

UNIVERSIDADE DO MINHO
DEPARTAMENTO DE INFORMÁTICA
REDES DE COMPUTADORES

Trabalho 2

Simão Brito (A89482)
António Silva (A89558)
José Martins (A90122)

1 de fevereiro de 2021



Simão Brito A89482



António Silva A89558



José Martins A90122

Conteúdo

1	Parte 1	4
1.1	Exercício 1	4
1.1.1	a. Active o wireshark ou o tcpdump no Cliente1. Numa shell do Cliente1, execute o comando traceroute -I para o endereço IP do Servidor1.	4
1.1.2	b. Registe e analise o tráfego ICMP enviado pelo Cliente1 e o tráfego ICMP recebido como resposta. Comente os resultados face ao comportamento esperado.	5
1.1.3	c. Qual deve ser o valor inicial mínimo do campo TTL para alcançar o Servidor1? Verifique na prática que a sua resposta está correta.	5
1.1.4	d. Calcule o valor médio do tempo de ida-e-volta (Round-Trip Time) obtido?	5
1.2	Exercício 2	6
1.2.1	a. Qual é o endereço IP da interface ativa do seu computador?	6
1.2.2	b. Qual é o valor do campo protocolo? O que identifica?	6
1.2.3	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?	6
1.2.4	d. O datagrama IP foi fragmentado? Justifique.	6
1.2.5	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.	6
1.2.6	f. Observa algum padrão nos valores do campo de Identificação do datagrama IP e TTL?	7
1.2.7	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?	7
1.3	Exercício 3	7
1.3.1	a. Localize a primeira mensagem ICMP. Porque é que houve necessidade de fragmentar o pacote inicial?	7
1.3.2	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?	8
1.3.3	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?	8
1.3.4	d. Quantos fragmentos foram criados a partir do datagrama original? Como se detecta o último fragmento correspondente ao datagrama original?	9
1.3.5	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.	9

2	Parte 2	10
2.1	Exercício 1	10
2.1.1	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.	10
2.1.2	b. Tratam-se de endereços públicos ou privados? Porquê?	11
2.1.3	c. Porque razão não é atribuído um endereço IP aos switches?	11
2.1.4	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).	11
2.1.5	e. Verifique se existe conectividade IP do router de acesso RISP para o servidor S1	12
2.2	Exercício 2	13
2.2.1	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).	13
2.2.2	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).	14
2.2.3	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.	15
2.2.4	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.	16
2.2.5	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.	16
2.3	Exercício 3	17
2.3.1	a. 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.	17
2.3.2	b. Qual a máscara de rede que usou (em formato decimal)? Quantos hosts IP pode interligar em cada departamento? Justifique.	18
2.3.3	c. Garanta e verifique que conectividade IP entre as várias redes locais da organização MIEI-RC é mantida. Explique como procedeu.	18
2.4	Conclusão	19

Parte 1

1.1 Exercício 1

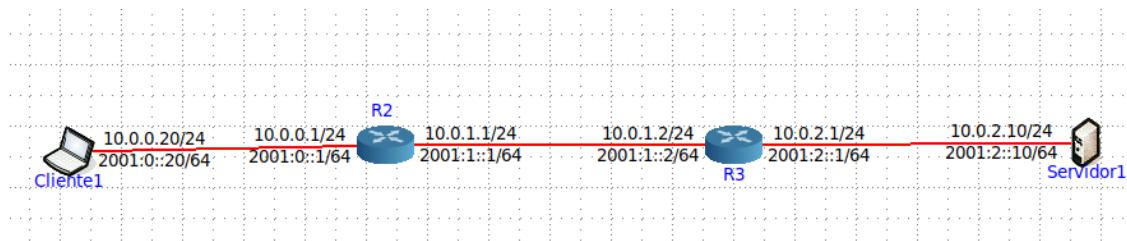


Figura 1.1: Topologia Core

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

```
root@Cliente1: /tmp/pycore.35761/Cliente1.conf
root@Cliente1:/tmp/pycore.35761/Cliente1.conf# 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) 0.034 ms 0.006 ms 0.004 ms
 2 10.0.1.2 (10.0.1.2) 0.015 ms 0.007 ms 0.007 ms
 3 10.0.2.10 (10.0.2.10) 0.015 ms 0.031 ms 0.009 ms
root@Cliente1:/tmp/pycore.35761/Cliente1.conf#
```

Figura 1.2: Resultados da execução do Traceroute

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

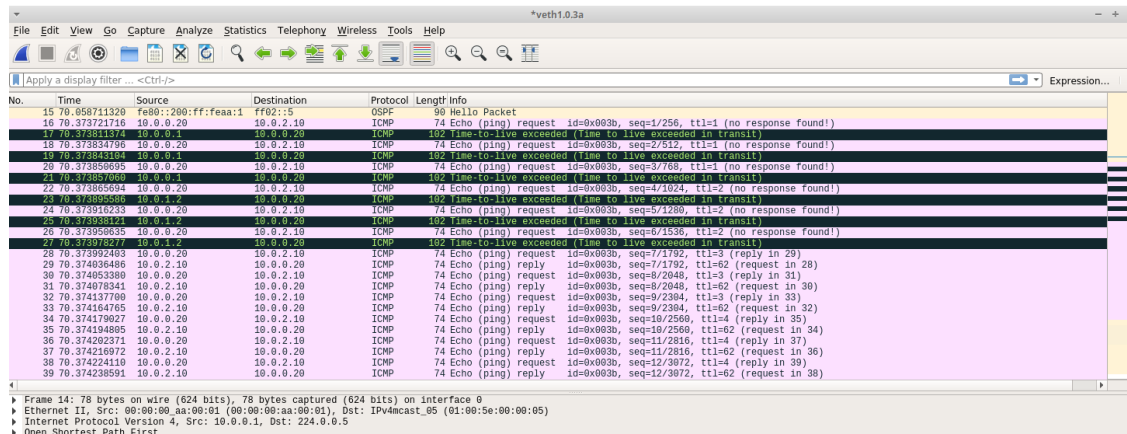


Figura 1.3: Análise tráfego wireshark

Pela topologia criada, seria de esperar que datagramas com TTL ≥ 3 não chegassem ao destino, uma vez que esta possui 2 routers, o que significa que vai ser decrementado o valor 2 no percurso até ao destino. Assim através da análise dos dados podemos concluir que ocorreu o esperado uma vez que datagramas com TTL = 1 ou TTL = 2 não conseguiram chegar ao servidor enviando uma mensagem de “time-to-live exceeded”. Também podemos verificar que o traceroute funciona como contávamos, uma vez que é enviado um ou mais datagramas com o campo TTL igual 1, depois com o TTL a 2; e assim sucessivamente.

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

TTL = 3, como está demonstrado nos dados recolhidos(fig 1.3), todos os datagramas que tem TTL maior que 2, proporcionam uma resposta do servidor de “time-to-live exceeded”.

1.1.4 d. Calcule o valor médio do tempo de ida-e-volta (Round-Trip Time) obtido?

Analisando a Figura 1.2 apresentada acima, podemos fazer o seguinte cálculo:

$$(0.015 + 0.031 + 0.009)/3 = 0.0156 \text{ ms}$$

Assim o valor médio do tempo de ida-e-volta é de 0.0156 ms.

1.2 Exercício 2

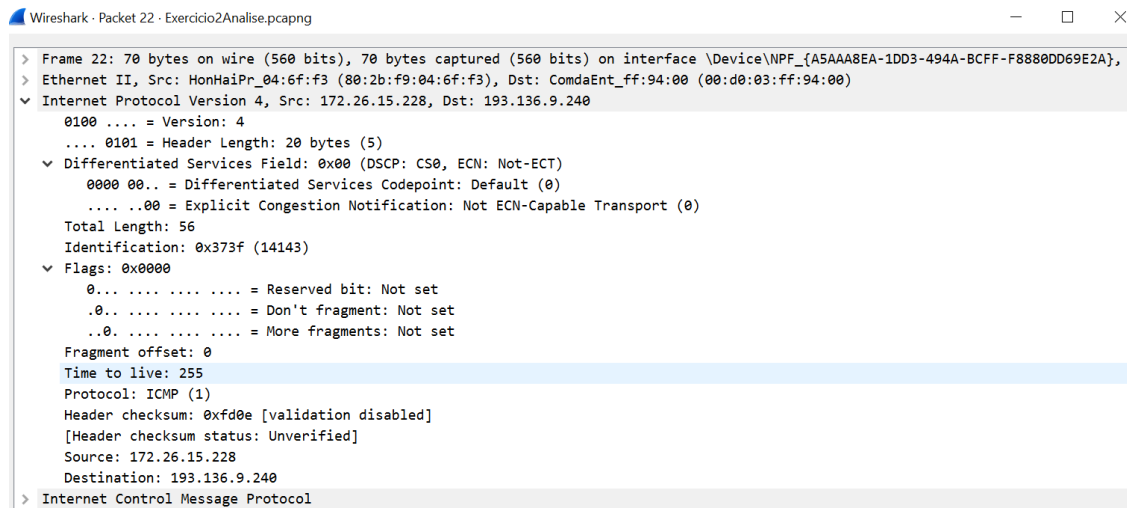


Figura 1.4: Cabeçalho da Comunicação

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

O endereço IP é 172.26.15.228.

1.2.2 b. Qual é o valor do campo protocolo? O que identifica?

O valor do campo protocolo que o identifica é ICMP(1). Identifica Internet Protocol.

1.2.3 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 (Header Length) tem 20 bytes. O payload corresponde à diferença entre a Total Length e a Head Length, pelo que tem 36 bytes (56-20).

1.2.4 d. O datagrama IP foi fragmentado? Justifique.

Não. Através dos dados recolhidos não é visível nenhuma mensagem de datagrama fragmentado e percebemos pela sua análise que os bits que correspondem ao “fragment offset” são igual a zero logo estamos no início do pacote. Para além disso, o campo “More Fragments” não está definido pelo que esta é a última parte do pacote. Assim, concluímos que o datagrama não é fragmentado.

1.2.5 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 de pacote para pacote são a Identificação, Time to live (TTL) e Header checksum.

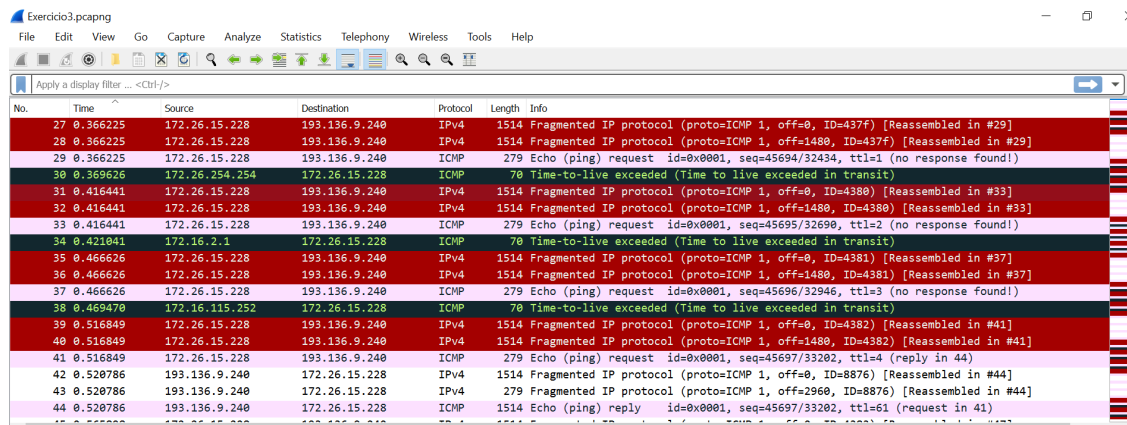
1.2.6 f. Observa algum padrão nos valores do campo de Identificação do datagrama IP e TTL?

A identificação incrementa sempre 1. No caso do TTL varia entre 1,2,3 e 255.

1.2.7 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?

O valor do campo TTL não é constante variando entre ff(255), fe(254) e fd(253). Este valor depende do router de origem a partir do qual a mensagem é enviada, ou seja, se a mensagem for enviada pelo primeiro router este valor será de 255, se for enviada pelo segundo router este valor será de 254 e pelo terceiro router será de 253.

1.3 Exercício 3



The image shows a Wireshark capture of network traffic. The packet list on the left shows several fragmented IP packets (27-30, 31-33, 34-36, 37-39, 40-42, 43-44) and ICMP Echo (ping) requests (29, 30, 33, 36, 39, 40, 43, 44). The packet details pane on the right shows the structure of the selected packet (No. 27), which is a fragmented IP packet (protocol=ICMP 1, offset=0, ID=437f) [Reassembled in #29]. The packet bytes pane on the right shows the raw data of the selected packet.

No.	Time	Source	Destination	Protocol	Length	Info
27	0.366225	172.26.15.228	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=437f) [Reassembled in #29]
28	0.366225	172.26.15.228	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=437f) [Reassembled in #29]
29	0.366225	172.26.15.228	193.136.9.240	ICMP	279	Echo (ping) request id=0x0001, seq=45694/32434, ttl=1 (no response found!)
30	0.369626	172.26.254.254	172.26.15.228	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
31	0.416441	172.26.15.228	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=4380) [Reassembled in #33]
32	0.416441	172.26.15.228	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=4380) [Reassembled in #33]
33	0.416441	172.26.15.228	193.136.9.240	ICMP	279	Echo (ping) request id=0x0001, seq=45695/32690, ttl=2 (no response found!)
34	0.421041	172.16.2.1	172.26.15.228	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
35	0.466626	172.26.15.228	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=4381) [Reassembled in #37]
36	0.466626	172.26.15.228	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=4381) [Reassembled in #37]
37	0.466626	172.26.15.228	193.136.9.240	ICMP	279	Echo (ping) request id=0x0001, seq=45696/32946, ttl=3 (no response found!)
38	0.469470	172.16.115.252	172.26.15.228	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
39	0.516849	172.26.15.228	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=4382) [Reassembled in #41]
40	0.516849	172.26.15.228	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=4382) [Reassembled in #41]
41	0.516849	172.26.15.228	193.136.9.240	ICMP	279	Echo (ping) request id=0x0001, seq=45697/33202, ttl=4 (reply in 44)
42	0.520786	193.136.9.240	172.26.15.228	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=8876) [Reassembled in #44]
43	0.520786	193.136.9.240	172.26.15.228	IPv4	279	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=8876) [Reassembled in #44]
44	0.520786	193.136.9.240	172.26.15.228	ICMP	1514	Echo (ping) reply id=0x0001, seq=45697/33202, ttl=61 (request in 41)

Figura 1.5: Fragmentos do datagrama IP

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

Mensagem com o numero 27(fig 1.5).

Houve necessidade de fragmentar o pacote inicial pois o tamanho da mensagem é superior ao limite máximo de transferência. Normalmente este valor é de 1500 bytes e pela análise de dados podemos verificar que a mensagem tem tamanho superior ao permitido pelo protocolo IPV4.

- 1.3.2 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?

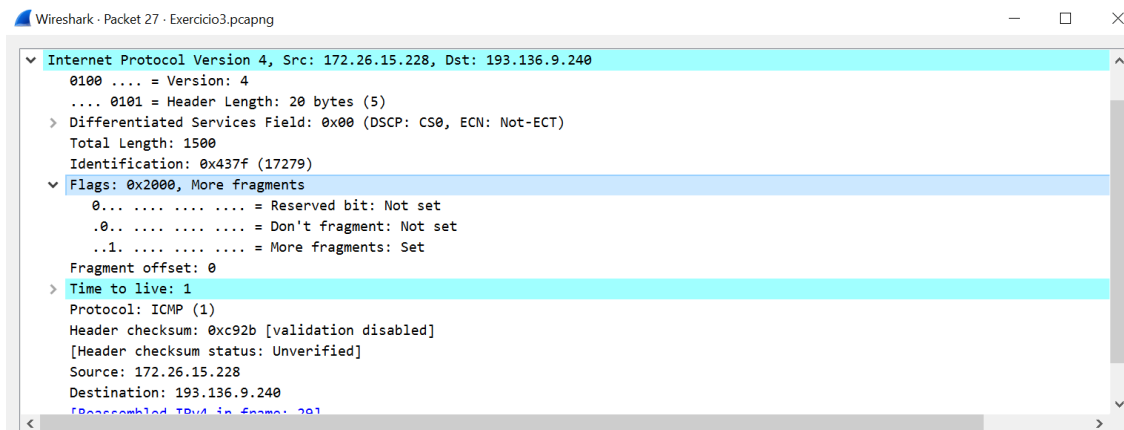


Figura 1.6: Cabeçalho do primeiro fragmento

No campo das flags, o bit que corresponde a "More Fragments" tem valor de 1, o que indica a fragmentação do datagrama. Uma vez que o "Fragment offset" é 0 então trata-se do primeiro fragmento. O tamanho é de 1500 bytes ("total length").

- 1.3.3 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?

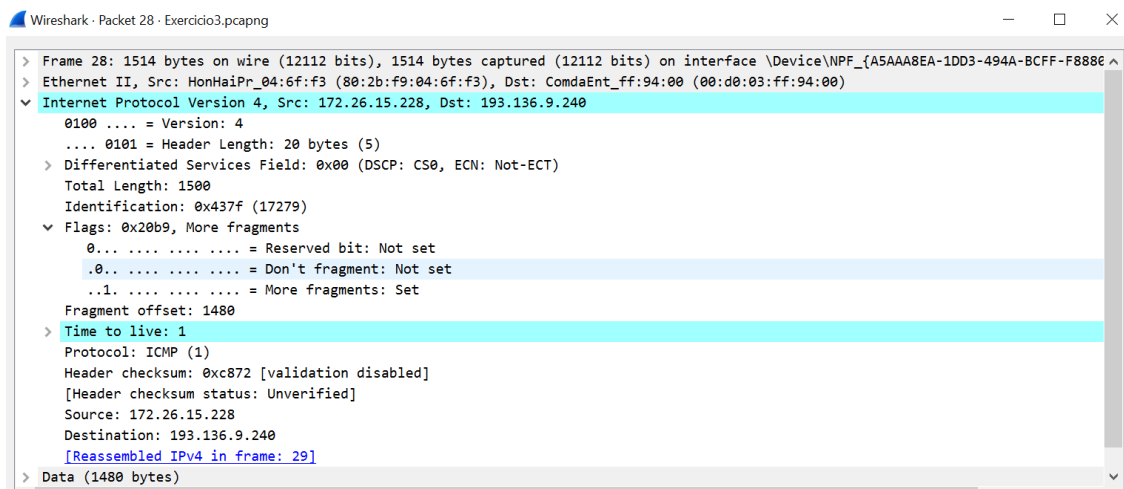


Figura 1.7: Cabeçalho do segundo fragmento

Não se trata do primeiro fragmento uma vez que o "Fragment offset" é diferente de 0 ("Fragment offset" = 1480.). Sim, pois o bit correspondente a "More Fragments" é 1.

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

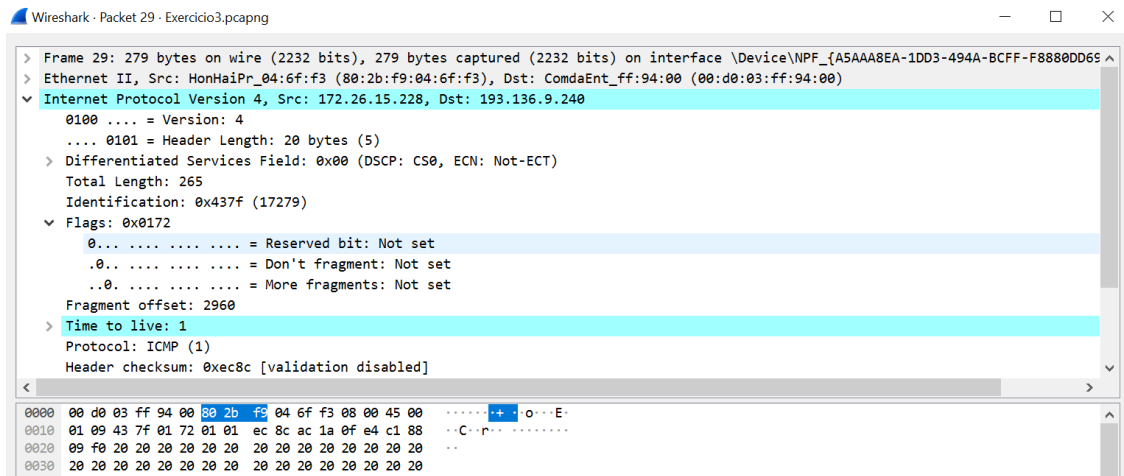


Figura 1.8: Cabeçalho do segundo fragmento

Foram criados 3 fragmentos. Uma vez que o bit correspondente a "More Fragments" é 0 então pode-se afirmar que já não há mais fragmentos, pelo que este é o último.

1.3.5 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 cabeçalhos IP dos 3 fragmentos varia o "More Fragments" e o "Fragment offset".

Parte 2

2.1 Exercício 1

- 2.1.1 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 máscara utilizada foi 255.255.255.0 uma vez que todos os equipamentos contêm o prefixo /24. Os endereços de cada equipamento são mostrados na figura. Nota: o CORE impossibilita uma ligação de 10Gbps, pelo que a ligação entre rA e rISP foi feita a 1Gbps.

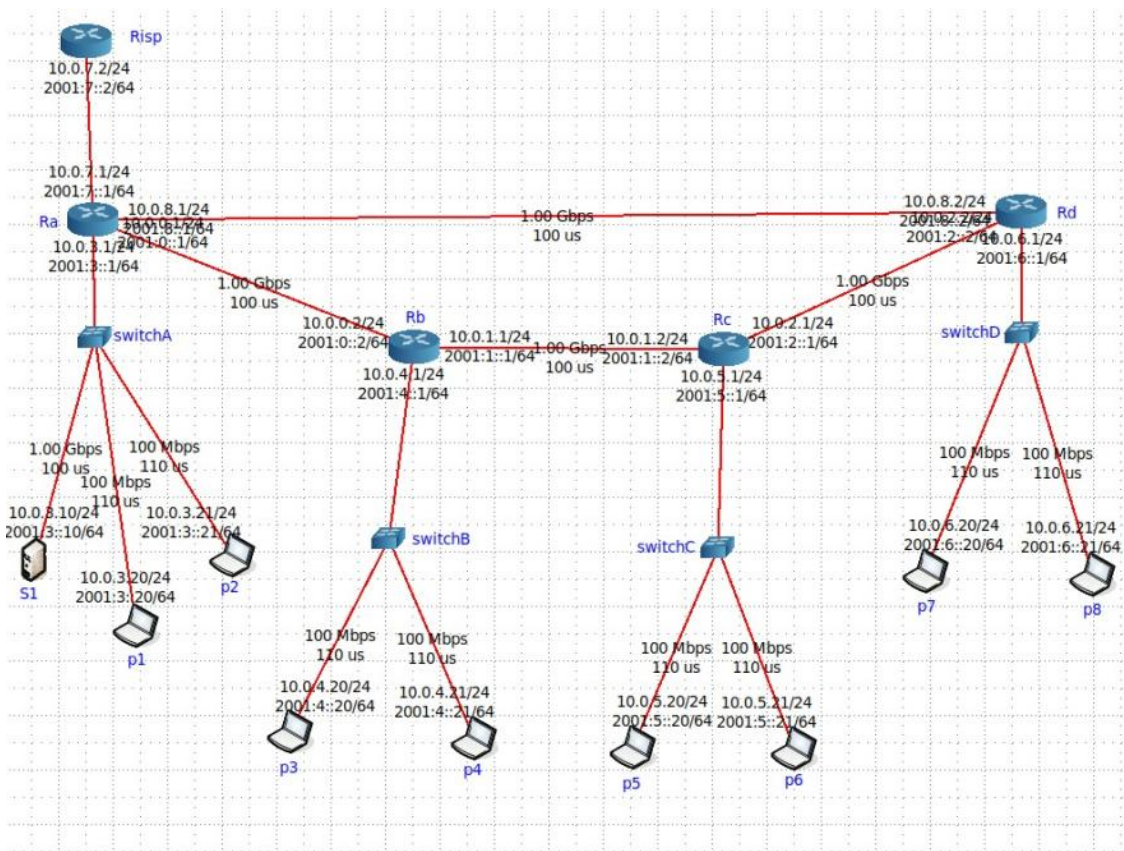


Figura 2.1: Topologia definida

2.1.2 b. Tratam-se de endereços públicos ou privados? Porquê?

São endereços privados uma vez que todos os endereços entre 10.0.0.0 e 10.255.255.255 são privados.

2.1.3 c. Porque razão não é atribuído um endereço IP aos switches?

Não é atribuído um endereço IP aos switches uma vez que estes operam numa camada abaixo (Layer 2). É de considerar que estes são Ethernet switches que comutam pacotes de Ethernet cujo nível não possui endereços de IP.

2.1.4 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).

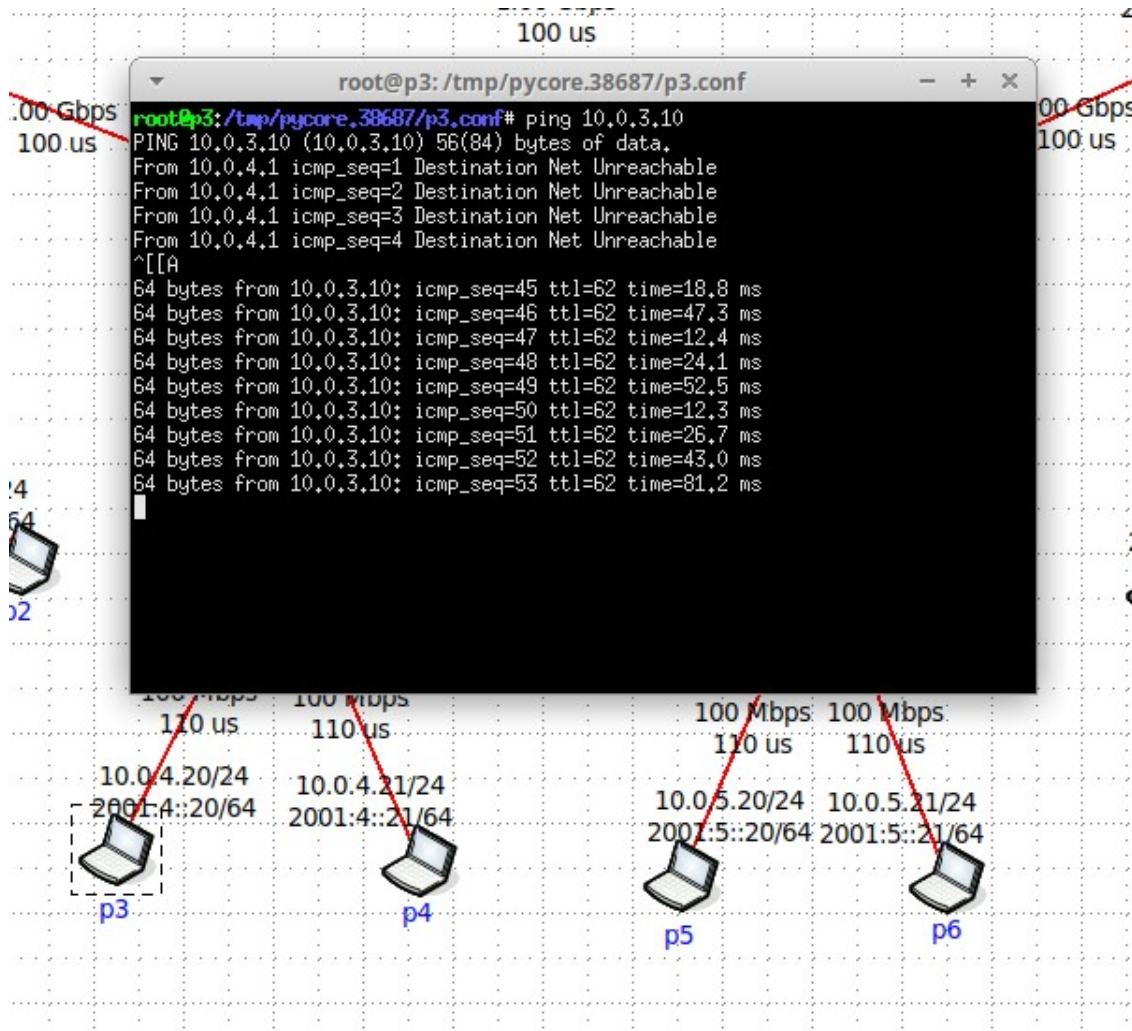
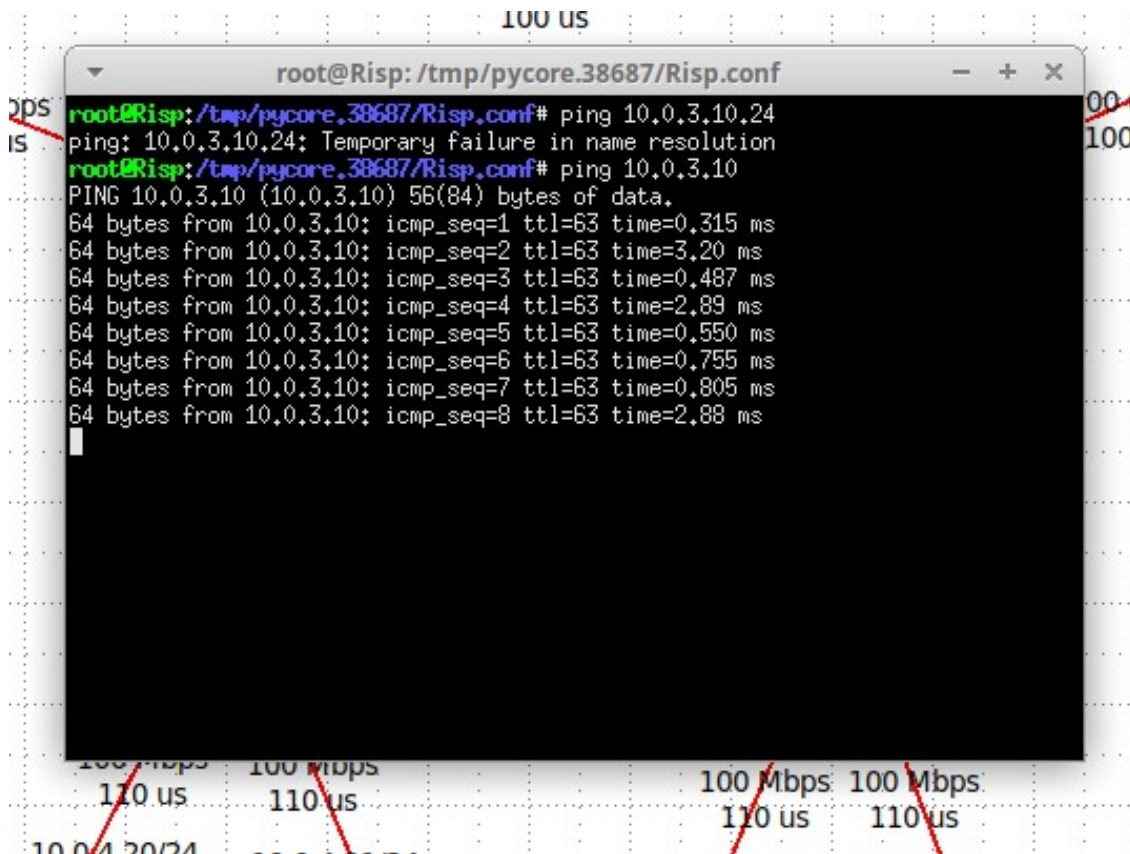


Figura 2.2: Utilização e resultado do comando ping

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

A terminal window titled 'root@Risp: /tmp/pycore.38687/Risp.conf' is shown. The terminal output displays the results of a ping command from the RISP router to the S1 server. The first attempt to ping 10.0.3.10,24 fails with a 'Temporary failure in name resolution' message. The second attempt to ping 10.0.3.10 is successful, showing 8 successful pings with varying response times. The terminal window is overlaid on a background that appears to be a network diagram or a grid with some text like '100 us' and '100 Mbps' visible through the window's transparency.

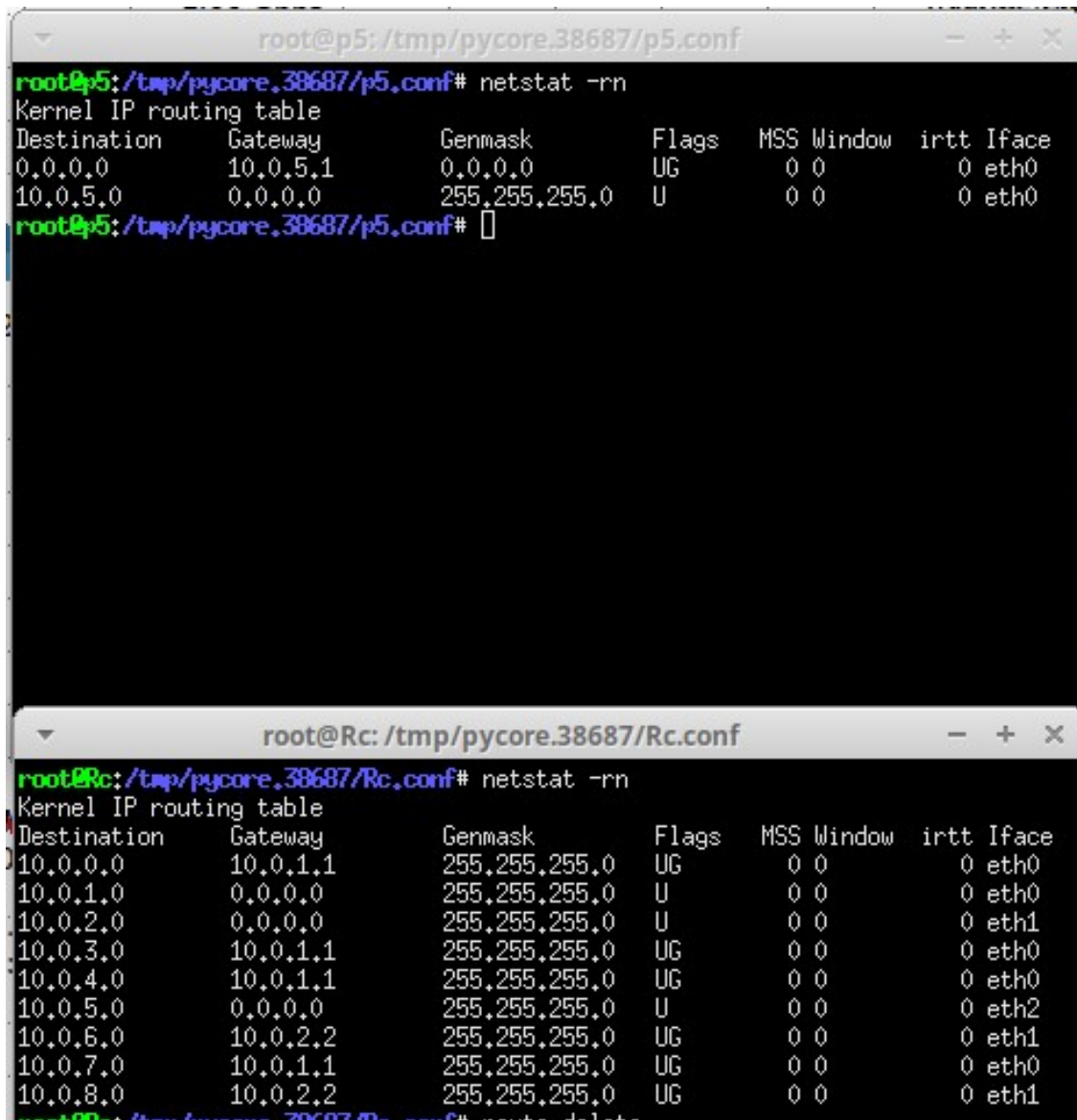
```
root@Risp: /tmp/pycore.38687/Risp.conf# ping 10.0.3.10,24
ping: 10.0.3.10,24: Temporary failure in name resolution
root@Risp: /tmp/pycore.38687/Risp.conf# ping 10.0.3.10
PING 10.0.3.10 (10.0.3.10) 56(84) bytes of data.
64 bytes from 10.0.3.10: icmp_seq=1 ttl=63 time=0.315 ms
64 bytes from 10.0.3.10: icmp_seq=2 ttl=63 time=3.20 ms
64 bytes from 10.0.3.10: icmp_seq=3 ttl=63 time=0.487 ms
64 bytes from 10.0.3.10: icmp_seq=4 ttl=63 time=2.89 ms
64 bytes from 10.0.3.10: icmp_seq=5 ttl=63 time=0.550 ms
64 bytes from 10.0.3.10: icmp_seq=6 ttl=63 time=0.755 ms
64 bytes from 10.0.3.10: icmp_seq=7 ttl=63 time=0.805 ms
64 bytes from 10.0.3.10: icmp_seq=8 ttl=63 time=2.88 ms
```

Figura 2.3: Verificação da conectividade IP entre router e S1

Como se pode ver na imagem acima confirma-se a conectividade IP do router de acesso RISP para o servidor S1.

2.2 Exercício 2

- 2.2.1 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`).



The image shows two terminal windows side-by-side. The top window is titled 'root@p5: /tmp/pycore.38687/p5.conf' and shows the output of 'netstat -rn' for a router. The bottom window is titled 'root@Rc: /tmp/pycore.38687/Rc.conf' and shows the output of 'netstat -rn' for a laptop.

```
root@p5: /tmp/pycore.38687/p5.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@p5: /tmp/pycore.38687/p5.conf#
```

```
root@Rc: /tmp/pycore.38687/Rc.conf# netstat -rn
Kernel IP routing table
Destination      Gateway         Genmask         Flags   MSS Window  irtt Iface
10.0.0.0         10.0.1.1        255.255.255.0   UG      0 0        0 eth0
10.0.1.0         0.0.0.0         255.255.255.0   U       0 0        0 eth0
10.0.2.0         0.0.0.0         255.255.255.0   U       0 0        0 eth1
10.0.3.0         10.0.1.1        255.255.255.0   UG      0 0        0 eth0
10.0.4.0         10.0.1.1        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 eth2
10.0.6.0         10.0.2.2        255.255.255.0   UG      0 0        0 eth1
10.0.7.0         10.0.1.1        255.255.255.0   UG      0 0        0 eth0
10.0.8.0         10.0.2.2        255.255.255.0   UG      0 0        0 eth1
root@Rc: /tmp/pycore.38687/Rc.conf# route delete
```

Figura 2.4: Tabelas de Encaminhamento do laptop do departamento C e do router do departamento C respetivamente

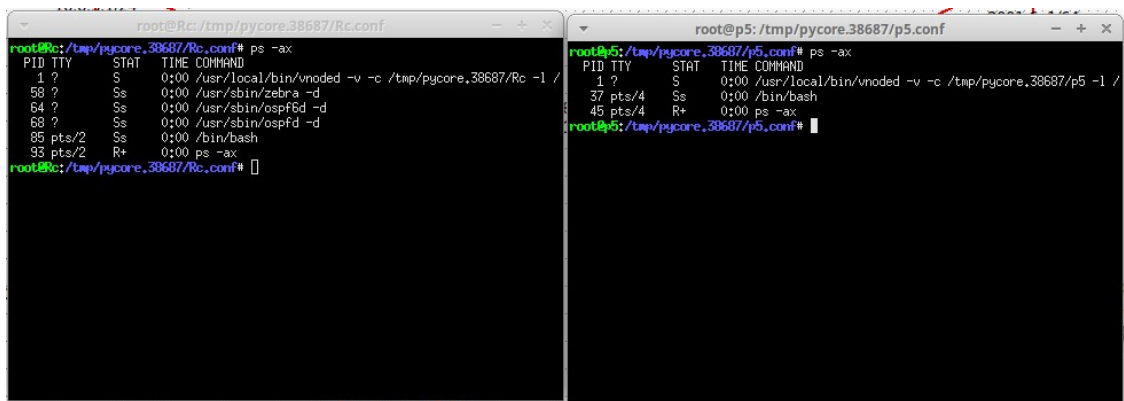
Análise da primeira tabela de encaminhamento relativa a um laptop do Departamento C:

Uma vez que a máscara é 0, o destino 0.0.0.0 identifica todas as redes possíveis, isto é, na eventualidade do tráfego não ser direcionado para o host 10.0.5.0 este é redirecionado por defeito para o Gateway 10.0.5.1.

Análise da segunda tabela de encaminhamento relativa ao router do Departamento C:

Podemos verificar que a primeira e segunda entrada da tabela diferem no Gateway (a primeira tem Gateway definido enquanto que a segunda tem o endereço default). Isto acontece pois, se os endereços não estiverem ligados diretamente é necessário saber qual o próximo salto, a passo que quando os endereços não estão ligados não é necessário que o Gateway esteja definido. Se observarmos as flags, estas dão-nos informação adicional, podendo ser U que indica que a route é válida mas o Gateway não está definido ou UG que indica que a route é válida mas redireciona o tráfego para um Gateway definido.

2.2.2 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).



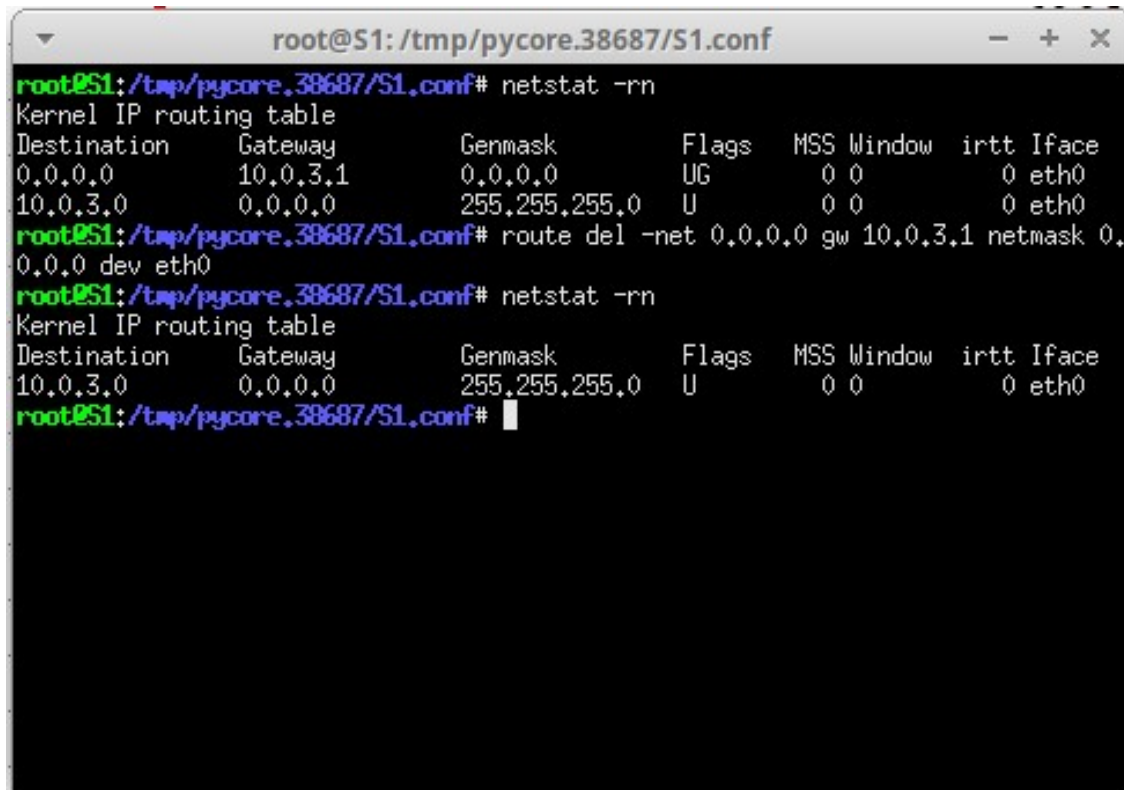
```
root@Rc:/tmp/pycore.38687/Rc.conf# ps -ax
PID TTY STAT TIME COMMAND
1 ? S 0:00 /usr/local/bin/vnoded -v -c /tmp/pycore.38687/Rc -l /
58 ? Ss 0:00 /usr/sbin/zebra -d
64 ? Ss 0:00 /usr/sbin/ospf6d -d
68 ? Ss 0:00 /usr/sbin/ospf6d -d
86 pts/2 Ss 0:00 /bin/bash
93 pts/2 R+ 0:00 ps -ax
root@Rc:/tmp/pycore.38687/Rc.conf#

root@p5:/tmp/pycore.38687/p5.conf# ps -ax
PID TTY STAT TIME COMMAND
1 ? S 0:00 /usr/local/bin/vnoded -v -c /tmp/pycore.38687/p5 -l /
37 pts/4 Ss 0:00 /bin/bash
45 pts/4 R+ 0:00 ps -ax
root@p5:/tmp/pycore.38687/p5.conf#
```

Figura 2.5: Análise de processos com comando ps -ax

Está a ser usado encaminhamento dinâmico, porque as rotas são definidas automaticamente através da troca de informação de routing entre routers.

- 2.2.3 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.38687/S1.conf
root@S1: /tmp/pycore.38687/S1.conf# netstat -rn
Kernel IP routing table
Destination      Gateway          Genmask          Flags      MSS Window  irtt Iface
0.0.0.0          10.0.3.1        0.0.0.0          UG          0 0        0 eth0
10.0.3.0         0.0.0.0         255.255.255.0    U           0 0        0 eth0
root@S1: /tmp/pycore.38687/S1.conf# route del -net 0.0.0.0 gw 10.0.3.1 netmask 0.0.0.0 dev eth0
root@S1: /tmp/pycore.38687/S1.conf# netstat -rn
Kernel IP routing table
Destination      Gateway          Genmask          Flags      MSS Window  irtt Iface
10.0.3.0         0.0.0.0         255.255.255.0    U           0 0        0 eth0
root@S1: /tmp/pycore.38687/S1.conf#
```

Figura 2.6: Remoção da rota por defeito da tabela de encaminhamento do servidor S1

O servidor S1 perde conectividade com todos os hosts que não pertencem à sua rede local, porque, uma vez que foi removida a rota por defeito, o servidor não tem definida a rota de envio de tráfego para redes não locais. Assim os utilizadores não conseguem receber dados do servidor.

- 2.2.4 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.38687/S1.conf# route add -net 10.0.4.0 netmask 255.255.255.0 gw 10.0.3.1
root@S1:/tmp/pycore.38687/S1.conf# route add -net 10.0.5.0 netmask 255.255.255.0 gw 10.0.3.1
root@S1:/tmp/pycore.38687/S1.conf# route add -net 10.0.6.0 netmask 255.255.255.0 gw 10.0.3.1
root@S1:/tmp/pycore.38687/S1.conf# route add 10.0.7.2 gw 10.0.3.1 eth0
root@S1:/tmp/pycore.38687/S1.conf# netstat -rn
Kernel IP routing table
Destination        Gateway           Genmask          Flags   MSS Window  irtt Iface
10.0.3.0           0.0.0.0          255.255.255.0    U        0 0        0 eth0
10.0.4.0           10.0.3.1         255.255.255.0    UG        0 0        0 eth0
10.0.5.0           10.0.3.1         255.255.255.0    UG        0 0        0 eth0
10.0.6.0           10.0.3.1         255.255.255.0    UG        0 0        0 eth0
10.0.7.2           10.0.3.1         255.255.255.255 UGH       0 0        0 eth0
root@S1:/tmp/pycore.38687/S1.conf# ping 10.0.4.21
```

Figura 2.7: Adição das rotas estáticas necessárias para restaurar a conectividade de S1

- 2.2.5 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@S1:/tmp/pycore.38687/S1.conf# ping 10.0.4.21
PING 10.0.4.21 (10.0.4.21) 56(84) bytes of data.
4 bytes from 10.0.4.21: icmp_seq=1 ttl=62 time=14.9 ms
4 bytes from 10.0.4.21: icmp_seq=2 ttl=62 time=36.7 ms
4 bytes from 10.0.4.21: icmp_seq=3 ttl=62 time=14.1 ms
C
-- 10.0.4.21 ping statistics --
3 packets transmitted, 3 received, 0% packet loss, time 2025ms
rtt min/avg/max/mdev = 14.198/21.978/36.751/10.451 ms
root@S1:/tmp/pycore.38687/S1.conf# ping 10.0.5.21
PING 10.0.5.21 (10.0.5.21) 56(84) bytes of data.
4 bytes from 10.0.5.21: icmp_seq=1 ttl=61 time=8.82 ms
4 bytes from 10.0.5.21: icmp_seq=2 ttl=61 time=4.97 ms
4 bytes from 10.0.5.21: icmp_seq=3 ttl=61 time=7.81 ms
C
-- 10.0.5.21 ping statistics --
3 packets transmitted, 3 received, 0% packet loss, time 2005ms
rtt min/avg/max/mdev = 4.970/7.205/8.828/1.634 ms
root@S1:/tmp/pycore.38687/S1.conf# ping 10.0.6.21
PING 10.0.6.21 (10.0.6.21) 56(84) bytes of data.
4 bytes from 10.0.6.21: icmp_seq=1 ttl=62 time=14.6 ms
4 bytes from 10.0.6.21: icmp_seq=2 ttl=62 time=74.6 ms
4 bytes from 10.0.6.21: icmp_seq=3 ttl=62 time=14.6 ms
C
-- 10.0.6.21 ping statistics --
3 packets transmitted, 3 received, 0% packet loss, time 2007ms
rtt min/avg/max/mdev = 14.605/34.632/74.667/28.309 ms
root@S1:/tmp/pycore.38687/S1.conf#
```

Figura 2.8: Remoção da rota por defeito da tabela de encaminhamento do servidor S1

Através da imagem acima percebemos que o Servidor S1 está novamente acessível.

2.3 Exercício 3

- 2.3.1 a. 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.

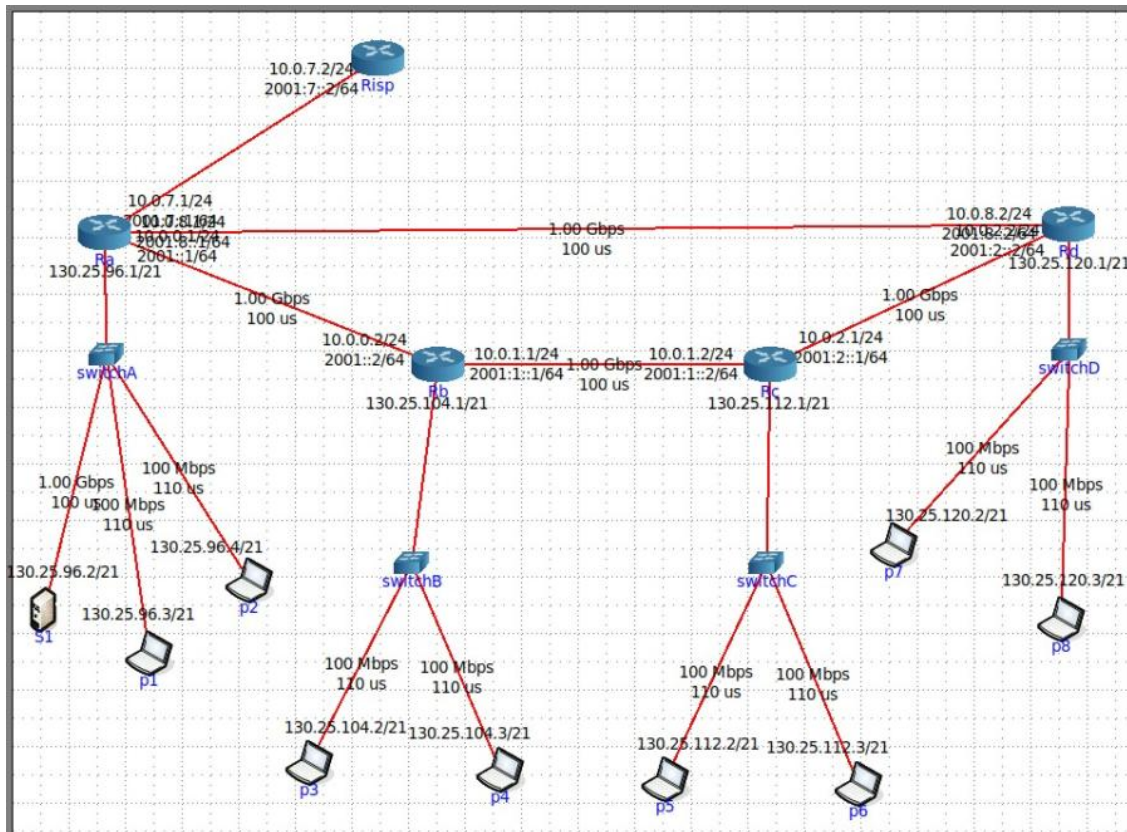


Figura 2.9: Topologia do novo esquema de endereçamento

No exercício proposto é-nos dito que dispomos apenas de um endereço /19, que precisamos dividir por 4 subredes. Assim para que isto fosse possível decidimos utilizar mais 2 bits para a representação dessas subredes(00/01/10/11). Feito isto, ficamos então com endereços /21. Este acréscimo de bits que fazemos ao endereço de origem vai também modificar o 3º octeto de bits dos endereços que formulamos como vemos abaixo:

3º octeto do Departamento:

A: 01100000 - 96

B: 01101000 - 104

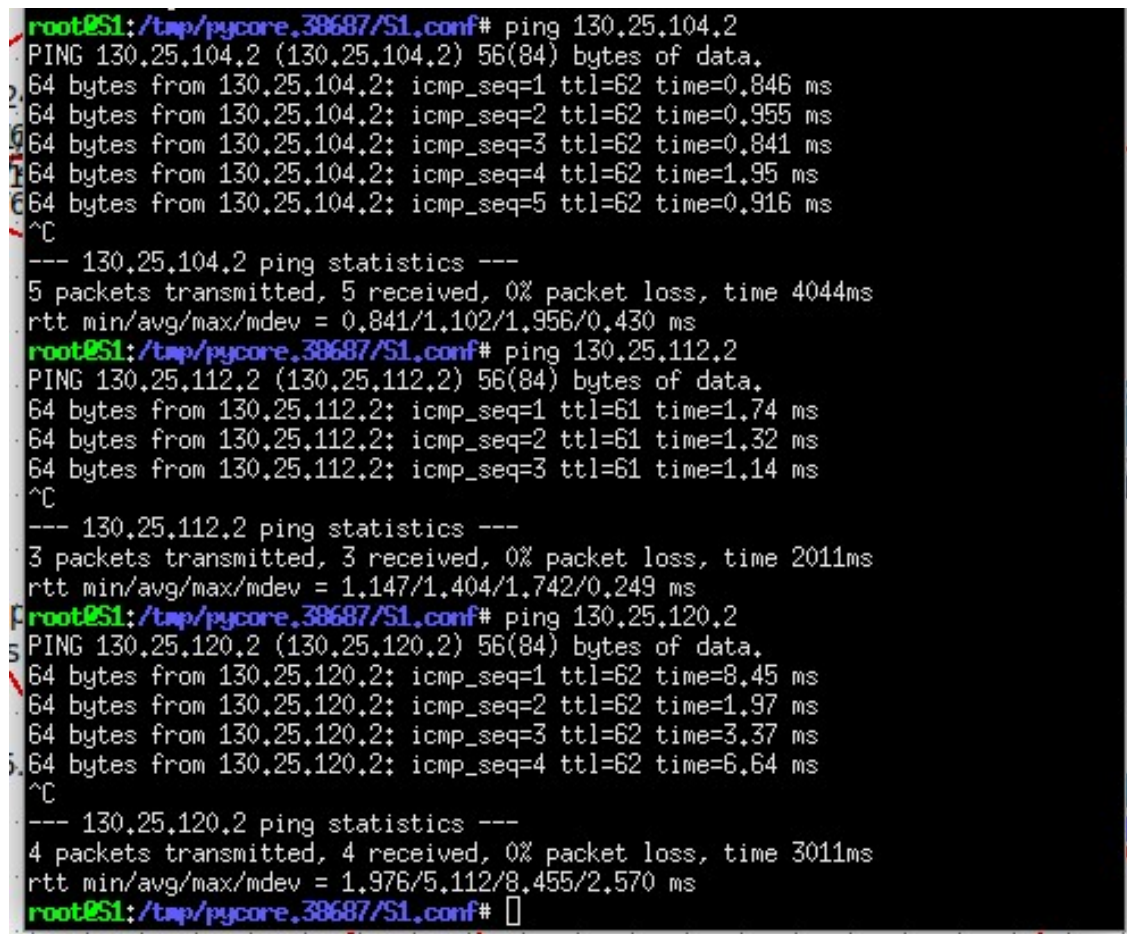
C: 01110000 - 112

D: 01111000 - 120

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

A máscara de rede foi /21 (255.255.248.0). Podemos interligar 2^{11} (2048) hosts IP em cada departamento.

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



```
root@S1:/tmp/pycore.38687/S1.conf# ping 130.25.104.2
PING 130.25.104.2 (130.25.104.2) 56(84) bytes of data.
64 bytes from 130.25.104.2: icmp_seq=1 ttl=62 time=0.846 ms
64 bytes from 130.25.104.2: icmp_seq=2 ttl=62 time=0.955 ms
64 bytes from 130.25.104.2: icmp_seq=3 ttl=62 time=0.841 ms
64 bytes from 130.25.104.2: icmp_seq=4 ttl=62 time=1.95 ms
64 bytes from 130.25.104.2: icmp_seq=5 ttl=62 time=0.916 ms
^C
--- 130.25.104.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4044ms
rtt min/avg/max/mdev = 0.841/1.102/1.956/0.430 ms
root@S1:/tmp/pycore.38687/S1.conf# ping 130.25.112.2
PING 130.25.112.2 (130.25.112.2) 56(84) bytes of data.
64 bytes from 130.25.112.2: icmp_seq=1 ttl=61 time=1.74 ms
64 bytes from 130.25.112.2: icmp_seq=2 ttl=61 time=1.32 ms
64 bytes from 130.25.112.2: icmp_seq=3 ttl=61 time=1.14 ms
^C
--- 130.25.112.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2011ms
rtt min/avg/max/mdev = 1.147/1.404/1.742/0.249 ms
root@S1:/tmp/pycore.38687/S1.conf# ping 130.25.120.2
PING 130.25.120.2 (130.25.120.2) 56(84) bytes of data.
64 bytes from 130.25.120.2: icmp_seq=1 ttl=62 time=8.45 ms
64 bytes from 130.25.120.2: icmp_seq=2 ttl=62 time=1.97 ms
64 bytes from 130.25.120.2: icmp_seq=3 ttl=62 time=3.37 ms
64 bytes from 130.25.120.2: icmp_seq=4 ttl=62 time=6.64 ms
^C
--- 130.25.120.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3011ms
rtt min/avg/max/mdev = 1.976/5.112/8.455/2.570 ms
root@S1:/tmp/pycore.38687/S1.conf#
```

Figura 2.10: Verificação da conectividade IP entre as redes locais de MIEI-RC

2.4 Conclusão

Este trabalho prático serviu de suporte às aulas teóricas e ajudou-nos a consolidar a matéria lecionada nas mesmas.

Através da máquina virtual disponibilizada tivemos a oportunidade de construir topologias CORE. Foi bastante esclarecedor para a nossa aprendizagem poder construir casos virtuais de tráfego ICMP.

No capítulo relativo ao IP, ficamos a perceber melhor como é que o TTL funciona através da análise de tráfego ICMP.

No que toca ao formato de um datagrama IP, identificamos com a ajuda das aulas teóricas os dois campos que o constituem: campo de dados(payload) e cabeçalho.

Averiguámos também a fragmentação de datagramas e percebemos a importância do campo relativo à fragmentation, bem como da análise das respetivas flags.

Tivemos de explorar também as diferenças entre endereços públicos e privados.

Analisamos tabelas de encaminhamento e retratamos as diferenças entre encaminhamento estático e dinâmico.

Por fim, abordamos os conceitos de subnetting, tendo posto à prova a nossa capacidade de divisão de endereços, identificação de subredes e manipulação de máscaras.