

# TP3 - Camada de Ligação Lógica

---



Engenharia Informática  
Universidade do Minho  
2019/2020

## Grupo - PL6.G06



Paulo Lima | a89983



Mafalda Costa | a83919



Maria Silva | a83840

# Objetivo

---

- Ganhar conhecimentos a nível da camada de ligação lógica
- Utilização as tecnologias Ethernet e o protocolo ARP

# Questões e Respostas

## 3 - Captura e análise de tramas Ethernet

+	41	6.958782563	192.168.100.220	193.136.19.40	HTTP	399	GET / HTTP/1.1
+	43	6.960143270	193.136.19.40	192.168.100.220	HTTP	547	HTTP/1.1 301 Moved Permanently (text/html)

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

O endereço MAC de origem é o [a0:2b:b8:23:fb:d9] e o endereço MAC de destino é o [00:0c:29:d2:19:f0].

```
▶ Frame 41: 399 bytes on wire (3192 bits), 399 bytes captured (3192 bits) on interface 0
▼ Ethernet II, Src: HewlettP_23:fb:d9 (a0:2b:b8:23:fb:d9), 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)
    .... ..0. .... = LG bit: Globally unique address (factory default)
    .... ..0. .... = IG bit: Individual address (unicast)
  ▼ Source: HewlettP_23:fb:d9 (a0:2b:b8:23:fb:d9)
    Address: HewlettP_23:fb:d9 (a0:2b:b8:23:fb:d9)
    .... ..0. .... = LG bit: Globally unique address (factory default)
    .... ..0. .... = IG bit: Individual address (unicast)
  Type: IPv4 (0x0800)
▶ Internet Protocol Version 4, Src: 192.168.100.220, Dst: 193.136.19.40
▶ Transmission Control Protocol, Src Port: 41074, Dst Port: 80, Seq: 1, Ack: 1, Len: 333
▼ Hypertext Transfer Protocol
  ▼ GET / HTTP/1.1\r\n
    ▼ [Expert Info (Chat/Sequence): GET / HTTP/1.1\r\n]
      [GET / HTTP/1.1\r\n]
      [Severity level: Chat]
      [Group: Sequence]
      Request Method: GET
      Request URI: /
      Request Version: HTTP/1.1
      Host: miei.di.uminho.pt\r\n
      User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:70.0) Gecko/20100101 Firefox/7
      Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
      Accept-Language: en-US,en;q=0.5\r\n
      Accept-Encoding: gzip, deflate\r\n
      DNT: 1\r\n
      Connection: keep-alive\r\n
      Upgrade-Insecure-Requests: 1\r\n
      \r\n
      [Full request URI: http://miei.di.uminho.pt/]
      [HTTP request 1/1]
      [Response in frame: 43]
```

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

O endereço mac refere-se a um endereço físico atribuído a uma interface de rede.

Como podemos observar na imagem, a origem(source) refere-se ao endereço físico do nosso computador e o destination refere-se ao endereço físico do router com o qual estamos a comunicar.

## 3) Qual o valor hexadecimal do campo Type da trama Ethernet? O que significa?

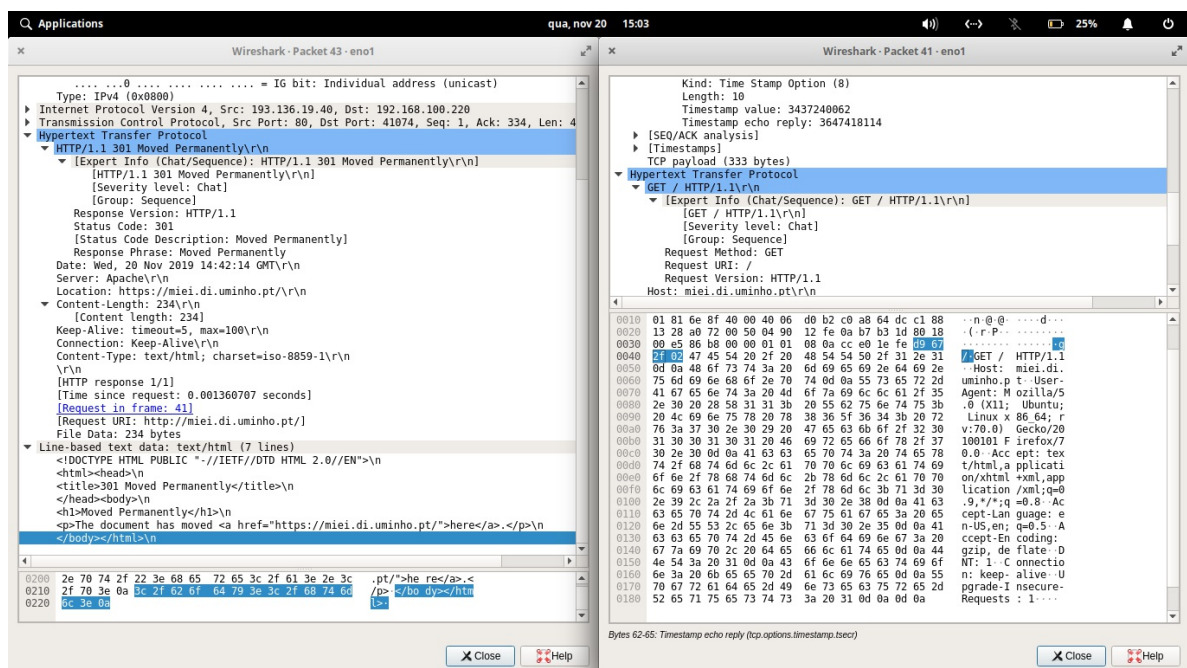
O valor hexadecimal do campo Type é [0x0800] o que significa que se trata do tipo IPv4.

## 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.

- Foram capturados 399 bytes.

```
▶ Frame 41: 399 bytes on wire (3192 bits), 399 bytes captured (3192 bits) on interface 0
▶ Ethernet II, Src: HewlettP_23:fb:d9 (a0:2b:b8:23:fb:d9), Dst: Vmware_d2:19:f0 (00:0c:29:
▶ Internet Protocol Version 4, Src: 192.168.100.220, Dst: 193.136.19.40
▶ Transmission Control Protocol, Src Port: 41074, Dst Port: 80, Seq: 1, Ack: 1, Len: 333
▶ Hypertext Transfer Protocol
```

- O caractere "G" encontra-se no byte 65.
- A percentagem da sobrecarga introduzida pela pilha protocolar é dada por:  $((65 \cdot 100) / 399)$  16,290726%



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

- O campo FCS não está a ser utilizado pois como foi utilizada uma rede ethernet que é considerada muito segura, não existiu necessidade de inserir esse campo.

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

O endereço MAC origem é o [00:0c:29:d2:19:f0] e o endereço IP correspondente o [193.136.19.40]. Este endereço corresponde ao endereço de destino do link dado (<http://miei.di.uminho.pt>).

```
▶ Source: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
```

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

O endereço MAC destino é o [a0:2b:b8:23:fb:d9] e o endereço IP correspondente o [192.168.100.220] corresponde ao nosso endereço, que deu origem ao pedido inicial.

```
▶ Ethernet II, Src: HewlettP_23:fb:d9 (a0:2b:b8:23:fb:d9), Dst: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
▶ Internet Protocol Version 4, Src: 192.168.100.220, Dst: 193.136.19.40
```

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

Os protocolos existentes são: IP, TCP, Ethernet e HTTP.

```
▶ Ethernet II, Src: Vmware_d2:19:f0 (00:0c:29:d2:19:f0), Dst: HewlettP_23:fb:d9 (a0:2b:b8:23:fb:d9)
▶ Internet Protocol Version 4, Src: 193.136.19.40, Dst: 192.168.100.220
▶ Transmission Control Protocol, Src Port: 80, Dst Port: 41074, Seq: 1, Ack: 334, Len: 481
▶ Hypertext Transfer Protocol
```

## 4 - Protocolo ARP

```
root@hostlinux:/home/linux/Documents/NON GIT/escola/3ano/rc/tp3# ping -c 6 192.168.100.190
PING 192.168.100.190 (192.168.100.190) 56(84) bytes of data:
64 bytes from 192.168.100.190: icmp_seq=1 ttl=64 time=0.706 ms
64 bytes from 192.168.100.190: icmp_seq=2 ttl=64 time=0.344 ms
64 bytes from 192.168.100.190: icmp_seq=3 ttl=64 time=0.347 ms
64 bytes from 192.168.100.190: icmp_seq=4 ttl=64 time=0.349 ms
64 bytes from 192.168.100.190: icmp_seq=5 ttl=64 time=0.394 ms
64 bytes from 192.168.100.190: icmp_seq=6 ttl=64 time=0.332 ms

--- 192.168.100.190 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5101ms
rtt min/avg/max/mdev = 0.332/0.412/0.706/0.132 ms
```

9) Observe o conteúdo da tabela ARP. Explique o significado de cada uma das colunas.

- Os valores hex source e destination na trama são(estao na foto).

```
root@hostlinux:/home/linux/Documents/NON GIT/escola/3ano/rc/tp3# arp -d -a
gateway (172.26.254.254) at 00:d0:03:ff:94:00 [ether] on wlp1s0
```

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 é o [192.168.100.203], e o endereço de destino é o endereço de broadcast [0.0.0.0]. A nossa tabela ARP não tem um endereço MAC associado ao ip destino do ping. Por isso perguntamos a todos os dispositivos associados à rede e apenas o que se identificar responde/manda um "reply".

676 14.674997720	BizlinkK_07:8b:e5	Broadcast	ARP	42 who has 192.168.100.190? Tell 192.168.100.203
677 14.675339650	AsustekC_0f:38:da	BizlinkK_07:8b:e5	ARP	60 192.168.100.190 is at 54:a0:50:0f:38:da

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

O valor hexadecimal do campo "Type" da trama ethernet é (0x0806), o que indica que estamos a usar o protocolo ARP.

Type : ARP (0x0806)

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

O valor do campo ARP opcode é 1, o que especifica um request.

```
▼ Address Resolution Protocol (request)
  Hardware type: Ethernet (1)
  Protocol type: IPv4 (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
  Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
  Target IP address: 192.168.100.203
```

## 13) Identifique que tipo de endereços está contido na mensagem ARP? Que conclui?

Temos endereços de IP e MAC, sendo que através do envio de um request pelo broadcast conseguimos informar todos os IPs presentes de forma a que apenas o Target IP responda com o seu endereço MAC correspondente.

```
Sender MAC address: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
Sender IP address: 192.168.100.254
Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
Target IP address: 192.168.100.203
```

## 14) Explícite que tipo de pedido ou pergunta é feito pelo host de origem?

É efetuado um pedido do tipo ARP. Como inicialmente não existe nenhuma associação do IP ao MAC então é enviada uma mensagem ARP para todos os hosts da rede de forma a que o endereço requisitado responda.







**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 Target IP do request ARP gratuito é diferente do request normal, sendo que o Target IP é igual ao Sender IP, logo apenas o router é responde ao request.

142	26.856726014	BizlinkK_07:8b:e5	Broadcast	ARP	42 Gratuitous ARP for 192.168.100.203 (Request)
145	27.375358798	Vmware_d2:19:f0	Broadcast	ARP	60 Who has 192.168.100.226? Tell 192.168.100.254
146	27.856876769	BizlinkK_07:8b:e5	Broadcast	ARP	42 Gratuitous ARP for 192.168.100.203 (Request)
148	28.374748741	Vmware_d2:19:f0	Broadcast	ARP	60 Who has 192.168.100.226? Tell 192.168.100.254
150	28.857000267	BizlinkK_07:8b:e5	Broadcast	ARP	42 Gratuitous ARP for 192.168.100.203 (Request)
154	29.375429620	Vmware_d2:19:f0	Broadcast	ARP	60 Who has 192.168.100.226? Tell 192.168.100.254
155	29.857128473	BizlinkK_07:8b:e5	Broadcast	ARP	42 Gratuitous ARP for 192.168.100.203 (Request)
159	30.411253916	Vmware_d2:19:f0	Broadcast	ARP	60 Who has 192.168.100.226? Tell 192.168.100.254
160	30.461462900	AsustekC_24:de:25	Broadcast	ARP	60 Who has 192.168.100.185? Tell 192.168.100.156
161	30.857213566	BizlinkK_07:8b:e5	Broadcast	ARP	42 Gratuitous ARP for 192.168.100.203 (Request)
163	31.410807452	Vmware_d2:19:f0	Broadcast	ARP	60 Who has 192.168.100.226? Tell 192.168.100.254
164	31.857352623	BizlinkK_07:8b:e5	Broadcast	ARP	42 Gratuitous ARP for 192.168.100.203 (Request)
174	32.411471869	Vmware_d2:19:f0	Broadcast	ARP	60 Who has 192.168.100.226? Tell 192.168.100.254

▶	Frame 142: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
▼	Ethernet II, Src: BizlinkK_07:8b:e5 (9c:eb:e8:07:8b:e5), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
▶	Destination: Broadcast (ff:ff:ff:ff:ff:ff)
▶	Source: BizlinkK_07:8b:e5 (9c:eb:e8:07:8b:e5)
	Type: ARP (0x0806)
▼	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: BizlinkK_07:8b:e5 (9c:eb:e8:07:8b:e5)
	Sender IP address: 192.168.100.203
	Target MAC address: Broadcast (ff:ff:ff:ff:ff:ff)
	Target IP address: 192.168.100.203

## 5 - Domínios de colisão

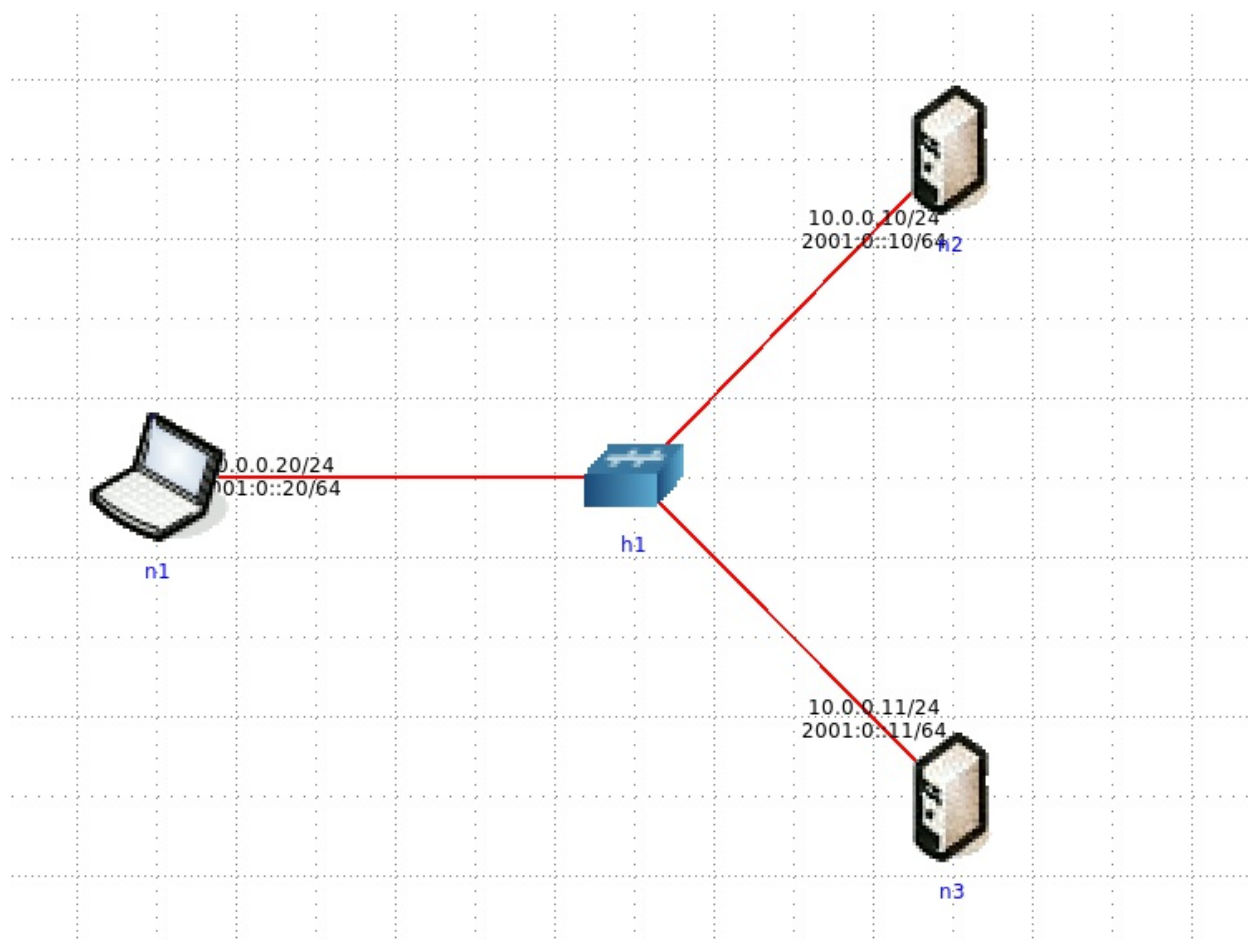
17) Faça ping de n1 para n2. Verifique com a opção tcpdump como flui o tráfego nas diversas interfaces dos vários dispositivos. Que conclui?

```
root@n1:/tmp/pycore.33317/n1.conf# ping -c 1 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_seq=1 ttl=64 time=0.155 ms

--- 10.0.0.10 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.155/0.155/0.155/0.000 ms
root@n1:/tmp/pycore.33317/n1.conf#
```

*Comando ping no terminal n1*

Um hub é um dispositivo que une várias portas num único segmento. Por isso, os hosts n2 e n3 recebem um tráfego uniforme, com a exceção do tráfego enviado por eles próprios.



*Topologia*

```

root@n2:/tmp/pycore.35317/n2.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
15:49:13.785027 IP6 n2 > ip6-allrouters: ICMP6, router solicitation, length 16
15:49:13.785045 IP6 fe80::200:ff:feaa:0 > ip6-allrouters: ICMP6, router solicitation, length 16
15:49:14.985054 IP6 fe80::7cd5:dfff:fe2:c399 > ip6-allrouters: ICMP6, router solicitation, length 16
15:49:15.497080 IP6 fe80::ac47:ffff:fe4a:c2f9 > ip6-allrouters: ICMP6, router solicitation, length 16
15:49:15.497104 IP6 fe80::200:ff:feaa:2 > ip6-allrouters: ICMP6, router solicitation, length 16
15:49:17.139096 IP6 fe80::7cd5:dfff:fe2:c399.mdns > ff02::fb.mdns: 0 [3a] [7q] PTR (QM)? _ipps._tcp.local. TXT (QM)? hostlinuz._http._tcp.local. SRV (QM)? hostlinuz._http._tcp.local. PTR (QM)? _http._tcp.local. PTR (QM)? _https._tcp.local. PTR (QM)? _ftp._tcp.local. PTR (QM)? _ipp._tcp.local. (160)
15:49:17.519512 IP6 fe80::ac47:ffff:fe4a:c2f9.mdns > ff02::fb.mdns: 0 [3a] [7a] TXT (QM)? hostlinuz._http._tcp.local. PTR (QM)? _http._tcp.local. PTR (QM)? _https._tcp.local. PTR (QM)? _ftp._tcp.local. PTR (QM)? _http._tcp.local. PTR (QM)? _ipps._tcp.local. SRV (QM)? hostlinuz._http._tcp.local. (160)
15:49:28.889013 IP6 fe80::200:ff:feaa:0 > ip6-allrouters: ICMP6, router solicitation, length 16
15:49:28.877061 IP6 n2 > ip6-allrouters: ICMP6, router solicitation, length 16

```

Comando tcdump no terminal n2 : hub

```

root@n3:/tmp/pycore.35317/n3.conf# tcpdump -vv
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
15:44:32.752914 IP6 (flowlabel 0x17068, hlim 255, next-header UDP (17) payload length: 52) fe80::6467:f8ff:fe5c:9c1.mdns > ff02::fb.mdns: [bad udp cksun 0x6449 -> 0x9e7f!] 0 SRV (QM)? hostlinuz._http._tcp.local. (44)
15:44:32.753091 IP6 (flowlabel 0x17068, hlim 255, next-header UDP (17) payload length: 104) fe80::6467:f8ff:fe5c:9c1.mdns > ff02::fb.mdns: [bad udp cksun 0x647d -> 0x6804!] 0*- [0q] 2/0/0 hostlinuz._http._tcp.local. (Cache flush) SRV hostlinuz.local.:80 0 0, hostlinuz.local. (Cache flush) AAAA fe80::6467:f8ff:fe5c:9c1 (96)
15:44:34.123502 IP6 (flowlabel 0xf6aa4, hlim 255, next-header UDP (17) payload length: 52) fe80::547b:a7ff:fe77:bd3a.mdns > ff02::fb.mdns: [bad udp cksun 0xb6f1 -> 0x4bd7!] 0 SRV (QM)? hostlinuz._http._tcp.local. (44)
15:44:34.123754 IP6 (flowlabel 0xf6aa4, hlim 255, next-header UDP (17) payload length: 104) fe80::547b:a7ff:fe77:bd3a.mdns > ff02::fb.mdns: [bad udp cksun 0xb725 -> 0xc2b3!] 0*- [0q] 2/0/0 hostlinuz._http._tcp.local. (Cache flush) SRV hostlinuz.local.:80 0 0, hostlinuz.local. (Cache flush) AAAA fe80::547b:a7ff:fe77:bd3a (96)
15:45:01.545120 IP6 (hlim 255, next-header ICMPv6 (58) payload length: 16) fe80::200:ff:feaa:0 > ip6-allrouters: [icmp6 sum ok] ICMP6, router solicitation, length 16
    source link-address option (1), length 8 (1): 00:00:00:aa:00:00
    0x0000: 0000 00aa 0000
15:45:17.929339 IP6 (hlim 255, next-header ICMPv6 (58) payload length: 16) fe80::200:ff:feaa:1 > ip6-allrouters: [icmp6 sum ok] ICMP6, router solicitation, length 16
    source link-address option (1), length 8 (1): 00:00:00:aa:00:01
    0x0000: 0000 00aa 0001
15:45:24.503532 IP6 (flowlabel 0xf6aa4, hlim 255, next-header UDP (17) payload length: 132) fe80::547b:a7ff:fe77:bd3a.mdns > ff02::fb.mdns: [bad udp cksun 0xb741 -> 0x8981!] 0 [2a] [6q] TXT (QM)? hostlinuz._http._tcp.local. PTR (QM)? _http._tcp.local. PTR (QM)? _https._tcp.local. PTR (QM)? _ftp._tcp.local. PTR (QM)? _ipp._tcp.local. PTR (QM)? _ipps._tcp.local. _http._tcp.local. PTR hostlinuz._http._tcp.local., hostlinuz._http._tcp.local. TXT "" (124)
15:45:25.179997 IP6 (flowlabel 0x17068, hlim 255, next-header UDP (17) payload length: 132) fe80::6467:f8ff:fe5c:9c1.mdns > ff02::fb.mdns: [bad udp cksun 0x6499 -> 0xdc29!] 0 [2a] [6q] TXT (QM)? hostlinuz._http._tcp.local. PTR (QM)? _http._tcp.local. PTR (QM)? _https._tcp.local. PTR (QM)? _ftp._tcp.local. PTR (QM)? _ipp._tcp.local. PTR (QM)? _ipps._tcp.local. _http._tcp.local. PTR hostlinuz._http._tcp.local., hostlinuz._http._tcp.local. TXT "" (124)
15:45:34.313054 IP6 (hlim 255, next-header ICMPv6 (58) payload length: 16) fe80::547b:a7ff:fe77:bd3a > ip6-allrouters: [icmp6 sum ok] ICMP6, router solicitation, length 16
    source link-address option (1), length 8 (1): 66:67:f8:5c:09:c1
    0x0000: 6667 f85c 09c1
15:46:10.770236 IP6 (flowlabel 0x17068, hlim 255, next-header UDP (17) payload length: 52) fe80::6467:f8ff:fe5c:9c1.mdns > ff02::fb.mdns: [bad udp cksun 0x6449 -> 0x9e7f!] 0 SRV (QM)? hostlinuz._http._tcp.local. (44)
15:46:10.778449 IP6 (flowlabel 0x17068, hlim 255, next-header UDP (17) payload length: 104) fe80::6467:f8ff:fe5c:9c1.mdns > ff02::fb.mdns: [bad udp cksun 0x647d -> 0x6804!] 0*- [0q] 2/0/0 hostlinuz._http._tcp.local. (Cache flush) SRV hostlinuz.local.:80 0 0, hostlinuz.local. (Cache flush) AAAA fe80::6467:f8ff:fe5c:9c1 (96)
15:46:10.952239 IP6 (flowlabel 0xf6aa4, hlim 255, next-header UDP (17) payload length: 52) fe80::547b:a7ff:fe77:bd3a.mdns > ff02::fb.mdns: [bad udp cksun 0xb6f1 -> 0x4bd7!] 0 SRV (QM)? hostlinuz._http._tcp.local. (44)
15:46:10.952486 IP6 (flowlabel 0xf6aa4, hlim 255, next-header UDP (17) payload length: 104) fe80::547b:a7ff:fe77:bd3a.mdns > ff02::fb.mdns: [bad udp cksun 0xb725 -> 0xc2b3!] 0*- [0q] 2/0/0 hostlinuz._http._tcp.local. (Cache flush) SRV hostlinuz.local.:80 0 0, hostlinuz.local. (Cache flush) AAAA fe80::547b:a7ff:fe77:bd3a (96)

```

Comando tcdump no terminal n3 : hub

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.

Com um switch ao invés de o tráfego ser enviado para todos os dispositivos (como acontece com um hub), o tráfego é apenas direcionado para um só nó. Com os switch o risco de existir colisões é menor, sendo que estes dispositivos segmentam a rede internamente e cada porta do dispositivo corresponde a um domínio de colisão diferente. Os hub, ao enviarem informação para todos os hosts, concentram toda esta informação num único canal de transmissão aumentando a probabilidade de colisões.

```
Activities Terminal x Wed 22:29
File Edit View Search Terminal Help
root@n2:/tmp/pcycore.40889/n2.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
22:28:38.557560 IP6 fe80::648e:bcbf:feac:77be > ip6-allrouters: ICMP6, router solicitation, length 16
22:28:39.321715 IP6 fe80::200:ff:feaa:0 > ip6-allrouters: ICMP6, router solicitation, length 16
22:28:39.833617 IP6 n2 > ip6-allrouters: ICMP6, router solicitation, length 16
22:28:40.089631 IP6 fe80::b452:7aff:fe4:c5de > ip6-allrouters: ICMP6, router solicitation, length 16
22:28:42.138689 IP6 fe80::b452:7aff:fe4:c5de.mdns > ff02::fb.mdns: 0 [3a] [13a] TXT (QM)? hostlinuz, htt
p, tcp.local, PTR (QM)? http, tcp.local, PTR (QM)? https, tcp.local, PTR (QM)?_ipp, tcp.local, PTR (QM)?_
nfs, tcp.local, PTR (QM)?_ftp, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_sftp-ssh, tcp.local, PTR (QM)?_smb, tcp.local, PTR (QM)?_afpovertcp, tcp.local, PTR (QM)?_nfs, tcp.local, SRV (QM)? hostlinuz, http, tcp.local, (245)
22:28:42.362696 IP6 fe80::648e:bcbf:feac:77be.mdns > ff02::fb.mdns: 0 [3a] [13a] TXT (QM)? hostlinuz, htt
p, tcp.local, PTR (QM)? http, tcp.local, PTR (QM)? https, tcp.local, PTR (QM)?_ipp, tcp.local, PTR (QM)?_
nfs, tcp.local, PTR (QM)?_ftp, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_sftp-ssh, tcp.local, PTR (QM)?_smb, tcp.local, PTR (QM)?_afpovertcp, tcp.local, PTR (QM)?_nfs, tcp.local, SRV (QM)? hostlinuz, http, tcp.local, (245)
22:28:51.065783 IP6 fe80::200:ff:feaa:2 > ip6-allrouters: ICMP6, router solicitation, length 16
22:28:53.145620 IP6 fe80::648e:bcbf:feac:77be > ip6-allrouters: ICMP6, router solicitation, length 16
22:28:55.962122 IP6 fe80::200:ff:feaa:0 > ip6-allrouters: ICMP6, router solicitation, length 16
22:28:58.009576 IP6 n2 > ip6-allrouters: ICMP6, router solicitation, length 16
22:28:58.009590 IP6 fe80::b452:7aff:fe4:c5de > ip6-allrouters: ICMP6, router solicitation, length 16
22:28:58.140024 IP6 fe80::b452:7aff:fe4:c5de.mdns > ff02::fb.mdns: 0 [3a] [13a] TXT (QM)? hostlinuz, htt
p, tcp.local, PTR (QM)? http, tcp.local, PTR (QM)? https, tcp.local, PTR (QM)?_ipp, tcp.local, PTR (QM)?_
nfs, tcp.local, PTR (QM)?_ftp, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_sftp-ssh, tcp.local, PTR (QM)?_smb, tcp.local, PTR (QM)?_afpovertcp, tcp.local, PTR (QM)?_nfs, tcp.local, SRV (QM)? hostlinuz, http, tcp.local, (245)
22:28:58.363356 IP6 fe80::648e:bcbf:feac:77be.mdns > ff02::fb.mdns: 0 [3a] [13a] TXT (QM)? hostlinuz, htt
p, tcp.local, PTR (QM)? http, tcp.local, PTR (QM)? https, tcp.local, PTR (QM)?_ipp, tcp.local, PTR (QM)?_
nfs, tcp.local, PTR (QM)?_ftp, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_sftp-ssh, tcp.local, PTR (QM)?_smb, tcp.local, PTR (QM)?_afpovertcp, tcp.local, PTR (QM)?_nfs, tcp.local, SRV (QM)? hostlinuz, http, tcp.local, (245)
22:29:20.530154 IP6 fe80::200:ff:feaa:2 > ip6-allrouters: ICMP6, router solicitation, length 16
22:29:24.633626 IP6 fe80::648e:bcbf:feac:77be > ip6-allrouters: ICMP6, router solicitation, length 16
22:29:28.729680 IP6 fe80::200:ff:feaa:0 > ip6-allrouters: ICMP6, router solicitation, length 16
22:29:30.140996 IP6 fe80::b452:7aff:fe4:c5de.mdns > ff02::fb.mdns: 0 [3a] [13a] TXT (QM)? hostlinuz, htt
p, tcp.local, PTR (QM)? http, tcp.local, PTR (QM)? https, tcp.local, PTR (QM)?_ipp, tcp.local, PTR (QM)?_
nfs, tcp.local, PTR (QM)?_ftp, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_sftp-ssh, tcp.local, PTR (QM)?_smb, tcp.local, PTR (QM)?_afpovertcp, tcp.local, PTR (QM)?_nfs, tcp.local, SRV (QM)? hostlinuz, http, tcp.local, (245)
22:29:30.364421 IP6 fe80::648e:bcbf:feac:77be.mdns > ff02::fb.mdns: 0 [3a] [13a] TXT (QM)? hostlinuz, htt
p, tcp.local, PTR (QM)? http, tcp.local, PTR (QM)? https, tcp.local, PTR (QM)?_ipp, tcp.local, PTR (QM)?_
nfs, tcp.local, PTR (QM)?_ftp, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_sftp-ssh, tcp.local, PTR (QM)?_smb, tcp.local, PTR (QM)?_afpovertcp, tcp.local, PTR (QM)?_nfs, tcp.local, SRV (QM)? hostlinuz, http, tcp.local, (245)
22:29:32.825872 IP6 fe80::b452:7aff:fe4:c5de > ip6-allrouters: ICMP6, router solicitation, length 16
22:29:34.873636 IP6 n2 > ip6-allrouters: ICMP6, router solicitation, length 16
^C
20 packets captured
20 packets received by filter
0 packets dropped by kernel
root@n2:/tmp/pcycore.40889/n2.conf#

root@n3:/tmp/pcycore.40889/n3.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
22:28:37.785608 IP6 n3 > ip6-allrouters: ICMP6, router solicitation, length 16
22:28:39.321711 IP6 fe80::200:ff:feaa:0 > ip6-allrouters: ICMP6, router solicitation, length 16
22:28:39.834020 IP6 fe80::200:ff:feaa:1 > ip6-allrouters: ICMP6, router solicitation, length 16
22:28:40.089618 IP6 fe80::e439:43ff:fec6:e097 > ip6-allrouters: ICMP6, router solicitation, length 16
22:28:42.089626 IP6 fe80::b452:7aff:fe4:c5de > ip6-allrouters: ICMP6, router solicitation, length 16
22:28:42.138688 IP6 fe80::b452:7aff:fe4:c5de.mdns > ff02::fb.mdns: 0 [3a] [13a] TXT (QM)? hostlinuz, htt
p, tcp.local, PTR (QM)? http, tcp.local, PTR (QM)? https, tcp.local, PTR (QM)?_ipp, tcp.local, PTR (QM)?_
nfs, tcp.local, PTR (QM)?_ftp, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_sftp-ssh, tcp.local, PTR (QM)?_smb, tcp.local, PTR (QM)?_afpovertcp, tcp.local, PTR (QM)?_nfs, tcp.local, SRV (QM)? hostlinuz, http, tcp.local, (245)
22:28:42.524008 IP6 fe80::e439:43ff:fec6:e097.mdns > ff02::fb.mdns: 0 [3a] [13a] TXT (QM)? hostlinuz, htt
p, tcp.local, PTR (QM)? http, tcp.local, PTR (QM)? https, tcp.local, PTR (QM)?_ipp, tcp.local, PTR (QM)?_
nfs, tcp.local, PTR (QM)?_ftp, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_sftp-ssh, tcp.local, PTR (QM)?_smb, tcp.local, PTR (QM)?_afpovertcp, tcp.local, PTR (QM)?_nfs, tcp.local, SRV (QM)? hostlinuz, http, tcp.local, (245)
22:28:51.065627 IP6 n3 > ip6-allrouters: ICMP6, router solicitation, length 16
22:28:55.962119 IP6 fe80::200:ff:feaa:0 > ip6-allrouters: ICMP6, router solicitation, length 16
22:28:58.009586 IP6 fe80::b452:7aff:fe4:c5de > ip6-allrouters: ICMP6, router solicitation, length 16
22:28:58.009727 IP6 fe80::200:ff:feaa:1 > ip6-allrouters: ICMP6, router solicitation, length 16
22:28:58.140020 IP6 fe80::b452:7aff:fe4:c5de.mdns > ff02::fb.mdns: 0 [3a] [13a] TXT (QM)? hostlinuz, htt
p, tcp.local, PTR (QM)? http, tcp.local, PTR (QM)? https, tcp.local, PTR (QM)?_ipp, tcp.local, PTR (QM)?_
nfs, tcp.local, PTR (QM)?_ftp, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_sftp-ssh, tcp.local, PTR (QM)?_smb, tcp.local, PTR (QM)?_afpovertcp, tcp.local, PTR (QM)?_nfs, tcp.local, SRV (QM)? hostlinuz, http, tcp.local, (245)
22:28:58.524056 IP6 fe80::e439:43ff:fec6:e097.mdns > ff02::fb.mdns: 0 [3a] [13a] TXT (QM)? hostlinuz, htt
p, tcp.local, PTR (QM)? http, tcp.local, PTR (QM)? https, tcp.local, PTR (QM)?_ipp, tcp.local, PTR (QM)?_
nfs, tcp.local, PTR (QM)?_ftp, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_sftp-ssh, tcp.local, PTR (QM)?_smb, tcp.local, PTR (QM)?_afpovertcp, tcp.local, PTR (QM)?_nfs, tcp.local, SRV (QM)? hostlinuz, http, tcp.local, (245)
22:29:20.530802 IP6 n3 > ip6-allrouters: ICMP6, router solicitation, length 16
22:29:28.729676 IP6 fe80::e439:43ff:fec6:e097 > ip6-allrouters: ICMP6, router solicitation, length 16
22:29:30.140992 IP6 fe80::b452:7aff:fe4:c5de.mdns > ff02::fb.mdns: 0 [3a] [13a] TXT (QM)? hostlinuz, htt
p, tcp.local, PTR (QM)? http, tcp.local, PTR (QM)? https, tcp.local, PTR (QM)?_ipp, tcp.local, PTR (QM)?_
nfs, tcp.local, PTR (QM)?_ftp, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_webdav, tcp.local, PTR (QM)?_sftp-ssh, tcp.local, PTR (QM)?_smb, tcp.local, PTR (QM)?_afpovertcp, tcp.local, PTR (QM)?_nfs, tcp.local, SRV (QM)? hostlinuz, http, tcp.local, (245)
22:29:32.825867 IP6 fe80::b452:7aff:fe4:c5de > ip6-allrouters: ICMP6, router solicitation, length 16
22:29:34.873696 IP6 fe80::200:ff:feaa:1 > ip6-allrouters: ICMP6, router solicitation, length 16
^C
21 packets captured
21 packets received by filter
0 packets dropped by kernel
root@n3:/tmp/pcycore.40889/n3.conf#
```

Comando tcpdump no terminal n2 e n3 : switch

# Conclusões

---

Com este trabalho expandimos o nosso conhecimento acerca de tramas Ethernet e dos protocolos a esta associados. Sabemos que as estão organizadas em bytes, exploramos, através delas, o conceito de mac adress, desenvolvemos o nosso conhecimento acerca do protocolo HTTP. Este protocolo permite comunicar informação com a World Wide Web, através do FCS (Frame Check Sequence) garantimos que a informação enviada é a correta.

Quanto ao protocolo ARP, concluímos que este permite que diferentes PC's se encontrem numa rede ethernet. Através do protocolo ARP descobrimos o mac adress de um certo IP da rede em que nos encontramos. Quando um PC quer comunicar com outro mas não sabe o endereço físico deste, apenas o IP adress, é enviada uma mensagem de broadcast de forma a questionar-se a quem pertence esse IP, ao que o outro PC responde com o seu endereço físico.

Cada um dos protocolos existentes é utilizado com um objetivo específico, estes são depois, estruturados através de camadas lógicas que facilitam a ligação de redes.

Compreendemos a diferença entre o uso de um hub e um switch e as consequências do uso de cada um na nossa topologia, concluindo que o uso de um switch é mais vantajoso pois diminui o número de colisões entre pacotes.