

UMinho

Mestrado em Engenharia Informática
Interligação de Redes IP (2022/2023)

GRUPO 4

Simão Cunha (a93262)
Gonçalo Pereira (a93168)
Rui Alves (pg50745)

Braga, 28 de março de 2023

Conteúdo

1	Exercício 1	2
	1.1 Resolução	2
2	Exercício 2	3
	2.1 Resolução	3
3	Exercício 3	4
	3.1 Resolução	4
4	Exercício 4	5
	4.1 Resolução	5
5	Exercício 5	6
	5.1 Resolução	6
	5.2 Resolução	8
6	Exercício 6	9
	6.1 Resolução	9
7	Exercício 7	10
	7.1 Resolução alínea (i)	10
	7.2 Resolução alínea (ii)	12
8	Exercício 8	14
	8.1 Resolução	14
9	Exercício 9	16
	9.1 Resolução	16
10	Exercício 10	18
	10.1 Resolução	18

1 Exercício 1

Definir uma topologia de uma rede de interligação envolvendo vários routers, os respectivos links, e redes cliente. A topologia criada deverá incluir pelo menos duas situações distintas de redundância: i) na perspectiva de pelo menos um dos routers existem pelo menos dois caminhos alternativos com o mesmo custo mínimo para atingir uma determinada rede e ii) na perspectiva de pelo menos um dos routers existem pelo menos dois caminhos de custo diferente para atingir uma ou mais redes.

1.1 Resolução

Seguindo o enunciado, elaboramos a seguinte topologia no emulador CORE:

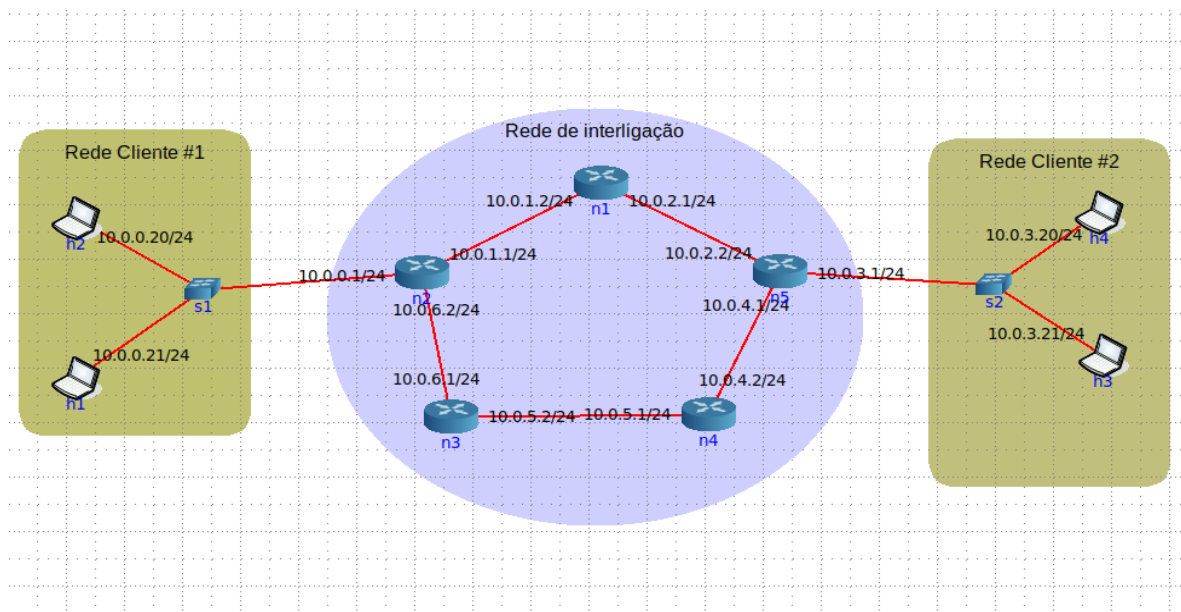


Figura 1: Topologia criada

Neste caso, de forma a respeitar o requisito i), podemos verificar que o *router* n3 tem, para atingir a *rede cliente* #1, pelo menos dois caminhos alternativos para o mesmo custo mínimo e que o *router* n2, para atingir a *rede cliente* #2, tem pelo menos dois caminhos de custo diferente.

2 Exercício 2

Analise/comente a configuração dos endereços (IPv4) das interfaces de rede dos routers/hosts que foi efetuada pelo CORE.

2.1 Resolução

Observando a topologia criada, é possível observar alguns factos:

- A máscara destes endereços é /24;
- Existe um endereço único atribuído a cada interface de cada router/host pelo CORE;
- Cada segmento/ligação é uma sub-rede;
- Estes endereços IP são usados para encaminhar pacotes de dados ao nível 3 da pilha OSI. Caso estejam mal configurados (por exemplo, existirem interfaces com endereços iguais), o *routing* irá funcionar de forma incorreta;

3 Exercício 3

Configure, na interface gráfica, todos os routers da topologia para usarem o protocolo RIP para difusão de todas as redes de interligação e redes clientes.

3.1 Resolução

De forma a configurarmos os routers da topologia de forma a utilizarem o protocolo RIP para a difusão de todas as redes de interligação e redes clientes, temos de seguir alguns passos:

1. *Right click* num certo router;
2. Selecionar *Services*;
3. Ativar as opções consoante a figura 2;
4. Aplicar os três passos acima em todos os routers.

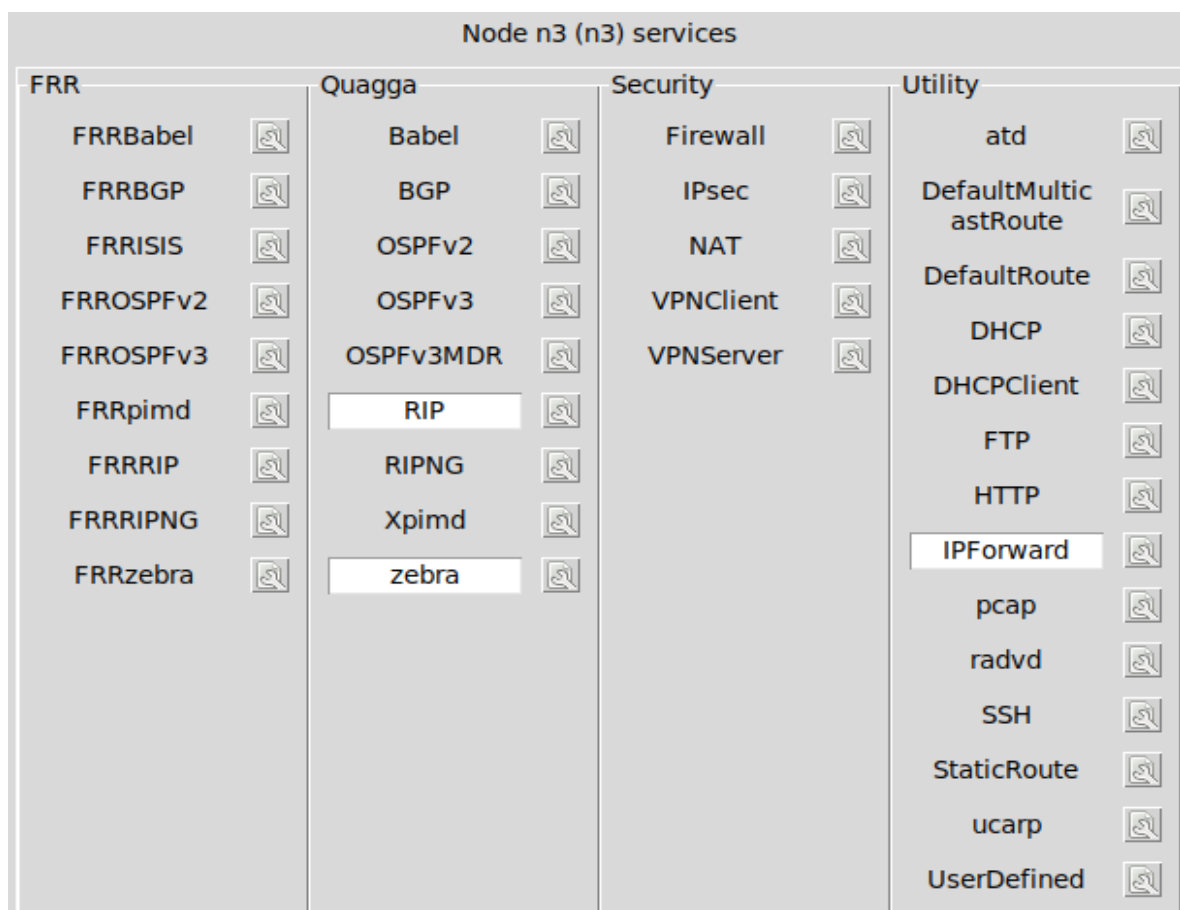


Figura 2: Configuração de um router da topologia através da interface gráfica

4 Exercício 4

Ative a topologia e proceda a testes de conectividade (e.g. ping, etc.) entre os diversos equipamentos. Verifique também as rotas utilizadas pelo tráfego que circula na rede.

4.1 Resolução

Para respondermos a esta alínea, mostraremos os testes de conectividade efetuados fazendo uso de *traceroute* entre diferentes nodos da topologia.

```
root@h1:/tmp/pycore.45789/h1.conf# traceroute 10.0.3.21
traceroute to 10.0.3.21 (10.0.3.21), 30 hops max, 60 byte packets
 1  10.0.0.1 (10.0.0.1)  0.076 ms  0.018 ms  0.010 ms
 2  10.0.1.2 (10.0.1.2)  0.025 ms  0.015 ms  0.014 ms
 3  10.0.2.2 (10.0.2.2)  0.028 ms  0.018 ms  0.017 ms
 4  10.0.3.21 (10.0.3.21) 0.034 ms  0.023 ms  0.022 ms
root@h1:/tmp/pycore.45789/h1.conf#
```

Figura 3: Traceroute host $h1 \rightarrow h3$

Observando a figura 3, podemos observar que existe conectividade e a rota dos pacotes entre $h1$ e $h3$ consiste em $h1 \rightarrow n2 \rightarrow n1 \rightarrow n5 \rightarrow h3$. A escolha desta rota justifica-se pelo facto da métrica usada pelo protocolo *RIP* ser o número de saltos.

Na figura 4, podemos observar que também existe conectividade e que existe uma rota entre $h2$ e $h4$.

```
root@h2:/tmp/pycore.41019/h2.conf# traceroute 10.0.3.20
traceroute to 10.0.3.20 (10.0.3.20), 30 hops max, 60 byte packets
 1  10.0.0.1 (10.0.0.1)  0.076 ms  0.004 ms  0.004 ms
 2  10.0.1.2 (10.0.1.2)  0.014 ms  0.005 ms  0.005 ms
 3  10.0.2.2 (10.0.2.2)  0.016 ms  0.008 ms  0.006 ms
 4  10.0.3.20 (10.0.3.20) 0.020 ms  0.009 ms  0.009 ms
root@h2:/tmp/pycore.41019/h2.conf#
```

Figura 4: Traceroute host $h2 \rightarrow h4$

Além disso, também é possível observar que existe conectividade entre $n3$ e $n5$.

```
root@n3:/tmp/pycore.36157/n3.conf# traceroute 10.0.4.1
traceroute to 10.0.4.1 (10.0.4.1), 30 hops max, 60 byte packets
 1  10.0.5.1 (10.0.5.1)  0.029 ms  0.005 ms  0.004 ms
 2  10.0.4.1 (10.0.4.1)  0.016 ms  0.006 ms  0.005 ms
```

Figura 5: Traceroute router $n3 \rightarrow n5$

5 Exercício 5

Visualize e explique as tabelas de routing que foram estabelecidas pelos routers da rede de interligação. [nota: selecione e explique em detalhe uma tabela de routing específica que considere relevante para esse efeito]

5.1 Resolução

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

n2# show 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, 00:05:15
R>* 10.0.3.0/24 [120/3] via 10.0.1.2, eth1, 00:05:09
R>* 10.0.4.0/24 [120/3] via 10.0.6.1, eth2, 00:05:14
R>* 10.0.5.0/24 [120/2] via 10.0.6.1, eth2, 00:05:14
C>* 10.0.6.0/24 is directly connected, eth2
C>* 127.0.0.0/8 is directly connected, lo
n2#
```

Figura 6: Tabela de routing do router *n2*

Escolhemos analisar a tabela de routing do router *n2* por se tratar de um router de acesso a uma rede cliente.

Em primeiro lugar, na tabela é possível observar as entradas marcadas com **C** que indicam as interfaces a que o router está ligado diretamente. As entradas marcadas com **R** indicam rotas obtidas através do protocolo *RIP*.

Neste caso, a tabela indica que:

- Por 10.0.1.2 atinge-se 10.0.2.0 com um custo de 2. O tempo desde a última atualização do RIP é 00:05:15.
- Por 10.0.1.2 atinge-se 10.0.3.0 com um custo de 3. O tempo desde a última atualização do RIP é 00:05:09.
- Por 10.0.6.1 atinge-se 10.0.4.0 com um custo de 3. O tempo desde a última atualização do RIP é 00:05:14.
- Por 10.0.6.1 atinge-se 10.0.5.0 com um custo de 2. O tempo desde a última atualização do RIP é 00:05:14.

Agora analisando as entradas com **C**, podemos observar que:

- Para a rede Cliente #1 (identificado pela subrede 10.0.0.0/24), o router **n2** encaminha o tráfego através da interface **eth0**;
- Para o nó **n3** (identificado pela subrede 10.0.1.0/24), o router **n2** encaminha o tráfego através da interface **eth1**;
- Para o nó **n1** (identificado pela subrede 10.0.6.0/24), o router **n2** encaminha o tráfego através da interface **eth2**;
- A última entrada refere-se ao endereço de *loopback*.

Por se tratar do protocolo *RIP* entende-se por custo o número de saltos, já que essa é a métrica usada. Além disso, é possível verificar que o custo administrativo deste protocolo é 120. O custo administrativo só é relevante quando num router existe mais do que um protocolo de encaminhamento, servindo como fator de desempate na escolha de uma rota.

Visualize e explique as tabelas de routing dos hosts das redes clientes. [nota: explique em detalhe uma tabela de routing específica de um dos hosts]

5.2 Resolução

Escolhemos analisar a tabela de routing do host *h1*, obtida através do comando **netstat -nr**. Como é possível verificar na imagem abaixo, o host só tem 2 rotas estabelecidas na sua tabela. A primeira entrada representa a rota que estabelece para onde os pacotes são encaminhados quando o próximo salto não é conhecido. A segunda entrada representa uma rota via *n2*.

```
root@h1:/tmp/pycore.34765/h1.conf# netstat -nr
Kernel IP routing table
Destination    Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         10.0.0.1        0.0.0.0         UG      0 0        0 eth0
10.0.0.0        0.0.0.0         255.255.255.0   U       0 0        0 eth0
root@h1:/tmp/pycore.34765/h1.conf#
```

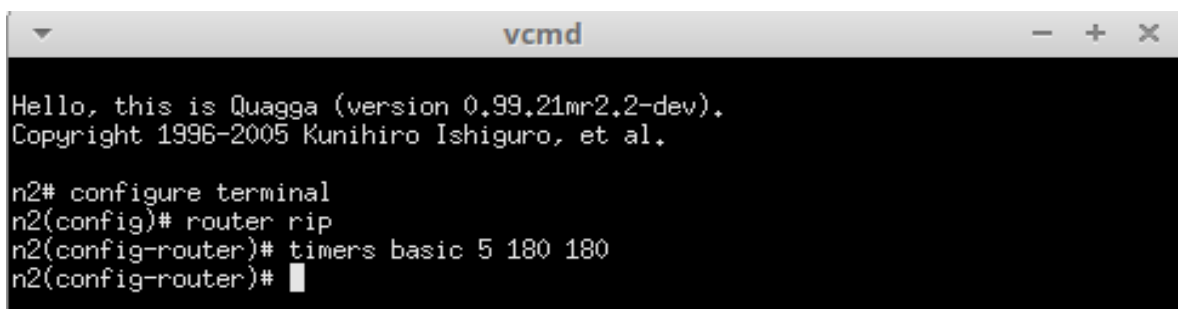
Figura 7: Tabela de routing do host *h1*

6 Exercício 6

Suponha que pretendia alterar o intervalo de tempo segundo o qual são gerados os updates enviados pelo protocolo RIP. Que comando utilizaria para o efeito?

6.1 Resolução

De forma a alterar o intervalo de tempo segundo os quais são gerados os updates enviados pelo protocolo RIP, consultamos a documentação [1] e fizemos uso do comando `timers basic` no router `n2` fixando o tempo de update nos 5 segundos. O comando além do tempo de update recebe mais 2 argumentos (os 2º, 3º argumentos são, respetivamente, *invalid*, *holddown* e *flush*), aos quais atribuímos os tempos *default*, tal como se observa na figura 8.



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

n2# configure terminal
n2(config)# router rip
n2(config-router)# timers basic 5 180 180
n2(config-router)#
```

Figura 8: Execução do comando `timers basic`

De forma a verificar se o novo valor do intervalo de tempo dos updates estava a ser cumprido, abrimos o Wireshark no router `n1` e verificamos que o `n2` estava de facto a enviar updates periódicos de 5 em 5 segundos.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.0.1.2	224.0.0.9	RIPv2	126	Response
2	0.253269702	10.0.1.1	224.0.0.9	RIPv2	126	Response
3	5.257855741	10.0.1.1	224.0.0.9	RIPv2	126	Response
4	10.261342432	10.0.1.1	224.0.0.9	RIPv2	126	Response
5	15.267253592	10.0.1.1	224.0.0.9	RIPv2	126	Response
6	20.273351945	10.0.1.1	224.0.0.9	RIPv2	126	Response

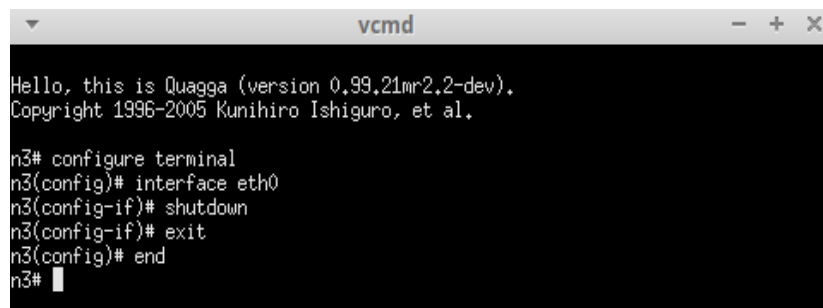
Figura 9: Captura no Wireshark no router `n1`

7 Exercício 7

Através da consola de configuração (vtysh) dos routers desative um (ou mais) links/interfaces de interligação por forma a originar duas situações distintas: i) apesar da(s) interface(s) desativadas todos os routers/redes da topologia podem manter conectividade; ii) alguns dos routers/redes ficam sem caminhos disponíveis para atingir outros routers/redes da topologia. Para cada uma das situações explique o que observou em relação às alterações das tabelas de routing dos equipamentos e ao tempo de propagação da informação de routing. [nota: para cada caso deverá ser também selecionada e explicada em detalhe uma tabela de routing específica que considere relevante para esse efeito]

7.1 Resolução alínea (i)

De modo a desativar a eth0 do router *n3* executamos o seguinte comando `shutdown` [4]:



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

n3# configure terminal
n3(config)# interface eth0
n3(config-if)# shutdown
n3(config-if)# exit
n3(config)# end
n3#
```

Figura 10: Execução do comando `shutdown`

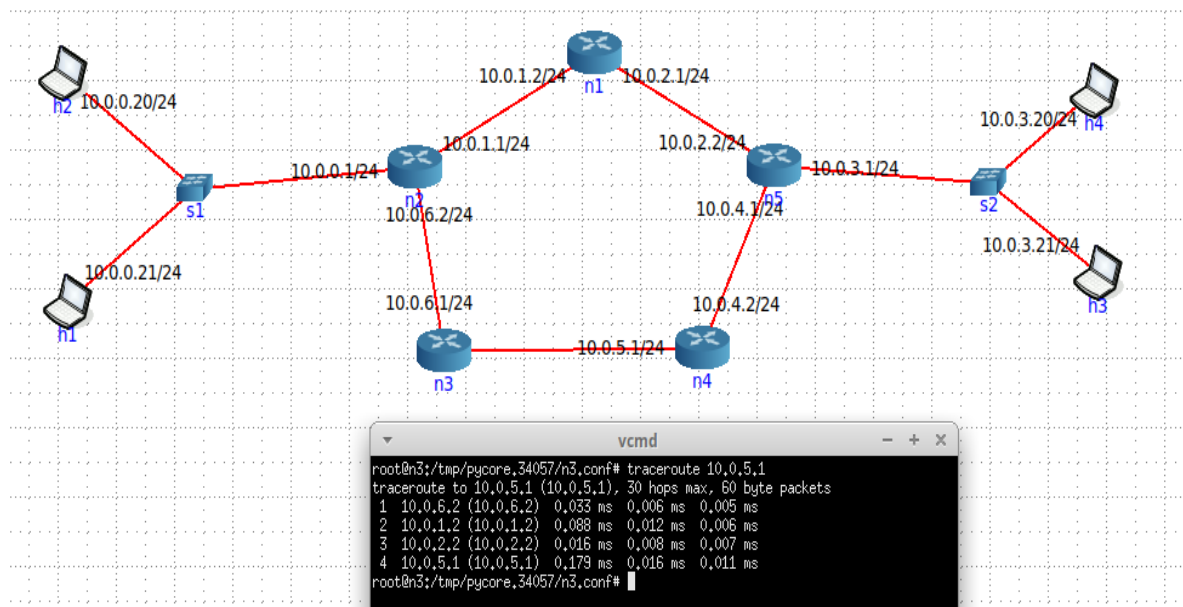


Figura 11: Topologia modificada e execução do comando `traceroute` *n3* → *n5*

Na figura 11, efetuamos o comando `traceroute` para verificar a rota dos pacotes *n3* para *n4*

e verificamos que a rota passa por $n3 \rightarrow n2 \rightarrow n1 \rightarrow n5 \rightarrow n4$.

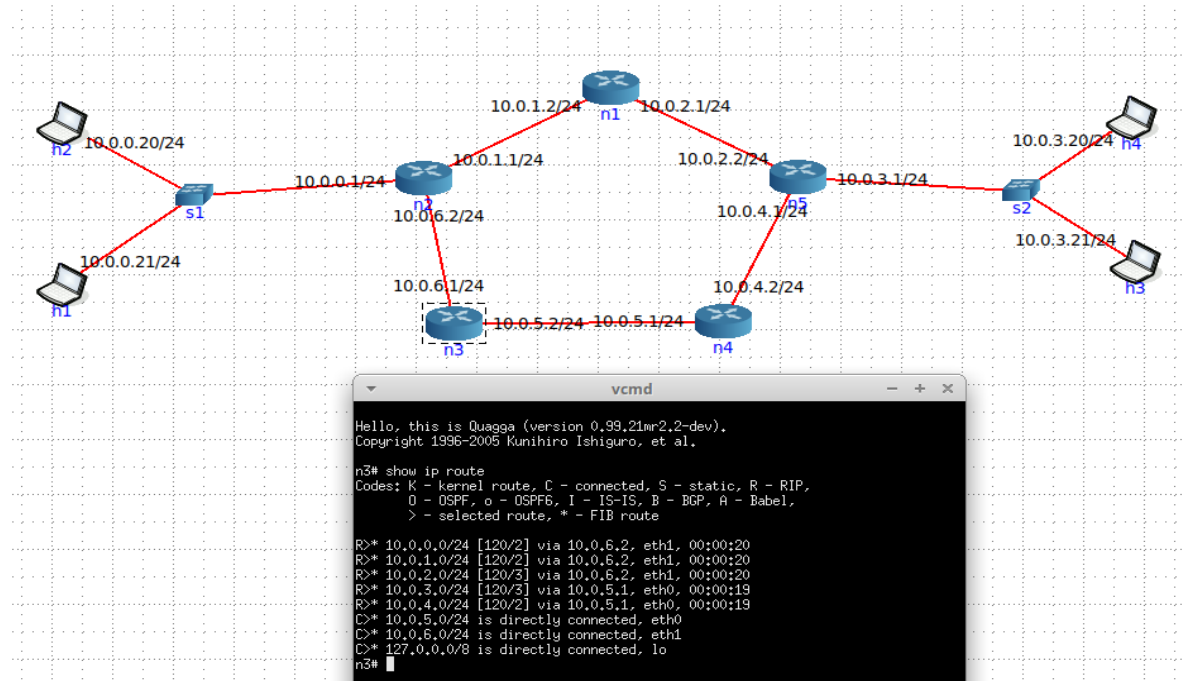


Figura 12: Tabela de routing do router $n3$ antes da remoção da interface

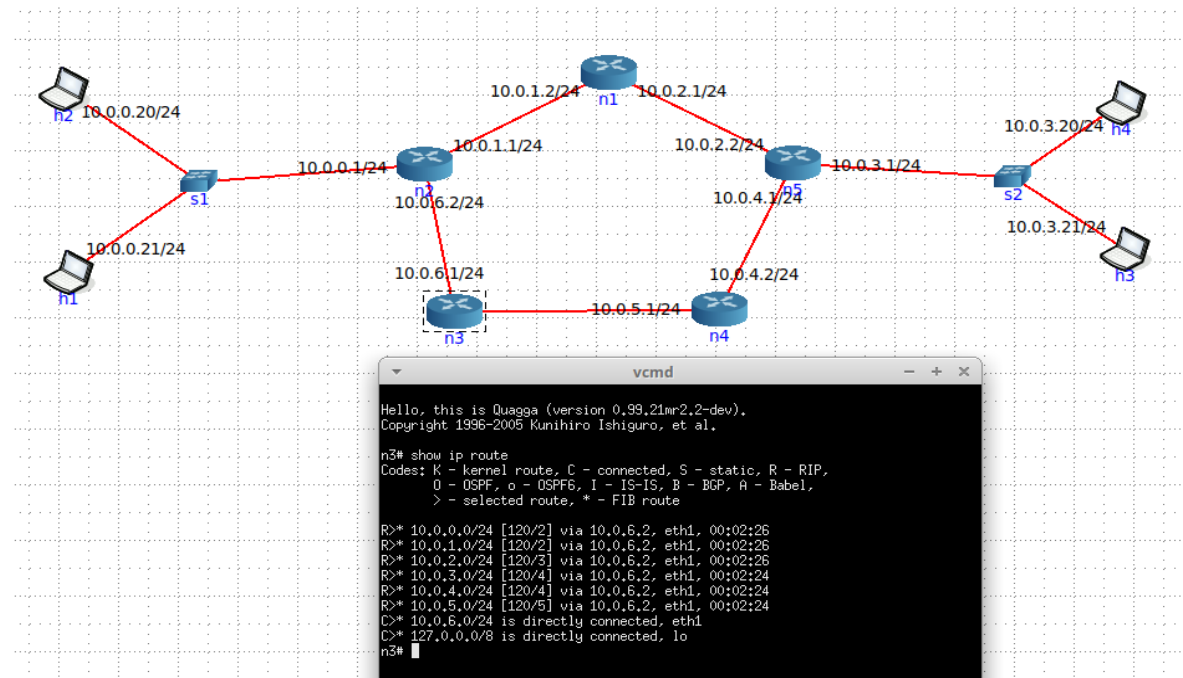


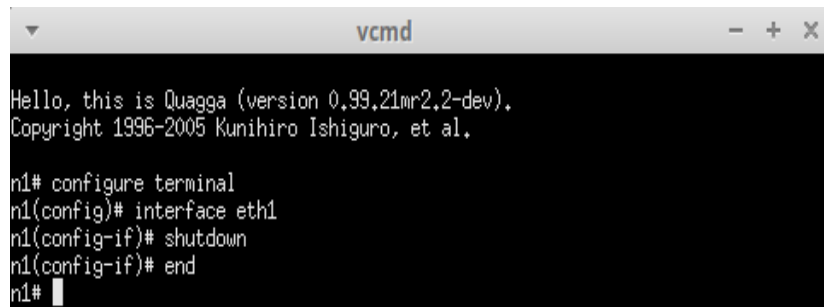
Figura 13: Tabela de routing do router $n3$ depois da remoção da interface

Após eliminarmos uma das interfaces de $n3$ consultamos a sua tabela de routing. Verifi-

camos que, de facto, a interface tinha sido eliminada pois a entrada do tipo **C** só indicava ligação à interface 1 (**eth1**). Além disso, analisando as entradas do tipo **R** verificamos que já não existia uma rota direta entre $n3$ e $n4$. Esse facto é também confirmado pelo `traceroute` executado acima que demonstra que o caminho mais longo foi usado já que era o único disponível.

7.2 Resolução alínea (ii)

Na figura 14, decidimos desativar a interface que liga o router $n1$ ao router $n5$ - (juntamente com a desativação feita na alínea i), cumprindo o requisito do enunciado da alínea



```

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

n1# configure terminal
n1(config)# interface eth1
n1(config-if)# shutdown
n1(config-if)# end
n1#

```

Figura 14: Execução do comando `shutdown`

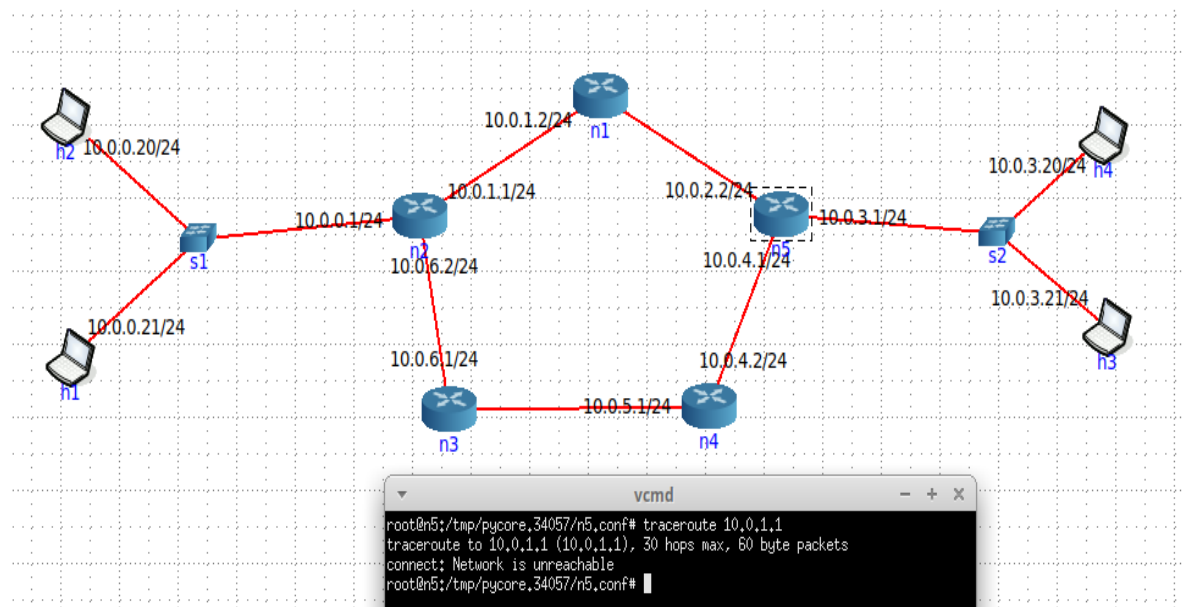


Figura 15: Topologia modificado e `traceroute` $n5 \rightarrow n2$

Efetuamos o comando `traceroute` para verificar a rota dos pacotes $n1$ para $n5$ e verificamos que já não existe conetividade.

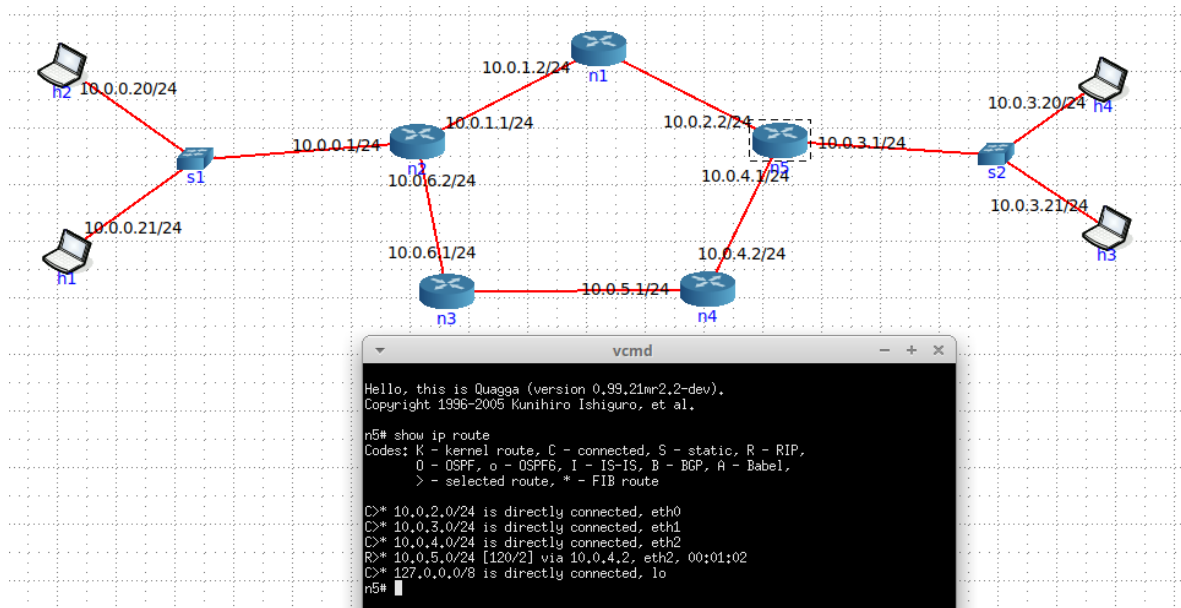


Figura 16: Tabela de routing do router *n5*

Na tabela de routing do router *n5* é possível observar que este mantém as interfaces a que está conectado, entradas do tipo **C**, visto que não removemos nenhuma interface desse router. Por outro lado, é possível verificar que só existe uma rota, sendo esta a rota para *n4*. A eliminação das interfaces de *n3* e *n1* fez com que o estabelecimento de rotas não fosse possível nem por *n4* nem por *n5* levando à perda de conectividade entre duas partes da topologia.

8 Exercício 8

Configure um dos routers da sua topologia por forma a que o tráfego que ele transmite para uma determinada rede destino não passe pelo caminho com um menor numero de saltos, mas sim por um outro caminho alternativo (ou seja, no exemplo da Figura 1, o router y para enviar tráfego destinado à rede B passaria a usar o caminho que passa pelo router x. Para todos os restantes destinos o caminho com o menor número de saltos continuaria a ser o escolhido). [sugestão: nas aulas teóricas foi mencionado um comando específico que permite resolver esta questão]

8.1 Resolução

Para responder a esta alínea, iremos focar nos routers *n2* e *n5*. Na imagem abaixo, é possível observar o traceroute que foi efetuado de *n2* para *n5* bem como a tabela de routing de *n2*. Como é possível verificar, a rota tomada foi *n2* → *n1* → *n5*, tal como seria de esperar já que é a rota que exige um menor número de saltos explícita na tabela para alcançar *n5*.

```
root@n2:/tmp/pycore.43143/n2.conf# traceroute 10.0.3.1
traceroute to 10.0.3.1 (10.0.3.1), 30 hops max, 60 byte packets
 1 10.0.1.2 (10.0.1.2) 0.042 ms 0.011 ms 0.009 ms
 2 10.0.3.1 (10.0.3.1) 0.026 ms 0.014 ms 0.013 ms
root@n2:/tmp/pycore.43143/n2.conf#

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

n2# show 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, 00:00:29
R>* 10.0.3.0/24 [120/3] via 10.0.1.2, eth1, 00:00:29
R>* 10.0.4.0/24 [120/3] via 10.0.6.1, eth2, 00:00:29
R>* 10.0.5.0/24 [120/2] via 10.0.6.1, eth2, 00:00:29
C>* 10.0.6.0/24 is directly connected, eth2
C>* 127.0.0.0/8 is directly connected, lo
n2#
```

Figura 17: Pre offset-list: Traceroute *n2* → *n5* e tabela de routing de *n2*

Como o objetivo é obrigar a que seja usada uma rota alternativa, fizemos uso do comando **offset-list 1 in 3 th1** [2] no router *n2*. Este comando define no router que para todos os updates que cheguem ao mesmo, pela *eth1* (10.0.1.1) provenientes de endereços contidos na accesslist 1 soma-se 3 ao valor anunciado. Para definir a access-list, fizemos *access-list 1 permit 10.0.3.0 0.0.0.255*, identificado a access-list 1 como sendo endereços da subrede 10.0.3.0 explicitando a wildcard mask 0.0.0.255. Na figura abaixo, está demonstrada a execução desses mesmos comandos.

```

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

n2# configure terminal
n2(config)# router rip
n2(config-router)# offset-list 1 in 3 eth1
n2(config-router)# exit
n2(config)# access-list 1 permit 10.0.3.0 0.0.0.255
n2(config)# █

```

Figura 18: Offsetlist

Por fim, para garantir que os comandos executados acima tiveram sucesso, voltamos a realizar um traceroute entre *n2* e *n5* e verificamos que agora a rota escolhida, desta vez, foi $n2 \rightarrow n3 \rightarrow n4 \rightarrow n5$. Observando a tabela de routing de *n2* nota-se que a rota com destino em 10.0.3.0 é via 10.0.6.1 (*n3*) com custo de 4. Esta mudança deve-se aos comandos executados anteriormente que fizeram com que a outra rota tivesse um maior custo que a rota atual levando a uma atualização da tabela de rotas para a rota de menor custo.

```

root@n2:/tmp/pycore.43149/n2.conf# traceroute 10.0.3.1
traceroute to 10.0.3.1 (10.0.3.1), 30 hops max, 60 byte packets
 1 10.0.6.1 (10.0.6.1)  0.044 ms  0.012 ms  0.009 ms
 2 10.0.5.1 (10.0.5.1)  0.023 ms  0.014 ms  0.013 ms
 3 10.0.3.1 (10.0.3.1)  0.036 ms  0.018 ms  0.017 ms
root@n2:/tmp/pycore.43149/n2.conf# █

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

n2# show 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, 00:04:40
R>* 10.0.3.0/24 [120/4] via 10.0.6.1, eth2, 00:00:34
R>* 10.0.4.0/24 [120/3] via 10.0.6.1, eth2, 00:04:40
R>* 10.0.5.0/24 [120/2] via 10.0.6.1, eth2, 00:04:40
C>* 10.0.6.0/24 is directly connected, eth2
C>* 127.0.0.0/8 is directly connected, lo
n2# █

```

Figura 19: Pos offset-list: Traceroute $n2 \rightarrow n5$ e tabela de routing de *n2*

9 Exercício 9

Assuma que a rede de interligação que definiu passará a estar ligada a uma outra rede externa através do router x (ver Figura 2). Apresente e explique o(s) comando(s) que usaria na sua rede de interligação para que todo o tráfego dirigido a redes externas saísse pelo router x .

9.1 Resolução

Para resolvermos esta alínea, atualizamos a nossa topologia inicial adicionando uma rede externa diretamente ligada a $n3$. Essa rede externa usa o OSPF como protocolo de encaminhamento. Na figura abaixo é possível observar a topologia atualizada.

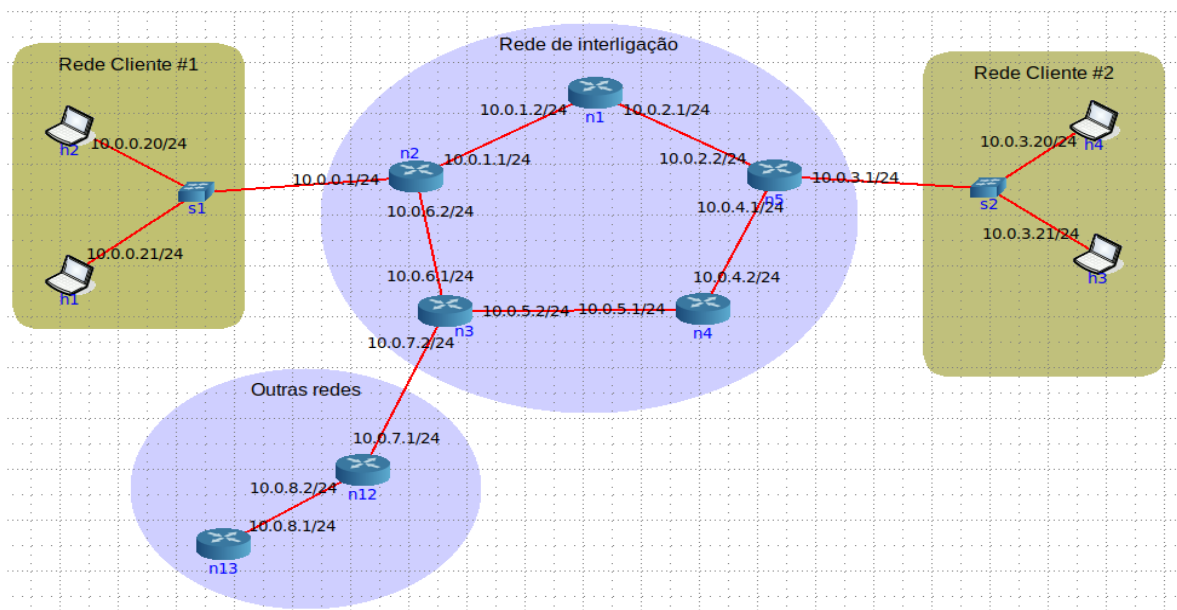


Figura 20: Topologia atualizada

De modo a permitir a rede de interligação ter acesso a redes externas, definimos neste caso o router $n3$ como sendo o default gateway, ou seja, o router para o qual o tráfego é enviado quando tem destinos fora da rede de interligação. Para tal, executamos no router $n3$ o comando **ip route 0.0.0.0 0.0.0.0 10.0.7.1 [3]**, os dois primeiros argumentos estabelecem um destino e máscara default e o terceiro argumento define o próximo salto. Neste caso, o próximo salto é numa interface do router $n12$ que é o router de interligação da rede externa.

```
n3# configure terminal
n3(config)# ip route 0.0.0.0 0.0.0.0 10.0.7.1
n3(config)#
```

Figura 21: Definição de uma default route no $n3$

Após a execução do comando a rota é adicionada na tabela de routing do router *n3* e a partir daí essa rota é propagada via RIP para o resto da rede. Na figura abaixo, é possível verificar que, na tabela de routing do router *n1*, a primeira rota é uma rota default e foi obtida através de RIP.

```
n1# show 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 [120/3] via 10.0.1.1, eth0, 00:08:51
R>* 10.0.0.0/24 [120/2] via 10.0.1.1, eth0, 00:10:51
C>* 10.0.1.0/24 is directly connected, eth0
C>* 10.0.2.0/24 is directly connected, eth1
R>* 10.0.3.0/24 [120/2] via 10.0.2.2, eth1, 00:10:49
R>* 10.0.4.0/24 [120/2] via 10.0.2.2, eth1, 00:10:49
R>* 10.0.5.0/24 [120/3] via 10.0.1.1, eth0, 00:10:50
R>* 10.0.6.0/24 [120/2] via 10.0.1.1, eth0, 00:10:51
R>* 10.0.7.0/24 [120/3] via 10.0.1.1, eth0, 00:10:50
C>* 127.0.0.0/8 is directly connected, lo
n1#
```

Figura 22: Tabela de routing de n1

10 Exercício 10

Após os procedimentos efetuados na questão 9, defina um cenário de rede que lhe permita verificar, e.g. através dos utilitários ping e traceroute, que os routers da topologia que definiu conseguem efetivamente conectividade a endereços externos (i.e. localizados em Outras Redes na Figura 2) através do router x.

10.1 Resolução

Na topologia criada na figura 20, para existir conectividade entre a rede de interligação e o router *n13* na rede externa, foi preciso seguir um processo semelhante realizado no exercício 9. Ou seja, definimos o router *12* como default gateway da rede externa, assegurando assim que também a rede externa tem uma saída para a rede de interligação. De modo a comprovar que existe conectividade entre a rede de interligação e a rede externa, executamos dois comandos traceroute: $h2 \rightarrow n13$ e $h4 \rightarrow n13$.

```
root@h2:/tmp/pycore.41249/h2.conf# traceroute 10.0.8.1
traceroute to 10.0.8.1 (10.0.8.1), 30 hops max, 60 byte packets
 1  10.0.0.1 (10.0.0.1)  0.035 ms  0.009 ms  0.004 ms
 2  10.0.6.1 (10.0.6.1)  0.015 ms  0.005 ms  0.006 ms
 3  10.0.7.1 (10.0.7.1)  0.048 ms  0.008 ms  0.008 ms
 4  10.0.8.1 (10.0.8.1)  0.022 ms  0.011 ms  0.009 ms
```

Figura 23: Traceroute $h2 \rightarrow n13$

```
root@h4:/tmp/pycore.41249/h4.conf# traceroute 10.0.8.1
traceroute to 10.0.8.1 (10.0.8.1), 30 hops max, 60 byte packets
 1  10.0.3.1 (10.0.3.1)  0.053 ms  0.005 ms  0.004 ms
 2  10.0.4.2 (10.0.4.2)  0.015 ms  0.006 ms  0.005 ms
 3  10.0.5.2 (10.0.5.2)  0.015 ms  0.008 ms  0.007 ms
 4  10.0.7.1 (10.0.7.1)  0.017 ms  0.009 ms  0.009 ms
 5  10.0.8.1 (10.0.8.1)  0.018 ms  0.011 ms  0.012 ms
```

Figura 24: Traceroute $h4 \rightarrow n13$

Bibliografia

- [1] Documentação `timers basic`: https://www.cisco.com/E-Learning/bulk/public/tac/cim/cib/using_cisco_ios_software/cmdrefs/timers_basic.htm (consultado em mar. 2023)
- [2] Documentação `offset-list`: <https://networklessons.com/rip/cisco-offset-list-command> (consultado em mar. 2023)
- [3] Documentação `static routes`: <https://www.learncisco.net/courses/icnd-1/ip-routing-technologies/static-routing.html> (consultado em mar. 2023)
- [4] Documentação `shutdown`: https://www.cisco.com/E-Learning/bulk/public/tac/cim/cib/using_cisco_ios_software/cmdrefs/shutdown.htm (consultado em mar. 2023)