

# **TP2: Protocolo IP**

Diogo Afonso Costa, Daniel Maia, and Vitor Castro

University of Minho, Department of Informatics, 4710-057 Braga, Portugal  
e-mail: {a78034,a77531,a77870}@alunos.uminho.pt

**Abstract.** Resumo...

## **1 Introdução**

## **2 Parte I - Datagramas e Fragmentação**

### **2.1 Exercício 1.b.**

**Questão**

**Resposta**

**Realização**

### **2.2 Exercício 1.c.**

**Questão**

**Resposta**

**Realização**

### **2.3 Exercício 1.d.**

**Questão**

**Resposta**

**Realização**

### **2.4 Exercício 2.a.**

**Questão**

Qual é o endereço IP da interface ativa do seu computador?

## Resposta

O endereço IP é 192.168.100.216.

## Realização

43	3.903920123	192.168.100.216	192.168.100.216	UNDEF	519	Standard query response 0x7472 A marco.uminho.pt A 192.168.9.248 NS unsa
44	3.903910826	192.168.100.216	193.136.9.240	ICMP	74	Echo (ping) request id=0x3eal, seq=1/256, ttl=1 (no response found!)
45	3.903938117	192.168.100.216	193.136.9.240	ICMP	74	Echo (ping) request id=0x3eal, seq=2/512, ttl=1 (no response found!)
46	3.903947143	192.168.100.216	193.136.9.240	ICMP	74	Echo (ping) request id=0x3eal, seq=3/768, ttl=1 (no response found!)
47	3.903956246	192.168.100.216	193.136.9.240	ICMP	74	Echo (ping) request id=0x3eal, seq=4/1024, ttl=2 (no response found!)
48	3.903964672	192.168.100.216	193.136.9.240	ICMP	74	Echo (ping) request id=0x3eal, seq=5/1280, ttl=2 (no response found!)
49	3.903970621	192.168.100.216	193.136.9.240	ICMP	74	Echo (ping) request id=0x3eal, seq=6/1536, ttl=2 (no response found!)
50	3.903975159	192.168.100.216	193.136.9.240	ICMP	74	Echo (ping) request id=0x3eal, seq=7/1792, ttl=3 (reply in 63)
51	3.903982289	192.168.100.216	193.136.9.240	ICMP	74	Echo (ping) request id=0x3eal, seq=8/2048, ttl=3 (reply in 64)
52	3.903988913	192.168.100.216	193.136.9.240	ICMP	74	Echo (ping) request id=0x3eal, seq=9/2304, ttl=3 (reply in 66)
53	3.903994876	192.168.100.216	193.136.9.240	ICMP	74	Echo (ping) request id=0x3eal, seq=10/2560, ttl=4 (reply in 67)
54	3.904000785	192.168.100.216	193.136.9.240	ICMP	74	Echo (ping) request id=0x3eal, seq=11/2816, ttl=4 (reply in 68)
55	3.904009311	192.168.100.216	193.136.9.240	ICMP	74	Echo (ping) request id=0x3eal, seq=12/3072, ttl=4 (reply in 69)
56	3.904015173	192.168.100.216	193.136.9.240	ICMP	74	Echo (ping) request id=0x3eal, seq=13/3328, ttl=5 (reply in 70)
57	3.904020348	192.168.100.216	193.136.9.240	ICMP	74	Echo (ping) request id=0x3eal, seq=14/3584, ttl=5 (reply in 71)
58	3.904023767	192.168.100.216	193.136.9.240	ICMP	74	Echo (ping) request id=0x3eal, seq=15/3840, ttl=5 (reply in 72)
59	3.904032851	192.168.100.216	193.136.9.240	ICMP	74	Echo (ping) request id=0x3eal, seq=16/4096, ttl=6 (reply in 73)
60	3.904260758	192.168.100.216	192.168.100.216	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)

Fig. 1: Identificação do endereço IP.

## 2.5 Exercício 2.b.

### Questão

Qual é o valor do campo protocolo? O que identifica?

### Resposta

O campo protocolo tem o valor "ICMP (1)". ICMP significa *Internet Control Message Protocol*. Este é utilizado para reportar erros no processamento de datagramas. Efetivamente, dentro dos possíveis erros temos, *destination unreachable* (quando o datagrama não consegue alcançar o destino), *time exceeded message* (quando um *gateway* processa um datagrama e descobre que o *TTL* é zero e tem que descartar o datagrama e consequentemente notificar o *host*), *echo request/reply* (quando são enviadas mensagens para funções de teste e controle da rede (*request*), caso a máquina esteja ligada responde com um *reply*) [1] [2]. Como as mensagens ICMP encontram-se ao nível de rede, estas são também elas encapsuladas em datagramas IP que, consequentemente, usam o protocolo IP.

Assim sendo, analisando a primeira mensagem ICMP, nomeadamente no separador do *Internet Protocol Version 4*, percebemos que se trata de uma mensagem que vem num protocolo *ICMP*. Além disso, é possível concluir que se trata de uma mensagem de *echo request*, se observarmos o campo *Type* no separador *Internet Control Message Protocol*. Desta forma, pode-se concluir que o computador usado para a resolução deste trabalho está a tentar perceber se consegue estabelecer uma ligação com o *host* marco.uminho.pt e para isso usa mensagens *ICMP* do tipo *echo request*.

### Realização

```

Internet Protocol Version 4, Src: 192.168.100.216, Dst: 193.136.9.240
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    ► Differentiated Services Field: 0x00 (DSCH: CS0, ECN: Not-ECT)
    Total Length: 60
    Identification: 0xb224 (45604)
    ► Flags: 0x00
    Fragment offset: 0
    Time to live: 1
    Protocol: ICMP (1)
        Header checksum: 0x16a4 [validation disabled]
        [Header checksum status: Unverified]
        Source: 192.168.100.216
        Destination: 193.136.9.240
        [Source GeoIP: Unknown]
        ► [Destination GeoIP: Portugal]
    Internet Control Message Protocol
        Type: 8 (Echo (ping) request)
            Code: 0
            Checksum: 0x43d8 [correct]
            [Checksum Status: Good]
            Identifier (BE): 16033 (0x3e1)
            Identifier (LE): 41278 (0xa13e)
            Sequence number (BE): 1 (0x0001)
            Sequence number (LE): 256 (0x0100)
        ► [No response seen]
        Data (32 bytes)

```

Fig. 2: Identificação do campo *Protocol*.

## 2.6 Exercício 2.c.

### Questão

Quantos bytes tem o cabeçalho IP(v4)? Quantos bytes tem o campo de dados (payload) do datagrama? Como se calcula o tamanho do payload?

### Resposta

O cabeçalho IPv4 tem 20 bytes.

O campo de dados (payload) do datagrama tem 40 bytes.

O cálculo do payload é feito retirando o tamanho do cabeçalho ao tamanho total do datagrama (60 bytes). Desta forma, basta fazer  $60 - 20 = 40\text{bytes}$ .

### Realização

```

Internet Protocol Version 4, Src: 192.168.100.216, Dst: 193.136.9.240
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    ► Differentiated Services Field: 0x00 (DSCH: CS0, ECN: Not-ECT)
    Total Length: 60
    Identification: 0xb224 (45604)
    ► Flags: 0x00
    Fragment offset: 0
    Time to live: 1
    Protocol: ICMP (1)
        Header checksum: 0x16a4 [validation disabled]
        [Header checksum status: Unverified]
        Source: 192.168.100.216
        Destination: 193.136.9.240
        [Source GeoIP: Unknown]
        ► [Destination GeoIP: Portugal]
    Internet Control Message Protocol

```

Fig. 3: Identificação do tamanho do *header* e do *payload*.

## 2.7 Exercício 2.d.

### Questão

O datagrama IP foi fragmentado? Justifique.

### Resposta

A fragmentação acontece quando o tamanho total do datagrama excede o *MTU* disponível. Tendo em conta que por defeito o *traceroute* usa 60 bytes por datagrama e tem-se um *MTU* disponível de 1500 bytes, podemos conjeturar que não haverá fragmentação.

A verificação se um datagrama foi ou não fragmentado é feita com base em dois valores, o *fragment offset* (indica o *offset* em que o datagrama atual encaixa no datagrama original) e a *flag more fragments* (indica se existe mais fragmentos). Neste datagrama em específico o *fragment offset* = 0 e a *flag more fragments* = 0. Desta forma, tendo em conta o *fragment offset*, sabe-se que se o datagrama foi fragmentado então ele é necessariamente o primeiro. Além disso, se analisarmos a *flag more fragments* concluímos que para além do datagrama atual não existe mais nenhum associado a este.

Assim sendo, conjugando a informação dos dois parâmetros percebe-se que se o datagrama é o primeiro e não existe mais nenhum associado, então este é único e não foi fragmentado.

### Realização

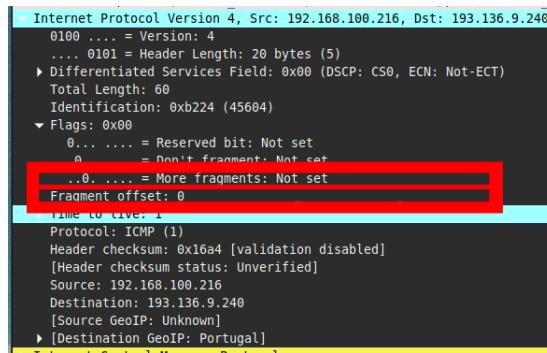


Fig. 4: Fragmentação.

## 2.8 Exercício 2.e.

### Questão

Ordene os pacotes capturados de acordo com o endereço IP fonte (e.g., selecionando o cabeçalho da coluna Source), e analise a sequência de tráfego ICMP gerado a partir do endereço IP atribuído à sua máquina. Para a sequência de mensagens ICMP enviadas pelo seu computador, indique que campos do cabeçalho IP variam de pacote para pacote.

## Resposta

Os campos que vêm os seus valores alterados correspondem à *identification*, *header checksum* e *time to live (TTL)*.

A *identificação* muda pois este campo identifica unicamente cada datagrama e visto que estes são sempre diferentes então o campo também o será.

O *header checksum* permite verificar que determinado header foi ou não corrompido. Desta forma o *checksum* identifica um determinado *header* num determinado estado. Assim sendo, o *checksum* muda pois este campo utiliza no seu algoritmo todas as palavras de 16 bits do *header* [1]. Efetivamente, sabendo que o *header*, propriamente dito, muda de datagrama para datagrama então o seu *checksum* também vai mudar.

O *TTL* muda pois a máquina que está a ser usada está a tentar contactar o *host* marco.uminho.pt. Começa por tentar com *TTL* = 1 e o pacote é descartado. Desta forma, vai aumentando o *TTL* até conseguir chegar ao *host* pretendido.

## Realização

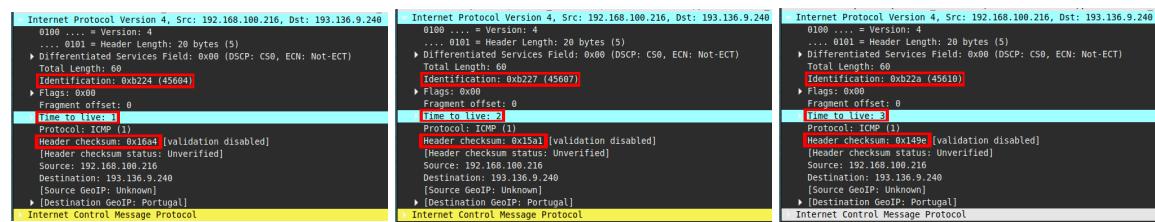


Fig. 5: Campos que mudam.

## 2.9 Exercício 2.f.

### Questão

Observa algum padrão nos valores do campo de Identificação do datagrama IP e TTL?

## Resposta

O campo da indentificação corresponde a um valor que é incrementado e que identifica unicamente o datagrama em questão. Por exemplo, se o primeiro datagrama tiver *Identification* 0xb224 (45604), então o datagrama seguinte terá o valor 0xb225 (45605).

O TTL corresponde a uma variável que vai sendo decrementada sempre que é interse-tada por um *router*. Visto que na primeira mensagem o TTL é 1, o datagrama é descartado imediatamente no primeiro *router*. Deste modo, é enviado, de seguida, um novo datagrama com TTL a 2 com a esperança de que este chegue desta vez ao destino. Caso não chegue, então no próximo datagrama o TTL será aumentado, e assim sucessivamente, até chegar ao destino.

## Realização

A relação entre *TTL* pode ser observada na figura 8.

<pre> Internet Protocol Version 4, Src: 192.168.100.216, Dst: 193.136.9.248   0100 .... = Version: 4   .... 0101 = Header Length: 20 bytes (5)   ► Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)   Total Length: 60   Identification: 0xb224 (45604)   ► Flags: 0x00   Fragment offset: 0   Time to live: 1   Protocol: ICMP (1)   Header checksum: 0x16a4 [validation disabled]   [Header checksum status: Unverified]   Source: 192.168.100.216   Destination: 193.136.9.248   [Source GeoIP: Unknown]   ► [Destination GeoIP: Portugal]   Internet Control Message Protocol </pre>	<pre> Internet Protocol Version 4, Src: 192.168.100.216, Dst: 193.136.9.248   0100 .... = Version: 4   .... 0101 = Header Length: 20 bytes (5)   ► Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)   Total Length: 60   Identification: 0xb225 (45605)   ► Flags: 0x00   Fragment offset: 0   Time to live: 1   Protocol: ICMP (1)   Header checksum: 0x16a3 [validation disabled]   [Header checksum status: Unverified]   Source: 192.168.100.216   Destination: 193.136.9.248   [Source GeoIP: Unknown]   ► [Destination GeoIP: Portugal]   Internet Control Message Protocol </pre>
---	---

Fig. 6: Identificação.

## 2.10 Exercício 2.g.

### Questão

Ordene o tráfego capturado por endereço destino e encontre a série de respostas ICMP TTL exceeded enviadas ao seu computador. Qual é o valor do campo TTL? Esse valor permanece constante para todas as mensagens de resposta ICMP TTL exceeded enviados ao seu host? Porquê?

### Resposta

O IP 192.168.100.254 referencia a interface do primeiro router de acesso (visto ter o três primeiros campos iguais ao da máquina em que os testes estão a ser feitos e o último utiliza um número convencionado para ser utilizado para identificar a interface IP do router dentro da rede 192.168.100, na qual a máquina de testes se encontra).

Desta forma, quando analisamos os datagramas do 192.168.100.254 percebemos que estes têm todos o *TTL* = 64. O *TTL* toma um valor exageradamente elevado, devido ao desconhecimento que este tem da distância a que o *host* de destino se encontra. Por defeito, este router quando não tem informação da distância a que o *host* de destino se encontra envia datagramas com *TTL* = 64 e por isso todas os seus datagramas tem *TTL* igual.

Além disso, é possível concluir que as três mensagens recebidas deste router com *ICMP TTL exceeded* são a resposta ao envio feito pela máquina de testes de três datagramas de *echo (ping) request* com *TTL* apenas de 1. Estes datagramas chegaram ao *router* de acesso e ficaram com o *TTL* a 0 e a exceção veio enviado de no datagrama *ICMP TTL exceeded*.

Após estes datagramas é possível identificar um segundo conjunto de datagramas desta vez da interface IP 193.136.19.254. Pelo mesmo raciocínio podemos assumir que o IP desta interface identifica um *router*. Este novo *router* envia datagramas com *TTL* = 254 (constante). Este valor é considerável pela mesma razão explicitada anteriormente. Estes datagramas são a resposta a 3 datagramas enviados pela máquina de testes com *TTL* = 2, o que nos leva a concluir que é o segundo router por qual o nosso datagrama teve de passar para chegar ao marco.uminho.pt.

### Realização

## 2.11 Exercício 3.a.

### Questão

Localize a primeira mensagem ICMP. Porque é que houve necessidade de fragmentar o pacote inicial?

<p>Internet Protocol Version 4, Src: 192.168.100.254, Dst: 192.168.100.216</p> <p>0100 .... = Version: 4</p> <p>.... 0101 = Header Length: 20 bytes (5)</p> <p>► Differentiated Services Field: 0xc0 (DSCP: CS6, ECN: Not-ECT)</p> <p>Total Length: 88</p> <p>Identification: 0xe4fc (61260)</p> <p>Flags: 0x00</p> <p>Fragment offset: 0</p> <p><b>TTL: 64</b></p> <p>Protocol: ICMP [1]</p> <p>Header checksum: 0x3f71 [validation disabled]</p> <p>[Header checksum status: Unverified]</p> <p>Source: 192.168.100.254</p> <p>Destination: 192.168.100.216</p> <p>[Source GeoIP: Unknown]</p> <p>[Destination GeoIP: Unknown]</p>	<p>Internet Protocol Version 4, Src: 192.168.100.254, Dst: 192.168.100.216</p> <p>0100 .... = Version: 4</p> <p>.... 0101 = Header Length: 20 bytes (5)</p> <p>► Differentiated Services Field: 0xc0 (DSCP: CS6, ECN: Not-ECT)</p> <p>Total Length: 88</p> <p>Identification: 0xe4fd (61261)</p> <p>Flags: 0x00</p> <p>Fragment offset: 0</p> <p><b>TTL: 64</b></p> <p>Protocol: ICMP [1]</p> <p>Header checksum: 0x3f70 [validation disabled]</p> <p>[Header checksum status: Unverified]</p> <p>Source: 192.168.100.254</p> <p>Destination: 192.168.100.216</p> <p>[Source GeoIP: Unknown]</p> <p>[Destination GeoIP: Unknown]</p>	<p>Internet Control Message Protocol</p>
		<p>Internet Control Message Protocol</p>

Fig. 7: Router de acesso com  $TTL = 64$ .

<p>Internet Protocol Version 4, Src: 193.136.19.254, Dst: 192.168.100.216</p> <p>0100 .... = Version: 4</p> <p>.... 0101 = Header Length: 20 bytes (5)</p> <p>► Differentiated Services Field: 0x00 (DSCP: CS6, ECN: Not-ECT)</p> <p>Total Length: 56</p> <p>Identification: 0x1147 (4423)</p> <p>Flags: 0x00</p> <p>Fragment offset: 0</p> <p>Header checksum: 0xaefb [validation disabled]</p> <p>[Header checksum status: Unverified]</p> <p>Source: 193.136.19.254</p> <p>Destination: 192.168.100.216</p> <p>► [Source GeoIP: Portugal]</p> <p>[Destination GeoIP: Unknown]</p> <p>Internet Control Message Protocol</p>	<p>Internet Protocol Version 4, Src: 193.136.19.254, Dst: 192.168.100.216</p> <p>0100 .... = Version: 4</p> <p>.... 0101 = Header Length: 20 bytes (5)</p> <p>► Differentiated Services Field: 0x00 (DSCP: CS6, ECN: Not-ECT)</p> <p>Total Length: 56</p> <p>Identification: 0x1148 (4424)</p> <p>Flags: 0x00</p> <p>Fragment offset: 0</p> <p>Header checksum: 0xaefc [validation disabled]</p> <p>[Header checksum status: Unverified]</p> <p>Source: 193.136.19.254</p> <p>Destination: 192.168.100.216</p> <p>► [Source GeoIP: Portugal]</p> <p>[Destination GeoIP: Unknown]</p> <p>Internet Control Message Protocol</p>	<p>Internet Protocol Version 4, Src: 193.136.19.254, Dst: 192.168.100.216</p> <p>0100 .... = Version: 4</p> <p>.... 0101 = Header Length: 20 bytes (5)</p> <p>► Differentiated Services Field: 0x00 (DSCP: CS6, ECN: Not-ECT)</p> <p>Total Length: 56</p> <p>Identification: 0x1149 (4425)</p> <p>Flags: 0x00</p> <p>Fragment offset: 0</p> <p>Header checksum: 0xaefd [validation disabled]</p> <p>[Header checksum status: Unverified]</p> <p>Source: 193.136.19.254</p> <p>Destination: 192.168.100.216</p> <p>► [Source GeoIP: Portugal]</p> <p>[Destination GeoIP: Unknown]</p> <p>Internet Control Message Protocol</p>
---	---	---

Fig. 8: Segundo router com  $TTL = 254$

## Resposta

O pacote inicial tem um *total length* de 3033 bytes. O MTU (*Maximum Transmission Unit*) disponível é de apenas 1500 bytes e, como tal, o pacote inicial tem de ser fragmentado para que este possa ser transmitido na rede.

## Realização

```

N. Time Source Destination Protocol Length Info
11. 9.318608576 192.168.100.216 192.168.100.254 DNS 80 Standard query 0xa8c8a1 dd-debug.dropbox.com
12. 9.318608341 192.168.100.216 192.168.100.254 DNS 80 Standard query 0x8cd44 AAAA dd-debug.dropbox.com
13. 9.319254679 192.168.100.254 192.168.100.216 DNS 190 Standard query response 0x8cd44 AAAA dd-debug.dropbox.com CNAME block-debug.xd...
14. 9.649429727 192.168.100.216 192.168.100.254 DNS 75 Standard query 0x3dcfa1 Marco.unimho.pt
15. 9.649437934 192.168.100.216 192.168.100.254 DNS 75 Standard query 0x70d8 0x4000 AAAA marco.unimho.pt
16. 9.650723567 192.168.100.216 192.168.100.254 DNS 56 Standard query response 0x70d8 0x4000 AAAA marco.unimho.pt A 193.136.9.240 ID=105 unim...
17. 9.650723567 192.168.100.254 192.168.100.216 DNS 129 Standard query response 0x70d8 0x4000 AAAA marco.unimho.pt A 193.136.9.240 ID=105 unim...
18. 9.656893095 192.168.100.216 193.136.9.240 ICMP 32 1514 Fragmented IP protocol [proto=ICMP 1, offset=1480, ID=813] [Reassembled in #20]
19. 9.656888397 192.168.100.216 193.136.9.240 IPv4 1514 Fragmented IP protocol [proto=ICMP 1, offset=1480, ID=813] [Reassembled in #20]
* 20. 9.656890359 192.168.100.216 193.136.9.240 ICMP 87 Echo (ping) request 0x8d4af, seq=1/250, ttl=1 (no response found!)
21. 9.656894641 192.168.100.216 193.136.9.240 IPv4 1514 Fragmented IP protocol [proto=ICMP 1, offset=1480, ID=814] [Reassembled in #23]
22. 9.656897423 192.168.100.216 193.136.9.240 IPv4 1514 Fragmented IP protocol [proto=ICMP 1, offset=1480, ID=814] [Reassembled in #23]
23. 9.656896558 192.168.100.216 193.136.9.240 ICMP 87 Echo (ping) request 0x8d4af, seq=2/252, ttl=1 (no response found!)
24. 9.656898000 192.168.100.216 193.136.9.240 ICMP 1514 Fragmented IP protocol [proto=ICMP 1, offset=1480, ID=815] [Reassembled in #26]
25. 9.656898460 192.168.100.216 193.136.9.240 ICMP 1514 Fragmented IP protocol [proto=ICMP 1, offset=1480, ID=815] [Reassembled in #26]
26. 9.656895558 192.168.100.216 193.136.9.240 ICMP 87 Echo (ping) request 0x8d4af, seq=3/788, ttl=1 (no response found!)
27. 9.656890521 192.168.100.216 193.136.9.240 IPv4 1514 Fragmented IP protocol [proto=ICMP 1, offset=0-1616] [Reassembled in #20]
28. 9.656891042 192.168.100.216 193.136.9.240 IPv4 1514 Fragmented IP protocol [proto=ICMP 1, offset=1480, ID=816] [Reassembled in #20]
29. 9.656911308 192.168.100.216 193.136.9.240 ICMP 87 Echo (ping) request 0x8d4af, seq=4/1024, ttl=1 (no response found!)

> Frame 18: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface 0
   Ethernet II, Src: Quantaco_6b:85:44 (2c:00:c0:6b:85:44), Dst: Intel(R) PRO/100 MT Desktop (00:0c:29:d2:19:f0)
Internet Protocol Version 4, Src: 192.168.100.216, Dst: 193.136.9.240
  Options: Version: 4
            . .101. Header Length: 20 bytes (5)
            Identification: 0x0000
            Flags: More Fragments Field: 0x0000 (OSCP: CS0, ECN: Not-ECT)
  Total Length: 1514
Identification: 0x0000 (43027)
Flags: 0x01 (More Fragments)
  0... .... = Reserved bit: Not set
  .0.... .... = Don't fragment: Not set
  .1.... .... = More fragments: Set
Fragment offset: 0
Time to live: 128
Protocol: ICMP (1)
Header checksum: 0xbfb14 [validation disabled]
[Header checksum status: Unverified]
Source: 192.168.100.216
Destination: 193.136.9.240
[Source Geop: Unknown]
```

Fig. 9: A MTU é menor do que o tamanho do pacote

### 2.12 Exercício 3.b.

## Questão

Imprima o primeiro fragmento do datagrama IP segmentado. Que informação no cabeçalho indica que o datagrama foi fragmentado? Que informação no cabeçalho IP indica que se trata do primeiro fragmento? Qual é o tamanho deste datagrama IP?

### Resposta

A informação de que o datagrama foi fragmentado é dada através da *flag more fragments* que se encontra a 1 em conjugação com o campo *fragment offset*, que neste caso se encontra a 0. Desta forma, podemos concluir que este datagrama é efetivamente o primeiro (*fragment offset = 0*) de um PDU fragmentado (*flag more fragments = 1*). Este é um datagrama IP de 1480 bytes (MTU de 1500 bytes, dos quais 20 bytes servem de *header*).

### Realização

No.	Time	Source	Destination	Protocol	Length	Info
12	9.318609076	192.168.100.216	192.168.100.254	DNS	80	Standard query 0x8cfa A dl-debug.dropbox.com
12	9.318609341	192.168.100.216	192.168.100.254	DNS	80	Standard query 0x8cd4 AAAA dl-debug.dropbox.com
13	9.319254879	192.168.100.254	192.168.100.216	DNS	198	Standard query response 0xcbc4 AAAA dl-debug.dropbox.com CNAME block-debug.x.drc
14	9.649429727	192.168.100.216	192.168.100.254	DNS	75	Standard query 0x3dct A marco.unimnho.pt
15	9.64943783	192.168.100.216	192.168.100.254	DNS	75	Standard query 0x78d5 AAAA marco.unimnho.pt
16	9.65913624	192.168.100.254	192.168.100.216	DNS	347	Standard query response 0x3dcf A marco.unimnho.pt A 193.136.9.240 NS dns3.unimnho.pt
17	9.659723657	192.168.100.254	192.168.100.216	DNS	129	Standard query response 0x78d8 AAAA marco.unimnho.pt S04 dns.unimnho.pt
18	9.659835086	192.168.100.216	193.136.9.240	IPv4	1514	Fragmented IP protocol [proto=ICMP, off=0, ID=813] [Reassembled in #20]
19	9.659835119	192.168.100.216	193.136.9.240	IPv4	1514	Fragmented IP protocol [proto=ICMP, off=0, ID=814] [Reassembled in #20]
*	20	9.659890359	192.168.100.216	193.136.9.240	ICMP	87 Echo (ping) request id=0x4af3, seq=1/254, ttl=1 (no response found!)
21	9.659896464	192.168.100.216	193.136.9.240	IPv4	1514	Fragmented IP protocol [proto=ICMP 1, off=0, ID=814] [Reassembled in #23]
22	9.65989742	192.168.100.216	193.136.9.240	IPv4	1514	Fragmented IP protocol [proto=ICMP 1, off=1480, ID=814] [Reassembled in #23]
23	9.65989829	192.168.100.216	193.136.9.240	ICMP	87 Echo (ping) request id=0x4af3, seq=2/254, ttl=1 (no response found!)	
24	9.659903569	192.168.100.216	193.136.9.240	IPv4	1514	Fragmented IP protocol [proto=ICMP 1, off=0, ID=815] [Reassembled in #26]
25	9.659903608	192.168.100.216	193.136.9.240	IPv4	1514	Fragmented IP protocol [proto=ICMP 1, off=1480, ID=815] [Reassembled in #26]
26	9.659905558	192.168.100.216	193.136.9.240	ICMP	87 Echo (ping) request id=0x4af3, seq=3/254, ttl=1 (no response found!)	
27	9.659909521	192.168.100.216	193.136.9.240	IPv4	1514	Fragmented IP protocol [proto=ICMP 1, off=0, ID=816] [Reassembled in #29]
28	9.659910424	192.168.100.216	193.136.9.240	IPv4	1514	Fragmented IP protocol [proto=ICMP 1, off=1488, ID=816] [Reassembled in #29]
29	9.659911383	192.168.100.216	193.136.9.240	ICMP	87	Echo (ping) request id=0x4af3, seq=4/1024, ttl=2 (no response found!)
	0100	.... + Version: 4				
	.... 0101	= Header length: 20 bytes (5)				
	> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)					
	> Identification: 0x011 (43027)					
	> Flags: 0x01 (More Fragments)					
	0... .... + Reserved bit: Not set					
	0... .... + Don't Fragment: Not set					
	0... .... + More Fragments: Set					
	> Fragment offset: 0					
	>					
	Protocol: ICMP (1)					
	Header checksum: 0xbfb14 [validation disabled]					
	[Header checksum status: Unverified]					
	Source: 192.168.100.216					
	Destination: 193.136.9.240					
	[Destination IP: Unknown]					
	> Reassembled IP in frame: #28					
	> Data (1480 bytes)					

Fig. 10: As flags e o fragment offset confirmam que é o primeiro fragmento.

### 2.13 Exercício 3.c.

#### Questão

#### Resposta

#### Realização

### 2.14 Exercício 3.d.

#### Questão

**Resposta**

**Realização**

**2.15 Exercício 3.e.**

**Questão**

**Resposta**

**Realização**

### **3 Parte II - Endereçamento e Encaminhamento IP**

#### **3.1 Exercício 2.1.a**

**Questão**

**Resposta**

**Realização**

#### **3.2 Exercício 2.1.b**

**Questão**

**Resposta**

**Realização**

#### **3.3 Exercício 2.1.c**

**Questão**

**Resposta**

**Realização**

#### **3.4 Exercício 2.1.d**

**Questão**

**Resposta**

**Realização**

#### **3.5 Exercício 2.1.e**

**Questão**

**Resposta**

**Realização**

**3.6 Exercício 2.2.a**

**Questão**

**Resposta**

**Realização**

**3.7 Exercício 2.2.b**

**Questão**

**Resposta**

**Realização**

**3.8 Exercício 2.2.c**

**Questão**

**Resposta**

**Realização**

**3.9 Exercício 2.2.d**

**Questão**

**Resposta**

**Realização**

### **3.10 Exercício 2.2.e**

**Questão**

**Resposta**

**Realização**

### **3.11 Exercício 3.1**

**Questão**

**Resposta**

**Realização**

### **3.12 Exercício 3.2**

**Questão**

**Resposta**

**Realização**

### **3.13 Exercício 3.3**

**Questão**

**Resposta**

**Realização**

According to Table 11...

## **4 Conclusions**

Neste trabalho...

## **References**

1. : Internet Control Message Protocol. RFC 792 (1981)
2. Wikipedia: Internet control message protocol. [https://pt.wikipedia.org/wiki/Internet\\_Control\\_Message\\_Protocol](https://pt.wikipedia.org/wiki/Internet_Control_Message_Protocol) (2017) [Online; accessed a 4-Novembro-2017].

(a) Delay and jitter	(b) Delay and loss
(c) Delay and throughput	(d) Jitter and loss
(e) Jitter and throughput	(f) Loss and throughput

Fig. 11: Tabela exemplo.