

**UNIVERSIDAD SAN CARLOS DE GUATEMALA**

**CENTRO UNIVERSITARIO DE OCCIDENTE**

**-CUNOC-**



**MANUAL GENERAL PROYECTO REDES DE COMPUTADORAS 1**

Astrid Gabriela Martinez Castillo 201731318

Celia Esmeralda Vargas Lopez 201730930

Práctica Intermedia TI.

Ing. Francisco Rojas.

Quetzaltenango, 11 de mayo de 2021



## ÍNDICE

Introducción general.....	2
Marco Teórico.....	3
Arquitectura de la Red.....	12
Configuración de Servidor con OpenVPN.....	15
Configuración de Cliente.....	17
Configuración de las Redes.....	19
Restricciones.....	31



## Introducción

En el presente informe se desarrollaran las diferentes instrucciones que se ejecutaron para la creación de una red específica, presentando los conceptos básicos que se deben conocer para realizarla y de la manera mejor detallada cada paso, tanto como para la creación de un servidor vpn, un cliente así como de 4 redes conectadas a un router, cabe resaltar que cada red tiene distintas restricciones de acceso con otras redes.

## Marco Teórico

### SERVIDOR Y CLIENTE VPN

#### ¿Qué es una VPN?

Una VPN (Red Privada Virtual) es un servicio que se puede utilizar cuando se conecta a internet. Protege su tráfico y datos en línea, asegurándose de que disfrute de la navegación privada y del acceso sin restricciones a cualquier contenido en línea que desee.

#### ¿Qué es el software de cliente VPN?


Un cliente VPN es el software de red privada virtual que se instala en su dispositivo para establecer una conexión entre él y el servidor VPN y ofrecer acceso a los servicios VPN.

La mayoría de las plataformas populares como Windows, macOS, iOS y Android vienen con software de cliente VPN preinstalado, pero también puede instalar clientes de terceros para disfrutar de una mejor interfaz de usuario y más funciones.

- ***Clientes VPN de terceros***

Además, podrían dividirse en dos categorías:

1. Clientes VPN desarrollados por desarrolladores de protocolos VPN.  
Algunos ejemplos incluyen clientes OpenVPN y SoftEther .

- 
2. Clientes VPN desarrollados por proveedores de servicios VPN Las aplicaciones VPN desarrolladas por proveedores VPN vienen con características adicionales que los clientes VPN básicos desarrollados por desarrolladores de protocolos generalmente no ofrecen, como una aplicación. Killer, un Killswitch , la capacidad de agrupar servidores por país, elegir entre protocolos VPN, conectarse automáticamente al inicio y mucho más.


## ¿Qué es un servidor VPN?

Un servidor VPN es un servidor físico o virtual que está configurado para alojar y entregar servicios VPN a usuarios de todo el mundo. El servidor es una combinación de hardware VPN y software VPN que permite a los clientes VPN conectarse a una red privada segura. A diferencia de la mayoría de los servidores, un servidor VPN generalmente tiene más puertos de comunicaciones lógicos y físicos.

- *¿Cómo funciona un servidor VPN?*

Todo el proceso comienza con usted ejecutando un cliente VPN. Lo conecta con el servidor VPN y comienza a enviar su tráfico a través de su ISP. Sin embargo, esta vez todos los datos están encriptados por los protocolos VPN con los que está configurado el servidor, lo que significa que su ISP (o cualquier otra persona) no puede monitorearlos.

Una vez que el servidor VPN reciba toda la información encriptada, procederá a descifrar y la enviará al servidor web designado. Posteriormente, el servidor VPN



encriptará los datos que recibe de dicho servidor web y se los enviará a través de su ISP. Una vez que reciba esos datos en su dispositivo, el cliente VPN los descifrará por usted.

## ¿Qué es OpenVPN?


OpenVPN es tanto un protocolo VPN como un software que utiliza técnicas VPN para asegurar conexiones punto a punto y de sitio a sitio. Actualmente, es uno de los protocolos VPN más populares entre los usuarios de VPN.

## ¿Cómo funciona OpenVPN?

El protocolo OpenVPN es responsable de manejar las comunicaciones cliente-servidor. Básicamente, ayuda a establecer un “túnel” seguro entre el cliente VPN y el servidor VPN.

Cuando OpenVPN maneja el cifrado y la autenticación, usa la biblioteca OpenSSL de manera bastante extensa. Además, OpenVPN puede usar UDP (Protocolo de Datagramas de Usuario) o TCP (Protocolo de Control de Transmisión) para transmitir datos.

Si no está familiarizado con TCP y UDP, son protocolos de capa de transporte y se utilizan para transmitir datos en línea. TCP es más estable ya que ofrece funciones de corrección de errores (cuando se envía un paquete de red, TCP espera la confirmación



antes de enviarlo nuevamente o enviar un nuevo paquete). UDP no realiza corrección de errores, lo que lo hace un poco menos estable, pero mucho más rápido.

OpenVPN funciona mejor que UDP (de acuerdo con OpenVPN.net), por lo que el Servidor de Acceso de OpenVPN primero intenta establecer conexiones UDP. Si esas conexiones fallan, solo entonces el servidor intenta establecer conexiones TCP. La mayoría de los proveedores de VPN también ofrecen OpenVPN sobre UDP por defecto.

Debido a la forma en que está programado (es un protocolo de seguridad personalizado), el protocolo OpenVPN puede omitir fácilmente HTTP y NAT.

A diferencia de la mayoría de los protocolos VPN, OpenVPN es de código abierto. Eso significa que su código no es propiedad de una sola entidad, y terceros siempre pueden inspeccionarlo y mejorarlo continuamente.

## **El direccionamiento IP**

Los equipos y redes que funcionan mediante el protocolo TCP/IP (Protocolo de Control de Transmisión / Protocolo de Internet). Este protocolo necesita para su funcionamiento que los equipos que funcionan con él tengan dos parámetros configurados en su interfaz de red, estos son la dirección IP y la máscara de subred.

- ***Dirección IP***

En primer lugar, tenemos la dirección IP, que prácticamente todos conoceremos. Es una dirección lógica de 4 bytes o 32 bits cada uno de ellos separados por un punto, con la que se identifica unívocamente a un equipo o host en una red.

En la actualidad los equipos cuentan con dos tipos de direcciones IP, en primer lugar, está la dirección IPv4 que efectivamente tiene una longitud de 4 bytes (0 – 255) y que podríamos representarla de la siguiente forma:

<b>Notación decimal (la más conocida)</b>	192.168.3.120
<b>Notación binaria</b>	11000000.10101000.00000011.01111000
<b>Notación hexadecimal</b>	C0 A8 03 78

Y la dirección IPv6, que está diseñada para el caso en que el direccionamiento IP tradicional se quede corto. En este caso tendremos una dirección lógica de 128 bits, por lo que abarca un rango mucho más amplio que la dirección IPv4. Esta la veremos escrita casi siempre en formato hexadecimal:

**2010:DB92:AC32:FA10:00AA:1254:A03D:CC49**

- **Los campos de red y host y tipo de dirección IP**

Una dirección IP se puede dividir en dos partes llamadas red y host. En función de estos dos campos tendremos estos tipos de direcciones IP:



- Clase A: solamente utilizamos el primer byte para definir la red en donde nos encontramos. Los tres bytes siguientes estarán destinados a identificar al host dentro de esta red. El rango de direcciones va desde la 0.0.0.0 hasta la 127.255.255.255. La clase A se utiliza para redes muy grandes ya que tendremos direccionamiento hasta para 16 millones de equipos.
- Clase B: en este caso estaríamos utilizando los dos primeros bytes de la dirección para definir la red y los otros dos para definir el host. Este rango va desde 128.0.0.0 hasta la 191.255.255.255. también está destinado a redes de extensor tamaño.
- Clase C: en este caso utilizamos los tres primeros bytes para direccionar redes y el último byte para definir el host. De esta forma tendremos el muy conocido rango de 0.0.0 hasta 223.255.255.255.
- Clase D: el rango de IP de clase D no es de utilización común para usuarios normales, ya que está destinado a su uso experimental y grupos de máquinas concretos. Este rango va desde 224.0.0.0 hasta 239.255.255.255.
- Clase E: finalmente tenemos la clase E, la cual tampoco se utiliza en equipos de uso normal. En este caso tendremos un rango que comienza en el byte 223.0.0.0 hasta el resto.
- **Máscara de subred**

Una vez conocidas las propiedades de direccionamiento IP para los hosts dentro de una red, pasamos a ver otro parámetro no menos importante, que es la máscara de subred.

Para cada clase de IP se puede contar con un número de subredes determinado. Una subred es una red física independiente que comparte la misma dirección IP con otras redes físicas, es decir, ahora estamos identificando a la red principal en donde se conectan los hosts.

Precisamente la función de la máscara de subred es lograr que equipos que comparten el mismo identificador de red y que están situados en redes físicas distintas se puedan comunicar. Será nuestro router o servidor el que haga la correspondencia entre la información de la máscara de subred y la dirección IP de los hosts.

Existen tres tipos de máscaras de subred, para cada una de las clases utilizadas:

A	255.0.0.0
B	255.255.0.0
C	255.255.255.0

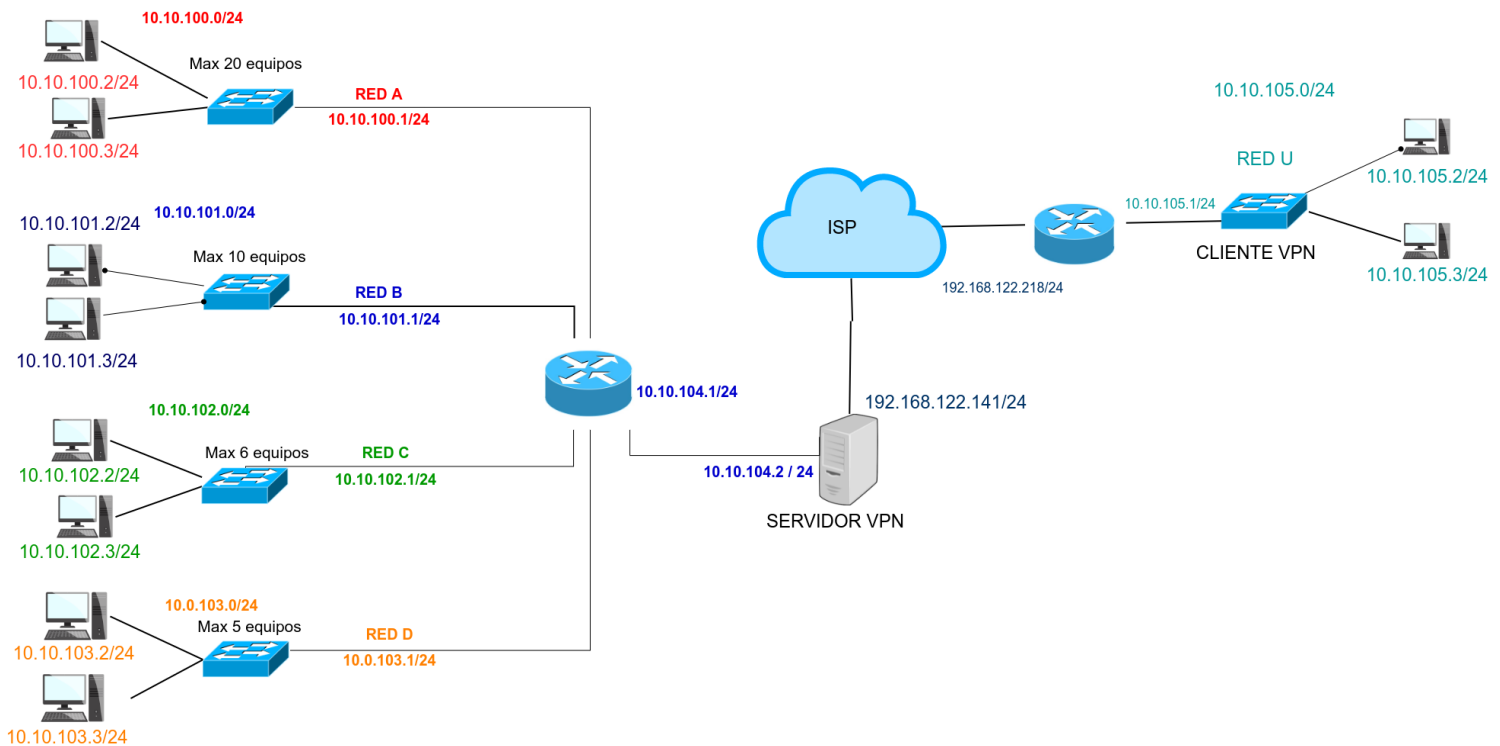
- **Notación abreviada dirección-máscara**
- Si tenemos 192.168.1.1/24, significa que los 24 primeros bits (en binario) están destinados a la red, por lo que la máscara de subred sería 255.255.255.0, y la red a la que pertenece sería 192.168.1.0.
- Si tenemos 180.10.1.1/16, significará que los primeros 16 bits están destinados a la red, entonces sería 255.25.0.0, y la red a la que pertenece sería 180.10.0.0.

—

\_\_\_\_\_

## Arquitectura de la Red

La red está estructurada de la siguiente manera segun topologia solicitada especificamente para el proyecto:



Estas redes cuentan con las siguientes limitaciones:

REDES	CONEXIÓN
A → B	✓
A → C	✓
B → <b>A</b>	✗
B → U	✗

$C \rightarrow A$	✓
$A \rightarrow B$	✓
$D \rightarrow A$	✓
$D \rightarrow B$	✓
$D \rightarrow C$	✓
todas las demás	✗

## Configuración de Servidor VPN con OpenVPN

Configuración de servidor con el cliente dentro de la red

Instalar el repositorio epel-release.

```
[root@localhost]# dnf install -y epel-release
```

Se instala la aplicación:

```
[root@localhost ~]# dnf install -y openvpn easy-rsa
```

Se accede al directorio de easy-rsa, donde se ubican los archivos necesarios para la configuración:

```
[root@localhost ~]# cd /usr/share/easy-rsa/3.0.8
```

Si se desea, se puede copiar el contenido al directorio de openvpn: (en mi caso lo haré, por comodidad de configuración)

```
[root@localhost ~]# cp -r * /etc/openvpn/server/.
```

Cambiamos al directorio dónde se copiaron los archivos:

```
[root@localhost ~]# cd /etc/openvpn/server/
```

Se inicia el PKI (Public Key Infrastructure) y se construye la autoridad de certificación (CA):

```
[root@localhost server]# ./easyrsa init-pki
```

```
[root@localhost server]# ./easyrsa build-ca nopass
```

Se genera y firma el certificado del servidor:

```
[root@localhost server]# ./easyrsa gen-req serverA nopass
```

```
[root@localhost server]# ./easyrsa sign-req server serverA nopass
```

Se genera y firma el certificado para el cliente:

```
[root@localhost server]# ./easyrsa gen-req clientA nopass
```

```
[root@localhost server]# ./easyrsa sign-req client clientA nopass
```

se genera el dh.pem (parámetros de Diffie-Hellman) que establece la fortaleza en el intercambio de las claves:

```
[root@localhost server]# ./easyrsa gen-dh
```

Copiar el archivo de configuración ejemplo al directorio de configuración:

```
[root@localhost openvpn]# cp
```

`/usr/share/doc/openvpn/sample/sample-config-files/server.conf`  
`/etc/openvpn/server/.`

Editamos el archivo `/etc/openvpn/server.conf`:

```
[root@localhost ~]# vi /etc/openvpn/server/server.conf
```

Se edita el archivo `/etc/openvpn/server.conf` y se configuran las rutas correctas de los certificados y llaves, para este ejemplo, quedaría como se muestra a continuación.

```
# Diffie hellman parameters.
# Generate your own with:
#   openssl dhparam -out dh2048.pem 2048

dh /etc/openvpn/server/pki/dh.pem
.
topology subnet
.
.
push "redirect-gateway def1 bypass-dhcp"
.
.
push "dhcp-option DNS 8.8.8.8"
push "dhcp-option DNS 8.8.4.4"
.
.
# The second parameter should be '0'
# on the server and '1' on the clients.
;tls-auth ta.key 0 # This file is secret
.
.
# You can uncomment this out on
# non-Windows systems.
user nobody
group nobody
```

Si alguno de los parámetros resaltados no se configura, el servicio no podrá ser iniciado.

Se habilita el ip forwarding.

```
[root@localhost ~]# sysctl -w net.ipv4.ip_forward=1
```

Para hacerlo de forma persistente:

//verifican como se llama el `.conf`



```
[root@localhost ~]# echo net.ipv4.ip_forward=1 >> /etc/sysctl.d/sysctl-additionals.conf
```

Establecer las reglas del firewall:

```
[root@localhost ~]# firewall-cmd --zone=trusted --add-masquerade --permanent
[root@localhost ~]# firewall-cmd --set-default-zone=trusted
[root@localhost ~]# firewall-cmd --add-service=openvpn --permanent
[root@localhost ~]# firewall-cmd --reload
[root@localhost ~]# firewall-cmd --list-all
```

Deshabilitar SELinux:

```
[root@localhost ~]# setenforce 0
```

Se comprueba su funcionamiento con:

```
[root@localhost ~]# cd /etc/openvpn/server/
[root@localhost server]# openvpn --config server.conf
```

La siguiente línea al final de la salida, indica que levantó correctamente:

```
.... Initialization Sequence Complete
```

Se inicia y habilita el servidor:

```
[root@localhost ~]# systemctl enable openvpn-server@server --now
[root@localhost ~]# systemctl status openvpn-server@server
```

Se puede comprobar que la interfaz virtual se haya habilitado con 'ip addr':

```
[root@server ~]# ip addr
.
.
5: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
state UNKNOWN group default qlen 100
    link/none
    inet 10.8.0.1/24 brd 10.8.0.255 scope global tun0
        valid_lft forever preferred_lft forever
    inet6 fe80::dab7:46e8:cc6e:4b79/64 scope link flags 800
        valid_lft forever preferred_lft forever
```

## Configuracion del Cliente

Instalar el repositorio epel-release.

```
[root@localhost ~]# dnf -y install epel-release
```

Se instala la aplicación:

```
[root@localhost ~]# dnf -y install openvpn
```

Se comprueba la conexión entre los equipos (servidor y cliente), y se copian los certificados del generados en el servidor hacia el cliente: (no olvides sustituir *ip\_client* por dirección ip de tu cliente):

```
[root@localhost ~]# cd /etc/openvpn/server/pki/
[root@localhost pki]# scp ca.crt root@ip_client:/etc/openvpn/client/.
[root@localhost pki]# scp issued/clientA.crt root@ip_client:/etc/openvpn/client/.
[root@localhost pki]# scp private/clientA.key root@ip_client:/etc/openvpn/client/.
```

Se edita el archivo de configuración para el **cliente**:

```
[root@client ~]# vi /etc/openvpn/client/clientA.conf
```

Y se agregan las siguientes líneas (no olvides sustituir *ip\_server* por la dirección ip de tu servidor):

```
client
dev tun
proto udp
remote ip_server 1194
ca ca.crt          #en caso de usar otra ruta, modificarla
cert clientA.crt   #en caso de usar otra ruta, modificarla
key clientA.key     #en caso de usar otra ruta, modificarla
verb 5

remote-cert-tls server
auth-nocache
cipher AES-256-CBC
```

Se guarda el archivo: y ejecuta la configuración para comprobar el funcionamiento:

```
[root@client ~]# cd /etc/openvpn/client/
```

```
[root@client openvpn]# openvpn --config clientA.conf
```

La siguiente línea al final de la salida indica que la VPN levantó correctamente, si no se despliega esa línea es necesario revisar la configuración:

```
.... Initialization Sequence Complete
```

Si la VPN levantó correctamente, se puede habilitar el servicio de la siguiente manera:

```
[root@client ~]# systemctl start openvpn-client@clientA
```

O bien, para iniciarla y habilitarla para que inicie con el arranque del sistema:

```
[root@client ~]# systemctl enable openvpn-client@clientA --now
```

Al igual que con el servidor, se debió crear una interfaz virtual:

```
[root@server ~]# ip addr
.
.
5: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN group
default qlen 100
    link/none
    inet 10.8.0.2/24 brd 10.8.0.255 scope global tun0
        valid_lft forever preferred_lft forever
    inet6 fe80::a2fc:5174:af3f:5c3e/64 scope link flags 800
        valid_lft forever preferred_lft forever
```

Con lo anterior, la VPN está establecida y funcional.

## Configuración de las Redes

Para la configuración de las redes se hizo uso de la herramienta nmtui para configurar las direcciones ip de cada planta y las puertas de enlace

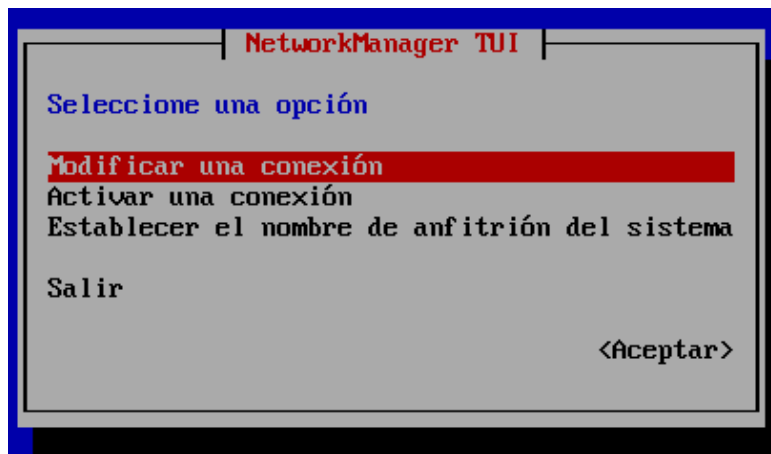
### **Router:**

Antes de iniciar la configuración sera necesario tener 5 interfaces de red para administrar las diferentes conexiones, estas tendran cada una una ip estatica tal como se muestra a continuación:

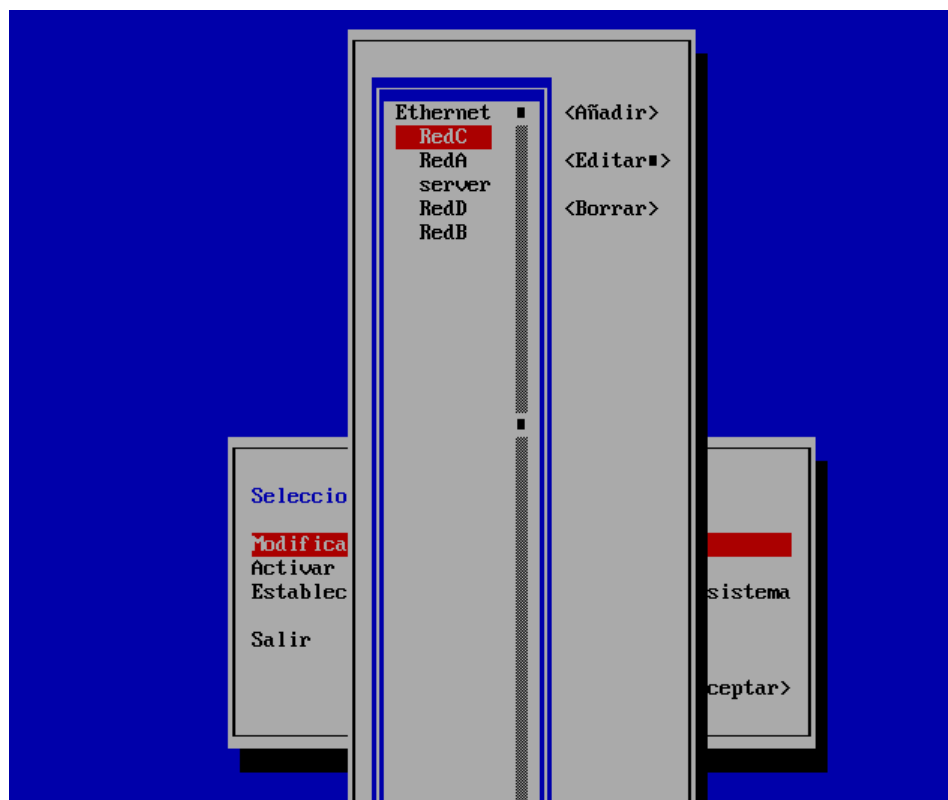
<b>interfaz</b>	<b>red</b>	<b>ip</b>
interfaz 1	Red A: 10.10.100.0/24	10.10.100.1/24
interfaz 2	Red B: 10.10.101.0/24	10.10.101.1/24
interfaz 3	Red C: 10.10.102.0/24	10.10.102.1/24
interfaz 4	Red D: 10.10.103.0/24	10.10.103.1/24
interfaz 5	servidor: 10.10.104.0/24	10.10.104.1/24

ya teniendo definidas las ips se procede a configurar cada una

En la línea de comandos abrimos nmtui y elegimos modificar una conexion



luego en modificar y se escoge la red que se desea editar, para el router deberán aparecer 5 interfaces como se muestra a continuación:



a cada una se le ira colocando el nombre conforme se vaya configurando cada vez asi que es normal que aparezca con el orden por defecto **conexion cableada**

The screenshot shows the 'Editar la conexión' window for a profile named 'RedA'. The device is '52:54:00:D3:11:B9 (ens13)'. The configuration is for an Ethernet interface. Under 'CONFIGURACIÓN IPv4', the mode is 'Manual', the address is '10.10.100.1/24', and the gateway is '3.8.8.8'. There are checkboxes for 'Enrutando' (No hay rutas personalizadas), 'Nunca utilice esta red para la ruta predeterminada', 'Ignorar rutas obtenidas automáticamente', and 'Ignorar los parámetros DNS obtenidos automáticamente'. There is also a checkbox for 'Solicitar la dirección IPv4 para esta conexión'. The 'CONFIGURACIÓN IPv6' section is set to 'Ignorar'. At the bottom, there are checkboxes for 'Conectar de forma automática' and 'Disponible para todos los usuarios', and buttons for 'Cancelar' and 'Aceptar'.

Configuración interfaz de la Red C en el router

Configuración interfaz de la Red C en el router:

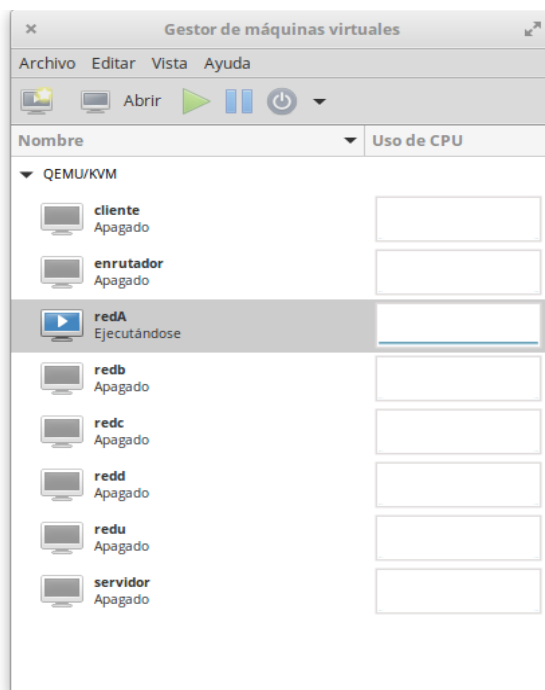
The screenshot shows the 'Editar la conexión' window for a profile named 'RedC'. The device is '52:54:00:DA:F5:46 (ens12)'. The configuration is for an Ethernet interface. Under 'CONFIGURACIÓN IPv4', the mode is 'Manual', the address is '10.10.102.1/24', and the gateway is '3.8.8.8'. There are checkboxes for 'Enrutando' (No hay rutas personalizadas), 'Nunca utilice esta red para la ruta predeterminada', 'Ignorar rutas obtenidas automáticamente', and 'Ignorar los parámetros DNS obtenidos automáticamente'. There is also a checkbox for 'Solicitar la dirección IPv4 para esta conexión'. The 'CONFIGURACIÓN IPv6' section is set to 'Automático'. At the bottom, there are checkboxes for 'Conectar de forma automática' and 'Disponible para todos los usuarios', and buttons for 'Cancelar' and 'Aceptar'.

y esta configuración se le dará a las demás interfaces según su IP y su nombre.

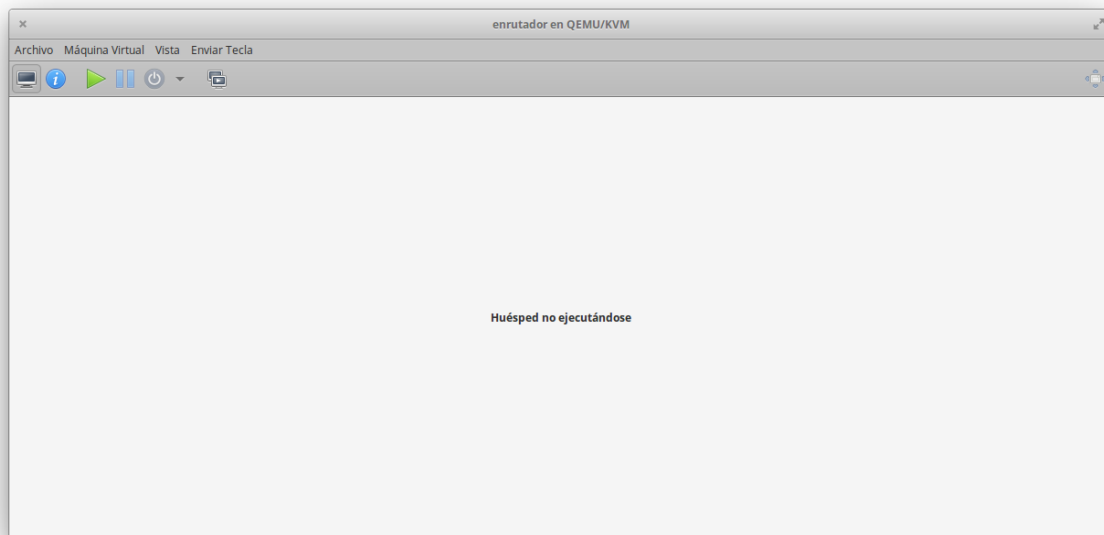
## Configuración de subredes

Se debe ingresar a cada equipo para configurar la dirección IP conforme la arquitectura de red, en el caso del servidor y el cliente se debe agregar una interfaz. A continuación describimos los pasos y campos a llenar.

Vista previa a los equipos. Usando KVM.



Para ingresar a cada equipo deberá seleccionar el equipo a utilizar, click derecho y seleccionar abrir, se desplegará una pantalla e inicializamos la máquina virtual.



Ingresamos nuestras credenciales para iniciar sesión.

```
CentOS Linux 7 (Core)
Kernel 3.10.0-1160.el7.x86_64 on an x86_64

localhost login: root
Password: _
```



Para configurar cada máquina ingresamos el comando << **nmtui** >> y nos desplegará la siguiente pantalla:



Seleccionamos la primera opción y nos mostrará una pantalla como esta:

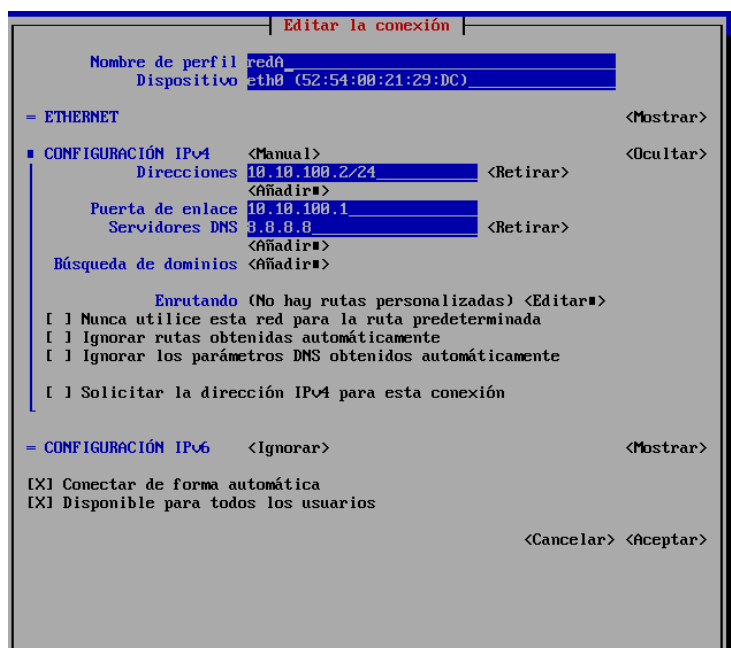


Seleccionamos nuestra interfaz y editamos.

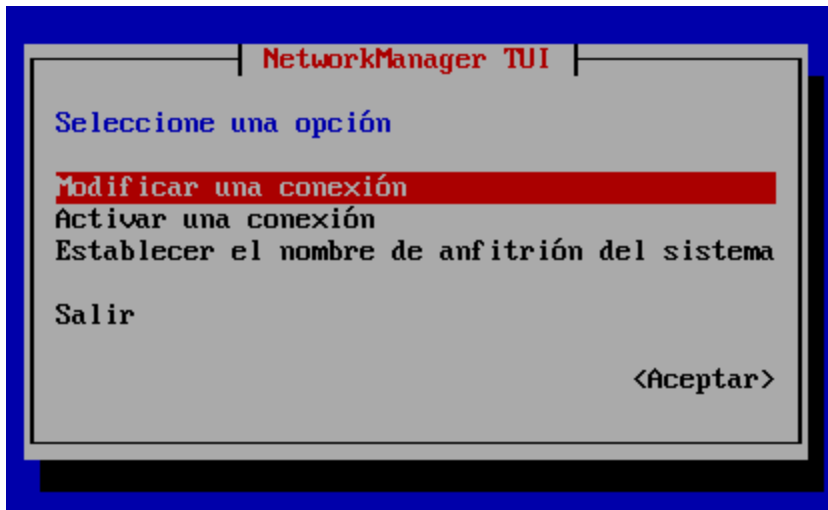
Llenamos los campos correspondientes según la red. Basándonos en la siguiente tabla y nuestro resultado quedaría como la pantalla.

Subred	IP	Puerta enlace de	Servidor DNS
red A	10.10.100.2/24	10.10.100.1	8.8.8.8
red B	10.10.101.2/24	10.10.101.1	8.8.8.8
red C	10.10.102.2/24	10.10.102.1	8.8.8.8
red D	10.10.103.2/24	10.10.103.1	8.8.8.8
red U	10.10.105.2/24	10.10.105.1	8.8.8.8

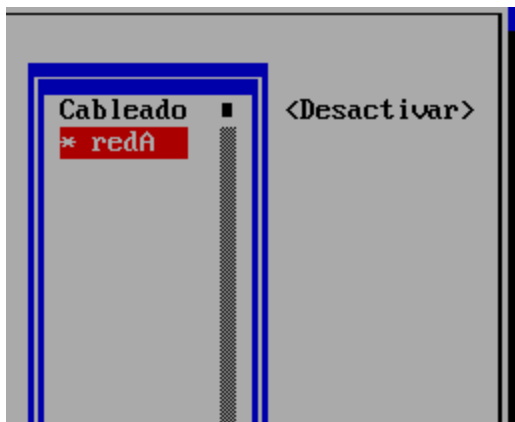
\*dns 8.8.8.8 es el que hemos ingresado en el archivo server.conf



Guardamos los cambios y volvemos al menú principal.



Seleccionamos activa una conexión y verificamos que se pueda activar y desactivar. Lo dejaremos como desactivar.



y regresamos al menú principal, salimos del menú principal y ejecutamos en consola el siguiente comando.

**systemctl restart network.service.**

verificamos que este correcto con el comando

## systemctl status network.service

Mostrando algo similar. Con esto estamos seguros que esta corriendo con éxito.

```
[root@localhost ~]# systemctl status network.service
■ network.service - LSB: Bring up/down networking
   Loaded: loaded (/etc/rc.d/init.d/network; bad; vendor preset: disabled)
   Active: active (exited) since mar 2021-05-11 18:39:40 CST; 18min ago
     Docs: man:systemd-sysv-generator(8)
   Process: 821 ExecStart=/etc/rc.d/init.d/network start (code=exited, status=0/SUCCESS)

may 11 18:39:39 localhost.localdomain systemd[1]: Starting LSB: Bring up/down networking...
may 11 18:39:40 localhost.localdomain network[821]: Activación de la interfaz de loopback: [ OK ]
may 11 18:39:40 localhost.localdomain network[821]: Activando interfaz eth0: [ OK ]
may 11 18:39:40 localhost.localdomain systemd[1]: Started LSB: Bring up/down networking.
[root@localhost ~]#
```

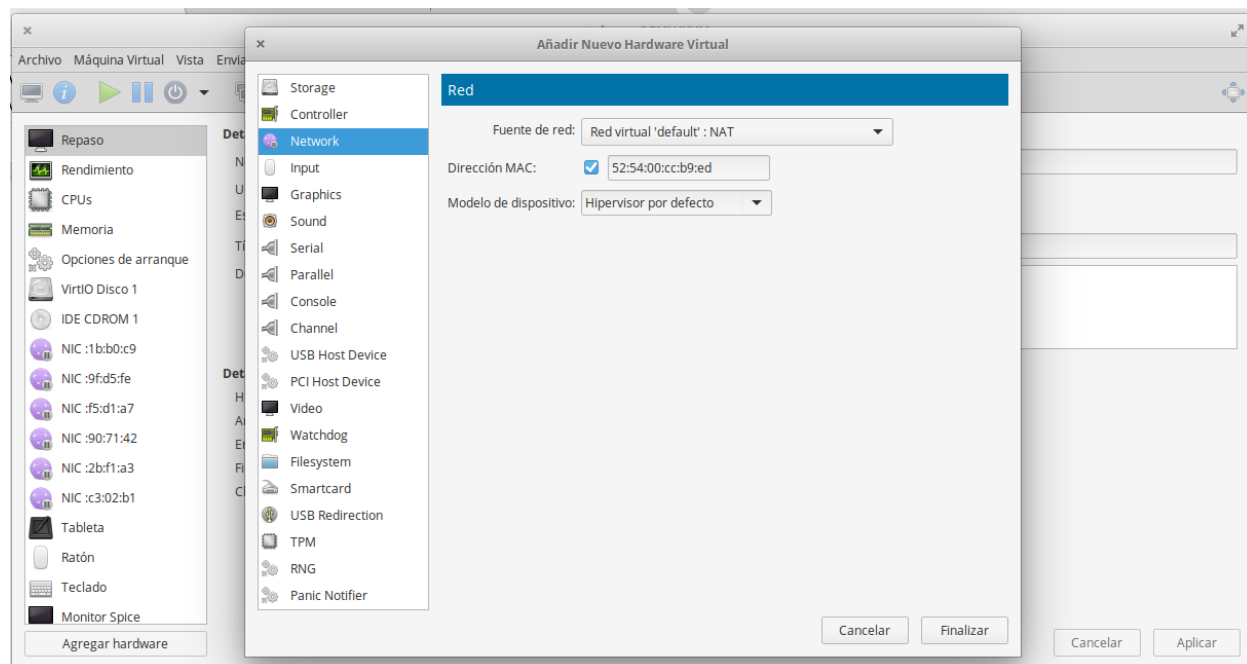
Podemos verificar si la ip ha sido cambiada exitosamente con el comando **ip addr**.

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 52:54:00:21:29:dc brd ff:ff:ff:ff:ff:ff
    inet 10.10.100.2/24 brd 10.10.100.255 scope global noprefixroute eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::5054:ff:fe21:29dc/64 scope link
        valid_lft forever preferred_lft forever
```

## Configuración del servidor con el enrutador y red del cliente

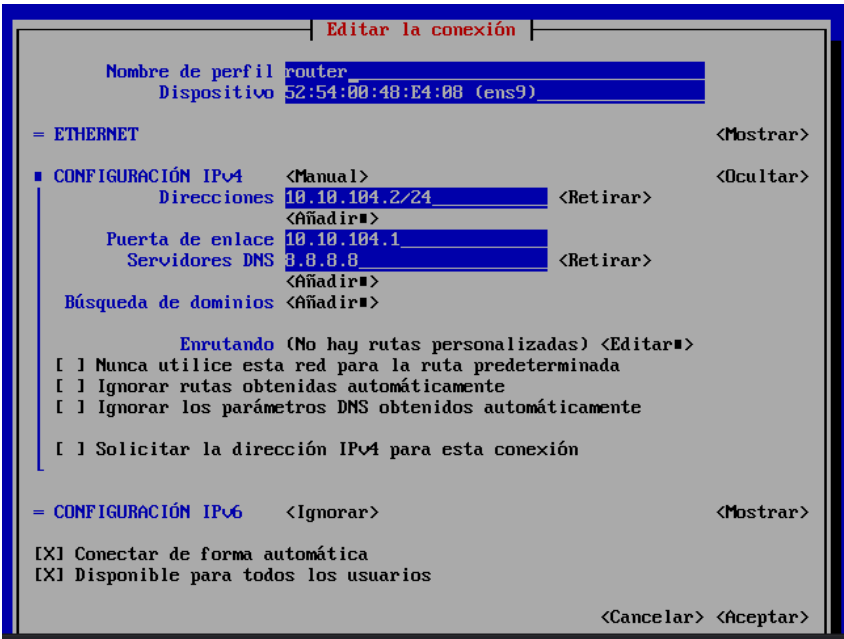
Para configurar el cliente y el servidor se debe agregar una interfaz, seleccionando el icono de información en la parte superior de la máquina virtual. Agregar hardware, seleccionamos red y la agregamos con la

configuración por defecto en finalizar. Cada interfaz que agregamos se mostrará de lado izquierdo con un icono morado.



Realizamos el paso anterior para cliente y servidor.

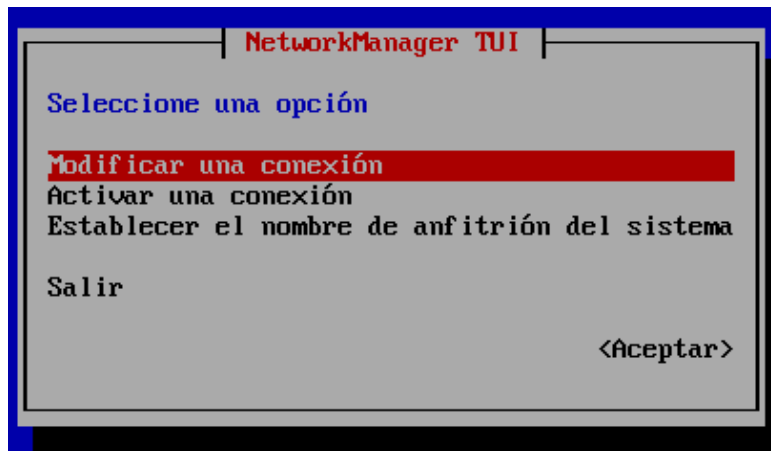
Ingresamos el comando `nmtui` y seleccionamos la opción modificar una conexión y dejamos la siguiente configuración.



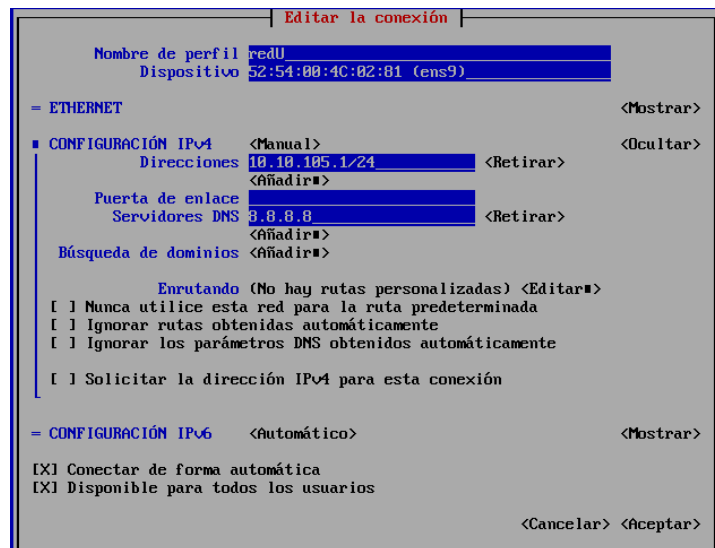
## Red del Cliente

ya teniendo la interfaz agregada procedemos con la configuración:

En la línea de comandos abrimos nmtui y elegimos modificar una conexión

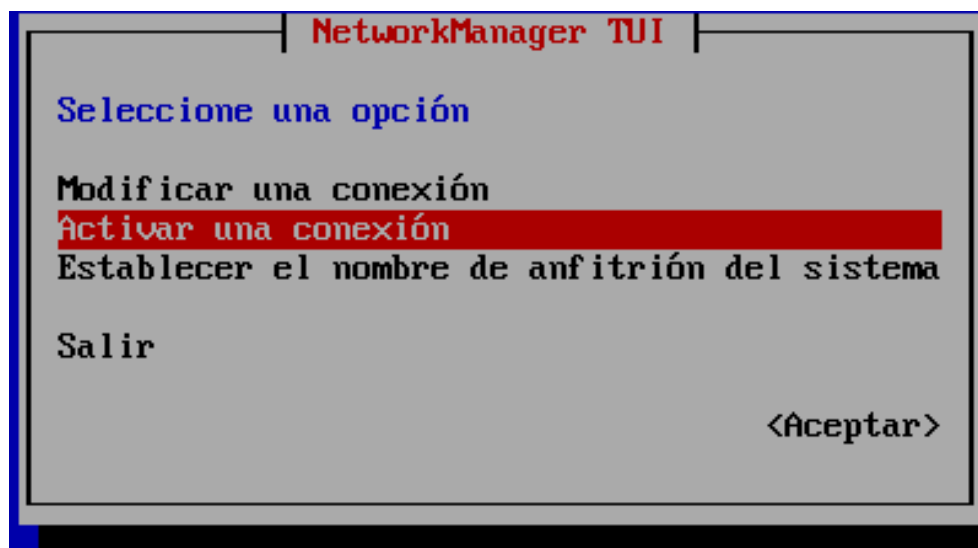


luego en modificar y se escoge la red que se acaba de agregar, se le da editar y se configura como se muestra a continuación:

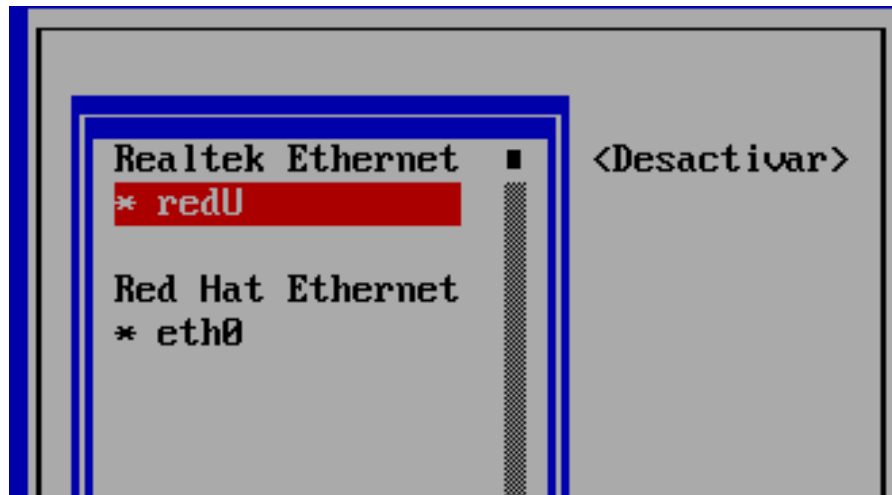


## NOTA IMPORTANTE

1. Cada vez que se termina de configurar una interfaz es necesario ir a la ventana



y desactivamos y activamos cada una de las interfaces para evitar problemas de reconocimiento de cambios.



2. para observar si la ip fue cambiada con exito utilizamos el comando `ip addr` que nos presentara la siguiente salida en una maquina a la que le acabamos de cambiar su ip a la 10.10.105.1/24

```
[root@localhost ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: ens9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 52:54:00:4c:02:81 brd ff:ff:ff:ff:ff:ff
    inet 10.10.105.1/24 brd 10.10.105.255 scope global noprefixroute ens9
        valid_lft forever preferred_lft forever
    inet6 fe80::cff6:2af6:ea08:babc/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 52:54:00:19:4d:97 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.218/24 brd 192.168.122.255 scope global noprefixroute dynamic eth0
        valid_lft 2369sec preferred_lft 2369sec
    inet6 fe80::5b99:8ed1:b061:c555/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
[root@localhost ~]# _
```



## Restricciones

Como se menciono anteriormente en la Arquitectura de la red, las comunicaciones entre redes estan limitadas es por esto que se mencionaran las mas importantes:

**miembros de la RedA: no puede enviar paquetes a los miembros de laRedC**

**redA:** 10.10.100.2/24    y **redB:**10.10.101.2/24

desde un miembro de la redA en este caso 10.10.100.2/24 hacemos ping hacia la redB y como se puede observar no puede enviar paquetes

```
[root@localhost ~]# ping 10.10.103.2
PING 10.10.103.2 (10.10.103.2) 56(84) bytes of data.
From 10.10.100.2 icmp_seq=1 Destination Host Unreachable
From 10.10.100.2 icmp_seq=2 Destination Host Unreachable
From 10.10.100.2 icmp_seq=3 Destination Host Unreachable
From 10.10.100.2 icmp_seq=4 Destination Host Unreachable
^C
--- 10.10.103.2 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 3004ms
pipe 4
```

desde un miembro de la redA en este caso 10.10.100.2/24 hacemos ping hacia la redB y como se puede observar no puede enviar paquetes

**miembros de la RedB: no puede enviar paquetes a los miembros de ninguna red**

**B:** 10.10.101.2/24

**ping con la red redA:**10.10.101.2/24

```
[root@localhost ~]# ping 10.10.103.2
PING 10.10.103.2 (10.10.103.2) 56(84) bytes of data.
From 10.10.100.2 icmp_seq=1 Destination Host Unreachable
From 10.10.100.2 icmp_seq=2 Destination Host Unreachable
From 10.10.100.2 icmp_seq=3 Destination Host Unreachable
From 10.10.100.2 icmp_seq=4 Destination Host Unreachable
^C
--- 10.10.103.2 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 3004ms
pipe 4
```

**ping con la red redC:10.10.102.2/24**

```
[root@localhost ~]# ping 10.10.102.2
PING 10.10.102.2 (10.10.102.2) 56(84) bytes of data.
From 10.10.101.2 icmp_seq=1 Destination Host Unreachable
From 10.10.101.2 icmp_seq=2 Destination Host Unreachable
From 10.10.101.2 icmp_seq=3 Destination Host Unreachable
From 10.10.101.2 icmp_seq=4 Destination Host Unreachable
^C
--- 10.10.102.2 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 2999ms
pipe 4
[root@localhost ~]# _
```

**ping con la red redD:10.10.103.2/24**

```
[root@localhost ~]# ping 10.10.103.2
PING 10.10.103.2 (10.10.103.2) 56(84) bytes of data.
From 10.10.101.2 icmp_seq=1 Destination Host Unreachable
From 10.10.101.2 icmp_seq=2 Destination Host Unreachable
From 10.10.101.2 icmp_seq=3 Destination Host Unreachable
From 10.10.101.2 icmp_seq=4 Destination Host Unreachable
^C
--- 10.10.103.2 ping statistics ---
5 packets transmitted, 0 received, +4 errors, 100% packet loss, time 4003ms
pipe 4
```