# Redes de computadores

## Análise de tráfego com Wireshark

Prof. Luís Eduardo Tenório Silva
luis.silva@garanhuns.ifpe.edu.br

INSTITUTO FEDERAL
Pernambuco
Campus Garanhuns

# Análise de tráfego



- *Trobleshooting* de rede;
- Investigação de incidentes de segurança;
- Estudo dos protocolos de rede;
- Análise de desempenho da rede.

# Análise de tráfego



- Ferramentas de análise de rede
  - tcpdump/wireshark
  - traceroute/tracert
  - ping
  - nmap
  - nc
  - iptables
  - nperf
  - ...

# Wireshark



- Ferramenta de **análise de pacotes** (*packet sniffer*)
  - » Realiza passivamente uma cópia dos pacotes que trafegam em uma ou mais interfaces de rede;
  - » Identifica todos os cabeçalhos (*headers*) e cargas (*payload*) dos protocolos utilizados.

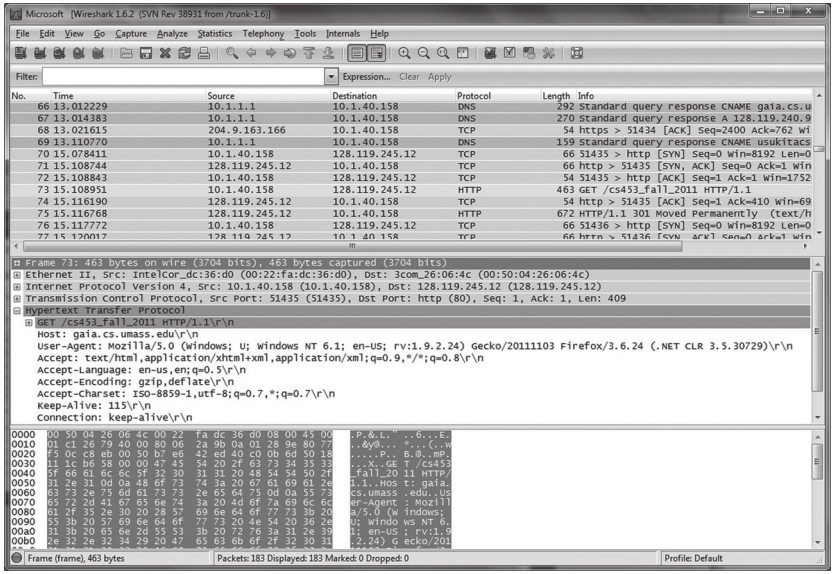



**Figura: Captura de pacotes usando o Wireshark**

# Wireshark

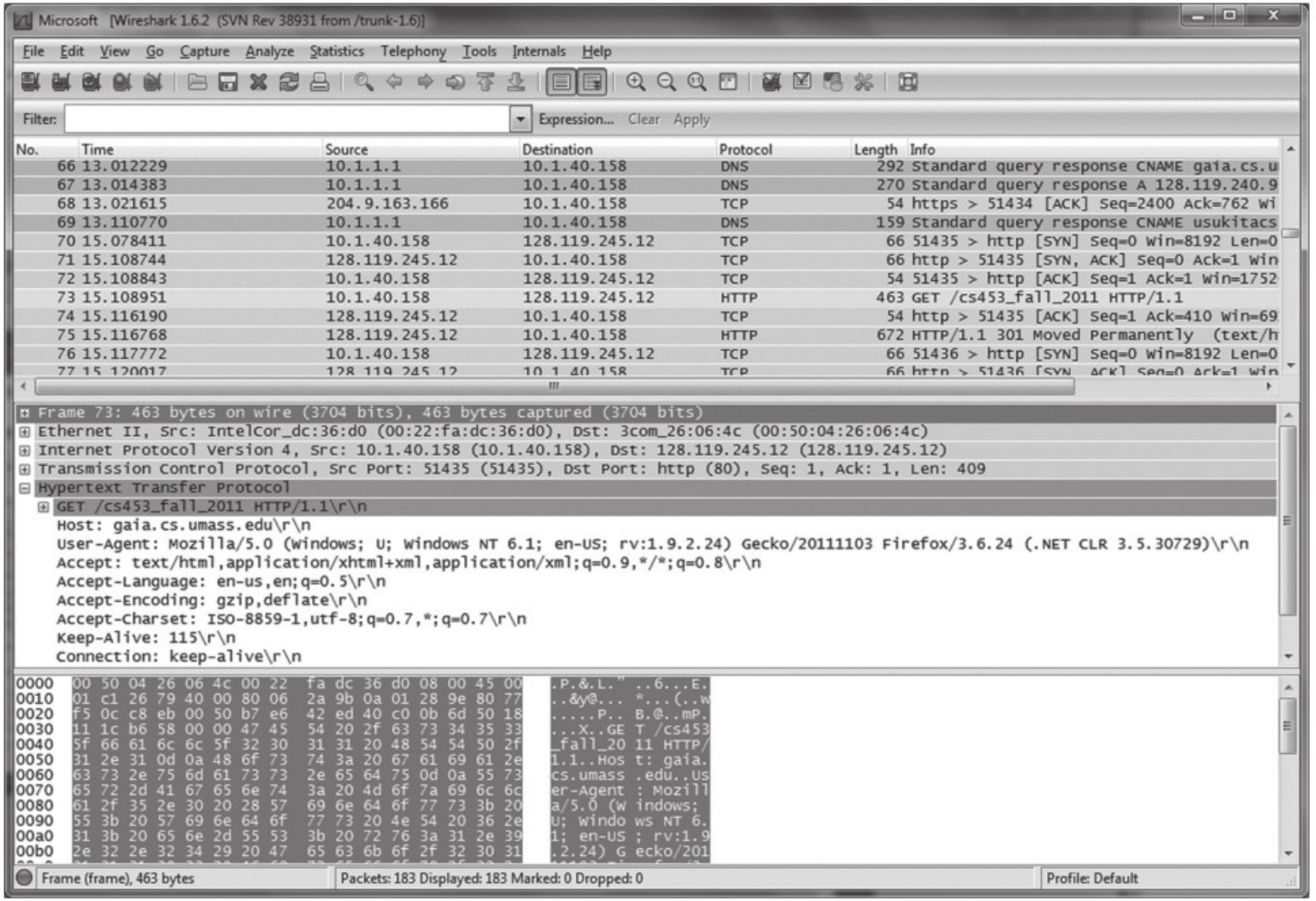


Menu de **comandos**

**Listagem** de pacotes capturados

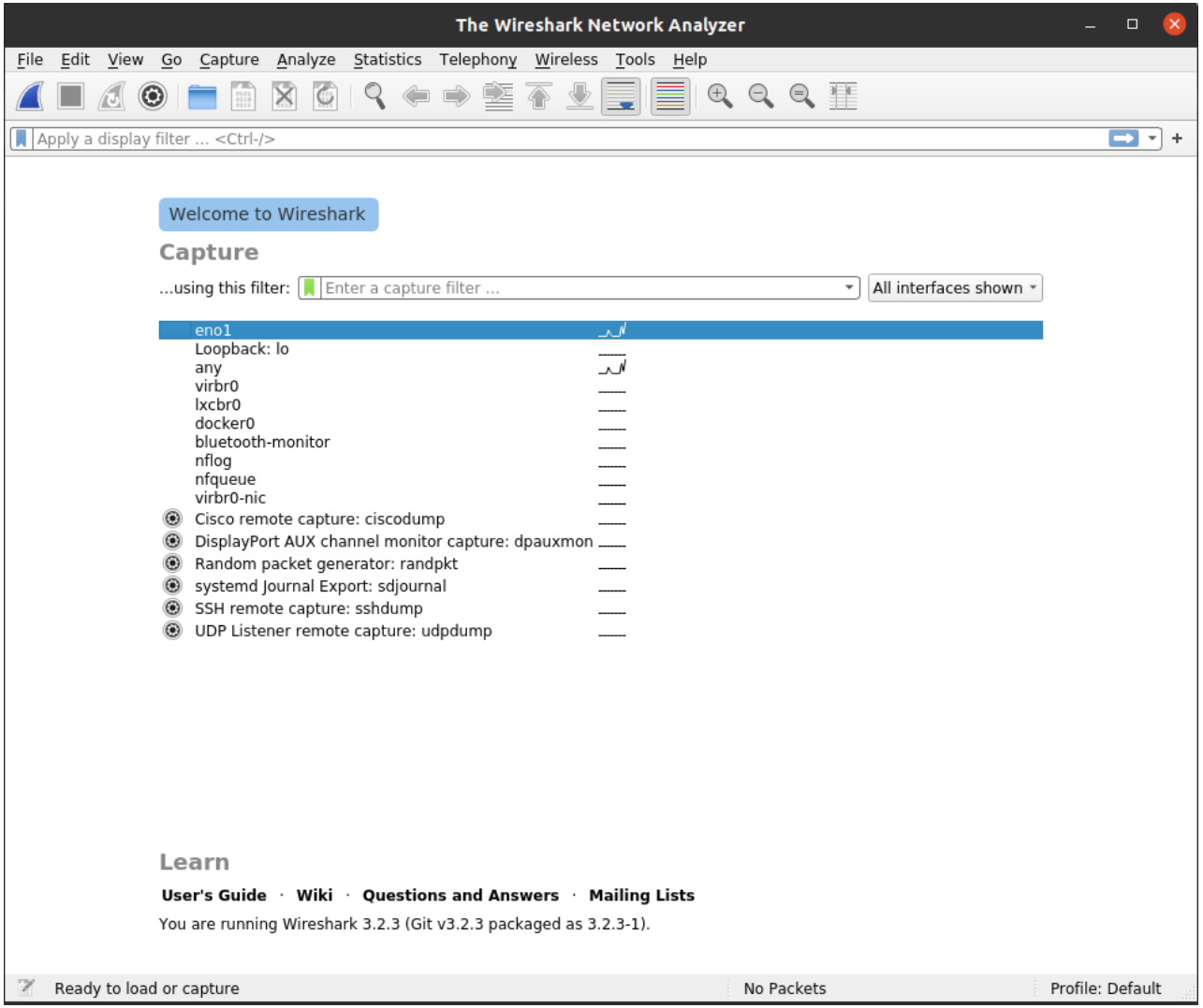
**Detalhes** do cabeçalho do pacote selecionado

**Conteúdo** do pacote em hexadecimal e ASCII



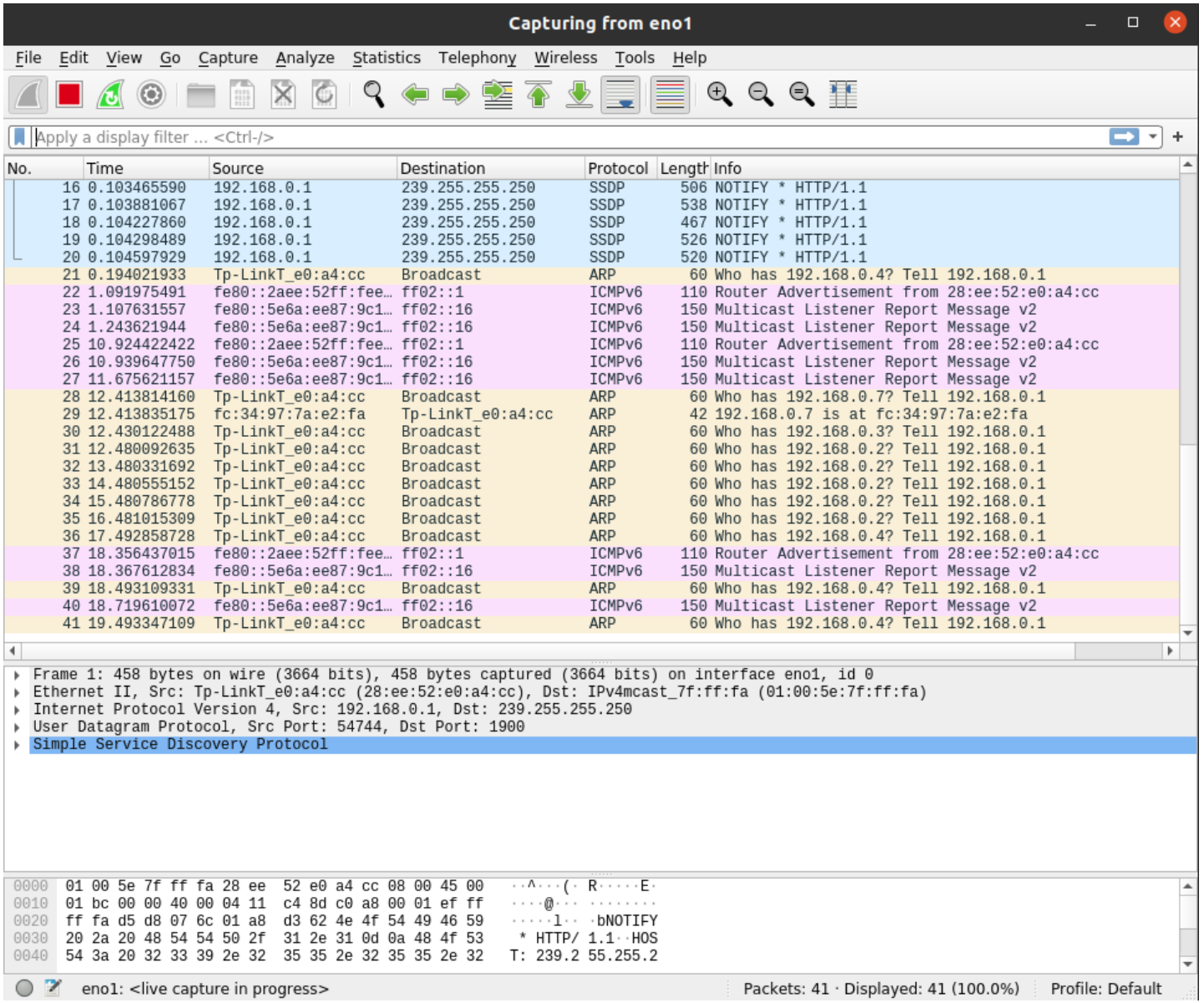
# Wireshark




The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Welcome to Wireshark
Capture
...using this filter: Enter a capture filter ...
All interfaces shown
eno1
Loopback: lo
any
virbr0
lxcbr0
docker0
bluetooth-monitor
nflog
nfqueue
virbr0-nic
Cisco remote capture: ciscodump
DisplayPort AUX channel monitor capture: dpauxmon
Random packet generator: randpkt
systemd Journal Export: sdjournal
SSH remote capture: sshdump
UDP Listener remote capture: udpdump
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists
You are running Wireshark 3.2.3 (Git v3.2.3 packaged as 3.2.3-1).
Ready to load or capture
No Packets
Profile: Default

# Wireshark




Capturing from eno1
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
16 0.103465590 192.168.0.1 239.255.255.250 SSDP 506 NOTIFY * HTTP/1.1
17 0.103881067 192.168.0.1 239.255.255.250 SSDP 538 NOTIFY * HTTP/1.1
18 0.104227860 192.168.0.1 239.255.255.250 SSDP 467 NOTIFY * HTTP/1.1
19 0.104298489 192.168.0.1 239.255.255.250 SSDP 526 NOTIFY * HTTP/1.1
20 0.104597929 192.168.0.1 239.255.255.250 SSDP 520 NOTIFY * HTTP/1.1
21 0.194021933 Tp-LinkT_e0:a4:cc Broadcast ARP 60 Who has 192.168.0.4? Tell 192.168.0.1
22 1.091975491 fe80::2aee:52ff:fee… ff02::1 ICMPv6 110 Router Advertisement from 28:ee:52:e0:a4:cc
23 1.107631557 fe80::5e6a:ee87:9c1… ff02::16 ICMPv6 150 Multicast Listener Report Message v2
24 1.243621944 fe80::5e6a:ee87:9c1… ff02::16 ICMPv6 150 Multicast Listener Report Message v2
25 10.924422422 fe80::2aee:52ff:fee… ff02::1 ICMPv6 110 Router Advertisement from 28:ee:52:e0:a4:cc
26 10.939647750 fe80::5e6a:ee87:9c1… ff02::16 ICMPv6 150 Multicast Listener Report Message v2
27 11.675621157 fe80::5e6a:ee87:9c1… ff02::16 ICMPv6 150 Multicast Listener Report Message v2
28 12.413814160 Tp-LinkT_e0:a4:cc Broadcast ARP 60 Who has 192.168.0.7? Tell 192.168.0.1
29 12.413835175 fc:34:97:7a:e2:fa Tp-LinkT_e0:a4:cc ARP 42 192.168.0.7 is at fc:34:97:7a:e2:fa
30 12.430122488 Tp-LinkT_e0:a4:cc Broadcast ARP 60 Who has 192.168.0.3? Tell 192.168.0.1
31 12.480092635 Tp-LinkT_e0:a4:cc Broadcast ARP 60 Who has 192.168.0.2? Tell 192.168.0.1
32 13.480331692 Tp-LinkT_e0:a4:cc Broadcast ARP 60 Who has 192.168.0.2? Tell 192.168.0.1
33 14.480555152 Tp-LinkT_e0:a4:cc Broadcast ARP 60 Who has 192.168.0.2? Tell 192.168.0.1
34 15.480786778 Tp-LinkT_e0:a4:cc Broadcast ARP 60 Who has 192.168.0.2? Tell 192.168.0.1
35 16.481015309 Tp-LinkT_e0:a4:cc Broadcast ARP 60 Who has 192.168.0.2? Tell 192.168.0.1
36 17.492858728 Tp-LinkT_e0:a4:cc Broadcast ARP 60 Who has 192.168.0.4? Tell 192.168.0.1
37 18.356437015 fe80::2aee:52ff:fee… ff02::1 ICMPv6 110 Router Advertisement from 28:ee:52:e0:a4:cc
38 18.367612834 fe80::5e6a:ee87:9c1… ff02::16 ICMPv6 150 Multicast Listener Report Message v2
39 18.493109331 Tp-LinkT_e0:a4:cc Broadcast ARP 60 Who has 192.168.0.4? Tell 192.168.0.1
40 18.719610072 fe80::5e6a:ee87:9c1… ff02::16 ICMPv6 150 Multicast Listener Report Message v2
41 19.493347109 Tp-LinkT_e0:a4:cc Broadcast ARP 60 Who has 192.168.0.4? Tell 192.168.0.1
Frame 1: 458 bytes on wire (3664 bits), 458 bytes captured (3664 bits) on interface eno1, id 0
Ethernet II, Src: Tp-LinkT_e0:a4:cc (28:ee:52:e0:a4:cc), Dst: IPv4mcast_7f:ff:fa (01:00:5e:7f:ff:fa)
Internet Protocol Version 4, Src: 192.168.0.1, Dst: 239.255.255.250
User Datagram Protocol, Src Port: 54744, Dst Port: 1900
Simple Service Discovery Protocol
0000 01 00 5e 7f ff fa 28 ee 52 e0 a4 cc 08 00 45 00 ··^···(· R·····E·
0010 01 bc 00 00 40 00 04 11 c4 8d c0 a8 00 01 ef ff ····@··· ········
0020 ff fa d5 d8 07 6c 01 a8 d3 62 4e 4f 54 49 46 59 ·····l·· ·bNOTIFY
0030 20 2a 20 48 54 54 50 2f 31 2e 31 0d 0a 48 4f 53 * HTTP/ 1.1··HOS
0040 54 3a 20 32 33 39 2e 32 35 35 2e 32 35 35 2e 32 T: 239.2 55.255.2
eno1: <live capture in progress>
Packets: 41 · Displayed: 41 (100.0%)
Profile: Default

# Wireshark



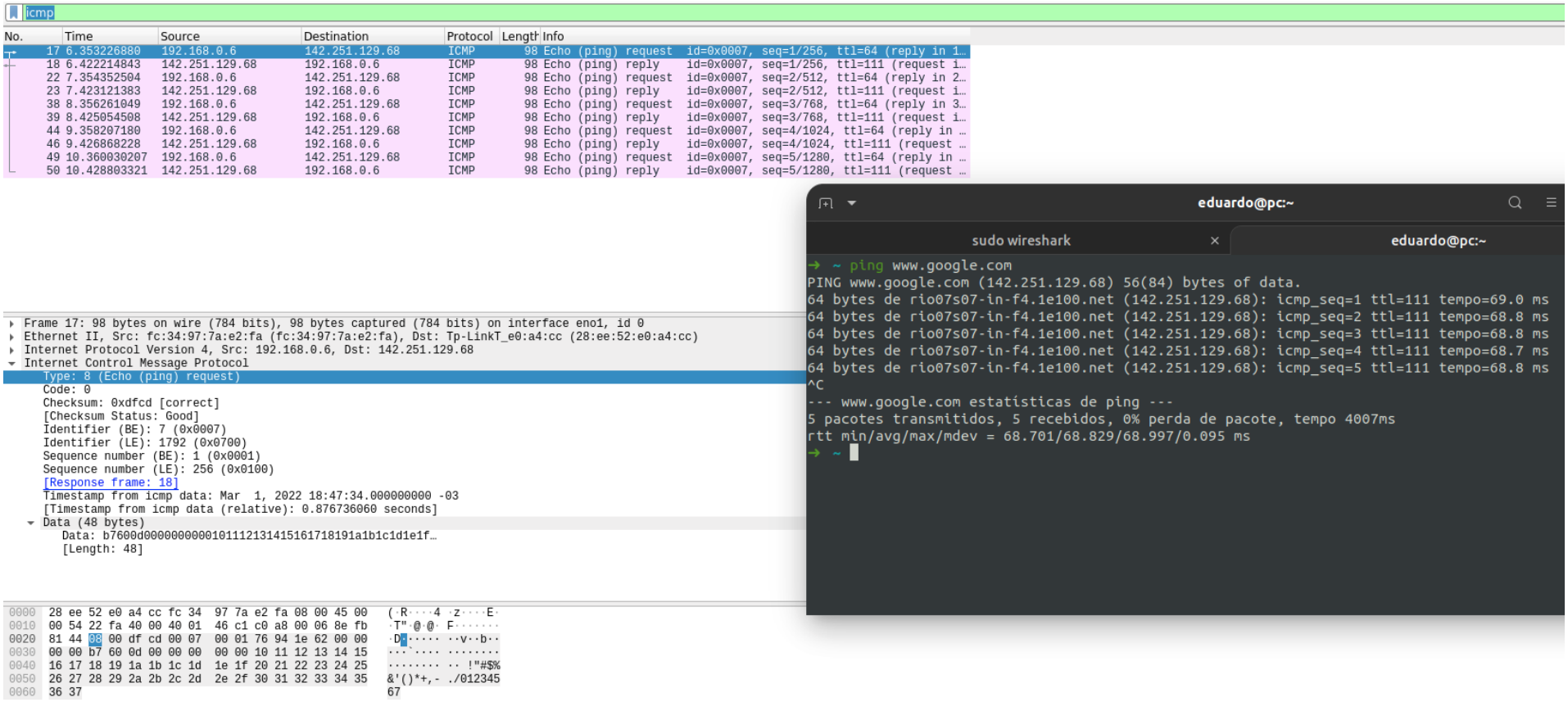
| Número | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| Número do pacote | Tempo que o pacote levou para chegar na placa de rede | Endereço de origem (rede/físico) | Endereço de destino (rede/físico) | Protocolo | Tamanho do pacote | Informações principais do pacote |

# Análise



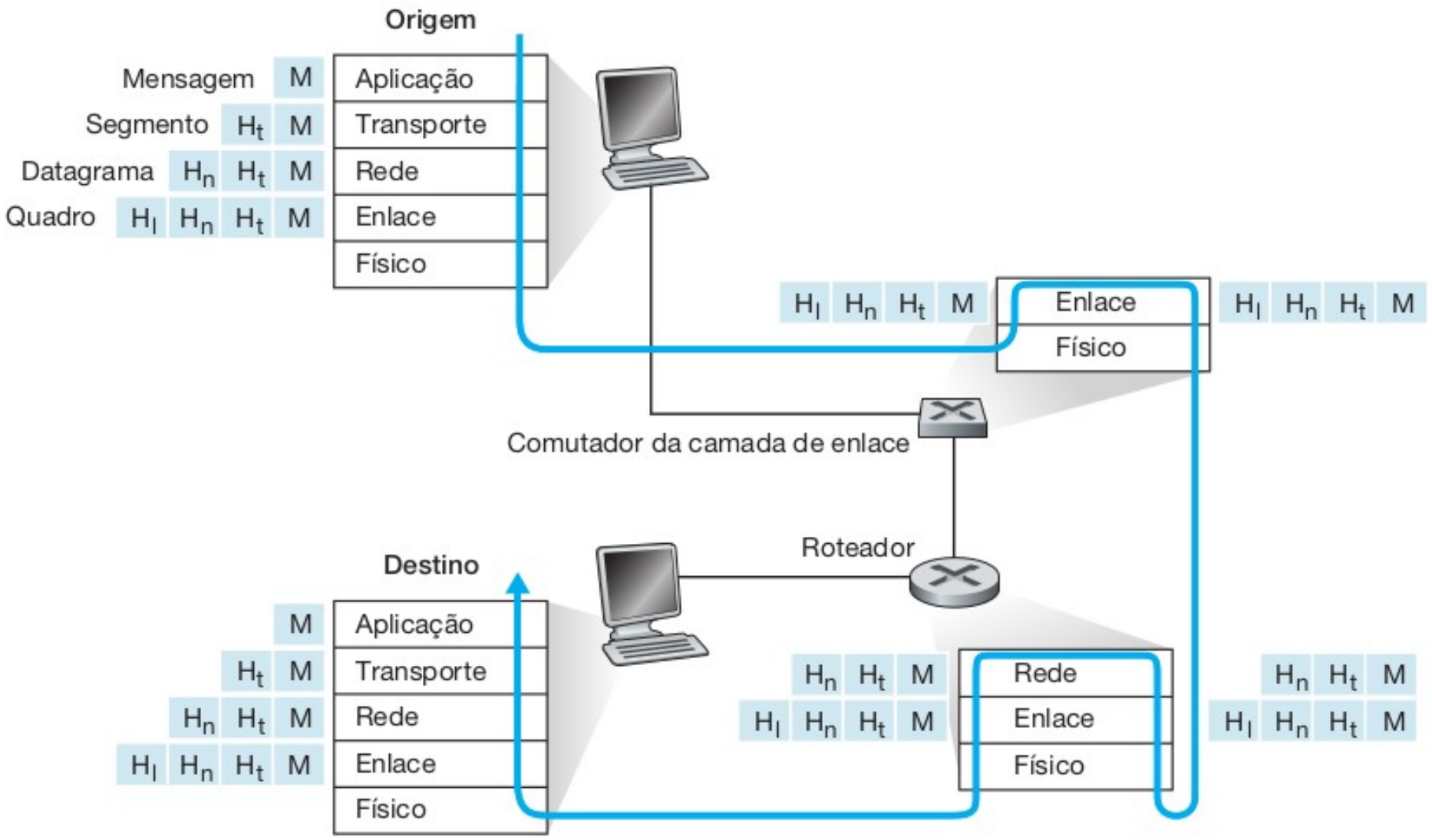
- Testar conectividade entre duas máquinas
  - Comando **ping**
    - Envia uma mensagem (**eco request**) para um dispositivo e recebe um resposta (**eco response**) caso o dispositivo receba a mensagem;
    - Utiliza o protocolo **ICMP** (Internet Control Message Protocol);
    - Trabalha na camada de rede (3) do modelo híbrido.
- Ex:

```
➜  ~ ping www.google.com
PING www.google.com (142.250.78.228) 56(84) bytes of data.
64 bytes de rio07s02-in-f4.1e100.net (142.250.78.228): icmp_seq=1 ttl=111 tempo=68.6 ms
64 bytes de rio07s02-in-f4.1e100.net (142.250.78.228): icmp_seq=2 ttl=111 tempo=68.8 ms
64 bytes de rio07s02-in-f4.1e100.net (142.250.78.228): icmp_seq=3 ttl=111 tempo=68.7 ms
64 bytes de rio07s02-in-f4.1e100.net (142.250.78.228): icmp_seq=4 ttl=111 tempo=68.7 ms
^C
--- www.google.com estatísticas de ping ---
4 pacotes transmitidos, 4 recebidos, 0% perda de pacote, tempo 3002ms
rtt min/avg/max/mdev = 68.577/68.697/68.784/0.076 ms
```


icmp
No. Time Source Destination Protocol Length Info
17 6.353226880 192.168.0.6 142.251.129.68 ICMP 98 Echo (ping) request id=0x0007, seq=1/256, ttl=64 (reply in 1…
18 6.422214843 142.251.129.68 192.168.0.6 ICMP 98 Echo (ping) reply id=0x0007, seq=1/256, ttl=111 (request i…
22 7.354352504 192.168.0.6 142.251.129.68 ICMP 98 Echo (ping) request id=0x0007, seq=2/512, ttl=64 (reply in 2…
23 7.423121383 142.251.129.68 192.168.0.6 ICMP 98 Echo (ping) reply id=0x0007, seq=2/512, ttl=111 (request i…
38 8.356261049 192.168.0.6 142.251.129.68 ICMP 98 Echo (ping) request id=0x0007, seq=3/768, ttl=64 (reply in 3…
39 8.425054508 142.251.129.68 192.168.0.6 ICMP 98 Echo (ping) reply id=0x0007, seq=3/768, ttl=111 (request i…
44 9.358207180 192.168.0.6 142.251.129.68 ICMP 98 Echo (ping) request id=0x0007, seq=4/1024, ttl=64 (reply in …
46 9.426868228 142.251.129.68 192.168.0.6 ICMP 98 Echo (ping) reply id=0x0007, seq=4/1024, ttl=111 (request …
49 10.360030207 192.168.0.6 142.251.129.68 ICMP 98 Echo (ping) request id=0x0007, seq=5/1280, ttl=64 (reply in …
50 10.428803321 142.251.129.68 192.168.0.6 ICMP 98 Echo (ping) reply id=0x0007, seq=5/1280, ttl=111 (request …
Frame 17: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface eno1, id 0
Ethernet II, Src: fc:34:97:7a:e2:fa (fc:34:97:7a:e2:fa), Dst: Tp-LinkT_e0:a4:cc (28:ee:52:e0:a4:cc)
Internet Protocol Version 4, Src: 192.168.0.6, Dst: 142.251.129.68
Internet Control Message Protocol
Type: 8 (Echo (ping) request)
Code: 0
Checksum: 0xdfcd [correct]
[Checksum Status: Good]
Identifier (BE): 7 (0x0007)
Identifier (LE): 1792 (0x0700)
Sequence number (BE): 1 (0x0001)
Sequence number (LE): 256 (0x0100)
[Response frame: 18]
Timestamp from icmp data: Mar 1, 2022 18:47:34.000000000 -03
[Timestamp from icmp data (relative): 0.876736060 seconds]
Data (48 bytes)
Data: b7600d000000000101112131415161718191a1b1c1d1e1f…
[Length: 48]
0000 28 ee 52 e0 a4 cc fc 34 97 7a e2 fa 08 00 45 00 (·R····4 ·z····E·
0010 00 54 22 fa 40 00 40 01 46 c1 c0 a8 00 06 8e fb ·T"·@·@· F·····
0020 81 44 08 00 df cd 00 07 00 01 76 94 1e 62 00 00 ·D····· ··v··b··
0030 00 00 b7 60 0d 00 00 00 00 00 10 11 12 13 14 15 ···`···· ····
0040 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25 ····· ·· !"#$%
0050 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35 &'()*+,- ./012345
0060 36 37 67
eduardo@pc:~
sudo wireshark
eduardo@pc:~
→ ~ ping www.google.com
PING www.google.com (142.251.129.68) 56(84) bytes of data.
64 bytes de rio07s07-in-f4.1e100.net (142.251.129.68): icmp_seq=1 ttl=111 tempo=69.0 ms
64 bytes de rio07s07-in-f4.1e100.net (142.251.129.68): icmp_seq=2 ttl=111 tempo=68.8 ms
64 bytes de rio07s07-in-f4.1e100.net (142.251.129.68): icmp_seq=3 ttl=111 tempo=68.8 ms
64 bytes de rio07s07-in-f4.1e100.net (142.251.129.68): icmp_seq=4 ttl=111 tempo=68.7 ms
64 bytes de rio07s07-in-f4.1e100.net (142.251.129.68): icmp_seq=5 ttl=111 tempo=68.8 ms
^C
--- www.google.com estatísticas de ping ---
5 pacotes transmitidos, 5 recebidos, 0% perda de pacote, tempo 4007ms
rtt min/avg/max/mdev = 68.701/68.829/68.997/0.095 ms
→ ~

# Análise




Origem
Mensagem M
Segmento $H_t$ M
Datagrama $H_n$ $H_t$ M
Quadro $H_l$ $H_n$ $H_t$ M
Aplicação
Transporte
Rede
Enlace
Físico
$H_l$ $H_n$ $H_t$ M
Enlace
Físico
$H_l$ $H_n$ $H_t$ M
Comutador da camada de enlace
Roteador
Destino
M
$H_t$ M
$H_n$ $H_t$ M
$H_l$ $H_n$ $H_t$ M
Aplicação
Transporte
Rede
Enlace
Físico
$H_n$ $H_t$ M
$H_l$ $H_n$ $H_t$ M
Rede
Enlace
Físico
$H_n$ $H_t$ M
$H_l$ $H_n$ $H_t$ M

# Análise



- **Camada física**: sinal eletromagnético (interface eno1)
- **Camada de enlace**: Protocolo ethernet
  - Endereço físico de origem (MAC): fc:34:97:7a:e2:fa
  - Endereço físico de destino (MAC): 28:ee:52:e0:a4:cc
- **Camada de rede**: Protocolo ICMP
  - Endereço lógico de origem (IP): 192.168.0.6
  - Endereço lógico de destino (IP): 142.251.129.68

# Análise



- **Camada física**: sinal eletromagnético (interface eno1)
- **Camada de enlace**: Protocolo ethernet
  - Endereço físico de origem (MAC): fc:34:97:7a:e2:fa
  - Endereço físico de destino (MAC): 28:ee:52:e0:a4:cc
- **Camada de rede**: Protocolo ICMP
  - Endereço lógico de origem (IP): 192.168.0.6
  - Endereço lógico de destino (IP): 142.251.129.68

**Como obter o endereço físico de destino?**

- Protocolo **ARP**

**Como obter o endereço lógico de destino usando o nome?**

- Protocolo **DNS**

# Análise



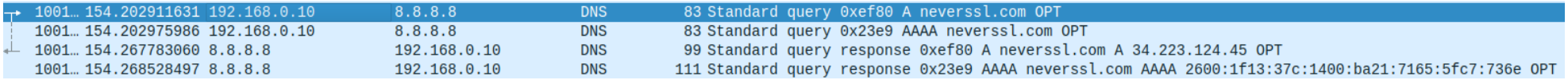
- Acessando uma página web e analisar o resultado do wireshark

- Acessar a página http://neverssl.com
  - Como o endereço neverssl.com é traduzido para seu endereço lógico?
  - Como descobrir o seu endereço lógico?
  - Qual protocolo usado para requisitar a páginas neverssl.com

# Análise



- Tradução de nome em IP (DNS)



| | | | | | | |
|---|---|---|---|---|---|---|
| 1001… | 154.202911631 | 192.168.0.10 | 8.8.8.8 | DNS | 83 | Standard query 0xef80 A neverssl.com OPT |
| 1001… | 154.202975986 | 192.168.0.10 | 8.8.8.8 | DNS | 83 | Standard query 0x23e9 AAAA neverssl.com OPT |
| 1001… | 154.267783060 | 8.8.8.8 | 192.168.0.10 | DNS | 99 | Standard query response 0xef80 A neverssl.com A 34.223.124.45 OPT |
| 1001… | 154.268528497 | 8.8.8.8 | 192.168.0.10 | DNS | 111 | Standard query response 0x23e9 AAAA neverssl.com AAAA 2600:1f13:37c:1400:ba21:7165:5fc7:736e OPT |

- Caso não apareça no wireshark o tráfego DNS, pode ser que o sistema esteja usando o **cache DNS**.
  - Como limpar o cache DNS e verificar a requisição DNS:
    - Linux (Ubuntu):

```
$ resolvectl flush-caches
```

    - Windows:

```
> ipconfig /flushdns
```

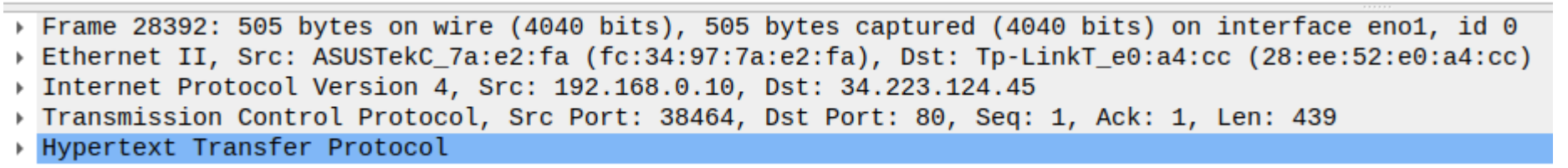
https://themeim.com/how-to-clear-dns-cache-on-windows-10/

# Análise



- Requisição de página Web

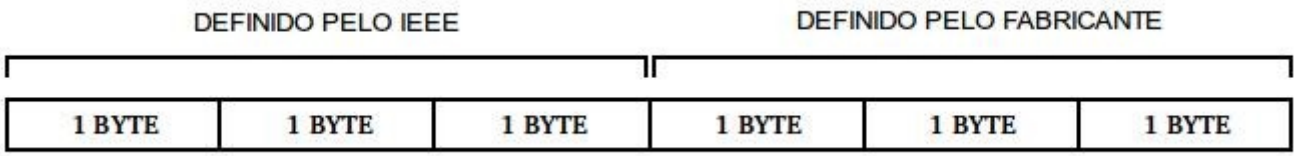
| | | | | | | |
|---|---|---|---|---|---|---|
| 28263 | 16.314383476 | 192.168.0.10 | 34.223.124.45 | TCP | 74 | 38464 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2101462406 TSecr=0 WS=128 |
| 28390 | 16.492180840 | 34.223.124.45 | 192.168.0.10 | TCP | 74 | 80 → 38464 [SYN, ACK] Seq=0 Ack=1 Win=26847 Len=0 MSS=1460 SACK_PERM=1 TSval=1158049544 TSecr |
| 28391 | 16.492229116 | 192.168.0.10 | 34.223.124.45 | TCP | 66 | 38464 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=2101462584 TSecr=1158049544 |
| 28392 | 16.492378033 | 192.168.0.10 | 34.223.124.45 | HTTP | 505 | GET /online HTTP/1.1 |
| 28395 | 16.669649633 | 34.223.124.45 | 192.168.0.10 | TCP | 66 | 80 → 38464 [ACK] Seq=1 Ack=440 Win=28032 Len=0 TSval=1158049722 TSecr=2101462584 |
| 28396 | 16.670199001 | 34.223.124.45 | 192.168.0.10 | HTTP | 607 | HTTP/1.1 301 Moved Permanently (text/html) |
| 28397 | 16.670270902 | 192.168.0.10 | 34.223.124.45 | TCP | 66 | 38464 → 80 [ACK] Seq=440 Ack=542 Win=64128 Len=0 TSval=2101462762 TSecr=1158049722 |
| 28398 | 16.676631617 | 192.168.0.10 | 34.223.124.45 | HTTP | 506 | GET /online/ HTTP/1.1 |
| 28400 | 16.854743219 | 34.223.124.45 | 192.168.0.10 | HTTP | 1585 | HTTP/1.1 200 OK (text/html) |
| 28401 | 16.854797303 | 192.168.0.10 | 34.223.124.45 | TCP | 66 | 38464 → 80 [ACK] Seq=880 Ack=2061 Win=64128 Len=0 TSval=2101462946 TSecr=1158049906 |
| 28402 | 16.953762452 | 192.168.0.10 | 34.223.124.45 | HTTP | 450 | GET /favicon.ico HTTP/1.1 |
| 28403 | 17.131267331 | 34.223.124.45 | 192.168.0.10 | HTTP | 482 | HTTP/1.1 200 OK (PNG) |
| 28404 | 17.177768939 | 192.168.0.10 | 34.223.124.45 | TCP | 66 | 38464 → 80 [ACK] Seq=1264 Ack=2477 Win=64128 Len=0 TSval=2101463269 TSecr=1158050183 |
| 29771 | 21.175229445 | 192.168.0.10 | 34.223.124.45 | TCP | 66 | 33656 → 80 [FIN, ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=2101467267 TSecr=1158049218 |
| 29772 | 21.175272135 | 192.168.0.10 | 34.223.124.45 | TCP | 66 | 33642 → 80 [FIN, ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=2101467267 TSecr=1158049193 |
| 30189 | 21.352369069 | 34.223.124.45 | 192.168.0.10 | TCP | 66 | 80 → 33642 [FIN, ACK] Seq=1 Ack=2 Win=26880 Len=0 TSval=1158054405 TSecr=2101467267 |
| 30190 | 21.352404545 | 192.168.0.10 | 34.223.124.45 | TCP | 66 | 33642 → 80 [ACK] Seq=2 Ack=2 Win=64256 Len=0 TSval=2101467444 TSecr=1158054405 |
| 30191 | 21.353322815 | 34.223.124.45 | 192.168.0.10 | TCP | 66 | 80 → 33656 [FIN, ACK] Seq=1 Ack=2 Win=26880 Len=0 TSval=1158054407 TSecr=2101467267 |
| 30192 | 21.353365761 | 192.168.0.10 | 34.223.124.45 | TCP | 66 | 33656 → 80 [ACK] Seq=2 Ack=2 Win=64256 Len=0 TSval=2101467445 TSecr=1158054407 |
| 31986 | 22.134374728 | 34.223.124.45 | 192.168.0.10 | TCP | 66 | 80 → 38464 [FIN, ACK] Seq=2477 Ack=1264 Win=30080 Len=0 TSval=1158055186 TSecr=2101463269 |
| 31987 | 22.134625761 | 192.168.0.10 | 34.223.124.45 | TCP | 66 | 38464 → 80 [FIN, ACK] Seq=1264 Ack=2478 Win=64128 Len=0 TSval=2101468226 TSecr=1158055186 |
| 32346 | 22.312528907 | 34.223.124.45 | 192.168.0.10 | TCP | 66 | 80 → 38464 [ACK] Seq=2478 Ack=1265 Win=30080 Len=0 TSval=1158055363 TSecr=2101468226 |



# Endereço MAC



- Identificador único de uma interface de rede (NIC);
- Usado em muitos protocolos de enlace (Ethernet, WiFi, Bluetooth);
- Endereço físico;
- Identificado por 6 grupos (bytes) de dois números hexadecimais (0-F):
  - **fc:34:97:7a:e2:fa**



# Endereço MAC



- Verificar o endereço MAC de um enlace

  Windows

```
> ipconfig /all
```

  Linux

```
> ip l l
```

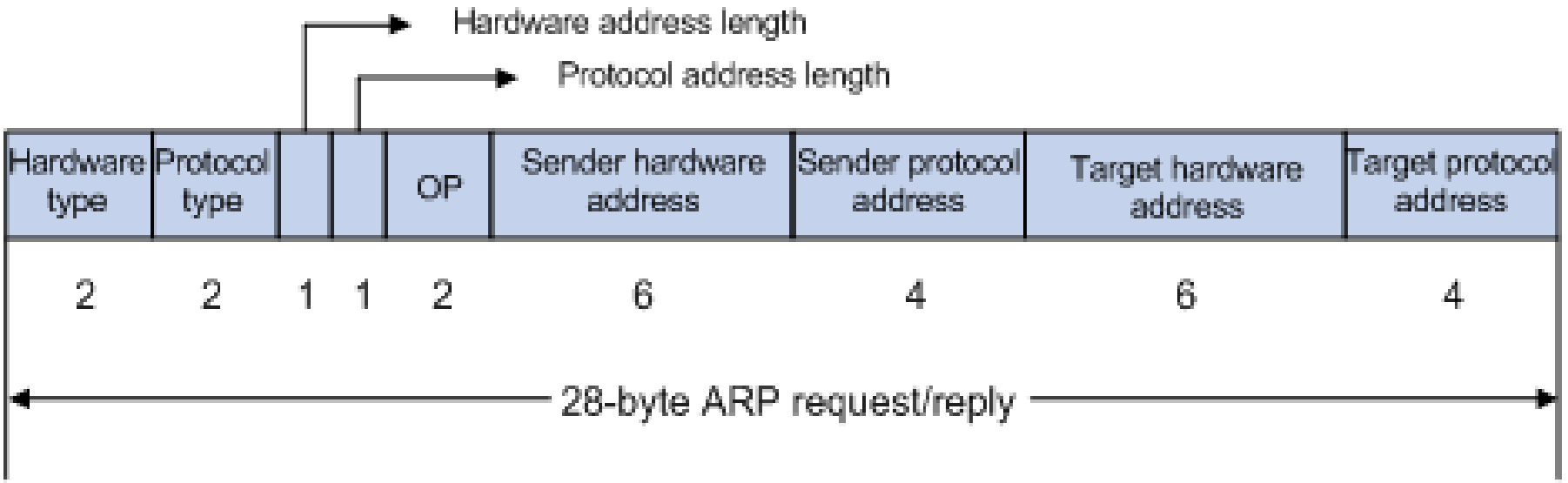
# Protocolo ARP



- *Address Resolution Protocol*
- Protocolo utilizado para resolver dinamicamente um endereço lógico (IP) em um endereço físico(MAC).
- Utiliza quadros de rede para se comunicar.

# Formato do Quadro ARP

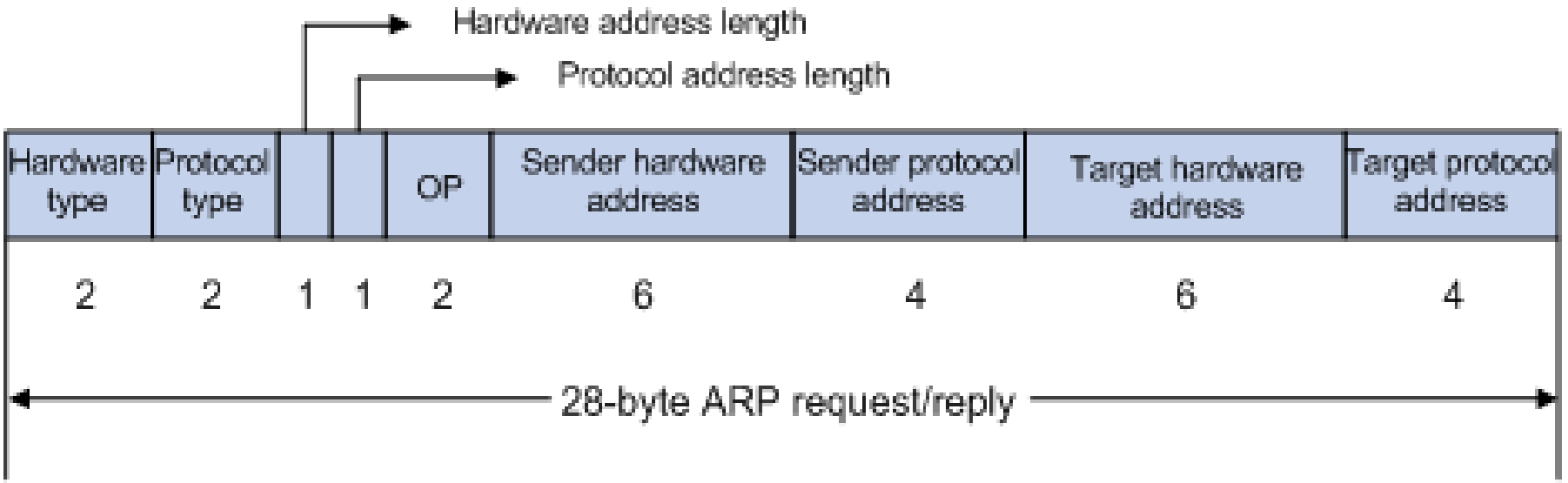





- **Hardware Type**: Identifica o tipo de enlace. Valor 1 para Ethernet.
- **Protocol Type**: Identifica qual será o protocolo de camada de rede. Valor 0x08 para IP.
- **Hardware Length**: Identifica o tamanho do endereço de hardware em grupos de 8 bits. MAC padrão ==6.

# Formato do Quadro ARP

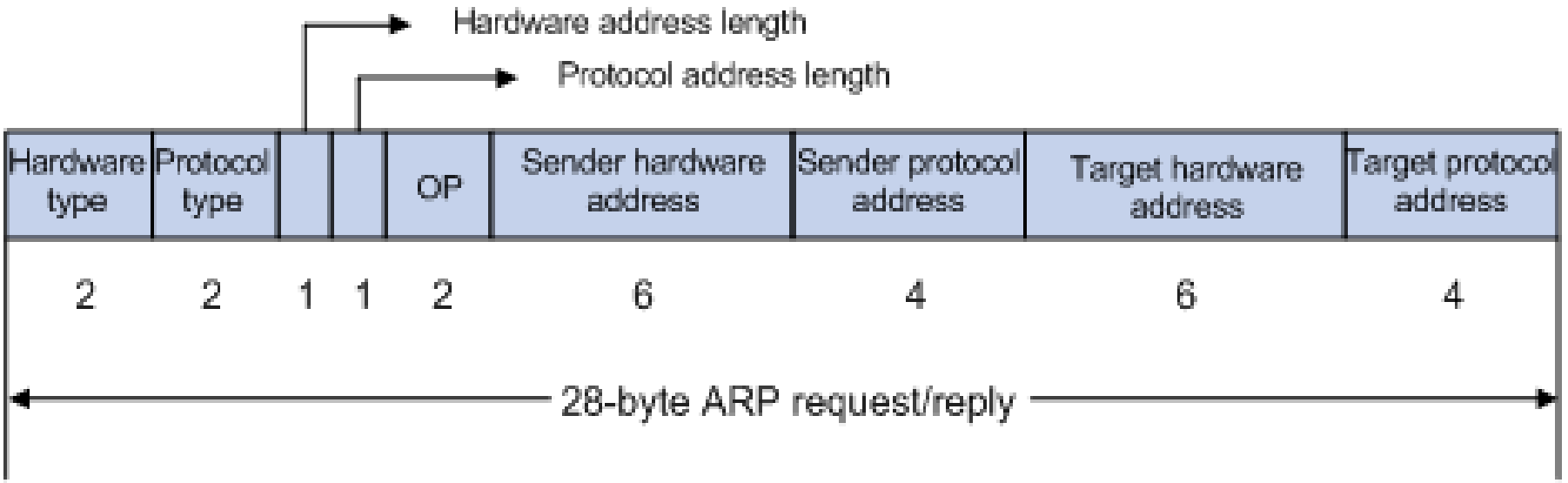





- **Protocol length**: Identifica o tamanho do endereço lógico em grupos de 8 bits. IPv4 == 4.
- **Operation**: Identifica a operação que está sendo realizada. Valores: 1 ARP Request; 2 ARP Reply.
- **Sender Hardware Address**: Endereço físico do remetente (MAC de origem).

# Formato do Quadro ARP






- **Sender Protocol Address**: Endereço lógico do remetente (IPv4 de origem).
- **Target Hardware Address**: Endereço físico do remetente (MAC de destino).
- **Target Protocol Address**: Endereço lógico do remetente (IPv4 de destino).

# ARP Cache



- Verificar a tabela de cache ARP

  Windows

```
> arp -a
```

  Linux

```
$ cat /proc/net/arp
```

```
$ arp -a
```

# ARP Cache



- Remover um item da tabela de cache ARP

  Windows

```
> arp -d <endereco_IP>
```

  Linux

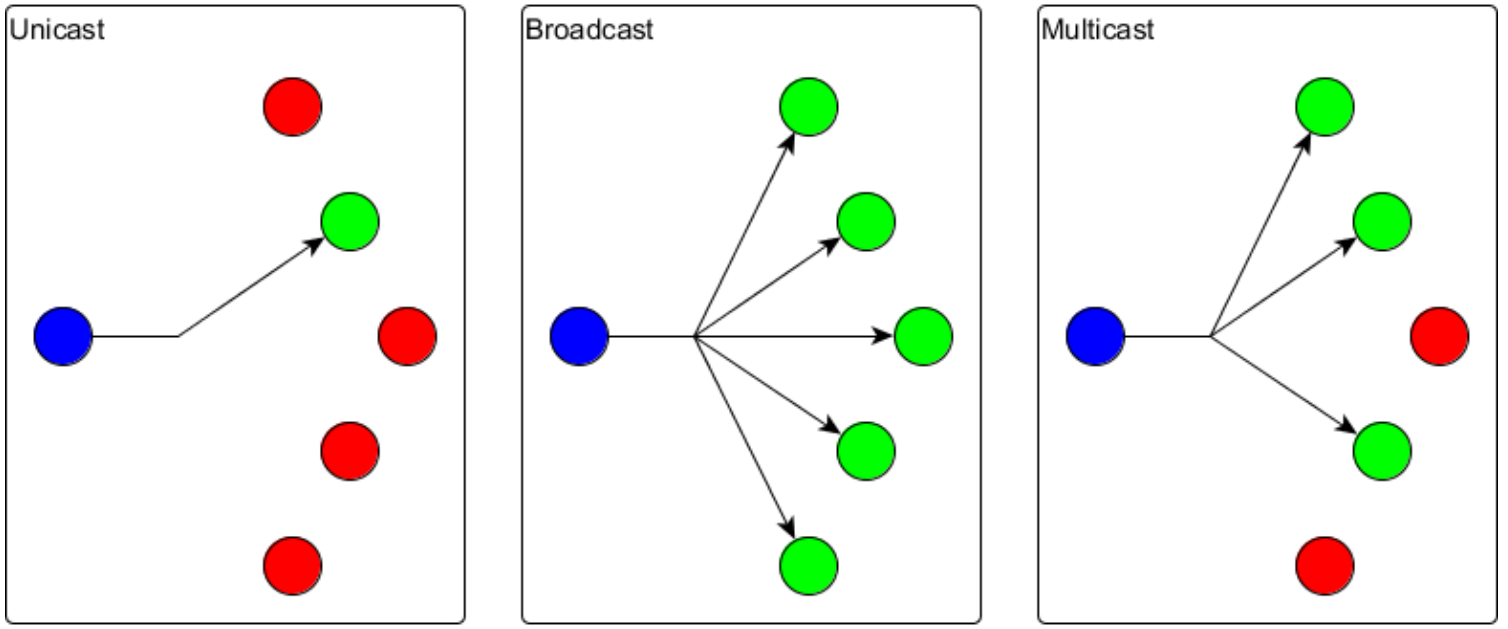
```
$ echo “”> cat /proc/net/arp
```

```
$ arp -d <endereco_IP>
```

# Tipos de Endereço



- **Unicast**: Endereçamento realizado a um único destino;
- **Multicast**: Endereçamento realizado a um grupo;
- **Broadcast**: Endereçamento realizado a todos.



# Endereço MAC



- Como **verificar** o endereço físico da minha placa de rede?
  - » Windows: ipconfig /all
  - » Linux: ip link list (ou: ip l l)

```
→ ~ ip l l
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eno1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT group default qlen 1000
    link/ether fc:34:97:7a:e2:fa brd ff:ff:ff:ff:ff:ff
    altname enp0s31f6
3: virbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN mode DEFAULT group default qlen 1000
    link/ether 52:54:00:12:4a:81 brd ff:ff:ff:ff:ff:ff
4: virbr0-nic: <BROADCAST,MULTICAST> mtu 1500 qdisc fq_codel master virbr0 state DOWN mode DEFAULT group default qlen 1000
    link/ether 52:54:00:12:4a:81 brd ff:ff:ff:ff:ff:ff
5: lxcbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN mode DEFAULT group default qlen 1000
    link/ether 00:16:3e:00:00:00 brd ff:ff:ff:ff:ff:ff
```

# Análise



- O que significa encaminhar **echo request** pelo nome e não receber **echo reply**?
- O que significa encaminhar **echo request** pelo endereço lógico e não receber **echo reply**?
- Como verificar o possível ponto de falha de um **echo request** usando o comando ping?
- O que é o protocolo ICMP?
- O que é o protocolo ARP?

# Dúvidas?