Trabalho Prático Nº 2

Protocolo IPv4 - Datagramas IP e Fragmentação

Realizado por: Ricardo Miguel Queirós de Jesus (a100066)
Jorge Emanuel Matos Teixeira (a100838)
Rui Pedro Fernandes Madeira Pinto (a100659)

Questões

PARTE 1

1)

a) Registe e analise o tráfego ICMP enviado pelo sistema Lost e o tráfego ICMP recebido como resposta. Explique os resultados obtidos tendo em conta o princípio de funcionamento do traceroute.

i)

```
root@Lost:/tmp/pycore.35101/Lost.conf# traceroute -I 10.0.5.10
traceroute to 10.0.5.10 (10.0.5.10), 30 hops max, 60 byte packets
 1  10.0.0.1 (10.0.0.1)  0.109 ms  0.069 ms  0.065 ms
 2  10.0.1.2 (10.0.1.2)  30.886 ms  30.878 ms  30.875 ms
 3  10.0.3.2 (10.0.3.2)  60.980 ms  60.978 ms  60.976 ms
 4  10.0.5.10 (10.0.5.10)  62.145 ms  62.143 ms  62.141 ms
root@Lost:/tmp/pycore.35101/Lost.conf#
```

R:No resultado obtido é possível verificar 4 registos de tempos diferentes,que correspondem aos tempos de ida-e-volta dos pacotes à medida que o TTL correspondente aumenta.

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

ii) **R:** O valor inicial mínimo do campo TTL para alcançar o servidor “Found” deveria ser 4, pois tem 3 routers intermediários. É possível confirmar este valor através do Wireshark em que obtemos resposta a quando do campo TTL com valor 4.

27	30.780221512	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request id=0x002c, seq=4/1024, ttl=2 (no respons…
28	30.780231094	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request id=0x002c, seq=5/1280, ttl=2 (no respons…
29	30.780234839	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request id=0x002c, seq=6/1536, ttl=2 (no respons…
30	30.780238780	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request id=0x002c, seq=7/1792, ttl=3 (no respons…
31	30.780242584	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request id=0x002c, seq=8/2048, ttl=3 (no respons…
32	30.780246092	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request id=0x002c, seq=9/2304, ttl=3 (no respons…
33	30.780249981	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request id=0x002c, seq=10/2560, ttl=4 (reply in …
34	30.780253527	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request id=0x002c, seq=11/2816, ttl=4 (reply in …
35	30.780257022	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request id=0x002c, seq=12/3072, ttl=4 (reply in …
36	30.780260939	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request id=0x002c, seq=13/3328, ttl=5 (reply in …
37	30.780264479	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request id=0x002c, seq=14/3584, ttl=5 (reply in …
38	30.780267985	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request id=0x002c, seq=15/3840, ttl=5 (reply in …
39	30.780272747	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request id=0x002c, seq=16/4096, ttl=6 (reply in …
40	30.780622944	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request id=0x002c, seq=17/4352, ttl=6 (reply in …
41	30.780629262	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request id=0x002c, seq=18/4608, ttl=6 (reply in …
42	30.780633391	10.0.0.20	10.0.5.10	ICMP	74 Echo (ping) request id=0x002c, seq=19/4864, ttl=7 (reply in …

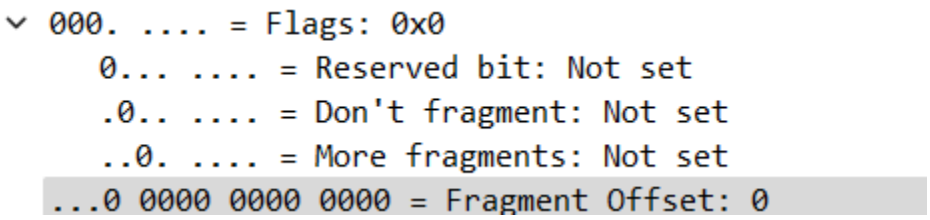
b) **Calcule o valor médio do tempo de ida-e-volta (RTT - Round-Trip Time) obtido no acesso ao servidor. Por modo a obter uma média mais confiável, poderá alterar o número pacotes de prova com a opção -q**
   i) **R:** (62,145 + 62,143 + 62,141) / 3 = 62,143 ms
c) **O valor médio do atraso num sentido (One-Way Delay) poderia ser calculado com precisão dividindo o RTT por dois? O que torna difícil o cálculo desta métrica numa rede real?**
   i) **R:** Neste caso específico poderia-se calcular o valor médio do atraso num sentido dividindo o RTT por dois, porque sabemos que as latências são iguais. Contudo, numa rede real, o facto do valor da latência ser variável e desconhecida, impede o cálculo preciso deste atraso.

2)

a) **Qual é o endereço IP da interface ativa do seu computador?**
   i) 172.26.16.19
b) **Qual é o valor do campo protocol? O que permite identificar?**
   i) ICMP e identifica o protocolo utilizado

c) **Quantos bytes tem o cabeçalho IPv4? Quantos bytes tem o campo de dados (payload) do datagrama? Como se calcula o tamanho do payload?**
   i) O cabeçalho IPv4 tem 20 bytes. O campo de dados do datagrama tem 36 bytes e é calculado a partir da subtração do número de bytes do cabeçalho ao número total de bytes utilizado (56 - 20 = 36).

d) **O datagrama IP foi fragmentado? Justifique.**
   i) O datagrama não foi fragmentado, assim como podemos ver na print a seguir: a flag “More fragments” está apresentada como “not set” e “Fragment Offset” está a 0 o que indica que não tem mais fragmentos e o offset do pacote atual é 0 (ou seja, é único).



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.**

R:De pacote para pacote, varia o campo **“Identification”, “Time to Live” e “Header Checksum”**

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

R: Verificamos que existe um padrão de crescimento proporcional entre o IP e o TTL, excluindo o primeiro IP da “sequência” que apresenta um TTL muito maior que os restantes, que por ser o primeiro tem um tamanho ttl pré determinado (neste caso, 255).

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

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

R:O valor do campo TTL apresentado é de 255 e não permanece constante para todas as mensagens de resposta. Este valor diminui porque

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

R:O valor do TTL é relativamente alto porque, quando o pacote chega ao destino, o router envia uma resposta (um pacote de resposta) e, para garantir que este chega à origem, o router de destino utiliza um TTL grande.

h) **Sabendo que o ICMP é um protocolo pertencente ao nível de rede, discuta se a informação contida no cabeçalho ICMP poderia ser incluída no cabeçalho IPv4? Quais seriam as vantagens/desvantagens resultantes dessa hipotética inclusão?**

R: A informação do cabeçalho ICMP poderia, hipoteticamente, ser incluída no cabeçalho do IPv4, desde que o seu limite de bytes aumentasse, claro que implicaria alterar todos os sistemas atuais para passarem a receber cabeçalhos com o novo número de bytes. No caso hipotético em que tal seria feito, seria vantajoso pois iria diminuir a complexidade, já que a informação do ICMP já não vinha encapsulada, também iria aumentar a velocidade de processamento do pacote, como a informação não vem encapsulada o acesso a ela é direto, aumentando significativamente a velocidade de processamento da mesma. Por outro lado, esta junção iria fazer com que o tamanho dos cabeçalhos aumentassem significativamente, o que iria provocar pacotes muito mais pesados que vão precisar de mais fragmentação do que primeiro. Outra desvantagem,seria a diminuição da flexibilidade do ICMP, isto porque, as informações do cabeçalho ICMP são usadas para interpretar e responder a uma variedade de funções de rede, ao juntar o ICMP com o IPv4, iria fazer com que

se perdesse parte da flexibilidade, complicando como se lida com certas mensagens ICMP.

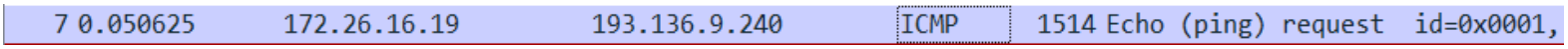
3)

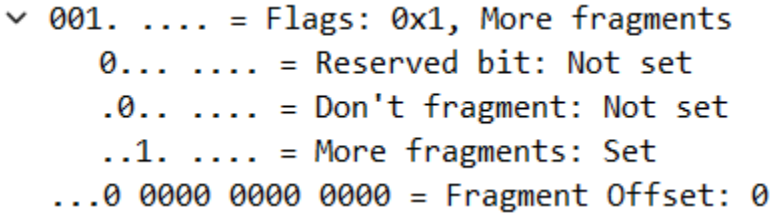
a) **Localize a primeira mensagem ICMP. Porque é que houve necessidade de fragmentar o pacote inicial?**

R: Houve necessidade de fragmentar o pacote inicial porque era demasiado grande para ser enviado de uma só vez, isto é, o seu comprimento excedia o seu MTU (que no caso é 3510).

b) **Imprima o primeiro fragmento do datagrama IP original. 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?**

R: Os 3 bits identificativos da flag (001) indicam através do terceiro bit (1) que existem mais fragmentos. É possível também ver que se trata do primeiro fragmento porque o campo “Fragment Offset” se encontra a 0 sendo o tamanho deste datagrama ()





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

```
Internet Protocol Version 4, Src: 172.26.16.19, 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: 0x08aa (2218)
  v 001. .... = Flags: 0x1, More fragments
      0... .... = Reserved bit: Not set
      .0.. .... = Don't fragment: Not set
      ..1. .... = More fragments: Set
    ...0 0000 1011 1001 = Fragment Offset: 1480
```

R: Sabemos que não se trata do 1º fragmento pois o valor do “Fragment Offset” é maior que zero (neste caso, 1480 bytes já foram incluídos noutro pacote) . Sabemos que existem mais fragmentos para além deste, pois o bit correspondente a “More fragments” está definido com 1.

d) **Estime teoricamente o número de fragmentos gerados a partir do datagrama IP original e o número de bytes transportados no último fragmento desse datagrama. Compare os dois valores estimados com os obtidos através do wireshark.**

Estimativa teórica:

Nº fragmentos = 3510 / (1500 - 20) ≈ 3

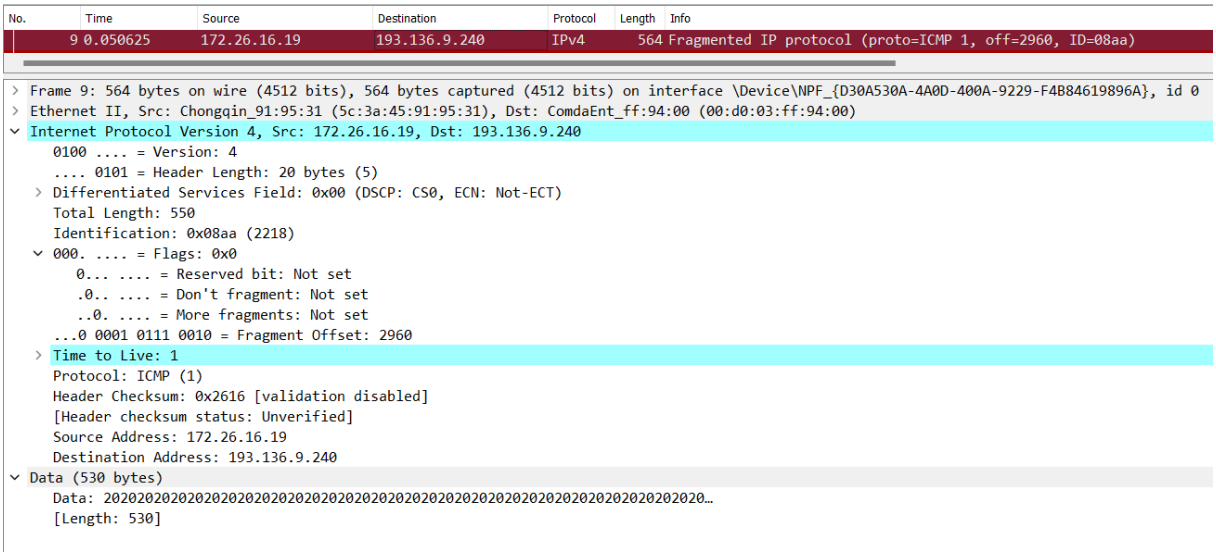
Nº Bytes no útimo frag. = 3510 - 2* 1480 = 550

```
7 0.050625   172.26.16.19   193.136.9.240   ICMP   1514 Echo (ping) request  id=0x0001, seq=291/8961, ttl=1 (no response found!)
8 0.050625   172.26.16.19   193.136.9.240   IPv4   1514 Fragmented IP protocol (proto=ICMP 1, off=1480, ID=08aa)
9 0.050625   172.26.16.19   193.136.9.240   IPv4    564 Fragmented IP protocol (proto=ICMP 1, off=2960, ID=08aa)
```

R: Como é possível analisar, houve a criação de 3 fragmentos e o fragmento final possuí um “Total length” de 550, tal como foi previsto.

e) **Como se deteta o último fragmento correspondente ao datagrama original? Estabeleça um filtro no Wireshark que permita listar o último fragmento do primeiro datagrama IP segmentado.**

R: Sabemos que se trata do último fragmento se o bit do “More Fragments” estiver definido a 0 e o seu campo de “Off Set” estiver com um valor diferente de 0. Assim temos o filtro, e um exemplo de pacotes que respeitam as suas condições:



f) Identifique o equipamento onde o datagrama IP original é reconstruído a partir dos fragmentos. A reconstrução poderia ter ocorrido noutro equipamento diferente do identificado? Porquê?

R: O datagrama IP original é reconstruído no equipamento de destino com ip:193.136.9.240. Efetivamente, a reconstrução poderia ter ocorrido noutro equipamento que não este. Contudo, existem desvantagens associadas à reconstrução ante a chegada ao destino, como por exemplo, se o datagrama tivesse de passar por outro host com um MTU diferente (mais pequeno, por exemplo) teria de se fragmentar novamente, indo posteriormente necessitar de uma nova reconstrução.Para além disso, os fragmentos de um datagrama não seguem necessariamente o mesmo caminho para alcançar um destino pelo que nesse caso a reconstrução seria impossível de acontecer num equipamento que não o de chegada.

g) 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.

R: De fragmento em fragmento os campos de cabeçalho que vão alterando são o “Fragment Offset”, que indica quanta informação já foi enviada antes deste fragmento (sendo preciso multiplicar por 8) e o “More fragments”, que muda caso o fragmento seja o último de todos.

h) Por que razão apenas o primeiro fragmento de cada pacote é identificado como sendo um pacote ICMP?

R: Apenas o primeiro fragmento de cada pacote é identificado como ICMP porque este tem uma pequena dimensão fazendo com que no ocorrer da fragmentação ele consiga “caber” no primeiro fragmento.

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

R: O valor do datagrama é comparado com o MTU (Maximum Transmission Unit) que é o tamanho máximo que um pacote pode ter. Um MTU maior iria fazer com que fosse preciso enviar menos pacotes, no entanto iria provocar uma maior fragmentação dos mesmos. Por outro lado, diminuindo o valor de MTU, a perda de pacotes e fragmentação seria menos propícia a acontecer.

j) **Sabendo que no comando ping a opção -f (Windows), -M do (Linux) ou –D (Mac) ativa a flag "Don't Fragment" (DF) no cabeçalho do IPv4, usando ping SIZE marco.uminho.pt, (opção pkt_size = –l (Windows) ou –s (Linux, Mac), determine o valor máximo de SIZE sem que ocorra fragmentação do pacote? Justifique o valor obtido.**

R: Sendo de conhecimento prévio que o MTU era de 1500 bytes, contudo o cabeçalho IP ocupa 20 bytes e o campo protocolo ocupa outros 8 bytes. Assim sendo ficamos com um tamanho máximo de 1472, tal como provado:

```
PS C:\Users\matos> ping -f -l 1472 marco.uminho.pt

Pinging marco.uminho.pt [193.136.9.240] with 1472 bytes of data:
Reply from 193.136.9.240: bytes=1472 time=11ms TTL=61
Reply from 193.136.9.240: bytes=1472 time=14ms TTL=61
Reply from 193.136.9.240: bytes=1472 time=14ms TTL=61
Reply from 193.136.9.240: bytes=1472 time=11ms TTL=61

Ping statistics for 193.136.9.240:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 11ms, Maximum = 14ms, Average = 12ms
PS C:\Users\matos> ping -f -l 1473 marco.uminho.pt

Pinging marco.uminho.pt [193.136.9.240] with 1473 bytes of data:
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.

Ping statistics for 193.136.9.240:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

PARTE 2

1.

a) **Averigue, através do comando ping, que AfonsoHenriques tem efetivamente conectividade com o servidor Financas e com os servidores da CDN.**

```
<core.33609/AfonsoHenriques.conf# ping 192.168.0.250
PING 192.168.0.250 (192.168.0.250) 56(84) bytes of data.
64 bytes from 192.168.0.250: icmp_seq=1 ttl=61 time=0.063 ms
64 bytes from 192.168.0.250: icmp_seq=2 ttl=61 time=0.121 ms
64 bytes from 192.168.0.250: icmp_seq=3 ttl=61 time=0.111 ms
64 bytes from 192.168.0.250: icmp_seq=4 ttl=61 time=0.076 ms
```

-> ping as Finanças com sucesso

```
<ycore.33609/AfonsoHenriques.conf# ping 192.168.0.202
PING 192.168.0.202 (192.168.0.202) 56(84) bytes of data.
64 bytes from 192.168.0.202: icmp_seq=1 ttl=55 time=0.232 ms
64 bytes from 192.168.0.202: icmp_seq=2 ttl=55 time=0.170 ms
^C
--- 192.168.0.202 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1027ms
rtt min/avg/max/mdev = 0.170/0.201/0.232/0.031 ms
<core.33609/AfonsoHenriques.conf# ping 192.168.0.203
PING 192.168.0.203 (192.168.0.203) 56(84) bytes of data.
64 bytes from 192.168.0.203: icmp_seq=1 ttl=55 time=0.144 ms
^C
--- 192.168.0.203 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.144/0.144/0.144/0.000 ms
<core.33609/AfonsoHenriques.conf# ping 192.168.0.204
PING 192.168.0.204 (192.168.0.204) 56(84) bytes of data.
64 bytes from 192.168.0.204: icmp_seq=1 ttl=55 time=0.143 ms
^C
--- 192.168.0.204 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.143/0.143/0.143/0.000 ms
<core.33609/AfonsoHenriques.conf# ping 192.168.0.210
PING 192.168.0.210 (192.168.0.210) 56(84) bytes of data.
64 bytes from 192.168.0.210: icmp_seq=1 ttl=55 time=0.142 ms
64 bytes from 192.168.0.210: icmp_seq=2 ttl=55 time=0.145 ms
^C
--- 192.168.0.210 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1022ms
rtt min/avg/max/mdev = 0.142/0.143/0.145/0.001 ms
<core.33609/AfonsoHenriques.conf# ping 192.168.0.218
PING 192.168.0.218 (192.168.0.218) 56(84) bytes of data.
64 bytes from 192.168.0.218: icmp_seq=1 ttl=55 time=0.124 ms
64 bytes from 192.168.0.218: icmp_seq=2 ttl=55 time=0.143 ms
^C
--- 192.168.0.218 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1016ms
rtt min/avg/max/mdev = 0.124/0.133/0.143/0.009 ms
```

-> ping servidores CDN

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

```
root@AfonsoHenriques:/tmp/pycore.33609/AfonsoHenriques.conf# netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         192.168.0.225   0.0.0.0         UG        0 0          0 eth0
192.168.0.224   0.0.0.0         255.255.255.248 U         0 0          0 eth0
```

->Tabela de encaminhamento Afonso

```
root@Teresa:/tmp/pycore.33609/Teresa.conf# netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         192.168.0.193   0.0.0.0         UG        0 0          0 eth0
192.168.0.192   0.0.0.0         255.255.255.248 U         0 0          0 eth0
```

->Tabela de encaminhamento Teresa

R: Não existe nenhum problema nas suas entradas.Em ambas as tabelas de encaminhamento vemos duas entradas, a primeira é denominada de rota default, e serve para,caso o tráfego não corresponda a nenhuma outra entrada na tabela(em ambos os casos só há mais uma), vai ser direcionado para o gateway desta entrada. A segunda entrada é o caminho para a rede local.

c)

```
root@AfonsoHenriques:/tmp/pycore.35819/AfonsoHenriques.conf# traceroute 192.16>
traceroute to 192.168.0.194 (192.168.0.194), 30 hops max, 60 byte packets
 1  192.168.0.225 (192.168.0.225)  0.036 ms  0.004 ms  0.004 ms
 2  172.16.143.1 (172.16.143.1)  0.016 ms  0.007 ms  0.006 ms
 3  10.0.0.29 (10.0.0.29)  0.017 ms !N  0.009 ms !N *
```

- > Afonso nao chega a Teresa por causa do “10.0.0.29”(n5)

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
10.0.0.0        10.0.0.25       255.255.255.252 UG        0 0          0 eth1
10.0.0.4        10.0.0.25       255.255.255.252 UG        0 0          0 eth1
10.0.0.8        10.0.0.25       255.255.255.252 UG        0 0          0 eth1
10.0.0.12       10.0.0.25       255.255.255.252 UG        0 0          0 eth1
10.0.0.16       10.0.0.25       255.255.255.252 UG        0 0          0 eth1
10.0.0.20       10.0.0.25       255.255.255.252 UG        0 0          0 eth1
10.0.0.24       0.0.0.0         255.255.255.252 U         0 0          0 eth1
10.0.0.28       0.0.0.0         255.255.255.252 U         0 0          0 eth0
172.0.0.0       10.0.0.30       255.0.0.0       UG        0 0          0 eth0
172.16.142.0    10.0.0.25       255.255.255.248 UG        0 0          0 eth1
172.16.143.0    10.0.0.30       255.255.255.252 UG        0 0          0 eth0
172.16.143.0    10.0.0.30       255.255.255.248 UG        0 0          0 eth0
172.16.143.4    10.0.0.30       255.255.255.252 UG        0 0          0 eth0
192.142.0.4     10.0.0.25       255.255.255.252 UG        0 0          0 eth1
192.168.0.200   10.0.0.25       255.255.255.248 UG        0 0          0 eth1
192.168.0.208   10.0.0.25       255.255.255.248 UG        0 0          0 eth1
192.168.0.216   10.0.0.25       255.255.255.248 UG        0 0          0 eth1
192.168.0.224   10.0.0.30       255.255.255.248 UG        0 0          0 eth0
192.168.0.232   10.0.0.30       255.255.255.248 UG        0 0          0 eth0
192.168.0.240   10.0.0.30       255.255.255.248 UG        0 0          0 eth0
192.168.0.248   10.0.0.30       255.255.255.248 UG        0 0          0 eth0
```

-> tabela de encaminhamento de “10.0.0.29”(n5), vemos que não há entrada de ip de Afonso para ip Teresa

```
<5.conf# route add -net 192.168.0.192 gw 10.0.0.25 netmask 255.255.255.248
```

-> adicionando a nova rota na tabela de encaminhamento de n5

```
traceroute to 192.168.0.194 (192.168.0.194), 30 hops max, 60 byte packets
 1  192.168.0.225 (192.168.0.225)  0.041 ms  0.005 ms  0.004 ms
 2  172.16.143.1 (172.16.143.1)  0.014 ms  0.007 ms  0.006 ms
 3  10.0.0.29 (10.0.0.29)  0.018 ms  0.008 ms  0.009 ms
 4  10.0.0.25 (10.0.0.25)  0.033 ms  0.010 ms  0.010 ms
 5  10.0.0.25 (10.0.0.25)  3063.157 ms !H  3063.144 ms !H  3063.134 ms !H
```

->o pacote falhou em “10.0.0.25”(n2)

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
10.0.0.0        10.0.0.13       255.255.255.252 UG        0 0          0 eth1
10.0.0.4        10.0.0.21       255.255.255.252 UG        0 0          0 eth0
10.0.0.8        10.0.0.13       255.255.255.252 UG        0 0          0 eth1
10.0.0.12       0.0.0.0         255.255.255.252 U         0 0          0 eth1
10.0.0.16       10.0.0.13       255.255.255.252 UG        0 0          0 eth1
10.0.0.20       0.0.0.0         255.255.255.252 U         0 0          0 eth0
10.0.0.24       0.0.0.0         255.255.255.252 U         0 0          0 eth2
10.0.0.28       10.0.0.26       255.255.255.252 UG        0 0          0 eth2
172.0.0.0       10.0.0.26       255.0.0.0       UG        0 0          0 eth2
172.16.142.0    10.0.0.13       255.255.255.252 UG        0 0          0 eth1
172.16.142.4    10.0.0.21       255.255.255.252 UG        0 0          0 eth0
172.16.143.0    10.0.0.26       255.255.255.252 UG        0 0          0 eth2
172.16.143.4    10.0.0.26       255.255.255.252 UG        0 0          0 eth2
192.168.0.192   10.0.0.13       255.255.255.248 UG        0 0          0 eth1
192.168.0.194   10.0.0.25       255.255.255.254 UG        0 0          0 eth2
192.168.0.200   10.0.0.21       255.255.255.248 UG        0 0          0 eth0
192.168.0.208   10.0.0.21       255.255.255.248 UG        0 0          0 eth0
192.168.0.216   10.0.0.21       255.255.255.248 UG        0 0          0 eth0
192.168.0.224   10.0.0.26       255.255.255.248 UG        0 0          0 eth2
192.168.0.232   10.0.0.26       255.255.255.248 UG        0 0          0 eth2
192.168.0.240   10.0.0.26       255.255.255.248 UG        0 0          0 eth2
192.168.0.248   10.0.0.26       255.255.255.248 UG        0 0          0 eth2
```

-> na tabela de encaminhamento de n2 tinha uma entrada errada

```
K2.conf# route del -net 192.168.0.194 netmask 255.255.255.254
```

-> eliminamos a entrada errada

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
10.0.0.0        10.0.0.9        255.255.255.252 UG        0 0          0 eth0
10.0.0.4        10.0.0.9        255.255.255.252 UG        0 0          0 eth0
10.0.0.8        0.0.0.0         255.255.255.252 U         0 0          0 eth0
10.0.0.12       0.0.0.0         255.255.255.252 U         0 0          0 eth1
10.0.0.16       10.0.0.9        255.255.255.252 UG        0 0          0 eth0
10.0.0.20       10.0.0.14       255.255.255.252 UG        0 0          0 eth1
10.0.0.24       10.0.0.14       255.255.255.252 UG        0 0          0 eth1
10.0.0.28       10.0.0.14       255.255.255.252 UG        0 0          0 eth1
172.0.0.0       10.0.0.14       255.0.0.0       UG        0 0          0 eth1
172.16.142.0    10.0.0.9        255.255.255.252 UG        0 0          0 eth0
172.16.142.4    10.0.0.9        255.255.255.252 UG        0 0          0 eth0
172.16.143.0    10.0.0.14       255.255.255.252 UG        0 0          0 eth1
172.16.143.4    10.0.0.14       255.255.255.252 UG        0 0          0 eth1
192.168.0.192   10.0.0.14       255.255.255.248 UG        0 0          0 eth1
192.168.0.200   10.0.0.9        255.255.255.248 UG        0 0          0 eth0
192.168.0.208   10.0.0.9        255.255.255.248 UG        0 0          0 eth0
192.168.0.216   10.0.0.9        255.255.255.248 UG        0 0          0 eth0
192.168.0.224   10.0.0.14       255.255.255.248 UG        0 0          0 eth1
192.168.0.232   10.0.0.14       255.255.255.248 UG        0 0          0 eth1
192.168.0.240   10.0.0.14       255.255.255.248 UG        0 0          0 eth1
192.168.0.248   10.0.0.14       255.255.255.248 UG        0 0          0 eth1
```

-> Tabela de encaminhamento de n1 possui uma entrada que provoca um loop

```
<1.conf# route del -net 192.168.0.192 netmask 255.255.255.248
```

->eliminar entrada errada em n1

```
<1.conf# route add -net 192.168.0.192 gw 10.0.0.9 netmask 255.255.255.248
```

->adicionamos a entrada correta

```
<5819/AfonsoHenriques.conf# traceroute 192.168.0.194
traceroute to 192.168.0.194 (192.168.0.194), 30 hops max, 60 byte packets
 1  192.168.0.225 (192.168.0.225)  0.040 ms  0.005 ms  0.004 ms
 2  172.16.143.1 (172.16.143.1)  0.015 ms  0.006 ms  0.006 ms
 3  10.0.0.29 (10.0.0.29)  0.017 ms  0.008 ms  0.009 ms
 4  10.0.0.25 (10.0.0.25)  0.021 ms  0.011 ms  0.010 ms
 5  10.0.0.13 (10.0.0.13)  0.036 ms  0.013 ms  0.013 ms
 6  10.0.0.13 (10.0.0.13)  3075.752 ms !H  3075.096 ms !H  3075.069 ms !H
```

->erro no "10.0.0.13"

```
root@n1:/tmp/pycore.35819/n1.conf# route add -net 192.168.0.192 gw 10.0.0.9 ne>
```

->adicionar a entrada correta

```
<5819/AfonsoHenriques.conf# traceroute 192.168.0.194
traceroute to 192.168.0.194 (192.168.0.194), 30 hops max, 60 byte packets
 1  192.168.0.225 (192.168.0.225)  0.039 ms  0.005 ms  0.004 ms
 2  172.16.143.1 (172.16.143.1)  0.015 ms  0.007 ms  0.006 ms
 3  10.0.0.29 (10.0.0.29)  0.016 ms  0.008 ms  0.008 ms
 4  10.0.0.25 (10.0.0.25)  0.021 ms  0.010 ms  0.011 ms
 5  10.0.0.13 (10.0.0.13)  0.023 ms  0.012 ms  0.012 ms
 6  10.0.0.17 (10.0.0.17)  0.068 ms  0.022 ms  0.014 ms
 7  10.0.0.5 (10.0.0.5)  0.036 ms  0.015 ms  0.016 ms
 8  10.0.0.1 (10.0.0.1)  0.033 ms  0.017 ms  0.018 ms
```

-> com sucesso

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

```
<core.35819/AfonsoHenriques.conf# ping 192.168.0.194
PING 192.168.0.194 (192.168.0.194) 56(84) bytes of data.
^C
--- 192.168.0.194 ping statistics ---
2 packets transmitted, 0 received, 100% packet loss, time 1014ms
```

```
16 24.010634052  192.168.0.193  224.0.0.5      OSPF   78 Hello Packet
17 25.701557330  192.168.0.226  192.168.0.194  ICMP   98 Echo (ping) request  id=0x0048, seq=1/256, ttl=55 (reply in 1…
18 25.701597362  192.168.0.194  192.168.0.226  ICMP   98 Echo (ping) reply    id=0x0048, seq=1/256, ttl=64 (request in…
19 25.701608783  192.168.0.193  192.168.0.194  ICMP  126 Destination unreachable (Network unreachable)
```

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

R: Como é possível verificar no wireshark, a conectividade está a ser quebrada no dispositivo de ip 192.168.0.193, falta a entry para Afonso

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
10.0.0.0        172.16.142.1    255.255.255.252 UG        0 0          0 eth0
10.0.0.4        172.16.142.1    255.255.255.252 UG        0 0          0 eth0
10.0.0.8        172.16.142.1    255.255.255.252 UG        0 0          0 eth0
10.0.0.12       172.16.142.1    255.255.255.252 UG        0 0          0 eth0
10.0.0.16       172.16.142.1    255.255.255.252 UG        0 0          0 eth0
10.0.0.20       172.16.142.1    255.255.255.252 UG        0 0          0 eth0
10.0.0.24       172.16.142.1    255.255.255.252 UG        0 0          0 eth0
10.0.0.28       172.16.142.1    255.255.255.252 UG        0 0          0 eth0
172.0.0.0       172.16.142.1    255.0.0.0       UG        0 0          0 eth0
172.16.142.0    0.0.0.0         255.255.255.252 U         0 0          0 eth0
172.16.142.4    172.16.142.1    255.255.255.252 UG        0 0          0 eth0
172.16.143.0    172.16.142.1    255.255.255.252 UG        0 0          0 eth0
172.16.143.4    172.16.142.1    255.255.255.252 UG        0 0          0 eth0
192.168.0.192   0.0.0.0         255.255.255.248 U         0 0          0 eth1
192.168.0.200   172.16.142.1    255.255.255.248 UG        0 0          0 eth0
192.168.0.208   172.16.142.1    255.255.255.248 UG        0 0          0 eth0
192.168.0.216   172.16.142.1    255.255.255.248 UG        0 0          0 eth0
192.168.0.232   172.16.142.1    255.255.255.248 UG        0 0          0 eth0
192.168.0.240   172.16.142.1    255.255.255.248 UG        0 0          0 eth0
192.168.0.248   172.16.142.1    255.255.255.248 UG        0 0          0 eth0
```

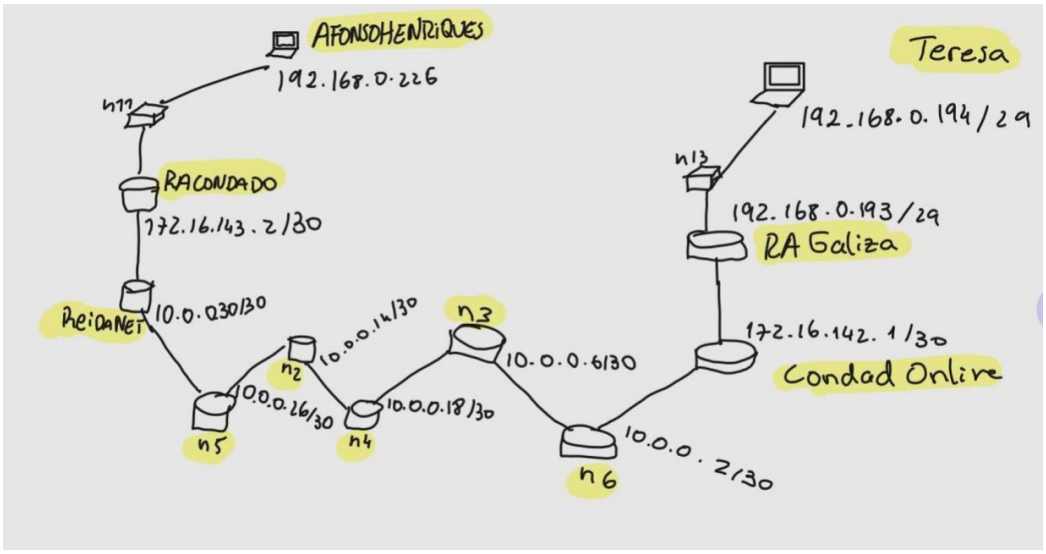
```
root@RAGaliza:/tmp/pycore.35819/RAGaliza.conf# route add -net 192.168.0.224 gw 172.16.142.1 netmask 255.255.255.248
```

->adição da entrada

```
root@Teresa:/tmp/pycore.35819/Teresa.conf# ping 192.168.0.226
PING 192.168.0.226 (192.168.0.226) 56(84) bytes of data.
64 bytes from 192.168.0.226: icmp_seq=1 ttl=55 time=0.099 ms
64 bytes from 192.168.0.226: icmp_seq=2 ttl=55 time=0.176 ms
^C
--- 192.168.0.226 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1027ms
```

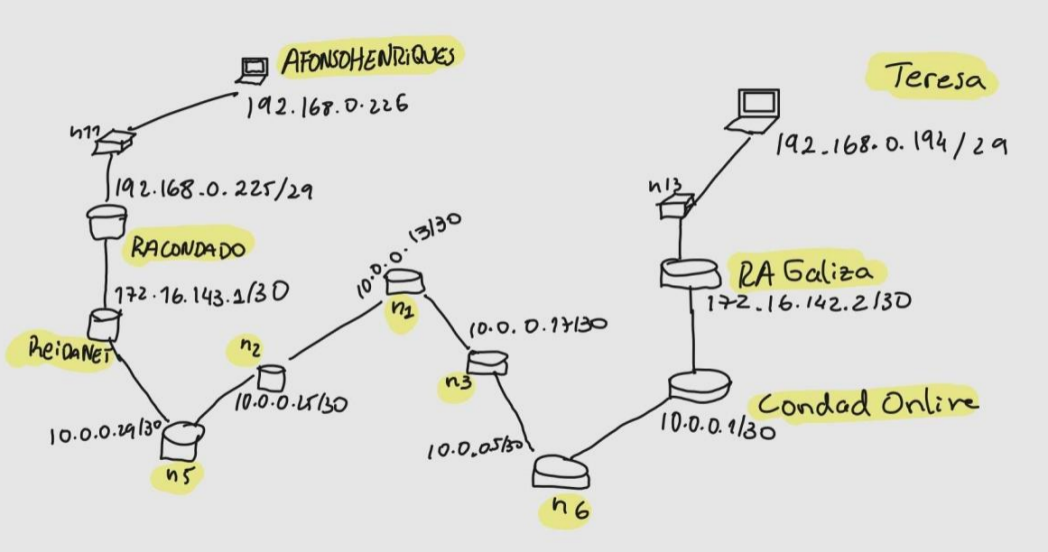
->ligação feita

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




```
root@Teresa:/tmp/pycore.35819/Teresa.conf# traceroute 192.168.0.226
traceroute to 192.168.0.226 (192.168.0.226), 30 hops max, 60 byte packets
 1  192.168.0.193 (192.168.0.193)  0.041 ms  0.005 ms  0.004 ms
 2  172.16.142.1 (172.16.142.1)  0.013 ms  0.006 ms  0.007 ms
 3  10.0.0.2 (10.0.0.2)  0.017 ms  0.008 ms  0.008 ms
 4  10.0.0.6 (10.0.0.6)  0.020 ms  0.010 ms  0.010 ms
 5  10.0.0.18 (10.0.0.18)  0.021 ms  0.011 ms  0.013 ms
 6  10.0.0.14 (10.0.0.14)  0.028 ms  0.023 ms  0.013 ms
 7  10.0.0.26 (10.0.0.26)  0.025 ms  0.016 ms  0.016 ms
 8  10.0.0.30 (10.0.0.30)  0.024 ms  0.018 ms  0.018 ms
 9  172.16.143.2 (172.16.143.2)  0.026 ms  0.020 ms  0.020 ms
10  192.168.0.226 (192.168.0.226)  0.035 ms  0.022 ms  0.021 ms
```

->traceroute de Teresa para Afonso




```
<5819/AfonsoHenriques.conf# traceroute 192.168.0.194
traceroute to 192.168.0.194 (192.168.0.194), 30 hops max, 60 byte packets
 1  192.168.0.225 (192.168.0.225)  0.034 ms  0.007 ms  0.004 ms
 2  172.16.143.1 (172.16.143.1)  0.014 ms  0.006 ms  0.006 ms
 3  10.0.0.29 (10.0.0.29)  0.015 ms  0.007 ms  0.008 ms
 4  10.0.0.25 (10.0.0.25)  0.017 ms  0.009 ms  0.009 ms
 5  10.0.0.13 (10.0.0.13)  0.019 ms  0.012 ms  0.011 ms
 6  10.0.0.17 (10.0.0.17)  0.023 ms  0.024 ms  0.013 ms
 7  10.0.0.5 (10.0.0.5)  0.020 ms  0.014 ms  0.015 ms
 8  10.0.0.1 (10.0.0.1)  0.023 ms  0.016 ms  0.017 ms
 9  172.16.142.2 (172.16.142.2)  0.024 ms  0.018 ms  0.018 ms
10  192.168.0.194 (192.168.0.194)  0.026 ms  0.020 ms  0.021 ms
```

->traceroute de Afonso para Teresa

R: Os caminhos diferem num dispositivo, enquanto que em Afonso para Teresa o pacote toma o caminho pelo n1, no caminho contrário, ele opta por n3.

e) Existe uma correspondência (match) nesta entrada para pacotes enviados para o polo Galiza? E para CDN? Caso seja essa a entrada utilizada para o encaminhamento, permitirá o funcionamento esperado do dispositivo?

R: ->Para CDN não existe correspondência nesta tabela de N3
-> Tem correspondência para Galicia ainda que este router não seja utilizado para reencaminhar pois existe um melhor

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

> **R:**Os endereços utilizados nos quatro protocolos são, de facto, todos privados, de acordo com o protocolo RCF1918 que estabelece regras de endereçamento para endereços privados.

Existem 3 grupos de endereços privados definidos pelo protocolo RCF1918 e, as redes inseridas no primeiro bloco, que é o bloco 192.168.0.0 - 192.168.255.255 /16.
As redes utilizadas pelos ISP's estão inseridas no segundo bloco, que é o bloco 172.16.0.0 - 172.31.255.255/12.
As redes do core estão inseridas no terceiro bloco, que é o bloco 10.0.0.0 - 10.255.255.255 /8.
Dado isto, é possível afirmar então, que todos os endereços utilizados são privados, seguindo as regras estabelecidas pelo protocolo RCF1918.

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

R: Não, os switches não possuem endereços ips pois os switches apenas operam endereços físicos e um endereço ip é um endereço lógica.

2)

a) Não estando satisfeito com a decoração do Castelo, opta por eliminar a sua rota default.
Adicione as rotas necessárias para que o Castelo continue a ter acesso a cada um dos três polos. Mostre que a conectividade é restabelecida, assim como a tabela de encaminhamento resultante. Explicite ainda a utilidade de uma rota default.

R: Uma rota default é útil para encaminhar todo o tráfego de rede que não possui uma rota específica na tabela de encaminhamento.

```
K819/Castelo.conf# route del -net 0.0.0.0  netmask 0.0.0.0
```

->del da rota default do Castelo

Foram adicionadas as rotas para as redes e subredes dos 3 polos.

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
192.168.0.192   192.168.0.225   255.255.255.248 UG        0 0          0 eth0
192.168.0.200   192.168.0.225   255.255.255.248 UG        0 0          0 eth0
192.168.0.208   192.168.0.225   255.255.255.248 UG        0 0          0 eth0
192.168.0.216   192.168.0.225   255.255.255.248 UG        0 0          0 eth0
192.168.0.224   0.0.0.0         255.255.255.248 U         0 0          0 eth0
192.168.0.232   192.168.0.225   255.255.255.248 UG        0 0          0 eth0
192.168.0.240   192.168.0.225   255.255.255.248 UG        0 0          0 eth0
192.168.0.248   192.168.0.225   255.255.255.248 UG        0 0          0 eth0
```

-> tabela de encaminhamento atualizada

Confirmações ping:

```
root@Castelo:/tmp/pycore.35819/Castelo.conf# ping 192.168.0.234
PING 192.168.0.234 (192.168.0.234) 56(84) bytes of data.
64 bytes from 192.168.0.234: icmp_seq=1 ttl=61 time=0.098 ms
64 bytes from 192.168.0.234: icmp_seq=2 ttl=61 time=0.077 ms
64 bytes from 192.168.0.234: icmp_seq=3 ttl=61 time=0.090 ms
^C
--- 192.168.0.234 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2042ms
rtt min/avg/max/mdev = 0.077/0.088/0.098/0.008 ms
```

-> UMinho

```
root@Castelo:/tmp/pycore.35819/Castelo.conf# ping 192.168.0.242
PING 192.168.0.242 (192.168.0.242) 56(84) bytes of data.
64 bytes from 192.168.0.242: icmp_seq=1 ttl=61 time=0.084 ms
64 bytes from 192.168.0.242: icmp_seq=2 ttl=61 time=0.112 ms
64 bytes from 192.168.0.242: icmp_seq=3 ttl=61 time=0.110 ms
^C
--- 192.168.0.242 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2051ms
rtt min/avg/max/mdev = 0.084/0.102/0.112/0.012 ms
```

->DI

```
root@Castelo:/tmp/pycore.35819/Castelo.conf# ping 192.168.0.250
PING 192.168.0.250 (192.168.0.250) 56(84) bytes of data.
64 bytes from 192.168.0.250: icmp_seq=1 ttl=61 time=0.066 ms
64 bytes from 192.168.0.250: icmp_seq=2 ttl=61 time=0.112 ms
64 bytes from 192.168.0.250: icmp_seq=3 ttl=61 time=0.100 ms
^C
--- 192.168.0.250 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2025ms
```

->Finanças

```
root@Castelo:/tmp/pycore.35819/Castelo.conf# ping 192.168.0.196
PING 192.168.0.196 (192.168.0.196) 56(84) bytes of data.
64 bytes from 192.168.0.196: icmp_seq=1 ttl=55 time=0.144 ms
64 bytes from 192.168.0.196: icmp_seq=2 ttl=55 time=0.178 ms
64 bytes from 192.168.0.196: icmp_seq=3 ttl=55 time=0.170 ms
^C
--- 192.168.0.196 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2033ms
```

->Torradeira

```
root@Castelo:/tmp/pycore.35819/Castelo.conf# ping 192.168.0.202
PING 192.168.0.202 (192.168.0.202) 56(84) bytes of data.
64 bytes from 192.168.0.202: icmp_seq=1 ttl=55 time=0.131 ms
64 bytes from 192.168.0.202: icmp_seq=2 ttl=55 time=0.185 ms
64 bytes from 192.168.0.202: icmp_seq=3 ttl=55 time=0.180 ms
^C
--- 192.168.0.202 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2039ms
```

->Youtube

```
root@Castelo:/tmp/pycore.35819/Castelo.conf# ping 192.168.0.210
PING 192.168.0.210 (192.168.0.210) 56(84) bytes of data.
64 bytes from 192.168.0.210: icmp_seq=1 ttl=55 time=0.115 ms
64 bytes from 192.168.0.210: icmp_seq=2 ttl=55 time=0.182 ms
64 bytes from 192.168.0.210: icmp_seq=3 ttl=55 time=0.173 ms
^C
--- 192.168.0.210 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2050ms
```

->Itunes

->Spotify

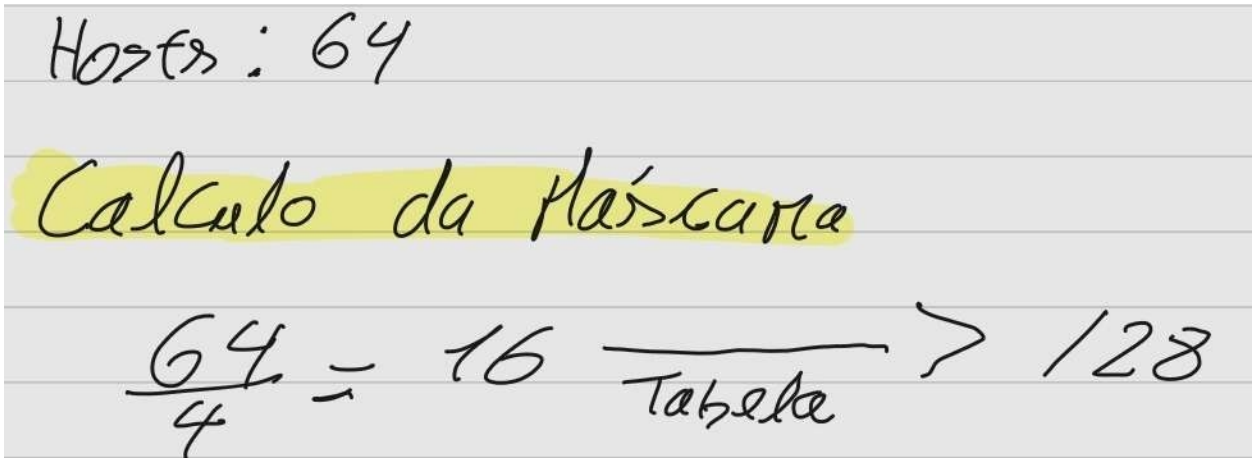
b) **Por modo a garantir uma posição estrategicamente mais vantajosa e ter casa de férias para relaxar entre batalhas, ordena também a construção de um segundo Castelo, em Braga. Não tendo qualquer queixa do serviço prestado, recorre novamente aos serviços do ISP ReiDaNet para ter acesso à rede no segundo Castelo. O ISP atribuiu-lhe o endereço de rede IP 172.16.XX.128/26 em que XX corresponde ao seu número de grupo (PLXX). Defina um esquema de endereçamento que permita o estabelecimento de pelo menos 3 redes e que garanta que cada uma destas possa ter 10 ou mais hosts. Assuma que todos os endereços de sub-redes são utilizáveis.**

Caracteristicas da subrede:

- Tabela de sIP 172.16.10.128/26
- Redes >= 3
- Cada rede tenha 10 ou mais Hosts

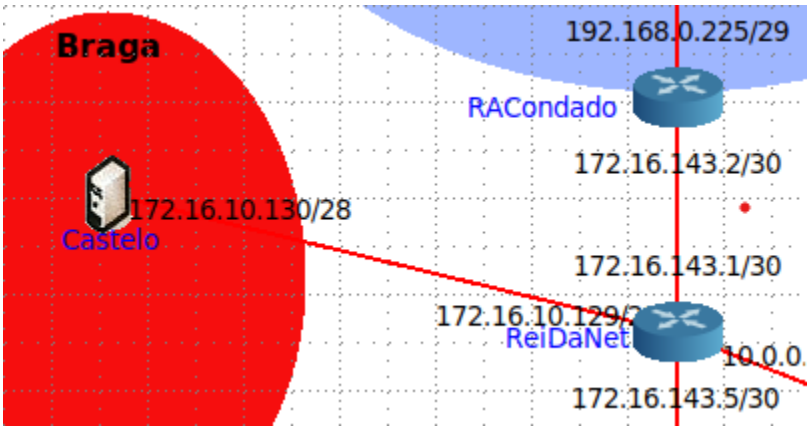
	Hosts	Netmask	Number of Subnets
/30	4	255.255.255.252	64
/29	8	255.255.255.248	32
/28	16	255.255.255.240	16
/27	32	255.255.255.224	8
/26	64	255.255.255.192	4

->Tabela Subnet

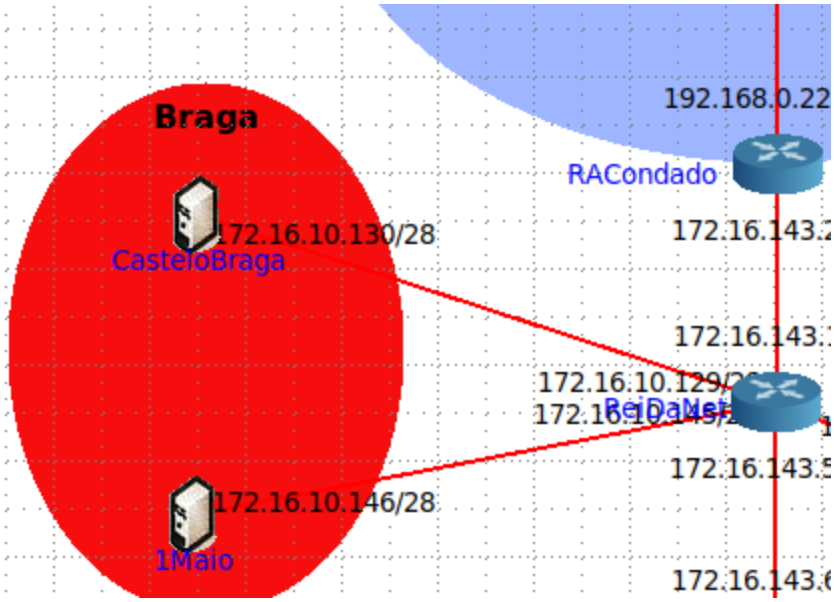


Logo a subrede para Braga vai ficar com as seguintes subredes:

- 172.16.10.128/28 -> hosts: 172.16.10.129-172.16.10.142 e broadcast 172.16.10.143
- 172.16.10.144/28 -> hosts: 172.16.10.145-172.16.10.158 e broadcast 172.16.10.159
- 172.16.10.160/28 -> hosts: 172.16.10.161-172.16.10.174 e broadcast 172.16.19.175
- 172.16.10.176/28 -> hosts: 172.16.10.177-172.16.10.186 e broadcast 172.16.19.187



c) **Ligue um novo host diretamente ao router ReiDaNet. Associe-lhe um endereço, à sua escolha, pertencente a uma sub-rede disponível das criadas na alínea anterior (garanta que a interface do router ReiDaNet utiliza o primeiro endereço da sub-rede escolhida). Verifique que tem conectividade com os diferentes polos. Existe algum host com o qual não seja possível comunicar? Porquê?**



->adição do ”1Maio”

```
root@1Maio:/tmp/pycore.37611/1Maio.conf# traceroute 192.168.0.228
traceroute to 192.168.0.228 (192.168.0.228), 30 hops max, 60 byte packets
 1  172.16.10.145 (172.16.10.145)  0.225 ms  0.194 ms  0.185 ms
 2  172.16.143.2 (172.16.143.2)  0.176 ms  0.156 ms  0.146 ms
 3  * * *
 4  * * *
 5  * * *
 6  * * *
 7  * * *
 8  *^C
```

-> Não há conectividade com o Castelo

R: O “1Maio” consegue estabelecer conexão normal com todos os hosts excepto o host “Castelo”, isto porque na chegada dos pacotes ao host “Castelo” estes não sabem o caminho de volta.

3)

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

R: Para fazer sumarizar as rotas(supernetting) vamos pegar no ip destination rotas que queremos juntar e passá-las para binário, posteriormente vamos verificar até que bit as rotas são idênticas. O número de bits que são idênticos vai determinar a máscara aplicada, depois basta deletar as antigas rotas e adicionar a nova rota para todas.

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
10.0.0.0        0.0.0.0         255.255.255.252 U         0 0          0 eth0
10.0.0.4        0.0.0.0         255.255.255.252 U         0 0          0 eth1
10.0.0.8        10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.12       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.16       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.20       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.24       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.28       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
172.0.0.0       10.0.0.6        255.0.0.0       UG        0 0          0 eth1
172.16.142.0    10.0.0.1        255.255.255.252 UG        0 0          0 eth0
172.16.142.4    10.0.0.1        255.255.255.252 UG        0 0          0 eth0
172.16.143.0    10.0.0.6        255.255.255.252 UG        0 0          0 eth1
172.16.143.4    10.0.0.6        255.255.255.252 UG        0 0          0 eth1
192.168.0.192   10.0.0.1        255.255.255.248 UG        0 0          0 eth0
192.168.0.200   10.0.0.1        255.255.255.248 UG        0 0          0 eth0
192.168.0.208   10.0.0.1        255.255.255.248 UG        0 0          0 eth0
192.168.0.216   10.0.0.1        255.255.255.248 UG        0 0          0 eth0
192.168.0.224   10.0.0.6        255.255.255.248 UG        0 0          0 eth1
192.168.0.232   10.0.0.6        255.255.255.248 UG        0 0          0 eth1
192.168.0.240   10.0.0.6        255.255.255.248 UG        0 0          0 eth1
192.168.0.248   10.0.0.6        255.255.255.248 UG        0 0          0 eth1
```

->antiga tabela

IPs para binário:

- 192.168.0.192/29 11000000.10101000.00000000.110 | 00000
- 192.168.0.200/29 11000000.10101000.00000000.110 | 01000
- 192.168.0.208/29 11000000.10101000.00000000.110 | 10000
- 192.168.0.216/29 11000000.10101000.00000000.110 | 11000

Ficamos com os bits iguais de:

11000000.10101000.00000000.110 | 00000

Logo temos uma máscara de 192.168.0.192/27

Ao retirar as entradas para Galiza e CDN e adicionando a nova entrada obtida através de supernetting, obtemos a seguinte tabela:

```
root@n6:/tmp/pycore.37611/n6.conf# netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
10.0.0.0        0.0.0.0         255.255.255.252 U         0 0          0 eth0
10.0.0.4        0.0.0.0         255.255.255.252 U         0 0          0 eth1
10.0.0.8        10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.12       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.16       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.20       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.24       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.28       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
172.0.0.0       10.0.0.6        255.0.0.0       UG        0 0          0 eth1
172.16.142.0    10.0.0.1        255.255.255.252 UG        0 0          0 eth0
172.16.142.4    10.0.0.1        255.255.255.252 UG        0 0          0 eth0
172.16.143.0    10.0.0.6        255.255.255.252 UG        0 0          0 eth1
172.16.143.4    10.0.0.6        255.255.255.252 UG        0 0          0 eth1
192.168.0.192   10.0.0.1        255.255.255.224 UG        0 0          0 eth0
192.168.0.224   10.0.0.6        255.255.255.248 UG        0 0          0 eth1
192.168.0.232   10.0.0.6        255.255.255.248 UG        0 0          0 eth1
192.168.0.240   10.0.0.6        255.255.255.248 UG        0 0          0 eth1
192.168.0.248   10.0.0.6        255.255.255.248 UG        0 0          0 eth1
```

->**nova tabela**

```
<ycore.37611/AfonsoHenriques.conf# ping 192.168.0.194
PING 192.168.0.194 (192.168.0.194) 56(84) bytes of data.
64 bytes from 192.168.0.194: icmp_seq=1 ttl=55 time=0.127 ms
^C
--- 192.168.0.194 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.127/0.127/0.127/0.000 ms
<ycore.37611/AfonsoHenriques.conf# ping 192.168.0.210
PING 192.168.0.210 (192.168.0.210) 56(84) bytes of data.
64 bytes from 192.168.0.210: icmp_seq=1 ttl=55 time=0.129 ms
^C
--- 192.168.0.210 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.129/0.129/0.129/0.000 ms
```

-> conectividade assegurada

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

```
root@n6:/tmp/pycore.37611/n6.conf# netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
10.0.0.0        0.0.0.0         255.255.255.252 U         0 0          0 eth0
10.0.0.4        0.0.0.0         255.255.255.252 U         0 0          0 eth1
10.0.0.8        10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.12       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.16       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.20       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.24       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.28       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
172.0.0.0       10.0.0.6        255.0.0.0       UG        0 0          0 eth1
172.16.142.0    10.0.0.1        255.255.255.252 UG        0 0          0 eth0
172.16.142.4    10.0.0.1        255.255.255.252 UG        0 0          0 eth0
172.16.143.0    10.0.0.6        255.255.255.252 UG        0 0          0 eth1
172.16.143.4    10.0.0.6        255.255.255.252 UG        0 0          0 eth1
192.168.0.192   10.0.0.1        255.255.255.224 UG        0 0          0 eth0
192.168.0.224   10.0.0.6        255.255.255.248 UG        0 0          0 eth1
192.168.0.232   10.0.0.6        255.255.255.248 UG        0 0          0 eth1
192.168.0.240   10.0.0.6        255.255.255.248 UG        0 0          0 eth1
192.168.0.248   10.0.0.6        255.255.255.248 UG        0 0          0 eth1
```

->antiga tabela

IPs para binário:

- **192.168.0.224/29 11000000.10101000.00000000.111 | 00000**
- **192.168.0.232/29 11000000.10101000.00000000.111 | 01000**
- **192.168.0.240/29 11000000.10101000.00000000.111 | 10000**
- **192.168.0.248/29 11000000.10101000.00000000.111 | 11000**

Ficamos com os bits iguais de:

11000000.10101000.00000000.111 | 00000

Logo temos máscara de 192.168.0.224/27

Ao retirar as entradas para CondadoPortucalense e Institucional adicionando a nova entrada obtida

através de supernetting, obtemos a seguinte tabela:

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
10.0.0.0        0.0.0.0         255.255.255.252 U         0 0          0 eth0
10.0.0.4        0.0.0.0         255.255.255.252 U         0 0          0 eth1
10.0.0.8        10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.12       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.16       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.20       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.24       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
10.0.0.28       10.0.0.6        255.255.255.252 UG        0 0          0 eth1
172.0.0.0       10.0.0.6        255.0.0.0       UG        0 0          0 eth1
172.16.142.0    10.0.0.1        255.255.255.252 UG        0 0          0 eth0
172.16.142.4    10.0.0.1        255.255.255.252 UG        0 0          0 eth0
172.16.143.0    10.0.0.6        255.255.255.252 UG        0 0          0 eth1
172.16.143.4    10.0.0.6        255.255.255.252 UG        0 0          0 eth1
192.168.0.192   10.0.0.1        255.255.255.224 UG        0 0          0 eth0
192.168.0.224   10.0.0.6        255.255.255.224 UG        0 0          0 eth1
```

->nova tabela

```
PING 192.168.0.226 (192.168.0.226) 56(84) bytes of data.
64 bytes from 192.168.0.226: icmp_seq=1 ttl=55 time=0.129 ms
^C
--- 192.168.0.226 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.129/0.129/0.129/0.000 ms
root@Teresa:/tmp/pycore.37611/Teresa.conf# ping 192.168.0.234
PING 192.168.0.234 (192.168.0.234) 56(84) bytes of data.
64 bytes from 192.168.0.234: icmp_seq=1 ttl=55 time=0.111 ms
^C
--- 192.168.0.234 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.111/0.111/0.111/0.000 ms
```

->conectividade assegurada

c) Comente os aspectos positivos e negativos do uso do Supperneting.

R: O uso de Supernetting possui tanto aspectos positivos como aspectos negativos.

Lados positivos:

- Diminui o tamanho das tabelas de reencaminhamento, o que facilita a leitura das entradas;
- Aumenta a eficiência do encaminhamento;
- Facilita o gerenciamento das tabelas;

Lados negativos:

- Perdesse flexibilidade, os IPs ao estarem agregados fica mais complicado de mexer individualmente;
- Cria possíveis conflitos de endereços;
- Aumenta o tráfego de Broadcast;