

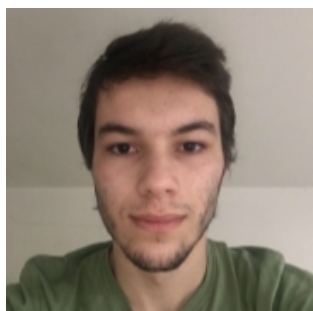


Universidade do Minho  
Escola de Engenharia

# Redes de Computadores

## Trabalho Prático 2

Francisco Correia Franco A89458  
António Jorge Nande Rodrigues A89585  
Luís Enes Sousa A89597



A89458



A89585



A89597

## Conteúdo

<b>1</b>	<b>Captura de tráfego IP</b>	<b>1</b>
1.1	Pergunta 1 . . . . .	1
1.2	Pergunta 2 . . . . .	3
1.3	Pergunta 3 . . . . .	7
<b>2</b>	<b>Endereçamento e Encaminhamento IP</b>	<b>10</b>
2.1	Pergunta 1 . . . . .	10
2.2	Pergunta 2 . . . . .	12
<b>3</b>	<b>Definição de Sub-redes</b>	<b>16</b>
3.1	Pergunta 1 . . . . .	16
3.2	Pergunta 2 . . . . .	17
3.3	Pergunta 3 . . . . .	18
<b>4</b>	<b>Conclusão</b>	<b>19</b>

# 1 Captura de tráfego IP

## 1.1 Pergunta 1

Prepare uma topologia CORE para verificar o comportamento do trace-route. 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).

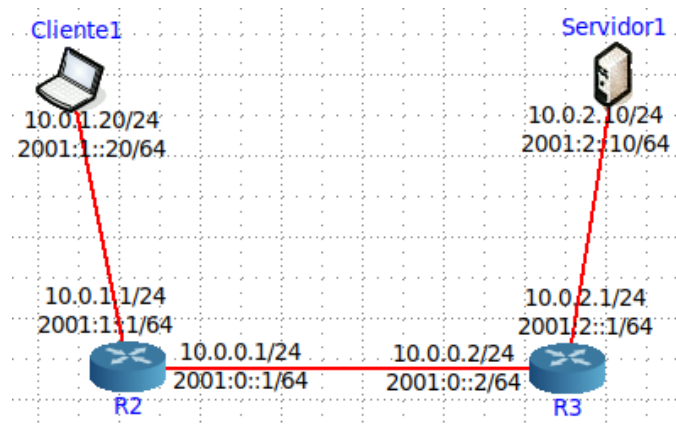


Figura 1: Topologia CORE

A - Ative o wireshark ou o tcpdump no pc h1. Numa shell de h1, execute o comando traceroute -I para o endereço IP do host s4.

```
root@Cliente1:/tmp/pycore.35579/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 10.0.1.1 (10.0.1.1)  0.054 ms  0.053 ms  0.016 ms
 2 10.0.0.2 (10.0.0.2)  0.034 ms  0.020 ms  0.017 ms
 3 10.0.2.10 (10.0.2.10)  0.073 ms  0.025 ms  0.023 ms
```

Figura 2: Traceroute do Cliente1 para o Servidor1

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

Inicialmente, o Cliente1 tenta comunicar com o Servidor1, mas, como o TTL é igual a 1, o pacote vai ser descartado ao chegar a R2, sendo retribuída uma mensagem de erro ao Cliente1. O TTL é, então, aumentado sucessivamente, até chegar a TTL 3, que é o mínimo necessário para chegar ao Servidor1.

No.	Time	Source	Destination	Protocol	Length	Info
76	35.344361341	10.0.1.20	10.0.2.10	ICMP	74	Echo (ping) request id=0x0026, seq=1/256, ttl=1 (no response found!)
77	35.344398104	10.0.1.1	10.0.1.20	ICMP	102	Time-to-live exceeded (time to live exceeded in transit)
78	35.344448376	10.0.1.20	10.0.2.10	ICMP	74	Echo (ping) request id=0x0026, seq=2/512, ttl=1 (no response found!)
79	35.344461511	10.0.1.1	10.0.1.20	ICMP	102	Time-to-live exceeded (time to live exceeded in transit)
80	35.344476619	10.0.1.20	10.0.2.10	ICMP	74	Echo (ping) request id=0x0026, seq=3/768, ttl=1 (no response found!)
81	35.344486808	10.0.1.1	10.0.1.20	ICMP	102	Time-to-live exceeded (time to live exceeded in transit)
82	35.344498848	10.0.1.20	10.0.2.10	ICMP	74	Echo (ping) request id=0x0026, seq=4/1024, ttl=2 (no response found!)
83	35.344529970	10.0.1.1	10.0.1.20	ICMP	102	Time-to-live exceeded (time to live exceeded in transit)
84	35.344535953	10.0.1.20	10.0.2.10	ICMP	74	Echo (ping) request id=0x0026, seq=5/1280, ttl=2 (no response found!)
85	35.344549853	10.0.1.1	10.0.1.20	ICMP	102	Time-to-live exceeded (time to live exceeded in transit)
86	35.344557846	10.0.1.20	10.0.2.10	ICMP	74	Echo (ping) request id=0x0026, seq=6/1536, ttl=2 (no response found!)
87	35.344597005	10.0.1.1	10.0.1.20	ICMP	102	Time-to-live exceeded (time to live exceeded in transit)
88	35.344581180	10.0.1.20	10.0.2.10	ICMP	74	Echo (ping) request id=0x0026, seq=7/1792, ttl=3 (reply in 89)
89	35.344649838	10.0.2.10	10.0.1.20	ICMP	74	Echo (ping) reply id=0x0026, seq=7/1792, ttl=62 (request in 88)
90	35.344680966	10.0.1.20	10.0.2.10	ICMP	74	Echo (ping) request id=0x0026, seq=8/2048, ttl=3 (reply in 91)
91	35.344680789	10.0.2.10	10.0.1.20	ICMP	74	Echo (ping) reply id=0x0026, seq=8/2048, ttl=62 (request in 90)
92	35.344689487	10.0.1.20	10.0.2.10	ICMP	74	Echo (ping) request id=0x0026, seq=9/2304, ttl=3 (reply in 93)
93	35.344707854	10.0.2.10	10.0.1.20	ICMP	74	Echo (ping) reply id=0x0026, seq=9/2304, ttl=62 (request in 92)

Figura 3: Tráfego ICMP

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

O valor inicial do campo TTL deverá ser 3. Na figura 3 podemos verificar que o valor mais baixo do TTL para o qual o pacote chega ao destino é 3.

**D - Calcule o valor médio do tempo de ida-e-volta (Round-Trip Time) obtido?**

O Round-Trip Time médio deverá ser 0.020 ms.

91	35.344680789	10.0.2.10	10.0.1.20	ICMP	74	Echo (ping) reply id=0x0026, seq=8/2048, ttl=62 (request in 90)
<p>Frame 91: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0</p> <p>Ethernet II, Src: 00:00:00:aa:00:05 (00:00:00:aa:00:05), Dst: 00:00:00:aa:00:04 (00:00:00:aa:00:04)</p> <p>Internet Protocol Version 4, Src: 10.0.2.10, Dst: 10.0.1.20</p> <p>Internet Control Message Protocol</p> <p>Type: 0 (Echo (ping) reply)</p> <p>Code: 0</p> <p>Checksum: 0x8a4c [correct]</p> <p>[Checksum Status: Good]</p> <p>Identifier (BE): 38 (0x0026)</p> <p>Identifier (LE): 9728 (0x2600)</p> <p>Sequence number (BE): 8 (0x0008)</p> <p>Sequence number (LE): 2048 (0x0800)</p> <p>[Request frame: 90]</p> <p>[Response time: 0.020 ms]</p> <p>Data (32 bytes)</p>						

Figura 4: Round-Trip Time

## 1.2 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 e centre a análise no nível protocolar IP. Através da análise do cabeçalho IP diga:

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

O endereço IP da interface ativa do nosso computador é 172.26.9.103.

No.	Time	Source	Destination	Protocol	Length	Info
5	0.274648	172.26.9.103	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=92/23552, ttl=1 (no response found!)

Figura 5: Endereço IP da interface ativa

**B - Qual é o valor do campo protocolo? O que identifica?**

O campo protocolo é igual a 1, referente ao protocolo ICMP (Internet Control Message Protocol).

```
✓ Internet Protocol Version 4, Src: 172.26.9.103, 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: 0x19c0 (6592)
  > Flags: 0x00
    Fragment Offset: 0
  > Time to Live: 1
    Protocol: ICMP (1)
```

Figura 6: Valor do campo protocolo

**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?**

O cabeçalho IP(v4) tem 20 bytes.

O campo de dados tem um tamanho de 36 bytes. Este valor é calculado através da diferença do tamanho total (56 bytes) com o tamanho do cabeçalho (20 bytes).

```
▼ Internet Protocol Version 4, Src: 172.26.9.103, 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
```

Figura 7: Header e Total Length

**D - O datagrama IP foi fragmentado? Justifique.**

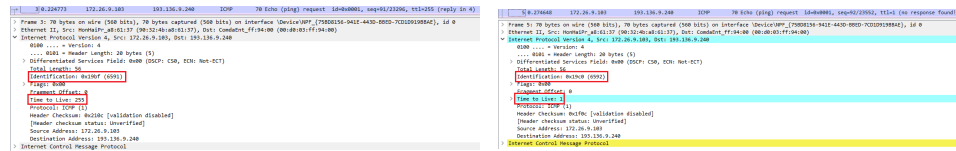
O valor das flags e do offset é igual a 0, logo o pacote não foi fragmentado.

```
▼ Internet Protocol Version 4, Src: 172.26.9.103, 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: 0x19c0 (6592)
    ▼ Flags: 0x00
      0... .... = Reserved bit: Not set
      .0.. .... = Don't fragment: Not set
      ..0. .... = More fragments: Not set
      Fragment Offset: 0
```

Figura 8: Flags e Offset

**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.**

Os campos do cabeçalho IP que variam são o campo de identificação e o TTL.



Pacote 1

Pacote 2

Figura 9: Exemplo de dois pacotes enviados pelo computador

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

Como se pode ver na Figura 9, referente a dois pacotes consecutivos, o campo de identificação é incrementado a cada pacote enviado.

Na Figura 10 verifica-se que o TTL também é incrementado a cada pacote enviado. No entanto, a cada 5 pacotes é enviado um pacote de teste (TTL = 255).

3	0.224773	172.26.9.103	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=91/23296,	t=255 (reply in 4)
5	0.274648	172.26.9.103	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=92/23552,	t=1 (no response found!)
8	0.324093	172.26.9.103	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=93/23808,	t=2 (no response found!)
11	0.375213	172.26.9.103	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=94/24064,	t=3 (no response found!)
13	0.425535	172.26.9.103	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=95/24320,	t=4 (reply in 14)
19	2.725333	172.26.9.103	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=96/24576,	t=255 (reply in 20)
21	2.775516	172.26.9.103	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=97/24832,	t=1 (no response found!)
23	2.825566	172.26.9.103	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=98/25088,	t=2 (no response found!)
25	2.875563	172.26.9.103	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=99/25344,	t=3 (no response found!)
27	2.925648	172.26.9.103	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=100/25600,	t=4 (reply in 28)
37	5.226381	172.26.9.103	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=101/25856,	t=255 (reply in 38)

Figura 10: Evolução do TTL

**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 enviadas ao seu host? Porquê?**

A Figura 12 mostra os diferentes valores dos TTL. O primeiro pacote tem TTL=255, o segundo TTL=254 e o terceiro TTL=253. Isto deve-se ao facto dos TTL enviados pelo nosso router serem incrementados ao longo do tempo fazendo com que cheguem mais longe e saltem por mais routers. Quando é necessário enviar uma mensagem de erro os pacotes são enviados do router mais distante até ao nosso router. Sendo assim, quanto maior o TTL do pacote enviado, menor o TTL do pacote recebido (vindo do router que enviou a mensagem de erro), pois este último tem de passar por mais routers intermédios, tendo o seu TTL sido decrementado em cada um.

6 0.278422	172.26.254.254	172.26.9.103	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
9 0.328185	172.16.2.1	172.26.9.103	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
12 0.378811	172.16.115.252	172.26.9.103	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)

Figura 11: Pacotes ordenados por destino



1º Pacote



2º Pacote



3º Pacote

Figura 12: Exemplo de três pacotes consecutivos recebidos pelo computador



### 1.3 Pergunta 3

Pretende-se agora analisar a fragmentação de pacotes IP.

Observe o tráfego depois do tamanho de pacote ter sido definido para 32XX bytes.

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

O pacote inicial teve de ser fragmentado, pois este tem 3252 bytes e o MTU (Maximum Transport Unit) da rede é 1500 bytes. Assim, o pacote tem de ser fragmentado em três partes, para poder seguir nesta rede.

40	2.347057	172.26.9.103	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=20b4) [Reassembled in #42]
41	2.347057	172.26.9.103	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=1480, ID=20b4) [Reassembled in #42]
42	2.347057	172.26.9.103	193.136.9.240	ICMP	306 Echo (ping) request id=0x0001, seq=1872/20487, ttl=1 (no response found!)
43	2.358901	172.26.254.254	172.26.9.103	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)

Figura 13: Primeira mensagem ICMP

**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?**

Pode-se concluir que o datagrama foi fragmentado, pois as flags têm um valor diferente de 0.

O facto de 'Fragment Offset' ser 0 e a flag 'More fragments' ser 1 indica que se trata do primeiro fragmento.

O tamanho deste datagrama IP é 1500 bytes.

40	2.347057	172.26.9.103	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=20b4) [Reassembled in #42]
> Frame 48: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface \Device\NPF_{758D8156-941E-443D-8BED-7CD1D9198BAE}, id 0					
> Ethernet II, Src: MonHaiPr_a8:61:37 (90:32:4b:a8:61:37), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)					
▼ Internet Protocol Version 4, Src: 172.26.9.103, 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: 0x20b5 (8373)					
▼ 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: 2					
Protocol: ICMP (1)					
Header Checksum: 0xf172 [validation disabled]					
[Header checksum status: Unverified]					
Source Address: 172.26.9.103					
Destination Address: 193.136.9.240					
[Reassembled IPv4 in frame: 50]					
> Data (1480 bytes)					

Figura 14: Primeiro fragmento do datagrama IP segmentado

**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?**

O 'Fragment Offset' é igual a 1480 (diferente de 0), logo não se trata do primeiro fragmento.

Como o datagrama tem a flag 'More Fragments' a 1, pode-se concluir que há mais fragmentos.

```

41 2.347057 172.26.9.103 193.136.9.240 IPv4 1514 Fragmented IP protocol (proto=ICMP 1, off=1480, ID=20b4) [Reassembled in #42]
> Frame 41: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface \Device\NPF_{75B08156-941E-4430-BBED-7CD1D9198BAE}, id 0
> Ethernet II, Src: HonHaiPr_a8:61:37 (90:32:4b:a8:61:37), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
v Internet Protocol Version 4, Src: 172.26.9.103, 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: 0x20b4 (8372)
  v 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
  Protocol: ICMP (1)
  Header Checksum: 0xf1ba [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 172.26.9.103
  Destination Address: 193.136.9.240
  [Reassembled IPv4 in frame: 42]
> Data (1480 bytes)

```

Figura 15: Segundo fragmento do datagrama IP segmentado

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

Foram criados 3 fragmentos a partir do datagrama original.

Sabe-se que este é o último fragmento do datagrama original, pois a flag 'More Fragments' está a 0 e o 'Fragment Offset' é diferente de 0.

```

42 2.347057 172.26.9.103 193.136.9.240 ICMP 306 Echo (ping) request id=0x0001, seq=1872/20487, ttl=1 (no response found!)
> Frame 42: 306 bytes on wire (2448 bits), 306 bytes captured (2448 bits) on interface \Device\NPF_{75B08156-941E-4430-BBED-7CD1D9198BAE}, id 0
> Ethernet II, Src: HonHaiPr_a8:61:37 (90:32:4b:a8:61:37), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
v Internet Protocol Version 4, Src: 172.26.9.103, 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: 292
  Identification: 0x20b4 (8372)
  v 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
  Protocol: ICMP (1)
  Header Checksum: 0x15ba [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 172.26.9.103
  Destination Address: 193.136.9.240
  > [3 IPv4 Fragments (3232 bytes): #40(1480), #41(1480), #42(272)]
> Internet Control Message Protocol

```

Figura 16: Último fragmento do datagrama IP segmentado

**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.**

Os campos que mudam entre os diferentes fragmentos são as flags 'More Fragments' e 'Fragment Offset'.

A flag 'More Fragments' indica se existem mais fragmentos. Já a flag 'Fragment Offset' permite identificar a posição do fragmento no datagrama original. Assim, é possível reconstruir o datagrama original juntando os fragmentos por ordem crescente do valor da flag 'Fragment Offset'.

## 2 Endereçamento e Encaminhamento IP

Considere que a organização MIEL-RC é constituída por quatro departamentos (A, B, C e D) e cada departamento possui um router de acesso à sua rede local. Estes routers de acesso (RA, RB, RC e RD) estão interligados entre si por ligações Ethernet a 1Gbps, formando um anel.

Por sua vez, existe um servidor (S1) na rede do departamento A e, pelo menos, dois laptops por departamento, interligados ao router respetivo através de um comutador (switch). S1 tem uma ligação a 1Gbps e os laptops ligações a 100Mbps. Considere apenas a existência de um comutador por departamento.

A conectividade IP externa da organização é assegurada através de um router de acesso RISP conectado a RA por uma ligação ponto-a-ponto a 10 Gbps.

### 2.1 Pergunta 1

**A - 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.**

A imagem seguinte ilustra os vários endereços IP e a respetiva máscara (255.255.255.0, ou /24 em notação CIDR) atribuídos pelo CORE a cada equipamento.

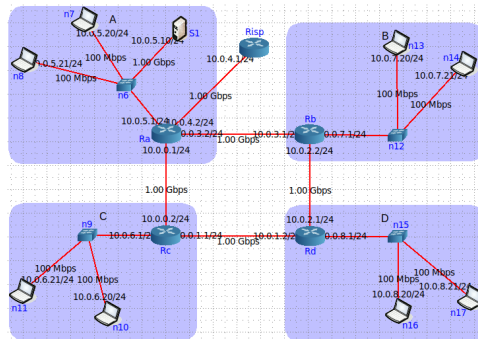


Figura 17: Topologia CORE

### B - Tratam-se de endereços públicos ou privados? Porquê?

Tratam-se de endereços privados, pois pertencem à Classe A (endereços entre 10.0.0.0 e 10.255.255.255).

### C - Porque razão não é atribuído um endereço IP aos switches?

O switch tem como principal função a interligação de equipamentos. Na primeira vez que um pacote é reencaminhado para um equipamento, o switch guarda o endereço MAC na sua tabela de endereços MAC (onde são guardados os endereços MAC dos equipamentos ligados a cada porta do switch).

Através da informação contida na sua tabela de endereços MAC, o switch é capaz de fazer o redirecionamento de pacotes entre os equipamentos ligados a ele.

### 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 podemos ver pelas imagens seguintes, os equipamentos de cada departamento conseguem receber e enviar pacotes entre eles e o servidor do departamento C. Sendo assim, conclui-se que existe conectividade IP entre os mesmos.

```
root@v7:/tmp/pgcore.42739/v7.conf# ping 10.0.5.10
PING 10.0.5.10 (10.0.5.10) 56(84) bytes of data:
64 bytes from 10.0.5.10: icmp_seq=1 ttl=64 time=0.050 ms
64 bytes from 10.0.5.10: icmp_seq=2 ttl=64 time=0.036 ms
64 bytes from 10.0.5.10: icmp_seq=3 ttl=64 time=0.037 ms
^C
--- 10.0.5.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2035ms
rtt min/avg/max/mdev = 0.036/0.041/0.050/0.006 ms
```

Departamento A

```
root@v13:/tmp/pgcore.42739/v13.conf# ping 10.0.5.10
PING 10.0.5.10 (10.0.5.10) 56(84) bytes of data:
64 bytes from 10.0.5.10: icmp_seq=1 ttl=62 time=0.064 ms
64 bytes from 10.0.5.10: icmp_seq=2 ttl=62 time=0.044 ms
64 bytes from 10.0.5.10: icmp_seq=3 ttl=62 time=0.046 ms
^C
--- 10.0.5.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2036ms
rtt min/avg/max/mdev = 0.044/0.051/0.064/0.010 ms
```

Departamento B

```
root@v10:/tmp/pgcore.42739/v10.conf# ping 10.0.5.10
PING 10.0.5.10 (10.0.5.10) 56(84) bytes of data:
64 bytes from 10.0.5.10: icmp_seq=1 ttl=62 time=0.079 ms
64 bytes from 10.0.5.10: icmp_seq=2 ttl=62 time=0.052 ms
64 bytes from 10.0.5.10: icmp_seq=3 ttl=62 time=0.051 ms
^C
--- 10.0.5.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2038ms
rtt min/avg/max/mdev = 0.051/0.060/0.079/0.015 ms
```

Departamento C

```
root@v16:/tmp/pgcore.42739/v16.conf# ping 10.0.5.10
PING 10.0.5.10 (10.0.5.10) 56(84) bytes of data:
64 bytes from 10.0.5.10: icmp_seq=1 ttl=61 time=0.074 ms
64 bytes from 10.0.5.10: icmp_seq=2 ttl=61 time=0.043 ms
64 bytes from 10.0.5.10: icmp_seq=3 ttl=61 time=0.061 ms
^C
--- 10.0.5.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2037ms
rtt min/avg/max/mdev = 0.043/0.061/0.074/0.012 ms
```

Departamento D

Figura 18: Ping dos departamentos para o servidor S1

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

Usando o mesmo método da pergunta anterior, verifica-se que existe conectividade IP entre o router RISP e o servidor S1.

```
root@Risp:/tmp/pscore_42739/Risp.conf# ping 10.0.5.10
PING 10.0.5.10 (10.0.5.10) 56(84) bytes of data:
64 bytes from 10.0.5.10: icmp_seq=1 ttl=63 time=0.070 ms
64 bytes from 10.0.5.10: icmp_seq=2 ttl=63 time=0.039 ms
64 bytes from 10.0.5.10: icmp_seq=3 ttl=63 time=0.042 ms
^C
--- 10.0.5.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2061ms
rtt min/avg/max/mdev = 0.039/0.050/0.070/0.015 ms
```

Figura 19: Ping do Router RISP para o servidor S1

## 2.2 Pergunta 2

Para o router e um laptop do departamento C:

**A - 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. Se necessário, consulte o manual respetivo (`man netstat`).**

A tabela de encaminhamento do laptop tem duas entradas. A linha da coluna 'Destination' com valor 0.0.0.0 (default) informa que um pacote que pretenda ser enviado para uma rede desconhecida deve seguir pelo respetivo Gateway. A outra linha, com o valor 10.0.6.0, é usada quando se pretende enviar um pacote para a própria rede.

```
root@l10:/tmp/pscore_43815/l10.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
```

Figura 20: Tabela de encaminhamento do laptop

A tabela de encaminhamento do router contém as redes alcançáveis pelo router e os respectivos gateways. Quando o destino de um pacote corresponde a um dos valores da coluna 'Destination', o pacote é enviado pelo respectivo valor na coluna 'Gateway'.

```
root@Rc:/tmp/pycore.43815/Rc.conf# netstat -rn
```

Destination	Gateway	Genmask	Flags	MSS Window	irtt	Iface
10.0.0.0	0.0.0.0	255.255.255.0	U	0 0		eth0
10.0.1.0	0.0.0.0	255.255.255.0	U	0 0		eth1
10.0.2.0	10.0.1.2	255.255.255.0	UG	0 0		eth1
10.0.3.0	10.0.0.1	255.255.255.0	UG	0 0		eth0
10.0.4.0	10.0.0.1	255.255.255.0	UG	0 0		eth0
10.0.5.0	10.0.0.1	255.255.255.0	UG	0 0		eth0
10.0.6.0	0.0.0.0	255.255.255.0	U	0 0		eth2
10.0.7.0	10.0.0.1	255.255.255.0	UG	0 0		eth0
10.0.8.0	10.0.1.2	255.255.255.0	UG	0 0		eth1

Figura 21: Tabela de encaminhamento do router

**B - Diga, justificando, se está a ser usado encaminhamento estático ou dinâmico (sugestão: analise que processos estão a correr em cada sistema, por exemplo, `ps -ax`).**

No router, está a ser usado encaminhamento dinâmico, pois o comando “`ps -ax`” mostra que estão a correr processos com o protocolo OSPF e ZEBRA. No encaminhamento dinâmico os routers trocam informação de routing entre si, sendo as rotas atualizadas ao longo do tempo.

```
root@Rc:/tmp/pycore.43815/Rc.conf# ps -ax
```

PID	TTY	STAT	TIME	COMMAND
1	?	S	0:00	/usr/local/bin/vnoded -v -c /tmp/pycore.43815/Rc -l /
58	?	Ss	0:00	/usr/sbin/zebra -d
64	?	Ss	0:00	/usr/sbin/ospfd -d
68	?	Ss	0:00	/usr/sbin/ospfd -d
76	pts/2	Ss	0:00	/bin/bash
88	pts/2	R+	0:00	ps -ax

Figura 22: Processos a correr no router

Pelo contrário, no laptop o encaminhamento é estático, pois apenas estão a correr os processos básicos da máquina. No encaminhamento estático as rotas permanecem fixas e são baseadas nas rotas pré-definidas.

```
root@n10:/tmp/pycore.43815/n10.conf# ps -ax
```

PID	TTY	STAT	TIME	COMMAND
1	?	S	0:00	/usr/local/bin/vnoded -v -c /tmp/pycore.43815/n10 -l
19	pts/4	Ss	0:00	/bin/bash
28	pts/4	R+	0:00	ps -ax

Figura 23: Processos a correr no laptop

**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.**

Ao retirar a rota por defeito, o servidor S1 deixa de conseguir comunicar com equipamentos fora da sua rede (10.0.5.0). Isto acontece pois a tabela de encaminhamento do servidor passa a conter apenas informação de como enviar pacotes para a rede 10.0.5.0. Todos os pacotes com destino diferente são descartados, pois o servidor não sabe para onde os enviar.

```
root@S1:/tmp/pqcore.43815/S1_conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
0.0.0.0 10.0.5.1 0.0.0.0 UG 0 0 0 eth0
10.0.5.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
root@S1:/tmp/pqcore.43815/S1_conf# route delete default
root@S1:/tmp/pqcore.43815/S1_conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
10.0.5.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
```

Figura 24: Tabela de encaminhamento do servidor S1

**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 registe os comandos que usou.**

```
root@S1:/tmp/pqcore.43815/S1_conf# route add -net 10.0.4.0 netmask 255.255.255.0 gw 10.0.5.1
root@S1:/tmp/pqcore.43815/S1_conf# route add -net 10.0.7.0 netmask 255.255.255.0 gw 10.0.5.1
root@S1:/tmp/pqcore.43815/S1_conf# route add -net 10.0.6.0 netmask 255.255.255.0 gw 10.0.5.1
root@S1:/tmp/pqcore.43815/S1_conf# route add -net 10.0.8.0 netmask 255.255.255.0 gw 10.0.5.1
root@S1:/tmp/pqcore.43815/S1_conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
10.0.4.0 10.0.5.1 255.255.255.0 UG 0 0 0 eth0
10.0.5.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
10.0.6.0 10.0.5.1 255.255.255.0 UG 0 0 0 eth0
10.0.7.0 10.0.5.1 255.255.255.0 UG 0 0 0 eth0
10.0.8.0 10.0.5.1 255.255.255.0 UG 0 0 0 eth0
```

Figura 25: Rotas adicionadas para os departamentos e para o router de acesso



**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.**

Nas imagens seguintes verifica-se que os diversos equipamentos (Laptops dos departamentos B, C e D e router RISP) conseguem transmitir pacotes com o servidor S1, concluindo-se que este está novamente acessível.

```
root@r13:/tmp/pgcore-43815/r13.conf# ping 10.0.5.10
PING 10.0.5.10 (10.0.5.10) 56(84) bytes of data.
64 bytes from 10.0.5.10: icmp_seq=1 ttl=62 time=0,155 ms
64 bytes from 10.0.5.10: icmp_seq=2 ttl=62 time=0,045 ms
64 bytes from 10.0.5.10: icmp_seq=3 ttl=62 time=0,054 ms
^C
--- 10.0.5.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2032ms
rtt min/avg/max/mdev = 0,045/0,084/0,155/0,050 ms
```

Departamento B

```
root@r10:/tmp/pgcore-43815/r10.conf# ping 10.0.5.10
PING 10.0.5.10 (10.0.5.10) 56(84) bytes of data.
64 bytes from 10.0.5.10: icmp_seq=1 ttl=62 time=0,267 ms
64 bytes from 10.0.5.10: icmp_seq=2 ttl=62 time=0,043 ms
64 bytes from 10.0.5.10: icmp_seq=3 ttl=62 time=0,047 ms
^C
--- 10.0.5.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2055ms
rtt min/avg/max/mdev = 0,043/0,119/0,267/0,104 ms
```

Departamento C

```
root@r16:/tmp/pgcore-43815/r16.conf# ping 10.0.5.10
PING 10.0.5.10 (10.0.5.10) 56(84) bytes of data.
64 bytes from 10.0.5.10: icmp_seq=1 ttl=61 time=0,080 ms
64 bytes from 10.0.5.10: icmp_seq=2 ttl=61 time=0,060 ms
64 bytes from 10.0.5.10: icmp_seq=3 ttl=61 time=0,060 ms
^C
--- 10.0.5.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2033ms
rtt min/avg/max/mdev = 0,060/0,066/0,080/0,013 ms
```

Departamento D

```
root@R1sp:/tmp/pgcore-43815/R1sp.conf# ping 10.0.5.10
PING 10.0.5.10 (10.0.5.10) 56(84) bytes of data.
64 bytes from 10.0.5.10: icmp_seq=1 ttl=63 time=0,036 ms
64 bytes from 10.0.5.10: icmp_seq=2 ttl=63 time=0,040 ms
64 bytes from 10.0.5.10: icmp_seq=3 ttl=63 time=0,040 ms
^C
--- 10.0.5.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2051ms
rtt min/avg/max/mdev = 0,036/0,038/0,040/0,007 ms
```

Router RISP

Figura 26: Ping dos departamentos e do router RISP para o servidor S1

### 3 Definição de Sub-redes

Considere a topologia definida anteriormente. Assuma que o endereçamento entre os routers se mantém inalterado, contudo, o endereçamento em cada departamento deve ser redefinido.

#### 3.1 Pergunta 1

Considere que dispõe apenas do endereço de rede IP 130.XX.96.0/19, em que XX é o decimal correspondendo ao seu número de grupo (PLXX). 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 nosso endereço de rede é 130.52.96.0/19. Uma vez que temos uma máscara de 19 significa que a nossa rede pode usar todos os endereços entre 130.52.96.0 e 130.52.127.255.

Como temos 4 departamentos, são necessários 2 bits para conseguir atribuir um endereço a cada departamento (sub-netting). No entanto, é boa prática deixar um endereço disponível entre cada sub-rede, caso seja preciso aumentar o número de hosts de um departamento. Desta forma, garante-se que, à medida que os IPs aumentam, a ordem alfabética dos departamentos também.

Usando 3 bits, temos então 8 opções disponíveis para fazer sub-netting. Assim, distribuímos os endereços seguindo o modelo **130.52.011XXX00.0**, em que XXX são os bits atribuídos a cada departamento.

000	Dep A
001	Livre (Dep A)
010	Dep B
011	Livre (Dep B)
100	Dep C
101	Livre (Dep C)
110	Dep D
111	Livre (Dep D)

Tabela 1: Bits atribuídos a cada departamento

Dep.	IP	IP inicial	IP final
A	130.52.96.0/22	130.52.96.0	130.52.99.255
B	130.52.104.0/22	130.52.104.0	130.52.107.255
C	130.52.112.0/22	130.52.112.0	130.52.115.255
D	130.52.120.0/22	130.52.120.0	130.52.123.255

Tabela 2: IPs de host para cada departamento

Assim, conseguimos atribuir um IP a cada equipamento, dentro do intervalo disponibilizado para cada departamento (exceto os IPs com tudo a 0 e tudo a 1).

Dep A	IP atribuído	Dep B	IP atribuído
n7	130.52.96.2/22	n13	130.52.104.2/22
n8	130.52.96.3/22	n14	130.52.104.3/22
S1	130.52.96.4/22	Rb	130.52.104.1/22
Ra	130.52.96.1/22	-	-
Dep C	IP atribuído	Dep D	IP atribuído
n10	130.52.112.2/22	n16	130.52.120.2/22
n11	130.52.112.3/22	n17	130.52.120.3/22
Rc	130.52.112.1/22	Rd	130.52.120.1/22

Tabela 3: Endereços atribuídos aos equipamentos dos departamentos

### 3.2 Pergunta 2

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

Sendo que reservamos 3 bits para sub-netting, a nossa máscara passa de 19 para 22 bits, logo, em formato decimal, 255.255.252.0.

Como a máscara ocupa 22 bits, ficam disponíveis 10 bits. O número de hosts IP que se pode interligar em cada departamento é igual a:  $2^{10} - 2$ , em que '10' são os bits disponíveis e '2' são os IPs com tudo a 0 e tudo a 1. Assim ficam disponíveis 1022 IPs para cada departamento.

### 3.3 Pergunta 3

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

Começamos por atribuir o endereço IP manualmente a cada equipamento, seguindo os valores da Tabela 3 (Figura 27). Depois, para verificar a conectividade, usamos o comando ping de n7 para um laptop de cada departamento (Figura 28).

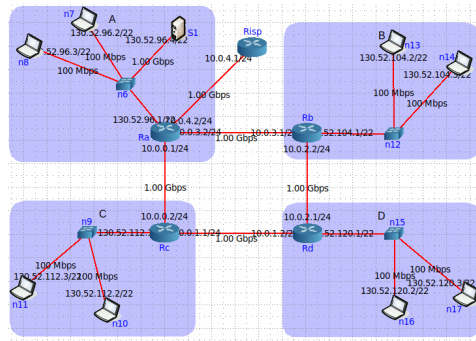


Figura 27: Topologia CORE com IPs atribuídos manualmente

```
root@n7:/tmp/pycore.37585/n7.conf# ping 130.52.96,3 Dep A
PING 130.52.96,3 (130.52.96,3) 56(84) bytes of data.
64 bytes from 130.52.96,3: icmp_seq=1 ttl=64 time=0,064 ms
64 bytes from 130.52.96,3: icmp_seq=2 ttl=64 time=0,125 ms
^C
--- 130.52.96,3 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1027ms
rtt min/avg/max/mdev = 0,064/0,094/0,125/0,032 ms
root@n7:/tmp/pycore.37585/n7.conf# ping 130.52,104,2 Dep B
PING 130.52,104,2 (130.52,104,2) 56(84) bytes of data.
64 bytes from 130.52,104,2: icmp_seq=1 ttl=62 time=0,154 ms
64 bytes from 130.52,104,2: icmp_seq=2 ttl=62 time=0,197 ms
^C
--- 130.52,104,2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1025ms
rtt min/avg/max/mdev = 0,154/0,175/0,197/0,025 ms
root@n7:/tmp/pycore.37585/n7.conf# ping 130.52,112,2 Dep C
PING 130.52,112,2 (130.52,112,2) 56(84) bytes of data.
64 bytes from 130.52,112,2: icmp_seq=1 ttl=62 time=0,114 ms
64 bytes from 130.52,112,2: icmp_seq=2 ttl=62 time=0,217 ms
^C
--- 130.52,112,2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1010ms
rtt min/avg/max/mdev = 0,114/0,165/0,217/0,053 ms
root@n7:/tmp/pycore.37585/n7.conf# ping 130.52,120,2 Dep D
PING 130.52,120,2 (130.52,120,2) 56(84) bytes of data.
64 bytes from 130.52,120,2: icmp_seq=1 ttl=61 time=0,113 ms
64 bytes from 130.52,120,2: icmp_seq=2 ttl=61 time=0,198 ms
^C
--- 130.52,120,2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1008ms
rtt min/avg/max/mdev = 0,113/0,155/0,198/0,044 ms
```

Figura 28: Ping de um laptop do Dep A (n7) para um laptop de cada departamento

## 4 Conclusão

Na primeira parte do trabalho, foi feita uma análise ao protocolo IPv4. Para tal, realizamos uma topologia CORE e analisamos o tráfego ICMP referente. Para além disso, estudamos o caso particular da fragmentação de pacotes IPv4. Nesta parte do trabalho, consolidamos os nossos conhecimentos sobre a transmissão de pacotes entre vários equipamentos ligados a uma rede, em específico, usando o protocolo IPv4.

Na segunda parte, construímos outra topologia CORE, com 4 departamentos distintos, de forma a estudar o comportamento destes entre si, nomeadamente das rotas que os pacotes seguem. Vimos ainda como poderíamos manipular endereços IP, podendo assim criar sub-redes personalizadas. Nesta parte, percebemos melhor como funcionam as interações entre sub-redes e aprendemos a manipular as rotas entre estas. Deparamo-nos, ainda, com a dificuldade de atribuir endereços IP manualmente a equipamentos de sub-redes.