



# Redes de Computadores - Relatório TP3

João Nunes (A82300)      Luís Braga (A82088)  
Luís Martins (A82298)  
Grupo 57

16 de Novembro de 2018

# 1 Questões e Respostas

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

Os endereços origem e destino são dados respectivamente por 4c:cc:6a:e1:c5:49 e 00:0c:29:d2:19:f0, como se pode verificar na figura 1.

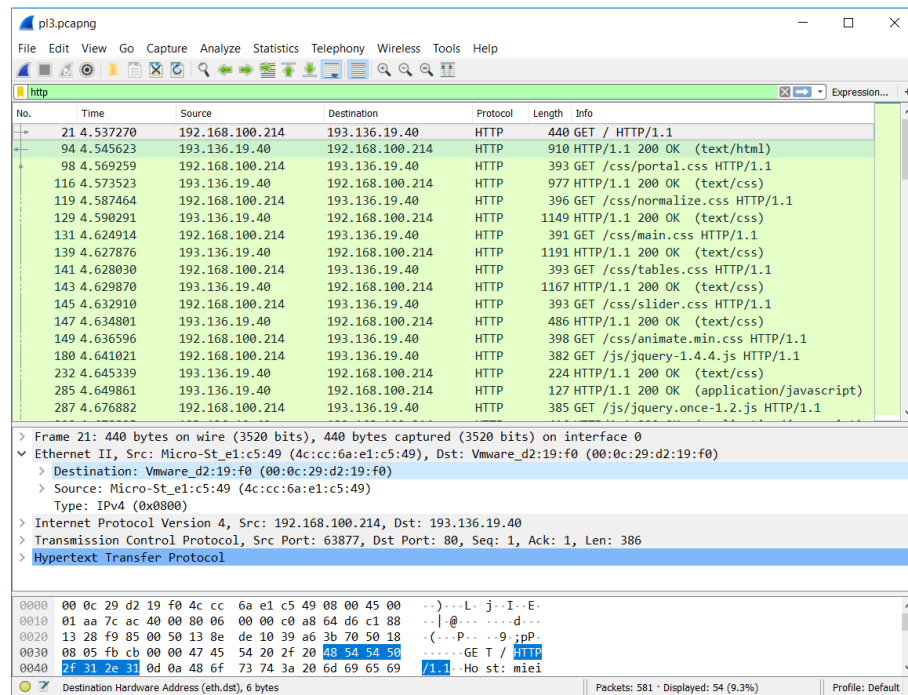


Figura 1: Printscreen da primeira trama ethernet que contém a mensagem HTTP GET.

## 2. Identifique que sistemas se referem. Justifique.

O endereço MAC origem pertence à interface ativa do computador em que está a ser feita a captura, o endereço MAC destino não se refere ao servidor mas sim ao router adjacente, uma vez que o nível de ligação destes serviços são prestados na ligação entre nós adjacentes.

## 3. Qual o valor hexadecimal do campo type da trama ethernet? O que significa?

O valor hexadecimal do campo type da trama (0x0800) significa que campo de dados são pacotes IPv4.

```

> Frame 21: 440 bytes on wire (3520 bits), 440 bytes captured (3520 bits) on interface 0
> Ethernet II, Src: Micro-St_e1:c5:49 (4c:cc:6a:e1:c5:49), Dst: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
> Destination: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
> Source: Micro-St_e1:c5:49 (4c:cc:6a:e1:c5:49)
  Type: IPv4 (0x0800)
> Internet Protocol Version 4, Src: 192.168.100.214, Dst: 193.136.19.40
> Transmission Control Protocol, Src Port: 63877, Dst Port: 80, Seq: 1, Ack: 1, Len: 386
> Hypertext Transfer Protocol

0000  00 0c 29 d2 19 f0 4c cc 6a e1 c5 49 08 00 45 00  ..)...L. j..I..E.
0010  01 aa 7c ac 40 00 80 06 00 00 c0 a8 64 d6 c1 88  ..|. @... ..d...
0020  13 28 f9 85 00 50 13 8e de 10 39 a6 3b 70 50 18  -(...P... ..9;pP-
0030  08 05 fb cb 00 00 47 45 54 20 2f 20 48 54 54 50  ....GET / HTTP
0040  2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 6d 69 65 69  /1.1..Ho st: miei

```

Figura 2: Printscreen da primeiro pacote que contem a mensagem HTTP GET.

4. Quantos bytes são usados deste 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.

Desde o início da trama até ao caractere "G"são usados 54 bytes, uma vez que o GET não pertence à pilha protocolar.

A sobrecarga é dada por  $54/440 = 12.27\%$ .

```

> Frame 21: 440 bytes on wire (3520 bits), 440 bytes captured (3520 bits) on interface 0
> Ethernet II, Src: Micro-St_e1:c5:49 (4c:cc:6a:e1:c5:49), Dst: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
> Internet Protocol Version 4, Src: 192.168.100.214, Dst: 193.136.19.40
> Transmission Control Protocol, Src Port: 63877, Dst Port: 80, Seq: 1, Ack: 1, Len: 386
> Hypertext Transfer Protocol
  GET / HTTP/1.1\r\n
    [Expert Info (Chat/Sequence): GET / HTTP/1.1\r\n]

0000  00 0c 29 d2 19 f0 4c cc 6a e1 c5 49 08 00 45 00  ..)...L. j..I..E.
0010  01 aa 7c ac 40 00 80 06 00 00 c0 a8 64 d6 c1 88  ..|. @... ..d...
0020  13 28 f9 85 00 50 13 8e de 10 39 a6 3b 70 50 18  -(...P... ..9;pP-
0030  08 05 fb cb 00 00 47 45 54 20 2f 20 48 54 54 50  ....GET / HTTP
0040  2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 6d 69 65 69  /1.1..Ho st: miei
0050  2e 64 69 2e 75 6d 69 6e 68 6f 2e 70 74 0d 0a 43  .di.umin ho.pt..C
0060  6f 6e 6e 65 63 74 69 6f 6e 3a 20 6b 65 65 70 2d  onnectio n: keep-
0070  61 6c 69 76 65 0d 0a 55 70 67 72 61 64 65 2d 49  alive..U pgrade-I
0080  6e 73 65 63 75 72 65 2d 52 65 71 75 65 73 74 73  nsecure- Requests
0090  3a 20 31 0d 0a 55 73 65 72 2d 41 67 65 6e 74 3a  : 1..Use r-Agent:
00a0  20 4d 6f 7a 69 6c 6c 61 2f 35 2e 30 20 28 57 69  Mozilla /5.0 (Wi
00b0  6e 64 6f 77 73 20 4e 54 20 31 30 2e 30 3b 20 57  ndows NT 10.0; W
00c0  69 6e 36 34 3b 20 78 36 34 29 20 41 70 70 6c 65  in64; x6 4) Apple
00d0  57 65 62 4b 69 74 2f 35 33 37 2e 33 36 20 28 4b  WebKit/5.37.36 (K
00e0  48 54 4d 4c 2c 20 6c 69 6b 65 20 47 65 63 6b 6f  HTML, li ke Gecko
00f0  29 20 43 68 72 6f 6d 65 2f 37 30 2e 30 2e 33 35  ) Chrome /70.0.35
0100  33 38 2e 31 30 32 20 53 61 66 61 72 69 2f 35 33  38.102 S afari/53
0110  37 2e 33 36 0d 0a 41 63 63 65 70 74 3a 20 74 65  7.36..Ac cept: te
0120  78 74 2f 68 74 6d 6c 2c 61 70 70 6c 69 63 61 74  xt/html, applicat
0130  69 6f 6e 2f 78 68 74 6d 6c 2b 78 6d 6c 2c 61 70  ion/xhtm l+xml,ap
0140  70 6c 69 63 61 74 69 6f 6e 2f 78 6d 6c 3b 71 3d  plicatio n/xml;q=
0150  30 2e 39 2c 69 6d 61 67 65 2f 77 65 62 70 2c 69  0.9,imag e/webp,i
0160  6d 61 67 65 2f 61 70 6e 67 2c 2a 2f 2a 3b 71 3d  mage/apn g,*/*;q=
0170  30 2e 38 0d 0a 41 63 63 65 70 74 2d 45 6e 63 6f  0.8..Acc ept-Enco
0180  64 69 6e 67 3a 20 67 7a 69 70 2c 20 64 65 66 6c  ding: gz ip, defl
0190  61 74 65 0d 0a 41 63 63 65 70 74 2d 4c 61 6e 67  ate..Acc ept-Lang
01a0  75 61 67 65 3a 20 65 6e 2d 55 53 2c 65 6e 3b 71  uage: en -US,en;q
01b0  3d 30 2e 39 0d 0a 0d 0a  =0.9....

```

Figura 3: Localização do GET no campo hexadecimal da trama.

Frame 21: 440 bytes on wire (3520 bits), 440 bytes captured (3520 bits) on interface 0

Figura 4: Tamanho total de bytes do pacote.

**5. Através da visualização direta de uma trama capturada, verifique que, possivelmente, o campo FCS (Frame Check Sequence) usado para a detecção de erros não está a ser usado. Em sua opinião porque será?.**

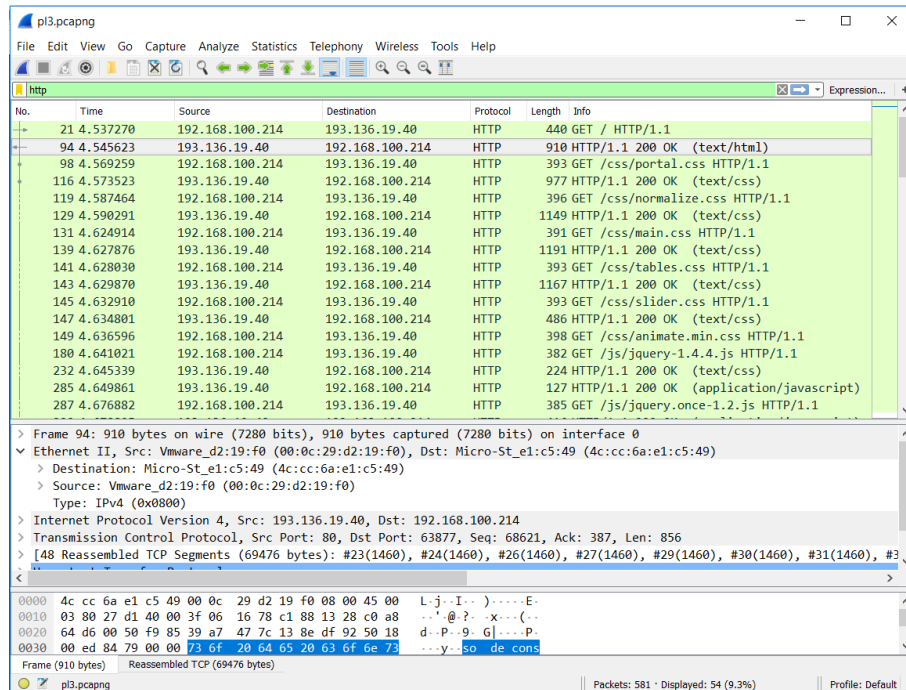
O campo FCS não está a ser usado, porque em ligações com fios pouco suscetíveis a erros, nem sempre as NICs geram o código de detecção de erros.

```
▼ Ethernet II, Src: Micro-St_e1:c5:49 (4c:cc:6a:e1:c5:49), Dst: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
  > Destination: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
  > Source: Micro-St_e1:c5:49 (4c:cc:6a:e1:c5:49)
  Type: IPv4 (0x0800)
```

Figura 5: Valor do campo Ethernet.

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

O endereço Ethernet da fonte é 00:0c:29:d2:19:f0 e corresponde ao router adjacente à máquina que o grupo usou, pois, em relação à pergunta 1, o endereço de origem e destino trocaram, uma vez que se trata da trama de resposta.



```

> Frame 94: 910 bytes on wire (7280 bits), 910 bytes captured (7280 bits) on interface 0
> Ethernet II, Src: Vmware_d2:19:f0 (00:0c:29:d2:19:f0), Dst: Micro-St_e1:c5:49 (4c:cc:6a:e1:c5:49)
> Internet Protocol Version 4, Src: 193.136.19.40, Dst: 192.168.100.214
> Transmission Control Protocol, Src Port: 80, Dst Port: 63877, Seq: 68621, Ack: 387, Len: 856
> [48 Reassembled TCP Segments (69476 bytes): #23(1460), #24(1460), #26(1460), #27(1460), #29(1460), #30(1460), #31(1460), #33(1460), i
> Hypertext Transfer Protocol
> Line-based text data: text/html (1573 lines)

```

Figura 7: Campos da trama Ethernet que contêm o primeiro byte de resposta HTTP.

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

A primeira tabela contém o endereço IP de uma determinada máquina, na segunda coluna estão discriminados os endereços MAC do nó adjacente à nossa máquina. Portanto, se quisermos comunicar com um IP da tabela teremos de enviar a informação para o nó adjacente com o MAC respectivo. E ainda temos a coluna *Type* que determina o tipo do endereço.

```

C:\Users\luisb>arp -a

Interface: 192.168.56.1 --- 0x8
    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
    224.0.0.251           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

Interface: 172.26.67.41 --- 0xd
    Internet Address      Physical Address      Type
    172.26.254.254        00-d0-03-ff-94-00    dynamic
    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
    255.255.255.255       ff-ff-ff-ff-ff-ff    static

```

Figura 8: Tabela ARP.

## 10. 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?)

O endereço de origem em hexadecimal é 88:d7:f6:1b:30:32 e o endereço de destino é o ff:ff:ff:ff:ff:ff. O endereço de destino usado é esse, porque como não sabemos qual é o endereço de ip destino é feito um broadcast para o nó adjacente, onde

depois ocorre um *flood* onde todas as máquinas adjacentes recebem o pedido mas só a pretendida há de responder.

```
C:\WINDOWS\system32>ping 192.168.100.207

Pinging 192.168.100.207 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 192.168.100.207:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Figura 9: Resultado do comando ping.

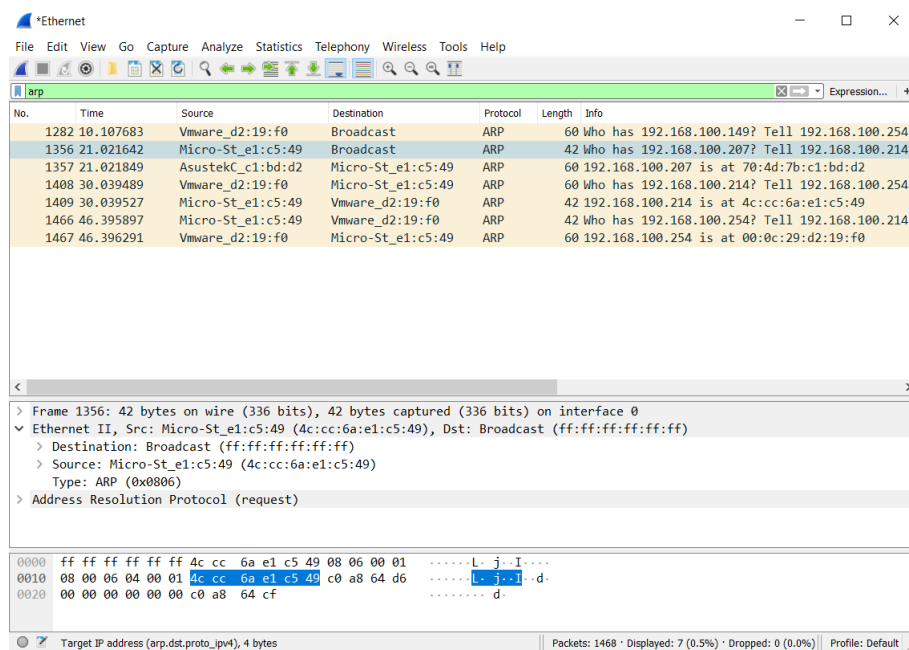


Figura 10: Printscreen dos pacotes ARP.

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

O valor do campo tipo é 0x0806 e indica que o protocolo do pacote é ARP, como se pode verificar na figura 10.

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

O campo ARP opcode é 1, que significa que é um request.

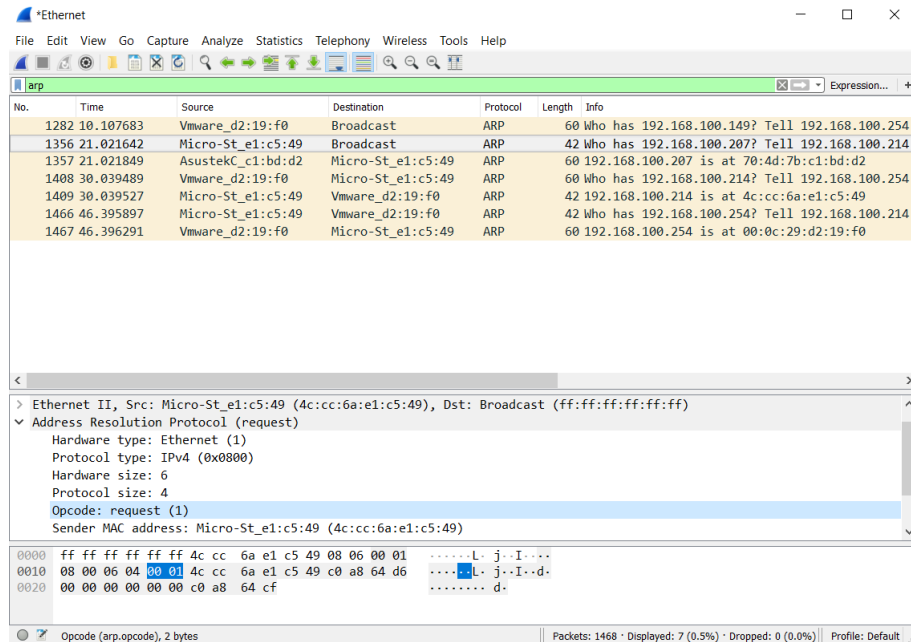


Figura 11: Valor do campo opcode.

## 13. Identifique que tipo de endereços estão contidos na mensagem ARP? Que conclui?

Os endereços contidos na mensagem ARP são os endereços IP e os endereços de ligação lógicos (MAC adress). Uma vez que sabemos qual é o endereço IP destino, o protocolo ARP permite que dispositivos da mesma network perguntem entre si quais os endereços MAC que têm o IP em questão. Todas as máquinas adjacentes recebem o pedido, mas apenas a pretendida responde ao request.

## 14. Explícite que tipo de pedido ou pergunta é feita pelo host de origem?

É feita um pedido onde se pergunta qual é o MAC adress do endereço IP destino.

42 Who has 192.168.100.207? Tell 192.168.100.214

Figura 12: Pedido feito pelo host de origem.



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

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

O valor do campo ARP opcode é 2, o que significa que é um reply.

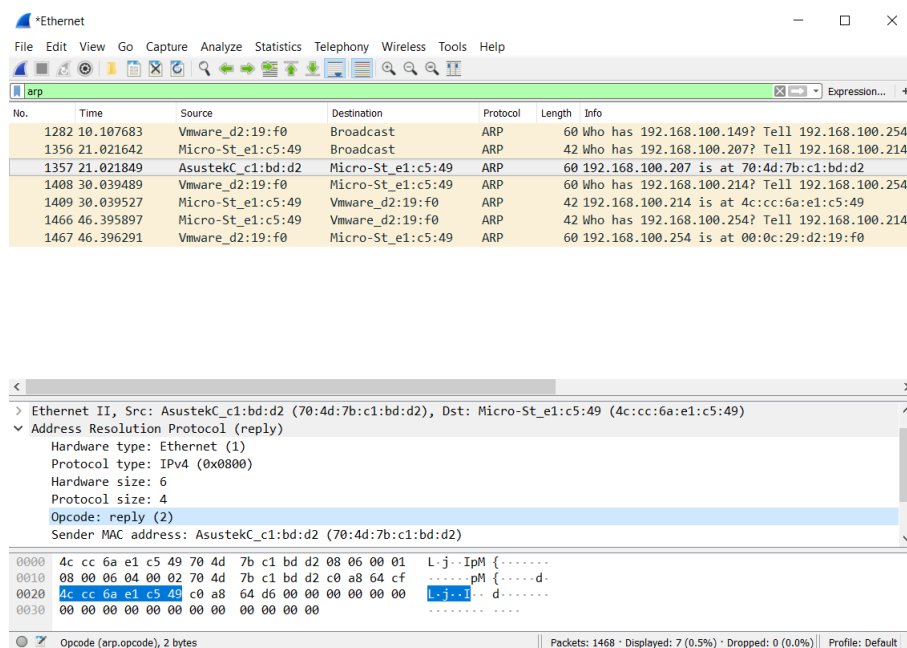


Figura 13: Primeira mensagem ARP reposta.

b. Em que posição da mensagem ARP está a resposta ao pedido ARP?

Está localizada nos bytes 22-26 e é identificado como o Sender MAC Address.

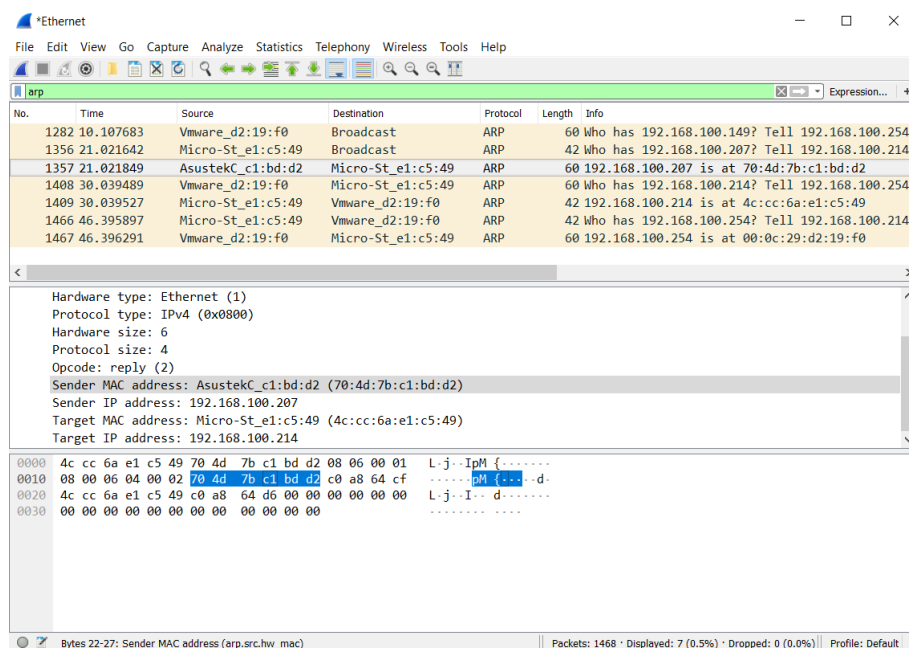


Figura 14: Posição da mensagem ARP.

**16. Identifique um pacote de pedido ARP gratuito originado pelo seu sistema. 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?**

O pedido ARP gratuito distingue-se na maneira em que envia para o mesmo ip da máquina um ARP request, de modo a anunciar a toda a rede o novo endereço físico que entrou na rede, de modo a todas as outras máquinas atualizarem as suas tabelas ARP.

As principais mudanças são a informação sobre a mensagem do pacote (normalmente é um request a perguntar) e é adicionado o campo Is Gratuitous no campo ARP.

O resultado esperado é que o novo endereço seja "anunciado" e as outras tabelas ARP da rede sejam atualizadas.

No.	Time	Source	Destination	Protocol	Length	Info
4	0.033585	Micro-St_e1:c5:49	Broadcast	ARP	42	Who has 192.168.100.254? Tell 192.168.100.214
5	0.197938	Micro-St_e1:c5:49	Broadcast	ARP	42	Who has 192.168.100.214? Tell 0.0.0.0
10	0.352598	Vmware_d2:19:f0	Broadcast	ARP	60	Who has 192.168.100.208? Tell 192.168.100.254
12	0.551470	Micro-St_e1:c5:49	Broadcast	ARP	42	Who has 192.168.100.254? Tell 192.168.100.214
13	0.551804	Vmware_d2:19:f0	Micro-St_e1:c5:49	ARP	60	192.168.100.254 is at 00:0c:29:d2:19:f0
20	1.197110	Micro-St_e1:c5:49	Broadcast	ARP	42	Who has 192.168.100.214? Tell 0.0.0.0
36	1.353158	Vmware_d2:19:f0	Broadcast	ARP	60	Who has 192.168.100.208? Tell 192.168.100.254
90	2.197439	Micro-St_e1:c5:49	Broadcast	ARP	42	Who has 192.168.100.214? Tell 0.0.0.0
100	3.197145	Micro-St_e1:c5:49	Broadcast	ARP	42	Gratuitous ARP for 192.168.100.214 (Request)
561	8.237170	Vmware_d2:19:f0	Micro-St_e1:c5:49	ARP	60	Who has 192.168.100.214? Tell 192.168.100.254
562	8.237216	Micro-St_e1:c5:49	Vmware_d2:19:f0	ARP	42	192.168.100.214 is at 4c:cc:6a:e1:c5:49

> Frame 100: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0  
 > Ethernet II, Src: Micro-St\_e1:c5:49 (4c:cc:6a:e1:c5:49), Dst: Broadcast (ff:ff:ff:ff:ff:ff)  
 > Address Resolution Protocol (request/gratuitous ARP)  
   Hardware type: Ethernet (1)  
   Protocol type: IPv4 (0x0800)  
   Hardware size: 6  
   Protocol size: 4  
   Opcode: request (1)  
   [Is gratuitous: True]  
   Sender MAC address: Micro-St\_e1:c5:49 (4c:cc:6a:e1:c5:49)  
   Sender IP address: 192.168.100.214

0000 ff ff ff ff ff 4c cc 6a e1 c5 49 08 06 00 01 .....L. j..I...  
 0010 08 00 06 04 00 01 4c cc 6a e1 c5 49 c0 a8 64 d6 .....L. j..I..d..  
 0020 00 00 00 00 00 00 c0 a8 64 d6 .....d.

Is gratuitous (arp.isgratuitous)    Packets: 645 · Displayed: 11 (1.7%)    Profile: Default

Figura 15: Informação sobre o pacote ARP gratuito.

17. Faça *ping* de n1 para n2. Verifique com a opção *tcpdump* como foi o tráfego nas diversas interfaces dos vários dispositivos. Que conclui?

A topologia CORE utilizada foi a seguinte:

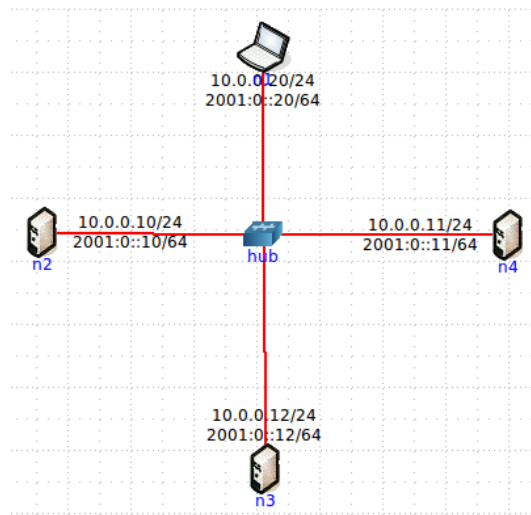
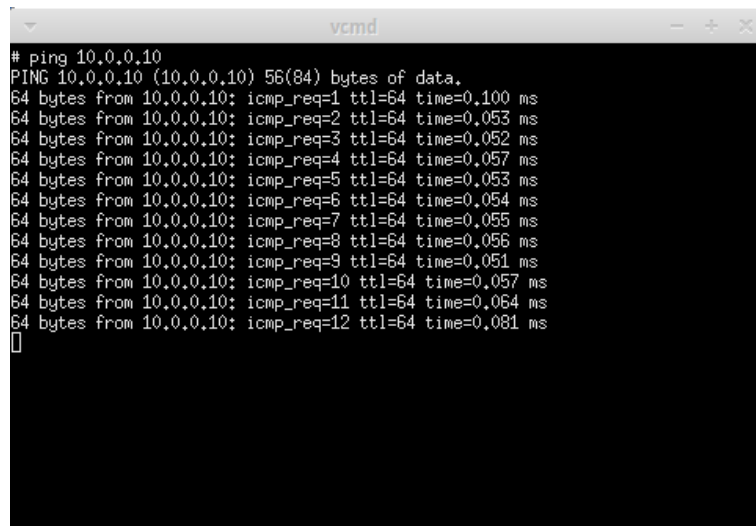


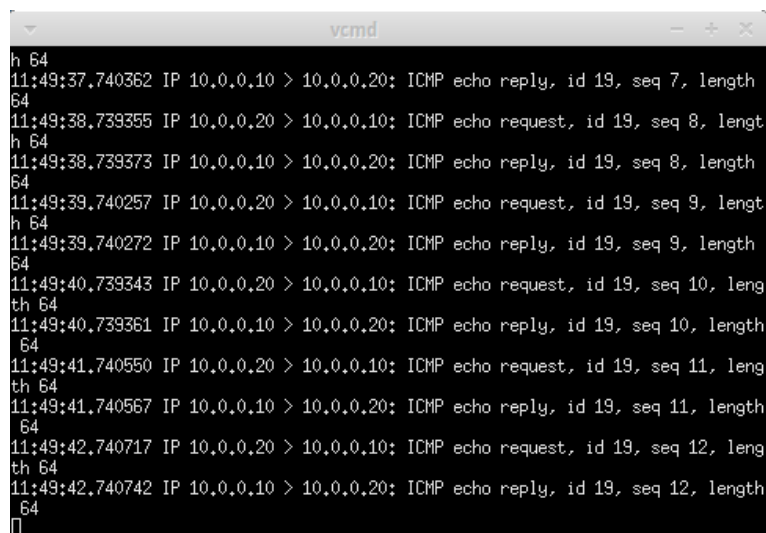
Figura 16: Topologia CORE com hub.

Ao fazer o ping do n1 para o host n2 (figura 17), a mensagem é transmitida para o hub. Os hubs são dispositivos de interligação, que repetem o sinal que chega através da porta de entrada para todas as outras portas. Desta maneira, quando um hub recebe a mensagem do n1, este repete a mensagem para o n2, n3 e o n4, sendo possível verificar o mesmo no tcpdump da figura 18, 19 e 20.



```
vcmd
# 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=0.100 ms
64 bytes from 10.0.0.10: icmp_req=2 ttl=64 time=0.053 ms
64 bytes from 10.0.0.10: icmp_req=3 ttl=64 time=0.052 ms
64 bytes from 10.0.0.10: icmp_req=4 ttl=64 time=0.057 ms
64 bytes from 10.0.0.10: icmp_req=5 ttl=64 time=0.053 ms
64 bytes from 10.0.0.10: icmp_req=6 ttl=64 time=0.054 ms
64 bytes from 10.0.0.10: icmp_req=7 ttl=64 time=0.055 ms
64 bytes from 10.0.0.10: icmp_req=8 ttl=64 time=0.056 ms
64 bytes from 10.0.0.10: icmp_req=9 ttl=64 time=0.051 ms
64 bytes from 10.0.0.10: icmp_req=10 ttl=64 time=0.057 ms
64 bytes from 10.0.0.10: icmp_req=11 ttl=64 time=0.064 ms
64 bytes from 10.0.0.10: icmp_req=12 ttl=64 time=0.081 ms
□
```

Figura 17: Resultado do comando ping feito do n1 para o n2.



```
vcmd
h 64
11:49:37.740362 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 7, length
64
11:49:38.739355 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 8, lengt
h 64
11:49:38.739373 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 8, length
64
11:49:39.740257 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 9, lengt
h 64
11:49:39.740272 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 9, length
64
11:49:40.739343 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 10, leng
th 64
11:49:40.739361 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 10, length
64
11:49:41.740550 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 11, leng
th 64
11:49:41.740567 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 11, length
64
11:49:42.740717 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 12, leng
th 64
11:49:42.740742 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 12, length
64
□
```

Figura 18: Resultado do tcpdump no n2.

```
vcmd
h 64
11:49:37.740365 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 7, length
64
11:49:38.739350 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 8, lengt
h 64
11:49:38.739377 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 8, length
64
11:49:39.740253 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 9, lengt
h 64
11:49:39.740275 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 9, length
64
11:49:40.739338 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 10, lengt
h 64
11:49:40.739365 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 10, length
64
11:49:41.740543 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 11, lengt
h 64
11:49:41.740572 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 11, length
64
11:49:42.740710 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 12, lengt
h 64
11:49:42.740747 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 12, length
64
```

Figura 19: Resultado do tcpdump no n3.

```
vcmd
h 64
11:49:37.740366 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 7, length
64
11:49:38.739353 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 8, lengt
h 64
11:49:38.739378 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 8, length
64
11:49:39.740255 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 9, lengt
h 64
11:49:39.740276 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 9, length
64
11:49:40.739341 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 10, lengt
h 64
11:49:40.739366 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 10, length
64
11:49:41.740547 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 11, lengt
h 64
11:49:41.740573 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 11, length
64
11:49:42.740713 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 12, lengt
h 64
11:49:42.740749 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 12, length
64
```

Figura 20: Resultado do tcpdump no n4.

18. 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.

A topologia CORE utilizada foi a seguinte:

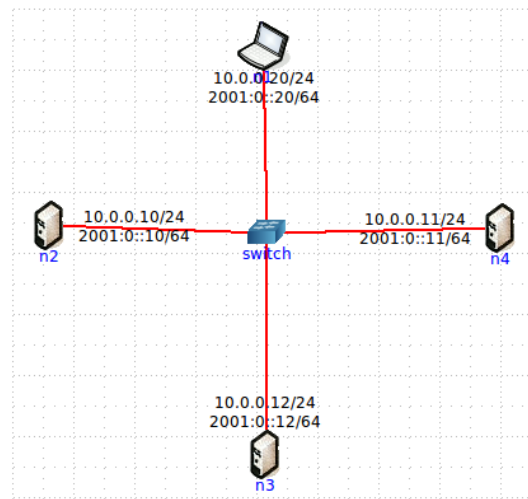


Figura 21: Topologia CORE com switch.

O hub neste caso foi substituído por um switch, os resultados obtidos são diferentes. Quando o laptop n1 faz ping para o host n2 (figura 22), a mensagem é transmitida para o switch. No entanto, ao contrário de repetir o sinal como o hub, este recebe a mensagem e transmite para o host pretendido. Desta forma, quando verificamos os resultados das figuras 23 à 25, verificando o tcpdump da figura 24 e 25, os hosts n3 e n4 estão inativos, não efetuam o *listen* como acontece com o hub.

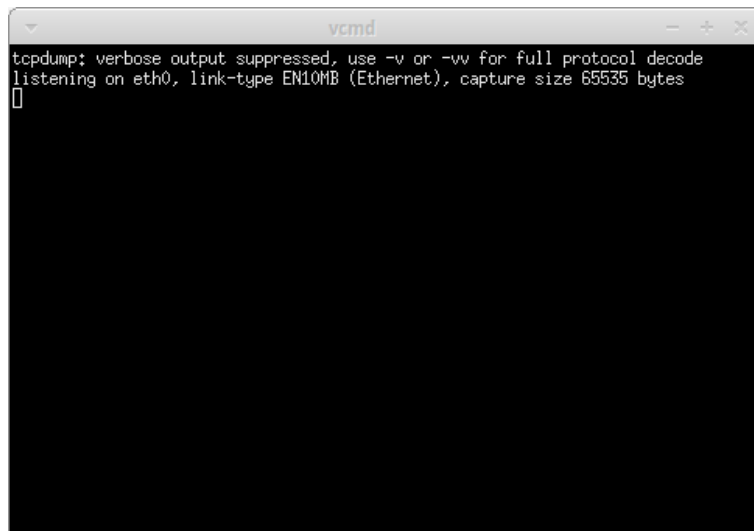
Com base nos resultados obtidos dos hubs e switches no contexto de controlar ou dividir os domínios de colisão, podemos afirmar que, como os hubs repetem as mensagens para todos os nodos ligados a ele, ao apenas ter um canal de comunicação, a probabilidade de colisão é muito grande, para além da necessidade de repetir o envio da mensagem. No entanto, os switches têm o principal objetivo de eliminar as colisões, uma vez que limitam o envio de mensagens apenas para o destino pretendido, funcionando como uma espécie de uma *ponte*. Desta forma, ao ter várias portas para cada interface, garante vários domínios de colisão, o que diminui a probabilidade de colisão, sendo esta a melhor escolha no que toca no contexto de controlar e dividir os domínios de colisão.

```
vcmd
64 bytes from 10.0.0.10: icmp_req=70 ttl=64 time=0,067 ms
64 bytes from 10.0.0.10: icmp_req=71 ttl=64 time=0,067 ms
64 bytes from 10.0.0.10: icmp_req=72 ttl=64 time=0,066 ms
64 bytes from 10.0.0.10: icmp_req=73 ttl=64 time=0,067 ms
64 bytes from 10.0.0.10: icmp_req=74 ttl=64 time=0,066 ms
64 bytes from 10.0.0.10: icmp_req=75 ttl=64 time=0,046 ms
64 bytes from 10.0.0.10: icmp_req=76 ttl=64 time=0,048 ms
64 bytes from 10.0.0.10: icmp_req=77 ttl=64 time=0,066 ms
64 bytes from 10.0.0.10: icmp_req=78 ttl=64 time=0,066 ms
64 bytes from 10.0.0.10: icmp_req=79 ttl=64 time=0,067 ms
64 bytes from 10.0.0.10: icmp_req=80 ttl=64 time=0,067 ms
64 bytes from 10.0.0.10: icmp_req=81 ttl=64 time=0,050 ms
64 bytes from 10.0.0.10: icmp_req=82 ttl=64 time=0,052 ms
64 bytes from 10.0.0.10: icmp_req=83 ttl=64 time=0,089 ms
64 bytes from 10.0.0.10: icmp_req=84 ttl=64 time=0,110 ms
64 bytes from 10.0.0.10: icmp_req=85 ttl=64 time=0,108 ms
64 bytes from 10.0.0.10: icmp_req=86 ttl=64 time=0,068 ms
64 bytes from 10.0.0.10: icmp_req=87 ttl=64 time=0,067 ms
64 bytes from 10.0.0.10: icmp_req=88 ttl=64 time=0,001 ms
64 bytes from 10.0.0.10: icmp_req=89 ttl=64 time=0,066 ms
64 bytes from 10.0.0.10: icmp_req=90 ttl=64 time=0,067 ms
64 bytes from 10.0.0.10: icmp_req=91 ttl=64 time=0,065 ms
64 bytes from 10.0.0.10: icmp_req=92 ttl=64 time=0,066 ms
```

Figura 22: Resultado do comando ping feito do n1 para o n2.

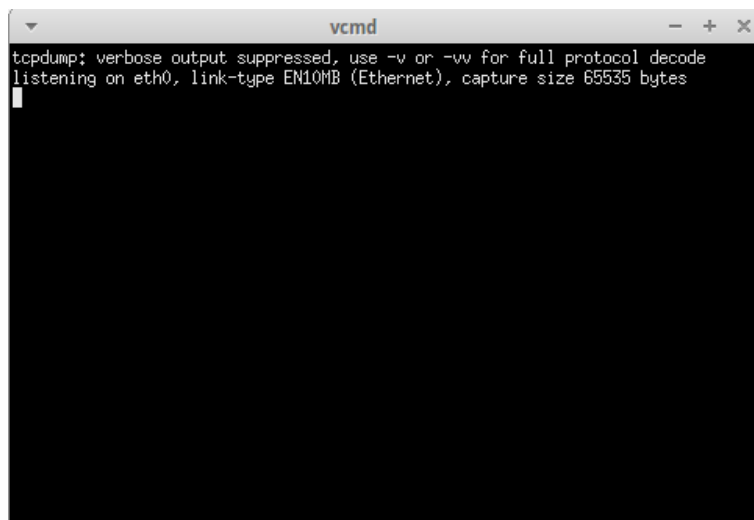
```
vcmd
th 64
11:47:22,595438 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 87, length
64
11:47:23,595724 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 88, leng
th 64
11:47:23,595725 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 88, length
64
11:47:24,595343 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 89, leng
th 64
11:47:24,595361 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 89, length
64
11:47:25,595401 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 90, leng
th 64
11:47:25,595420 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 90, length
64
11:47:26,596273 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 91, leng
th 64
11:47:26,596291 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 91, length
64
11:47:27,595399 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 19, seq 92, leng
th 64
11:47:27,595417 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 19, seq 92, length
64
```

Figura 23: Resultado do tcpdump no n2.



```
vcmd
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
█
```

Figura 24: Resultado do tcpdumpo no n3.



```
vcmd
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
█
```

Figura 25: Resultado do tcpdump no n4.



## 2 Conclusão

Este trabalho abordou temas como a detecção e correção de erros, protocolos de acesso e de controlo de ligação, endereços MAC, Address Resolution Protocol (ARP) e Ethernet, o que contribuiu para o grupo consolidar os conhecimentos relativamente aos temas em questão. Para além de fortalecer estes conhecimentos, a captura e análise de tramas Ethernet, através do Wireshark, foi bastante importante, uma vez que deu ao grupo uma visão mais realista das tramas e da sua constituição. A análise também ofereceu uma melhor percepção do mecanismo de envio e relação dos vários endereços existentes.

O estudo do protocolo ARP foi essencial também, no que toca na consolidação de conhecimentos relacionados com os mecanismos de mapeamento. O manuseamento de core, tal como no trabalho anterior, proporcionou uma noção ainda melhor do seu funcionamento, e as consequências de usar um certo tipo de equipamentos de ligação.