



## **TP2 - Protocolo IP**

### **Redes de Computadores - Grupo 12**

Ana Teresa Gíão Gomes A89536  
André Carvalho da Cunha Martins A89586  
Pedro Miguel de Soveral Pacheco Barbosa A89529



**Fig. 1.** A89536



**Fig. 2.** A89586



**Fig. 3.** A89529

25 de novembro de 2020

# Parte 1

André Martins, Ana Teresa Gomes, and Pedro Barbosa

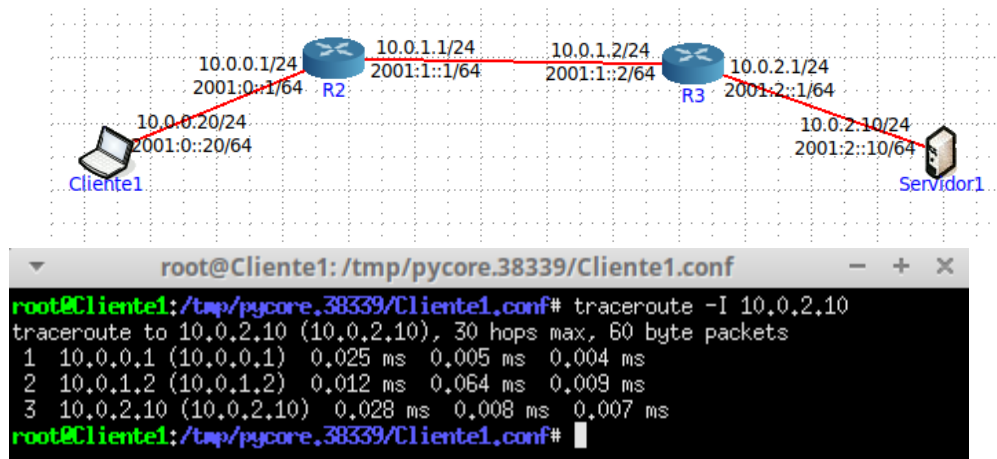
University of Minho, Department of Informatics, 4710-057 Braga, Portugal  
e-mail: {a89586,a89536,a89529}@alunos.uminho.pt

## 1 Pergunta 1

Prepare uma topologia CORE para verificar o comportamento do traceroute. Ligue um host (PC) Cliente1 a um router R2; o router R2 a um router R3, que por sua vez, se liga a um host (servidor) Servidor1. (Note que pode não existir conectividade IP imediata entre o Cliente1 e o Servidor1 até que o anúncio de rotas estabilize)

### 1.1 Pergunta 1 A)

Active o wireshark ou o tcpdump no Cliente1. Numa shell do Cliente1, execute o comando traceroute -I para o endereço IP do Servidor1.



### 1.2 Pergunta 1 B)

Registe e analise o tráfego ICMP enviado pelo Cliente1 e o tráfego ICMP recebido como resposta. Comente os resultados face ao comportamento esperado.

14994773	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x003f, seq=1/256, ttl=1 (no respon
5014754	10.0.0.1	10.0.0.20	ICMP	102 Time-to-live exceeded (time to live exceeded in transit)	
5021000	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x003f, seq=2/512, ttl=1 (no respon
5029180	10.0.0.1	10.0.0.20	ICMP	102 Time-to-live exceeded (time to live exceeded in transit)	
5041603	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x003f, seq=3/768, ttl=1 (no respon
5044993	10.0.0.1	10.0.0.20	ICMP	102 Time-to-live exceeded (time to live exceeded in transit)	
5067325	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x003f, seq=4/1024, ttl=2 (no respo
5080585	10.0.1.2	10.0.0.20	ICMP	102 Time-to-live exceeded (time to live exceeded in transit)	
5083903	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x003f, seq=5/1280, ttl=2 (no respo
5088242	10.0.1.2	10.0.0.20	ICMP	102 Time-to-live exceeded (time to live exceeded in transit)	
5091931	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x003f, seq=6/1536, ttl=2 (no respo
5096757	10.0.1.2	10.0.0.20	ICMP	102 Time-to-live exceeded (time to live exceeded in transit)	
5100415	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x003f, seq=7/1792, ttl=3 (reply in
5113739	10.0.2.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x003f, seq=7/1792, ttl=62 (request

Fig. 1. Tráfego ICMP

A primeira tentativa de comunicação entre o Cliente1 e o Servidor1 é falhada visto que o TTL é 1. O pacote ao chegar ao R2 será descartado, visto que o TTL será decrementado

e ficará igual a 0. Nas comunicações seguintes, o TTL é incrementado até atingir o valor de TTL = 3, que é o valor mínimo para a comunicação entre o Cliente1 e o Servidor1 ser bem sucedida.

### 1.3 Pergunta 1 C)

Qual deve ser o valor inicial mínimo do campo TTL para alcançar o Servidor1? Verifique na prática que a sua resposta está correta.

Como vemos na Figura 2, e como foi explicado acima, o TTL inicial mínimo tem que ser igual a 3

5100415	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x003f, seq=7/1792, ttl=3 (reply I
5113739	10.0.2.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x003f, seq=7/1792, ttl=62 (reques

**Fig. 2.** Primeiro pacote a chegar com sucesso ao Servidor1

### 1.4 Pergunta 1 D)

Calcule o valor médio do tempo de ida-e-volta (Round-TripTime) obtido?

O valor médio do tempo de ida e volta (Round-TripTime) será a média dos 3 valores obtidos no traceroute.

$$\frac{0.028 + 0.008 + 0.007}{3} = 0.014$$

O valor está expresso em milissegundos

```

root@Cliente1: /tmp/pycore.38339/Cliente1.conf
root@Cliente1:/tmp/pycore.38339/Cliente1.conf# traceroute -I 10.0.2.10
traceroute to 10.0.2.10 (10.0.2.10), 30 hops max, 60 byte packets
 1 10.0.0.1 (10.0.0.1) 0.025 ms 0.005 ms 0.004 ms
 2 10.0.1.2 (10.0.1.2) 0.012 ms 0.064 ms 0.009 ms
 3 10.0.2.10 (10.0.2.10) 0.028 ms 0.008 ms 0.007 ms
root@Cliente1:/tmp/pycore.38339/Cliente1.conf#

```

**Fig. 3.** Valores obtidos através do traceroute

## 2 Pergunta 2

Pretende-se agora usar o traceroute na sua máquina nativa, e gerar de datagramas IP de diferentes tamanhos.

Selecione a primeira mensagem ICMP capturada (referente a (i) tamanho por defeito)

7 0.861830	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=281/6401, ttl=255 (reply in 8)
8 0.864482	193.136.9.240	172.26.21.229	ICMP	70 Echo (ping) reply	id=0x0001, seq=281/6401, ttl=61 (request in 7)
9 0.901594	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=282/6657, ttl=1 (no response found!)
10 0.901950	172.26.21.229	193.136.9.240	ICMP	70 Time-to-live exceeded (time to live exceeded in transit)	
11 0.940195	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=283/6913, ttl=2 (no response found!)
12 0.942101	172.26.21.229	193.136.9.240	ICMP	70 Time-to-live exceeded (time to live exceeded in transit)	
13 0.979920	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=284/7159, ttl=3 (no response found!)
14 0.981892	172.26.21.229	193.136.9.240	ICMP	70 Time-to-live exceeded (time to live exceeded in transit)	
15 1.018225	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=285/7425, ttl=4 (reply in 16)
16 1.019949	193.136.9.240	172.26.21.229	ICMP	70 Echo (ping) reply	id=0x0001, seq=285/7425, ttl=61 (request in 15)
17 1.361610	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=286/7681, ttl=255 (reply in 18)
18 1.365568	193.136.9.240	172.26.21.229	ICMP	70 Echo (ping) reply	id=0x0001, seq=286/7681, ttl=61 (request in 17)
19 1.414314	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=287/7937, ttl=1 (no response found!)
20 1.415681	172.26.21.229	193.136.9.240	ICMP	70 Time-to-live exceeded (time to live exceeded in transit)	
21 1.466656	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=288/8193, ttl=2 (no response found!)
22 1.466623	172.26.21.229	193.136.9.240	ICMP	70 Time-to-live exceeded (time to live exceeded in transit)	
23 1.514925	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=289/8449, ttl=3 (no response found!)

Fig. 4. Captura efetuado com packet size = 56 bytes

### 2.1 Pergunta 2 A)

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

9 0.901594	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=282/6657, ttl=1 (no response found!)
10 0.901950	172.26.21.229	193.136.9.240	ICMP	70 Time-to-live exceeded (time to live exceeded in transit)	
11 0.940195	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=283/6913, ttl=2 (no response found!)

Fig. 5. IP da interface ativa

Como podemos ver na imagem, o IP da interface ativa do nosso computador é 172.26.21.229

### 2.2 Pergunta 2 B)

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

Internet Protocol Version 4, Src: 172.26.21.229, 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: 56
Identification: 0x3e3b (15931)
> Flags: 0x00
Fragment Offset: 0
Time to Live: 255
Protocol: ICMP (1)
Header Checksum: 0xF811 [validation disabled]
[Header checksum status: Unverified]
Source Address: 172.26.21.229
Destination Address: 193.136.9.240

Fig. 6. Valor do campo protocolo

Como podemos observar na figura, o valor do campo protocolo é igual a 1, representando este o protocolo ICMP - Internet Control Message Protocol

### 2.3 Pergunta 2 C)

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

```
Internet Protocol Version 4, Src: 172.26.21.229, 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: 56
Identification: 0x3e3b (15931)
Flags: 0x00
0... .... = Reserved bit: Not set
..0... .... = Don't fragment: Not set
...0... .... = More fragments: Not set
Fragment Offset: 0
Time to Live: 255
Protocol: ICMP (1)
Header Checksum: 0xf011 [validation disabled]
[Header checksum status: Unverified]
Source Address: 172.26.21.229
Destination Address: 193.136.9.240
```

Fig. 7. Header Length

Como é possível observar, o cabeçalho IP(v4) tem 20 bytes.

```
Internet Protocol Version 4, Src: 172.26.21.229, 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: 56
Identification: 0x3e3b (15931)
Flags: 0x00
0... .... = Reserved bit: Not set
..0... .... = Don't fragment: Not set
...0... .... = More fragments: Not set
Fragment Offset: 0
Time to Live: 255
Protocol: ICMP (1)
Header Checksum: 0xf011 [validation disabled]
[Header checksum status: Unverified]
Source Address: 172.26.21.229
Destination Address: 193.136.9.240
```

Fig. 8. Total Length

O campo de dados (payload) do datagrama tem 36 bytes. Obtém-se este valor ao fazer a diferença entre o tamanho total (56 bytes) e o tamanho do cabeçalho do IP(v4) (20 bytes).

### 2.4 Pergunta 2 D)

O datagrama IP foi fragmentado? Justifique.

```
Flags: 0x00
0... .... = Reserved bit: Not set
..0... .... = Don't fragment: Not set
...0... .... = More fragments: Not set
Fragment Offset: 0
Time to Live: 255
Protocol: ICMP (1)
```

Fig. 9. Flags

Como a figura mostra, o valor das flags e do offset é igual a 0, logo, pode-se concluir que o pacote não foi fragmentado.

## 2.5 Pergunta 2 E)

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 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.

7	0.861830	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=281/6401, ttl=255 (reply in 8)
9	0.901594	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=282/6657, ttl=1 (no response found)
11	0.940195	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=283/6913, ttl=2 (no response found)
13	0.979926	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=284/7169, ttl=3 (no response found)
15	1.018225	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=285/7425, ttl=4 (reply in 16)
17	1.363610	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=286/7681, ttl=255 (reply in 18)
19	1.414314	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=287/7937, ttl=1 (no response found)
21	1.464656	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=288/8193, ttl=2 (no response found)
23	1.514925	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=289/8449, ttl=3 (no response found)
25	1.565307	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=290/8705, ttl=4 (reply in 26)

Fig. 10. Pacotes ordenados de acordo com o IP fonte

7	0.861830	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=281/6401, ttl=255 (reply in 8)
9	0.901594	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=282/6657, ttl=1 (no response found)
11	0.940195	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=283/6913, ttl=2 (no response found)
13	0.979926	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=284/7169, ttl=3 (no response found)
15	1.018225	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=285/7425, ttl=4 (reply in 16)
17	1.363610	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=286/7681, ttl=255 (reply in 18)
19	1.414314	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=287/7937, ttl=1 (no response found)
21	1.464656	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=288/8193, ttl=2 (no response found)
23	1.514925	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=289/8449, ttl=3 (no response found)
25	1.565307	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=290/8705, ttl=4 (reply in 26)
27	5.000293	172.26.21.229	93.184.216.14	TCP	66 49955 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 S=256 SACK_PERM=1	

> Frame 9: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface vDevice\VF\_51B3331B-0084-4054-9C1C-81A8B4A00AF3, id 0  
 > Ethernet II, Src: IntelCor\_Bd-F2-61 (ac:ed:5c:bd:f2:61), Dst: ComdatEnt\_ff:94:00 (00:00:03:ff:94:00)  
 ▼ Internet Protocol Version 4, Src: 172.26.21.229, 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: 56  
 Identification: 0x3e3d (15932)  
 ▼ Flags: 0x00  
 0... .... = Reserved bit: Not set  
 .0... .... = Don't fragment: Not set  
 .0... .... = More fragments: Not set  
 Fragment Offset: 0  
 > Time to Live: 1  
 Protocol: ICMP (1)  
 Header Checksum: 0xe11 [validation disabled]  
 [Header checksum status: Unverified]  
 Source Address: 172.26.21.229  
 Destination Address: 193.136.9.240

Fig. 11. Pacote 1

7	0.861830	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=281/6401, ttl=255 (reply in 8)
9	0.901594	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=282/6657, ttl=1 (no response found)
11	0.940195	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=283/6913, ttl=2 (no response found)
13	0.979926	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=284/7169, ttl=3 (no response found)
15	1.018225	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=285/7425, ttl=4 (reply in 16)
17	1.363610	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=286/7681, ttl=255 (reply in 18)
19	1.414314	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=287/7937, ttl=1 (no response found)
21	1.464656	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=288/8193, ttl=2 (no response found)
23	1.514925	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=289/8449, ttl=3 (no response found)
25	1.565307	172.26.21.229	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=290/8705, ttl=4 (reply in 26)
27	17.5.000293	172.26.21.229	93.184.216.14	TCP	66 49955 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 S=256 SACK_PERM=1	

> Frame 11: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface vDevice\VF\_51B3331B-0084-4054-9C1C-81A8B4A00AF3, id 0  
 > Ethernet II, Src: IntelCor\_Bd-F2-61 (ac:ed:5c:bd:f2:61), Dst: ComdatEnt\_ff:94:00 (00:00:03:ff:94:00)  
 > Internet Protocol Version 4, Src: 172.26.21.229, 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: 56  
 Identification: 0x3e3d (15933)  
 > Flags: 0x00  
 0... .... = Reserved bit: Not set  
 .0... .... = Don't fragment: Not set  
 .0... .... = More fragments: Not set  
 Fragment Offset: 0  
 > Time to Live: 2

Fig. 12. Pacote 2

Como é comprovado pelas imagens acima, os campos TTL e Identification variam ao longo do tempo

## 2.6 Pergunta 2 F)

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

Como é possível observar nas figuras 11 e 12, que são imagens de 2 pacotes consecutivos, o campo da Identificação é incrementado a cada pacote enviado. À semelhança do campo Identificação, o TTL também é incrementado a cada pacote enviado

## 2.7 Pergunta 2 G)

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?

10.0.903290	172.26.254.254	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
12.0.942101	172.16.2.1	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
14.0.981862	172.16.115.252	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)

Fig. 13. Pacotes ordenados pelo destino

10.0.903290	172.26.254.254	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
12.0.942101	172.16.2.1	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
14.0.981862	172.16.115.252	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
16.1.019949	193.136.9.240	172.26.21.229	ICMP	70 Echo (ping) reply id=0x0001, seq=285/7425, ttl=61 (request in 15)
18.3.365568	193.136.9.240	172.26.21.229	ICMP	70 Echo (ping) reply id=0x0001, seq=286/7681, ttl=61 (request in 17)
20.3.416631	172.26.254.254	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
22.3.466623	172.16.2.1	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
24.3.517368	172.16.115.252	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
26.3.567399	193.136.9.240	172.26.21.229	ICMP	70 Echo (ping) reply id=0x0001, seq=290/8705, ttl=61 (request in 25)
28.5.127230	93.184.216.34	172.26.21.229	TCP	66.80 → 49955 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1250 SACK_PERM=1 WS=512
31.5.254484	93.184.216.34	172.26.21.229	TCP	54.80 → 49955 [FIN, ACK] Seq=1 Ack=2 Win=65536 Len=0
34.5.866245	193.136.9.240	172.26.21.229	ICMP	70 Echo (ping) reply id=0x0001, seq=291/8961, ttl=61 (request in 33)
36.5.916624	172.26.254.254	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
38.5.967694	172.16.2.1	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)

Frame 10: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface \Device\NPF\_{51B3331B-0DB4-4654-9C1C-81ABBA0A0AF3}, id 0  
Ethernet II, Src: CondaInt\_ff:94:00 (00:00:03:ff:94:00), Dst: IntelCor\_Bd:f2:61 (accd:5c:b4:f2:61)  
Internet Protocol Version 4, Src: 172.26.254.254, Dst: 172.26.21.229  
0100 .... = Version: 4  
.... 0101 = Header Length: 20 bytes (5)  
> Differentiated Services Field: 0x00 (DSCP: CS6, ECN: Not-ECT)  
Total Length: 56  
Identification: 0x495c (18780)  
> Flags: 0x00  
Fragment Offset: 0  
Time to Live: 255  
Protocol: ICMP (1)  
Header Checksum: 0xb090 (validation disabled)  
[Header checksum status: Unverified]  
Source Address: 172.26.254.254  
Destination Address: 172.26.21.229  
Internet Control Message Protocol

Fig. 14. Primeiros 3 pacotes

10.0.903290	172.26.254.254	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
12.0.942101	172.16.2.1	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
14.0.981862	172.16.115.252	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
16.1.019949	193.136.9.240	172.26.21.229	ICMP	70 Echo (ping) reply id=0x0001, seq=285/7425, ttl=61 (request in 15)
18.3.365568	193.136.9.240	172.26.21.229	ICMP	70 Echo (ping) reply id=0x0001, seq=286/7681, ttl=61 (request in 17)
20.3.416631	172.26.254.254	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
22.3.466623	172.16.2.1	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
24.3.517368	172.16.115.252	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
26.3.567399	193.136.9.240	172.26.21.229	ICMP	70 Echo (ping) reply id=0x0001, seq=290/8705, ttl=61 (request in 25)
28.5.127230	93.184.216.34	172.26.21.229	TCP	66.80 → 49955 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1250 SACK_PERM=1 WS=512
31.5.254484	93.184.216.34	172.26.21.229	TCP	54.80 → 49955 [FIN, ACK] Seq=1 Ack=2 Win=65536 Len=0
34.5.866245	193.136.9.240	172.26.21.229	ICMP	70 Echo (ping) reply id=0x0001, seq=291/8961, ttl=61 (request in 33)
36.5.916624	172.26.254.254	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
38.5.967694	172.16.2.1	172.26.21.229	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)

Frame 12: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface \Device\NPF\_{51B3331B-0DB4-4654-9C1C-81ABBA0A0AF3}, id 0  
Ethernet II, Src: CondaInt\_ff:94:00 (00:00:03:ff:94:00), Dst: IntelCor\_Bd:f2:61 (accd:5c:b4:f2:61)  
Internet Protocol Version 4, Src: 172.16.2.1, Dst: 172.26.21.229  
0100 .... = Version: 4  
.... 0101 = Header Length: 20 bytes (5)  
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)  
Total Length: 56  
Identification: 0x5adb (23259)  
> Flags: 0x00  
Fragment Offset: 0  
Time to Live: 254  
Protocol: ICMP (1)  
Header Checksum: 0xf1d8 (validation disabled)  
[Header checksum status: Unverified]  
Source Address: 172.16.2.1  
Destination Address: 172.26.21.229  
Internet Control Message Protocol

Fig. 15. Segundos 3 pacotes

18 0.001250	172.26.254.254	172.26.21.229	ICMP	70 Time-to-live exceeded (time to live exceeded in transit)
12 0.042191	172.26.2.1	172.26.21.229	ICMP	70 Time-to-live exceeded (time to live exceeded in transit)
14 0.081802	172.26.115.252	172.26.21.229	ICMP	70 Time-to-live exceeded (time to live exceeded in transit)
16 1.010949	193.136.9.240	172.26.21.229	ICMP	70 Echo (ping) reply id=0x0001, seq=285/7425, ttl=61 (request in 15)
18 1.365568	193.136.9.240	172.26.21.229	ICMP	70 Echo (ping) reply id=0x0001, seq=286/7368, ttl=61 (request in 17)
20 1.410631	172.26.254.254	172.26.21.229	ICMP	70 Time-to-live exceeded (time to live exceeded in transit)
22 1.466623	172.26.2.1	172.26.21.229	ICMP	70 Time-to-live exceeded (time to live exceeded in transit)
24 1.511704	172.26.115.252	172.26.21.229	ICMP	70 Time-to-live exceeded (time to live exceeded in transit)
26 1.567399	193.136.9.240	172.26.21.229	ICMP	70 Echo (ping) reply id=0x0001, seq=290/8705, ttl=61 (request in 25)
28 5.127230	93.184.216.34	172.26.21.229	TCP	66 80 → 49955 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1250 SACK_PERM=1 WS=512
31 5.254404	93.184.216.34	172.26.21.229	TCP	54 80 → 49955 [FIN, ACK] Seq=1 Ack=2 Win=65536 Len=0
34 5.866245	193.136.9.240	172.26.21.229	ICMP	70 Echo (ping) reply id=0x0001, seq=291/8961, ttl=61 (request in 33)
36 5.916624	172.26.254.254	172.26.21.229	ICMP	70 Time-to-live exceeded (time to live exceeded in transit)
38 5.967694	172.26.2.1	172.26.21.229	ICMP	70 Time-to-live exceeded (time to live exceeded in transit)
4 0.000000	172.26.2.1	172.26.21.229	ICMP	70 Time-to-live exceeded (time to live exceeded in transit)

Frame 14: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface \Device\NPF>{51833318-0D84-4654-9C1C-81ABBA000F3}, id 0	
Ethernet II, Src: Comadant_ff:94:00 (00:00:03:ff:94:00), Dst: IntelCor_Bd:f2:61 (ac:ed:5c:bd:f2:61)	
Internet Protocol Version 4, Src: 172.16.115.252, Dst: 172.26.21.229	
..... -> Version: 4	
..... 0101 = Header Length: 20 bytes (5)	
..... Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)	
Total Length: 56	
Identification: 0x38a5 (14501)	
Flags: 0x00	
Fragment Offset: 0	
Time to Live: 253	
Protocol: ICMP (1)	
Header Checksum: 0xa313 [validation disabled]	
[Header checksum status: Unverified]	
Source Address: 172.16.115.252	
Destination Address: 172.26.21.229	
Internet Control Message Protocol	

Fig. 16. Últimos 3 pacotes

As figuras 14, 15 e 16 mostram os valores do TTL. O valor do TTL apresentado na figura 14 é igual a 255. Contudo, este valor não se mantém constante(Por cada salto efetuado, é decrementado). Isto acontece dado que, a cada mensagem de erro enviada, vamo-nos aproximando da origem do nosso pacote, logo o TTL vai diminuindo neste caminho de regresso até à fonte do pacote (Nas imagens 15 e 16, o TTL apresentado já é de 254 e 253, respetivamente).

### 3 Pergunta 3

Pretende-se agora analisar a fragmentação de pacotes IP.

5 0.028779	193.136.9.240	172.26.21.229	IPv4	266 Fragmented IP protocol (proto:ICMP 1, off=2960, ID=4519) [Reassembled in #6]
6 0.034116	193.136.9.240	172.26.21.229	ICMP	1514 Echo (ping) reply id=0x0001, seq=121/3070, ttl=61 (request in 3)
7 0.033942	172.26.21.229	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto:ICMP 1, off=0, ID=4842) [Reassembled in #9]
8 0.033942	172.26.21.229	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto:ICMP 1, off=1480, ID=4841) [Reassembled in #9]
9 0.033942	172.26.21.229	193.136.9.240	ICMP	266 Echo (ping) request id=0x0001, seq=122/31232, ttl=1 (no response found)
10 0.048483	172.26.254.254	172.26.21.229	ICMP	70 Time-to-live exceeded (time to live exceeded in transit)
11 0.076834	172.26.21.229	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto:ICMP 1, off=0, ID=4842) [Reassembled in #13]
12 0.076834	172.26.21.229	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto:ICMP 1, off=1480, ID=4842) [Reassembled in #13]
13 0.076834	172.26.21.229	193.136.9.240	ICMP	266 Echo (ping) request id=0x0001, seq=123/31488, ttl=2 (no response found)
14 0.080250	172.26.2.1	172.26.21.229	ICMP	70 Time-to-live exceeded (time to live exceeded in transit)
15 0.115283	172.26.21.229	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto:ICMP 1, off=0, ID=4843) [Reassembled in #17]
16 0.115283	172.26.21.229	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto:ICMP 1, off=1480, ID=4843) [Reassembled in #17]
17 0.115283	172.26.21.229	193.136.9.240	ICMP	266 Echo (ping) request id=0x0001, seq=124/31744, ttl=3 (no response found)
18 0.124067	172.16.115.252	172.26.21.229	ICMP	70 Time-to-live exceeded (time to live exceeded in transit)
19 0.155326	172.26.21.229	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto:ICMP 1, off=0, ID=4844) [Reassembled in #21]
20 0.155326	172.26.21.229	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto:ICMP 1, off=1480, ID=4844) [Reassembled in #21]

Fig. 17. Pacotes com fragmentação



### 3.1 Pergunta 3 A)

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

7 0.0.0.0	172.26.21.229	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, id=041) [Reassembled in #0]
8 0.0.0.0	172.26.21.229	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=1408, id=041) [Reassembled in #0]
9 0.0.0.0	172.26.21.229	193.136.9.240	ICMP	200 Echo (ping) request id=0x0001, seq=122/31232, ttl=1 (no response found)

Fig. 18. Primeiro pacote e fragmentação

O pacote que tentamos enviar tem o tamanho de 3212 bytes. Uma que a MTU utilizada apenas consegue enviar pacotes com o tamanho de 1500 bytes, foi necessário fragmentar o nosso pacote para que este pudesse ser enviado

### 3.2 Pergunta 3 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?

```
Frame 7: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface \Device\NPF_{51B3331B-8064-4054-9C1C-B1A8B4A00A73}, id 0
Ethernet II, Src: IntelCor_8d:f2:61 (acd:5c:8d:f2:61), Dst: CondaInt_ff:94:00 (00:00:03:ff:94:00)
Internet Protocol Version 4, Src: 172.26.21.229, Dst: 193.136.9.240
0200 .... = Version: 4
.... 0101 = Header Length: 20 bytes (5)
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 1500
Identification: 0x041 (57409)
Flags: 0x20, More fragments
0... .... = Reserved bit: Not set
0... .... = Don't fragment: Not set
..1. .... = More fragments: Set
Fragment Offset: 0
> Time to live: 1
```

Fig. 19. Flags

O primeiro fragmento do datagrama é identificado quando o valor do offset é igual a 0 e o valor da flag More Fragments é igual a 1. Como é possível analisar na imagem 19, os valores das flags offset e More fragments são, respetivamente, 0 e 1, logo este é o primeiro fragmento. Como também foi referido em cima, a MTU utilizada só consegue enviar pacotes de 1500 bytes, logo o tamanho deste datagrama é de 1500 bytes.

### 3.3 Pergunta 3 C)

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

```
Frame 8: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface \Device\NPF_{51B3331B-0084-4654-9C1C-81A8B4A00AF3}, id 0
Ethernet II, Src: IntelCor_Bd:f2:61 (ac:ed:5c:8d:f2:61), Dst: Comdant_ff:94:00 (00:d0:03:ff:94:00)
Internet Protocol Version 4, Src: 172.26.21.229, 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: 1500
Identification: 0xe041 (57409)
Flags: 0x20, More fragments
0... .... = Reserved bit: Not set
0... .... = Don't fragment: Not set
...1. .... = More fragments: Set
Fragment Offset: 1480
> Time to Live: 1
```

Fig. 20. Segundo fragmento

Como foi dito na pergunta 2 B), sabemos que um fragmento é o primeiro quando o offset tem valor igual a 0. Neste caso, temos um offset diferente de 0 (Offset = 1480), logo não se trata do primeiro fragmento. Sabemos, também, que existem mais fragmentos, uma vez que o valor da flag More fragments se mantém igual a 1.

### 3.4 Pergunta 3 D)

Quantos fragmentos foram criados a partir do datagrama original? Como se detecta o último fragmento correspondente ao datagrama original?

```
Frame 9: 266 bytes on wire (2128 bits), 266 bytes captured (2128 bits) on interface \Device\NPF_{51B3331B-0084-4654-9C1C-81A8B4A00AF3}, id 0
Ethernet II, Src: IntelCor_Bd:f2:61 (ac:ed:5c:8d:f2:61), Dst: Comdant_ff:94:00 (00:d0:03:ff:94:00)
Internet Protocol Version 4, Src: 172.26.21.229, 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: 392
Identification: 0xe041 (57409)
Flags: 0x01
0... .... = Reserved bit: Not set
0... .... = Don't fragment: Not set
...0. .... = More fragments: Not set
Fragment Offset: 2960
> Time to Live: 1
```

Fig. 21. Último fragmento

Conseguimos identificar o último fragmento quando o valor da flag More fragments é igual a 0 e o valor do offset é diferente de 0, que é o caso apresentado na figura de cima. Logo, podemos concluir que foram criados 3 fragmentos a partir do datagrama original

### 3.5 Pergunta 3 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.

Entre os diferentes fragmentos, os valores das flags vão variando. Os valores das flags Offset e More fragments variam de fragmento para fragmento, permitindo o valor da flag More fragments avaliar se ainda existem mais fragmentos, enquanto que o valor da flag Offset permite organizar os fragmentos por ordem, uma vez que a ordem dos fragmentos é dada pelo valor do Offset, colocados em ordem crescente

## Parte 2

André Martins, Ana Teresa Gomes, and Pedro Barbosa

University of Minho, Department of Informatics, 4710-057 Braga, Portugal  
e-mail: {a89586,a89536,a89529}@alunos.uminho.pt

### 1 Pergunta 1

#### 1.1 Pergunta 1 A)

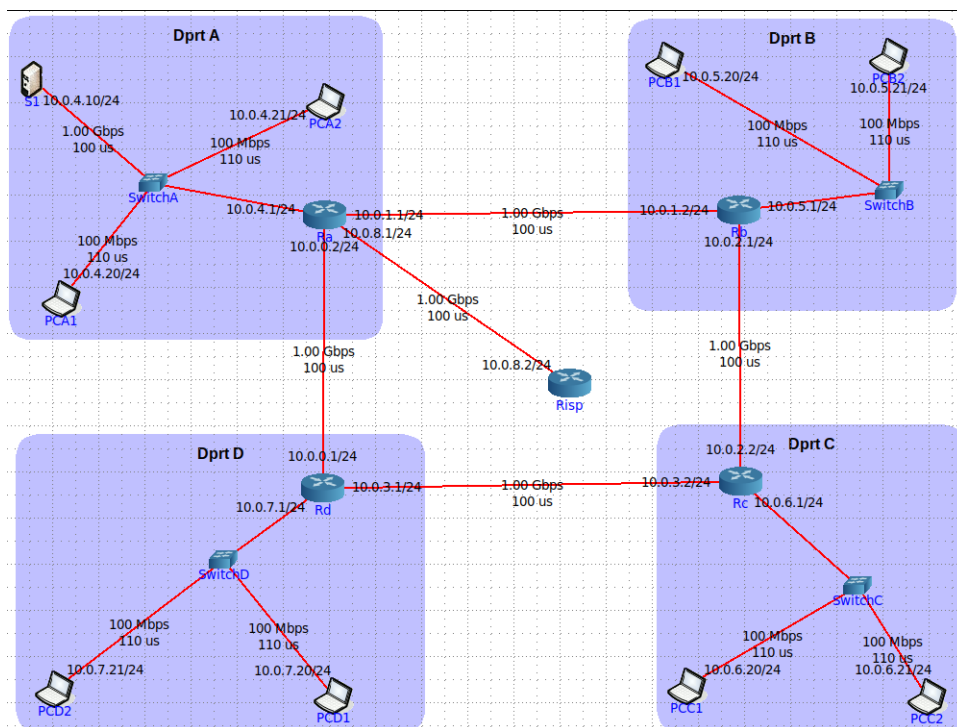


Fig. 1. Equipamentos e Departamentos

Indique que endereços IP e máscaras de rede foram atribuídos pelo CORE a cada equipamento. Para simplificar, pode incluir uma imagem que ilustre de forma clara a topologia definida e o endereçamento usado.

Como é possível verificar na figura acima, cada endereço possui um /24, logo, as máscaras utilizadas são 255.255.255.0

## 1.2 Pergunta 1 B)

Tratam-se de endereços públicos ou privados? Porquê?

Todos os endereços que estejam entre 10.0.0.0 e 10.255.255.255 são endereços privados, uma vez que se encaixam na Classe A. Como é possível analisar na figura 1, todos endereços estão dentro desse intervalo, logo, são endereços privados.

## 1.3 Pergunta 1 C)

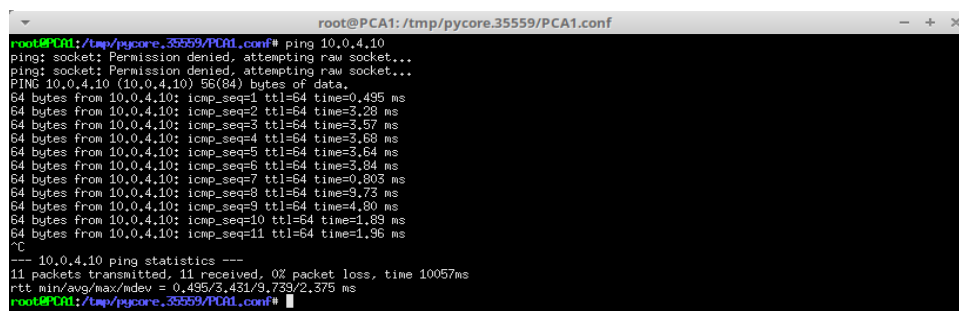
Porque razão não é atribuído um endereço IP aos switches?

A principal função de um switch é fazer a ligação entre equipamentos de uma rede. O switch regista o endereço MAC (Media Access Control) dos dispositivos que estão conectados. Depois de feita a análise deste endereço, são associadas as máquinas a que está ligado às respetivas entradas físicas do equipamento, sendo a informação enviada diretamente para o seu destino. Portanto, e devido à maneira de como se desenrola este processo, não existe a necessidade da atribuição de um endereço IP ao switch, uma vez que ele apenas decide para onde vão os pacotes, após a análise do endereço MAC de cada equipamento ligado entre si.

## 1.4 Pergunta 1 D)

Usando o comando ping certifique-se que existe conectividade IP entre os laptops dos vários departamentos e o servidor do departamento A (basta certificar-se da conectividade de um laptop por departamento).

Como é possível analisar nas 4 figuras abaixo, os PC's de cada departamento conseguem enviar e receber packets entre eles e o servidor S1 do Departamento A, provando, assim, que existe conexão entre os equipamentos e o Servidor S1.



```
root@PCA1: /tmp/pycore.35559/PCA1.conf# ping 10.0.4.10
ping: socket: Permission denied, attempting raw socket...
PING 10.0.4.10 (10.0.4.10) 56(84) bytes of data:
64 bytes from 10.0.4.10: icmp_seq=1 ttl=64 time=0.495 ms
64 bytes from 10.0.4.10: icmp_seq=2 ttl=64 time=3.28 ms
64 bytes from 10.0.4.10: icmp_seq=3 ttl=64 time=3.57 ms
64 bytes from 10.0.4.10: icmp_seq=4 ttl=64 time=3.68 ms
64 bytes from 10.0.4.10: icmp_seq=5 ttl=64 time=3.64 ms
64 bytes from 10.0.4.10: icmp_seq=6 ttl=64 time=3.84 ms
64 bytes from 10.0.4.10: icmp_seq=7 ttl=64 time=0.803 ms
64 bytes from 10.0.4.10: icmp_seq=8 ttl=64 time=9.73 ms
64 bytes from 10.0.4.10: icmp_seq=9 ttl=64 time=4.80 ms
64 bytes from 10.0.4.10: icmp_seq=10 ttl=64 time=1.89 ms
64 bytes from 10.0.4.10: icmp_seq=11 ttl=64 time=1.96 ms
^C
--- 10.0.4.10 ping statistics ---
11 packets transmitted, 11 received, 0% packet loss, time 10057ms
rtt min/avg/max/mdev = 0.495/3.431/9.739/2.375 ms
root@PCA1: /tmp/pycore.35559/PCA1.conf#
```

Fig. 2. Ping Departamento A, PCA1 para Servidor S1

```
root@PCB1:/tmp/pycore.38269/PCB1.conf# ping 10.0.4.10
PING 10.0.4.10 (10.0.4.10) 56(84) bytes of data:
64 bytes from 10.0.4.10: icmp_seq=1 ttl=62 time=2.39 ms
64 bytes from 10.0.4.10: icmp_seq=2 ttl=62 time=13.9 ms
64 bytes from 10.0.4.10: icmp_seq=3 ttl=62 time=40.0 ms
64 bytes from 10.0.4.10: icmp_seq=4 ttl=62 time=17.4 ms
64 bytes from 10.0.4.10: icmp_seq=5 ttl=62 time=2.82 ms
64 bytes from 10.0.4.10: icmp_seq=6 ttl=62 time=20.1 ms
64 bytes from 10.0.4.10: icmp_seq=7 ttl=62 time=14.5 ms
64 bytes from 10.0.4.10: icmp_seq=8 ttl=62 time=14.0 ms
64 bytes from 10.0.4.10: icmp_seq=9 ttl=62 time=6.68 ms
64 bytes from 10.0.4.10: icmp_seq=10 ttl=62 time=12.6 ms
^C
--- 10.0.4.10 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9041ms
rtt min/avg/max/mdev = 2.396/14.470/40.037/10.202 ms
root@PCB1:/tmp/pycore.38269/PCB1.conf#
```

Fig. 3. Ping Departamento B, PCB1 para Servidor S1

```
root@PCC1:/tmp/pycore.38269/PCC1.conf# ping 10.0.4.10
PING 10.0.4.10 (10.0.4.10) 56(84) bytes of data:
64 bytes from 10.0.4.10: icmp_seq=1 ttl=61 time=1.88 ms
64 bytes from 10.0.4.10: icmp_seq=2 ttl=61 time=1.23 ms
64 bytes from 10.0.4.10: icmp_seq=3 ttl=61 time=2.94 ms
64 bytes from 10.0.4.10: icmp_seq=4 ttl=61 time=1.14 ms
64 bytes from 10.0.4.10: icmp_seq=5 ttl=61 time=1.15 ms
64 bytes from 10.0.4.10: icmp_seq=6 ttl=61 time=1.14 ms
64 bytes from 10.0.4.10: icmp_seq=7 ttl=61 time=1.13 ms
^C
--- 10.0.4.10 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6027ms
rtt min/avg/max/mdev = 1.135/1.520/2.941/0.632 ms
root@PCC1:/tmp/pycore.38269/PCC1.conf#
```

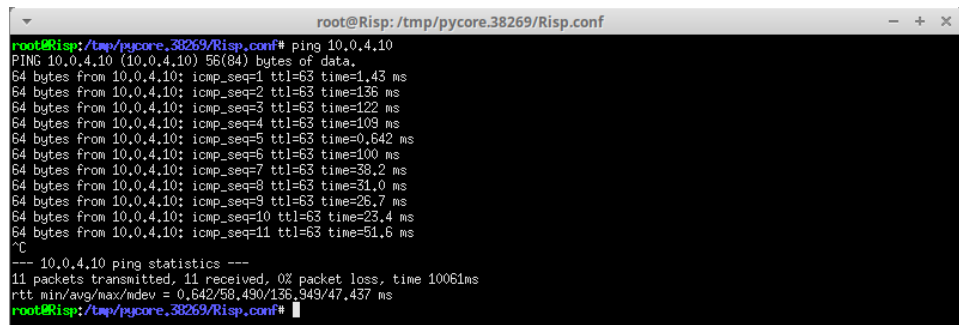
Fig. 4. Ping Departamento C, PCC1 para Servidor S1

```
root@PCD1:/tmp/pycore.38269/PCD1.conf# ping 10.0.4.10
PING 10.0.4.10 (10.0.4.10) 56(84) bytes of data:
64 bytes from 10.0.4.10: icmp_seq=1 ttl=62 time=1.70 ms
64 bytes from 10.0.4.10: icmp_seq=2 ttl=62 time=28.0 ms
64 bytes from 10.0.4.10: icmp_seq=3 ttl=62 time=37.1 ms
64 bytes from 10.0.4.10: icmp_seq=4 ttl=62 time=47.7 ms
64 bytes from 10.0.4.10: icmp_seq=5 ttl=62 time=61.6 ms
64 bytes from 10.0.4.10: icmp_seq=6 ttl=62 time=36.7 ms
^C
--- 10.0.4.10 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5030ms
rtt min/avg/max/mdev = 1.702/35.493/61.604/18.407 ms
root@PCD1:/tmp/pycore.38269/PCD1.conf#
```

Fig. 5. Ping Departamento D, PCD1 para Servidor S1

## 1.5 Pergunta 1 E)

Verifique se existe conectividade IP do router de acesso RISP para o servidor S1.



```
root@Risp:/tmp/pycore.38269/Risp.conf# ping 10.0.4.10
PING 10.0.4.10 (10.0.4.10) 56(84) bytes of data:
64 bytes from 10.0.4.10: icmp_seq=1 ttl=63 time=1.43 ms
64 bytes from 10.0.4.10: icmp_seq=2 ttl=63 time=136 ms
64 bytes from 10.0.4.10: icmp_seq=3 ttl=63 time=122 ms
64 bytes from 10.0.4.10: icmp_seq=4 ttl=63 time=109 ms
64 bytes from 10.0.4.10: icmp_seq=5 ttl=63 time=0.642 ms
64 bytes from 10.0.4.10: icmp_seq=6 ttl=63 time=100 ms
64 bytes from 10.0.4.10: icmp_seq=7 ttl=63 time=38.2 ms
64 bytes from 10.0.4.10: icmp_seq=8 ttl=63 time=31.0 ms
64 bytes from 10.0.4.10: icmp_seq=9 ttl=63 time=26.7 ms
64 bytes from 10.0.4.10: icmp_seq=10 ttl=63 time=23.4 ms
64 bytes from 10.0.4.10: icmp_seq=11 ttl=63 time=51.6 ms
^C
--- 10.0.4.10 ping statistics ---
11 packets transmitted, 11 received, 0% packet loss, time 10061ms
rtt min/avg/max/mdev = 0.642/58.490/136.949/47.437 ms
root@Risp:/tmp/pycore.38269/Risp.conf#
```

Fig. 6. Ping RISP para Servidor S1

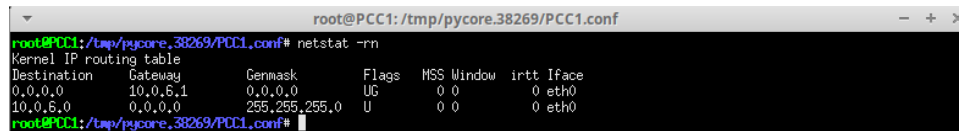
Como foi referido na pergunta anterior, comprova-se, assim, a conectividade entre o router RISP e o Servidor S1

## 2 Pergunta 2

### 2.1 Pergunta 2 A)

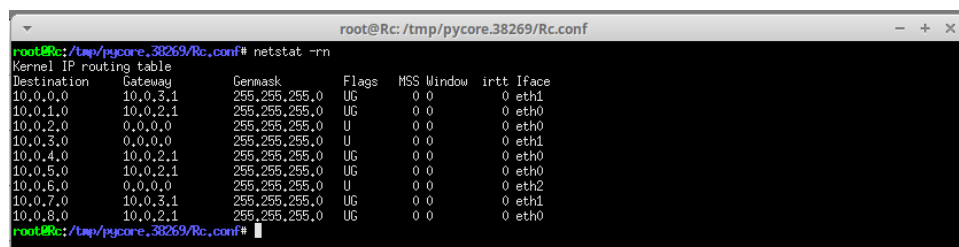
Para o router e um laptop do departamento C:

Execute o comando netstat -rn por forma a poder consultar a tabela de encaminhamento unicast (IPv4). Inclua no seu relatório as tabelas de encaminhamento obtidas; interprete as várias entradas de cada tabela.



```
root@PCC1:/tmp/pycore.38269/PCC1.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
0.0.0.0 10.0.6.1 0.0.0.0 UG 0 0 0 eth0
10.0.6.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
root@PCC1:/tmp/pycore.38269/PCC1.conf#
```

Fig. 7. Tabela de encaminhamento PC C1



```
root@Rc:/tmp/pycore.38269/Rc.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
10.0.0.0 10.0.3.1 255.255.255.0 UG 0 0 0 eth1
10.0.1.0 10.0.2.1 255.255.255.0 UG 0 0 0 eth0
10.0.2.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
10.0.3.0 0.0.0.0 255.255.255.0 U 0 0 0 eth1
10.0.4.0 10.0.2.1 255.255.255.0 UG 0 0 0 eth0
10.0.5.0 10.0.2.1 255.255.255.0 UG 0 0 0 eth0
10.0.6.0 0.0.0.0 255.255.255.0 U 0 0 0 eth2
10.0.7.0 10.0.3.1 255.255.255.0 UG 0 0 0 eth1
10.0.8.0 10.0.2.1 255.255.255.0 UG 0 0 0 eth0
root@Rc:/tmp/pycore.38269/Rc.conf#
```

Fig. 8. Tabela de encaminhamento router Rc

Através do comando netstat -rn, conseguimos obter as tabelas de endereçamento do router RC e do PC C1. Estas tabelas apresenta algumas informações acerca das rotas que

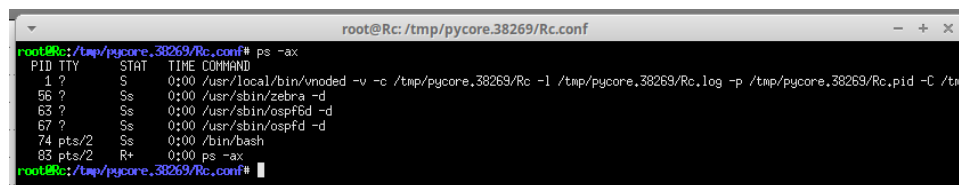
os pacotes terão que efetuar. A coluna *Destination* indica a sub-rede de destino, a coluna *Gateway* indica por que equipamento terá que esse pacote passar e a coluna *Genmask* indica o tipo de máscara utilizado.

Utilizando o exemplo do router Rc, se um pacote tiver como destino um equipamento da sub-rede 10.0.0.0 tem que passar pelo router 10.0.3.1. No entanto, se o destino for um equipamento da sub-rede 10.0.3.0, o pacote pode seguir qualquer caminho, uma vez que o router não está especificado.

Se usarmos agora o exemplo do PC PCC1, existem apenas duas rotas possíveis. Tem a rota default que, independentemente do destino do pacote, este terá sempre que passar pelo router 10.0.6.1 (Router Rc). A outra rota especifica o caminho a seguir quando o pacote tem como endereço de destino um endereço da sub-rede do Departamento C. Neste caso, o pacote pode optar por um destino qualquer.

## 2.2 Pergunta 2 B)

Diga, justificando, se está a ser usado encaminhamento estático ou dinâmico

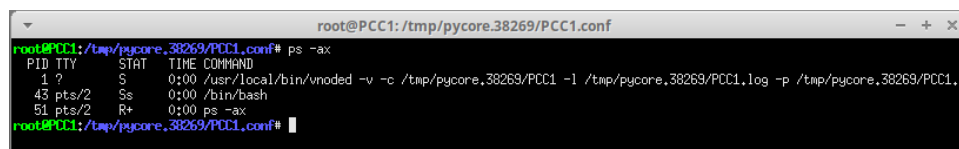


```

root@Rc:/tmp/pycore.38269/Rc.conf# ps -ax
PID TTY          STAT       TIME COMMAND
  1 ?            S          0:00 /usr/local/bin/vnoded -v -c /tmp/pycore.38269/Rc -l /tmp/pycore.38269/Rc.log -p /tmp/pycore.38269/Rc.pid -C /tm
 56 ?          Ss         0:00 /usr/sbin/zebra -d
 63 ?          Ss         0:00 /usr/sbin/ospfd -d
 67 ?          Ss         0:00 /usr/sbin/ospfd -d
 74 pts/2      Ss         0:00 /bin/bash
 83 pts/2      R+        0:00 ps -ax
root@Rc:/tmp/pycore.38269/Rc.conf#

```

Fig. 9. Processos a correr no router Rc



```

root@PCC1:/tmp/pycore.38269/PCC1.conf# ps -ax
PID TTY          STAT       TIME COMMAND
  1 ?            S          0:00 /usr/local/bin/vnoded -v -c /tmp/pycore.38269/PCC1 -l /tmp/pycore.38269/PCC1.log -p /tmp/pycore.38269/PCC1.p
 43 pts/2      Ss         0:00 /bin/bash
 51 pts/2      R+        0:00 ps -ax
root@PCC1:/tmp/pycore.38269/PCC1.conf#

```

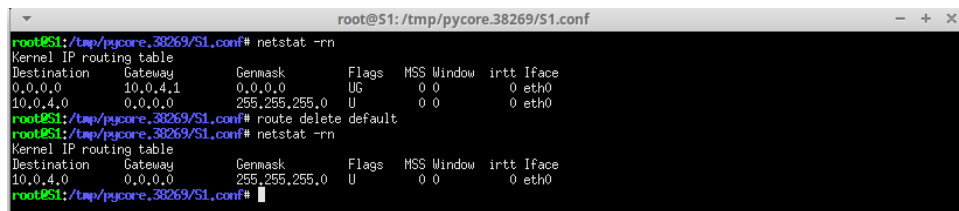
Fig. 10. Processos a correr no PC PCC1

Como podemos observar na figura 3, que representa os processos a correr no router Rc, existem processos a correr os protocolos ZEBRA e OSPF, logo, podemos concluir que o router Rc está a utilizar endereçamento dinâmico. No endereçamento dinâmico as rotas são atualizadas ao longo do tempo. Os protocolos permitem a um pacote seguir um destino distinto quando não é possível fazer a rota predefinida.

Por outro lado, o PC PCC1 está a utilizar endereçamento estático, ou seja, quando um pacote não consegue seguir as rotas predefinidas nas tabelas de endereçamento, este é descartado.

### 2.3 Pergunta 2 C)

Admita que, por questões administrativas, a rota por defeito (0.0.0.0 ou default) deve ser retirada definitivamente da tabela de encaminhamento do servidor S1 localizado no departamento A. Use o comando `route delete` para o efeito. Que implicações tem esta medida para os utilizadores da organização MIEI-RC que acedem ao servidor. Justifique.



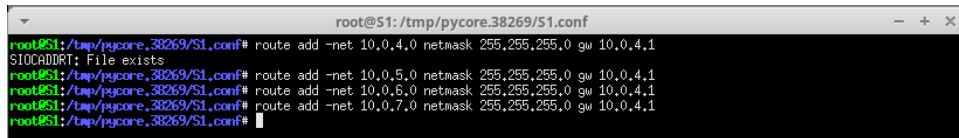
```
root@S1:/tmp/pycore.38269/S1.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
0.0.0.0 10.0.4.1 0.0.0.0 UG 0 0 0 eth0
10.0.4.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
root@S1:/tmp/pycore.38269/S1.conf# route delete default
root@S1:/tmp/pycore.38269/S1.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
10.0.4.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
root@S1:/tmp/pycore.38269/S1.conf#
```

Fig. 11. Remover route default

Ao realizar a eliminação da rota por defeito, impossibilitamos o servidor S1 de efetuar qualquer rota que se situe fora do seu departamento, ficando apenas limitado aos equipamentos do seu departamento.

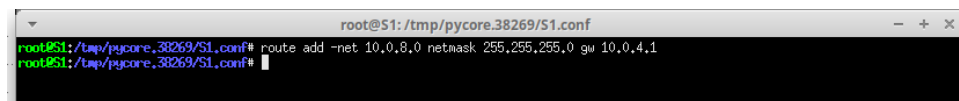
### 2.4 Pergunta 2 D)

Adicione as rotas estáticas necessárias para restaurar a conectividade para o servidor S1, por forma a contornar a restrição imposta na alínea c). Utilize para o efeito o comando `route add` e registre os comandos que usou.



```
root@S1:/tmp/pycore.38269/S1.conf# route add -net 10.0.4.0 netmask 255.255.255.0 gw 10.0.4.1
SIOCADDRT: File exists
root@S1:/tmp/pycore.38269/S1.conf# route add -net 10.0.5.0 netmask 255.255.255.0 gw 10.0.4.1
root@S1:/tmp/pycore.38269/S1.conf# route add -net 10.0.6.0 netmask 255.255.255.0 gw 10.0.4.1
root@S1:/tmp/pycore.38269/S1.conf# route add -net 10.0.7.0 netmask 255.255.255.0 gw 10.0.4.1
root@S1:/tmp/pycore.38269/S1.conf#
```

Fig. 12. Add routers de cada departamentoAdd RISP



```
root@S1:/tmp/pycore.38269/S1.conf# route add -net 10.0.8.0 netmask 255.255.255.0 gw 10.0.4.1
root@S1:/tmp/pycore.38269/S1.conf#
```

Fig. 13. Add RISP



## 2.5 Pergunta 2 E)

Teste a nova política de encaminhamento garantindo que o servidor está novamente acessível, utilizando para o efeito o comando ping. Registe a nova tabela de encaminhamento do servidor.

```
root@PCA1:/tmp/pycore.38269/PCA1.conf# ping 10.0.4.10
PING 10.0.4.10 (10.0.4.10) 56(84) bytes of data:
64 bytes from 10.0.4.10: icmp_seq=1 ttl=63 time=9.42 ms
64 bytes from 10.0.4.10: icmp_seq=2 ttl=63 time=0.537 ms
64 bytes from 10.0.4.10: icmp_seq=3 ttl=63 time=4.79 ms
64 bytes from 10.0.4.10: icmp_seq=4 ttl=63 time=26.2 ms
64 bytes from 10.0.4.10: icmp_seq=5 ttl=63 time=13.2 ms
64 bytes from 10.0.4.10: icmp_seq=6 ttl=63 time=3.88 ms
^C
-- 10.0.4.10 ping statistics --
5 packets transmitted, 5 received, 0% packet loss, time 5042ms
rtt min/avg/max/ndev = 0.597/9.688/26.174/9.415 ms
root@PCA1:/tmp/pycore.38269/PCA1.conf#
```

Fig. 14. Ping PCA1 para Servidor S1

```
root@PCB1:/tmp/pycore.38269/PCB1.conf# ping 10.0.4.10
PING 10.0.4.10 (10.0.4.10) 56(84) bytes of data:
64 bytes from 10.0.4.10: icmp_seq=1 ttl=62 time=1.72 ms
64 bytes from 10.0.4.10: icmp_seq=2 ttl=62 time=21.6 ms
64 bytes from 10.0.4.10: icmp_seq=3 ttl=62 time=69.9 ms
64 bytes from 10.0.4.10: icmp_seq=4 ttl=62 time=52.5 ms
64 bytes from 10.0.4.10: icmp_seq=5 ttl=62 time=41.1 ms
^C
-- 10.0.4.10 ping statistics --
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/ndev = 1.726/23.449/69.945/25.368 ms
root@PCB1:/tmp/pycore.38269/PCB1.conf#
```

Fig. 15. Ping PCB1 para Servidor S1

```
root@PCC1:/tmp/pycore.38269/PCC1.conf# ping 10.0.4.10
PING 10.0.4.10 (10.0.4.10) 56(84) bytes of data:
64 bytes from 10.0.4.10: icmp_seq=1 ttl=61 time=3.35 ms
64 bytes from 10.0.4.10: icmp_seq=2 ttl=61 time=55.1 ms
64 bytes from 10.0.4.10: icmp_seq=3 ttl=61 time=50.1 ms
64 bytes from 10.0.4.10: icmp_seq=4 ttl=61 time=53.6 ms
64 bytes from 10.0.4.10: icmp_seq=5 ttl=61 time=54.6 ms
64 bytes from 10.0.4.10: icmp_seq=6 ttl=61 time=23.4 ms
^C
-- 10.0.4.10 ping statistics --
6 packets transmitted, 6 received, 0% packet loss, time 5053ms
rtt min/avg/max/ndev = 3.253/40.079/65.149/25.202 ms
root@PCC1:/tmp/pycore.38269/PCC1.conf#
```

Fig. 16. Ping PCC1 para Servidor S1

```
root@PCD1:/tmp/pycore.38269/PCD1.conf# ping 10.0.4.10
PING 10.0.4.10 (10.0.4.10) 56(84) bytes of data:
64 bytes from 10.0.4.10: icmp_seq=1 ttl=62 time=3.42 ms
64 bytes from 10.0.4.10: icmp_seq=2 ttl=62 time=60.0 ms
64 bytes from 10.0.4.10: icmp_seq=3 ttl=62 time=16.4 ms
64 bytes from 10.0.4.10: icmp_seq=4 ttl=62 time=2.52 ms
64 bytes from 10.0.4.10: icmp_seq=5 ttl=62 time=1.40 ms
^C
-- 10.0.4.10 ping statistics --
5 packets transmitted, 5 received, 0% packet loss, time 4013ms
rtt min/avg/max/ndev = 1.483/15.798/60.092/22.323 ms
root@PCD1:/tmp/pycore.38269/PCD1.conf#
```

Fig. 17. Ping PCD1 para Servidor S1

```
root@S1:/tmp/pycore.38269/S1.conf# netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
10.0.4.0         10.0.4.1       255.255.255.0   UG      0 0         0 eth0
10.0.4.0         0.0.0.0        255.255.255.0   U        0 0         0 eth0
10.0.5.0         10.0.4.1       255.255.255.0   UG      0 0         0 eth0
10.0.6.0         10.0.4.1       255.255.255.0   UG      0 0         0 eth0
10.0.7.0         10.0.4.1       255.255.255.0   UG      0 0         0 eth0
10.0.8.0         10.0.4.1       255.255.255.0   UG      0 0         0 eth0
root@S1:/tmp/pycore.38269/S1.conf#
```

Fig. 18. Tabela de encaminhamento Servidor S1

### 3 Pergunta 3

#### 3.1 Pergunta 3 A)

Considere que dispõe apenas do endereço de rede IP 130.12.96.0/19. Defina um novo esquema de endereçamento para as redes dos departamentos (mantendo a rede de acesso e core inalteradas) e atribua endereços às interfaces dos vários sistemas envolvidos. Assuma que todos os endereços de sub-redes são usáveis. Deve justificar as opções usadas.

O endereço de redes que dispomos é o 130.12.96.0/19 e temos que criar 4 sub-redes (1 para cada departamento). Iremos necessitar de 3 bits para criar as sub-redes necessárias, uma vez que, com 3 bits, é possível criar 6 sub-redes ( $2^3 - 2$ ). Com a utilização destes 3 bits para subnetting, a nossa máscara passará a ter 22 bits.

**Table 1.** Sub-redes criadas

SR1	000	Reservado	—
SR2	001	130.12.100.0/22	Departamento A
SR3	010	130.12.104.0/22	Livre
SR4	011	130.12.108.0/22	Departamento B
SR5	100	130.12.112.0/22	Livre
SR6	101	130.12.116.0/22	Departamento C
SR7	110	130.12.120.0/22	Departamento D
SR8	111	Reservado	—

**Table 2.** IP's de host atribuídos a cada departamento

Departamento	IP	IP- Início	IP- Fim
A	130.12.100.0/22	130.12.100.0	130.12.103.255
B	130.12.108.0/22	130.12.108.0	130.12.111.255
C	130.12.116.0/22	130.12.116.0	130.12.119.255
D	130.12.120.0/22	130.12.120.0	130.12.123.255

**Table 3.** Endereços de cada equipamento dos departamentos

Dprt A	IP	Dprt B	IP	Dprt C	IP	Dprt D	IP
PCA1	130.12.100.2	PCB1	130.12.108.2	PCC1	130.12.116.2	PCD1	130.12.120.2
PCA2	130.12.100.3	PCB2	130.12.108.3	PCC2	130.12.116.3	PCD2	130.12.120.3
Ra	130.12.100.1	Rb	130.12.108.1	Rc	130.12.116.1	Rd	130.12.120.1
S1	130.12.100.4	-	-	-	-	-	-

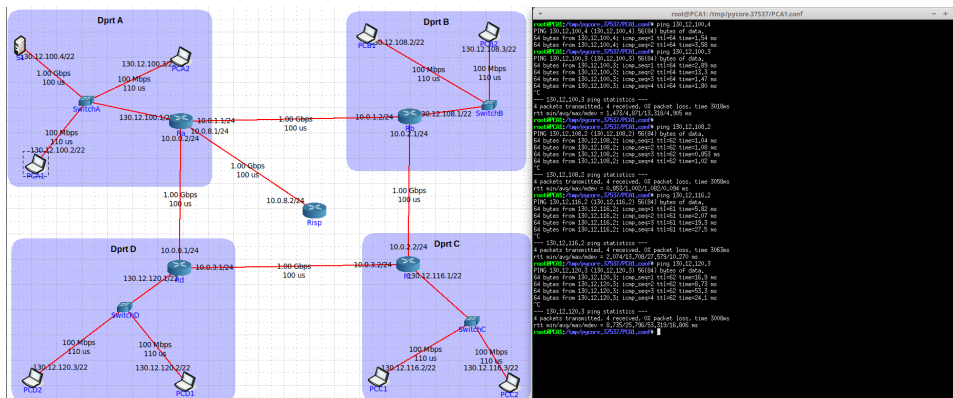
### 3.2 Pergunta 3 B)

Qual a máscara de rede que usou (em formato decimal)? Quantos hosts IP pode interligar em cada departamento? Justifique.

Uma vez que utilizamos 3 bits para realizar o subnetting, a nossa máscara passou de 19 para 22 bits, sendo o seu valor decimal 255.255.252.0. Sendo que a máscara tem 22 bits, ficamos com  $(32 - 22)$  bits que podem ser alterados. Portanto, em cada departamento será possível atribuir  $(2^{(32-22)} - 1 - 2)$  1021 endereços de IP para hosts, sendo que cada rede reserva sempre 2 endereços.

### 3.3 Pergunta 3 C)

Garanta e verifique que conectividade IP entre as várias redes locais da organização MIEI-RC é mantida. Explique como procedeu.



Ao executar o comando ping de um PC do Departamento A (PCA1) para PC's localizados noutros departamentos, rapidamente percebemos que existe conectividade IP entre as várias redes locais. O PCA1 consegue, com sucesso, conectar-se com equipamentos presentes noutros departamentos e noutras sub-redes.

## 4 Conclusão

Neste trabalho prático, na primeira parte, analisamos o protocolo IPv4 e o tráfego ICMP. Analisamos, também, alguns casos particulares de IPv4 onde ocorria a fragmentação, quando um pacote IP tinha um tamanho superior ao MTU da rede. Esta primeira parte foi importante para aprofundar e ganhar prática na análise de tráfego ICMP em ferramentas como o Wireshark, saber analisar os resultados obtidos e as flags representadas, aprofundar a análise de comandos como o `tracert` e saber em como o valor do TTL influencia a comunicação entre vários equipamentos.

Na segunda parte do trabalho prático, fizemos uma análise mais profunda e complexa, nomeadamente de tabelas de endereçamento e encaminhamento IP. Esta parte já nos permitiu aprofundar os conhecimentos acerca do subnetting, como é feita a manipulação de endereços IP para efeitos de subnetting, como é realizado o encaminhamento e como se podem definir sub-redes. Também aprendemos a analisar o resultado de comandos com o `ping` ou o `netstat -rn`, analisando, assim, cada coluna da tabela resultante da execução desse comando. O exercício 3 desta parte merece especial destaque, uma vez que foi o exercício que envolveu mais cálculos e que nos permitiu perceber de que maneiras é possível definir subnets através de um endereço de rede IP. Ao "pôr a mão na massa", digamos assim, foi possível entender a lógica e estrutura por detrás do subnetting, tornando este trabalho prático ainda mais interessante e enriquecedor.