

Redes de Computadores

2019/2020

Mestrado Integrado em Engenharia Informática

TP2: Protocolo IPv4

Grupo 06/PL06



Mafalda Costa (A83919)



Maria Moutinho (A83840)



Paulo Lima (A89983)

13 de novembro de 2019

Parte 1

Exercício 1

Prepare uma topologia no CORE para verificar o comportamento do traceroute. Ligue um host (servidor) s1 a um router r2; o router r2 a um router r3, o router r3 a um router r4, que por sua vez, se liga a um host (pc) h5. (Note que pode não existir conectividade IP imediata entre s1 e h5 até que o routing estabilize). Ajuste o nome dos equipamentos atribuídos por defeito para a topologia do enunciado.

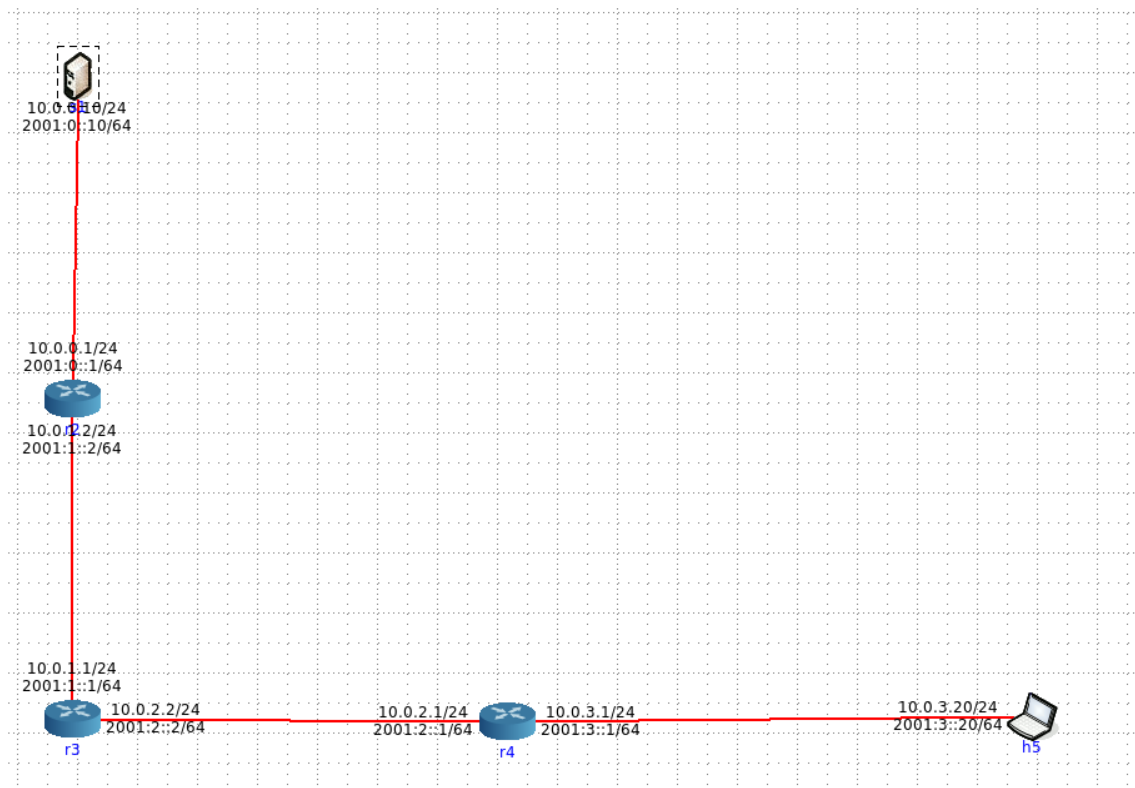


Figura 1.1: Topologia Core

- 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.

```
root@s1:/tmp/pycore.36141/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 gateway (10.0.0.1)  0.145 ms  0.032 ms  0.025 ms
 2 10.0.1.1 (10.0.1.1)  0.055 ms  0.024 ms  0.022 ms
 3 10.0.2.1 (10.0.2.1)  0.048 ms  0.031 ms  0.029 ms
 4 10.0.3.20 (10.0.3.20)  0.070 ms  0.034 ms  0.032 ms
root@s1:/tmp/pycore.36141/s1.conf#
```

Figura 1.2

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

119	426	254337084	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request	id=0x0023, seq=1/256, ttl=1 (no response found!)
120	426	254395667	10.0.0.1	10.0.0.10	ICMP	102 Time-to-live exceeded	(Time to live exceeded in transit)
121	426	254424242	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request	id=0x0023, seq=2/512, ttl=1 (no response found!)
122	426	254443642	10.0.0.1	10.0.0.10	ICMP	102 Time-to-live exceeded	(Time to live exceeded in transit)
123	426	254459155	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request	id=0x0023, seq=3/768, ttl=1 (no response found!)
124	426	254475695	10.0.0.1	10.0.0.10	ICMP	102 Time-to-live exceeded	(Time to live exceeded in transit)
125	426	254492408	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request	id=0x0023, seq=4/1024, ttl=2 (no response found!)
126	426	254509080	10.0.0.1	10.0.0.10	ICMP	102 Time-to-live exceeded	(Time to live exceeded in transit)
127	426	254524802	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request	id=0x0023, seq=5/1280, ttl=2 (no response found!)
128	426	254539230	10.0.0.1	10.0.0.10	ICMP	102 Time-to-live exceeded	(Time to live exceeded in transit)
129	426	254559271	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request	id=0x0023, seq=6/1536, ttl=2 (no response found!)
130	426	254574226	10.0.0.1	10.0.0.10	ICMP	102 Time-to-live exceeded	(Time to live exceeded in transit)
131	426	254592366	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request	id=0x0023, seq=7/1792, ttl=3 (no response found!)
132	426	254710874	10.0.0.1	10.0.0.10	ICMP	102 Time-to-live exceeded	(Time to live exceeded in transit)
133	426	254727664	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request	id=0x0023, seq=8/2048, ttl=3 (no response found!)
134	426	254766896	10.0.0.1	10.0.0.10	ICMP	102 Time-to-live exceeded	(Time to live exceeded in transit)
135	426	254782239	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request	id=0x0023, seq=9/2304, ttl=3 (no response found!)
136	426	254819183	10.0.0.1	10.0.0.10	ICMP	102 Time-to-live exceeded	(Time to live exceeded in transit)
137	426	254837088	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request	id=0x0023, seq=10/2560, ttl=4 (reply in 138)
138	426	254901491	10.0.0.3.20	10.0.0.10	ICMP	74 Echo (ping) reply	id=0x0023, seq=10/2560, ttl=61 (request in 137)
139	426	254920511	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request	id=0x0023, seq=11/2816, ttl=4 (reply in 140)

Figura 1.3: Tráfego ICMP

Inicialmente, o servidor vai tentar comunicar com o host, sem sucesso porque o TTL inicial vai ser 1. O pacote vai chegar a r2 e é desprezado, enviando uma mensagem ao servidor, com essa informação. Posteriormente, o TTL é sucessivamente aumentado até 4, repetindo o processo anterior. Este TTL vai ser o necessário para que os dados cheguem ao host (h5).

- 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.

137	426	254837088	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request	id=0x0023, seq=10/2560, ttl=4 (reply in 138)
138	426	254901491	10.0.0.3.20	10.0.0.10	ICMP	74 Echo (ping) reply	id=0x0023, seq=10/2560, ttl=61 (request in 137)
139	426	254920511	10.0.0.10	10.0.3.20	ICMP	74 Echo (ping) request	id=0x0023, seq=11/2816, ttl=4 (reply in 140)

Figura 1.4

O TTL inicial deverá ser 4, sendo que é o momento, em que o primeiro pacote é transmitido com sucesso a h5.

Exercício 2

Pretende-se agora usar o traceroute na sua máquina nativa, e gerar de datagramas IP de diferentes tamanhos.

- a. Qual é o endereço IP da interface ativado seu computador?

192.168.100.220

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

ICMP

- 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?

20 | 1246 | cabeçalho ICMP + payload ICMP

- d. O datagrama IP foi fragmentado? Justifique.

Não, pois o fragment offset é maior que 0

- 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.

(IP) identificar os frames | (ICMP) sequence number (BE) e (LE)

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

IP aumenta 1 em cada request | TTL aumenta 1 em cada 3 requests

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?

TTL=64, e não permanece constante.

Exercício 3

Pretende-se agora analisar a fragmentação de pacotes IP. Reponha a ordem do tráfego capturado usando a coluna do tempo de captura. Observe o tráfego depois do tamanho de pacote ter sido definido para 4206 bytes. (PL6)

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

O protocolo IPV4 apenas pode ter 1500 bytes, dos quais 20 para cabeçalho logo, sobram 1480, como o valor do payload é superior necessitamos de fragmentar, neste caso 3 vezes.

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?

The screenshot shows the Wireshark interface with two main panes. The left pane displays the 'Packet Details' for the selected packet (Packet 409). The right pane displays the 'Packet List' showing multiple ICMP packets.

Packet Details (Left Pane):

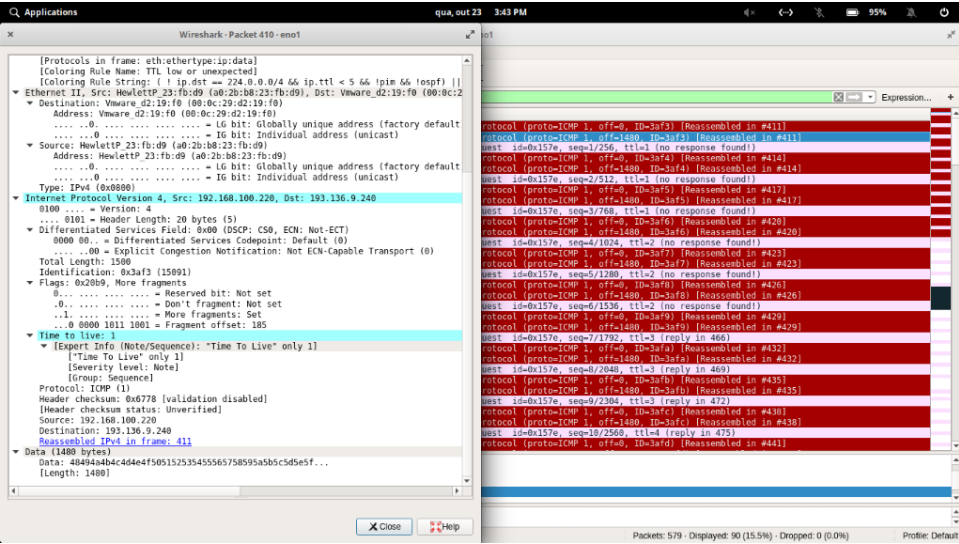
- Protocols in frame: eth:ethertype:ip:data
- Coloring Rule Name: TTL low or unexpected
- Filter: ip.dst == 224.0.0.4 && ip.ttl < 5 && (pin && !ospf) ||
- Ethernet II, Src: HewlettP_23:fb:d9 (a0:2b:b8:23:fb:d9), Dst: Vmware_d2:19:f0 (00:0c:29:02:19:f0)
- Address: Vmware_d2:19:f0 (00:0c:29:02:19:f0)
- Destination: Vmware_d2:19:f0 (00:0c:29:02:19:f0)
- Type: IPv4 (0x0800)
- Internet Protocol Version 4, Src: 192.168.100.228, Dst: 193.136.9.240
- 8100 ... = Version: 4
- 8101 ... = Header Length: 20 bytes (5)
- Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
- 0000 00 ... = Differentiated Services Codepoint: Default (0)
- 0000 00 ... = Explicit Congestion Notification: Not ECN-Capable Transport (0)
- Total Length: 1500
- Identification: 0x3af3 (15891)
- Flags: 0x2000, More fragments
- 0 ... = Reserved bit: Not set
- 0 ... = Don't fragment: Not set
- 0 ... = More fragments: Set
- 0 0000 0000 0000 ... = Fragment Offset: 0
- Time to live: 1
- [Expert Info (Note/Sequence): "Time To Live" only 1]
- [Time To Live] only 1
- [Severity Level: Note]
- [Group: Sequence]
- Protocol: ICMP (1)
- Header checksum: 0x0831 [validation disabled]
- Header checksum status: Unverified
- Source: 192.168.100.228
- Destination: 193.136.9.240
- Reassembled IPv4 in frame: 411
- Data (1480 bytes)
- Data: 0800209e157e00148494a4b4c444e4f505125354555657...
- [Length: 1480]

Packet List (Right Pane):

- Protocol (proto=ICMP), offset 0, ID=3af3 [Reassembled in #411]
- Protocol (proto=ICMP), offset 1480, ID=3af3 [Reassembled in #411]
- Protocol (proto=ICMP), offset 0, ID=3af4 [Reassembled in #414]
- Protocol (proto=ICMP), offset 1480, ID=3af4 [Reassembled in #414]
- Protocol (proto=ICMP), offset 0, ID=3af5 [Reassembled in #417]
- Protocol (proto=ICMP), offset 1480, ID=3af5 [Reassembled in #417]
- Protocol (proto=ICMP), offset 0, ID=3af6 [Reassembled in #420]
- Protocol (proto=ICMP), offset 1480, ID=3af6 [Reassembled in #420]
- Protocol (proto=ICMP), offset 0, ID=3af7 [Reassembled in #423]
- Protocol (proto=ICMP), offset 1480, ID=3af7 [Reassembled in #423]
- Protocol (proto=ICMP), offset 0, ID=3af8 [Reassembled in #426]
- Protocol (proto=ICMP), offset 1480, ID=3af8 [Reassembled in #426]
- Protocol (proto=ICMP), offset 0, ID=3af9 [Reassembled in #429]
- Protocol (proto=ICMP), offset 1480, ID=3af9 [Reassembled in #429]
- Protocol (proto=ICMP), offset 0, ID=3afa [Reassembled in #432]
- Protocol (proto=ICMP), offset 1480, ID=3afa [Reassembled in #432]
- Protocol (proto=ICMP), offset 0, ID=3afb [Reassembled in #435]
- Protocol (proto=ICMP), offset 1480, ID=3afb [Reassembled in #435]
- Protocol (proto=ICMP), offset 0, ID=3afc [Reassembled in #438]
- Protocol (proto=ICMP), offset 1480, ID=3afc [Reassembled in #438]
- Protocol (proto=ICMP), offset 0, ID=3afd [Reassembled in #441]
- Protocol (proto=ICMP), offset 1480, ID=3afd [Reassembled in #441]

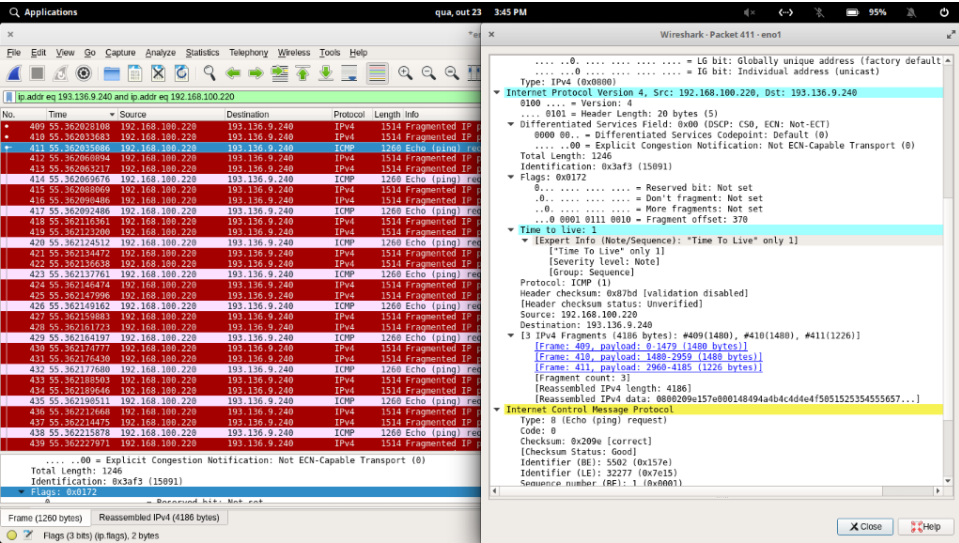
O primeiro fragmento é indicado pelos valores das flags quando o offset tem o valor 0 e o valor do more fragments é 1, tal como o indicado na imagem acima. Sabemos que o segmento foi fragmentado quando o valor das flags é diferente de 0. O tamanho deste datagrama é 1500.

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?



Como explicado na resposta anterior, um certo fragmento é o primeiro quando o offset tem o valor 0 e ‘more fragments’ é 1. Na imagem acima vemos o fragmento diferente de 0 logo trata-se de outro que não o primeiro. Devido á flag more fragments sabemos que existem mais fragmentos depois do atual.

D) Quantos fragmentos foram criados a partir do datagrama original? Como se deteta o último fragmento correspondente ao datagrama original?



O primeiro e o segundo estão expostos nas alíneas anteriores e como o terceiro é último sabemos que foram criados três fragmentos. O último fragmento é dado pela flag ‘more fragments’ igual a zero e o valor do offset diferente de zero como vemos neste caso.

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 ao longo dos diversos fragmentos são os valores das flags 'offset' e 'more fragments'. A flag more fragments permite saber se ainda há ou não fragmentos do datagrama original a circular na rede. O campo offset serve para saber por que ordem devem ser juntos os fragmentos de modo a obter o datagrama original. Os fragmentos são organizados por ordem crescente através dos valores do offset.

Parte 2

Exercício 1

Atenda aos endereços IP atribuídos automaticamente pelo CORE aos diversos equipamentos da topologia.

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

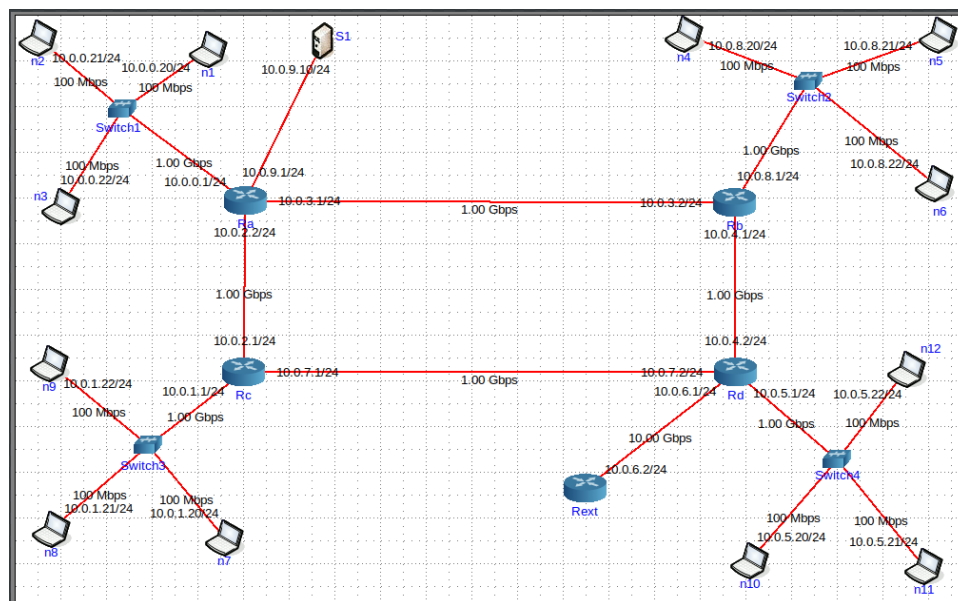


Figura 2.1: Equipamentos e Departamentos

A máscara de rede usada foi: 255.255.255.0 pois todos os equipamentos possuem /24 nos seus endereços.

- b) Trata-se de endereços públicos ou privados? Porquê?
Entre 10.0.0.0 e 255.255.255.0 os endereços são privados. Sabemos que, na nossa topologia, todos os endereços começam com 10, logo, todos eles são privados.
- c) Por que razão não é atribuído um endereço IP aos switches?

Um switch é um equipamento que tem como principal funcionalidade a interligação de equipamentos, permitindo o envio e a receção de informação. Devido à sua funcionalidade, não há necessidade de atribuir um endereço IP a um switch, sendo que, este apenas decide para onde são enviados os pacotes de dados após análise dos endereços MAC dos equipamentos ligados a si.

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

```
root@n5:/tmp/pycore.41593/n5.conf# ping 10.0.0.21
PING 10.0.0.21 (10.0.0.21) 56(84) bytes of data.
64 bytes from 10.0.0.21: icmp_seq=1 ttl=63 time=0.135 ms
64 bytes from 10.0.0.21: icmp_seq=2 ttl=63 time=0.084 ms
64 bytes from 10.0.0.21: icmp_seq=3 ttl=63 time=0.070 ms
64 bytes from 10.0.0.21: icmp_seq=4 ttl=63 time=0.086 ms
64 bytes from 10.0.0.21: icmp_seq=5 ttl=63 time=0.099 ms
64 bytes from 10.0.0.21: icmp_seq=6 ttl=63 time=0.079 ms
64 bytes from 10.0.0.21: icmp_seq=7 ttl=63 time=0.092 ms
64 bytes from 10.0.0.21: icmp_seq=8 ttl=63 time=0.066 ms
64 bytes from 10.0.0.21: icmp_seq=9 ttl=63 time=0.074 ms
64 bytes from 10.0.0.21: icmp_seq=10 ttl=63 time=0.075 ms
64 bytes from 10.0.0.21: icmp_seq=11 ttl=63 time=0.090 ms
^C
--- 10.0.0.21 ping statistics ---
11 packets transmitted, 11 received, 0% packet loss, time 10220ms
rtt min/avg/max/mdev = 0.066/0.086/0.135/0.019 ms
```

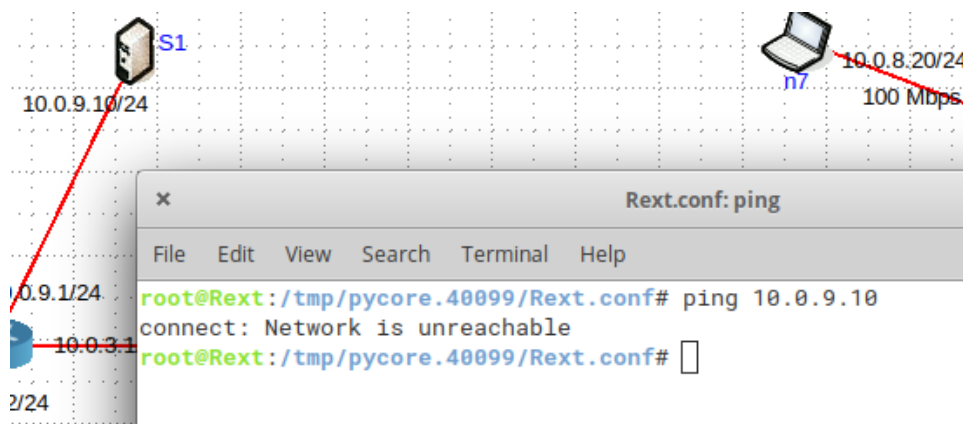
Figura 2.2: Ping Departamento A para S1

```
root@n5:/tmp/pycore.41593/n5.conf# ping 10.0.1.21
PING 10.0.1.21 (10.0.1.21) 56(84) bytes of data.
64 bytes from 10.0.1.21: icmp_seq=1 ttl=62 time=0.147 ms
64 bytes from 10.0.1.21: icmp_seq=2 ttl=62 time=0.087 ms
^C
--- 10.0.1.21 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1026ms
rtt min/avg/max/mdev = 0.087/0.117/0.147/0.030 ms
```

Figura
2.3:
Ping

Departamento B para S1

- e) Verifique se existe conectividade IP do router de acesso R_{ext} para o servidor S1.



Logo, não existe conectividade.

Exercício 2

- 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`).

```

n16.conf: netstat
File Edit View Search Terminal Help

root@n16:/tmp/pycore.41593/n16.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
0.0.0.0 10.0.8.1 0.0.0.0 UG 0 0 0 eth0
10.0.8.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
root@n16:/tmp/pycore.41593/n16.conf#

```

(O n16 da foto é na topologia o n5)

Legenda: tabela de encaminhamento do laptop B

```

n4.conf: netstat
File Edit View Search Terminal Help

root@n4:/tmp/pycore.41593/n4.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
10.0.3.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
10.0.4.0 0.0.0.0 255.255.255.0 U 0 0 0 eth1
10.0.8.0 0.0.0.0 255.255.255.0 U 0 0 0 eth2
root@n4:/tmp/pycore.41593/n4.conf#

```

(O n4 da foto é na topologia o Rb)

Legenda: tabela de encaminhamento do router B

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

Dinâmico porque a rota não foi feita manualmente, foi endereçada automaticamente, por isso, as rotas são obtidas através de protocolos de encaminhamento.


```

root@n16:/tmp/pycore.41593/n16.conf# ps -ef
UID          PID    PPID  C STIME TTY          TIME CMD
root           1         0  0  14:58 ?           00:00:00 /usr/local/bin/vnoded -v -c /tmp
root          19         1  0  14:58 pts/3       00:00:00 /bin/bash
root         103        19  0  15:05 pts/3       00:00:00 ps -ef
root@n16:/tmp/pycore.41593/n16.conf#

```

Legenda: Processos a correr no laptop

```

root@n4:/tmp/pycore.41593/n4.conf# ps -ef
UID          PID    PPID  C STIME TTY          TIME CMD
root           1         0  0  14:58 ?           00:00:00 /usr/local/bin/vnoded -v -c /tmp
quagga         59         1  0  14:58 ?           00:00:00 /usr/sbin/zebra -d
quagga         65         1  0  14:58 ?           00:00:00 /usr/sbin/ospf6d -d
quagga         69         1  0  14:58 ?           00:00:00 /usr/sbin/ospfd -d
root          77         1  0  14:58 pts/5       00:00:00 /bin/bash
root         106        77  0  15:05 pts/5       00:00:00 ps -ef
root@n4:/tmp/pycore.41593/n4.conf#

```

Legenda: Processos a correr no router

Como podemos ver na imagem de baixo, vemos que existem processos a correr o protocolo ZEBRA, o que nos permite concluir que de facto o router está a usar encaminhamento dinâmico. No encaminhamento estático as rotas permanecem fixas e são baseadas nas rotas pré-definidas. Por isso não existe nenhum processo a correr além dos da própria máquina. Analisando a imagem de cima, vemos que os processos que estão a correr são os processos básicos da máquina, podendo concluir assim que o laptop está a usar encaminhamento estático.

- 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.

Este servidor vai deixar de ser acessível através da rede 10.0.8.1. Como o computador, neste momento, não tem o default gateway, não pode aceder à internet pois este é responsável por enviar os pacotes de dados para outras redes. Esta rota, como foi retirada, deixa de ser utilizável.

```

root@n5:/tmp/pycore.41593/n5.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
0.0.0.0 10.0.9.1 0.0.0.0 UG 0 0 0 eth0
10.0.9.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
root@n5:/tmp/pycore.41593/n5.conf# route delete default
root@n5:/tmp/pycore.41593/n5.conf# netstat -rn
Kernel IP routing table
Destination Gateway Genmask Flags MSS Window irtt Iface
10.0.9.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
root@n5:/tmp/pycore.41593/n5.conf#

```

(n5 da imagem == S1 da topologia)

Legenda: Tabela de Encaminhamento do Servidor S1 antes e depois, assim como o comando utilizado para retirar a default route.

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

```

root@n5:/tmp/pycore.41593/n5.conf# route add -net 10.0.6.0 netmask 255.255.255.0
gw 10.0.9.1
root@n5:/tmp/pycore.41593/n5.conf# netstat -rn
Kernel IP routing table
Destination      Gateway          Genmask         Flags   MSS Window  irtt Iface
10.0.0.0         10.0.9.1        255.255.255.0   UG      0 0        0 eth0
10.0.1.0         10.0.9.1        255.255.255.0   UG      0 0        0 eth0
10.0.5.0         10.0.9.1        255.255.255.0   UG      0 0        0 eth0
10.0.6.0         10.0.9.1        255.255.255.0   UG      0 0        0 eth0
10.0.8.0         10.0.9.1        255.255.255.0   UG      0 0        0 eth0
10.0.9.0         0.0.0.0         255.255.255.0   U       0 0        0 eth0
root@n5:/tmp/pycore.41593/n5.conf# ping 10.0.6.2
PING 10.0.6.2 (10.0.6.2) 56(84) bytes of data.
64 bytes from 10.0.6.2: icmp_seq=1 ttl=61 time=0.108 ms
64 bytes from 10.0.6.2: icmp_seq=2 ttl=61 time=0.112 ms
64 bytes from 10.0.6.2: icmp_seq=3 ttl=61 time=0.128 ms
^C
--- 10.0.6.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2045ms
rtt min/avg/max/mdev = 0.108/0.116/0.128/0.008 ms
root@n5:/tmp/pycore.41593/n5.conf#

```

Legenda: Tabela de Encaminhamento do Servidor S1, assim como o comando utilizado para adicionar a rota do Rext.

- 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.
Respondida na imagem anterior.

Exercício 3

1. 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.

O nosso IP é dado por 172.66.32.0/22

O nosso ip da rede é dado por 172.66.48.0/20. Uma vez que, apenas temo uma máscara de 20 significa que a nossa rede pode usar todos os endereços entre 172.66.48.0 e 172.66.63.255. Como temos 3 departamentos, inicialmente, iríamos precisar de 2 bits para conseguir fazer subnetting, no entanto como a possibilidade 00 e 11 estão reservadas ficaríamos com apenas 2 opções, 00 e 01 para representar os 3 departamentos o que torna a situação impossível. Aumentamos assim de 2 bits para 3 bits para representar as 3 redes.

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

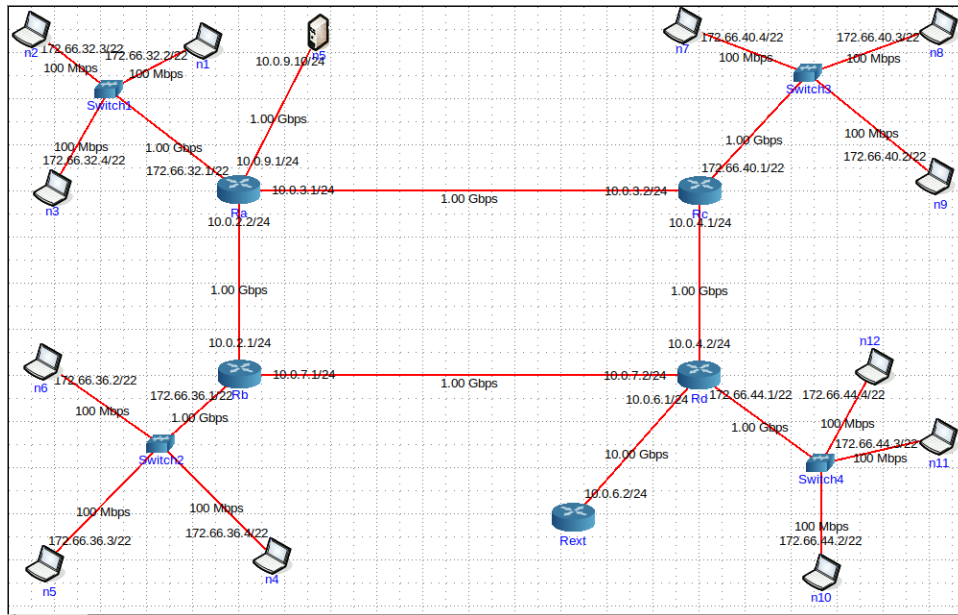
Uma vez que reservamos 2 bits para fazer subnetting a nossa máscara passa de 20 para 22 ficando o seu valor decimal em 255.255.252.0. Como a máscara usa 22 bits ficamos com 10 bits em que podemos mexer. O número de host é então dado por $2^{(10)} - 1$. Como cada rede guarda 2 endereço para broadcast e outro para comunicar com todos os dispositivos, o número de hosts é reduzido em 2 ficando em 1022.

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

Alteramos os valores dos ip's dos dispositivos para os valores atribuídos na tabela da pergunta 2. Para mostrar que a conectividade se mantém usamos o comando ping de n7 para um laptop de cada dispositivo como mostra a figura abaixo.

```
root@n7:/tmp/pycore.33851/n7.conf# ping 172.66.32.1
PING 172.66.32.1 (172.66.32.1) 56(84) bytes of data.
64 bytes from 172.66.32.1: icmp_seq=1 ttl=64 time=0.131 ms
64 bytes from 172.66.32.1: icmp_seq=2 ttl=64 time=0.088 ms
^C
--- 172.66.32.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1004ms
rtt min/avg/max/mdev = 0.088/0.109/0.131/0.023 ms
root@n7:/tmp/pycore.33851/n7.conf# ping 172.66.36.4
PING 172.66.36.4 (172.66.36.4) 56(84) bytes of data.
64 bytes from 172.66.36.4: icmp_seq=1 ttl=62 time=0.164 ms
64 bytes from 172.66.36.4: icmp_seq=2 ttl=62 time=0.141 ms
64 bytes from 172.66.36.4: icmp_seq=3 ttl=62 time=0.188 ms
^C
--- 172.66.36.4 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2045ms
rtt min/avg/max/mdev = 0.141/0.164/0.188/0.021 ms
root@n7:/tmp/pycore.33851/n7.conf# ping 172.66.44.2
PING 172.66.44.2 (172.66.44.2) 56(84) bytes of data.
64 bytes from 172.66.44.2: icmp_seq=1 ttl=61 time=0.221 ms
64 bytes from 172.66.44.2: icmp_seq=2 ttl=61 time=0.170 ms
^C
--- 172.66.44.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1015ms
rtt min/avg/max/mdev = 0.170/0.195/0.221/0.029 ms
root@n7:/tmp/pycore.33851/n7.conf# ping 172.66.40.3
PING 172.66.40.3 (172.66.40.3) 56(84) bytes of data.
64 bytes from 172.66.40.3: icmp_seq=1 ttl=62 time=0.226 ms
64 bytes from 172.66.40.3: icmp_seq=2 ttl=62 time=0.137 ms
64 bytes from 172.66.40.3: icmp_seq=3 ttl=62 time=0.118 ms
^C
--- 172.66.40.3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2042ms
rtt min/avg/max/mdev = 0.118/0.160/0.226/0.048 ms
root@n7:/tmp/pycore.33851/n7.conf#
```

Imagem: Ping para as diversas redes.Equipamentos e Departamentos com novos IP.



Modelo: Equipamentos e Departamentos com novos IP.