Ministério da Educação

Universidade Federal do Piauí – UFPI

Campus Senador Helvídio Nunes de Barros – Picos

Bacharelado Em Sistemas de Informação

Redes de Computadores I

Aluno: Jorge Luis Ferreira Luz

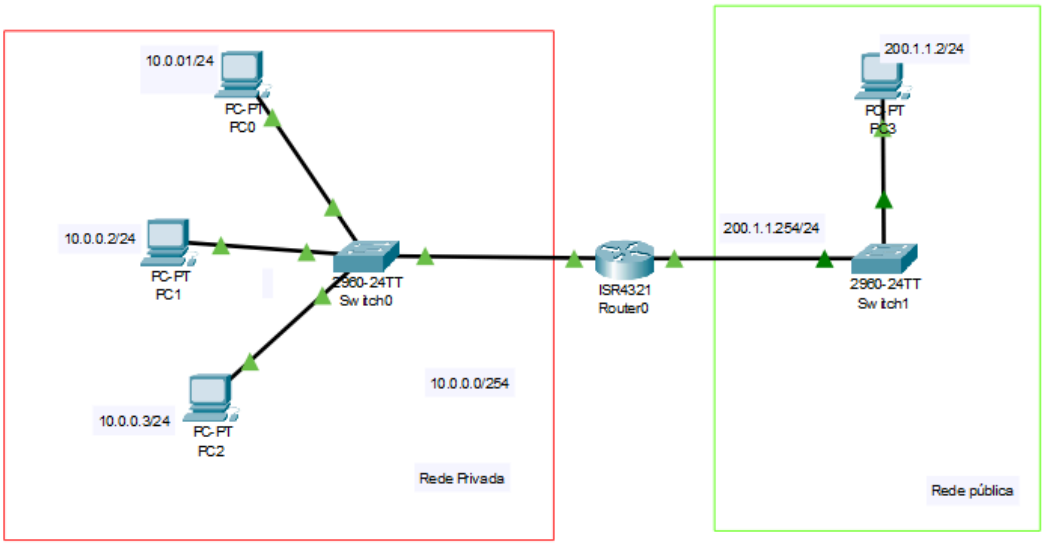
Prof.Rayner Gomes Sousa

1 Questão- Configuração de NAT

Objetivo:

Permitir que os hosts da rede privada 10.0.0.0/24 acessem a rede pública 200.1.1.0/24, especificamente o servidor 200.1.1.2, utilizando NAT configurado no roteador Router0.

Topologia:



Rede Privada: 10.0.0.0/24

- Host PC0: 10.0.0.1
- Host PC2: 10.0.0.2

- Host PC3: 10.0.0.3
- Gateway (Router0 G0/0): 10.0.0.254

Rede Pública: 200.1.1.0/24

- Servidor: 200.1.1.2
- Gateway (Router0 G0/1): 200.1.1.254

Configuração Passo a Passo no Router:

1. Acessar o roteador:
    - Clique no Router0.
    - Vá até a aba CLI.
    - Digite no para desabilitar o modo de configuração automática, se solicitado.
2. Entrar no modo privilegiado e configuração global:
    - `enable`
    - `configure terminal`
3. Configurar a interface G0/0 (rede privada):
    - `interface gigabitEthernet 0/0`
    - `ip address 10.0.0.254 255.255.255.0`
    - `ip nat inside`
    - `no shutdown`
    - `exit`
4. Configurar a interface G0/1 (rede pública):
    - `interface gigabitEthernet 0/1`
    - `ip address 200.1.1.254 255.255.255.0`
    - `ip nat outside`
    - `no shutdown`
    - `exit`
5. Criar uma ACL (Access Control List) para permitir NAT da rede privada:
    - `access-list 1 permit 10.0.0.0 0.0.0.255`
6. Configurar o NAT com sobrecarga (PAT):

- `ip nat inside source list 1 interface gigabitEthernet 0/1 overload`

7. Salvar as configurações
   - `end`
   - `write memory`

Configuração dos PCs

Para cada PC (PC0, PC2, PC3):

1. Clique no PC.
2. Vá até a aba Desktop > IP Configuration.
3. Configure:
   - `IP Address: (ex: 10.0.0.1, 10.0.0.2, 10.0.0.3)`
   - `Subnet Mask: 255.255.255.0`
   - `Default Gateway: 10.0.0.254`

Configuração do Servidor

1. Clique no Server.
2. Vá até Desktop > IP Configuration.
3. Configure:
   - `IP Address: 200.1.1.2`
   - `Subnet Mask: 255.255.255.0`
   - `Default Gateway: 200.1.1.254`

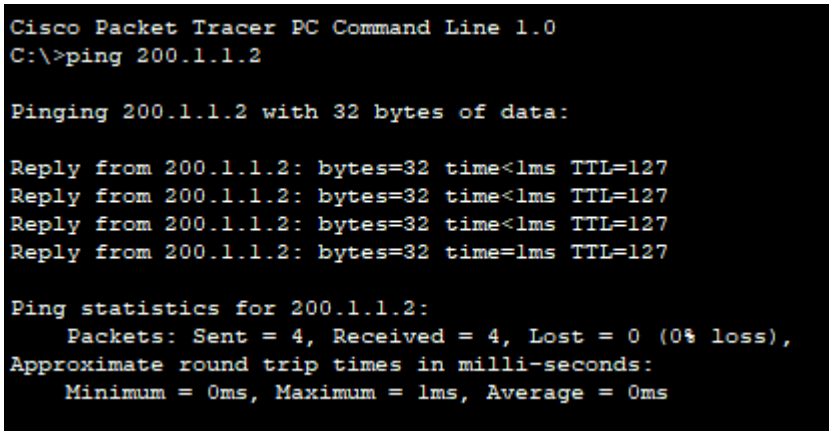
Testes de Conectividade

No PC0, vá em:

- Desktop > Command Prompt

Teste com:

ping 200.1.1.2



- O resultado deve mostrar respostas bem-sucedidas, confirmando que o NAT está funcionando corretamente.

Conclusão

Com o NAT configurado no Router0, os hosts da rede privada podem agora acessar a rede pública. Isso simula uma situação real de acesso à internet a partir de IPs privados, protegendo a rede interna e permitindo comunicação com IPs públicos.

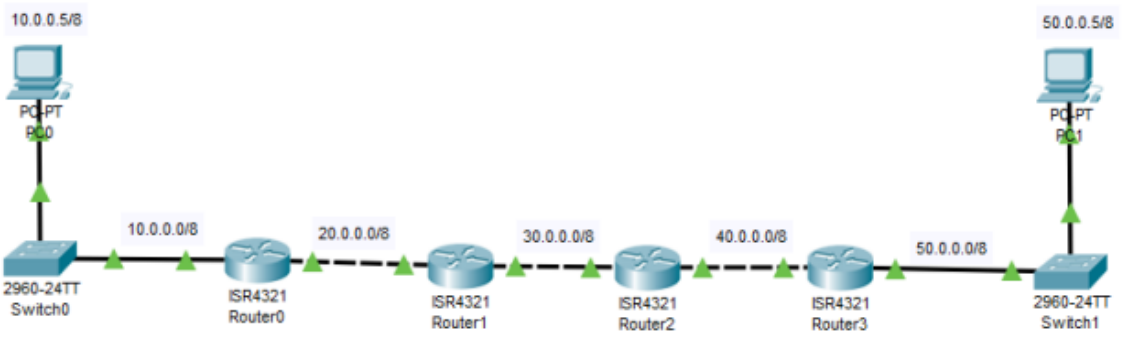
2 Questão- Roteamento Estático

Objetivo:

Permitir que os computadores das redes 10.0.0.0/8 e 50.0.0.0/8 se comuniquem entre si utilizando roteamento estático em quatro roteadores conectados em série.

Topologia

A rede é composta por dois computadores e quatro roteadores conectados entre si por links ponto-a-ponto. Abaixo, as interfaces utilizadas:



Configuração dos Hosts

PC0 (lado esquerdo):

- IP Address: 10.0.0.5
- Subnet Mask: 255.0.0.0
- Default Gateway: 10.0.0.1

PC1 (lado direito):

- IP Address: 50.0.0.5
- Subnet Mask: 255.0.0.0
- Default Gateway: 50.0.0.1

Configuração dos Roteadores

```
Router0

enable

configure terminal

interface g0/0/0

ip address 10.0.0.1 255.0.0.0

no shutdown

exit

interface g0/0/1

ip address 20.0.0.1 255.0.0.0
```

```
no shutdown
exit
ip route 50.0.0.0 255.0.0.0 20.0.0.2
end
write memory

Router1
enable
configure terminal
interface g0/0/0
ip address 20.0.0.2 255.0.0.0
no shutdown
exit
interface g0/0/1
ip address 30.0.0.1 255.0.0.0
no shutdown
exit
ip route 10.0.0.0 255.0.0.0 20.0.0.1
ip route 50.0.0.0 255.0.0.0 30.0.0.2
end
write memory

Router2
enable
configure terminal
interface g0/0/0
ip address 30.0.0.2 255.0.0.0
```

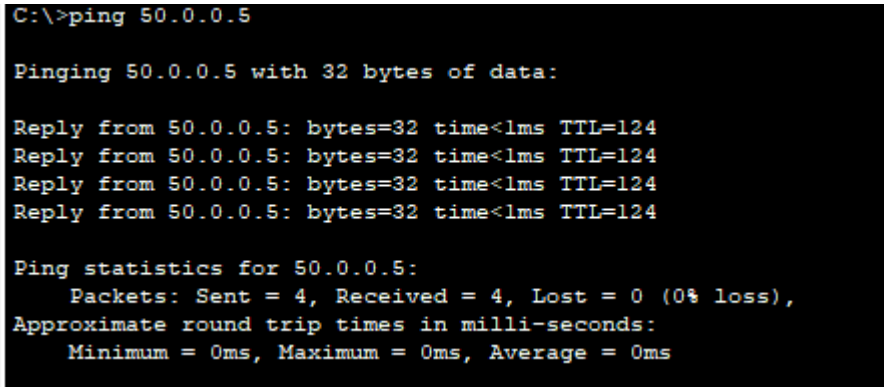
```
no shutdown
exit
interface g0/0/1
ip address 40.0.0.1 255.0.0.0
no shutdown
exit
ip route 10.0.0.0 255.0.0.0 30.0.0.1
ip route 50.0.0.0 255.0.0.0 40.0.0.2
end
write memory
Router3
enable
configure terminal
interface g0/0/0
ip address 40.0.0.2 255.0.0.0
no shutdown
exit
interface g0/0/1
ip address 50.0.0.1 255.0.0.0
no shutdown
exit
ip route 10.0.0.0 255.0.0.0 40.0.0.1
end
write memory
```

Testes de Conectividade

1. Clique no PC0 (rede 10.0.0.0).

2. Vá até Desktop > Command Prompt.

3. Execute o comando:
   ping 50.0.0.5



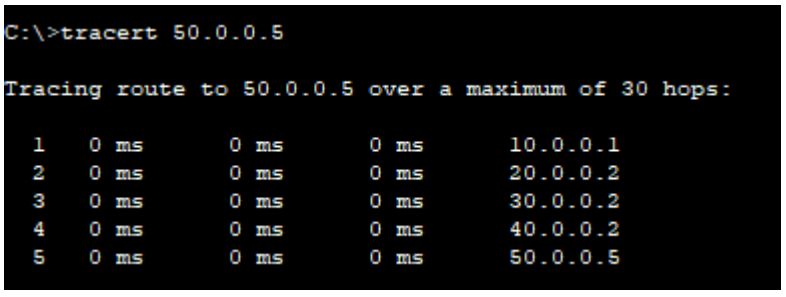
A comunicação entre os hosts 10.0.0.5 e 50.0.0.5 foi possível por meio da correta configuração dos IPs nas interfaces e da adição de rotas estáticas em todos os roteadores, garantindo que cada um conheça os caminhos até as redes de destino. A topologia demonstra bem o conceito de roteamento manual em redes segmentadas.

3 Questão - Funcionamento do Traceroute

Objetivo:

Demonstrar o funcionamento do comando tracert (equivalente ao traceroute) no Cisco Packet Tracer, evidenciando o caminho percorrido pelos pacotes desde o computador da rede 10.0.0.0/8 até o destino na rede 50.0.0.0/8.

Resultado Obtido:



O comando tracert permite visualizar o caminho que os pacotes percorrem entre o dispositivo de origem e o de destino, mostrando os roteadores intermediários por onde os dados passam.

No cenário da topologia utilizada, o comando foi executado a partir do computador localizado na rede 10.0.0.0/8 com destino ao computador da rede 50.0.0.0/8. O resultado do comando mostrou cinco saltos, correspondendo aos dispositivos intermediários e ao destino..

- O primeiro salto foi até o roteador Router0, que atua como gateway da rede de origem, com o IP 10.0.0.1.
- O segundo salto chegou ao roteador Router1, na interface 20.0.0.2, que está conectado ao Router0.

- O terceiro salto passou pelo roteador Router2, na interface 30.0.0.2.

- O quarto salto atingiu o roteador Router3, através do IP 40.0.0.2.

- Por fim, o quinto salto chegou ao destino final, o computador com IP 50.0.0.5.

Esse resultado comprova que o roteamento estático foi configurado corretamente, permitindo que os pacotes trafeguem de ponta a ponta pela rede, passando por todos os roteadores intermediários até alcançar o destino. Cada linha do tracert indica um salto bem-sucedido, e o tempo de resposta 0 ms reflete a simulação instantânea do Cisco Packet Tracer.

4 Questão -Implementação, Configuração e Testes de Topologia de Rede Virtual com Docker

O objetivo desta questão é implementar, configurar e testar uma topologia de rede virtual utilizando Docker. Esta topologia deverá incluir múltiplas redes locais (LANs), roteadores para interconectá-las, hosts clientes e servidores web. A finalidade é demonstrar a capacidade de comunicação entre os diferentes segmentos de rede, o correto funcionamento do roteamento de pacotes e o acesso aos serviços web hospedados nos containers Docker.

Passo 1: Criar as Redes Docker

```
echo "Removendo redes existentes (se houver)..."

docker network rm lan1 lan2 lan3 lan4 2>/dev/null || true

echo "Criando as 4 redes (LANs)..."

docker network create --driver bridge --subnet=192.168.1.0/24 lan1

docker network create --driver bridge --subnet=192.168.2.0/24 lan2

docker network create --driver bridge --subnet=192.168.3.0/24 lan3

docker network create --driver bridge --subnet=192.168.4.0/24 lan4

echo "Redes criadas."
```

Passo 2: Criar os Containers dos Roteadores

Comandos (extraídos do script .sh):

```
echo "### Passo 3: Criar os Containers dos Roteadores ###"
echo "Criando roteador1..."
docker run -d --name roteador1 --privileged --cap-add=NET_ADMIN
--cap-add=SYS_ADMIN --net lan1 --ip 192.168.1.10 ubuntu:latest
sleep infinity

echo "Criando roteador2..."
docker run -d --name roteador2 --privileged --cap-add=NET_ADMIN
--cap-add=SYS_ADMIN --net lan1 --ip 192.168.1.20 ubuntu:latest
sleep infinity

echo "Criando roteador3..."
docker run -d --name roteador3 --privileged --cap-add=NET_ADMIN
--cap-add=SYS_ADMIN --net lan1 --ip 192.168.1.30 ubuntu:latest
sleep infinity
echo "Containers dos roteadores criados."
echo "------------------------------------------------"
```

Passo 3: Conectar Roteadores às Redes Adicionais

Comandos (extraídos do script .sh):

```
echo "### Passo 4: Conectar Roteadores às Redes Adicionais ###"
echo "Conectando Roteador 2 às outras redes..."
docker network connect --ip 192.168.2.20 lan2 roteador2
docker network connect --ip 192.168.3.20 lan3 roteador2

echo "Conectando Roteador 3 às outras redes..."
docker network connect --ip 192.168.3.30 lan3 roteador3
docker network connect --ip 192.168.4.30 lan4 roteador3
echo "Roteadores conectados às redes adicionais."
echo "------------------------------------------------"
```

Passo 4: Criar os Hosts

```
echo "### Passo 5: Criar os Hosts ###"
# ... (comandos para criar host-lan2, host-lan3, host-lan4) ...
docker run -d --name host-lan2 --privileged --cap-add=NET_ADMIN
--net lan2 --ip 192.168.2.100 ubuntu:latest sleep infinity
docker run -d --name host-lan3 --privileged --cap-add=NET_ADMIN
--net lan3 --ip 192.168.3.100 ubuntu:latest sleep infinity
docker run -d --name host-lan4 --privileged --cap-add=NET_ADMIN
--net lan4 --ip 192.168.4.100 ubuntu:latest sleep infinity
echo "Hosts criados."
echo "------------------------------------------------"
```

Passo 5: Criar os Servidores Web com Mapeamento de Portas

```
docker run -d --name webserver-lan2 --privileged
--cap-add=NET_ADMIN --net lan2 --ip 192.168.2.200 -p 8081:80
nginx:latest
docker run -d --name webserver-lan3 --privileged
--cap-add=NET_ADMIN --net lan3 --ip 192.168.3.200 -p 8082:80
nginx:latest
docker run -d --name webserver-lan4 --privileged
--cap-add=NET_ADMIN --net lan4 --ip 192.168.4.200 -p 8083:80
nginx:latest
echo "Servidores web criados."
echo "------------------------------------------------"
```

Passo 6: Instalar Ferramentas em Todos os Containers.

```
docker exec roteador1 bash -c "apt update && apt install -y
iproute2 iputils-ping net-tools"
docker exec roteador2 bash -c "apt update && apt install -y
iproute2 iputils-ping net-tools"
docker exec roteador3 bash -c "apt update && apt install -y
iproute2 iputils-ping net-tools"
docker exec host-lan2 bash -c "apt update && apt install -y
iproute2 iputils-ping net-tools links"
docker exec host-lan3 bash -c "apt update && apt install -y
iproute2 iputils-ping net-tools links"
docker exec host-lan4 bash -c "apt update && apt install -y
iproute2 iputils-ping net-tools links"
docker exec webserver-lan2 bash -c "apt update && apt install -y
iproute2 iputils-ping net-tools"
docker exec webserver-lan3 bash -c "apt update && apt install -y
iproute2 iputils-ping net-tools"
docker exec webserver-lan4 bash -c "apt update && apt install -y
iproute2 iputils-ping net-tools"
echo "Ferramentas instaladas."
echo "------------------------------------------------"
```

Passo 7: Habilitar IP Forwarding nos Roteadores

```
echo "### Passo 8: Habilitar IP Forwarding nos Roteadores ###"
docker exec roteador1 bash -c "echo 1 >
/proc/sys/net/ipv4/ip_forward"
docker exec roteador2 bash -c "echo 1 >
/proc/sys/net/ipv4/ip_forward"
docker exec roteador3 bash -c "echo 1 >
/proc/sys/net/ipv4/ip_forward"
echo "IP Forwarding habilitado."
echo "------------------------------------------------"
```

Passo 8: Configurar Rotas nos Roteadores

```
docker exec roteador1 bash -c "ip route replace 192.168.2.0/24 via
192.168.1.20"
```

```
docker exec roteador1 bash -c "ip route replace 192.168.3.0/24 via
192.168.1.20"
docker exec roteador1 bash -c "ip route replace 192.168.4.0/24 via
192.168.1.30"
docker exec roteador2 bash -c "ip route replace 192.168.4.0/24 via
192.168.1.30"
docker exec roteador3 bash -c "ip route replace 192.168.2.0/24 via
192.168.1.20"
echo "Rotas configuradas nos roteadores."
echo "------------------------------------------------"
```

Passo 9: Configurar Gateway Padrão nos Hosts e Servidores

```
(comandos docker exec ip route add default via ...) ...
docker exec host-lan2 bash -c "ip route del default || true && ip
route add default via 192.168.2.20"
docker exec host-lan3 bash -c "ip route del default || true && ip
route add default via 192.168.3.20"
docker exec host-lan4 bash -c "ip route del default || true && ip
route add default via 192.168.4.30"
docker exec webserver-lan2 bash -c "ip route del default || true
&& ip route add default via 192.168.2.20"
docker exec webserver-lan3 bash -c "ip route del default || true
&& ip route add default via 192.168.3.20"
docker exec webserver-lan4 bash -c "ip route del default || true
&& ip route add default via 192.168.4.30"
echo "Gateways padrão configurados."
```

Passo 10: Personalizar Conteúdo Web dos Servidores Nginx

```
echo "Criando index.html para Portal Empresarial Alpha
(webserver-lan2)..."
docker exec webserver-lan2 bash -c 'cat >
/usr/share/nginx/html/index.html <<EOF
# ... (conteúdo HTML para webserver-lan2, conforme fornecido no
script original) ...
EOF'
echo "Criando index.html para Centro de Inovação Beta
(webserver-lan3)..."
docker exec webserver-lan3 bash -c 'cat >
/usr/share/nginx/html/index.html << EOF
# ... (conteúdo HTML para webserver-lan3, conforme fornecido no
script original) ...
EOF'
echo "Criando index.html para Hub Tecnológico Gamma
(webserver-lan4)..."
docker exec webserver-lan4 bash -c 'cat >
/usr/share/nginx/html/index.html << EOF
# ... (conteúdo HTML para webserver-lan4, conforme fornecido no
script original) ...
```

```
EOF'
echo "Páginas web personalizadas criadas."
echo "------------------------------------------------"
```

Passo 11: Configurar Resolução de Nomes (DNS/Hosts) nos Containers

```
echo "### Passo 12: Configurar DNS/Hosts nos Containers ###"
HOSTS_ENTRIES="
192.168.2.200 www.alpha-empresa.com alpha-empresa.com sitea
192.168.3.200 www.beta-inovacao.com beta-inovacao.com siteb
192.168.4.200 www.gamma-tech.com gamma-tech.com sitec
"
ROUTER_HOSTS_ENTRIES="
192.168.2.200 www.alpha-empresa.com alpha-empresa.com
192.168.3.200 www.beta-inovacao.com beta-inovacao.com
192.168.4.200 www.gamma-tech.com gamma-tech.com
"
# ... (comandos docker exec echo ... >> /etc/hosts) ...
docker exec host-lan2 bash -c "echo \"${HOSTS_ENTRIES}\" >>
/etc/hosts"
docker exec host-lan3 bash -c "echo \"${HOSTS_ENTRIES}\" >>
/etc/hosts"
docker exec host-lan4 bash -c "echo \"${HOSTS_ENTRIES}\" >>
/etc/hosts"
docker exec roteador1 bash -c "echo \"${ROUTER_HOSTS_ENTRIES}\" >>
/etc/hosts"
docker exec roteador2 bash -c "echo \"${ROUTER_HOSTS_ENTRIES}\" >>
/etc/hosts"
docker exec roteador3 bash -c "echo \"${ROUTER_HOSTS_ENTRIES}\" >>
/etc/hosts"
echo "Arquivos /etc/hosts configurados nos containers."
echo "------------------------------------------------"
```

Passo 13: Testes de Conectividade

```
echo "=== Teste de Conectividade entre Hosts ==="
echo "Ping de host-lan2 para host-lan3 (192.168.3.100):"
docker exec host-lan2 ping -c 3 192.168.3.100:
```

```
PS C:\Users\jorge> docker exec host-lan2 ping -c 3 192.168.3.100
PING 192.168.3.100 (192.168.3.100) 56(84) bytes of data.
64 bytes from 192.168.3.100: icmp_seq=1 ttl=63 time=0.176 ms
64 bytes from 192.168.3.100: icmp_seq=2 ttl=63 time=0.089 ms
64 bytes from 192.168.3.100: icmp_seq=3 ttl=63 time=0.106 ms

--- 192.168.3.100 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2030ms
rtt min/avg/max/mdev = 0.089/0.123/0.176/0.037 ms
```

```
echo "Ping de host-lan2 para host-lan4 (192.168.4.100):"
docker exec host-lan2 ping -c 3 192.168.4.100:
```

```
PS C:\Users\jorge> docker exec host-lan2 ping -c 3 192.168.4.100
PING 192.168.4.100 (192.168.4.100) 56(84) bytes of data.
64 bytes from 192.168.4.100: icmp_seq=1 ttl=62 time=0.231 ms
64 bytes from 192.168.4.100: icmp_seq=2 ttl=62 time=0.380 ms
64 bytes from 192.168.4.100: icmp_seq=3 ttl=62 time=0.122 ms
```

```
echo "Ping de host-lan3 para host-lan4 (192.168.4.100):"
docker exec host-lan3 ping -c 3 192.168.4.100:
```

```
PS C:\Users\jorge> docker exec host-lan3 ping -c 3 192.168.4.100
PING 192.168.4.100 (192.168.4.100) 56(84) bytes of data.
64 bytes from 192.168.4.100: icmp_seq=1 ttl=63 time=0.110 ms
64 bytes from 192.168.4.100: icmp_seq=2 ttl=63 time=0.095 ms
64 bytes from 192.168.4.100: icmp_seq=3 ttl=63 time=0.106 ms

--- 192.168.4.100 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2048ms
rtt min/avg/max/mdev = 0.095/0.103/0.110/0.006 ms
```

Teste de Acesso aos Servidores Web (links)

Comandos e Resultados Esperados (conforme script .sh):

```
echo "=== Teste de Acesso aos Servidores Web (links) ==="

echo "--- De host-lan2 ---"
echo "Acessando webserver-lan2 (Site A) por IP:"
docker exec host-lan2 links -dump http://192.168.2.200:
```

```
PS C:\Users\jorge> echo "Acessando webserver-lan2 (Site A) por IP:"
Acessando webserver-lan2 (Site A) por IP:
PS C:\Users\jorge> docker exec host-lan2 links -dump http://192.168.2.200
                    **** Portal Empresarial Alpha ****

   Bem-vindo ao nosso portal corporativo!

   Localizacao: Servidor na LAN2

   IP do Servidor: 192.168.2.200

   Dominio: www.alpha-empresa.com

   Solucoes empresariais de alta qualidade
```

```
echo "Acessando webserver-lan2 (Site A) por domínio:"
docker exec host-lan2 links -dump http://www.alpha-empresa.com:
```

```
PS C:\Users\jorge> echo "Acessando webserver-lan2 (Site A) por domínio:"
Acessando webserver-lan2 (Site A) por domínio:
PS C:\Users\jorge> docker exec host-lan2 links -dump http://www.alpha-empresa.com
                      **** Portal Empresarial Alpha ****

   Bem-vindo ao nosso portal corporativo!

   Localizacao: Servidor na LAN2

   IP do Servidor: 192.168.2.200

   Dominio: www.alpha-empresa.com

   Solucoes empresariais de alta qualidade
```

```
echo "---"
echo "Acessando webserver-lan3 (Site B) por IP:"
docker exec host-lan2 links -dump http://192.168.3.200:
```

```
PS C:\Users\jorge> echo "Acessando webserver-lan3 (Site B) por IP:"
Acessando webserver-lan3 (Site B) por IP:
PS C:\Users\jorge> docker exec host-lan2 links -dump http://192.168.3.200
                    **** Centro de Inovacao Beta ****

   Hub de tecnologia e inovacao!

   Localizacao: Servidor na LAN3

   IP do Servidor: 192.168.3.200

   Dominio: www.beta-inovacao.com

   Desenvolvendo o futuro da tecnologia
```

```
echo "Acessando webserver-lan3 (Site B) por domínio:"
docker exec host-lan2 links -dump http://www.beta-inovacao.com :
```

```
   Desenvolvendo o futuro da tecnologia
PS C:\Users\jorge> echo "Acessando webserver-lan3 (Site B) por domínio:"
Acessando webserver-lan3 (Site B) por domínio:
PS C:\Users\jorge> docker exec host-lan2 links -dump http://www.beta-inovacao.com
                      **** Centro de Inovacao Beta ****

   Hub de tecnologia e inovacao!

   Localizacao: Servidor na LAN3

   IP do Servidor: 192.168.3.200

   Dominio: www.beta-inovacao.com

   Desenvolvendo o futuro da tecnologia
```

```
echo "---"
echo "Acessando webserver-lan4 (Site C) por IP:"
docker exec host-lan2 links -dump http://192.168.4.200:
```

```
PS C:\Users\jorge> echo "Acessando webserver-lan4 (Site C) por IP:"
Acessando webserver-lan4 (Site C) por IP:
PS C:\Users\jorge> docker exec host-lan2 links -dump http://192.168.4.200
                        ** Hub Tecnologico Gamma **

   Centro de excelencia em tecnologia!

   Localizacao: Servidor na LAN4

   IP do Servidor: 192.168.4.200

   Dominio: www.gamma-tech.com

   Transformando ideias em realidade digital
```

```
echo "Acessando webserver-lan4 (Site C) por domínio:"
docker exec host-lan2 links -dump http://www.gamma-tech.com:
```

```
PS C:\Users\jorge> docker exec host-lan2 links -dump http://www.gamma-tech.com
                        ** Hub Tecnologico Gamma **

   Centro de excelencia em tecnologia!

   Localizacao: Servidor na LAN4

   IP do Servidor: 192.168.4.200

   Dominio: www.gamma-tech.com

   Transformando ideias em realidade digital
```

```
echo "--- De host-lan3 ---"
echo "Acessando webserver-lan2 (Site A) por IP:"
docker exec host-lan3 links -dump http://192.168.2.200:
```

```
--- De host-lan3 ---
PS C:\Users\jorge> echo "Acessando webserver-lan2 (Site A) por IP:"
Acessando webserver-lan2 (Site A) por IP:
PS C:\Users\jorge> docker exec host-lan3 links -dump http://192.168.2.200
                      **** Portal Empresarial Alpha ****

   Bem-vindo ao nosso portal corporativo!

   Localizacao: Servidor na LAN2

   IP do Servidor: 192.168.2.200

   Dominio: www.alpha-empresa.com

   Solucoes empresariais de alta qualidade
```

```
echo "Acessando webserver-lan2 (Site A) por domínio:"
docker exec host-lan3 links -dump http://www.alpha-empresa.com:
```

```
PS C:\Users\jorge> echo "Acessando webserver-lan2 (Site A) por domínio:"
Acessando webserver-lan2 (Site A) por domínio:
PS C:\Users\jorge> docker exec host-lan3 links -dump http://www.alpha-empresa.com
                      **** Portal Empresarial Alpha ****

   Bem-vindo ao nosso portal corporativo!

   Localizacao: Servidor na LAN2

   IP do Servidor: 192.168.2.200

   Dominio: www.alpha-empresa.com

   Solucoes empresariais de alta qualidade
```

```
echo "---"
echo "Acessando webserver-lan3 (Site B) por IP:"
docker exec host-lan3 links -dump http://192.168.3.200:
```

```
PS C:\Users\jorge> echo "Acessando webserver-lan3 (Site B) por IP:"
Acessando webserver-lan3 (Site B) por IP:
PS C:\Users\jorge> docker exec host-lan3 links -dump http://192.168.3.200
                      **** Centro de Inovacao Beta ****

   Hub de tecnologia e inovacao!

   Localizacao: Servidor na LAN3

   IP do Servidor: 192.168.3.200

   Dominio: www.beta-inovacao.com

   Desenvolvendo o futuro da tecnologia
```

```
echo "Acessando webserver-lan3 (Site B) por domínio:"
docker exec host-lan3 links -dump http://www.beta-inovacao.com:
```

```
PS C:\Users\jorge> echo "Acessando webserver-lan3 (Site B) por domínio:"
Acessando webserver-lan3 (Site B) por domínio:
PS C:\Users\jorge> docker exec host-lan3 links -dump http://www.beta-inovacao.com
                      **** Centro de Inovacao Beta ****

   Hub de tecnologia e inovacao!

   Localizacao: Servidor na LAN3

   IP do Servidor: 192.168.3.200

   Dominio: www.beta-inovacao.com

   Desenvolvendo o futuro da tecnologia
```

```
echo "---"
echo "Acessando webserver-lan4 (Site C) por IP:"
docker exec host-lan3 links -dump http://192.168.4.200:
```

```
PS C:\Users\jorge> echo "Acessando webserver-lan4 (Site C) por IP:"
Acessando webserver-lan4 (Site C) por IP:
PS C:\Users\jorge> docker exec host-lan3 links -dump http://192.168.4.200
                        ** Hub Tecnologico Gamma **

   Centro de excelencia em tecnologia!

   Localizacao: Servidor na LAN4

   IP do Servidor: 192.168.4.200

   Dominio: www.gamma-tech.com

   Transformando ideias em realidade digital
```

```
echo "Acessando webserver-lan4 (Site C) por domínio:"
docker exec host-lan3 links -dump http://www.gamma-tech.com:
```

```
PS C:\Users\jorge> echo "Acessando webserver-lan4 (Site C) por domínio:"
Acessando webserver-lan4 (Site C) por domínio:
PS C:\Users\jorge> docker exec host-lan3 links -dump http://www.gamma-tech.com
                         ** Hub Tecnologico Gamma **

   Centro de excelencia em tecnologia!

   Localizacao: Servidor na LAN4

   IP do Servidor: 192.168.4.200

   Dominio: www.gamma-tech.com

   Transformando ideias em realidade digital
```

```
echo "------------------------------------------------"

echo "--- De host-lan4 ---"
echo "Acessando webserver-lan2 (Site A) por IP:"
docker exec host-lan4 links -dump http://192.168.2.200:
```

```
--- De host-lan4 ---
PS C:\Users\jorge> echo "Acessando webserver-lan2 (Site A) por IP:"
Acessando webserver-lan2 (Site A) por IP:
PS C:\Users\jorge> docker exec host-lan4 links -dump http://192.168.2.200
                      **** Portal Empresarial Alpha ****

   Bem-vindo ao nosso portal corporativo!

   Localizacao: Servidor na LAN2

   IP do Servidor: 192.168.2.200

   Dominio: www.alpha-empresa.com

   Solucoes empresariais de alta qualidade
```

```
echo "Acessando webserver-lan2 (Site A) por domínio:"
docker exec host-lan4 links -dump http://www.alpha-empresa.com:
```

```
PS C:\Users\jorge> echo "Acessando webserver-lan2 (Site A) por domínio:"
Acessando webserver-lan2 (Site A) por domínio:
PS C:\Users\jorge> docker exec host-lan4 links -dump http://www.alpha-empresa.com
                      **** Portal Empresarial Alpha ****

   Bem-vindo ao nosso portal corporativo!

   Localizacao: Servidor na LAN2

   IP do Servidor: 192.168.2.200

   Dominio: www.alpha-empresa.com

   Solucoes empresariais de alta qualidade
```

```
echo "---"
echo "Acessando webserver-lan3 (Site B) por IP:"
docker exec host-lan4 links -dump http://192.168.3.200:
```

```
PS C:\Users\jorge> echo "Acessando webserver-lan3 (Site B) por IP:"
Acessando webserver-lan3 (Site B) por IP:
PS C:\Users\jorge> docker exec host-lan4 links -dump http://192.168.3.200
                     **** Centro de Inovacao Beta ****

   Hub de tecnologia e inovacao!

   Localizacao: Servidor na LAN3

   IP do Servidor: 192.168.3.200

   Dominio: www.beta-inovacao.com

   Desenvolvendo o futuro da tecnologia
```

```
echo "Acessando webserver-lan3 (Site B) por domínio:"
docker exec host-lan4 links -dump http://www.beta-inovacao.com:
```

```
PS C:\Users\jorge> echo "Acessando webserver-lan3 (Site B) por domínio:"
Acessando webserver-lan3 (Site B) por domínio:
PS C:\Users\jorge> docker exec host-lan4 links -dump http://www.beta-inovacao.co
                      **** Centro de Inovacao Beta ****

   Hub de tecnologia e inovacao!

   Localizacao: Servidor na LAN3

   IP do Servidor: 192.168.3.200

   Dominio: www.beta-inovacao.com

   Desenvolvendo o futuro da tecnologia
```

```
echo "---"
echo "Acessando webserver-lan4 (Site C) por IP:"
docker exec host-lan4 links -dump http://192.168.4.200:
```

```
PS C:\Users\jorge> echo "Acessando webserver-lan4 (Site C) por IP:"
Acessando webserver-lan4 (Site C) por IP:
PS C:\Users\jorge> docker exec host-lan4 links -dump http://192.168.4.200
                        ** Hub Tecnologico Gamma **

   Centro de excelencia em tecnologia!

   Localizacao: Servidor na LAN4

   IP do Servidor: 192.168.4.200

   Dominio: www.gamma-tech.com

   Transformando ideias em realidade digital
```

```
echo "Acessando webserver-lan4 (Site C) por domínio:"
docker exec host-lan4 links -dump http://www.gamma-tech.com:
```

```
PS C:\Users\jorge> echo "Acessando webserver-lan4 (Site C) por domínio:"
Acessando webserver-lan4 (Site C) por domínio:
PS C:\Users\jorge> docker exec host-lan4 links -dump http://www.gamma-tech.com
                        ** Hub Tecnologico Gamma **

   Centro de excelencia em tecnologia!

   Localizacao: Servidor na LAN4

   IP do Servidor: 192.168.4.200

   Dominio: www.gamma-tech.com

   Transformando ideias em realidade digital
```

Conclusão:

A implementação da topologia de rede virtual com Docker foi concluída com sucesso, conforme detalhado nesta Questão 4. Todos os passos de criação, configuração e teste, executados por meio de um script shell (.sh) disponível no repositório GitHub anexo, foram demonstrados. Os resultados obtidos (evidenciados pelas capturas de tela sugeridas) demonstram o correto funcionamento da comunicação entre as LANs, o roteamento de pacotes e o acesso aos serviços web. Esta configuração serve como uma base prática para a compreensão de conceitos de rede e virtualização.

5 Questão - Implementação de Roteamento Dinâmico com RIP (FRR)

Objetivo:

O objetivo desta questão é refazer o roteamento estático implementado na Questão 04, substituindo-o pelo roteamento dinâmico RIP (Routing Information Protocol) utilizando FRRouting (FRR). Serão demonstradas a instalação e configuração do FRR nos containers Docker que atuam como roteadores, a configuração do protocolo RIP, a verificação da propagação de rotas e os testes de conectividade na topologia de rede estabelecida.

Instalar Ferramentas e FRR nos Containers

Nesta etapa, além das ferramentas de rede padrão, o FRRouting (FRR) é instalado nos containers designados como roteadores.

```
ROUTERS="roteador1 roteador2 roteador3"

for router in $ROUTERS; do

  echo "Instalando ferramentas e FRR no $router..."

  # Adicionar -y para apt-get install e garantir noninteractive
para frr

  docker exec $router bash -c "DEBIAN_FRONTEND=noninteractive
apt-get update && apt-get install -y iproute2 iputils-ping
net-tools frr"

done

HOSTS="host-lan2 host-lan3 host-lan4"

# ... (instalação de ferramentas nos hosts) ...

WEBSERVERS="webserver-lan2 webserver-lan3 webserver-lan4"

# ... (instalação de ferramentas nos webservers) ...

echo "Ferramentas instaladas."

echo ""
```

Configurar RIP com FRR nos Roteadores

```
echo "### Passo 9: Configurando RIP com FRR nos Roteadores ###"

FRR_DAEMONS_CONFIG='sed -i "s/zebra=no/zebra=yes/"
/etc/frr/daemons && sed -i "s/ripd=no/ripd=yes/" /etc/frr/daemons'

# Configurar Roteador 1

echo "Configurando FRR no roteador1..."

docker exec roteador1 bash -c "$FRR_DAEMONS_CONFIG && \

cat > /etc/frr/frr.conf << EOF

frr defaults traditional

hostname roteador1

log syslog informational

!
```

```
router rip
 version 2
 network 192.168.1.0/24
 redistribute connected
!
line vty
!
EOF
service frr restart"

# Configurar Roteador 2
echo "Configurando FRR no roteador2..."
docker exec roteador2 bash -c "$FRR_DAEMONS_CONFIG && \
cat > /etc/frr/frr.conf << EOF
frr defaults traditional
hostname roteador2
log syslog informational
!
router rip
 version 2
 network 192.168.1.0/24
 network 192.168.2.0/24
 network 192.168.3.0/24
 redistribute connected
!
line vty
!
```

```
EOF
service frr restart"
# Configurar Roteador 3
echo "Configurando FRR no roteador3..."
docker exec roteador3 bash -c "$FRR_DAEMONS_CONFIG && \
cat > /etc/frr/frr.conf << EOF
frr defaults traditional
hostname roteador3
log syslog informational
!
router rip
 version 2
 network 192.168.1.0/24
 network 192.168.3.0/24
 network 192.168.4.0/24
 redistribute connected
!
line vty
!
EOF
service frr restart"
echo "Configuração RIP (FRR) concluída e serviços reiniciados."
echo "Aguardando convergência do RIP (20 segundos)..."
sleep 20 # Dar tempo para o RIP convergir
```

Testes de Conectividade

Verifica a conectividade entre hosts em diferentes LANs e o acesso aos servidores web por IP e domínio. Com o RIP configurado, o roteamento deve ocorrer dinamicamente.

```
echo "### Passo 14: Testes de Conectividade (via RIP com FRR) ###"

echo "=== Teste de Conectividade entre Hosts ==="

docker exec host-lan2 ping -c 3 192.168.3.100

docker exec host-lan2 ping -c 3 192.168.4.100

# ... (outros pings) ...

echo ""

echo "=== Teste de Acesso por IP ==="

docker exec host-lan2 links -dump http://192.168.2.200 # Site A

docker exec host-lan2 links -dump http://192.168.3.200 # Site B

docker exec host-lan2 links -dump http://192.168.4.200 # Site C

echo ""

echo "=== Teste de Acesso por Domínio ==="

docker exec host-lan2 links -dump http://www.alpha-empresa.com
```

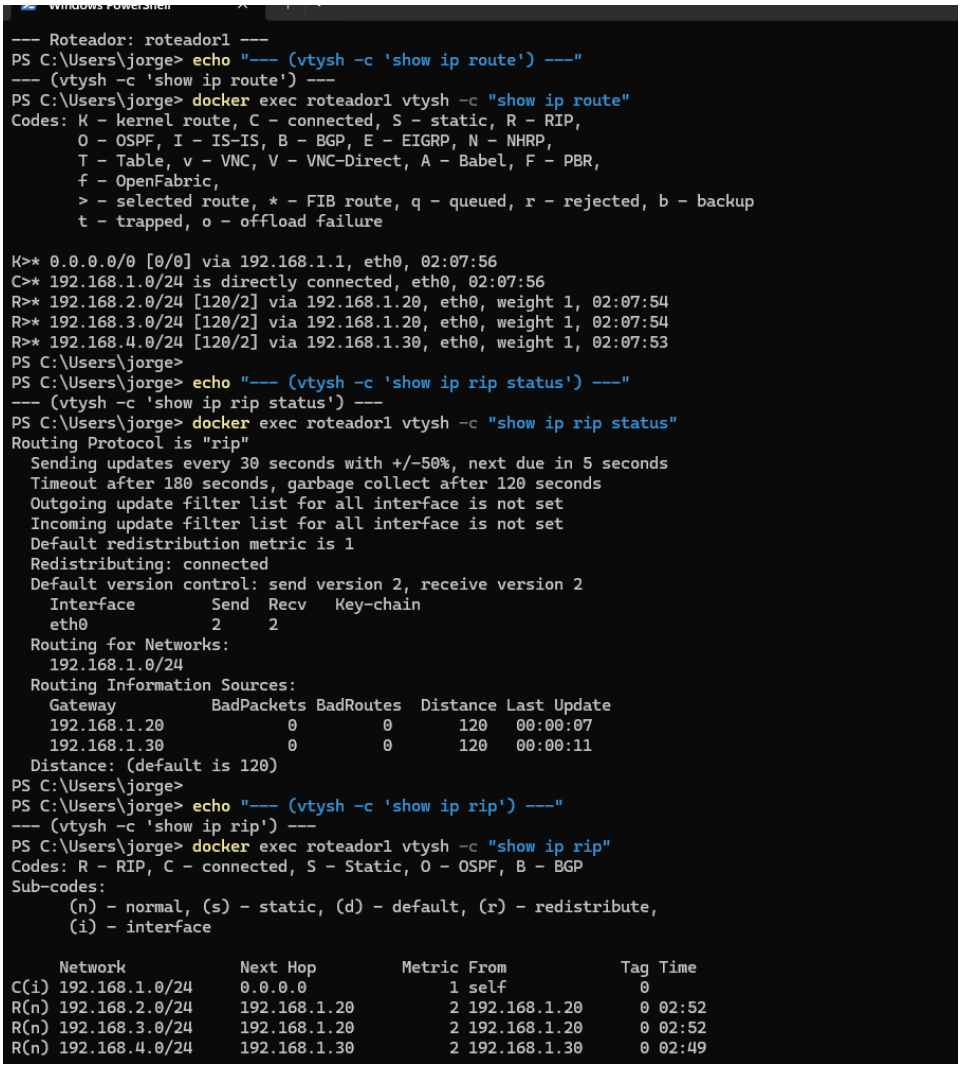
Verificação do Funcionamento do RIP nos Roteadores

Esta etapa é crucial para verificar o estado e o correto funcionamento do RIP nos roteadores, confirmando que as rotas estão sendo aprendidas dinamicamente.

```
for router in $ROUTERS; do

  echo ""

  echo "--- Roteador: $router ---"

  echo "--- (vtysh -c 'show ip route') ---"

  docker exec $router vtysh -c "show ip route"

  sleep 1 # Pequena pausa para não sobrecarregar o docker exec em
loops rápidos

  echo "--- (vtysh -c 'show ip rip status') ---"

  docker exec $router vtysh -c "show ip rip status"

  sleep 1

  echo "--- (vtysh -c 'show ip rip') ---"
```

```
  docker exec $router vtysh -c "show ip rip" # Mostra rotas RIP
específicas e seus timers

  sleep 1

done

echo ""
```

Roteador1 – Análise do Protocolo RIP



```
--- Roteador: roteador1 ---
PS C:\Users\jorge> echo "--- (vtysh -c 'show ip route') ---"
--- (vtysh -c 'show ip route') ---
PS C:\Users\jorge> docker exec roteador1 vtysh -c "show ip route"
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

K>* 0.0.0.0/0 [0/0] via 192.168.1.1, eth0, 02:07:56
C>* 192.168.1.0/24 is directly connected, eth0, 02:07:56
R>* 192.168.2.0/24 [120/2] via 192.168.1.20, eth0, weight 1, 02:07:54
R>* 192.168.3.0/24 [120/2] via 192.168.1.20, eth0, weight 1, 02:07:54
R>* 192.168.4.0/24 [120/2] via 192.168.1.30, eth0, weight 1, 02:07:53
PS C:\Users\jorge>
PS C:\Users\jorge> echo "--- (vtysh -c 'show ip rip status') ---"
--- (vtysh -c 'show ip rip status') ---
PS C:\Users\jorge> docker exec roteador1 vtysh -c "show ip rip status"
Routing Protocol is "rip"
  Sending updates every 30 seconds with +/-50%, next due in 5 seconds
  Timeout after 180 seconds, garbage collect after 120 seconds
  Outgoing update filter list for all interface is not set
  Incoming update filter list for all interface is not set
  Default redistribution metric is 1
  Redistributing: connected
  Default version control: send version 2, receive version 2
    Interface        Send  Recv   Key-chain
    eth0             2     2
  Routing for Networks:
    192.168.1.0/24
  Routing Information Sources:
    Gateway          BadPackets BadRoutes  Distance Last Update
    192.168.1.20             0         0       120   00:00:07
    192.168.1.30             0         0       120   00:00:11
  Distance: (default is 120)
PS C:\Users\jorge>
PS C:\Users\jorge> echo "--- (vtysh -c 'show ip rip') ---"
--- (vtysh -c 'show ip rip') ---
PS C:\Users\jorge> docker exec roteador1 vtysh -c "show ip rip"
Codes: R - RIP, C - connected, S - Static, O - OSPF, B - BGP
Sub-codes:
      (n) - normal, (s) - static, (d) - default, (r) - redistribute,
      (i) - interface

     Network            Next Hop         Metric From            Tag Time
C(i) 192.168.1.0/24     0.0.0.0               1 self              0
R(n) 192.168.2.0/24     192.168.1.20          2 192.168.1.20      0 02:52
R(n) 192.168.3.0/24     192.168.1.20          2 192.168.1.20      0 02:52
R(n) 192.168.4.0/24     192.168.1.30          2 192.168.1.30      0 02:49
```

O roteador1 está configurado com o protocolo de roteamento dinâmico RIP (Routing Information Protocol). Através dos comandos executados, foi possível verificar que o roteador está operando corretamente, aprendendo rotas de redes vizinhas e atualizando sua tabela de roteamento.

A tabela de rotas (show ip route) mostra que o roteador possui uma rota padrão (0.0.0.0/0) via 192.168.1.1, uma rede diretamente conectada (192.168.1.0/24), e rotas aprendidas via RIP para as redes 192.168.2.0/24, 192.168.3.0/24 e 192.168.4.0/24. Todas essas rotas têm distância administrativa de 120 e métrica 2, indicando dois saltos de distância.

O comando show ip rip status confirmou que o roteador está enviando atualizações RIP a cada 30 segundos e está roteando apenas a rede 192.168.1.0/24. As informações de roteamento estão sendo recebidas de dois vizinhos: 192.168.1.20 e 192.168.1.30, o que indica troca ativa de rotas.

Por fim, o comando show ip rip detalhou as rotas conhecidas via RIP. A rede local aparece como conectada (C), enquanto as demais foram aprendidas normalmente (R). As rotas para 192.168.2.0/24 e 192.168.3.0/24 estão sendo encaminhadas para 192.168.1.20, e a rota para 192.168.4.0/24 para 192.168.1.30.

Com isso, confirma-se que o roteador1 está funcionando corretamente dentro da topologia RIP, aprendendo rotas e repassando atualizações conforme o esperado.

Roteador2 – Análise do Protocolo RIP:

```
PS C:\Users\jorge> echo "--- Roteador: roteador2 ---"
--- Roteador: roteador2 ---
PS C:\Users\jorge> echo "--- (vtysh -c 'show ip route') ---"
--- (vtysh -c 'show ip route') ---
PS C:\Users\jorge> docker exec roteador2 vtysh -c "show ip route"
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

K>* 0.0.0.0/0 [0/0] via 192.168.1.1, eth0, 02:10:55
C>* 192.168.1.0/24 is directly connected, eth0, 02:10:55
C>* 192.168.2.0/24 is directly connected, eth1, 02:10:55
C>* 192.168.3.0/24 is directly connected, eth2, 02:10:55
R>* 192.168.4.0/24 [120/2] via 192.168.1.30, eth0, weight 1, 02:10:52
PS C:\Users\jorge>
PS C:\Users\jorge> echo "--- (vtysh -c 'show ip rip status') ---"
--- (vtysh -c 'show ip rip status') ---
PS C:\Users\jorge> docker exec roteador2 vtysh -c "show ip rip status"
Routing Protocol is "rip"
  Sending updates every 30 seconds with +/-50%, next due in 9 seconds
  Timeout after 180 seconds, garbage collect after 120 seconds
  Outgoing update filter list for all interface is not set
  Incoming update filter list for all interface is not set
  Default redistribution metric is 1
  Redistributing: connected
  Default version control: send version 2, receive version 2
    Interface        Send  Recv   Key-chain
    eth0             2     2
    eth1             2     2
    eth2             2     2
  Routing for Networks:
    192.168.1.0/24
    192.168.2.0/24
    192.168.3.0/24
  Routing Information Sources:
    Gateway          BadPackets BadRoutes  Distance Last Update
    192.168.1.30             0         0       120   00:00:25
    192.168.3.30             0         0       120   00:00:25
  Distance: (default is 120)
PS C:\Users\jorge>
PS C:\Users\jorge> echo "--- (vtysh -c 'show ip rip') ---"
--- (vtysh -c 'show ip rip') ---
PS C:\Users\jorge> docker exec roteador2 vtysh -c "show ip rip"
Codes: R - RIP, C - connected, S - Static, O - OSPF, B - BGP
Sub-codes:
      (n) - normal, (s) - static, (d) - default, (r) - redistribute,
      (i) - interface

     Network            Next Hop         Metric From            Tag Time
C(i) 192.168.1.0/24     0.0.0.0               1 self              0
C(i) 192.168.2.0/24     0.0.0.0               1 self              0
C(i) 192.168.3.0/24     0.0.0.0               1 self              0
R(n) 192.168.4.0/24     192.168.1.30          2 192.168.1.30      0 02:35
```

O roteador2 também está utilizando o protocolo RIP para troca dinâmica de rotas. Através dos comandos analisados, observamos que ele está conectado diretamente às redes 192.168.1.0/24, 192.168.2.0/24 e 192.168.3.0/24, conforme exibido na tabela de roteamento com prefixo C (connected).

Além disso, ele possui uma rota padrão (0.0.0.0/0) configurada via o gateway 192.168.1.1. A rota para a rede 192.168.4.0/24 está sendo aprendida dinamicamente via RIP através do

endereço 192.168.1.30, com distância administrativa padrão de 120 e métrica 2, indicando dois saltos até o destino.

O comando show ip rip status confirmou que o roteador está roteando três redes locais (192.168.1.0/24, 192.168.2.0/24 e 192.168.3.0/24) por meio do protocolo RIP, com envios e recebimentos ativos nas interfaces eth0, eth1 e eth2. As atualizações estão sendo recebidas de dois vizinhos: 192.168.1.30 e 192.168.3.30, demonstrando que há troca efetiva de rotas com outros roteadores da rede.

Por fim, o comando show ip rip mostra que as redes locais aparecem como conectadas diretamente (C(i)), enquanto a rede 192.168.4.0/24 foi aprendida dinamicamente de forma normal (R(n)), via o roteador em 192.168.1.30.

Essas informações confirmam que o roteador2 está corretamente configurado no ambiente RIP, atuando como um ponto intermediário para outras redes e redistribuindo rotas de forma eficiente.

Roteador3 – Análise do Protocolo RIP

```
PS C:\Users\jorge> echo "--- Roteador: roteador3 ---"
--- Roteador: roteador3 ---
PS C:\Users\jorge> echo "--- (vtysh -c 'show ip route') ---"
--- (vtysh -c 'show ip route') ---
PS C:\Users\jorge> docker exec roteador3 vtysh -c "show ip route"
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
       f - OpenFabric,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

K>* 0.0.0.0/0 [0/0] via 192.168.1.1, eth0, 02:14:41
C>* 192.168.1.0/24 is directly connected, eth0, 02:14:41
R>* 192.168.2.0/24 [120/2] via 192.168.1.20, eth0, weight 1, 02:14:40
C>* 192.168.3.0/24 is directly connected, eth1, 02:14:41
C>* 192.168.4.0/24 is directly connected, eth2, 02:14:41
PS C:\Users\jorge>
PS C:\Users\jorge> echo "--- (vtysh -c 'show ip rip status') ---"
--- (vtysh -c 'show ip rip status') ---
PS C:\Users\jorge> docker exec roteador3 vtysh -c "show ip rip status"
Routing Protocol is "rip"
  Sending updates every 30 seconds with +/-50%, next due in 17 seconds
  Timeout after 180 seconds, garbage collect after 120 seconds
  Outgoing update filter list for all interface is not set
  Incoming update filter list for all interface is not set
  Default redistribution metric is 1
  Redistributing: connected
  Default version control: send version 2, receive version 2
    Interface        Send  Recv   Key-chain
    eth0             2     2
    eth1             2     2
    eth2             2     2
  Routing for Networks:
    192.168.1.0/24
    192.168.3.0/24
    192.168.4.0/24
  Routing Information Sources:
    Gateway          BadPackets BadRoutes  Distance Last Update
    192.168.1.20             0         0       120   00:00:07
    192.168.3.20             0         0       120   00:00:07
  Distance: (default is 120)
PS C:\Users\jorge>
PS C:\Users\jorge> echo "--- (vtysh -c 'show ip rip') ---"
--- (vtysh -c 'show ip rip') ---
PS C:\Users\jorge> docker exec roteador3 vtysh -c "show ip rip"
Codes: R - RIP, C - connected, S - Static, O - OSPF, B - BGP
Sub-codes:
      (n) - normal, (s) - static, (d) - default, (r) - redistribute,
      (i) - interface

     Network            Next Hop         Metric From            Tag Time
C(i) 192.168.1.0/24     0.0.0.0               1 self              0
R(n) 192.168.2.0/24     192.168.1.20          2 192.168.1.20      0 02:52
C(i) 192.168.3.0/24     0.0.0.0               1 self              0
C(i) 192.168.4.0/24     0.0.0.0               1 self              0
```

O roteador3 também está configurado para utilizar o protocolo RIP (Routing Information Protocol) para propagação dinâmica de rotas na rede. Ele possui conexões diretas com as redes 192.168.1.0/24, 192.168.3.0/24 e 192.168.4.0/24, identificadas como rotas conectadas diretamente (C(i)) nas tabelas de roteamento.

A rota para 192.168.2.0/24 é aprendida dinamicamente via RIP, com próximo salto 192.168.1.20 e métrica 2, indicando que há dois saltos até alcançar essa rede. Isso demonstra o funcionamento correto da troca de informações entre roteadores, já que essa rede não está diretamente conectada ao roteador3.

Segundo a saída do comando show ip rip status, o roteador está roteando três redes (192.168.1.0/24, 192.168.3.0/24 e 192.168.4.0/24) e está recebendo atualizações de dois vizinhos RIP: 192.168.1.20 e 192.168.3.20, com distância administrativa padrão de 120.

O protocolo está operando na versão 2, com envio e recebimento de atualizações habilitado nas três interfaces (eth0, eth1 e eth2). A redistribuição de rotas conectadas também está ativa, garantindo que as redes diretamente conectadas ao roteador3 sejam anunciadas a seus vizinhos.

O comando show ip rip confirma o correto funcionamento do protocolo, com destaque para a rota RIP aprendida (R(n)) para 192.168.2.0/24, recebida do roteador em 192.168.1.20.

Com isso, é possível concluir que o roteador3 está devidamente configurado para participar da rede dinâmica RIP, trocando rotas com os demais roteadores e permitindo o roteamento eficiente entre todas as sub-redes do ambiente.

6 Questão - Implementação de Roteamento Dinâmico com OSPF (FRR)

O objetivo desta questão é refazer o roteamento estático implementado na Questão 04, substituindo-o pelo roteamento dinâmico OSPF (Open Shortest Path First) utilizando FRRouting (FRR). Serão demonstradas a instalação e configuração do FRR nos containers Docker que atuam como roteadores, a configuração do protocolo OSPF, a verificação da formação de adjacências, a propagação de rotas e os testes de conectividade na topologia de rede estabelecida.

Configurar OSPF com FRR nos Roteadores

Esta é a etapa central onde o OSPF é configurado.

```
FRR_DAEMONS_CONFIG_OSPF='sed -i "s/zebra=no/zebra=yes/"
/etc/frr/daemons && sed -i "s/ospfd=no/ospfd=yes/"
/etc/frr/daemons'

# Configurar Roteador 1

echo "Configurando FRR (OSPF) no roteador1..."

docker exec roteador1 bash -c "$FRR_DAEMONS_CONFIG_OSPF && \

cat > /etc/frr/frr.conf << EOF
```

```
frr defaults traditional
hostname roteador1
log syslog informational
!
router ospf
 ospf router-id 192.168.1.10
 network 192.168.1.0/24 area 0.0.0.0
!
line vty
!
EOF
service frr restart"
# Configurar Roteador 2
echo "Configurando FRR (OSPF) no roteador2..."
docker exec roteador2 bash -c "$FRR_DAEMONS_CONFIG_OSPF && \
cat > /etc/frr/frr.conf << EOF
frr defaults traditional
hostname roteador2
log syslog informational
!
router ospf
 ospf router-id 192.168.1.20
 network 192.168.1.0/24 area 0.0.0.0
 network 192.168.2.0/24 area 0.0.0.0
 network 192.168.3.0/24 area 0.0.0.0
!
line vty
```

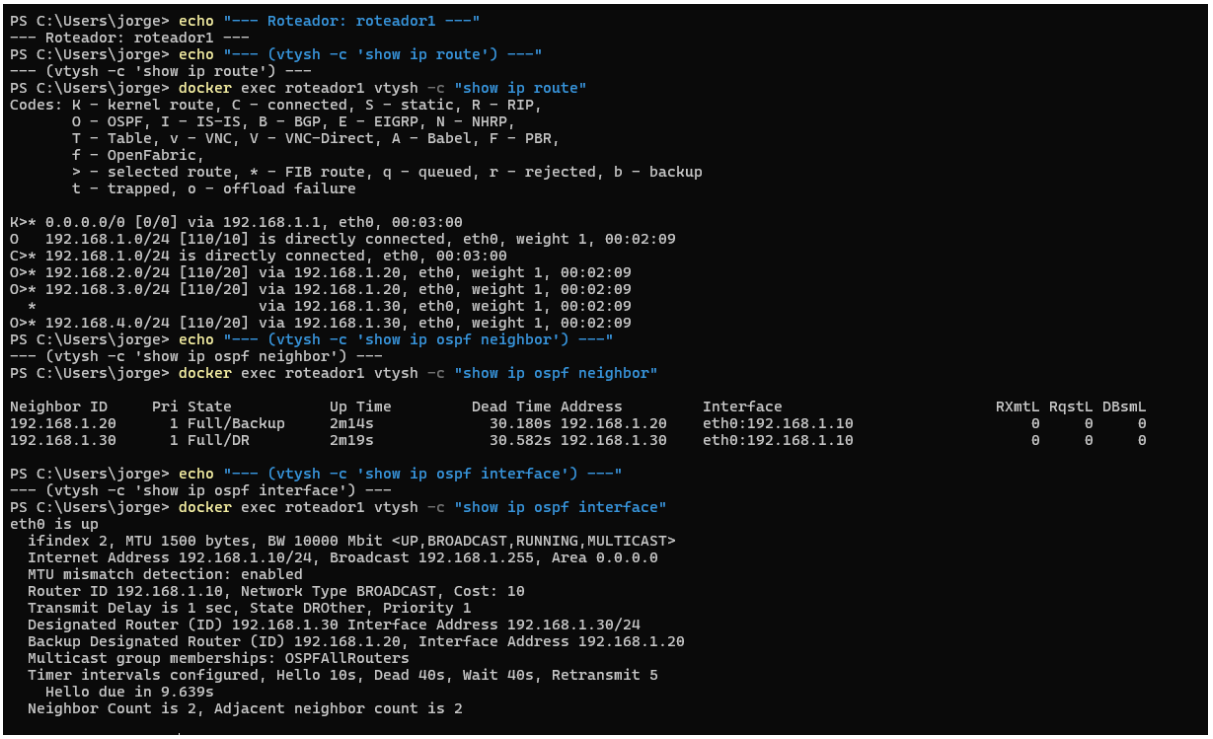
```
!
EOF
service frr restart"
# Configurar Roteador 3
echo "Configurando FRR (OSPF) no roteador3..."
docker exec roteador3 bash -c "$FRR_DAEMONS_CONFIG_OSPF && \
cat > /etc/frr/frr.conf << EOF
frr defaults traditional
hostname roteador3
log syslog informational
!
router ospf
 ospf router-id 192.168.1.30
 network 192.168.1.0/24 area 0.0.0.0
 network 192.168.3.0/24 area 0.0.0.0
 network 192.168.4.0/24 area 0.0.0.0
!
line vty
!
EOF
service frr restart"
echo "Configuração OSPF (FRR) concluída e serviços reiniciados."
echo "Aguardando convergência do OSPF (45 segundos)..."
sleep 45 # OSPF pode demorar um pouco mais para convergir
echo ""
```

Verificação do Funcionamento do OSPF nos Roteadores

Esta é a etapa fundamental para comprovar que o OSPF está operando corretamente e que as rotas estão sendo aprendidas dinamicamente.

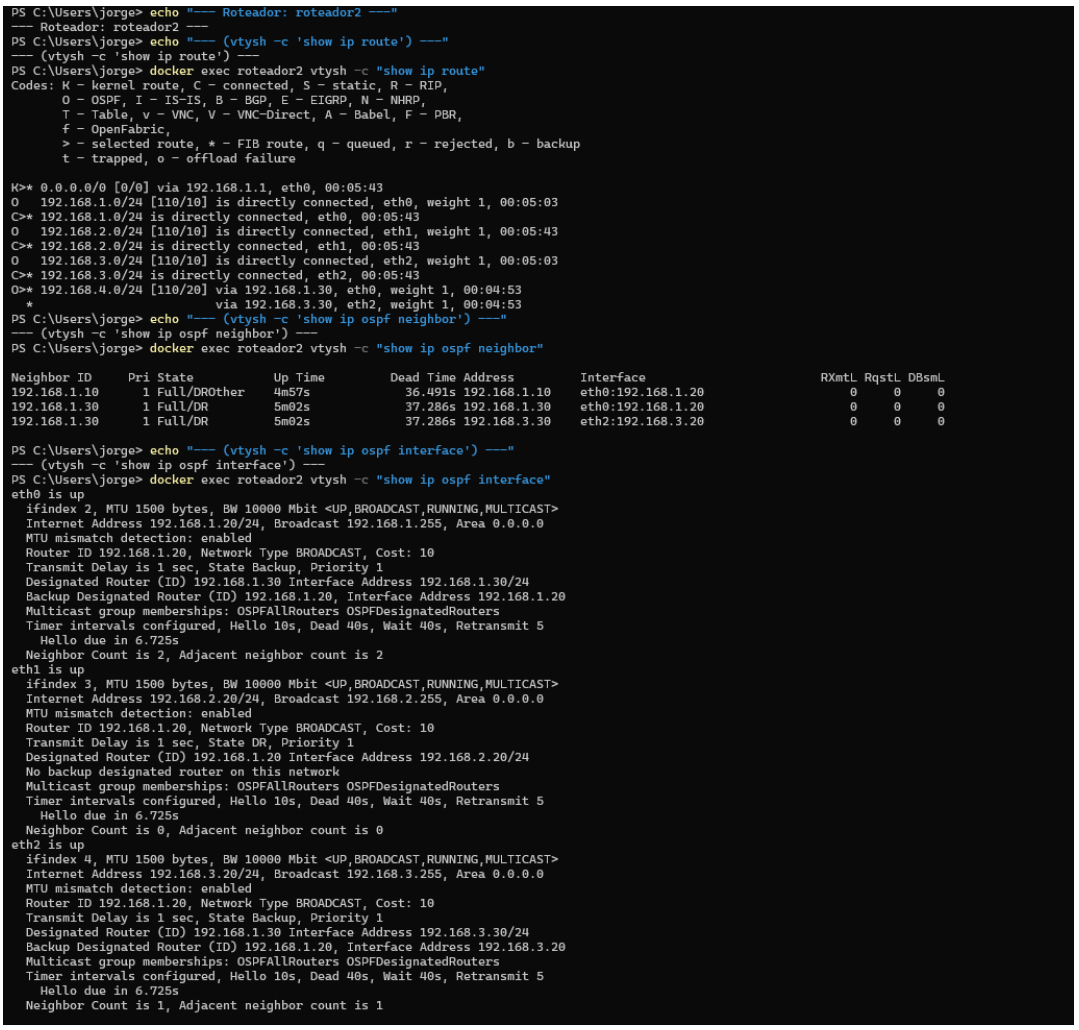
```
for router in $ROUTERS; do

  echo ""

  echo "--- Roteador: $router ---"

  echo "--- (vtysh -c 'show ip route') ---"

  docker exec $router vtysh -c "show ip route"

  sleep 1

  echo "--- (vtysh -c 'show ip ospf neighbor') ---"

  docker exec $router vtysh -c "show ip ospf neighbor"

  sleep 1

  echo "--- (vtysh -c 'show ip ospf interface') ---"

  docker exec $router vtysh -c "show ip ospf interface"

  sleep 1

  # echo "--- $router (vtysh -c 'show ip ospf database') ---" #
Descomente para mais detalhes

  # docker exec $router vtysh -c "show ip ospf database"

  # sleep 1

done
```

Roteador1 – Análise do Protocolo OSPF:



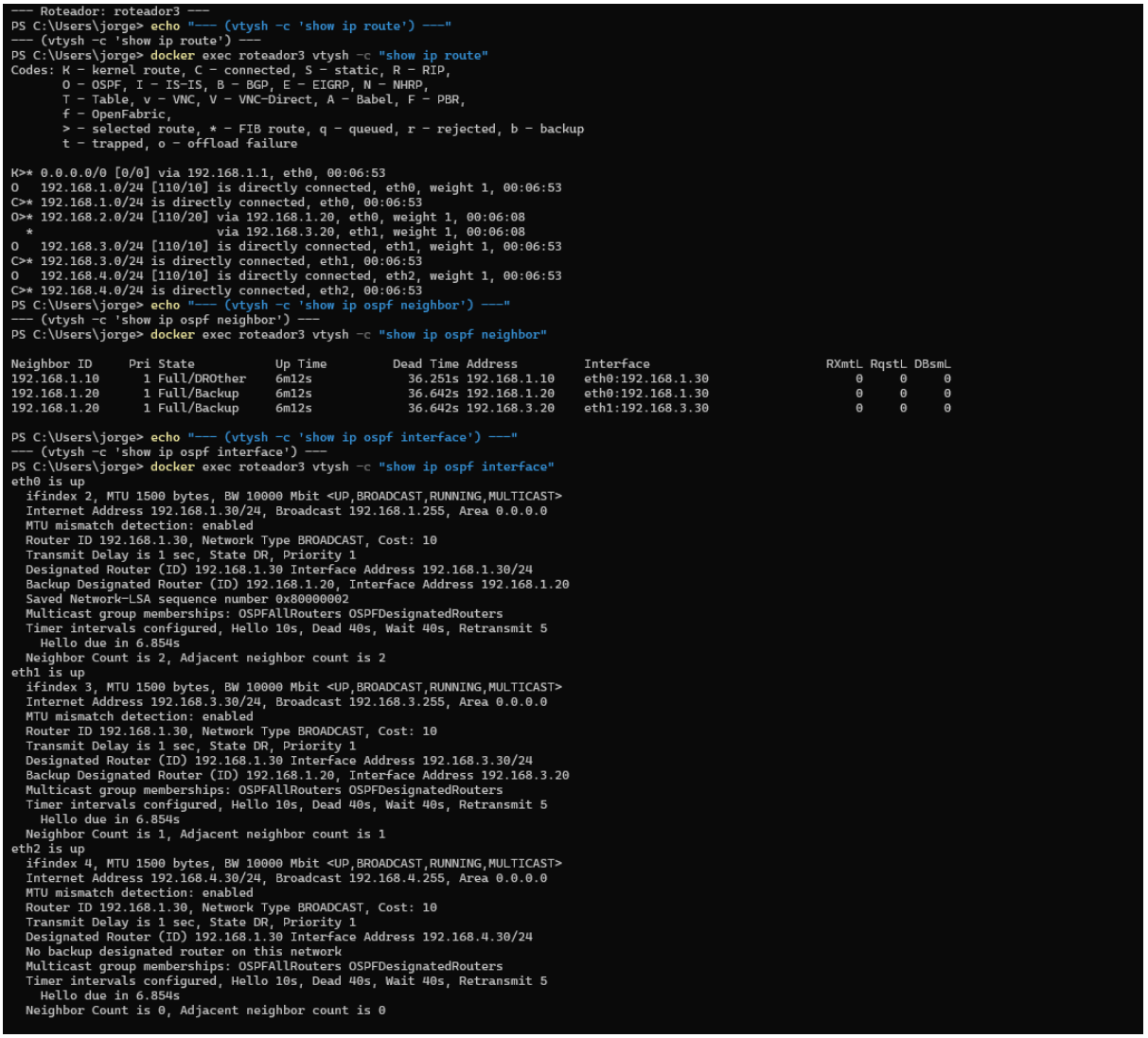
O roteador1 está aprendendo rotas via OSPF, conforme mostra a tabela de roteamento onde aparecem redes diretamente conectadas e rotas OSPF para outras sub-redes. A presença de múltiplas rotas OSPF indica que ele está recebendo informações de diferentes vizinhos. O comando que mostra os vizinhos OSPF revela que o roteador está totalmente adjacente a dois vizinhos, um com papel de Designated Router (DR) e outro como Backup DR, o que é típico em redes de broadcast para garantir redundância e estabilidade. A interface eth0 está ativa e configurada corretamente para OSPF, participando da área 0.0.0.0, com timers normais de hello e dead, e o roteador reconhece os roles dos vizinhos na rede, confirmando que a comunicação e o processo de eleição do DR estão funcionando como esperado. Isso tudo indica que o roteador está integrado corretamente na rede OSPF.

Roteador2 – Análise do Protocolo OSPF:



O roteador2 está participando da rede OSPF em múltiplas interfaces e áreas. Na tabela de roteamento, ele possui rotas diretamente conectadas para as redes 192.168.1.0/24, 192.168.2.0/24 e 192.168.3.0/24, além de rotas OSPF para a rede 192.168.4.0/24 que chegam por dois caminhos diferentes, indicando redundância e múltiplos links ativos. No vizinho OSPF, ele mantém adjacências completas com três vizinhos, incluindo um roteador com papel de Designated Router (DR) para as redes em que participa. Nas interfaces, eth0 está configurada com estado de Backup Designated Router (Backup DR), eth1 está no papel de Designated Router (DR) sem backup, e eth2 também está como Backup DR. A presença dessas funções de DR e Backup DR e a existência de adjacências completas indicam que a eleição e a comunicação entre roteadores OSPF estão funcionando corretamente, garantindo estabilidade e redundância na rede. Além disso, a configuração dos timers e o estado das interfaces indicam que a rede está operando normalmente.

Roteador3 – Análise do Protocolo OSPF:



No roteador3, a análise da tabela de roteamento mostra que a rota padrão está configurada como uma rota do kernel, apontando para o gateway 192.168.1.1 pela interface eth0. As redes diretamente conectadas, como 192.168.1.0/24, 192.168.3.0/24 e 192.168.4.0/24, estão ativas e presentes na tabela. Além disso, o protocolo OSPF está operando corretamente, pois as rotas para as redes 192.168.1.0/24, 192.168.2.0/24 e 192.168.3.0/24 são aprendidas via OSPF, incluindo a rede 192.168.2.0/24 que possui múltiplos caminhos, indicando redundância e balanceamento de carga entre as interfaces eth0 e eth1.

A verificação dos vizinhos OSPF mostra que o roteador3 mantém adjacências completas (estado Full) com os roteadores vizinhos identificados pelos IPs 192.168.1.10 e 192.168.1.20, sendo que este último aparece como backup em duas interfaces (eth0 e eth1), garantindo alta disponibilidade da rede.

Por fim, a análise das interfaces OSPF revela que as três interfaces do roteador (eth0, eth1 e eth2) estão ativas e configuradas para a área OSPF 0.0.0.0, com a função de Designated Router (DR) atribuída ao próprio roteador3 em todas as interfaces. A interface eth0, por exemplo, está com prioridade 1 e mantém um Backup Designated Router (BDR) no endereço 192.168.1.20. A interface eth2 não possui BDR configurado, indicando uma topologia possivelmente ponto a ponto ou com menos vizinhos nessa rede. Os timers de

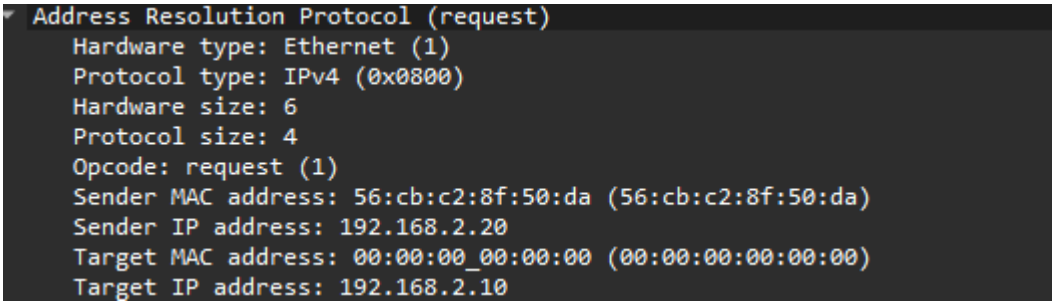
OSPF, como Hello Interval e Dead Interval, estão configurados para valores padrão, garantindo a estabilidade da comunicação entre os roteadores.

7 Questão - WireShark

Para realizar a análise do tráfego de rede nas sub-redes mencionadas na questão 04, utilizei a ferramenta tcpdump para capturar os pacotes que circulam na rede. O tcpdump é uma ferramenta de captura de pacotes que permite registrar o tráfego em tempo real, salvando os dados para posterior análise. Após a captura, utilizei o Wireshark, uma ferramenta gráfica poderosa para análise detalhada dos pacotes, que permite visualizar os diferentes protocolos, seus campos e valores de forma organizada e clara.

Com essas ferramentas, foi possível obter amostras representativas dos protocolos ARP, ICMP e IP presentes nas redes monitoradas. A análise dos pacotes capturados permitiu identificar requisições e respostas desses protocolos, entender a comunicação entre os dispositivos, o funcionamento do processo de resolução de endereços, bem como o diagnóstico de conectividade entre máquinas em diferentes sub-redes. A seguir, serão apresentadas as capturas realizadas e a explicação detalhada dos campos mais importantes dos cabeçalhos Ethernet, ARP, IP e ICMP presentes em cada tipo de pacote.

Protocolo ARP:



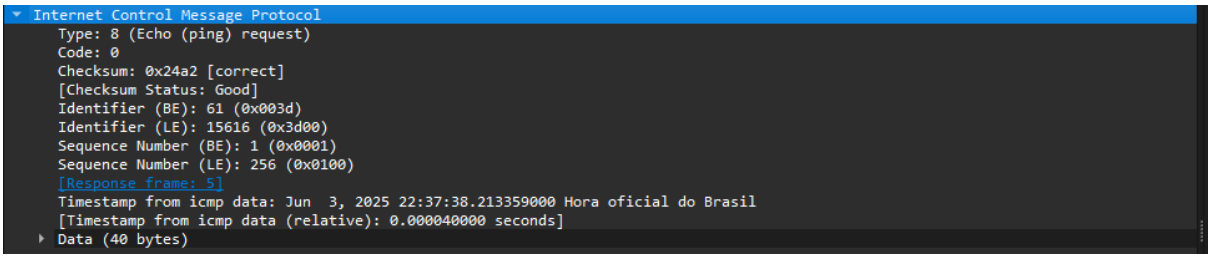
O quadro capturado possui 42 bytes e é um pacote do protocolo ARP (Address Resolution Protocol). A estrutura começa com o cabeçalho Ethernet, onde o endereço MAC de origem é 56:cb:c2:8f:50:da, que identifica o dispositivo que enviou o pacote. O endereço de destino é o broadcast (ff:ff:ff:ff:ff:ff), ou seja, o pacote é enviado para todos os dispositivos da rede local. O tipo de protocolo indicado no cabeçalho Ethernet é 0x0806, que corresponde ao protocolo ARP.

Dentro do protocolo ARP, este pacote é uma requisição, indicada pelo código de operação . O hardware utilizado é Ethernet, identificado pelo valor 1, e o protocolo é IPv4, indicado pelo valor 0x0800. O tamanho do endereço de hardware é 6 bytes (correspondente a um endereço MAC) e o tamanho do endereço de protocolo é 4 bytes (correspondente a um endereço IPv4).

No corpo do pacote, o endereço MAC do remetente é 56:cb:c2:8f:50:da e seu endereço IP é 192.168.2.20. O endereço MAC do destino está zerado (00:00:00:00:00:00) pois ainda não é conhecido — é justamente essa informação que o protocolo ARP busca obter. O endereço

IP alvo da requisição é 192.168.2.10. Ou seja, esse pacote é uma consulta enviada em broadcast para a rede local perguntando “Quem tem o IP 192.168.2.10?

Protocolo ICM:



Este quadro Ethernet tem 98 bytes e começa com o cabeçalho Ethernet, onde o endereço MAC de origem é 5e:20:49:a9:99:ef e o endereço MAC de destino é 56:cb:c2:8f:50:da, indicando uma comunicação direta entre esses dois dispositivos na rede local. O tipo de protocolo no cabeçalho Ethernet indica que o pacote contém um protocolo IP (IPv4).

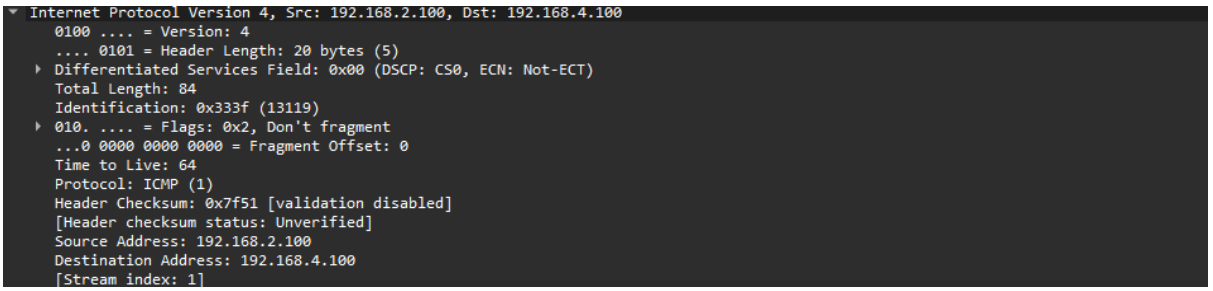
No cabeçalho IP, o pacote foi enviado do endereço 192.168.2.100 para o endereço 192.168.3.100. Isso mostra que a comunicação ocorre entre duas redes diferentes (lan2 e lan3, conforme suas sub-redes Docker). O protocolo transportado dentro do IP é o ICMP, que é usado para mensagens de controle e diagnóstico.

Dentro do ICMP, o pacote é um "Echo Request", que é basicamente um pedido de "ping" para testar a conectividade entre os hosts. O tipo 8 indica esse pedido, e o código 0 confirma que é um pedido padrão. O checksum está correto, garantindo a integridade do pacote. Os campos de identificador e número de sequência ajudam a rastrear solicitações e respostas correspondentes.

Além disso, o pacote contém 40 bytes de dados que normalmente são usados para medir o tempo de resposta, e há um timestamp indicando quando o ping foi enviado.

Resumindo, este pacote é um pedido de ping enviado pelo dispositivo com IP 192.168.2.100 para o dispositivo com IP 192.168.3.100, buscando verificar se o segundo está ativo e acessível na rede.

Protocolo Ip:



Este quadro Ethernet possui 98 bytes e apresenta o cabeçalho Ethernet, onde o endereço MAC de origem é 5e:20:49:a9:99:ef e o endereço MAC de destino é 56:cb:c2:8f:50:da, indicando uma comunicação direta entre dois dispositivos na rede local. O campo tipo mostra que o protocolo encapsulado é o IPv4.

No cabeçalho IPv4, observamos que a versão do protocolo é 4, e o comprimento do cabeçalho é de 20 bytes. O campo de serviços diferenciados está zerado, indicando que não há prioridade especial para esse pacote. O comprimento total do pacote IP é 84 bytes. A identificação do pacote é 0x333f, usada para distinguir fragmentos do mesmo pacote, se necessário.

O campo Flags indica “Don't fragment” (não fragmentar), o que significa que o pacote não pode ser dividido em fragmentos durante o roteamento. O deslocamento de fragmento está em zero, confirmando que esse não é um fragmento de pacote. O Time To Live (TTL) está em 64, limitando o número de saltos que o pacote pode fazer na rede para evitar loops.

O protocolo transportado pelo IP é o ICMP, identificado pelo número 1, utilizado para controle e diagnóstico de rede.

O endereço IP de origem é 192.168.2.100 e o destino é 192.168.4.100, indicando que o pacote está indo de um dispositivo na rede lan2 para um dispositivo na rede lan4.

No cabeçalho ICMP (não detalhado no trecho enviado), provavelmente está presente uma mensagem relacionada a diagnóstico, como um “ping” (echo request ou reply).

Anexos:

GITHUB: https://github.com/JorgeLuis8/SegundaAvalicaoRedes1
Vídeo no YouTube: https://youtu.be/s2UsmsNGVgA