

Universidade do Minho

”TP2: Protocolo IPv4”
Datagramas e Fragmentação
Endereçamento e Encaminhamento IP
Grupo 52

Licenciatura em Engenharia Informática

Redes de Computadores

a85646	a98286	a87978
Hugo Teles Silva	Luís Ferreira	Tiago Cunha

Braga, 20 de Abril de 2023

Pergunta 1

- 1 Prepare uma topologia CORE para verificar o comportamento do traceroute. Na topologia deve existir: um host (pc) cliente designado Lost, cujo router de acesso é RA1; o router RA1 está simultaneamente ligado a dois routers no core da rede RC1 e RC2; estes estão conectados a um router de acesso RA2, que por sua vez, se liga a um host (servidor) designado Found. Ajuste o nome dos equipamentos atribuídos por defeito para o enunciado. Apenas nas ligações (links) da rede de core, estabeleça um tempo de propagação de 15 ms. Após ativar a topologia, note que pode não existir conectividade IP imediata entre Lost e Found até que o anúncio de rotas entre routers estabilize.

A topologia criada pelo grupo é a que se apresenta de seguida:

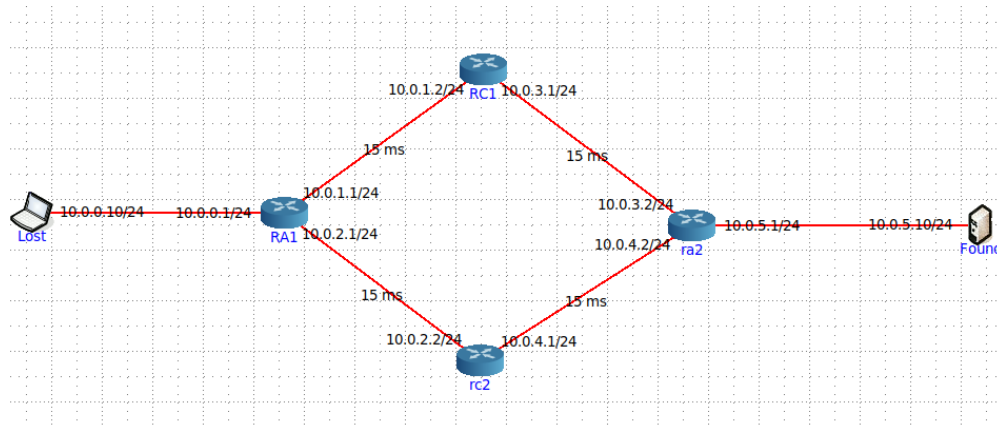


Figure 1: Topologia criada

- 1.a Active o Wireshark no host Lost. Numa shell de Lost execute o comando traceroute -I para o endereço IP do Found. 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.

```
root@Lost:/tmp/pycore.40227/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)  0.057 ms  0.016 ms  0.013 ms
 2  10.0.1.2 (10.0.1.2)  30.101 ms  30.194 ms  30.186 ms
 3  10.0.3.2 (10.0.3.2)  60.294 ms  60.288 ms  60.280 ms
 4  10.0.5.10 (10.0.5.10)  60.321 ms  60.313 ms  60.305 ms
```

Figure 2: Output do comando traceroute no terminal

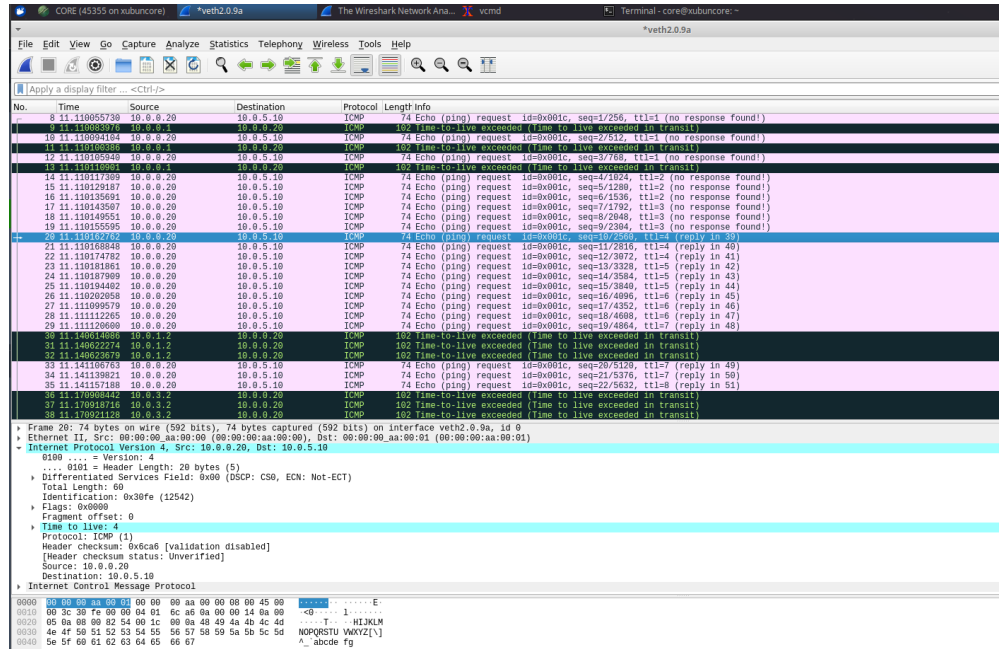


Figure 3: Output Wireshark

O comportamento da rota do tráfego ICMP entre o PC Lost e o servidor Found foi obtido através do comando do traceroute tal como se pode verificar na Figure 2, este mesmo comando que permite descobrir a rota (salto-a-salto) desde uma origem IP até um destino IP, neste caso desde o IP do sistema Lost e o IP do servidor Found, respetivamente.

Na shell de Lost foi executado o comando *traceroute -I* para o endereço IP do Found (10.0.5.10), cada linha da Figure 2 representa um salto e cada uma das 3 colunas representam um RTT (*Round-Trip Time*). Existem 3 tempos/colunas pois o comando *traceroute* envia por defeito 3 pacotes. Na Figure 2 é evidenciado o delay colocado nos links da rede CORE na linha 2 e 3. Como é visível nestas linhas o RTT é maior que nas linhas 1 e 4 (links fora da rede CORE). O tráfego ICMP enviado pelo sistema Lost e o tráfego ICMP recebido como resposta podem ser verificados através da Figure 3.

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

Os resultados do Wireshark permitem-nos chegar à conclusão que os primeiros pacotes enviados pela origem do PC Lost e que chegaram ao destino do servidor Found sem tendo sido retornado nenhum erro foi o pacote com o ID 0x001c cujo TTL (Time-To-Live) é igual a 4.

Tendo em conta o funcionamento do traceroute, sabemos que o traceroute opera da seguinte maneira: inicialmente, é enviado um ou mais datagramas com o campo TTL igual a 1; seguidamente, é enviado um ou mais datagramas com o TTL a 2; depois com o TTL a 3; e assim sucessivamente.

Cada router no percurso até ao destino deve decrementar de 1 o TTL de cada datagrama recebido, se o TTL atingir o valor 0, o router terá que descartar o datagrama e devolver uma mensagem de controlo ICMP ao host de origem

Uma vez que estes primeiros pacotes não retornaram nenhum erro, isto é, o TTL não atingiu o valor 0 e portanto, o router não descartou o datagrama nem devolveu uma mensagem de controlo ICMP ao host de origem, devido ao facto que o TTL não foi excedido, é possível verificar que o TTL necessário para chegar ao servidor Found é igual a 4.

1.c 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.

```
root@Lost:/tmp/pycore.45355/Lost.conf# traceroute -q 6 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) 0.070 ms 0.010 ms 0.007 ms 0.007 ms 0.007 ms 0.008
   ms
 2 10.0.1.2 (10.0.1.2) 31.201 ms 31.178 ms 31.158 ms 31.140 ms 31.122 ms
   31.102 ms
 3 10.0.3.2 (10.0.3.2) 61.188 ms 61.168 ms 61.149 ms 61.130 ms 60.306 ms
   60.264 ms
 4 10.0.5.10 (10.0.5.10) 60.883 ms 60.834 ms 60.807 ms 60.786 ms 60.476 ms
   60.441 ms
```

Figure 4: Tempos de RTT no acesso ao servidor

Calculando a média:

$$\frac{60.883 + 60.834 + 60.807 + 60.786 + 60.476 + 60.441}{6} = 60.7045ms \quad (1)$$

1.d 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?.

Como neste caso o caminho de volta é o mesmo caminho que o caminho de ida e não há alteração nos links para deixarem de ser simétricos podemos afirmar que o valor médio do atraso num sentido (One-Way Delay) poderia ser calculado com precisão dividindo o RTT por dois. Numa rede real tal não acontece pois existem rotas assimétricas onde o caminho de ida e o caminho de volta não são o mesmo, congestionamento em routers, jitter e fatores externos como falhas ou quebras que influenciam o resultado do RTT num dos sentidos o que afeta a precisão da divisão para calcular o One-Way Delay.

Pergunta 2

- 2 Pretende-se agora usar o traceroute na sua máquina nativa e gerar datagramas IP de diferentes tamanhos. Usando o wireshark capture o tráfego gerado pelo traceroute sem especificar o tamanho do pacote, i.e., quando é usado o tamanho do pacote de prova por defeito. Utilize como máquina destino o host marco.uminho.pt. Pare a captura. Com base no tráfego capturado, identifique os pedidos ICMP Echo Request e o conjunto de mensagens devolvidas como resposta. Selecione a primeira mensagem ICMP capturada e centre a análise no nível protocolar IP e, em particular, do cabeçalho IP (expanda o tab correspondente na janela de detalhe do wireshark).



```
68 26.444928683 172.26.42.221 193.136.9.254 ICMP 528 Echo (ping) request id=0x0002, seq=1/256, ttl=1 (no response)
```

Figure 5: Mensagem capturada

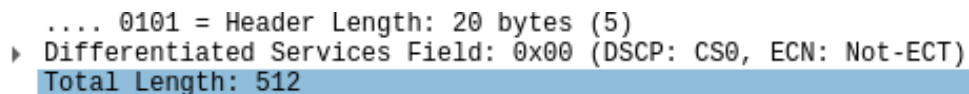
- 2.a Qual é o endereço IP da interface ativa do seu computador?

172.26.42.221

- 2.b Qual é o valor do campo protocol? O que permite identificar?

ICMP, permite identificar que iniciamos o processo de traceroute, onde enviamos um ou mais TTL.

- 2.c Quantos bytes tem o cabeçalho IPv4? Quantos bytes tem o campo de dados (payload) do datagrama? Como se calcula o tamanho do payload?



```
.... 0101 = Header Length: 20 bytes (5)
▶ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 512
```

Figure 6: Header

O cabeçalho possui 20 bytes. Ao valor da total length (512) retiramos o tamanho do cabeçalho para obter o payload, ou seja, o payload tem 492 bytes.

2.d O datagrama IP foi fragmentado? Justifique.

```
Flags: 0x0000
Fragment offset: 0
```

Figure 7: Fragment offset

Devido ao fragment offset estar a 0, o datagram IP não foi fragmentado

2.e Ordene os pacotes capturados de acordo com o endereço IP fonte (e.g., selecionando o cabeçalho da coluna Source), e analise a sequência de tráfego ICMP gerado a partir do endereço IP atribuído à interface da sua máquina. Para a sequência de mensagens ICMP enviadas pelo seu computador, indique que campos do cabeçalho IP variam de pacote para pacote.

```
95 26.458332938 172.26.254.254 172.26.42.221 ICMP 70 Time-to-live exceeded (Time to live exceeded in transit)
96 26.458345491 172.26.254.254 172.26.42.221 ICMP 70 Time-to-live exceeded (Time to live exceeded in transit)
97 26.458357307 172.26.254.254 172.26.42.221 ICMP 70 Time-to-live exceeded (Time to live exceeded in transit)
```

Figure 8: Packets ordenados

Os campos que variam de pacote para pacote são o 'Identification', 'time to live' e 'Header Checksum'

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

O 'Identification' aumenta em cada pacote. Já o TTL aumenta de 3 em 3 devido ao Traceroute enviar 3 pacotes com o mesmo TTL de cada vez

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

2.g.1 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?

Os TTL não se mantêm constantes, devido a terem origens diferentes

2.g.2 Porque razão as mensagens de resposta ICMP TTL Exceeded são sempre enviadas na origem com um valor TTL relativamente alto?

As mensagens enviadas à origem com TTL alto são assim porque o objetivo é informar a origem que a packet chegou ao seu limite máximo de saltos. Se o TTL fosse baixo não obteríamos a informação do quão longe a packet foi.

2.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?

A informação contida no cabeçalho ICMP não deve ser contida no cabeçalho IPv4 devido as suas distintas funções. O ICMP tem de dar feedback sobre problemas de comunicação na rede e o IPv4 tem endereçar e rotear os packets. Caso a informação fosse enviada junta o overhead seria reduzida, o que levaria a uma maior eficiência do protocolo, mas o cabeçalho IPv4 tornaria-se mais complexo, aumentando a possibilidade de erros de comunicação e tornaria a resolução de problemas de rede mais difícil, exigindo análise mais profunda do tráfego da rede.

Pergunta 3

3 Pretende-se agora analisar a fragmentação de pacotes IP. Usando o Wireshark, capture e observe o tráfego gerado depois do tamanho de pacote ter sido definido para $(3500 + X)$ bytes, em que X é o número do grupo de trabalho. Documente e justifique todas as respostas às seguintes alíneas:

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

Houve necessidade de fragmentar, pois o tamanho do pacote excede o MTU (Max Transfer Unit) de 1500 bytes.

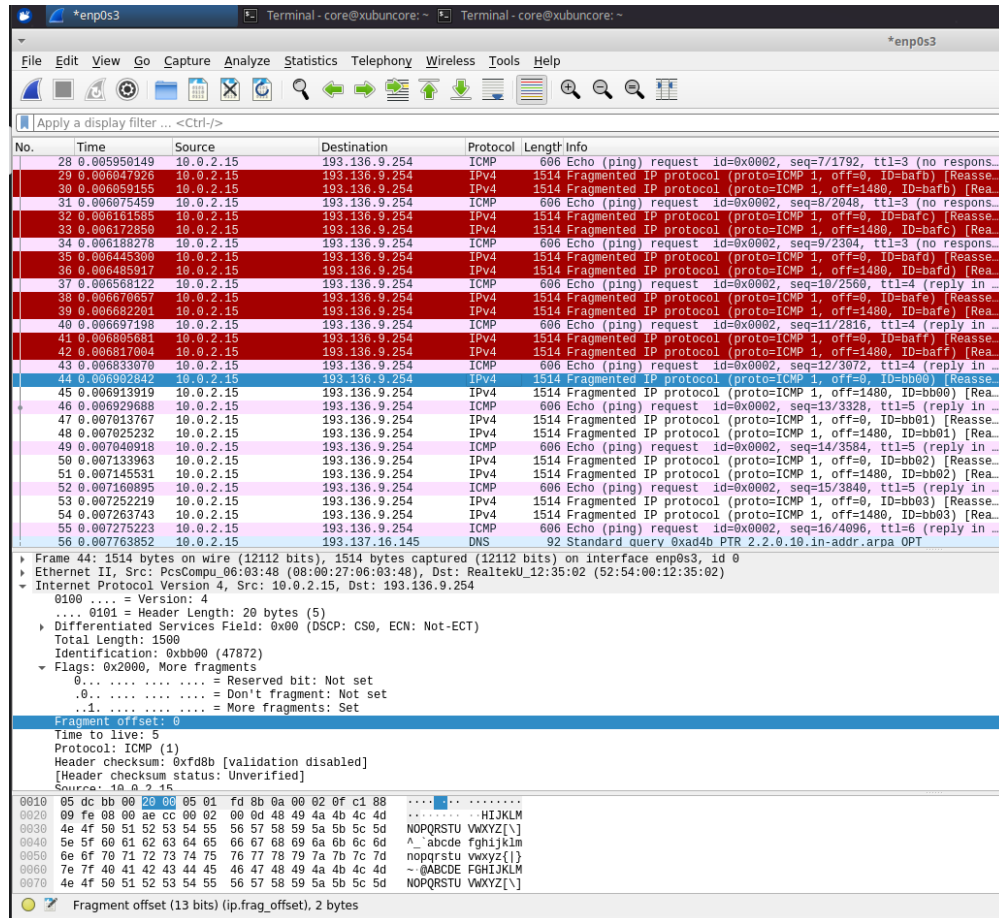


Figure 9: Packets capturados

```

Flags: 0x2000, More fragments
0... .. = Reserved bit: Not set
.0... .. = Don't fragment: Not set
..1... .. = More fragments: Set

```

Figure 10: Fragments

3.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?

Através do valor da flag (more fragments) que é 1, é possível verificar que o datagrama foi fragmentado.

O fragment offset com valor 0, indica que se trata do primeiro fragmento. O tamanho do datagrama IP é 1500 que coincide com o MTU.

3.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?

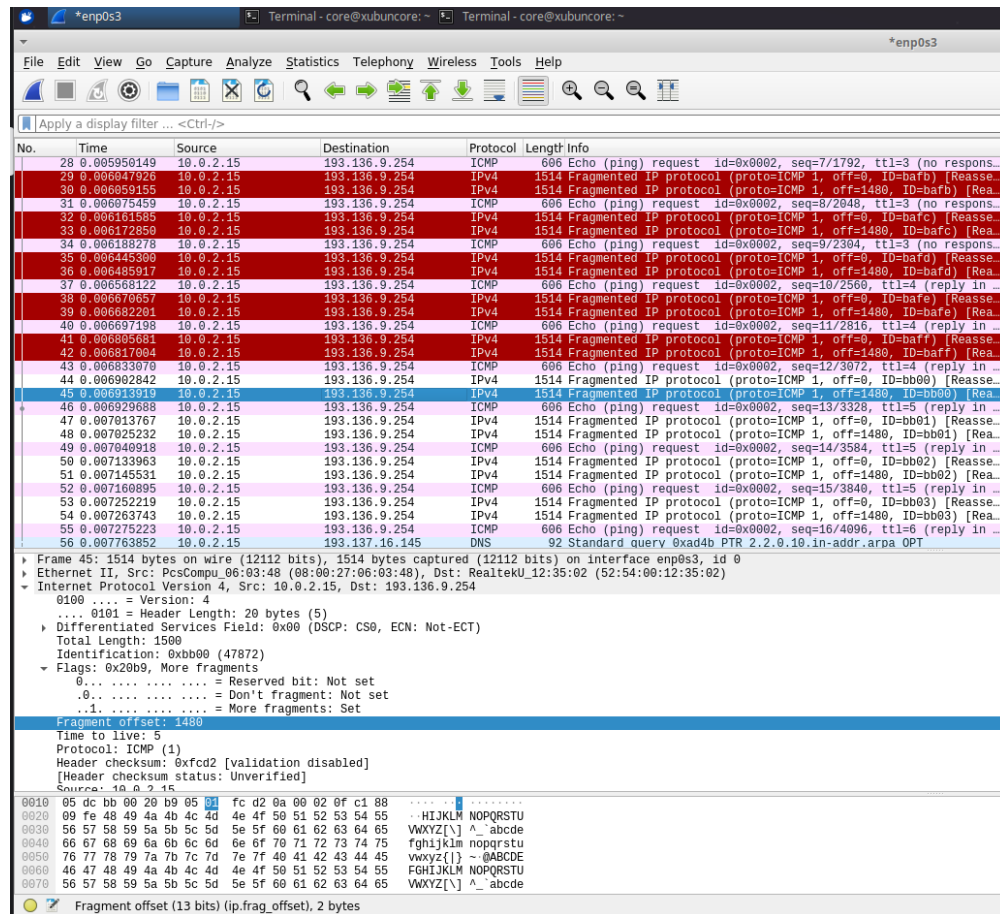


Figure 11: Segundo fragmento

Através do fragment offset que é 1480, indica que não se trata do primeiro fragmento, mas sim do segundo. Quanto à existência de mais fragmentos, a flag "more fragments" permite verificar a existência de mais fragmentos.

3.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.

Como fizemos o traceroute para 3552bytes e sendo o MTU 1500 bytes, dividindo 3552 bytes em fragmentos de, no máximo, 1500 bytes, na teoria teríamos 3 fragmentos, dois com 1500 bytes cada e um ultimo com 552 bytes restantes. É possível ver no wireshark o total length de cada fragmento e no ultimo a flag "more fragments" a 0 com offset de 2960.

Nos tamanhos dos fragmentos apresentados no wireshark temos 1480 para cada um dos dois fragmentos iniciais.

Aos quais se soma o length do ultimo fragmento (sem contabilizar o cabeçalho IP de 20 bytes) 572 bytes.

A soma destes valores não corresponde ao nosso pacote enviado (3352).

3.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.

O último fragmento correspondente ao datagrama original pode ser detetado através do valor da flag more fragments.

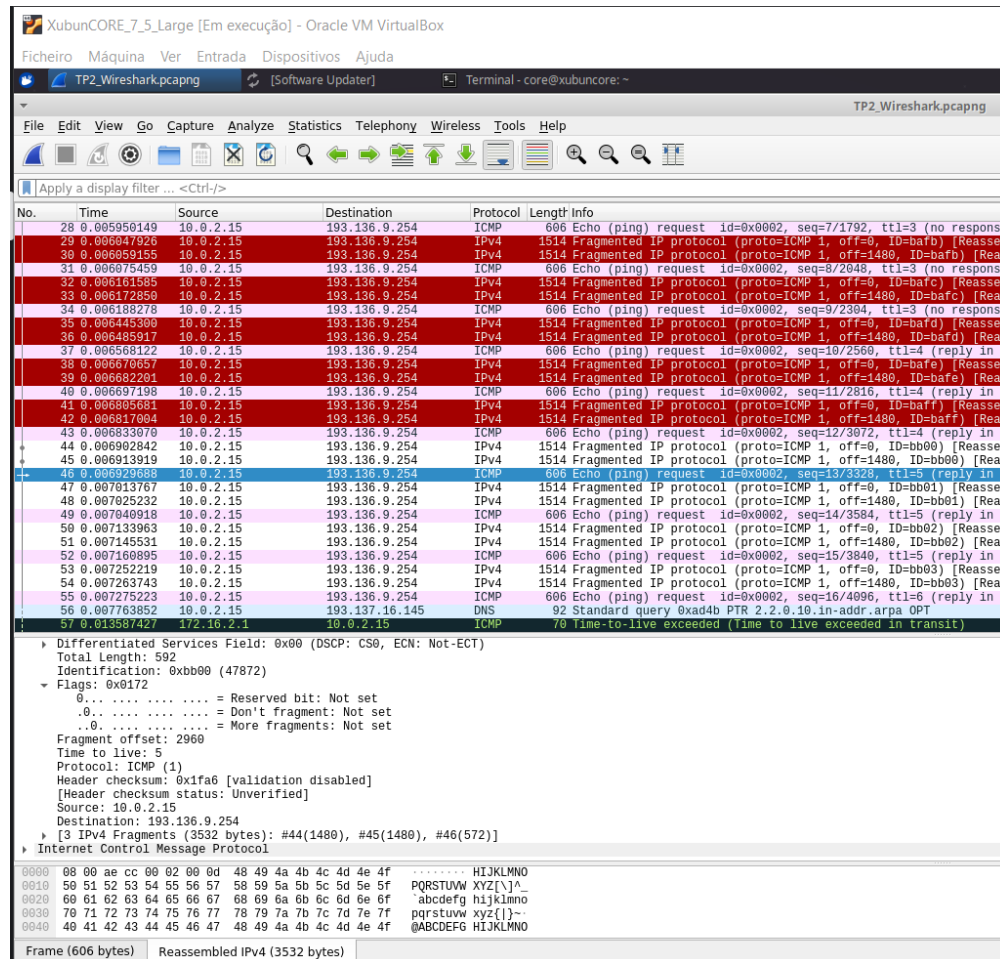


Figure 12: Ultimo fragmento

Como se pode verificar, para o datagrama com o identificador apresentado, 0xbb00, a flag more fragments tem como valor 0, o que indica que se trata do último fragmento. O filtro "ip.flags.mf == 0" permite listar

os últimos fragmentos de cada datagrama IP segmentado.

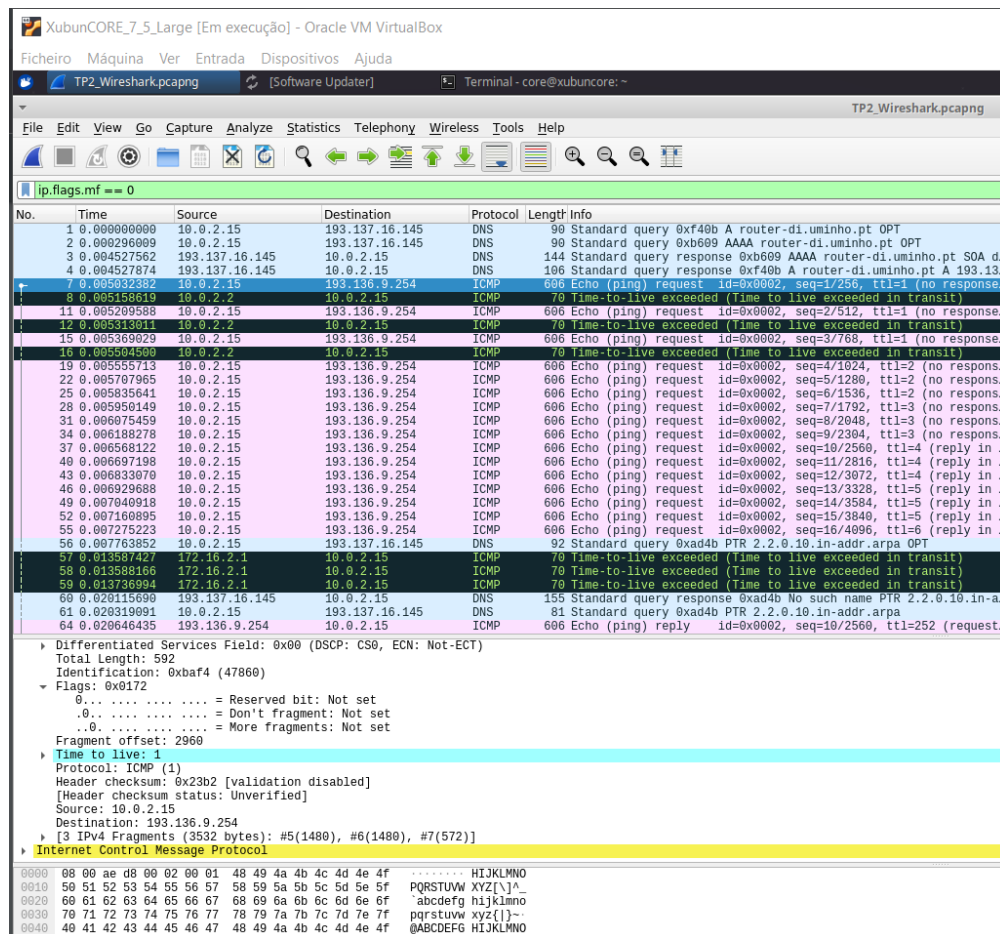


Figure 13: Ultimo fragmento pt2

Tendo sido selecionado o último fragmento do primeiro datagrama IP segmentado.

3.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?

O datagrama IP original é reconstruído a partir dos fragmentos no destinatário. A reconstrução não poderia ter ocorrido noutro equipamento diferente do identificado, uma vez que esse equipamento seria um equipamento intermédio e teria que voltar a ser fragmentado para chegar ao destino.

3.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.

Campos que variam:

Através da flag do fragmento (more fragments, setado a 1), podemos determinar a existência de mais datagramas, de modo a permitir a reconstrução do datagrama.

Através do fragment offset, que indica a posição do datagrama, permitindo ordenar os datagramas da maneira correta.

Campo que não variam:

O ID do fragmento, que permite identificar fragmentos do mesmo pacote, é um dos campos que permite distinguir diferentes fragmentos, de modo a permitir a reconstrução do datagrama original.

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

Apenas o primeiro fragmento de cada pacote é identificado como sendo um pacote ICMP, pois é o único fragmento que contém o cabeçalho ICMP completo, que contém as informações necessárias para identificar a mensagem ICMP e interpretar seu significado.

3.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?

O tamanho dos datagramas é comparado com o valor do MTU definido na rede (1500 bytes).

Aumentar o valor do MTU pode ter como efeito positivo na rede o aumento da eficiência da rede.

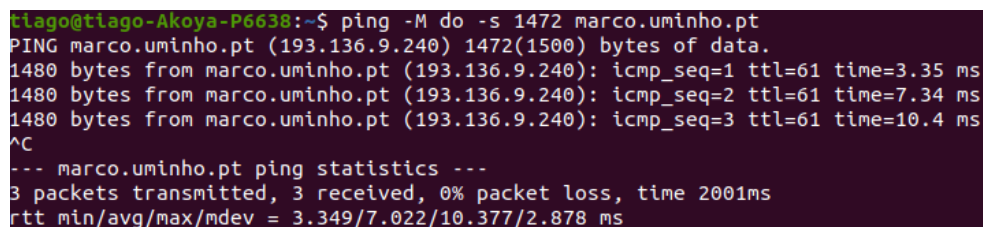
No entanto, o aumento do valor do MTU também pode ter oferecer aspetos desvantajosos na rede, tais como, o aumento da probabilidade de fragmentação e o aumento da latência.

Diminuir o valor do MTU pode ter o seguinte efeito positivo na rede, a redução da probabilidade de fragmentação.

Contudo, diminuir o valor do MTU também pode ter alguns efeitos negativos na rede, tais como, o aumento do tráfego de rede.

3.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 (opção DF) (opção pkt size) 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

Para sabermos o tamanho máximo permitido precisamos de saber o MTU. Como foi dito anteriormente este têm um valor de 1500. Para saber o size maximo temos de subtrair ao MTU 28 bytes, sendo 20 bytes do cabeçalho IPV4 e 8 bytes do cabeçalho ICMP, sendo o resultado 1472 bytes.



```
tiago@tiago-Akoya-P6638:~$ ping -M do -s 1472 marco.uminho.pt
PING marco.uminho.pt (193.136.9.240) 1472(1500) bytes of data.
1480 bytes from marco.uminho.pt (193.136.9.240): icmp_seq=1 ttl=61 time=3.35 ms
1480 bytes from marco.uminho.pt (193.136.9.240): icmp_seq=2 ttl=61 time=7.34 ms
1480 bytes from marco.uminho.pt (193.136.9.240): icmp_seq=3 ttl=61 time=10.4 ms
^C
--- marco.uminho.pt ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 3.349/7.022/10.377/2.878 ms
```

Figure 14: Ping com o tamanho máximo

```

tiago@tiago-Akoya-P6638:~$ ping -M do -s 1473 marco.uminho.pt
PING marco.uminho.pt (193.136.9.240) 1473(1501) bytes of data.
ping: local error: message too long, mtu=1500
ping: local error: message too long, mtu=1500
^C
--- marco.uminho.pt ping statistics ---
2 packets transmitted, 0 received, +2 errors, 100% packet loss, time 1030ms

```

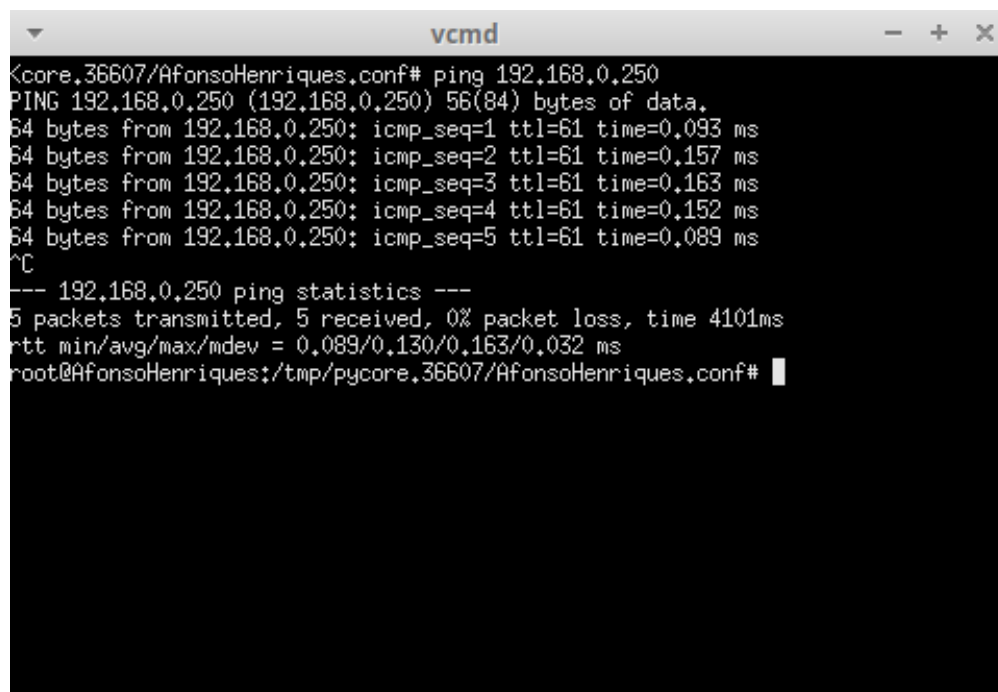
Figure 15: Acima de 1472 o ping da erro

Parte II

Pergunta 1

- 1 D.Afonso Henriques afirma ter problemas de comunicação com a sua mãe, D.Teresa. Este alega que o problema deverá estar no dispositivo de D.Teresa, uma vez que no dia anterior conseguiu enviar a sua declaração do IRS para o portal das finanças, e não tem qualquer problema em ver as suas séries favoritas disponíveis na rede de conteúdos.
- 1.a Averigue, através do comando ping, que AfonsoHenriques tem efetivamente conectividade com o servidor Financas e com os servidores da CDN.

Através do comando ping concluímos que AfonsoHenriques tem efetivamente conectividade com o servidor Financas e com os servidores da CDN.



```

vcmd
Kcore.36607/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=0.093 ms
64 bytes from 192.168.0.250: icmp_seq=2 ttl=61 time=0.157 ms
64 bytes from 192.168.0.250: icmp_seq=3 ttl=61 time=0.163 ms
64 bytes from 192.168.0.250: icmp_seq=4 ttl=61 time=0.152 ms
64 bytes from 192.168.0.250: icmp_seq=5 ttl=61 time=0.089 ms
^C
--- 192.168.0.250 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4101ms
rtt min/avg/max/mdev = 0.089/0.130/0.163/0.032 ms
root@AfonsoHenriques:/tmp/pycore.36607/AfonsoHenriques.conf#

```

Figure 16: Conexão Finanças

```
vcmd
<core.36607/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=0.163 ms
64 bytes from 192.168.0.218: icmp_seq=2 ttl=55 time=0.269 ms
64 bytes from 192.168.0.218: icmp_seq=3 ttl=55 time=0.297 ms
64 bytes from 192.168.0.218: icmp_seq=4 ttl=55 time=0.237 ms
64 bytes from 192.168.0.218: icmp_seq=5 ttl=55 time=0.262 ms
^C
--- 192.168.0.218 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4100ms
rtt min/avg/max/mdev = 0.163/0.245/0.297/0.045 ms
root@AfonsoHenriques:/tmp/pycore.36607/AfonsoHenriques.conf#
```

Figure 17: Conexão CDN

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

```
vcmd
root@AfonsoHenriques:/tmp/pycore.36607/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
root@AfonsoHenriques:/tmp/pycore.36607/AfonsoHenriques.conf#
```

Figure 18: Netstat AfonsoHenriques

```
vcmd
root@Teresa:/tmp/pycore.36607/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
root@Teresa:/tmp/pycore.36607/Teresa.conf#
```

Figure 19: Netstat Teresa

Não existem problemas com as suas entradas. A primeira entrada refere-se a uma default route, isto é, todas os endereços não conhecidos irão ser encaminhados para o gateway 192.168.0.225 (no caso do AfonsoHenriques), a segunda linha refere-se a uma rota específica para a subnet 192, o que significa que se trata da rede local 192.168.0.224 (no caso do AfonsoHenriques).

- 1.c Utilize o Wireshark para investigar o comportamento dos routers do core da rede (n1 a n6) quando tenta estabelecer comunicação entre os hosts AfonsoHenriques e Teresa. Indique que dispositivo(s) não permite(m) o encaminhamento correto dos pacotes. Seguidamente, avalie e explique a(s) causa(s) do funcionamento incorreto do dispositivo. Utilize o comando `ip route add/del` para adicionar as rotas necessárias ou remover rotas incorretas. Verifique a sintaxe completa do comando a usar com `man ip-route` ou `man route`. Poderá também utilizar o comando `traceroute` para se certificar do caminho nó a nó. Considere a alínea resolvida assim que houver tráfego a chegar ao ISP CondadOnline.

```
Kycore.45329/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) 0.084 ms 0.061 ms 0.012 ms
 2 172.16.143.1 (172.16.143.1) 0.042 ms 0.013 ms 0.012 ms
 3 10.0.0.29 (10.0.0.29) 0.041 ms !N 0.018 ms !N *
```

Figure 20: Traceroute: Afonso - Teresa

vcmd						
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

Figure 21: Tabela de encaminhamento - router n5

Utilizando o comando `traceroute` no terminal do AfonsoHenriques para a Teresa, tem-se 3 hops, sendo o último em 10.0.0.29 ao qual se refer ao router n5. Através do comando `netstat -rn` verifica-se que a destination da Teresa não se encontra na tabela, tendo sido necessário adicionar o IP no router n5.

```
route add -net 192.168.0.192 netmask 255.255.255.248 gw 10.0.0.25
```

```
<.36607/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) 0.060 ms 0.013 ms 0.011 ms
 2 172.16.143.1 (172.16.143.1) 0.030 ms 0.016 ms 0.015 ms
 3 10.0.0.29 (10.0.0.29) 0.031 ms 0.020 ms 0.020 ms
 4 10.0.0.25 (10.0.0.25) 0.045 ms 0.026 ms 0.023 ms
 5 10.0.0.25 (10.0.0.25) 3052.542 ms !H 3052.413 ms !H 3052.382 ms !H
```

Figure 22: Traceroute: Afonso - Teresa

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

Figure 23: Tabela de encaminhamento - router n2

Voltando a utilizar o comando traceroute que parou no 5º hop com a gateway 10.0.0.25 que corresponde ao router n2. Através da tabela obtida pelo comando netstat -rn no router n2, verificou-se que o destinatário 192.168.0.194 corresponde ao IP da Teresa, porém este destinatário está incorreto, uma vez que o destinatário anterior já é referente ao subnet da Teresa, sendo 192.168.0.192. O encaminhamento incorreto dos pacotes levou à necessidade de remover o IP em causa.

```
route del -net 192.168.0.194 netmask 255.255.255.254 gw 10.0.0.25
```



```

<5329/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) 0.132 ms 0.012 ms 0.009 ms
 2 172.16.143.1 (172.16.143.1) 0.020 ms 0.010 ms 0.009 ms
 3 10.0.0.29 (10.0.0.29) 0.021 ms 0.013 ms 0.012 ms
 4 10.0.0.25 (10.0.0.25) 0.043 ms 0.016 ms 0.015 ms
 5 10.0.0.13 (10.0.0.13) 0.047 ms 0.018 ms 0.020 ms
 6 10.0.0.25 (10.0.0.25) 0.022 ms 0.052 ms 0.017 ms
 7 10.0.0.13 (10.0.0.13) 0.017 ms 0.016 ms 0.016 ms
 8 * * *
 9 * * *
10 * * *
11 * * *
12 * * *
13 * 10.0.0.13 (10.0.0.13) 0.070 ms 0.026 ms
14 10.0.0.25 (10.0.0.25) 0.026 ms 0.022 ms 0.022 ms
15 10.0.0.13 (10.0.0.13) 0.024 ms 0.023 ms 0.024 ms
16 10.0.0.25 (10.0.0.25) 0.023 ms 0.025 ms *
17 * * *
18 * * *
19 * * *
20 * * *
21 * * *
22 * 10.0.0.25 (10.0.0.25) 0.096 ms 0.047 ms
23 10.0.0.13 (10.0.0.13) 0.049 ms 0.046 ms 0.047 ms
24 10.0.0.25 (10.0.0.25) 0.046 ms 0.048 ms 0.081 ms
25 10.0.0.13 (10.0.0.13) 0.052 ms 0.048 ms *
26 * * *
27 * * *
28 * * *
29 * * *
30 * * *

```

Figure 24: Traceroute: Afonso - Teresa

```
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

Figure 25: Tabela de endereçamento - router n1

Através do comando traceroute, verificamos um loop no que toca ao envio de tráfego entre o router n1 e o n2 , isto é, o tráfego enviado volta para trás. De acordo com a tabela obtida pelo comando netstat -rn no router n1, temos que o tráfego do destinatário 192.168.0.192 está a ser enviado para trás, o que leva à necessidade de remover o IP e adicionar um IP mas desta vez com a gateway correta.

```
route del -net 192.168.0.192 netmask 255.255.255.248 gw 10.0.0.14
```

```
route add -net 192.168.0.192 netmask 255.255.255.248 gw 10.0.0.9
```

```

<.36607/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) 0.053 ms 0.009 ms 0.007 ms
 2 172.16.143.1 (172.16.143.1) 0.021 ms 0.013 ms 0.019 ms
 3 10.0.0.29 (10.0.0.29) 0.024 ms 0.015 ms 0.015 ms
 4 10.0.0.25 (10.0.0.25) 0.034 ms 0.018 ms 0.019 ms
 5 10.0.0.13 (10.0.0.13) 0.040 ms 0.023 ms 0.023 ms
 6 10.0.0.17 (10.0.0.17) 0.078 ms 0.109 ms 0.031 ms
 7 10.0.0.5 (10.0.0.5) 0.087 ms 0.045 ms 0.032 ms
 8 10.0.0.1 (10.0.0.1) 0.050 ms 0.035 ms 0.033 ms
 9 * * *
10 * * *
11 * * *
12 * * *
13 * * *
14 * * *
15 * * *
16 * * *
17 * * *
18 * * *
19 * * *
20 * * *
21 * * *
22 * * *
23 * * *
24 * * *
25 * * *
26 * * *
27 * * *
28 * * *
29 * * *
30 * * *

```

Figure 26: Traceroute: Afonso - Teresa

Através do comando traceroute, verifica-se que é enviado tráfego até 10.0.0.1, ou seja, até o CondaOnline.

1.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.

1.d.1 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.

Não existe conectividade entre o Afonso e a Teresa, uma vez que os pacotes são entregues do Afonso à Teresa, contudo o Afonso não obtém nenhuma resposta, isto é, não existe conectividade bi-direcional.

```

root@RAGaliza:/tmp/pycore.45329/RAGaliza.conf# netstat -rn
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

```

Figure 27: Tabela de endereçamento -Galiza

Verifica-se que a Galiza não possui conectividade com o Afonso, pois não possui uma rota para a sua rede, havendo a necessidade de adicioná-la, através do seguinte comando:

```
route add -net 192.168.224 netmask 255.255.255.248 gw 172.16.142.1
```

Assim, a conectividade é restabelecida e o confronto entre os dois é evitado.

1.d.2 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.

Não, como se pode verificar, os pacotes enviados da Teresa ao Afonso passam pelo router n1, enquanto os pacotes enviados do Afonso à Teresa, passam pelo router n4.

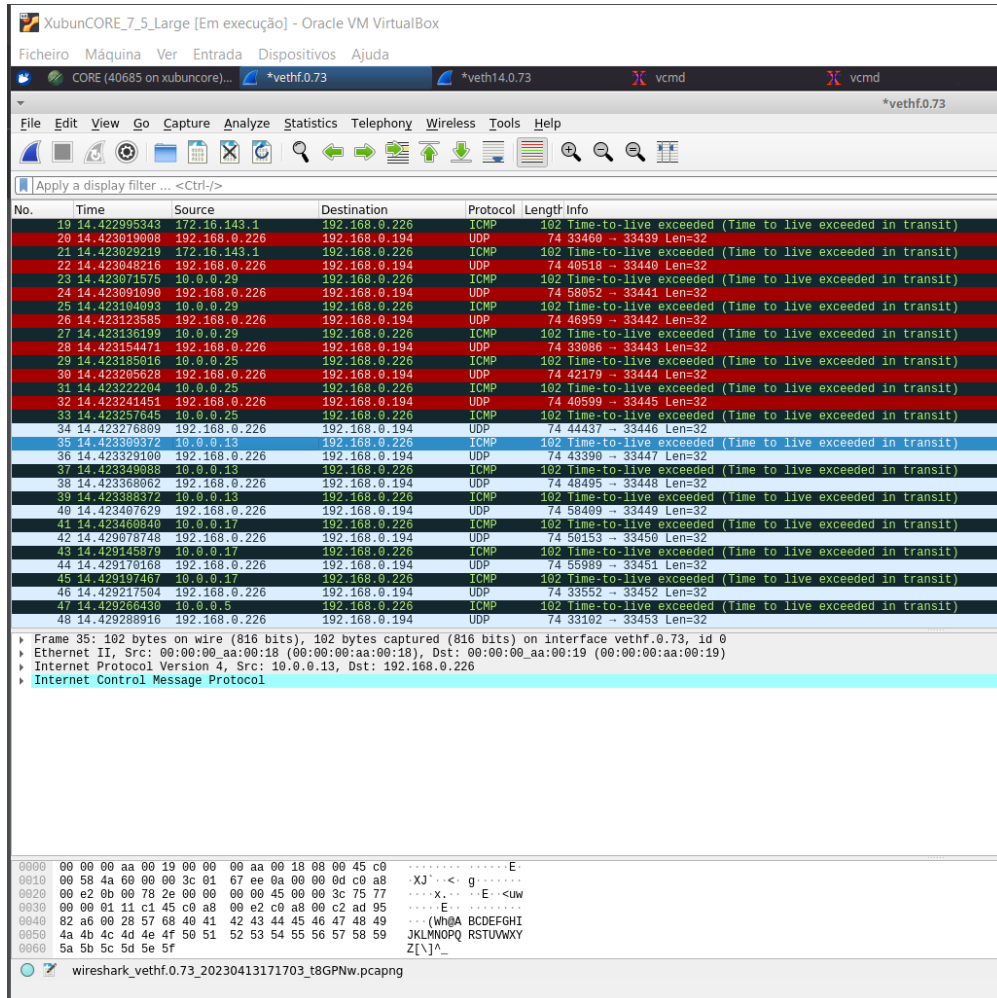


Figure 28: Traceroute Afonso - Teresa

Passa pelo 10.0.0.13 (router n1)

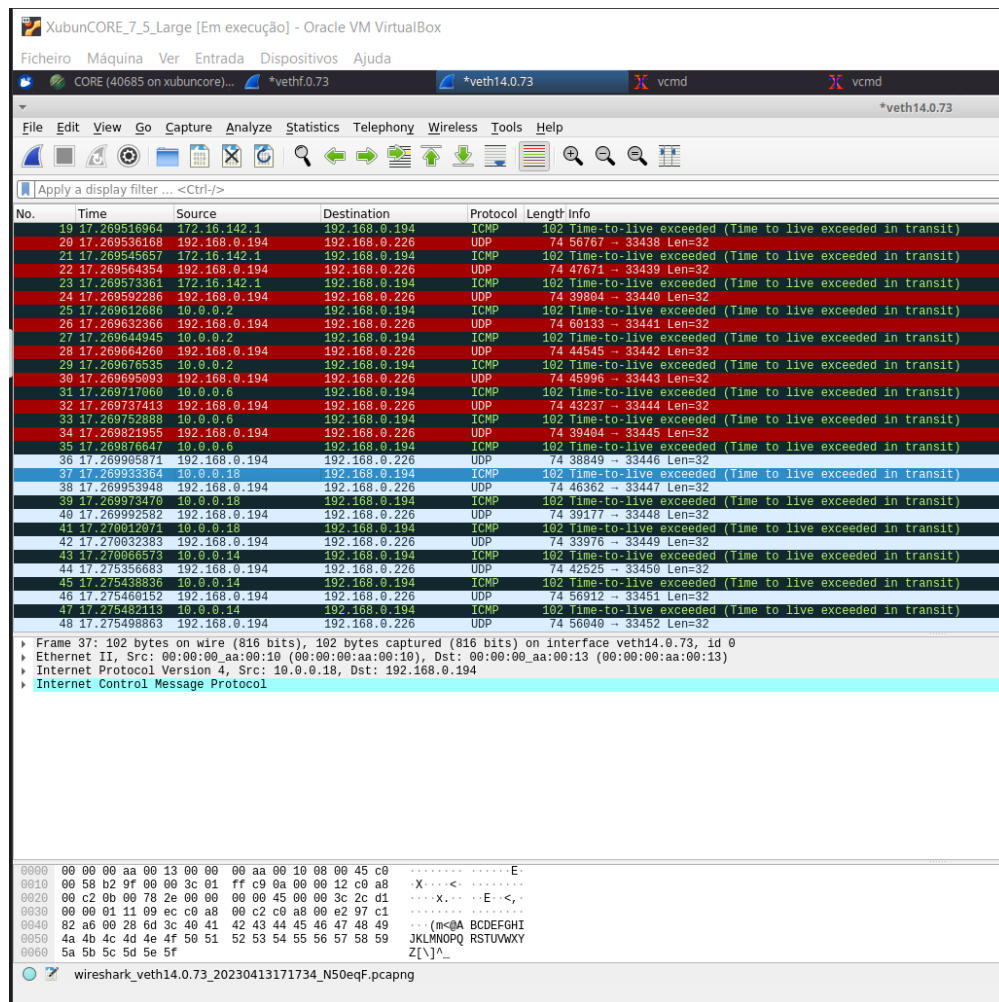


Figure 29: Traceroute Teresa - Afonso

Passa pelo 10.0.0.18 (router n4)

- 1.e Estando restabelecida a conectividade entre os dois hosts, obtenha a tabela de encaminhamento de n3 e foque-se na seguinte entrada. 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? Ofereça uma explicação pela qual essa entrada é ou não utilizada.

A entrada esta errada mas n3 continua a encaminhar os pacotes corretamente para Galiza. Isto é devido a n3 ter outra entrada com o mesmo destino mas com um ip menor (10.0.0.5) que é o encaminhamento correto. Por default é escolhido o ip menor, o que neste caso em especifico é o correto, mas numa situação em que os ips corretos fossem os contrários ele daria erro.

Para pacotes enviados para CDN não há correspondencia na entrada

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

Os endereços utilizados pelos quatro polos são endereços privados, bem como os endereços utilizados no core da rede/ISPs. Esses endereços IP são privados, pois o endereço 192.168.0.0 é privado de classe C e o endereço 10.0.0.0 é privado de classe A.

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

Os switches localizados em cada um dos polos não têm um endereço IP atribuído, uma vez que operam na 2ª camada. Por outro lado, o IP é protocolo da 3ª camada, por este motivo, os switches não precisam de um endereço IP, pois não precisam de se comunicar com outros dispositivos fora da rede local.

Pergunta 2

2 Tendo feito as pazes com a mãe, D. Afonso Henriques vê-se com algum tempo livre e decide fazer remodelações no condado:

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. Explícite ainda a utilidade de uma rota default.

Eliminamos a rota default através do comando `route del default`. De modo que o Castelo continue a ter acesso a cada um dos três polos, houve a necessidade de adicionar as seguintes rotas:

```
route add -net 192.168.0.192 netmask 255.255.255.248 gw 192.168.0.225
route add -net 192.168.0.200 netmask 255.255.255.248 gw 192.168.0.225
route add -net 192.168.0.208 netmask 255.255.255.248 gw 192.168.0.225
route add -net 192.168.0.216 netmask 255.255.255.248 gw 192.168.0.225
route add -net 192.168.0.232 netmask 255.255.255.248 gw 192.168.0.225
route add -net 192.168.0.240 netmask 255.255.255.248 gw 192.168.0.225
route add -net 192.168.0.248 netmask 255.255.255.248 gw 192.168.0.225
```

Assim, a conectividade é restabelecida como se pode verificar nas seguintes figuras. Verificamos que o Castelo tem conectividade com o RAIInstitucional

```
root@Castelo:/tmp/pycore.44849/Castelo.conf# traceroute 192.168.0.249
traceroute to 192.168.0.249 (192.168.0.249), 30 hops max, 60 byte packets
 1 192.168.0.225 (192.168.0.225) 0.035 ms 0.006 ms 0.005 ms
 2 * * *
 3 192.168.0.249 (192.168.0.249) 0.026 ms 0.009 ms 0.009 ms
```

Figure 30: Castelo - RAIInstitucional

Verificamos que o Castelo tem conectividade com o RACDN

```
traceroute to 192.168.0.201 (192.168.0.201), 30 hops max, 60 byte packets
 1 192.168.0.225 (192.168.0.225) 0.031 ms 0.015 ms 0.008 ms
 2 * * *
 3 * * *
 4 * * *
 5 * * *
 6 * * *
 7 * * *
 8 * * *
 9 192.168.0.201 (192.168.0.201) 0.053 ms 0.029 ms 0.029 ms
```

Figure 31: Castelo - RACDN

Verificamos que o Castelo tem conectividade com o RAGaliza

```
root@Castelo:/tmp/pycore.44849/Castelo.conf# traceroute 192.168.0.193
traceroute to 192.168.0.193 (192.168.0.193), 30 hops max, 60 byte packets
 1 192.168.0.225 (192.168.0.225) 0.031 ms 0.005 ms 0.005 ms
 2 * * *
 3 * * *
 4 * * *
 5 * * *
 6 * * *
 7 * * *
 8 * * *
 9 192.168.0.193 (192.168.0.193) 0.040 ms 0.030 ms 0.029 ms
```

Figure 32: Castelo - RAGaliza

Utilidade de uma rota default:

Uma rota default é uma rota usada por um router para encaminhar pacotes para um destino desconhecido ou para o qual não há uma rota específica definida. Na ausência de uma rota default, não é possível enviar esses pacotes para destinos que não se encontram na tabela de encaminhamento.

- 2.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.

Esquema de endereçamento:

Sub-rede 1:

Endereço de sub-rede: 172.16.52.128/27

Endereços de host utilizáveis: 172.16.52.129 - 172.16.52.158

Sub-rede 2:

Endereço de sub-rede: 172.16.52.160/27

Endereços de host utilizáveis: 172.16.52.161 - 172.16.52.190

Sub-rede 3:

Endereço de sub-rede: 172.16.52.192/27

Endereços de host utilizáveis: 172.16.52.193 - 172.16.52.222

Neste esquema de endereçamento, cada sub-rede possui uma máscara de sub-rede /27, que permite o uso de 30 endereços de host. Dessa forma, todas as três redes possuem pelo menos 10 endereços de host disponíveis.

- 2.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?

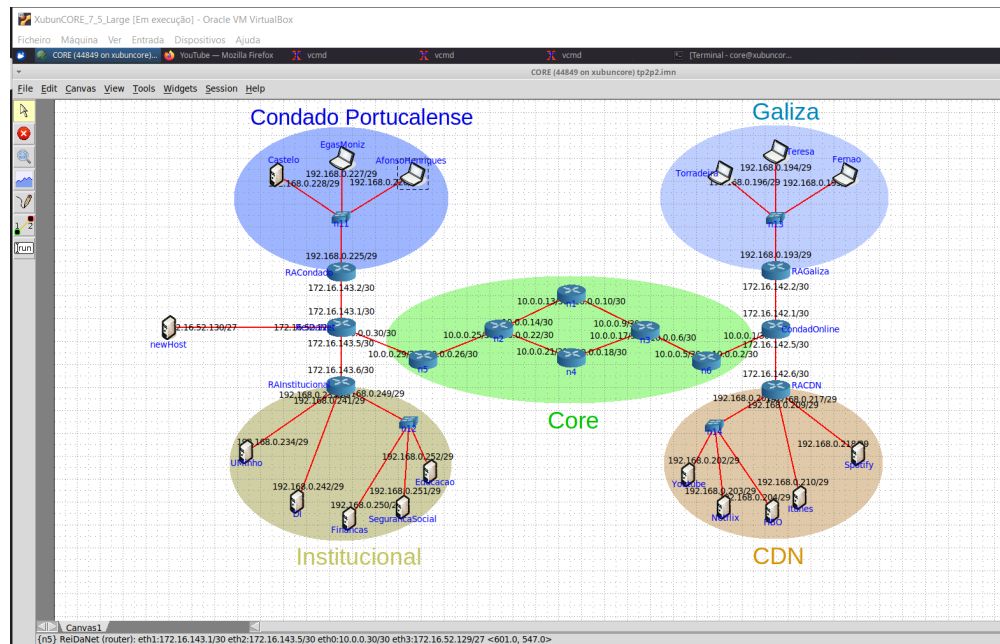


Figure 33: Novo host

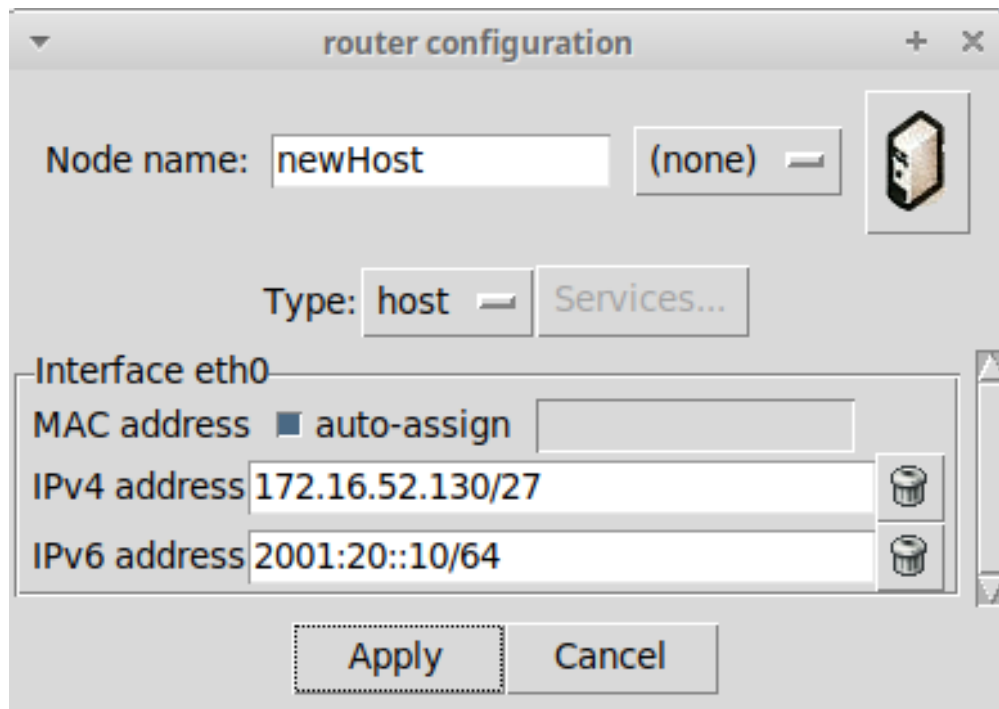


Figure 34: Settings do novo host

Como se pode verificar através da seguinte figura, o novo host adicionado tem conectividade com todos os polos:

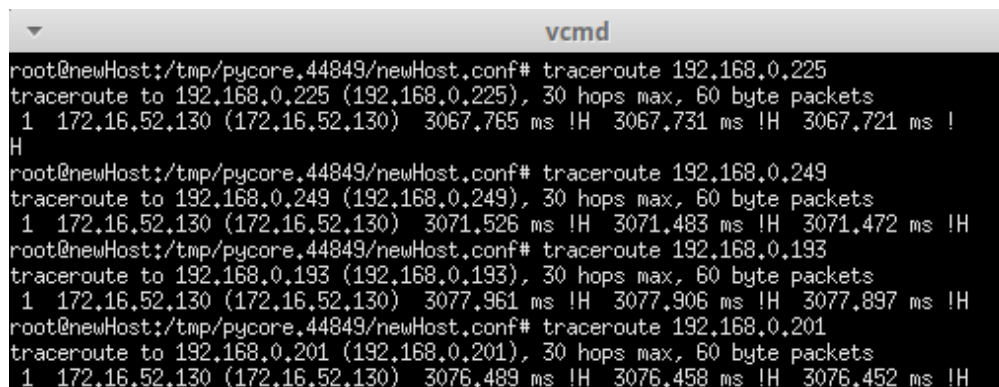


Figure 35: Conexões do novo host

Contudo, não é possível se comunicar com o Castelo, como se pode verificar através da seguinte figura:

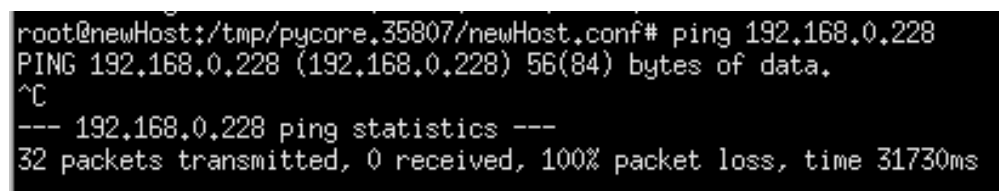


Figure 36: Novo host - Castelo

Pergunta 3

- 3** Ao planejar um novo ataque, D. Afonso Henriques constata que o seu exército não só perde bastante tempo a decidir que direção tomar a cada salto como, por vezes, inclusivamente se perde.
- 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.

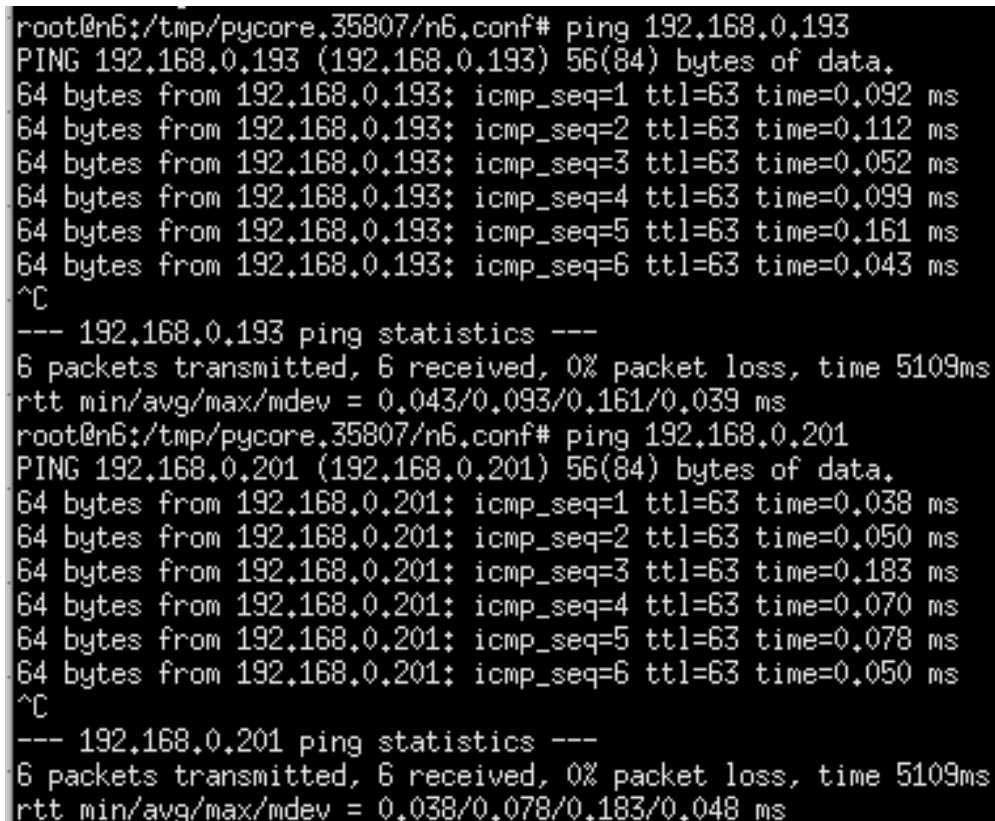
De modo a facilitar a travessia, foi eliminada as rotas referentes a Galiza e CDN no dispositivo n6, através dos seguintes comandos:

```
route del -net 192.168.0.192 netmask 255.255.255.248 gw 10.0.0.1
route del -net 192.168.0.200 netmask 255.255.255.248 gw 10.0.0.1
```

Foi definido um esquema de sumarização de rotas (Supernetting) que permite o uso de apenas uma rota para ambos os polos, tendo sido adicionado a seguinte rota:

```
route add -net 192.168.0.192/28 gw 10.0.0.1
```

Assim, podemos comprovar através da seguinte figura, que a conectividade é mantida



```
root@n6:/tmp/pycore.35807/n6.conf# ping 192.168.0.193
PING 192.168.0.193 (192.168.0.193) 56(84) bytes of data.
64 bytes from 192.168.0.193: icmp_seq=1 ttl=63 time=0.092 ms
64 bytes from 192.168.0.193: icmp_seq=2 ttl=63 time=0.112 ms
64 bytes from 192.168.0.193: icmp_seq=3 ttl=63 time=0.052 ms
64 bytes from 192.168.0.193: icmp_seq=4 ttl=63 time=0.099 ms
64 bytes from 192.168.0.193: icmp_seq=5 ttl=63 time=0.161 ms
64 bytes from 192.168.0.193: icmp_seq=6 ttl=63 time=0.043 ms
^C
--- 192.168.0.193 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5109ms
rtt min/avg/max/mdev = 0.043/0.093/0.161/0.039 ms
root@n6:/tmp/pycore.35807/n6.conf# ping 192.168.0.201
PING 192.168.0.201 (192.168.0.201) 56(84) bytes of data.
64 bytes from 192.168.0.201: icmp_seq=1 ttl=63 time=0.038 ms
64 bytes from 192.168.0.201: icmp_seq=2 ttl=63 time=0.050 ms
64 bytes from 192.168.0.201: icmp_seq=3 ttl=63 time=0.183 ms
64 bytes from 192.168.0.201: icmp_seq=4 ttl=63 time=0.070 ms
64 bytes from 192.168.0.201: icmp_seq=5 ttl=63 time=0.078 ms
64 bytes from 192.168.0.201: icmp_seq=6 ttl=63 time=0.050 ms
^C
--- 192.168.0.201 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5109ms
rtt min/avg/max/mdev = 0.038/0.078/0.183/0.048 ms
```

Figure 37: Conectividade mantida

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

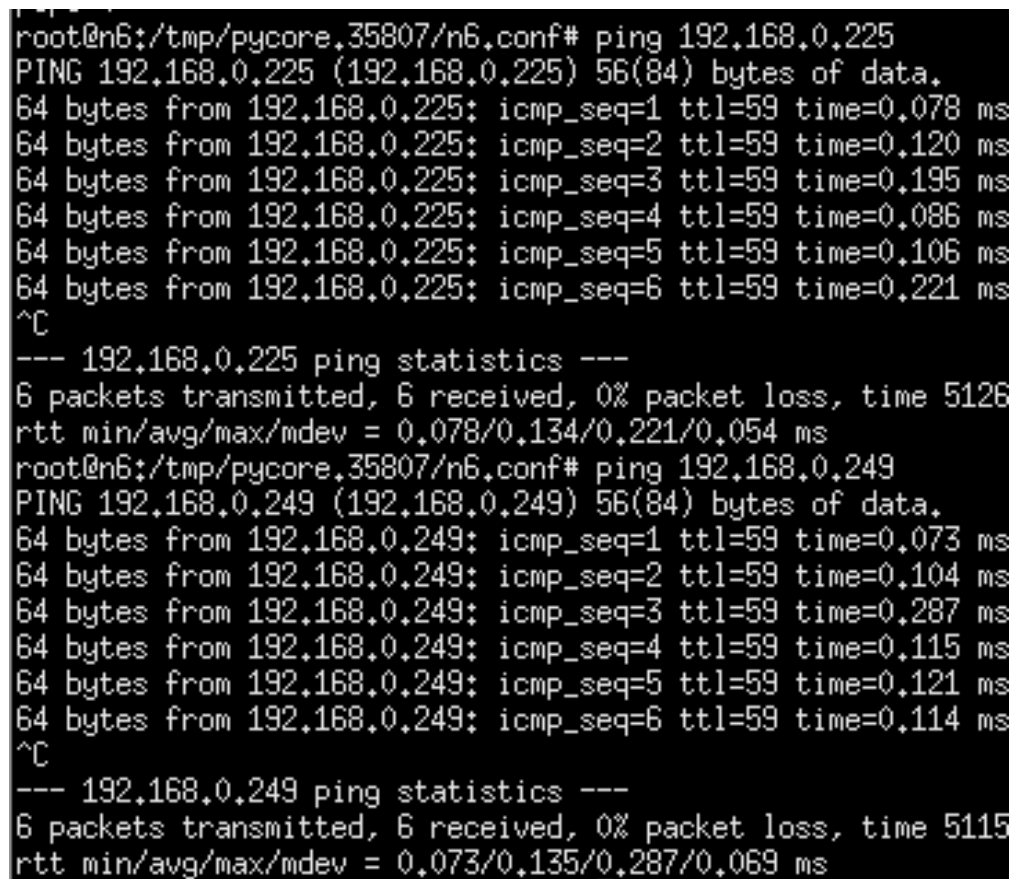
Foram eliminadas as rotas referentes ao CondadoPortucalense e Institucional, através dos seguintes comandos:

```
route del -net 192.168.0.224 netmask 255.255.255.248 gw 10.0.0.6
route del -net 192.168.0.248 netmask 255.255.255.248 gw 10.0.0.6
```

Tendo sido definido o seguinte esquema de sumarização de rotas que irá permitir o uso de apenas uma rota para os polos em questão:

```
route add -net 192.168.0.224/27 gw 10.0.0.6
```

Assim, podemos comprovar através da seguinte figura, que a conectividade é mantida:



```
root@n6:/tmp/pycore.35807/n6.conf# ping 192.168.0.225
PING 192.168.0.225 (192.168.0.225) 56(84) bytes of data.
64 bytes from 192.168.0.225: icmp_seq=1 ttl=59 time=0.078 ms
64 bytes from 192.168.0.225: icmp_seq=2 ttl=59 time=0.120 ms
64 bytes from 192.168.0.225: icmp_seq=3 ttl=59 time=0.195 ms
64 bytes from 192.168.0.225: icmp_seq=4 ttl=59 time=0.086 ms
64 bytes from 192.168.0.225: icmp_seq=5 ttl=59 time=0.106 ms
64 bytes from 192.168.0.225: icmp_seq=6 ttl=59 time=0.221 ms
^C
--- 192.168.0.225 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5126
rtt min/avg/max/mdev = 0.078/0.134/0.221/0.054 ms
root@n6:/tmp/pycore.35807/n6.conf# ping 192.168.0.249
PING 192.168.0.249 (192.168.0.249) 56(84) bytes of data.
64 bytes from 192.168.0.249: icmp_seq=1 ttl=59 time=0.073 ms
64 bytes from 192.168.0.249: icmp_seq=2 ttl=59 time=0.104 ms
64 bytes from 192.168.0.249: icmp_seq=3 ttl=59 time=0.287 ms
64 bytes from 192.168.0.249: icmp_seq=4 ttl=59 time=0.115 ms
64 bytes from 192.168.0.249: icmp_seq=5 ttl=59 time=0.121 ms
64 bytes from 192.168.0.249: icmp_seq=6 ttl=59 time=0.114 ms
^C
--- 192.168.0.249 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5115
rtt min/avg/max/mdev = 0.073/0.135/0.287/0.069 ms
```

Figure 38: Conectividade mantida

3.c Comente os aspetos positivos e negativos do uso do Supernetting.

Aspetos positivos e negativos do supernetting:

O Supernetting, também conhecido como super-rede, agregação de rotas ou resumo de rotas, é uma técnica utilizada para reduzir o tamanho da tabela de roteamento. Embora o Supernetting possa trazer pontos positivos para a rede, também pode apresentar algumas desvantagens.

Aspectos positivos:

Redução do tamanho da tabela de roteamento: a agregação de várias rotas em uma única rota permite reduzir o tamanho da tabela de roteamento, tornando a rede mais eficiente.

Redução do tráfego na rede: como há menos rotas na tabela de roteamento, há menos tráfego de roteamento na rede, o que pode melhorar o desempenho.

Maior segurança: a agregação de rotas pode ajudar a reduzir o risco de ataques de negação de serviço por meio da redução do número de rotas que precisam ser processadas pelos dispositivos de rede.

Aspectos negativos:

Perda de flexibilidade: a agregação de rotas pode limitar a flexibilidade da rede, tornando mais difícil adicionar ou remover redes sem afetar outras partes da rede.

Perda de redundância: a agregação de rotas pode tornar a rede menos redundante, pois a falha de uma rota pode afetar várias redes, em vez de apenas uma.

Problemas de escalabilidade: a agregação de rotas pode tornar a rede menos escalável, pois a tabela de roteamento pode crescer rapidamente à medida que mais redes são adicionadas.

Possível aumento da latência: a agregação de rotas pode levar a um aumento da latência devido ao aumento da quantidade de tráfego na rede.

Em resumo, o Supernetting pode trazer benefícios para a rede, como a redução do tamanho da tabela de encaminhamento e a redução do tráfego na rede, mas também pode apresentar desvantagens, como a perda de flexibilidade, a perda de redundância e possíveis problemas de escalabilidade e latência.