Práctica P01- SIPD: Redes privadas virtuales

Gabriel Lazovsky Igual

Alejandro Rodríguez Ferrer

CEU

Índice

Parte obligatoria:

a) Crear el escenario virtualizado propuesto. Configurar las direcciones IP de todos los interfaces y activar el forwarding en el Gateway. Establecer como router por defecto el Gateway en los restantes equipos y comprobar la conectividad. Instalar algún servicio en los equipos PC1, PC2 y Servidor (servidor SSH, Apache, …).

Antes de nada, instalaremos los paquetes necesarios en todas las maquinas como por ejemplo en el pc1 que se observa en la siguiente imagen:

Instalación de paquetes:

```
root@vpnservergabriel:~# apt install -y apache2 openvpn easy-rsa
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  apache2-bin apache2-data apache2-utils libapr1t64 libaprutil1-dbd-sqlite3
  libaprutil1-ldap libaprutil1t64 libccid libeac3 liblua5.4-0 libpcsclite1
  libpkcs11-helper1t64 opensc opensc-pkcs11 pcscd ssl-cert
Paquetes sugeridos:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom www-browser pcmciautils
  openvpn-dco-dkms openvpn-systemd-resolved
Se instalarán los siguientes paquetes NUEVOS:
  apache2 apache2-bin apache2-data apache2-utils easy-rsa libapr1t64
  libaprutil1-dbd-sqlite3 libaprutil1-ldap libaprutil1t64 libccid libeac3 liblua5.4-0
```

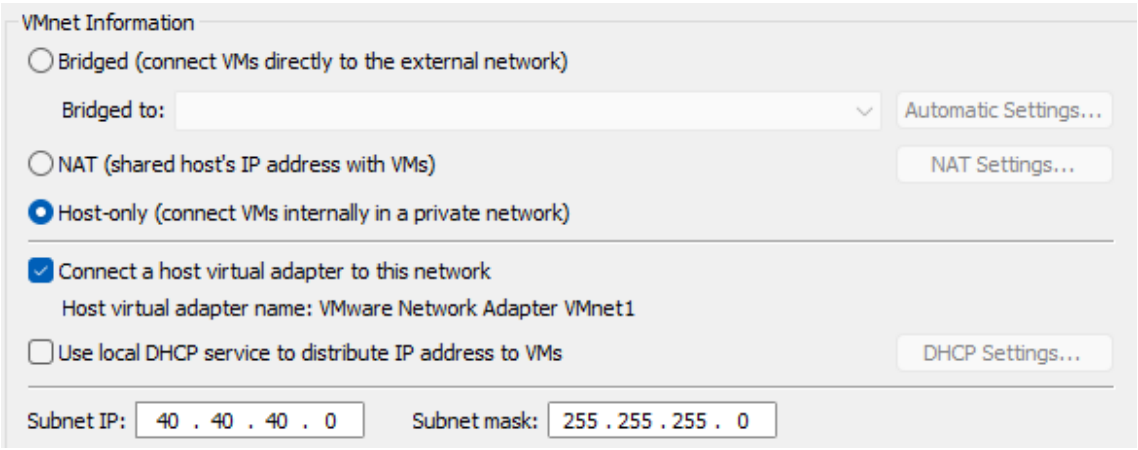
Para esta práctica, usaremos 4 máquinas Ubuntu desde VMware. Para la red externa usaremos host-only y para la interna NAT. Para crearlas, abriremos VMware en modo administrador, dentro “Edit” entraremos en Virtual Network Editor. Asignaremos el PC1 y el Gateway a la interfaz en Host-Only mientras que el Gateway tendrá otra interfaz con el NAT junto con el PC2 y el servidor.

Esto es lo que deberíamos de tener:

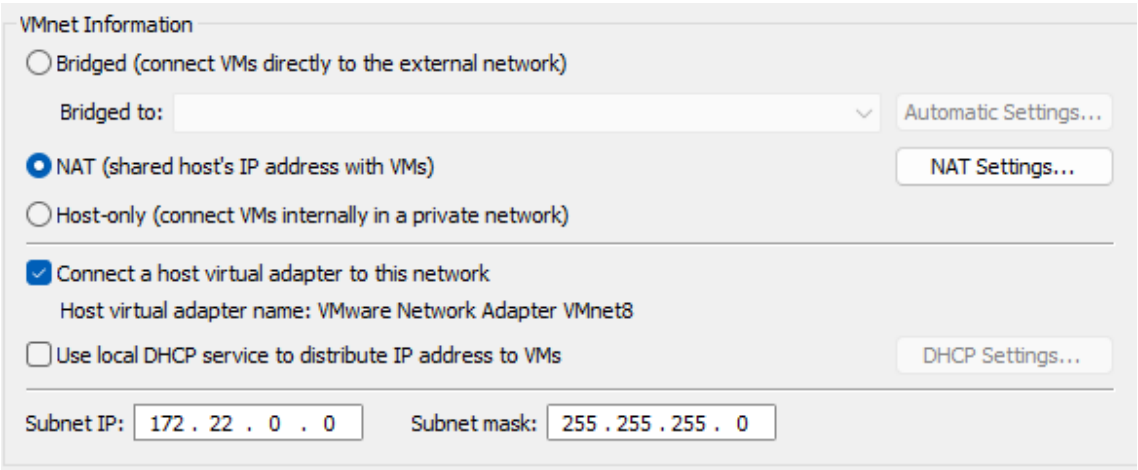
Virtual Network Editor

Name	Type	External Connection	Host Connection	DHCP	Subnet Address
VMnet0	Custom	-	-	-	192.168.47.0
VMnet1	Host-only	-	Connected	-	40.40.40.0
VMnet8	NAT	NAT	Connected	-	172.22.0.0

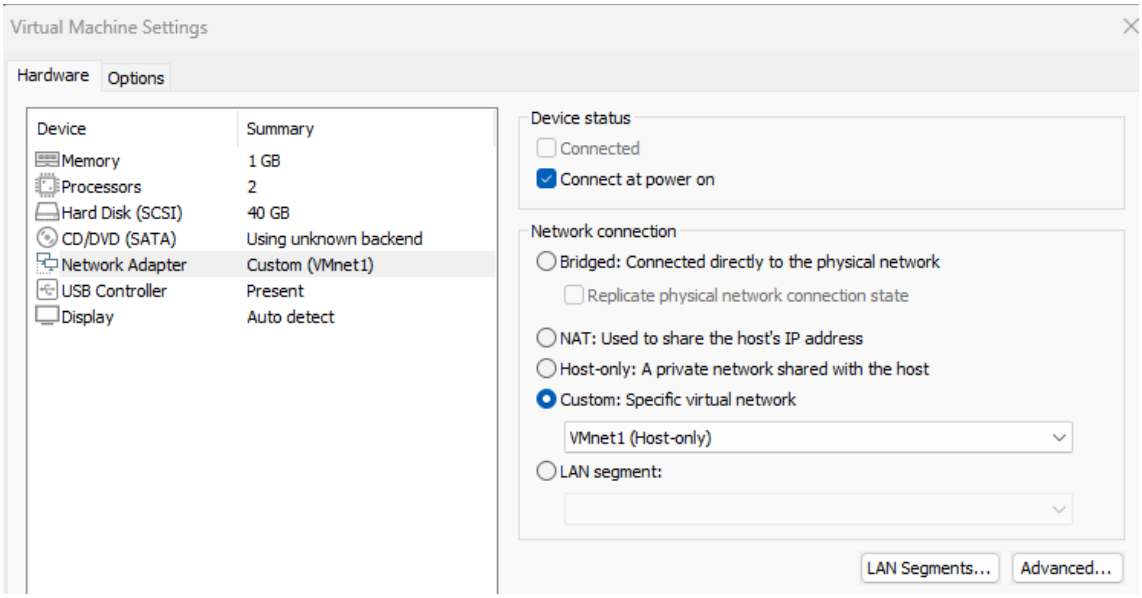
Esta es la host-only (VMnet1):



Esta en la NAT (VMnet8):

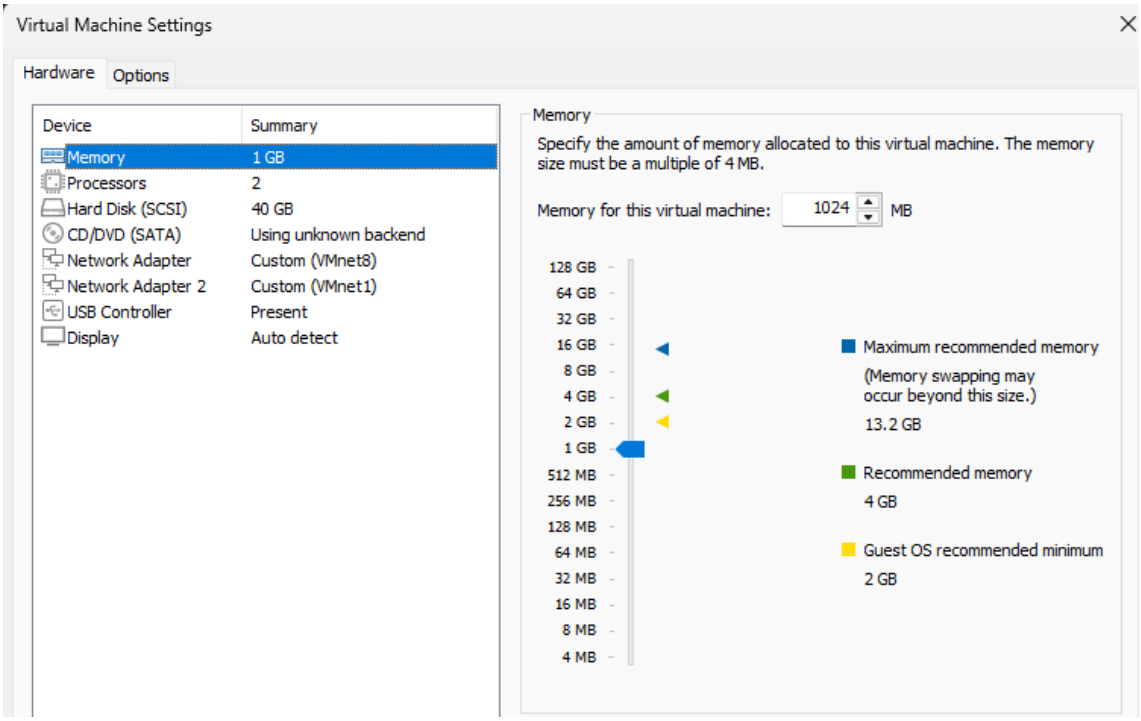


Ponemos DHCP disable, ya que asignaremos las ips manualmente desde el netplan, pero primero aclarar que para asignar nuestros nets (h-o /nat) lo haremos dándole click derecho a la máquina y entrando en settings:



Ejemplo (PC1)

Para el GW, debemos añadir una nueva interfaz de red y seleccionamos las nets que deseamos:



A continuación, se cambian manualmente las ip en los documentos con el siguiente comando:

```
sudo vim /etc/netplan/00-installer-config.yaml
```

Y se edita de la siguiente forma cada *netplan* cada máquina con la configuración:

PC1:

```
# This is the network config written by 'subiquity'
network:
  version: 2
  ethernets:
    eth0:
      dhcp4: no
      addresses: [40.40.40.30/24]
```

GW:

```
# This is the network config written by 'subiquity'
network:
  version: 2
  ethernets:
    eth0:
      dhcp4: no
      addresses: [172.22.0.10/24]
      routes:
        - to: default
          via: 172.22.0.1
    eth1:
      dhcp4: no
      addresses: [40.40.40.10/24]
```

ServidorVPN:

```
# This is the network config written by 'subiquity'
network:
  version: 2
  ethernets:
    eth0:
      dhcp4: no
      addresses: [172.22.0.80/24]
      routes:
        - to: default
          via: 172.22.0.10
```

PC2:

```
# This is the network config written by 'subiquity'
network:
  version: 2
  ethernets:
    eth0:
      dhcp4: no
      addresses: [172.22.0.50/24]
      routes:
        - to: default
          via: 172.22.0.10
```

Si todo ha sido configurado correctamente, tendría que haber conectividad entre máquina conectadas a la red interna (PC2, ServidorVPN y GW) y externa (GW y PC1).

Por lo que para comprobarlo se realiza primero el comando

ifconfig

Para comprobar si se han actualizado las ips de forma correcta y posteriormente con el comando ping para ver la conectividad de dentro de las redes.

PC1:

```
gabriel@pc1gabriel:~$ ifconfig
ens160: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 40.40.40.30  netmask 255.255.255.0  broadcast 40.40.40.255
        inet6 fe80::20c:29ff:feab:cc5b  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:ab:cc:5b  txqueuelen 1000  (Ethernet)
        RX packets 786723  bytes 56813857 (56.8 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 741177  bytes 59999224 (59.9 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 45  memory 0x3fe00000-3fe20000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 786529  bytes 60560778 (60.5 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 786529  bytes 60560778 (60.5 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

GW:

```
root@gatewaygabriel:~# ifconfig
ens160: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 40.40.40.10  netmask 255.255.255.0  broadcast 40.40.40.255
        inet6 fe80::20c:29ff:fe4a:7155  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:4a:71:55  txqueuelen 1000  (Ethernet)
        RX packets 9759  bytes 944618 (944.6 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4565  bytes 518784 (518.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 44  memory 0x3fe00000-3fe20000

ens256: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.22.0.10  netmask 255.255.255.0  broadcast 172.22.0.255
        inet6 fe80::20c:29ff:fe4a:715f  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:4a:71:5f  txqueuelen 1000  (Ethernet)
        RX packets 1406  bytes 215949 (215.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 674  bytes 112029 (112.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 47  memory 0x3e600000-3e620000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 589068  bytes 41832190 (41.8 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 589068  bytes 41832190 (41.8 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

VPNServer:

```
gabriel@vpnservergabriel:~$ ifconfig
ens160: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.22.0.80  netmask 255.255.255.0  broadcast 172.22.0.255
        inet6 fe80::20c:29ff:fe3d:3029  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:3d:30:29  txqueuelen 1000  (Ethernet)
        RX packets 165821  bytes 79258406 (79.2 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 70424  bytes 7032350 (7.0 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 44  memory 0x3fe00000-3fe20000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 8822  bytes 653371 (653.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8822  bytes 653371 (653.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

PC2:

```
gabriel@pc2gabriel:~$ ifconfig
ens160: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.22.0.50  netmask 255.255.255.0  broadcast 172.22.0.255
        inet6 fe80::20c:29ff:fe2a:3d19  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:2a:3d:19  txqueuelen 1000  (Ethernet)
        RX packets 4804  bytes 470132 (470.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2171  bytes 222604 (222.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 44  memory 0x3fe00000-3fe20000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 664304  bytes 47174832 (47.1 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 664304  bytes 47174832 (47.1 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Ahora se comprobará la conectividad entre las diferentes maquinas.

Comenzando desde el PC1 al GW

```
gabriel@pc1gabriel:~$ ping 40.40.40.10
PING 40.40.40.10 (40.40.40.10) 56(84) bytes of data.
64 bytes from 40.40.40.10: icmp_seq=1 ttl=64 time=3.58 ms
64 bytes from 40.40.40.10: icmp_seq=2 ttl=64 time=0.691 ms
64 bytes from 40.40.40.10: icmp_seq=3 ttl=64 time=1.04 ms
64 bytes from 40.40.40.10: icmp_seq=4 ttl=64 time=0.793 ms
^C
--- 40.40.40.10 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3019ms
rtt min/avg/max/mdev = 0.691/1.525/3.576/1.190 ms
gabriel@pc1gabriel:~$
```

Desde el PC2 al server:

```
gabriel@pc2gabriel:~$ ping 172.22.0.80
PING 172.22.0.80 (172.22.0.80) 56(84) bytes of data.
64 bytes from 172.22.0.80: icmp_seq=1 ttl=64 time=5.03 ms
64 bytes from 172.22.0.80: icmp_seq=2 ttl=64 time=0.753 ms
```

En el GW debemos poner el valor de forwarding a 1, esto lo hacemos añadiendo esto en /etc/sysctl.conf:

```
net.ipv4.ip_forward=1
```

Podemos reiniciar o ejecutar “sudo sysctl –p" para que se aplique directamente, para comprobar que esta con valor 1, ejecutamos:

```
sysctl net.ipv4.ip_forward
```

b) Crear una Autoridad de Certificación en el Servidor y emitir los elemento necesarios para nuestra configuración (valores DH y claves/certificados para el servidor y el cliente).

DEBEMOS TENER YA CONECTIVIDAD ENTRE PC1 – GW y GW - ServidorVPN

En primer lugar, instalamos en cliente y servidor Open-VPN y sus dependencias:

```
sudo apt install -y openvpn easy-rsa (openssl lzo)
```

```
gabriel@pc1gabriel:~$ sudo apt install openvpn
[sudo] password for gabriel:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  libpkcs11-helper1t64
Paquetes sugeridos:
  openvpn-dco-dkms openvpn-systemd-resolved easy-rsa
Se instalarán los siguientes paquetes NUEVOS:
  libpkcs11-helper1t64 openvpn
0 actualizados, 2 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 719 kB de archivos.
Se utilizarán 2.054 kB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] S
Des:1 http://ports.ubuntu.com/ubuntu-ports noble/main arm64 libpkcs11-helper1t64 arm64 1.29.0-2.1build2 [49,3 kB]
Des:2 http://ports.ubuntu.com/ubuntu-ports noble-updates/main arm64 openvpn arm64 2.6.14-0ubuntu0.24.04.1 [669 kB]
Descargados 719 kB en 1s (1.026 kB/s)
Preconfigurando paquetes ...
Seleccionando el paquete libpkcs11-helper1t64:arm64 previamente no seleccionado.
(Leyendo la base de datos ... 90886 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../libpkcs11-helper1t64 1.29.0-2.1build2 arm64.deb ...
gabriel@pc1gabriel:~$ sudo apt install easy-rsa
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  libccid libeac3 libpcsclite1 opensc opensc-pkcs11 pcscd
Paquetes sugeridos:
  pcmciautils
Se instalarán los siguientes paquetes NUEVOS:
  easy-rsa libccid libeac3 libpcsclite1 opensc opensc-pkcs11 pcscd
```

Dentro del servidor ejecutaremos estos comandos de easy-rsa para generar los archivos necesarios:

cd ~
sudo /usr/share/easy-rsa/easyrsa init-pki

Inicializa la infraestructura de clave pública (**PKI**: Public Key Infrastructure).
Crea el directorio /pki donde se guardarán certificados, claves privadas y peticiones de firma.

sudo /usr/share/easy-rsa/easyrsa build-ca nopass

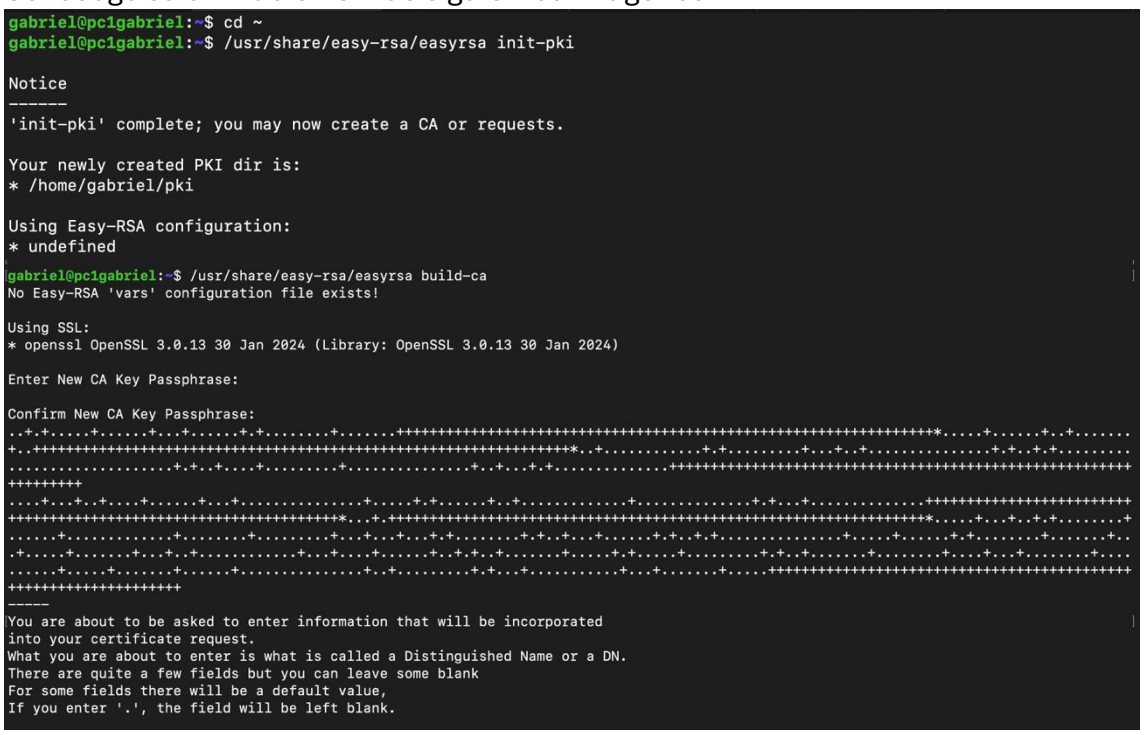
Con este comando anterior se genera la CA que firmará los certificados que se presentan a continuación. Primero el Diffie-Hellman;

sudo /usr/share/easy-rsa/easyrsa gen-dh

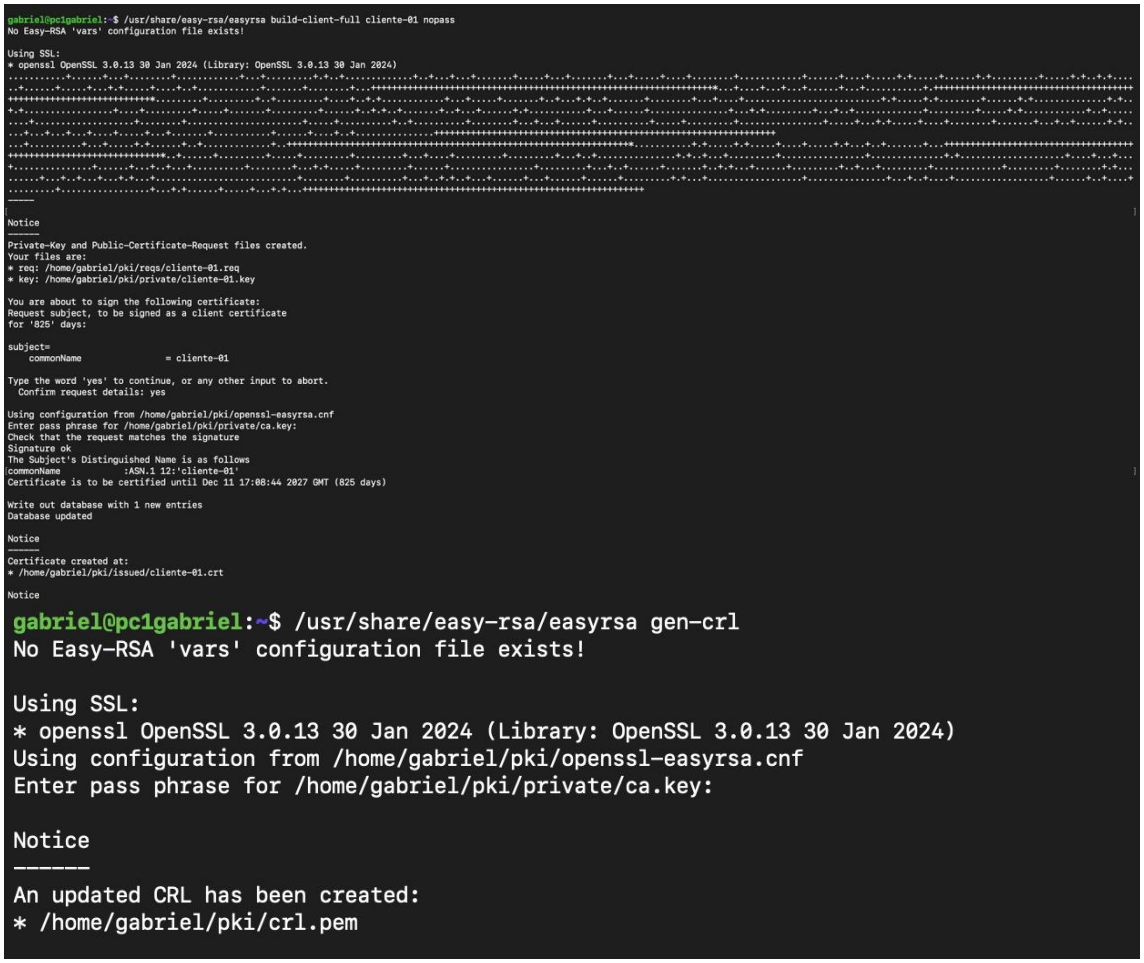
A continuacion el certificado y clave privada del servidor OpenVPN y el del cliente OpenVPN asi como la clave estática utilizada para el tls-auth.

sudo /usr/share/easy-rsa/easyrsa build-server-full servidor nopass
sudo /usr/share/easy-rsa/easyrsa build-client-full cliente-01 nopass
sudo openvpn --genkey secret /etc/openvpn/ta.key

Se recoge su utilización en las siguientes imágenes:




...sh gabriel@40.40.40.30
...gatewaygabriel: ~ — ssh
...ssh gabriel@172.22.0.80
...ssh gabriel@172.22.0.50
...@pc2gabriel: ~ — -bash
...riel@pc1gabriel: ~ — vim
...with Rankings — -bash
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
Common Name (eg: your user, host, or server name) [Easy-RSA CA]:pc1
Notice
CA creation complete. Your new CA certificate is at:
* /home/gabriel/pki/ca.crt
gabriel@pc1gabriel:~$ /usr/share/easy-rsa/easyrsa gen-dh
No Easy-RSA 'vars' configuration file exists!
Using SSL:
* openssl OpenSSL 3.0.13 30 Jan 2024 (Library: OpenSSL 3.0.13 30 Jan 2024)
Generating DH parameters, 2048 bit long safe prime
DH parameters appear to be ok.
Notice
DH parameters of size 2048 created at:
* /home/gabriel/pki/dh.pem
gabriel@pc1gabriel:~$ /usr/share/easy-rsa/easyrsa build-server-full servidor nopass
No Easy-RSA 'vars' configuration file exists!
Using SSL:
* openssl OpenSSL 3.0.13 30 Jan 2024 (Library: OpenSSL 3.0.13 30 Jan 2024)
Notice
Private-Key and Public-Certificate-Request files created.
Your files are:
* req: /home/gabriel/pki/reqs/servidor.req
* key: /home/gabriel/pki/private/servidor.key
You are about to sign the following certificate:
Request subject, to be signed as a server certificate
for '825' days:
subject=
commonName = servidor
Type the word 'yes' to continue, or any other input to abort.
Confirm request details: yes
Using configuration from /home/gabriel/pki/openssl-easyrsa.cnf
Enter pass phrase for /home/gabriel/pki/private/ca.key:
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName :ASN.1 12:'servidor'
Certificate is to be certified until Dec 11 17:08:32 2027 GMT (825 days)
Write out database with 1 new entries
Database updated
Notice
Certificate created at:
* /home/gabriel/pki/issued/servidor.crt
Notice
Inline file created:
* /home/gabriel/pki/inline/servidor.inline



Una vez completados los anteriores comandos, podemos observar los archivos generados:

```
gabriel@pc1gabriel:~$ ls
pki
gabriel@pc1gabriel:~$ ls pki/
ca.crt           crl.pem  index.txt       index.txt.attr.old  inline  openssl-easyrsa.cnf  reqs     serial
certs_by_serial  dh.pem   index.txt.attr  index.txt.old       issued  private              revoked  serial.old
gabriel@pc1gabriel:~$ ls pki/ -a
.   ca.crt           crl.pem  index.txt       index.txt.attr.old  inline  openssl-easyrsa.cnf  reqs     serial
..  certs_by_serial  dh.pem   index.txt.attr  index.txt.old       issued  private              revoked  serial.old
gabriel@pc1gabriel:~$ ls pki/certs_by_serial/ -a
.  ..  06A5C44B8792C97882BF7E7BF2CD37FF.pem  F7477C7269484069C6737104503A8405.pem
gabriel@pc1gabriel:~$ ls pki/private/ -a
.  ..  ca.key  cliente-01.key  servidor.key
gabriel@pc1gabriel:~$ ls pki/
ca.crt               index.txt           inline/             reqs/
certs_by_serial/     index.txt.attr      issued/             revoked/
crl.pem              index.txt.attr.old  openssl-easyrsa.cnf serial
dh.pem               index.txt.old       private/            serial.old
gabriel@pc1gabriel:~$ ls pki/inline/
cliente-01.inline  servidor.inline
gabriel@pc1gabriel:~$ ls pki/issued/
cliente-01.crt  servidor.crt
gabriel@pc1gabriel:~$ openvpn --genkey secret ta.key
gabriel@pc1gabriel:~$ systemctl start openvpn-server@servidor
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ====
Authentication is required to start 'openvpn-server@servidor.service'.
Authenticating as: gabriel
Password:
```

Cuando salga common name, es simplemente el nombre del CA entre servidor y un cliente en específico.

Ahora debemos ubicar cada archivo en el servidor y en el cliente:

Para el servidor nos valdrá con ejecutar esto:

sudo chown -R root:root /etc/openvpn/server
sudo cp /usr/share/easy-rsa/pki/ca.crt /etc/openvpn/server/
sudo cp /usr/share/easy-rsa/pki/issued/servidor.crt /etc/openvpn/server/
sudo cp /usr/share/easy-rsa/pki/private/servidor.key /etc/openvpn/server/
sudo cp /usr/share/easy-rsa/pki/dh.pem /etc/openvpn/server/
sudo cp /etc/openvpn/ta.key /etc/openvpn/server/

Permisos:

sudo chmod 600 /etc/openvpn/server/servidor.key
sudo chmod 600 /etc/openvpn/server/ta.key
sudo chown root:root /etc/openvpn/server/*

Para el cliente, haremos un Jump al GW, ya que no tenemos acceso a ServidorVPN, y asi poder copiar los archivos que necesitamos.

scp -o ProxyJump=root@40.40.40.10 root@172.22.0.80:/usr/share/easy-rsa/pki/ca.crt /etc/openvpn/client/
scp -o ProxyJump=root@40.40.40.10 root@172.22.0.80:/usr/share/easy-rsa/pki/issued/cliente-01.crt /etc/openvpn/client/
scp -o ProxyJump=root@40.40.40.10 root@172.22.0.80:/usr/share/easy-rsa/pki/private/cliente-01.key /etc/openvpn/client/
scp -o ProxyJump=root@40.40.40.10 root@172.22.0.80:/etc/openvpn/ta.key /etc/openvpn/client/

Permisos:

sudo chmod 600 /etc/openvpn/client/cliente-01.key
sudo chown root:root /etc/openvpn/client/*

c) Configurar el servidor OpenVPN en modo TUN (actuará como router), usando el archivo server.conf). Usaremos UDP como protocolo de transporte. Configurar su arranque automático al inicio y arrancar el servicio. También habilitaremos la función de forwarding en este equipo Servidor, para que pueda encaminar el tráfico entre sus interfaces.

Para configurar el servidor es necesario editar los archivos de configuración tanto del servidor como del cliente.

El primero de los dos será el del servidor que editaremos con el comando

```
sudo vim /etc/openvpn/server/servidor.conf
```

Archivos de configuración:
Servidor: "/etc/openvpn/server/servidor.conf"

```
port 1194
proto udp
dev tun

ca /etc/openvpn/server/ca.crt
cert /etc/openvpn/server/servidor.crt
key /etc/openvpn/server/servidor.key
dh /etc/openvpn/server/dh.pem

server 172.16.0.0 255.255.255.0
ifconfig-pool-persist /var/log/openvpn/ipp.txt

# que los clientes conozcan la LAN interna
push "route 172.22.0.0 255.255.255.0"

keepalive 10 120
tls-auth /etc/openvpn/server/ta.key 0
cipher AES-256-CBC
persist-key
persist-tun
verb 4
```

Y posteriormente con el cliente con el comando

```
sudo vim etc/openvpn/client/cliente.conf
```

Importante que en ambos se configuren en modo tun “dev tun” actuando como router y el protocolo “proto udp”.

Cliente: "/etc/openvpn/client/cliente.conf"

```
client
dev tun
proto udp
remote 40.40.40.10 1194
resolv-retry infinite
nobind

ca /etc/openvpn/client/ca.crt
cert /etc/openvpn/client/cliente-01.crt
key  /etc/openvpn/client/cliente-01.key
remote-cert-tls server
tls-auth /etc/openvpn/client/ta.key 1
cipher AES-256-CBC
persist-key
persist-tun
verb 3
```

Arrancamos y lo configuramos para su arranque automatico al inicio

```
sudo systemctl enable openvpn-server@servidor
sudo systemctl start  openvpn-server@servidor
sudo journalctl -u openvpn-server@servidor -f  /Para confirmar .conf
```

En otra terminal de PC1 ejecutamos:

```
sudo openvpn --config /etc/openvpn/client/cliente.conf
sudo systemctl start openvpn-client@cliente-01
sudo journalctl -u openvpn-client@cliente-01 -f
```

Y ponemos la variable también en el servidor de ip_foward a 1:

```
sudo sysctl -w net.ipv4.ip_forward=1
```

d) Configurar en el Gateway un port-forwarding para reenviar al servidor interno todo el tráfico recibido en el puerto udp/1194 de su interface externo (es necesaria una regla iptables nat en PREROUTING).

Se usa el siguiente comando que reenviara el trafico del Gateway al servidor.

```
sudo iptables -t nat -A PREROUTING -i eth1 -p udp --dport 1194 -j DNAT --to-destination 172.22.0.80:1194
```

e) Configurar en el Gateway el enmascaramiento para todo el tráfico saliente Comprobar que siguen pudiendo alcanzarse recursos externos desde la red interna (PC2 a PC1). Verificar que se realiza el enmascaramiento usando tcpdump

Permitirnos forwarding:

```
sudo iptables -A FORWARD -p udp -d 172.22.0.80 --dport 1194 -j ACCEPT
```

Enmascaramiento en el GW:

```
sudo iptables -t nat -A POSTROUTING -s 172.22.0.0/24 -o eth1 -j MASQUERADE
sudo iptables -t nat -A POSTROUTING -s 40.40.40.0/24 -o eth0 -j MASQUERADE
sudo iptables -t nat -A POSTROUTING -s 172.22.0.0/24 -o eth0 -j MASQUERADE
```

Una vez realizado los comandos anteriores comprobaos que el PC2 hace ping a PC1:

```
root@server:~# ping 40.40.40.30
PING 40.40.40.30 (40.40.40.30) 56(84) bytes of data.
64 bytes from 40.40.40.30: icmp_seq=1 ttl=63 time=1.39 ms
64 bytes from 40.40.40.30: icmp_seq=2 ttl=63 time=2.69 ms
64 bytes from 40.40.40.30: icmp_seq=3 ttl=63 time=2.06 ms
64 bytes from 40.40.40.30: icmp_seq=4 ttl=63 time=2.67 ms
64 bytes from 40.40.40.30: icmp_seq=5 ttl=63 time=2.59 ms
64 bytes from 40.40.40.30: icmp_seq=6 ttl=63 time=2.85 ms
```

Y verificamos haciendo uso del *tcpdump*:

GW – eth1:

```
root@server:~# sudo tcpdump -i eth1 -n icmp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), snapshot length 262144 bytes
10:16:07.574156 IP 40.40.40.10 > 40.40.40.30: ICMP echo request, id 1694, seq 1, length 64
10:16:07.574801 IP 40.40.40.30 > 40.40.40.10: ICMP echo reply, id 1694, seq 1, length 64
10:16:08.576502 IP 40.40.40.10 > 40.40.40.30: ICMP echo request, id 1694, seq 2, length 64
10:16:08.577634 IP 40.40.40.30 > 40.40.40.10: ICMP echo reply, id 1694, seq 2, length 64
10:16:09.578511 IP 40.40.40.10 > 40.40.40.30: ICMP echo request, id 1694, seq 3, length 64
10:16:09.579415 IP 40.40.40.30 > 40.40.40.10: ICMP echo reply, id 1694, seq 3, length 64
10:16:10.580230 IP 40.40.40.10 > 40.40.40.30: ICMP echo request, id 1694, seq 4, length 64
10:16:10.581344 IP 40.40.40.30 > 40.40.40.10: ICMP echo reply, id 1694, seq 4, length 64
10:16:11.582170 IP 40.40.40.10 > 40.40.40.30: ICMP echo request, id 1694, seq 5, length 64
10:16:11.583408 IP 40.40.40.30 > 40.40.40.10: ICMP echo reply, id 1694, seq 5, length 64
10:16:12.584308 IP 40.40.40.10 > 40.40.40.30: ICMP echo request, id 1694, seq 6, length 64
10:16:12.585562 IP 40.40.40.30 > 40.40.40.10: ICMP echo reply, id 1694, seq 6, length 64
```

GW-eth0:

```
root@server:~# sudo tcpdump -i eth0 -n icmp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
10:16:07.574106 IP 172.22.0.50 > 40.40.40.30: ICMP echo request, id 1694, seq 1, length 64
10:16:07.574813 IP 40.40.40.30 > 172.22.0.50: ICMP echo reply, id 1694, seq 1, length 64
10:16:08.576459 IP 172.22.0.50 > 40.40.40.30: ICMP echo request, id 1694, seq 2, length 64
10:16:08.577661 IP 40.40.40.30 > 172.22.0.50: ICMP echo reply, id 1694, seq 2, length 64
10:16:09.578467 IP 172.22.0.50 > 40.40.40.30: ICMP echo request, id 1694, seq 3, length 64
10:16:09.579435 IP 40.40.40.30 > 172.22.0.50: ICMP echo reply, id 1694, seq 3, length 64
10:16:10.580184 IP 172.22.0.50 > 40.40.40.30: ICMP echo request, id 1694, seq 4, length 64
10:16:10.581370 IP 40.40.40.30 > 172.22.0.50: ICMP echo reply, id 1694, seq 4, length 64
10:16:11.582127 IP 172.22.0.50 > 40.40.40.30: ICMP echo request, id 1694, seq 5, length 64
10:16:11.583435 IP 40.40.40.30 > 172.22.0.50: ICMP echo reply, id 1694, seq 5, length 64
10:16:12.584259 IP 172.22.0.50 > 40.40.40.30: ICMP echo request, id 1694, seq 6, length 64
10:16:12.585588 IP 40.40.40.30 > 172.22.0.50: ICMP echo reply, id 1694, seq 6, length 64
gabriel@gatewaygabriel:~$ sudo iptables -t nat -A PREROUTING -p udp --dport 1194 -i ens160 -j DNAT --to-destination 172.22.0.80:1194
gabriel@gatewaygabriel:~$ sudo iptables -t nat -L -n -v
Chain PREROUTING (policy ACCEPT 302 packets, 22130 bytes)
 pkts bytes target     prot opt in     out     source               destination
    4   280 DNAT       17   --  ens160 *       0.0.0.0/0            0.0.0.0/0           udp dpt:1194 to:172.22.0.80:1194

Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain POSTROUTING (policy ACCEPT 2745 packets, 195K bytes)
 pkts bytes target     prot opt in     out     source               destination
    5   309 MASQUERADE  0    --  *      ens256  0.0.0.0/0            0.0.0.0/0
gabriel@gatewaygabriel:~$
```

f) Configurar el cliente OpenVPN en PC1 (client.conf) para conectar al servidor (a través de la IP del Gateway). Para completar esta tarea será necesario copiar los archivos necesarios obtenidos en el apartado 2

Ya se copiaron a la hora de generarlos previamente, pero en el caso de no haberlo hecho todavía se deja preparado para arrancar el servicio con todo ya listo.

Ahora en el servidor arrancamos el servicio vpn:

sudo systemctl enable openvpn-server@servidor
sudo systemctl start openvpn-server@servidor
sudo journalctl -u openvpn-server@servidor -f /Para confirmar .conf

En otra terminal de PC1 ejecutamos:

sudo openvpn --config /etc/openvpn/client/cliente.conf
sudo systemctl start openvpn-client@cliente-01

```
sudo journalctl -u openvpn-client@cliente-01 -f
```

g) Iniciar el cliente y comprobar que se conecta al servidor. Verificar la configuración que se establece en el interface virtual tun del cliente, su tabla de encaminamiento y que es posible alcanzar desde PC1 los recursos ofrecidos por el Servidor a través de la VPN (puede hacer una captura del tráfico intercambiado para verificar que los paquetes se transportan sobre tun0, que a su vez se envía encriptado sobre eth0). Desactivar el router por defecto en PC1 y comprobar que sigue siendo posible acceder al Servidor.

PC1:

```
root@server:~# 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 noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:85:07:02 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    altname ens33
    inet 40.40.40.30/24 brd 40.40.40.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe85:702/64 scope link
       valid_lft forever preferred_lft forever
6: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UNKNOWN group default
 qlen 500
    link/none
    inet 172.16.0.6 peer 172.16.0.5/32 scope global tun0
       valid_lft forever preferred_lft forever
    inet6 fe80::781b:4cfd:b47a:af27/64 scope link stable-privacy
       valid_lft forever preferred_lft forever
```

Servidor:

```
root@server:~# 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 noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:e0:f2:f1 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    altname ens33
    inet 172.22.0.80/24 brd 172.22.0.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fee0:f2f1/64 scope link
       valid_lft forever preferred_lft forever
3: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UNKNOWN group default
 qlen 500
    link/none
    inet 172.16.0.1 peer 172.16.0.2/32 scope global tun0
       valid_lft forever preferred_lft forever
    inet6 fe80::d2d7:6447:cb42:f62e/64 scope link stable-privacy
       valid_lft forever preferred_lft forever
```

Desde otra terminal en PC1, ejecutamos este comando para establecer la conexión:

```
sudo openvpn –config /etc/openvpn/client/cliente.conf
```

```
root@server:~# sudo openvpn --config /etc/openvpn/client/cliente.conf
2025-09-21 15:39:07 DEPRECATED OPTION: --cipher set to 'AES-256-CBC' but missing in --data-ciphers (AES-256-GCM:AES
-128-GCM:CHACHA20-POLY1305). OpenVPN ignores --cipher for cipher negotiations.
2025-09-21 15:39:07 Note: Kernel support for ovpn-dco missing, disabling data channel offload.
2025-09-21 15:39:07 OpenVPN 2.6.14 x86_64-pc-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4] [EPOLL] [PKCS11] [MH/PKTINFO] [A
EAD] [DCO]
2025-09-21 15:39:07 library versions: OpenSSL 3.0.13 30 Jan 2024, LZO 2.10
2025-09-21 15:39:07 DCO version: N/A
2025-09-21 15:39:07 TCP/UDP: Preserving recently used remote address: [AF_INET]40.40.40.10:1194
2025-09-21 15:39:07 Socket Buffers: R=[212992->212992] S=[212992->212992]
2025-09-21 15:39:07 UDPv4 link local: (not bound)
2025-09-21 15:39:07 UDPv4 link remote: [AF_INET]40.40.40.10:1194
2025-09-21 15:39:07 TLS: Initial packet from [AF_INET]40.40.40.10:1194, sid=990174d5 568ddc1d
2025-09-21 15:39:07 VERIFY OK: depth=1, CN=vpn-ca
2025-09-21 15:39:07 VERIFY KU OK
2025-09-21 15:39:07 Validating certificate extended key usage
2025-09-21 15:39:07 ++ Certificate has EKU (str) TLS Web Server Authentication, expects TLS Web Server Authenticati
on
2025-09-21 15:39:07 VERIFY EKU OK
2025-09-21 15:39:07 VERIFY OK: depth=0, CN=servidor
2025-09-21 15:39:07 Control Channel: TLSv1.3, cipher TLSv1.3 TLS_AES_256_GCM_SHA384, peer certificate: 2048 bits RS
A, signature: RSA-SHA256, peer temporary key: 253 bits X25519
2025-09-21 15:39:07 [servidor] Peer Connection Initiated with [AF_INET]40.40.40.10:1194
2025-09-21 15:39:07 TLS: move_session: dest=TM_ACTIVE src=TM_INITIAL reinit_src=1
2025-09-21 15:39:07 TLS: tls_multi_process: initial untrusted session promoted to trusted
2025-09-21 15:39:07 PUSH: Received control message: 'PUSH_REPLY,route 172.22.0.0 255.255.255.0,route 172.16.0.1,top
ology net30,ping 10,ping-restart 120,ifconfig 172.16.0.6 172.16.0.5,peer-id 1,cipher AES-256-GCM,protocol-flags cc-
exit tls-ekm dyn-tls-crypt,tun-mtu 1500'
2025-09-21 15:39:07 OPTIONS IMPORT: --ifconfig/up options modified
2025-09-21 15:39:07 OPTIONS IMPORT: route options modified
2025-09-21 15:39:07 OPTIONS IMPORT: tun-mtu set to 1500
2025-09-21 15:39:07 net_route_v4_best_gw query: dst 0.0.0.0
2025-09-21 15:39:07 net_route_v4_best_gw result: via 0.0.0.0 dev
2025-09-21 15:39:07 ROUTE_GATEWAY 0.0.0.0
2025-09-21 15:39:07 TUN/TAP device tun0 opened
2025-09-21 15:39:07 net_iface_mtu_set: mtu 1500 for tun0
2025-09-21 15:39:07 net_iface_up: set tun0 up
2025-09-21 15:39:07 net_addr_ptp_v4_add: 172.16.0.6 peer 172.16.0.5 dev tun0
2025-09-21 15:39:07 net_route_v4_add: 172.22.0.0/24 via 172.16.0.5 dev [NULL] table 0 metric -1
2025-09-21 15:39:07 net_route_v4_add: 172.16.0.1/32 via 172.16.0.5 dev [NULL] table 0 metric -1
2025-09-21 15:39:07 Initialization Sequence Completed
2025-09-21 15:39:07 Data Channel: cipher 'AES-256-GCM', peer-id: 1
2025-09-21 15:39:07 Timers: ping 10, ping-restart 120
2025-09-21 15:39:07 Protocol options: protocol-flags cc-exit tls-ekm dyn-tls-crypt
```

Habiendo comentado la línea del gateway4/router por defecto comprobamos que es posible acceder al servidor.

PC1 ping Servidor:

```
root@server:~# ping 172.22.0.80
PING 172.22.0.80 (172.22.0.80) 56(84) bytes of data.
64 bytes from 172.22.0.80: icmp_seq=1 ttl=64 time=1.53 ms
64 bytes from 172.22.0.80: icmp_seq=2 ttl=64 time=1.94 ms
64 bytes from 172.22.0.80: icmp_seq=3 ttl=64 time=3.20 ms
^C
--- 172.22.0.80 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 1.526/2.220/3.195/0.709 ms
root@server:~# ping 172.16.0.6
PING 172.16.0.6 (172.16.0.6) 56(84) bytes of data.
64 bytes from 172.16.0.6: icmp_seq=1 ttl=64 time=2.86 ms
64 bytes from 172.16.0.6: icmp_seq=2 ttl=64 time=0.039 ms
64 bytes from 172.16.0.6: icmp_seq=3 ttl=64 time=0.046 ms
^C
--- 172.16.0.6 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2063ms
rtt min/avg/max/mdev = 0.039/0.982/2.861/1.328 ms
```

Servidor ping PC1:

```
root@server:~# ping 172.16.0.6
PING 172.16.0.6 (172.16.0.6) 56(84) bytes of data.
64 bytes from 172.16.0.6: icmp_seq=1 ttl=64 time=1.54 ms
64 bytes from 172.16.0.6: icmp_seq=2 ttl=64 time=2.55 ms
64 bytes from 172.16.0.6: icmp_seq=3 ttl=64 time=2.73 ms
^C
--- 172.16.0.6 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 1.544/2.273/2.726/0.520 ms
```

h) Compruebe si es posible alcanzar el equipo PC2 desde PC1 a través de la conexión VPN. ¿Qué está sucediendo?

No funciona debido a que el paquete desde PC1 le llega a PC2, pero este no sabe por dónde devolverlo.

i) Modificar la configuración de PC2 para que pueda comunicar con PC1 a través de la VPN. Nota: también es posible configurar el Gateway para el retorno de los paquetes a través de la VPN, sin modificar el encaminamiento en PC2. Compruebe ambas opciones.

Para conseguir la conexión a través de la VPN, hemos realizado los siguientes cambios/pasos:

Modificar netplan de GW:

```
# This is the network config written by 'subiquity'
network:
  version: 2
  ethernets:
    eth0:
      dhcp4: no
      addresses: [172.22.0.10/24]
      routes:
        - to: 172.16.0.0/24
          via: 172.22.0.80
        #routes:
        #- to: default
        #  via: 172.22.0.1
        #nameservers:
        #addresses: [8.8.8.8, 1.1.1.1]
    eth1:
      dhcp4: no
      addresses: [40.40.40.10/24]
```

Modificamos netplan PC2:

```
# This is the network config written by 'subiquity'
network:
  version: 2
  ethernets:
    eth0:
      dhcp4: no
      addresses: [172.22.0.50/24]
      routes:
        - to: 172.16.0.0/24
          via: 172.22.0.10
      nameservers:
        addresses: [8.8.8.8,1.1.1.1]
```

Ejecutamos lo siguiente en el GW:

sudo ip route add 172.16.0.0/24 via 172.22.0.80 dev eth0
sudo iptables -A FORWARD -s 172.22.0.0/24 -d 172.16.0.0/24 -j ACCEPT
sudo iptables -A FORWARD -s 172.16.0.0/24 -d 172.22.0.0/24 -j ACCEPT

Ejecutamos lo siguiente en el ServidorVPN, añadir como ya sabemos para persistencia:

sudo sysctl -w net.ipv4.ip_forward=1

```
gabriel@pc2gabriel:~$ ping 172.16.0.6
PING 172.16.0.6 (172.16.0.6) 56(84) bytes of data.
64 bytes from 172.16.0.6: icmp_seq=1 ttl=62 time=2.90 ms
64 bytes from 172.16.0.6: icmp_seq=2 ttl=62 time=1.53 ms
64 bytes from 172.16.0.6: icmp_seq=3 ttl=62 time=1.53 ms
```

Partes opcionales:

j) Realizar una nueva configuración de la conexión OpenVPN, ahora en modo TAP o bridged. Para ello, en el servidor debe configurarse un bridge entre el interface tap (normalmente tap0) y el interface de conexión a la LAN. Esta configuración se realiza con un script adicional (suele incluirse dentro del paquete de OpenVPN). En el modo bridged, hay que asociar una IP dinámica de la red destino al cliente VPN. Esto puede hacerse con el servicio DHCP de la red destino, o reservando un rango de direcciones en el propio servidor VPN (en la directiva server-bridge)

En el servidor:

sudo apt install bridge-utils

sudo ip link add name br0 type bridge sudo ip link set dev eth0 master br0 sudo ip link set dev tap0 master br0

```
sudo ip link set br0 up
```

Comenzamos creando los files /etc/openvpn/bridge-up.sh y /etc/openvpn/bridge-down.sh:

```
#!/bin/bash
# args: <bridge-name> <physical-iface> <tap-iface>
BR="$1"
PHY="$2"
TAP="$3"

# move IP from PHY to BR
IP_INFO=$(ip -4 addr show dev $PHY | awk '/inet /{print $2; exit}')
# create bridge if not exists
ip link add name $BR type bridge 2>/dev/null || true
# set interfaces down, assign master
ip link set dev $PHY down
ip link set dev $TAP down
ip link set dev $PHY master $BR
ip link set dev $TAP master $BR

# assign previous IP to bridge
if [ -n "$IP_INFO" ]; then
  ip addr flush dev $PHY
  ip addr add $IP_INFO dev $BR
fi

# bring up bridge and members
ip link set dev $BR up
ip link set dev $PHY up
ip link set dev $TAP up

# enable STP off (optional), set forward delay small
ip link set dev $BR type bridge stp_state 0
```

```
#!/bin/bash
BR="$1"
PHY="$2"
TAP="$3"

# bring down tap, remove from bridge
ip link set dev $TAP down
ip link set dev $TAP nomaster || true
ip link set dev $PHY nomaster || true

# move IP back to PHY (if br had one)
IP_INFO=$(ip -4 addr show dev $BR | awk '/inet /{print $2; exit}')
if [ -n "$IP_INFO" ]; then
```

```
  ip addr flush dev $BR
  ip addr add $IP_INFO dev $PHY
fi

ip link set dev $BR down
ip link del $BR 2>/dev/null || true
ip link set dev $PHY up
```

Entramos en servidor.conf, y como tenemos dhcp activado, no tenemos que habilitar server-bridge. Solo vamos a cambiar dev tun por dev tap en server.conf y cliente.conf.

Despues reinciamos openvpn:

```
sudo systemctl restart openvpn-server@servidor.service
```

Ha abrimos una terminal con PC1 y probamos a iniciar la conexión, si todo sale bien se habrá añadido una ip a PC1 y se podrán hacer ping entre ellas:

```
root@server:~# sudo openvpn --config /etc/openvpn/client/cliente.conf
2025-09-26 13:32:12 DEPRECATED OPTION: --cipher set to 'AES-256-CBC' but missing in --data-ciphers (AE
S-256-GCM:AES-128-GCM:CHACHA20-POLY1305). OpenVPN ignores --cipher for cipher negotiations.
2025-09-26 13:32:12 Note: dev-type not tun, disabling data channel offload.
2025-09-26 13:32:12 OpenVPN 2.6.14 x86_64-pc-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4] [EPOLL] [PKCS11] [M
H/PKTINFO] [AEAD] [DCO]
2025-09-26 13:32:12 library versions: OpenSSL 3.0.13 30 Jan 2024, LZO 2.10
2025-09-26 13:32:12 DCO version: N/A
2025-09-26 13:32:12 TCP/UDP: Preserving recently used remote address: [AF_INET]40.40.40.10:1194
2025-09-26 13:32:12 Socket Buffers: R=[212992->212992] S=[212992->212992]
2025-09-26 13:32:12 UDPv4 link local: (not bound)
2025-09-26 13:32:12 UDPv4 link remote: [AF_INET]40.40.40.10:1194
2025-09-26 13:32:12 read UDPv4 [ECONNREFUSED]: Connection refused (fd=3,code=111)
^C2025-09-26 13:32:13 event_wait : Interrupted system call (fd=-1,code=4)
2025-09-26 13:32:13 SIGINT[hard,] received, process exiting
root@server:~# sudo openvpn --config /etc/openvpn/client/cliente.conf
2025-09-26 13:33:50 DEPRECATED OPTION: --cipher set to 'AES-256-CBC' but missing in --data-ciphers (AE
S-256-GCM:AES-128-GCM:CHACHA20-POLY1305). OpenVPN ignores --cipher for cipher negotiations.
2025-09-26 13:33:50 Note: dev-type not tun, disabling data channel offload.
2025-09-26 13:33:50 OpenVPN 2.6.14 x86_64-pc-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4] [EPOLL] [PKCS11] [M
H/PKTINFO] [AEAD] [DCO]
```

ServidorVPN

```
root@server:~# ping 172.16.0.4
PING 172.16.0.4 (172.16.0.4) 56(84) bytes of data.
64 bytes from 172.16.0.4: icmp_seq=1 ttl=64 time=1.93 ms
64 bytes from 172.16.0.4: icmp_seq=2 ttl=64 time=2.48 ms
64 bytes from 172.16.0.4: icmp_seq=3 ttl=64 time=2.27 ms
^C
```

PC1

```
root@server:~# ping 172.16.0.1
PING 172.16.0.1 (172.16.0.1) 56(84) bytes of data.
64 bytes from 172.16.0.1: icmp_seq=1 ttl=64 time=2.87 ms
64 bytes from 172.16.0.1: icmp_seq=2 ttl=64 time=2.54 ms
64 bytes from 172.16.0.1: icmp_seq=3 ttl=64 time=2.59 ms
64 bytes from 172.16.0.1: icmp_seq=4 ttl=64 time=1.96 ms
^C
```

k) Estudiar las características de la solución VPN WireGuard (www.wireguard.com). Preparar un escenario para poner en marcha una configuración básica de esta VPN (puede usarse como base la maqueta anterior).

No hace falta desactivar opnevpn (usan diferentes puertos).

Comenzamos instalando en el PC1 y en el Servidor "wireguard"

```
sudo apt update
sudo apt install wireguard
```

Ahora debemos crear las claves:

Para el ServidorVPN:

```
wg genkey | tee /etc/wireguard/server.key | wg pubkey > /etc/wireguard/server.pub
```

Para PC1:

```
wg genkey | tee /etc/wireguard/client.key | wg pubkey > /etc/wireguard/client.pub
```

Posteriormente vamos a crear los archivos de configuración para wg0 en cliente y servidor:

PC1: vim /etc/wireguard/wg0.conf

```
[Interface]
PrivateKey = 6GkgKNWHVb...                    cat /etc/wireguard/client.key
Address = 10.8.0.2/24

[Peer]
PublicKey = pu41vFHU0YDC...                   cat /etc/wireguard/servidor.pub
Endpoint = 40.40.40.10:51820
AllowedIPs = 172.22.0.0/24, 10.8.0.0/24
PersistentKeepalive = 25
```

ServidorVPN: vim /etc/wireguard/wg0.conf

```
[Interface]
PrivateKey = KJi7hxusZtJo...                  cat /etc/wireguard/server.key
Address = 10.8.0.1/24
ListenPort = 51820

# Peer PC1
[Peer]
PublicKey = oR57icy5ORGlT...                  cat /etc/wireguard/client.pub
AllowedIPs = 10.8.0.2/32
```

Debemos ejecutar estos comandos (Reglas NAT) en el Servidor:

```
sudo sysctl -w net.ipv4.ip_forward=1
sudo iptables -t nat -A POSTROUTING -s 10.8.0.0/24 -o eth0 -j MASQUERADE
```

En el GW:

```
sudo iptables -t nat -A PREROUTING -i eth1 -p udp --dport 51820 -j DNAT --to-destination 172.22.0.80:51820
```

```
sudo iptables -A FORWARD -p udp -d 172.22.0.80 --dport 51820 -j ACCEPT
```

Para arrancar el servicio:
Servidor y PC1:

```
sudo wg-quick up wg0
sudo systemctl enable wg-quick@wg0
```

Ver el estado de la interfaz:

```
sudo wg show
```

Comprobar hacer pings:
Desde PC1 –> ServidorVPN: ping 10.8.0.1
Desde ServidorVPN -> PC1: ping 10.8.0.2

Para cualquier problema, podemos usar tcpdump para ver el tráfico:

```
sudo tcpdump -n -i any udp port 51820 -vv
```

Caso conectividad PC1 a PC2:
Solución, crear una interfaz más general:
Primero borramos la regla anterior, y añadimos la general en ***ServidorVPN***

```
sudo iptables -t nat -D POSTROUTING -s 10.8.0.0/24 -o eth0 -j MASQUERADE
2>/dev/null || true
sudo iptables -t nat -A POSTROUTING -s 10.8.0.0/24 -j MASQUERADE
```