

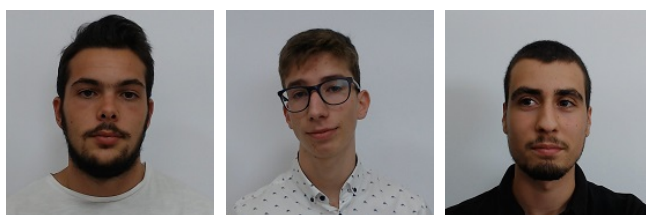


Universidade do Minho
Mestrado Integrado em Engenharia Informática
3ºano - 1º Semestre

Redes de Computadores

TP2: Protocolo IP

Grupos 1 - PL1



a83732 – Gonçalo Rodrigues Pinto
a84197 – João Pedro Araújo Parente
a84829 – José Nuno Martins da Costa

14 de Novembro de 2019

Conteúdo

1	Introdução	4
2	Questões e Resposta	4
2.1	Parte I	4
2.1.1	Questão 1	4
2.1.2	Questão 2	7
2.1.3	Questão 3	11
2.2	Parte II	14
2.2.1	Questão 1	14
2.2.2	Questão 2	18
2.2.3	Questão 3	22
3	Conclusão	26

Lista de Figuras

1	Topologia criada para verificar o comportamento do traceroute.	5
2	Execução do comando traceroute -I para o endereço IP do host s5	5
3	Tráfego ICMP enviado por s1 e o tráfego ICMP recebido como resposta	6
4	Tráfego capturado.	7
5	Primeira mensagem ICMP capturada.	7
6	Detalhes do IPv4 da primeira mensagem capturada.	8
7	Os pacotes capturados ordenados de acordo com o endereço IP da fonte	9
8	Comparação de 2 pacotes ICMP enviados seguidos.	9
9	Os pacotes capturados ordenados de acordo com o endereço IP destino.	10
10	O primeiro fragmento do datagrama IP segmentado.	11
11	O segundo fragmento do datagrama IP original.	12
12	O último fragmento do datagrama IP original que corresponde ao mesmo tempo à primeira mensagem ICMP registada. . . .	13
13	Topologia criada no CORE para reflectir uma rede local de uma empresa.	15
14	Conexão entre o laptop do departamento A e o servidor. . . .	16
15	Conexão entre o laptop do departamento B e o servidor. . . .	16
16	Conexão entre o laptop do departamento C e o servidor. . . .	17
17	Conexão entre o laptop do departamento D e o servidor. . . .	17
18	Conectividade IP do router de acesso Rext para o servidor S1.	18
19	Tabela de encaminhamento de um laptop do departamento B, neste caso n13.	19
20	Tabela de encaminhamento do router do departamento B. . . .	19
21	Processos a correr no router do departamento B.	20
22	Tabela de encaminhamento do servidor 1.	20
23	Tabela de encaminhamento do servidor 1 após termos executado o comando route delete.	20
24	Tentativa de comunicação de um laptop do departamento B com o servidor 1.	21
25	Adição de uma rota estática de forma a restaurar a conectividade para o servidor S1.	21
26	Teste com o objectivo de certificar que o servidor está novamente acessível.	22
27	Topologia CORE, depois de ser alterada.	23
28	Ping de um laptop do departamento A ao Servidor1.	24

29	Ping de um laptop do departamento B ao Servidor1.	25
30	Ping de um laptop do departamento C ao Servidor1.	25
31	Ping de um laptop do departamento D ao Servidor1.	26

1 Introdução

O principal objectivo deste trabalho foi o estudo do Internet Protocol (IP) nas suas principais vertentes, nomeadamente:

1. Estudo do formato de um pacote ou datagrama IP;
2. Fragmentação de pacotes IP;
3. Endereçamento IP;
4. Encaminhamento IP;

2 Questões e Resposta

2.1 Parte I

Na primeira parte deste trabalho foi realizado o registo de datagramas IP enviados e recebidos através da execução do programa traceroute. Foi analisado os vários campos de um datagrama IP e analisado o processo de fragmentação realizado pelo IP.

2.1.1 Questão 1

Após termos preparado uma topologia no CORE para verificar o comportamento do traceroute, ligamos um host s1 a um router r2, este a um router r3, este por sua vez ligado a um router r4, que por fim, ligou-se a um host h5.

- (a) **Active o wireshark ou o tcpdump no PC s1. Numa shell de s1, execute o comando traceroute -I para o endereço IP do host h5.**

R: Ativámos o software wireshark no host s1 como foi indicado e executamos o traceroute -I para o endereço IP do host h5, como se apresenta na figura 2.

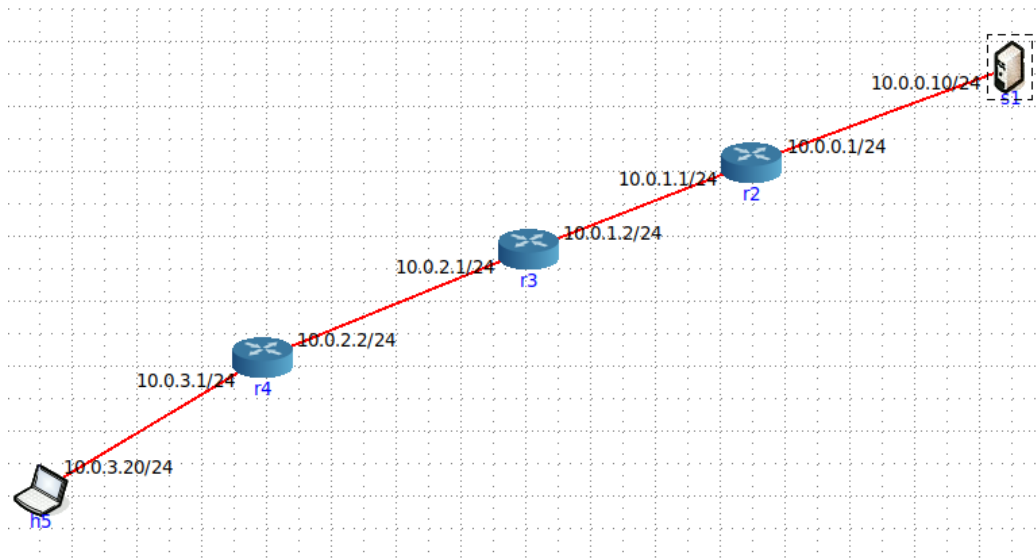


Figura 1: Topologia criada para verificar o comportamento do traceroute.

```

root@s1:/tmp/pycore.44215/s1.conf# traceroute -I 10.0.3.20
traceroute to 10.0.3.20 (10.0.3.20), 30 hops max, 60 byte packets
 1  10.0.0.1 (10.0.0.1)  0.067 ms  0.010 ms  0.007 ms
 2  10.0.1.2 (10.0.1.2)  0.020 ms  0.010 ms  0.009 ms
 3  10.0.2.2 (10.0.2.2)  0.020 ms  0.012 ms  0.014 ms
 4  10.0.3.20 (10.0.3.20)  0.058 ms  0.030 ms  0.027 ms
  
```

Figura 2: Execução do comando traceroute -I para o endereço IP do host s5

- (b) **Registre e analise o tráfego ICMP enviado por s1 e o tráfego ICMP recebido como resposta. Comente os resultados face ao comportamento esperado.**

R: Registou-se e analisou-se o tráfego ICMP (Internet Control Message Protocol) enviado por s1 e o tráfego ICMP recebido como resposta, como se apresenta na figura 3. A topologia criada possui 3 routers entre o servidor s1 e o host h5, cada router no percurso até ao destino decrementa de 1 o TTL (Time-to-live) de cada datagrama recebido. Após os resultados obtidos, podemos concluir que os valores registados foram ao encontro do esperado, isto é, para o servidor s1 comunicar com o host h5 os pacotes enviados tem que ter pelo menos o TTL igual a 4 pois cada router do percurso decrementa o TTL como foi referido anteriormente pois inicialmente o s1 tenta comunicar com h5 com um datagrama com TTL igual a 1 onde o router r1 devolve uma

mensagem de controlo ICMP ao host de origem, indicando que o TTL foi excedido, ou seja, o TTL atingiu o valor 0 e o router descartou o datagrama recebido; consequentemente o servidor aumenta o TTL para 2 contudo o router 2 descarta o datagrama recebido porque o TTL foi excedido; novamente o servidor tenta comunicar com o host aumentando o TTL para 3 contudo o router r3 descarta porque TTL foi mais uma vez excedido; após aumentar o TTL para 4 o servidor consegue finalmente comunicar com o host.

18	52.707284952	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request id=0x0025, seq=1/256, ttl=1 (no response found!)
19	52.707300932	10.0.0.1	10.0.0.10	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)
20	52.707311865	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request id=0x0025, seq=2/512, ttl=1 (no response found!)
21	52.707318422	10.0.0.1	10.0.0.10	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)
22	52.707323006	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request id=0x0025, seq=3/768, ttl=1 (no response found!)
23	52.707327476	10.0.0.1	10.0.0.10	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)
24	52.707332664	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request id=0x0025, seq=4/1024, ttl=2 (no response found!)
25	52.707335037	10.0.1.2	10.0.0.10	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)
26	52.707354851	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request id=0x0025, seq=5/1280, ttl=2 (no response found!)
27	52.707363024	10.0.1.2	10.0.0.10	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)
28	52.707367118	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request id=0x0025, seq=6/1536, ttl=2 (no response found!)
29	52.707371228	10.0.1.2	10.0.0.10	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)
30	52.707378895	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request id=0x0025, seq=7/1792, ttl=3 (no response found!)
31	52.707396973	10.0.2.2	10.0.0.10	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)
32	52.707401432	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request id=0x0025, seq=8/2048, ttl=3 (no response found!)
33	52.707412144	10.0.2.2	10.0.0.10	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)
34	52.707416094	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request id=0x0025, seq=9/2304, ttl=3 (no response found!)
35	52.707428844	10.0.2.2	10.0.0.10	ICMP	102 Time-to-live exceeded (Time to live exceeded in transit)
36	52.707436107	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request id=0x0025, seq=10/2560, ttl=4 (reply in 37)
37	52.707491257	10.0.3.20	10.0.0.10	ICMP	74 Echo (ping) reply id=0x0025, seq=10/2560, ttl=61 (request in 36)
38	52.707502378	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request id=0x0025, seq=11/2816, ttl=4 (reply in 39)
39	52.707527712	10.0.3.20	10.0.0.10	ICMP	74 Echo (ping) reply id=0x0025, seq=11/2816, ttl=61 (request in 38)
40	52.707535487	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request id=0x0025, seq=12/3072, ttl=4 (reply in 41)
41	52.707558345	10.0.3.20	10.0.0.10	ICMP	74 Echo (ping) reply id=0x0025, seq=12/3072, ttl=61 (request in 40)
42	52.707567060	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request id=0x0025, seq=13/3328, ttl=5 (reply in 43)
43	52.707589203	10.0.3.20	10.0.0.10	ICMP	74 Echo (ping) reply id=0x0025, seq=13/3328, ttl=61 (request in 42)
44	52.707596172	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request id=0x0025, seq=14/3584, ttl=5 (reply in 45)
45	52.707616981	10.0.3.20	10.0.0.10	ICMP	74 Echo (ping) reply id=0x0025, seq=14/3584, ttl=61 (request in 44)
46	52.707623954	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request id=0x0025, seq=15/3840, ttl=5 (reply in 47)
47	52.707643445	10.0.3.20	10.0.0.10	ICMP	74 Echo (ping) reply id=0x0025, seq=15/3840, ttl=61 (request in 46)
48	52.707651540	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request id=0x0025, seq=16/4096, ttl=6 (reply in 49)
49	52.707672344	10.0.3.20	10.0.0.10	ICMP	74 Echo (ping) reply id=0x0025, seq=16/4096, ttl=61 (request in 48)

Figura 3: Tráfego ICMP enviado por s1 e o tráfego ICMP recebido como resposta

- (c) Qual deve ser o valor inicial mínimo do campo TTL para alcançar o destino h5? Verifique na prática que a sua resposta está correta.

R: Deduziu-se que o valor inicial mínimo do campo TTL para alcançar o destino s5 deve ser de 4 pois é o valor adequado para os pacotes alcançarem o destino pretendido dado haver 3 routers no percurso. Temos a oportunidade de comprovar este valor através da figura 3.

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

R: O valor médio do tempo de ida-e-volta obtido foi 0,038(3) este valor resultou da média dos valores do linha número 4 da figura 2 que corresponde ao TTL igual a 4 pois foi quando os pacotes enviados chegaram ao host h5.

2.1.2 Questão 2

De forma a gerar datagramas IP de diferentes tamanhos, utilizámos o programa Pingplotter para mudar o tamanho das mensagens a enviar porque o sistema operativo da nossa máquina nativa foi Windows desta forma permitiu-nos maior flexibilidade para efectuar o traceroute. Usando o programa Wireshark capturámos o tráfego com um tamanho predefinido pelo Pingplotter que foi de 56 bytes. Utilizou-se como máquina destino o host marco.uminho.pt. Com base no tráfego capturado, identificámos os pedidos ICMP Echo Request e o conjunto de mensagens devolvidas em resposta a esses pedidos. De seguida analisou-se a primeira mensagem ICMP capturada e foi efectuada uma análise ao nível protocolar. Partindo da análise do cabeçalho IP dessa mensagem foi possível responder as seguintes questões.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	Cisco_Sb:13:52	Spanning-tree-for...	STP	60	Conf. Root = 4096/720/00:0a:8a:97:74:80 Cost = 3004 Port = 0x8012
2	0.054738	fe80::e8c6:2fe2:e5...	ff02::c	UDP	718	58275 → 3702 Len=656
3	1.659723	192.168.100.178	239.255.255.250	SSDP	179	M-SEARCH * HTTP/1.1
4	1.829030	Tp-LinkT_26:31:4f	Broadcast	ARP	60	Who has 192.168.25.1? Tell 192.168.25.7
5	2.000243	Cisco_Sb:13:52	Spanning-tree-for...	STP	60	Conf. Root = 4096/720/00:0a:8a:97:74:80 Cost = 3004 Port = 0x8012
6	2.139418	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3983/36623, ttl=255 (reply in 7)
7	2.140890	193.136.9.240	192.168.100.172	ICMP	70	Echo (ping) reply id=0x0001, seq=3983/36623, ttl=62 (request in 6)
8	2.189916	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3984/36879, ttl=1 (no response found!)
9	2.191074	192.168.100.254	192.168.100.172	ICMP	98	Time-to-live exceeded (Time to live exceeded in transit)
10	2.239951	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3985/37135, ttl=2 (no response found!)
11	2.240924	193.136.9.240	192.168.100.172	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
12	2.290479	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3986/37391, ttl=3 (reply in 13)
13	2.291670	193.136.9.240	192.168.100.172	ICMP	70	Echo (ping) reply id=0x0001, seq=3986/37391, ttl=62 (request in 12)
14	2.342634	Cisco_Sb:13:52	Cisco_Sb:13:52	LOOP	60	Reply
15	2.823146	Tp-LinkT_26:31:4f	Broadcast	ARP	60	Who has 192.168.25.1? Tell 192.168.25.7
16	3.086383	192.168.100.202	224.0.0.251	MDNS	103	Standard query 0x0001 PTR _googlecast._tcp.local, "QU" question PTR _goo...
17	3.086383	192.168.100.202	224.0.0.251	MDNS	103	Standard query 0x0001 PTR _googlecast._tcp.local, "QU" question PTR _goo...
18	3.822411	Tp-LinkT_26:31:4f	Broadcast	ARP	60	Who has 192.168.25.1? Tell 192.168.25.7
19	3.902448	192.168.100.202	239.255.255.250	SSDP	167	M-SEARCH * HTTP/1.1
20	4.003951	Cisco_Sb:13:52	Spanning-tree-for...	STP	60	Conf. Root = 4096/720/00:0a:8a:97:74:80 Cost = 3004 Port = 0x8012
21	4.320566	fe80::9417:e7f4:cb...	ff02::2	ICMPv6	86	Multicast Listener Done
22	4.321368	fe80::ee08:6bff:fe...	ff02::1	ICMPv6	86	Multicast Listener Query
23	4.340875	192.168.100.187	239.255.255.250	SSDP	179	M-SEARCH * HTTP/1.1

Figura 4: Tráfego capturado.

No.	Time	Source	Destination	Protocol	Length	Info
6	2.139418	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3983/36623, ttl=255 (reply in 7)

Frame 6: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface 0
 Ethernet II, Src: AsustekC_1b:1b:73 (88:d7:f6:1b:1b:73), Dst: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
 Internet Protocol Version 4, Src: 192.168.100.172, Dst: 193.136.9.240
 Internet Control Message Protocol

Figura 5: Primeira mensagem ICMP capturada.

- (a) **Qual é o endereço IP da interface ativa do seu computador?**
R: O endereço IP da interface ativa do nosso computador foi 192.168.100.172, como se pode comprovar na figura 5 no campo "Source".
- (b) **Qual é o valor do campo protocolo? O que identifica?**
R: O valor do campo protocolo é ICMP ou Internet Control Message Protocol identifica as mensagens de controlo de máquina para máquina, sendo que estas podem ser de sucesso ou de erro.

```

No.      Time          Source            Destination      Protocol Length Info
 6 2.139418      192.168.100.172    193.136.9.240    ICMP      70      Echo (ping) request id=0x0001, seq=3983/36623,
ttl=255 (reply in 7)
Frame 6: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface 0
Ethernet II, Src: AsustekC_1b:1b:73 (88:d7:f6:1b:1b:73), Dst: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
Internet Protocol Version 4, Src: 192.168.100.172, 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: 56
 Identification: 0x55ab (21931)
 Flags: 0x0000
  0... .. = Reserved bit: Not set
  .0.. .. = Don't fragment: Not set
  ..0. .. = More fragments: Not set
  ...0 0000 0000 0000 = Fragment offset: 0
 Time to live: 255
 Protocol: ICMP (1)
 Header checksum: 0x0000 [validation disabled]
 [Header checksum status: Unverified]
 Source: 192.168.100.172
 Destination: 193.136.9.240
Internet Control Message Protocol

```

Figura 6: Detalhes do IPv4 da primeira mensagem capturada.

- (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?**
R: O cabeçalho do IP(v4) tem 20 bytes. O campo de dados (payload) têm 36 bytes. Para calcular o tamanho do payload faz-se o tamanho total, que neste caso foi de 56 bytes, menos o tamanho do cabeçalho do IP(v4) . A figura 6 comprova esta afirmação.
- (d) **O datagrama IP foi fragmentado? Justifique.**
R: O datagrama IP não foi fragmentado porque o tamanho dos pacotes enviados, que neste caso foi de 56 bytes, que é menor que o Maximum Transmission Unit ou MTU que é o tamanho máximo (1500 bytes) que um pacote pode ter para ser enviado sem ser fragmentado.A figura 6 fundamenta esta resposta.

No.	Time	Source	Destination	Protocol	Length	Info
6	2.139418	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3983/36623, ttl=255 (reply in 7)
8	2.189916	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3984/36679, ttl=1 (no response found!)
10	2.239951	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3985/37135, ttl=2 (no response found!)
12	2.299479	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3986/37391, ttl=3 (reply in 13)
41	4.641220	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3987/37647, ttl=255 (reply in 42)
48	4.691948	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3988/37903, ttl=1 (no response found!)
50	4.742902	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3989/38159, ttl=2 (no response found!)
53	4.792948	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3990/38415, ttl=3 (reply in 54)
87	7.141222	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3991/38671, ttl=255 (reply in 88)
89	7.191646	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3992/38927, ttl=1 (no response found!)
91	7.242218	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3993/39183, ttl=2 (no response found!)
93	7.292246	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3994/39439, ttl=3 (reply in 94)
102	8.642286	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3995/39695, ttl=255 (reply in 103)
104	8.692961	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3996/39951, ttl=1 (no response found!)
106	9.743227	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3997/40207, ttl=2 (no response found!)
108	9.793802	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3998/40463, ttl=3 (reply in 109)
137	12.142781	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=3999/40719, ttl=255 (reply in 138)
140	12.193333	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=4000/40975, ttl=1 (no response found!)
142	12.244200	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=4001/41231, ttl=2 (no response found!)
144	12.294166	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=4002/41487, ttl=3 (reply in 145)
161	14.643485	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=4003/41743, ttl=255 (reply in 162)
163	14.693497	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=4004/41999, ttl=1 (no response found!)
165	14.744189	192.168.100.172	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=4005/42255, ttl=2 (no response found!)

Figura 7: Os pacotes capturados ordenados de acordo com o endereço IP da fonte .

Field	Packet 10 (Left)	Packet 8 (Right)
Frame	10: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface 4	8: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface 4
Ethernet II	Src: AsustekC_Lb:1b:73 (88:d7:f6:1b:73), Dst: Vmware_d2:19:f0 (08:00:0e:54:00:19)	Src: AsustekC_Lb:1b:73 (88:d7:f6:1b:73), Dst: Vmware_d2:19:f0 (08:00:0e:54:00:19)
Internet Protocol Version 4	Src: 192.168.100.172, Dst: 193.136.9.240	Src: 192.168.100.172, Dst: 193.136.9.240
ICMP	Version: 4, Header Length: 20 bytes (5), Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT), Total Length: 56, Identification: 0x55ad (21933), Flags: 0x0000, Time to live: 2, Protocol: ICMP (1), Header checksum: 0x0000 [validation disabled], Source: 192.168.100.172, Destination: 193.136.9.240	Version: 4, Header Length: 20 bytes (5), Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT), Total Length: 56, Identification: 0x55ac (21932), Flags: 0x0000, Time to live: 1, Protocol: ICMP (1), Header checksum: 0x0000 [validation disabled], Source: 192.168.100.172, Destination: 193.136.9.240
Internet Control Message Protocol		

Figura 8: Comparação de 2 pacotes ICMP enviados seguidos.

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

R: Na sequência de mensagens ICMP enviadas pelo nosso computador, os campos do cabeçalho IP que variaram de pacote para pacote foram o TTL e a identificação do datagrama IP.

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

R: Os valores do campo de Identificação do datagrama IP e TTL foram incrementados.

No.	Time	Source	Destination	Protocol	Length	Info
7	2.148999	193.136.9.248	192.168.100.172	ICMP	78	Echo (ping) reply id=0x0001, seq=3983/36623, ttl=62 (request in 6)
9	2.151074	192.168.100.254	192.168.100.172	ICMP	98	Time-to-live exceeded (Time to live exceeded in transit)
11	2.240744	193.136.19.254	192.168.100.172	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
13	2.291678	193.136.9.248	192.168.100.172	ICMP	78	Echo (ping) reply id=0x0001, seq=3986/37391, ttl=62 (request in 12)
42	4.642036	193.136.9.248	192.168.100.172	ICMP	78	Echo (ping) reply id=0x0001, seq=3987/37647, ttl=62 (request in 41)
49	4.692858	192.168.100.254	192.168.100.172	ICMP	98	Time-to-live exceeded (Time to live exceeded in transit)
51	4.744802	193.136.19.254	192.168.100.172	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
54	4.793825	193.136.9.248	192.168.100.172	ICMP	78	Echo (ping) reply id=0x0001, seq=3990/38415, ttl=62 (request in 53)
88	7.142325	193.136.9.248	192.168.100.172	ICMP	78	Echo (ping) reply id=0x0001, seq=3991/38671, ttl=62 (request in 87)
90	7.192409	192.168.100.254	192.168.100.172	ICMP	98	Time-to-live exceeded (Time to live exceeded in transit)
92	7.243323	193.136.19.254	192.168.100.172	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
94	7.293317	193.136.9.248	192.168.100.172	ICMP	78	Echo (ping) reply id=0x0001, seq=3994/39439, ttl=62 (request in 93)
103	9.643396	193.136.9.248	192.168.100.172	ICMP	78	Echo (ping) reply id=0x0001, seq=3995/39695, ttl=62 (request in 102)
105	9.694055	192.168.100.254	192.168.100.172	ICMP	98	Time-to-live exceeded (Time to live exceeded in transit)
107	9.744286	193.136.19.254	192.168.100.172	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
109	9.794920	193.136.9.248	192.168.100.172	ICMP	78	Echo (ping) reply id=0x0001, seq=3998/48463, ttl=62 (request in 108)
138	12.143883	193.136.9.248	192.168.100.172	ICMP	78	Echo (ping) reply id=0x0001, seq=3999/48719, ttl=62 (request in 137)
141	12.194451	192.168.100.254	192.168.100.172	ICMP	98	Time-to-live exceeded (Time to live exceeded in transit)
143	12.249345	193.136.19.254	192.168.100.172	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)
145	12.299323	193.136.9.248	192.168.100.172	ICMP	78	Echo (ping) reply id=0x0001, seq=4002/41407, ttl=62 (request in 144)
162	14.680313	193.136.9.248	192.168.100.172	ICMP	78	Echo (ping) reply id=0x0001, seq=4003/41743, ttl=62 (request in 161)
164	14.694649	192.168.100.254	192.168.100.172	ICMP	98	Time-to-live exceeded (Time to live exceeded in transit)
166	14.745286	193.136.19.254	192.168.100.172	ICMP	78	Time-to-live exceeded (Time to live exceeded in transit)

Figura 9: Os pacotes capturados ordenados de acordo com o endereço IP destino.

- (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?

R: O valor do campo TTL foi bastante grande contudo não foi constante para todas as mensagens de resposta ICMP TTL exceeded enviados ao nosso computador isto acontece por causa da implementação de cada router onde o TTL é excedido e de forma a avisar a máquina que tentou enviar o pacote que não foi possível efectuar a comunicação envia uma mensagem ICMP com TTL o suficientemente grande variando de fabricante para fabricante de forma a que passe por uma enorme quantidade de routers e o host de origem saiba que não foi possível haver comunicação. A figura 8 fundamenta a nossa resposta.

2.1.3 Questão 3

Pretendeu-se analisar a fragmentação de pacotes IP para isso definimos no programa pingplotter o tamanho do pacote a enviar de 4201 bytes. Partindo da ordenação do tráfego capturado usando a coluna do tempo de captura é possível responder as diferentes questões.

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

R: Existiu a necessidade de fragmentar o pacote inicial porque o pacote enviado tem um tamanho de 4201 bytes que é maior que o MTU, que é o tamanho máximo, como foi dito anteriormente tem um valor de 1500 bytes, que um dado pacote pode ter para ser enviado de uma vez só.

```
No.      Time          Source          Destination      Protocol Length Info
 66 14.594386      192.168.100.172 193.136.9.240    IPv4          1514    Fragmented IP protocol (proto=ICMP 1, off=0,
ID=6047) [Reassembled in #68]
Frame 66: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface 0
Ethernet II, Src: AsustekC_1b:1b:73 (88:d7:f6:1b:1b:73), Dst: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
Internet Protocol Version 4, Src: 192.168.100.172, 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: 0x6047 (24647)
 Flags: 0x2000, 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: 255
 Protocol: ICMP (1)
 Header checksum: 0x0000 [validation disabled]
 [Header checksum status: Unverified]
 Source: 192.168.100.172
 Destination: 193.136.9.240
 Reassembled IPv4 in frame: 68
 Data (1480 bytes)
```

Figura 10: O primeiro fragmento do datagrama IP segmentado.

- (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?**

R: A informação no cabeçalho que indica que o datagrama foi fragmentado, é através da secção das flags através dos campos "More fragments" e "Fragment offset", se "More fragments" estiver a "Set" no caso da figura 10, e "Fragment offset" a 0, podemos concluir que se trata do primeiro fragmento. O tamanho deste datagrama vai ser igual à MTU, dado que a razão para ter sido fragmentado foi devido ao facto de o tamanho do pacote ser maior que a MTU.

```

No.      Time            Source                Destination           Protocol Length Info
 67 14.594387      192.168.100.172      193.136.9.240         IPv4      1514    Fragmented IP protocol (proto=ICMP 1, off=1480,
ID=6047) [Reassembled in #68]
Frame 67: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface 0
Ethernet II, Src: AsustekC_1b:1b:73 (88:d7:f6:1b:1b:73), Dst: Vmware_d2:19:f0 (00:0c:29:d2:19:f0)
Internet Protocol Version 4, Src: 192.168.100.172, 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: 0x6047 (24647)
Flags: 0x20b9, More fragments
 0... .. = Reserved bit: Not set
 .0.. .. = Don't fragment: Not set
 ..1. .... = More fragments: Set
 ...0 0000 1011 1001 = Fragment offset: 185
Time to live: 255
Protocol: ICMP (1)
Header checksum: 0x0000 [validation disabled]
[Header checksum status: Unverified]
Source: 192.168.100.172
Destination: 193.136.9.240
Reassembled IPv4 in frame: 68
Data (1480 bytes)

```

Figura 11: O segundo fragmento do datagrama IP original.

- (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?**

R: Após analisar o segundo fragmento do datagrama IP original, através do campo "Fragment offset" presente no cabeçalho IP podemos concluir que não se trata do primeiro fragmento pois o campo é diferente de zero. Também é possível afirmar que existe mais fragmentos pois o campo "More fragments" indica "Set".

- (d) **Quantos fragmentos foram criados a partir do datagrama original? Como se detecta o último fragmento correspondente ao datagrama original?**

R: Foram criados 3 fragmentos a partir do datagrama original, para detectar o último fragmento do datagrama original tem que se observar o campo "More fragments" que tem de estar "Not Set" e o campo "Fragment offset" ser diferente de 0 e desta forma é possível detectar o último fragmento

2.2 Parte II

Na segunda parte continuou-se o estudo do protocolo IPv4 com ênfase no endereçamento e encaminhamento IP. Foram estudadas algumas das técnicas mais relevantes que foram propostas para aumentar a escalabilidade do protocolo IP, mitigar a exaustão dos endereços IPv4 (as técnicas estudadas apenas solucionam o problema a curto prazo, a solução para responder ao aumento significativo do número de endereços é necessário o uso progressivo do IPv6) e também reduzir os recursos de memória necessários nos routers para manter as tabelas de encaminhamento.

2.2.1 Questão 1

Após termos preparado uma topologia no CORE para reflectir uma rede local de uma empresa constituída por quatro departamentos (A, B, C e D) e cada departamento possui um router de acesso à sua rede local. Estes routers de acesso (Ra, Rb, Rc, e Rd) estão interligados entre si por ligações Ethernet a 1 Gbps, formando um anel. Por sua vez, existe um servidor (S1) na rede do departamento A e, pelo menos, três laptops por departamento, interligados ao router respetivo através de um comutador (switch). S1 tem uma ligação a 1 Gbps e os laptops ligações a 100 Mbps. Considere apenas a existência de um comutador por departamento. A conectividade IP externa da organização é assegurada através de um router de acesso Rext conectado a Rd por uma ligação ponto-a-ponto a 1 Gbps. Atendendo aos endereços IP atribuídos automaticamente pelo CORE aos diferentes equipamentos da topologia podemos responder às seguintes questões.

- (a) **Indique que endereços IP e máscaras de rede foram atribuídos pelo CORE a cada equipamento.**

R: Para simplificar, incluímos a imagem apresentada abaixo que ilustra de forma clara a topologia definida e o endereçamento usado.

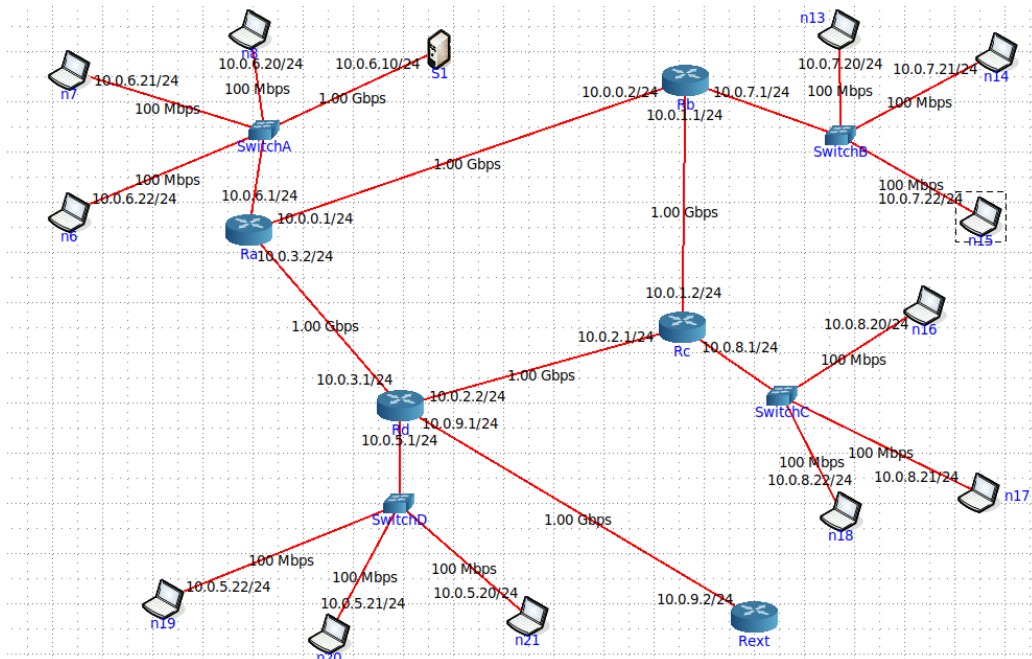


Figura 13: Topologia criada no CORE para reflectir uma rede local de uma empresa.

(b) **Trata-se de endereços públicos ou privados? Porquê?**

R: Os endereços IP atribuídos automaticamente pelo CORE aos diferentes equipamentos da topologia tratam-se de endereços privados porque os endereços atribuídos automaticamente encontram-se no intervalo 10.0.0.0 a 10.255.255.255/8, esta faixa de endereços IP atualmente é reservada a redes privadas que foi definido no RFC 1918.

(c) **Por que razão não é atribuído um endereço IP aos switches?**

R: Não foi atribuído um endereço IP aos switches porque é um "interruptor" de Internet, isto é, altera os pacotes Ethernet e neste nível não existe endereços IP.

(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).**

R: Para simplificar, incluímos as figuras apresentadas abaixo que ilustram de forma clara o comando ping entre os laptops de diferentes departamentos e o servidor do departamento A.


```
root@n8:/tmp/pycore.39003/n8.conf# ping 10.0.6.10
PING 10.0.6.10 (10.0.6.10) 56(84) bytes of data.
64 bytes from 10.0.6.10: icmp_seq=1 ttl=64 time=0.098 ms
64 bytes from 10.0.6.10: icmp_seq=2 ttl=64 time=0.045 ms
64 bytes from 10.0.6.10: icmp_seq=3 ttl=64 time=0.065 ms
64 bytes from 10.0.6.10: icmp_seq=4 ttl=64 time=0.064 ms
64 bytes from 10.0.6.10: icmp_seq=5 ttl=64 time=0.046 ms
64 bytes from 10.0.6.10: icmp_seq=6 ttl=64 time=0.050 ms
^C
--- 10.0.6.10 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 111ms
rtt min/avg/max/mdev = 0.045/0.061/0.098/0.019 ms
```

Figura 14: Conexão entre o laptop do departamento A e o servidor.

```
root@n13:/tmp/pycore.39003/n13.conf# ping 10.0.6.10
PING 10.0.6.10 (10.0.6.10) 56(84) bytes of data.
64 bytes from 10.0.6.10: icmp_seq=1 ttl=62 time=0.228 ms
64 bytes from 10.0.6.10: icmp_seq=2 ttl=62 time=0.112 ms
64 bytes from 10.0.6.10: icmp_seq=3 ttl=62 time=0.085 ms
64 bytes from 10.0.6.10: icmp_seq=4 ttl=62 time=0.120 ms
64 bytes from 10.0.6.10: icmp_seq=5 ttl=62 time=0.117 ms
64 bytes from 10.0.6.10: icmp_seq=6 ttl=62 time=0.112 ms
^C
--- 10.0.6.10 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 114ms
rtt min/avg/max/mdev = 0.085/0.129/0.228/0.045 ms
```

Figura 15: Conexão entre o laptop do departamento B e o servidor.

```

root@n16:/tmp/pycore.39003/n16.conf# ping 10.0.6.10
PING 10.0.6.10 (10.0.6.10) 56(84) bytes of data.
64 bytes from 10.0.6.10: icmp_seq=1 ttl=61 time=0.256 ms
64 bytes from 10.0.6.10: icmp_seq=2 ttl=61 time=0.146 ms
64 bytes from 10.0.6.10: icmp_seq=3 ttl=61 time=0.138 ms
64 bytes from 10.0.6.10: icmp_seq=4 ttl=61 time=0.176 ms
64 bytes from 10.0.6.10: icmp_seq=5 ttl=61 time=0.137 ms
64 bytes from 10.0.6.10: icmp_seq=6 ttl=61 time=0.136 ms
^C
--- 10.0.6.10 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 115ms
rtt min/avg/max/mdev = 0.136/0.164/0.256/0.046 ms

```

Figura 16: Conexão entre o laptop do departamento C e o servidor.

```

root@n20:/tmp/pycore.39003/n20.conf# ping 10.0.6.10
PING 10.0.6.10 (10.0.6.10) 56(84) bytes of data.
64 bytes from 10.0.6.10: icmp_seq=1 ttl=62 time=0.080 ms
64 bytes from 10.0.6.10: icmp_seq=2 ttl=62 time=0.138 ms
64 bytes from 10.0.6.10: icmp_seq=3 ttl=62 time=0.116 ms
64 bytes from 10.0.6.10: icmp_seq=4 ttl=62 time=0.119 ms
64 bytes from 10.0.6.10: icmp_seq=5 ttl=62 time=0.123 ms
64 bytes from 10.0.6.10: icmp_seq=6 ttl=62 time=0.096 ms
^C
--- 10.0.6.10 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 107ms
rtt min/avg/max/mdev = 0.080/0.112/0.138/0.018 ms

```

Figura 17: Conexão entre o laptop do departamento D e o servidor.

- (e) **Verifique se existe conectividade IP do router de acesso Rext para o servidor S1.**

R: Para simplificar, incluímos a figura apresentada abaixo que ilustra de forma clara conectividade IP do router de acesso Rext para o servidor S1.

```

root@Rext:/tmp/pycore.39003/Rext.conf# ping 10.0.6.10
PING 10.0.6.10 (10.0.6.10) 56(84) bytes of data.
64 bytes from 10.0.6.10: icmp_seq=1 ttl=62 time=0.093 ms
64 bytes from 10.0.6.10: icmp_seq=2 ttl=62 time=0.121 ms
64 bytes from 10.0.6.10: icmp_seq=3 ttl=62 time=0.121 ms
64 bytes from 10.0.6.10: icmp_seq=4 ttl=62 time=0.124 ms
64 bytes from 10.0.6.10: icmp_seq=5 ttl=62 time=0.109 ms
64 bytes from 10.0.6.10: icmp_seq=6 ttl=62 time=0.119 ms
^C
--- 10.0.6.10 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 132ms
rtt min/avg/max/mdev = 0.093/0.114/0.124/0.015 ms

```

Figura 18: Conectividade IP do router de acesso Rext para o servidor S1.

2.2.2 Questão 2

Após termos preparado uma topologia no CORE como foi dito anteriormente para reflectir uma rede local de uma empresa, apresentada na figura 13. Partindo da análise do router e um laptop do departamento B, podemos responder às seguintes questões.

- (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`).**

R: Tanto os routers como os laptops dos diferentes departamentos, possuem uma tabela de encaminhamento.

As entradas da tabela incluem: *1ª coluna*-Endereço da Rede de Destino; *2ª coluna*-Endereço IP da interface de entrega (next hop); *3ª coluna* - Máscara da rede para uma determinada rota; *4ª coluna*-Opções que descrevem a rota; *5ª coluna*-O MSS é o Tamanho Máximo do Segmento e o tamanho do maior datagrama que o kernel construirá para transmissão para uma dada rota; *6ª coluna*-Janela é a quantidade máxima de dados que o sistema aceitará em uma única rajada de um host remoto; *7ª coluna*-O acrônimo irtt significa "tempo inicial de ida e volta"; *8ª Coluna*-Identificador da interface de saída da máquina local (Iface).

A entrega (forwarding), ou salto (hop) seguinte de um datagrama IP, é decidida em função do endereço IP destino do datagrama.

Por exemplo, se o laptop pretender comunicar com outro laptop do mesmo departamento (10.0.7.0/24) , pela segunda linha da tabela de encaminhamento, que se pode observar na figura 19, então o pacote é enviado para a rede local. Por outro lado, se pretender comunicar com outro departamento ou servidor é utilizado a primeira linha da tabela de encaminhamento onde é enviado a informação ao router 'Rb' (10.0.7.1) pela interface eth0 onde este trata de encaminhar para o destino pretendido.

```
root@n13:/tmp/pycore.33565/n13.conf# netstat -rn
Tabela de Roteamento IP do Kernel
Destino      Roteador      MáscaraGen.   Opções   MSS Janela   irtt Iface
0.0.0.0      10.0.7.1      0.0.0.0       UG       0 0       0 eth0
10.0.7.0     0.0.0.0       255.255.255.0 U        0 0       0 eth0
root@n13:/tmp/pycore.33565/n13.conf#
```

Figura 19: Tabela de encaminhamento de um laptop do departamento B, neste caso n13.

```
root@Rb:/tmp/pycore.39003/Rb.conf# netstat -rn
Tabela de Roteamento IP do Kernel
Destino      Roteador      MáscaraGen.   Opções   MSS Janela   irtt Iface
10.0.0.0     0.0.0.0       255.255.255.0 U        0 0       0 eth0
10.0.1.0     0.0.0.0       255.255.255.0 U        0 0       0 eth1
10.0.2.0     10.0.1.2      255.255.255.0 UG       0 0       0 eth1
10.0.3.0     10.0.0.1      255.255.255.0 UG       0 0       0 eth0
10.0.5.0     10.0.0.1      255.255.255.0 UG       0 0       0 eth0
10.0.6.0     10.0.0.1      255.255.255.0 UG       0 0       0 eth0
10.0.7.0     0.0.0.0       255.255.255.0 U        0 0       0 eth2
10.0.8.0     10.0.1.2      255.255.255.0 UG       0 0       0 eth1
10.0.9.0     10.0.0.1      255.255.255.0 UG       0 0       0 eth0
```

Figura 20: Tabela de encaminhamento do router do departamento B.

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

R: Como se pode observar na figura 21, ao examinar os processos do router do departamento B, com o comando 'top', deparamo-nos com o processo 'ospfd' a ser executado pelo user 'quagga', este processo está constantemente a atualizar as tabelas de encaminhamento das máquinas. Tornando assim o encaminhamento dinâmico.

PID	USUARIO	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TEMPO+	COMANDO
1	root	20	0	2768	1564	1424	S	0,0	0,0	0:00.01	vnoded
54	quagga	20	0	4904	2980	2004	S	0,0	0,0	0:00.00	zebra
64	quagga	20	0	4700	2900	2152	S	0,0	0,0	0:00.00	ospf6d
68	quagga	20	0	5120	2804	1968	S	0,0	0,0	0:00.00	ospfd
76	root	20	0	10704	4728	3280	S	0,0	0,1	0:00.02	bash
82	root	20	0	10704	4704	3260	S	0,0	0,1	0:00.02	bash
88	root	20	0	11772	3700	3228	R	0,0	0,1	0:00.00	top
89	root	20	0	10276	956	832	S	0,0	0,0	0:00.00	ping

Figura 21: Processos a correr no router do departamento B.

- (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 implicação tem esta medida para os utilizadores da empresa que acedem ao servidor? Justifique.

R: Ao retirar-se a rota por defeito da tabela de encaminhamento do servidor S1 localizado no departamento A por questões administrativas esta medida traz consequências para os utilizadores que acedem ao servidor pois não conseguem estabelecer conexão com o servidor, isto é, vão conseguir enviar pacotes para o servidor mas este não vai conseguir responder porque falta a rota por defeito o que leva a que os pacotes transmitidos sejam todos perdidos, como é demonstrado na figura 24.

```
root@S1:/tmp/pycore.39003/S1.conf# netstat -rn
```

Tabela de Roteamento IP do Kernel						
Destino	Roteador	MáscaraGen.	Opções	MSS	Janela	irtt Iface
0.0.0.0	10.0.6.1	0.0.0.0	UG	0	0	0 eth0
10.0.6.0	0.0.0.0	255.255.255.0	U	0	0	0 eth0

Figura 22: Tabela de encaminhamento do servidor 1.

```
root@S1:/tmp/pycore.39003/S1.conf# route delete default
root@S1:/tmp/pycore.39003/S1.conf# netstat -rn
```

Tabela de Roteamento IP do Kernel						
Destino	Roteador	MáscaraGen.	Opções	MSS	Janela	irtt Iface
10.0.6.0	0.0.0.0	255.255.255.0	U	0	0	0 eth0

Figura 23: Tabela de encaminhamento do servidor 1 após termos executado o comando `route delete`.

```

root@n13:/tmp/pycore.39003/n13.conf# ping 10.0.6.10
PING 10.0.6.10 (10.0.6.10) 56(84) bytes of data.
^C
--- 10.0.6.10 ping statistics ---
7 packets transmitted, 0 received, 100% packet loss, time 150ms

```

Figura 24: Tentativa de comunicação de um laptop do departamento B com o servidor 1.

- (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.**

R: Com o intuito de restaurar a conectividade para o servidor S1 de forma a contornar a restrição imposta na alínea c), adicionámos a rota estática com o comando route add onde o destino é 10.0.7.0 que é o endereço da rede B. Para restaurar a conectividade aos outros departamentos, seria um processo análogo, sendo que o destino teria um endereço de rede correspondente.

```

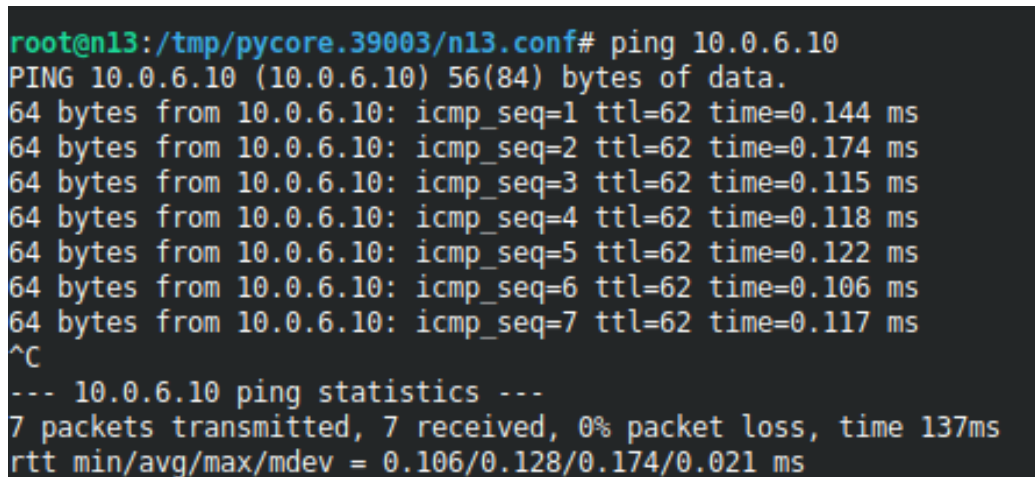
root@S1:/tmp/pycore.39003/S1.conf# route add -net 10.0.7.0 netmask 255.255.255.0 gw
10.0.6.1
root@S1:/tmp/pycore.39003/S1.conf# netstat -rn
Tabela de Roteamento IP do Kernel
Destino      Roteador      MáscaraGen.   Opções    MSS Janela   irtt Iface
10.0.6.0     0.0.0.0       255.255.255.0 U         0 0       0 eth0
10.0.7.0     10.0.6.1     255.255.255.0 UG        0 0       0 eth0

```

Figura 25: Adição de uma rota estática de forma a restaurar a conectividade para o servidor S1.

- (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.

R: Para simplificar, incluímos a figura apresentada abaixo que ilustra de forma clara que o servidor está novamente acessível utilizando para o efeito o comando ping.



```
root@n13:/tmp/pycore.39003/n13.conf# ping 10.0.6.10
PING 10.0.6.10 (10.0.6.10) 56(84) bytes of data.
64 bytes from 10.0.6.10: icmp_seq=1 ttl=62 time=0.144 ms
64 bytes from 10.0.6.10: icmp_seq=2 ttl=62 time=0.174 ms
64 bytes from 10.0.6.10: icmp_seq=3 ttl=62 time=0.115 ms
64 bytes from 10.0.6.10: icmp_seq=4 ttl=62 time=0.118 ms
64 bytes from 10.0.6.10: icmp_seq=5 ttl=62 time=0.122 ms
64 bytes from 10.0.6.10: icmp_seq=6 ttl=62 time=0.106 ms
64 bytes from 10.0.6.10: icmp_seq=7 ttl=62 time=0.117 ms
^C
--- 10.0.6.10 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 137ms
rtt min/avg/max/mdev = 0.106/0.128/0.174/0.021 ms
```

Figura 26: Teste com o objectivo de certificar que o servidor está novamente acessível.

2.2.3 Questão 3

De forma a minimizar a falta de endereços IPv4 é comum a utilização de sub-redes. Além disso, a definição de sub-redes permite uma melhor organização do espaço de endereçamento das redes em questão. Para definir endereços de sub-rede é necessário usar a parte prevista para endereçamento de host, não sendo possível alterar o endereço de rede original. Considerámos a topologia definida anteriormente. Assumimos que o endereçamento entre os routers se mantém inalterado, contudo, o endereçamento em cada departamento foi redefinido.

- (a) Considere que dispõe apenas do endereço de rede IP 172.yyx.32.0/20, em que “yy” são os dígitos correspondendo ao seu número de grupo (Gyy) e “x” é o dígito correspondente ao seu turno prático (PLx). 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. Deve justificar as opções usadas.

R: O endereço que temos à nossa disposição é 172.11.32.0/20. Para simplificar, incluímos a figura apresentada abaixo que ilustra de forma clara que os diferentes endereços atribuídos às interfaces dos vários sistemas envolvidos. Como existe 4 departamentos nesta instituição vai equivaler a 4 sub-redes, para tal, precisamos de, no mínimo, 3 bits para as representar. Tirando aos 32 bits do endereço IP, os bits de máscara (20) e os bits que alocamos para as subredes (3), resta-nos então 9 bits para identificar as interfaces IP.

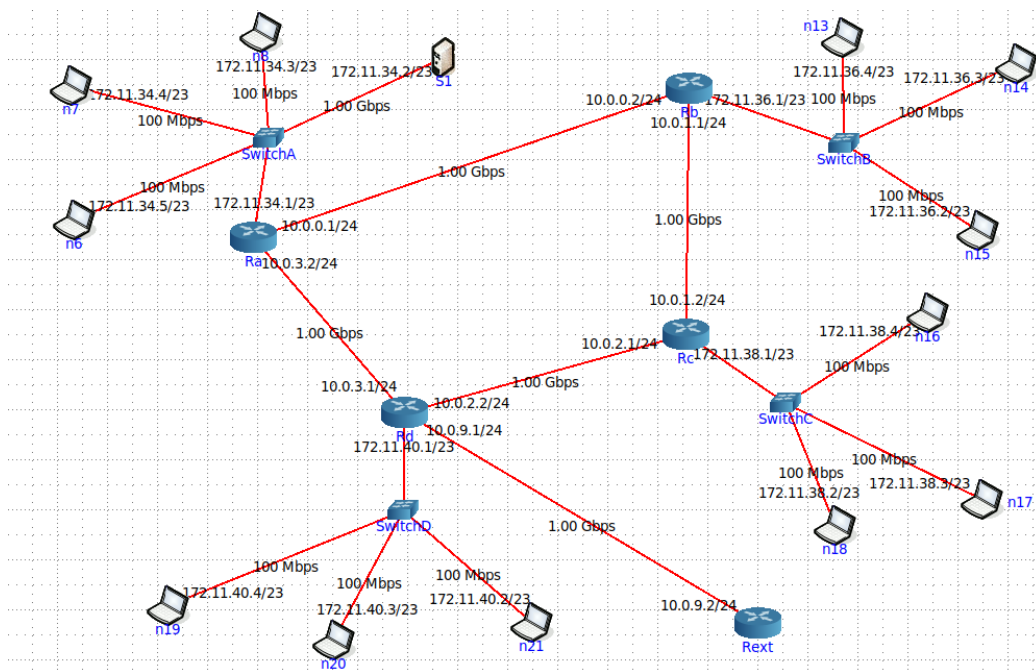


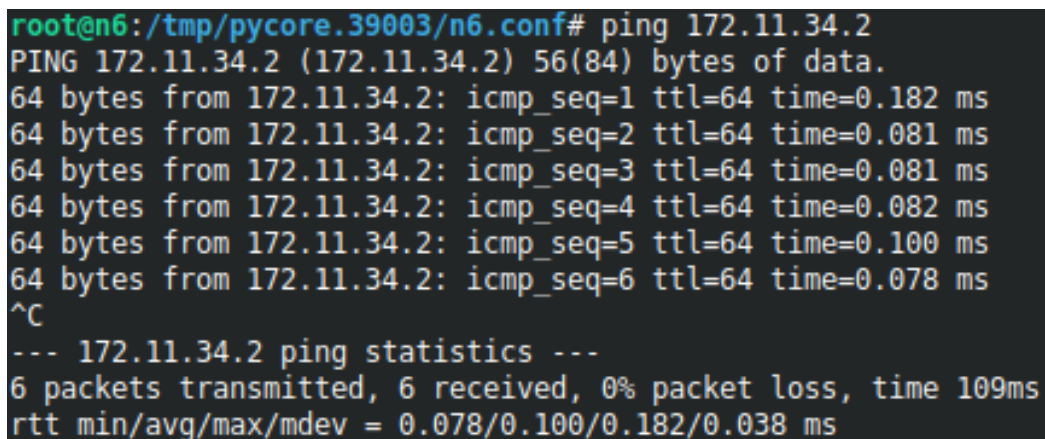
Figura 27: Topologia CORE, depois de ser alterada.

- (b) **Qual a máscara de rede que usou (em notação decimal)? Quantos interfaces IP pode interligar em cada departamento? Justifique.**

R: A máscara de rede que foi usada, em notação decimal, foi 255.255.254.0 ou em notação CIDR /23. Como definimos anteriormente que foi utilizado 9 bits para as interfaces IP, então podemos definir 512 (2^9) interfaces diferentes, sendo que temos que remover ainda as duas interfaces reservadas (os bits da interface todos a 0 e todos a 1), disponibilizando então 510 interfaces.

- (c) **Garanta e verifique que a conectividade IP entre as várias redes locais da organização MIEI-RC é mantida. Explique como procedeu.**

R: Para simplificar, incluímos as figuras apresentadas abaixo que ilustram de forma clara que a conectividade IP entre as várias redes locais da organização MIEI-RC é mantida.



```
root@n6:/tmp/pycore.39003/n6.conf# ping 172.11.34.2
PING 172.11.34.2 (172.11.34.2) 56(84) bytes of data.
64 bytes from 172.11.34.2: icmp_seq=1 ttl=64 time=0.182 ms
64 bytes from 172.11.34.2: icmp_seq=2 ttl=64 time=0.081 ms
64 bytes from 172.11.34.2: icmp_seq=3 ttl=64 time=0.081 ms
64 bytes from 172.11.34.2: icmp_seq=4 ttl=64 time=0.082 ms
64 bytes from 172.11.34.2: icmp_seq=5 ttl=64 time=0.100 ms
64 bytes from 172.11.34.2: icmp_seq=6 ttl=64 time=0.078 ms
^C
--- 172.11.34.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 109ms
rtt min/avg/max/mdev = 0.078/0.100/0.182/0.038 ms
```

Figura 28: Ping de um laptop do departamento A ao Servidor1.

```
root@n13:/tmp/pycore.39003/n13.conf# ping 10.0.6.10
PING 10.0.6.10 (10.0.6.10) 56(84) bytes of data.
64 bytes from 10.0.6.10: icmp_seq=1 ttl=62 time=0.228 ms
64 bytes from 10.0.6.10: icmp_seq=2 ttl=62 time=0.112 ms
64 bytes from 10.0.6.10: icmp_seq=3 ttl=62 time=0.085 ms
64 bytes from 10.0.6.10: icmp_seq=4 ttl=62 time=0.120 ms
64 bytes from 10.0.6.10: icmp_seq=5 ttl=62 time=0.117 ms
64 bytes from 10.0.6.10: icmp_seq=6 ttl=62 time=0.112 ms
^C
--- 10.0.6.10 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 114ms
rtt min/avg/max/mdev = 0.085/0.129/0.228/0.045 ms
```

Figura 29: Ping de um laptop do departamento B ao Servidor1.

```
root@n16:/tmp/pycore.39003/n16.conf# ping 172.11.34.2
PING 172.11.34.2 (172.11.34.2) 56(84) bytes of data.
64 bytes from 172.11.34.2: icmp_seq=1 ttl=61 time=0.274 ms
64 bytes from 172.11.34.2: icmp_seq=2 ttl=61 time=0.133 ms
64 bytes from 172.11.34.2: icmp_seq=3 ttl=61 time=0.137 ms
64 bytes from 172.11.34.2: icmp_seq=4 ttl=61 time=0.137 ms
64 bytes from 172.11.34.2: icmp_seq=5 ttl=61 time=0.136 ms
64 bytes from 172.11.34.2: icmp_seq=6 ttl=61 time=0.140 ms
^C
--- 172.11.34.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 125ms
rtt min/avg/max/mdev = 0.133/0.159/0.274/0.052 ms
```

Figura 30: Ping de um laptop do departamento C ao Servidor1.

```
root@n19:/tmp/pycore.39003/n19.conf# ping 172.11.34.2
PING 172.11.34.2 (172.11.34.2) 56(84) bytes of data.
64 bytes from 172.11.34.2: icmp_seq=1 ttl=62 time=0.190 ms
64 bytes from 172.11.34.2: icmp_seq=2 ttl=62 time=0.115 ms
64 bytes from 172.11.34.2: icmp_seq=3 ttl=62 time=0.057 ms
64 bytes from 172.11.34.2: icmp_seq=4 ttl=62 time=0.121 ms
64 bytes from 172.11.34.2: icmp_seq=5 ttl=62 time=0.057 ms
64 bytes from 172.11.34.2: icmp_seq=6 ttl=62 time=0.121 ms
^C
--- 172.11.34.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 111ms
rtt min/avg/max/mdev = 0.057/0.110/0.190/0.045 ms
```

Figura 31: Ping de um laptop do departamento D ao Servidor1.

3 Conclusão

O presente relatório descreveu, de forma sucinta, a resolução das questões propostas utilizando os softwares disponibilizados (Wireshark, CORE, Ping-Plotter) pelos docentes.

Após a realização deste trabalho, ficamos conscientes do formato de um pacote/datagrama IP, a fragmentação de pacotes IP, endereçamento IP e o encaminhamento IP.

Consideramos que os principais objetivos foram cumpridos, no entanto, há que ter em conta que na questão 3 da parte II após termos definido um novo esquema de endereçamento para as redes dos departamentos e a respetiva atribuição dos endereços às interfaces dos vários sistemas envolvidos ao executar no software CORE o esquema não é imediato estabelecer a conexão entre os diferentes departamentos, inicialmente as tabelas de encaminhamento encontram-se desatualizadas. Após alguns segundos as tabelas de encaminhamento são atualizadas devido aos processos que estão a executar nos routers ao mesmo tempo a tentar atualizar as diferentes tabelas de encaminhamento e depois de ser atualizado já é possível estabelecer a conexão entre os diferentes departamentos.

Sentimos que a realização deste trabalho prático consolidou os nossos conhecimentos do Internet Protocol(IPv4).