

Trabalho Prático Nº2

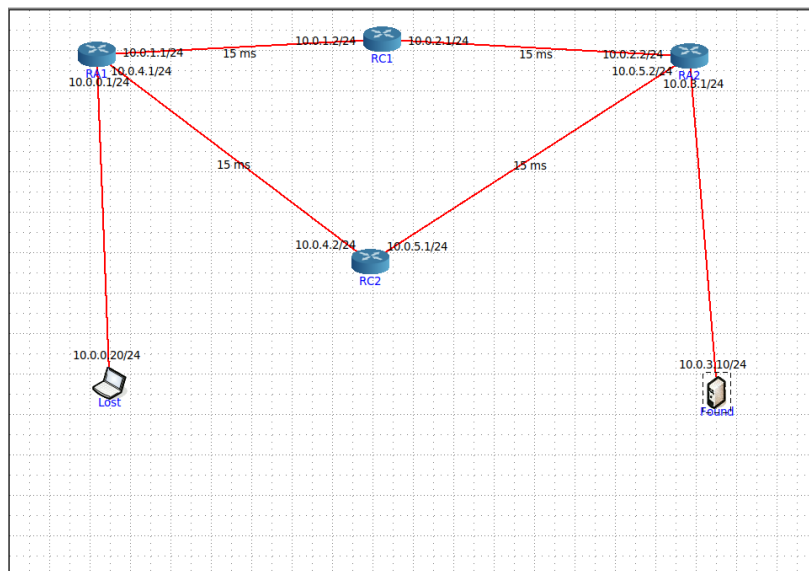
Protocolo IPv4 :: Datagramas IP e Fragmentação (1ª Parte)

Rafael Peixoto ^[96807], João Lopes ^[100829], João Vale ^[100697]

Universidade do Minho

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. 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.



O traceroute é um comando que, a partir do pc Lost pretende chegar ao servidor Found enviando 3 datagramas de cada vez com tamanhos diferentes. No início o TTL é igual a 1, então quando chega ao primeiro router RA1 ele descarta o datagramas pois ainda não chegou ao Found e o seu TTL esta igual a 0. Assim são enviadas três mensagens de controlo ICMP com TTL exceeded. Logo de seguida envia mais 3 datagramas com TTL é igual a 2 e ira fazer dois saltos ate o router RC1 e o TTL desses datagramas ficam a 0. Assim também são enviadas três

mensagens de controlo ICMP com TTL exceeded. O mesmo acontece com o TTL é igual a 3, chega até o router RA2. Quando o TTL é igual a 4 ele chega ao Found e assim envia três respostas.

No.	Time	Source	Destination	Protocol	Length	Info
3	2.449424956	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=1/256, ttl=1 (no response found!)
4	2.449447762	10.0.0.1	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
5	2.449455737	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=2/512, ttl=1 (no response found!)
6	2.449459995	10.0.0.1	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
7	2.449463118	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=3/768, ttl=1 (no response found!)
8	2.449466240	10.0.0.1	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
9	2.449470135	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=4/1024, ttl=2 (no response found!)
10	2.449470970	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=5/1280, ttl=2 (no response found!)
11	2.449483931	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=6/1536, ttl=2 (no response found!)
12	2.449487081	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=7/1792, ttl=3 (no response found!)
13	2.449490446	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=8/2048, ttl=3 (no response found!)
14	2.449494531	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=9/2304, ttl=3 (no response found!)
15	2.449498708	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=10/2560, ttl=4 (reply in 34)
16	2.449502969	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=11/2816, ttl=4 (reply in 35)
17	2.449506738	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=12/3072, ttl=4 (reply in 36)
18	2.449510841	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=13/3328, ttl=5 (reply in 37)
19	2.449514436	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=14/3584, ttl=5 (reply in 38)
20	2.449517807	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=15/3840, ttl=5 (reply in 39)
21	2.449522241	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=16/4096, ttl=6 (reply in 40)
22	2.450368920	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=17/4352, ttl=6 (reply in 41)
23	2.45037435	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=18/4608, ttl=6 (reply in 42)
24	2.450393080	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=19/4864, ttl=7 (reply in 43)
25	2.450395301	10.0.0.20	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
26	2.485656805	10.0.1.2	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
27	2.485657549	10.0.1.2	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
28	2.486218282	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=20/5120, ttl=7 (reply in 44)
29	2.486232900	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=21/5376, ttl=7 (reply in 45)
30	2.486238916	10.0.0.20	10.0.3.10	ICMP	74	Echo (ping) request id=0x0036, seq=22/5632, ttl=8 (reply in 46)
31	2.518534398	10.0.2.2	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
32	2.518541632	10.0.2.2	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
33	2.518543045	10.0.2.2	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
34	2.518544348	10.0.3.10	10.0.0.20	ICMP	74	Echo (ping) reply id=0x0036, seq=10/2560, ttl=61 (request in 15)
35	2.518545627	10.0.3.10	10.0.0.20	ICMP	74	Echo (ping) reply id=0x0036, seq=11/2816, ttl=61 (request in 16)
36	2.518546907	10.0.3.10	10.0.0.20	ICMP	74	Echo (ping) reply id=0x0036, seq=12/3072, ttl=61 (request in 17)
37	2.518548196	10.0.3.10	10.0.0.20	ICMP	74	Echo (ping) reply id=0x0036, seq=13/3328, ttl=61 (request in 18)
38	2.518549471	10.0.3.10	10.0.0.20	ICMP	74	Echo (ping) reply id=0x0036, seq=14/3584, ttl=61 (request in 19)
39	2.518550762	10.0.3.10	10.0.0.20	ICMP	74	Echo (ping) reply id=0x0036, seq=15/3840, ttl=61 (request in 20)
40	2.518552045	10.0.3.10	10.0.0.20	ICMP	74	Echo (ping) reply id=0x0036, seq=16/4096, ttl=61 (request in 21)
41	2.518553337	10.0.3.10	10.0.0.20	ICMP	74	Echo (ping) reply id=0x0036, seq=17/4352, ttl=61 (request in 22)
42	2.518554622	10.0.3.10	10.0.0.20	ICMP	74	Echo (ping) reply id=0x0036, seq=18/4608, ttl=61 (request in 23)
43	2.518555913	10.0.3.10	10.0.0.20	ICMP	74	Echo (ping) reply id=0x0036, seq=19/4864, ttl=61 (request in 24)
44	2.551207056	10.0.3.10	10.0.0.20	ICMP	74	Echo (ping) reply id=0x0036, seq=20/5120, ttl=61 (request in 28)
45	2.551216431	10.0.3.10	10.0.0.20	ICMP	74	Echo (ping) reply id=0x0036, seq=21/5376, ttl=61 (request in 29)
46	2.551218124	10.0.3.10	10.0.0.20	ICMP	74	Echo (ping) reply id=0x0036, seq=22/5632, ttl=61 (request in 30)

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.

O TTL tem de ser no mínimo 4 para alcançar o servidor Found, isto verifica-se na imagem do wireshark acima, onde se obtém a primeira resposta na linha 15. Isto mantém-se verifica com execuções anteriores e está de acordo com o diagrama na imagem do core.

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.35621/Lost.conf# traceroute -I 10.0.3.10 -q 3
traceroute to 10.0.3.10 (10.0.3.10), 30 hops max, 60 byte packets
 1 10.0.0.1 (10.0.0.1) 0.033 ms 0.005 ms 0.004 ms
 2 10.0.1.2 (10.0.1.2) 40.943 ms 40.934 ms 40.931 ms
 3 10.0.2.2 (10.0.2.2) 72.852 ms 72.850 ms 72.847 ms
 4 10.0.3.10 (10.0.3.10) 72.844 ms 72.841 ms 72.839 ms
root@Lost:/tmp/pycore.35621/Lost.conf# traceroute -I 10.0.3.10 -q 3
traceroute to 10.0.3.10 (10.0.3.10), 30 hops max, 60 byte packets
 1 10.0.0.1 (10.0.0.1) 0.032 ms 0.007 ms 0.005 ms
 2 10.0.1.2 (10.0.1.2) 32.183 ms 32.176 ms 32.172 ms
 3 10.0.2.2 (10.0.2.2) 66.590 ms 66.589 ms 66.586 ms
 4 10.0.3.10 (10.0.3.10) 66.583 ms 66.579 ms 66.574 ms
root@Lost:/tmp/pycore.35621/Lost.conf# traceroute -I 10.0.3.10 -q 3
traceroute to 10.0.3.10 (10.0.3.10), 30 hops max, 60 byte packets
 1 10.0.0.1 (10.0.0.1) 0.032 ms 0.005 ms 0.004 ms
 2 10.0.1.2 (10.0.1.2) 31.095 ms 31.088 ms 31.084 ms
 3 10.0.2.2 (10.0.2.2) 64.023 ms 64.020 ms 64.018 ms
 4 10.0.3.10 (10.0.3.10) 64.014 ms 64.012 ms 64.010 ms
root@Lost:/tmp/pycore.35621/Lost.conf#

```

Os valores médios do RTT no acesso ao servidor para as tentativas na imagem são 72.84ms, 66.58ms e 64.012ms respetivamente. Sendo assim, o RTT tem um valor médio de 67.81ms no total.

O atraso num sentido na topologia CORE é de 30ms devido aos 15ms de atraso em cada ligação da rede de core. O RTT dividido por 2 é ligeiramente superior a 30ms, mas como o valor do RTT pode variar, a precisão nunca será muito alta, o que faz com que seja bastante difícil obter um valor concreto para o One-Way Delay ao dividir o RTT por dois. Sendo que numa rede real o delay é variado, esta tarefa torna-se muito mais complicada.

[illegible]

IP: 172.26.35.254

O valor do protocolo é ICMP (1) e permite identificar que é uma mensagem do tipo “Echo Request” (Type 8).

O cabeçalho tem 20 bytes, o tamanho total é 512 bytes. O payload calcula-se subtraindo o tamanho total pelo cabeçalho, 492 bytes.

D. O datagrama IP foi fragmentado? Justifique.

```
✓ 000. .... = Flags: 0x0
  0... .... = Reserved bit: Not set
  .0.. .... = Don't fragment: Not set
  ..0. .... = More fragments: Not set
  ...0 0000 0000 0000 = Fragment Offset: 0
```

Não foi fragmentado pois o “Fragment offset” tem valor 0 e podemos observar também que não há mais fragmentos em “More fragments”.

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.

```
594 41.749646 172.16.115.252 172.26.35.254 ICMP 70 Time-to-live exceeded (Time to live exceeded in transit)
> Frame 594: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface en0, id 0
> Ethernet II, Src: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00), Dst: Apple_24:cd:3a (c0:95:6d:24:cd:3a)
> Internet Protocol Version 4, Src: 172.16.115.252, Dst: 172.26.35.254
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 56
  Identification: 0x6b15 (27413)
  000. .... = Flags: 0x0
  ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 253
  Protocol: ICMP (1)
  Header Checksum: 0x628a [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 172.16.115.252
  Destination Address: 172.26.35.254
  ✓ Internet Control Message Protocol
    Type: 11 (Time-to-live exceeded)
    Code: 0 (Time to live exceeded in transit)
    Checksum: 0xf4ff [correct]
    [Checksum Status: Good]
    Unused: 00000000
  > Internet Protocol Version 4, Src: 172.26.35.254, Dst: 193.136.9.240
  ✓ Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0x152f [unverified] [in ICMP error packet]
    [Checksum Status: Unverified]
    Identifier (BE): 58055 (0xe2c7)
    Identifier (LE): 51170 (0xc7e2)
    Sequence Number (BE): 9 (0x0009)
    Sequence Number (LE): 2304 (0x0900)
```

```
559 41.575802 172.26.254.254 172.26.35.254 ICMP 70 Time-to-live exceeded (Time to live exceeded in transit)
> Frame 559: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface en0, id 0
> Ethernet II, Src: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00), Dst: Apple_24:cd:3a (c0:95:6d:24:cd:3a)
> Internet Protocol Version 4, Src: 172.26.254.254, Dst: 172.26.35.254
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0xc0 (DSCP: CS6, ECN: Not-ECT)
  Total Length: 56
  Identification: 0x9783 (38787)
  000. .... = Flags: 0x0
  ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 255
  Protocol: ICMP (1)
  Header Checksum: 0xa84f [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 172.26.254.254
  Destination Address: 172.26.35.254
  ✓ Internet Control Message Protocol
    Type: 11 (Time-to-live exceeded)
    Code: 0 (Time to live exceeded in transit)
    Checksum: 0xf4ff [correct]
    [Checksum Status: Good]
    Unused: 00000000
  > Internet Protocol Version 4, Src: 172.26.35.254, Dst: 193.136.9.240
  > Internet Control Message Protocol
```

Alterações observadas:

- Time to live;
- Identification;
- Header Checksum.

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

558	41.572332	172.26.35.254	193.136.9.240	ICMP	526	Echo (ping) request	id=0xe2c7, seq=2/512, ttl=1 (no response found!)
560	41.575961	172.26.35.254	193.136.9.240	ICMP	526	Echo (ping) request	id=0xe2c7, seq=3/768, ttl=1 (no response found!)
562	41.579491	172.26.35.254	193.136.9.240	ICMP	526	Echo (ping) request	id=0xe2c7, seq=4/1024, ttl=2 (no response found!)
574	41.655916	172.26.35.254	193.136.9.240	ICMP	526	Echo (ping) request	id=0xe2c7, seq=5/1280, ttl=2 (no response found!)
576	41.661692	172.26.35.254	193.136.9.240	ICMP	526	Echo (ping) request	id=0xe2c7, seq=6/1536, ttl=2 (no response found!)
579	41.664435	172.26.35.254	193.136.9.240	ICMP	526	Echo (ping) request	id=0xe2c7, seq=7/1792, ttl=3 (no response found!)
591	41.740662	172.26.35.254	193.136.9.240	ICMP	526	Echo (ping) request	id=0xe2c7, seq=8/2048, ttl=3 (no response found!)
593	41.745439	172.26.35.254	193.136.9.240	ICMP	526	Echo (ping) request	id=0xe2c7, seq=9/2304, ttl=3 (no response found!)
595	41.749759	172.26.35.254	193.136.9.240	ICMP	526	Echo (ping) request	id=0xe2c7, seq=10/2560, ttl=4 (reply in 596)
603	42.115110	172.26.35.254	193.136.9.240	ICMP	526	Echo (ping) request	id=0xe2c7, seq=11/2816, ttl=4 (reply in 604)
605	42.119404	172.26.35.254	193.136.9.240	ICMP	526	Echo (ping) request	id=0xe2c7, seq=12/3072, ttl=4 (reply in 606)

O campo de identificação do datagrama IP aumenta sempre 1 a cada datagrama e o TTL aumenta 1 a cada 3 datagramas.

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?

580	41.668630	172.16.115.252	172.26.35.254	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
592	41.745314	172.16.115.252	172.26.35.254	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
594	41.749646	172.16.115.252	172.26.35.254	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
563	41.582992	172.16.2.1	172.26.35.254	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
575	41.661335	172.16.2.1	172.26.35.254	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
578	41.664190	172.16.2.1	172.26.35.254	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
553	41.513111	172.26.254.254	172.26.35.254	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
559	41.575802	172.26.254.254	172.26.35.254	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
561	41.579369	172.26.254.254	172.26.35.254	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)

O campo TTL tem os valores 253, 254 e 255 e não se mantém constante. A cada router que passa, o TTL é decrementado por 1.

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

As mensagens são enviadas com um valor alto pois evita que as mensagens de resposta sejam descartas antes de chegar à origem.

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?

O cabeçalho do ICMP pode ser incluído no do IPV4, trazendo vantagens e desvantagens. A principal vantagem é a redução do overhead de cabeçalho, o que diminui a quantidade de informação necessária para enviar os pacotes, já a principal desvantagem é a complexidade que poderia trazer ao cabeçalho do IPV4, causando maior dificuldade de leitura e a problemas de desempenho e segurança.

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 3525.

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

```
1 0.000000 172.26.35.254 193.136.9.240 ICMP 1514 Echo (ping) request id=0xe611, seq=1/256, ttl=1 (no response found!)
```

Porque era demasiado grande para ser transmitido de uma única vez, devido ao MTU ser igual 1500 (Maximum Transmission Unit).

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?

```
1 0.000000 172.26.35.254 193.136.9.240 ICMP 1514 Echo (ping) request id=0xe611, seq=1/256, ttl=1 (no response found!)
2 0.000045 172.26.35.254 193.136.9.240 IPv4 1514 Fragmented IP protocol (proto=ICMP 1, off=1480, ID=e612)
3 0.000055 172.26.35.254 193.136.9.240 IPv4 579 Fragmented IP protocol (proto=ICMP 1, off=2960, ID=e612)
4 0.012826 172.26.254.254 172.26.35.254 ICMP 70 Time-to-live exceeded (Time to live exceeded in transit)
5 0.014161 172.26.35.254 193.137.16.65 DNS 87 Standard query 0xc514 PTR 254.254.26.172.in-addr.arpa

> Frame 1: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface en0, id 0
> Ethernet II, Src: Apple_24:cd:3a (c0:95:6d:24:cd:3a), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
> Internet Protocol Version 4, Src: 172.26.35.254, Dst: 193.136.9.240
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  ✓ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    0000 00.. = Differentiated Services Codepoint: Default (0)
    .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
  Total Length: 1500
  Identification: 0xe612 (58898)
  ✓ 001. .... = Flags: 0x1, More fragments
    0... .... = Reserved bit: Not set
    .0.. .... = Don't fragment: Not set
    .1. .... = More fragments: Set
    ...0 0000 0000 0000 = Fragment Offset: 0
  ✓ Time to Live: 1
    ✓ [Expert Info (Note/Sequence): "Time To Live" only 1]
      ["Time To Live" only 1]
      [Severity level: Note]
      [Group: Sequence]
  Protocol: ICMP (1)
  Header Checksum: 0x127e [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 172.26.35.254
  Destination Address: 193.136.9.240
> Internet Control Message Protocol
```

Sabemos que foi fragmentado pois tem a flag “More Fragments” e é o primeiro fragmento pois o seu “Fragment offset” é igual a 0. Tendo um tamanho de 1500 bytes.

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?

```

2 0.000045 172.26.35.254 193.136.9.240 IPv4 1514 Fragmented IP protocol (proto=ICMP 1, off=1480, ID=e612)
> Frame 2: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface en0, id 0
> Ethernet II, Src: Apple_24:cd:3a (c0:95:6d:24:cd:3a), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
> Internet Protocol Version 4, Src: 172.26.35.254, Dst: 193.136.9.240
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    0000 00.. = Differentiated Services Codepoint: Default (0)
    .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
  Total Length: 1500
  Identification: 0xe612 (58898)
  > 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
  > Time to Live: 1
    > [Expert Info (Note/Sequence): "Time To Live" only 1]
      ["Time To Live" only 1]
      [Severity level: Note]
      [Group: Sequence]
    Protocol: ICMP (1)
    Header Checksum: 0x11c5 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 172.26.35.254
    Destination Address: 193.136.9.240

```

Pois o “Fragment Offset” não é 0 e existem mais fragmentos pois a flag “More Fragments” assim o indica.

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.

```

1 0.000000 172.26.35.254 193.136.9.240 ICMP 1514 Echo (ping) request id=0xe611, seq=1/256, ttl=1 (no response found!)
2 0.000045 172.26.35.254 193.136.9.240 IPv4 1514 Fragmented IP protocol (proto=ICMP 1, off=1480, ID=e612)
3 0.000055 172.26.35.254 193.136.9.240 IPv4 579 Fragmented IP protocol (proto=ICMP 1, off=2960, ID=e612)
4 0.012826 172.26.35.254 172.26.35.254 ICMP 70 Time-to-live exceeded (Time to live exceeded in transit)
5 0.014161 172.26.35.254 193.137.16.65 DNS 87 Standard query 0xc514 PTR 254.254.26.172.in-addr.arpa
> Frame 1: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface en0, id 0
> Ethernet II, Src: Apple_24:cd:3a (c0:95:6d:24:cd:3a), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
> Internet Protocol Version 4, Src: 172.26.35.254, Dst: 193.136.9.240
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    0000 00.. = Differentiated Services Codepoint: Default (0)
    .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
  Total Length: 1500
  Identification: 0xe612 (58898)
  > 001. .... = Flags: 0x1, More fragments
    0... .... = Reserved bit: Not set
    .0.. .... = Don't fragment: Not set
    ..1. .... = More fragments: Set
    ...0 0000 0000 0000 = Fragment Offset: 0
  > Time to Live: 1
    > [Expert Info (Note/Sequence): "Time To Live" only 1]
      ["Time To Live" only 1]
      [Severity level: Note]
      [Group: Sequence]
    Protocol: ICMP (1)
    Header Checksum: 0x127e [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 172.26.35.254
    Destination Address: 193.136.9.240
> Internet Control Message Protocol

```

```

2 0.000045 172.26.35.254 193.136.9.240 IPv4 1514 Fragmented IP protocol (proto=ICMP 1, off=1480, ID=e612)
> Frame 2: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface en0, id 0
> Ethernet II, Src: Apple_24:cd:3a (c0:95:6d:24:cd:3a), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
> Internet Protocol Version 4, Src: 172.26.35.254, Dst: 193.136.9.240
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    0000 00.. = Differentiated Services Codepoint: Default (0)
    .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
  Total Length: 1500
  Identification: 0xe612 (58898)
  > 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
  > Time to Live: 1
    > [Expert Info (Note/Sequence): "Time To Live" only 1]
      ["Time To Live" only 1]
      [Severity level: Note]
      [Group: Sequence]
    Protocol: ICMP (1)
    Header Checksum: 0x11c5 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 172.26.35.254
    Destination Address: 193.136.9.240

3 0.000055 172.26.35.254 193.136.9.240 IPv4 579 Fragmented IP protocol (proto=ICMP 1, off=2960, ID=e612)
> Frame 3: 579 bytes on wire (4632 bits), 579 bytes captured (4632 bits) on interface en0, id 0
> Ethernet II, Src: Apple_24:cd:3a (c0:95:6d:24:cd:3a), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
> Internet Protocol Version 4, Src: 172.26.35.254, Dst: 193.136.9.240
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    0000 00.. = Differentiated Services Codepoint: Default (0)
    .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
  Total Length: 565
  Identification: 0xe612 (58898)
  > 000. .... = Flags: 0x0
    0... .... = Reserved bit: Not set
    .0.. .... = Don't fragment: Not set
    ..0. .... = More fragments: Not set
    ...0 0001 0111 0010 = Fragment Offset: 2960
  > Time to Live: 1
    > [Expert Info (Note/Sequence): "Time To Live" only 1]
      ["Time To Live" only 1]
      [Severity level: Note]
      [Group: Sequence]
    Protocol: ICMP (1)
    Header Checksum: 0x34b3 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 172.26.35.254
    Destination Address: 193.136.9.240

```

Teoricamente teríamos 3 fragmentos. Os dois primeiros com um tamanho de 1500 bytes e o último com 565 bytes. De onde os todos tem ambos um header de 20 bytes.

Na prática acontece o que foi descrito anterior menos os 8 bits para cada fragmento provindos do header ICMP.

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.

Pois no último fragmento, a flag “More Fragments” altera de Set para Not Set.

```

3 0.000055 172.26.35.254 193.136.9.240 IPv4 579 Fragmented IP protocol (proto=ICMP 1, off=2960, ID=e612)
> Frame 3: 579 bytes on wire (4632 bits), 579 bytes captured (4632 bits) on interface en0, id 0
> Ethernet II, Src: Apple_24:cd:3a (c0:95:6d:24:cd:3a), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
> Internet Protocol Version 4, Src: 172.26.35.254, Dst: 193.136.9.240
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    0000 00.. = Differentiated Services Codepoint: Default (0)
    .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
  Total Length: 565
  Identification: 0xe612 (58898)
  > 000. .... = Flags: 0x0
    0... .... = Reserved bit: Not set
    .0.. .... = Don't fragment: Not set
    ..0. .... = More fragments: Not set
    ...0 0001 0111 0010 = Fragment Offset: 2960
  > Time to Live: 1
    > [Expert Info (Note/Sequence): "Time To Live" only 1]
      ["Time To Live" only 1]
      [Severity level: Note]
      [Group: Sequence]
    Protocol: ICMP (1)
    Header Checksum: 0x34b3 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 172.26.35.254
    Destination Address: 193.136.9.240

```


Filtro: ip.flags.mf == 0 && ip.frag_offset != 0

No.	Time	Source	Destination	Protocol	Length	Info
3	0.000055	172.26.35.254	193.136.9.240	IPv4	579	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=e612)
9	0.019645	172.26.35.254	193.136.9.240	IPv4	579	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=e613)
13	0.023650	172.26.35.254	193.136.9.240	IPv4	579	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=e614)
17	0.028378	172.26.35.254	193.136.9.240	IPv4	579	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=e615)
21	0.033434	172.26.35.254	193.136.9.240	IPv4	579	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=e616)
25	0.037730	172.26.35.254	193.136.9.240	IPv4	579	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=e617)
29	0.042032	172.26.35.254	193.136.9.240	IPv4	579	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=e618)
33	0.048148	172.26.35.254	193.136.9.240	IPv4	579	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=e619)
37	0.052722	172.26.35.254	193.136.9.240	IPv4	579	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=e61a)
41	0.056834	172.26.35.254	193.136.9.240	IPv4	579	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=e61b)
43	0.063057	193.136.9.240	172.26.35.254	IPv4	579	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=90e5)
47	0.063875	172.26.35.254	193.136.9.240	IPv4	579	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=e61c)
49	0.070086	193.136.9.240	172.26.35.254	IPv4	579	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=90e6)
53	0.070329	172.26.35.254	193.136.9.240	IPv4	579	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=e61d)
55	0.076699	193.136.9.240	172.26.35.254	IPv4	579	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=90e7)

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 é reconstruído no destino (equipamento 193.26.35.254).

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.

```
1 0.000000 172.26.35.254 193.136.9.240 ICMP 1514 Echo (ping) request id=0xe611, seq=1/256, ttl=1 (no response found!)
> Frame 1: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface en0, id 0
> Ethernet II, Src: Apple_24:cd:3a (c0:95:6d:24:cd:3a), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
> Internet Protocol Version 4, Src: 172.26.35.254, Dst: 193.136.9.240
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    0000 00.. = Differentiated Services Codepoint: Default (0)
    .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
  Total Length: 1500
  Identification: 0xe612 (58898)
  > 001. .... = Flags: 0x1, More fragments
    0... .... = Reserved bit: Not set
    .0.. .... = Don't fragment: Not set
    ..1. .... = More fragments: Set
    ...0 0000 0000 0000 = Fragment Offset: 0
  > Time to Live: 1
    > [Expert Info (Note/Sequence): "Time To Live" only 1]
      ["Time To Live" only 1]
      [Severity level: Note]
      [Group: Sequence]
    Protocol: ICMP (1)
    Header Checksum: 0x127e [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 172.26.35.254
    Destination Address: 193.136.9.240
> Internet Control Message Protocol
```

```
3 0.000055 172.26.35.254 193.136.9.240 IPv4 579 Fragmented IP protocol (proto=ICMP 1, off=2960, ID=e612)
> Frame 3: 579 bytes on wire (4632 bits), 579 bytes captured (4632 bits) on interface en0, id 0
> Ethernet II, Src: Apple_24:cd:3a (c0:95:6d:24:cd:3a), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
> Internet Protocol Version 4, Src: 172.26.35.254, Dst: 193.136.9.240
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    0000 00.. = Differentiated Services Codepoint: Default (0)
    .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
  Total Length: 565
  Identification: 0xe612 (58898)
  > 000. .... = Flags: 0x0
    0... .... = Reserved bit: Not set
    .0.. .... = Don't fragment: Not set
    ..0. .... = More fragments: Not set
    ...0 0001 0111 0010 = Fragment Offset: 2960
  > Time to Live: 1
    > [Expert Info (Note/Sequence): "Time To Live" only 1]
      ["Time To Live" only 1]
      [Severity level: Note]
      [Group: Sequence]
    Protocol: ICMP (1)
    Header Checksum: 0x34b3 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 172.26.35.254
    Destination Address: 193.136.9.240
```

Alterações observadas:

- Fragment Offset
- More Fragments
- Header Checksum
- Total length

Através do Fragment Offset é possível que no destino sejam reordenados de modo a reconstruir o datagrama original.

Quando o valor do More fragments é igual a 0, identifica que este é o último fragmento, e que não falta mais nenhum.

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

O primeiro fragmento de cada pacote é identificado como ICMP, pois o pacote original é um ICMP.

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 valor com que é comparado é 1500 bytes porque é o MTU (Maximum transmission unit), ao aumentar este valor corríamos o risco de perder dados visto que é possível que a capacidade máxima do canal tenha sido ultrapassada antes de ocorrer a fragmentação. Ao diminuir iria haver um número maior de fragmentos do que é necessário, o que leva a um aumento do overhead, podendo provocar um buffer overflow.

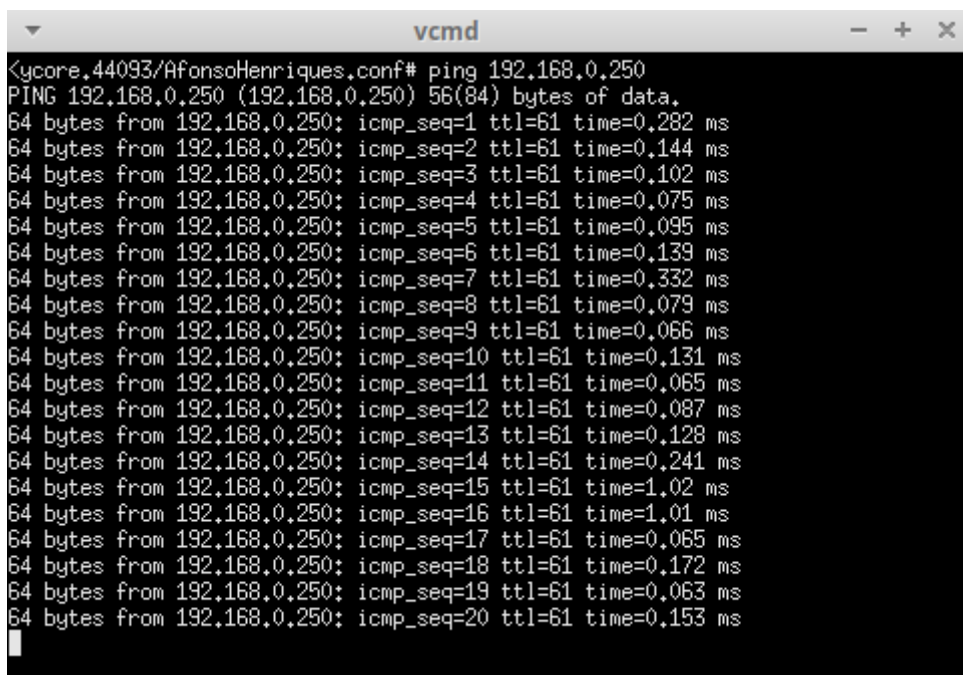
J. Sabendo que no comando ping a opção -f (Windows), -M do (Linux) ou -D (Mac) ativa a flag “Don’t Fragment” (DF) no cabeçalho do IPv4, usando ping <opção DF> <opção pkt_size> SIZE marco.uminho.pt, (opção pkt_size = -l (Windows) ou -s (Linux, Mac), determine o valor máximo de SIZE sem que ocorra fragmentação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

O valor máximo observado foi 1472. O valor do MTU máximo é de 1500, dos quais 20 bytes vão para o header, e outros 8 vão para o header do ICMP. Assim o valor 1472 é um valor plausível.

PARTE II

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.

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



```
<ycore.44093/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=0.282 ms
64 bytes from 192.168.0.250: icmp_seq=2 ttl=61 time=0.144 ms
64 bytes from 192.168.0.250: icmp_seq=3 ttl=61 time=0.102 ms
64 bytes from 192.168.0.250: icmp_seq=4 ttl=61 time=0.075 ms
64 bytes from 192.168.0.250: icmp_seq=5 ttl=61 time=0.095 ms
64 bytes from 192.168.0.250: icmp_seq=6 ttl=61 time=0.139 ms
64 bytes from 192.168.0.250: icmp_seq=7 ttl=61 time=0.332 ms
64 bytes from 192.168.0.250: icmp_seq=8 ttl=61 time=0.079 ms
64 bytes from 192.168.0.250: icmp_seq=9 ttl=61 time=0.066 ms
64 bytes from 192.168.0.250: icmp_seq=10 ttl=61 time=0.131 ms
64 bytes from 192.168.0.250: icmp_seq=11 ttl=61 time=0.065 ms
64 bytes from 192.168.0.250: icmp_seq=12 ttl=61 time=0.087 ms
64 bytes from 192.168.0.250: icmp_seq=13 ttl=61 time=0.128 ms
64 bytes from 192.168.0.250: icmp_seq=14 ttl=61 time=0.241 ms
64 bytes from 192.168.0.250: icmp_seq=15 ttl=61 time=1.02 ms
64 bytes from 192.168.0.250: icmp_seq=16 ttl=61 time=1.01 ms
64 bytes from 192.168.0.250: icmp_seq=17 ttl=61 time=0.065 ms
64 bytes from 192.168.0.250: icmp_seq=18 ttl=61 time=0.172 ms
64 bytes from 192.168.0.250: icmp_seq=19 ttl=61 time=0.063 ms
64 bytes from 192.168.0.250: icmp_seq=20 ttl=61 time=0.153 ms

Kycore.44093/AfonsoHenriques.conf# ping 172.16.142.6
PING 172.16.142.6 (172.16.142.6) 56(84) bytes of data.
From 172.16.143.1 icmp_seq=1 Destination Net Unreachable
From 172.16.143.1 icmp_seq=2 Destination Net Unreachable
From 172.16.143.1 icmp_seq=3 Destination Net Unreachable
From 172.16.143.1 icmp_seq=4 Destination Net Unreachable
From 172.16.143.1 icmp_seq=40 Destination Net Unreachable
From 172.16.143.1 icmp_seq=81 Destination Net Unreachable
```

AfonsoHenriques tem conectividade com o servidor Financas, mas não tem com os servidores da CDN.

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.44093/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.44093/AfonsoHenriques.conf#
```

```
vcmd
root@Teresa:/tmp/pycore.44093/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.44093/Teresa.conf#
```

As tabelas de encaminhamento não aparentam ter problemas.

Na tabela de encaminhamento de AfonsoHenriques tem uma entrada com destino 192.168.0.224 que é o endereço da subnet onde AfonsoHenriques se encontra, esta entrada é para casos em que se quer enviar algo para alguém na mesma subnet, sendo que o próximo salto é o próprio AfonsoHenriques de modo a ser ele mesmo a encaminhar o pacote até ao destino. A outra entrada tem destino 0.0.0.0, ou seja, é a entrada default da tabela, sendo que o próximo salto é 192.168.0.225 que é o RACondado que já apresenta condições de transmitir pacotes para alguém fora da subnet.

O mesmo aplica-se à Teresa, exceto que na entrada default o próximo salto é RAGaliza, o router de acesso à subnet onde a Teresa se encontra.

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.

```
vcmd
pycore.42087/AfonsoHenriques.conf# traceroute -I 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) 0.049 ms 0.007 ms 0.004 ms
 2 172.16.143.1 (172.16.143.1) 0.023 ms 0.007 ms 0.007 ms
 3 10.0.0.29 (10.0.0.29) 0.319 ms !N 0.387 ms !N *
```

Não somos capazes de estabelecer ligação do AfonsoHenriques a Teresa devido ao router n5 que não possui a respetiva entrada na tabela de encaminhamento, ou seja, não consegue encaminhar para a subnet onde se encontra Teresa. Para corrigir o problema adicionamos uma entrada à tabela que tem como destino o endereço 192.168.0.192 que é a subnet Galiza, e como próximo salto o router n2.

```

root@n2:/tmp/pycore.46495/n2.conf# netstat -rn
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

```

No router n2 existia uma entrada na tabela com destino no ip da Teresa, esta entrada tinha maior prioridade que a entrada que encaminha para a subnet Galiza e tinha um next hop errado, portanto foi removida.

```

vcmd
root@n1:/tmp/pycore.42087/n1.conf# netstat -rn
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
root@n1:/tmp/pycore.42087/n1.conf#

```

```

R5/HfonsoHenriques.conf# traceroute -l 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) 0.041 ms 0.006 ms 0.005 ms
 2 172.16.143.1 (172.16.143.1) 0.059 ms 0.012 ms 0.008 ms
 3 10.0.0.29 (10.0.0.29) 0.020 ms 0.009 ms 0.009 ms
 4 10.0.0.25 (10.0.0.25) 0.033 ms 0.014 ms 0.011 ms
 5 10.0.0.13 (10.0.0.13) 0.027 ms 0.014 ms 0.014 ms
 6 10.0.0.25 (10.0.0.25) 0.014 ms 0.042 ms 0.015 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) 0.188 ms 0.068 ms
14 10.0.0.25 (10.0.0.25) 0.062 ms 0.061 ms 0.059 ms
15 10.0.0.13 (10.0.0.13) 0.064 ms 0.065 ms 0.065 ms
16 10.0.0.25 (10.0.0.25) 0.064 ms 0.066 ms *^C

```

O router n1 esta a enviar de volta ao n2, que consequentemente envia de volta ao n1.

```
root@n1:/tmp/pycore.45275/n1.conf# netstat -rn
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
root@n1:/tmp/pycore.45275/n1.conf#
```

```
192.168.0.192 10.0.0.9 255.255.255.248 UG 0 0 0 eth0
```

Ao chegar ao RAGaliza não existia maneira encaminhar de volta. Adicionamos o destino 192.168.0.224/29 para corrigir o problema.

```
vcmd
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
192.168.0.224 172.16.142.1 255.255.255.248 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.46495/RAGaliza.conf#
```

Ao fim destas alterações Afonso Henriques conseguiu estabelecer a teresa e a teresa ao Afonso Henriques.

```
vcmd
AfonsoHenriques.conf# traceroute -l 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) 0.041 ms 0.006 ms 0.005 ms
 2 172.16.143.1 (172.16.143.1) 0.019 ms 0.007 ms 0.007 ms
 3 10.0.0.29 (10.0.0.29) 0.020 ms 0.009 ms 0.009 ms
 4 10.0.0.25 (10.0.0.25) 0.032 ms 0.012 ms 0.013 ms
 5 10.0.0.13 (10.0.0.13) 0.029 ms 0.015 ms 0.014 ms
 6 10.0.0.17 (10.0.0.17) 0.038 ms 0.054 ms 0.020 ms
 7 10.0.0.5 (10.0.0.5) 0.038 ms 0.021 ms 0.021 ms
 8 10.0.0.1 (10.0.0.1) 0.037 ms 0.084 ms 0.026 ms
 9 172.16.142.2 (172.16.142.2) 0.038 ms 0.026 ms 0.026 ms
10 192.168.0.194 (192.168.0.194) 0.045 ms 0.029 ms 0.028 ms
root@AfonsoHenriques:/tmp/pycore.46495/AfonsoHenriques.conf#
```


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. 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.

i)

Os pacotes já são recebidos tanto do lado da teresa com o lado do afonsoHenriques existindo conexão entre os dois.

57	6.145915560	192.168.0.194	192.168.0.226	ICMP	526 Echo (ping) request	id=0x001f, seq=27/6912, ttl=9 (no respon...
58	6.145957175	172.16.143.2	192.168.0.194	ICMP	554 Time-to-live exceeded (Time to live exceeded in transit)	
59	6.145961638	192.168.0.194	192.168.0.226	ICMP	526 Echo (ping) request	id=0x001f, seq=28/7168, ttl=10 (reply in ...
60	6.145997731	192.168.0.226	192.168.0.194	ICMP	526 Echo (ping) reply	id=0x001f, seq=28/7168, ttl=55 (request ...
61	6.146003369	192.168.0.194	192.168.0.226	ICMP	526 Echo (ping) request	id=0x001f, seq=29/7424, ttl=10 (reply in ...
62	6.146021341	192.168.0.226	192.168.0.194	ICMP	526 Echo (ping) reply	id=0x001f, seq=29/7424, ttl=55 (request ...
63	6.146023932	192.168.0.194	192.168.0.226	ICMP	526 Echo (ping) request	id=0x001f, seq=30/7680, ttl=10 (reply in ...
64	6.146041800	192.168.0.226	192.168.0.194	ICMP	526 Echo (ping) reply	id=0x001f, seq=30/7680, ttl=55 (request ...
65	6.146046304	192.168.0.194	192.168.0.226	ICMP	526 Echo (ping) request	id=0x001f, seq=31/7936, ttl=11 (reply in ...
66	6.146079339	192.168.0.226	192.168.0.194	ICMP	526 Echo (ping) reply	id=0x001f, seq=31/7936, ttl=55 (request ...
67	6.146082149	192.168.0.194	192.168.0.226	ICMP	526 Echo (ping) request	id=0x001f, seq=32/8192, ttl=11 (reply in ...
68	6.146096649	192.168.0.226	192.168.0.194	ICMP	526 Echo (ping) reply	id=0x001f, seq=32/8192, ttl=55 (request ...

62	13.009101818	192.168.0.226	192.168.0.194	ICMP	526 Echo (ping) request	id=0x0071, seq=27/6912, ttl=8 (no respon...
63	13.009118577	172.16.142.2	192.168.0.226	ICMP	554 Time-to-live exceeded (Time to live exceeded in transit)	
64	13.009124557	192.168.0.226	192.168.0.194	ICMP	526 Echo (ping) request	id=0x0071, seq=28/7168, ttl=9 (reply in ...
65	13.009156119	192.168.0.194	192.168.0.226	ICMP	526 Echo (ping) reply	id=0x0071, seq=28/7168, ttl=56 (request ...
66	13.009164934	192.168.0.226	192.168.0.194	ICMP	526 Echo (ping) request	id=0x0071, seq=29/7424, ttl=9 (reply in ...
67	13.009185922	192.168.0.194	192.168.0.226	ICMP	526 Echo (ping) reply	id=0x0071, seq=29/7424, ttl=56 (request ...
68	13.009192136	192.168.0.226	192.168.0.194	ICMP	526 Echo (ping) request	id=0x0071, seq=30/7680, ttl=9 (reply in ...
69	13.009212750	192.168.0.194	192.168.0.226	ICMP	526 Echo (ping) reply	id=0x0071, seq=30/7680, ttl=56 (request ...
70	13.009219244	192.168.0.226	192.168.0.194	ICMP	526 Echo (ping) request	id=0x0071, seq=31/7936, ttl=10 (reply in ...
71	13.009239526	192.168.0.194	192.168.0.226	ICMP	526 Echo (ping) reply	id=0x0071, seq=31/7936, ttl=56 (request ...
72	13.009245627	192.168.0.226	192.168.0.194	ICMP	526 Echo (ping) request	id=0x0071, seq=32/8192, ttl=10 (reply in ...
73	13.009265308	192.168.0.194	192.168.0.226	ICMP	526 Echo (ping) reply	id=0x0071, seq=32/8192, ttl=56 (request ...

ii)

Afonso:

Afonso -> RACondado -> ReiDaNet -> n5 -> n2 -> n1 -> n3 -> n6 -> CondadOnline -> RAGaliza -> Teresa

```

1 192.168.0.225 (192.168.0.225) 2.117 ms 0.020 ms 0.003 ms
2 172.16.143.1 (172.16.143.1) 0.195 ms 0.025 ms 0.005 ms
3 10.0.0.29 (10.0.0.29) 0.575 ms 0.013 ms 0.006 ms
4 10.0.0.25 (10.0.0.25) 0.256 ms 0.012 ms 0.007 ms
5 10.0.0.13 (10.0.0.13) 0.347 ms 0.023 ms 0.049 ms
6 10.0.0.17 (10.0.0.17) 0.808 ms 0.075 ms 0.023 ms
7 10.0.0.5 (10.0.0.5) 0.395 ms 0.025 ms 0.013 ms
8 10.0.0.1 (10.0.0.1) 0.507 ms 0.026 ms 0.015 ms
9 172.16.142.2 (172.16.142.2) 0.357 ms 0.065 ms 0.033 ms
10 192.168.0.194 (192.168.0.194) 0.725 ms 0.082 ms 0.028 ms
root@AfonsoHenriques:/tmp/pycore.45275/AfonsoHenriques.conf#

```

Teresa:

Teresa -> RAGaliza -> CondadOnline -> n6 -> n3 -> n4 -> n2 -> n5 -> ReiDaNet -> RACondado -> Afonso.

```

1 192.168.0.193 (192.168.0.193) 0.705 ms 0.011 ms 0.005 ms
2 172.16.142.1 (172.16.142.1) 0.019 ms 0.008 ms 0.006 ms
3 10.0.0.2 (10.0.0.2) 0.350 ms 0.011 ms 0.006 ms
4 10.0.0.6 (10.0.0.6) 0.131 ms 0.010 ms 0.007 ms
5 10.0.0.18 (10.0.0.18) 0.584 ms 1.488 ms 1.484 ms
6 10.0.0.14 (10.0.0.14) 1.543 ms 0.036 ms 0.015 ms
7 10.0.0.26 (10.0.0.26) 0.032 ms 0.019 ms 0.018 ms
8 10.0.0.30 (10.0.0.30) 0.423 ms 0.019 ms 0.013 ms
9 172.16.143.2 (172.16.143.2) 0.140 ms 0.018 ms 0.015 ms
10 192.168.0.226 (192.168.0.226) 0.048 ms 0.019 ms 0.037 ms
root@Teresa:/tmp/pycore.45275/Teresa.conf#

```

Como se pode ver acima a única diferença visível no percurso da Teresa é o uso do router n4 em vez do n1 que o Afonso usa.

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.

Não, pois, n3 vai realizar o longest prefix match. Entre os dois o qual que ira ser utilizado é o 192.168.0.192/29 pois possui mais um bit “reservado” invés do 192.168.0.192/28.

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

Os endereços usados nos polos são endereços privados devido estarem na gama 192.168.0.0-192.168.255.255/16. Já os endereços usados no core, como ISPs também são privados, devido a pertencerem a gama 172.16.0.0 -172.31.255.255/12 e da gama 10.0.0.0-10.255.255.255/8.

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

Não, os switches tem a utilidade de conectar todos os dispositivos com apenas um ip do router.

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

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.

Ao adicionar a rota da imagem abaixo o Castelo vai conseguir o acesso aos polos outra vez. A rota default é bastante útil para que cada dispositivo não tenha de conhecer todas as rotas para cada destino. Quando ele não conhece o endereço ele envia na rota default para outro dispositivo, e acontece o mesmo até a um dispositivo saber a rota até ao destino.

```
192.168.0.192 192.168.0.225 255.255.255.192 UG 0 0 0 eth0

root@Castelo:/tmp/pycore.45275/Castelo.conf# traceroute -I 192.168
traceroute to 192.168,0,194 (192.168,0,194), 30 hops max, 512 byte
 1 192.168,0,225 (192.168,0,225) 0.038 ms 0.008 ms 0.002 ms
 2 * * *
 3 * * *
 4 * * *
 5 * * *
 6 * * *
 7 * * *
 8 * * *
 9 * * *
10 192.168,0,194 (192.168,0,194) 0.037 ms 0.031 ms 0.025 ms
```

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.

Em binário o endereço de rede IP:

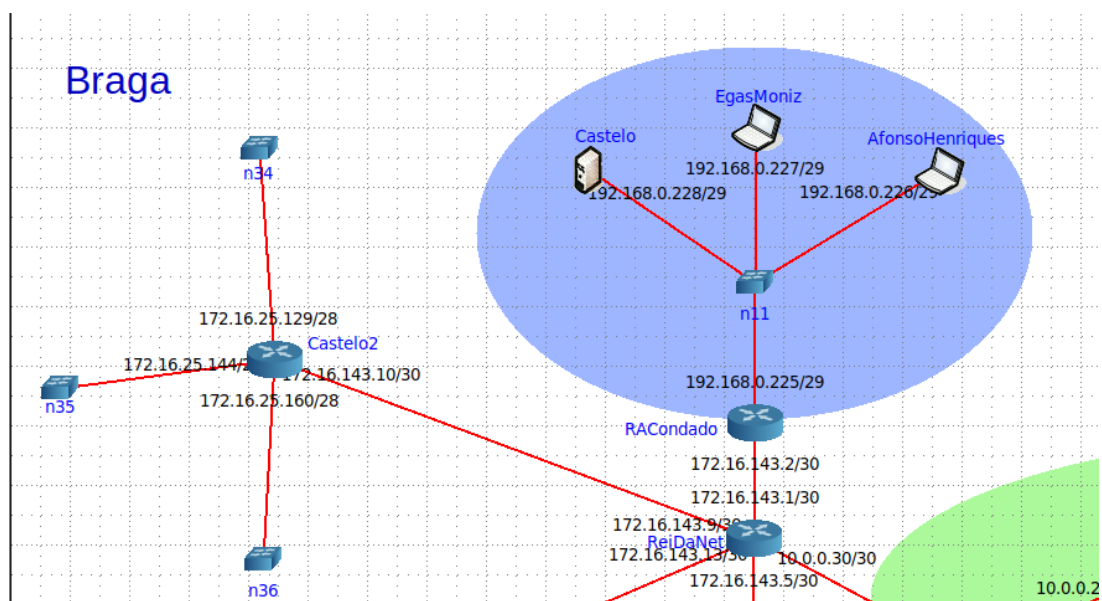
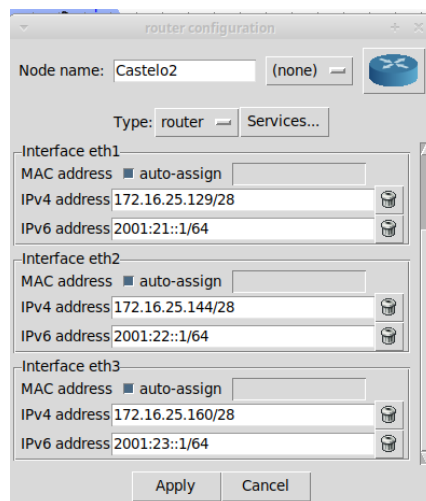
10101100.00010000.00011001.10XXXXXX

Como a máscara.

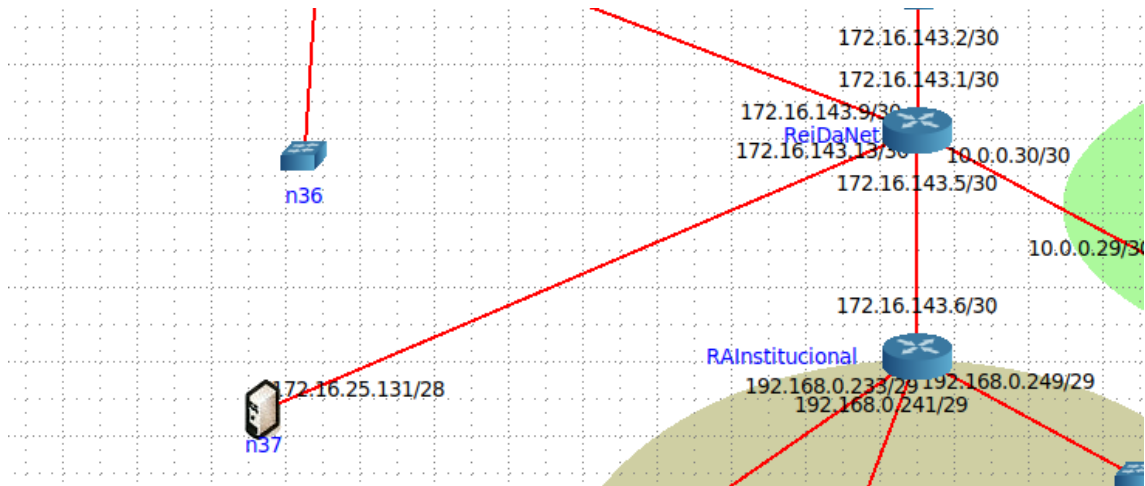
Bits para identificar a sub-rede.

O endereço de rede IP 172.16.25.128/26 terá de ser dividido em 3 sub-redes das quais, 2 bits serão usados para identificar a sub-rede e os restantes 4 serão usados para os hosts., resultando numa máscara /28.

A primeira sub-rede tem endereço de rede IP de 172.16.25.129/28 (subnet 00), a segunda tem endereço de rede IP 172.16.25.144/28 (subnet 01) e a terceira tem endereço de rede 172.16.25.160/28 (subnet 10). Na terceira sub-rede a máscara poderia ser igual a 26 pois todos os endereços a partir 172.16.25.160 não vão ser usados por mais nenhuma sub-rede.



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?



3) Ao planear 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.

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.

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

```
root@n6:/tmp/pycore.46495/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
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
root@n6:/tmp/pycore.46495/n6.conf#
```

Ao apagar todas as rotas referentes a todos os polos (Galiza, CDN, CondadoPortucalense e Institucional), o dispositivo n6 ficou com apenas as rotas para os dispositivos do core.

A) Ao adicionar o destino 192.168.0.192/27 com gateway 10.0.0.1 (CondadOnline) a conectividade entre os polos é mantém-se. (supernetting referente aos polos de Galiza e CDN).

B) Ao adicionar o destino 192.168.0.224/27 com gateway 10.0.0.6 (n3) a conectividade entre os polos mantém-se. (supernetting referente aos polos de CondadoPortucalense e Institucional).

A máscara ser /27 é o resultado do supernetting feito para ambos os lados da topologia.

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

O supernetting é uma boa técnica devido a otimizar o uso de endereços de rede IP e a redução da tabela de endereços já que a partir de uma única rota podem-se definir se definir rotas para vários dispositivos, tornando a manutenção da mesma mais fácil. O maior aspeto negativo é o aumento da complexidade.