Redes: Capa de red

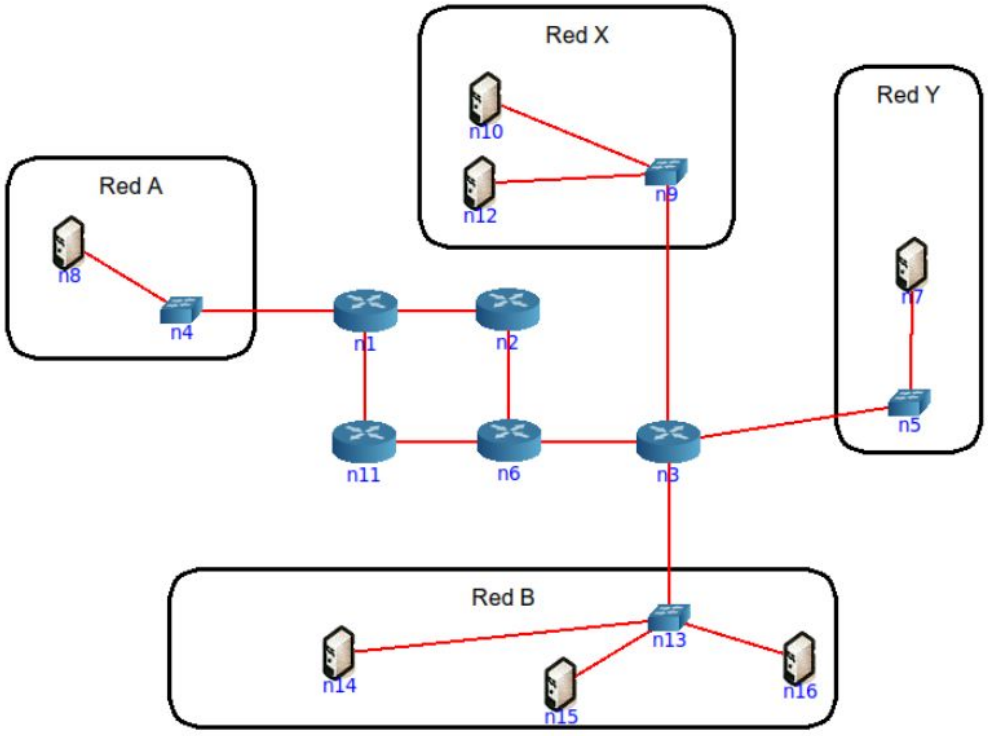
Grupo: 96

Autores
Valentín Colato 15655/7
Nicolas Cesar Champane Peñalva 15974/9
Nahuel Bigurrarena 15782/3

Practica 7

15. (Ejercicio de promoción) Utilizando la siguiente topología y el bloque IPv4 190.80.0.0/22, arme el plan de direccionamiento IPv4 teniendo en cuenta las siguientes restricciones:

NOTA: para quienes hagan la promoción, este será un ejercicio entregable. En la entrega deberán estar todas las preguntas respondidas y debidamente justificadas. En los puntos donde es necesario ejecutar comandos, los mismos deberán adjuntarse a la entrega



a. La red A tiene 70 hosts y se espera un crecimiento máximo de 20 hosts.
b. La red X tiene 150 hosts.
c. La red B cuenta con 20 hosts
d. La red Y tiene 35 hosts y se espera un crecimiento máximo de 30 hosts.
e. En cada red, se debe desperdiciar la menor cantidad de direcciones IP posibles. En este sentido, las redes utilizadas para conectar los routers deberán utilizar segmentos de red /30 de modo de desperdiciar la menor cantidad posible de direcciones IP

Bueno nuestro plan de direccionamiento va a ser generar las distintas subredes a partir de la dirección base 190.80.0.0/22 realizando subnetting. Empezaremos por asignarle una dirección a la **Red X** la cual necesita una subred con 150 host, por eso dividiremos la red con 8 bits de host para asignar una dirección que pueda soportar los 150 host necesarios.

Luego procederemos a generar una subred para la **Red A** , la cual tiene 70 host y espera un crecimiento máximo de 20 host , por ende crearemos una subred que soporte 90 host para ya estar preparado para el posible crecimiento. La crearemos con la siguiente dirección libre y utilizando solo 7 bits para los host. Luego necesitaremos para la **Red Y** 35 host pero como también se espera un crecimiento de 30 host como máximo, le asignaremos una red que tenga como mínimo 65 host, para esto necesitamos 7 bits para host en la dirección. Y por último la **Red B** necesita 20 host la cual le asignaremos una dirección con 5 bits para host. Por último con la primera dirección libre la dividiremos para crear subredes de 2 bits de host para asignarles direcciones a los 5 enlaces entre routers.

16. (Ejercicio de promoción) Asigne direcciones IP en los equipos de la topología según el plan anterior.

Arrancamos subneteando para la red que más host tiene: Red X.
Para referenciar a los 150 host de la Red X se precisan 8 bits (254) , entonces:
Aumentamos la máscara en 2 bits a la derecha para subnetear y que queden 8 bits para host para evitar desperdiciar hosts. Esto genera 4 subredes por el corrimiento de 2 bits (**color naranja**).

Dirección de red: 190.80.0.0 / 22 10111110.01010000.00000000.00000000
Mascara de red: 255.255.252.0 11111111.11111111.11111100.00000000
Mascara de subred: 255.255.255.0 11111111.11111111.11111111.00000000

Como solo quedan 2 bits para subredes (sólo se pueden direccionar 4 subredes con 2^8 -2 host) se asignará la primera subred a la Red X la cual utilizara 150 host y desperdiciaria 103, y las otras se volverán a dividir

190.80.0.0/24 para la red X
190.80.1.0/24 Libre
190.80.2.0/24 Libre
190.80.3.0/24 Libre

Subneteamos **190.80.1.0/24** para la siguiente red con más host: **Red A**.Entonces:
Aumentamos la máscara en un bit a la derecha para subnetear y que queden 7 bits para host para evitar desperdiciar hosts.Esto genera 2 subredes por el corrimiento de 1 bits (**color verde**).
Dirección de red: 190.80.1.0/24 10111110.01010000.00000001.00000000
Mascara de red: 255.255.255.0 11111111.11111111.11111111.00000000
Mascara de subred: 255.255.225.128 11111111.11111111.11111111.10000000

Queda 1 bits para subredes (sólo se pueden direccionar 2 subredes con 2^7 -2 host) se asignará la primera subred a la red A la cual utilizara 90 host y desperdiciaria 36, y las otra se volverán a dividir
190.80.1.0/25 para la red A

190.80.1.128/25 Libre

La siguiente red con más host: **Red Y**.
Como esta necesita también 7 bits para referenciar a los host ya que utilizara 65 host y desperdiciara 61 host. No se le podría asignar una con 6 bits porque con 6 solo tendríamos 62 host y no alcanzaría para los 65.
Por eso le asignamos la red que sobró de la última división el cual es una red con 7 bits para host.
190.80.1.128/25 para la **Red Y**

Nos queda la red B que necesita una red con 20 host ,
Para referenciar a los host de la red B se precisan 5 bits, entonces:
Agarramos la siguiente red libre que nos queda para dividir: **190.80.2.0/24**
Aumentamos la máscara en 3 bits a la derecha para subnetear y que queden 5 bits para host para evitar desperdiciar hosts. Esto genera 8 subredes por el corrimiento de 3 bits (**color azul**).
Dirección de red: 190.80.2.0/24 10111110.01010000.00000010.00000000
Mascara de red: 255.255.255.0 11111111.11111111.11111111.00000000
Mascara de subred: 255.255.225.224 11111111.11111111.11111111.11100000

Quedan 3 bits para subredes (sólo se pueden direccionar 8 subredes con 2^5 -2 host) se asignará la primera subred a la red B la cual utilizara 20 host y desperdiciaria 10, y las otras se volverán a dividir

190.80.2.0/27 para la red B
190.80.2.32/27 Libre
190.80.2.64/27 Libre
190.80.2.96/27 Libre
190.80.2.128/27 Libre
190.80.2.160/27 Libre
190.80.2.192/27 Libre
190.80.2.224/27 Libre

Ya terminamos de asignar direcciones a las distintas redes, ahora quedan asignar ips a las interfaces de routers.Para esto usaremos direcciones con sólo 2 bits de host para desperdiciar la menor cantidad posible de direcciones IP.
Usamos la primera red libre (**190.80.2.32/27**) y hacemos subnetting para llegar a solo 2 bits de host:
Aumentamos la máscara en 3 bits a la derecha para subnetear y que queden 2 bits para host para evitar desperdiciar hosts. Esto genera 8 subredes por el corrimiento de 3 bits (**color rojo**).
Dirección de red: 190.80.2.32/27 10111110.01010000.00000010.00100000
Mascara de red: 255.255.255.224 11111111.11111111.11111111.11100000
Mascara de subred: 255.255.225.252 11111111.11111111.11111111.11111100

Quedan 3 bits para subredes (sólo se pueden direccionar 8 subredes con 2^3 - 2 host)

Iremos asignando las direcciones ip a cada enlace entre los distintos routers

190.80.2.32/30 (Enlace entre n1-n2)
190.80.2.36/30 (Enlace entre n1-n11)
190.80.2.40/30 (Enlace entre n11-n6)
190.80.2.44/30 (Enlace entre n2-n6)
190.80.2.48/30 (Enlace entre n3-n6)
190.80.2.52/30 libre
190.80.2.56/30 libre
190.80.2.60/30 libre

Direcciones finales para las redes y enlaces

190.80.0.0/24 Para la Red X
190.80.1.0/25 Para la Red A
190.80.1.128/25 Para la Red Y
190.80.2.0/27 Para la Red B
190.80.2.32/30 Enlace entre n1-n2
190.80.2.36/30 Enlace entre n1-n11
190.80.2.40/30 Enlace entre n11-n6
190.80.2.44/30 Enlace entre n2-n6
190.80.2.48/30 Enlace entre n3-n6

Para mantener un mejor ordenado las direcciones disponibles, realizaremos **CIDR** con las direcciones libres

Libres:
190.80.2.52/30

190.80.2.56/29
190.80.2.56/30 0011 1|000
190.80.2.60/30 0011 1|100

190.80.2.64/26
190.80.2.64/27 **01|0**0 0000
190.80.2.96/27 **01|1**0 0000

190.80.2.128/25
190.80.2.128/27 **1|00**0 0000
190.80.2.160/27 **1|01**0 0000
190.80.2.192/27 **1|10**0 0000
190.80.2.224/27 **1|11**0 0000

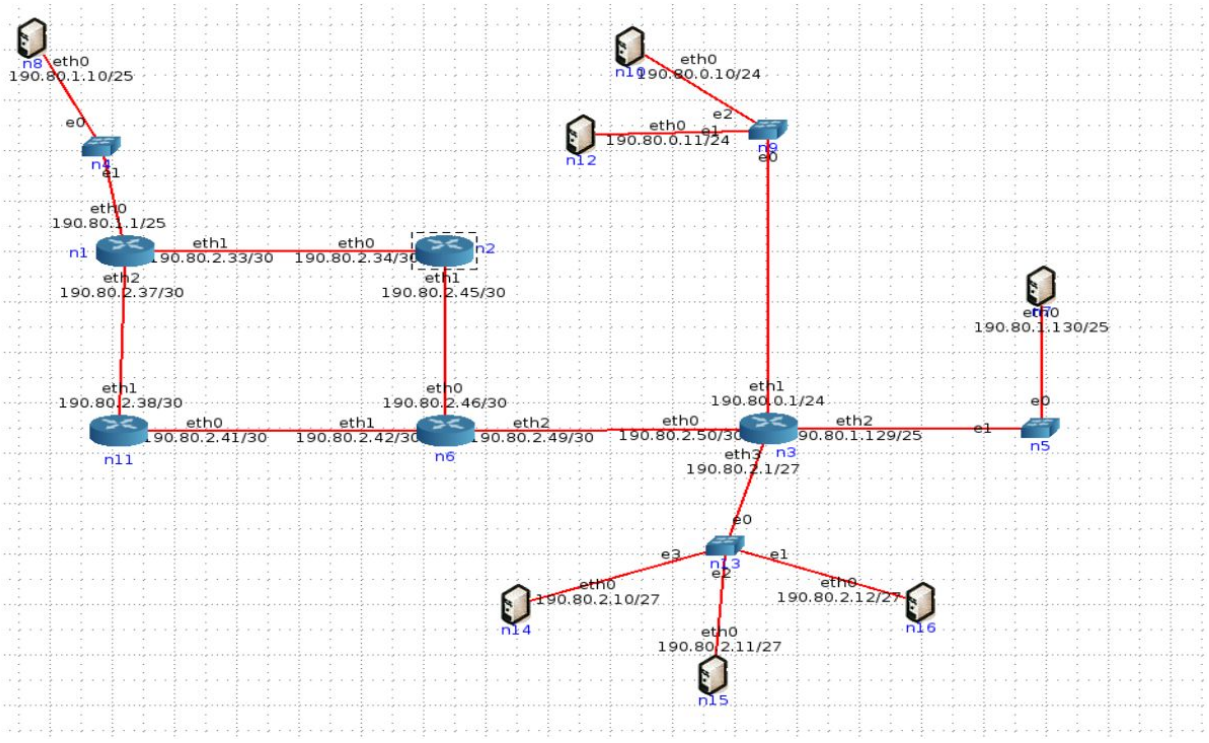
190.80.3.0/24

Practica 8

5. (Ejercicio de promoción) Para resolver este ejercicio utilice la topología del ejercicio de promoción de la anterior práctica, junto con la información de direccionamiento que le corresponda.
NOTA: para quienes hagan la promoción, este será un ejercicio entregable. En la entrega deberán estar todas las preguntas respondidas y debidamente justificadas. En los puntos donde es necesario ejecutar comandos, los mismos deberán adjuntarse a la entrega.

a. Antes de empezar el ejercicio ejecute en una terminal el siguiente comando: sudo iptables -P FORWARD ACCEPT

b. Arme en CORE la topología y configure las IPs de cada dispositivo según el plan generado. Realice la configuración utilizando alguno de los comandos vistos.
Para asignarle las IPs a los dispositivos utilizaremos el comando "ifconfig ethx xxx.xx.x.xx



Red A

- En **n8** ifconfig eth0 190.80.1.10/25

Red B

- En **n14** ifconfig eth0 190.80.2.10/27
- En **n15** ifconfig eth0 190.80.2.11/27
- En **n16** ifconfig eth0 190.80.2.12/27

Red Y

- En **n7** ifconfig eth0 190.80.1.130/25

Red X

- En **n10** ifconfig eth0 190.80.0.10/24
- En **n12** ifconfig eth0 190.80.0.11/24

Routers

- En **n1** ifconfig eth0 190.80.1.1/25
- En **n1** ifconfig eth1 190.80.2.33/30
- En **n1** ifconfig eth2 190.80.2.37/30

- En **n2** ifconfig eth0 190.80.2.34/30
- En **n2** ifconfig eth1 190.80.2.45/30

- En **n11** ifconfig eth0 190.80.2.41/30
- En **n11** ifconfig eth1 190.80.2.38/30

- En **n6** ifconfig eth0 190.80.2.46/30
- En **n6** ifconfig eth1 190.80.2.42/30
- En **n6** ifconfig eth2 190.80.2.49/30

- En **n3** ifconfig eth0 190.80.2.50/30
- En **n3** ifconfig eth1 190.80.0.1/24
- En **n3** ifconfig eth2 190.80.1.129/25
- En **n3** ifconfig eth3 190.80.2.1/27

c. Configure rutas por defecto para todas las PCs.

Para asignarle la ruta default de salida:
route add default gw {ip}

- En **n8** route add default gw 190.80.1.1
- En **n10** route add default gw 190.80.0.1
- En **n12** route add default gw 190.80.0.1
- En **n7** route add default gw 190.80.1.129
- En **n14** route add default gw 190.80.2.1
- En **n15** route add default gw 190.80.2.1
- En **n16** route add default gw 190.80.2.1

Con esto las PCs cuando no encuentre en la tabla de ruteo la ip destino entrara por esta entrada de default la cual le enviará el dato al router de su red. Esto funciona ya que se tiene como direccion 0.0.0.0 y mascara 0.0.0.0 por lo cual cualquier direccion que se le aplique la mascara 0.0.0.0 con un AND se quedara todo en 0.0.0.0 por lo cual coincidira con la direccion 0.0.0.0 y utiliza el gateway seleccionado.

d. Configure las tablas de ruteo de cada router en base al siguiente criterio:

Router n3 envía todo el tráfico desconocido a Router n6.
Router n6 envía todo el tráfico desconocido a Router n3.
Los demás routers no deberán utilizar rutas por defecto.
El tráfico entre Red A y red B deberá utilizar el enlace con n2.
El tráfico entre Red A, Red Y y Red X deberá utilizar el enlace con n11.

Para asignarle la ruta default de salida a los routers n6 y n3 utilizamos el siguiente comando:
route add default gw {ip}

- *En **n3** route add default gw* 190.80.2.49
- *En **n6** route add default gw* 190.80.2.50

Con esto generamos que cualquier mensaje que reciba n3 de la **Red X** ,**Y** , **B** y no conozca a dónde enviar el mensaje , se lo envíe por defecto a n6 . El cual va a hacer lo mismo que n3 , cualquier dirección que no coincida en su tabla de ruteo se lo enviará a n3. Generando así un ciclo en el que si llega un mensaje con una dirección en la cual n3 y n6 no conozcan se va a quedar dando saltos entre estos routers hasta que se termine su ttl y se envía un mensaje al origen con Time to live exceeded.

Respetaremos los mensajes que se envíen de la **Red A** hacia las distintas rutas considerando la consigna de que si es hacia la **Red B** pase por n2 y si es para la **Red X** o **Red Y** pase por n11.
Pero no se puede configurar los tráficos de regreso hacia la **Red A** de forma que discriminen si viene desde la **Red B**, **Red Y** o **Red X** para saber si al partir del router 6 se tiene que enviar hacia el router 11 o el router 2, esto se debe a que no se puede discriminar por la dirección de origen, solo por la de destino. Teniendo esto en cuenta debemos seleccionar una ruta para dirigir toda la información entre el Router 2 y 11. Nosotros vamos a hacer que se redirija los mensajes por n11 ya que se adaptaría más a la consigna/pedido ya que entre la **Red X** y **Red Y** hay muchos mas host que la **Red B**, por ende habría “más” posibles comunicaciones entre la **Red A** con las **Red X** y **Red Y** .

Por eso entonces en el n6 vamos a agregar a la tabla de ruteo la direccion 190.80.1.0/25 que lo envie el tráfico a n11 (190.80.2.41) con el siguiente comando:

```
route add -net 190.80.1.0/25 gw 190.80.2.41
```

Ahora en n11 agregamos al igual que en el paso anterior , la direccion 190.80.1.0/25 que lo envie a n1 (190.80.2.37) con el siguiente comando:

```
route add -net 190.80.1.0/25 gw 190.80.2.37
```

Con esto configuramos los mensajes de ida desde la **Red X , Y y B** hacia la A
Ahora en n1 configuremos para que se envíe a n11 el tráfico que sea para **Red X** y **Red Y,** para hacer que los mensajes de la **Red A** hacia estas redes pase si o si por el router 11.

```
route add -net 190.80.0.0/24 gw 190.80.2.38
route add -net 190.80.1.128/25 gw 190.80.2.38
```

Repetimos el mismo paso anterior para n11 en el cual se va a enviar todo el tráfico que sea para **Red X** o **Red Y** a el router 6 (190.80.2.42)

route add -net 190.80.0.0/24 gw 190.80.2.42

route add -net 190.80.1.128/25 gw 190.80.2.42

Con esto concluimos la conexión entre la **Red A** y las **Redes X** y **Y** , pasando el tráfico por n11

Ahora pasaremos a configurar el tráfico de la **Red A** hacia la **Red B** y que pase por n2
Configuraremos n1 para que el tráfico hacia la **Red B** lo envía a n2 con el comando:

route add -net 190.80.2.0/27 gw 190.80.2.34

Faltaria configurar n2 para que el tráfico hacia la **Red B** se envia a n6,usaremos el comando:

route add -net 190.80.2.0/27 gw 190.80.2.46

Con esto ya estaría configurado todas las tablas de ruteo pasando por los routers especificados.

e. Verifique conectividad entre las PCs, utilizando los comandos ping y traceroute. Asegúrese que los caminos que siguen los paquetes en cada caso son correctos. Una vez verificado lo anterior, tome capturas de pantalla de las tablas de ruteo de cada router; las mismas deben adjuntarse a la entrega.

Como decidimos pasar todo el tráfico de las redes X,Y,B hacia A que pase por n11 no agregamos en la tabla de ruteo de n2 la dirección que envíe a la Red A al router n1 .Para confirmar que llega a n2 el tráfico que se envía de la Red A hacia la B , la agregamos temporalmente con el comando:

```
root@n2:/tmp/pycore.46875/n2.conf# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
190.80.1.0      190.80.2.33     255.255.255.128 UG    0      0        0 eth0
190.80.2.0      190.80.2.46     255.255.255.224 UG    0      0        0 eth1
190.80.2.32     0.0.0.0         255.255.255.252 U     0      0        0 eth0
190.80.2.44     0.0.0.0         255.255.255.252 U     0      0        0 eth1
```

```
root@n8:/tmp/pycore.46875/n8.conf# traceroute 190.80.2.12
traceroute to 190.80.2.12 (190.80.2.12), 30 hops max, 60 byte packets
 1  190.80.1.1 (190.80.1.1)  0.052 ms  0.012 ms  0.011 ms
 2  190.80.2.34 (190.80.2.34)  0.050 ms  0.018 ms  0.015 ms
 3  190.80.2.42 (190.80.2.42)  0.032 ms  0.021 ms  0.021 ms
 4  190.80.2.50 (190.80.2.50)  0.035 ms  0.027 ms  0.027 ms
 5  190.80.2.12 (190.80.2.12)  0.047 ms  0.032 ms  0.032 ms
```

Con esto confirmamos que desde la red A hacia la red B pasa por 190.80.2.34 que es la dirección del router n2(eth0).

Y de la red A hacia la red X/Y :

```
root@n8:/tmp/pycore.46875/n8.conf# traceroute 190.80.1.130
traceroute to 190.80.1.130 (190.80.1.130), 30 hops max, 60 byte packets
 1  190.80.1.1 (190.80.1.1)  0.050 ms  0.014 ms  0.011 ms
 2  190.80.2.38 (190.80.2.38)  0.027 ms  0.016 ms  0.015 ms
 3  190.80.2.42 (190.80.2.42)  0.043 ms  0.025 ms  0.020 ms
 4  190.80.2.50 (190.80.2.50)  0.034 ms  0.035 ms  0.028 ms
 5  190.80.1.130 (190.80.1.130)  0.048 ms  0.034 ms  0.034 ms
root@n8:/tmp/pycore.46875/n8.conf# traceroute 190.80.0.10
traceroute to 190.80.0.10 (190.80.0.10), 30 hops max, 60 byte packets
 1  190.80.1.1 (190.80.1.1)  0.053 ms  0.014 ms  0.010 ms
 2  190.80.2.38 (190.80.2.38)  0.025 ms  0.017 ms  0.016 ms
 3  190.80.2.42 (190.80.2.42)  0.031 ms  0.022 ms  0.022 ms
 4  190.80.2.50 (190.80.2.50)  0.034 ms  0.028 ms  0.027 ms
 5  190.80.0.10 (190.80.0.10)  0.050 ms  0.036 ms  0.033 ms
```

Confirmamos que pasa por n11 ya que aparece la direccion 190.80.2.38 que eth1 de n11

Utilizamos el comando ping y traceroute para verificar todas las conectividades y funcionan

Tablas de ruteo:

N1:

```
root@n1:/tmp/pycore.46875/n1.conf# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
190.80.0.0      190.80.2.38     255.255.255.0   UG    0      0        0 eth2
190.80.1.0      0.0.0.0         255.255.255.128 U     0      0        0 eth0
190.80.1.128    190.80.2.38     255.255.255.128 UG    0      0        0 eth2
190.80.2.0      190.80.2.34     255.255.255.224 UG    0      0        0 eth1
190.80.2.32     0.0.0.0         255.255.255.252 U     0      0        0 eth1
190.80.2.36     0.0.0.0         255.255.255.252 U     0      0        0 eth2
root@n1:/tmp/pycore.46875/n1.conf#
```

N2:

```
root@n2:/tmp/pycore.46875/n2.conf# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
190.80.1.0      190.80.2.33     255.255.255.128 UG    0      0        0 eth0
190.80.2.0      190.80.2.46     255.255.255.224 UG    0      0        0 eth1
190.80.2.32     0.0.0.0         255.255.255.252 U     0      0        0 eth0
190.80.2.44     0.0.0.0         255.255.255.252 U     0      0        0 eth1
```

N11:

```
root@n11:/tmp/pycore.46875/n11.conf# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
190.80.0.0      190.80.2.42     255.255.255.0   UG    0      0        0 eth0
190.80.1.0      190.80.2.37     255.255.255.128 UG    0      0        0 eth1
190.80.1.128    190.80.2.42     255.255.255.128 UG    0      0        0 eth0
190.80.2.36     0.0.0.0         255.255.255.252 U     0      0        0 eth1
190.80.2.40     0.0.0.0         255.255.255.252 U     0      0        0 eth0
```

N6:

```
root@n6:/tmp/pycore.46875/n6.conf# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         190.80.2.50     0.0.0.0         UG    0      0        0 eth2
190.80.1.0      190.80.2.41     255.255.255.128 UG    0      0        0 eth1
190.80.2.40     0.0.0.0         255.255.255.252 U     0      0        0 eth1
190.80.2.44     0.0.0.0         255.255.255.252 U     0      0        0 eth0
190.80.2.48     0.0.0.0         255.255.255.252 U     0      0        0 eth2
```

N3:

```
root@n3:/tmp/pycore.46875/n3.conf# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         190.80.2.49     0.0.0.0         UG    0      0        0 eth0
190.80.0.0      0.0.0.0         255.255.255.0   U     0      0        0 eth1
190.80.1.128    0.0.0.0         255.255.255.128 U     0      0        0 eth2
190.80.2.0      0.0.0.0         255.255.255.224 U     0      0        0 eth3
190.80.2.48     0.0.0.0         255.255.255.252 U     0      0        0 eth0
```

f. Diagnóstico de red.

i. Desde la PC n8 realice un ping a una red que no exista en su topología. ¿Qué respuesta recibe y qué significa? ¿Quién le da la respuesta y por qué? ¿Qué protocolo se utiliza en la respuesta?

```
root@n8:/tmp/pycore.46875/n8.conf# traceroute 200.80.21.1
traceroute to 200.80.21.1 (200.80.21.1), 30 hops max, 60 byte packets
 1  190.80.1.1 (190.80.1.1)  0.045 ms !N  0.013 ms !N *
root@n8:/tmp/pycore.46875/n8.conf# ping 200.80.21.1
PING 200.80.21.1 (200.80.21.1) 56(84) bytes of data.
From 190.80.1.1 icmp_seq=1 Destination Net Unreachable
From 190.80.1.1 icmp_seq=2 Destination Net Unreachable
From 190.80.1.1 icmp_seq=3 Destination Net Unreachable
^C
--- 200.80.21.1 ping statistics ---
```

Realizamos el comando ping y traceroute desde la PC n8

ping 200.80.21.1 | traceroute 200.80.21.1

La respuesta fue Destination Net Unreacheable, la devolvio el router n1 porque al revisar en su tabla de ruteo no encuentra ninguna entrada que coincida con la red(ya que no tiene default),esto significa que no tiene una ruta para llegar a la red y devuelve un mensaje del protocolo ICMP diciendo que no la red es inalcanzable.

ii. Desde la PC n8 realice un ping a una IP de la red B que no esté asignada a un host. ¿Qué respuesta recibe y qué significa? ¿Quién le da la respuesta y por qué? ¿Qué protocolo se utiliza en la respuesta?

```
root@n8:/tmp/pycore.46875/n8.conf# ping 190.80.2.15
PING 190.80.2.15 (190.80.2.15) 56(84) bytes of data.
From 190.80.2.50 icmp_seq=1 Destination Host Unreachable
From 190.80.2.50 icmp_seq=2 Destination Host Unreachable
From 190.80.2.50 icmp_seq=3 Destination Host Unreachable
^C
--- 190.80.2.15 ping statistics ---
5 packets transmitted, 0 received, +3 errors, 100% packet loss, time 4016ms
pipe 3
root@n8:/tmp/pycore.46875/n8.conf# traceroute 190.80.2.15
traceroute to 190.80.2.15 (190.80.2.15), 30 hops max, 60 byte packets
 1  190.80.1.1 (190.80.1.1)  0.045 ms  0.012 ms  0.010 ms
 2  190.80.2.34 (190.80.2.34)  0.024 ms  0.015 ms  0.015 ms
 3  190.80.2.42 (190.80.2.42)  0.030 ms  0.020 ms  0.021 ms
 4  190.80.2.50 (190.80.2.50)  0.034 ms  0.027 ms  0.028 ms
 5  190.80.2.50 (190.80.2.50)  2996.710 ms !H  2996.676 ms !H  2996.655 ms !H
root@n8:/tmp/pycore.46875/n8.conf#
```

La petición fue pasando por los routers n1 luego n2, n6 y finalmente llegó hasta el router n3 donde no pudo encontrar el host el cual devolvió el mensaje de error “Host Unreachable” a través del protocolo ICMP indicando que el Host no se pudo alcanzar debido a que no se encuentra o no está disponible en la red.

(Ping utiliza el protocolo ICMP debido que es para el control de mensajes de internet, es parte del conjunto de protocolos IP. Se utiliza para enviar mensajes de error e información operativa, por ejemplo, que host no puede ser localizado entre otros.)

iii. Desde la PC n14 realice un ping a una red que no exista en su topología. ¿Qué respuesta recibe y qué significa? Explique por qué se produce esta situación. ¿Qué protocolo se utiliza en la respuesta?

```
root@n14:/tmp/pycore.46875/n14.conf# ping 200.1.2.3
PING 200.1.2.3 (200.1.2.3) 56(84) bytes of data.
From 190.80.2.49 icmp_seq=1 Time to live exceeded
From 190.80.2.49 icmp_seq=2 Time to live exceeded
From 190.80.2.49 icmp_seq=3 Time to live exceeded
From 190.80.2.49 icmp_seq=4 Time to live exceeded
^C
--- 200.1.2.3 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 2999ms
```

La petición fue enviada desde n14 a n3 el cual como no tiene la dirección en la tabla de ruteo lo envía por defecto a n6 , en n6 ocurre lo mismo pero esta vez se lo envía a n3. Generando así un ciclo donde cada vez que pasa por un router se decrementa el TTL por eso devuelve Time to live exceeded , esto lo envía n6.El TTL es la cantidad máxima de saltos que puede dar el datagrama hasta llegar a destino, si este llega a 0 el protocolo ICMP es el que le envía un mensaje de error a la dirección de origen.

iv. Compare lo que ocurre cuando hace un ping desde n8 a una red que no existe en la topología con lo que ocurre si el ping lo hace desde n14. ¿Por qué obtiene respuestas diferentes en ambos casos?

Se tienen respuestas distintas porque en la **Red B** , **Red X** y **Red Y** envían sus datagramas al router n3 el cual como tiene una ruta por defecto al route n6 (para aquellas dirección que no coinciden en su tabla de ruteo) y n6 tiene también como ruta por defecto al router n3 genera un ciclo que cualquier dirección que no conozca n3 o n6 se va a terminar quedando sin TTL porque se va a quedar yendo de n3 a n6 y de n6 a n3 tantas veces como tenga su TTL.
La diferencia cuando se hace desde la **Red A** es que como le envían los mensajes al router n1 como no encuentra la red en su tabla de ruteo y no tiene por defecto una red a la cual enviar el paquete, devuelve Destination Net Unreachable.

g. Mantenimiento de red.

i. Simule que se corta el enlace entre n1 y n2. Para hacerlo, elimine la conexión en la topología. Al ocurrir esto, parte de la red perderá conectividad. Solucione el problema modificando las tablas de ruteo que corresponda hasta que vuelva a tener conectividad completa. Al finalizar, realice una captura de pantalla de las tablas de ruteo que modificó y adjunte las mismas en la entrega.

Utilizamos el comando siguiente para simular que se corte el enlace entre n1 y n2:

En n1:

ifconfig eth1 down Para simular el corte entre n1 y n2

Luego agregamos la ruta para que todo el tráfico hacia B pase por n11 :

route add -net 190.80.2.0/27 gw 190.80.2.38

```
root@n1:/tmp/pycore.46875/n1.conf# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
190.80.0.0      190.80.2.38     255.255.255.0   UG    0      0        0 eth2
190.80.1.0      0.0.0.0         255.255.255.128 U     0      0        0 eth0
190.80.1.128    190.80.2.38     255.255.255.128 UG    0      0        0 eth2
190.80.2.0      190.80.2.38     255.255.255.224 UG    0      0        0 eth2
190.80.2.36     0.0.0.0         255.255.255.252 U     0      0        0 eth2
```

En n11:

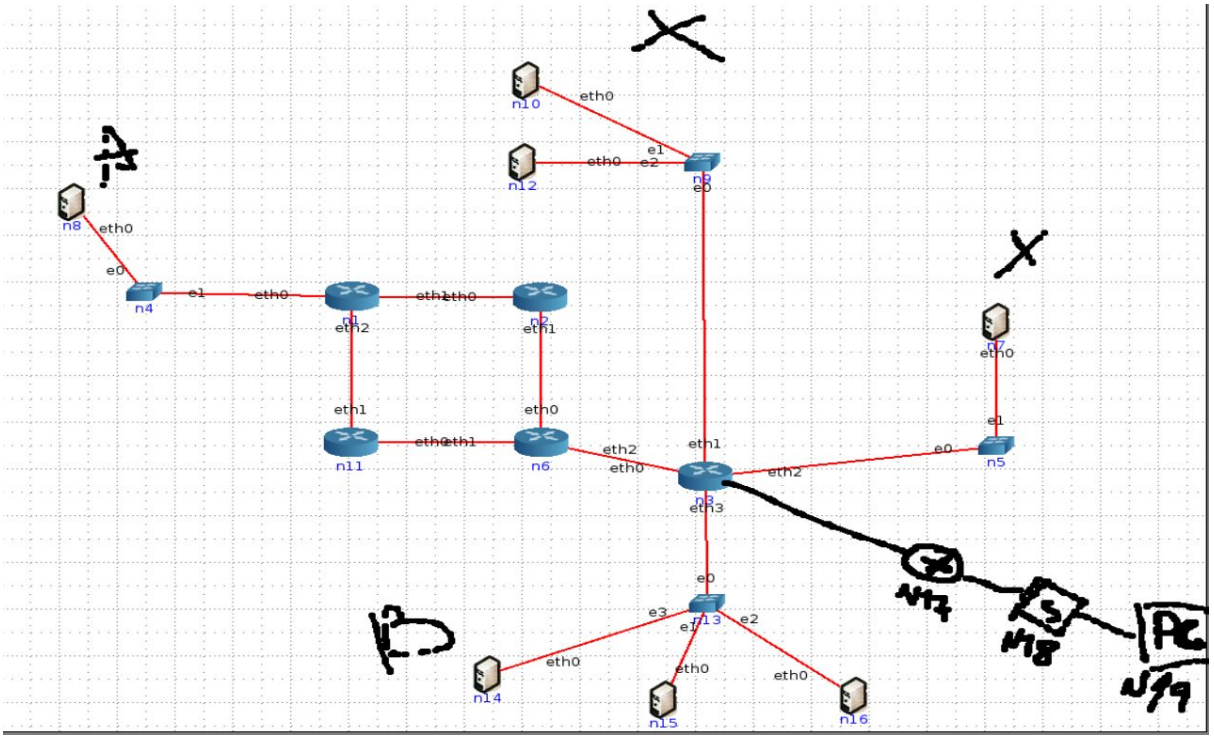
Agregamos la ruta para que todo el tráfico que llegue se redirija al router n6:

route add -net 190.80.2.0/27 gw 190.80.2.42

```
root@n11:/tmp/pycore.46875/n11.conf# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
190.80.0.0      190.80.2.42     255.255.255.0   UG    0      0        0 eth0
190.80.1.0      190.80.2.37     255.255.255.128 UG    0      0        0 eth1
190.80.1.128    190.80.2.42     255.255.255.128 UG    0      0        0 eth0
190.80.2.0      190.80.2.42     255.255.255.224 UG    0      0        0 eth0
190.80.2.36     0.0.0.0         255.255.255.252 U     0      0        0 eth1
190.80.2.40     0.0.0.0         255.255.255.252 U     0      0        0 eth0
```

Con esto todo el tráfico que se envíe de A hacia B va a pasar por n11 en vez de n2 ya que este enlace esta caido. Esto es debido a que el tráfico desde B hacia A antes ya pasaba por n11 por la decisión que tomamos al comienzo.

ii. La red se amplía. Por necesidades de su organización, se debe incorporar una nueva red. Para hacerlo, se conecta un router nuevo a n3 y, al router mencionado, se conectan un switch y una PC. Explique los cambios que debería realizar para que todos los dispositivos puedan comunicarse con la nueva red (no es necesario realizarlo en CORE. Si lo hace, considere que al apagar la topología perderá los cambios).



Primero se configuraría la dirección IP de la PC(n19) y la dirección por defecto que tendría que enviarse al router n17.

El paso siguiente será configurar el router n17 para que todo direccion de red desconocida sea redirigida por defecto al router n3.

Agregaremos una nueva dirección en el router n3 para que todo tráfico que tiene que ser enviado para la nueva red se envíe al router n17. De esta forma hasta el momento estan conectadas las redes X,B,Y con la nueva red.

Luego se debería configurar para conectar con la red A.

Procederemos a agregar en n1 la nueva red en la tabla de ruteo para que sea enviado a n11 y en esta última también agregar que todo el tráfico hacia la nueva red sea enviado a n6. De esta forma estaría conectado la red A con la red nueva.

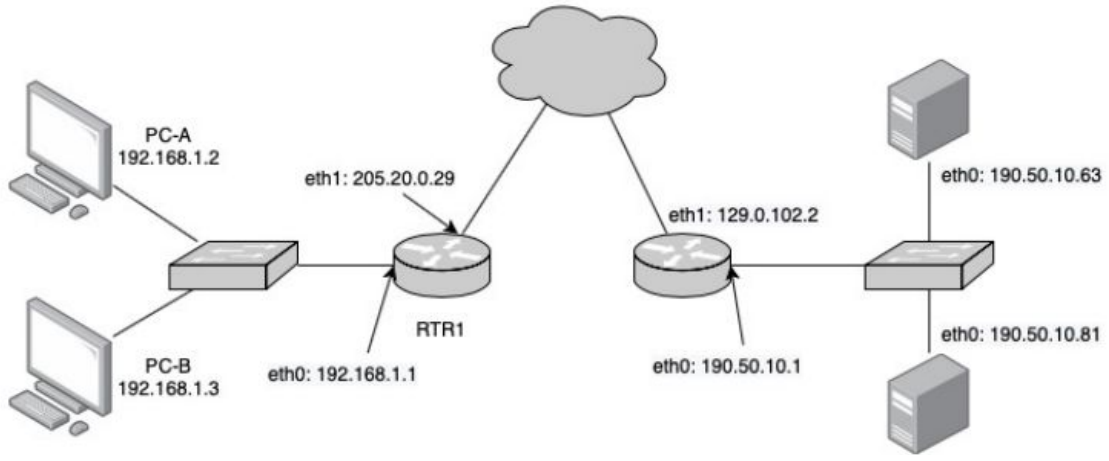
iii. En base a los cambios realizados en los dos puntos anteriores se puede ver que es compleja la gestión de una red con pocos dispositivos cuando ocurre el menor cambio. Incluso, en el caso de un corte en un enlace, hasta que un administrador no interviene algunas redes pierden conectividad a pesar de existir un camino que las une. Indique de qué forma podría simplificar la gestión y minimizar los cortes de servicio.

La forma para simplificar la gestión y minimizar los cortes de servicio sería utilizando el ruteo dinámico ya que en contraposición con el método estático, el ruteo dinámico utiliza diferentes protocolos cuyo fin es el de intercambiar rutas entre dispositivos intermedios con el objetivo de tener una red totalmente accesible. En este caso, los routers envían y reciben información de enrutamiento que utilizan para armar sus tablas de ruteo. Por esto mismo nos ahorramos tener que configurar esas modificaciones manualmente. Las tablas de ruteo son construidas a través de algoritmos e información relativa de la red, teniendo en cuenta a la hora de construir las tablas los conteo de saltos,ancho de banda, carga de la red y así elegir el mejor camino.

Una de las ventajas para facilitar la gestión es que permite configurar el ruteo en una red mediana a grande implica mucho menos trabajo para el administrador, a la vez que permite que la red completa se ponga en funcionamiento en un tiempo mucho menor.
Si nos referimos a errores como cuando se corta un enlace como pudimos apreciar en el ejercicio anterior es capaz de adaptarse, ya que puede detectar la falla de un enlace principal y utilizar entonces un enlace alternativo para alcanzar el destino (si lo hubiera).

Las desventajas son que, al intercambiar información entre los dispositivos y requerir que cada router procese dicha información se utiliza tanto ancho de banda de los enlaces como tiempo de procesamiento en los equipos, lo cual en algunas circunstancias puede convertirse en un problema. Adicionalmente, dependiendo del protocolo que se utilice, el enrutamiento dinámico requiere un mayor conocimiento por parte del administrador, tanto para configurarlo de forma correcta como para solucionar problemas.

10. (Ejercicio de promoción) Resuelva las consignas que se dan a continuación. NOTA: para quienes hagan la promoción, este será un ejercicio entregable. En la entrega deberán estar todas las preguntas respondidas y debidamente justificadas. En los puntos donde es necesario ejecutar comandos, los mismos deberán adjuntarse a la entrega. a. En base a la siguiente topología y a las tablas que se muestran, complete los datos que faltan.



PC-A (ss)

Local Address:Port	Peer Address:Port
192.168.1.2:49273	***190.50.10.63:80***
192.168.1.2:37484	190.50.10.63:25
192.168.1.2:***51238***	190.50.10.81:8080

PC-B (ss)

Local Address:Port	Peer Address:Port
192.168.1.3:52734	***190.50.10.81:8081***
192.168.1.3:39275	***190.50.10.81:8080***

RTR-1 (Tabla de NAT)

Lado LAN	Lado WAN
192.168.1.2:49273	205.20.0.29:25192
192.168.1.2:51238	***205.20.0.29:16345***
192.168.1.3:52734	205.20.0.29:51091
192.168.1.2:37484	205.20.0.29:41823
192.168.1.3:39275	205.20.0.29:9123

SRV-A (ss)

Local Address:Port	Peer Address:Port
190.50.10.63:80	205.20.0.29:25192
190.50.10.63:25	205.20.0.29:41823

SRV-B (ss)

Local Address:Port	Peer Address:Port
190.50.10.81:8080	205.20.0.29:16345
190.50.10.81:8081	205.20.0.29:51091
190.50.10.81:8080	205.20.0.29:9123

b. En base a lo anterior, responda:

i. ¿Cuántas conexiones establecidas hay y entre qué dispositivos?

Hay 5 conexiones establecidas:

PC A |205.20.0.29:25192	<--->	190.50.10.63:80	SRV-A
PC A |205.20.0.29:16345	<--->	190.50.10.81:8080	SRV-B
PC A |205.20.0.29:41823	<--->	190.50.10.63:25	SRV-A
PC B |205.20.0.29:51091	<--->	190.50.10.81:8081	SRV-B
PC B |205.20.0.29:9123	<--->	190.50.10.81:8080	SRV-B

ii. ¿Quién inició cada una de las conexiones? ¿Podrían haberse iniciado en sentido inverso? ¿Por qué? Investigue qué es port forwarding y si serviría como solución en este caso.

Las conexiones las iniciaron las PCs porque no se podría iniciar una conexión desde afuera ya que la red está utilizando NAPT lo cual produce que cada PC tenga una dirección privada y se comunique hacia afuera con única dirección pública a través de diferentes puertos. No se podría iniciar desde afuera ya que no se sabría la dirección exacta de la pc porque es privada. Y no estaría en la tabla de nat la transformación de la ip pública a la ip privada.

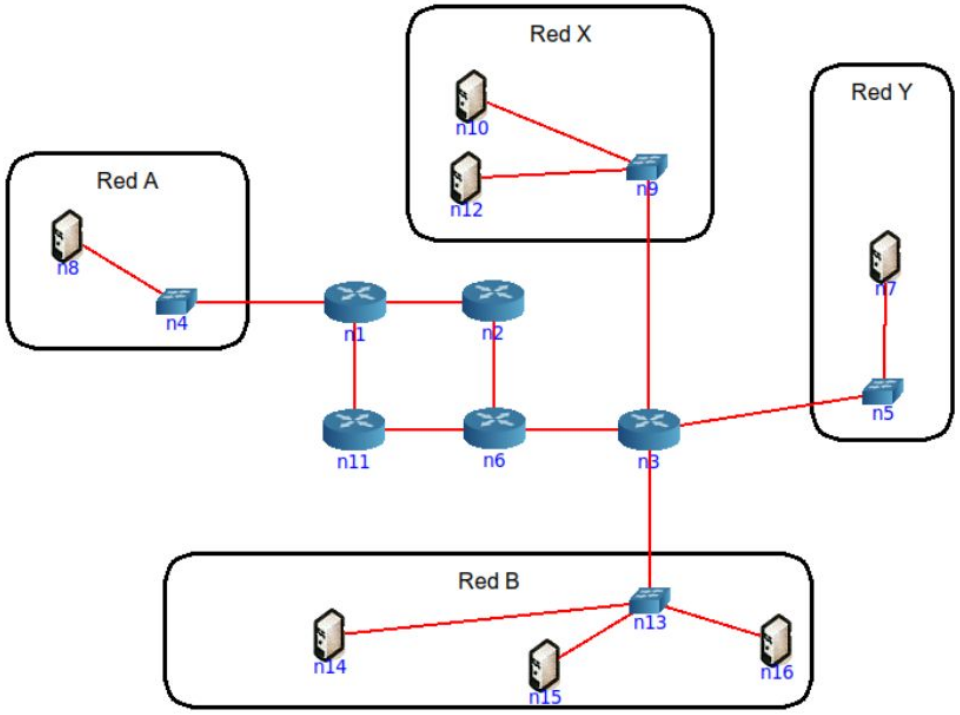
La solución a este problema podría ser ”port forwarding” debido a que se encarga del redireccionamiento de puertos, es decir la acción de redirigir el tráfico ingresado a un puerto específico a un host con ip privada. En otras palabras, el host con ip privada estaría “escuchando” en X puerto con la ip publica del router, este le enviaría todo lo que llegue a este puerto. Esta técnica lo que permite es poder tener servicios en una red privada accesibles desde “afuera”, en otras palabras una PC/Servidor desde el exterior tendrá acceso a un puerto en una dirección IP privada de la red LAN.

Practica 9

13. (Ejercicio de promoción) Responda y justifique los siguientes ejercicios.

NOTA: para quienes hagan la promoción, este será un ejercicio entregable. En la entrega deberán estar todas las preguntas respondidas y debidamente justificadas. En los puntos donde es necesario ejecutar comandos, los mismos deberán adjuntarse a la entrega.

a. Sobre la topología trabajada en las prácticas anteriores, utilizando el direccionamiento establecido y asumiendo que en la misma la conectividad IPv4 está dada de acuerdo a las pautas de la práctica anterior, realice las siguientes configuraciones:



i. Configure direcciones IPv6 y GW en los dispositivos conectados a las redes A, B, X e Y. Para ello, utilice los siguientes bloques IPv6:

Como tenemos la dirección de ipv6: **2001:db8:1a32::/48**
La transformaremos a como esta en el enunciado a redes de /64.
Esto lo haremos tomando 8 octetos. **2001:db8:1a32:0::/64**
Y la repartiremos para las distintas redes.

Red A: 2001:db8:1a32:0::/64
Red B: 2001:db8:1a32:1::/64
Red X: 2001:db8:1a32:2::/64
Red Y: 2001:db8:1a32:3::/64

NOTA: Los routers n2, n6 y n11 no deben tener direcciones IPv6

Comandos usados:

- **En N8:**
  - **ip -6 addr add 2001:DB8:1A32:0::2/64 dev eth0**
- **EN N1:**
  - **ip -6 addr add 2001:DB8:1A32:0::1/64 dev eth0**
- **En N3**
  - **ip -6 addr add 2001:DB8:1A32:1::1/64 dev eth3**
  - **ip -6 addr add 2001:DB8:1A32:2::1/64 dev eth1**
  - **ip -6 addr add 2001:DB8:1A32:3::1/64 dev eth2**
- **En N10**
  - **ip -6 addr add 2001:DB8:1A32:2::2/64 dev eth0**
- **En N12**
  - **ip -6 addr add 2001:DB8:1A32:2::3/64 dev eth0**
- **En N7**
  - **ip -6 addr add 2001:DB8:1A32:3::2/64 dev eth0**
- **En N14**
  - **ip -6 addr add 2001:DB8:1A32:1::2/64 dev eth0**
- **En N15**
  - **ip -6 addr add 2001:DB8:1A32:1::3/64 dev eth0**
- **En N16**
  - **ip -6 addr add 2001:DB8:1A32:1::4/64 dev eth0**

Default:

- **En n14**
  - **ip -6 route add default via 2001:db8:1a32:1::1**
- **En n15**
  - **ip -6 route add default via 2001:db8:1a32:1::1**
- **En n16**
  - **ip -6 route add default via 2001:db8:1a32:1::1**
- **En n7**
  - **ip -6 route add default via 2001:db8:1a32:3::1**
- **En n10**
  - **ip -6 route add default via 2001:db8:1a32:2::1**
- **En n12**
  - **ip -6 route add default via 2001:db8:1a32:2::1**
- **En n8**
  - **ip -6 route add default via 2001:db8:1a32:0::1**

ii. Verifique la conectividad IPv6 nativa entre los hosts de las redes B, X e Y

Probamos conectividad entre el router n14 y n7 **(Red B con Red Y)**:

```
root@n14:/tmp/pycore.38863/n14.conf# ping6 2001:db8:1a32:3::2
PING 2001:db8:1a32:3::2(2001:db8:1a32:3::2) 56 data bytes
64 bytes from 2001:db8:1a32:3::2: icmp_seq=1 ttl=63 time=0.051 ms
64 bytes from 2001:db8:1a32:3::2: icmp_seq=2 ttl=63 time=0.057 ms
64 bytes from 2001:db8:1a32:3::2: icmp_seq=3 ttl=63 time=0.050 ms
64 bytes from 2001:db8:1a32:3::2: icmp_seq=4 ttl=63 time=0.057 ms
^C
--- 2001:db8:1a32:3::2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms
rtt min/avg/max/mdev = 0.050/0.053/0.057/0.009 ms
root@n14:/tmp/pycore.38863/n14.conf#
```

Probamos conectividad entre el router n14 y n10**(Red B con Red X)**:

```
root@n14:/tmp/pycore.38863/n14.conf# ping6 2001:db8:1a32:2::2
PING 2001:db8:1a32:2::2(2001:db8:1a32:2::2) 56 data bytes
64 bytes from 2001:db8:1a32:2::2: icmp_seq=1 ttl=63 time=0.049 ms
64 bytes from 2001:db8:1a32:2::2: icmp_seq=2 ttl=63 time=0.056 ms
64 bytes from 2001:db8:1a32:2::2: icmp_seq=3 ttl=63 time=0.052 ms
^C
--- 2001:db8:1a32:2::2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 0.049/0.052/0.056/0.006 ms
root@n14:/tmp/pycore.38863/n14.conf#
```

Probamos la conectividad entre el router n17 y n12 **(Red Y con Red X)**:

```
root@n7:/tmp/pycore.38863/n7.conf# ping6 2001:db8:1a32:2::3
PING 2001:db8:1a32:2::3(2001:db8:1a32:2::3) 56 data bytes
64 bytes from 2001:db8:1a32:2::3: icmp_seq=1 ttl=63 time=0.053 ms
64 bytes from 2001:db8:1a32:2::3: icmp_seq=2 ttl=63 time=0.046 ms
64 bytes from 2001:db8:1a32:2::3: icmp_seq=3 ttl=63 time=0.049 ms
^C
--- 2001:db8:1a32:2::3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.046/0.049/0.053/0.006 ms
root@n7:/tmp/pycore.38863/n7.conf#
```

iii. Configure un tunel GRE entre los routers n1 y n3 de modo de poder tunelear el tráfico IPv6 hacia y desde la Red A. Para ello utilice los siguientes comandos:

 # Establecer el tunel (en ambos extremos)
 # ip tunnel add <nombre_de_la_interfaz> mode gre remote <ip_otro_extremo_del_tunel> \ local <ip_mi_extremo_del_tunel> ttl 255
ip tunnel add mi_tunel mode gre remote 2.2.2.2 local 1.1.1.1 ttl 255
 # Prender la interfaz
 # ip link set <nombre_de_la_interfaz> up
ip link set mi_tunel up
 # Rutear tráfico a través del tunel
 # ip -6 route add <IPv6_network>/<mask> dev <nombre_de_la_interfaz>
ip -6 route add 2010:55::/64 dev mi_tunel

Antes de hacer el túnel , los routers tienen que saber cómo llegar desde la interfaz de n1 a la interfaz de n3 y viceversa (interfaces seleccionadas en el túnel) en ipv4, para eso ejecutamos los siguientes comandos para agregar en las tablas de ruteo el direccionamiento correcto.

N1

route add -net 190.80.2.48/30 gw 190.80.2.38

N11

route add -net 190.80.2.48/30 gw 190.80.2.42

N6

route add -net 190.80.2.36/30 gw 190.80.2.41

-El siguiente paso fue configurar el túnel gre a través de los siguientes comandos:

N1

ip tunnel add tunnel_n1_n3 mode gre remote 190.80.2.50 local 190.80.2.37 ttl 255
ip link set tunnel_n1_n3 up
ip -6 route add 2001:db8:1a32:1::/64 dev tunnel_n1_n3
ip -6 route add 2001:db8:1a32:2::/64 dev tunnel_n1_n3
ip -6 route add 2001:db8:1a32:3::/64 dev tunnel_n1_n3

N3

ip tunnel add tunnel_n1_n3 mode gre remote 190.80.2.37 local 190.80.2.50 ttl 255
ip link set tunnel_n1_n3 up
ip -6 route add 2001:db8:1a32::/64 dev tunnel_n1_n3

iv. Verifique la conectividad IPv6 entre los hosts de las redes A, B, X e Y.

Red A con Red B(N8 y N14)

```
root@n8:/tmp/pycore.45227/n8.conf# ping6 2001:db8:1a32:1::2
PING 2001:db8:1a32:1::2(2001:db8:1a32:1::2) 56 data bytes
64 bytes from 2001:db8:1a32:1::2: icmp_seq=1 ttl=62 time=0.098 ms
64 bytes from 2001:db8:1a32:1::2: icmp_seq=2 ttl=62 time=0.110 ms
^C
--- 2001:db8:1a32:1::2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 999ms
rtt min/avg/max/mdev = 0.098/0.104/0.110/0.006 ms
root@n8:/tmp/pycore.45227/n8.conf#
```

Red A con Red Y (N8 y N7)

```
root@n8:/tmp/pycore.45227/n8.conf# ping6 2001:db8:1a32:3::2
PING 2001:db8:1a32:3::2(2001:db8:1a32:3::2) 56 data bytes
64 bytes from 2001:db8:1a32:3::2: icmp_seq=1 ttl=62 time=0.147 ms
64 bytes from 2001:db8:1a32:3::2: icmp_seq=2 ttl=62 time=0.121 ms
64 bytes from 2001:db8:1a32:3::2: icmp_seq=3 ttl=62 time=0.138 ms
^C
--- 2001:db8:1a32:3::2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.121/0.135/0.147/0.014 ms
root@n8:/tmp/pycore.45227/n8.conf#
```

Red A con Red X(N8 y N10)

```
root@n8:/tmp/pycore.45227/n8.conf# ping6 2001:db8:1a32:2::2
PING 2001:db8:1a32:2::2(2001:db8:1a32:2::2) 56 data bytes
64 bytes from 2001:db8:1a32:2::2: icmp_seq=1 ttl=62 time=0.152 ms
64 bytes from 2001:db8:1a32:2::2: icmp_seq=2 ttl=62 time=0.115 ms
64 bytes from 2001:db8:1a32:2::2: icmp_seq=3 ttl=62 time=0.113 ms
^C
--- 2001:db8:1a32:2::2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.113/0.126/0.152/0.022 ms
root@n8:/tmp/pycore.45227/n8.conf#
```

La conectividad como se puede apreciar a través de los comandos realizados fue exitosa.

v. Evalúe y analice las diferencias observadas entre un traceroute IPv4 y un traceroute IPv6 respecto de:

a) Cantidad de saltos que la traza nos muestra en cada caso. ¿esperaba otra respuesta? ¿por qué?

Red A con Red X(N8 y N10)

IPV6:

```
root@n8:/tmp/pycore.45227/n8.conf# traceroute6 2001:db8:1a32:2::2
traceroute to 2001:db8:1a32:2::2 (2001:db8:1a32:2::2), 30 hops max, 80 byte pack
ets
 1  2001:db8:1a32::1 (2001:db8:1a32::1)  0.047 ms  0.008 ms  0.007 ms
 2  2001:db8:1a32:1::1 (2001:db8:1a32:1::1)  0.076 ms  0.029 ms  0.027 ms
 3  2001:db8:1a32:2::2 (2001:db8:1a32:2::2)  0.035 ms  0.027 ms  0.026 ms
```

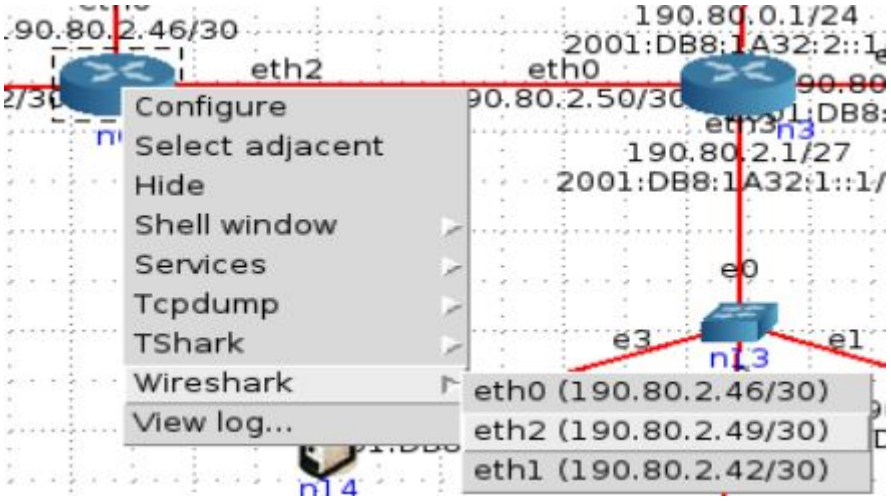
IPV4:

```
root@n8:/tmp/pycore.45227/n8.conf# traceroute 190.80.0.10
traceroute to 190.80.0.10 (190.80.0.10), 30 hops max, 60 byte packets
 1  190.80.1.1 (190.80.1.1)  0.048 ms  0.012 ms  0.009 ms
 2  190.80.2.38 (190.80.2.38)  0.023 ms  0.015 ms  0.014 ms
 3  190.80.2.42 (190.80.2.42)  0.053 ms  0.021 ms  0.020 ms
 4  190.80.2.50 (190.80.2.50)  0.035 ms  0.026 ms  0.025 ms
 5  190.80.0.10 (190.80.0.10)  0.058 ms  0.034 ms  0.032 ms
```

En ipv6 vemos que realizar 3 saltos nada más , comparado contra ipv4 que realiza 5 saltos.La diferencia esta en el túnel ya que ignora las ip's intermedias tipo ipv4 porque el router es el que encapsula la ipv6 en ipv4 creando un nuevo datagrama con origen n1 hasta n3 reiniciando el TTL, cuando se desempaqueta en n3 ahí se decrementa el ttl del mensaje ipv6 y recien ahi volvería el mensaje de ICMP al origen N8, por esto no se registra los routers intermedios.

b) Capture el tráfico entre en3 y n6 durante el traceroute IPv6 para observar el valor asociado en el header GRE al protocolo encapsulado.

Utilizamos la herramienta de Core para capturar el tráfico con Wireshark solo para una interfaz



1	0.000000000	2001:db8:1a32::2	2001:db8:1a32:1::2	UDP	118	Source port: 46522 Destination port: 33437
2	0.000033000	2001:db8:1a32:1::1	2001:db8:1a32::2	ICMPv6	166	Time Exceeded (hop limit exceeded in transit)
3	0.000089000	2001:db8:1a32::2	2001:db8:1a32:1::2	UDP	118	Source port: 52782 Destination port: 33438
4	0.000122000	2001:db8:1a32:1::1	2001:db8:1a32::2	ICMPv6	166	Time Exceeded (hop limit exceeded in transit)
5	0.000165000	2001:db8:1a32::2	2001:db8:1a32:1::2	UDP	118	Source port: 39466 Destination port: 33439
6	0.000179000	2001:db8:1a32:1::1	2001:db8:1a32::2	ICMPv6	166	Time Exceeded (hop limit exceeded in transit)
7	0.000217000	2001:db8:1a32::2	2001:db8:1a32:1::2	UDP	118	Source port: 47763 Destination port: 33440
8	0.000243000	2001:db8:1a32:1::2	2001:db8:1a32::2	ICMPv6	166	Destination Unreachable (Port unreachable)
9	0.000281000	2001:db8:1a32::2	2001:db8:1a32:1::2	UDP	118	Source port: 53235 Destination port: 33441
10	0.000295000	2001:db8:1a32:1::2	2001:db8:1a32::2	ICMPv6	166	Destination Unreachable (Port unreachable)
11	0.000333000	2001:db8:1a32::2	2001:db8:1a32:1::2	UDP	118	Source port: 33227 Destination port: 33442
12	0.000346000	2001:db8:1a32:1::2	2001:db8:1a32::2	ICMPv6	166	Destination Unreachable (Port unreachable)
13	0.000383000	2001:db8:1a32::2	2001:db8:1a32:1::2	UDP	118	Source port: 50885 Destination port: 33443
14	0.000397000	2001:db8:1a32:1::2	2001:db8:1a32::2	ICMPv6	166	Destination Unreachable (Port unreachable)
15	0.000433000	2001:db8:1a32::2	2001:db8:1a32:1::2	UDP	118	Source port: 45033 Destination port: 33444
16	0.000447000	2001:db8:1a32:1::2	2001:db8:1a32::2	ICMPv6	166	Destination Unreachable (Port unreachable)
17	0.000483000	2001:db8:1a32::2	2001:db8:1a32:1::2	UDP	118	Source port: 43015 Destination port: 33445
18	0.000497000	2001:db8:1a32:1::2	2001:db8:1a32::2	ICMPv6	166	Destination Unreachable (Port unreachable)
19	0.000534000	2001:db8:1a32::2	2001:db8:1a32:1::2	UDP	118	Source port: 46342 Destination port: 33446
20	0.000569000	2001:db8:1a32::2	2001:db8:1a32:1::2	UDP	118	Source port: 47618 Destination port: 33447
21	0.000605000	2001:db8:1a32::2	2001:db8:1a32:1::2	UDP	118	Source port: 42579 Destination port: 33448
22	0.000695000	2001:db8:1a32::2	2001:db8:1a32:1::2	UDP	118	Source port: 45535 Destination port: 33449

```
▾ Generic Routing Encapsulation (IPv6)
  ▾ Flags and Version: 0x0000
      0... .... .... .... = Checksum Bit: No
      .0.. .... .... .... = Routing Bit: No
      ..0. .... .... .... = Key Bit: No
      ...0 .... .... .... = Sequence Number Bit: No
      .... 0... .... .... = Strict Source Route Bit: No
      .... .000 .... .... = Recursion control: 0
      .... .... 0000 0... = Flags (Reserved): 0
      .... .... .... .000 = Version: GRE (0)
    Protocol Type: IPv6 (0x86dd)
```

Se encuentra en el encabezado GRE , “0x86DD” que significa que encapsula el protocolo IPv6. Este campo es importante ya que el GRE podría encapsular otros tipos de protocolos y no únicamente ipv6.