

Universidade do Minho

Redes de Computadores

TP2 - Protocolo IPv4

MIEI-3º Ano- 1º semestre

Grupo 1

PL5



Catarina Gil
A85266



Margarida Campos
A85166

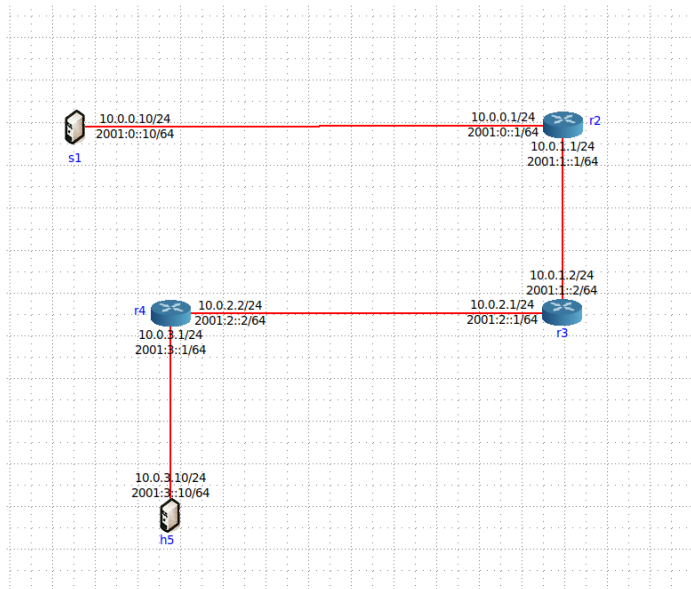


Tânia Rocha
A85176

Parte 1

Pergunta 1

- a) Active o wireshark ou o tcpdump no pc s1. Numa shell de s1, execute o comando `tracert -I` para o endereço IP do host h5.



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

Verificamos que o ttl começa com o valor de 1. Enquanto o ttl é menor ou igual a 3 não obtemos resposta (“no response found”). Sendo que a partir daí a resposta já chega ao destino (para ttl=4).

No.	Time	Source	Destination	Protocol	Length	Info
65	212.901802898	10.0.0.10	10.0.3.10	ICMP	74	Echo (ping) request id=0x0023, seq=1/256, ttl=1 (no response found!)
66	212.901849011	10.0.0.1	10.0.0.10	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
67	212.901882318	10.0.0.10	10.0.3.10	ICMP	74	Echo (ping) request id=0x0023, seq=2/512, ttl=1 (no response found!)
68	212.901904595	10.0.0.1	10.0.0.10	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
69	212.901923996	10.0.0.10	10.0.3.10	ICMP	74	Echo (ping) request id=0x0023, seq=3/768, ttl=1 (no response found!)
70	212.901942565	10.0.0.1	10.0.0.10	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
71	212.901963886	10.0.0.10	10.0.3.10	ICMP	74	Echo (ping) request id=0x0023, seq=4/1024, ttl=2 (no response found!)
72	212.902014924	10.0.1.2	10.0.0.10	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
73	212.902035955	10.0.0.10	10.0.3.10	ICMP	74	Echo (ping) request id=0x0023, seq=5/1280, ttl=2 (no response found!)
74	212.902064030	10.0.1.2	10.0.0.10	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
75	212.902080244	10.0.0.10	10.0.3.10	ICMP	74	Echo (ping) request id=0x0023, seq=6/1536, ttl=2 (no response found!)
76	212.902108913	10.0.1.2	10.0.0.10	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
77	212.902130138	10.0.0.10	10.0.3.10	ICMP	74	Echo (ping) request id=0x0023, seq=7/1792, ttl=3 (no response found!)
78	212.902190784	10.0.2.2	10.0.0.10	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
79	212.902230375	10.0.0.10	10.0.3.10	ICMP	74	Echo (ping) request id=0x0023, seq=8/2048, ttl=3 (no response found!)
80	212.902274552	10.0.2.2	10.0.0.10	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
81	212.902292508	10.0.0.10	10.0.3.10	ICMP	74	Echo (ping) request id=0x0023, seq=9/2304, ttl=3 (no response found!)
82	212.902332206	10.0.2.2	10.0.0.10	ICMP	102	Time-to-live exceeded (Time to live exceeded in transit)
83	212.902355053	10.0.0.10	10.0.3.10	ICMP	74	Echo (ping) request id=0x0023, seq=10/2560, ttl=4 (reply in 84)
84	212.902479940	10.0.3.10	10.0.0.10	ICMP	74	Echo (ping) reply id=0x0023, seq=10/2560, ttl=61 (request in 83)

- 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. O valor mínimo deve ser 4, que é quando o destino começa a responder.

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

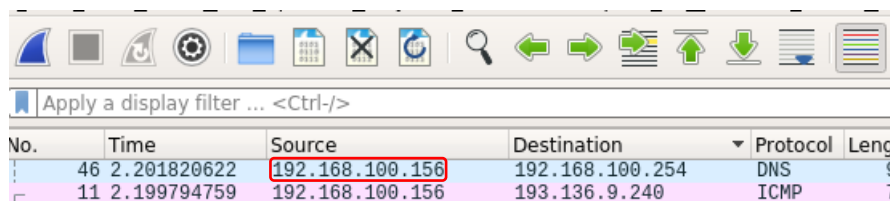
O valor médio = $(0.034+0.011+0.007)/3 + (0.018+0.014+0.015)/3 + (0.025+0.014+0.014)/3 + (0.023+0.017+0.016)/3 = 0.181 \text{ ms}$

```
# traceroute -I 10.0.3.10
traceroute to 10.0.3.10 (10.0.3.10), 30 hops max, 60 byte packets
 1  _gateway (10.0.0.1)  0.034 ms  0.011 ms  0.007 ms
 2  10.0.1.2 (10.0.1.2)  0.018 ms  0.014 ms  0.015 ms
 3  10.0.2.2 (10.0.2.2)  0.025 ms  0.014 ms  0.014 ms
 4  10.0.3.10 (10.0.3.10)  0.023 ms  0.017 ms  0.016 ms
```

Pergunta 2

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

IP: 192.168.100.156



No.	Time	Source	Destination	Protocol	Length
46	2.201820622	192.168.100.156	192.168.100.254	DNS	9
11	2.199794759	192.168.100.156	193.136.9.240	ICMP	7

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

O valor é 1 e identifica o ICMP.

```
Total Length: 60
Identification: 0x7393 (29587)
Flags: 0x0000
Time to live: 1
Protocol: ICMP (1)
Header checksum: 0x5571 [validation disabled]
[Header checksum status: Unverified]
Source: 192.168.100.156
Destination: 193.136.9.240
Internet Control Message Protocol
Type: 8 (Echo (ping) request)
Code: 0
```

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?

cabeçalho = 20 bytes

campo de dados = total length – cabeçalho = 60-20 = 40 bytes

```
Internet Protocol Version 4, Src: 192.168.100.156, Dst: 193.136.9.240
0100 .... = Version: 4
.... 0101 = Header Length: 20 bytes (5)
Differentially Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 60
Identification: 0x7394 (29588)
```

d) O datagrama IP foi fragmentado? Justifique.

Não foi fragmentado porque a flag "Fragment offset" está a zero.

```
▼ Flags: 0x0000
 0... .. = Reserved bit: Not set
.0... .. = Don't fragment: Not set
..0. .... = More fragments: Not set
...0 0000 0000 0000 = Fragment offset: 0
```

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.

Identificação, o ttl e o checksum.

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

O campo de identificação do datagrama ip aumenta uma unidade por pacote e o ttl aumenta uma unidade de 3 em 3 pacotes.

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 permanece constante com o valor de 64.

Pergunta 3

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

Como o MTU é no máximo 1241 bytes e o pacote enviado foi de 4173 bytes, logo houve a necessidade de fragmentar o pacote inicial.

- 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?

O facto da flag “More fragments” estar a Set indica que o datagrama foi fragmentado.
O Fragment offset tem valor 0, indicando que é o primeiro fragmento.
O tamanho deste datagrama IP é de 1500.

```
▼ Flags: 0x2000, More fragments
  0... .. = Reserved bit: Not set
  .0.. .. = Don't fragment: Not set
  ..1. .... = More fragments: Set
  ...0 0000 0000 0000 = Fragment offset: 0

  0000 00.. = Differentiated Service
  .... ..00 = Explicit Congestion Notification
Total Length: 1241
Identification: 0x497c (18812)
▼ Flags: 0x0172
  0... .. = Reserved bit
```

- 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?

Não se trata do 1º fragmento porque a flag offset não tem o valor 0.
Sim, porque a flag more fragments continua a Set.

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

Foram criados 3 fragmentos.
Quando a flag fragment offset é diferente de 0 e não há mais fragmentos.

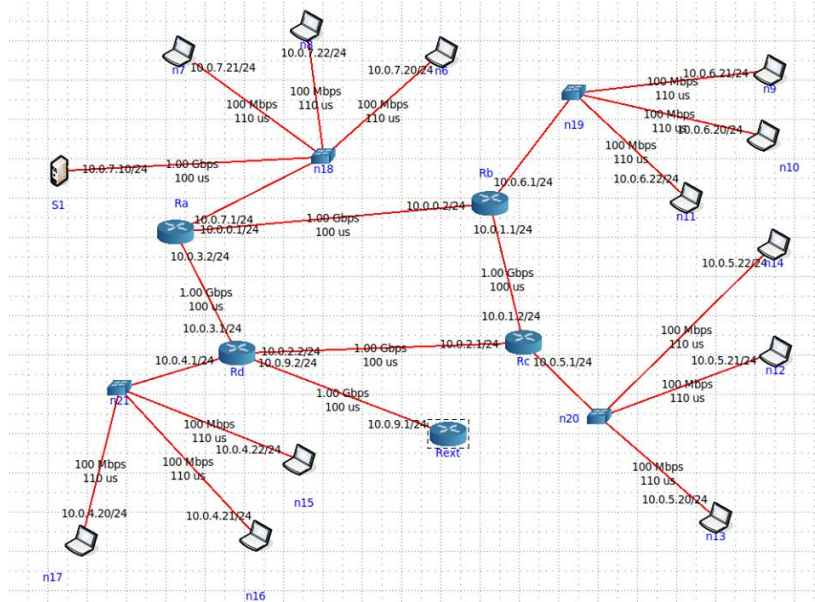
- e)** Indique, resumindo, os campos que mudam no cabeçalho IP entre os diferentes fragmentos, e explique a forma como essa informação permite reconstruir o datagrama original.

Os campos que mudam no cabeçalho IP são a flag fragment offset e no último fragmento a flag more fragments. Para reconstruir o datagrama original ordenamos por ordem crescente de fragment offset os fragmentos.

Parte 2

Pergunta 1

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



- b) Trata-se de endereços públicos ou privados? Porquê?
São privados, porque os seus endereços encontram-se todos dentro do intervalo 10.0.0.0 - 10.255.255.255.
- c) Por que razão não é atribuído um endereço IP aos switches?
Porque não é um equipamento do nível de rede (é de nível 2).
- 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@n7:/tmp/pycore.46381/n7.conf# ping 10.0.7.10
PING 10.0.7.10 (10.0.7.10) 56(84) bytes of data.
64 bytes from 10.0.7.10: icmp_seq=1 ttl=64 time=0.294 ms
64 bytes from 10.0.7.10: icmp_seq=2 ttl=64 time=0.302 ms
^C
--- 10.0.7.10 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1026ms
rtt min/avg/max/mdev = 0.294/0.298/0.302/0.004 ms
```

```
root@n9:/tmp/pycore.46381/n9.conf# ping 10.0.7.10
PING 10.0.7.10 (10.0.7.10) 56(84) bytes of data.
64 bytes from 10.0.7.10: icmp_seq=1 ttl=62 time=0.690 ms
64 bytes from 10.0.7.10: icmp_seq=2 ttl=62 time=0.547 ms
64 bytes from 10.0.7.10: icmp_seq=3 ttl=62 time=0.547 ms
^C
--- 10.0.7.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2033ms
rtt min/avg/max/mdev = 0.547/0.594/0.690/0.073 ms
```

```
root@n15:/tmp/pycore.46381/n15.conf# ping 10.0.7.10
PING 10.0.7.10 (10.0.7.10) 56(84) bytes of data.
64 bytes from 10.0.7.10: icmp_seq=1 ttl=62 time=0.744 ms
64 bytes from 10.0.7.10: icmp_seq=2 ttl=62 time=0.550 ms
^C
--- 10.0.7.10 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1015ms
rtt min/avg/max/mdev = 0.550/0.647/0.744/0.097 ms
```

```
root@n12:/tmp/pycore.46381/n12.conf# ping 10.0.7.10
PING 10.0.7.10 (10.0.7.10) 56(84) bytes of data.
64 bytes from 10.0.7.10: icmp_seq=1 ttl=61 time=1.01 ms
64 bytes from 10.0.7.10: icmp_seq=2 ttl=61 time=0.771 ms
^C
--- 10.0.7.10 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.771/0.892/1.013/0.121 ms
```

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

```
root@Rext:/tmp/pycore.46381/Rext.conf# ping 10.0.7.10
PING 10.0.7.10 (10.0.7.10) 56(84) bytes of data.
64 bytes from 10.0.7.10: icmp_seq=1 ttl=62 time=0.586 ms
64 bytes from 10.0.7.10: icmp_seq=2 ttl=62 time=0.659 ms
^C
--- 10.0.7.10 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1003ms
rtt min/avg/max/mdev = 0.586/0.622/0.659/0.044 ms
```


Pergunta 2

- 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.

Cada tabela é constituída pela Destination, que indica o destino para onde se pode enviar um dado pacote, pelo Gateway, mostra o endereço da interface de saída. Posto isto, os endereços contidos na coluna Destination são os endereços dos Departamentos, dos seus respetivos routers e do servidor s1. Os endereços da coluna Gateway são as ligações diretas através do router em questão. É através destes endereços que é possível chegar a todos os destinos. A Iface diz qual é a interface da rede que deve ser usada para enviar um dado pacote de acordo com o seu destino.

```
root@n9:/tmp/pycore.46381/n9.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@Rb:/tmp/pycore.46381/Rb.conf# netstat -rn
Kernel IP routing table
Destination      Gateway         Genmask         Flags   MSS Window  irtt Iface
10.0.0.0         0.0.0.0         255.255.255.0   U       0 0        0 eth0
10.0.1.0         0.0.0.0         255.255.255.0   U       0 0        0 eth1
10.0.2.0         10.0.1.2        255.255.255.0   UG      0 0        0 eth1
10.0.3.0         10.0.0.1        255.255.255.0   UG      0 0        0 eth0
10.0.4.0         10.0.0.1        255.255.255.0   UG      0 0        0 eth0
10.0.5.0         10.0.1.2        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         10.0.0.1        255.255.255.0   UG      0 0        0 eth0
10.0.9.0         10.0.0.1        255.255.255.0   UG      0 0        0 eth0
```

- b) Diga, justificando, se está a ser usado encaminhamento estático ou dinâmico.

O endereçamento que está a ser utilizado é dinâmico, pois existe mais do que uma forma de aceder aos diferentes equipamentos. Podemos ver isso ao utilizar `traceroute` e ver que nem sempre é tomada a mesma rota.

```
root@n7:/tmp/pycore.34631/n7.conf# traceroute -I 10.0.6.21
traceroute to 10.0.6.21 (10.0.6.21), 30 hops max, 60 byte packets
 1  _gateway (10.0.7.1)  0.202 ms  0.138 ms  0.129 ms
 2  10.0.0.2 (10.0.0.2)  0.382 ms  0.364 ms  0.354 ms
 3  10.0.6.21 (10.0.6.21)  0.526 ms  0.545 ms  0.524 ms
```

```
root@n16:/tmp/pycore.34631/n16.conf# traceroute -I 10.0.6.21
traceroute to 10.0.6.21 (10.0.6.21), 30 hops max, 60 byte packets
 1  _gateway (10.0.4.1)  0.431 ms  0.338 ms  0.332 ms
 2  10.0.3.2 (10.0.3.2)  0.630 ms  0.625 ms  0.622 ms
 3  10.0.1.1 (10.0.1.1)  0.910 ms  0.906 ms  0.903 ms
 4  10.0.6.21 (10.0.6.21)  1.103 ms  1.100 ms  1.096 ms
```


- 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.

Na primeira imagem a partir de S1 conseguimos aceder a duas redes: 0.0.0.0 e 10.0.7.0 .

Após o comando `route delete default`, deixamos de ter acesso à 0.0.0.0 através da 10.0.7.1 .

Com isto, a conexão do s1 com o Ra deixa de existir. Como os pc's dos departamentos B,C e D necessitam de passar em Ra para se conectarem com s1, também não têm acesso a s1.

No entanto, os pc's do departamento A têm acesso ao s1, uma vez que estão ligados por um switch (n18) e não pelo Ra.

Apesar de tudo todos os pc's de todos os departamentos têm conexão entre si.

```
root@S1:/tmp/pycore.34631/S1.conf# netstat -rn
Kernel IP routing table
Destination    Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0        10.0.7.1        0.0.0.0         UG      0  0        0 eth1
10.0.7.0       0.0.0.0         255.255.255.0   U       0  0        0 eth1
```

```
root@S1:/tmp/pycore.34631/S1.conf# netstat -rn
Kernel IP routing table
Destination    Gateway         Genmask         Flags   MSS Window  irtt Iface
10.0.7.0       0.0.0.0         255.255.255.0   U       0  0        0 eth1
```

- 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.

```
route add -net 10.0.0.0 netmask 255.255.255.0 gw 10.0.7.1 eth1
route add -net 10.0.1.0 netmask 255.255.255.0 gw 10.0.7.1 eth1
route add -net 10.0.2.0 netmask 255.255.255.0 gw 10.0.7.1 eth1
route add -net 10.0.3.0 netmask 255.255.255.0 gw 10.0.7.1 eth1
route add -net 10.0.4.0 netmask 255.255.255.0 gw 10.0.7.1 eth1
route add -net 10.0.5.0 netmask 255.255.255.0 gw 10.0.7.1 eth1
route add -net 10.0.6.0 netmask 255.255.255.0 gw 10.0.7.1 eth1
route add -net 10.0.7.0 netmask 255.255.255.0 gw 10.0.7.1 eth1
route add -net 10.0.9.0 netmask 255.255.255.0 gw 10.0.7.1 eth1
```

Com estes comandos garantimos a conectividade entre o servidor s1 e os departamentos.

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

```
root@S1:/tmp/pycore.38201/S1.conf# ping 10.0.6.1
4PING 10.0.6.1 (10.0.6.1) 56(84) bytes of data.
64 bytes from 10.0.6.1: icmp_seq=1 ttl=63 time=0.437 ms
64 bytes from 10.0.6.1: icmp_seq=2 ttl=63 time=0.429 ms
64 bytes from 10.0.6.1: icmp_seq=3 ttl=63 time=0.376 ms
^C
--- 10.0.6.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2047ms
rtt min/avg/max/mdev = 0.376/0.414/0.437/0.027 ms
```

Conectividade entre s1 e departamento B

```
root@S1:/tmp/pycore.38201/S1.conf# netstat -rn
Kernel IP routing table
Destination      Gateway         Genmask         Flags        MSS Window  irtt Iface
10.0.0.0          10.0.7.1        255.255.255.0   UG           0 0        0 eth1
10.0.1.0          10.0.7.1        255.255.255.0   UG           0 0        0 eth1
10.0.2.0          10.0.7.1        255.255.255.0   UG           0 0        0 eth1
10.0.3.0          10.0.7.1        255.255.255.0   UG           0 0        0 eth1
10.0.4.0          10.0.7.1        255.255.255.0   UG           0 0        0 eth1
10.0.5.0          10.0.7.1        255.255.255.0   UG           0 0        0 eth1
10.0.6.0          10.0.7.1        255.255.255.0   UG           0 0        0 eth1
10.0.7.0          0.0.0.0         255.255.255.0   U            0 0        0 eth1
10.0.9.0          10.0.7.1        255.255.255.0   UG           0 0        0 eth1
```

Pergunta 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.

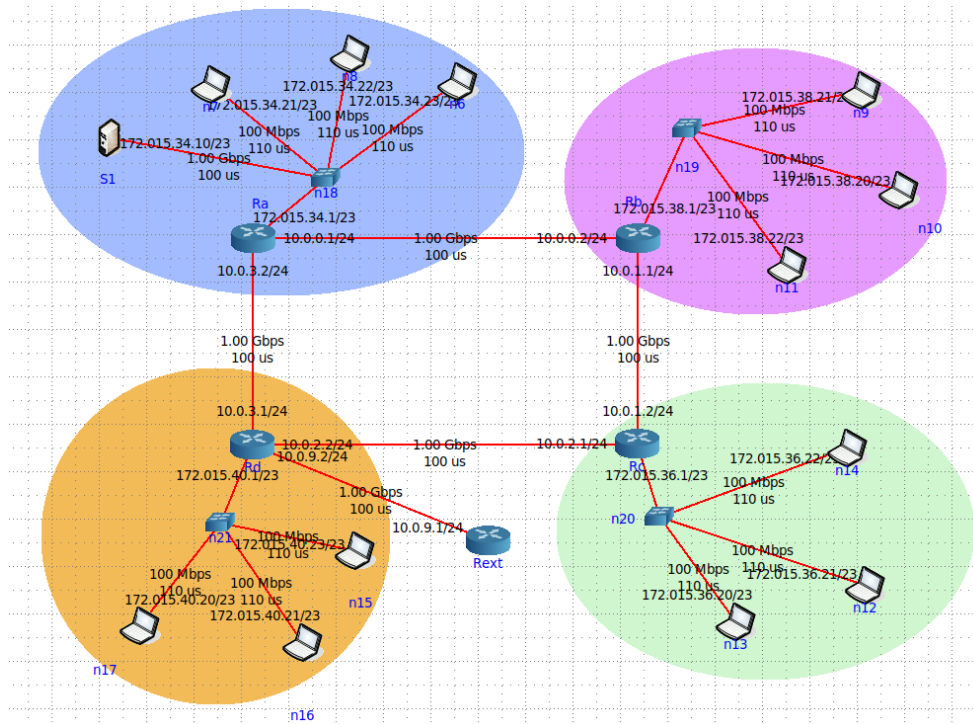
IP= 172.15.32.0/20

Temos $2^n - 2$ sub-redes. Para os 4 departamentos existentes, precisamos de definir 4 sub-redes. Para isto optamos por usar 3 bits ficando com $(2^3 - 2) = 6$ endereços disponíveis.

A máscara da rede passa por isso a ser 255.255.254.0(/23).

Os endereços atribuídos aos diferentes departamentos encontram-se na tabela seguinte:

000	172.15.32.0/23	Reservado para default
001	172.15.34.0/23	Departamento A
010	172.15.36.0/23	Departamento C
011	172.15.38.0/23	Departamento B
100	172.15.40.0/23	Departamento D
101	172.15.42.0/23	Livre
110	172.15.44.0/23	Livre
111	172.15.46.0/23	Reservado para broadcast



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

Máscara da rede (/23) : 255.255.254.0

Em cada departamento é possível atribuir $(2^{(32-23)}-2-1 =)$ 509 interfaces IP.

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

Para garantir a conectividade IP entre as várias redes locais da organização fizemos o comando ping de um computador de um departamento a computadores de outros departamentos.

```
root@n7:/tmp/pycore.43721/n7.conf# ping 172.015.36.20
PING 172.015.36.20 (172.13.36.20) 56(84) bytes of data.
64 bytes from 172.13.36.20: icmp_seq=1 ttl=61 time=0.993 ms
64 bytes from 172.13.36.20: icmp_seq=2 ttl=61 time=0.790 ms
64 bytes from 172.13.36.20: icmp_seq=3 ttl=61 time=0.781 ms
^C
--- 172.015.36.20 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2017ms
rtt min/avg/max/mdev = 0.781/0.854/0.993/0.103 ms
```

```
root@n7:/tmp/pycore.43721/n7.conf# ping 172.015.38.20
PING 172.015.38.20 (172.13.38.20) 56(84) bytes of data.
64 bytes from 172.13.38.20: icmp_seq=1 ttl=62 time=0.723 ms
64 bytes from 172.13.38.20: icmp_seq=2 ttl=62 time=0.579 ms
64 bytes from 172.13.38.20: icmp_seq=3 ttl=62 time=0.560 ms
^C
--- 172.015.38.20 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2033ms
rtt min/avg/max/mdev = 0.560/0.620/0.723/0.078 ms
```

```
root@n7:/tmp/pycore.43721/n7.conf# ping 172.015.34.20
PING 172.015.34.20 (172.13.34.20) 56(84) bytes of data.
64 bytes from 172.13.34.20: icmp_seq=1 ttl=62 time=0.909 ms
64 bytes from 172.13.34.20: icmp_seq=2 ttl=62 time=0.558 ms
64 bytes from 172.13.34.20: icmp_seq=3 ttl=62 time=0.556 ms
^C
--- 172.015.34.20 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2047ms
rtt min/avg/max/mdev = 0.556/0.674/0.909/0.167 ms
```

Conclusão

Como referido, a primeira componente do trabalho envolveu um estudo ao protocolo IPv4. A partir da ferramenta Core foi possível aplicar os conhecimentos obtidos e assim desenvolver a topologia pretendida. Através da ferramenta Wireshark foram captados pacotes com o objetivo de analisar a fragmentação dos pacotes IP, situação que ocorre quando o tamanho do mesmo é superior ao MTU da rede.

Na segunda componente do trabalho aprofundamos conhecimentos relativos ao endereçamento e encaminhamento IP. Foi estudada como funciona o encaminhamento entre 4 redes diferentes (4 departamentos) e como é feita a manipulação de endereços IP para efeitos de subnetting.