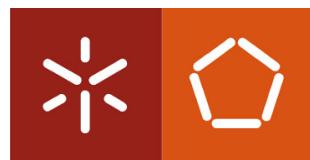


UNIVERSIDADE DO MINHO
DEPARTAMENTO DE INFORMÁTICA



TRABALHO PRÁTICO 2

Redes de Computadores

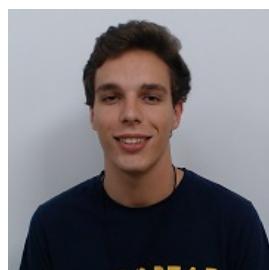
Grupo 14



Inês Bastos A89522



João Freitas A83782



João Félix A89460

Novembro de 2020

Conteúdo

1 1 TP2: Protocolo IPv4 (Parte I)	3
1.1 Exercício 1	3
1.1.1 a. Active o wireshark ou o tcpdump no Cliente1. Numa shell do Cliente1, execute o comando traceroute -I para o endereço IP do Servidor1.	3
1.1.2 b. Registe e analise o tráfego ICMP enviado pelo Cliente1 e o tráfego ICMP recebido como resposta. Comente os resultados face ao comportamento esperado.	4
1.1.3 c. Qual deve ser o valor inicial mínimo do campo TTL para alcançar o Servidor1? Verifique na prática que a sua resposta está correta.	4
1.1.4 d. Calcule o valor médio do tempo de ida-e-volta (Round-Trip Time) obtido?	5
1.2 Exercício 2	5
1.2.1 a. Qual é o endereço IP da interface ativa do seu computador?	5
1.2.2 b. Qual é o valor do campo protocolo? O que identifica?	5
1.2.3 c. Quantos bytes tem o cabeçalho IP(v4)? Quantos bytes tem o campo de dados (payload) do datagrama? Como se calcula o tamanho do payload?	5
1.2.4 d. O datagrama IP foi fragmentado? Justifique.	5
1.2.5 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.	6
1.2.6 f. Observa algum padrão nos valores do campo de Identificação do datagrama IP e TTL?	6
1.2.7 g. Ordene o tráfego capturado por endereço destino e encontre a série de respostas ICMP TTL exceeded enviadas ao seu computador. Qual é o valor do campo TTL? Esse valor permanece constante para todas as mensagens de resposta ICMP TTL exceeded enviados ao seu host? Porquê?	7
1.3 Exercício 3	8
1.3.1 a. Localize a primeira mensagem ICMP. Porque é que houve necessidade de fragmentar o pacote inicial?	8
1.3.2 b. Imprima o primeiro fragmento do datagrama IP segmentado. Que informação no cabeçalho indica que o datagrama foi fragmentado? Que informação no cabeçalho IP indica que se trata do primeiro fragmento? Qual é o tamanho deste datagrama IP?	8
1.3.3 c. Imprima o segundo fragmento do datagrama IP original. Que informação do cabeçalho IP indica que não se trata do 1º fragmento? Há mais fragmentos? O que nos permite afirmar isso?	9
1.3.4 d. Quantos fragmentos foram criados a partir do datagrama original? Como se detecta o último fragmento correspondente ao datagrama original?	10
1.3.5 e. Indique, resumindo, os campos que mudam no cabeçalho IP entre os diferentes fragmentos, e explique a forma como essa informação permite reconstruir o datagrama original.	10
2 2 TP2: Protocolo IP (Parte II)	11
2.1 Exercício 1	11

2.1.1	a. Indique que endereços IP e máscaras de rede foram atribuídos pelo CORE a cada equipamento. Para simplificar, pode incluir uma imagem que ilustre de forma clara a topologia definida e o endereçamento usado.	11
2.1.2	b. Tratam-se de endereços públicos ou privados? Porquê?	11
2.1.3	c. Porque razão não é atribuído um endereço IP aos switches?	12
2.1.4	d. Usando o comando ping certifique-se que existe conectividade IP entre os laptops dos vários departamentos e o servidor do departamento A (basta certificar-se da conectividade de um laptop por departamento).	12
2.1.5	e. Verifique se existe conectividade IP do router de acesso RISP para o servidor S1.	12
2.2	Exercício 2	13
2.2.1	a. Execute o comando netstat -rn por forma a poder consultar a tabela de encaminhamento unicast (IPv4). Inclua no seu relatório as tabelas de encaminhamento obtidas; interprete as várias entradas de cada tabela. Se necessário, consulte o manual respetivo (man netstat).	13
2.2.2	b. Diga, justificando, se está a ser usado encaminhamento estático ou dinâmico (sugestão: analise que processos estão a correr em cada sistema, por exemplo, ps -ax).	13
2.2.3	c. Admita que, por questões administrativas, a rota por defeito (0.0.0.0 ou default) deve ser retirada definitivamente da tabela de encaminhamento do servidor S1 localizado no departamento A. Use o comando route delete para o efeito. Que implicações tem esta medida para os utilizadores da organização MIEI-RC que accedem ao servidor. Justifique.	14
2.2.4	d. Adicione as rotas estáticas necessárias para restaurar a conectividade para o servidor S1, por forma a contornar a restrição imposta na alínea c). Utilize para o efeito o comando route add e registe os comandos que usou.	14
2.2.5	e. Teste a nova política de encaminhamento garantindo que o servidor está novamente acessível, utilizando para o efeito o comando ping. Registe a nova tabela de encaminhamento do servidor.	15
2.3	Exercício 3	15
2.3.1	1) Considere que dispõe apenas do endereço de rede IP 130.XX.96.0/19, em que XX é o decimal correspondendo ao seu número de grupo (PLXX). Defina um novo esquema de endereçamento para as redes dos departamentos (mantendo a rede de acesso e core inalteradas) e atribua endereços às interfaces dos vários sistemas envolvidos. Assuma que todos os endereços de sub-redes são usáveis. Deve justificar as opções usadas.	15
2.3.2	2) Qual a máscara de rede que usou (em formato decimal)? Quantos hosts IP pode interligar em cada departamento? Justifique.	15
2.3.3	3) Garanta e verifique que conectividade IP entre as várias redes locais da organização MIEI-RC é mantida. Explique como procedeu.	16
3	Conclusão	18

Capítulo 1

1 TP2: Protocolo IPv4 (Parte I)

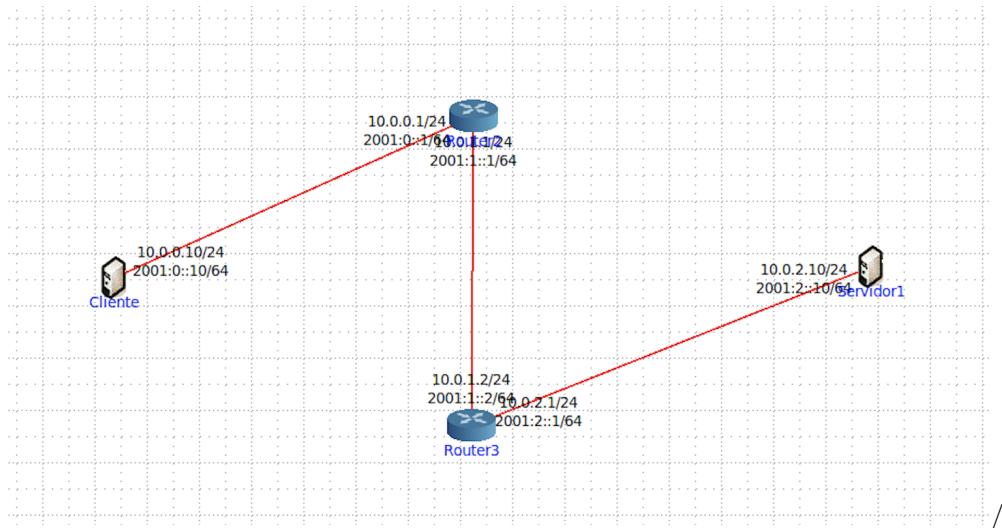


Figura 1.1: Topologia CORE.

1.1 Exercício 1

- 1.1.1 a. Active o wireshark ou o tcpdump no Cliente1. Numa shell do Cliente1, execute o comando traceroute -I para o endereço IP do Servidor1.

```
root@Cliente:/tmp/pycore.36771/Cliente.conf# traceroute -I 10.0.2.10
traceroute to 10.0.2.10 (10.0.2.10), 30 hops max, 60 byte packets
 1  10.0.0.1 (10.0.0.1)  0.037 ms  0.007 ms  0.005 ms
 2  10.0.1.2 (10.0.1.2)  0.016 ms  0.007 ms  0.006 ms
 3  10.0.2.10 (10.0.2.10)  0.027 ms  0.016 ms  0.014 ms
root@Cliente:/tmp/pycore.36771/Cliente.conf#
```

Figura 1.2: Resultados da execução do traceroute.

1.1.2 b. Registe e analise o tráfego ICMP enviado pelo Cliente1 e o tráfego ICMP recebido como resposta. Comente os resultados face ao comportamento esperado.

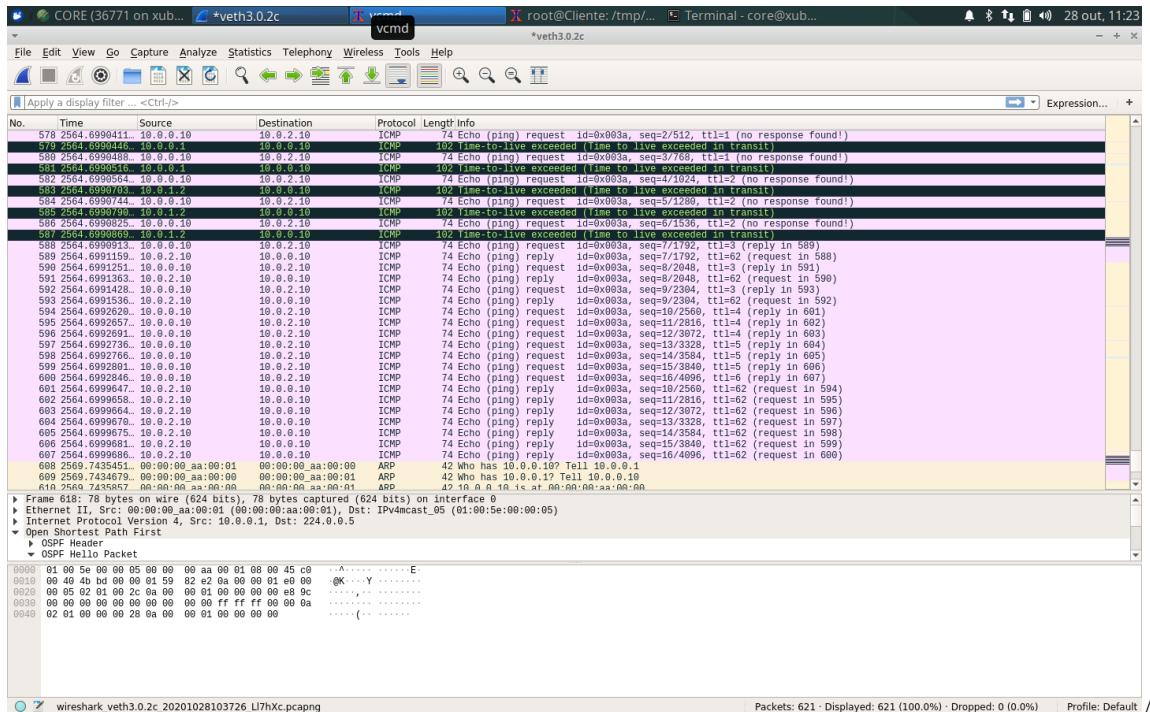


Figura 1.3: Análise tráfego wireshark.

Face aos resultados analisados, observamos que se verificou o comportamento esperado tendo sido enviado de h1 vários pacotes, "Echo (ping) request". Numa primeira fase, foram enviados vários pacotes com TTL = 1, descartados por r2. Em seguida, foram enviados pacotes com TTL = 2, também descartados por r3 e, por fim, foram enviados pacotes com TTL = 3 que chegaram ao seu destino, s4. Para cada pacote descartado foi recebido um outro pacote do router que o descartou, "Time-to-live exceeded". Como resposta aos pacotes que alcançaram o destino foi recebido um pacote "Echo (ping) reply".

1.1.3 c. Qual deve ser o valor inicial mínimo do campo TTL para alcançar o Servidor1? Verifique na prática que a sua resposta está correta.

TTL = 3

311 1374.9328835...	10.0.0.10	10.0.2.10	ICMP	74 Echo (ping) request id=0x0039, seq=7/1792, ttl=3 (reply in 312)
312 1374.9329241...	10.0.2.10	10.0.0.10	ICMP	74 Echo (ping) reply id=0x0039, seq=7/1792, ttl=62 (request in 311)
313 1374.9329339...	10.0.0.10	10.0.2.10	ICMP	74 Echo (ping) request id=0x0039, seq=8/2048, ttl=3 (reply in 314)
314 1374.9329445...	10.0.2.10	10.0.0.10	ICMP	74 Echo (ping) reply id=0x0039, seq=8/2048, ttl=62 (request in 313)

Figura 1.4: Comunicação entre as máquinas com sucesso.

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

$$\text{RTT1} = (0.037 + 0.007 + 0.005)/3 = 0,016333333\text{ms}$$

$$\text{RTT2} = (0.016 + 0.007 + 0.006)/3 = 0,00966667\text{ms}$$

$$\text{RTT3} = (0.027 + 0.016 + 0.014)/3 = 0,019\text{ms}$$

1.2 Exercício 2

```
> Frame 23685: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface \Device\NPF_{24E6145C-24B6-4920-9EAB-678E32B370A7}, id 0
> Ethernet II, Src: IntelCor_e4:29:38 (94:b8:6d:e4:29:38), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
  Internet Protocol Version 4, Src: 172.26.58.210, Dst: 193.136.9.240
    Version: 4
    Header Length: 20 bytes (5)
    Differentiated Services Field: 0x00 (DS0, ECN: Not-ECT)
      Total Length: 56
      Identification: 0xaeef2 (44786)
    Flags: 0x00
    Fragment Offset: 0
    Time to Live: 4
    Protocol: ICMP (1)
    Header Checksum: 0x0000 [validation disabled]
      [Header checksum status: Unverified]
    Source Address: 172.26.58.210
    Destination Address: 193.136.9.240
  Internet Control Message Protocol
```

Figura 1.5: Cabeçalho de comunicação.

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

172.26.58.210

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

ICMP(1).

Identifica Internet Protocol.

1.2.3 c. Quantos bytes tem o cabeçalho IP(v4)? Quantos bytes tem o campo de dados (payload) do datagrama? Como se calcula o tamanho do payload?

O cabeçalho (Header length) tem 20 bytes.

Payload = 56 - 20 = 36 bytes.

O tamanho do campo de dados (payload) é calculado a partir da subtração do tamanho total do pacote(56 bytes) pelo tamanho do cabeçalho(20 bytes).

1.2.4 d. O datagrama IP foi fragmentado? Justifique.

Como podemos observar, o Fragment Offset está definido a 0 e o campo "More Fragments" está como "Not set", ou seja, não definido. Com isto, podemos concluir que estamos no início do pacote e, como o campo "More Fragments" não está definido, estamos também na última parte do pacote. Concluimos, portanto que ele não está fragmentado.

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

Os campos do cabeçalho IP que variam de pacote são os seguintes: TTL, Header e identification.

1 0.000000	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=976/53251, ttl=255 (reply in 2)
3 0.050564	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=977/53507, ttl=1 (no response found!)
5 0.101362	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=978/53763, ttl=2 (no response found!)
7 0.152711	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=979/54019, ttl=3 (no response found!)
9 0.202866	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=980/54275, ttl=4 (reply in 10)
11 1.123011	172.26.58.210	162.159.134.234	TLSv1.2	105 Application Data	
14 1.300080	172.26.58.210	162.159.134.234	TCP	54 49723 + 443 [ACK] Seq=52 Ack=34 Win=509 Len=0	
15 2.500038	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=981/54531, ttl=255 (reply in 16)
17 2.550413	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=982/54787, ttl=1 (no response found!)
19 2.601506	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=983/55043, ttl=2 (no response found!)
21 2.652375	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=984/55299, ttl=3 (no response found!)
23 2.702562	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=985/55555, ttl=4 (reply in 24)
25 5.000935	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=986/55811, ttl=255 (reply in 26)
27 5.050647	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=987/56067, ttl=1 (no response found!)
29 5.101346	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=988/56323, ttl=2 (no response found!)
31 5.151573	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=989/56579, ttl=3 (no response found!)
33 5.202156	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=990/56835, ttl=4 (reply in 34)
35 6.946223	172.26.58.210	216.158.209.67	TCP	55 49810 + 443 [ACK] Seq=1 Ack=1 Win=510 Len=1 [TCP segment of a reassembled PDU]	
37 7.501149	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=991/57091, ttl=255 (reply in 38)
39 7.551513	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=992/57347, ttl=1 (no response found!)
41 7.601628	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=993/57603, ttl=2 (no response found!)
43 7.652201	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=994/57859, ttl=3 (no response found!)
45 7.702193	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=995/58115, ttl=4 (reply in 46)
47 8.262799	172.26.58.210	64.233.166.188	TCP	55 [TCP segment of a reassembled PDU]	
49 8.279700	172.26.58.210	239.255.255.250	UDP	698 49866 + 3702 Len=656	
52 8.513648	172.26.58.210	239.255.255.250	UDP	698 49866 + 3702 Len=656	
54 8.972700	172.26.58.210	239.255.255.250	UDP	698 49866 + 3702 Len=656	
56 9.885316	172.26.58.210	239.255.255.250	UDP	698 49866 + 3702 Len=656	
57 10.003130	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=996/58371, ttl=255 (reply in 58)
59 10.053224	172.26.58.210	193.136.9.240	ICMP	70 Echo (ping) request	id=0x0001, seq=997/58627, ttl=1 (no response found!)

Figura 1.6: Tráfego wireshark ordenado por endereço fonte.

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

Identificação do datagrama IP: Identificamos que os 8 primeiros bits são iguais (0x0001).

TTL: é incrementado sequencialmente.

- 1.2.7 g. Ordene o tráfego capturado por endereço destino e encontre a série de respostas ICMP TTL exceeded enviadas ao seu computador. Qual é o valor do campo TTL? Esse valor permanece constante para todas as mensagens de resposta ICMP TTL exceeded enviados ao seu host? Porquê?

25434 2946.200486	193.136.9.240	172.26.58.210	ICMP	70 Echo (ping) reply id=0x0001, seq=6870/54810, ttl=61 (request in 25433)
25432 2946.144189	172.16.115.252	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25430 2946.091993	172.16.2.1	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25428 2946.038463	172.26.254.254	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25426 2945.990425	193.136.9.240	172.26.58.210	ICMP	70 Echo (ping) reply id=0x0001, seq=6866/53786, ttl=61 (request in 25425)
25424 2943.686498	193.136.9.240	172.26.58.210	ICMP	70 Echo (ping) reply id=0x0001, seq=6865/53530, ttl=61 (request in 25423)
25422 2943.636744	172.16.115.252	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25420 2943.611895	172.16.2.1	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25419 2943.610746	172.26.254.254	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25418 2943.602002	193.136.9.240	172.26.58.210	ICMP	70 Echo (ping) reply id=0x0001, seq=6861/52506, ttl=61 (request in 25415)
25414 2943.396728	173.194.76.188	172.26.58.210	TCP	66 [TCP Keep-Alive ACK] 5228 → 49912 [ACK] Seq=6180 Ack=828 Win=67840 Len=0 SLE=827 SRE=828
25412 2941.199853	193.136.9.240	172.26.58.210	ICMP	70 Echo (ping) reply id=0x0001, seq=6860/52250, ttl=61 (request in 25411)
25410 2941.132110	172.16.115.252	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25408 2941.078886	172.16.2.1	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25406 2941.029962	172.26.254.254	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25404 2940.984315	193.136.9.240	172.26.58.210	ICMP	70 Echo (ping) reply id=0x0001, seq=6856/51226, ttl=61 (request in 25403)
25402 2938.680392	193.136.9.240	172.26.58.210	ICMP	70 Echo (ping) reply id=0x0001, seq=6855/50970, ttl=61 (request in 25401)
25400 2938.628749	172.16.115.252	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25398 2938.590197	172.16.2.1	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25396 2938.526876	172.26.254.254	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25394 2938.485477	193.136.9.240	172.26.58.210	ICMP	70 Echo (ping) reply id=0x0001, seq=6851/49946, ttl=61 (request in 25393)
25391 2937.619886	35.186.224.47	172.26.58.210	TLSv1.2	94 Application Data
25390 2937.607369	35.186.224.47	172.26.58.210	TCP	54.443 → 49728 [ACK] Seq=3881 Ack=4215 Win=267 Len=0
25388 2936.197383	193.136.9.240	172.26.58.210	ICMP	70 Echo (ping) reply id=0x0001, seq=6850/49690, ttl=61 (request in 25387)
25386 2936.129773	172.16.115.252	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25384 2936.075853	172.16.2.1	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25382 2936.033985	172.26.254.254	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25380 2935.986281	193.136.9.240	172.26.58.210	ICMP	70 Echo (ping) reply id=0x0001, seq=6846/48666, ttl=61 (request in 25379)
25378 2933.790446	193.136.9.240	172.26.58.210	ICMP	70 Echo (ping) reply id=0x0001, seq=6845/48410, ttl=61 (request in 25377)
25376 2933.633751	172.16.115.252	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25374 2933.601946	172.16.2.1	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25372 2933.546769	172.26.254.254	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
25370 2933.473736	193.136.9.240	172.26.58.210	ICMP	70 Echo (ping) reply id=0x0001, seq=6841/47386, ttl=61 (request in 25369)
25368 2931.283426	193.136.9.240	172.26.58.210	ICMP	70 Echo (ping) reply id=0x0001, seq=6840/47130, ttl=61 (request in 25364)

Figura 1.7: Tráfego wireshark ordenado por endereço de destino.

O valor do campo TTL é 61. Este valor permanece constante para todas as mensagens de resposta ICMP TTL exceeded enviados ao nosso host. O TTL predefinido (hardocored) pelo destino é 64 (garante que o pacote chegue da origem ao destino). No final, quando o pacote chega ao seu destino, como passou por 3 routers intermédios o valor foi incrementado 3 vezes, daí o TTL ser 61.

1.3 Exercício 3

No.	Time ^	Source	Destination	Protocol	Length	Info
10	0.880252	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=cc1f) [Reassembled in #12]
11	0.880252	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=cc1f) [Reassembled in #12]
12	0.880252	172.26.58.210	193.136.9.240	ICMP	68	Echo (ping) request id=0x0001, seq=4628/5138, ttl=1 (no response found!)
13	0.882896	172.26.254.254	172.26.58.210	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
14	0.930557	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=cc20) [Reassembled in #16]
15	0.930557	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=cc20) [Reassembled in #16]
16	0.938057	172.26.58.210	193.136.9.240	ICMP	68	Echo (ping) request id=0x0001, seq=4629/5394, ttl=2 (no response found!)
17	0.933180	172.16.2.1	172.26.58.210	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
18	0.981096	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=cc21) [Reassembled in #20]
19	0.981096	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=cc21) [Reassembled in #20]
20	0.981096	172.26.58.210	193.136.9.240	ICMP	68	Echo (ping) request id=0x0001, seq=4630/5650, ttl=3 (no response found!)
21	0.984621	172.16.115.252	172.26.58.210	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
22	1.031594	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=cc22) [Reassembled in #24]
23	1.031594	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=cc22) [Reassembled in #24]
24	1.031594	172.26.58.210	193.136.9.240	ICMP	68	Echo (ping) request id=0x0001, seq=4631/5906, ttl=4 (reply in 27)
25	1.044579	193.136.9.240	172.26.58.210	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=c794) [Reassembled in #27]
26	1.044579	193.136.9.240	172.26.58.210	IPv4	68	Fragmented IP protocol (proto=ICMP 1, off=0, ID=2960, ID=c794) [Reassembled in #27]
27	1.044579	193.136.9.240	172.26.58.210	ICMP	1514	Echo (ping) reply id=0x0001, seq=4631/5906, ttl=61 (request in 24)
28	1.748210	172.26.58.210	162.159.134.234	TLSv1.2	105	Application Data
29	1.757009	162.159.134.234	172.26.58.210	TCP	54	443 → 49747 [ACK] Seq=1 Ack=52 Win=73 Len=0
30	1.889695	162.159.134.234	172.26.58.210	TLSv1.2	87	Application Data
31	1.935229	172.26.58.210	162.159.134.234	TCP	54	49747 → 443 [ACK] Seq=52 Ack=34 Win=509 Len=0
32	3.331099	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=cc23) [Reassembled in #34]
33	3.331099	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=cc23) [Reassembled in #34]
34	3.331099	172.26.58.210	193.136.9.240	ICMP	68	Echo (ping) request id=0x0001, seq=4632/6162, ttl=255 (reply in 37)
35	3.337405	193.136.9.240	172.26.58.210	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=c7b) [Reassembled in #37]
36	3.338916	193.136.9.240	172.26.58.210	IPv4	68	Fragmented IP protocol (proto=ICMP 1, off=2960, ID=c7b) [Reassembled in #37]
37	3.357774	193.136.9.240	172.26.58.210	ICMP	1514	Echo (ping) reply id=0x0001, seq=4632/6162, ttl=61 (request in 34)
38	3.381809	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=cc24) [Reassembled in #40]
39	3.381809	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=cc24) [Reassembled in #40]
40	3.381809	172.26.58.210	193.136.9.240	ICMP	68	Echo (ping) request id=0x0001, seq=4633/6418, ttl=1 (no response found!)
41	3.387269	172.26.254.254	172.26.58.210	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
42	3.432532	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=cc25) [Reassembled in #44]
43	3.432532	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=cc25) [Reassembled in #44]
44	3.432532	172.26.58.210	193.136.9.240	ICMP	68	Echo (ping) request id=0x0001, seq=4634/6674, ttl=2 (no response found!)
45	3.436645	172.16.2.1	172.26.58.210	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
46	3.483124	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=cc26) [Reassembled in #48]
47	3.483124	172.26.58.210	193.136.9.240	TCP	4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=cc26) [Reassembled in #48]

Figura 1.8: Fragmentos do datagrama IP.

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

A primeira mensagem ICMP é a mensagem número 12 como podemos observar pela figura 1.8. Houve necessidade de fragmentar porque o tamanho do PDU é maior do que o máximo permitido pelo protocolo IPv4.

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

Nas Flags, o bit correspondente a "More fragments" tem de valor 1 (Set), o que indica que o diagrama foi fragmentado.

Como o offset é 0 ("Fragment offset"), este é o primeiro fragmento.

O tamanho do datagrama IP é 1500 bytes ("Total Length").

No.	Time	Source	Destination	Protocol	Length	Info
10	0.880252	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=cc1f) [Reassembled in #12]
11	0.880252	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=cc1f) [Reassembled in #12]
12	0.880252	172.26.58.210	193.136.9.240	ICMP	68	Echo (ping) request id=0x0001, seq=4628/5138, ttl=1 (no response found!)
	13 0.882896	172.26.254.254	172.26.58.210	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
14	0.930557	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=cc20) [Reassembled in #16]
15	0.930557	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=cc20) [Reassembled in #16]
16	0.930557	172.26.58.210	193.136.9.240	ICMP	68	Echo (ping) request id=0x0001, seq=4629/5394, ttl=2 (no response found!)
17	0.933188	172.16.2.1	172.26.58.210	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
18	0.981096	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=cc21) [Reassembled in #20]
19	0.981096	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=cc21) [Reassembled in #20]
20	0.981096	172.26.58.210	193.136.9.240	ICMP	68	Echo (ping) request id=0x0001, seq=4630/5650, ttl=3 (no response found!)
21	0.984621	172.16.115.252	172.26.58.210	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)

```

> Frame 10: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface \Device\NPF_{24E6145C-24B6-4920-9EAB-678E32B370A7}, id 0
> Ethernet II, Src: IntelCor_e4:29:38 (94:b8:6d:e4:29:38), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
< Internet Protocol Version 4, Src: 172.26.58.210, 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: 0xcc1f (52255)
  Flags: 0x20, More fragments
    0... .... = Reserved bit: Not set
    .0... .... = Don't fragment: Not set
    ..1.... = More fragments: Set
  Fragment Offset: 0
  > Time to Live: 1
    Protocol: ICMP (1)
    Header Checksum: 0x0000 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 172.26.58.210
    Destination Address: 193.136.9.240
    [Reassembled IPv4 in frame: 12]
  > Data (1480 bytes)

```

Figura 1.9: Primeiro fragmento do datagrama IP.

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

No.	Time	Source	Destination	Protocol	Length	Info
10	0.880252	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=cc1f) [Reassembled in #12]
11	0.880252	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=cc1f) [Reassembled in #12]
12	0.880252	172.26.58.210	193.136.9.240	ICMP	68	Echo (ping) request id=0x0001, seq=4628/5138, ttl=1 (no response found!)
	13 0.882896	172.26.254.254	172.26.58.210	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
14	0.930557	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=cc20) [Reassembled in #16]
15	0.930557	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=cc20) [Reassembled in #16]
16	0.930557	172.26.58.210	193.136.9.240	ICMP	68	Echo (ping) request id=0x0001, seq=4629/5394, ttl=2 (no response found!)
17	0.933188	172.16.2.1	172.26.58.210	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
18	0.981096	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=cc21) [Reassembled in #20]
19	0.981096	172.26.58.210	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=cc21) [Reassembled in #20]
20	0.981096	172.26.58.210	193.136.9.240	ICMP	68	Echo (ping) request id=0x0001, seq=4630/5650, ttl=3 (no response found!)
21	0.984621	172.16.115.252	172.26.58.210	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)

```

> Frame 11: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface \Device\NPF_{24E6145C-24B6-4920-9EAB-678E32B370A7}, id 0
> Ethernet II, Src: IntelCor_e4:29:38 (94:b8:6d:e4:29:38), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
< Internet Protocol Version 4, Src: 172.26.58.210, 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: 0xcc1f (52255)
  Flags: 0x20, More fragments
    0... .... = Reserved bit: Not set
    .0... .... = Don't fragment: Not set
    ..1.... = More fragments: Set
  Fragment Offset: 1480
  > Time to Live: 1
    Protocol: ICMP (1)
    Header Checksum: 0x0000 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 172.26.58.210
    Destination Address: 193.136.9.240
    [Reassembled IPv4 in frame: 12]
  > Data (1480 bytes)

```

Figura 1.10: Segundo fragmento do datagrama IP.

Não é o 1º fragmento do pacote porque o offset não é 0 ("Fragment offset" = 1480). Sim, há mais fragmentos porque o bit correspondente a "More fragments" é 1.

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

10 0.880252	172.26.58.210	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=cc1f) [Reassembled in #12]
11 0.880252	172.26.58.210	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=1480, ID=cc1f) [Reassembled in #12]
12 0.880252	172.26.58.210	193.136.9.240	ICMP	68 Echo (ping) request id=0x0001, seq=4628/5138, ttl=1 (no response found!)
13 0.882896	172.26.254.254	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
14 0.930557	172.26.58.210	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=cc20) [Reassembled in #16]
15 0.930557	172.26.58.210	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=1480, ID=cc20) [Reassembled in #16]
16 0.930557	172.26.58.210	193.136.9.240	ICMP	68 Echo (ping) request id=0x0001, seq=4629/5394, ttl=2 (no response found!)
17 0.933180	172.16.2.1	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)
18 0.981096	172.26.58.210	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=cc21) [Reassembled in #20]
19 0.981096	172.26.58.210	193.136.9.240	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=1480, ID=cc21) [Reassembled in #20]
20 0.981096	172.26.58.210	193.136.9.240	ICMP	68 Echo (ping) request id=0x0001, seq=4630/5650, ttl=3 (no response found!)
21 0.984621	172.16.115.252	172.26.58.210	ICMP	70 Time-to-live exceeded (Time to live exceeded in transit)

```

rame 12: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) on interface \Device\NPF_{24E6145C-24B6-4920-9EAB-678E32B370A7}, id 0
thernet II, Src: IntelCor_e4:29:38 (94:b8:6d:e4:29:38), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
Internet Protocol Version 4, Src: 172.26.58.210, 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: 54
  Identification: 0xcc1f (52255)
  Flags: 0x01
    0... .... = Reserved bit: Not set
    .0... .... = Don't fragment: Not set
    ..0.... = More fragments: Not set
  Fragment Offset: 2960
> Time to Live: 1
  Protocol: ICMP (1)
  Header Checksum: 0x0000 [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 172.26.58.210
  Destination Address: 193.136.9.240
> [3 IPv4 Fragments (2994 bytes): #10(1480), #11(1480), #12(34)]
  Internet Control Message Protocol

```

Figura 1.11: Terceiro fragmento do datagrama IP.

Foram criados 3 fragmentos, o número 10, 11 e o 12 (Figura 1.11). O bit correspondente a "More fragments" é 0 (Not Set), e portanto não há mais fragmentos.

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

Entre os cabeçalhos IP dos fragmentos variam as propriedades "More fragments" e "Fragment offset". Ordenam-se os fragmentos por ordem crescente de "Fragment offset" (sabendo que primeiro é 0) até que o bit de "More fragment" seja 0 (Not Set), isto significa que estamos no último fragmento.

Capítulo 2

2 TP2: Protocolo IP (Parte II)

2.1 Exercício 1

- 2.1.1 a. Indique que endereços IP e máscaras de rede foram atribuídos pelo CORE a cada equipamento. Para simplificar, pode incluir uma imagem que ilustre de forma clara a topologia definida e o endereçamento usado.

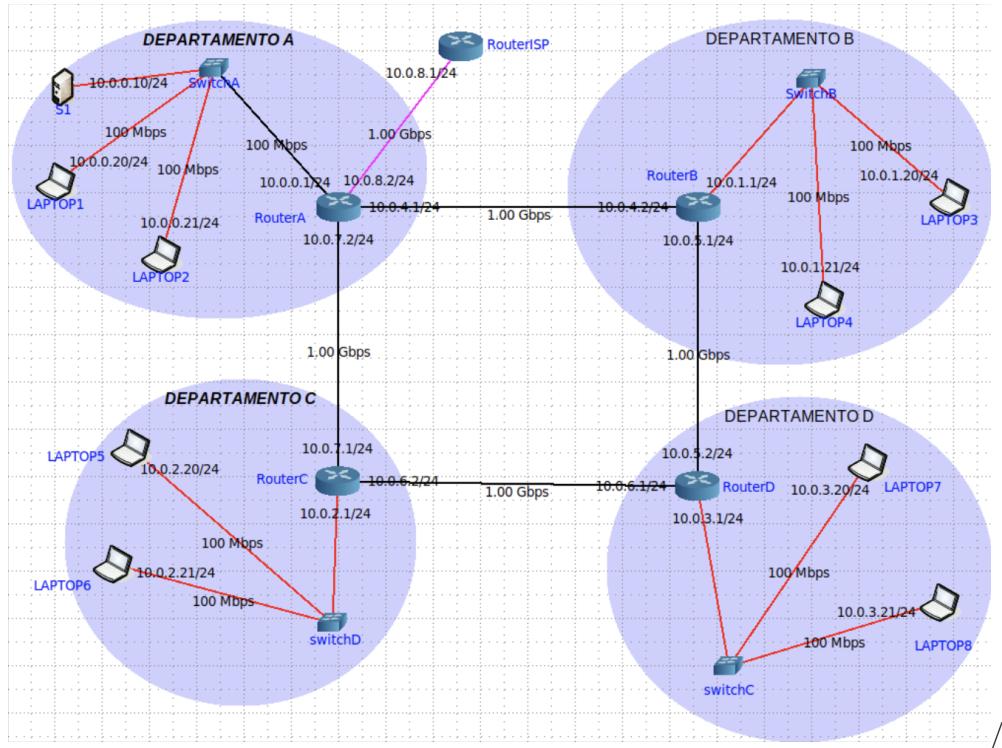


Figura 2.1: Topologia CORE.

- 2.1.2 b. Tratam-se de endereços públicos ou privados? Porquê?

São endereços privados porque utilizam como prefixo um dos blocos reservados a endereços privados na norma RFC 1918 pela IANA ("10.0.0.0 - 10.255.255.255 (10/8 prefix)").

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

Eles não têm um IP atribuído porque operam numa camada abaixo.

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

```
root@S1:/tmp/pycore.34897/S1.conf# ping 10.0.0.20
PING 10.0.0.20 (10.0.0.20) 56(84) bytes of data.
64 bytes from 10.0.0.20: icmp_seq=1 ttl=64 time=0.061 ms
64 bytes from 10.0.0.20: icmp_seq=2 ttl=64 time=0.033 ms
64 bytes from 10.0.0.20: icmp_seq=3 ttl=64 time=0.069 ms
^C
--- 10.0.0.20 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2044ms
rtt min/avg/max/mdev = 0.033/0.054/0.069/0.016 ms
root@S1:/tmp/pycore.34897/S1.conf# ping 10.0.1.20
PING 10.0.1.20 (10.0.1.20) 56(84) bytes of data.
64 bytes from 10.0.1.20: icmp_seq=1 ttl=62 time=0.054 ms
64 bytes from 10.0.1.20: icmp_seq=2 ttl=62 time=0.058 ms
64 bytes from 10.0.1.20: icmp_seq=3 ttl=62 time=0.078 ms
^C
--- 10.0.1.20 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2055ms
rtt min/avg/max/mdev = 0.054/0.063/0.078/0.012 ms
root@S1:/tmp/pycore.34897/S1.conf# ping 10.0.2.20
PING 10.0.2.20 (10.0.2.20) 56(84) bytes of data.
64 bytes from 10.0.2.20: icmp_seq=1 ttl=62 time=0.044 ms
64 bytes from 10.0.2.20: icmp_seq=2 ttl=62 time=0.057 ms
64 bytes from 10.0.2.20: icmp_seq=3 ttl=62 time=0.050 ms
^C
--- 10.0.2.20 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2037ms
rtt min/avg/max/mdev = 0.044/0.050/0.057/0.007 ms
root@S1:/tmp/pycore.34897/S1.conf# ping 10.0.3.20
PING 10.0.3.20 (10.0.3.20) 56(84) bytes of data.
64 bytes from 10.0.3.20: icmp_seq=1 ttl=61 time=0.090 ms
64 bytes from 10.0.3.20: icmp_seq=2 ttl=61 time=0.058 ms
64 bytes from 10.0.3.20: icmp_seq=3 ttl=61 time=0.065 ms
^C
--- 10.0.3.20 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2039ms
rtt min/avg/max/mdev = 0.058/0.071/0.090/0.013 ms
root@S1:/tmp/pycore.34897/S1.conf#
```

Figura 2.2: Ping para Laptop a partir de S1 .

2.1.5 e. Verifique se existe conectividade IP do router de acesso RISP para o servidor S1.

```
root@RouterISP:/tmp/pycore.34897/RouterISP.conf# ping 10.0.0.10
PING 10.0.0.10 (10.0.0.10) 56(84) bytes of data.
64 bytes from 10.0.0.10: icmp_seq=1 ttl=63 time=0.068 ms
64 bytes from 10.0.0.10: icmp_seq=2 ttl=63 time=0.046 ms
64 bytes from 10.0.0.10: icmp_seq=3 ttl=63 time=0.046 ms
^C
--- 10.0.0.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2034ms
rtt min/avg/max/mdev = 0.046/0.053/0.068/0.012 ms
root@RouterISP:/tmp/pycore.34897/RouterISP.conf#
```

Figura 2.3: Ping para S1 a partir de Rext.

2.2 Exercício 2

- 2.2.1 a. Execute o comando netstat -rn por forma a poder consultar a tabela de encaminhamento unicast (IPv4). Inclua no seu relatório as tabelas de encaminhamento obtidas; interprete as várias entradas de cada tabela. Se necessário, consulte o manual respetivo (man netstat).

Kernel IP routing table							
Destination	Gateway	Genmask	Flags	MSS	Window	irtt	Iface
10.0.0.0	10.0.7.2	255.255.255.0	UG	0	0	0	eth1
10.0.1.0	10.0.6.1	255.255.255.0	UG	0	0	0	eth0
10.0.2.0	0.0.0.0	255.255.255.0	U	0	0	0	eth2
10.0.3.0	10.0.6.1	255.255.255.0	UG	0	0	0	eth0
10.0.4.0	10.0.7.2	255.255.255.0	UG	0	0	0	eth1
10.0.5.0	10.0.6.1	255.255.255.0	UG	0	0	0	eth0
10.0.6.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
10.0.7.0	0.0.0.0	255.255.255.0	U	0	0	0	eth1
10.0.8.0	10.0.7.2	255.255.255.0	UG	0	0	0	eth1

Figura 2.4: Tabela de Encaminhamento do router do departamento A.

Analisando as entradas da tabela da Figura 2.4: Leitura de cada linha: Um datagrama destinado à rede "Destination" será entregue na interface de endereço "Gateway" saindo pela interface local "Iface". É obrigatório que a máscara seja mencionada (uma vez que estamos a utilizar Classless), que será sempre 24 (255.255.255.0). Analisando agora a segunda e a terceira entrada da tabela verificamos que diferem no Gateway (a terceira tem o endereço default, ao contrário da segunda que tem Gateway definido). Isto acontece porque se os dois endereços estão ligados diretamente, o Gateway não precisa de estar definido, enquanto que se não estiverem ligados diretamente é necessário saber qual é o próximo salto. As flags apenas acrescentam informação adicional, sendo U (a route é válida) utilizada quando o Gateway não está definido, e UG (a route é válida mas redireciona para um gateway e não diretamente a uma rede ou host) caso contrário.

Kernel IP routing table							
Destination	Gateway	Genmask	Flags	MSS	Window	irtt	Iface
0.0.0.0	10.0.2.1	0.0.0.0	UG	0	0	0	eth0
10.0.2.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0

Figura 2.5: Tabela de Encaminhamento de um laptop do departamento A.

Analisando as duas entradas da tabela da Figura 2.5: Como a máscara é 0 o destino 0.0.0.0 identifica todas as redes possíveis, ou seja, caso o tráfego não seja para um host de 10.0.2.0 ele é redirecionado, por defeito, para o Gateway 10.0.2.1.

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

É usado um encaminhamento dinâmico porque as rotas são definidas automaticamente através da troca de informação de routing entre routers.

- 2.2.3 c.** Admita que, por questões administrativas, a rota por defeito (0.0.0.0 ou default) deve ser retirada definitivamente da tabela de encaminhamento do servidor S1 localizado no departamento A. Use o comando route delete para o efeito. Que implicações tem esta medida para os utilizadores da organização MIEI-RC que acedem ao servidor. Justifique.

```
root@S1:/tmp/pycore.34897/S1.conf# netstat -r
Kernel IP routing table
Destination     Gateway         Genmask        Flags   MSS Window irtt Iface
default         10.0.0.1        0.0.0.0       UG        0 0          0 eth0
10.0.0.0        0.0.0.0        255.255.255.0  U         0 0          0 eth0
/
```

Figura 2.6

```
root@S1:/tmp/pycore.34897/S1.conf# route del default
root@S1:/tmp/pycore.34897/S1.conf# netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask        Flags   MSS Window irtt Iface
10.0.0.0        0.0.0.0        255.255.255.0  U         0 0          0 eth0
/
```

Figura 2.7: Tabela de Encaminhamento depois de delete route por defeito.

O servidor perde a conectividade com todos os hosts que não pertencem à sua rede local (Departamento A) isto porque, removendo a rota por defeito, o Servidor não tem definida a rota de envio de tráfego para redes não locais. Assim, os utilizadores da empresa conseguem enviar dados para o servidor mas não conseguem receber.

- 2.2.4 d.** Adicione as rotas estáticas necessárias para restaurar a conectividade para o servidor S1, por forma a contornar a restrição imposta na alínea c). Utilize para o efeito o comando route add e registe os comandos que usou.

```
root@S1:/tmp/pycore.34897/S1.conf# route add 10.0.8.1 gw 10.0.0.1
root@S1:/tmp/pycore.34897/S1.conf# route add -net 10.0.1.0 netmask 255.255.255.0 gw 10.0.0.1
root@S1:/tmp/pycore.34897/S1.conf# route add -net 10.0.2.0 netmask 255.255.255.0 gw 10.0.0.1
root@S1:/tmp/pycore.34897/S1.conf# route add -net 10.0.3.0 netmask 255.255.255.0 gw 10.0.0.1
/
```

Figura 2.8: Adição de static routes.

A partir do S1:

```
route add 10.0.0.1 gw 10.0.0.1

route add -net 10.0.2.0 netmask 255.255.255.0 gw 10.0.0.1

route add -net 10.0.3.0 netmask 255.255.255.0 gw 10.0.0.1
```

- 2.2.5 e. Teste a nova política de encaminhamento garantindo que o servidor está novamente acessível, utilizando para o efeito o comando ping. Registe a nova tabela de encaminhamento do servidor.

```

root@S1:/tmp/pycore.34897/S1.conf# ping 10.0.1.20
PING 10.0.1.20 (10.0.1.20) 56(84) bytes of data.
64 bytes from 10.0.1.20: icmp_seq=1 ttl=62 time=0.146 ms
64 bytes from 10.0.1.20: icmp_seq=2 ttl=62 time=0.067 ms
64 bytes from 10.0.1.20: icmp_seq=3 ttl=62 time=0.199 ms

--- 10.0.1.20 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2050ms
rtt min/avg/max/mdev = 0.067/0.137/0.199/0.055 ms
^C
root@S1:/tmp/pycore.34897/S1.conf# ping 10.0.2.20
PING 10.0.2.20 (10.0.2.20) 56(84) bytes of data.
64 bytes from 10.0.2.20: icmp_seq=1 ttl=62 time=0.050 ms
64 bytes from 10.0.2.20: icmp_seq=2 ttl=62 time=0.091 ms
64 bytes from 10.0.2.20: icmp_seq=3 ttl=62 time=0.081 ms
^C
--- 10.0.2.20 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2050ms
rtt min/avg/max/mdev = 0.050/0.074/0.091/0.017 ms
root@S1:/tmp/pycore.34897/S1.conf# ping 10.0.3.20
PING 10.0.3.20 (10.0.3.20) 56(84) bytes of data.
64 bytes from 10.0.3.20: icmp_seq=1 ttl=61 time=0.058 ms
64 bytes from 10.0.3.20: icmp_seq=2 ttl=61 time=0.059 ms
64 bytes from 10.0.3.20: icmp_seq=3 ttl=61 time=0.076 ms
^C
--- 10.0.3.20 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2051ms
rtt min/avg/max/mdev = 0.058/0.064/0.076/0.010 ms
root@S1:/tmp/pycore.34897/S1.conf# 

```

Figura 2.9: Teste de conectividade e tabela de encaminhamento.

Como podemos observar pela figura 2.9, através do comando ping, o servidor S1 está novamente acessível.

2.3 Exercício 3

- 2.3.1 1) Considere que dispõe apenas do endereço de rede IP 130.XX.96.0/19, em que XX é o decimal correspondendo ao seu número de grupo (PLXX). Defina um novo esquema de endereçamento para as redes dos departamentos (mantendo a rede de acesso e core inalteradas) e atribua endereços às interfaces dos vários sistemas envolvidos. Assuma que todos os endereços de sub-redes são usáveis. Deve justificar as opções usadas.

O nosso grupo é o 14, logo, o nosso endereço IP é 130.14.96.0/19. Sabendo que queremos representar 4 departamentos, nós dividimos em /23 de forma a termos um número de subredes maior para facilitar e adicionar mais departamentos

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

O número de hosts de IP para cada departamento é 2 elevado ao número de bits não utilizados como identificador de rede, menos 2 (o endereço de broadcast e o endereço universal da rede.) A máscara de rede que escolhemos foi /23, ou seja, 130.14.96.0/23. Assim, temos 9 bits disponíveis para os hosts, o que significa que temos $2^9 - 2$ para cada departamento.

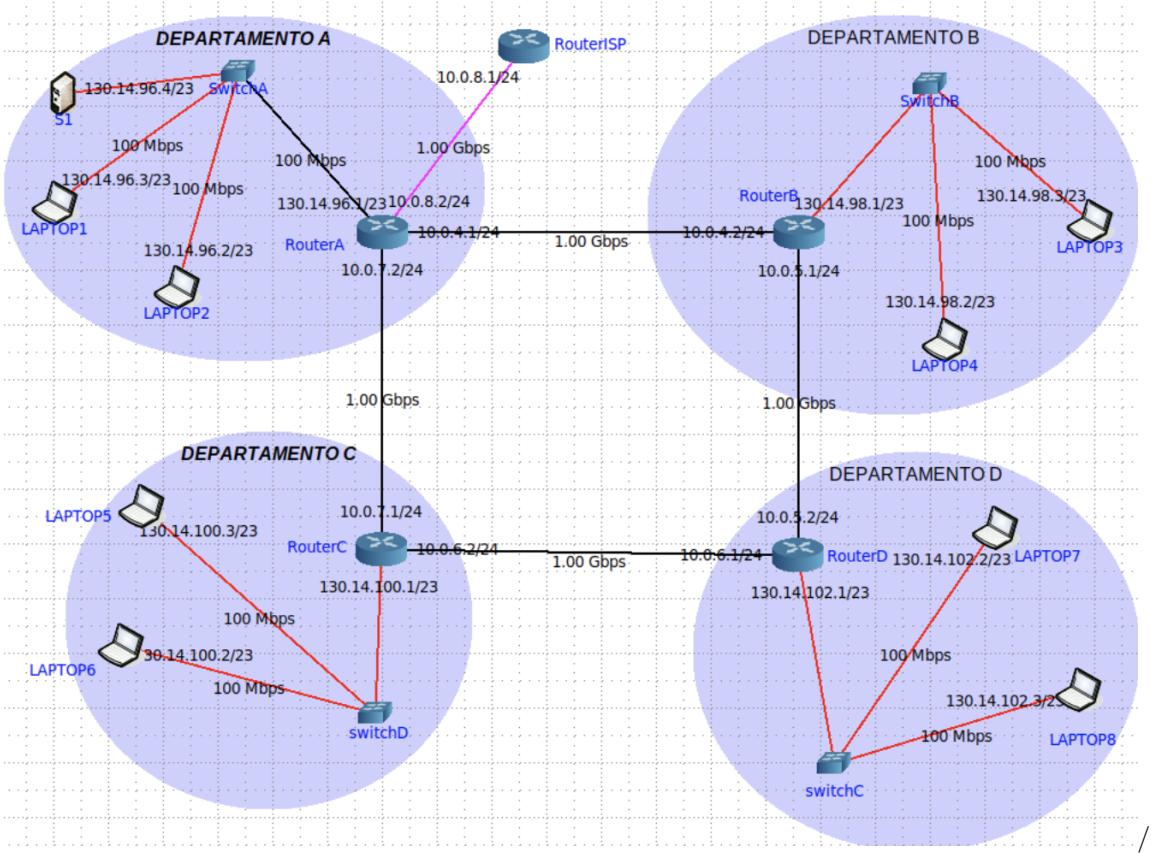


Figura 2.10: Divisão de endereços 130.14.96.0/19.

Departamento A: $2^9 - 2 = 510$

Departamento B: $2^9 - 2 = 510$

Departamento C: $2^9 - 2 = 510$

Departamento D: $2^9 - 2 = 510$

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

De forma a validar as alterações efetuadas ao esquema de endereçamento para as redes dos vários departamentos, fizemos vários pings a partir do servidor S1, para os hosts de todos os outros departamentos e verificamos também a tabela de endereçamento no do Router do departamento A.

Como podemos observar na primeira figura, através do comando ping, confirmamos que todos os *packets* transmitidos foram devidamente recebidos na extremidade oposta. Quanto à interpretação da tabela de endereçamento, conseguimos observar os novos endereços de destino. A gateway para o Departamento A é o endereço 0.0.0.0, como era esperado. Para os outros destinos, os endereços também correspondem ao esquema de endereçamento novo. Podemos ainda ver que, como foi usada a mascara de rede /23, os valores para o Germask 255.255.254.0.

```

root@S1:/tmp/pycore.34897/S1.conf# ping 130.14.100.2
PING 130.14.100.2 (130.14.100.2) 56(84) bytes of data.
64 bytes from 130.14.100.2: icmp_seq=1 ttl=62 time=0.112 ms
64 bytes from 130.14.100.2: icmp_seq=2 ttl=62 time=0.197 ms
64 bytes from 130.14.100.2: icmp_seq=3 ttl=62 time=0.132 ms
^C
--- 130.14.100.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2050ms
rtt min/avg/max/mdev = 0.112/0.147/0.197/0.036 ms
root@S1:/tmp/pycore.34897/S1.conf# ping 130.14.102.3
PING 130.14.102.3 (130.14.102.3) 56(84) bytes of data.
64 bytes from 130.14.102.3: icmp_seq=1 ttl=61 time=0.086 ms
64 bytes from 130.14.102.3: icmp_seq=2 ttl=61 time=0.163 ms
64 bytes from 130.14.102.3: icmp_seq=3 ttl=61 time=0.228 ms
^C
--- 130.14.102.3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2061ms
rtt min/avg/max/mdev = 0.086/0.159/0.228/0.058 ms
root@S1:/tmp/pycore.34897/S1.conf# ping 130.14.98.3
PING 130.14.98.3 (130.14.98.3) 56(84) bytes of data.
64 bytes from 130.14.98.3: icmp_seq=1 ttl=62 time=0.055 ms
64 bytes from 130.14.98.3: icmp_seq=2 ttl=62 time=0.156 ms
64 bytes from 130.14.98.3: icmp_seq=3 ttl=62 time=0.130 ms
^C
--- 130.14.98.3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2040ms
rtt min/avg/max/mdev = 0.055/0.113/0.156/0.044 ms
root@S1:/tmp/pycore.34897/S1.conf#

```

Figura 2.11: Conectividade do servidor 1- Departamento A para os restantes .

Kernel IP routing table								Iface
Destination	Gateway	Genmask	Flags	MSS	Window	irtt		
10.0.4.0	0.0.0.0	255.255.255.0	U	0	0			eth1
10.0.5.0	10.0.4.2	255.255.255.0	UG	0	0			eth1
10.0.6.0	10.0.7.1	255.255.255.0	UG	0	0			eth2
10.0.7.0	0.0.0.0	255.255.255.0	U	0	0			eth2
10.0.8.0	0.0.0.0	255.255.255.0	U	0	0			eth5
130.14.96.0	0.0.0.0	255.255.254.0	U	0	0			eth0
130.14.98.0	10.0.4.2	255.255.254.0	UG	0	0			eth1
130.14.100.0	10.0.7.1	255.255.254.0	UG	0	0			eth2
130.14.102.0	10.0.4.2	255.255.254.0	UG	0	0			eth1

Figura 2.12: Tabela de endereçamento do router do departamento A.

Capítulo 3

Conclusão

A realização deste trabalho facilitou a nossa compreensão dos assuntos lecionados nas aulas e enriquecer o nosso conhecimento relativo a redes. Para isso, foram usadas duas ferramentas: Core, para simulação de redes, e Wireshark, para captura de tráfego. Está dividido em duas partes em que cada uma delas consiste em 3 questões.

A primeira parte permitiu a nossa familiarização com a topologia CORE. O uso desta topologia ajudou também a consolidação dos nossos conhecimentos acerca do tráfego ICMP. Quanto ao datagrama IP, este trabalho permitiu-nos por em prática os conhecimentos que adquirimos nas aulas teóricas. Gostaríamos de notar que achámos bastante interessante a aplicação da fragmentação de datagramas.

Resumindo, todo o capítulo de Protocolo IP foi abrangido, desde a construção e posterior análise de um esquema de endereçamento de uma topologia CORE, onde relembramos conceitos como: as máscaras de rede, os endereços públicos e privados, tabelas de encaminhamentos, encaminhamento estático e dinâmico e comandos como o ping para testar a conectividade entre todos os departamentos. Todos os tópicos abordados nesta parte do trabalho prático foram assim praticados e consolidados.