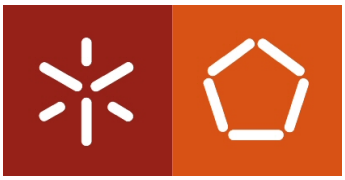


Universidade do Minho

Departamento de Informática

Relatório de Redes de Computadores
Grupo 3

Inês Alves (A81368) João Lopes (A80397)
Sofia Teixeira (a80624)

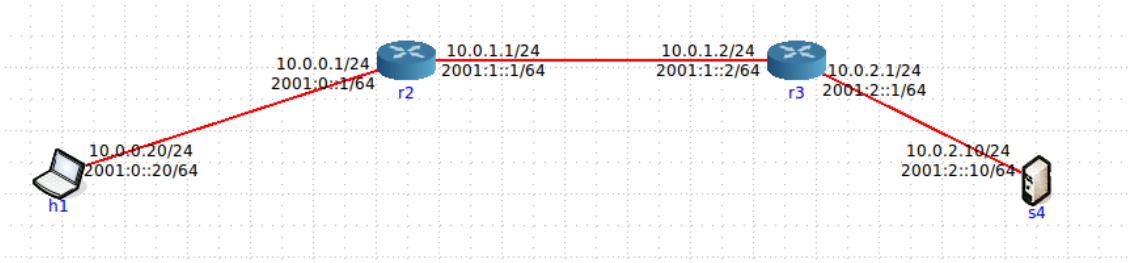
16 de Novembro de 2018

Nota: Nas questões em que era pedido utilizar o número do turno e o número do grupo, ou seja, PL 5, Grupo 3 (53), foi utilizado, por lapso, apenas o número do grupo, tendo ficado 03.

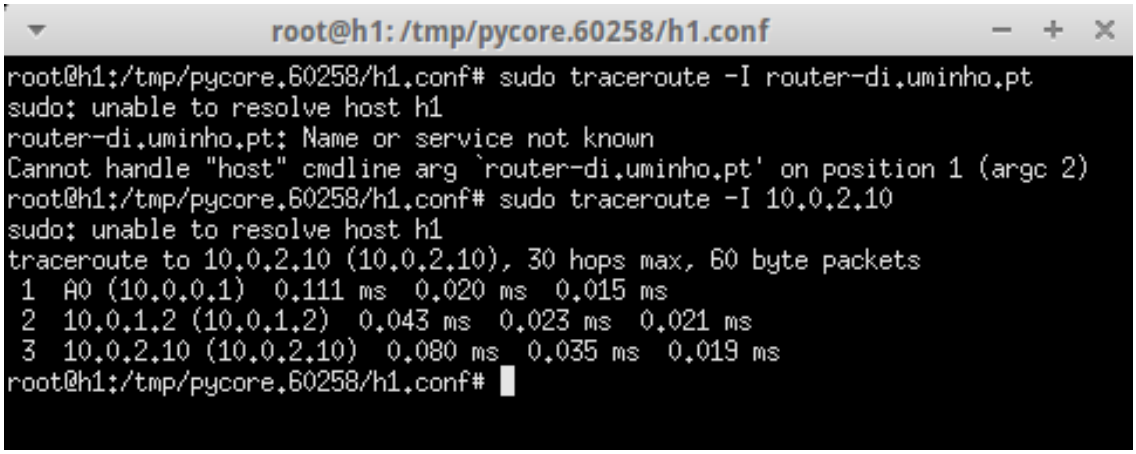
Conteúdo

1 Questões e Respostas: Parte 1

1.1 Pergunta 1.



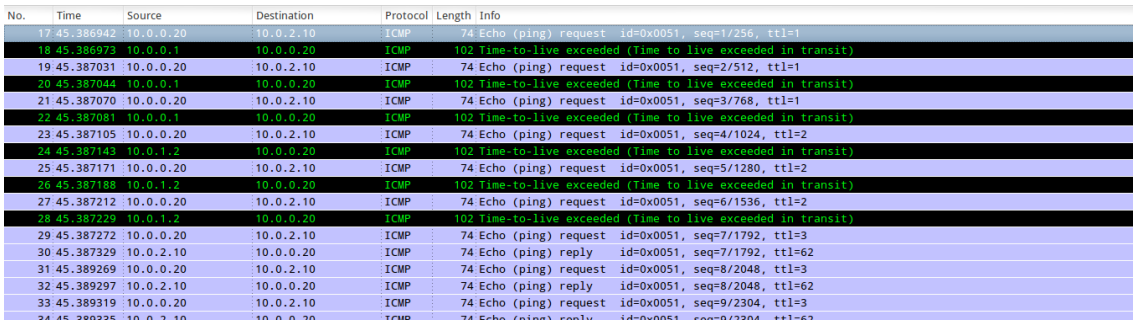
a) Active o wireshark ou o tcpdump no pc h1. Numa shell de h1, execute o comando traceroute -I para o endereço IP do host s4.



b) Registe e analise o tráfego ICMP enviado por h1 e o tráfego ICMP recebido como resposta. Comente os resultados face ao comportamento esperado.

H1 começa por enviar 3 pacotes com TTL = 1. De seguida, incrementa o TTL e envia novamente 3 pacotes e assim sucessivamente, como podemos observar na figura abaixo. Os pacotes vão precisar de efetuar 3 "saltos"para conseguir chegar a s4, logo vai precisar de TTL = 3, no mínimo. Assim, podemos observar na mesma figura que, efetivamente, enquanto o TTL é menor que 3 os pacotes dão erro de *time-to-live exceeded*.

Isto significa que TTL = 3 é o valor mínimo para alcançar s4.



No.	Time	Source	Destination	Protocol	Length	Info
17	45.386942	10.0.0.20	10.0.2.10	ICMP	74	Echo (ping) request id=0x0051, seq=1/256, ttl=1
18	45.386973	10.0.0.1	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
19	45.387031	10.0.0.20	10.0.2.10	ICMP	74	Echo (ping) request id=0x0051, seq=2/512, ttl=1
20	45.387044	10.0.0.1	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
21	45.387070	10.0.0.20	10.0.2.10	ICMP	74	Echo (ping) request id=0x0051, seq=3/768, ttl=1
22	45.387081	10.0.0.1	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
23	45.387105	10.0.0.20	10.0.2.10	ICMP	74	Echo (ping) request id=0x0051, seq=4/1024, ttl=2
24	45.387143	10.0.1.2	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
25	45.387171	10.0.0.20	10.0.2.10	ICMP	74	Echo (ping) request id=0x0051, seq=5/1280, ttl=2
26	45.387188	10.0.1.2	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
27	45.387212	10.0.0.20	10.0.2.10	ICMP	74	Echo (ping) request id=0x0051, seq=6/1536, ttl=2
28	45.387229	10.0.1.2	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
29	45.387272	10.0.0.20	10.0.2.10	ICMP	74	Echo (ping) request id=0x0051, seq=7/1792, ttl=3
30	45.387329	10.0.2.10	10.0.0.20	ICMP	74	Echo (ping) reply id=0x0051, seq=7/1792, ttl=62
31	45.389269	10.0.0.20	10.0.2.10	ICMP	74	Echo (ping) request id=0x0051, seq=8/2048, ttl=3
32	45.389297	10.0.2.10	10.0.0.20	ICMP	74	Echo (ping) reply id=0x0051, seq=8/2048, ttl=62
33	45.389319	10.0.0.20	10.0.2.10	ICMP	74	Echo (ping) request id=0x0051, seq=9/2304, ttl=3

c) Qual deve ser o valor inicial mínimo do campo TTL para alcançar o destino s4? Verifique na prática que a sua resposta está correta.

O valor inicial mínimo do campo TTL para alcançar o destino s4 tem que ser 3, como verificamos na alínea anterior.

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

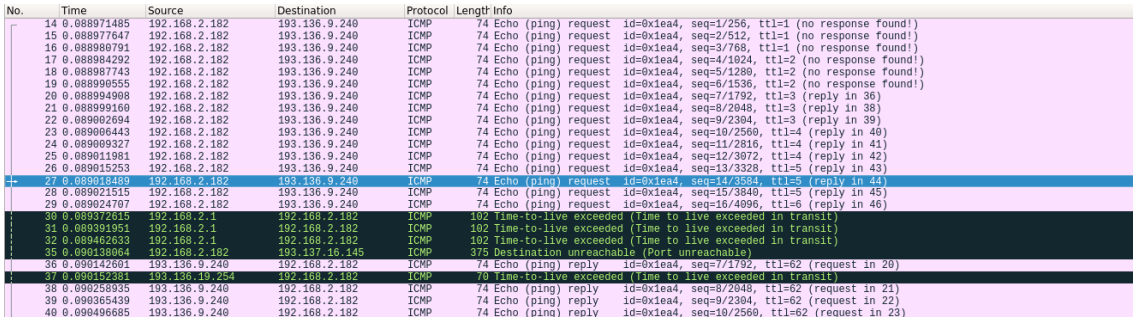
O valor médio do tempo de ida-e-volta é:

$$\frac{0.080 + 0.035 + 0.019}{3} = 0.0446667ms$$

1.2 Pergunta 2.

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

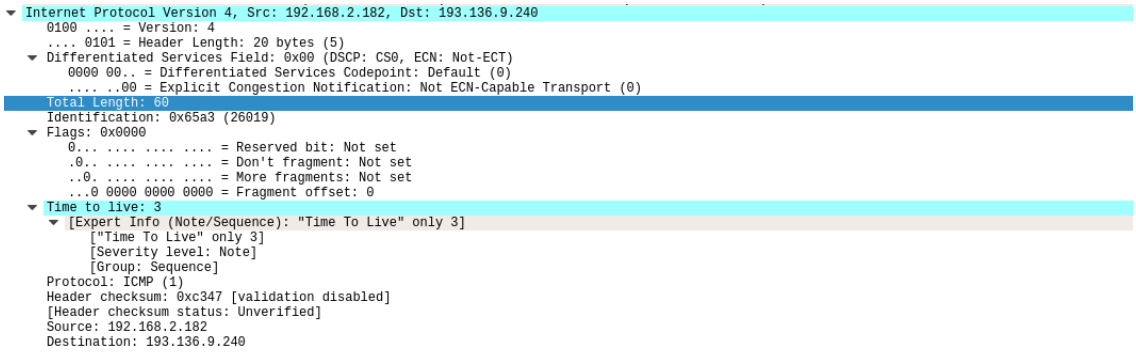
O endereço IP da interface ativa do nosso computador é 192.168.2.182.



No.	Time	Source	Destination	Protocol	Length	Info
14	0.088971485	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=1/256, ttl=1 (no response found!)
15	0.088977647	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=2/512, ttl=1 (no response found!)
16	0.088980791	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=3/768, ttl=1 (no response found!)
17	0.088984292	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=4/1024, ttl=2 (no response found!)
18	0.088987743	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=5/1280, ttl=2 (no response found!)
19	0.088990555	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=6/1536, ttl=2 (no response found!)
20	0.088994908	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=7/1792, ttl=3 (reply in 36)
21	0.088999160	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=8/2048, ttl=3 (reply in 38)
22	0.089002694	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=9/2304, ttl=3 (reply in 39)
23	0.089006443	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=10/2560, ttl=4 (reply in 40)
24	0.089009327	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=11/2816, ttl=4 (reply in 41)
25	0.089011981	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=12/3072, ttl=4 (reply in 42)
26	0.089015253	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=13/3328, ttl=5 (reply in 43)
27	0.089018489	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=14/3584, ttl=5 (reply in 44)
28	0.089021515	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=15/3840, ttl=5 (reply in 45)
29	0.089024707	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=16/4096, ttl=6 (reply in 46)
30	0.089372615	192.168.2.1	192.168.2.182	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
31	0.089391951	192.168.2.1	192.168.2.182	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
32	0.089482833	192.168.2.1	192.168.2.182	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
35	0.090138064	192.168.2.182	193.137.16.145	ICMP	375	Destination unreachable (Port unreachable)
36	0.090142601	193.136.9.240	192.168.2.182	ICMP	74	Echo (ping) reply id=0x1ea4, seq=7/1792, ttl=62 (request in 20)
37	0.090152381	193.136.19.254	192.168.2.182	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
38	0.090258935	193.136.9.240	192.168.2.182	ICMP	74	Echo (ping) reply id=0x1ea4, seq=8/2048, ttl=62 (request in 21)
39	0.090365439	193.136.9.240	192.168.2.182	ICMP	74	Echo (ping) reply id=0x1ea4, seq=9/2304, ttl=62 (request in 22)
40	0.090496685	193.136.9.240	192.168.2.182	ICMP	74	Echo (ping) reply id=0x1ea4, seq=10/2560, ttl=62 (request in 23)

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

O valor do campo protocolo é 1.



```
Internet Protocol Version 4, Src: 192.168.2.182, Dst: 193.136.9.240
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
      0000 00.. = Differentiated Services Codepoint: Default (0)
      .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
    Total Length: 60
    Identification: 0x65a3 (26019)
  Flags: 0x0000
      0... .... .... .... = Reserved bit: Not set
      .0.. .... .... .... = Don't fragment: Not set
      ..0. .... .... .... = More fragments: Not set
      ...0 0000 0000 0000 = Fragment offset: 0
  Time to live: 3
    [Expert Info (Note/Sequence): "Time To Live" only 3]
        ["Time To Live" only 3]
        [Severity level: Note]
        [Group: Sequence]
    Protocol: ICMP (1)
    Header checksum: 0xc347 [validation disabled]
    [Header checksum status: Unverified]
    Source: 192.168.2.182
    Destination: 193.136.9.240
```

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 (Heather length: 20 bytes). O cálculo do tamanho do payload faz-se da seguinte forma: Total length - Header length. Sabemos que Total length = 60 bytes e que, como referido em cima, Heather length = 20 bytes. Sendo assim, o tamanho do payload é 60-20 = 40 bytes. Tal como podemos verificar a partir da imagem da alínea anterior

d) O datagrama IP foi fragmentado? Justifique.

Para sabermos se o datagrama IP foi fragmentado ou não temos que olhar para a flag *Fragment offset*. Ora, esta flag encontra-se com valor 0. Para além de concluirmos que o datagrama não foi fragmentado sabemos também que, no caso de existirem mais fragmentos, este é o primeiro de todos.

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 os campos *ID* e *TTL*.

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

O TTL incrementa 1 valor de 3 em 3 pacotes.

No.	Time	Source	Destination	Protocol	Length	Info
30	0.089372615	192.168.2.1	192.168.2.182	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
31	0.089391951	192.168.2.1	192.168.2.182	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
32	0.089462633	192.168.2.1	192.168.2.182	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
14	0.088971485	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=1/256, ttl=1 (no response found!)
15	0.088977647	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=2/512, ttl=1 (no response found!)
16	0.088980791	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=3/768, ttl=1 (no response found!)
17	0.088984292	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=4/1024, ttl=2 (no response found!)
18	0.088987743	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=5/1280, ttl=2 (no response found!)
19	0.088990555	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=6/1536, ttl=2 (no response found!)
20	0.088994908	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=7/1792, ttl=3 (reply in 36)
21	0.088999160	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=8/2048, ttl=3 (reply in 38)
22	0.089002694	192.168.2.182	193.136.9.240	ICMP	74	Echo (ping) request id=0x1ea4, seq=9/2304, ttl=3 (reply in 39)

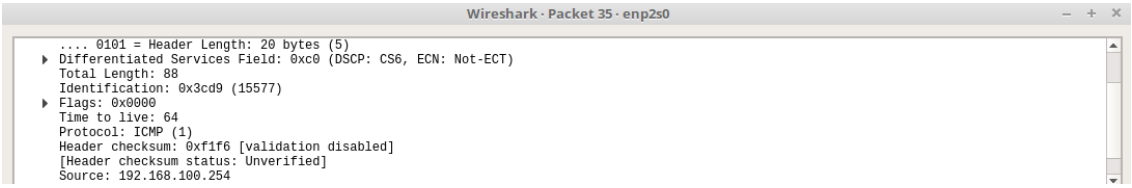
O campo de Identificação incrementa de 1 em 1.

```
▸ Frame 20: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
▸ Ethernet II, Src: AsustekC_04:54:36 (2c:56:dc:04:54:36), Dst: Vmware_5e:69:ad (00:0c:29:5e:69:ad)
▾ Internet Protocol Version 4, Src: 192.168.2.182, Dst: 193.136.9.240
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▾ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
      0000 00.. = Differentiated Services Codepoint: Default (0)
      .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
    Total Length: 60
    Identification: 0x65a3 (26019)
```

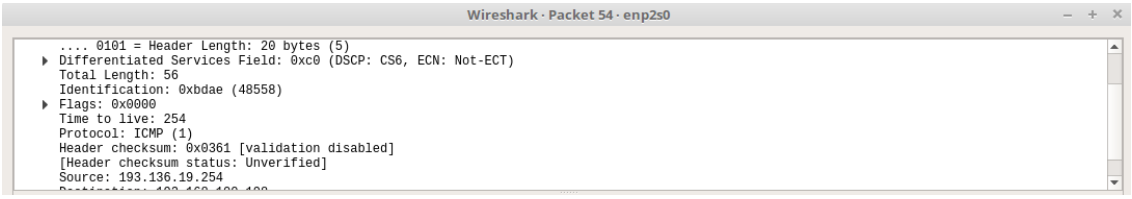
```
▸ Frame 21: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
▸ Ethernet II, Src: AsustekC_04:54:36 (2c:56:dc:04:54:36), Dst: Vmware_5e:69:ad (00:0c:29:5e:69:ad)
▾ Internet Protocol Version 4, Src: 192.168.2.182, Dst: 193.136.9.240
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 60
    Identification: 0x65a4 (26020)
```

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 permanece constante para todas as mensagens de resposta *ICMP TTL exceeded* enviados ao seu host. Como verificamos nas figuras seguintes, temos valores de TTL = 254 e TTL = 64.



Sempre que um pacote dá um "salto", o TTL decrementa. Apesar não estar representado, obtivemos um valor de TTL = 255 e, representado na figura abaixo, encontra-se um valor de TTL = 254, exemplificando assim esta diminuição de valor.



Quando são emitidas mensagens de erro, estas vêm de *Routers* que se encontram mais longe. Assim, para regressar ao *Router* inicial passam, logicamente, por outros *Routers*, levando a que o TTL decremente em cada "salto", uma vez que, quando um pacote passa por um *Router*, o TTL é decrementado uma unidade, como inicialmente explicado.

1.3 Pergunta 3.

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

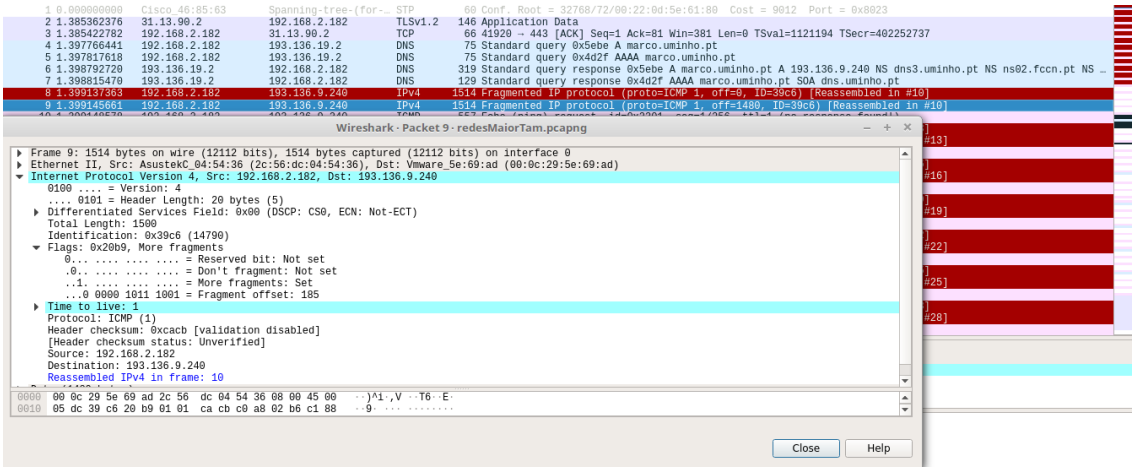
Houve necessidade de fragmentar o pacote inicial (dividir em fragmentos mais pequenos) para que estes conseguissem passar por um link com menor MTU, uma vez que este tem 1500 bytes de tamanho e o pacote que se pretende enviar tem 3503 bytes.

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

A partir do cabeçalho podemos observar que o datagrama foi fragmentado, uma vez que a flag *More Fragments* tem valor 1. Neste mesmo cabeçalho, sabemos que a flag *Fragment offset* tem valor 0, o que nos indica que se trata do primeiro fragmento. Para obtermos o tamanho do datagrama IP, subtraimos o *Header Length* ao *Total Length* (1500-20), o que dá como resultado 1480 bytes.

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

Como já foi referido, conseguimos saber se um fragmento é ou não o primeiro olhando para a flag *Fragment offset*. Como ilustrado na imagem seguinte, o valor desta é diferente de 0, logo, este fragmento não é o primeiro. Sabemos ainda que há ou não mais fragmentos olhando para a flag *More fragments*. Esta tem valor 1(como se vê na figura), o que nos leva a concluir que há mais fragmentos para além deste.



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

O pacote que enviamos é de 3503 bytes, no entanto, o nosso *Total Length* é de 1500, o que nos indica que só podem ser enviados 1500 bytes de cada vez. Posto isto, é necessário dividir o nosso pacote inicial (3503 bytes) em 3 fragmentos, de modo a que todo o pacote seja enviado.

O último fragmento é o pacote 10, uma vez que é neste que verificamos que a flag *More Fragments* se encontra a 0 e a flag *Fragment offset* é diferente de 0, ou seja, não há mais fragmentos criados a partir do datagrama original, mas o fragmento referido também não é o primeiro. Fazendo contagem manual, foram precisos 3 fragmentos para fragmentar na totalidade todo o datagrama, como era de esperar. Esta mesma conclusão pode ser reforçada se tivermos em conta o tamanho no segundo datagrama IP original (=185). Apesar de não haver ilustração gráfica, a flag *Fragment offset* do último fragmento possuía o valor 370. Ou seja (0+185+185), indicando, novamente, que foram criados 3 fragmentos.

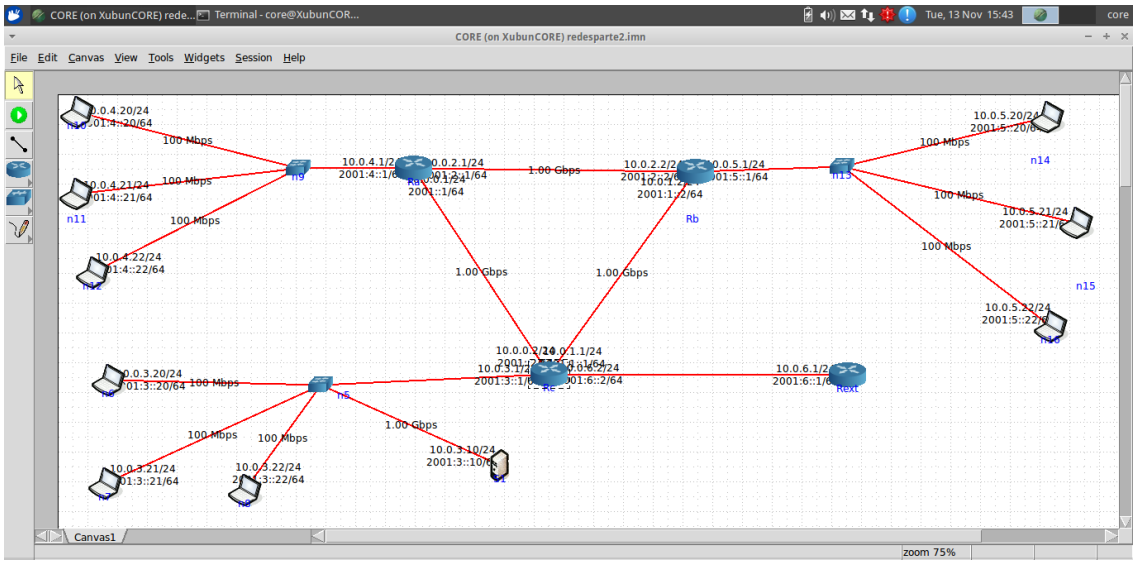
e) Indique, resumindo, os campos que mudam no cabeçalho IP entre os diferentes fragmentos, e explique a forma como essa informação permite reconstruir o datagrama original.

Os campos que mudam são o campo *More Fragments* e o campo *Fragment offset*.O primeiro campo indica se existem ou não mais fragmentos, enquanto que o segundo campo indica em que posição o fragmento se encontra.

2 Questões e Respostas: Parte 2

2.1 Pergunta 1.

a) Indique que endereços IP e máscaras de rede foram atribuídos pelo CORE a cada equipamento.



b) Tratam-se de endereços públicos ou privados? Porquê?

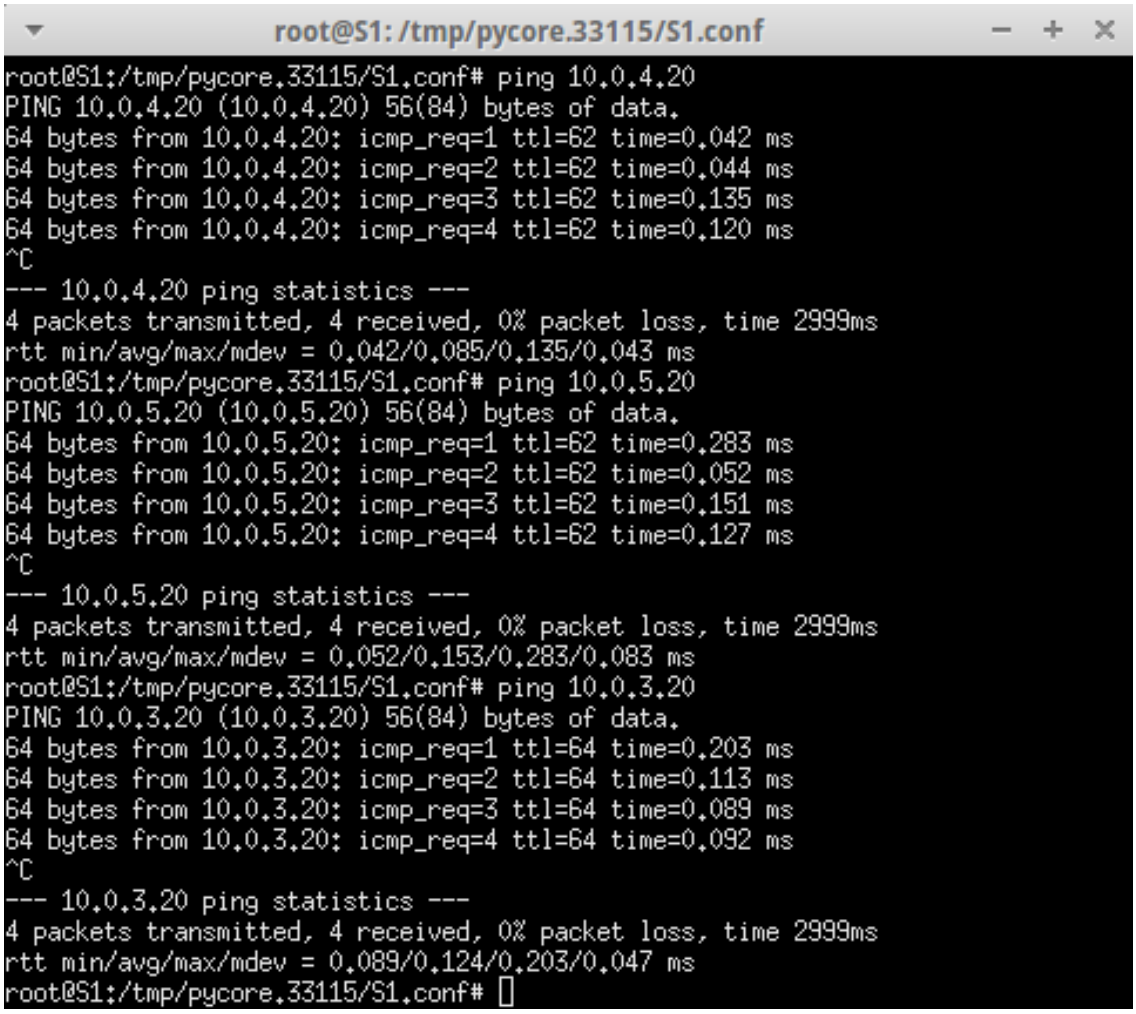
Os endereços tratam-se de endereços privados porque todos se encontram compreendidos entre 10.0.0.0 e 10.255.255.255.

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

Um *switch* é um equipamento activo que funciona, normalmente, na camada 2 do modelo OSI e tem como principal função a interligação de equipamentos de uma rede. Quando um *switch* recebe informação numa determinada porta, transmite essa mesma informação por todas as outras portas, exceto por aquela por onde recebeu essa informação e regista o endereço MAC dos dispositivos que estão ligados a cada porta sua. O *switch* analisa este endereço e envia a informação diretamente para o endereço de destino correspondente. Posto isto, não é atribuído um endereço IP aos *switches* porque não é necessário, dado que estes apenas reencaminham os pacotes para o destino.

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

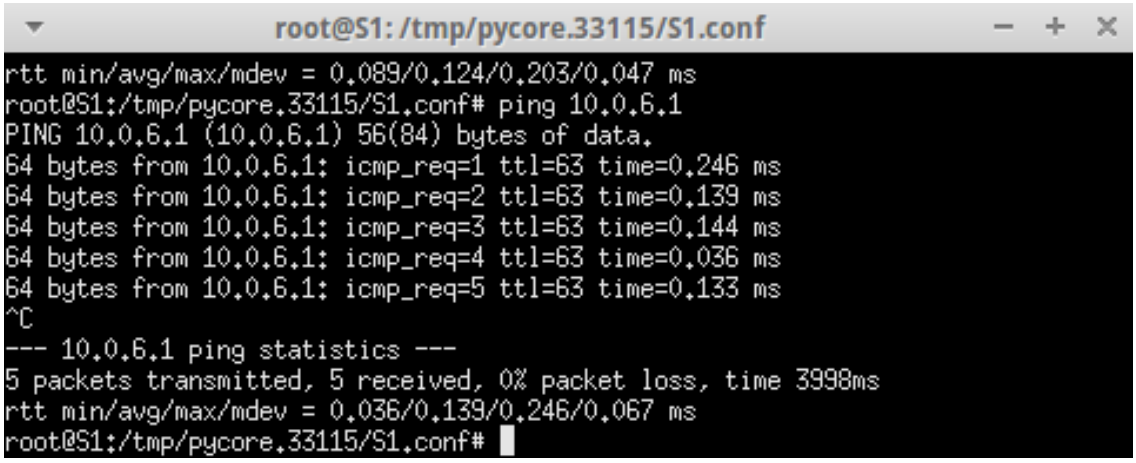
Após a execução do comando ping, obtivemos o seguinte resultado:



Com este resultado, conseguimos afirmar que existe conectividade entre os *laptops* dos vários departamentos e o servidor do departamento. Isto verifica-se uma vez que todos os pacotes enviados a partir do servidor chegam aos *laptops*.

e) Verifique se existe conectividade IP do router de acesso Rext para o servidor S1.

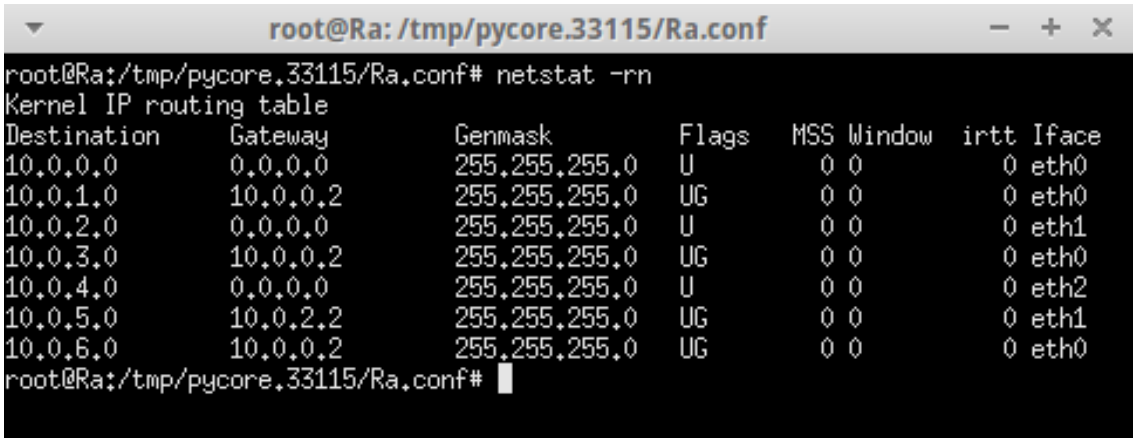
Há conectividade IP do router de acesso Rext para o servidor S1, comprovada pela imagem seguinte:



2.2 Pergunta 2.

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).

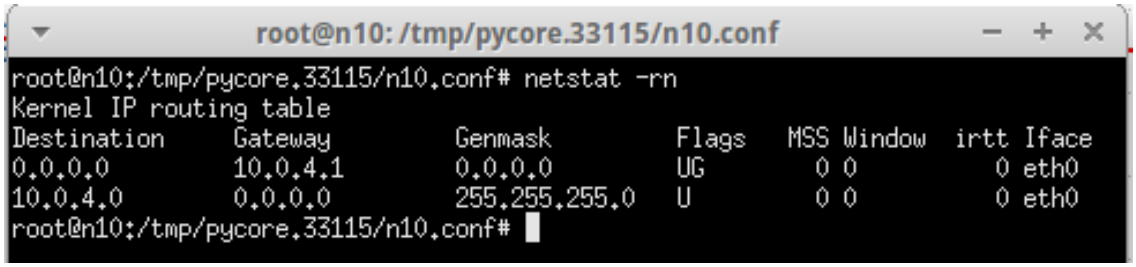
A coluna "Destination" indica a sub-rede de destino, a coluna "Gateway" indica por onde tem que passar o pacote e a "Genmask" refere-se à máscara do pacote que queremos enviar.



A partir da figura verificamos que um pacote que se encontre no *Router A* e tenha como destino uma máquina da sub-rede 10.0.1.0 tem que passar, obrigatoriamente, pelo *Router* 10.0.0.2.

No entanto, se o destino for uma máquina da sub-rede 10.0.0.0, o pacote segue uma qualquer rota, sem restrições.

Já para o *laptop n10* o pacote só pode seguir 2 rotas:

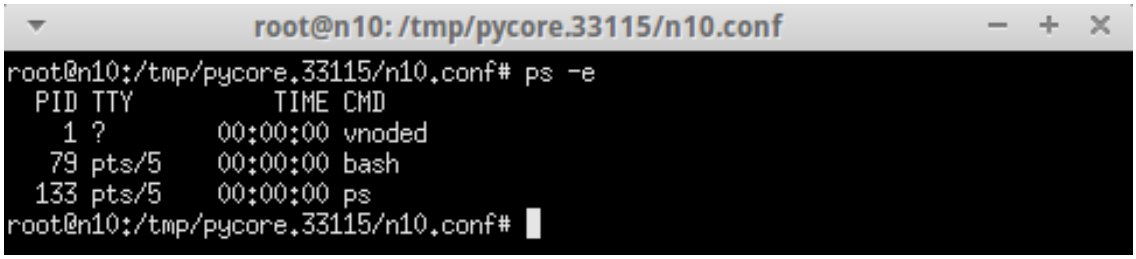


- Uma rota pré-definida. Nesta rota, independentemente do endereço de destino, o pacote tem que passar pelo *Router* cujo endereço é 10.0.4.1(*Router A*).
- Uma rota especifica. Nesta rota, não é obrigatório passar por nenhum *Router* em específico, mas o destino é um endereço da sub-rede do departamento em que esse *laptop* se encontra.

b) Diga, justificando, se está a ser usado encaminhamento estático ou dinâmico (sugestão: analise que processos estão a correr em cada sistema).

Para sabermos se está a ser usado encaminhamento estático ou dinâmico executamos o comando *ps -e*, que analisa que processos estão a correr em cada sistema. Com a execução deste comando verficamos que está em execução o protocolo OSPF (*Open Shortest Path First*). Posto isto, sabemos no *Router A* está a ser usado encaminho dinâmico, uma vez que este protocolo permite que o pacote faça outros caminhos, quando não consegue realizar o caminho pré-definido.

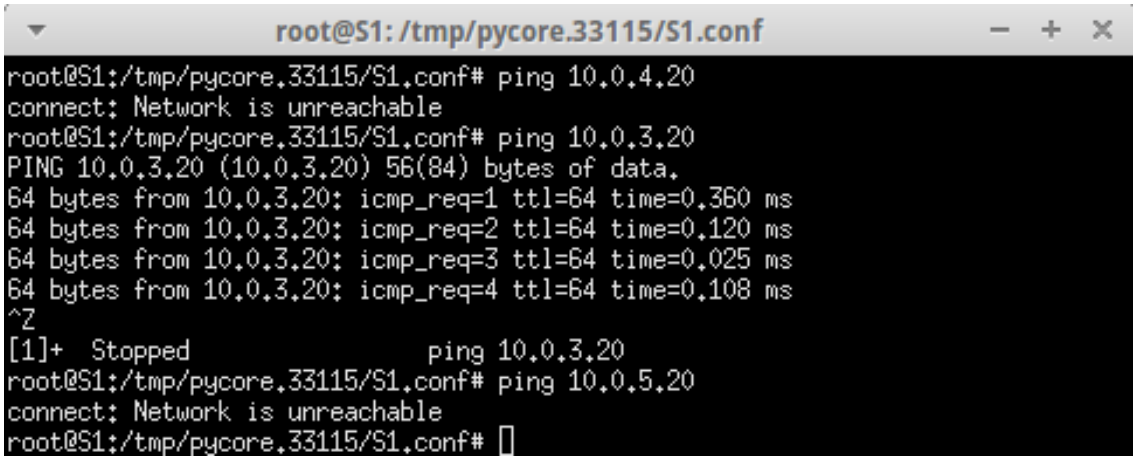
Já no caso do *laptop n10* o encaminhamento é estático:



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 C. Use o comando route delete para o efeito. Que implicações tem esta medida para os utilizadores da empresa que acedem ao servidor. Justifique.

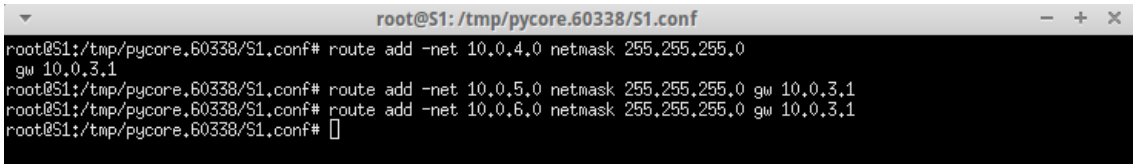
Após execução do comando *route delete default* foi feita a experiência de fazer *ping* para o Departamento B e conseguimos obter conectividade, uma vez que o *laptop* em questão se encontra

neste mesmo departamento. Já na tentativa de fazer ping para o Departamento A e C, não foi possível, tal como era esperado, uma vez que a conexão do *laptop* referido não se encontra em nenhum destes departamentos.



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.

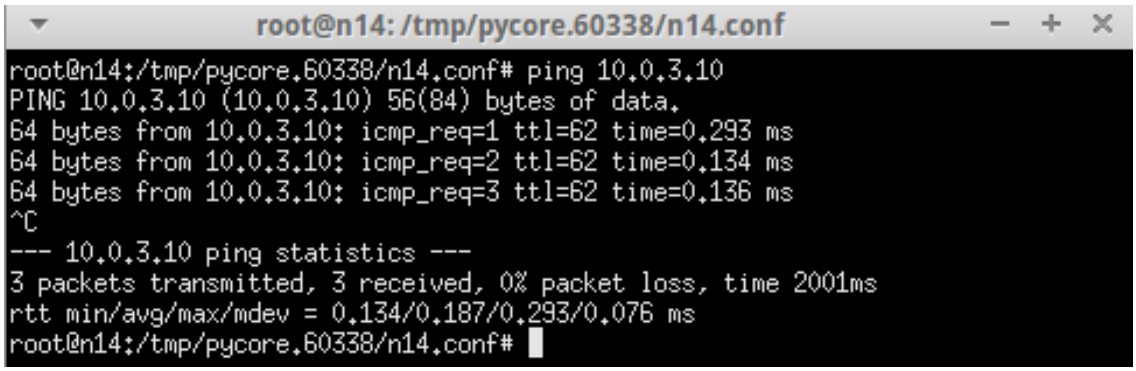
Como podemos verificar na imagem seguinte, utilizamos os seguintes comandos:

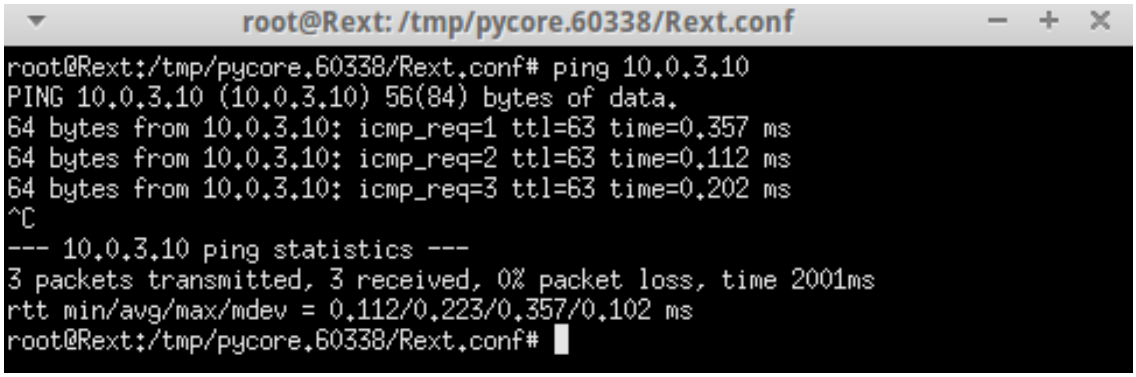


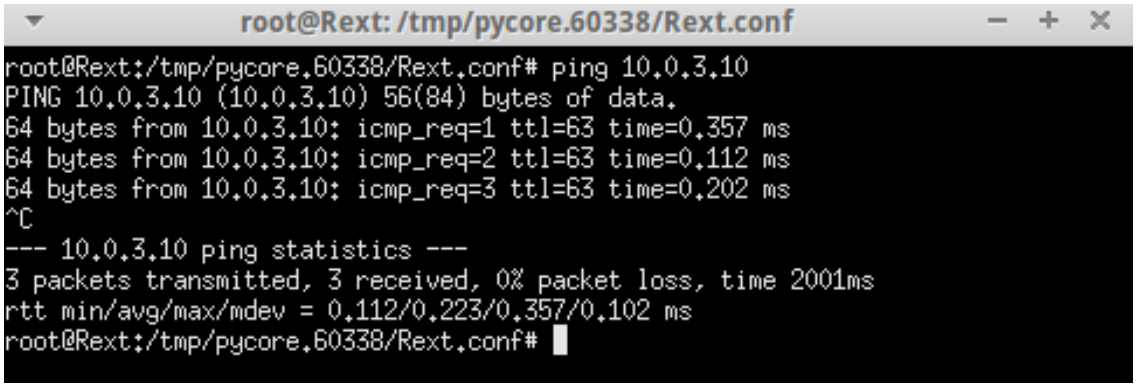
Estes comandos restauram a conectividade entre o servidor e os departamentos A e B, bem como a conectividade com o *Rext*.

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.

Pelas imagens seguintes podemos afirmar que os *laptops* do Departamento A e do departamento B, assim como o *Rext*, conseguem transmitir e receber pacotes com o servidor S1, logo podemos concluir que a ligação com o departamento C foi restabelecida.


root@n14: /tmp/pycore.60338/n14.conf
root@n14:/tmp/pycore.60338/n14.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_req=1 ttl=62 time=0.293 ms
64 bytes from 10.0.3.10: icmp_req=2 ttl=62 time=0.134 ms
64 bytes from 10.0.3.10: icmp_req=3 ttl=62 time=0.136 ms
^C
--- 10.0.3.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 0.134/0.187/0.293/0.076 ms
root@n14:/tmp/pycore.60338/n14.conf#


root@Rext: /tmp/pycore.60338/Rext.conf
root@Rext:/tmp/pycore.60338/Rext.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_req=1 ttl=63 time=0.357 ms
64 bytes from 10.0.3.10: icmp_req=2 ttl=63 time=0.112 ms
64 bytes from 10.0.3.10: icmp_req=3 ttl=63 time=0.202 ms
^C
--- 10.0.3.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 0.112/0.223/0.357/0.102 ms
root@Rext:/tmp/pycore.60338/Rext.conf#


root@Rext: /tmp/pycore.60338/Rext.conf
root@Rext:/tmp/pycore.60338/Rext.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_req=1 ttl=63 time=0.357 ms
64 bytes from 10.0.3.10: icmp_req=2 ttl=63 time=0.112 ms
64 bytes from 10.0.3.10: icmp_req=3 ttl=63 time=0.202 ms
^C
--- 10.0.3.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 0.112/0.223/0.357/0.102 ms
root@Rext:/tmp/pycore.60338/Rext.conf#

2.3 Pergunta 3.

1) Considere que dispõe apenas do endereço de rede IP 172.XX.48.0/20, 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. Deve justificar as opções usadas.

O nosso endereço de rede IP é 172.03.48.0/20. Necessitamos de criar 3 sub-redes (1 para cada departamento). Para criarmos sub-redes utilizamos a expressão matemática: 2^n-2. Posto isto, vamos precisar de 3 bits para satisfazer as condições necessárias (2^3 -2 = 6 sub-redes), uma vez que 2 bits não eram suficientes (2^2 -2) = 2 sub-redes).

A nossa máscara de rede passa a ser /23, devido aos 3 bits que serão utilizados para subnetting.

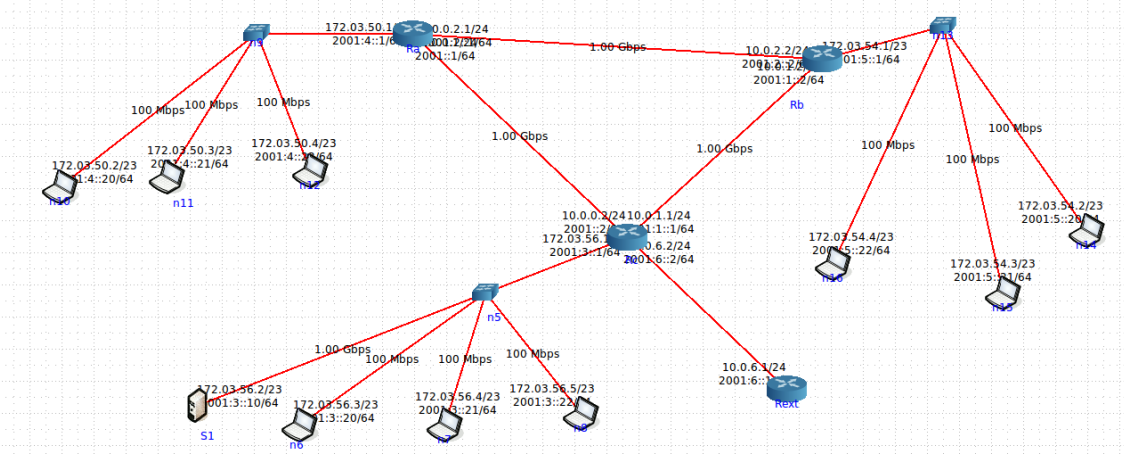
000	Reservado	
001	172.03.50.0/23	Departamento A
010	172.03.52.0/23	Livre
011	172.03.54.0/23	Departamento B
100	172.03.56.0/23	Livre
101	172.03.58.0/23	Departamento C
110	172.03.60.0/23	Livre
111	Reservado (broadcast)	

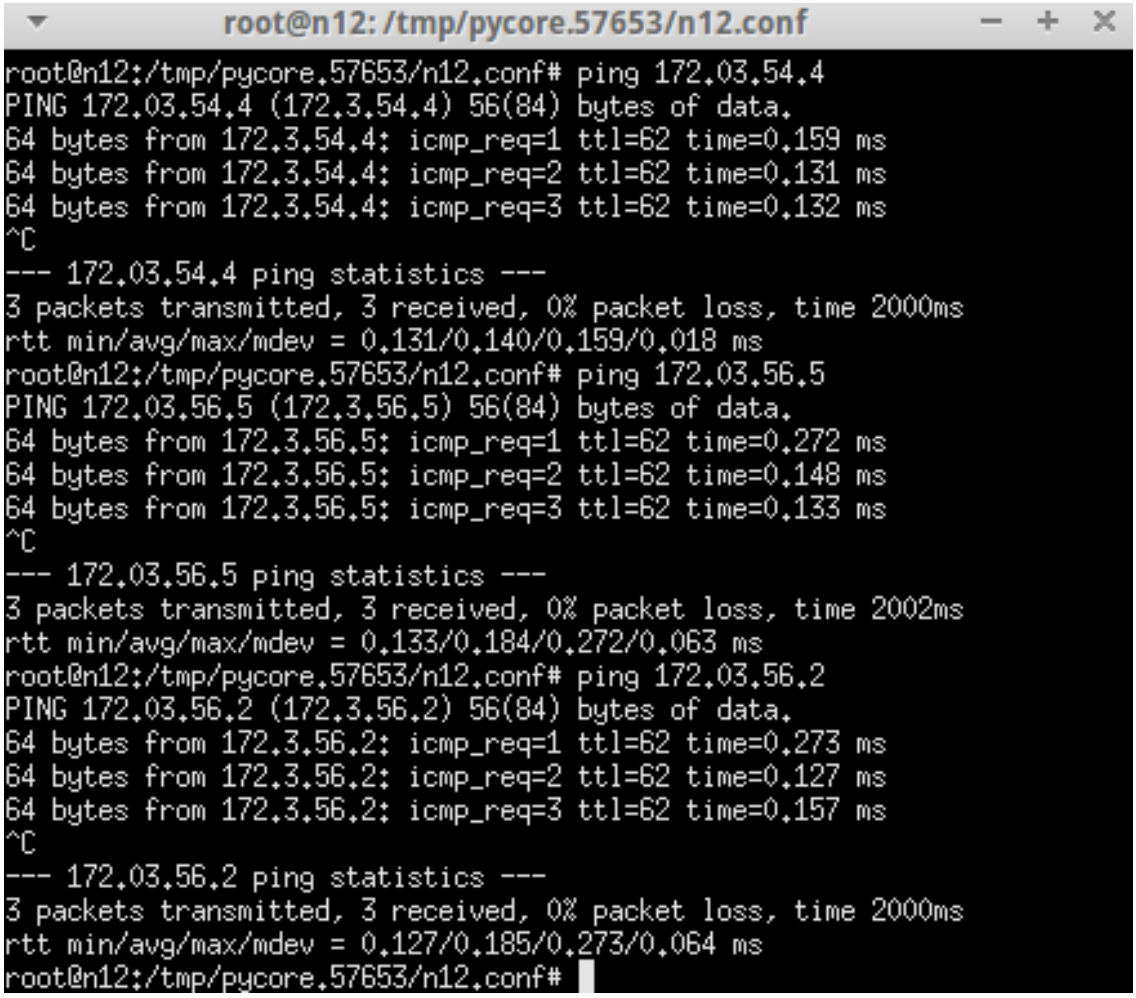
2) Qual a máscara de rede que usou (em formato decimal)? Quantos hosts IP pode interligar em cada departamento? Justifique.

Tal como justificado na alínea anterior, a máscara de rede passou de /20 para /23, pois necessitamos de reservar 3 bits para subnetting. Sendo assim, a máscara de rede que usamos, em formato decimal, foi: 255.255.254.0. Sabemos então que não podemos mexer nos primeiros 23 bits. Restam-nos 9 bits que podemos alterar (32-23 = 9 bits). Sendo assim, podemos interligar 2^9 -2 = 510 *hosts*. Cada rede tem 2 endereços de reserva: um para *broadcast* e outro que corresponde ao endereço IP de rede.

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

Inicialmente modificamos os endereços IP das máquinas conforme a tabela da alínea anterior.





Com isto, vimos que existe conectividade entre as várias redes locais.

3 Conclusão

Este relatório foi dividido em 2 partes. Na primeira parte, foi feita uma pesquisa mais aprofundada ao protocolo IP(v4), utilizando a topologia CORE, observando datagramas IP, tráfego ICMP, entre outros. Também a fragmentação foi uma parte bastante estudada nesta fase, ajudando-nos a entender como se processa o envio de pacotes. Já na segunda parte, como referido no enunciado, estudamos o endereçamento e encaminhamento IP. Assim, como na primeira parte, visualizamos um datagrama. Desta vez com 3 departamentos e as suas respetivas máquinas. Nesta fase procuramos aprofundar o nosso conhecimento sobre manipulação de endereços IP.