Universidade do Minho

Licenciatura em Engenharia Informática

Redes de Computadores

Grupo 52

TP3 : Redes Ethernet e Protocolo ARP

António Luís Braga Mendes (A84675)

Maria Eugénia Bessa Cunha (A93264)

Vicente Gonçalves Moreira (A93296)

Maio 2022

Questões e Respostas

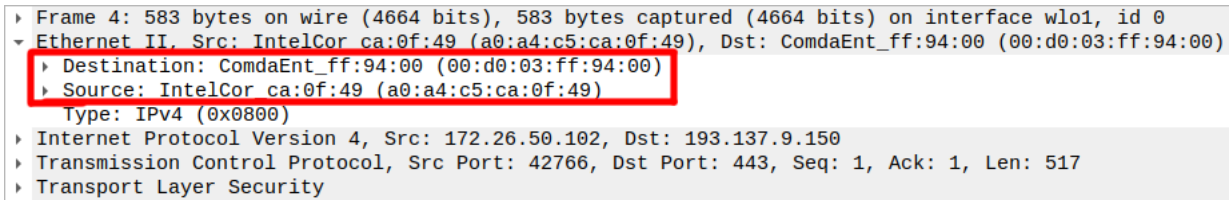
3 Captura e análise de Tramas Ethernet

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

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

Analisando a trama respetiva ao primeiro pacote *"Application Data"* enviado pela máquina, podemos descobrir o endereço MAC de origem **(a0:a4:c5:ca:0f:49)** e o endereço MAC de destino **(00:d0:03:ff:94:00).**

Figura 1: Endereços MAC origem/destino



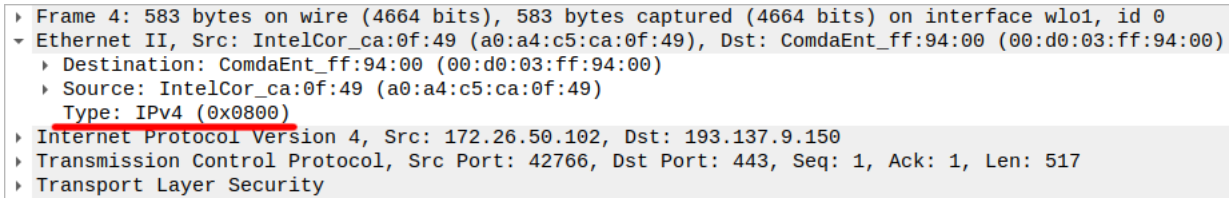
3.2 Identifique a que sistemas se referem. Justifique.

O endereço MAC de origem refere-se à máquina utilizada para aceder ao *website*, enquanto que o endereço de destino refere-se ao equipamento de nível de rede presente no primeiro "salto"na rede, ou seja, o primeiro *router* encontrado na rede, pertencente à Universidade do Minho.

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

O valor encontrado neste campo tem o valor de **0x0800**. Este valor é utilizado para identificar o protocolo utilizado pela camada protocolar superior, ou seja, o "formato"da *payload* presente na trama.

Figura 2: Valor "type"MAC



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

Para calcular os *bytes* utilizados no encapsulamento protocolar, avaliamos o tamanho de cada *header* presente no pacote. Apesar da falta de informação sobre o tamanho do cabeçalho do nível de ligação lógica neste, conseguimos entender que este tem um tamanho fixo de **14 *bytes***, (6 *bytes* MAC Org. + 6 *bytes* MAC Dest. + 2 *bytes* Type), sendo esta a razão pela qual este não necessita de ser enviado. Para os restantes cabeçalhos quer do nível de Rede ou nível de Transporte contêm informação acerca do tamanho do cabeçalho, que utilizamos para calcular o total de *bytes* transmitidos pelas camadas protocolares.

Figura 3: Tamanho cabeçalho camada de Rede

```
▸ Frame 14: 913 bytes on wire (7304 bits), 913 bytes captured (7304 bits) on interface wlo1, id 0
▸ Ethernet II, Src: IntelCor_ca:0f:49 (a0:a4:c5:ca:0f:49), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
▾ Internet Protocol Version 4, Src: 172.26.50.102, Dst: 193.137.9.150
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 899
```

Figura 4: Tamanho cabeçalho camada de Transporte

```
▸ Frame 14: 913 bytes on wire (7304 bits), 913 bytes captured (7304 bits) on interface wlo1, id 0
▸ Ethernet II, Src: IntelCor_ca:0f:49 (a0:a4:c5:ca:0f:49), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
▸ Internet Protocol Version 4, Src: 172.26.50.102, Dst: 193.137.9.150
▾ Transmission Control Protocol, Src Port: 42766, Dst Port: 443, Seq: 569, Ack: 138, Len: 847
    Source Port: 42766
    Destination Port: 443
    [Stream index: 0]
    [TCP Segment Len: 847]
    Sequence number: 569    (relative sequence number)
    Sequence number (raw): 3372685835
    [Next sequence number: 1416    (relative sequence number)]
    Acknowledgment number: 138    (relative ack number)
    Acknowledgment number (raw): 4131657560
    1000 .... = Header Length: 32 bytes (8)
  ▸ Flags: 0x018 (PSH, ACK)
```

Somamos assim o tamanho dos vários cabeçalhos:

- Camada de Ligação Lógica = 14 bytes
- Camada de Rede = 20 bytes
- Camada de Transporte = 32 bytes
- **Total:** 66 bytes

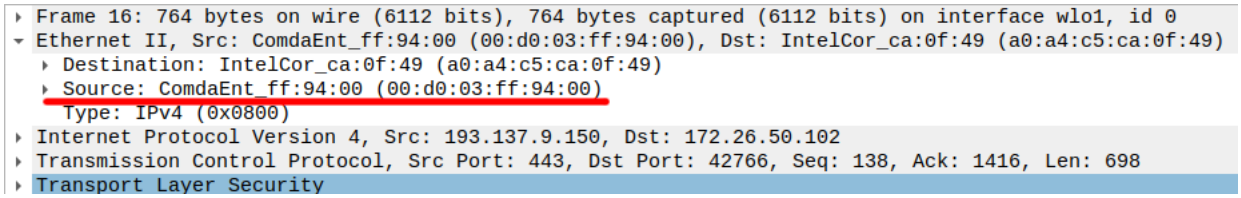
Visto que o pacote tem um tamanho total de **913 *bytes*** (Visto na linha da "Frame"e verificado com a soma da *payload* TCP(847 *bytes*) + total *Headers* (66 *bytes*)), obtemos uma percentagem de *overhead* de **7.23%**, ou seja, cerca de 7.23% dos *bytes* totais transmitidos "pela rede"foram utilizados para realizar o envio da informação "real".

• **A seguir responda às seguintes perguntas, baseado no conteúdo da trama Ethernet que contém o primeiro byte da resposta HTTP proveniente do servidor.**

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

O endereço MAC/*Ethernet* da fonte é **00:d0:03:ff:94:00** e corresponde ao último *router* encontrado na rede (tendo em conta que o pacote viajou do servidor à máquina), pertencente à Universidade do Minho.

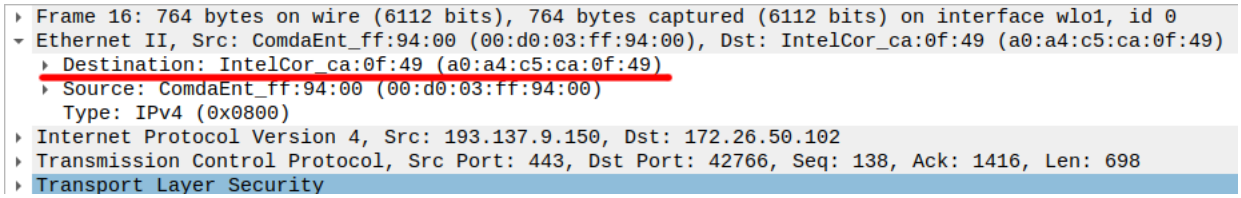
Figura 5: Endereço MAC Fonte



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

O endereço MAC/*Ethernet* do destino é **a0:a4:c5:ca:0f:49** e corresponde à máquina utilizada para efetuar a ligação ao *website*.

Figura 6: Endereço MAC Destino



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

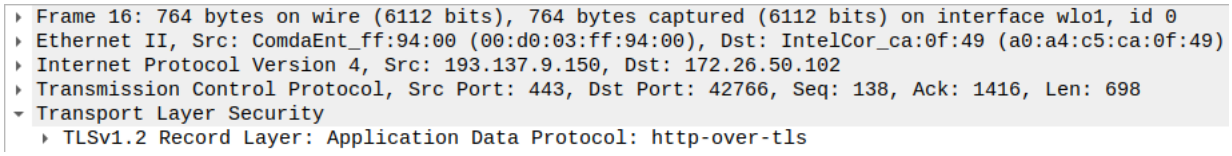
Segundo o *OSI model* mais básico, existem 5 camadas protocolares. Estas dividem de forma clara quais as responsabilidades e funções que cada camada necessita de "prestar"para o bom funcionamento da *internet*. Uma vantagem deste modelo é a grande modularidade da *internet*, pois várias camadas podem ser "trocadas"sem haver efeitos negativos nas restantes camadas, como por exemplo, a utilização dos protocolos de transporte TCP ou UDP, ou uma máquina ser capaz de se conectar à rede por fios (via *Ethernet*) ou por um sistema sem fios (*WiFi*) sem haver compromissos no acesso a esta.

No entanto, uma desvantagem presente neste modelo é a necessidade de cada camada necessitar de enviar informação relativa à sua camada/protocolo para realizar a sua função, havendo maiores *overheads* do que num modelo integrado.

Neste exemplo, podemos ver as 4 camadas protocolares presentes na trama enviada (apenas 4 pois a camada "Física"não é representável), sendo estes:

- Camada de Ligação Lógica = **Ethernet**
- Camada de Rede = **IPv4** (Internet Protocol Version 4)
- Camada de Transporte = **TCP** (Transmission Control Protocol)
- Camada Aplicacional = **TLSv1.2** (Transport Layer Security)

Figura 7: Camadas Protocolares

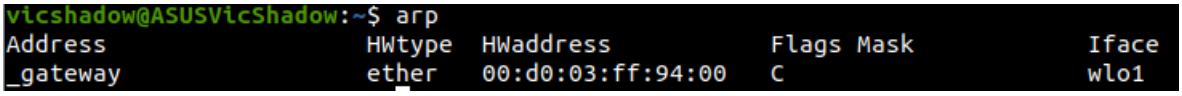


4 Protocolo ARP

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

Para este exercício executamos o comando "arp", obtendo o seguinte resultado:

Figura 8: Tabela ARP



Depois de analisada a tabela em questão, indicamos o significado de cada coluna presente:

Coluna	Descrição
Address	Endereço IP de destino
HWtype	Tipo de interface de saída
HWaddress	Endereço MAC do Hardware conectado à saída
Flags	Flags de informação acerca da entrada na tabela
Mask	Máscara do Endereço IP
Iface	Nome da interface de saída na máquina

Tabela 1: Colunas Tabela ARP

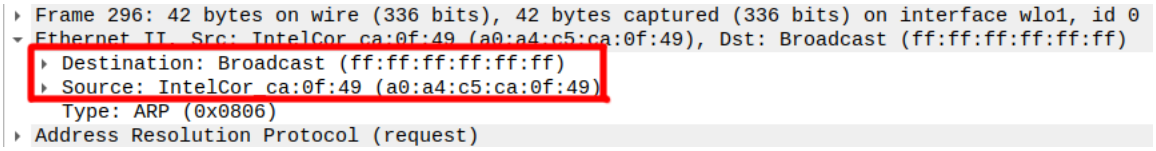
Também decidimos investigar o significado das várias *Flags* possíveis, obtendo a seguinte informação:

- **Flag C** = Entrada definida dinamicamente pelo protocolo ARP
- **Flag M** = Entrada inserida manualmente
- **Flag P** = Entrada de "Publicação". Indica que esta entrada serve como resposta a pacotes do tipo *ARP request* e *ARP response*.

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

Avaliando a trama *Ehternet* contendo o pedido ARP, o valor hexadecimal do endereço de origem é **0xa0a4c5ca0f49** (a0:a4:c5:ca:0f:49) que corresponde ao endereço MAC da máquina utilizada. Já o endereço de destino é **0xffffffffff** (ff:ff:ff:ff:ff:ff) sendo que este representa um "endereço vazio", identificador de um pacote ARP do tipo *Request*. Este endereço é utilizado visto que a máquina não conhece o endereço MAC das várias máquinas vizinhas ligadas às suas interfaces, sendo este endereço genérico utilizado para efetuar *Broadcasts*.

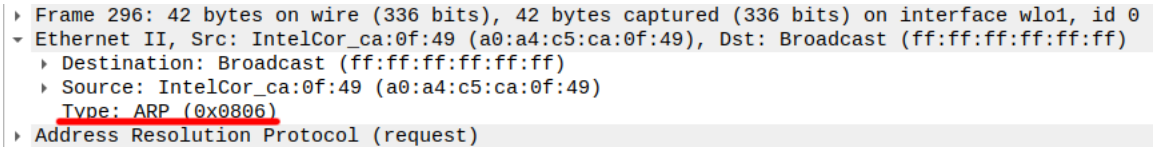
Figura 9: Trama Ethernet - ARP Request



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

O valor hexadecimal presente no campo "Type"da trama *Ethernet* é **0x0806**. Este valor indica o protocolo utilizado pela mensagem encapsulada na trama, para que esta seja lida de forma correta. Apesar do protocolo ARP pertencer à camada de ligação lógica, este é encapsulado na trama, visto que pertence a uma "subcamada"da camada de ligação lógica, a subcamada MAC.

Figura 10: Trama Ethernet - Type



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

Expandindo a informação inserida na mensagem ARP encapsulada na trama, podemos ver que existe informação relativa ao tipo de conexão utilizado, o *Opcode* da mensagem e os endereços IP/MAC da máquina que efetuou o pedido (Sender) e os endereços IP/MAC da máquina "alvo"(Target).

Podemos assim confirmar que esta trama se trata efetivamente de um pedido ARP visto que, não só contém o endereço MAC de destino de *Broadcast* como também contém o valor do *Opcode* da mensagem ARP com o valor 1, indicando que se trata de um *APR Request*.

Dentro da mensagem ARP, estão contidos vários endereços MAC e IP, sendo que contêm os endereços MAC e IP da máquina que efetuou o *ARP Request* (Sender), de forma a que a resposta a este pedido possa ser "devolvida"e, por último, o endereço IP da máquina alvo (**172.26.254.254**), ao qual se pretende descobrir o endereço MAC. Por esta razão o campo "Target MAC address"encontra-se vazio (00:00:00:00:00:00), pois será preenchido pela máquina alvo identificada pelo endereço IP.

Figura 11: Conteúdo Mensagem ARP

```
▸ Frame 296: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface wlo1, id 0
▾ Ethernet II, Src: IntelCor_ca:0f:49 (a0:a4:c5:ca:0f:49), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
  ▸ Destination: Broadcast (ff:ff:ff:ff:ff:ff)
  ▸ Source: IntelCor_ca:0f:49 (a0:a4:c5:ca:0f:49)
    Type: ARP (0x0806)
▾ Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: IntelCor_ca:0f:49 (a0:a4:c5:ca:0f:49)
    Sender IP address: 172.26.50.102
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 172.26.254.254
```

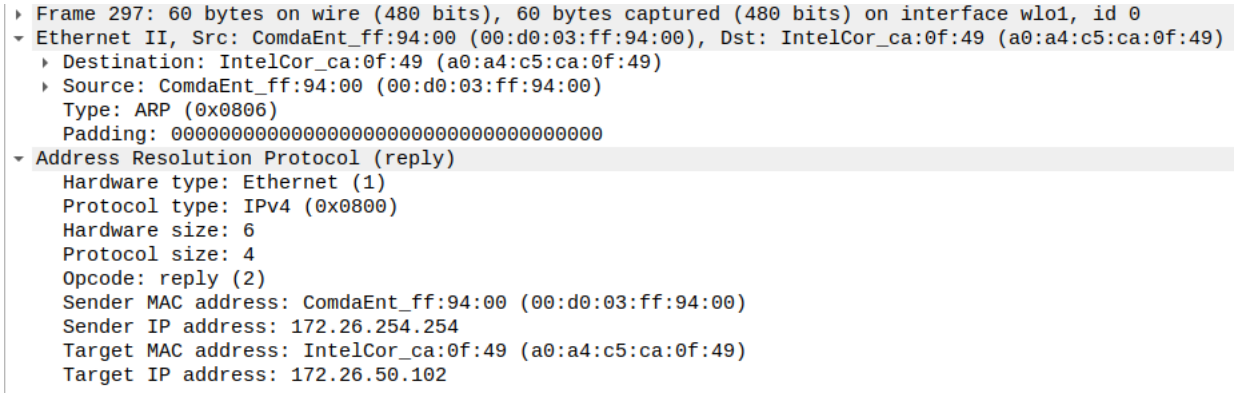
4.12 Explicite que tipo de pedido ou pergunta é feita pelo host de origem.

O *host* de origem, ao efetuar este *ARP Request*, pretende saber o endereço MAC da máquina cujo IP na rede é 172.26.254.254. Este pedido será transmitido pela rede até que o dono do IP em questão for encontrado e este efetuar um *ARP Reply* com a informação desejada do pedido. É de notar que, neste exemplo, o IP utilizado (172.26.254.254, máscara 255.255.0.0) é normalmente utilizado na rede por *routers* responsáveis da sub-rede (Endereço de topo).

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

Localizada a mensagem, procedemos à sua análise. Observamos estes resultados:

Figura 12: Trama Ethernet - ARP Reply



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

O campo **ARP Opcode** tem um valor de 2, indicando que esta mensagem ARP se trata efetivamente de uma *ARP Reply*, respondendo ao pedido anterior.

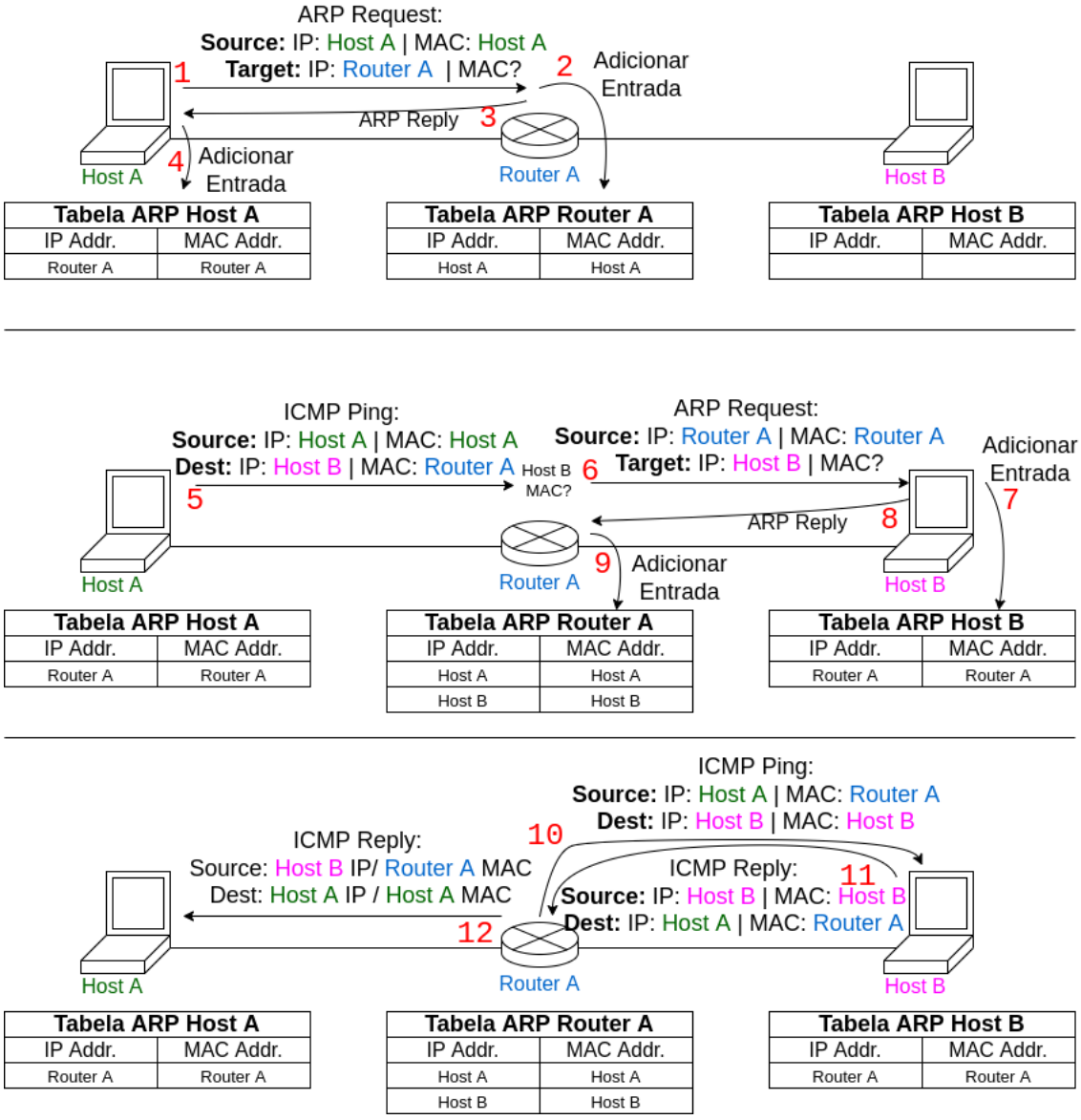
b. Em que campo da mensagem ARP está a resposta ao pedido ARP?

A resposta encontra-se no campo "Sender MAC Address". Como podemos ver, o valor deste foi alterado para um MAC address distinto, correspondente ao endereço IP pedido.

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

Para esta pergunta esboçamos um esquema contendo as várias fases e pedidos feitos, ordenados numericamente ("Adicionar Entrada"não conta como mensagem):

Figura 13: Mensagens Trocadas



5 Domínios de Colisão

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

A principal diferença entre uma LAN partilhada (utilização de um *hub*) e uma LAN comutada (utilização de um *switch*) é a forma como a transmissão das tramas ocorre na subrede. Enquanto que numa LAN partilhada, todas as tramas enviadas ao *hub* são difundidas pela rede toda, visto que o *hub* não faz operações lógicas de comparação de endereços, numa LAN comutada, dirigida por um *switch*, as tramas enviadas a este são reencaminhadas apenas para os endereços de destino presente na trama.

Esta comutação traz vantagens pois não só evita o envio e receção de tramas que serão "desperdiçadas", ocupando assim a largura de rede (colisões), como também evita potenciais ataques, como um agente maligno receber e interpretar todas as tramas partilhadas na subrede.

Para a resolução deste exercício, e de forma a demonstrar a diferença das tramas partilhadas entre LAN's partilhadas e LAN's comutadas, fizemos as modificações pedidas na topologia e, de seguida, efetuamos o comando *tcpdump* num dispositivo "ouvinte"(neste caso, escolhemos o Servidor A e Servidor B), efetuando depois um *ping* entre ambos os *hosts* presentes no Departamento.

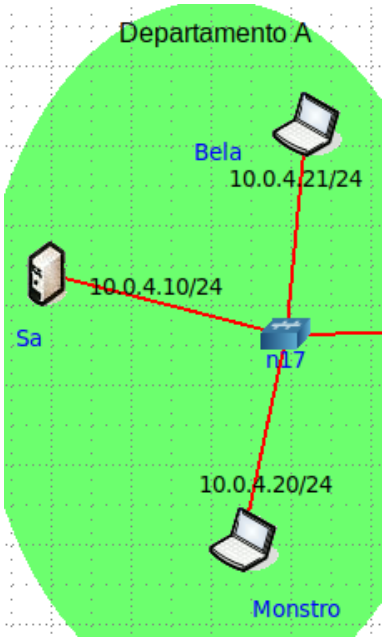


Figura 14: Departamento A

```
root@Monstro:/tmp/pycore.46735/Monstro.conf# ping 10.0.4.21
PING 10.0.4.21 (10.0.4.21) 56(84) bytes of data.
64 bytes from 10.0.4.21: icmp_seq=1 ttl=64 time=0.146 ms
64 bytes from 10.0.4.21: icmp_seq=2 ttl=64 time=0.131 ms
64 bytes from 10.0.4.21: icmp_seq=3 ttl=64 time=0.097 ms
64 bytes from 10.0.4.21: icmp_seq=4 ttl=64 time=0.096 ms
64 bytes from 10.0.4.21: icmp_seq=5 ttl=64 time=0.088 ms
^C
--- 10.0.4.21 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4068ms
rtt min/avg/max/mdev = 0.088/0.111/0.146/0.022 ms
root@Monstro:/tmp/pycore.46735/Monstro.conf#
```

Figura 15: Ping Monstro → *Bela*

Figura 16: tcpdump Servidor A

```
root@Sa:/tmp/pycore.46735/Sa.conf# tcpdump
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
^C16:07:55.302000 IP 10.0.4.1 > 224.0.0.5: OSPFv2, Hello, length 44
16:07:57.303244 IP 10.0.4.1 > 224.0.0.5: OSPFv2, Hello, length 44
16:07:58.998385 ARP, Request who-has 10.0.4.21 tell 10.0.4.20, length 28
16:07:58.998436 ARP, Reply 10.0.4.21 is-at 00:00:00:aa:00:18 (oui Ethernet), length 28
16:07:58.998450 IP 10.0.4.20 > 10.0.4.21: ICMP echo request, id 27, seq 1, length 64
16:07:58.998478 IP 10.0.4.21 > 10.0.4.20: ICMP echo reply, id 27, seq 1, length 64
16:07:59.238497 IP6 fe80::200:ff:feaa:19 > ff02::5: OSPFv3, Hello, length 36
16:07:59.303514 IP 10.0.4.1 > 224.0.0.5: OSPFv2, Hello, length 44
16:07:59.998763 IP 10.0.4.20 > 10.0.4.21: ICMP echo request, id 27, seq 2, length 64
16:07:59.998807 IP 10.0.4.21 > 10.0.4.20: ICMP echo reply, id 27, seq 2, length 64
16:08:01.018470 IP 10.0.4.20 > 10.0.4.21: ICMP echo request, id 27, seq 3, length 64
16:08:01.018513 IP 10.0.4.21 > 10.0.4.20: ICMP echo reply, id 27, seq 3, length 64
16:08:01.303810 IP 10.0.4.1 > 224.0.0.5: OSPFv2, Hello, length 44
16:08:02.042459 IP 10.0.4.20 > 10.0.4.21: ICMP echo request, id 27, seq 4, length 64
16:08:02.042503 IP 10.0.4.21 > 10.0.4.20: ICMP echo reply, id 27, seq 4, length 64
16:08:03.066422 IP 10.0.4.20 > 10.0.4.21: ICMP echo request, id 27, seq 5, length 64
16:08:03.066461 IP 10.0.4.21 > 10.0.4.20: ICMP echo reply, id 27, seq 5, length 64
16:08:03.305020 IP 10.0.4.1 > 224.0.0.5: OSPFv2, Hello, length 44
16:08:04.154354 ARP, Request who-has 10.0.4.20 tell 10.0.4.21, length 28
16:08:04.154380 ARP, Reply 10.0.4.20 is-at 00:00:00:aa:00:17 (oui Ethernet), length 28

20 packets captured
20 packets received by filter
0 packets dropped by kernel
root@Sa:/tmp/pycore.46735/Sa.conf#
```

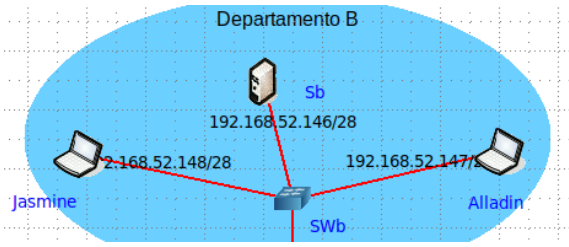


Figura 17: Departamento B

```
root@Jasmine:/tmp/pycore.46735/Jasmine.conf# ping 192.168.52.147
PING 192.168.52.147 (192.168.52.147) 56(84) bytes of data.
64 bytes from 192.168.52.147: icmp_seq=1 ttl=64 time=1.11 ms
64 bytes from 192.168.52.147: icmp_seq=2 ttl=64 time=0.266 ms
64 bytes from 192.168.52.147: icmp_seq=3 ttl=64 time=0.286 ms
64 bytes from 192.168.52.147: icmp_seq=4 ttl=64 time=0.310 ms
64 bytes from 192.168.52.147: icmp_seq=5 ttl=64 time=0.338 ms
64 bytes from 192.168.52.147: icmp_seq=6 ttl=64 time=0.277 ms
^C
--- 192.168.52.147 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5086ms
rtt min/avg/max/mdev = 0.266/0.431/1.112/0.305 ms
root@Jasmine:/tmp/pycore.46735/Jasmine.conf#
```

Figura 18: Ping Jasmine → *Alladin*

Figura 19: tcpdump Servidor B

```
root@Sb:/tmp/pycore.46735/Sb.conf# tcpdump
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
^C16:09:39.396627 IP6 fe80::200:ff:feaa:c > ff02::5: OSPFv3, Hello, length 36
16:09:39.461166 IP 192.168.52.145 > 224.0.0.5: OSPFv2, Hello, length 44
16:09:41.461590 IP 192.168.52.145 > 224.0.0.5: OSPFv2, Hello, length 44
16:09:43.462034 IP 192.168.52.145 > 224.0.0.5: OSPFv2, Hello, length 44
16:09:45.436964 ARP, Request who-has 192.168.52.147 tell 192.168.52.148, length 28
16:09:45.462833 IP 192.168.52.145 > 224.0.0.5: OSPFv2, Hello, length 44
16:09:47.463088 IP 192.168.52.145 > 224.0.0.5: OSPFv2, Hello, length 44
16:09:49.370764 IP6 fe80::200:ff:feaa:c > ff02::5: OSPFv3, Hello, length 36
16:09:49.463530 IP 192.168.52.145 > 224.0.0.5: OSPFv2, Hello, length 44
16:09:51.463558 IP 192.168.52.145 > 224.0.0.5: OSPFv2, Hello, length 44

10 packets captured
10 packets received by filter
0 packets dropped by kernel
root@Sb:/tmp/pycore.46735/Sb.conf#
```

Como podemos observar, no departamento A, gerido por um *hub* (LAN partilhada), apesar de o Servidor A não estar envolvido no *Ping Request* efetuado no Departamento A, acabamos por receber os pacotes relativos a este *Ping*. Já no Departamento B, gerido por um *switch* (LAN comutada), não foi possível observar qualquer pacote que não fosse dirigido ao servidor, através do endereço MAC correspondente.

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

Apresentamos de seguida um esquema representativo dos dispositivos presentes no departamento B, assim como o *Router* ligado ao *Switch* em questão. Para descobrir os vários endereços MAC dos dispositivos ligados à sub rede, efetuamos o comando **ifconfig**, anotando o valor de endereço MAC obtido. No caso do *router*, tivemos o cuidado de analisar a interface de saída e o seu IP correspondente. Com esta informação reunida, construimos a tabela de comutação deste *Switch*:

Figura 20: ifconfig Router A

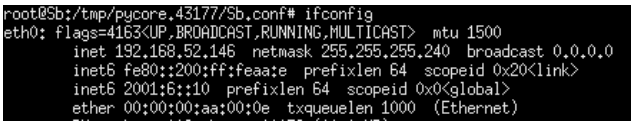
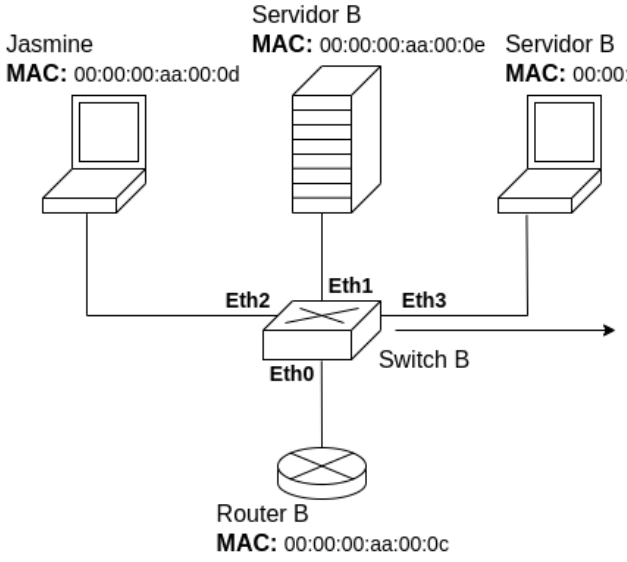


Figura 21: ifconfig Server A

Figura 22: ifconfig Jasmine

Figura 23: ifconfig Alladin A

Figura 24: Tabela Comutação Switch B



Endereço MAC	Interface	TTL
00:00:00:aa:00:0c	Eth0	60
00:00:00:aa:00:0e	Eth1	60
00:00:00:aa:00:0d	Eth2	60
00:00:00:aa:00:0f	Eth3	60

Conclusão

Com o terminar desta trabalho prático, encontramo-nos satisfeitos com o trabalho desenvolvido, tendo alcançado todas as metas propostas pelos docentes, assim como ter justificado adequadamente os resultados observados e analisados. No entanto, existem possíveis ajustes a serem feitos, como uma estruturação mais cuidada de algumas respostas, assim como a falta de algumas explicações mais breves e sucintas.

Este trabalho também nos permitiu aprofundar o nosso conhecimento acerca do funcionamento de um dos níveis mais básicos das redes, percebendo melhor os vários detalhes e processos envolvidos nesta camada. Pudemos analisar o conteúdo das tramas *Ethernet* e entender a sua importância, investigamos o funcionamento do protocolo ARP ao pormenor e entedemos as várias diferenças entre LAN's partilhadas e LAN's comutadas.