

Universidade do Minho
Departamento de Informática

TP2 - Protocolo IP (1ª parte)
Redes de Computadores
Grupo 37

Catarina Pais Vieira (a89524)

José Duarte Pereira de Castro Alves (a89563)

Leonardo de Freitas Marreiros (a89537)

Conteúdo

Parte 1	4
1. Exercício 1.....	4
1.1 Alínea a.....	4
1.2 Alínea b	5
1.3 Alínea c.....	5
1.4 Alínea d	5
2. Exercício 2.....	6
2.1 Alínea a.....	6
2.2 Alínea b	6
2.3 Alínea c.....	6
2.4 Alínea d	7
2.5 Alínea e.....	7
2.6 Alínea f	7
2.7 Alínea g.....	8
3. Exercício 3.....	8
3.1 Alínea a.....	9
3.2 Alínea b	9
3.3 Alínea c.....	10
3.4 Alínea d	10
3.5 Alínea e.....	11
Parte 2	12
1. Exercício 1.....	12
1.1 Alínea a.....	12
1.2 Alínea b	12
1.3 Alínea c.....	12
1.4 Alínea d	13
1.5 Alínea e.....	13
2. Exercício 2.....	14
2.1 Alínea a.....	14
2.2 Alínea b	15
2.3 Alínea c.....	16
2.4 Alínea d	16
2.5 Alínea e.....	17

3. Exercício 3	18
3.1 Alínea a	18
3.2 Alínea b	19
3.3 Alínea c	19
Conclusão	22

Parte 1

1. Exercício 1

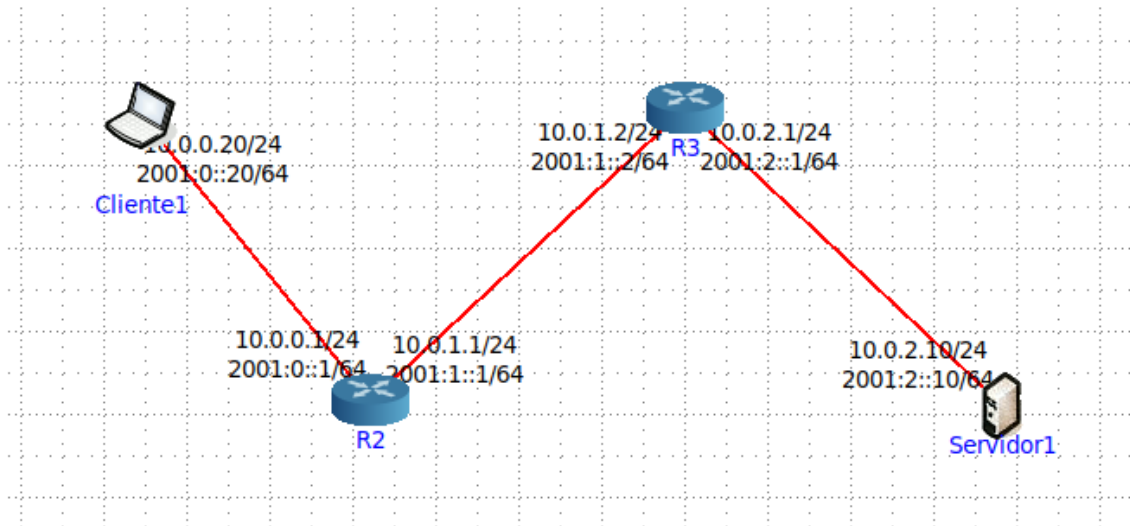


Figura 1- Topologia CORE

1.1 Alínea 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@n1:/tmp/pycore.45519/n1.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)  0.128 ms  0.014 ms  0.022 ms
 2 10.0.1.2 (10.0.1.2)  0.034 ms  0.015 ms  0.014 ms
 3 10.0.2.10 (10.0.2.10)  0.031 ms  0.019 ms  0.017 ms
```

Figura 2- Traceroute

1.2 Alínea b

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

42	172.165385021	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x0046, seq=1/256, ttl=1 (no response found!)
43	172.165488286	10.0.0.1	10.0.0.20	ICMP	102 Time-to-live exceeded	(Time to live exceeded in transit)
44	172.165561149	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x0046, seq=2/512, ttl=1 (no response found!)
45	172.165694449	10.0.0.1	10.0.0.20	ICMP	102 Time-to-live exceeded	(Time to live exceeded in transit)
46	172.165695615	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x0046, seq=3/768, ttl=1 (no response found!)
47	172.165695680	10.0.0.1	10.0.0.20	ICMP	102 Time-to-live exceeded	(Time to live exceeded in transit)
48	172.1656924589	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x0046, seq=4/1024, ttl=2 (no response found!)
49	172.165695992	10.0.1.2	10.0.0.20	ICMP	102 Time-to-live exceeded	(Time to live exceeded in transit)
50	172.165661598	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x0046, seq=5/1280, ttl=2 (no response found!)
51	172.165672824	10.0.1.2	10.0.0.20	ICMP	102 Time-to-live exceeded	(Time to live exceeded in transit)
52	172.165679099	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x0046, seq=6/1536, ttl=2 (no response found!)
53	172.165680945	10.0.1.2	10.0.0.20	ICMP	102 Time-to-live exceeded	(Time to live exceeded in transit)
54	172.165696570	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x0046, seq=7/1792, ttl=3 (reply in 55)
55	172.165724447	10.0.2.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x0046, seq=7/1792, ttl=62 (request in 54)
56	172.165733398	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x0046, seq=8/2048, ttl=3 (reply in 57)
57	172.165747913	10.0.2.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x0046, seq=8/2048, ttl=62 (request in 56)
58	172.165754207	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x0046, seq=9/2304, ttl=3 (reply in 59)
59	172.165767960	10.0.2.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x0046, seq=9/2304, ttl=62 (request in 58)
60	172.165775167	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x0046, seq=10/2560, ttl=4 (reply in 61)
61	172.165788959	10.0.2.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x0046, seq=10/2560, ttl=62 (request in 60)
62	172.165795345	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x0046, seq=11/2816, ttl=4 (reply in 63)
63	172.165810831	10.0.2.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x0046, seq=11/2816, ttl=62 (request in 62)
64	172.165817016	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x0046, seq=12/3072, ttl=4 (reply in 65)
65	172.165830438	10.0.2.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x0046, seq=12/3072, ttl=62 (request in 64)
66	172.165837214	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x0046, seq=13/3328, ttl=5 (reply in 67)
67	172.165850726	10.0.2.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x0046, seq=13/3328, ttl=62 (request in 66)
68	172.165856961	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x0046, seq=14/3584, ttl=5 (reply in 69)
69	172.165871285	10.0.2.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x0046, seq=14/3584, ttl=62 (request in 68)
70	172.165877711	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x0046, seq=15/3840, ttl=5 (reply in 71)
71	172.165891484	10.0.2.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x0046, seq=15/3840, ttl=62 (request in 70)
72	172.165898209	10.0.0.20	10.0.2.10	ICMP	74 Echo (ping) request	id=0x0046, seq=16/4096, ttl=6 (reply in 73)
73	172.165922056	10.0.2.10	10.0.0.20	ICMP	74 Echo (ping) reply	id=0x0046, seq=16/4096, ttl=62 (request in 72)

Figura 3- Wireshark

Em relação aos resultados obtidos é importante referir os seguintes aspetos: primeiramente, os pacotes enviados com TTL=1 foram rejeitados pelo router r2; da mesma forma, os pacotes enviados para o router r3 com TTL=2 foram descartados. Nesta fase, foi recebido um pacote “Time to live exceeded” como resposta. Finalmente, os pacotes enviados com TTL=3 chegaram ao seu destino, Servidor1, e foi obtida como resposta pacotes “Echo (ping) reply”.

1.3 Alínea c

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

Tal como foi referido na questão anterior, TTL=3.

1.4 Alínea d

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

$$\begin{aligned} \text{RTT} &= 2 * ((0.128 + 0.014 + 0.022) / 3 + (0.034 + 0.015 + 0.014) / 3 + (0.031 + 0.019 + 0.017) / 3) \text{ms} \\ &= 0.196 \text{ ms.} \end{aligned}$$

2. Exercício 2

```
> Ethernet II, Src: LiteonTe_d6:f7:57 (cc:b0:da:d6:f7:57), Dst: ComdaEnt_ff:94:00 (00:d0:03:ff:94:00)
✓ Internet Protocol Version 4, Src: 172.26.88.43, 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: 0xf5b7 (62903)
  > Flags: 0x0000
    Fragment offset: 0
    Time to live: 255
    Protocol: ICMP (1)
    Header checksum: 0xf64e [validation disabled]
    [Header checksum status: Unverified]
    Source: 172.26.88.43
    Destination: 193.136.9.240
  > Internet Control Message Protocol
```

Figura 4- Cabeçalho IP

2.1 Alínea a

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

172.26.88.43

2.2 Alínea b

Qual é o valor do campo protocolo? O que identifica?

ICMP (1). Identifica Internet Protocol.

2.3 Alínea 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 IP(v4) tem 20 bytes. O payload calcula-se subtraindo o total length do pacote pelo header length (56-20) que resulta um tamanho de 36 bytes.

2.4 Alínea d

O datagrama IP foi fragmentado? Justifique.

Ao analisarmos a figura 4 podemos verificar que o campo “Fragment offset” está com o valor 0. Além disso, a Flag “More fragments” estava definida como “Not Set” o que significa que não existem mais fragmentos. Logo, concluímos que o datagrama não foi fragmentado.

2.5 Alínea 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.

No.	Time	Source	Destination	Protocol	Length	Info
78	33.880154	172.26.88.43	162.159.136.232	TCP	55	55273 → 443 [ACK] Seq=1 Ack=1 Win=511 Len=1 [TCP segment of a reassembled PDU]
81	34.417310	172.26.88.43	162.159.133.234	TCP	54	51459 → 443 [ACK] Seq=55 Ack=1640 Win=511 Len=0
82	34.609817	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=770/515, ttl=255 (reply in 83)
84	34.660474	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=771/771, ttl=1 (no response found!)
86	34.711315	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=772/1827, ttl=2 (no response found!)
88	34.744349	172.26.88.43	162.159.129.232	TCP	55	55253 → 443 [ACK] Seq=1 Ack=1 Win=512 Len=1 [TCP segment of a reassembled PDU]
90	34.762041	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=773/1283, ttl=3 (no response found!)
92	34.812952	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=774/1539, ttl=4 (reply in 93)
95	36.171182	172.26.88.43	162.159.133.234	TCP	54	51459 → 443 [ACK] Seq=55 Ack=1797 Win=510 Len=0
96	37.189987	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=775/1795, ttl=255 (reply in 97)
98	37.169296	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=776/2851, ttl=1 (no response found!)
100	37.211114	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=777/2307, ttl=2 (no response found!)
103	37.260293	172.26.88.43	162.159.133.234	TCP	54	51459 → 443 [ACK] Seq=55 Ack=1958 Win=510 Len=0
104	37.261395	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=778/2563, ttl=3 (no response found!)
106	37.312151	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=779/2819, ttl=4 (reply in 107)
108	39.610577	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=780/3075, ttl=255 (reply in 109)
110	39.661286	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=781/3331, ttl=1 (no response found!)
112	39.712412	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=782/3587, ttl=2 (no response found!)
114	39.762982	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=783/3843, ttl=3 (no response found!)
116	39.813854	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=784/4099, ttl=4 (reply in 117)
119	40.948045	172.26.88.43	162.159.133.234	TCP	54	51459 → 443 [ACK] Seq=55 Ack=2129 Win=508 Len=0
120	42.118025	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=785/4355, ttl=255 (reply in 121)
122	42.161088	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=786/4611, ttl=1 (no response found!)
124	42.211834	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=787/4867, ttl=2 (no response found!)
126	42.262199	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=788/5123, ttl=3 (no response found!)
128	42.312922	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=789/5379, ttl=4 (reply in 129)
130	44.611814	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=790/5635, ttl=255 (reply in 131)
132	44.661121	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=791/5891, ttl=1 (no response found!)
134	44.711819	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=792/6147, ttl=2 (no response found!)
136	44.762592	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=793/6403, ttl=3 (no response found!)
138	44.813385	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=794/6659, ttl=4 (reply in 139)
140	47.112242	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=795/6915, ttl=255 (reply in 141)
142	47.162984	172.26.88.43	193.136.9.240	ICMP	70	Echo (ping) request id=0x0001, seq=796/7171, ttl=1 (no response found!)

Figura 5- Pacotes capturados no Wireshark ordenados pelo endereço IP fonte

Os campos do cabeçalho IP que variam de pacote para pacote são: TTL, Header checksum e Identification.

2.6 Alínea f

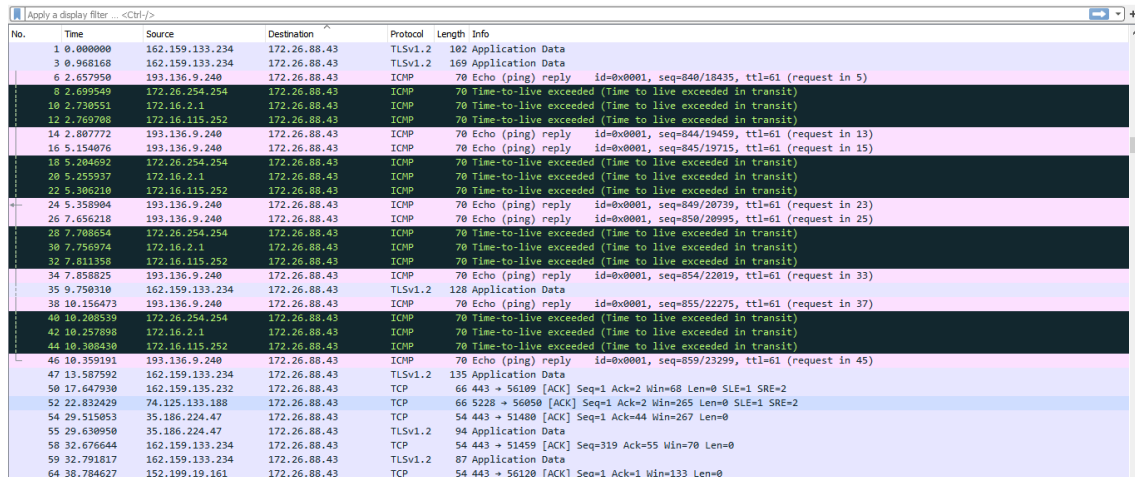
Observa algum padrão nos valores do campo de Identificação do datagrama IP e TTL?

Identificação do datagrama IP: os primeiros 8 bits são iguais e os restantes aumentam sequencialmente.

TTL: aumenta sequencialmente.

2.7 Alínea 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?



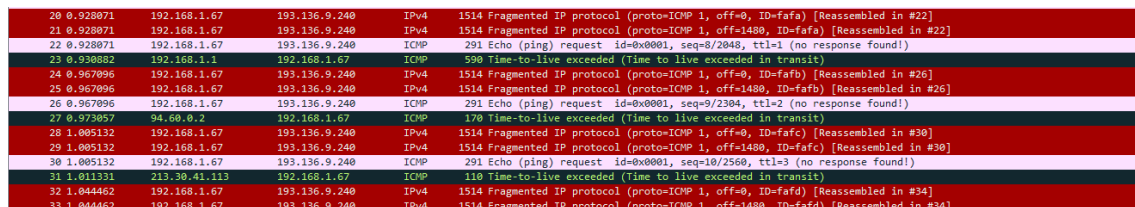
The image shows a Wireshark packet capture window with a display filter set to 'icmp'. The packets are sorted by destination address. The table below represents the data visible in the packet list pane.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	162.159.133.234	172.26.88.43	TLsv1.2	162	Application Data
3	0.968168	162.159.133.234	172.26.88.43	TLsv1.2	169	Application Data
6	2.657950	193.136.9.240	172.26.88.43	ICMP	70	Echo (ping) reply id=0x0001, seq=840/18435, ttl=61 (request in 5)
8	2.699549	172.26.254.254	172.26.88.43	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
10	2.730551	172.16.2.1	172.26.88.43	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
12	2.769708	172.16.115.252	172.26.88.43	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
14	2.807772	193.136.9.240	172.26.88.43	ICMP	70	Echo (ping) reply id=0x0001, seq=844/19459, ttl=61 (request in 13)
16	5.154076	193.136.9.240	172.26.88.43	ICMP	70	Echo (ping) reply id=0x0001, seq=845/19715, ttl=61 (request in 15)
18	5.204592	172.26.254.254	172.26.88.43	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
20	5.255937	172.16.2.1	172.26.88.43	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
22	5.286210	172.16.115.252	172.26.88.43	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
24	5.358094	193.136.9.240	172.26.88.43	ICMP	70	Echo (ping) reply id=0x0001, seq=849/20739, ttl=61 (request in 23)
26	7.656218	193.136.9.240	172.26.88.43	ICMP	70	Echo (ping) reply id=0x0001, seq=850/20995, ttl=61 (request in 25)
28	7.708654	172.26.254.254	172.26.88.43	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
30	7.756974	172.16.2.1	172.26.88.43	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
32	7.811558	172.16.115.252	172.26.88.43	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
34	7.858825	193.136.9.240	172.26.88.43	ICMP	70	Echo (ping) reply id=0x0001, seq=854/22019, ttl=61 (request in 33)
35	9.790310	162.159.133.234	172.26.88.43	TLsv1.2	128	Application Data
38	10.156473	193.136.9.240	172.26.88.43	ICMP	70	Echo (ping) reply id=0x0001, seq=855/22275, ttl=61 (request in 37)
40	10.208539	172.26.254.254	172.26.88.43	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
42	10.257898	172.16.2.1	172.26.88.43	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
44	10.308430	172.16.115.252	172.26.88.43	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
46	10.359191	193.136.9.240	172.26.88.43	ICMP	70	Echo (ping) reply id=0x0001, seq=859/23299, ttl=61 (request in 45)
47	13.587592	162.159.133.234	172.26.88.43	TLsv1.2	135	Application Data
50	17.647930	162.159.135.232	172.26.88.43	TCP	66	443 → 56109 [ACK] Seq=1 Ack=2 Win=68 Len=0 SLE=1 SRE=2
52	22.832429	74.125.133.188	172.26.88.43	TCP	66	5228 → 56050 [ACK] Seq=1 Ack=2 Win=265 Len=0 SLE=1 SRE=2
54	29.515053	35.186.224.47	172.26.88.43	TCP	54	443 → 51480 [ACK] Seq=1 Ack=44 Win=267 Len=0
55	29.630950	35.186.224.47	172.26.88.43	TLsv1.2	94	Application Data
58	32.676644	162.159.133.234	172.26.88.43	TCP	54	443 → 51459 [ACK] Seq=319 Ack=55 Win=70 Len=0
59	32.791817	162.159.133.234	172.26.88.43	TLsv1.2	87	Application Data
64	38.784627	152.199.19.161	172.26.88.43	TCP	54	443 → 56120 [ACK] Seq=1 Ack=1 Win=133 Len=0

Figura 6- Pacotes capturados no Wireshark ordenados pelo endereço destino

O valor do TTL permanece constante com um valor de 61 para todas as mensagens de resposta ICMP TTL exceeed. Este valor advém do facto do TTL predefinido pelo destino ser 64; quando o pacote chega ao seu destino, como passou por 3 routers intermédios,e sabendo que por cada salto diminui em uma unidade, o TTL foi decrementado 3 vezes: $64 - 3 = 61$.

3. Exercício 3



The image shows a Wireshark packet capture window with a display filter set to 'ip'. The packets are sorted by destination address. The table below represents the data visible in the packet list pane.

No.	Time	Source	Destination	Protocol	Length	Info
20	0.928071	192.168.1.67	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=fafa) [Reassembled in #22]
21	0.928071	192.168.1.67	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=fafa) [Reassembled in #22]
22	0.928071	192.168.1.67	193.136.9.240	ICMP	291	Echo (ping) request id=0x0001, seq=8/2045, ttl=1 (no response found!)
23	0.930882	192.168.1.1	192.168.1.67	ICMP	590	Time-to-live exceeded (Time to live exceeded in transit)
24	0.967096	192.168.1.67	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=fafb) [Reassembled in #26]
25	0.967096	192.168.1.67	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=fafb) [Reassembled in #26]
26	0.967096	192.168.1.67	193.136.9.240	ICMP	291	Echo (ping) request id=0x0001, seq=9/2304, ttl=2 (no response found!)
27	0.973057	94.60.0.2	192.168.1.67	ICMP	170	Time-to-live exceeded (Time to live exceeded in transit)
28	1.005132	192.168.1.67	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=fafc) [Reassembled in #30]
29	1.005132	192.168.1.67	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=fafc) [Reassembled in #30]
30	1.005132	192.168.1.67	193.136.9.240	ICMP	291	Echo (ping) request id=0x0001, seq=10/2560, ttl=3 (no response found!)
31	1.011331	213.30.41.113	192.168.1.67	ICMP	110	Time-to-live exceeded (Time to live exceeded in transit)
32	1.044462	192.168.1.67	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=fafd) [Reassembled in #34]
33	1.044462	192.168.1.67	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=fafd) [Reassembled in #34]

Figura 7- Fragmentos do datagrama IP

3.1 Alínea a

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

A primeira mensagem ICMP é a 20. Há necessidade de fragmentar os pacotes quando o seu tamanho ultrapassa o MTU (Maximum Transmission Unit) que pela análise dos dados dos pacotes capturados é 1500 bytes. Ora, como o pacote enviado foi de 3237 bytes então foi fragmentado em 3 pacotes mais pequenos.

3.2 Alínea 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?

```
> Frame 20: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits)
> Ethernet II, Src: LiteonTe_d6:f7:57 (cc:b0:da:d6:f7:57), Dst: HuaweiTe_d0:a5
▼ Internet Protocol Version 4, Src: 192.168.1.67, 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: 0xfafa (64250)
    ▼ Flags: 0x2000, 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: 0x0bc3 [validation disabled]
    [Header checksum status: Unverified]
    Source: 192.168.1.67
    Destination: 193.136.9.240
    [Reassembled IPv4 in frame: 22]
> Data (1480 bytes)
```

Figura 8- Cabeçalho do primeiro fragmento do datagrama IP segmentado

É possível verificar que o datagrama foi fragmentado pois a Flag “More Fragments” tem o valor 1. Trata-se do primeiro fragmento porque tem “Fragment offset “ 0. O tamanho deste datagrama IP é 1500 bytes, informação que se encontra no campo “Total Length”.


```

> Frame 22: 291 bytes on wire (2328 bits), 291 bytes captured (2328 b
> Ethernet II, Src: LiteonTe_d6:f7:57 (cc:b0:da:d6:f7:57), Dst: Huawe
✓ Internet Protocol Version 4, Src: 192.168.1.67, 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: 277
    Identification: 0xfafa (64250)
✓ Flags: 0x0172
    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: 0x2f18 [validation disabled]
    [Header checksum status: Unverified]
    Source: 192.168.1.67
    Destination: 193.136.9.240
> [3 IPv4 Fragments (3217 bytes): #20(1480), #21(1480), #22(257)]

```

Figura 10- - Cabeçalho do último fragmento do datagrama IP segmentado

3.5 Alínea 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.

Os campos que mudam no cabeçalho IP entre os fragmentos são: o “Fragment offset” (que indica onde informação vai ser agregada), a flag “More Fragments” e o “Header checksum”. A flag “More Fragments” em conjunto com o offset permitem reconstruir o pacote na medida em que se ordenam por ordem crescente de offset até que a flag “More Fragments” seja 0. O campo header checksum serve apenas para verificar a integridade dos fragmentos.

Parte 2

1. Exercício 1

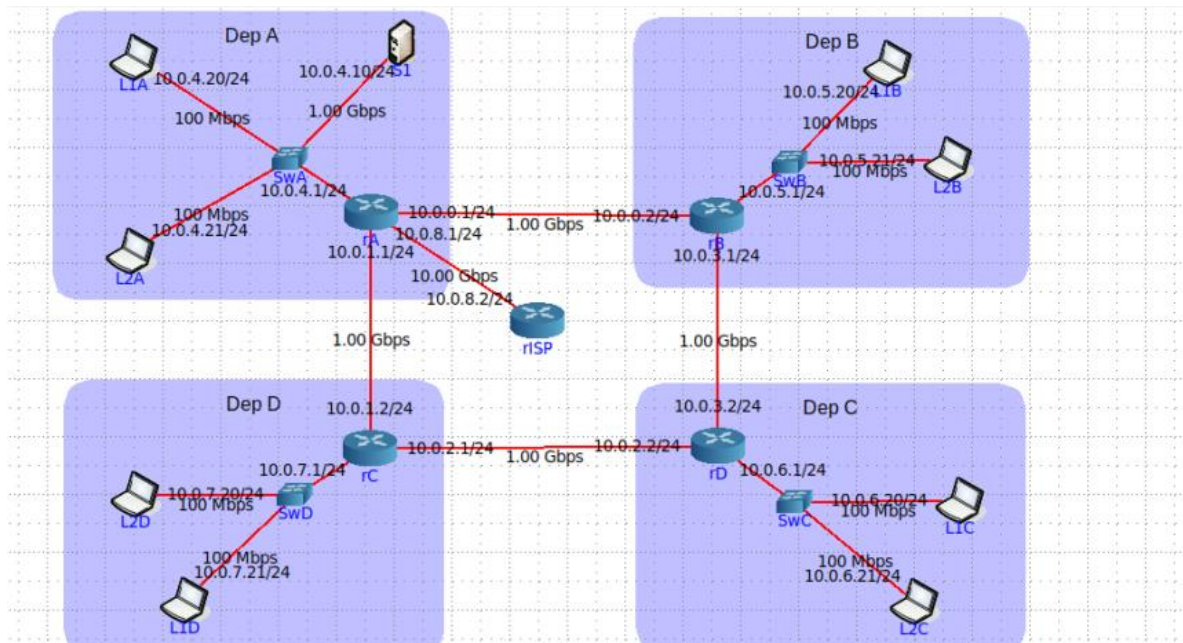


Figura 12 - Topologia CORE

1.1 Alínea 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.

Representado na figura 12.

1.2 Alínea b

Tratam-se de endereços públicos ou privados? Porquê?

Tratam-se de endereços privados pois os endereços de IP são valores entre 10.0.0.0 e 10.255.255.255 /8.

1.3 Alínea c

Porque razão não é atribuído um endereço IP aos switches?

Não lhe são atribuídos IP porque servem para interligar dispositivos e passar informação entre eles; Fazem parte da camada de ligação 2 pelo que não são visíveis na camada de ligação 3. O switch aprende com a rede e depois apenas encaminha para os endereços MAC conhecidos.

1.4 Alínea 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).

```
# ping -c3 10.0.4.20
PING 10.0.4.20 (10.0.4.20) 56(84) bytes of data.
64 bytes from 10.0.4.20: icmp_seq=1 ttl=64 time=0.040 ms
64 bytes from 10.0.4.20: icmp_seq=2 ttl=64 time=0.038 ms
64 bytes from 10.0.4.20: icmp_seq=3 ttl=64 time=0.083 ms

--- 10.0.4.20 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2049ms
rtt min/avg/max/ndev = 0.038/0.053/0.083/0.022 ms
# ping -c3 10.0.5.20
PING 10.0.5.20 (10.0.5.20) 56(84) bytes of data.
64 bytes from 10.0.5.20: icmp_seq=1 ttl=62 time=0.121 ms
64 bytes from 10.0.5.20: icmp_seq=2 ttl=62 time=0.061 ms
64 bytes from 10.0.5.20: icmp_seq=3 ttl=62 time=0.065 ms

--- 10.0.5.20 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2051ms
rtt min/avg/max/ndev = 0.061/0.082/0.121/0.028 ms
# ping -c3 10.0.6.20
PING 10.0.6.20 (10.0.6.20) 56(84) bytes of data.
64 bytes from 10.0.6.20: icmp_seq=1 ttl=61 time=0.118 ms
64 bytes from 10.0.6.20: icmp_seq=2 ttl=61 time=0.085 ms
64 bytes from 10.0.6.20: icmp_seq=3 ttl=61 time=0.075 ms

--- 10.0.6.20 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2054ms
rtt min/avg/max/ndev = 0.075/0.092/0.118/0.021 ms
# ping -c3 10.0.7.20
PING 10.0.7.20 (10.0.7.20) 56(84) bytes of data.
64 bytes from 10.0.7.20: icmp_seq=1 ttl=62 time=0.121 ms
64 bytes from 10.0.7.20: icmp_seq=2 ttl=62 time=0.054 ms
64 bytes from 10.0.7.20: icmp_seq=3 ttl=62 time=0.062 ms

--- 10.0.7.20 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2054ms
rtt min/avg/max/ndev = 0.054/0.079/0.121/0.029 ms
#
```

Figura 13- Conetividade entre o servidor e laptops de cada departamento

Como podemos observar na figura 13, existe conetividade IP entre os laptops dos vários departamentos e o servidor do departamento A. O comando ping teve resposta do servidor em cada um dos laptops.

1.5 Alínea e

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

```

vcmd
# ping -c3 10.0.4.10
PING 10.0.4.10 (10.0.4.10) 56(84) bytes of data.
64 bytes from 10.0.4.10: icmp_seq=1 ttl=63 time=0.063 ms
64 bytes from 10.0.4.10: icmp_seq=2 ttl=63 time=0.057 ms
64 bytes from 10.0.4.10: icmp_seq=3 ttl=63 time=0.053 ms

--- 10.0.4.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2036ms
rtt min/avg/max/mdev = 0.053/0.057/0.063/0.009 ms
# █

```

Figura 14- Ping para S1 a partir de RISP

Observando a figura 14, verifica-se que existe conectividade IP do router RISP para o servidor S1.

2. Exercício 2

2.1 Alínea 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`).

```

root@rC: /tmp/pycore.42805/rC.conf
root@rC: /tmp/pycore.42805/rC.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
10.0.0.0 10.0.3.1 255.255.255.0 UG 0 0 0 eth1
10.0.1.0 10.0.2.1 255.255.255.0 UG 0 0 0 eth0
10.0.2.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
10.0.3.0 0.0.0.0 255.255.255.0 U 0 0 0 eth1
10.0.4.0 10.0.2.1 255.255.255.0 UG 0 0 0 eth0
10.0.5.0 10.0.3.1 255.255.255.0 UG 0 0 0 eth1
10.0.6.0 0.0.0.0 255.255.255.0 U 0 0 0 eth2
10.0.7.0 10.0.2.1 255.255.255.0 UG 0 0 0 eth0
10.0.8.0 10.0.2.1 255.255.255.0 UG 0 0 0 eth0
root@rC: /tmp/pycore.42805/rC.conf# █

```

Figura 15- Tabela de encaminhamento do router do Departamento C

```

root@L1C: /tmp/pycore.42805/L1C.conf
root@L1C: /tmp/pycore.42805/L1C.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window 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
root@L1C: /tmp/pycore.42805/L1C.conf# █

```

Figura 16 - Tabela de endereçamento de um laptop do Departamento C

As tabelas das imagens 15 e 16 são interpretadas da seguinte forma: um datagrama tem destino “Destination”, e é entregue na interface de endereço “Gateway” saindo pela interface “Iface”. A “Genmask” representa o tipo da máscara.

Mudando o foco de análise para as primeiras linhas da tabela da figura 15, reparamos que diferem no Gateway. Quando os endereços estão ligados diretamente, o Gateway não precisa de estar definido. Por sua vez, se não estiverem ligados diretamente, é necessário saber qual é o próximo salto.

Finalmente, em relação ao campo Flags, estas adicionam informação adicional. No caso, a flag “U” - rota válida; e “G” - a rota é para um router de gateway, e não para uma rede ou host diretamente conectado.

2.2 Alínea 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).

```
root@C:/tmp/pycore.42805/rC.conf# ps -ax
PID TTY      STAT   TIME COMMAND
  1 ?        S      0:00 /usr/local/bin/vnoded -v -c /tmp/pycore.42805/rC -l /
 56 ?        Ss     0:00 /usr/sbin/zebra -d
 62 ?        Ss     0:00 /usr/sbin/ospf6d -d
 66 ?        Ss     0:00 /usr/sbin/ospfd -d
101 pts/4    Ss+    0:00 /bin/bash
110 pts/8    Ss     0:00 /bin/bash
119 pts/8    R+     0:00 ps -ax
root@C:/tmp/pycore.42805/rC.conf#
```

Figura 17- Comando ps -ax no Router C

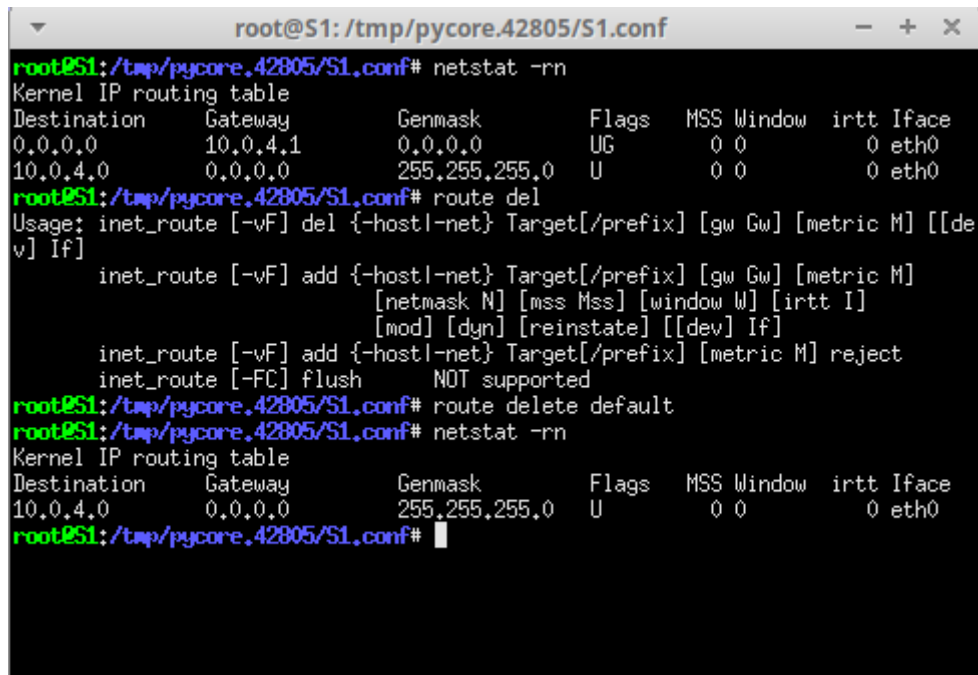
```
root@L1C:/tmp/pycore.42805/L1C.conf# ps -ax
PID TTY      STAT   TIME COMMAND
  1 ?        S      0:00 /usr/local/bin/vnoded -v -c /tmp/pycore.42805/L1C -l
 17 pts/2    Ss+    0:00 /bin/bash
 26 pts/6    Ss+    0:00 /bin/bash
 37 pts/10   Ss     0:00 /bin/bash
 45 pts/10   R+     0:00 ps -ax
root@L1C:/tmp/pycore.42805/L1C.conf#
```

Figura 18 - Comando ps -ax num laptop do Departamento C

Ao analisar a figura 17, vemos que o router usa o protocolo OSPF (Open Shortest Path First) logo é usado encaminhamento dinâmico. As rotas são atualizadas ao longo do tempo.

2.3 Alínea 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.42805/S1.conf
root@S1:/tmp/pycore.42805/S1.conf# netstat -rn
Kernel IP routing table
Destination      Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0          10.0.4.1        0.0.0.0         UG        0 0        0 eth0
10.0.4.0         0.0.0.0         255.255.255.0   U        0 0        0 eth0
root@S1:/tmp/pycore.42805/S1.conf# route del
Usage: inet_route [-vF] del {-host|-net} Target[/prefix] [gw Gw] [metric M] [[dev] If]
       inet_route [-vF] add {-host|-net} Target[/prefix] [gw Gw] [metric M]
                               [netmask N] [mss Mss] [window W] [irtt I]
                               [mod] [dyn] [reinststate] [[dev] If]
       inet_route [-vF] add {-host|-net} Target[/prefix] [metric M] reject
       inet_route [-FC] flush      NOT supported
root@S1:/tmp/pycore.42805/S1.conf# route delete default
root@S1:/tmp/pycore.42805/S1.conf# netstat -rn
Kernel IP routing table
Destination      Gateway         Genmask         Flags   MSS Window  irtt Iface
10.0.4.0         0.0.0.0         255.255.255.0   U        0 0        0 eth0
root@S1:/tmp/pycore.42805/S1.conf#
```

Figura 19 - Tabela de encaminhamento antes e depois da remoção da rota por defeito

Ao remover a rota por defeito é perdida a conectividade entre o servidor S1 e os restantes hosts que não pertencem à sua rede local. Isto acontece porque, ao eliminar a rota default, o servidor S1 passa a não saber para onde enviar de volta o que recebeu, isto é, não tem definida a rota de envio de tráfego para redes não locais. É possível enviar dados, no entanto, não é possível recebê-los de volta.

2.4 Alínea 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 registre os comandos que usou.

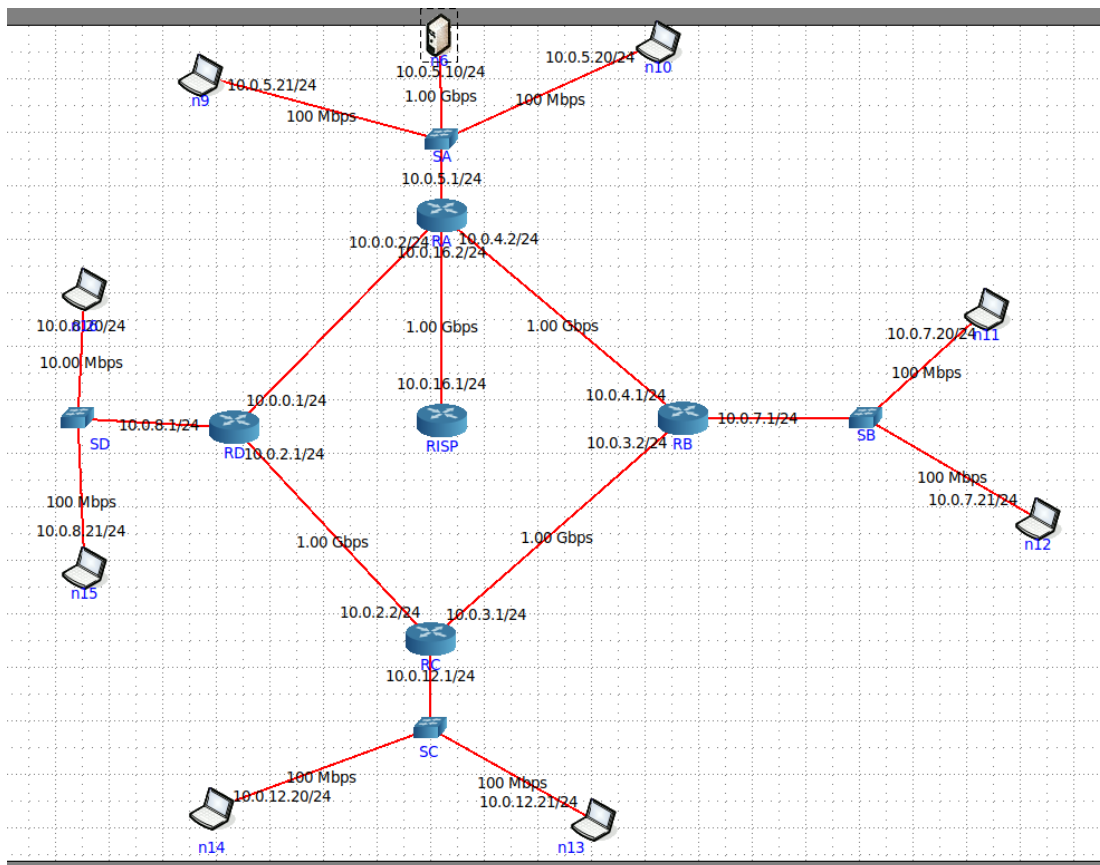


Figura 20 - Topologia CORE

Nota: Devido a um problema na máquina virtual, a topologia a considerar a partir deste ponto é a apresentada acima.

As rotas adicionadas, a partir de S1, foram:

- route add -net 10.0.8.0 netmask 255.255.255.0 gw 10.0.5.1
- route add -net 10.0.12.0 netmask 255.255.255.0 gw 10.0.5.1
- route add -net 10.0.7.0 netmask 255.255.255.0 gw 10.0.5.1

2.5 Alínea 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@n6: /tmp/pycore.42259/n6.conf# netstat -rn
Kernel IP routing table
Destination    Gateway         Genmask         Flags    MSS Window  irtt  Iface
10.0.5.0       0.0.0.0         255.255.255.0   U        0 0        0     eth0
10.0.7.0       10.0.5.1        255.255.255.0   UG       0 0        0     eth0
10.0.8.0       10.0.5.1        255.255.255.0   UG       0 0        0     eth0
10.0.12.0      10.0.5.1        255.255.255.0   UG       0 0        0     eth0
```

Figura 21- Tabela de endereçamento de S1 após serem repostas as rotas

Através da figura 21 verificamos que o servidor S1 está novamente acessível.

3. Exercício 3

3.1 Alínea a

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.

É dado o endereço 130.39.96.0/19 para o *subnetting* dos departamentos.

- Assim para quatro sub-redes e tendo em conta que todos os endereços da sub-rede são usáveis, são necessários 2 bits para diferenciar os quatro departamentos (130.39.011|00|000.0/21).

Desta maneira, os endereços para as sub-redes foram atribuídos da seguinte forma:

Subrede 1(Departamento A) : 130.39.011|00|000(96).0/21

Subrede 2(Departamento B) : 130.39.011|01|000(104).0/21

Subrede 3(Departamento C) : 130.39.011|10|000(112).0/21

Subrede 4(Departamento D) : 130.39.011|11|000(120).0/21

- Assumindo que os primeiros e últimos endereços estão reservados, cada departamento dispõe da respetiva gama de endereçamento:

Departamento A: 130.39.96.1/21 - 130.39.103.254/21

Departamento B: 130.39.104.1/21 - 130.39.111.254/21

Departamento C: 130.39.112.1/21 - 130.39.119.254/21

Departamento D: 130.39.120.1/21 - 130.39.127.254/21

- Tendo em mente a gama de cada departamento, foi escolhido o primeiro endereço de cada gama para a interface do router do respetivo departamento. Quanto aos hosts e ao servidor estes podem levar qualquer endereço na gama e, assim sendo, a cada host foi atribuído o endereço menor disponível e ao servidor foi atribuído o maior endereço do seu departamento (Departamento A) para fácil identificação.

A) Interface do router 130.39.96.1/21

HostA1.....130.39.96.2/21

HostA2.....130.39.96.3/21

Server.....130.39.96.254/21

B) Interface do router 130.39.104.1/21

HostB1.....130.39.104.2/21

HostB2.....130.39.104.3/21

C) Interface do router 130.39.112.1/21

HostC1.....130.39.112.2/21

HostC2.....130.39.112.3/21

D) Interface do router 130.39.120.1/21

HostD1.....130.39.120.2/21

HostD2.....130.39.120.3/21

3.2 Alínea b

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

A máscara usada foi 255.255.248.0.

A quantidade de endereços IP disponíveis para interligação em cada departamento, tendo em conta que sobram 11 bits para a identificação dos hosts e que o primeiro e último endereço são reservados, é $2^{11} - 2 = 2046$ endereços.

3.3 Alínea c

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

Alterar a tabela de routing IP de cada host de forma a:

- Remover as rotas antigas. (1)
- Ter o ip novo da respetiva interface do router do respetivo departamento como a rota default (0.0.0.0). (2)
- Ter a rota da subrede do respetivo departamento. (3)

Alterar a tabela de routing IP de cada router de forma a:

- Remover as rotas antigas para os departamentos. (4)
- Ter as novas rotas para os outros departamentos. (5)

(1) Every Host: route del default

Hosts A: route del -net 10.0.5.0/24

Hosts B: route del -net 10.0.7.0/24

Hosts C: route del -net 10.0.12.0/24

Hosts D: route del -net 10.0.8.0/24

(2) Hosts A: route add -net default gw 130.39.96.1

Hosts B: route add -net default gw 130.39.104.1

Hosts C: route add -net default gw 130.39.112.1

Hosts D: route add -net default gw 130.39.120.1

(3) Hosts A: route add -net 130.39.96.0/21 dev eth0

Hosts B: route add -net 130.39.104.0/21 dev eth0

Hosts C: route add -net 130.39.112.0/21 dev eth0

Hosts D: route add -net 130.39.120.0/21 dev eth0

(4) Every Router: route del -net 10.0.7.0/24

route del -net 10.0.12.0/24

route del -net 10.0.8.0/24

route del -net 10.0.5.0/24

(5) Router A: route add -net 130.39.96.0/21 dev eth2

route add -net 130.39.104.0/21 dev eth1

route add -net 130.39.112.0/21 dev eth1

route add -net 130.39.120.0/21 dev eth0

Router B: route add -net 130.39.96.0/21 dev eth1

route add -net 130.39.104.0/21 dev eth2

```
route add -net 130.39.112.0/21 dev eth0
```

```
route add -net 130.39.120.0/21 dev eth0
```

Router C:

```
route add -net 130.39.96.0/21 dev eth0
```

```
route add -net 130.39.104.0/21 dev eth1
```

```
route add -net 130.39.112.0/21 dev eth2
```

```
route add -net 130.39.120.0/21 dev eth0
```

Router D:

```
route add -net 130.39.96.0/21 dev eth0
```

```
route add -net 130.39.104.0/21 dev eth0
```

```
route add -net 130.39.112.0/21 dev eth1
```

```
route add -net 130.39.120.0/21 dev eth2
```

O resultado desta operação foi:

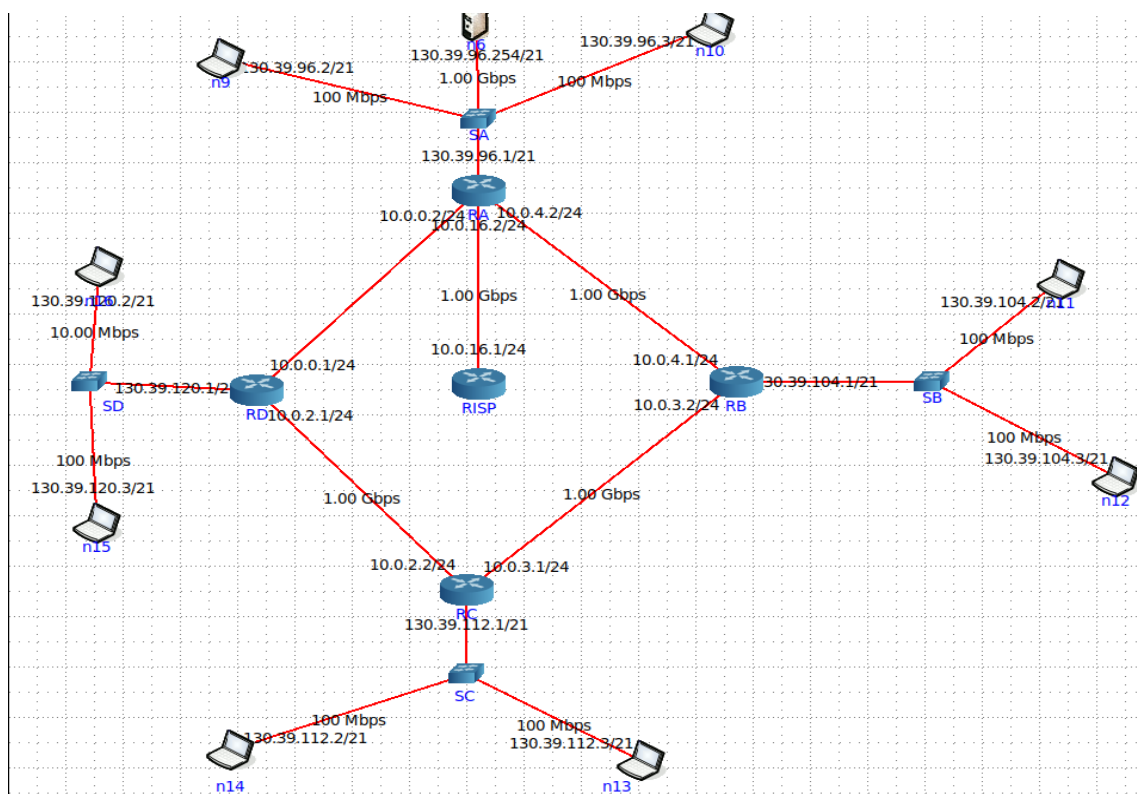


Figura 22-Nova Topologia CORE

Nota: Um dos alunos neste grupo estava inserido erroneamente no grupo 39 da Blackboard, que não corresponde ao nosso grupo. Apenas reparamos neste pormenor após o exercício estar concluído pelo que o valor de XX utilizado foi 39 em vez de 37 que seria o número correto.

Conclusão

Neste trabalho prático foi nos dada a oportunidade de consolidar a matéria que foi dada nas aulas teóricas ao longo destas semanas.

Para isso foram usadas duas ferramentas: Core, para simulação de redes, e Wireshark, para capturar tráfego. Com estas ferramentas desenvolvemos as nossas capacidades de construção de Topologias CORE. Foi bastante útil para a aprendizagem poder construir um caso virtual de tráfego ICMP.

Relativamente ao capítulo de Internet Protocol, ficamos a perceber melhor a funcionalidade do TTL uma vez que tivemos a oportunidade de analisar vários exemplos. Percebemos também a importância de tentar ter o TTL o menor possível, mas que consiga chegar ao destino desejado.

Relativamente ao formato de um datagrama IP, identificamos os dois campos constituintes no datagrama mencionado em cima: campo de dados e cabeçalho.

Vimos também a fragmentação de datagramas e, conseqüentemente, fomos confrontados com a importância do campo relativo à fragmentation, com as respetivas flags.

Relembramos ainda a definição de endereços públicos e privados e de switches.

Analisamos tabelas de encaminhamento e recordamos a definição de encaminhamento estático e dinâmico.

O conceito de subnetting foi também posto à prova com a necessidade da divisão de endereços e a importância das máscaras de rede foi também realçada.

Concluindo, a maior parte do capítulo de Protocolo IP foi abrangido e lembrado, e os conceitos inerentes ao mesmo foram consolidados.