



Universidade do Minho

Braga, Portugal

# TRABALHO PRÁTICO 2 - RELATÓRIO

## INTERNET PROTOCOL (IP)

### Redes de Computadores

Departamento de Informática

Engenharia Informática 2024/25

Grupo 69:

Duarte Escairo Brandão Reis Silva

Pedro Emanuel Organista Silva

Tiago Silva Figueiredo

Março 2025

# Índice

1. Parte I .....	1
1.1. Questão 1 .....	1
1.2. Questão 2 .....	4
1.3. Questão 3 .....	7
2. Parte II .....	11
2.1. Questão 1 .....	11
2.2. Questão 2 .....	14
2.3. Questão 3 .....	21

## 1. Parte I

### 1.1. Questão 1

Prepare uma topologia CORE para verificar o comportamento do traceroute. Na topologia deve existir: um host (pc) cliente designado Lost, cujo router de acesso é RA1; o router RA1 está simultaneamente ligado a dois routers no core da rede RC1 e RC2; estes estão conectados a um router de acesso RA2, que por sua vez, se liga a um host (servidor) designado Found. Ajuste o nome dos equipamentos atribuídos por defeito para o enunciado. Apenas nas ligações (links) da rede de core, estabeleça um tempo de propagação de 15ms. Após ativar a topologia, note que pode não existir conectividade IP imediata entre Lost e Found até que o anúncio de rotas entre routers estabilize.

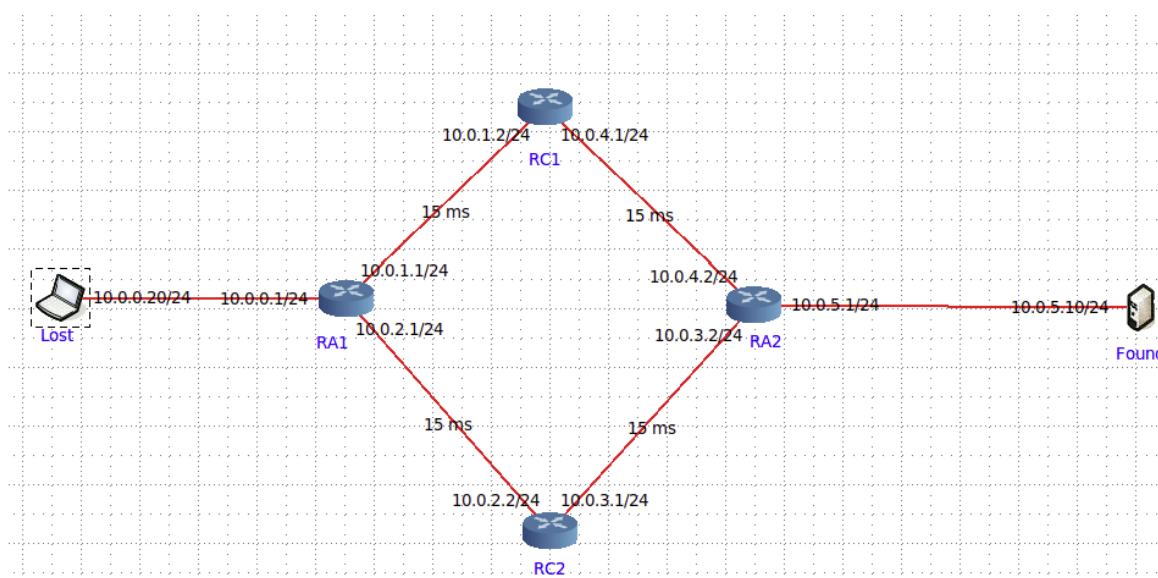


Figura 1: Topologia criada para o problema

a)

Ative o Wireshark no host Lost. Numa shell de Lost execute o comando `traceroute -I` para o endereço IP do Found. Registe e analise o tráfego ICMP enviado pelo sistema Lost e o tráfego ICMP recebido como resposta. Explique os resultados obtidos tendo em conta o princípio de funcionamento do traceroute.

RESPOSTA:

O comando `tracerout` acompanha os datagramas que saem do IP origem, incrementando o campo TTL em cada um. Os primeiros três tipos de datagramas (TTL = 1,2,3) dão timeout (é enviada uma mensagem de controlo ICMP) devido a não terem saltos suficientes para chegar ao IP destino definido. Os restantes chegam ao destino.

3	2.839887104	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=1/256, ttl=1 (no response..
4	2.839912055	10.0.0.1	10.0.0.20	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)	
5	2.839919047	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=2/512, ttl=1 (no response..
6	2.839922833	10.0.0.1	10.0.0.20	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)	
7	2.839925929	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=3/768, ttl=1 (no response..
8	2.839928633	10.0.0.1	10.0.0.20	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)	
9	2.839931568	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=4/1024, ttl=2 (no respons..
10	2.839940302	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=5/1280, ttl=2 (no respons..
11	2.839943468	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=6/1536, ttl=2 (no respons..
12	2.839946723	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=7/1792, ttl=3 (no respons..
13	2.839949458	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=8/2048, ttl=3 (no respons..
14	2.839952743	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=9/2304, ttl=3 (no respons..
15	2.839956409	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=10/2560, ttl=4 (reply in ..
16	2.839959334	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=11/2816, ttl=4 (reply in ..
17	2.839962039	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=12/3072, ttl=4 (reply in ..
18	2.839965104	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=13/3328, ttl=5 (reply in ..
19	2.839968239	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=14/3584, ttl=5 (reply in ..
20	2.839970953	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=15/3840, ttl=5 (reply in ..
21	2.839974149	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=16/4096, ttl=6 (reply in ..
22	2.840444271	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=17/4352, ttl=6 (reply in ..
23	2.840452324	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=18/4608, ttl=6 (reply in ..
24	2.840456752	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=19/4864, ttl=7 (reply in ..
25	2.872490268	10.0.1.2	10.0.0.20	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)	
26	2.872493824	10.0.1.2	10.0.0.20	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)	
27	2.872494505	10.0.1.2	10.0.0.20	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)	
28	2.872859272	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=20/5120, ttl=7 (reply in ..
29	2.872871322	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=21/5376, ttl=7 (reply in ..
30	2.872876311	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request	id=0x002c, seq=22/5632, ttl=8 (reply in ..
31	2.903817303	10.0.3.2	10.0.0.20	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)	
32	2.903822642	10.0.3.2	10.0.0.20	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)	
33	2.903824065	10.0.3.2	10.0.0.20	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)	
34	2.903825106	10.0.5.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x002c, seq=10/2560, ttl=61 (request ..
35	2.903826158	10.0.5.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x002c, seq=11/2816, ttl=61 (request ..
36	2.903827250	10.0.5.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x002c, seq=12/3072, ttl=61 (request ..
37	2.903828202	10.0.5.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x002c, seq=13/3328, ttl=61 (request ..
38	2.903829143	10.0.5.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x002c, seq=14/3584, ttl=61 (request ..
39	2.903830145	10.0.5.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x002c, seq=15/3840, ttl=61 (request ..
40	2.903831146	10.0.5.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x002c, seq=16/4096, ttl=61 (request ..
41	2.903832148	10.0.5.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x002c, seq=17/4352, ttl=61 (request ..
42	2.903833060	10.0.5.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x002c, seq=18/4608, ttl=61 (request ..
43	2.903833971	10.0.5.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x002c, seq=19/4864, ttl=61 (request ..
44	2.934512213	10.0.5.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x002c, seq=20/5120, ttl=61 (request ..
45	2.934518023	10.0.5.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x002c, seq=21/5376, ttl=61 (request ..
46	2.934519415	10.0.5.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x002c, seq=22/5632, ttl=61 (request ..

Figura 2: Tráfego capturado com o comando *traceroute*

b)

Qual deve ser o valor inicial mínimo do campo TTL para alcançar o servidor Found? Esboce um esquema com o valor do campo TTL à chegada a cada um dos routers percorridos até ao servidor Found. Verifique na prática que a sua resposta está correta.

RESPOSTA:

**TTL = 4**

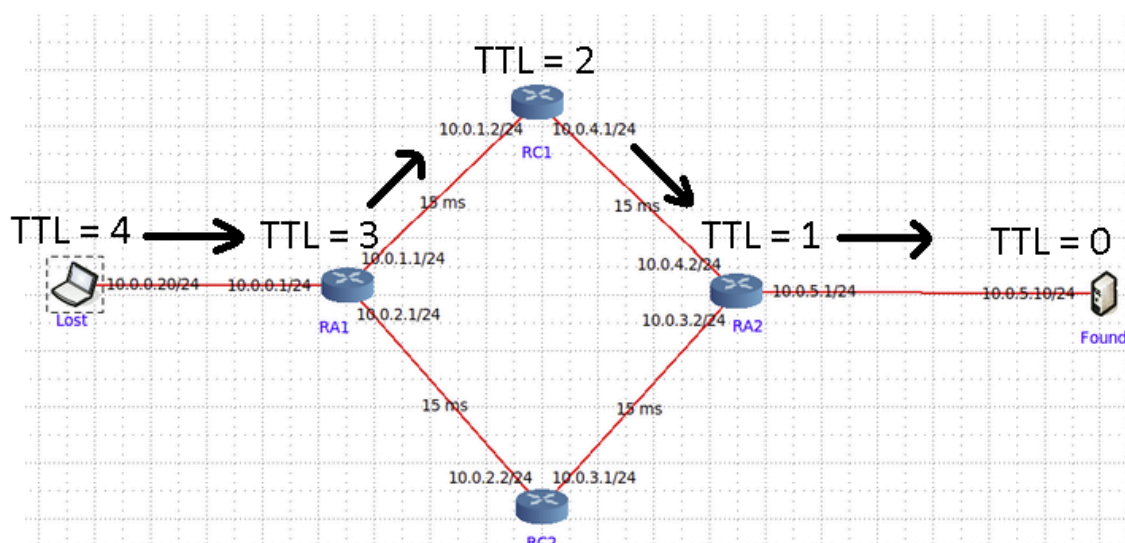


Figura 3: Esquema com o valor do TTL

c)

Calcule o valor médio do tempo de ida-e-volta (RTT - Round-Trip Time) obtido no acesso ao servidor. Por modo a obter uma média mais confiável, poderá alterar o número pacotes de prova com a opção -q.

RESPOSTA:

O valor médio do RTT obtido no acesso ao servidor é: 61.338195ms.

d)

O valor médio do atraso num sentido (One-Way Delay) poderia ser calculado com precisão dividindo o RTT por dois? O que torna difícil o cálculo desta métrica numa rede real?

RESPOSTA:

No caso de estudo em concreto é possível obter um resultado relativamente preciso devido ao facto de não haver congestionamento e os caminhos serem muito parecidos. Num caso real, o resultado não iria ser necessariamente preciso, porque os caminhos podem ser diferentes e grandes, o que influencia no tempo de pedido e de resposta.

## 1.2. Questão 2

Usando o Wireshark capture o tráfego gerado pelo traceroute sem especificar o tamanho do pacote, i.e., quando é usado o tamanho do pacote de prova por defeito. Utilize como máquina destino o host marco.uminho.pt. Pare a captura. Com base no tráfego capturado, identifique os pedidos ICMP Echo Request e o conjunto de mensagens devolvidas como resposta.

a)

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

RESPOSTA:

```
75 0.357879276 193.136.9.240 172.26.80.96 ICMP 74 Echo (ping) reply
```

O endereço IP da interface ativa do computador é : 172.26.80.96.

b)

Qual é o valor do campo protocol? O que permite identificar?

RESPOSTA:

```
Protocol: ICMP (1)
```

O campo protocol tem o valor 1 e permite identificar o payload encapsulado no pacote ICMP.

c)

Quantos bytes tem o cabeçalho IPv4? Quantos bytes tem o campo de dados (payload) do datagrama? Como se calcula o tamanho do payload?

RESPOSTA:

```
Internet Protocol Version 4, Src: 172.26.80.96, Dst: 193.136.9.240
  0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▶ Differentiated Services Field: 0x00 (DSCP: CS0)
    Total Length: 60
```

O cabeçalho IPV4 tem 20 bytes. O payload tem 40 bytes. O tamanho do payload pode ser calculado através da subtração do tamanho do cabeçalho ao tamanho total da mensagem.

d)

O datagrama IP foi fragmentado? Justifique.

RESPOSTA:

```
Fragment Offset: 0
```

O datagrama não foi fragmentado porque os bits do offset e da flag do pacote estão a 0.

e)

Análise a sequência de tráfego ICMP gerado a partir do endereço IP atribuído à interface da sua máquina. Para a sequência de mensagens ICMP enviadas pelo seu computador, indique que campos do cabeçalho IP variam de pacote para pacote. Justifique estas mudanças.

RESPOSTA:

```
Internet Protocol Version 4, Src: 172.26.80.96, Dst: 193.136.9.240
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 60
  Identification: 0x3bd5 (15317)
  000. .... = Flags: 0x0
  ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 1
  Protocol: ICMP (1)
  Header Checksum: 0xb5f9 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 172.26.80.96
  Destination Address: 193.136.9.240

Internet Protocol Version 4, Src: 172.26.80.96, Dst: 193.136.9.240
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 60
  Identification: 0x3bd8 (15320)
  000. .... = Flags: 0x0
  ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 2
  Protocol: ICMP (1)
  Header Checksum: 0xb4f6 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 172.26.80.96
  Destination Address: 193.136.9.240
```

Os campos do cabeçalho IP que mudam de pacote para pacote são: o Identifier, o TTL e o Header Checksum. O Identifier muda porque é o identificador único de cada pacote, o TTL muda porque com o envio dos vários pacotes, o TTL é aumentado para se poder verificar se consegue ou não chegar ao destino e o Header Checksum .

f)

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

RESPOSTA:

O valor do campo TTL incrementa em uma unidade de três em três pacotes. O ID do datagrama aumenta de um em um.

g)

Ordene o tráfego capturado por endereço destino e encontre a série de respostas ICMP TTL Exceeded enviadas ao seu computador.

21	0.006836586	172.16.2.1	172.26.80.96	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
23	0.007385373	172.16.115.252	172.26.80.96	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
24	0.007385572	172.16.115.252	172.26.80.96	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25	0.007418719	172.16.2.1	172.26.80.96	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
26	0.007418998	172.26.254.254	172.26.80.96	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
27	0.007418776	172.16.2.1	172.26.80.96	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
28	0.007419048	172.26.254.254	172.26.80.96	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
29	0.007419194	172.26.254.254	172.26.80.96	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
30	0.007423353	193.136.9.240	172.26.80.96	ICMP	74 Echo (ping) reply id=0x1a1b, seq=10/2560, ttl=61 (requ
35	0.008032415	172.16.115.252	172.26.80.96	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)

i)

Qual é o valor do campo TTL recebido no seu computador? Esse valor permanece constante para todas as mensagens de resposta ICMP TTL Exceeded recebidas no seu computador? Porquê?

RESPOSTA:

Os valores do campo TTL recebidos no computador são: 253, 254 e 255. Estes valores não permanecem constantes para todas as mensagens porque o TTL mínimo para que o pacote chegasse ao destino é 4. Pacotes enviados com TTL 1 têm na mensagem de resposta um TTL Exceeded de 255, os de TTL 2 uma resposta de TTL Exceeded de 254 e os de TTL 3, uma resposta de TTL Exceeded de 253. Isto porque a soma do TTL inicial do pacote e do TTL da resposta é sempre 256.

ii)

Por que razão as mensagens de resposta ICMP TTL Exceeded são sempre enviadas na origem com um valor relativamente alto?

RESPOSTA:

As mensagens de resposta ICMP TTL Exceeded são sempre enviadas na origem com um valor relativamente alto porque assim garante-se que ela consegue percorrer toda a rota de volta até à origem.

h)

A informação contida no cabeçalho ICMP poderia ser incluída no cabeçalho IPv4? Se sim, quais seriam as suas vantagens/desvantagens?

RESPOSTA:

Sim, informação contida no cabeçalho ICMP poderia ser incluída no cabeçalho IPv4.

Vantagens: Menos overhead

Desvantagens: Menos modular, Caso o pacote não fosse ICMP seriam encapsulados 8 bytes de ICMP desnecessariamente



### 1.3. Questão 3

Pretende-se agora analisar a fragmentação de pacotes IP. Usando o Wireshark, capture e observe o tráfego gerado depois do tamanho de pacote ter sido definido para (3800 + X) bytes, em que X é o número do grupo de trabalho (e.g., X=22 para o grupo PL22). De modo a poder visualizar os fragmentos, aceda a Edit -> Preferences -> Protocols e em IPv4 desative a opção “Reassemble fragmented IPv4 datagrams”.

a)

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

RESPOSTA:

**Houve necessidade de fragmentar o pacote inicial porque o tamanho do mesmo era superior ao MTU da interface de rede usada (tipicamente e até neste caso, 1500 bytes).**

b)

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 no cabeçalho que indica que o datagrama foi fragmentada é o bit da flag que indica que existem mais fragmentos. A informação que indica que se trata do primeiro fragmento é o facto dos bits do offset serem 0. O tamanho deste datagrama é de 1500 bytes.**

```
Total Length: 1500
Identification: 0x083e (2110)
001. .... = Flags: 0x1, More fragments
...0 0000 0000 0000 = Fragment Offset: 0
```

c)

Imprima o segundo fragmento do datagrama IP original. Que informação do cabeçalho IP indica que não se trata do 1º fragmento? Existem mais fragmentos? O que nos permite afirmar isso?

RESPOSTA:

**A informação do cabeçalho IP que indica que não se trata do 1º fragmento é o facto de os bits do offset serem 1480. Existem mais fragmentos porque os bits da flag de ‘more fragments’ é 1.**

```
001. .... = Flags: 0x1, More fragments
0... .... = Reserved bit: Not set
.0.. .... = Don't fragment: Not set
..1. .... = More fragments: Set
...0 0000 1011 1001 = Fragment Offset: 1480
```

d)

Quantos fragmentos foram criados a partir do datagrama original? Como se detecta o último fragmento correspondente ao datagrama original? Estabeleça um filtro no Wireshark que permita listar apenas o último fragmento do primeiro datagrama IP segmentado.

RESPOSTA:

A partir do datagrama original foram criados 3 fragmentos. O último fragmento é detectado a partir da flag 'more fragments' estar a 0. O comando usado foi: 'ip.flags.mf==0 and ip.frag\_offset!= 0'.

```
Total Length: 937
Identification: 0x083e (2110)
000. .... = Flags: 0x0
    0... .... = Reserved bit: Not set
    .0.. .... = Don't fragment: Not set
    ..0. .... = More fragments: Not set
    ...0 0001 0111 0010 = Fragment Offset: 2960
```

e)

Indique, resumindo, os campos que mudam no cabeçalho IP entre os diferentes fragmentos, e explique a forma como essa informação permite reconstruir o datagrama original.

RESPOSTA:

A informação que muda no cabeçalho IP entre os diferentes fragmentos são: a flag 'more fragments', o offset, o 'header checksum' e, eventualmente, o comprimento (neste caso, do terceiro fragmento). Esta informação permite reconstruir o datagrama original pois eles têm todos o mesmo identificador, logo, pertencem ao mesmo pacote. De seguida, quando são recebidos, através da verificação da flag 'more fragments' e do offset, é possível reconstruir a ordem do pacote toda.

```
Total Length: 1500
Identification: 0x1933 (6451)
001. .... = Flags: 0x1, More fragments
...0 0000 1011 1001 = Fragment Offset: 1480

Header Checksum: 0x7342 [validation disabled]
```

```
Total Length: 937
Identification: 0x1933 (6451)
000. .... = Flags: 0x0
...0 0001 0111 0010 = Fragment Offset: 2960

Header Checksum: 0x94bc [validation disabled]
```

f)

Estime teoricamente o número de fragmentos gerados e o número de bytes transportados em cada um dos fragmentos. Apresente todos os cálculos efetuados, incluindo os campos do cabeçalho IP relevantes para cada um dos fragmentos.

RESPOSTA:

**Total de bytes a enviar: 3869 bytes**

**MTU: 1500 bytes**

Número de fragmentos:  $\frac{3869}{1500} = 2.579$  , logo são necessários 3 fragmentos.

Os headers têm 20 bytes.

1º fragmento:  $1500 - 20 - 8 = 1472$  bytes de payload

2º fragmento:  $1500 - 20 = 1480$  bytes de payload

3º fragmento:  $3869 - (1480 + 1472) = 917$  bytes de payload

O offset do primeiro fragmento será 0, o offset do segundo fragmento será 1480, e o offset do terceiro e último fragmento será 2960. A flag “more fragments” dos dois primeiros estará a 1 a do último estará a 0.

```
Total Length: 1500
Identification: 0x20ba (8378)
001. .... = Flags: 0x1, More fragments
...0 0000 0000 0000 = Fragment Offset: 0

Total Length: 1500
Identification: 0x20ba (8378)
001. .... = Flags: 0x1, More fragments
...0 0000 1011 1001 = Fragment Offset: 1480

Total Length: 937
Identification: 0x20ba (8378)
000. .... = Flags: 0x0
...0 0001 0111 0010 = Fragment Offset: 2960
```

g)

Por que razão apenas o primeiro fragmento de cada pacote é identificado pelo Wireshark como sendo um pacote ICMP? Justifique a sua resposta com base no conceito de Fragmentação apresentado nas aulas teóricas.

RESPOSTA:

O cabeçalho ICMP fica apenas no primeiro fragmento porque seria desnecessário manter o mesmo cabeçalho ICMP para os restantes fragmentos visto que a refragmentação é sempre feita no destino, logo, o destino sabe a qual pacote os fragmentos pertencem através do seu campo de identificação.

h)

Com que valor é o tamanho do datagrama comparado a fim de se determinar se este deve ser fragmentado? Quais seriam os efeitos na rede ao aumentar/diminuir este valor?

RESPOSTA:

O tamanho do datagrama é comparado com o valor do MTU. Caso esse fosse aumentado, o aumento do tamanho dos datagramas poderia sobrecarregar a rede. Caso diminuísse, haveria ainda mais fragmentação.

i)

Sabendo que no comando ping a opção “-f” (Windows), “-M do” (Linux) ou “-D” (Mac) ativa a flag “Don’t Fragment” (DF) no cabeçalho do IPv4, usando ping < opção DF > < opção pkt\_size > SIZE marco.uminho.pt, (opção pkt\_size = -1 (Windows) ou -s (Linux, Mac), determine o valor máximo de SIZE sem que ocorra fragmentação do pacote? Justifique o valor obtido.

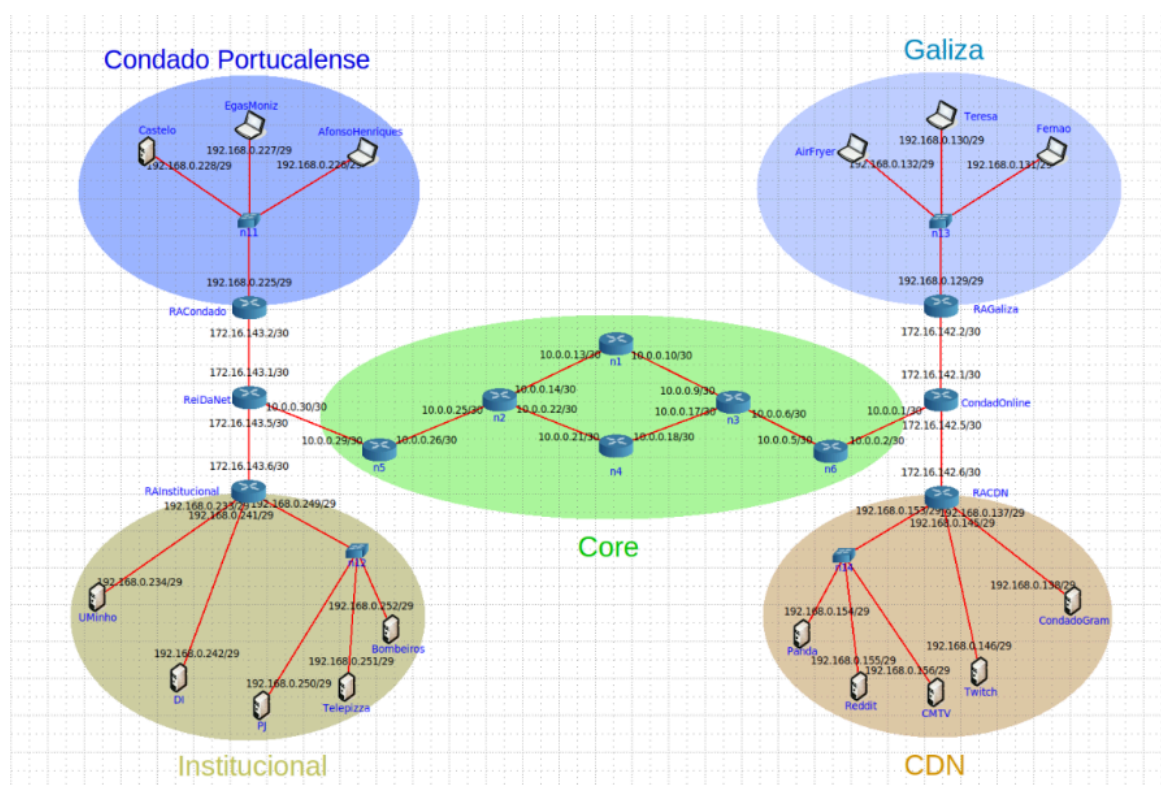
RESPOSTA:

**O valor máximo obtido de SIZE sem que ocorra fragmentação é 1472 bytes. Isto acontece porque o MTU é de 1500 e como o cabeçalho IP ocupa 20 bytes e o cabeçalho ICMP ocupa 8 bytes,  $1500 - 20 - 8 = 1472$ .**

## 2. Parte II

Nos polos Condado e Galiza existe um router de acesso (que permite comunicação com o exterior) ligado a um comutador (switch),cpor modo a que todos os dispositivos partilhem a mesma rede local. Em cada um dos polos Institucional e CDN encontram-se três sub-redes distintas, criando-se uma segmentação dos dispositivos existentes.

Os polos Condado Portucalense e Institucional estão ligados ao router do ISP ReiDaNet, enquanto Galiza e CDN estão ligados ao ISP CondadOnline. Interligando os dois ISPs existe um ISP de trânsito cuja rede Core é constituída pelos dispositivos n1 a n6.



### 2.1. Questão 1

Com os avanços da Inteligência Artificial, D. Afonso Henriques termina todas as suas tarefas mais cedo e vê-se com algum tempo livre. Decide então fazer remodelações no reino:

a)

De modo a garantir uma posição estrategicamente mais vantajosa e ter casa de férias para relaxar entre batalhas, ordena a construção de um segundo Castelo, em Braga. Não tendo qualquer queixa do serviço prestado, recorre aos serviços do ISP ReiDaNet, que já utiliza no condado, para ter acesso à rede no segundo Castelo. O ISP atribuiu-lhe o endereço de rede IP 172.68.XX.192/26 em que XX corresponde ao seu número de grupo (PLXX). Defina um esquema de endereçamento que permita o estabelecimento de pelo menos 6 redes e que garanta que cada uma destas possa ter 5 ou mais hosts. Assuma que todos os endereços de sub-redes são utilizáveis.

RESPOSTA:

Sub-rede	Min	Max
SR1: 172.68.69.192/29	.193	.198
SR2: 172.68.69.200/29	.201	.206
SR3: 172.68.69.208/29	.209	.214
SR4: 172.68.69.216/29	.217	.222
SR5: 172.68.69.224/29	.225	.230
SR6: 172.68.69.232/29	.233	.238
SR7: 172.68.69.240/29	.241	.246
SR8: 172.68.69.248/29	.249	.254

Ligue um novo host Castelo2 diretamente ao router ReiDaNet. Associe-lhe um endereço, à sua escolha, pertencente a uma sub-rede disponível das criadas na alínea anterior (garanta que a interface do router ReiDaNet utiliza o primeiro endereço válido da sub-rede escolhida). Verifique que tem conectividade com os dispositivos do Condado Portucalense.

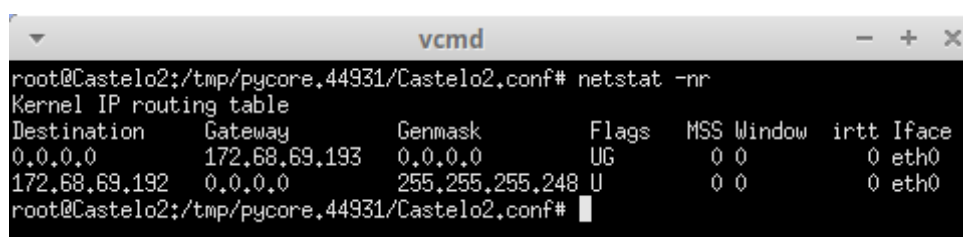
Como se pode ver na imagem a baixo, foi adicionado um novo host (Castelo2) ao router ReiDaNet tal como pedido no enunciado. Também podemos verificar que existe conectividade entre o Castelo2 e o Condado Portucalense.



Não estando satisfeito com a decoração deste novo Castelo, opta por eliminar a sua rota default. Adicione as rotas necessárias para que o Castelo2 continue a ter acesso ao Condado Portucalense e à rede Institucional. Mostre que a conectividade é restabelecida, assim como a tabela de encaminhamento resultante. Explícite ainda a utilidade de uma rota default.

RESPOSTA:

A tabela de encaminhamento do Castelo2 antes das alterações pedidas era a seguinte:



```

root@Castelo2:/tmp/pycore.44931/Castelo2.conf# netstat -nr
Kernel IP routing table
Destination      Gateway         Genmask        Flags   MSS Window  irtt Iface
0.0.0.0          172.68.69.193  0.0.0.0        UG      0 0        0 eth0
172.68.69.192    0.0.0.0        255.255.255.248 U        0 0        0 eth0
root@Castelo2:/tmp/pycore.44931/Castelo2.conf#

```

Após a remoção da rota default, a tabela era esta:

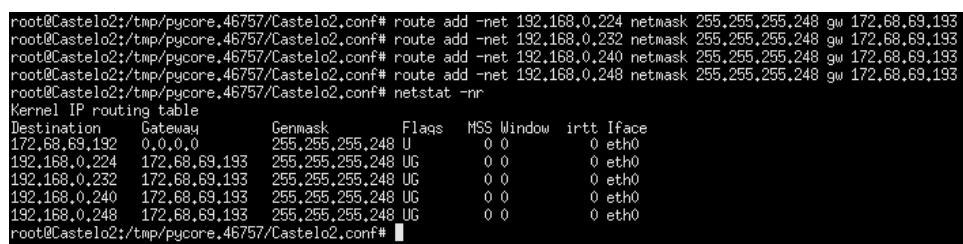


```

<4931/Castelo2.conf# route del -net 0.0.0.0 gw 172.68.69.193 netmask 0.0.0.0
root@Castelo2:/tmp/pycore.44931/Castelo2.conf# netstat -nr
Kernel IP routing table
Destination      Gateway         Genmask        Flags   MSS Window  irtt Iface
172.68.69.192    0.0.0.0        255.255.255.248 U        0 0        0 eth0
root@Castelo2:/tmp/pycore.44931/Castelo2.conf#

```

Com a adição de novas entradas na tabela, para garantir a conectividade entre o Condado Portucalense e rede Institucional, a tabela tinha este aspeto:



```

root@Castelo2:/tmp/pycore.46757/Castelo2.conf# route add -net 192.168.0.224 netmask 255.255.255.248 gw 172.68.69.193
root@Castelo2:/tmp/pycore.46757/Castelo2.conf# route add -net 192.168.0.232 netmask 255.255.255.248 gw 172.68.69.193
root@Castelo2:/tmp/pycore.46757/Castelo2.conf# route add -net 192.168.0.240 netmask 255.255.255.248 gw 172.68.69.193
root@Castelo2:/tmp/pycore.46757/Castelo2.conf# route add -net 192.168.0.248 netmask 255.255.255.248 gw 172.68.69.193
root@Castelo2:/tmp/pycore.46757/Castelo2.conf# netstat -nr
Kernel IP routing table
Destination      Gateway         Genmask        Flags   MSS Window  irtt Iface
172.68.69.192    0.0.0.0        255.255.255.248 U        0 0        0 eth0
192.168.0.224    172.68.69.193  255.255.255.248 UG      0 0        0 eth0
192.168.0.232    172.68.69.193  255.255.255.248 UG      0 0        0 eth0
192.168.0.240    172.68.69.193  255.255.255.248 UG      0 0        0 eth0
192.168.0.248    172.68.69.193  255.255.255.248 UG      0 0        0 eth0
root@Castelo2:/tmp/pycore.46757/Castelo2.conf#

```

A conectividade entre as diversas redes foi garantida com estas alterações na tabela, como prova, foram enviados “pings” para os hosts Egas Moniz, UMinho, DI e PJ respetivamente. Os resultados encontram-se na imagem a baixo.

```
root@Castelo2:/tmp/pycore.46757/Castelo2.conf# ping 192.168.0.227
PING 192.168.0.227 (192.168.0.227) 56(84) bytes of data.
64 bytes from 192.168.0.227: icmp_seq=1 ttl=62 time=0.062 ms
^Z
[5]+  Stopped                  ping 192.168.0.227
root@Castelo2:/tmp/pycore.46757/Castelo2.conf# ping 192.168.0.234
PING 192.168.0.234 (192.168.0.234) 56(84) bytes of data.
64 bytes from 192.168.0.234: icmp_seq=1 ttl=62 time=0.045 ms
^Z
[6]+  Stopped                  ping 192.168.0.234
root@Castelo2:/tmp/pycore.46757/Castelo2.conf# ping 192.168.0.242
PING 192.168.0.242 (192.168.0.242) 56(84) bytes of data.
64 bytes from 192.168.0.242: icmp_seq=1 ttl=62 time=0.047 ms
^Z
[7]+  Stopped                  ping 192.168.0.242
root@Castelo2:/tmp/pycore.46757/Castelo2.conf# ping 192.168.0.250
PING 192.168.0.250 (192.168.0.250) 56(84) bytes of data.
64 bytes from 192.168.0.250: icmp_seq=1 ttl=62 time=0.089 ms
^Z
[8]+  Stopped                  ping 192.168.0.250
```

Uma rota default é útil numa tabela de routing, pois permite efetuar saltos para IP's que não estejam especificados na tabela.

## 2.2. Questão 2

D.Afonso Henriques quer enviar fotos do novo Castelo à sua mãe, D.Teresa, mas está a ter alguns problemas de comunicação. Este alega que o problema deverá estar no dispositivo de D.Teresa, uma vez que no dia anterior conseguiu fazer stream deFortnite para todos os seus subscritores da Twitch, e acabou de sair de uma discussão política no Reddit.

a)

Confirme, através do comando ping, que AfonsoHenriques tem efetivamente conectividade com os servidores Reddit e Twitch.

RESPOSTA:

Como podemos ver na imagem a baixo, existe conectividade entre AfonsoHenriques e os servidores Reddit e Twitch, devido ao uso do comando ping ter sido bem sucedido.

```
root@AfonsoHenriques:/tmp/pycore.46757/AfonsoHenriques.conf# ping 192.168.0.155
PING 192.168.0.155 (192.168.0.155) 56(84) bytes of data.
64 bytes from 192.168.0.155: icmp_seq=1 ttl=55 time=0.099 ms
^Z
[2]+  Stopped                  ping 192.168.0.155
root@AfonsoHenriques:/tmp/pycore.46757/AfonsoHenriques.conf# ping 192.168.0.146
PING 192.168.0.146 (192.168.0.146) 56(84) bytes of data.
64 bytes from 192.168.0.146: icmp_seq=1 ttl=55 time=0.165 ms
^Z
[3]+  Stopped                  ping 192.168.0.146
```

b)

Recorrendo ao comando netstat -rn, analise as tabelas de encaminhamento dos dispositivos AfonsoHenriques e Teresa. Existe algum problema com as suas entradas? Identifique e descreva a utilidade de cada uma das entradas destes dois hosts.



RESPOSTA:

```
root@AfonsoHenriques:/tmp/pycore.46757/AfonsoHenriques.conf# netstat -rn
Kernel IP routing table
Destination      Gateway          Genmask         Flags   MSS Window  irtt Iface
0.0.0.0          192.168.0.225   0.0.0.0         UG        0 0        0 eth0
192.168.0.224    0.0.0.0         255.255.255.248 U        0 0        0 eth0
```

```
root@Teresa:/tmp/pycore.46757/Teresa.conf# netstat -rn
Kernel IP routing table
Destination      Gateway          Genmask         Flags   MSS Window  irtt Iface
0.0.0.0          192.168.0.129   0.0.0.0         UG        0 0        0 eth0
192.168.0.128    0.0.0.0         255.255.255.248 U        0 0        0 eth0
```

Ambas as tabelas de routing dos dispositivos AfonsoHenriques e Teresa, não apresentam entradas para redes internas. Cada uma delas apresenta 2 entradas, uma delas é a default, que serve para enviar mensagens cujo o IP destino não está especificado na tabela, e a outra entrada é usada quando o destino é a própria sub rede.

c)

Análise o comportamento dos routers do core da rede (n1 a n6) quando tenta estabelecer comunicação entre os hosts AfonsoHenriques e Teresa. Indique que dispositivo(s) não permite(m) o encaminhamento correto dos pacotes. Seguidamente, avalie e explique a(s) causa(s) do funcionamento incorreto do dispositivo.

Utilize o comando `ip route add/del` para adicionar as rotas necessárias ou remover rotas incorretas. Verifique a sintaxe completa do comando a usar com `man ip-route` ou `man route`. Poderá também utilizar o comando `traceroute` para se certificar do caminho nó a nó. Considere a alínea resolvida assim que houver tráfego a chegar ao ISP CondadOnline.

RESPOSTA:

Inicialmente não existe conectividade entre os dispositivos AfonsoHenriques e Teresa.

```
<6751/AfonsoHenriques.conf# traceroute 192.168.0.130
traceroute to 192.168.0.130 (192.168.0.130), 30 hops max, 60 byte packets
 1 192.168.0.225 (192.168.0.225) 0.065 ms 0.005 ms 0.003 ms
 2 172.16.143.1 (172.16.143.1) 0.016 ms 0.006 ms 0.006 ms
 3 10.0.0.29 (10.0.0.29) 0.017 ms 0.008 ms 0.008 ms
 4 10.0.0.25 (10.0.0.25) 0.027 ms 0.010 ms 0.009 ms
 5 10.0.0.25 (10.0.0.25) 3065.474 ms !H 3065.458 ms !H 3065.446 ms !H
```

Com o auxílio do comando `traceroute`, percebemos que os pacotes paravam no endereço 10.0.0.25, que correspondia ao router n2.

```

root@n2:/tmp/pycore.36751/n2.conf# netstat -nr
Kernel IP routing table

```

Destination	Gateway	Genmask	Flags	MSS	Window	irtt	Iface
10.0.0.0	10.0.0.13	255.255.255.252	UG	0	0	0	eth1
10.0.0.4	10.0.0.21	255.255.255.252	UG	0	0	0	eth0
10.0.0.8	10.0.0.13	255.255.255.252	UG	0	0	0	eth1
10.0.0.12	0.0.0.0	255.255.255.252	U	0	0	0	eth1
10.0.0.16	10.0.0.13	255.255.255.252	UG	0	0	0	eth1
10.0.0.20	0.0.0.0	255.255.255.252	U	0	0	0	eth0
10.0.0.24	0.0.0.0	255.255.255.252	U	0	0	0	eth2
10.0.0.28	10.0.0.26	255.255.255.252	UG	0	0	0	eth2
172.0.0.0	10.0.0.26	255.0.0.0	UG	0	0	0	eth2
172.16.142.0	10.0.0.13	255.255.255.252	UG	0	0	0	eth1
172.16.142.4	10.0.0.21	255.255.255.252	UG	0	0	0	eth0
172.16.143.0	10.0.0.26	255.255.255.252	UG	0	0	0	eth2
172.16.143.4	10.0.0.26	255.255.255.252	UG	0	0	0	eth2
192.168.0.128	10.0.0.13	255.255.255.248	UG	0	0	0	eth1
192.168.0.130	10.0.0.25	255.255.255.254	UG	0	0	0	eth2
192.168.0.136	10.0.0.21	255.255.255.248	UG	0	0	0	eth0
192.168.0.144	10.0.0.21	255.255.255.248	UG	0	0	0	eth0
192.168.0.152	10.0.0.21	255.255.255.248	UG	0	0	0	eth0
192.168.0.224	10.0.0.26	255.255.255.248	UG	0	0	0	eth2
192.168.0.232	10.0.0.26	255.255.255.248	UG	0	0	0	eth2
192.168.0.240	10.0.0.26	255.255.255.248	UG	0	0	0	eth2
192.168.0.248	10.0.0.26	255.255.255.248	UG	0	0	0	eth2

Ao analisar a tabela, vimos que a entrada com destino 192.168.0.130 (Teresa) era enviada para o próprio router.

```

root@n2:/tmp/pycore.39931/n2.conf# route del -net 192.168.0.130 gw 10.0.0.25 n>
root@n2:/tmp/pycore.39931/n2.conf# netstat -nr
Kernel IP routing table

```

Destination	Gateway	Genmask	Flags	MSS	Window	irtt	Iface
10.0.0.0	10.0.0.13	255.255.255.252	UG	0	0	0	eth1
10.0.0.4	10.0.0.21	255.255.255.252	UG	0	0	0	eth0
10.0.0.8	10.0.0.13	255.255.255.252	UG	0	0	0	eth1
10.0.0.12	0.0.0.0	255.255.255.252	U	0	0	0	eth1
10.0.0.16	10.0.0.13	255.255.255.252	UG	0	0	0	eth1
10.0.0.20	0.0.0.0	255.255.255.252	U	0	0	0	eth0
10.0.0.24	0.0.0.0	255.255.255.252	U	0	0	0	eth2
10.0.0.28	10.0.0.26	255.255.255.252	UG	0	0	0	eth2
172.0.0.0	10.0.0.26	255.0.0.0	UG	0	0	0	eth2
172.16.142.0	10.0.0.13	255.255.255.252	UG	0	0	0	eth1
172.16.142.4	10.0.0.21	255.255.255.252	UG	0	0	0	eth0
172.16.143.0	10.0.0.26	255.255.255.252	UG	0	0	0	eth2
172.16.143.4	10.0.0.26	255.255.255.252	UG	0	0	0	eth2
192.168.0.128	10.0.0.13	255.255.255.248	UG	0	0	0	eth1
192.168.0.136	10.0.0.21	255.255.255.248	UG	0	0	0	eth0
192.168.0.144	10.0.0.21	255.255.255.248	UG	0	0	0	eth0
192.168.0.152	10.0.0.21	255.255.255.248	UG	0	0	0	eth0
192.168.0.224	10.0.0.26	255.255.255.248	UG	0	0	0	eth2
192.168.0.232	10.0.0.26	255.255.255.248	UG	0	0	0	eth2
192.168.0.240	10.0.0.26	255.255.255.248	UG	0	0	0	eth2
192.168.0.248	10.0.0.26	255.255.255.248	UG	0	0	0	eth2

Removemos essa entrada da tabela e desta forma os pacotes seriam enviados para n1 (10.0.0.13) quando o IP destino fosse 192.168.0.128, que identifica a sub rede onde está contida a Teresa.

```
<9931/AfonsoHenriques.conf# traceroute 192.168.0.130
traceroute to 192.168.0.130 (192.168.0.130), 30 hops max, 60 byte packets
 1 192.168.0.225 (192.168.0.225)  0.025 ms  0.003 ms  0.002 ms
 2 172.16.143.1 (172.16.143.1)  0.011 ms  0.004 ms  0.003 ms
 3 10.0.0.29 (10.0.0.29)  0.012 ms  0.008 ms  0.005 ms
 4 10.0.0.25 (10.0.0.25)  0.022 ms  0.006 ms  0.006 ms
 5 10.0.0.13 (10.0.0.13)  0.016 ms  0.007 ms  0.006 ms
 6 10.0.0.25 (10.0.0.25)  0.007 ms  0.013 ms  0.006 ms
 7 10.0.0.13 (10.0.0.13)  0.008 ms  0.007 ms  0.007 ms
 8 * * *
 9 * * *
10 * * *
11 * * *
12 * * *
13 * 10.0.0.13 (10.0.0.13)  0.118 ms  0.025 ms
14 10.0.0.25 (10.0.0.25)  0.023 ms  0.025 ms  0.026 ms
15 10.0.0.13 (10.0.0.13)  0.036 ms  0.034 ms  0.026 ms
16 10.0.0.25 (10.0.0.25)  0.026 ms  0.027 ms  *^Z
[1]+  Stopped                  traceroute 192.168.0.130
root@AfonsoHenriques:/tmp/pycore.39931/AfonsoHenriques.conf#
```

Voltamos a usar o comando `traceroute`, e desta vez os pacotes paravam em n1.

```
root@n1:/tmp/pycore.39931/n1.conf# netstat -nr
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
10.0.0.0 10.0.0.9 255.255.255.252 UG 0 0 0 eth0
10.0.0.4 10.0.0.9 255.255.255.252 UG 0 0 0 eth0
10.0.0.8 0.0.0.0 255.255.255.252 U 0 0 0 eth0
10.0.0.12 0.0.0.0 255.255.255.252 U 0 0 0 eth1
10.0.0.16 10.0.0.9 255.255.255.252 UG 0 0 0 eth0
10.0.0.20 10.0.0.14 255.255.255.252 UG 0 0 0 eth1
10.0.0.24 10.0.0.14 255.255.255.252 UG 0 0 0 eth1
10.0.0.28 10.0.0.14 255.255.255.252 UG 0 0 0 eth1
172.0.0.0 10.0.0.14 255.0.0.0 UG 0 0 0 eth1
172.16.142.0 10.0.0.9 255.255.255.252 UG 0 0 0 eth0
172.16.142.4 10.0.0.9 255.255.255.252 UG 0 0 0 eth0
172.16.143.0 10.0.0.14 255.255.255.252 UG 0 0 0 eth1
172.16.143.4 10.0.0.14 255.255.255.252 UG 0 0 0 eth1
192.168.0.128 10.0.0.14 255.255.255.248 UG 0 0 0 eth1
192.168.0.136 10.0.0.9 255.255.255.248 UG 0 0 0 eth0
192.168.0.144 10.0.0.9 255.255.255.248 UG 0 0 0 eth0
192.168.0.152 10.0.0.9 255.255.255.248 UG 0 0 0 eth0
192.168.0.224 10.0.0.14 255.255.255.248 UG 0 0 0 eth1
192.168.0.232 10.0.0.14 255.255.255.248 UG 0 0 0 eth1
192.168.0.240 10.0.0.14 255.255.255.248 UG 0 0 0 eth1
192.168.0.248 10.0.0.14 255.255.255.248 UG 0 0 0 eth1
```

Fomos verificar a tabela e descobrimos que quando o destino era a sub rede onde a Teresa estava contida (192.168.0.128) o próximo nó era o n2, ou seja voltava para trás.

```
root@n1:/tmp/pycore.39931/n1.conf# route add -net 192.168.0.128 gw 10.0.0.9 netmask 255.255.255.248
root@n1:/tmp/pycore.39931/n1.conf# netstat -nr
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
10.0.0.0 10.0.0.9 255.255.255.252 UG 0 0 0 eth0
10.0.0.4 10.0.0.9 255.255.255.252 UG 0 0 0 eth0
10.0.0.8 0.0.0.0 255.255.255.252 U 0 0 0 eth0
10.0.0.12 0.0.0.0 255.255.255.252 U 0 0 0 eth1
10.0.0.16 10.0.0.9 255.255.255.252 UG 0 0 0 eth0
10.0.0.20 10.0.0.14 255.255.255.252 UG 0 0 0 eth1
10.0.0.24 10.0.0.14 255.255.255.252 UG 0 0 0 eth1
10.0.0.28 10.0.0.14 255.255.255.252 UG 0 0 0 eth1
172.0.0.0 10.0.0.14 255.0.0.0 UG 0 0 0 eth1
172.16.142.0 10.0.0.9 255.255.255.252 UG 0 0 0 eth0
172.16.142.4 10.0.0.9 255.255.255.252 UG 0 0 0 eth0
172.16.143.0 10.0.0.14 255.255.255.252 UG 0 0 0 eth1
172.16.143.4 10.0.0.14 255.255.255.252 UG 0 0 0 eth1
192.168.0.128 10.0.0.9 255.255.255.248 UG 0 0 0 eth0
192.168.0.136 10.0.0.9 255.255.255.248 UG 0 0 0 eth0
192.168.0.144 10.0.0.9 255.255.255.248 UG 0 0 0 eth0
192.168.0.152 10.0.0.9 255.255.255.248 UG 0 0 0 eth0
192.168.0.224 10.0.0.14 255.255.255.248 UG 0 0 0 eth1
192.168.0.232 10.0.0.14 255.255.255.248 UG 0 0 0 eth1
192.168.0.240 10.0.0.14 255.255.255.248 UG 0 0 0 eth1
192.168.0.248 10.0.0.14 255.255.255.248 UG 0 0 0 eth1
```

Corrigimos essa entrada da tabela, alterando o próximo nó como sendo n3 (10.0.0.9).

```
<4149/AfonsoHenriques.conf# traceroute 192.168.0.130
traceroute to 192.168.0.130 (192.168.0.130), 30 hops max, 60 byte packets
 1 192.168.0.225 (192.168.0.225) 0.039 ms 0.004 ms 0.003 ms
 2 172.16.143.1 (172.16.143.1) 0.014 ms 0.005 ms 0.005 ms
 3 10.0.0.29 (10.0.0.29) 0.019 ms 0.007 ms 0.006 ms
 4 10.0.0.25 (10.0.0.25) 0.015 ms 0.008 ms 0.008 ms
 5 10.0.0.13 (10.0.0.13) 0.018 ms 0.010 ms 0.009 ms
 6 10.0.0.17 (10.0.0.17) 0.024 ms !N 0.042 ms !N *
```

Ao voltar a usar o comando traceroute, desta vez o problema esta no n3.

```
root@n3:/tmp/pycore.34149/n3.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
10.0.0.0 10.0.0.5 255.255.255.252 UG 0 0 0 eth2
10.0.0.4 0.0.0.0 255.255.255.252 U 0 0 0 eth2
10.0.0.8 0.0.0.0 255.255.255.252 U 0 0 0 eth0
10.0.0.12 10.0.0.10 255.255.255.252 UG 0 0 0 eth0
10.0.0.16 0.0.0.0 255.255.255.252 U 0 0 0 eth1
10.0.0.20 10.0.0.18 255.255.255.252 UG 0 0 0 eth1
10.0.0.24 10.0.0.18 255.255.255.252 UG 0 0 0 eth1
10.0.0.28 10.0.0.10 255.255.255.252 UG 0 0 0 eth0
172.0.0.0 10.0.0.10 255.0.0.0 UG 0 0 0 eth0
172.16.142.0 10.0.0.5 255.255.255.252 UG 0 0 0 eth2
172.16.142.4 10.0.0.5 255.255.255.252 UG 0 0 0 eth2
172.16.143.0 10.0.0.18 255.255.255.252 UG 0 0 0 eth1
172.16.143.4 10.0.0.10 255.255.255.252 UG 0 0 0 eth0
192.168.0.136 10.0.0.5 255.255.255.248 UG 0 0 0 eth2
192.168.0.144 10.0.0.5 255.255.255.248 UG 0 0 0 eth2
192.168.0.152 10.0.0.5 255.255.255.248 UG 0 0 0 eth2
192.168.0.224 10.0.0.18 255.255.255.248 UG 0 0 0 eth1
192.168.0.232 10.0.0.10 255.255.255.248 UG 0 0 0 eth0
192.168.0.240 10.0.0.10 255.255.255.248 UG 0 0 0 eth0
192.168.0.248 10.0.0.10 255.255.255.248 UG 0 0 0 eth0
```

Ao consultar a tabela de routing de n3 verificou-se que não existia nenhuma entrada para quando o destino fosse a sub rede da Teresa (192.168.0.128)

```
root@n3:/tmp/pycore.34149/n3.conf# route add -net 192.168.0.128 gw 10.0.0.5 netmask 255.255.255.248
root@n3:/tmp/pycore.34149/n3.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
10.0.0.0 10.0.0.5 255.255.255.252 UG 0 0 0 eth2
10.0.0.4 0.0.0.0 255.255.255.252 U 0 0 0 eth2
10.0.0.8 0.0.0.0 255.255.255.252 U 0 0 0 eth0
10.0.0.12 10.0.0.10 255.255.255.252 UG 0 0 0 eth0
10.0.0.16 0.0.0.0 255.255.255.252 U 0 0 0 eth1
10.0.0.20 10.0.0.18 255.255.255.252 UG 0 0 0 eth1
10.0.0.24 10.0.0.18 255.255.255.252 UG 0 0 0 eth1
10.0.0.28 10.0.0.10 255.255.255.252 UG 0 0 0 eth0
172.0.0.0 10.0.0.10 255.0.0.0 UG 0 0 0 eth0
172.16.142.0 10.0.0.5 255.255.255.252 UG 0 0 0 eth2
172.16.142.4 10.0.0.5 255.255.255.252 UG 0 0 0 eth2
172.16.143.0 10.0.0.18 255.255.255.252 UG 0 0 0 eth1
172.16.143.4 10.0.0.10 255.255.255.252 UG 0 0 0 eth0
192.168.0.128 10.0.0.5 255.255.255.248 UG 0 0 0 eth2
192.168.0.136 10.0.0.5 255.255.255.248 UG 0 0 0 eth2
192.168.0.144 10.0.0.5 255.255.255.248 UG 0 0 0 eth2
192.168.0.152 10.0.0.5 255.255.255.248 UG 0 0 0 eth2
192.168.0.224 10.0.0.18 255.255.255.248 UG 0 0 0 eth1
192.168.0.232 10.0.0.10 255.255.255.248 UG 0 0 0 eth0
192.168.0.240 10.0.0.10 255.255.255.248 UG 0 0 0 eth0
192.168.0.248 10.0.0.10 255.255.255.248 UG 0 0 0 eth0
```

Adicionamos então uma entrada que permitisse que o n3 pudesse encaminhar pacotes para a sub rede da Teresa.

```
<4149/AfonsoHenriques.conf# traceroute 192.168.0.130
traceroute to 192.168.0.130 (192.168.0.130), 30 hops max, 60 byte packets
 1 192.168.0.225 (192.168.0.225) 0.034 ms 0.003 ms 0.003 ms
 2 172.16.143.1 (172.16.143.1) 0.013 ms 0.005 ms 0.004 ms
 3 10.0.0.29 (10.0.0.29) 0.016 ms 0.006 ms 0.007 ms
 4 10.0.0.25 (10.0.0.25) 0.016 ms 0.008 ms 0.007 ms
 5 10.0.0.13 (10.0.0.13) 0.019 ms 0.008 ms 0.008 ms
 6 10.0.0.17 (10.0.0.17) 0.025 ms 0.028 ms 0.011 ms
 7 10.0.0.5 (10.0.0.5) 0.039 ms 0.012 ms 0.012 ms
 8 10.0.0.1 (10.0.0.1) 0.020 ms 0.014 ms 0.016 ms
```

Voltamos a correr o traceroute e desta vez os pacotes chegavam até ao router CondadOnline, o que bastou para a resolução da alínea.

d)

Uma vez que o core da rede esteja a encaminhar corretamente os pacotes enviados por AfonsoHenriques, confira com o Wireshark se estes são recebidos por Teresa.

i)

Em caso afirmativo, porque é que continua a não existir conectividade entre D.Teresa e D.Afonso Henriques? Efetue as alterações necessárias para garantir que a conectividade é restabelecida e o confronto entre os dois é evitado.

RESPOSTA:

```
<core,34149/AfonsoHenriques.conf# ping 192.168.0.130
PING 192.168.0.130 (192.168.0.130) 56(84) bytes of data.
```

Fazendo ping do AfonsoHenriques para a Teresa nenhuma resposta é obtida, contudo, usando o Wireshark para capturar o tráfego, é possível perceber que os dados são recebidos pela Teresa, apenas existe um problema na resposta dada por ela.

2	0.237088914	192.168.0.193	224.0.0.5	OSPF	78	Hello Packet
3	0.715879153	192.168.0.226	192.168.0.130	ICMP	98	Echo (ping) request id=0x0071, seq=1/256, ttl=55 (reply in 4)
4	0.715892333	192.168.0.130	192.168.0.226	ICMP	98	Echo (ping) reply id=0x0071, seq=1/256, ttl=64 (request in...)
5	0.715897987	192.168.0.129	192.168.0.130	ICMP	126	Destination unreachable (Network unreachable)

Ao consultar a tabela de routing de RAGaliza, percebeu-se que não existia nenhuma entrada referente à sub rede do Condado Portucalense, onde se encontra AfonsoHenriques, daí não haver conectividade entre os dois dispositivos.

```
<Galiza.conf# route add -net 192.168.0.224 gw 172.16.142.1 netmask 255.255.255.248
```

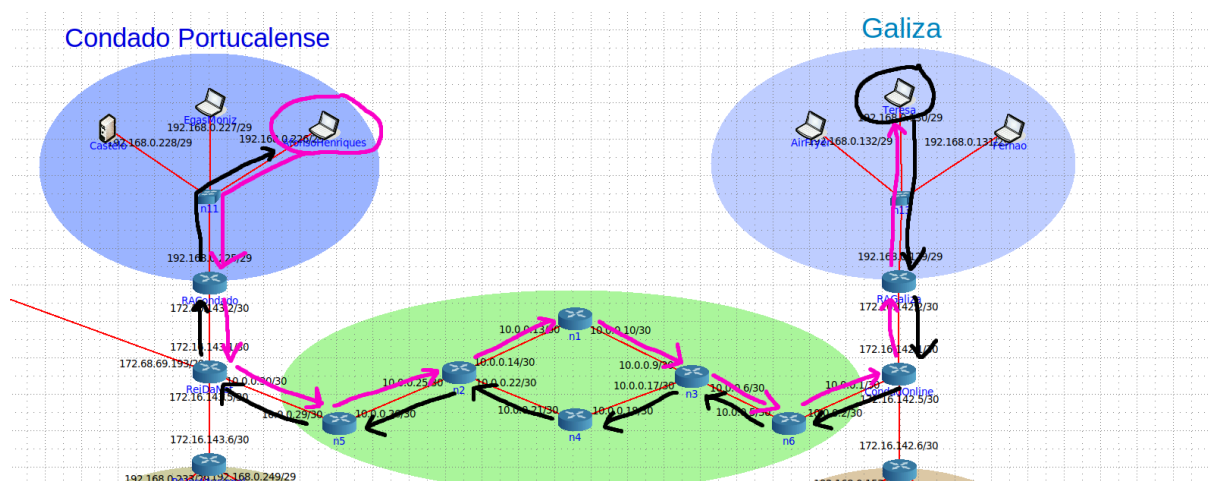
Depois de adicionada esta entrada na tabela de RAGaliza já foi possível obter resposta do comando ping quando enviado por AfonsoHenriques.

```
<core,34149/AfonsoHenriques.conf# ping 192.168.0.130
PING 192.168.0.130 (192.168.0.130) 56(84) bytes of data.
64 bytes from 192.168.0.130: icmp_seq=1 ttl=55 time=0.109 ms
64 bytes from 192.168.0.130: icmp_seq=2 ttl=55 time=0.123 ms
64 bytes from 192.168.0.130: icmp_seq=3 ttl=55 time=0.112 ms
64 bytes from 192.168.0.130: icmp_seq=4 ttl=55 time=0.123 ms
64 bytes from 192.168.0.130: icmp_seq=5 ttl=55 time=0.137 ms
64 bytes from 192.168.0.130: icmp_seq=6 ttl=55 time=0.122 ms
64 bytes from 192.168.0.130: icmp_seq=7 ttl=55 time=0.142 ms
64 bytes from 192.168.0.130: icmp_seq=8 ttl=55 time=0.114 ms
64 bytes from 192.168.0.130: icmp_seq=9 ttl=55 time=0.166 ms
64 bytes from 192.168.0.130: icmp_seq=10 ttl=55 time=0.111 ms
64 bytes from 192.168.0.130: icmp_seq=11 ttl=55 time=0.132 ms
64 bytes from 192.168.0.130: icmp_seq=12 ttl=55 time=0.177 ms
64 bytes from 192.168.0.130: icmp_seq=13 ttl=55 time=0.202 ms
64 bytes from 192.168.0.130: icmp_seq=14 ttl=55 time=0.111 ms
64 bytes from 192.168.0.130: icmp_seq=15 ttl=55 time=0.163 ms
64 bytes from 192.168.0.130: icmp_seq=16 ttl=55 time=0.208 ms
```

ii) As rotas dos pacotes ICMP echo reply são as mesmas, mas em sentido inverso, que as rotas dos pacotes ICMP echo request enviados entre AfonsoHenriques e Teresa? (Sugestão: analise as rotas nos dois sentidos com o traceroute). Mostre graficamente a rota seguida nos dois sentidos por esses pacotes ICMP.

RESPOSTA:

Não, as rotas são ligeiramente diferentes, uma opta por usar o router n2 (request) e a outra usa o n4 (reply). Como se pode ver nas imagem.



e)

Estando restabelecida a conectividade entre os dois hosts, obtenha a tabela de encaminhamento de n5 e foque-se na seguinte entrada:

**ip route 192.168.0.0 255.255.255.0 10.0.0.30**

Existe uma correspondência (match) nesta entrada para pacotes enviados para o polo Galiza? E para CDN? Caso seja essa a entrada utilizada para o encaminhamento, permitirá o funcionamento esperado do dispositivo? Ofereça uma explicação pela qual essa entrada é ou não utilizada.

RESPOSTA:

Não dá match em nenhum dos casos porque existem outras entradas que dirigem os pacotes para essas redes que são preferíveis, já que o IP encaminha sempre pelo longest match. E mesmo que enviasse, os pacotes nunca lá chegariam porque seriam enviados para o ReiDaNet, voltando para trás.

f)

Os endereços utilizados pelos quatro polos são endereços públicos ou privados? E os utilizados no core da rede/ISPs? Justifique convenientemente.

RESPOSTA:

**O alcance dos endereços privados está dividido em três blocos:**

- 192.168.0.0 – 192.168.255.255 /16
- 172.16.0.0 – 172.31.255.255 /12
- 10.0.0.0 – 10.255.255.255 /8

Como podemos observar na topologia do exercício, os endereços dentro dos quatro polos (Condado Portucalense, Institucional, Galiza e CDN), pertencem ao primeiro bloco de endereços privados. Já os endereços usados no Core e nos ISP's (ReiDaNet e CondadOnline) pertencem, respetivamente, ao segundo e terceiro bloco de endereços privados. Logo, os endereços utilizados nos quatro polos e no core da rede/ISP's são de facto endereços privados.

g)

Os switches localizados em cada um dos polos têm um endereço IP atribuído? Porquê?

RESPOSTA:

**Não, os switches em cada um dos polos não têm um endereço IP atribuído, e isto deve-se ao facto de estes pertencerem à layer 2, logo não reconhecem o protocolo IP. Estes funcionam à base de endereços MAC para enviar informação.**

### 2.3. Questão 3

Ao ver as fotos no CondadoGram, D. Teresa não ficou convencida com as novas alterações e ordena que Afonso Henriques vá arrumar o castelo. Inconformado, este decide planear um novo ataque, mas constata que o seu exército não só perde bastante tempo a decidir que direção tomar a cada salto como, por vezes, inclusivamente se perde.

a)

De modo a facilitar a travessia, elimine as rotas referentes a Galiza e CDN no dispositivo n6 e defina um esquema de sumarização de rotas (Supernetting) que permita o uso de apenas uma rota para ambos os polos. Confirme que a conectividade é mantida.

RESPOSTA:

**Após removidas as rotas referentes à Galiza e ao CDN é possível adicionar um esquema de sumarização de rotas:**

192.168.0.128/29 = 11000000.10101000.00000000.100 00000

192.168.0.136/29 = 11000000.10101000.00000000.100 01000

192.168.0.144/29 = 11000000.10101000.00000000.100 10000

192.168.0.152/29 = 11000000.10101000.00000000.100 11000

Como é possível verificar pelos bits que estão a negrito, apesar da máscara de rede ter em conta os primeiros 29 bits, é possível diminuir esta para 27 (já que os 3 primeiros bits do último octeto são comuns). Logo, na rota do dispositivo n6 é possível adicionar a entrada:

```
net 192.168.0.128 gw 10.0.0.1 netmask 255.255.255.224
```

Com isto, a tabela de encaminhamento do dispositivo n6 ficou:

```
root@n6:/tmp/pycore.36577/n6.conf# netstat -rn
Kernel IP routing table
Destination      Gateway         Genmask         Flags        MSS Window  irtt Iface
10.0.0.0          0.0.0.0         255.255.255.252 U             0 0        0 eth0
10.0.0.4          0.0.0.0         255.255.255.252 U             0 0        0 eth1
10.0.0.8          10.0.0.6        255.255.255.252 UG            0 0        0 eth1
10.0.0.12         10.0.0.6        255.255.255.252 UG            0 0        0 eth1
10.0.0.16         10.0.0.6        255.255.255.252 UG            0 0        0 eth1
10.0.0.20         10.0.0.6        255.255.255.252 UG            0 0        0 eth1
10.0.0.24         10.0.0.6        255.255.255.252 UG            0 0        0 eth1
10.0.0.28         10.0.0.6        255.255.255.252 UG            0 0        0 eth1
172.0.0.0         10.0.0.6        255.0.0.0       UG            0 0        0 eth1
172.16.142.0      10.0.0.1        255.255.255.252 UG            0 0        0 eth0
172.16.142.4      10.0.0.1        255.255.255.252 UG            0 0        0 eth0
172.16.143.0      10.0.0.6        255.255.255.252 UG            0 0        0 eth1
172.16.143.4      10.0.0.6        255.255.255.252 UG            0 0        0 eth1
192.168.0.128     10.0.0.1        255.255.255.224 UG            0 0        0 eth0
192.168.0.224     10.0.0.6        255.255.255.248 UG            0 0        0 eth1
192.168.0.232     10.0.0.6        255.255.255.248 UG            0 0        0 eth1
192.168.0.240     10.0.0.6        255.255.255.248 UG            0 0        0 eth1
192.168.0.248     10.0.0.6        255.255.255.248 UG            0 0        0 eth1
```

Assim, é possível encaminhar pacotes para tanto para hosts pertencentes à Galiza, como ao CDN à mesma:

```
root@n6:/tmp/pycore.36577/n6.conf# ping 192.168.0.154
PING 192.168.0.154 (192.168.0.154) 56(84) bytes of data.
64 bytes from 192.168.0.154: icmp_seq=1 ttl=62 time=0.045 ms
^Z
[1]+  Stopped                  ping 192.168.0.154
root@n6:/tmp/pycore.36577/n6.conf# ping 192.168.0.146
PING 192.168.0.146 (192.168.0.146) 56(84) bytes of data.
64 bytes from 192.168.0.146: icmp_seq=1 ttl=62 time=0.045 ms
^Z
[2]+  Stopped                  ping 192.168.0.146
root@n6:/tmp/pycore.36577/n6.conf# ping 192.168.0.132
PING 192.168.0.132 (192.168.0.132) 56(84) bytes of data.
64 bytes from 192.168.0.132: icmp_seq=1 ttl=62 time=0.036 ms
^Z
[3]+  Stopped                  ping 192.168.0.132
root@n6:/tmp/pycore.36577/n6.conf# ping 192.168.0.130
PING 192.168.0.130 (192.168.0.130) 56(84) bytes of data.
64 bytes from 192.168.0.130: icmp_seq=1 ttl=62 time=0.044 ms
^Z
[4]+  Stopped                  ping 192.168.0.130
```

b)

Repita o processo descrito na alínea anterior para CondadoPortugalense e Institucional, também no dispositivo n6.

RESPOSTA:

Após removidas as rotas referentes à CondadoPortugalense e ao Institucional é possível adicionar um esquema de sumarização de rotas:



192.168.0.224/29 = **11000000.10101000.00000000.111** 00000

192.168.0.232/29 = **11000000.10101000.00000000.111** 01000

192.168.0.240/29 = **11000000.10101000.00000000.111** 10000

192.168.0.248/29 = **11000000.10101000.00000000.111** 11000

Como é possível verificar pelos bits que estão a negrito, apesar da máscara de rede ter em conta os primeiros 29 bits, é possível diminuir esta para 27(já que os 3 primeiros bits do último octeto são comuns). Logo, na rota do dispositivo n6 é possível adicionar a entrada:

net 192.168.0.224 gw 10.0.0.6 mask 255.255.255.224

Com isto, a tabela de encaminhamento do dispositivo n6 ficou:

```
root@n6:/tmp/pycore.36577/n6.conf# route add -net 192.168.0.224 gw 10.0.0.6 netmask 255.255.255.224
root@n6:/tmp/pycore.36577/n6.conf# netstat -rn
Kernel IP routing table
Destination    Gateway         Genmask         Flags   MSS Window  irtt Iface
10.0.0.0        0.0.0.0         255.255.255.252 U        0 0          0 eth0
10.0.0.4        0.0.0.0         255.255.255.252 U        0 0          0 eth1
10.0.0.8        10.0.0.6        255.255.255.252 UG       0 0          0 eth1
10.0.0.12       10.0.0.6        255.255.255.252 UG       0 0          0 eth1
10.0.0.16       10.0.0.6        255.255.255.252 UG       0 0          0 eth1
10.0.0.20       10.0.0.6        255.255.255.252 UG       0 0          0 eth1
10.0.0.24       10.0.0.6        255.255.255.252 UG       0 0          0 eth1
10.0.0.28       10.0.0.6        255.255.255.252 UG       0 0          0 eth1
172.0.0.0       10.0.0.6        255.0.0.0       UG       0 0          0 eth1
172.16.142.0    10.0.0.1        255.255.255.252 UG       0 0          0 eth0
172.16.142.4    10.0.0.1        255.255.255.252 UG       0 0          0 eth0
172.16.143.0    10.0.0.6        255.255.255.252 UG       0 0          0 eth1
172.16.143.4    10.0.0.6        255.255.255.252 UG       0 0          0 eth1
192.168.0.128   10.0.0.1        255.255.255.224 UG       0 0          0 eth0
192.168.0.224   10.0.0.6        255.255.255.224 UG       0 0          0 eth1
```

Assim, é possível encaminhar pacotes para tanto para hosts pertencentes à CondadoPortucalense, como ao Institucional à mesma:

```
root@n6:/tmp/pycore.36577/n6.conf# ping 192.168.0.227
PING 192.168.0.227 (192.168.0.227) 56(84) bytes of data.
64 bytes from 192.168.0.227: icmp_seq=1 ttl=58 time=0.068 ms
^Z
[5]+  Stopped                  ping 192.168.0.227
root@n6:/tmp/pycore.36577/n6.conf# ping 192.168.0.228
PING 192.168.0.228 (192.168.0.228) 56(84) bytes of data.
64 bytes from 192.168.0.228: icmp_seq=1 ttl=58 time=0.100 ms
^Z
[6]+  Stopped                  ping 192.168.0.228
root@n6:/tmp/pycore.36577/n6.conf# ping 192.168.0.242
PING 192.168.0.242 (192.168.0.242) 56(84) bytes of data.
64 bytes from 192.168.0.242: icmp_seq=1 ttl=58 time=0.081 ms
^Z
[7]+  Stopped                  ping 192.168.0.242
root@n6:/tmp/pycore.36577/n6.conf# ping 192.168.0.234
PING 192.168.0.234 (192.168.0.234) 56(84) bytes of data.
64 bytes from 192.168.0.234: icmp_seq=1 ttl=58 time=0.075 ms
^Z
[8]+  Stopped                  ping 192.168.0.234
```

c)

Comente os aspetos positivos e negativos do uso do Supernetting.

RESPOSTA:

**Aspetos Positivos:**

- Redução do tamanho da tabela de routing – Ao combinar várias redes numa única rota, diminui-se a quantidade de entradas na tabela de routing, economizando a memória e o processamento nos routers
- Melhoria do encaminhamento – Como há menos rotas a serem analisadas, o routing torna-se mais rápido e eficiente

**Aspetos Negativos:**

- Menor controlo do tráfego – Como várias redes são agrupadas, pode ser difícil aplicar políticas específicas para cada sub-rede individualmente.
- Legibilidade das rotas - Com o agrupamento das rotas torna-se mais difícil ler e analisar as rotas, ao contrário do que acontece quando estas estão especificadas