekC\_08:85:76 (10:bf:48:08:85:76)  
 Sender IP address: 192.168.100.187 (192.168.100.187)  
 Target MAC address: 00:00:00\_00:00:00 (00:00:00:00:00:00)  
 Target IP address: 192.168.100.222 (192.168.100.222)

### Arp gratuito:

Sender MAC address: AsustekC\_08:85:76 (10:bf:48:08:85:76)  
 Sender IP address: 192.168.100.187 (192.168.100.187)  
 Target MAC address: 00:00:00\_00:00:00 (00:00:00:00:00:00)  
 Target IP address: 192.168.100.187 (192.168.100.187)

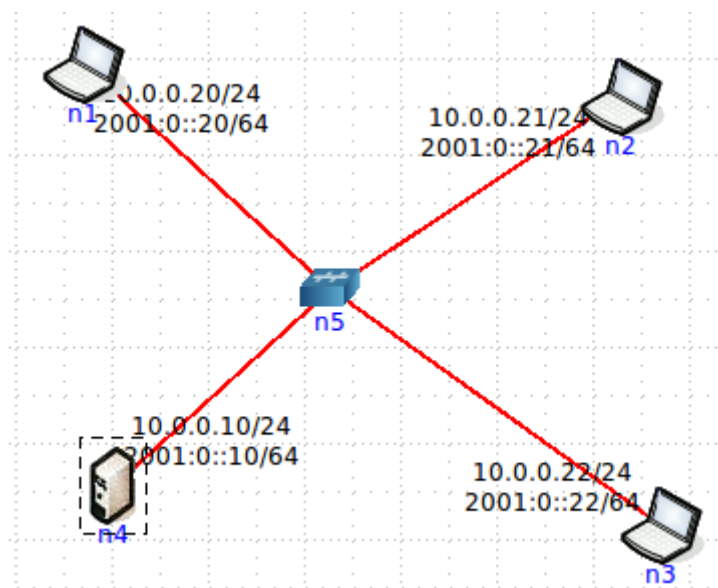
R: Enquanto num arp normal se procura o Mac correspondente a um determinado IP, no gratuito, a maquina questiona-se a si mesma pelo Mac correspondente ao próprio IP, para descobrir se tem mais alguma maquina a usar o nosso IP.

Esta é a trama correspondente ao arp gratuito:

|      |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |          |
|------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-------|----------|
| 0000 | ff | ff | ff | ff | ff | ff | 10 | bf | 48 | 08 | 85 | 76 | 08 | 06 | 00 | 01 | ..... | H..v.... |
| 0010 | 08 | 00 | 06 | 04 | 00 | 01 | 10 | bf | 48 | 08 | 85 | 76 | c0 | a8 | 64 | bb | ..... | H..v..d. |
| 0020 | 00 | 00 | 00 | 00 | 00 | 00 | c0 | a8 | 64 | bb |    |    |    |    |    |    | ..... | d.       |

R: O resultado final esperado será “verdadeiro” se a nossa máquina for a única a usar este endereço IP.

3. Faça ping de n2 para n4. Verifique com a opção tcpdump como flui o tráfego nas diversas interfaces dos vários dispositivos. Que conclui?



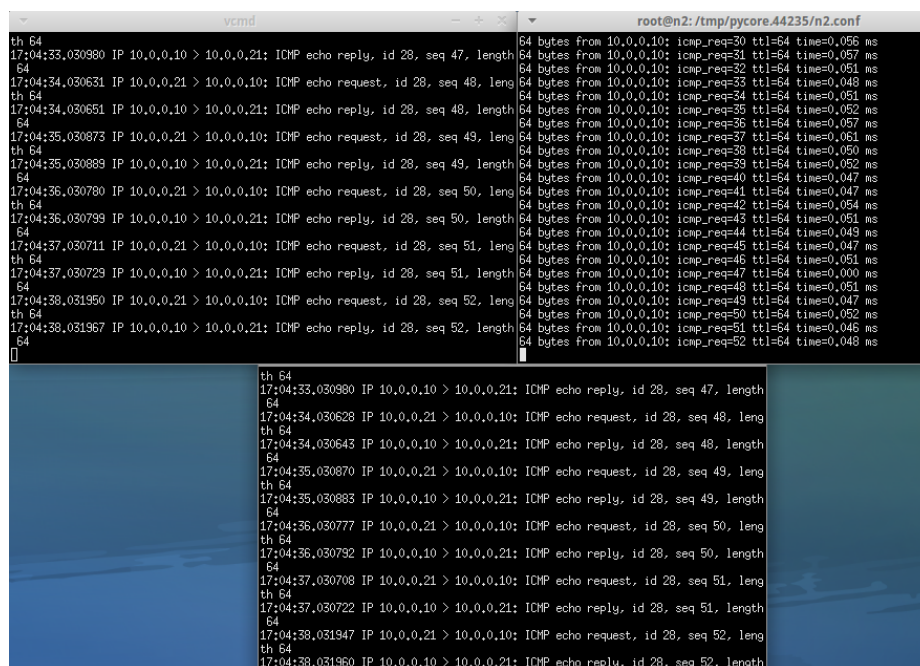
3.1 Faça ping de n2 para n4. Verifique com a opção tcpdump como flui o tráfego nas diversas interfaces dos vários dispositivos. Que conclui?

R: O pedido vai fluir por todas as máquinas conectadas ao hub, com pedidos e respostas.

3.2 Na topologia de rede substitua o hub por um switch. Repita os procedimentos que realizou na pergunta anterior. Comente os resultados obtidos quanto à utilização de hubs e switches no contexto de controlar ou dividir domínios de colisão. Documente as suas observações e conclusões com base no tráfego observado/capturado.

R: Como já mencionado no enunciado, os switches eliminam as colisões, conectando cada dispositivo a uma porta do computador, enquanto nos hubs estas colisões podem existir. Na utilização do core, ao utilizar um hub para comunicar entre uma máquina e um servidor, também passava tráfego pelas outras máquinas do esquema, no entanto, ao mudarmos para o switch observamos que apenas passava tráfego pela máquina e pelo servidor ao efetuar o ping, cortando assim, o tráfego pelas outras máquinas e evitando colisões.

#### HUB:



```
th 64
17:04:33,030980 IP 10.0.0.10 > 10.0.0.21: ICMP echo reply, id 28, seq 47, length 64 bytes from 10.0.0.10: icmp_req=30 ttl=64 time=0.056 ms
64
17:04:34,030631 IP 10.0.0.21 > 10.0.0.10: ICMP echo request, id 28, seq 48, length 64 bytes from 10.0.0.10: icmp_req=31 ttl=64 time=0.057 ms
th 64
17:04:34,030651 IP 10.0.0.10 > 10.0.0.21: ICMP echo reply, id 28, seq 48, length 64 bytes from 10.0.0.10: icmp_req=32 ttl=64 time=0.051 ms
64
17:04:35,030873 IP 10.0.0.21 > 10.0.0.10: ICMP echo request, id 28, seq 49, length 64 bytes from 10.0.0.10: icmp_req=33 ttl=64 time=0.048 ms
th 64
17:04:35,030888 IP 10.0.0.10 > 10.0.0.21: ICMP echo reply, id 28, seq 49, length 64 bytes from 10.0.0.10: icmp_req=34 ttl=64 time=0.051 ms
64
17:04:36,030780 IP 10.0.0.21 > 10.0.0.10: ICMP echo request, id 28, seq 50, length 64 bytes from 10.0.0.10: icmp_req=35 ttl=64 time=0.052 ms
th 64
17:04:36,030798 IP 10.0.0.10 > 10.0.0.21: ICMP echo reply, id 28, seq 50, length 64 bytes from 10.0.0.10: icmp_req=36 ttl=64 time=0.057 ms
64
17:04:37,030711 IP 10.0.0.21 > 10.0.0.10: ICMP echo request, id 28, seq 51, length 64 bytes from 10.0.0.10: icmp_req=37 ttl=64 time=0.061 ms
th 64
17:04:37,030728 IP 10.0.0.10 > 10.0.0.21: ICMP echo reply, id 28, seq 51, length 64 bytes from 10.0.0.10: icmp_req=38 ttl=64 time=0.050 ms
64
17:04:38,031950 IP 10.0.0.21 > 10.0.0.10: ICMP echo request, id 28, seq 52, length 64 bytes from 10.0.0.10: icmp_req=39 ttl=64 time=0.052 ms
th 64
17:04:38,031967 IP 10.0.0.10 > 10.0.0.21: ICMP echo reply, id 28, seq 52, length 64 bytes from 10.0.0.10: icmp_req=40 ttl=64 time=0.047 ms
64
17:04:39,032000 IP 10.0.0.21 > 10.0.0.10: ICMP echo request, id 28, seq 53, length 64 bytes from 10.0.0.10: icmp_req=41 ttl=64 time=0.047 ms
th 64
17:04:39,032017 IP 10.0.0.10 > 10.0.0.21: ICMP echo reply, id 28, seq 53, length 64 bytes from 10.0.0.10: icmp_req=42 ttl=64 time=0.054 ms
64
17:04:40,032034 IP 10.0.0.21 > 10.0.0.10: ICMP echo request, id 28, seq 54, length 64 bytes from 10.0.0.10: icmp_req=43 ttl=64 time=0.051 ms
th 64
17:04:40,032051 IP 10.0.0.10 > 10.0.0.21: ICMP echo reply, id 28, seq 54, length 64 bytes from 10.0.0.10: icmp_req=44 ttl=64 time=0.049 ms
64
17:04:41,032068 IP 10.0.0.21 > 10.0.0.10: ICMP echo request, id 28, seq 55, length 64 bytes from 10.0.0.10: icmp_req=45 ttl=64 time=0.047 ms
th 64
17:04:41,032085 IP 10.0.0.10 > 10.0.0.21: ICMP echo reply, id 28, seq 55, length 64 bytes from 10.0.0.10: icmp_req=46 ttl=64 time=0.051 ms
64
17:04:42,032102 IP 10.0.0.21 > 10.0.0.10: ICMP echo request, id 28, seq 56, length 64 bytes from 10.0.0.10: icmp_req=47 ttl=64 time=0.000 ms
th 64
17:04:42,032119 IP 10.0.0.10 > 10.0.0.21: ICMP echo reply, id 28, seq 56, length 64 bytes from 10.0.0.10: icmp_req=48 ttl=64 time=0.051 ms
64
17:04:43,032136 IP 10.0.0.21 > 10.0.0.10: ICMP echo request, id 28, seq 57, length 64 bytes from 10.0.0.10: icmp_req=49 ttl=64 time=0.047 ms
th 64
17:04:43,032153 IP 10.0.0.10 > 10.0.0.21: ICMP echo reply, id 28, seq 57, length 64 bytes from 10.0.0.10: icmp_req=50 ttl=64 time=0.047 ms
64
17:04:44,032170 IP 10.0.0.21 > 10.0.0.10: ICMP echo request, id 28, seq 58, length 64 bytes from 10.0.0.10: icmp_req=51 ttl=64 time=0.046 ms
th 64
17:04:44,032187 IP 10.0.0.10 > 10.0.0.21: ICMP echo reply, id 28, seq 58, length 64 bytes from 10.0.0.10: icmp_req=52 ttl=64 time=0.048 ms
64
```

Como podemos observar na imagem, ao fazer ping da máquina n2, para o servidor n4, para além do tráfego a passar pelo servidor, iremos ter também tráfego a passar também pela máquina n1.



## SWITCH

```
vcmd root@n2: /tmp/pycore.44236/n2.conf
17:07:48,658578 IP 10.0.0.22 > 10.0.0.10: ICMP echo request, id 28, seq 6, length 64
17:07:48,658590 IP 10.0.0.10 > 10.0.0.22: ICMP echo reply, id 28, seq 6, length 64
17:07:48,675267 ARP, Request who-has 10.0.0.22 tell 10.0.0.10, length 28
17:07:48,675376 ARP, Reply 10.0.0.22 is-at 00:00:00:aa:00:03, length 28
17:07:49,658577 IP 10.0.0.22 > 10.0.0.10: ICMP echo request, id 28, seq 7, length 64
17:07:49,658590 IP 10.0.0.10 > 10.0.0.22: ICMP echo reply, id 28, seq 7, length 64
17:07:50,658541 IP 10.0.0.22 > 10.0.0.10: ICMP echo request, id 28, seq 8, length 64
17:07:50,658552 IP 10.0.0.10 > 10.0.0.22: ICMP echo reply, id 28, seq 8, length 64
17:07:51,658546 IP 10.0.0.22 > 10.0.0.10: ICMP echo request, id 28, seq 9, length 64
17:07:51,658557 IP 10.0.0.10 > 10.0.0.22: ICMP echo reply, id 28, seq 9, length 64
17:07:52,659191 IP 10.0.0.22 > 10.0.0.10: ICMP echo request, id 28, seq 10, length 64
17:07:52,659201 IP 10.0.0.10 > 10.0.0.22: ICMP echo reply, id 28, seq 10, length 64
root@n2: /tmp/pycore.44236/n2.conf# ping 10.0.0.10
PING 10.0.0.10 (10.0.0.10) 56(84) bytes of data:
64 bytes from 10.0.0.10: icmp_req=1 ttl=64 time=0.079 ms
64 bytes from 10.0.0.10: icmp_req=2 ttl=64 time=0.043 ms
64 bytes from 10.0.0.10: icmp_req=3 ttl=64 time=0.043 ms
64 bytes from 10.0.0.10: icmp_req=4 ttl=64 time=0.043 ms
64 bytes from 10.0.0.10: icmp_req=5 ttl=64 time=0.045 ms
64 bytes from 10.0.0.10: icmp_req=6 ttl=64 time=0.043 ms
64 bytes from 10.0.0.10: icmp_req=7 ttl=64 time=0.047 ms
64 bytes from 10.0.0.10: icmp_req=8 ttl=64 time=0.043 ms
64 bytes from 10.0.0.10: icmp_req=9 ttl=64 time=0.040 ms
64 bytes from 10.0.0.10: icmp_req=10 ttl=64 time=0.042 ms
^C
root@n2: /tmp/pycore.44236/n2.conf#
listening on eth0, link-type ETHERNET, capture size 65535 bytes
17:07:20,626671 IP6 fe80::2861:69ff:fe31:44c8 > ff02::1:6: HBH ICMP6, multicast 1
listener report v2, 1 group record(s), length 28
17:07:43,660130 ARP, Request who-has 10.0.0.10 tell 10.0.0.22, length 28
^C
```

Já com a utilização do switch, conseguimos observar que apenas esta a passar tráfego pelo servidor e pela maquina n2, que fez o ping.

## **CONCLUSAO**

Neste relatório foram abordados vários temas de forma genérica, dos quais, a camada de ligação lógica, focando o uso da tecnologia Ethernet, o protocolo ARP. Na parte da ligação lógica abordamos transferência de dados, detecção e correção de erros, protocolos de acesso de controlo de ligação, endereços MAC, Address Resolution Protocol, Ethernet e interligação de redes locais.

Capturamos e analisámos tramas Ethernet, verificando endereços MAC(origem e destino), tipos e respostas de servidores.

Na parte do protocolo ARP abordamos o arp gratuito e domínios de colisão. Utilizando o CORE testamos diversos tipos de ligações, as diferenças entre HUB's e Switch's.