

# TP1 - Encaminhamento de Tráfego

## Grupo 4

Bruno Silva  
(a71385)

João Bernardo Freitas  
(a74814)

Eduardo Gil Rocha  
(a77048)

17 de Outubro 2019

### 1 Pergunta 1

A topologia da rede foi desenhada da seguinte forma:

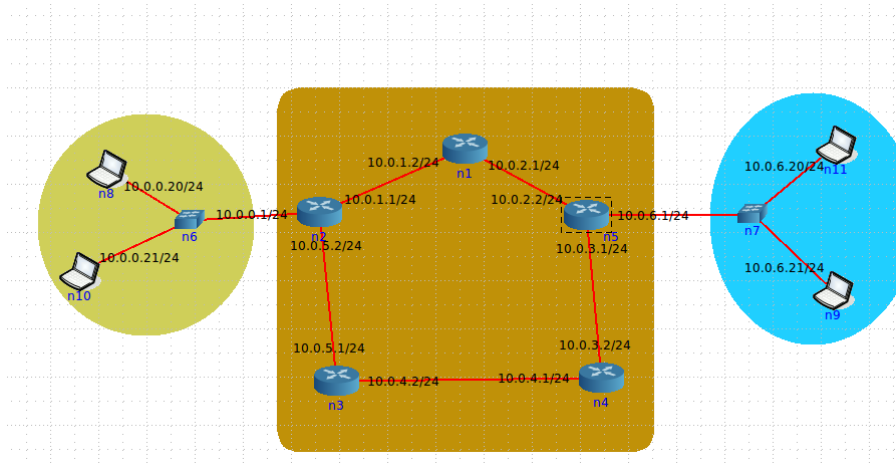


Figure 1: Topologia da rede

## 2 Pergunta 2

Após a criação da topologia reparamos que haviam 5 redes criadas entre os 5 *routers* e 2 redes criadas para os clientes, levando o total de redes para 7.

- Rede 0 - n2, n6, n8, n10 (10.0.0.0)
- Rede 1 - n1, n2 (10.0.1.0)
- Rede 2 - n1, n5 (10.0.2.0)
- Rede 3 - n4, n5 (10.0.3.0)
- Rede 4 - n4, n4 (10.0.4.0)
- Rede 5 - n2, n3 (10.0.5.0)
- Rede 6 - n5, n7, n9, n11 (10.0.6.0)

## 3 Pergunta 3

Para ser utilizado o protocolo *RIP*, foi preciso alterar as configurações dos *routers* para incluir estas opções:

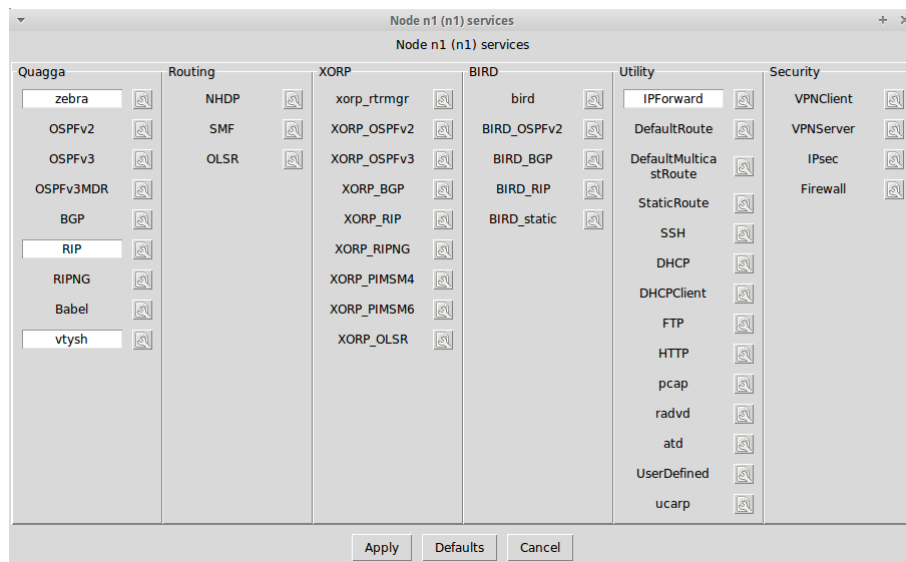


Figure 2: Configuração do uso do protocolo *RIP*

## 4 Pergunta 4

De seguida iremos mostrar várias imagens com testes à topologia criada.

A primeira imagem mostra um teste de *ping* a partir de o nó *n2* até ao nó *n9*, em que são enviados quatro pacotes.

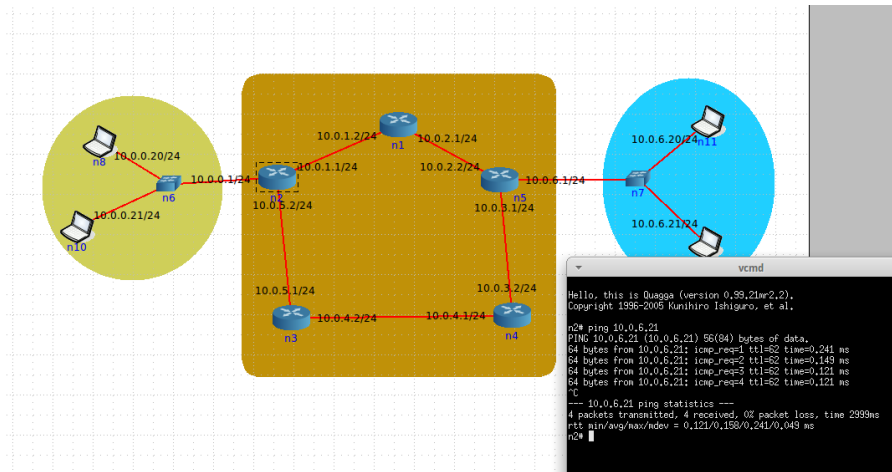


Figure 3: Testes *ping* a partir do nó *n2* para o nó *n9*

A segunda imagem mostra o uso do comando *tracert* usado para mostrar o caminho registado a partir do nó *n9* até ao endereço *10.0.0.21*, usado na máquina *n10*.

```
root@n9:/tmp/pycore.44394/n9.conf# tracert 10.0.0.21
 1: 10.0.6.21                                0.182ms pmtu 1500
 1: 10.0.6.1                                0.284ms
 1: 10.0.6.1                                0.111ms
 2: 10.0.2.1                                0.149ms
 3: 10.0.1.1                                0.217ms
 4: 10.0.0.21                              0.178ms reached
Resume: pmtu 1500 hops 4 back 61
root@n9:/tmp/pycore.44394/n9.conf#
```

Figure 4: Testes *ping* a partir do nó *n9*

Por último, temos um segundo teste usando o comando *tracert*, desta vez usado a partir do nó *n11*, mas com o mesmo destino que o teste anterior, o nó *n10*.

```

root@n11:/tmp/pycore.44394/n11.conf# tracepath 10.0.0.21
 1: 10.0.6.20          0.164ms pmtu 1500
 1: 10.0.6.1           0.254ms
 1: 10.0.6.1           0.077ms
 2: 10.0.2.1           0.269ms
 3: 10.0.1.1           0.195ms
 4: 10.0.0.21          0.188ms reached
Resume: pmtu 1500 hops 4 back 61

```

Figure 5: Testes *ping* a partir do nó *n11*

## 5 Pergunta 5

### 5.1 Tabela Routing Routers

Pela topologia reparamos que o nó *n2* tem interface ligadas a 3 outros nós, nomeadamente *n6*, *n1* e *n3* com endereços 10.0.0.0/24, 10.0.1.0/24 e 10.0.5.0/24. Também reparamos que tem ligação aos restantes nós da topologia através desses primeiros 3.

- 10.0.2.0/24 (*n5*) através de 10.0.1.2/24 (*n1*) com custo 2
- 10.0.3.0/24 (*n5*) através de 10.0.5.1/24 (*n3*) com custo 3
- 10.0.4.0/24 (*n4*) através de 10.0.5.1/24 (*n3*) com custo 2
- 10.0.6.0/24 (*n7*) através de 10.0.1.2/24 (*n1*) com custo 3

```

n2# sh ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, o - OSPF6, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

C>* 10.0.0.0/24 is directly connected, eth0
C>* 10.0.1.0/24 is directly connected, eth1
R>* 10.0.2.0/24 [120/2] via 10.0.1.2, eth1, 00:01:55
R>* 10.0.3.0/24 [120/3] via 10.0.5.1, eth2, 00:01:54
R>* 10.0.4.0/24 [120/2] via 10.0.5.1, eth2, 00:01:54
C>* 10.0.5.0/24 is directly connected, eth2
R>* 10.0.6.0/24 [120/3] via 10.0.1.2, eth1, 00:01:50
C>* 127.0.0.0/8 is directly connected, lo
n2#

```

Figure 6: Uso do comando *sh ip route* no nó *n2*

## 5.2 Tabela Routing Hosts

Vemos aqui que a tabela de routing dos hosts é mais simples visto que para comunicarem com qualquer outro nó da topologia têm que enviar para o nó n7, logo essa rota fica como a default.

```
root@n11:/tmp/pycore.44392/n11.conf# ip route
default via 10.0.6.1 dev eth0
10.0.6.0/24 dev eth0 proto kernel scope link src 10.0.6.20
```

Figure 7: Uso do comando *ip route* no host *n11*

## 6 Pergunta 6

Por defeito, o *update time* do protocolo *RIP* está definido a 30 segundos. Para alterar o tempo de update seria necessário utilizar o comando "*timers basic*".

```
Hello, this is Quagga (version 0.99.21mr2.2).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

n1# configure terminal
n1(config)# router rip
n1(config-router)# timers basic 30
<5-2147483647> Routing table update timer value in second. Default is 30.
n1(config-router)# timers basic 30 180
<5-2147483647> Routing information timeout timer. Default is 180.
n1(config-router)# timers basic 30 180
<5-2147483647> Garbage collection timer. Default is 120.
n1(config-router)# timers basic 30 180 120
n1(config-router)#
```

Figure 8: Alteração do *update timer* para 30 segundos

## 7 Pergunta 7

### 7.1

Pela seguinte imagem, do nó n2, decidimos remover a interface *eth2* visto que é a interface que liga o nó n2 ao nó n3.

```
n2# sh ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       0 - OSPF, o - OSPF6, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

C>* 10.0.0.0/24 is directly connected, eth0
C>* 10.0.1.0/24 is directly connected, eth1
R>* 10.0.2.0/24 [120/2] via 10.0.1.2, eth1, 00:01:55
R>* 10.0.3.0/24 [120/3] via 10.0.5.1, eth2, 00:01:54
R>* 10.0.4.0/24 [120/2] via 10.0.5.1, eth2, 00:01:54
C>* 10.0.5.0/24 is directly connected, eth2
R>* 10.0.6.0/24 [120/3] via 10.0.1.2, eth1, 00:01:50
C>* 127.0.0.0/8 is directly connected, lo
n2#
```

Figure 9: Rotas antes da remoção da interface eth2

A remoção foi feita através da edição da configuração no router n2 adicionando a linha "shutdown" na parte relativa á interface 2.

```
interface eth0
 ip address 10.0.0.1/24
!
interface eth1
 ip address 10.0.1.1/24
!
interface eth2
 ip address 10.0.5.2/24
 shutdown
!
router rip
 redistribute static
 redistribute connected
 redistribute ospf
 network 0.0.0.0/0
!
```

Figure 10: Remoção de eth2

Após nova aprendizagem das rotas através do protocolo RIP obtemos as seguinte rotas.

```
n2# sh ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, o - OSPF6, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

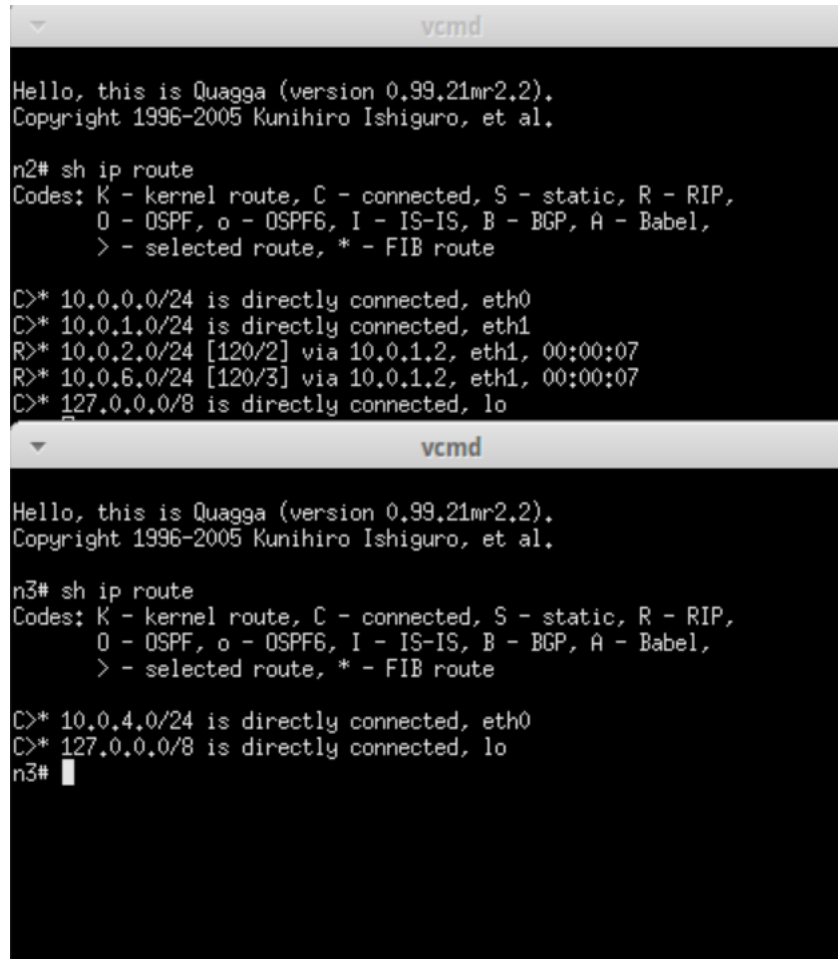
C>* 10.0.0.0/24 is directly connected, eth0
C>* 10.0.1.0/24 is directly connected, eth1
R>* 10.0.2.0/24 [120/2] via 10.0.1.2, eth1, 00:00:08
R>* 10.0.3.0/24 [120/3] via 10.0.1.2, eth1, 00:00:02
R>* 10.0.4.0/24 [120/4] via 10.0.1.2, eth1, 00:00:02
R>* 10.0.5.0/24 [120/5] via 10.0.1.2, eth1, 00:00:02
R>* 10.0.6.0/24 [120/3] via 10.0.1.2, eth1, 00:00:02
C>* 127.0.0.0/8 is directly connected, lo
n2#
```

Figure 11: Rotas depois da remoção da interface eth2 no nó 2

Como podemos observar o nó n2 sabe uma rota para todos os outros nós/redes mas nenhuma das rotas tem como intermédio rede 10.0.5.0. Como também não há ligação directa entre n2 e n3 o número de saltos necessários para uma mensagem ser enviada do nó n2 para a rede 10.0.5.0 sobe de 1 para 5 saltos.

## 7.2

Para este exercício decidimos isolar os nós n3 e n4 do resto da topologia, para tal é necessário efectuar a mesma mudança na configuração do exercício anterior nos nós n2, n3, n4 e n5, removendo as interfaces eth2, eth1, eth0 e eth1 respetivamente.



```
vcmd
Hello, this is Quagga (version 0.99.21mr2.2).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

n2# sh ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, o - OSPF6, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

C>* 10.0.0.0/24 is directly connected, eth0
C>* 10.0.1.0/24 is directly connected, eth1
R>* 10.0.2.0/24 [120/2] via 10.0.1.2, eth1, 00:00:07
R>* 10.0.6.0/24 [120/3] via 10.0.1.2, eth1, 00:00:07
C>* 127.0.0.0/8 is directly connected, lo

vcmd
Hello, this is Quagga (version 0.99.21mr2.2).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

n3# sh ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, o - OSPF6, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

C>* 10.0.4.0/24 is directly connected, eth0
C>* 127.0.0.0/8 is directly connected, lo
n3#
```

Figure 12: Rotas após remoção de todas as interfaces relevantes

Como podemos observar, o nó n2 e o resto da topologia que está conectada ao nó n2, não tem acesso aos nós n3 e n4, enquanto que os nós n3 e n4 só têm acesso um ao outro e não ao resto da topologia.



## 8 Pergunta 8

Como, no protocolo RIP, a rota escolhida é a rota com menor número de hops temos que adicionar hops artificialmente a um router. Para conseguirmos isso temos que utilizar os comandos `offset-list` e `access-list` na configuração do router n1, visto que é este o router que queremos evitar. Começamos então por obter um `tracpath` antes de editar a configuração do router n1.

```
root@n2:/tmp/pycore.55034/n2.conf# tracepath 10.0.6.20
  0: 10.0.1.1 0.162ms pmtu 1500
  1: 10.0.1.2 0.142ms
  1: 10.0.1.2 0.102ms
  2: 10.0.2.2 0.165ms
  3: 10.0.6.20 0.216ms reached
Resume: pmtu 1500 hops 3 back 62
root@n2:/tmp/pycore.55034/n2.conf#
```

Figure 13: Tracepath do router n2 para n1 antes do `offset-list`

Como podemos observar qualquer mensagem enviada do router n2 para n1 tem primeiro de passar por n1 e n5. Adicionamos agora os comandos:

- **`offset-list 1 out 5`** (cria a lista com nome 1, updates outbound que adiciona 5 hops)
- **`access-list 1 permit 10.0.6.0 0.0.0.255`** (faz com que a lista 1 só afecte comunicações com destino á rede 10.0.6.0)

```
interface eth0
 ip address 10.0.1.2/24
!
interface eth1
 ip address 10.0.2.1/24
!
access-list 1 permit 10.0.6.0 0.0.0.255

router rip
 redistribute static
 redistribute connected
 redistribute ospf
 network 0.0.0.0/0
 offset-list 1 out 5
!
```

Figure 14: Configuração do router n1

Agora podemos observar que quando n2 envia mensagem para n11 passa por n3,n4 e n5 evitando n1.

```
root@n2:/tmp/pycore.55037/n2.conf# tracepath 10.0.6.20
  1: 10.0.5.2                                0.187ms pmtu 1500
  1: 10.0.5.1                                0.208ms
  1: 10.0.5.1                                0.152ms
  2: 10.0.4.1                                0.261ms
  3: 10.0.2.2                                0.369ms asymm 2
  4: 10.0.6.20                               0.596ms reached
Resume: pmtu 1500 hops 4 back 62
```

Figure 15: Tracepath do router n2 para n11 depois do offset-list

Observamos também que as rotas que não têm como destino a rede **10.0.6.0** não são afetadas.

```
root@n2:/tmp/pycore.55040/n2.conf# tracepath 10.0.2.2
  1: 10.0.1.1                                0.152ms pmtu 1500
  1: 10.0.1.2                                0.095ms
  1: 10.0.1.2                                0.038ms
  2: 10.0.2.2                                0.067ms reached
Resume: pmtu 1500 hops 2 back 63
```

Figure 16: Tracepath do router n2 para n5 depois do offset-list

## 9 Pergunta 9

Começamos por alterar a topologia anterior de forma a incluir uma rede externa ligada ao *router x*.

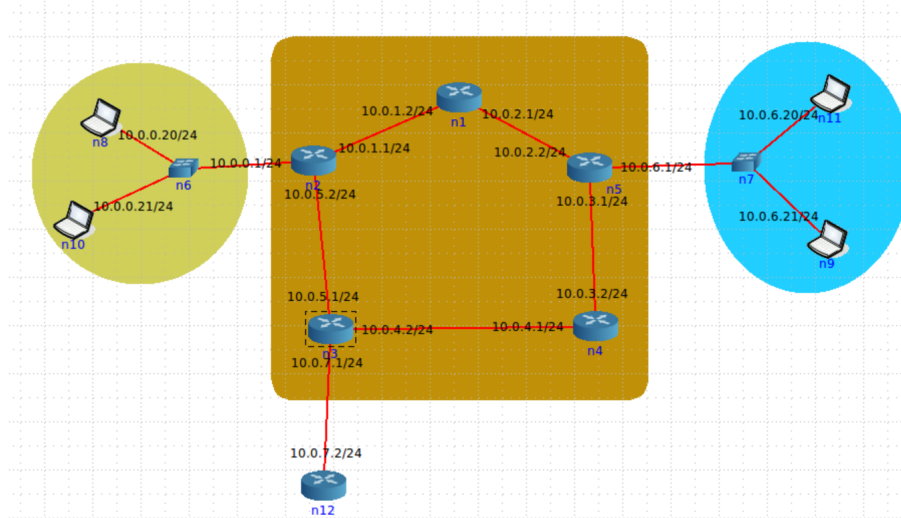


Figure 17: Topologia com rede externa

Há duas possibilidades para forçar que todo o tráfego dirigido a n12 passe por n3:

- Adicionámos rota estática em todos os routers da rede de interligação
- Adicionámos rota estática a n3 e propagamos essa rota através do protocolo RIP

Para resolver este exercício decidimos utilizar a segunda hipótese, para tal editamos a configuração de n3 de forma a incluir estes dois comandos:

- `ip route 10.0.7.2/24 eth2` (cria a rota estática para n3)
- `default-information originate` (propaga a rota através do protocolo RIP)

```

interface eth0
 ip address 10.0.4.2/24
!
interface eth1
 ip address 10.0.5.1/24
!
interface eth2
 ip address 10.0.7.1/24
!
ip route 10.0.7.2/24 eth2
router rip
 redistribute static
 redistribute connected
 redistribute ospf
 network 0.0.0.0/0
 default-information originate
!

```

Figure 18: Configuração de n3

Obtendo então estas tabelas de routing.

```

n3# sh ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, o - OSPF6, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

R>* 10.0.0.0/24 [120/2] via 10.0.5.2, eth1, 00:00:12
R>* 10.0.1.0/24 [120/2] via 10.0.5.2, eth1, 00:00:12
R>* 10.0.2.0/24 [120/3] via 10.0.5.2, eth1, 00:00:12
R>* 10.0.3.0/24 [120/2] via 10.0.4.1, eth0, 00:00:11
C>* 10.0.4.0/24 is directly connected, eth0
C>* 10.0.5.0/24 is directly connected, eth1
R>* 10.0.6.0/24 [120/3] via 10.0.4.1, eth0, 00:00:11
S 10.0.7.0/24 [1/0] is directly connected, eth2
C>* 10.0.7.0/24 is directly connected, eth2
C>* 127.0.0.0/8 is directly connected, lo
n3#

```

Figure 19: sh ip route n3

```

n4# sh ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, o - OSPF6, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

R>* 0.0.0.0/0 [120/2] via 10.0.4.2, eth1, 00:00:49
R>* 10.0.0.0/24 [120/3] via 10.0.4.2, eth1, 00:00:49
R>* 10.0.1.0/24 [120/3] via 10.0.4.2, eth1, 00:00:49
R>* 10.0.2.0/24 [120/2] via 10.0.3.1, eth0, 00:00:48
C>* 10.0.3.0/24 is directly connected, eth0
C>* 10.0.4.0/24 is directly connected, eth1
R>* 10.0.5.0/24 [120/2] via 10.0.4.2, eth1, 00:00:49
R>* 10.0.6.0/24 [120/2] via 10.0.3.1, eth0, 00:00:48
R>* 10.0.7.0/24 [120/2] via 10.0.4.2, eth1, 00:00:49
C>* 127.0.0.0/8 is directly connected, lo
n4#

```

Figure 20: sh ip route n4

## 10 Pergunta 10

Para cada router na rede de interligação foi necessário atribuir o valor da largura de banda a cada link adjacente.

```
interface eth0
  ip address 10.0.0.1/24
interface eth1
  ip address 10.0.1.1/24
  bandwidth 100000000
!
interface eth2
  ip address 10.0.2.1/24
  bandwidth 100000000
!
```

Figure 21: Largura de banda atribuída aos links saídos do router (n3)

Após a configuração dos valores da largura de banda, ficamos então com a seguinte visão geral da topologia :

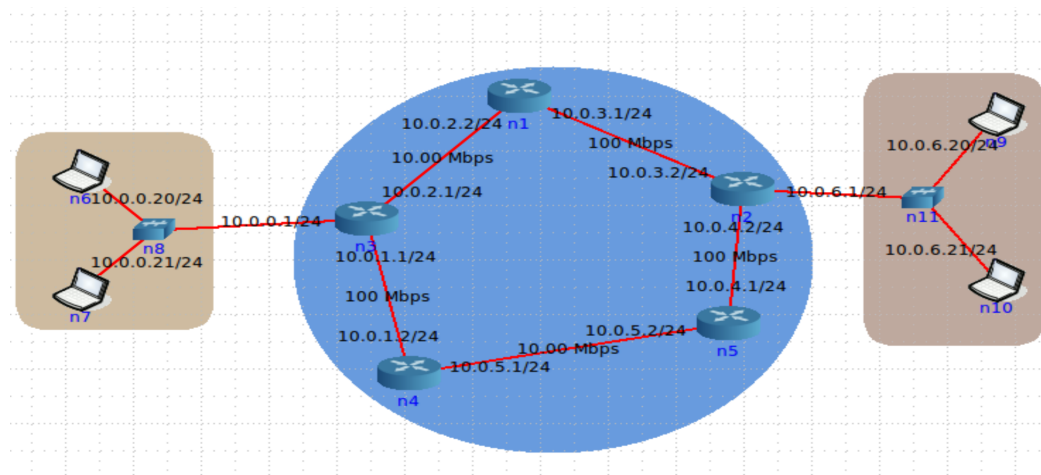


Figure 22: Topologia com links de 10 e 100Mbps de largura de banda

## 11 Pergunta 11

Foi definida na interface gráfica a configuração dos routers para OSPF. Estes estão localizados na área 0.

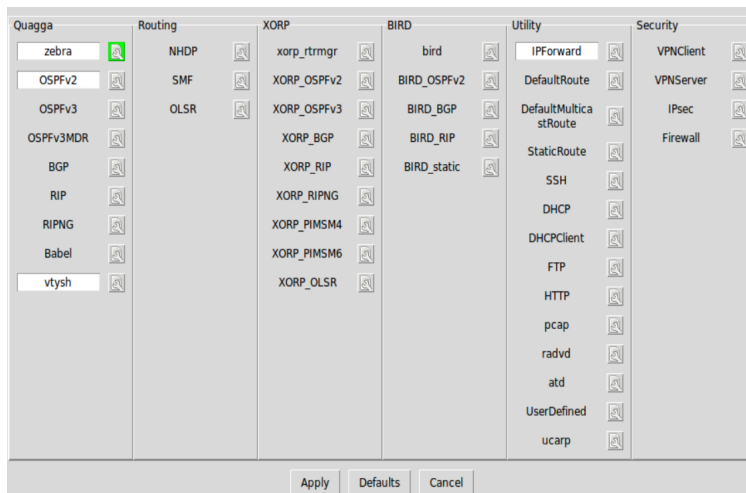


Figure 23: Configuração do uso do protocolo OSPF

```
ns2# sh ip ospf database
OSPF Router with ID (10.0.0.1)

Router Link States (Area 0.0.0.0)

Link ID      ADV Router   Age  Seq#       CkSum  Link count
10.0.0.1     10.0.0.1     184  0x80000009 0x883b  3
10.0.1.2     10.0.1.2     185  0x80000007 0x9549  2
10.0.2.2     10.0.2.2     190  0x80000007 0xa935  2
10.0.3.2     10.0.3.2     195  0x80000009 0xb5ed  3
10.0.4.1     10.0.4.1     195  0x80000006 0xb521  2

Net Link States (Area 0.0.0.0)

Link ID      ADV Router   Age  Seq#       CkSum
10.0.1.2     10.0.1.2     185  0x80000001 0x5acd
10.0.2.2     10.0.2.2     190  0x80000001 0x55cf
10.0.3.2     10.0.3.2     195  0x80000001 0x6cb2
10.0.4.1     10.0.4.1     196  0x80000001 0x72ab
10.0.5.2     10.0.4.1     196  0x80000001 0x4bd2
```

Figure 24: Redes localizadas na área 0

## 12 Pergunta 12

Foi estabelecida ligação entre o router n3 e o host n10 para efeitos de teste de conectividade.

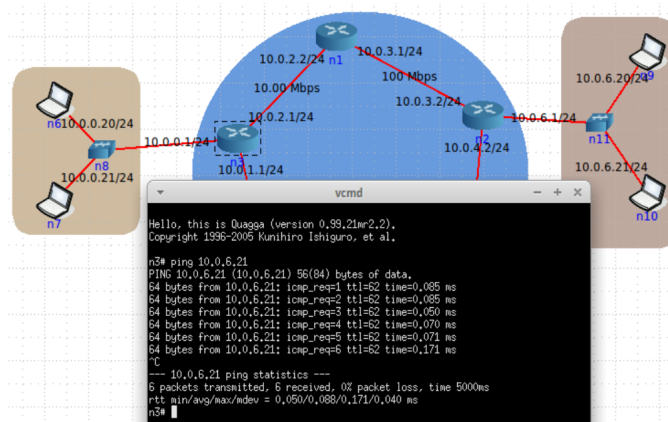


Figure 25: Ping do router n3 para o host n10

Foram analisadas as rotas estabelecidas a partir do host n10 para o host n7

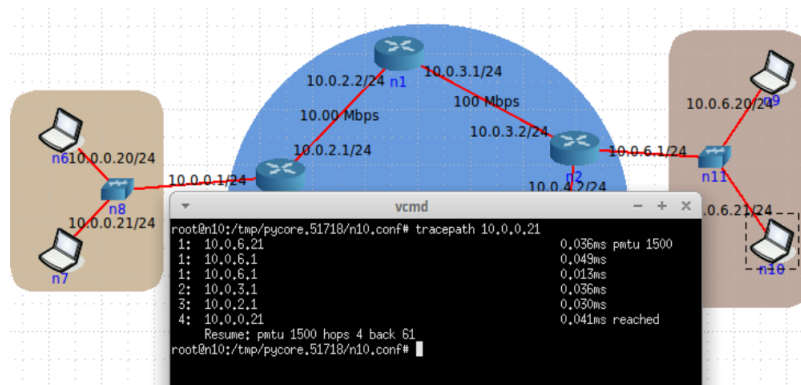


Figure 26: Testes a partir do nó n10

## 13 Pergunta 13

Nos routers de interligação, foram introduzidos os endereços dos nós vizinhos, assim como a largura de banda de cada link. Além disso é definida uma área onde se encontram esses vizinhos.

```
interface eth0
 ip address 10.0.0.1/24
!
interface eth1
 ip address 10.0.1.1/24
 bandwidth 100000000
!
interface eth2
 ip address 10.0.2.1/24
 bandwidth 10000000
!
router ospf
 router-id 10.0.0.1
 network 10.0.0.0/24 area 0
 network 10.0.1.0/24 area 0
 network 10.0.2.0/24 area 0
!
```

Figure 27: Configuração de um router na rede de interligação

## 14 Pergunta 14

Foi selecionada a tabela de routing no nó n3, de maneira a que se possa fazer uma análise detalhada da ligação de toda a rede. Tendo em conta a topologia já explícita em imagens anteriores, podemos concluir o seguinte:

- O router n3 está diretamente ligado às redes 10.0.0.0, 10.0.1.0 e 10.0.2.0
- Para chegar à rede 10.0.3.0, o tráfego é encaminhado por 10.0.2.2
- Para chegar à rede 10.0.5.0, o tráfego é encaminhado por 10.0.1.2
- Para chegar à rede 10.0.4.0, o tráfego tanto pode ir por 10.0.1.2 como por 10.0.2.2, isto porque ambos os caminhos têm a mesma largura de banda na sua totalidade.
- Para alcançar a rede 10.0.6.0, o tráfego segue pelo caminho com a métrica mais baixa, sendo este feito por 10.0.2.2

Concluindo, no protocolo OSPF existem alguns fatores a considerar aquando da escolha do encaminhamento do tráfego, porém neste caso foram considerados apenas os custos dos links.



```

Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, o - OSPF6, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

O 10.0.0.0/24 [110/10] is directly connected, eth0, 00:01:26
C>* 10.0.0.0/24 is directly connected, eth0
O 10.0.1.0/24 [110/10] is directly connected, eth1, 00:01:26
C>* 10.0.1.0/24 is directly connected, eth1
O 10.0.2.0/24 [110/1] is directly connected, eth2, 00:01:26
C>* 10.0.2.0/24 is directly connected, eth2
O>* 10.0.3.0/24 [110/11] via 10.0.2.2, eth2, 00:00:36
O>* 10.0.4.0/24 [110/21] via 10.0.2.2, eth2, 00:00:36
    * via 10.0.1.2, eth1, 00:00:36
O>* 10.0.5.0/24 [110/11] via 10.0.1.2, eth1, 00:00:36
O>* 10.0.6.0/24 [110/21] via 10.0.2.2, eth2, 00:00:36
C>* 127.0.0.0/8 is directly connected, lo
n3#

```

Figure 28: Tabela routing OSPF do router n3

## 15 Pergunta 15

O comando que decidimos utilizar é o `sh ip route`.

```

n4# sh ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, o - OSPF6, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

O>* 10.0.0.0/24 [110/20] via 10.0.1.1, eth0, 00:00:20
O 10.0.1.0/24 [110/10] is directly connected, eth0, 00:01:10
C>* 10.0.1.0/24 is directly connected, eth0
O>* 10.0.2.0/24 [110/11] via 10.0.1.1, eth0, 00:00:20
O>* 10.0.3.0/24 [110/21] via 10.0.5.2, eth1, 00:00:20
    * via 10.0.1.1, eth0, 00:00:20
O>* 10.0.4.0/24 [110/11] via 10.0.5.2, eth1, 00:00:25
O 10.0.5.0/24 [110/1] is directly connected, eth1, 00:00:25
C>* 10.0.5.0/24 is directly connected, eth1
O>* 10.0.6.0/24 [110/21] via 10.0.5.2, eth1, 00:00:20
C>* 127.0.0.0/8 is directly connected, lo

```

Figure 29: Tabela routing OSPF do router n4

Os custos, em OSPF, são atribuídos através da divisão da bandwidth de referência (100Mbps) pela da bandwidth da ligação, sendo que, quanto maior for a bandwidth da ligação menor é o custo. Como todos os links da nossa topologia têm 100Mbps ou 10Mbps de bandwidth os custos desses links irão ser  $\frac{100Mbps}{100Mbps} = 1$  e  $\frac{100Mbps}{10Mbps} = 10$  respectivamente.

## 16 Pergunta 16

Foram então alterados os custos dos links, 1 e 10 para ligações de 10Mbps e 100Mbps respetivamente. Para tal recorreu-se ao comando **ip ospf cost valorCusto**, sendo a variável valorCusto o valor 1 ou 10, dependendo da largura de banda da ligação. Este comando teve que ser introduzido em cada router, para as suas ligações adjacentes.

```
n3# configure terminal
n3(config)# interface eth2
n3(config-if)# ip ospf cost 10
n3(config-if)#
```

Figure 30: Definição do custo de uma ligação de 10Mbps para 10

```
n3# configure terminal
n3(config)# interface eth1
n3(config-if)# ip ospf cost 1
n3(config-if)#
```

Figure 31: Definição do custo de uma ligação de 100Mbps para 1

## 17 Pergunta 17

Tendo em conta que o OSPF define o seu encaminhamento de tráfego consoante as métricas estabelecidas pelos links, é expectável que os caminhos escolhidos mudem, visto que agora os links de 100Mbps têm prioridade sobre os de 10Mbps (medida standard em grande parte dos routers, mas não no CORE). É possível notar esta diferença no nosso router n1 para chegar à rede 10.0.5.0, que inicialmente escolhia ir por 10.0.2.1 mas agora é encaminhado por 10.0.3.2. Isto acontece porque agora o custo total dos links é mais baixo se for encaminhado por 10.0.3.2 (custo total de 12), enquanto que pelo caminho antigo teria um custo de 21.

As imagens seguintes demonstram a diferença no encaminhamento feito a partir do n1 até à rede 10.0.5.0, antes e depois da alteração dos custos.

```

n1# sh ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, o - OSPF6, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

O>* 10.0.0.0/24 [110/11] via 10.0.2.1, eth0, 00:01:02
O>* 10.0.1.0/24 [110/11] via 10.0.2.1, eth0, 00:01:02
O  10.0.2.0/24 [110/1] is directly connected, eth0, 00:01:52
C>* 10.0.2.0/24 is directly connected, eth0
O  10.0.3.0/24 [110/10] is directly connected, eth1, 00:01:52
C>* 10.0.3.0/24 is directly connected, eth1
O>* 10.0.4.0/24 [110/20] via 10.0.3.2, eth1, 00:01:02
O>* 10.0.5.0/24 [110/12] via 10.0.2.1, eth0, 00:00:52
O>* 10.0.6.0/24 [110/20] via 10.0.3.2, eth1, 00:01:02
C>* 127.0.0.0/8 is directly connected, lo
n1#

```

Figure 32: Encaminhamento feito por 10.0.2.1, antes da alteração

```

n1# sh ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, o - OSPF6, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

O>* 10.0.0.0/24 [110/20] via 10.0.2.1, eth0, 06:11:59
O>* 10.0.1.0/24 [110/11] via 10.0.2.1, eth0, 06:11:59
O  10.0.2.0/24 [110/10] is directly connected, eth0, 06:16:04
C>* 10.0.2.0/24 is directly connected, eth0
O  10.0.3.0/24 [110/1] is directly connected, eth1, 06:15:52
C>* 10.0.3.0/24 is directly connected, eth1
O>* 10.0.4.0/24 [110/2] via 10.0.3.2, eth1, 06:13:15
O>* 10.0.5.0/24 [110/12] via 10.0.3.2, eth1, 06:11:59
O>* 10.0.6.0/24 [110/11] via 10.0.3.2, eth1, 06:15:52
C>* 127.0.0.0/8 is directly connected, lo
n1#

```

Figure 33: Encaminhamento feito por 10.0.3.2, depois da alteração

## 18 Pergunta 18

Se a bandwidth de referência for 100Mbps todas as ligações com 100Mbps ou mais de bandwidth irão ficar com custo igual a 1, logo a bandwidth de referência tem de ser maior ou igual á melhor ligação existente na topologia.

Para além disso todos os routers têm que ter a mesma bandwidth de referência e routers vizinhos têm que definir a mesma bandwidth para o link que os liga. De salientar que é melhor dar prioridade a uma ligação com maior banda larga por ter maior capacidade de transmissão mas também é necessário manter um certo equilíbrio para não sobrecarregar a rede ao encaminhar o tráfego todo por uma certa ligação.

## 19 Pergunta 19

```
n4# sh ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, o - OSPF6, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

O>* 10.0.0.0/24 [110/20] via 10.0.1.1, eth0, 00:00:20
O  10.0.1.0/24 [110/10] is directly connected, eth0, 00:01:10
C>* 10.0.1.0/24 is directly connected, eth0
O>* 10.0.2.0/24 [110/11] via 10.0.1.1, eth0, 00:00:20
O>* 10.0.3.0/24 [110/21] via 10.0.5.2, eth1, 00:00:20
   *                via 10.0.1.1, eth0, 00:00:20
O>* 10.0.4.0/24 [110/11] via 10.0.5.2, eth1, 00:00:25
O  10.0.5.0/24 [110/1] is directly connected, eth1, 00:00:25
C>* 10.0.5.0/24 is directly connected, eth1
O>* 10.0.6.0/24 [110/21] via 10.0.5.2, eth1, 00:00:20
C>* 127.0.0.0/8 is directly connected, lo
```

Figure 34: Tabela routing OSPF do router n4

Como podemos observar através da tabela de routing de n4 há duas rotas **n4->rede 10.0.3.0**.

- **n4,n3,n1** com custo 21
- **n4,n5,n2** com custo 21

Através do comando **sh ip ospf** nos routers n5 e n3 conseguimos ver os IDs dos dois routers.

```
n5# sh ip ospf
OSPF Routing Process, Router ID: 10.0.4.1
```

Figure 35: ID router n5

```
n3# sh ip ospf
OSPF Routing Process, Router ID: 10.0.0.1
```

Figure 36: ID router n3

Como o tie-breaker, no caso do OSPF, é o maior router ID podemos afirmar que a rota escolhida seria a que passa por **n5**.

```
root@n4:/tmp/pycore.5109/n4.conf# tracepath 10.0.3.1
1: 10.0.5.1                                0.131ms pmtu 1500
1: 10.0.5.2                                0.102ms
1: 10.0.5.2                                0.084ms
2: 10.0.4.2                                0.196ms
3: 10.0.3.1                                0.196ms reached
Resume: pmtu 1500 hops 3 back 63
```

Figure 37: Tracepath n4 para 10.0.3.1

A desvantagem de utilizar este "tie-breaker" é que a rota escolhida vai ser sempre a mesma, independentemente do quão sobrecarregada está.

Outra opção seria activar **ECMP**, Equal Cost Multi-Path, que permite equilibrar a carga através da divisão de pacotes entre as várias rotas. A desvantagem desta opção é que o receptor pode receber pacotes fora de ordem, que pode causar problemas de performance.

Em **OSPFv2** essa função é activada através do comando **maximum-paths x** onde **x** é o nº máximo de rotas a utilizar.

## 20 Pergunta 20

Nesse caso as rotas OSPF iam ser escolhidas visto que a distância administrativa do protocolo OSPF (110) é menor que a distância administrativa do protocolo RIP (120), logo todas as tabelas de routing iriam ser as criadas por OSPF.

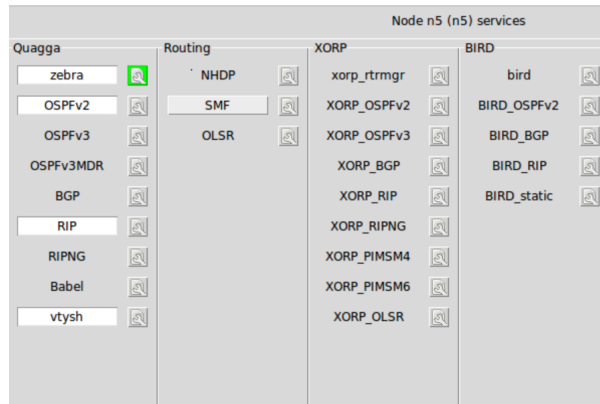


Figure 38: Serviços activados router n5

```
n5# sh ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, o - OSPF6, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

O>* 10.0.0.0/24 [110/21] via 10.0.5.1, eth1, 00:00:49
O>* 10.0.1.0/24 [110/11] via 10.0.5.1, eth1, 00:01:04
O>* 10.0.2.0/24 [110/12] via 10.0.5.1, eth1, 00:00:49
O>* 10.0.3.0/24 [110/20] via 10.0.4.2, eth0, 00:00:54
O  10.0.4.0/24 [110/10] is directly connected, eth0, 00:01:44
C>* 10.0.4.0/24 is directly connected, eth0
O  10.0.5.0/24 [110/1] is directly connected, eth1, 00:01:44
C>* 10.0.5.0/24 is directly connected, eth1
O>* 10.0.6.0/24 [110/20] via 10.0.4.2, eth0, 00:00:54
C>* 127.0.0.0/8 is directly connected, lo
```

Figure 39: Tabela routing OSPF/RIP do router n5

Como podemos observar, todas as rotas escolhidas foram as rotas de OSPF.