

Camada de Ligação Lógica: Ethernet e Protocolo ARP

João Monteiro, Mário Leite, and Miguel Pinto

University of Minho, Department of Informatics, 4710-057 Braga, Portugal
e-mail: {a53690,a61021,a61049}@alunos.uminho.pt

1 Questões e Respostas

Parte I

1.1 Captura e análise de Tramas Ethernet

Questão 1: Qual o endereço MAC da interface ativa do seu computador?

O endereço MAC da interface ativa é `10 : 9a : dd : 69 : d0 : 09`, como se pode comprovar pela imagem seguinte.

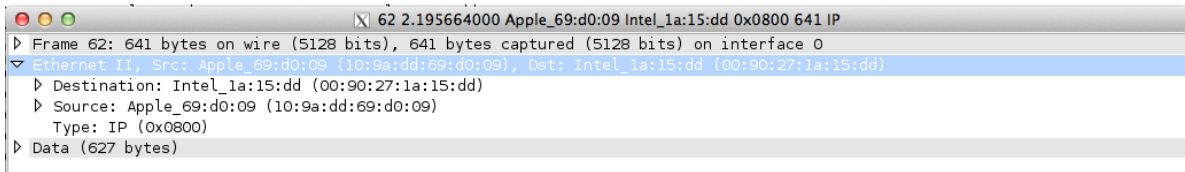


Fig. 1. Captura do Quadro Ethernet HTTP GET.

Questão 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 *marco.uminho.pt*? Justifique.

Pela análise do quadro anterior o endereço Mac destino da trama é `00.90.27.1a.15.dd`, sendo este endereço do router de destino.

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

O valor apresentado no campo type é `0x0800`, que corresponde a um pacote IP.

Questão 4: Quantos bytes são usados desde o início da trama até ao caractere ASCII "G" do método HTTP GET? Tente obter uma indicação do overhead introduzido pela pilha protocolar.

São usados 66 bytes até ao carácter 'G', 67º byte na trama.

0000	00 90 27 1a 15 dd 10 9a dd 69 d0 09 08 00 45 00	..'.i....E.
0010	02 73 a2 1c 40 00 40 06 00 00 c0 a8 64 c9 c1 88	.s..@.@.d...
0020	09 f0 ca 32 00 50 ce fe c1 c8 8b 9c 0c af 80 18	...2.P..
0030	20 2b f3 4f 00 00 01 01 08 0a 1a f5 9d 36 16 9b	+..0....6..
0040	a3 95 47 45 54 20 2f 43 43 47 2f 20 48 54 54 50	..GET /C CG/ HTTP

Fig. 2. Quadro referente aos bytes transmitidos.

Questão 5: Qual é o valor hexadecimal do campo FCS na trama capturada (poderá não estar a ser utilizado)?

O campo FCS não está a ser utilizado na trama capturada.

Questão 6: Qual é o endereço Ethernet da fonte? É o endereço do seu computador ou do servidor *marco.uminho.pt*? Qual o dispositivo que enviou a trama?

O endereço Ethernet da fonte é 00.90.27.1a.15.dd, que corresponde, como dito anteriormente, ao endereço do router.

```

> Frame 64: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface 0
▼ Ethernet II, Src: Intel_1a:15:dd (00:90:27:1a:15:dd), Dst: Apple_69:d0:09 (10:9a:dd:69:d0:09)
  > Destination: Apple_69:d0:09 (10:9a:dd:69:d0:09)
  > Source: Intel_1a:15:dd (00:90:27:1a:15:dd)
    Type: IP (0x0800)
  > Data (1500 bytes)

```

Fig. 3. Captura do Quadro Ethernet HTTP RESPONSE.

Questão 7: Qual é o endereço MAC do destino? Reconhece-o?

O endereço MAC do destino é: 10 : 9a : dd : 69 : d0.09. É o MAC do meu computador.

Questão 8: Qual o valor hexadecimal do campo tipo (*Type*)?

O valor hexadecimal é 0x0800, que corresponde a um valor IP.

Questão 9: Quantos bytes contém a trama Ethernet antes do caractere ASCII corresponde ao 200 OK (isto é, o código de resposta do HTTP)?

A trama Ethernet contém 75 bytes antes dos 200 OK.

0000	10 9a dd 69 d0 09 00 90 27 1a 15 dd 08 00 45 00	...i.... '.....E.
0010	05 dc a8 2c 40 00 3e 06 9e 05 c1 88 09 f0 c0 a8	...,@.>.
0020	64 c9 00 50 ca 32 8b 9c 0c af ce fe c4 07 80 10	d..P.2..
0030	00 7b 82 b1 00 00 01 01 08 0a 16 9b a4 84 1a f5	.{.....
0040	9d 36 48 54 54 50 2f 31 2e 31 20 32 30 30 20 4f	.6HTTP/1 .1 200 0
0050	4b 0d 0a 44 61 74 65 3a 20 54 75 65 2c 20 32 39	K..Date: Tue, 29

Fig. 4. Quadro referente aos bytes transmitidos.

Questão 10: Qual é o valor hexadecimal do campo FCS da trama Ethernet?

O campo FCS não está a ser utilizado na trama capturada.

1.2 Protocolo ARP

Questão 11: Verifique o conteúdo da cache ARP do seu computador.

Após o comando arp obteve-se o seguinte resultado, realizado em sistema OSx:

```
sh-3.2# arp -a
zonzhub.home (192.168.1.1) at 0:5:ca:ab:d5:25 on en1 ifscope [ethernet]
windows-phone.home (192.168.1.2) at 40:7a:80:e5:99:d4 on en1 ifscope [ethernet]
prettfortess.home (192.168.1.5) at 78:dd:8:db:cf:57 on en1 ifscope [ethernet]
? (192.168.1.255) at ff:ff:ff:ff:ff:ff on en1 ifscope [ethernet]
```

Fig. 5. Informação obtida após executado o comando arp.

Questão 12: Observe o conteúdo da tabela ARP. O que significa cada uma das colunas?

Pela informação obtida, a primeira coluna indica o nome do dispositivo ligado e o seu endereço IP, a segunda indica o endereço físico (endereço MAC) e a última mostra o tipo de ligação.

Questão 13: Qual é o valor hexadecimal dos endereços fonte e destino na trama Ethernet que contém a mensagem com o pedido ARP (ARP Request)? Como interpreta e justifica o endereço destino usado?

Pela análise do quadro com o filtro arp, podemos verificar que o endereço fonte é o 78:dd:08:db:cf:57 e o destino ff:ff:ff:ff:ff:ff. Todas as máquinas usam o mesmo endereço de destino, assim o servidor consegue responder ao arp com o seu MAC.

Filter: <input type="text" value="arp"/> Expression... Clear Apply Save						
No.	Time	Source	Destination	Protocol	Length	Info
473	39.128768000	HonHaiPr_db:cf:57	Broadcast	ARP	42	who has 192.168.1.3? Tell 192.168.1.5
474	39.743508000	HonHaiPr_db:cf:57	Broadcast	ARP	42	who has 192.168.1.54? Tell 192.168.1.5
475	39.747564000	HonHaiPr_db:cf:57	Broadcast	ARP	42	who has 192.168.1.100? Tell 192.168.1.5
476	39.748478000	HonHaiPr_db:cf:57	Broadcast	ARP	42	who has 192.168.1.85? Tell 192.168.1.5
478	40.049689000	HonHaiPr_db:cf:57	Broadcast	ARP	42	who has 192.168.1.4? Tell 192.168.1.5
479	40.049694000	HonHaiPr_db:cf:57	Broadcast	ARP	42	who has 192.168.1.3? Tell 192.168.1.5
480	40.049789000	HonHaiPr_db:cf:57	Broadcast	ARP	42	who has 192.168.1.75? Tell 192.168.1.5
481	40.049809000	Apple_18:04:d8	HonHaiPr_db:cf:57	ARP	42	192.168.1.4 is at e0:f8:47:18:04:d8
482	40.050010000	HonHaiPr_db:cf:57	Broadcast	ARP	42	who has 192.168.1.54? Tell 192.168.1.5
Frame 480: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0						
Ethernet II, Src: HonHaiPr_db:cf:57 (78:dd:08:db:cf:57), Dst: Broadcast (ff:ff:ff:ff:ff:ff)						
Destination: Broadcast (ff:ff:ff:ff:ff:ff)						
Source: HonHaiPr_db:cf:57 (78:dd:08:db:cf:57)						
Type: ARP (0x0806)						
Address Resolution Protocol (request)						
0000	ff ff ff ff ff ff 78 dd	08 db cf 57 08 06 00 01X. ...W....			
0010	08 00 06 04 00 01 78 dd	08 db cf 57 c0 a8 01 05X. ...W....			
0020	00 00 00 00 00 00 c0 a8	01 4bK			

Fig. 6. Captura Mensagem ARP.

Questão 14: Qual é o valor hexadecimal do campo tipo da trama Ethernet? O que significa?

```

> Frame 480: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
> Ethernet II, Src: HonHaiPr_db:cf:57 (78:dd:08:db:cf:57), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
  ▸ Destination: Broadcast (ff:ff:ff:ff:ff:ff)
  ▸ Source: HonHaiPr_db:cf:57 (78:dd:08:db:cf:57)
    Type: ARP (0x0806)
> Address Resolution Protocol (request)

```

Fig. 7. Quadro Ethernet em detalhe.

O valor é 0x0806 e é usado para indicar qual protocolo está encapsulado numa carga de um quadro Ethernet.

Questão 15: 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>.

O valor do campo opcode é o 0x0001 como se pode confirmar pela imagem abaixo. Indica o tipo de mensagem arp (pedido ou resposta) que neste caso é um pedido.

```

▼ Address Resolution Protocol (request)
  Hardware type: Ethernet (1)
  Protocol type: IP (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: request (1)
    Sender MAC address: HonHaiPr_db:cf:57 (78:dd:08:db:cf:57)
    Sender IP address: 192.168.1.5 (192.168.1.5)
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 192.168.1.75 (192.168.1.75)

```

0000	ff	ff	ff	ff	ff	ff	78	dd	08	db	cf	57	08	06	00	01x. ...W....
0010	08	00	06	04	00	01	78	dd	08	db	cf	57	c0	a8	01	05x. ...W....
0020	00	00	00	00	00	00	c0	a8	01	4b						K

Fig. 8. Campo ARP *opcode*.

Questão 16: A mensagem ARP contém o endereço IP do originador? Que tipo de pergunta é feita?

Contém o endereço IP do originador e do destinatário. A pergunta feita é: “Quem tem o endereço IP 192.168.1.75?”

Questão 17: Localize a mensagem ARP que é a resposta ao pedido ARP efectuado.

```

    ▾ 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_18:04:d8 (e0:f8:47:18:04:d8)
      Sender IP address: 192.168.1.4 (192.168.1.4)
      Target MAC address: HonHaiPr_db:cf:57 (78:dd:08:db:cf:57)
      Target IP address: 192.168.1.5 (192.168.1.5)
0000  78 dd 08 db cf 57 e0 f8 47 18 04 d8 08 06 00 01  x....W.. G.....
0010  08 00 06 04 00 02 e0 f8 47 18 04 d8 c0 a8 01 04  ..... G.....
0020  78 dd 08 db cf 57 c0 a8 01 05  x....W.. ..

```

Fig. 9. Mensagem ARP.

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

O valor é 0x0002 e indica que a mensagem arp é uma resposta.

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

Apartir do valor 0x0002 e os restantes bytes.

Questão 18: Quais são os valores hexadecimais para os endereços fonte e destino da trama que contém a resposta ARP? Que conclui?

Endereço fonte: e0:f8:47:18:04:d8

Endereço de destino: 78:dd:08:db:cf:57

Conclui-se que o endereço de destino da mensagem de resposta é o mesmo que o endereço fonte da mensagem de pedido (obteve-se uma resposta ao pedido).

```

    ▸ Frame 481: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
    ▾ Ethernet II, Src: Apple_18:04:d8 (e0:f8:47:18:04:d8), Dst: HonHaiPr_db:cf:57 (78:dd:08:db:cf:57)
      ▸ Destination: HonHaiPr_db:cf:57 (78:dd:08:db:cf:57)
      ▸ Source: Apple_18:04:d8 (e0:f8:47:18:04:d8)
      Type: ARP (0x0806)

```

Fig. 10. Quadro Ethernet Endereço Fonte e destino.

1.3 ARP numa topologia CORE

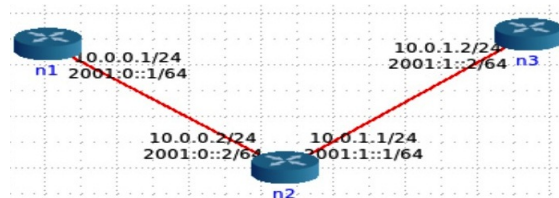


Fig. 11. Topologia CORE.

Questão 19: Com auxílio do *ifconfig* obtenha os endereços Ethernet das interfaces dos diversos routers.

Pela análise da imagem apresentada em baixo, podemos ver que o endereço Ethernet do router n1 é 00:00:00:aa:00:00.

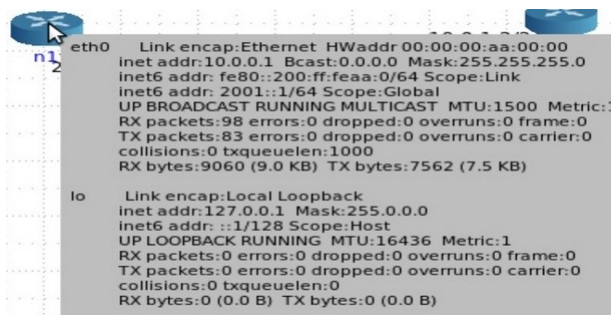
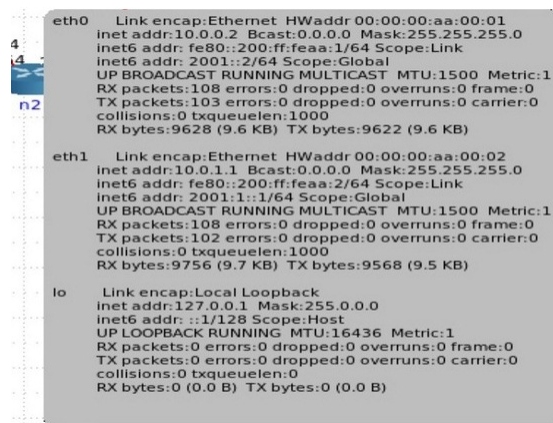


Fig. 12. *ifconfig* Router n1.

O endereço Ethernet do router n2 é 00:00:00:aa:00:02.



```

eth0  Link encap:Ethernet  HWaddr 00:00:00:aa:00:01
      inet addr:10.0.0.2  Bcast:0.0.0.0  Mask:255.255.255.0
      inet6 addr: fe80::200:ff:feaa:1/64 Scope:Link
      inet6 addr: 2001::2/64 Scope:Global
      UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
      RX packets:108 errors:0 dropped:0 overruns:0 frame:0
      TX packets:103 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:1000
      RX bytes:9628 (9.6 KB)  TX bytes:9622 (9.6 KB)

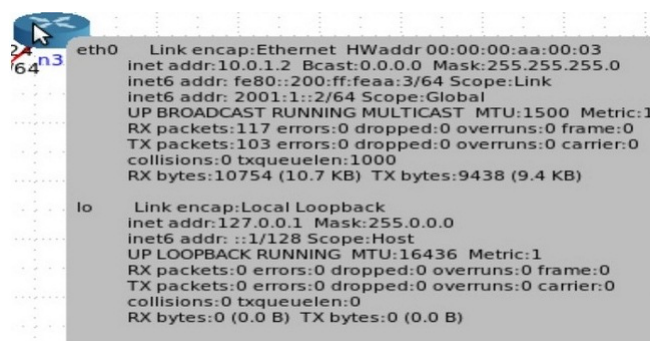
eth1  Link encap:Ethernet  HWaddr 00:00:00:aa:00:02
      inet addr:10.0.1.1  Bcast:0.0.0.0  Mask:255.255.255.0
      inet6 addr: fe80::200:ff:feaa:2/64 Scope:Link
      inet6 addr: 2001::1/64 Scope:Global
      UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
      RX packets:108 errors:0 dropped:0 overruns:0 frame:0
      TX packets:102 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:1000
      RX bytes:9756 (9.7 KB)  TX bytes:9568 (9.5 KB)

lo    Link encap:Local Loopback
      inet addr:127.0.0.1  Mask:255.0.0.0
      inet6 addr: ::1/128 Scope:Host
      UP LOOPBACK RUNNING  MTU:16436  Metric:1
      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:0
      RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

```

Fig. 13. *ifconfig* Router n2.

E o endereço Ethernet do router n3 é 00:00:00:aa:00:03.



```

eth0  Link encap:Ethernet  HWaddr 00:00:00:aa:00:03
      inet addr:10.0.1.2  Bcast:0.0.0.0  Mask:255.255.255.0
      inet6 addr: fe80::200:ff:feaa:3/64 Scope:Link
      inet6 addr: 2001::2/64 Scope:Global
      UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
      RX packets:117 errors:0 dropped:0 overruns:0 frame:0
      TX packets:103 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:1000
      RX bytes:10754 (10.7 KB)  TX bytes:9438 (9.4 KB)

lo    Link encap:Local Loopback
      inet addr:127.0.0.1  Mask:255.0.0.0
      inet6 addr: ::1/128 Scope:Host
      UP LOOPBACK RUNNING  MTU:16436  Metric:1
      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:0
      RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

```

Fig. 14. *ifconfig* Router n3.

Questão 20: Usando o comando arp obtenha o conteúdo das caches arp dos diversos sistemas.

Através das três imagens de baixo, é possível ver o conteúdo das caches de n1, n2 e n3, respectivamente.

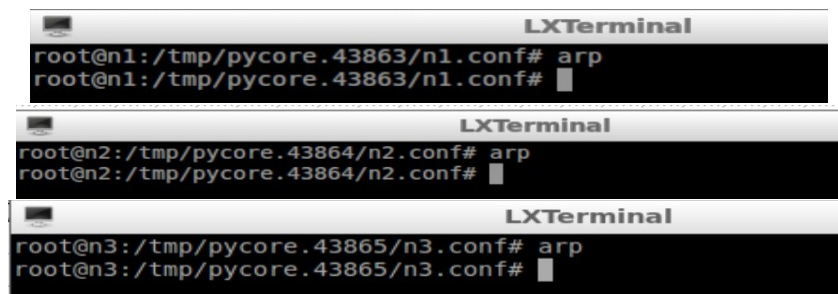


Fig. 15. Caches dos sistema.

Questão 21: Faça ping de n1 para n2. Que modificações observa nas caches ARP dos sistemas envolvidos.

As caches, com ping, passam a não estar vazias.

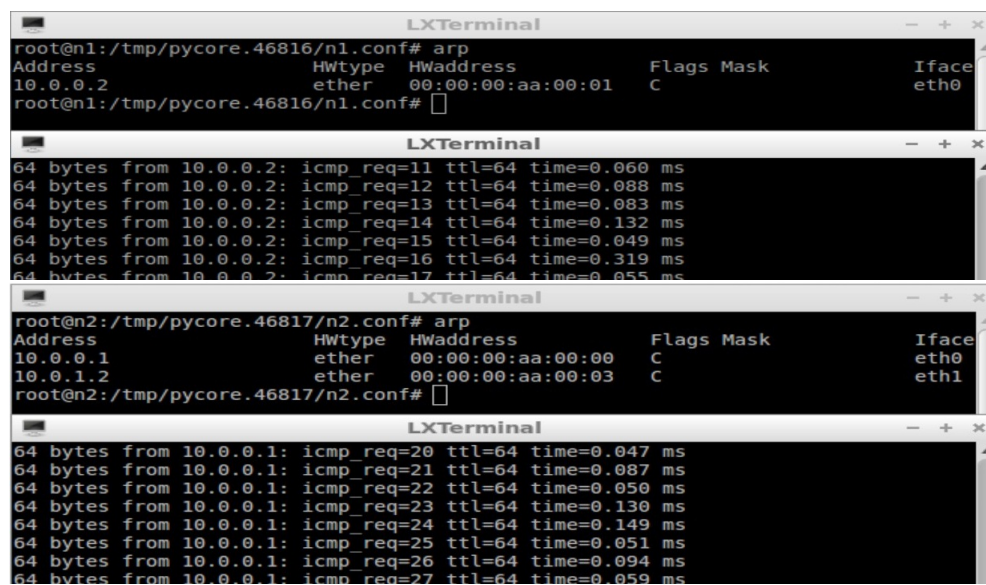


Fig. 16. Caches do sistema após ping.

Questão 22: Faça ping de n1 para n3. Consulte as caches ARP. Que conclui?

Na nossa topologia, não havia uma ligação directa de n1 para n3. Como tal, houve necessidade de haver ligação a um outro nodo (n2). Na cache de n1 podemos verificar o *address* destino da ligação de n1 para n2. Já na cache de n3, podes ver o *address* de origem do ligação n2 para n3.

The figure consists of three stacked screenshots of LXTerminal windows. The top window shows the ARP table for node n1, with an entry for 10.0.0.2 (ether 00:00:00:aa:00:01) on interface eth0. The middle window shows the ping results for node n1 to 10.0.1.2, with five successful pings. The bottom window shows the ARP table for node n3, with an entry for 10.0.1.1 (ether 00:00:00:aa:00:02) on interface eth0. The bottom window also shows the ping results for node n1 to 10.0.1.2, with five successful pings.

```
root@n1:/tmp/pycore.46822/n1.conf# arp
Address      HWtype  HWaddress      Flags Mask    Iface
10.0.0.2     ether   00:00:00:aa:00:01 C              eth0
root@n1:/tmp/pycore.46822/n1.conf#

root@n1:/tmp/pycore.46822/n1.conf# ping 10.0.1.2
PING 10.0.1.2 (10.0.1.2) 56(84) bytes of data:
64 bytes from 10.0.1.2: icmp_req=1 ttl=63 time=0.082 ms
64 bytes from 10.0.1.2: icmp_req=2 ttl=63 time=0.089 ms
64 bytes from 10.0.1.2: icmp_req=3 ttl=63 time=0.103 ms
64 bytes from 10.0.1.2: icmp_req=4 ttl=63 time=0.376 ms
64 bytes from 10.0.1.2: icmp_req=5 ttl=63 time=0.154 ms

root@n3:/tmp/pycore.46822/n3.conf# arp
Address      HWtype  HWaddress      Flags Mask    Iface
10.0.1.1     ether   00:00:00:aa:00:02 C              eth0
root@n3:/tmp/pycore.46822/n3.conf#

root@n1:/tmp/pycore.46822/n1.conf# ping 10.0.1.2
PING 10.0.1.2 (10.0.1.2) 56(84) bytes of data:
64 bytes from 10.0.1.2: icmp_req=1 ttl=63 time=0.082 ms
64 bytes from 10.0.1.2: icmp_req=2 ttl=63 time=0.089 ms
64 bytes from 10.0.1.2: icmp_req=3 ttl=63 time=0.103 ms
64 bytes from 10.0.1.2: icmp_req=4 ttl=63 time=0.376 ms
64 bytes from 10.0.1.2: icmp_req=5 ttl=63 time=0.154 ms
```

Fig. 17. Caches do sistema após ping.

Questão 23: Em n1 remova a entrada correspondente a n2. Coloque uma nova entrada para n2 com endereço Ethernet errado. O que acontece?

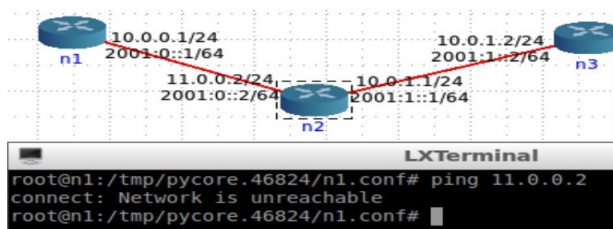


Fig. 18. Topologia com uma remoção.

Alteramos o address de 10.0.0.2 para 11.0.0.2 no nodo 2. Não é estabelecida a ligação.

Questão 24: 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.

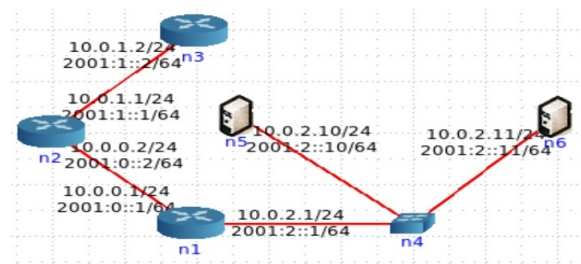


Fig. 19. Nova Topologia.

Será de esperar que na tabela ARP do n5 surja o endereço do n6.
Após verificar a cache do nodo n5, confirmarmos o nosso prognóstico.

```

LXTerminal
root@n5: /tmp/pycore.52725/n5.conf# arp
Address      HWtype  HWaddress  Flags Mask  Iface
10.0.2.11    ether    00:00:00:aa:00:06  C          eth0
root@n5: /tmp/pycore.52725/n5.conf# 

LXTerminal
root@n5: /tmp/pycore.52725/n5.conf# ping 10.0.2.11
PING 10.0.2.11 (10.0.2.11) 56(84) bytes of data:
64 bytes from 10.0.2.11: icmp_req=1 ttl=64 time=0.167 ms
64 bytes from 10.0.2.11: icmp_req=2 ttl=64 time=0.087 ms
64 bytes from 10.0.2.11: icmp_req=3 ttl=64 time=0.073 ms
64 bytes from 10.0.2.11: icmp_req=4 ttl=64 time=0.159 ms
64 bytes from 10.0.2.11: icmp_req=5 ttl=64 time=0.162 ms

```

Fig. 20. Cache dos sistemas.

Parte II

1.4 ARP Gratuito

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

Foram enviados 4 pacotes gratuitos, intercalados com os restantes pacotes.

Time	Source	Destination	Type	Length	Protocol	Info
35 6.041980000	Apple_b7:65:b8	Broadcast	ARP	64	Gratuitous ARP for 192.168.100.226 (Request)	[ETHERNET II]
36 6.348777000	AsustekC_db:5d:54	Broadcast	ARP	42	Who has 192.168.100.254? Tell 192.168.100.159	
37 6.349668000	Apple_b7:65:b8	Broadcast	ARP	64	Gratuitous ARP for 192.168.100.226 (Request)	[ETHERNET II]
38 6.451181000	Apple_b7:65:b8	Broadcast	ARP	64	Who has 169.254.255.255? Tell 192.168.100.226 [ETHERNET II]	

Fig. 21. Pacote Gratuito.

O primeiro pacote chegou às 16:36:36.93:

```
Arrival Time: Nov  5, 2013 16:36:36.936505000 WET
[Time shift for this packet: 0.000000000 seconds]
Epoch Time: 1383669396.936505000 seconds
[Time delta from previous captured frame: 0.502842000 seconds]
[Time delta from previous displayed frame: 0.502842000 seconds]
```

o segundo às 16:36:37.24:

```
Arrival Time: Nov  5, 2013 16:36:37.244193000 WET
[Time shift for this packet: 0.000000000 seconds]
Epoch Time: 1383669397.244193000 seconds
[Time delta from previous captured frame: 0.000891000 seconds]
[Time delta from previous displayed frame: 0.000891000 seconds]
```

o terceiro às 16:36:59.79:

```
Frame 97: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
Interface id: 0
Encapsulation type: Ethernet (1)
Arrival Time: Nov  5, 2013 16:36:59.795372000 WET
[Time shift for this packet: 0.000000000 seconds]
```

e o quarto às 16:37:01.27:

▼ Frame 129: 42 bytes on wire (336 bits), 42 bytes captured (336 b:
 Interface id: 0
 Encapsulation type: Ethernet (1)
 Arrival Time: Nov 5, 2013 16:37:01.271956000 WET
 [Time shift for this packet: 0.000000000 seconds]

Questão 2: Analise o conteúdo de um pedido ARP gratuito e identifique 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 é usado para determinar o endereço MAC do router que está na mesma rede enquanto que o ARP gratuito verifica se há outro host na rede com o mesmo endereço IP que o originador.

Primeiro pacote gratuito:

35	6.041980000	Apple_b7:65:b8	Broadcast	ARP	64 Gratuitous ARP for 192.168.100.226 (Request) [ETHERNET II]
36	6.348777000	AsustekC_db:5d:54	Broadcast	ARP	42 Who has 192.168.100.254? Tell 192.168.100.159
37	6.349668000	Apple_b7:65:b8	Broadcast	ARP	64 Gratuitous ARP for 192.168.100.226 (Request) [ETHERNET II]
38	6.451181000	Apple_b7:65:b8	Broadcast	ARP	64 Who has 169.254.255.255? Tell 192.168.100.226 [ETHERNET II]
43	9.726645000	Apple_18:04:d8	HitronTe_ab:d5:25	ARP	42 Who has 192.168.1.1? Tell 192.168.1.5

.....

Frame 35: 64 bytes on wire (512 bits), 64 bytes captured (512 bits) on interface 0

Ethernet II, Src: Apple_b7:65:b8 (a8:88:08:b7:65:b8), Dst: Broadcast (ff:ff:ff:ff:ff:ff)

Address Resolution Protocol (request/gratuitous ARP)

Fig. 22.

Segundo pacote:

35	6.041980000	Apple_b7:65:b8	Broadcast	ARP	64 Gratuitous ARP for 192.168.100.226 (Request) [ETHERNET II]
36	6.348777000	AsustekC_db:5d:54	Broadcast	ARP	42 Who has 192.168.100.254? Tell 192.168.100.159
37	6.349668000	Apple_b7:65:b8	Broadcast	ARP	64 Gratuitous ARP for 192.168.100.226 (Request) [ETHERNET II]
38	6.451181000	Apple_b7:65:b8	Broadcast	ARP	64 Who has 169.254.255.255? Tell 192.168.100.226 [ETHERNET II]
43	9.726645000	Apple_18:04:d8	HitronTe_ab:d5:25	ARP	42 Who has 192.168.1.1? Tell 192.168.1.5

.....

Frame 35: 64 bytes on wire (512 bits), 64 bytes captured (512 bits) on interface 0

Ethernet II, Src: Apple_b7:65:b8 (a8:88:08:b7:65:b8), Dst: Broadcast (ff:ff:ff:ff:ff:ff)

Address Resolution Protocol (request/gratuitous ARP)

Fig. 23.

Terceiro pacote:

97	28.900847000	Apple_18:04:d8	Broadcast	ARP	42 Gratuitous ARP for 192.168.100.166 (Request) [ETHERNET II]
98	28.905209000	Apple_18:04:d8	Broadcast	ARP	42 Who has 169.254.255.255? Tell 192.168.100.166
100	28.939858000	Apple_18:04:d8	Broadcast	ARP	42 Who has 192.168.100.254? Tell 192.168.100.166

.....

Frame 97: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0

Ethernet II, Src: Apple_18:04:d8 (e0:f8:47:18:04:d8), Dst: Broadcast (ff:ff:ff:ff:ff:ff)

Address Resolution Protocol (request/gratuitous ARP)

Fig. 24.

Quarto pacote:

124	30.087139000	Apple_18:04:d8	Broadcast	ARP	42 Who has 169.254.255.255? Tell 192.168.100.166
129	30.377431000	Apple_18:04:d8	Broadcast	ARP	42 Gratuitous ARP for 192.168.100.166 (Request)
132	30.407368000	Apple_18:04:d8	Broadcast	ARP	42 Who has 169.254.255.255? Tell 192.168.100.166
135	30.515640000	Intel_1a:15:dd	Broadcast	ARP	60 Who has 192.168.100.150? Tell 192.168.100.254
147	30.697901000	Apple_18:04:d8	Broadcast	ARP	42 Who has 192.168.100.254? Tell 192.168.100.166

.....

Frame 129: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
Ethernet II, Src: Apple_18:04:d8 (e0:f8:47:18:04:d8), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Address Resolution Protocol (request/gratuitous ARP)

Fig. 25.

1.5 Domínios de colisão

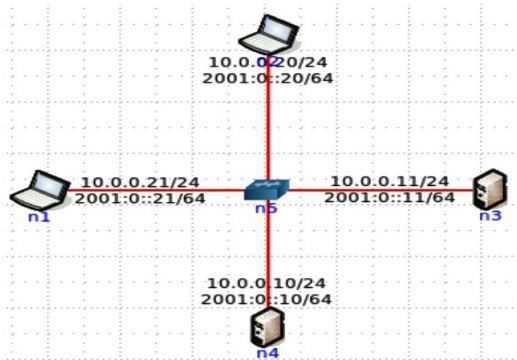
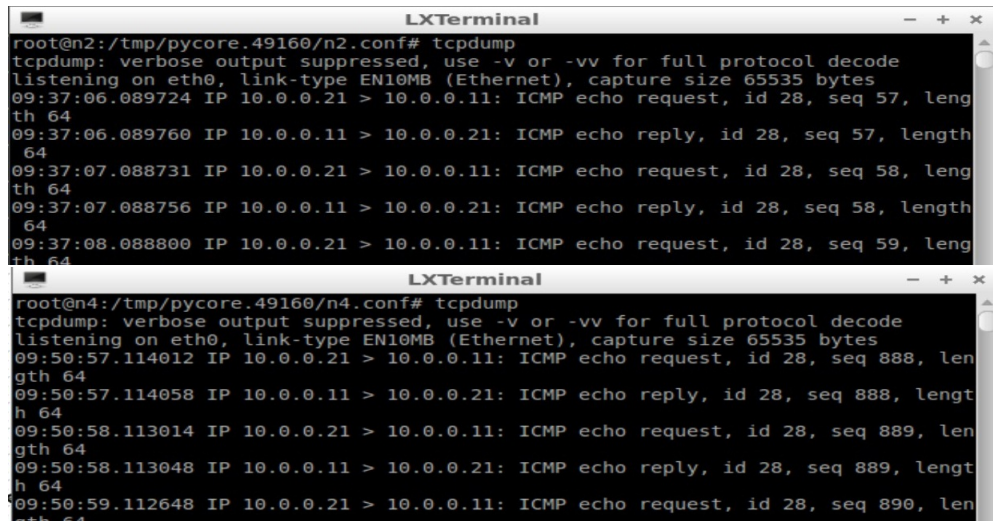


Fig. 26. Topologia CORE.

Questão 1: Faça ping de n1 para n3 e de n2 para n4. Verifique com a opção tcpdump como flui o tráfego nas diversas interfaces dos vários dispositivos.

Efetando o ping de n1 para n3, o tráfego em n2 e n4 é o seguinte:



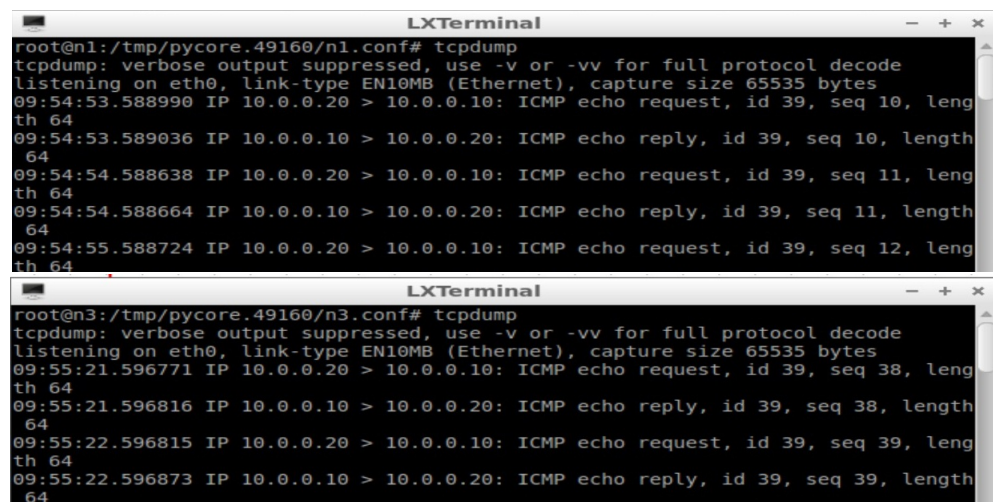
The image shows two terminal windows, LXTerminal, displaying tcpdump output. The top window is for n2, showing ICMP echo requests and replies between 10.0.0.21 and 10.0.0.11. The bottom window is for n4, showing ICMP echo requests and replies between 10.0.0.21 and 10.0.0.11.

```
root@n2:/tmp/pycore.49160/n2.conf# tcpdump
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
09:37:06.089724 IP 10.0.0.21 > 10.0.0.11: ICMP echo request, id 28, seq 57, length 64
09:37:06.089760 IP 10.0.0.11 > 10.0.0.21: ICMP echo reply, id 28, seq 57, length 64
09:37:07.088731 IP 10.0.0.21 > 10.0.0.11: ICMP echo request, id 28, seq 58, length 64
09:37:07.088756 IP 10.0.0.11 > 10.0.0.21: ICMP echo reply, id 28, seq 58, length 64
09:37:08.088800 IP 10.0.0.21 > 10.0.0.11: ICMP echo request, id 28, seq 59, length 64

root@n4:/tmp/pycore.49160/n4.conf# tcpdump
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
09:50:57.114012 IP 10.0.0.21 > 10.0.0.11: ICMP echo request, id 28, seq 888, length 64
09:50:57.114058 IP 10.0.0.11 > 10.0.0.21: ICMP echo reply, id 28, seq 888, length 64
09:50:58.113014 IP 10.0.0.21 > 10.0.0.11: ICMP echo request, id 28, seq 889, length 64
09:50:58.113048 IP 10.0.0.11 > 10.0.0.21: ICMP echo reply, id 28, seq 889, length 64
09:50:59.112648 IP 10.0.0.21 > 10.0.0.11: ICMP echo request, id 28, seq 890, length 64
```

Fig. 27. Tráfego de n2 e n4.

Efetando o ping de n2 para n4, o tráfego em n1 e n3 é o seguinte:



The image shows two terminal windows, LXTerminal, displaying tcpdump output. The top window is for n1, showing ICMP echo requests and replies between 10.0.0.20 and 10.0.0.10. The bottom window is for n3, showing ICMP echo requests and replies between 10.0.0.20 and 10.0.0.10.

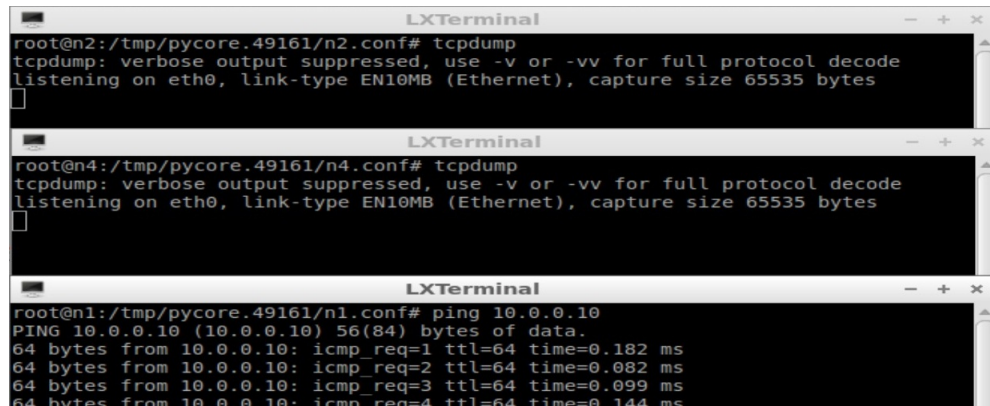
```
root@n1:/tmp/pycore.49160/n1.conf# tcpdump
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
09:54:53.588990 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 39, seq 10, length 64
09:54:53.589036 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 39, seq 10, length 64
09:54:54.588638 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 39, seq 11, length 64
09:54:54.588664 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 39, seq 11, length 64
09:54:55.588724 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 39, seq 12, length 64

root@n3:/tmp/pycore.49160/n3.conf# tcpdump
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
09:55:21.596771 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 39, seq 38, length 64
09:55:21.596816 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 39, seq 38, length 64
09:55:22.596815 IP 10.0.0.20 > 10.0.0.10: ICMP echo request, id 39, seq 39, length 64
09:55:22.596873 IP 10.0.0.10 > 10.0.0.20: ICMP echo reply, id 39, seq 39, length 64
```

Fig. 28. Tráfego de n1 e n3.

Questão 2: Na topologia de rede substitua o hub por um switch. Faça os mesmos procedimentos que realizou na pergunta anterior.

Efetuada o ping de n1 para n3, o tráfego em n2 e n4 é o seguinte:



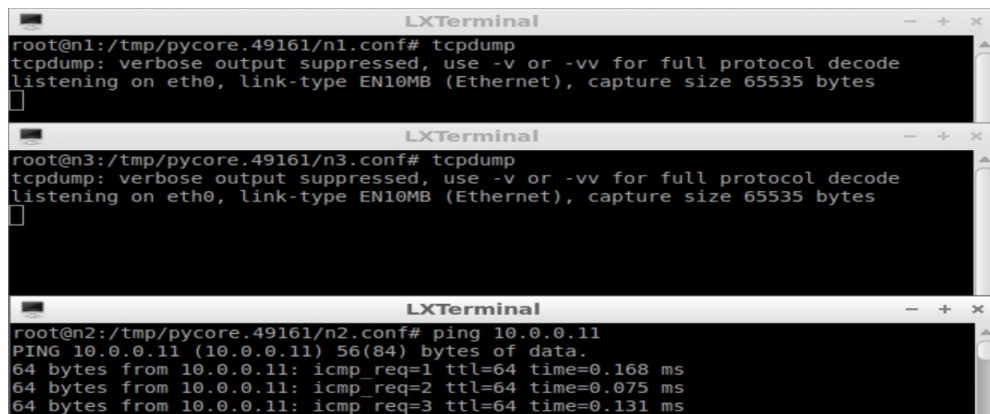
```
root@n2:/tmp/pycore.49161/n2.conf# tcpdump
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes

root@n4:/tmp/pycore.49161/n4.conf# tcpdump
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes

root@n1:/tmp/pycore.49161/n1.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=0.182 ms
64 bytes from 10.0.0.10: icmp_req=2 ttl=64 time=0.082 ms
64 bytes from 10.0.0.10: icmp_req=3 ttl=64 time=0.099 ms
64 bytes from 10.0.0.10: icmp_req=4 ttl=64 time=0.144 ms
```

Fig. 29. Tráfego de n2 e n4.

Efetuada o ping de n2 para n4, o tráfego em n1 e n3 é o seguinte:



```
root@n1:/tmp/pycore.49161/n1.conf# tcpdump
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes

root@n3:/tmp/pycore.49161/n3.conf# tcpdump
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes

root@n2:/tmp/pycore.49161/n2.conf# ping 10.0.0.11
PING 10.0.0.11 (10.0.0.11) 56(84) bytes of data.
64 bytes from 10.0.0.11: icmp_req=1 ttl=64 time=0.168 ms
64 bytes from 10.0.0.11: icmp_req=2 ttl=64 time=0.075 ms
64 bytes from 10.0.0.11: icmp_req=3 ttl=64 time=0.131 ms
```

Fig. 30. Tráfego de n1 e n3.

Questão 3: Comente os resultados obtidos e discuta cenários de 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.

Com a utilização de hubs temos a informação a ser reenviada para todos os dispositivos. Podemos ver através das imagens apresentadas na pergunta 1 essa mesma informação a ser passada.

Enquanto que, utilizando um switch, o tráfego apenas é enviado para o endereço de destino de cada pacote. Evitando assim, informação desnecessária e colisões de pacotes.

Pode-se concluir que o dispositivo Hub deve ser usado numa rede com poucos utilizadores. Já o switch, sendo mais sofisticado, deve ser usado para redes de maior proporção.

2 Conclusões

Neste trabalho prático: **Camada de Ligação Lógica: Ethernet e Protocolo ARP**, foram vários os conceitos novos.

Na primeira parte no exercício 3, sobre a **captura e análise de Tramas Ethernet** tivemos um primeiro contacto com WireShark. A principal dificuldade foi descobrir a mensagem que continha o *HTTPGET* e o *HTTPResponse*. Depois disto, o processo ficou mais simplificando.

Sendo, essencialmente, de observar a trama Ethernet e descobrir os endereços de origem e destino dos dispositivos.

Já no exercício 4, sobre o **Protocolo ARP**, a maior dificuldade foi, numa fase inicial, a captura de resultados porque apenas apresentava os *ARP* request e não as respostas(reply). Depois de limpar novamente a cache com o comando `arp -d *` e fazer a captura, obteve-se melhores resultados. Depois ficou-se a saber que o endereço de destino no request é o mesmo para todas as máquinas em broadcast.

No exercício 5, conseguimos uma maior familiarização com o *CORE*. Criamos uma série de topologias e realizamos um conjunto de comandos (`ifconfig`, `arp`) que nos permitiu ver vários resultados.

Quanto à parte 2, no exercício 2, basicamente soubemos da diferença entre um ARP normal e um ARP gratuito e quantos pacotes **ARP gratuitos** foram recebidos. No exercício 3, tivemos novamente contacto com o emulador *CORE*. Foi importante para verificar em termos práticos a diferença entre a utilização de um dispositivo Hub e um dispositivo Switch.

Em suma, consideramos o trabalho prático proveitoso.