

**Universidade do Minho**

Escola de Engenharia

Licenciatura em Engenharia Informática

## **Unidade Curricular de Redes de Computadores**

Ano Letivo de 2022/2023

### **Trabalho Prático Nº2** **Protocolo IPv4 :: Datagramas IP e Fragmentação**

#### **Grupo 57**

A94942 Miguel Velho Raposo  
A78823 João Carlos Cotinho Sotomaior Neto  
A91775 José Pedro Batista Fonte

20 de abril de 2023

# Índice

Lista de Figuras . . . . .	3
<b>1 Introdução</b>	<b>5</b>
<b>2 Parte 1</b>	<b>6</b>
2.1 Exercício 1 . . . . .	6
2.2 Exercício 2 . . . . .	9
2.3 Exercício 3 . . . . .	13
<b>3 Parte 2</b>	<b>20</b>
3.1 Exercício 1 . . . . .	21
3.2 Exercício 2 . . . . .	30
3.3 Exercício 3 . . . . .	36
<b>4 Conclusão</b>	<b>40</b>

## Lista de Figuras

2.1	Topologia da Rede do Exercício 1 da Parte 1 . . . . .	6
2.2	Resultados do traceroute ao Found . . . . .	7
2.3	Análise do Wireshark do Lost . . . . .	7
2.4	Traceroute com 5 pacotes . . . . .	8
2.5	Resultados do traceroute . . . . .	9
2.6	Resultados do ifconfig . . . . .	9
2.7	Campo Protocol . . . . .	9
2.8	Datagrama com a flag do protocolo em evidência . . . . .	10
2.9	Datagrama IP/ICMP . . . . .	10
2.10	Comparação entre datagramas . . . . .	11
2.11	Análise Wireshark . . . . .	11
2.12	Análise Wireshark . . . . .	12
2.13	Resultados do ping . . . . .	13
2.14	Análise Wireshark . . . . .	14
2.15	1º fragmento do pacote . . . . .	14
2.16	2º fragmento do pacote . . . . .	15
2.17	3º fragmento do pacote . . . . .	15
2.18	3º fragmento do pacote . . . . .	16
2.19	3º fragmento do pacote . . . . .	16
2.20	Flags dos fragmentos . . . . .	17
2.21	Teste da Unidade Máxima de Transmissão (MTU) . . . . .	19
3.1	Topologia de Rede da Parte 2 . . . . .	20
3.2	Testes de Conetividade ao Institucional e CDN . . . . .	21
3.3	Ping ao PC Teresa . . . . .	21
3.4	Tabelas de Encaminhamento . . . . .	22
3.5	1º Traceroute a Teresa . . . . .	23
3.6	Wireshark - Captura de tráfego em N5 . . . . .	23
3.7	Tabela de Encaminhamento do Router N5 . . . . .	24
3.8	2º Traceroute a Teresa . . . . .	24
3.9	Tabela de Encaminhamento do Router N5 . . . . .	24
3.10	3º Traceroute a Teresa . . . . .	25
3.11	Tabela de Encaminhamento do Router N1 . . . . .	25
3.12	4º Traceroute a Teresa . . . . .	26
3.13	Teste de Conetividade AfonsoHenriques ↔ Teresa . . . . .	26
3.14	Tabela de Encaminhamento do Router RAGaliza . . . . .	27
3.15	Teste de Conetividade AfonsoHenriques ↔ Teresa . . . . .	27
3.16	Traceroute AfonsoHenriques ↔ Teresa . . . . .	28
3.17	Caminhos AfonsoHenriques ↔ Teresa . . . . .	28
3.18	Tabela de Encaminhamento de Castelo . . . . .	30
3.19	Tabela de Encaminhamento de Castelo sem rota default . . . . .	30
3.20	Nova Tabela de Encaminhamento de Castelo . . . . .	31
3.21	Testes de conetividade ao Institucional . . . . .	31
3.22	Testes de conetividade à CDN . . . . .	31

3.23	Teste de conectividade à Galiza . . . . .	31
3.24	Testes de conectividade ao ISP ReiDaNet . . . . .	32
3.25	Cálculo das Subredes . . . . .	33
3.26	Novo Polo Braga . . . . .	33
3.27	Novo host DJ8 . . . . .	34
3.28	Tabelas de Encaminhamento do ISP ReiDaNet . . . . .	35
3.29	Testes de Conetividade a Braga . . . . .	35
3.30	Tabelas de Encaminhamento do Router n6 . . . . .	36
3.31	Cálculo do Supernetting da Galiza e CDN . . . . .	36
3.32	Router n6 com Supernetting . . . . .	37
3.33	Tabelas de Encaminhamento do Router n6 . . . . .	37
3.34	Cálculo do Supernetting do Instuticional e CondadoPortucalense . . . . .	38
3.35	Router n6 com Supernetting . . . . .	38

# 1 Introdução

O Internet Protocol (IP) é um dos principais protocolos de comunicação utilizado na Internet e em redes de computadores. Ele é responsável pelo roteamento e entrega de pacotes de dados entre diferentes dispositivos conectados numa rede. O estudo detalhado do IP é fundamental para entender seu funcionamento e suas principais vertentes, como o formato de um datagrama IP, a fragmentação de pacotes IP, o endereçamento IP e o encaminhamento IP.

Neste relatório de duas partes, abordamos os principais temas de estudo do protocolo IPv4. Na primeira parte, realizamos a análise e registo de datagramas IP enviados e recebidos através da execução do programa "traceroute" e "ping". Com o programa Wireshark analisamos os diversos campos de um datagrama IP e detalhamos o processo de fragmentação realizado pelo protocolo IP.

Na segunda parte, damos continuidade ao estudo do protocolo IPv4, com ênfase no endereçamento e encaminhamento IP. Abordaremos algumas das técnicas mais relevantes propostas para aumentar a escalabilidade do protocolo IP, mitigar a exaustão dos endereços IPv4 e reduzir os recursos de memória necessários nos router para manter as tabelas de encaminhamento. Entre as técnicas mais comuns, destacam-se o Classless InterDomain Routing (CIDR), as Subredes (subnetting), o Variable Length Subnet Masking (VLSM), a sumarização de rotas (supernetting), a atribuição dinâmica de endereços usando o DHCP (Dynamic Host Configuration Protocol) e a utilização de endereços privados conforme o protocolo RFC 1918.

É importante realçar que estas técnicas resolvem o problema da exaustão de endereços IPv4 apenas no curto/médio prazo. Para responder ao aumento significativo do número de endereços necessários a longo prazo, uma solução é a implementação progressiva do Internet Protocol versão 6 (IPv6).

No decorrer deste relatório, exploramos em detalhe cada uma das vertentes do protocolo IP, destacando suas principais características, benefícios e limitações. A compreensão desses conceitos é fundamental para a gestão eficiente, segura e confiável de redes de computadores, especialmente no contexto atual de crescente procura por endereços IP e a evolução constante das tecnologias de rede.

## 2 Parte 1

O principal objetivo desta parte trabalho é o estudo do Internet Protocol (IP) nas suas principais vertentes, nomeadamente: (i) estudo do formato de um pacote ou datagrama IP; (ii) fragmentação de pacotes IP. Na primeira parte deste estudo é realizado o registo de datagramas IP enviados e recebidos através da execução do programa "traceroute" e "ping". Os vários campos de um datagrama IP/ICMP são analisados e detalhados de modo a entender os cabeçalhos IP e ICMP e como funciona o processo de fragmentação do IP.

### 2.1 Exercício 1

Para verificar o comportamento do "traceroute" o grupo construiu uma topologia de acordo com as seguintes instruções:

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.

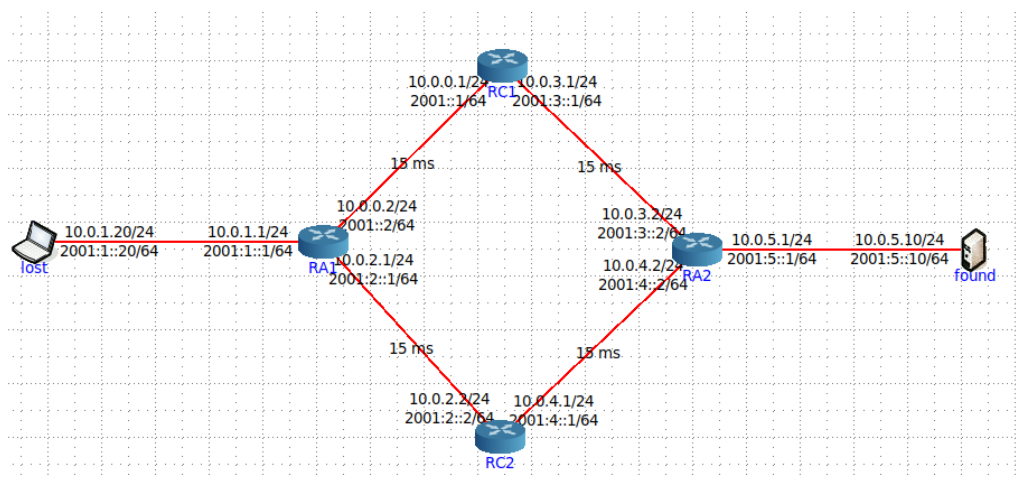


Figura 2.1: Topologia da Rede do Exercício 1 da Parte 1

a. Active o Wireshark no host Lost. Numa shell de Lost execute o comando **tracert -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 baseado na envio sequencial de pacotes ICMP com o objetivo de mostrar o caminho de nodos intermédios e os tempos de latência. Por norma o traceroute envia da origem 3 pacotes ICMP com um Time to Live (TTL) com valores incrementalmente mais altos, cada vez que o router intermédio recebe um pacote decrementa 1 ao seu TTL, se o TTL chegar a zero, descarta o pacote e envia-o de volta para a origem com a mensagem "TTL exceeded". Ao chegar à origem, este regista os IPs de origem dos pacotes com a mensagem "TTL exceeded" e calcula o seu RTT. Desse modo o traceroute mantém uma lista do percurso e dos tempos de ida e volta de cada pacote, permitindo ter resultados como o da figura 2.2.

Como é possível ver na figura 2.2, o grupo abriu uma shell no Lost e executou o traceroute alvejando o Found (IP: 10.0.5.10). Os resultados provam a topologia criada pelo grupo, o primeiro salto é para RA1 (IP: 10.0.1.1), o segundo salto para RC1 (10.0.0.1), o terceiro salto para RA2 (IP: 10.0.3.2) e o último salto para o Found (IP: 10.0.5.10). Os valores obtidos pelo traceroute representam o RTT (Round Trip Time) de cada pacote enviado.

```
root@lost:/tmp/pycore.37185/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 10.0.1.1 (10.0.1.1) 0.042 ms 0.010 ms 0.008 ms
 2 10.0.0.1 (10.0.0.1) 30.782 ms 30.770 ms 30.765 ms
 3 10.0.3.2 (10.0.3.2) 61.786 ms 61.782 ms 61.779 ms
 4 10.0.5.10 (10.0.5.10) 61.772 ms 61.768 ms 61.764 ms
```

Figura 2.2: Resultados do traceroute ao Found

No.	Time	Source	Destination	Protocol	Length	Info
18	29.634123116	10.0.1.1	224.0.0.5	OSPF	78	Hello Packet
19	29.537912477	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=1/256, ttl=1 (no response found!)
20	29.537939848	10.0.1.1	10.0.1.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
21	29.537949551	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=2/512, ttl=1 (no response found!)
22	29.537959376	10.0.1.1	10.0.1.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
23	29.537981347	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=3/768, ttl=1 (no response found!)
24	29.537986454	10.0.1.1	10.0.1.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
25	29.537972732	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=4/1024, ttl=2 (no response found!)
26	29.537987917	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=5/1280, ttl=2 (no response found!)
27	29.537992905	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=6/1536, ttl=2 (no response found!)
28	29.538000348	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=7/1792, ttl=3 (no response found!)
29	29.538006393	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=8/2048, ttl=3 (no response found!)
30	29.538012626	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=9/2304, ttl=3 (no response found!)
31	29.538019334	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=10/2560, ttl=4 (reply in 50)
32	29.538025856	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=11/2816, ttl=4 (reply in 51)
33	29.538032138	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=12/3072, ttl=4 (reply in 52)
34	29.538038674	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=13/3328, ttl=5 (reply in 53)
35	29.538045663	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=14/3584, ttl=5 (reply in 54)
36	29.538051502	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=15/3840, ttl=5 (reply in 55)
37	29.538058890	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=16/4096, ttl=6 (reply in 56)
38	29.539238494	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=17/4352, ttl=6 (reply in 57)
39	29.539381898	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=18/4608, ttl=6 (reply in 58)
40	29.539311084	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=19/4864, ttl=7 (reply in 59)
41	29.568748955	10.0.1.1	10.0.1.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
42	29.568753374	10.0.1.1	10.0.1.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
43	29.568754955	10.0.1.1	10.0.1.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
44	29.569179247	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=20/5120, ttl=7 (reply in 60)
45	29.569298578	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=21/5376, ttl=7 (reply in 61)
46	29.569299512	10.0.1.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002f, seq=22/5632, ttl=8 (reply in 62)
47	29.599776492	10.0.3.2	10.0.1.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
48	29.599785640	10.0.3.2	10.0.1.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
49	29.599787705	10.0.3.2	10.0.1.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
50	29.599793927	10.0.5.10	10.0.1.20	ICMP	74	Echo (ping) reply id=0x002f, seq=10/2560, ttl=61 (request in 31)
51	29.599791516	10.0.5.10	10.0.1.20	ICMP	74	Echo (ping) reply id=0x002f, seq=11/2816, ttl=61 (request in 32)
52	29.599793417	10.0.5.10	10.0.1.20	ICMP	74	Echo (ping) reply id=0x002f, seq=12/3072, ttl=61 (request in 33)
53	29.599795295	10.0.5.10	10.0.1.20	ICMP	74	Echo (ping) reply id=0x002f, seq=13/3328, ttl=61 (request in 34)
54	29.599797165	10.0.5.10	10.0.1.20	ICMP	74	Echo (ping) reply id=0x002f, seq=14/3584, ttl=61 (request in 35)
55	29.599799048	10.0.5.10	10.0.1.20	ICMP	74	Echo (ping) reply id=0x002f, seq=15/3840, ttl=61 (request in 36)
56	29.599800924	10.0.5.10	10.0.1.20	ICMP	74	Echo (ping) reply id=0x002f, seq=16/4096, ttl=61 (request in 37)
57	29.603124679	10.0.5.10	10.0.1.20	ICMP	74	Echo (ping) reply id=0x002f, seq=17/4352, ttl=61 (request in 38)
58	29.603130820	10.0.5.10	10.0.1.20	ICMP	74	Echo (ping) reply id=0x002f, seq=18/4608, ttl=61 (request in 39)
59	29.603132877	10.0.5.10	10.0.1.20	ICMP	74	Echo (ping) reply id=0x002f, seq=19/4864, ttl=61 (request in 40)
60	29.63081718	10.0.5.10	10.0.1.20	ICMP	74	Echo (ping) reply id=0x002f, seq=20/5120, ttl=61 (request in 44)
61	29.630875985	10.0.5.10	10.0.1.20	ICMP	74	Echo (ping) reply id=0x002f, seq=21/5376, ttl=61 (request in 45)
62	29.630838352	10.0.5.10	10.0.1.20	ICMP	74	Echo (ping) reply id=0x002f, seq=22/5632, ttl=61 (request in 46)
63	30.025095315	10.0.1.1	224.0.0.5	OSPF	78	Hello Packet

Figura 2.3: Análise do Wireshark do Lost

Na figura 2.3 é possível analisar a captura de tráfego no Lost e observar o protocolo descrito em

cima. No total observa-se, em três grupos distintos, os 9 pacotes recebidos com a mensagem "Time-to-live exceeded" destacados a preto, que correspondem aos três pacotes retornados por cada router intermédio. De notar que a partir de TTL igual a 4 o Lost já recebe *replies* válidas, o que confirma a topologia criada.

**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.**

Observando a topologia, em teoria são necessários 4 saltos no mínimo para estabelecer uma ligação entre Lost e Found, daí ser expectável um TTL mínimo de 4. Verificando na prática, através da figura 2.3, é possível observar que todos os pacotes com TTL menor que 4 são retornados com a mensagem "Time-to-Live Exceeded" e que a partir de TTL igual a 4 existem *replies* válidas, logo o valor mínimo do TTL é mesmo 4.

**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.**

Executando o traceroute com 5 pacotes é possível obter os resultados da figura 2.4.

```
root@lost:/tmp/pycore.37185/lost.conf# traceroute -I -q 5 10.0.5.10
traceroute to 10.0.5.10 (10.0.5.10), 30 hops max, 60 byte packets
 1 10.0.1.1 (10.0.1.1) 0.030 ms 0.007 ms 0.006 ms 0.006 ms 0.005 ms
 2 10.0.0.1 (10.0.0.1) 30.664 ms 30.655 ms 30.650 ms 30.646 ms 30.643 ms
 3 10.0.3.2 (10.0.3.2) 61.324 ms 61.321 ms 61.318 ms 61.316 ms 61.313 ms
 4 10.0.5.10 (10.0.5.10) 61.310 ms 60.804 ms 60.796 ms 60.792 ms 60.790 ms
root@lost:/tmp/pycore.37185/lost.conf#
```

Figura 2.4: Traceroute com 5 pacotes

Fazendo um simples cálculo da média dos valores da linha 4 obtêm-se um valor de RTT de 61 ms entre Lost e Found.

**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?**

No caso da nossa topologia construída sim, porque o grupo definiu delays simétricos de 15 ms, no entanto, o cálculo desta métrica numa rede real torna-se muito difícil e impreciso que faz com que esta abordagem não seja correta. Os principais motivos que resultam na imprecisão deste cálculo é o dinamismo de uma rede real, que muda com a taxa de congestão da rede, o número de routers no caminho, a latência da rede e a qualidade do equipamento de rede, isto leva os tempos entre routes a mudar constantemente que, por conseguinte, levam à atualização constante dos tabelas de encaminhamento que alteram os caminhos dos pacotes.



Em poucas palavras, o dinamismo da rede leva a que não haja certeza que o caminho de ida de um pacote seja o mesmo que o caminho de volta.

## 2.2 Exercício 2

Estudo com o comando "tracert -I router-di.uminho.pt 512"

```
jfonte@ubuntu:~$ tracert -I router-di.uminho.pt 512
tracert to router-di.uminho.pt (193.136.9.254), 30 hops max, 512 byte packets
 1 _gateway (10.0.2.2) 0.498 ms 0.474 ms 0.465 ms
 2 172.26.254.254 (172.26.254.254) 25.258 ms 25.510 ms 25.503 ms
 3 172.16.2.1 (172.16.2.1) 3.951 ms 4.351 ms 4.343 ms
 4 router-di.uminho.pt (193.136.9.254) 4.335 ms 5.009 ms 4.999 ms
```

Figura 2.5: Resultados do traceroute

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

Como é possível observar na figura 2.6 o endereço IP da interface ativa é o IP 10.0.2.15. Esta informação está de acordo com o tráfego capturado no Wireshark, presente na figura 2.11

```
jfonte@ubuntu:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
    inet6 fe80::4ccf:fe3d:7728:d588 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:28:6c:57 txqueuelen 1000 (Ethernet)
    RX packets 200945 bytes 301551677 (301.5 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 21026 bytes 1403772 (1.4 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Figura 2.6: Resultados do ifconfig

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

O valor do campo protocolo é ICMP(1) e é possível identificá-lo a partir de um campo presente no datagrama. Observando a figura 2.8 é possível ver que o valor do campo "Protocolo" é igual a 01, o que significa que se está a utilizar o protocolo ICMP.

```
Protocol: ICMP (1)
Header Checksum: 0x13ae [validation disabled]
```

Figura 2.7: Campo Protocol

0000	52	54	00	12	35	02	08	00	27	28	6c	57	08	00	45	00	RT	5	...	'(LW	E
0010	02	00	ba	71	00	00	01	31	25	f7	0a	00	02	0f	c1	88		q	...	%	.....
0020	09	fe	08	00	3b	4e	00	03	00	01	48	49	4a	4b	4c	4d		N	...	HIJKLM	
0030	4e	4f	50	51	52	53	54	55	56	57	58	59	5a	5b	5c	5d		NOPQRSTU	VWXYZ	[	]
0040	5e	5f	60	61	62	63	64	65	66	67	68	69	6a	6b	6c	6d		^_`abcde	fghijklm		
0050	6e	6f	70	71	72	73	74	75	76	77	78	79	7a	7b	7c	7d		nopqrstu	vwxyz	{	}
0060	7e	7f	40	41	42	43	44	45	46	47	48	49	4a	4b	4c	4d		~@ABCDE	FGHIJKLM		
0070	4e	4f	50	51	52	53	54	55	56	57	58	59	5a	5b	5c	5d		NOPQRSTU	VWXYZ	[	]
0080	5e	5f	60	61	62	63	64	65	66	67	68	69	6a	6b	6c	6d		^_`abcde	fghijklm		
0090	6e	6f	70	71	72	73	74	75	76	77	78	79	7a	7b	7c	7d		nopqrstu	vwxyz	{	}
00a0	7e	7f	40	41	42	43	44	45	46	47	48	49	4a	4b	4c	4d		~@ABCDE	FGHIJKLM		
00b0	4e	4f	50	51	52	53	54	55	56	57	58	59	5a	5b	5c	5d		NOPQRSTU	VWXYZ	[	]
00c0	5e	5f	60	61	62	63	64	65	66	67	68	69	6a	6b	6c	6d		^_`abcde	fghijklm		
00d0	6e	6f	70	71	72	73	74	75	76	77	78	79	7a	7b	7c	7d		nopqrstu	vwxyz	{	}
00e0	7e	7f	40	41	42	43	44	45	46	47	48	49	4a	4b	4c	4d		~@ABCDE	FGHIJKLM		
00f0	4e	4f	50	51	52	53	54	55	56	57	58	59	5a	5b	5c	5d		NOPQRSTU	VWXYZ	[	]
0100	5e	5f	60	61	62	63	64	65	66	67	68	69	6a	6b	6c	6d		^_`abcde	fghijklm		
0110	6e	6f	70	71	72	73	74	75	76	77	78	79	7a	7b	7c	7d		nopqrstu	vwxyz	{	}
0120	7e	7f	40	41	42	43	44	45	46	47	48	49	4a	4b	4c	4d		~@ABCDE	FGHIJKLM		
0130	4e	4f	50	51	52	53	54	55	56	57	58	59	5a	5b	5c	5d		NOPQRSTU	VWXYZ	[	]
0140	5e	5f	60	61	62	63	64	65	66	67	68	69	6a	6b	6c	6d		^_`abcde	fghijklm		
0150	6e	6f	70	71	72	73	74	75	76	77	78	79	7a	7b	7c	7d		nopqrstu	vwxyz	{	}
0160	7e	7f	40	41	42	43	44	45	46	47	48	49	4a	4b	4c	4d		~@ABCDE	FGHIJKLM		
0170	4e	4f	50	51	52	53	54	55	56	57	58	59	5a	5b	5c	5d		NOPQRSTU	VWXYZ	[	]
0180	5e	5f	60	61	62	63	64	65	66	67	68	69	6a	6b	6c	6d		^_`abcde	fghijklm		
0190	6e	6f	70	71	72	73	74	75	76	77	78	79	7a	7b	7c	7d		nopqrstu	vwxyz	{	}
01a0	7e	7f	40	41	42	43	44	45	46	47	48	49	4a	4b	4c	4d		~@ABCDE	FGHIJKLM		
01b0	4e	4f	50	51	52	53	54	55	56	57	58	59	5a	5b	5c	5d		NOPQRSTU	VWXYZ	[	]
01c0	5e	5f	60	61	62	63	64	65	66	67	68	69	6a	6b	6c	6d		^_`abcde	fghijklm		
01d0	6e	6f	70	71	72	73	74	75	76	77	78	79	7a	7b	7c	7d		nopqrstu	vwxyz	{	}
01e0	7e	7f	40	41	42	43	44	45	46	47	48	49	4a	4b	4c	4d		~@ABCDE	FGHIJKLM		
01f0	4e	4f	50	51	52	53	54	55	56	57	58	59	5a	5b	5c	5d		NOPQRSTU	VWXYZ	[	]
0200	5e	5f	60	61	62	63	64	65	66	67	68	69	6a	6b				^_`abcde	fghijk		

Figura 2.8: Datagrama com a flag do protocolo em evidência

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

Um cabeçalho IPV4 são 20 bytes do datagrama e tal informação é possível observar no campo "Header Length" do cabeçalho IPV4. Quanto ao tamanho do payload calcula-se subtraindo ao valor total do datagrama, o cabeçalho IPV4 (20 bytes) e o cabeçalho ICMP (8 bytes), que no caso da figura 2.8 resulta num datagrama de 60 bytes com 32 bytes de payload. Os valores podem ser confirmados observando o campo "Length" do "Data" presente no cabeçalho ICMP.

```

Frame 17: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface enp0s3, id 0
Ethernet II, Src: PcsCompu, 28:6c:57 (08:00:27:28:6c:57), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
Internet Protocol Version 4, Src: 10.0.2.15, Dst: 193.136.9.254
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 60
  Identification: 0xa497 (42135)
  Flags: 0x00
  ... 0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 3
  Protocol: ICMP (1)
  Header Checksum: 0x3b95 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 10.0.2.15
  Destination Address: 193.136.9.254
Internet Control Message Protocol
  Type: 8 (Echo (ping) request)
  Code: 0
  Checksum: 0x026e [correct]
  [Checksum Status: Good]
  Identifier (BE): 3 (0x0003)
  Identifier (LE): 768 (0x0300)
  Sequence Number (BE): 9 (0x0009)
  Sequence Number (LE): 2304 (0x0900)
[No response seen]
  [Expert Info (Warning/Sequence): No response seen to ICMP request]
Data (32 bytes)
  Data: 48494a4b4c4d4e4f505152535455565758595a5b5c5d5e5f6061626364656667
  [Length: 32]

```

Figura 2.9: Datagrama IP/ICMP

### d. O datagrama IP foi fragmentado? Justifique.

Um datagrama IP tem um conjunto de bits dedicado à especificação da fragmentação de pacotes, os campos observáveis na figura 2.10 (a) tem a seguinte função:

- **Reserved bit** : Este bit é reservado para uso futuro e não é atualmente utilizado.

- **Don't Fragment** : Quando esse bit está definido como 1, indica que o pacote IP não deve ser fragmentado. Isso significa que, se o pacote for maior que o MTU do caminho de transmissão, ele será descartado em vez de ser fragmentado.
- **More Fragments** : Quando esse bit está definido como 1, indica que o pacote IP é parte de uma série de fragmentos de um datagrama maior. Isso significa que ainda há mais fragmentos a seguir.
- **Fragment Offset** : Este campo de 13 bits e indica a posição do fragmento atual em relação ao datagrama original. Este campo é usado para reagrupar corretamente os fragmentos no destino.

O datagrama IP não foi fragmentado e, tal facto, é possível observar pelas flags do cabeçalho IP identificadas na figura 2.10 a). Como é possível ver a flag "More Fragments" está definida a 0 logo não existe fragmentação, um exemplo do contrário é a figura 2.10 b) com a flag a 1.

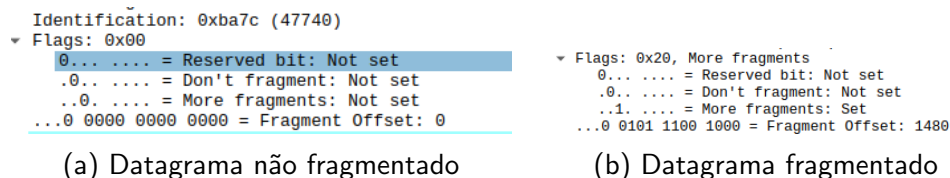


Figura 2.10: Comparação entre datagramas

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

Os campos do cabeçalho que variam de pacote para pacote são o Identificador do pacote e o TTL.

No.	Time	Source	Destination	Protocol	Length	Info
407	0.228471146	10.0.2.15	193.137.16.75	DNS	90	Standard query 0xb8b8 AAAA router-d1.uninho.pt OPT
408	0.238467266	193.137.16.75	10.0.2.15	DNS	153	Standard query response 0xb8b8 AAAA router-d1.uninho.pt SOA dns.uninho.pt OPT
409	0.238467487	193.137.16.75	10.0.2.15	DNS	106	Standard query response 0x7d11 A router-d1.uninho.pt A 193.136.9.254 OPT
410	0.238906410	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=1/204, ttl=1 (no response found)
411	0.238927241	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=2/512, ttl=1 (no response found)
412	0.238933968	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=3/768, ttl=1 (no response found)
413	0.238943414	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=4/1024, ttl=2 (no response found)
414	0.238950318	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=5/1280, ttl=2 (no response found)
415	0.238957343	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=6/1536, ttl=2 (no response found)
416	0.238959124	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=7/1792, ttl=3 (no response found)
417	0.238974444	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=8/2048, ttl=3 (no response found)
418	0.238982681	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=9/2304, ttl=3 (no response found)
419	0.239001151	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=10/2560, ttl=4 (reply in 435)
420	0.239006144	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=11/2816, ttl=4 (reply in 434)
421	0.239009387	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=12/3072, ttl=4 (reply in 435)
422	0.239017084	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=13/3328, ttl=5 (reply in 438)
423	0.239027490	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=14/3584, ttl=5 (reply in 439)
424	0.239036803	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=15/3840, ttl=5 (reply in 440)
425	0.239045089	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=16/4096, ttl=6 (reply in 441)
426	0.239087905	10.0.2.2	10.0.2.15	ICMP	70	Time to live exceeded (Time to live exceeded in transit)
427	0.239089016	10.0.2.2	10.0.2.15	ICMP	70	Time to live exceeded (Time to live exceeded in transit)
428	0.239089077	10.0.2.2	10.0.2.15	ICMP	70	Time to live exceeded (Time to live exceeded in transit)
429	0.239145021	10.0.2.15	193.137.16.75	DNS	92	Standard query 0x725b PTR 2.2.0.10.in-addr.arpa OPT
430	0.240197133	172.26.254.254	10.0.2.15	ICMP	70	Time to live exceeded (Time to live exceeded in transit)
431	0.241324506	172.26.254.254	10.0.2.15	ICMP	70	Time to live exceeded (Time to live exceeded in transit)
432	0.241324504	172.26.254.254	10.0.2.15	ICMP	70	Time to live exceeded (Time to live exceeded in transit)
433	0.241324640	193.136.9.254	10.0.2.15	ICMP	528	Echo (ping) reply id=0x0003, seq=10/2560, ttl=252 (request in 419)
434	0.242006768	193.136.9.254	10.0.2.15	ICMP	528	Echo (ping) reply id=0x0003, seq=11/2816, ttl=252 (request in 420)
435	0.242006869	193.136.9.254	10.0.2.15	ICMP	528	Echo (ping) reply id=0x0003, seq=12/3072, ttl=252 (request in 421)
436	0.242074765	193.137.16.75	10.0.2.15	DNS	103	Standard query response 0x725b PTR 2.2.0.10.in-addr.arpa SOA dns.uninho.pt OPT
437	0.242473909	10.0.2.15	193.137.16.75	DNS	81	Standard query 0x725b PTR 2.2.0.10.in-addr.arpa
438	0.243188440	193.136.9.254	10.0.2.15	ICMP	528	Echo (ping) reply id=0x0003, seq=13/3328, ttl=252 (request in 422)
439	0.243188570	193.136.9.254	10.0.2.15	ICMP	528	Echo (ping) reply id=0x0003, seq=14/3584, ttl=252 (request in 423)
440	0.243188628	193.136.9.254	10.0.2.15	ICMP	528	Echo (ping) reply id=0x0003, seq=15/3840, ttl=252 (request in 424)
441	0.243500615	193.136.9.254	10.0.2.15	ICMP	528	Echo (ping) reply id=0x0003, seq=16/4096, ttl=252 (request in 425)
442	0.246067646	193.137.16.75	10.0.2.15	DNS	172	Standard query response 0x725b No such name PTR 2.2.0.10.in-addr.arpa SOA dns.uninho.pt
443	0.250639956	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=17/4352, ttl=6 (reply in 446)
444	0.250663285	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=18/4608, ttl=6 (reply in 447)
445	0.250671142	10.0.2.15	193.136.9.254	ICMP	528	Echo (ping) request id=0x0003, seq=19/4864, ttl=7 (reply in 448)
446	0.250402197	193.136.9.254	10.0.2.15	ICMP	528	Echo (ping) reply id=0x0003, seq=17/4352, ttl=252 (request in 443)
447	0.258173988	193.136.9.254	10.0.2.15	ICMP	528	Echo (ping) reply id=0x0003, seq=18/4608, ttl=252 (request in 444)
448	0.250167570	193.136.9.254	10.0.2.15	ICMP	528	Echo (ping) reply id=0x0003, seq=19/4864, ttl=252 (request in 445)
449	0.262458795	172.26.254.254	10.0.2.15	ICMP	70	Time to live exceeded (Time to live exceeded in transit)
450	0.262458795	172.26.254.254	10.0.2.15	ICMP	70	Time to live exceeded (Time to live exceeded in transit)
451	0.262458915	172.26.254.254	10.0.2.15	ICMP	70	Time to live exceeded (Time to live exceeded in transit)

Figura 2.11: Análise Wireshark

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

O identificador do pacote incrementa um valor a cada pacote enviado e o TTL incrementa um valor a cada 3 pacotes enviados, este último campo altera-se devido ao modo de funcionamento do traceroute.

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

Pela figura 2.12 é possível observar os pacotes ICMP *TTL Exceeded* a preto e, de acordo com o esperado, são nove pacotes dado o número de nodos intermédios ser três.

No.	Time	Source	Destination	Protocol	Length	Info
3	0.004017428	193.137.16.145	10.0.2.15	DNS	962	Standard query response 0x4331 A router-di.uninho.pt A 193.136.9.254 NS dns.uninho.pt NS dns2.uninho.pt NS dns3.uninho.pt
4	0.004030559	193.137.16.145	10.0.2.15	DNS	144	Standard query response 0x4329 AAAA router-di.uninho.pt SOA dns.uninho.pt OPT
14	0.032815930	10.0.2.2	10.0.2.15	ICMP	40	Time-to-live exceeded (time to live exceeded in transit)
16	0.032817852	10.0.2.2	10.0.2.15	ICMP	70	Time-to-live exceeded (time to live exceeded in transit)
15	0.032817971	10.0.2.2	10.0.2.15	ICMP	70	Time-to-live exceeded (time to live exceeded in transit)
22	0.044343005	193.137.16.145	10.0.2.15	DNS	150	Standard query response 0x4319 No such name PTR 2.2.0.10.in-addr.arpa SOA dns.uninho.pt OPT
24	0.044350219	172.16.2.1	10.0.2.15	ICMP	40	Time-to-live exceeded (time to live exceeded in transit)
26	0.044350267	172.16.2.1	10.0.2.15	ICMP	70	Time-to-live exceeded (time to live exceeded in transit)
29	0.044790319	172.16.2.1	10.0.2.15	ICMP	70	Time-to-live exceeded (time to live exceeded in transit)
30	0.044790345	172.16.254.254	10.0.2.15	ICMP	70	Time-to-live exceeded (time to live exceeded in transit)
31	0.044790370	172.16.254.254	10.0.2.15	ICMP	70	Time-to-live exceeded (time to live exceeded in transit)
32	0.045216478	193.136.9.254	10.0.2.15	ICMP	526	Echo (ping) reply id=0x0008, seq=1673560, ttl=252 (request in 1)
33	0.045216507	193.136.9.254	10.0.2.15	ICMP	526	Echo (ping) reply id=0x0008, seq=1673560, ttl=252 (request in 1)
34	0.045216559	193.136.9.254	10.0.2.15	ICMP	526	Echo (ping) reply id=0x0008, seq=1712816, ttl=252 (request in 18)
35	0.045216617	193.136.9.254	10.0.2.15	ICMP	526	Echo (ping) reply id=0x0008, seq=1712816, ttl=252 (request in 18)
36	0.045617945	193.136.9.254	10.0.2.15	ICMP	526	Echo (ping) reply id=0x0008, seq=1373328, ttl=252 (request in 29)
37	0.045618003	193.136.9.254	10.0.2.15	ICMP	526	Echo (ping) reply id=0x0008, seq=1413384, ttl=252 (request in 21)
38	0.045618058	193.136.9.254	10.0.2.15	ICMP	526	Echo (ping) reply id=0x0008, seq=1513840, ttl=252 (request in 22)
39	0.0456180614	193.136.9.254	10.0.2.15	ICMP	526	Echo (ping) reply id=0x0008, seq=1514096, ttl=252 (request in 23)
40	0.046031731	193.137.16.145	10.0.2.15	DNS	144	Standard query response 0x3519 No such name PTR 2.2.0.10.in-addr.arpa SOA dns.uninho.pt
43	0.087055141	193.137.16.145	10.0.2.15	DNS	152	Standard query response 0x3090 No such name PTR 254.254.26.172.in-addr.arpa SOA 26.172.IN-ADDR.ARPA OPT
47	0.088245166	193.136.9.254	10.0.2.15	ICMP	526	Echo (ping) reply id=0x0008, seq=174352, ttl=252 (request in 41)
48	0.088245248	193.136.9.254	10.0.2.15	ICMP	526	Echo (ping) reply id=0x0008, seq=1814680, ttl=252 (request in 42)
49	0.088245281	193.136.9.254	10.0.2.15	ICMP	526	Echo (ping) reply id=0x0008, seq=1914864, ttl=252 (request in 43)
50	0.091643363	193.137.16.145	10.0.2.15	DNS	141	Standard query response 0x3090 No such name PTR 254.254.26.172.in-addr.arpa SOA 26.172.IN-ADDR.ARPA

Figura 2.12: Análise Wireshark

**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?**

As mensagens de retorno com a mensagem *TTL Exceeded* apresentam um TTL com um valor que varia entre o 253-255. O valor não permanece constante porque o caminho utilizado desde a origem até ao destino pode variar, por consequente, o número de nodos varia e assim o valor do TTL final também varia.

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

Os valores de TTL são relativamente altos de modo a garantir que a mensagem de resposta chega ao destino. Se o valor do TTL fosse muito perto do valor real do número de nodos intermédios haveria o risco de a resposta ficar perdida pelo caminho caso houvesse ligeiras alterações na rede.

**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 inclusão das informações do cabeçalho ICMP no cabeçalho IPv4 é teoricamente possível, uma vez que ambos pertencem ao mesmo nível de rede. Isso poderia resultar numa redução do overhead, uma vez que a carga útil total do pacote seria menor, e simplificaria o processamento, já que as informações estariam contidas num único cabeçalho.

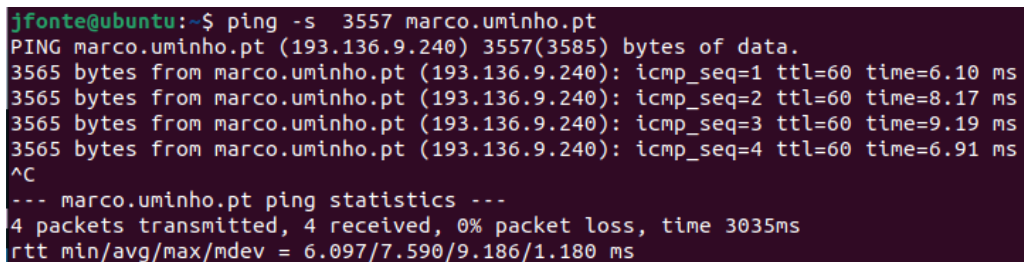
No entanto, há algumas desvantagens potenciais nessa abordagem. Primeiro, a inclusão de informações do ICMP no cabeçalho IPv4 poderia diminuir a flexibilidade do protocolo ICMP, uma vez que as funcionalidades adicionais do ICMP teriam que ser incorporadas ao cabeçalho IPv4, tornando-o mais complexo e difícil de ser estendido no futuro. Além disso, essa abordagem poderia enfrentar problemas de compatibilidade com dispositivos de rede existentes, uma vez que muitos dispositivos têm implementações específicas para o processamento do cabeçalho ICMP separadamente.

Outra desvantagem é que o cabeçalho IPv4 possui um tamanho fixo, enquanto o cabeçalho ICMP pode variar em tamanho, dependendo do tipo de mensagem ICMP. Isso poderia resultar em problemas de ajuste de tamanho do pacote, especialmente em casos de mensagens ICMP grandes, o que poderia levar a fragmentação de pacotes ou perda de informações.

Dessa forma, a inclusão das informações do cabeçalho ICMP no cabeçalho IPv4 pode ter algumas vantagens em termos de redução de overhead e simplificação do processamento, mas também pode apresentar desvantagens em termos de flexibilidade, compatibilidade e ajuste de tamanho do pacote.

## 2.3 Exercício 3

Estudo com o comando : "ping -s 3557 marco.uminho.pt"



```
jfonte@ubuntu:~$ ping -s 3557 marco.uminho.pt
PING marco.uminho.pt (193.136.9.240) 3557(3585) bytes of data.
3565 bytes from marco.uminho.pt (193.136.9.240): icmp_seq=1 ttl=60 time=6.10 ms
3565 bytes from marco.uminho.pt (193.136.9.240): icmp_seq=2 ttl=60 time=8.17 ms
3565 bytes from marco.uminho.pt (193.136.9.240): icmp_seq=3 ttl=60 time=9.19 ms
3565 bytes from marco.uminho.pt (193.136.9.240): icmp_seq=4 ttl=60 time=6.91 ms
^C
--- marco.uminho.pt ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3035ms
rtt min/avg/max/mdev = 6.097/7.590/9.186/1.180 ms
```

Figura 2.13: Resultados do ping

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

A primeira mensagem ICMP encontra-se na linha 3, logo depois da resolução DNS do domínio marco.uminho.pt.

Houve necessidade de fragmentar o pacote inicial porque o seu tamanho de 3557 é superior ao mtu definido (*maximum transmission unit*), que estabelece o tamanho máximo de um pacote sem que ocorra fragmentação.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.00000000	10.0.2.15	193.136.9.240	DNS	75	Standard query 0xb00f AAAA marco.uminho.pt
2	0.02072996	193.136.9.240	10.0.2.15	DNS	129	Standard query response 0xb00f AAAA marco.uminho.pt 50A dns.uminho.pt
3	0.03040059	10.0.2.15	193.136.9.240	ICMP	1514	Echo (ping) request id=0x0002, seq=1/256, ttl=64 (reply in 8)
4	0.03060252	10.0.2.15	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, offset=1488, ID=0x0002)
5	0.03060272	10.0.2.15	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, offset=1488, ID=0x0002)
6	0.06307097	193.136.9.240	10.0.2.15	ICMP	1514	Echo (ping) reply id=0x0002, seq=1/256, ttl=59 (request in 3)
7	0.06307136	193.136.9.240	10.0.2.15	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, offset=1488, ID=0x0002)
8	0.06307154	193.136.9.240	10.0.2.15	IPv4	639	Fragmented IP protocol (proto=ICMP 1, offset=2960, ID=0x0002)
9	0.03103460	10.0.2.15	193.136.9.240	ICMP	1514	Echo (ping) request id=0x0002, seq=2/512, ttl=64 (reply in 32)
10	1.03310223	10.0.2.15	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, offset=1488, ID=0x0002)
11	1.03320479	10.0.2.15	193.136.9.240	IPv4	639	Fragmented IP protocol (proto=ICMP 1, offset=2960, ID=0x0002)
12	1.05067177	193.136.9.240	10.0.2.15	ICMP	1514	Echo (ping) reply id=0x0002, seq=2/512, ttl=59 (request in 9)
13	1.05067182	193.136.9.240	10.0.2.15	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, offset=1488, ID=0x0002)
14	1.05067204	193.136.9.240	10.0.2.15	IPv4	639	Fragmented IP protocol (proto=ICMP 1, offset=2960, ID=0x0002)
15	2.05030770	10.0.2.15	193.136.9.240	ICMP	1514	Echo (ping) request id=0x0002, seq=3/768, ttl=64 (reply in 38)
16	2.05030813	10.0.2.15	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, offset=1488, ID=0x0002)
17	2.05030854	10.0.2.15	193.136.9.240	IPv4	639	Fragmented IP protocol (proto=ICMP 1, offset=2960, ID=0x0002)
18	2.05030858	193.136.9.240	10.0.2.15	ICMP	1514	Echo (ping) reply id=0x0002, seq=3/768, ttl=59 (request in 15)
19	2.05030862	193.136.9.240	10.0.2.15	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, offset=1488, ID=0x0002)
20	2.05030866	193.136.9.240	10.0.2.15	IPv4	639	Fragmented IP protocol (proto=ICMP 1, offset=2960, ID=0x0002)
21	3.05030866	10.0.2.15	193.136.9.240	ICMP	1514	Echo (ping) request id=0x0002, seq=4/1024, ttl=64 (reply in 24)
22	3.05030866	10.0.2.15	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, offset=1488, ID=0x0002)
23	3.05030866	10.0.2.15	193.136.9.240	IPv4	639	Fragmented IP protocol (proto=ICMP 1, offset=2960, ID=0x0002)
24	3.08490039	193.136.9.240	10.0.2.15	ICMP	1514	Echo (ping) reply id=0x0002, seq=4/1024, ttl=59 (request in 21)
25	3.08490043	193.136.9.240	10.0.2.15	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, offset=1488, ID=0x0002)
26	3.08490051	193.136.9.240	10.0.2.15	IPv4	639	Fragmented IP protocol (proto=ICMP 1, offset=2960, ID=0x0002)
27	4.04061703	10.0.2.15	193.136.9.240	ICMP	1514	Echo (ping) request id=0x0002, seq=5/1280, ttl=64 (reply in 30)
28	4.04061917	10.0.2.15	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, offset=1488, ID=0x0002)
29	4.04061977	10.0.2.15	193.136.9.240	IPv4	639	Fragmented IP protocol (proto=ICMP 1, offset=2960, ID=0x0002)
30	4.07219787	193.136.9.240	10.0.2.15	ICMP	1514	Echo (ping) reply id=0x0002, seq=5/1280, ttl=59 (request in 27)
31	4.07219807	193.136.9.240	10.0.2.15	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, offset=1488, ID=0x0002)
32	4.07219830	193.136.9.240	10.0.2.15	IPv4	639	Fragmented IP protocol (proto=ICMP 1, offset=2960, ID=0x0002)

Figura 2.14: Análise Wireshark

## 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?

Como mencionado no exercício 2.d), é possível consultar as flags de fragmentação no cabeçalho IP. Neste caso é possível ver na figura 2.15, que a flag *"More fragments"* está a *"Set"*, logo existe fragmentação do pacote, também é possível observar a flag do *"Fragment Offset"* a 0, o que indica tratar-se do primeiro pacote. O tamanho total deste datagrama é de 1500 bytes, e essa informação é possível confirmar pelo campo *"Total Length"* do cabeçalho IP.

```
> Frame 3: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface enp0s3, id 0
> Ethernet II, Src: PcsCompu_28:6c:57 (08:00:27:28:6c:57), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
  > Internet Protocol Version 4, Src: 10.0.2.15, Dst: 193.136.9.240
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
      Total Length: 1500
      Identification: 0x09ec (27116)
    > Flags: 0x20, 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: 64
      Protocol: ICMP (1)
      Header Checksum: 0x13ae [validation disabled]
      [Header checksum status: Unverified]
      Source Address: 10.0.2.15
      Destination Address: 193.136.9.240
    > Internet Control Message Protocol
      Type: 8 (Echo (ping) request)
      Code: 0
      Checksum: 0xe677 [unverified] [fragmented datagram]
      [Checksum Status: Unverified]
      Identifier (BE): 2 (0x0002)
      Identifier (LE): 512 (0x0200)
      Sequence Number (BE): 1 (0x0001)
      Sequence Number (LE): 256 (0x0100)
      [Response frame: 6]
      Timestamp from icmp data: Apr  2, 2023 16:47:29.000000000 BST
      [Timestamp from icmp data (relative): 0.952690627 seconds]
    > Data (1464 bytes)
```

Figura 2.15: 1º fragmento do pacote

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?

Observando a figura 2.16 é possível concluir que não se trata do 1º fragmento porque o *Fragment Offset* é maior que 0 (neste caso 1480) logo na junção dos pacotes existe um pacote antes deste, também sabemos que este não é o último fragmento pois observando o campo *More Fragments* está a "Set" e isso indica que existem mais pacotes depois deste fragmento.

```

> Frame 4: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface enp0s3, id 0
> Ethernet II, Src: PcsCompu_28:6c:57 (08:00:27:28:6c:57), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
> Internet Protocol Version 4, Src: 10.0.2.15, 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: 0x69ec (27116)
  > Flags: 0x20, More fragments
    0... .... = Reserved bit: Not set
    .0.. .... = Don't fragment: Not set
    ..1. .... = More fragments: Set
    ...0 0101 1100 1000 = Fragment Offset: 1480
  Time to Live: 64
  Protocol: ICMP (1)
  Header Checksum: 0x12f5 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 10.0.2.15
  Destination Address: 193.136.9.240
  > Data (1480 bytes)
    Data: c0c1c2c3c4c5c6c7c8c9cacbcccdcecf0d1d2d3d4d5d6d7d8d9dadbdcddeedfe0e1e2e3...
    [Length: 1480]

```

Figura 2.16: 2º fragmento do pacote

Como se pode confirmar pelo cabeçalho do terceiro e último pacote o campo " *More Fragments* está a "Not Set", logo não se espera mais fragmentos do datagrama original posteriores a este.

```

> Frame 5: 639 bytes on wire (5112 bits), 639 bytes captured (5112 bits) on interface enp0s3, id 0
> Ethernet II, Src: PcsCompu_28:6c:57 (08:00:27:28:6c:57), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
> Internet Protocol Version 4, Src: 10.0.2.15, 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: 625
  Identification: 0x69ec (27116)
  > Flags: 0x01
    0... .... = Reserved bit: Not set
    .0.. .... = Don't fragment: Not set
    ..0. .... = More fragments: Not set
    ...0 1011 1001 0000 = Fragment Offset: 2960
  Time to Live: 64
  Protocol: ICMP (1)
  Header Checksum: 0x35a7 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 10.0.2.15
  Destination Address: 193.136.9.240
  > Data (605 bytes)
    Data: 88898a8b8c8d8e8f909192939495969798999a9b9c9d9e9fa0a1a2a3a4a5a6a7a8a9aaab...
    [Length: 605]

```

Figura 2.17: 3º fragmento do pacote



**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.**

Teoricamente o limite máximo de cada pacote sem que ocorra fragmentação é de 1500 bytes (mtu = 1500), sabendo que o cabeçalho IP ocupa 20 bytes e o ICMP ocupa 8 bytes sobram 1472 bytes, e esse é o limite real de cada pacote IP/ICMP sem que ocorra fragmentação. Como apenas o primeiro fragmento do pacote IP/ICMP leva o cabeçalho ICMP, os restantes fragmentos tem um limite máximo de 1480 bytes.

Então, fazendo as contas sem contar cabeçalhos, temos um pacote de 3557 bytes que será dividido em 3 pacotes, o primeiro de 1472 bytes, o segundo de 1480 bytes e o terceiro de 605 bytes.

A estimativa teórica corresponde à prática como é possível ver pela figura 2.15, 2.16 e 2.17 .

**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 pode ser detetado a partir do campo *More fragments* a "Not Set", que significa que não tem mais pacotes a seguir, e o campo *Fragment Offset* diferente de zero, que indica que não é o primeiro fragmento. Desse modo, construi-se o filtro da figura 2.18 .

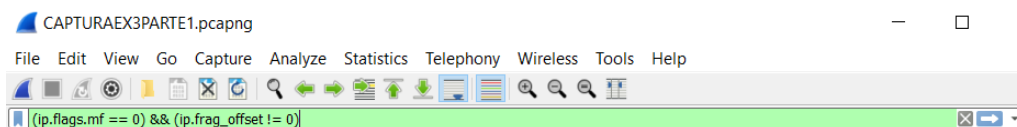


Figura 2.18: 3º fragmento do pacote

Como se pode ver pela figura 2.18 o terceiro e último pacote corresponde aos parâmetros estabelecidos.

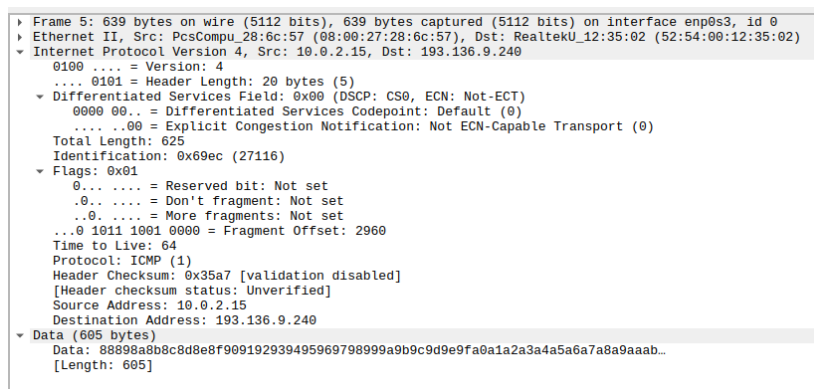


Figura 2.19: 3º fragmento do pacote



**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 equipamento responsável por reconstruir o datagrama IP original a partir dos fragmentos é o destino final do pacote. É neste equipamento que a camada de rede irá recolher todos os fragmentos recebidos e reagrupá-los, de forma a reconstruir o datagrama original.

Em teoria, é possível que a reconstrução ocorra em outro equipamento, desde que esteja configurado para receber todos os fragmentos do datagrama IP original. No entanto, na prática, o processo de reconstrução do datagrama é feito no destino final porque o objetivo da fragmentação é não congestionar a rede com pacotes muito grandes, e caso a união dos pacotes fosse feita num nodo intermédio a fragmentação perderia o seu propósito pois iria congestionar a última fase de transmissão. Deste modo, o destino é o único equipamento que tem a informação completa sobre o tamanho e as informações contidas no datagrama original, bem como sobre a identificação e a ordem dos fragmentos recebidos para assim poder reagrupar os diferentes fragmentos.

**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.**

Os campos que mudam entre os fragmentos são as *Flags*, "More fragments" e "Fragment Offset", que indicam se existe mais fragmentos do datagrama original e qual o *Offset* a partir do qual se agrupa este fragmento.

No caso em análise, o pacote de 3557 bytes é dividido em 3 fragmentos, com as seguintes flags no cabeçalho IP:

Identification: 0x69ec (27116) ▼ Flags: 0x20, More fragments 0... .. = Reserved bit: Not set .0... .. = Don't fragment: Not set ..1... .. = More fragments: Set ...0 0000 0000 0000 = Fragment Offset: 0	Identification: 0x69ec (27116) ▼ Flags: 0x20, More fragments 0... .. = Reserved bit: Not set .0... .. = Don't fragment: Not set ..1... .. = More fragments: Set ...0 0101 1100 1000 = Fragment Offset: 1480	Identification: 0x69ec (27116) ▼ Flags: 0x01 0... .. = Reserved bit: Not set .0... .. = Don't fragment: Not set ..0... .. = More fragments: Not set ...0 1011 1001 0000 = Fragment Offset: 2960
(a) 1º fragmento	(b) 2º fragmento	(c) 3º fragmento

Figura 2.20: Flags dos fragmentos

No destino o dispositivo identifica na flag "*More fragments*" do primeiro fragmento e sabe que se trata de um datagrama fragmentado logo terá de reagrupar os fragmentos.

O processo de junção de pacotes dá-se da seguinte forma: pelo campo "*Fragment Offset*" o dispositivo sabe que irá colocar a informação do fragmento da posição "*Fragment Offset*" → "*Fragment Offset*" + Payload . Depois, ao receber pacotes futuros identifica os que tem o mesmo ID (como é possível ver pelas três figuras todos tem o mesmo Id) e repete o mesmo processo até chegar o último fragmento com a flag "*More fragments*" a "Not Set".

No caso dos fragmentos da figura 2.15, a informação do 1º fragmento começa na posição 0 até ao tamanho do payload, a do 2º fragmento começa na posição 1480 até 1480 + Payload

e a do 3º fragmento começa na posição 2960 até 2960 + Payload. Tendo assim o pacote no destino com o 3557 bytes de informação inicial.

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

O primeiro fragmento está identificado como sendo ICMP porque no processo de fragmentação este é o único que contém o cabeçalho ICMP. O cabeçalho ICMP contém informação de controlo e não possui dados relevantes para a reconstrução do pacote no destino, deste modo no processo de fragmentação não se adiciona o cabeçalho ICMP nos restantes fragmentos para não acrescentar overhead no tamanho já limitado do pacote.

**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 do MTU é de 1500 bytes incluindo todos os cabeçalhos. Este valor é definido em muitos sistemas e foi determinado como sendo o valor ideal para a rede de modo a balancear os prós e contras.

Aumentar o valor do MTU permite transmissão de pacotes maiores e assim reduzir a latência e sobrecarga na rede, no entanto maiores pacotes estão sujeitos a mais erros de transmissão, o que leva a sua retransmissão a ser muito pesada na rede. Valores muito altos também podem trazer problemas de largura de banda para algumas partes da rede, o que inevitavelmente leva à congestão e sobrecarga de uma rede e inevitavelmente a atrasos e perda de pacotes, assim como, problemas gerais de latência e desempenho.

Pacotes muito pequenos têm o mesmo efeito negativo, a constante necessidade de fragmentação leva a um aumento exponencial do tráfego na rede, conduzindo aos mesmos problemas anteriormente referidos, como latência, perda de pacotes e mau desempenho geral.

j. Utilizando o comando "ping -M do -s SIZE marco.uminho.pt", determine o valor máximo de SIZE sem que ocorra fragmentação do pacote? Justifique o valor obtido.

O grupo tendo o conhecimento prévio de que o MTU definido era de 1500 bytes testou um valor inferior (1000 bytes) e um valor superior (1600 bytes). Como é possível observar pela figura 2.17 a rede aceita o valor inferior e rejeita o valor superior.

```
jfonte@ubuntu:~$ ping -M do -s 1000 marco.uminho.pt
PING marco.uminho.pt (193.136.9.240) 1000(1028) bytes of data:
1000 bytes from marco.uminho.pt (193.136.9.240): icmp_seq=1 ttl=60 time=14.4 ms
1000 bytes from marco.uminho.pt (193.136.9.240): icmp_seq=2 ttl=60 time=9.61 ms
1000 bytes from marco.uminho.pt (193.136.9.240): icmp_seq=3 ttl=60 time=8.92 ms
1000 bytes from marco.uminho.pt (193.136.9.240): icmp_seq=4 ttl=60 time=7.46 ms
1000 bytes from marco.uminho.pt (193.136.9.240): icmp_seq=5 ttl=60 time=4.86 ms
1000 bytes from marco.uminho.pt (193.136.9.240): icmp_seq=6 ttl=60 time=5.42 ms
^C1000 bytes from marco.uminho.pt (193.136.9.240): icmp_seq=7 ttl=60 time=7.59 ms
^C
--- marco.uminho.pt ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6049ms
rtt: min/avg/max/ndev = 4.862/8.319/14.378/2.936 ms
jfonte@ubuntu:~$ ping -M do -s 1600 marco.uminho.pt
PING marco.uminho.pt (193.136.9.240) 1600(1628) bytes of data.
ping: local error: message too long, mtu=1500
ping: local error: message too long, mtu=1500
ping: local error: message too long, mtu=1500
ping: local error: message too long, mtu=1500
```

Figura 2.21: Teste da Unidade Máxima de Transmissão (MTU)



## 3.1 Exercício 1

**Enunciado :** 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.**

Com os testes da figura 3.2 é possível confirmar que existe conectividade com o Institucional, nomeadamente as Finanças, e com a CDN, testando cada subrede presente.

```
<ycore.34167/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=0.050 ms
64 bytes from 192.168.0.250: icmp_seq=2 ttl=61 time=0.045 ms
64 bytes from 192.168.0.250: icmp_seq=3 ttl=61 time=0.099 ms
64 bytes from 192.168.0.250: icmp_seq=4 ttl=61 time=0.122 ms
64 bytes from 192.168.0.250: icmp_seq=5 ttl=61 time=0.047 ms
64 bytes from 192.168.0.250: icmp_seq=6 ttl=61 time=0.110 ms
64 bytes from 192.168.0.250: icmp_seq=7 ttl=61 time=0.096 ms
64 bytes from 192.168.0.250: icmp_seq=8 ttl=61 time=0.043 ms
^C
--- 192.168.0.250 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7166ms
rtt min/avg/max/mdev = 0.043/0.076/0.122/0.031 ms
root@AfonsoHenriques:/tmp/pycore.34167/AfonsoHenriques.conf#
```

(a) Ping ao Servidor Finanças

```
<ycore.34167/AfonsoHenriques.conf# ping 192.168.0.210
PING 192.168.0.210 (192.168.0.210) 56(84) bytes of data.
64 bytes from 192.168.0.210: icmp_seq=1 ttl=55 time=0.089 ms
64 bytes from 192.168.0.210: icmp_seq=2 ttl=55 time=0.077 ms
64 bytes from 192.168.0.210: icmp_seq=3 ttl=55 time=0.077 ms
64 bytes from 192.168.0.210: icmp_seq=4 ttl=55 time=0.077 ms
64 bytes from 192.168.0.210: icmp_seq=5 ttl=55 time=0.111 ms
64 bytes from 192.168.0.210: icmp_seq=6 ttl=55 time=0.077 ms
64 bytes from 192.168.0.210: icmp_seq=7 ttl=55 time=0.071 ms
64 bytes from 192.168.0.210: icmp_seq=8 ttl=55 time=0.075 ms
^C
--- 192.168.0.210 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7171ms
rtt min/avg/max/mdev = 0.071/0.081/0.111/0.012 ms
root@AfonsoHenriques:/tmp/pycore.34167/AfonsoHenriques.conf#
```

(b) Ping ao servidor Itunes

```
<ycore.34167/AfonsoHenriques.conf# ping 192.168.0.202
PING 192.168.0.202 (192.168.0.202) 56(84) bytes of data.
64 bytes from 192.168.0.202: icmp_seq=1 ttl=55 time=0.227 ms
64 bytes from 192.168.0.202: icmp_seq=2 ttl=55 time=0.227 ms
64 bytes from 192.168.0.202: icmp_seq=3 ttl=55 time=0.077 ms
64 bytes from 192.168.0.202: icmp_seq=4 ttl=55 time=0.224 ms
64 bytes from 192.168.0.202: icmp_seq=5 ttl=55 time=0.220 ms
64 bytes from 192.168.0.202: icmp_seq=6 ttl=55 time=0.146 ms
64 bytes from 192.168.0.202: icmp_seq=7 ttl=55 time=0.073 ms
64 bytes from 192.168.0.202: icmp_seq=8 ttl=55 time=0.072 ms
64 bytes from 192.168.0.202: icmp_seq=9 ttl=55 time=0.190 ms
64 bytes from 192.168.0.202: icmp_seq=10 ttl=55 time=0.181 ms
^C
--- 192.168.0.202 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9200ms
rtt min/avg/max/mdev = 0.072/0.163/0.227/0.063 ms
root@AfonsoHenriques:/tmp/pycore.34167/AfonsoHenriques.conf#
```

(c) Ping ao servidor Youtube

```
<ycore.34167/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=0.140 ms
64 bytes from 192.168.0.218: icmp_seq=2 ttl=55 time=0.090 ms
64 bytes from 192.168.0.218: icmp_seq=3 ttl=55 time=0.223 ms
64 bytes from 192.168.0.218: icmp_seq=4 ttl=55 time=0.241 ms
64 bytes from 192.168.0.218: icmp_seq=5 ttl=55 time=0.088 ms
64 bytes from 192.168.0.218: icmp_seq=6 ttl=55 time=0.170 ms
64 bytes from 192.168.0.218: icmp_seq=7 ttl=55 time=0.095 ms
64 bytes from 192.168.0.218: icmp_seq=8 ttl=55 time=0.151 ms
64 bytes from 192.168.0.218: icmp_seq=9 ttl=55 time=0.117 ms
64 bytes from 192.168.0.218: icmp_seq=10 ttl=55 time=0.105 ms
^C
--- 192.168.0.218 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9215ms
rtt min/avg/max/mdev = 0.088/0.142/0.241/0.052 ms
root@AfonsoHenriques:/tmp/pycore.34167/AfonsoHenriques.conf#
```

(d) Ping ao servidor Spotify

Figura 3.2: Testes de Conetividade ao Institucional e CDN

O teste de conetividade a D.Teresa, na Galiza, confirma os problemas de conetividade, com de cada pacote a retornar a mensagem de erro "*Destination Net Unreachable*".

```
<ycore.34167/AfonsoHenriques.conf# ping 192.168.0.194
PING 192.168.0.194 (192.168.0.194) 56(84) bytes of data.
From 10.0.0.29 icmp_seq=1 Destination Net Unreachable
From 10.0.0.29 icmp_seq=2 Destination Net Unreachable
From 10.0.0.29 icmp_seq=3 Destination Net Unreachable
From 10.0.0.29 icmp_seq=4 Destination Net Unreachable
^C
--- 192.168.0.194 ping statistics ---
5 packets transmitted, 0 received, +4 errors, 100% packet loss, time 4086ms
root@AfonsoHenriques:/tmp/pycore.34167/AfonsoHenriques.conf#
```

Figura 3.3: Ping ao PC Teresa

**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**

Analisando as tabelas de encaminhamento de AfonsoHenriques (figura 3.4 (a)) e Teresa (figura 3.4 (b)) não é detetado qualquer problema nas suas entradas. A utilidade das entradas é a mesma, tanto no AfonsoHenriques e Teresa, a primeira entrada é denominada de entrada *default* que redireciona qualquer endereço não listado na tabela para a interface do router de acesso da sua rede e, por consequente, para fora da rede local, a segunda entrada é utilizada quando o endereço destino pertence à sua subrede local, por isso, ele enviará diretamente os pacotes para o dispositivo correspondente.

```
root@AfonsoHenriques:/tmp/pycore.34167/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.34167/AfonsoHenriques.conf#
```

(a) AfonsoHenriques

```
root@Teresa:/tmp/pycore.34167/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.34167/Teresa.conf#
```

(b) D.Teresa

Figura 3.4: Tabelas de Encaminhamento

**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.**

Analisando a topologia da rede o grupo tem como hipótese que a não conectividade entre os dois host deve-se às tabelas de encaminhamento dos routers intermédios, desse modo até se verificar a conectividade entre AfonsoHenriques e D.Teresa, o método iterativo adotado foi, a cada iteração executar os seguintes passos :

- traceroute AfonsoHenriques → D.Teresa, se o tráfego chegar ao ISP CondadOnline dar o exercício por resolvido.
- Identificar o router intermédio com erros
- Analisar a tabela de encaminhamento do router
- Corrigir erros na tabela através do comando `ip route [add/del/change]`

## 1ª Iteração - Router n5

Em primeiro lugar executa-se um traceroute AfonsoHenriques → D.Teresa. Pela figura 3.5 verifica-se que o primeiro router com problemas é o router n5 pois o último salto registado é para o endereço IP : 10.0.0.29.

```
<core.34167/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)  0.222 ms  0.193 ms  0.183 ms
 2 172.16.143.1 (172.16.143.1)  0.174 ms  0.156 ms  0.145 ms
 3 10.0.0.29 (10.0.0.29)  0.135 ms IN  0.116 ms IN *
```

Figura 3.5: 1ª Traceroute a Teresa

No.	Time	Source	Destination	Protocol	Length	Info
17	14.005629274	10.0.0.29	224.0.0.5	OSPF	82	Hello Packet
18	14.005908823	10.0.0.30	224.0.0.5	OSPF	82	Hello Packet
19	16.007018972	10.0.0.30	224.0.0.5	OSPF	82	Hello Packet
20	16.007134187	10.0.0.29	224.0.0.5	OSPF	82	Hello Packet
21	17.226298125	fe80::200:ff:feaa:24	ff02::5	OSPF	94	Hello Packet
22	17.702293398	fe80::200:ff:feaa:25	ff02::5	OSPF	94	Hello Packet
23	18.007511722	10.0.0.30	224.0.0.5	OSPF	82	Hello Packet
24	18.007600374	10.0.0.29	224.0.0.5	OSPF	82	Hello Packet
25	20.008256841	10.0.0.30	224.0.0.5	OSPF	82	Hello Packet
26	20.008642650	10.0.0.29	224.0.0.5	OSPF	82	Hello Packet
27	20.110751544	192.168.0.226	192.168.0.194	UDP	74	51567 → 33448 Len=32
28	20.110757908	10.0.0.29	192.168.0.226	ICMP	102	Destination unreachable (Network unreachable)
29	20.110772601	192.168.0.226	192.168.0.194	UDP	74	37447 → 33441 Len=32
30	20.110775235	10.0.0.29	192.168.0.226	ICMP	102	Destination unreachable (Network unreachable)
31	20.110788135	192.168.0.226	192.168.0.194	UDP	74	51567 → 33442 Len=32
32	20.110799175	192.168.0.226	192.168.0.194	UDP	74	48555 → 33443 Len=32
33	20.110809427	192.168.0.226	192.168.0.194	UDP	74	48124 → 33444 Len=32
34	20.110819858	192.168.0.226	192.168.0.194	UDP	74	37846 → 33445 Len=32
35	20.110830735	192.168.0.226	192.168.0.194	UDP	74	45166 → 33446 Len=32
36	20.110841086	192.168.0.226	192.168.0.194	UDP	74	43433 → 33447 Len=32
37	20.110851541	192.168.0.226	192.168.0.194	UDP	74	51828 → 33448 Len=32
38	20.110862539	192.168.0.226	192.168.0.194	UDP	74	59746 → 33449 Len=32

Figura 3.6: Wireshark - Captura de tráfego em N5

A captura de tráfego do Wireshark no router n5 comprova que o pacotes ICMP retornam com a mensagem de erro "*Destination Unreachable*".

Numa análise às entradas tabela de encaminhamento (figura 3.7 (a)) verifica-se que, de facto, não existe nenhuma entrada que direcione o tráfego endereçado à sub-rede da Galiza. Desse modo tratou-se de adicionar uma entrada referente a esse tráfego. Sendo os IPs da subrede da Galiza 192.168.0.194/28, 192.168.0.195/28 e 192.168.0.196/28, o grupo calculou que o endereço IP da subrede Galiza é 192.168.0.192/28.

Assim sendo, executou-se o seguinte comando :

```
ip route add 192.168.0.192/29 via 10.0.0.25
```

Que significa que todo o tráfego para a subrede 192.168.0.192/28 (Galiza) é direccionado para 10.0.0.25 (router n2). Isto resulta numa nova tabela de endereçamento (figura 3.7 (b)), onde se verifica a nova entrada.

## 2ª Iteração - Router n2

Na 2ª iteração executa-se novamente o traceroute (figura 3.8) e verifica-se que o tráfego fica "preso" em 10.0.0.25 (Router n2).

Analisando a tabela de encaminhamento (figura 3.9) verifica-se que existe duas entradas para a rede Galiza, uma com destino 192.168.0.192(Galiza) e outra 192.168.0.194 (Teresa). A entrada para Teresa está incorreta por três motivos,

```
root@ns:/tmp/pqcore,34167/n5.conf# netstat -rn
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	eth0
10.0.0.8	10.0.0.25	255.255.255.252	UG	0 0	0	eth0
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	UG	0 0	0	eth1
10.0.0.28	0.0.0.0	255.255.255.252	UG	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	10.0.0.30	255.255.255.248	UG	0 0	0	eth0
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
172.142.0.4	10.0.0.25	255.255.255.252	UG	0 0	0	eth0
132.168.0.200	10.0.0.25	255.255.255.248	UG	0 0	0	eth1
132.168.0.208	10.0.0.25	255.255.255.248	UG	0 0	0	eth1
132.168.0.216	10.0.0.25	255.255.255.248	UG	0 0	0	eth1
132.168.0.224	10.0.0.30	255.255.255.248	UG	0 0	0	eth0
132.168.0.232	10.0.0.30	255.255.255.248	UG	0 0	0	eth0
132.168.0.240	10.0.0.30	255.255.255.248	UG	0 0	0	eth0
132.168.0.248	10.0.0.30	255.255.255.248	UG	0 0	0	eth0

```
root@ns:/tmp/pqcore,34167/n5.conf#
```

(a) Antiga

```
root@ns:/tmp/picore.34167/n5.conf# ip route add 192.168.0.192/29 via 10.0.0.25 dev eth1
root@ns:/tmp/picore.34167/n5.conf# netstat -rn
Kernel IP routing table
Destination        Gateway            Genmask           Flags        MSS Window  irtt Iface
0.0.0.0            0.0.0.0           0.0.0.0           UG           0 0 0      eth1
10.0.0.0           10.0.0.0           0.0.0.0           UG           0 0 0      eth1
10.0.0.4           10.0.0.0           255.255.255.252   UG           0 0 0      eth1
10.0.0.8           10.0.0.0           255.255.255.252   UG           0 0 0      eth1
10.0.0.12          10.0.0.0           255.255.255.252   UG           0 0 0      eth1
10.0.0.16          10.0.0.0           255.255.255.252   UG           0 0 0      eth1
10.0.0.20          10.0.0.0           255.255.255.252   UG           0 0 0      eth1
10.0.0.24          0.0.0.0            255.255.255.252   UG           0 0 0      eth1
10.0.0.28          0.0.0.0            255.255.255.252   UG           0 0 0      eth0
172.0.0.0          10.0.0.0           255.0.0.0         UG           0 0 0      eth0
172.0.0.4         10.0.0.0           255.0.0.0         UG           0 0 0      eth1
172.16.142.0      10.0.0.0           255.255.255.248   UG           0 0 0      eth1
172.16.143.0      10.0.0.0           255.255.255.252   UG           0 0 0      eth0
172.16.143.4      10.0.0.0           255.255.255.248   UG           0 0 0      eth0
172.16.143.4      10.0.0.0           255.255.255.252   UG           0 0 0      eth1
192.0.0.0          0.0.0.0            255.0.0.0         UG           0 0 0      eth1
192.168.0.192     10.0.0.0           255.255.255.248   UG           0 0 0      eth1
192.168.0.200     10.0.0.0           255.255.255.248   UG           0 0 0      eth1
192.168.0.208     10.0.0.0           255.255.255.248   UG           0 0 0      eth1
192.168.0.216     10.0.0.0           255.255.255.248   UG           0 0 0      eth1
192.168.0.224     10.0.0.0           255.255.255.248   UG           0 0 0      eth0
192.168.0.232     10.0.0.0           255.255.255.248   UG           0 0 0      eth0
192.168.0.240     10.0.0.0           255.255.255.248   UG           0 0 0      eth0
192.168.0.248     10.0.0.0           255.255.255.248   UG           0 0 0      eth1
```

(b) Nova

Figura 3.7: Tabela de Encaminhamento do Router N5

```
traceroute to 192.168.0.194 (192.168.0.194), 30 hops max, 60 byte packets
 1 192.168.0.225 (192.168.0.225)  0,041 ms  0,008 ms  0,007 ms
 2 172.16.143.1 (172.16.143.1)  0,018 ms  0,011 ms  0,010 ms
 3 10.0.0.29 (10.0.0.29)  0,023 ms  0,014 ms  0,014 ms
 4 10.0.0.25 (10.0.0.25)  0,031 ms  0,019 ms  0,018 ms
 5 10.0.0.25 (10.0.0.25)  3071.380 ms IH 3071.347 ms IH 3071.325 ms IH
```

Figura 3.8: 2º Traceroute a Teresa

- redireciona o tráfego para 10.0.0.25 (criando o loop)
- a máscara de rede está errada, é /31 em vez de /29
- viola aos princípios de agregação de rotas.

Deste modo o grupo considera que a melhor abordagem é eliminá-la.

Assim sendo, executou-se o seguinte comando :

```
ip route del 192.168.0.194/31 via 10.0.0.25
```

Como a entrada para o IP 192.168.0.192/28 está correta mais nenhuma alteração foi feita.

```
root@bz2:~# tcpdump -i eth0 netstat -rn
Kernel IP routing table

```

Destination	Gateway	Genmask	Flags	MSS window	irtt	iface
0.0.0.0	0.0.0.0	255.255.255.255	U	0 0	0	eth0
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	10.0.0.15	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
172.16.142.0	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
172.16.142.8	10.0.0.13	255.255.255.252	UG	0 0	0	eth1
172.16.143.4	10.0.0.26	255.255.255.252	UG	0 0	0	eth2
192.168.0.132	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	10.0.0.25	255.255.255.254	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

```
root@bz2:~# tcpdump -i eth0 netstat -rn
```

(a) Antiga

```

22.conf# ip route del 192.168.0.134/31 -- 0.0.0.25 dev eth2
root@n2:~# tcpdump -i tap.pcap.34167/n2.conf# metstat -rrn
Kernel IP routing table

```

Destination	Gateway	Genmask	Flags	MSS	Window	irtt	ifac
0.0.0.0	0.0.0.13	255.255.255.252	U	0	0	0	eth1
0.0.0.0	0.0.0.21	255.255.255.252	U	0	0	0	eth0
0.0.0.0	0.0.0.13	255.255.255.252	UG	0	0	0	eth1
0.0.0.12	0.0.0.0	255.255.255.252	U	0	0	0	eth1
0.0.0.16	0.0.0.13	255.255.255.252	U	0	0	0	eth1
0.0.0.20	0.0.0.0	255.255.255.252	U	0	0	0	eth0
0.0.0.24	0.0.0.0	255.255.255.252	U	0	0	0	eth2
0.0.0.28	0.0.0.25	255.255.255.252	UG	0	0	0	eth2
0.0.0.32	0.0.0.25	255.255.255.252	UG	0	0	0	eth2
172.16.142.0	0.0.0.13	255.255.255.252	UG	0	0	0	eth1
172.16.142.4	0.0.0.21	255.255.255.252	UG	0	0	0	eth0
172.16.143.0	0.0.0.26	255.255.255.252	UG	0	0	0	eth2
172.16.143.4	0.0.0.26	255.255.255.252	UG	0	0	0	eth2
192.168.0.132	0.0.0.13	255.255.255.248	U	0	0	0	eth1
192.168.0.200	0.0.0.21	255.255.255.248	U	0	0	0	eth0
192.168.0.208	0.0.0.21	255.255.255.248	UG	0	0	0	eth0
192.168.0.216	0.0.0.21	255.255.255.248	UG	0	0	0	eth0
192.168.0.224	0.0.0.26	255.255.255.248	UG	0	0	0	eth2
192.168.0.232	0.0.0.26	255.255.255.248	UG	0	0	0	eth2
192.168.0.240	0.0.0.26	255.255.255.248	UG	0	0	0	eth2
192.168.0.248	0.0.0.26	255.255.255.248	UG	0	0	0	eth2

(b) Nova

Figura 3.9: Tabela de Encaminhamento do Router N5



### 3ª Iteração - Router n1

Voltando a executar o traceroute para Teresa verifica-se problemas no direcionamento do router 10.0.0.13 (Router n1) que, ao que parece, induz o tráfego num loop entre n1 (10.0.0.13) e n2 (10.0.0.25).

```
<4167/AfonsoHenriques.conf# traceroute 192.168.0.194
traceroute to 192.168.0.194 (192.168.0.194), 30 hops max, 60 byte packets
 1 192.168.0.225 (192.168.0.225)  0,040 ms  0,009 ms  0,007 ms
 2 172.16.143.1 (172.16.143.1)  0,020 ms  0,124 ms  0,040 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)  0,024 ms  0,018 ms  0,027 ms
 5 10.0.0.13 (10.0.0.13)  0,027 ms  0,022 ms  0,022 ms
 6 10.0.0.25 (10.0.0.25)  0,022 ms  0,017 ms  0,012 ms
 7 10.0.0.13 (10.0.0.13)  0,013 ms  0,013 ms  0,013 ms
 8 ***
 9 ***
10 ***
11 ***
12 ***
13 * 10.0.0.13 (10.0.0.13)  0,168 ms  0,095 ms
14 10.0.0.25 (10.0.0.25)  0,092 ms  0,089 ms  0,088 ms
15 10.0.0.13 (10.0.0.13)  0,098 ms  0,099 ms  0,097 ms
16 10.0.0.25 (10.0.0.25)  0,097 ms  0,098 ms *
17 ***
18 ***
19 ***
20 ***
21 ***
22 * 10.0.0.25 (10.0.0.25)  0,179 ms  0,110 ms
23 10.0.0.13 (10.0.0.13)  0,114 ms  0,110 ms  0,109 ms
24 10.0.0.25 (10.0.0.25)  0,110 ms  0,110 ms  0,109 ms
25 10.0.0.13 (10.0.0.13)  0,116 ms  0,116 ms *C
root@AfonsoHenriques:/tmp/pycore.34167/AfonsoHenriques.conf#
```

Figura 3.10: 3ª Traceroute a Teresa

Obseevando a tabela de encaminhamento do Router N1 (figura 3.11) percebe-se que o tráfego para a Galiza (192.168.0.192) está a ser direcionado para o router n2 (10.0.0.14), desse modo a direção tem de ser alterada para o Router n3 (10.0.0.9).

Assim sendo, executou-se o seguinte comando :

```
ip route change 192.168.0.192/29 via 10.0.0.9
```

O comando muda o gateway da rota 192.168.0.192/29 para o IP 10.0.0.9 .

```
root@n1:/tmp/pycore.34167/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
192.168.0.132 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.34167/n1.conf#
```

(a) Antiga

```
<l1.conf# ip route change 192.168.0.192/29 via 10.0.0.9 dev eth0
root@n1:/tmp/pycore.34167/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 10.0.0.9 255.255.255.248 UG 0 0 0 eth0
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.34167/n1.conf#
```

(b) Nova

Figura 3.11: Tabela de Encaminhamento do Router N1

#### 4ª Iteração - Router

Na quarta e última iteração, após o traceroute verifica-se que o tráfego chega a 10.0.0.1 (ISP CondadOnline) se erros, assim sendo, segundo o enunciado dá-se esta alínea por concluída.

```
<pycore.34167/AfonsoHenriques.conf# traceroute 192.168.0.192
traceroute to 192.168.0.192 (192.168.0.192), 30 hops max, 60 byte packets
 1 192.168.0.225 (192.168.0.225) 0.302 ms 0.272 ms 0.262 ms
 2 172.16.143.1 (172.16.143.1) 0.253 ms 0.236 ms 0.224 ms
 3 10.0.0.29 (10.0.0.29) 0.212 ms 0.193 ms 0.180 ms
 4 10.0.0.25 (10.0.0.25) 0.168 ms 0.148 ms 0.133 ms
 5 10.0.0.13 (10.0.0.13) 0.118 ms 0.097 ms 0.079 ms
 6 10.0.0.17 (10.0.0.17) 0.063 ms 0.335 ms 0.312 ms
 7 10.0.0.5 (10.0.0.5) 0.293 ms 0.267 ms 0.247 ms
 8 10.0.0.1 (10.0.0.1) 0.227 ms 0.201 ms 0.179 ms
 9 ***
10 ***
11 ***
12 ***
13 ***
14 ***
15 ***
16 ***
17 ***
18 ***
19 *^X^C
root@AfonsoHenriques:/tmp/pycore.34167/AfonsoHenriques.conf#
```

Figura 3.12: 4º Traceroute a Teresa

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.

Num primeiro teste de conectividade entre AfonsoHenriques e Teresa, verifica-se pela figura 3.13 (b) que a conexão no sentido AfonsoHenriques → Teresa está funcional, isto porque os pacotes *ICMP-request* chegam a Teresa. No entanto, o ping da figura 3.13 (a) regista 100% *packet loss*, o que significa que no sentido contrário, Teresa → AfonsoHenriques, a conexão está a falhar - a suspeita confirma-se observando o pacotes *ICMP-Reply*, que retornam a Teresa com origem em RAGaliza, com a mensagem "*Destination Unreachable*".

```
<4167/AfonsoHenriques.conf# ping 192.168.0.194
PING 192.168.0.194 (192.168.0.194) 56(84) bytes of data.
^C
--- 192.168.0.194 ping statistics ---
12 packets transmitted, 0 received, 100% packet loss, time 11257ms
```

(a) Ping de AfonsoHenriques a Teresa

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.0.192	224.0.0.5	OSPF	78	Hello Packet
2	1.320021124	192.168.0.226	192.168.0.194	ICMP	98	Echo (ping) request id=0x008f, seq=1/256, ttl=55 (reply in 3)
3	1.320033441	192.168.0.194	192.168.0.226	ICMP	98	Echo (ping) reply id=0x008f, seq=1/256, ttl=64 (request in 2)
4	1.320043531	192.168.0.193	192.168.0.194	ICMP	120	Destination unreachable (Network unreachable)
5	2.000010512	192.168.0.193	224.0.0.5	OSPF	78	Hello Packet
6	2.336056047	192.168.0.226	192.168.0.194	ICMP	98	Echo (ping) request id=0x008f, seq=2/512, ttl=55 (reply in 7)
7	2.336070374	192.168.0.194	192.168.0.226	ICMP	98	Echo (ping) reply id=0x008f, seq=2/512, ttl=64 (request in 6)
8	2.336087699	192.168.0.193	192.168.0.194	ICMP	120	Destination unreachable (Network unreachable)
9	3.363021559	192.168.0.226	192.168.0.194	ICMP	98	Echo (ping) request id=0x008f, seq=3/768, ttl=55 (reply in 10)
10	3.363035948	192.168.0.194	192.168.0.226	ICMP	98	Echo (ping) reply id=0x008f, seq=3/768, ttl=64 (request in 9)
11	3.363048570	192.168.0.193	192.168.0.194	ICMP	120	Destination unreachable (Network unreachable)
12	4.000007348	192.168.0.193	224.0.0.5	OSPF	78	Hello Packet
13	4.384013223	192.168.0.226	192.168.0.194	ICMP	98	Echo (ping) request id=0x008f, seq=4/1024, ttl=55 (reply in 14)
14	4.384033198	192.168.0.194	192.168.0.226	ICMP	98	Echo (ping) reply id=0x008f, seq=4/1024, ttl=64 (request in 13)
15	4.384050599	192.168.0.193	192.168.0.194	ICMP	120	Destination unreachable (Network unreachable)
16	5.408446875	192.168.0.226	192.168.0.194	ICMP	98	Echo (ping) request id=0x008f, seq=5/1280, ttl=55 (reply in 17)
17	5.408455631	192.168.0.194	192.168.0.226	ICMP	98	Echo (ping) reply id=0x008f, seq=5/1280, ttl=64 (request in 16)
18	6.001158915	192.168.0.193	224.0.0.5	OSPF	78	Hello Packet

(b) Wireshark - Tráfego capturado em Teresa

Figura 3.13: Teste de Conetividade AfonsoHenriques ↔ 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.

As razões para continuar a não existir conectividade entre Teresa e AfonsoHenriques devem-se possivelmente ao encaminhamento de pacotes no sentido Teresa → AfonsoHenriques. Para resolver esta situação utiliza-se o método anteriormente mencionado.

Como foi mencionado na alínea anterior, os pacotes *ICMP-reply* com mensagem de erro "Destination Unreachable" tem como origem o IP 192.168.0.193 (RAGaliza) logo este deve ser o primeiro a ser investigado.

Pela tabela de encaminhamento da figura 3.14 (a) verifica-se que não há direcionamento para o tráfego destinado ao Condado Portucalense, com IP 192.168.0.224/28. O tratamento deste tráfego deve ser direcionado para o gateway 172.16.142.1.

Assim sendo, executou-se o seguinte comando :

```
ip route add 192.168.0.224/29 via 172.16.142.1
```

```
root@RAGaliza:/tmp/pycore,34167/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
```

(a) Antiga

```
root@RAGaliza.conf# ip route add 192.168.0.224/29 via 172.16.142.1 dev eth0
root@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
192.168.0.224 172.16.142.1 255.255.255.248 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
```

(b) Nova

Figura 3.14: Tabela de Encaminhamento do Router RAGaliza

Fazendo o mesmo teste do início, verifica-se que o ping da figura 3.15 (a) já funciona e que o tráfego capturado pelo wireshark da figura 3.15(b) já não apresenta pacotes com erros.

```
core,34167/AfonsoHenriques.conf# ping 192.168.0.194
PING 192.168.0.194 (192.168.0.194) 56(84) bytes of data,
64 bytes from 192.168.0.194: icmp_seq=1 ttl=55 time=0.110 ms
64 bytes from 192.168.0.194: icmp_seq=2 ttl=55 time=1.00 ms
64 bytes from 192.168.0.194: icmp_seq=3 ttl=55 time=0.095 ms
64 bytes from 192.168.0.194: icmp_seq=4 ttl=55 time=0.148 ms
64 bytes from 192.168.0.194: icmp_seq=5 ttl=55 time=0.083 ms
64 bytes from 192.168.0.194: icmp_seq=6 ttl=55 time=0.107 ms
64 bytes from 192.168.0.194: icmp_seq=7 ttl=55 time=0.232 ms
64 bytes from 192.168.0.194: icmp_seq=8 ttl=55 time=0.151 ms
^C
--- 192.168.0.194 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7127ms
rtt min/avg/max/mdev = 0.083/0.241/1.004/0.291 ms
root@AfonsoHenriques.conf#
```

(a) Ping a Teresa

13.16.80864027	192.168.0.193	224.0.0.5	OSPF	78 Hello Packet
14.18.80864029	192.168.0.193	224.0.0.5	OSPF	78 Hello Packet
15.18.552217318	192.168.0.226	192.168.0.194	ICMP	88 Echo (ping) request id=0x0088, seq=1/256, ttl=55 (reply in 16)
16.18.552224448	192.168.0.194	192.168.0.226	ICMP	88 Echo (ping) reply id=0x0088, seq=1/256, ttl=55 (reply in 15)
17.18.56770712	192.168.0.226	192.168.0.194	ICMP	88 Echo (ping) request id=0x0088, seq=2/512, ttl=55 (reply in 16)
18.18.56781818	192.168.0.194	192.168.0.226	ICMP	88 Echo (ping) reply id=0x0088, seq=2/512, ttl=55 (reply in 17)
19.18.618454208	192.168.0.193	224.0.0.5	OSPF	78 Hello Packet
20.20.56805739	192.168.0.226	192.168.0.194	ICMP	88 Echo (ping) request id=0x0088, seq=3/768, ttl=55 (reply in 21)
21.20.56805763	192.168.0.194	192.168.0.226	ICMP	88 Echo (ping) reply id=0x0088, seq=3/768, ttl=55 (reply in 20)
22.21.58357468	192.168.0.226	192.168.0.194	ICMP	88 Echo (ping) request id=0x0088, seq=4/1024, ttl=55 (reply in 23)
23.21.583583848	192.168.0.194	192.168.0.226	ICMP	88 Echo (ping) reply id=0x0088, seq=4/1024, ttl=55 (reply in 22)
24.22.61172021	192.168.0.193	224.0.0.5	OSPF	78 Hello Packet
25.22.60400228	192.168.0.226	192.168.0.194	ICMP	88 Echo (ping) request id=0x0088, seq=5/1280, ttl=55 (reply in 26)
26.22.60400888	192.168.0.194	192.168.0.226	ICMP	88 Echo (ping) reply id=0x0088, seq=5/1280, ttl=55 (reply in 25)
27.22.61877327	192.168.0.194	192.168.0.226	ICMP	88 Hello Packet
28.23.631777528	192.168.0.226	192.168.0.194	ICMP	88 Echo (ping) request id=0x0088, seq=6/1536, ttl=55 (reply in 29)
29.23.63187328	192.168.0.194	192.168.0.226	ICMP	88 Echo (ping) reply id=0x0088, seq=6/1536, ttl=55 (reply in 28)
30.24.61238447	192.168.0.193	224.0.0.5	OSPF	78 Hello Packet
31.24.605482978	192.168.0.226	192.168.0.194	ICMP	88 Echo (ping) request id=0x0088, seq=7/1792, ttl=55 (reply in 32)
32.24.605508961	192.168.0.194	192.168.0.226	ICMP	88 Echo (ping) reply id=0x0088, seq=7/1792, ttl=55 (reply in 31)
33.25.679211784	192.168.0.226	192.168.0.194	ICMP	88 Echo (ping) request id=0x0088, seq=8/2048, ttl=55 (reply in 34)
34.25.679229299	192.168.0.194	192.168.0.226	ICMP	88 Echo (ping) reply id=0x0088, seq=8/2048, ttl=55 (reply in 33)
35.25.67927027	192.168.0.193	224.0.0.5	OSPF	78 Hello Packet

(b) Wireshark - Tráfego capturado em Teresa

Figura 3.15: Teste de Conetividade AfonsoHenriques ↔ Teresa

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

Executando o traceroute nos dois sentidos, tem-se o seguinte resultado :

```
<gcore.34167/AfonsoHenriques.conf# traceroute 192.168.0.194
traceroute to 192.168.0.194 (192.168.0.194), 30 hops max, 60 byte packets
 1 192.168.0.226 (192.168.0.226) 0.356 ms 0.328 ms 0.318 ms
 2 172.16.143.1 (172.16.143.1) 0.309 ms 0.272 ms 0.246 ms
 3 10.0.0.29 (10.0.0.29) 0.221 ms 0.187 ms 0.174 ms
 4 10.0.0.25 (10.0.0.25) 0.161 ms 0.142 ms 0.128 ms
 5 10.0.0.13 (10.0.0.13) 0.114 ms 0.094 ms 0.079 ms
 6 10.0.0.17 (10.0.0.17) 0.061 ms 0.492 ms 0.470 ms
 7 10.0.0.5 (10.0.0.5) 0.452 ms 0.428 ms 0.020 ms
 8 10.0.0.1 (10.0.0.1) 0.021 ms 0.017 ms 0.016 ms
 9 172.16.142.2 (172.16.142.2) 0.024 ms 0.018 ms 0.019 ms
10 192.168.0.194 (192.168.0.194) 0.025 ms 0.020 ms 0.020 ms
```

(a) Traceroute AfonsoHenriques → Teresa

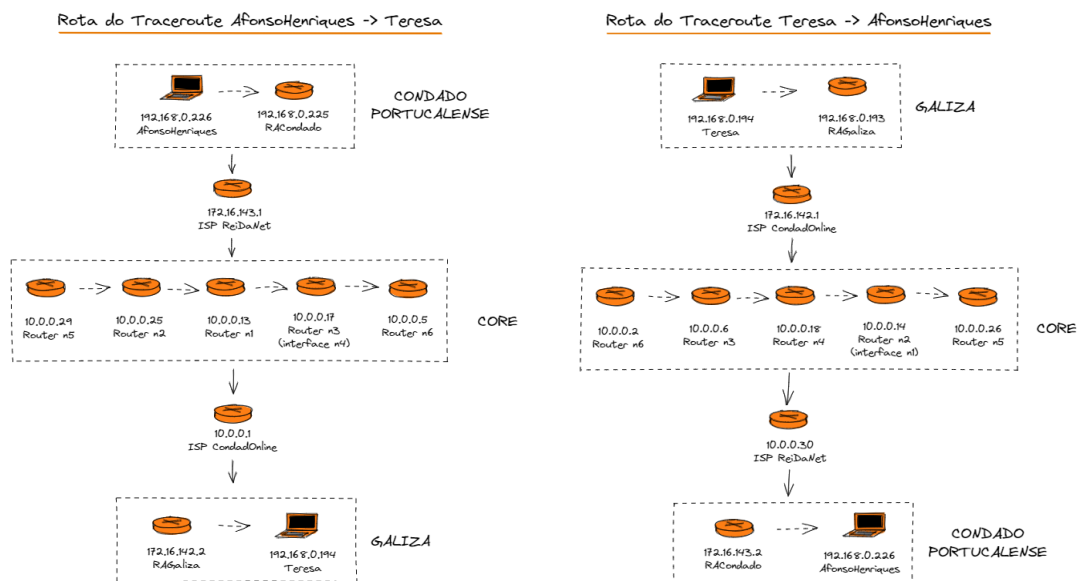
```
root@Teresa:/tmp/pycore.34167/Teresa.conf# traceroute 192.168.0.226
traceroute to 192.168.0.226 (192.168.0.226), 30 hops max, 60 byte packets
 1 192.168.0.193 (192.168.0.193) 0.034 ms 0.008 ms 0.007 ms
 2 172.16.142.1 (172.16.142.1) 0.019 ms 0.011 ms 0.010 ms
 3 10.0.0.2 (10.0.0.2) 0.022 ms 0.014 ms 0.023 ms
 4 10.0.0.6 (10.0.0.6) 0.024 ms 0.017 ms 0.016 ms
 5 10.0.0.18 (10.0.0.18) 0.026 ms 0.021 ms 0.020 ms
 6 10.0.0.14 (10.0.0.14) 0.034 ms 0.036 ms 0.025 ms
 7 10.0.0.26 (10.0.0.26) 0.035 ms 0.027 ms 0.027 ms
 8 10.0.0.30 (10.0.0.30) 0.037 ms 0.033 ms 0.032 ms
 9 172.16.143.2 (172.16.143.2) 0.041 ms 0.035 ms 0.036 ms
10 192.168.0.226 (192.168.0.226) 0.047 ms 0.040 ms 0.039 ms
```

(b) Traceroute Teresa → AfonsoHenriques

Figura 3.16: Traceroute AfonsoHenriques ↔ Teresa

Os diagramas da figura 3.17 ilustram rotas dos pacotes em cada sentido. Como é possível observar, tanto num sentido como no outro, existe um router que comprova que as rotas do reply e do request são diferentes - na figura 3.17 (a) é o Router n3 e na figura 3.17 (b) é o Router n2. Isto porque os pacotes seguem os caminhos das tabelas de endereçamento, no entanto, no caso de um router ter mais de uma rota, o caminho num sentido pode não ser o mesmo no outro sentido, e esse é o caso do router n3 e n2.

No caso da figura 3.17 (a) o traceroute envia o pacote ICMP Echo Request e chega ao router n3 do route n1. Ao ser enviado de volta pelo n3, a tabela de encaminhamento envia-o para o router n4 que ao chegar ao AfonsoHenriques registra a origem do pacote. Vemos assim que o caminho do request (n1→n3) foi diferente do reply (n4→n4).



(a) Caminho AfonsoHenriques → Teresa

(b) Caminho Teresa → AfonsoHenriques

Figura 3.17: Caminhos AfonsoHenriques ↔ Teresa

e. Estando restabelecida a conectividade entre os dois hosts, obtenha a tabela de encaminhamento de n3 e foque-se na seguinte entrada:

192.168.0.192	10.0.0.18	255.255.255.240	UG	0 0	0 eth1
---------------	-----------	-----------------	----	-----	--------

**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.**

Para o polo CDN não existe correspondência porque os pacotes direcionados para o CDN utilizam um ip a partir de 192.168.0.200, e a entrada só dá match para a subrede 192.168.0.192.

Para o polo Galiza existe correspondência, no entanto, o SO não reencaminha para este router porque existe uma entrada que também tem correspondência e com melhor métricas, logo esta entrada nunca é utilizada. A entrada com correspondência equivalente é:

192.168.0.192	10.0.0.5	255.255.255.248	UG	0 0	0 eth2
---------------	----------	-----------------	----	-----	--------

**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 são todos privados pois estão de acordo com o protocolo RFC1918, que estabelece regras de endereçamento para endereços privados.

O protocolo RFC1918 estabelece 3 grupos de endereços privados:

- bloco 192.168.0.0 - 192.168.255.255 /16
- bloco 172.16.0.0 - 172.31.255.255/12
- bloco 10.0.0.0 - 10.255.255.255 /8

As redes utilizadas nos quatro polos estão inseridas no primeiro bloco, as dos ISP estão inseridas no segundo bloco e as do core no terceiro bloco.

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

Não, os switches são dispositivos que operam na camada de dados do modelo OSI e estão responsáveis por encaminhar pacotes entre diferentes interfaces de rede com base no endereço MAC de destino. Como os switches não precisam de operar na camada de rede não implementam o protocolo IP logo não tem um IP a si associado.

## 3.2 Exercício 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.

Executando o comando "*netstat -rn*" observa-se que o router Castelo tem a seguinte tabela de encaminhamento:

```
root@Castelo:/tmp/pycore.38491/Castelo.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
```

Figura 3.18: Tabela de Encaminhamento de Castelo

Retirando a rota default, ou seja, a primeira linha da tabela, a nova tabela fica com o seguinte aspeto:

```
root@Castelo:/tmp/pycore.38491/Castelo.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
192.168.0.224 0.0.0.0 255.255.255.248 U 0 0 0 eth0
```

Figura 3.19: Tabela de Encaminhamento de Castelo sem rota default

De modo à conectividade aos polos ser restabelecida tem de se adicionar a rota específica para cada subrede de cada polo, assim sendo serão adicionadas as seguintes entradas:

- Polo Galiza : 192.168.0.192/29
- CDN - subrede1 : 192.168.0.200/29
- CDN - subrede2 : 192.168.0.208/29
- CDN - subrede3 : 192.168.0.216/29
- Instutucional - subrede1 : 192.168.0.232/29
- Instutucional - subrede2 : 192.168.0.240/29
- Instutucional - subrede3 : 192.168.0.248/29

Adicionando cada entrada com o comando:

```
ip route add [IP da subrede] via 192.168.0.225
```

E assim obtém-se a nova tabela de encaminhamento :

```

root@Castelo:/tmp/pycore.38491/Castelo.conf# netstat -rn
Kernel IP routing table
Destination        Gateway           Genmask          Flags   MSS Window  irtt Iface
192.168.0.192      192.168.0.225    255.255.255.248 UG        0 0        0 eth0
192.168.0.200      192.168.0.225    255.255.255.248 UG        0 0        0 eth0
192.168.0.208      192.168.0.225    255.255.255.248 UG        0 0        0 eth0
192.168.0.216      192.168.0.225    255.255.255.248 UG        0 0        0 eth0
192.168.0.224      0.0.0.0          255.255.255.248 U        0 0        0 eth0
192.168.0.232      192.168.0.225    255.255.255.248 UG        0 0        0 eth0
192.168.0.240      192.168.0.225    255.255.255.248 UG        0 0        0 eth0
192.168.0.248      192.168.0.225    255.255.255.248 UG        0 0        0 eth0

```

Figura 3.20: Nova Tabela de Encaminhamento de Castelo

```

root@Castelo:/tmp/pycore.38491/Castelo.conf# ping 192.168.0.234
PING 192.168.0.234 (192.168.0.234) 56(84) bytes of data:
64 bytes from 192.168.0.234: icmp_seq=1 ttl=61 time=0.104 ms
64 bytes from 192.168.0.234: icmp_seq=2 ttl=61 time=0.050 ms
64 bytes from 192.168.0.234: icmp_seq=3 ttl=61 time=0.052 ms
^C
64 bytes from 192.168.0.234: icmp_seq=4 ttl=61 time=0.054 ms
^C
--- 192.168.0.234 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 305ms
rtt min/avg/max/mdev = 0.050/0.065/0.104/0.022 ms

```

(a) Ping a UMinho

```

root@Castelo:/tmp/pycore.38491/Castelo.conf# ping 192.168.0.242
PING 192.168.0.242 (192.168.0.242) 56(84) bytes of data:
64 bytes from 192.168.0.242: icmp_seq=1 ttl=61 time=0.092 ms
64 bytes from 192.168.0.242: icmp_seq=2 ttl=61 time=0.085 ms
64 bytes from 192.168.0.242: icmp_seq=3 ttl=61 time=0.084 ms
^C
--- 192.168.0.242 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 204ms
rtt min/avg/max/mdev = 0.084/0.087/0.092/0.003 ms

```

(b) Ping ao DI

```

root@Castelo:/tmp/pycore.38491/Castelo.conf# ping 192.168.0.252
PING 192.168.0.252 (192.168.0.252) 56(84) bytes of data:
64 bytes from 192.168.0.252: icmp_seq=1 ttl=61 time=0.121 ms
64 bytes from 192.168.0.252: icmp_seq=2 ttl=61 time=0.187 ms
64 bytes from 192.168.0.252: icmp_seq=3 ttl=61 time=0.101 ms
^C
--- 192.168.0.252 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 204ms
rtt min/avg/max/mdev = 0.101/0.136/0.187/0.036 ms

```

(c) Ping a Educação

Figura 3.21: Testes de conectividade ao Institucional

```

root@Castelo:/tmp/pycore.38491/Castelo.conf# ping 192.168.0.210
PING 192.168.0.210 (192.168.0.210) 56(84) bytes of data:
64 bytes from 192.168.0.210: icmp_seq=1 ttl=55 time=0.170 ms
64 bytes from 192.168.0.210: icmp_seq=2 ttl=55 time=0.140 ms
64 bytes from 192.168.0.210: icmp_seq=3 ttl=55 time=0.098 ms
^C
--- 192.168.0.210 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 204ms
rtt min/avg/max/mdev = 0.098/0.136/0.170/0.029 ms

```

(a) Ping ao Itunes

```

root@Castelo:/tmp/pycore.38491/Castelo.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=0.184 ms
64 bytes from 192.168.0.218: icmp_seq=2 ttl=55 time=0.108 ms
64 bytes from 192.168.0.218: icmp_seq=3 ttl=55 time=0.156 ms
^C
--- 192.168.0.218 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 204ms
rtt min/avg/max/mdev = 0.108/0.149/0.184/0.031 ms

```

(b) Ping ao Spotify

```

root@Castelo:/tmp/pycore.38491/Castelo.conf# ping 192.168.0.202
PING 192.168.0.202 (192.168.0.202) 56(84) bytes of data:
64 bytes from 192.168.0.202: icmp_seq=1 ttl=55 time=0.133 ms
64 bytes from 192.168.0.202: icmp_seq=2 ttl=55 time=0.271 ms
64 bytes from 192.168.0.202: icmp_seq=3 ttl=55 time=0.236 ms
^C
--- 192.168.0.202 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 204ms
rtt min/avg/max/mdev = 0.133/0.213/0.271/0.068 ms

```

(c) Ping ao Youtube

Figura 3.22: Testes de conectividade à CDN

```

root@Castelo:/tmp/pycore.38491/Castelo.conf# ping 192.168.0.194
PING 192.168.0.194 (192.168.0.194) 56(84) bytes of data:
64 bytes from 192.168.0.194: icmp_seq=1 ttl=55 time=0.383 ms
64 bytes from 192.168.0.194: icmp_seq=2 ttl=55 time=0.257 ms
64 bytes from 192.168.0.194: icmp_seq=3 ttl=55 time=0.160 ms
^C
--- 192.168.0.194 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 204ms
rtt min/avg/max/mdev = 0.160/0.266/0.383/0.091 ms

```

Figura 3.23: Teste de conectividade à Galiza

Testando a conectividade com os polos, verifica-se que está restabelecida a conexão:

Verifica-se assim que conectividade está restabelecida, no entanto, a rota default ser eliminada de uma tabela de encaminhamento, deixa uma tarefa muito importante sem qualquer tratamento. Sem rota default, os casos em que a rota indicada ao router não está especificada na tabela o router não sabe para onde direcionar o tráfego e portanto dá erro.

Um teste que demonstra estes casos é um ping um endereço não especificado na tabela, por exemplo, um teste de conexão ao ISP ReiDaNet:

```
root@Castelo:/tmp/pycore.38491/Castelo.conf# ping 172.16.143.1
PING 172.16.143.1 (172.16.143.1) 56(84) bytes of data:
64 bytes from 172.16.143.1: icmp_seq=1 ttl=63 time=0.060 ms
64 bytes from 172.16.143.1: icmp_seq=2 ttl=63 time=0.127 ms
64 bytes from 172.16.143.1: icmp_seq=3 ttl=63 time=0.048 ms
^C
--- 172.16.143.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2025ms
rtt min/avg/max/ndev = 0.048/0.078/0.127/0.034 ms
```

(a) Com rota default

```
root@Castelo:/tmp/pycore.38491/Castelo.conf# ping 172.16.143.1
ping: connect: Network is unreachable
```

(b) Sem rota default

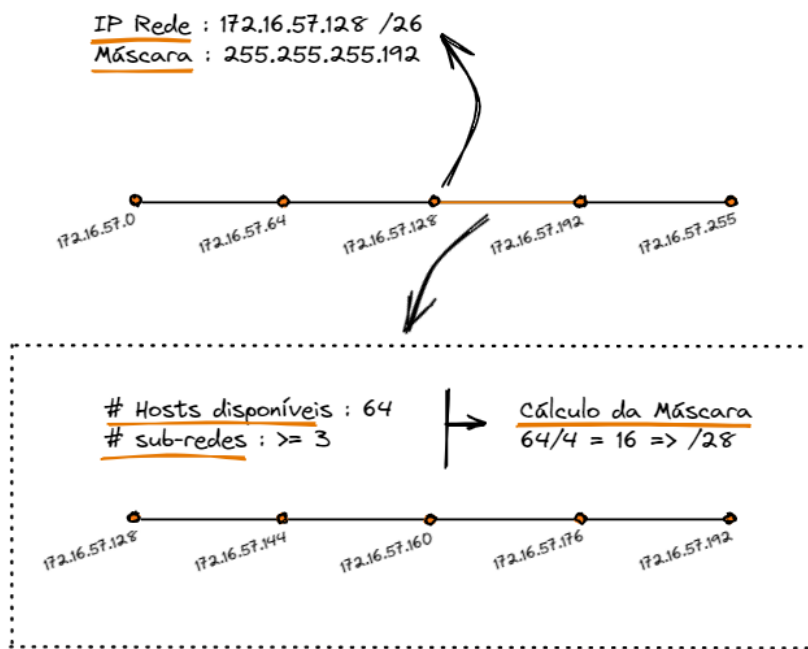
Figura 3.24: Testes de conectividade ao ISP ReiDaNet

Desta forma, é possível concluir que uma rota default é fundamental para tratar tráfego não especificado na tabela de encaminhamento.

**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**

Como o número do grupo é o 57, o endereço atribuído pelo ISP foi o 172.16.57.128/26, e assim se calculou espaços de endereçamento requeridos como demonstrado na figura 3.25. De seguida, foi adicionado à topologia o polo Braga (figura 3.26) com três novos hosts nas três primeiras novas subredes. De notar que, se no futuro fosse necessário adicionar mais hosts a cada subrede, teria-se de adicionar um switch a cada ligação para dividir a subrede.





Nome	IP da rede	Máscara	Range de hosts	#Hosts	IP de broadcast
Rede 1	172.16.57.128	/28	172.16.57.129 $\Rightarrow$ 172.16.57.142	14	172.16.57.143
Rede 2	172.16.57.144	/28	172.16.57.145 $\Rightarrow$ 172.16.57.158	14	172.16.57.159
Rede 3	172.16.57.160	/28	172.16.57.161 $\Rightarrow$ 172.16.57.174	14	172.16.57.175
Rede 4	172.16.57.176	/28	172.16.57.177 $\Rightarrow$ 172.16.57.190	14	172.16.57.191

Figura 3.25: Cálculo das Subredes

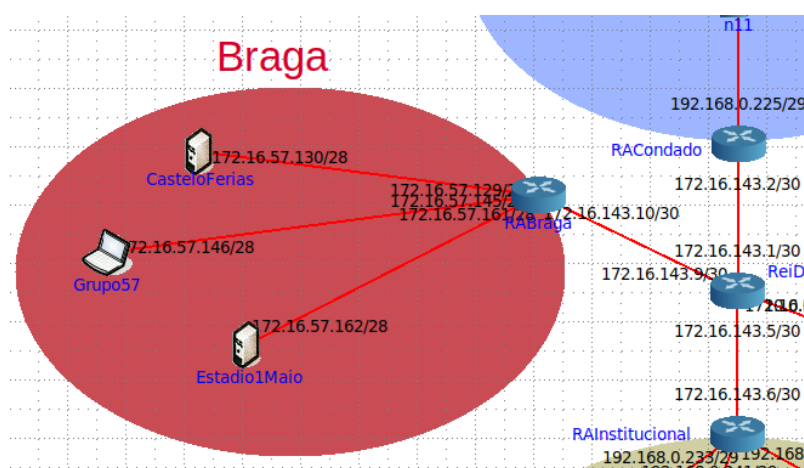


Figura 3.26: Novo Polo Braga

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?

O novo host, denominado "DJ8" foi adicionado à topologia, dentro da subrede 4 (com o IP: 172.16.57.178/28) e ligado diretamente ao ISP ReiDaNet, ficando a topologia com o seguinte aspeto:

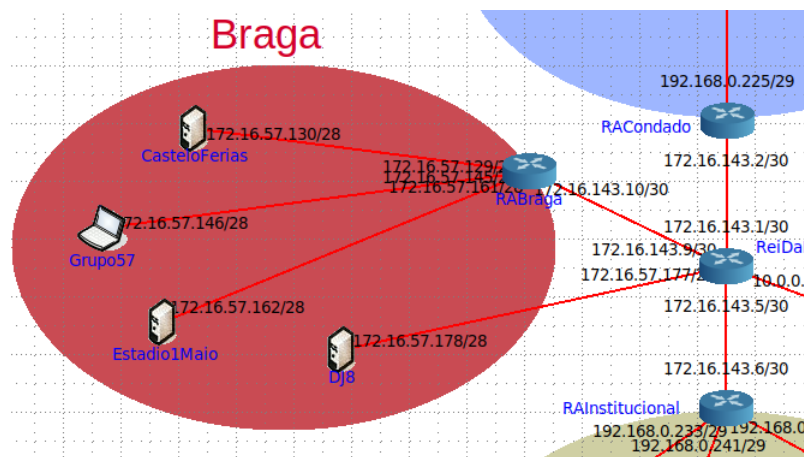


Figura 3.27: Novo host DJ8

Para testar a conectividade executou-se um ping a todos os polos. Os testes ao polo Condado-Portugalense, Galiza e CDN funcionam normalmente, no entanto, o teste aos hosts de Braga não funcionam, isto porque o host apesar de pertencer à subrede de Braga não se encontra na mesma LAN logo está dependente da tabela de encaminhamento do ISP ReiDaNet.

Verificando a tabela de encaminhamento do ISP ReiDaNet na figura 3.28, ainda não existe qualquer entrada para direcionar tráfego para Braga (172.16.57.128 ↔ 172.16.57.191).

Desta forma tem de se adicionar os seguintes entradas, para serem direcionadas para o gateway 172.16.143.10:

- Braga - Subrede1 : 172.16.57.128/28
- Braga - Subrede2 : 172.16.57.144/28
- Braga - Subrede3 : 172.16.57.160/28

```

root@ReiDaNet:/tmp/pycore.38491/ReiDaNet.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
10.0.0.0 10.0.0.23 255.255.255.252 UG 0 0 0 eth0
10.0.0.4 10.0.0.23 255.255.255.252 UG 0 0 0 eth0
10.0.0.8 10.0.0.23 255.255.255.252 UG 0 0 0 eth0
10.0.0.12 10.0.0.23 255.255.255.252 UG 0 0 0 eth0
10.0.0.16 10.0.0.23 255.255.255.252 UG 0 0 0 eth0
10.0.0.20 10.0.0.23 255.255.255.252 UG 0 0 0 eth0
10.0.0.24 10.0.0.23 255.255.255.252 UG 0 0 0 eth0
10.0.0.28 0.0.0.0 255.255.255.252 U 0 0 0 eth0
172.16.57.176 0.0.0.0 255.255.255.240 U 0 0 0 eth4
172.16.142.0 10.0.0.23 255.255.255.252 UG 0 0 0 eth0
172.16.143.0 0.0.0.0 255.255.255.252 U 0 0 0 eth1
172.16.143.4 0.0.0.0 255.255.255.252 U 0 0 0 eth2
172.16.143.8 0.0.0.0 255.255.255.252 U 0 0 0 eth3
192.142.0.4 10.0.0.23 255.255.255.252 UG 0 0 0 eth0
192.168.0.192 10.0.0.23 255.255.255.248 UG 0 0 0 eth0
192.168.0.200 10.0.0.23 255.255.255.248 UG 0 0 0 eth0
192.168.0.208 10.0.0.23 255.255.255.248 UG 0 0 0 eth0
192.168.0.216 10.0.0.23 255.255.255.248 UG 0 0 0 eth0
192.168.0.224 172.16.143.2 255.255.255.248 UG 0 0 0 eth1
192.168.0.232 172.16.143.6 255.255.255.248 UG 0 0 0 eth2
192.168.0.240 172.16.143.6 255.255.255.248 UG 0 0 0 eth2
192.168.0.248 172.16.143.6 255.255.255.248 UG 0 0 0 eth2

```

(a) Tabela Antiga

```

root@ReiDaNet:/tmp/pycore.38491/ReiDaNet.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
10.0.0.0 10.0.0.23 255.255.255.252 UG 0 0 0 eth0
10.0.0.4 10.0.0.23 255.255.255.252 UG 0 0 0 eth0
10.0.0.8 10.0.0.23 255.255.255.252 UG 0 0 0 eth0
10.0.0.12 10.0.0.23 255.255.255.252 UG 0 0 0 eth0
10.0.0.16 10.0.0.23 255.255.255.252 UG 0 0 0 eth0
10.0.0.20 10.0.0.23 255.255.255.252 UG 0 0 0 eth0
10.0.0.24 10.0.0.23 255.255.255.252 UG 0 0 0 eth0
10.0.0.28 0.0.0.0 255.255.255.252 U 0 0 0 eth0
172.16.57.128 172.16.143.10 255.255.255.240 UG 0 0 0 eth3
172.16.57.144 172.16.143.10 255.255.255.240 UG 0 0 0 eth3
172.16.57.160 172.16.143.10 255.255.255.240 UG 0 0 0 eth3
172.16.57.176 0.0.0.0 255.255.255.240 U 0 0 0 eth4
172.16.142.0 10.0.0.23 255.255.255.252 UG 0 0 0 eth0
172.16.143.0 0.0.0.0 255.255.255.252 U 0 0 0 eth1
172.16.143.4 0.0.0.0 255.255.255.252 U 0 0 0 eth2
172.16.143.8 0.0.0.0 255.255.255.252 U 0 0 0 eth3
192.142.0.4 10.0.0.23 255.255.255.252 UG 0 0 0 eth0
192.168.0.192 10.0.0.23 255.255.255.248 UG 0 0 0 eth0
192.168.0.200 10.0.0.23 255.255.255.248 UG 0 0 0 eth0
192.168.0.208 10.0.0.23 255.255.255.248 UG 0 0 0 eth0
192.168.0.216 10.0.0.23 255.255.255.248 UG 0 0 0 eth0
192.168.0.224 172.16.143.2 255.255.255.248 UG 0 0 0 eth1
192.168.0.232 172.16.143.6 255.255.255.248 UG 0 0 0 eth2
192.168.0.240 172.16.143.6 255.255.255.248 UG 0 0 0 eth2
192.168.0.248 172.16.143.6 255.255.255.248 UG 0 0 0 eth2

```

(b) Tabela Nova

Figura 3.28: Tabelas de Encaminhamento do ISP ReiDaNet

Testando a conectividade na figura 3.30, verifica-se que já é possível estabelecer conexão com todos os hosts de Braga.

```

root@DJ8:/tmp/pycore.38491/DJ8.conf# ping 172.16.57.130
PING 172.16.57.130 (172.16.57.130) 56(84) bytes of data.
64 bytes from 172.16.57.130: icmp_seq=1 ttl=62 time=0.052 ms
64 bytes from 172.16.57.130: icmp_seq=2 ttl=62 time=0.109 ms
64 bytes from 172.16.57.130: icmp_seq=3 ttl=62 time=0.117 ms
64 bytes from 172.16.57.130: icmp_seq=4 ttl=62 time=0.107 ms
64 bytes from 172.16.57.130: icmp_seq=5 ttl=62 time=0.107 ms
^C64 bytes from 172.16.57.130: icmp_seq=6 ttl=62 time=0.107 ms
^C
--- 172.16.57.130 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5101ms
rtt min/avg/max/mdev = 0.052/0.099/0.117/0.021 ms
root@DJ8:/tmp/pycore.38491/DJ8.conf# ping 172.16.57.146
PING 172.16.57.146 (172.16.57.146) 56(84) bytes of data.
64 bytes from 172.16.57.146: icmp_seq=1 ttl=62 time=0.049 ms
64 bytes from 172.16.57.146: icmp_seq=2 ttl=62 time=0.107 ms
64 bytes from 172.16.57.146: icmp_seq=3 ttl=62 time=0.170 ms
64 bytes from 172.16.57.146: icmp_seq=4 ttl=62 time=0.071 ms
^C
--- 172.16.57.146 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3062ms
rtt min/avg/max/mdev = 0.049/0.099/0.170/0.045 ms
root@DJ8:/tmp/pycore.38491/DJ8.conf# ping 172.16.57.162
PING 172.16.57.162 (172.16.57.162) 56(84) bytes of data.
64 bytes from 172.16.57.162: icmp_seq=1 ttl=62 time=0.050 ms
64 bytes from 172.16.57.162: icmp_seq=2 ttl=62 time=0.107 ms
64 bytes from 172.16.57.162: icmp_seq=3 ttl=62 time=0.108 ms
64 bytes from 172.16.57.162: icmp_seq=4 ttl=62 time=0.102 ms
^C
--- 172.16.57.162 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3073ms
rtt min/avg/max/mdev = 0.050/0.091/0.108/0.024 ms
root@DJ8:/tmp/pycore.38491/DJ8.conf#

```

Figura 3.29: Testes de Conetividade a Braga

### 3.3 Exercício 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.

Procedeu-se à eliminação das rotas da Galiza e do CDN no router n6, as rotas a eliminar são:

- Galiza : 192.168.0.192/29
- CDN - subrede1 : 192.168.0.200/29
- CDN - subrede2 : 192.168.0.208/29
- CDN - subrede3 : 192.168.0.216/29

```
root@n6:/tmp/pycore.38491/n6.conf# netstat -rn
```

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

(a) Tabela Antiga

```
root@n6:/tmp/pycore.38491/n6.conf# netstat -rn
```

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

(b) Tabela depois da eliminação

Figura 3.30: Tabelas de Encaminhamento do Router n6

Para fazer agregação das rotas da Galiza e do CDN, recorre-se à técnica de supernetting, resultando num IP geral de 192.168.0.192/27, como é possível ver pela figura 3.31.

#### SuperNetting da Galiza e CDN

	Endereço IP	Máscara
192.168.0.192/29	1100 0000 . 1010 1000 . 0000 0000 . 1100 0000	1100 0000
192.168.0.200/29	1100 0000 . 1010 1000 . 0000 0000 . 1100 1000	1100 1000
192.168.0.208/29	1100 0000 . 1010 1000 . 0000 0000 . 1101 0000	1101 0000
192.168.0.216/29	1100 0000 . 1010 1000 . 0000 0000 . 1101 1000	1101 1000
	↓	↓
	192.168.0.192	/27

Figura 3.31: Cálculo do Supernetting da Galiza e CDN

Assim sendo, como é possível observar na figura 3.32, adicionou-se a nova entrada à tabela de encaminhamento (a) e testou-se a conectividade entre o router UMinho e o polo Galiza e CDN (b):

```
root@n6:/tmp/pycore.38491/n6.conf# netstat -rn
```

Destination	Gateway	Genmask	Flags	MSS Window	irrt	Iface
10.0.0.0	0.0.0.0	255.255.255.252	U	0 0	0	eth0
10.0.0.4	0.0.0.0	255.255.255.252	U	0 0	0	eth1
10.0.0.8	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
10.0.0.12	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
10.0.0.16	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
10.0.0.20	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
10.0.0.24	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
10.0.0.28	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
172.0.0.0	10.0.0.6	255.0.0.0	UG	0 0	0	eth1
172.16.142.0	10.0.0.1	255.255.255.252	UG	0 0	0	eth0
172.16.142.4	10.0.0.1	255.255.255.252	UG	0 0	0	eth0
172.16.143.0	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
172.16.143.4	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
192.168.0.132	10.0.0.1	255.255.255.224	UG	0 0	0	eth0
192.168.0.224	10.0.0.6	255.255.255.248	UG	0 0	0	eth1
192.168.0.232	10.0.0.6	255.255.255.248	UG	0 0	0	eth1
192.168.0.240	10.0.0.6	255.255.255.248	UG	0 0	0	eth1
192.168.0.248	10.0.0.6	255.255.255.248	UG	0 0	0	eth1

(a) Nova Tabela com Supernetting

```
root@Uminho:/tmp/pycore.38491/Uminho.conf# ping 192.168.0.202
PING 192.168.0.202 (192.168.0.202) 56(84) bytes of data:
64 bytes from 192.168.0.202: icmp_seq=1 ttl=55 time=0.334 ms
64 bytes from 192.168.0.202: icmp_seq=2 ttl=55 time=0.309 ms
64 bytes from 192.168.0.202: icmp_seq=3 ttl=55 time=0.123 ms
64 bytes from 192.168.0.202: icmp_seq=4 ttl=55 time=0.219 ms
64 bytes from 192.168.0.202: icmp_seq=5 ttl=55 time=0.209 ms
^C
--- 192.168.0.202 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4086ms
rtt min/avg/max/ndev = 0.123/0.238/0.334/0.075 ms
root@Uminho:/tmp/pycore.38491/Uminho.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=0.136 ms
64 bytes from 192.168.0.218: icmp_seq=2 ttl=55 time=0.176 ms
64 bytes from 192.168.0.218: icmp_seq=3 ttl=55 time=0.077 ms
64 bytes from 192.168.0.218: icmp_seq=4 ttl=55 time=0.078 ms
^C
--- 192.168.0.218 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3054ms
rtt min/avg/max/ndev = 0.077/0.116/0.176/0.041 ms
root@Uminho:/tmp/pycore.38491/Uminho.conf# ping 192.168.0.194
PING 192.168.0.194 (192.168.0.194) 56(84) bytes of data:
64 bytes from 192.168.0.194: icmp_seq=1 ttl=55 time=0.161 ms
64 bytes from 192.168.0.194: icmp_seq=2 ttl=55 time=0.228 ms
64 bytes from 192.168.0.194: icmp_seq=3 ttl=55 time=0.224 ms
64 bytes from 192.168.0.194: icmp_seq=4 ttl=55 time=0.088 ms
^C
--- 192.168.0.194 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3063ms
rtt min/avg/max/ndev = 0.088/0.175/0.228/0.056 ms
```

(b) Testes de Conetividade

Figura 3.32: Router n6 com Supernetting

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

O mesmo processo foi executado para as rotas do CondadoPortugalense e Institucional no router n6. Procedeu-se à eliminação das suas rotas, sendo estas:

- CondadoPortugalense : 192.168.0.224/29
- Institucional - subrede1 : 192.168.0.232/29
- Institucional - subrede2 : 192.168.0.240/29
- Institucional - subrede3 : 192.168.0.248/29

```
root@n6:/tmp/pycore.38491/n6.conf# netstat -rn
```

Destination	Gateway	Genmask	Flags	MSS Window	irrt	Iface
10.0.0.0	0.0.0.0	255.255.255.252	U	0 0	0	eth0
10.0.0.4	0.0.0.0	255.255.255.252	U	0 0	0	eth1
10.0.0.8	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
10.0.0.12	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
10.0.0.16	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
10.0.0.20	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
10.0.0.24	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
10.0.0.28	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
172.0.0.0	10.0.0.6	255.0.0.0	UG	0 0	0	eth1
172.16.142.0	10.0.0.1	255.255.255.252	UG	0 0	0	eth0
172.16.142.4	10.0.0.1	255.255.255.252	UG	0 0	0	eth0
172.16.143.0	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
192.168.0.132	10.0.0.1	255.255.255.192	UG	0 0	0	eth0
192.168.0.224	10.0.0.6	255.255.255.248	UG	0 0	0	eth1
192.168.0.232	10.0.0.6	255.255.255.248	UG	0 0	0	eth1
192.168.0.240	10.0.0.6	255.255.255.248	UG	0 0	0	eth1
192.168.0.248	10.0.0.6	255.255.255.248	UG	0 0	0	eth1

(a) Tabela Antiga

```
root@n6:/tmp/pycore.38491/n6.conf# netstat -rn
```

Destination	Gateway	Genmask	Flags	MSS Window	irrt	Iface
10.0.0.0	0.0.0.0	255.255.255.252	U	0 0	0	eth0
10.0.0.4	0.0.0.0	255.255.255.252	U	0 0	0	eth1
10.0.0.8	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
10.0.0.12	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
10.0.0.16	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
10.0.0.20	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
10.0.0.24	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
10.0.0.28	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
172.0.0.0	10.0.0.6	255.0.0.0	UG	0 0	0	eth1
172.16.142.0	10.0.0.1	255.255.255.252	UG	0 0	0	eth0
172.16.142.4	10.0.0.1	255.255.255.252	UG	0 0	0	eth0
172.16.143.0	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
172.16.143.4	10.0.0.6	255.255.255.252	UG	0 0	0	eth1
192.168.0.132	10.0.0.1	255.255.255.192	UG	0 0	0	eth0

(b) Tabela depois da eliminação

Figura 3.33: Tabelas de Encaminhamento do Router n6

Para fazer agregação das rotas, recorre-se à técnica de supernetting, resultando num IP geral de 192.168.0.224/27, como é possível ver pela figura 3.34.

## SuperNetting do Instuticional e CondadoPortucalense

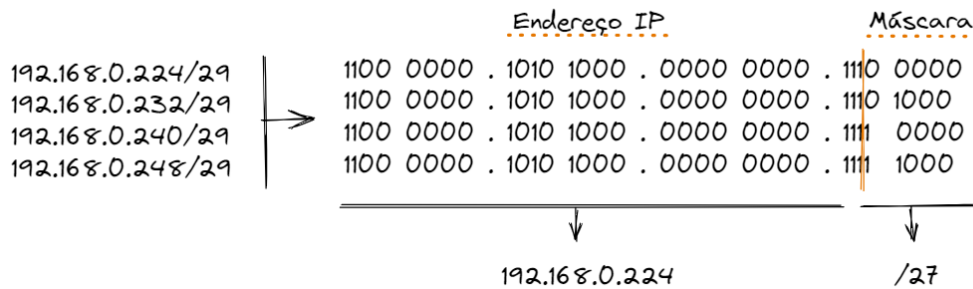


Figura 3.34: Cálculo do Supernetting do Instuticional e CondadoPortucalense

Assim sendo, como é possível observar na figura 3.35, adicionou-se a nova entrada à tabela de encaminhamento (a) e testou-se a conectividade entre o router Teresa e o polo Instuticional e CondadoPortucalense (b):

```
root@n6:/tmp/pycore.38491/n6.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
10.0.0.0 0.0.0.0 255.255.255.252 U 0 0 0 eth0
10.0.0.4 0.0.0.0 255.255.255.252 U 0 0 0 eth1
10.0.0.8 10.0.0.6 255.255.255.252 UG 0 0 0 eth1
10.0.0.12 10.0.0.6 255.255.255.252 UG 0 0 0 eth1
10.0.0.16 10.0.0.6 255.255.255.252 UG 0 0 0 eth1
10.0.0.20 10.0.0.6 255.255.255.252 UG 0 0 0 eth1
10.0.0.24 10.0.0.6 255.255.255.252 UG 0 0 0 eth1
10.0.0.28 10.0.0.6 255.255.255.252 UG 0 0 0 eth1
172.0.0.0 10.0.0.6 255.0.0.0 UG 0 0 0 eth1
172.16.142.0 10.0.0.1 255.255.255.252 UG 0 0 0 eth0
172.16.142.4 10.0.0.1 255.255.255.252 UG 0 0 0 eth0
172.16.143.0 10.0.0.6 255.255.255.252 UG 0 0 0 eth1
172.16.143.4 10.0.0.6 255.255.255.252 UG 0 0 0 eth1
192.168.0.192 10.0.0.1 255.255.255.192 UG 0 0 0 eth0
192.168.0.224 10.0.0.6 255.255.255.224 UG 0 0 0 eth1
```

(a) Nova Tabela com Supernetting

```
root@teresa:/tmp/pycore.38491/Teresa.conf# ping 192.168.0.234
PING 192.168.0.234 (192.168.0.234) 56(84) bytes of data:
64 bytes from 192.168.0.234: icmp_seq=1 ttl=55 time=0.146 ms
64 bytes from 192.168.0.234: icmp_seq=2 ttl=55 time=0.205 ms
64 bytes from 192.168.0.234: icmp_seq=3 ttl=55 time=0.078 ms
^C
--- 192.168.0.234 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2048ms
rtt min/avg/max/ndev = 0.078/0.143/0.206/0.091 ms
root@teresa:/tmp/pycore.38491/Teresa.conf# ping 192.168.0.242
PING 192.168.0.242 (192.168.0.242) 56(84) bytes of data:
64 bytes from 192.168.0.242: icmp_seq=1 ttl=55 time=0.102 ms
64 bytes from 192.168.0.242: icmp_seq=2 ttl=55 time=0.223 ms
64 bytes from 192.168.0.242: icmp_seq=3 ttl=55 time=0.114 ms
^C
--- 192.168.0.242 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2053ms
rtt min/avg/max/ndev = 0.102/0.146/0.223/0.054 ms
root@teresa:/tmp/pycore.38491/Teresa.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=55 time=0.108 ms
64 bytes from 192.168.0.250: icmp_seq=2 ttl=55 time=0.121 ms
64 bytes from 192.168.0.250: icmp_seq=3 ttl=55 time=0.100 ms
64 bytes from 192.168.0.250: icmp_seq=4 ttl=55 time=0.137 ms
^C
--- 192.168.0.250 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3053ms
rtt min/avg/max/ndev = 0.100/0.131/0.197/0.038 ms
root@teresa:/tmp/pycore.38491/Teresa.conf# ping 192.168.0.227
PING 192.168.0.227 (192.168.0.227) 56(84) bytes of data:
64 bytes from 192.168.0.227: icmp_seq=1 ttl=55 time=0.153 ms
64 bytes from 192.168.0.227: icmp_seq=2 ttl=55 time=0.118 ms
64 bytes from 192.168.0.227: icmp_seq=3 ttl=55 time=0.083 ms
^C
--- 192.168.0.227 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2035ms
rtt min/avg/max/ndev = 0.083/0.118/0.153/0.028 ms
```

(b) Testes de Conetividade

Figura 3.35: Router n6 com Supernetting

### **c. Comente os aspetos positivos e negativos do uso do Supernetting**

O Supernetting, também conhecido como agregação de rotas, é uma técnica de redes que permite a combinação de várias redes menores em uma única rede maior com o objetivo de reduzir o número de entradas nas tabelas de encaminhamento. Essa técnica tem aspetos positivos e negativos, tais como:

#### **Aspetos Positivos**

- Melhoria da eficiência do endereçamento
- Simplificação da tabela de encaminhamento

#### **Aspetos Negativos**

- Possíveis problemas de compatibilidade
- Necessidade de Configuração cuidadosa

## 4 Conclusão

Em conclusão, o estudo aprofundado dos temas abordados neste trabalho, como o formato de um datagrama IP, a fragmentação de pacotes IP, o endereçamento IP, o encaminhamento IP, proporcionou uma compreensão teórica mais sólida destes conceitos fundamentais para o funcionamento da Internet e das redes IP.

Ao explorar detalhadamente cada um desses temas, pudemos compreender as nuances e complexidades envolvidas em como os dados são estruturados em pacotes IP, como pacotes grandes são fragmentados e reagrupados durante a transmissão, como os dispositivos são identificados e localizados na Internet através do endereçamento IP, como os pacotes são encaminhados para alcançar seu destino e como técnicas avançadas de subnetting e supernetting podem ser aplicadas para otimizar o uso do espaço de endereçamento IP.

Além disso, o estudo desses temas também nos permitiu compreender a importância de considerações de segurança e privacidade relacionadas ao IP, bem como estar cientes das tendências emergentes, como a crescente adoção de redes IPv6.

Em resumo, este trabalho aprofundou nossa compreensão teórica dos temas abordados, permitindo-nos ter uma visão mais abrangente e aprofundada sobre o funcionamento dos protocolos IP e suas aplicações em redes modernas. O conhecimento adquirido neste trabalho será valioso para a compreensão e gestão de redes IP de forma eficiente, segura e confiável em futuros estudos e práticas relacionadas a redes de computadores.