Protocolo IPv4

Filipa Rebelo and Joana Oliveira

Universidade do Minho, Departamento de Informática, 4710-057 Braga, Portugal
email: {a90234,a87956}@alunos.uminho.pt

1 Parte I

1.1 Exercício 1

Prepare uma topologia CORE para verificar o comportamento do traceroute. Na topologia deve existir: um host (pc) cliente designado Bela cujo router de acesso é R2; o router R2 está simultaneamente ligado a dois routers R3 e R4; estes estão conectados a um router R5, que por sua vez, se liga a um host (servidor) designado Monstro. Ajuste o nome dos equipamentos atribuídos por defeito para o enunciado. Nas ligações (links) da rede de core estabeleça um tempo de propagação de 10ms. Após ativar a topologia, note que pode não existir conectividade IP imediata entre a Bela e o Monstro até que o anúncio de rotas entre routers estabilize.

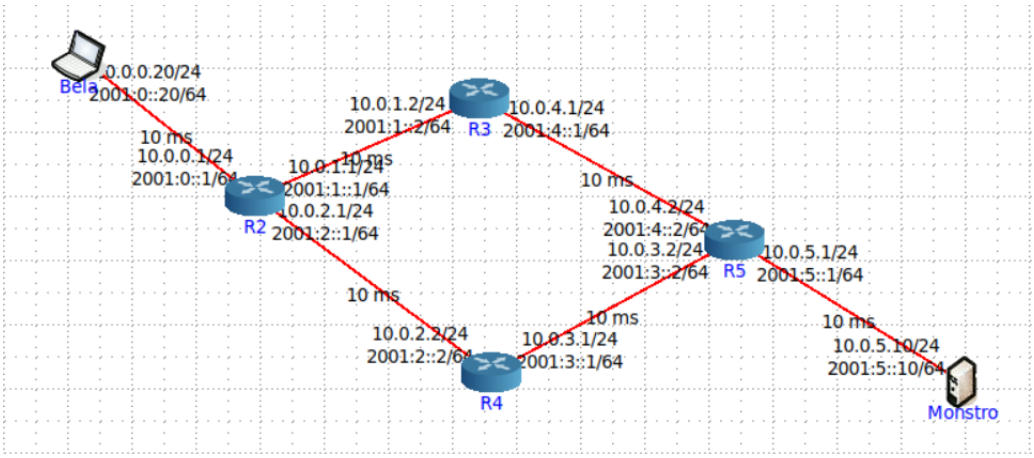


Fig. 1. Topologia Core

Alínea a) Active o wireshark ou o tcpdump no host Bela. Numa shell de Bela execute o comando traceroute -I para o endereço IP do Monstro.

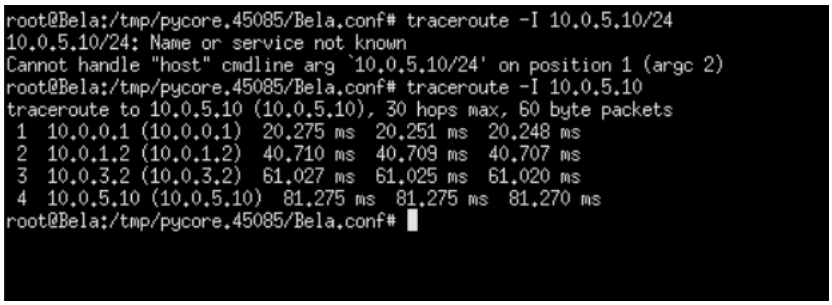
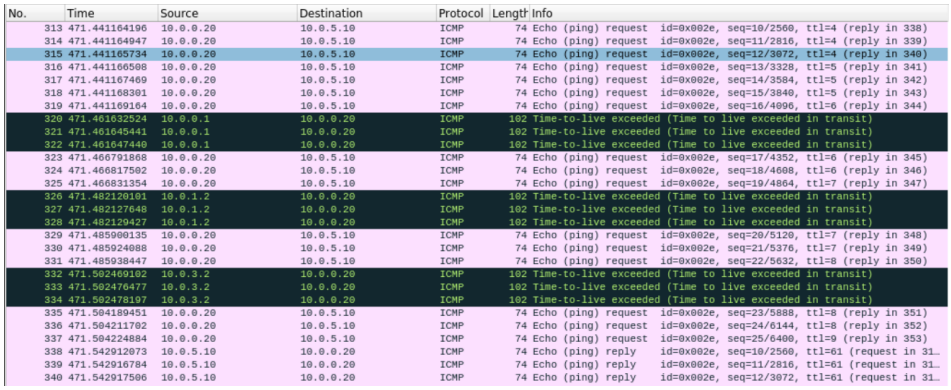


Fig. 2. Traceroute

Alínea b) Registe e analise o tráfego ICMP enviado pelo sistema Bela e o tráfego ICMP recebido como resposta. Comente os resultados face ao comportamento esperado.

Foi possível observar que o Bela envia diversos pacotes. Estes tinham inicialmente TTL = 1, tendo sido este valor incrementado, até se conseguir estabelecer uma conexão com sucesso.
Os pacotes com TTL = 6 foram descartados por R2, em seguida os pacotes com TTL = 7 foram descartados por R3 e com TTL = 8 foram descartados por R5. Para cada pacote descartado foi recebida a resposta *Time to live exceeded* proveniente do router que o descartou.
A partir de TTL = 9, os pacotes deixam de ser descartados, conseguindo finalmente chegar ao seu destino e passando assim a receber como resposta *Echo (ping) reply*.



No.	Time	Source	Destination	Protocol	Length	Info
313	471.441164196	10.0.0.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002e, seq=10/2560, ttl=4 (reply in 338)
314	471.441164947	10.0.0.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002e, seq=11/2816, ttl=4 (reply in 339)
315	471.441165734	10.0.0.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002e, seq=12/3072, ttl=4 (reply in 340)
316	471.441166508	10.0.0.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002e, seq=13/3328, ttl=5 (reply in 341)
317	471.441167469	10.0.0.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002e, seq=14/3584, ttl=5 (reply in 342)
318	471.441168301	10.0.0.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002e, seq=15/3840, ttl=5 (reply in 343)
319	471.441169164	10.0.0.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002e, seq=16/4096, ttl=6 (reply in 344)
320	471.461632524	10.0.0.1	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
321	471.461645441	10.0.0.1	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
322	471.461647440	10.0.0.1	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
323	471.466791868	10.0.0.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002e, seq=17/4352, ttl=6 (reply in 345)
324	471.466817502	10.0.0.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002e, seq=18/4608, ttl=6 (reply in 346)
325	471.466831354	10.0.0.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002e, seq=19/4864, ttl=7 (reply in 347)
326	471.482120101	10.0.1.2	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
327	471.482127648	10.0.1.2	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
328	471.482129427	10.0.1.2	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
329	471.485900135	10.0.0.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002e, seq=20/5120, ttl=7 (reply in 348)
330	471.485924088	10.0.0.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002e, seq=21/5376, ttl=7 (reply in 349)
331	471.485938447	10.0.0.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002e, seq=22/5632, ttl=8 (reply in 350)
332	471.502469102	10.0.3.2	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
333	471.502476477	10.0.3.2	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
334	471.502478197	10.0.3.2	10.0.0.20	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
335	471.504189451	10.0.0.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002e, seq=23/5888, ttl=8 (reply in 351)
336	471.504211702	10.0.0.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002e, seq=24/6144, ttl=8 (reply in 352)
337	471.504224884	10.0.0.20	10.0.5.10	ICMP	74	Echo (ping) request id=0x002e, seq=25/6400, ttl=9 (reply in 353)
338	471.542912073	10.0.5.10	10.0.0.20	ICMP	74	Echo (ping) reply id=0x002e, seq=10/2560, ttl=61 (request in 31…
339	471.542916784	10.0.5.10	10.0.0.20	ICMP	74	Echo (ping) reply id=0x002e, seq=11/2816, ttl=61 (request in 31…
340	471.542917506	10.0.5.10	10.0.0.20	ICMP	74	Echo (ping) reply id=0x002e, seq=12/3072, ttl=61 (request in 31…

Fig. 3. Wireshark

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

O valor mínimo de TTL deverá ser 9, tal como verificado na alínea anterior. É a partir deste valor que os pacotes deixam de ser descartados e que o cliente recebe uma resposta do servidor.

Alínea d) Calcule o valor médio do tempo de ida-e-volta (RTT - Round-Trip Time) obtido no acesso ao servidor. Para melhorar a média, poderá alterar o número pacotes de prova com a opção -q.

Tendo em conta os pacotes que conseguiram chegar ao destino e os valores da Figura 2:

$$\frac{81.275+81.275+81.270}{3} = 81,273ms$$

Alínea e) O valor médio do atraso num sentido (One-Way Delay) poderia ser calculado com precisão dividindo o RTT por dois? O que torna difícil o cálculo desta métrica?

Não, uma vez que o número de saltos a ser feito em cada sentido pode variar de acordo com o número de ligações existentes na rota escolhida. Além disso, mesmo que este número seja igual em ambos os sentidos, há ainda a possibilidade de existência de congestionamentos nas travessias que contribuem para o aumento deste atraso.

1.2 Exercício 2

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

Através da imagem abaixo verifica-se que o IP Source é 172.26.78.94.

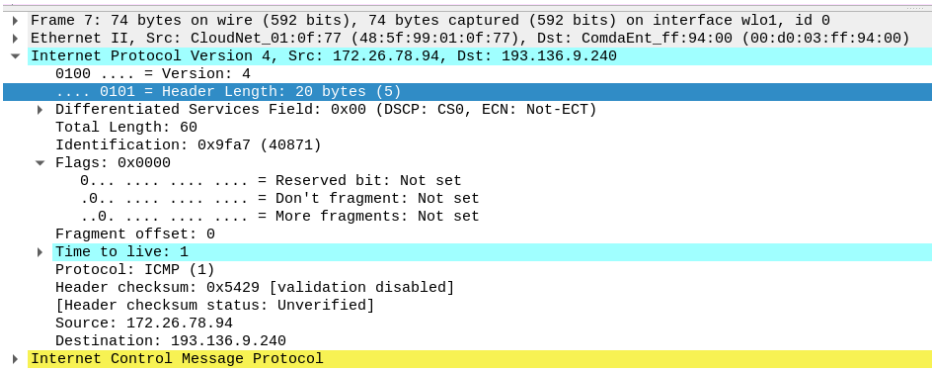


Fig. 4. Informação da primeira mensagem ICMP capturada

Alínea b) Qual é o valor do campo protocolo? O que permite identificar?

O valor do campo protocolo é 1, que identifica o ICMP, ou seja, o protocolo utilizado para enviar a mensagem.

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

Através da figura anterior e consultando o campo *Header Length* é possível perceber que o cabeçalho tem 20 bytes.
O cálculo do *payload* resulta da subtração do valor do *Header Length* ao valor que se encontra no *Total Length*. Assim sendo, o *payload* é então de 60-20 = 40 bytes.

Alínea d) O datagrama IP foi fragmentado? Justifique

Como se pode observar na Figura 4, o campo *Fragment offset* tem o valor 0, o que poderá significar que este datagrama não foi fragmentado ou então que este é o primeiro fragmento a ser enviado.
No entanto, e como o campo *More fragments* se encontra como *Not set*, é possível concluir que não existem mais fragmentos, pelo que o datagrama não foi fragmentado.

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.

Através da Figura 5 é possível verificar que os campos do cabeçalho IP que variam de pacote para pacote são o *Time to Live*, *Identification* e o *Header Checksum*.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	172.26.78.94	185.199.111.154	TCP	66	60308 → 443 [ACK] Seq=1 Ack=1 Win=501 Len=0 TSval=3445528525 …
3	0.131133033	172.26.78.94	193.137.16.145	DNS	86	Standard query 0xf6a3 A marco.uminho.pt OPT
4	0.131476048	172.26.78.94	193.137.16.145	DNS	86	Standard query 0xce3a AAAA marco.uminho.pt OPT
7	0.134571340	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=1/256, ttl=1 (no response…
8	0.134609522	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=2/512, ttl=1 (no response…
9	0.134635464	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=3/768, ttl=1 (no response…
10	0.134663917	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=4/1024, ttl=2 (no respons…
11	0.134687574	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=5/1280, ttl=2 (no respons…
12	0.134712216	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=6/1536, ttl=2 (no respons…
13	0.134739306	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=7/1792, ttl=3 (no respons…
14	0.134761729	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=8/2048, ttl=3 (no respons…
15	0.134787301	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=9/2304, ttl=3 (no respons…
16	0.134815737	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=10/2560, ttl=4 (reply in …
17	0.134838849	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=11/2816, ttl=4 (reply in …
18	0.134865837	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=12/3072, ttl=4 (reply in …
19	0.134897429	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=13/3328, ttl=5 (reply in …
20	0.134921292	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=14/3584, ttl=5 (reply in …
21	0.134949467	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=15/3840, ttl=5 (reply in …
22	0.134978557	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=16/4096, ttl=6 (reply in …
24	0.135929169	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=17/4352, ttl=6 (reply in …
27	0.137331653	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=18/4608, ttl=6 (reply in …
28	0.137383261	172.26.78.94	193.136.9.240	ICMP	74	Echo (ping) request id=0x0005, seq=19/4864, ttl=7 (reply in …
43	0.140794583	172.26.78.94	193.137.16.145	DNS	98	Standard query 0x9eaf PTR 254.254.26.172.in-addr.arpa OPT
47	0.143045005	172.26.78.94	193.137.16.145	DNS	87	Standard query 0x9eaf PTR 254.254.26.172.in-addr.arpa
49	0.147555100	172.26.78.94	193.137.16.145	DNS	94	Standard query 0xc0d3 PTR 1.2.16.172.in-addr.arpa OPT
51	0.150237136	172.26.78.94	193.137.16.75	DNS	94	Standard query 0xc0d3 PTR 1.2.16.172.in-addr.arpa OPT
53	0.153064561	172.26.78.94	193.137.16.65	DNS	94	Standard query 0xc0d3 PTR 1.2.16.172.in-addr.arpa OPT

Fig. 5. Tráfego ICMP

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

É possível observar que os valores contidos no campo relativo à identificação vão sendo sucessivamente incrementados a cada nova mensagem e que todos eles começam por "0x9f".
O campo TTL incrementa 1 valor por cada 3 mensagens, até que seja estabelecida ligação com o servidor.

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?

O valor do TTL varia entre 253 e 255. Isto deve-se ao funcionamento do *traceroute*, uma vez que cada pacote poderá ser rejeitado ao fim de um número de passos diferente, o que significa que cada mensagem de retorno irá também dar números de saltos diferentes.

No.	Time	Source	Destination	Protocol	Length	Info
261	20.223961145	172.26.78.94	140.82.114.26	TCP	66	34348 → 443 [ACK] Seq=1 Ack=1 Win=501 Len=0 TSval=1029230069 …
508	34.086245610	172.26.78.94	140.82.114.26	TLSv1.2	95	[TCP Previous segment not captured] , Application Data
2	0.015403920	185.199.111.154	172.26.78.94	TCP	66	[TCP ACKed unseen segment] 443 → 60308 [ACK] Seq=1 Ack=2 Win=…
5	0.133797716	193.137.16.145	172.26.78.94	DNS	140	Standard query response 0xce3a AAAA marco.uminho.pt SOA dns.u…
6	0.134026039	193.137.16.145	172.26.78.94	DNS	102	Standard query response 0xf6a3 A marco.uminho.pt A 193.136.9.…
23	0.135807769	172.16.2.1	172.26.78.94	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
25	0.137204234	172.16.2.1	172.26.78.94	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
26	0.137204482	172.16.2.1	172.26.78.94	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
29	0.139500505	193.136.9.240	172.26.78.94	ICMP	74	Echo (ping) reply id=0x0005, seq=10/2560, ttl=61 (request …
30	0.139500831	193.136.9.240	172.26.78.94	ICMP	74	Echo (ping) reply id=0x0005, seq=11/2816, ttl=61 (request …
31	0.139500930	193.136.9.240	172.26.78.94	ICMP	74	Echo (ping) reply id=0x0005, seq=12/3072, ttl=61 (request …
32	0.139501024	193.136.9.240	172.26.78.94	ICMP	74	Echo (ping) reply id=0x0005, seq=13/3328, ttl=61 (request …
33	0.139501123	193.136.9.240	172.26.78.94	ICMP	74	Echo (ping) reply id=0x0005, seq=14/3584, ttl=61 (request …
34	0.139578388	193.136.9.240	172.26.78.94	ICMP	74	Echo (ping) reply id=0x0005, seq=15/3840, ttl=61 (request …
35	0.139578517	193.136.9.240	172.26.78.94	ICMP	74	Echo (ping) reply id=0x0005, seq=16/4096, ttl=61 (request …
36	0.139578607	172.16.115.252	172.26.78.94	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
37	0.139578702	172.16.115.252	172.26.78.94	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
38	0.139578805	172.16.115.252	172.26.78.94	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
39	0.139846080	193.136.9.240	172.26.78.94	ICMP	74	Echo (ping) reply id=0x0005, seq=17/4352, ttl=61 (request …
40	0.139846273	172.26.254.254	172.26.78.94	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
41	0.139846373	172.26.254.254	172.26.78.94	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
42	0.139886637	172.26.254.254	172.26.78.94	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
44	0.141974992	193.136.9.240	172.26.78.94	ICMP	74	Echo (ping) reply id=0x0005, seq=18/4608, ttl=61 (request …
45	0.142464477	193.136.9.240	172.26.78.94	ICMP	74	Echo (ping) reply id=0x0005, seq=19/4864, ttl=61 (request …
46	0.142786006	193.137.16.145	172.26.78.94	DNS	152	Standard query response 0x9eaf No such name PTR 254.254.26.17…
48	0.145283267	193.137.16.145	172.26.78.94	DNS	141	Standard query response 0x9eaf No such name PTR 254.254.26.17…
50	0.149904998	193.137.16.145	172.26.78.94	DNS	94	Standard query response 0xc0d3 Refused PTR 1.2.16.172.in-addr…
52	0.152793202	193.137.16.75	172.26.78.94	DNS	94	Standard query response 0xc0d3 Refused PTR 1.2.16.172.in-addr…

Fig. 6. Tráfego ICMP

1.3 Exercício 3

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

Foi necessário fragmentar o pacote incial uma vez que este possuía 4037 bytes e apenas é possível transmitir 1500 bytes de cada vez.

No.	Time	Source	Destination	Protocol	Length	Info
263	24.127851185	172.26.78.94	193.137.16.145	DNS	86	Standard query 0x39d0 A marco.uminho.pt OPT
264	24.128244886	172.26.78.94	193.137.16.145	DNS	86	Standard query 0x3e69 AAAA marco.uminho.pt OPT
265	24.131356735	193.137.16.145	172.26.78.94	DNS	102	Standard query response 0x39d0 A marco.uminho.pt A 193.136.9.…
266	24.131357373	193.137.16.145	172.26.78.94	DNS	140	Standard query response 0x3e69 AAAA marco.uminho.pt SOA dns.u…
267	24.132506504	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=a5d7) [Reasse…
268	24.132559855	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=a5d7) [Rea…
269	24.132571061	172.26.78.94	193.136.9.240	ICMP	1091	Echo (ping) request id=0x0006, seq=1/256, ttl=1 (no response…
270	24.132601154	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=a5d8) [Reasse…
271	24.132612055	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=a5d8) [Rea…
272	24.132645059	172.26.78.94	193.136.9.240	ICMP	1091	Echo (ping) request id=0x0006, seq=2/512, ttl=1 (no response…
273	24.132667087	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=a5d9) [Reasse…
274	24.132676722	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=a5d9) [Rea…
275	24.132703654	172.26.78.94	193.136.9.240	ICMP	1091	Echo (ping) request id=0x0006, seq=3/768, ttl=1 (no response…
276	24.132743263	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=a5da) [Reasse…
277	24.132760061	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=a5da) [Rea…
278	24.132770148	172.26.78.94	193.136.9.240	ICMP	1091	Echo (ping) request id=0x0006, seq=4/1024, ttl=2 (no respons…
279	24.132796082	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=a5db) [Reasse…
280	24.132811624	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=a5db) [Rea…
281	24.132834557	172.26.78.94	193.136.9.240	ICMP	1091	Echo (ping) request id=0x0006, seq=5/1280, ttl=2 (no respons…
282	24.132870882	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=a5dc) [Reasse…
283	24.132891679	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=a5dc) [Rea…
284	24.132900931	172.26.78.94	193.136.9.240	ICMP	1091	Echo (ping) request id=0x0006, seq=6/1536, ttl=2 (no respons…
285	24.132925819	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=a5dd) [Reasse…
286	24.132940687	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=a5dd) [Rea…
287	24.132963761	172.26.78.94	193.136.9.240	ICMP	1091	Echo (ping) request id=0x0006, seq=7/1792, ttl=3 (no respons…
288	24.132999651	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=a5de) [Reasse…
289	24.133020834	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=a5de) [Rea…
290	24.133030235	172.26.78.94	193.136.9.240	ICMP	1091	Echo (ping) request id=0x0006, seq=8/2048, ttl=3 (no respons…
291	24.133052046	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=a5df) [Reasse…
292	24.133071127	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=a5df) [Rea…
293	24.133093987	172.26.78.94	193.136.9.240	ICMP	1091	Echo (ping) request id=0x0006, seq=9/2304, ttl=3 (no respons…
294	24.133134647	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=a5e0) [Reasse…
295	24.133151369	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=a5e0) [Rea…
296	24.133160952	172.26.78.94	193.136.9.240	ICMP	1091	Echo (ping) request id=0x0006, seq=10/2560, ttl=4 (reply in …
297	24.133183818	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=a5e1) [Reasse…
298	24.133199016	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=a5e1) [Rea…
299	24.133221777	172.26.78.94	193.136.9.240	ICMP	1091	Echo (ping) request id=0x0006, seq=11/2816, ttl=4 (reply in …
300	24.133258433	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=a5e2) [Reasse…
301	24.133279208	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=a5e2) [Rea…
302	24.133290433	172.26.78.94	193.136.9.240	ICMP	1091	Echo (ping) request id=0x0006, seq=12/3072, ttl=4 (reply in …
303	24.133315966	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=a5e3) [Reasse…
304	24.133331077	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=1480, ID=a5e3) [Rea…
305	24.133354260	172.26.78.94	193.136.9.240	ICMP	1091	Echo (ping) request id=0x0006, seq=13/3328, ttl=5 (reply in …
306	24.133389901	172.26.78.94	193.136.9.240	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, off=0, ID=a5e4) [Reasse…

Fig. 7. Fragmentação do pacote inicial

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?

Através da observação da Figura 8 verifica-se que o valor da flag *More Fragments* tem o seu bit a 1, o que indica que existirão mais fragmentos associados ao datagrama original.
O valor do campo *Fragment offset* encontra-se a 0. Isto, juntamente com a informação anteriormente observada, permite concluir que este se trata do primeiro fragmento do pacote.
O tamanho deste datagrama encontra-se no campo *Total Length*, sendo este de 1500 bytes.

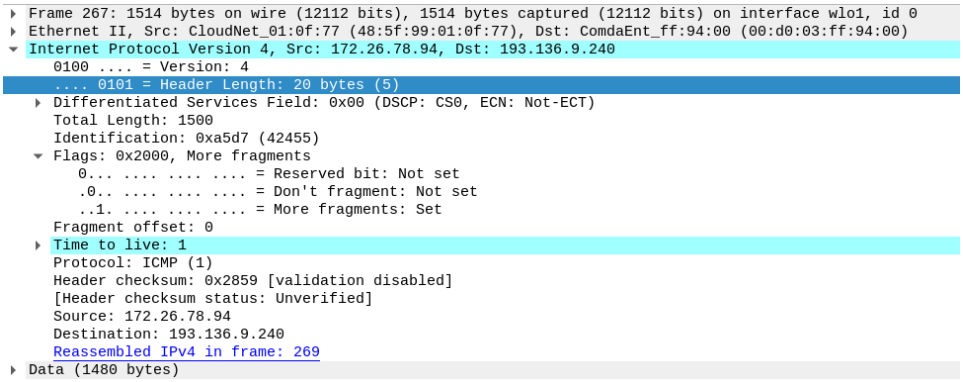


Fig. 8. Primeiro segmento do datagrama IP

Alínea 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?

Através da observação da Figura 9 verificamos que o campo *Fragment offset* tem valor 1488. Isto permite-nos concluir que se não se trata do primeiro fragmento uma vez que, nesse caso, o seu valor seria de 0.
Dado que a flag *More fragments* tem o seu bit a 1, é possível concluir que existem mais fragmentos do datagrama original.

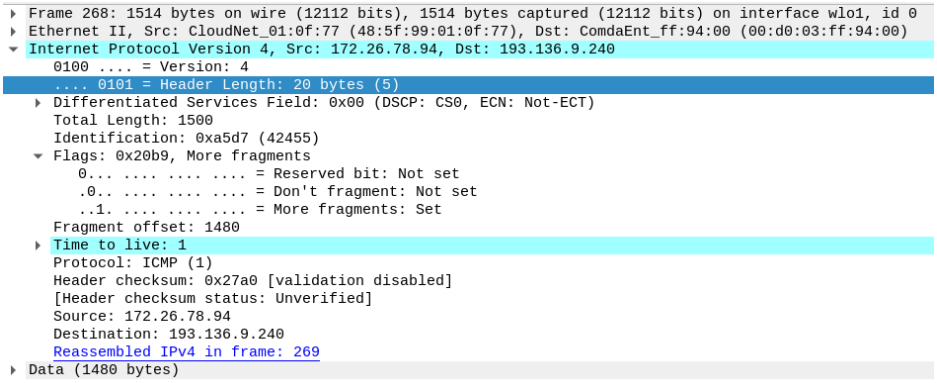


Fig. 9. Segundo segmento do datagrama IP

Alínea d) Quantos fragmentos foram criados a partir do datagrama original?

Foram criados 3 fragmentos a partir do datagrama inicial, tal como é possível observar na parte final da Figura 10. O último fragmento deverá conter a flag *More fragments* como *Not Set*, bem como a sua *Identification* igual aos restantes fragments, o que se pode observar. Sendo este o 3ºfragmento e uma vez que cumpre com os requisitos anteriores é possível concluir que, de facto, foram criados 3 fragmentos.

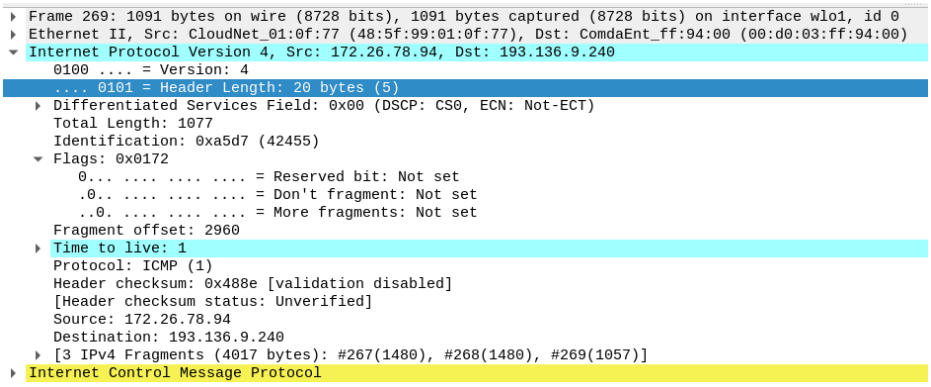


Fig. 10. Último segmento do datagrama IP

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 do cabeçalho IP que mudam entre os diferentes fragmentos do mesmo datagrama IP são a flag *More fragments* e o campo *Fragment offset*. Um bit a 0 na flag *More Fragments* permite-nos saber que nos encontramos no último fragmento e que, caso contrário, nos encontramos em qualquer um dos restantes. O campo *Fragment offset* terá o seu valor a 0 no caso de nos encontrarmos no primeiro fragmento e o valor correspondente ao número de bytes já enviados, caso nos encontremos em qualquer outro. Ordenando os fragmentos por ordem crescente de *offset* é possível obter o datagrama ordenado pela ordem em que os dados se encontravam no datagrama original.

Alínea f) Verifique o processo de fragmentação através de um processo de cálculo

Sendo o tamanho total do datagrama igual a 4037 e o tamanho máximo de cada fragmento igual a 1500, sendo que 20 são referentes ao cabeçalho, teríamos:

$$\frac{4037}{1480} = 2,73$$

No entanto, e como é necessário considerar que cada fragmento terá o seu cabeçalho, iriam ser necessários, no mínimo 3 fragmentos, uma vez que:

$$1480 * 2 = 2960$$

Ou seja, com 2 fragmentos apenas conseguimos enviar 2960 bytes, pelo que nos faltam ainda 1077 bytes. Como cada fragmento permite enviar 1480 bytes de informação, então são suficientes 3 fragmentos.

Alínea g) Escreva uma expressão lógica que permita detetar o último fragmento correspondente ao datagrama original.

O último fragmento será aquele cujo campo *Fragment offset* se encontre diferente de 0, o que significa que já foram enviados anteriormente outros fragmentos, e a flag *More Fragments* se encontre com o bit a 0, indicando que não existem mais fragmentos a ser enviados. Além disso, o campo Identification deverá ser igual aos restantes fragmentos do datagrama.

Expressão Lógica: *More Fragments* == 0 && *Fragment offset* >0 && $\forall 0 \leq i<n(Identification[i] == Identification[n])$

2 Parte II

2.1 Exercício 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.

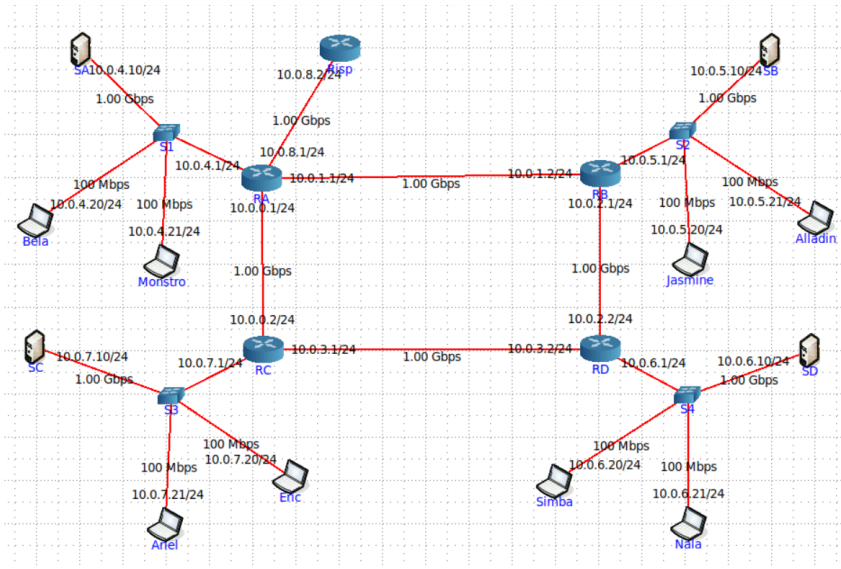


Fig. 11. Topologia Core

Alínea b) Tratam-se de endereços públicos ou privados? Porquê?

Tratam-se de endereços privados, uma vez que a gama 10.0.0.0-10.255.255.255/8 faz parte das gamas de endereços privados.

Alínea c) Porque razão não é atribuído um endereço IP aos switches?

Os switches operam uma camada abaixo (camada 2), relativa à ligação de dados, em que são utilizados os endereços MAC, e não os endereços IP, de cada equipamento para fazer a distribuição de pacotes.

Alínea d) Usando o comando ping certifique-se que existe conectividade IP interna a cada departamento (e.g. entre um laptop e o servidor respetivo.)

De forma a verificar a existência de conectividade interna entre cada portátil e o servidor respetivo foi então utilizado o comando *ping* em cada um deles. A imagem abaixo permite confirmar que foi possível estabelecer ligação com cada um dos servidores respetivos.

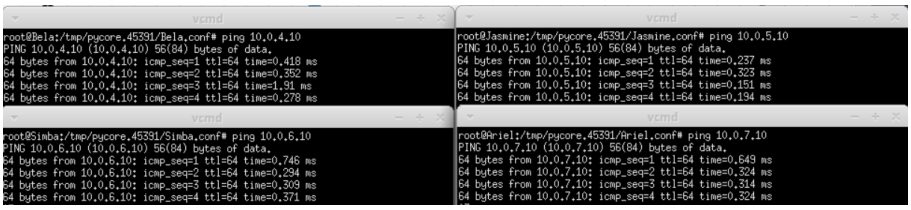


Fig. 12. Verificação de conectividade interna em todos os departamentos

Alínea e) Execute o número mínimo de comandos ping que lhe permite verificar a existência de conetividade IP entre departamentos.

De modo a utilizar o menor número possível de comandos *ping* para verificar a conectividade entre departamentos, apenas foram executados 3 comandos a partir do departamento A, 1 para cada departamento. Daqui é possível concluir que existe conectividade entre A e todos os outros departamentos. De modo a verificar que os restantes departamentos estão também conectados entre si, é necessário analisar a rede. Por exemplo, se quisermos confirmar que existe conectividade de B para D, uma vez que sabemos que existe conectividade de A para B, C e D, sabemos também que de modo a A estabelecer ligação com D teve que percorrer A-B-D ou então A-C-D. Qualquer que tenha sido a travessia é sempre possível estabelecer conexão entre B e D, quer seja diretamente de B para D ou através de B-A-D-C. O mesmo raciocínio pode ser aplicado aos restantes departamentos.

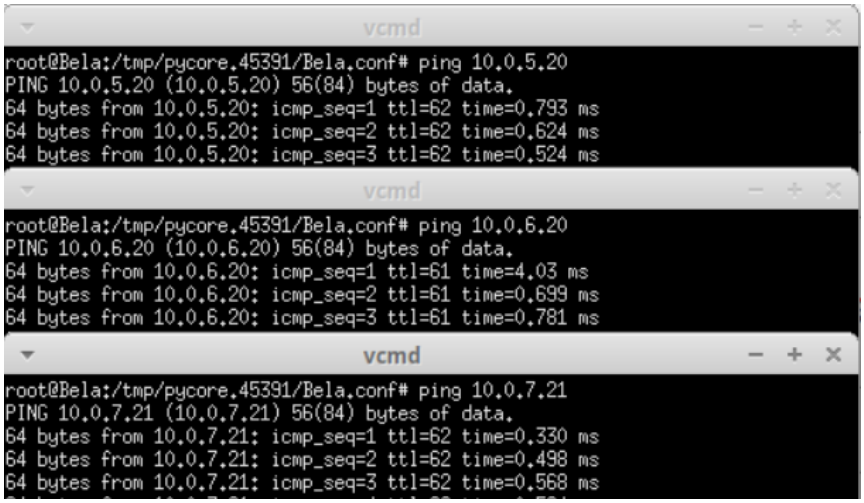


Fig. 13. Verificação de conectividade IP entre departamentos

Alínea f) Verifique se existe conectividade IP do portátil Bela para o router de acesso RISP.

Tal como na alínea d) foi utilizado o comando *ping* para verificar a existência de conectividade entre o portátil Bela e o router RISP. A imagem seguinte permite confirmar a sua existência.

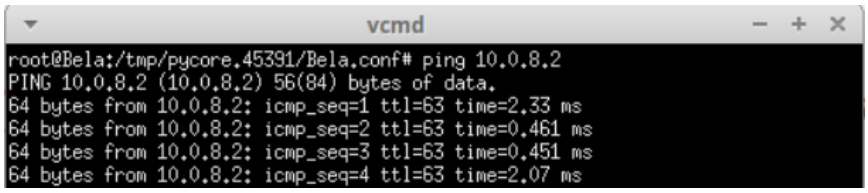


Fig. 14. Verificação de conectividade entre o portátil Bela e o router RISP

2.2 Exercício 2

Para o router RA e o portátil Bela:

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).

Na imagem seguinte é possível observar as tabelas de encaminhamento relativas ao router RA e ao portátil Bela. O campo *Destination* indica o endereço IP da rede de destino, o campo *Gateway* representa o próximo salto a ser dado pelo datagrama para chegar ao seu destino, o campo *Genmask* indica a máscara utilizada, as *Flags* permitem saber se as rotas são válidas (U) e poderão ser utilizadas ou se necessitarão ainda de um salto intermédio, ou seja, um *gateway* (G) e, por fim, o campo *Iface* contém o identificador da interface de saída da máquina local.

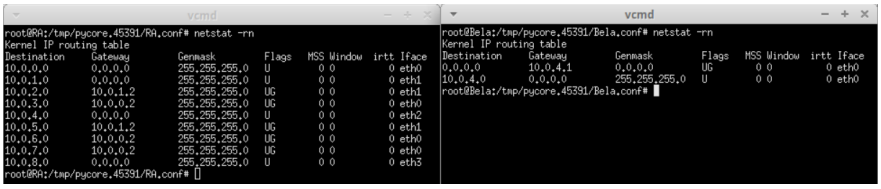


Fig. 15. Tabelas de encaminhamento do router RA e do portátil Bela

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 ou equivalente).

Na figura seguinte é possível observar que está a correr um processo *ospfd*, que se trata de um protocolo de encaminhamento dinâmico.

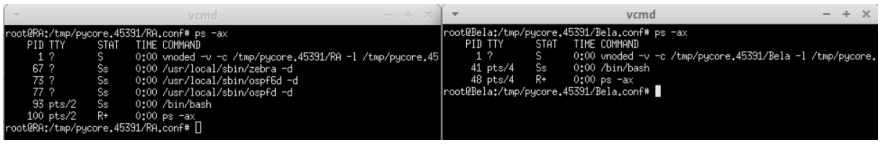


Fig. 16. Processos em execução no router RA e no portátil Bela

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 SA. Use o comando route delete para o efeito. Que implicações tem esta medida para os utilizadores da LEI-RC que acedem ao servidor. Justifique.

Ao retirar definitivamente a rota por defeito da tabela de encaminhamento de S1, este deixa de conseguir enviar pacotes para fora da sua sub-rede, uma vez que não estão definidas rotas para além dela. Assim sendo não é possível a conexão com outros departamentos.
Tal como é possível observar na Figura 18, o servidor SA tenta estabeler conexão com um portátil de outro departamento, mas sem sucesso.

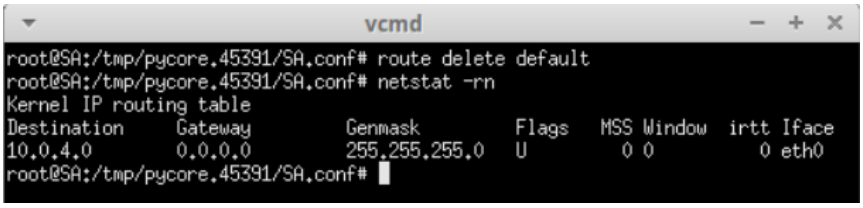


Fig. 17. Remoção da rota *default* do servidor SA

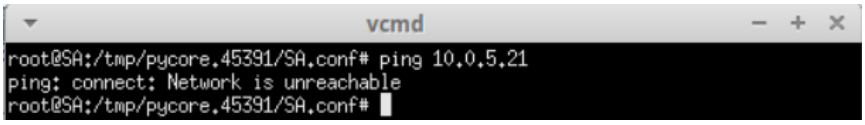


Fig. 18. Conectividade do servidor SA para o portátil Alladin

Alínea d) Não volte a repor a rota por defeito. Adicione todas as rotas estáticas necessárias para restaurar a conectividade para o servidor SA, por forma a contornar a restrição imposta na alínea c). Utilize para o efeito o comando route add e registe os comandos que usou

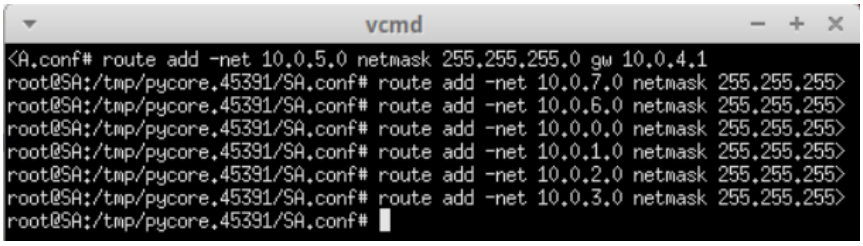


Fig. 19. Adição das rotas estáticas

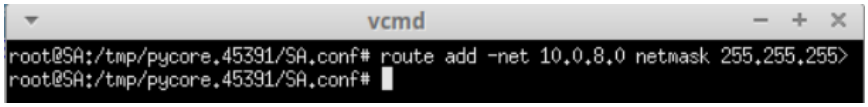


Fig. 20. Adição das rotas estáticas

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.

Através da Figura 22 é possível perceber que o servidor já consegue novamente estabelecer conexões para fora da sua sub-rede. Foi utilizado o comando *ping* para um portátil de cada departamento e observamos que não houve perda de pacotes, uma vez que todos eles são enviados e são recebidas as respetivas respostas.

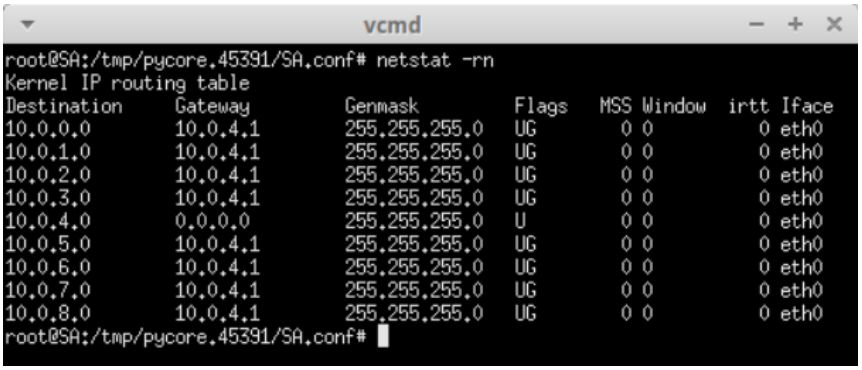


Fig. 21. Tabela de encaminhamento de SA

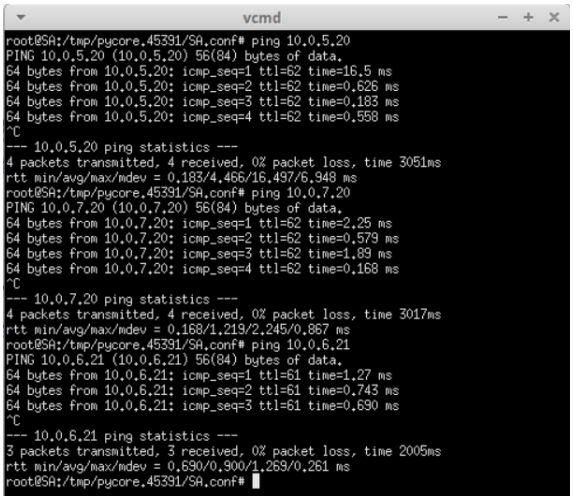


Fig. 22. Teste de conectividade entre SA e um portátil de cada departamento

2.3 Exercício 3

Considere a topologia definida anteriormente. Assuma que o endereçamento entre os routers (rede de backbone) se mantém inalterado, contudo, o endereçamento em cada departamento deve ser redefinido.

Alínea 1) Considere que dispõe apenas do endereço de rede IP 192.168.XXX.128/25, em que XXX é o decimal correspondendo ao seu número de grupo (PLXX). Defina um novo esquema de endereçamento para as redes dos departamentos (mantendo as redes de acesso externo e backbone inalteradas), sabendo que o número de departamentos pode vir a aumentar no curto prazo. Atribua endereços às interfaces dos vários sistemas envolvidos. Assuma que todos os endereços de subredes são usáveis. Justifique as opções tomadas no planeamento.

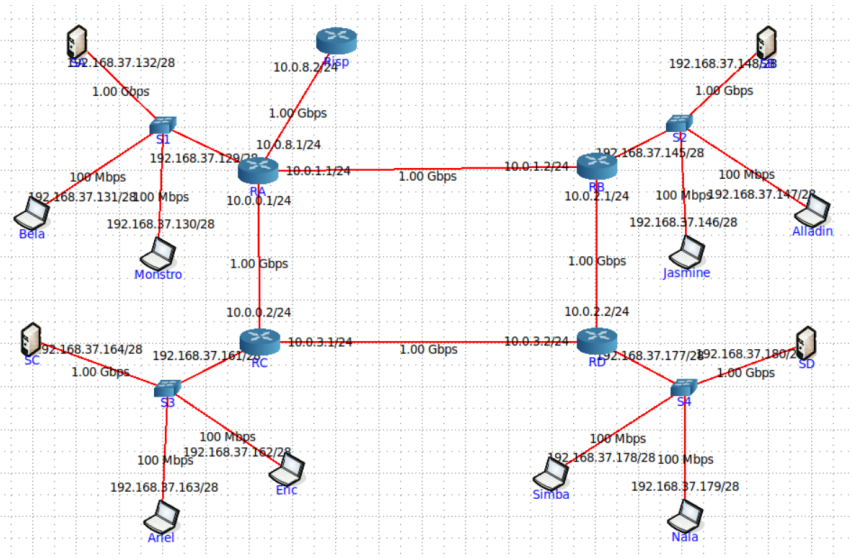


Fig. 23. Topologia Core

Uma vez que o número do nosso grupo é o 37, o endereço de rede IP que nos foi atríbuido é o 192.168.37.128/25, cuja representação em binário é 11000000.10101000.00100101.10000000.
Dada a existência de 4 departamentos, são então necessários, pelo menos, 2 bits. No entanto, e como é necessário considerar possíveis expansões futuras em termos de departamentos, 2 bits não chegam pois apenas permitem $2^2 = 4$ combinações.

Assim sendo iremos então reservar 3 bits para a identificação de sub-redes, ficando com uma nova máscara de 28 bits e 4 bits de sobra para hosts.
As sub-redes atríbuidas a cada departamento foram então:
Departamento A: 192.168.37.128/28
Departamento B: 192.168.37.144/28
Departamento C: 192.168.37.160/28
Departamento D: 192.168.37.176/28

Alínea 2) Qual a máscara de rede que usou (em formato decimal)? Quantos hosts IP pode interligar em cada departamento? Quantos prefixos de sub-rede ficam disponíveis para uso futuro? Justifique.

A máscara de rede utilizada foi a /28 (notação CIDR), correspondente a 255.255.255.240.
Tal como dito anteriormente, restam 4 bits para a identificação de hosts, o que significa que teremos $2^4 - 2 = 14$ possíveis hosts, uma vez que é necessário remover os endereços correspondentes ao endereço de broadcast e ao endereço da rede.
Uma vez que existem disponíveis $2^3 = 8$ prefixos de sub-rede e já estão a ser utilizados 4 deles, o número de prefixos de sub-rede disponíveis corresponde a 4 (8-4 = 4).

Alínea 3) Verifique e garanta que a conectividade IP interna na rede local LEI-RC é mantida. No caso de não existência de conetividade, reveja a atribuição de endereços efetuada e eventuais erros de encaminhamento por forma a realizar as correções necessárias. Explique como procedeu.

Para verificar a conectividade IP entre as várias redes utilizamos o router RA, pertencente ao departamento A. Através do comando *ping* tentamos então estabelecer conexão com os portáteis Monstro, Jasmine, Simba e Ariel, que se encontram todos em sub-redes distintas.
A figura seguinte permite confirmar a existência de conectividade entre o departamento A e os restantes departamentos. Para confirmar a conectividade entre os restantes departamentos (por exemplo, entre B e C), pode ser aplicada mesma lógica que foi utilizada na alínea e) do 1º exercício relativo a esta parte.

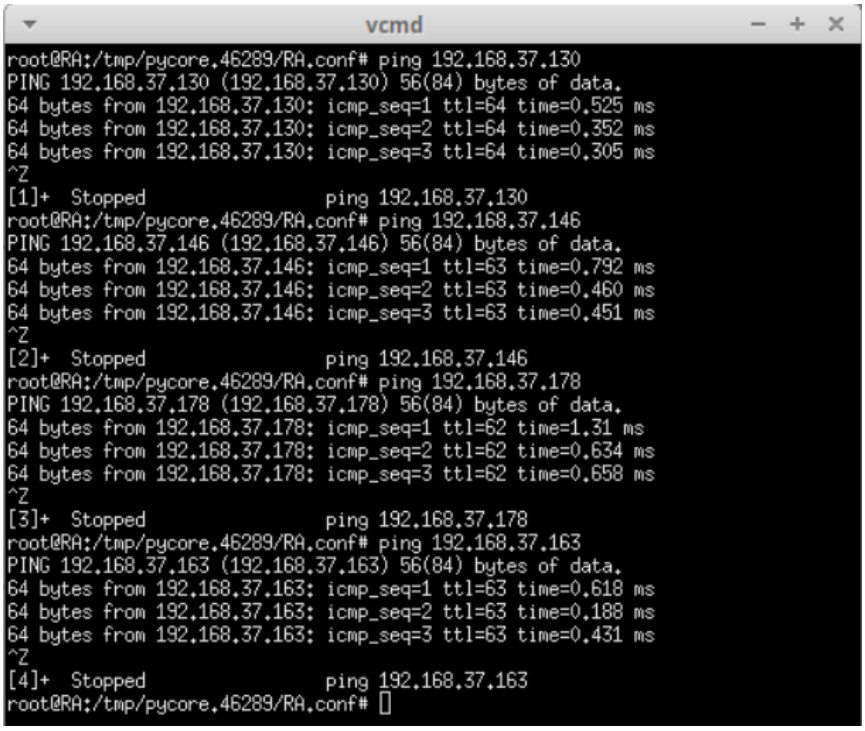


Fig. 24. Verificação de conetividade

3 Conclusão

A realização deste trabalho permitiu-nos consolidar os conceitos abordados nas aulas teóricas desta UC, mais especificamente sobre **Datagramas IP e Fragmentação**, **Endereçamento e Encaminhamento** e ainda, **Subnetting**.
O trabalho encontrava-se divido em duas partes, sendo que em ambas as partes foram utilizadas as ferramentas de gestão e monotorização de redes, *Core* e *Wireshark*.
Na primeira parte começamos por definir uma topologia de rede simples, recorrendo ao *Core* e focamo-nos principalmente no envio e rececão de pacotes, através da análise do tráfego ICMP, bem como na constituição dos datagramas IP e em como funciona a sua fragmentação.
Na segunda parte foi definida uma nova topologia mais complexa, constituída por diferentes sub-redes.
Esta parte centrou-se no redirecionamento de pacotes IP, através da manipulação de tabelas de encaminhamento dos vários routers e da inserção e remoção de rotas. Centrou-se também nos tipos de encaminhamento estático e dinâmico, bem como nos endereços públicos e privados.
Ainda nesta parte foi também abordado o conceito de *subnetting*, nomeadamente as máscaras de redes, os endereços de sub-rede e as respetivas gamas.
Por fim, acreditamos ter cumprido o objetivo deste trabalho e obtido conhecimento importante na área das redes de computadores.