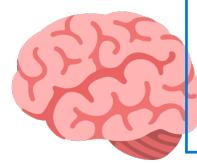


Conocimientos previos



Historia de Linux ↗ [comandos básicos de Linux](#) ↗ [Curso básico desde 0 \[Linux Mint / Ubuntu / Centos / Fedora\] - YouTube](#)

Comandos Básicos ↗ [curso LINUX - ¿qué es BASH SCRIPTING?- Como hacer una CALCULADORA con VARIABLES en LINUX en VIM - YouTube](#)

Primeros Scripts ↗ [Menú de OPCIONES en BASH](#) ✓ [estructuras WHILE y CASE - YouTube](#)

Usuarios/Grupos/Permisos ↗ [USUARIOS GRUPOS Y PERMISOS Linux](#) ↗ [\[EJEMPLOS\] COMANDOS useradd SUDO y CHMOD - YouTube](#)

SSH

Comandos para distribuciones Red Hat

SSH (Secure Shell) es un protocolo de red seguro que permite a los usuarios **acceder y administrar sistemas remotos de manera segura**. Proporciona una forma cifrada de comunicación entre dos dispositivos, lo que significa que la información que se envía y recibe a través de SSH está encriptada y protegida de posibles amenazas de seguridad. SSH es ampliamente utilizado para administrar servidores remotos, transferir archivos de manera segura y ejecutar comandos en máquinas remotas.

Comandos útiles

```
dnf install openssh-server
```

```
[root@fedoraserver ~]# systemctl status sshd
● sshd.service - OpenSSH server daemon
   Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; preset: enabled)
   Drop-In: /usr/lib/systemd/system/service.d
             └─10-timeout-abort.conf
     Active: active (running) since Tue 2023-08-08 14:47:18 -03; 11min ago
```

```
vim /etc/ssh/sshd_config
```

Dentro del **archivo de configuración** se puede cambiar el número de puerto, permitir o denegar acceso a usuarios, configurar la autenticación y más ...

```
# To modify the system-wide sshd configuration, create a *.conf file under
# /etc/ssh/sshd_config.d/ which will be automatically included below
Include /etc/ssh/sshd_config.d/*.conf

# If you want to change the port on a SELinux system, you have to tell
# SELinux about this change.
# semanage port -a -t ssh_port_t -p tcp #PORTNUMBER
#
#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
PermitRootLogin no
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes

# The default is to check both .ssh/authorized_keys and .ssh/authorized_keys2
# but this is overridden so installations will only check .ssh/authorized_keys
AuthorizedKeysFile      .ssh/authorized_keys
```

Recordar permitir el tráfico SSH (**puerto 22**).

```
[root@fedoraserver ~]# firewall-cmd --add-service=ssh
success
```

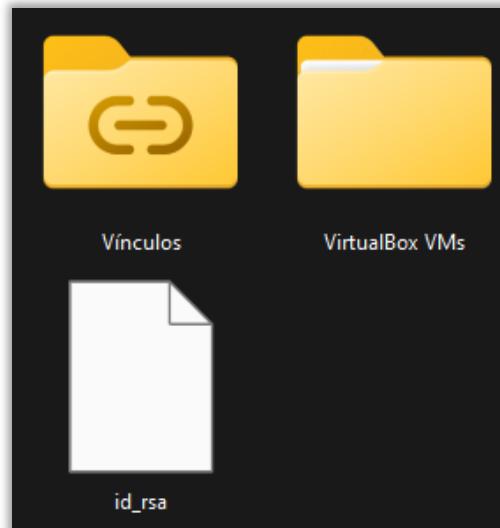
*La sintaxis dependerá del firewall y distribución que se esté utilizando.

Pueden generarse un **par de claves SSH** para la autenticación (eliminando o no la necesidad de una contraseña).

```
[root@fedoraserver ssh]# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa
Your public key has been saved in /root/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:9sf+JGXwvGSLgU+g9m4JDrybYe1P35rMHR0nmcmhys root@fedoraserver
The key's randomart image is:
+---[RSA 3072]----+
|                               |
|                               |
|                               |
|                               |
|                               |
|                               |
|                               |
|                               |
|                               |
+---[SHA256]-----+
```

Por cada par de claves pública/privada:

La **privada se copia al cliente** (directorio raíz del usuario) y la **.pub queda en el servidor**, agregándose al archivo \$HOME/.ssh/authorized_keys (se agrega automáticamente la primera vez).



Al conectarse desde el cliente se puede utilizar la siguiente sintaxis (indicando o no la clave privada, dependiendo del tipo de autenticación seleccionada).

```
C:\Users\santi>ssh -i id_rsa usuariorremoto@192.168.1.13
Web console: https://fedoraserver:9090/ or https://192.168.1.13:9090/
Last login: Tue Aug  8 15:49:29 2023
[usuariorremoto@fedoraserver ~]$
```

Transferencia de archivos (FTP, SCP, SMB).

A la hora de transferir archivos entre Windows y Linux existen varias posibilidades. Cada protocolo tiene sus características particulares y dependiendo el contexto, puede interesar más uno u otro.

FTP (puertos 20 y 21)

FTP (File Transfer Protocol) es un protocolo estándar utilizado para transferir archivos entre computadoras a través de una red. Es ampliamente utilizado para compartir

```
dnf install vsftpd  
systemctl start vsftpd
```

archivos y administrar sitios web, ya que permite a los usuarios cargar y descargar archivos desde y hacia un servidor remoto de manera eficiente.

***Debo permitir las comunicaciones desde el firewall y en caso de que se utilice el modo pasivo para conectarse, es necesario abrir otro rango de puertos e indicarlos en el archivo de configuración.**

```
#  
# Allow anonymous FTP? (Beware - allowed by default if you comment this out)  
anonymous_enable=NO  
#  
# Uncomment this to allow local users to log in.  
local_enable=YES  
#  
# Uncomment this to enable any form of FTP write command.  
write_enable=YES  
#  
# Default umask for local users is 077. You may wish to change this to 022,  
# if your users expect that (022 is used by most other ftptd's)  
local_umask=022
```

/etc/vsftpd/vsftpd.conf

```
rsa_cert_file=/etc/pki/tls/certs/vsftpd.crt  
rsa_private_key_file=/etc/pki/tls/private/vsftpd.key  
  
pasv_min_port=31500  
pasv_max_port=32500  
ssl_enable=yes
```

Rango puertos del modo pasivo e implementación SSL/TLS.

```
chroot_local_user=YES  
allow_writeable_chroot=YES
```



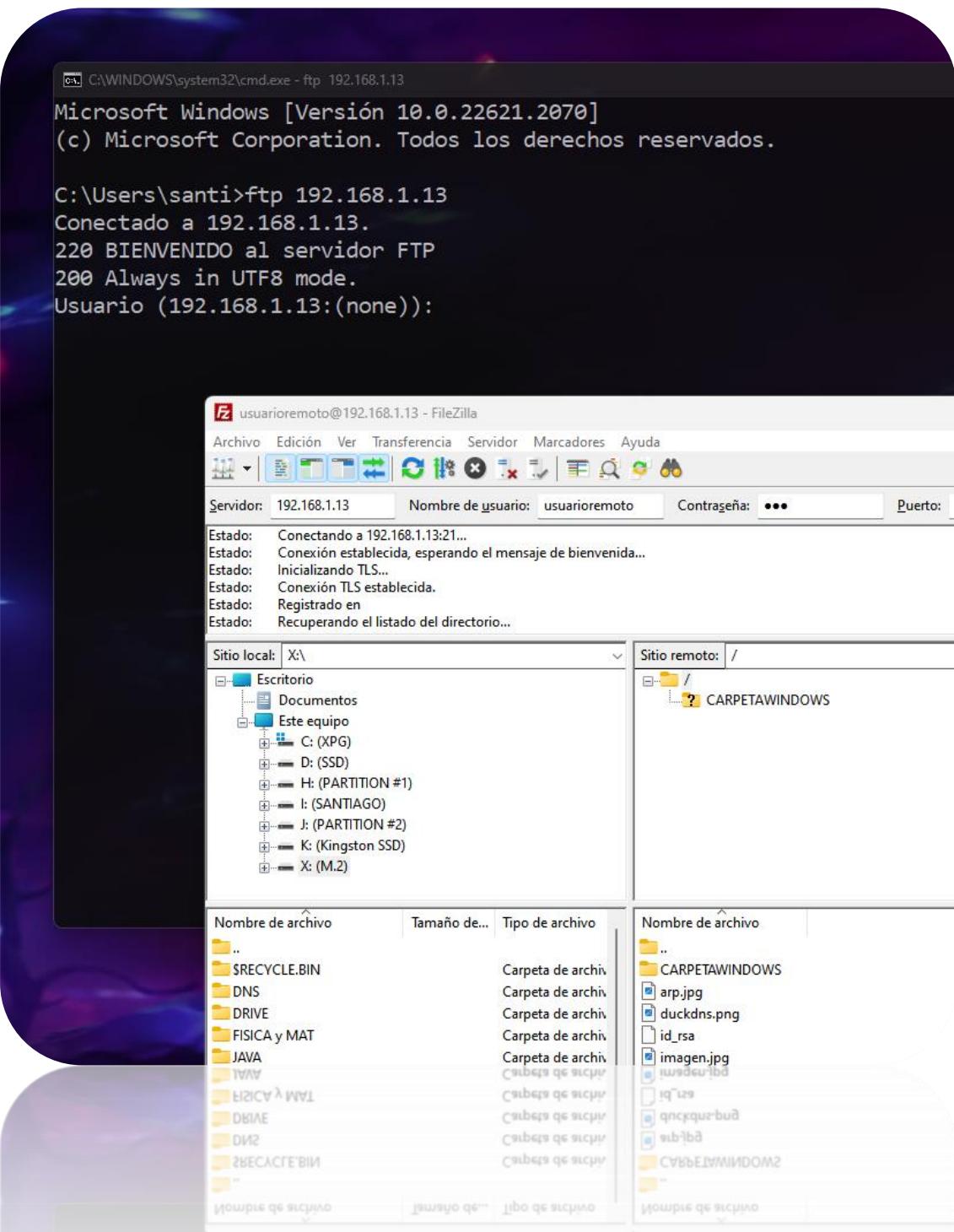
Enjaular usuario.

Recordar que FTP de base es un **protocolo inseguro** ya que no utiliza ningún tipo de cifrado en la comunicación.

[COMO configurar FTP Linux en servidor CentOs/FEDORA \[vsftpd\] Claves RSA, TLS y monitoreo con WIRESHARK - YouTube](#)

Es necesario tomar las medidas adecuadas para proteger un servidor FTP, como utilizar contraseñas seguras, limitar el acceso según sea necesario y considerar el uso de conexiones seguras (**FTP sobre SSL/TLS**)

Desde el cliente podemos conectarnos con aplicaciones como **FileZilla** o directamente utilizar un terminal de comandos para transferir archivos.



SCP (puerto 22) → depende de SSH

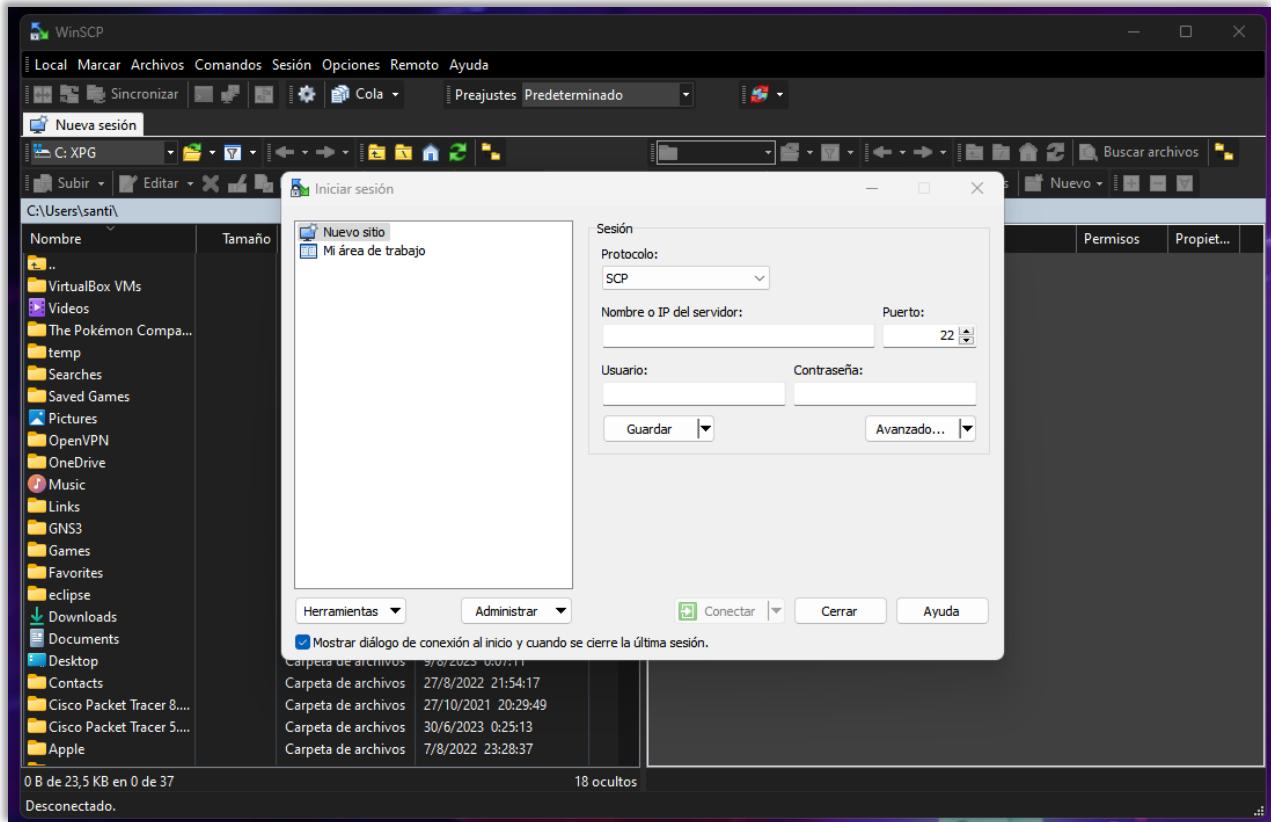
Si la conexión SSH está funcionando se puede utilizar el comando SCP para realizar una copia utilizando ese canal seguro.

El comando **scp** puede utilizarse desde Linux o en Windows PowerShell.

La sintaxis es similar al comando cp teniendo en cuenta que el origen o el destino es un equipo remoto (se necesita usuario y host para establecer la conexión).

```
Seleccionar Windows PowerShell
PS C:\Users\santi> scp usuarioremoto@192.168.1.13:/home/ssh/imagen.jpg C:\Users\santi\Desktop
usuarioremoto@192.168.1.13's password:
imagen.jpg                                         100%    77KB
PS C:\Users\santi>
```

También pueden transferirse archivos utilizando este protocolo con herramientas gráficas como **WinSCP o similares.**

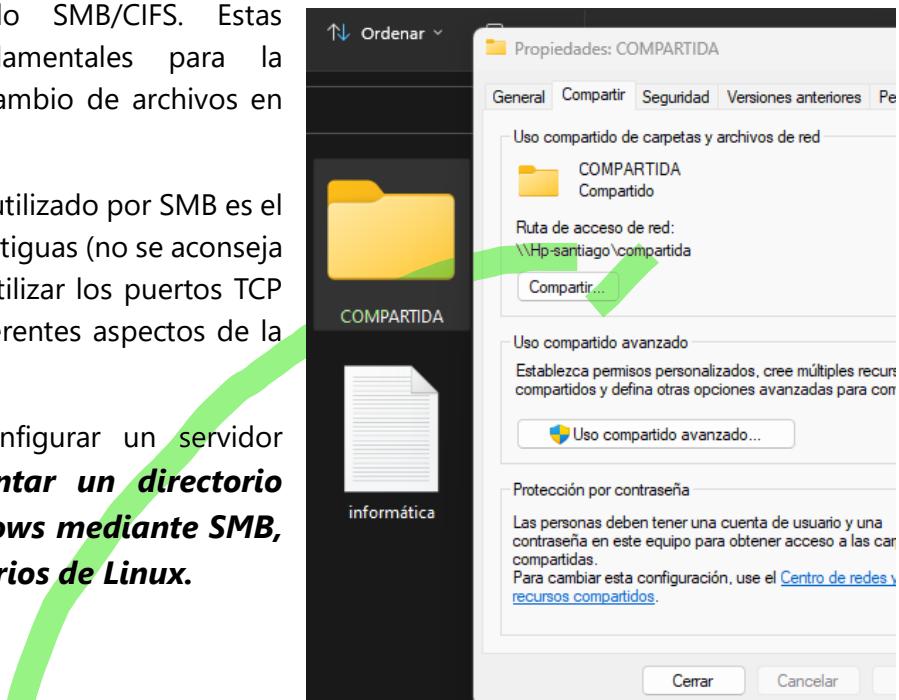


SMB/CIFS/SAMBA

SMB es un protocolo utilizado para compartir archivos y recursos en redes, mientras que **CIFS** es una extensión modernizada de SMB para funcionar en entornos más amplios (incluyendo internet y conexiones a través de firewalls). **SAMBA** es una implementación de código abierto que permite a los sistemas Unix comunicarse con sistemas Windows utilizando el protocolo SMB/CIFS. Estas tecnologías son fundamentales para la colaboración y el intercambio de archivos en entornos de red mixtos.

Normalmente el puerto utilizado por SMB es el **445** aunque versiones antiguas (no se aconseja su utilización) pueden utilizar los puertos TCP 137, 138 y 139 para diferentes aspectos de la comunicación.

Si bien es posible configurar un servidor SAMBA **se pudo montar un directorio compartido en Windows mediante SMB, en el árbol de directorios de Linux.**



En equipo Windows

[Crear CARPETA COMPARTIDA Windows 10 y Windows 11](#) - YouTube

En equipo Linux:

```
dnf install cifs-utils
```

```
mount -t cifs //IP_Windows/Nombre_Carpeta /ruta/enLinux  
-o username=Usuario_en_Windows
```

De esta manera el contenido del directorio remoto (ubicado en Windows) será accesible desde un directorio local.

Firewall → (firewalld/iptables)

Establece un control sobre el tráfico de datos que entra y sale de una red, permitiendo o bloqueando el acceso según las reglas de seguridad establecidas.



Existen varias posibilidades para gestionar el tráfico en sistemas Linux:

ufw, firewalld, shorewall, iptables, etc ...

Con **firewalld** se pueden listar los servicios que están permitidos, además de bloquear/permitir algún tipo de comunicación utilizando una sintaxis sencilla.

```
[root@localhost ~]# firewall-cmd --list-services  
cockpit dhcpc6-client ssh  
[root@localhost ~]# firewall-cmd --permanent --add-service=http  
success
```

Las reglas de **iptables** se organizan en "cadenas" que definen cómo se manejará el tráfico de red. Permite configurar y controlar las reglas de filtrado de paquetes en el kernel de Linux para gestionar el tráfico de red entrante y saliente.

INPUT (tráfico entrante), **OUTPUT** (tráfico saliente) y **FORWARD** (tráfico enrutado a través del sistema). Las reglas pueden especificar diversos criterios, como direcciones IP, puertos, protocolos y estados de conexión.

*Estas chains pertenecen a la tabla filter (tabla por defecto y donde se aplican las reglas en caso de no especificarlo).

Algunos modificadores útiles:

- A (Append): Agrega una nueva regla al final de una cadena existente.
- I (Insert): Agrega la regla al principio de la cadena (se puede indicar posición exacta).
- D (Delete): Se utiliza para eliminar una regla específica de una cadena.
- P (Policy): Cambia la política por defecto de una cadena. Puedes usarlo para especificar qué acción debe tomarse si no coincide ninguna regla en una cadena.
- s (Source): Especifica la dirección IP de origen para la cual se aplicará la regla.
- d (Destination): Especifica la dirección IP de destino para la cual se aplicará la regla.
- p (Protocol): Indica el protocolo de red que se debe filtrar, como TCP, UDP, ICMP, etc.
- dport (Destination Port): Establece el número de puerto de destino en la regla.
- j (Jump): Especifica la acción que se debe realizar si un paquete coincide con la regla.
ACCEPT (aceptar el paquete), DROP (descartar el paquete), REJECT (rechazar el paquete y enviar una respuesta), etc.
- i (Input Interface): Define a través de qué interfaz llega el paquete.

Recordar que las reglas se evalúan en orden, por lo que es importante la posición que ocupan en la cadena.

Ejemplos

```
iptables -A INPUT -s 192.168.3.0 -p tcp --dport 22 -j DROP
```

Bloquear el tráfico entrante que venga de una IP particular y con destino el puerto 22.

```
iptables -I INPUT 1 -i enp0s3 -p icmp --icmp-type echo-request -j DROP
```

Bloquea las peticiones echo del protocolo icmp (PING) que ingresan por una tarjeta de red en particular. (La regla se agrega al principio de la cadena).

persistencia

iptables-save > archivo
iptables-restore < archivo

```
iptables -L -v //
```

lista las reglas activas

DNS



Sistema que se utiliza para asociar direcciones IP con nombres de dominio legibles para los humanos.

Los **servidores DNS recursivos** (normalmente brindados por el ISP aunque existen públicos). No almacenan información de nombres de dominio, pero realizan búsquedas en la jerarquía de DNS y actúan como intermediarios.

tipos de servidores

Los **servidores DNS raíz** son el primer punto de contacto en la cadena de consultas DNS. Estos contienen información sobre la ubicación de los **servidores DNS de nivel superior** (como los dominios de nivel superior ".com", ".org", ".net", etc.). Aunque a menudo se habla de 13 servidores DNS raíz, en realidad hay múltiples instancias de estos servidores distribuidos en todo el mundo para garantizar la redundancia y la resistencia ante fallos.

Los **servidores DNS autoritativos** son los que almacenan la información oficial y actualizada sobre los nombres de dominio y sus correspondientes direcciones IP.



WWW: Es un subdominio opcional y generalmente se usa para indicar una dirección web o un servidor específico.

profesantiago: Es el nombre principal del dominio.

COM: Es el dominio de nivel superior (TLD), que indica el tipo de organización o el propósito del sitio web (.com para comerciales, .org para organizaciones sin fines de lucro, etc.).

El sistema DNS organiza los nombres de dominio en una jerarquía de servidores, comenzando desde los servidores raíz en la cima, pasando por los servidores de dominio de nivel superior (TLD), y finalmente llegando a los servidores autoritativos que almacenan la información específica del dominio.

Servidor DNS interno con BIND -> puerto 53

De esta manera podemos utilizar nombres de dominio para referirnos a los equipos de nuestra red interna y no utilizar direcciones IP. Puede configurarse además la recursividad para resolver dominios de internet.

```
dnf install bind  
systemctl start named
```

En el archivo de configuración puedo modificar puerto, red o interfaz por la que responde o activar/desactivar recursividad.

Debo definir aquí las nuevas **zonas DNS** que va a gestionar el servidor.

/etc/named.conf

```
# //////////////////////////////////////////////////////////////////// DEFINO ZONAS ////////////////////////////////////////////////////////////////////  
  
zone "midominio.lan" IN {  
    type master;  
    file "/var/named/db.midominio.lan";  
};  
  
zone "0.16.172.in-addr.arpa" IN {  
    type master;  
    file "/var/named/rev.midominio.lan";  
};
```

```
$TTL 86400  
@ IN SOA servidor.midominio.lan. root.midominio.lan. ( 0 ; Serial  
                                         3600 ; Refresh  
                                         1H ; Retry  
                                         1W ; expire  
                                         3H ; minimum  
)  
  
servidor IN NS servidor.midominio.lan.  
ubuntu   IN A   172.16.0.2  
win7     IN A   172.16.0.5  
@        IN MX 10 ubuntu  
www      IN CNAME servidor
```

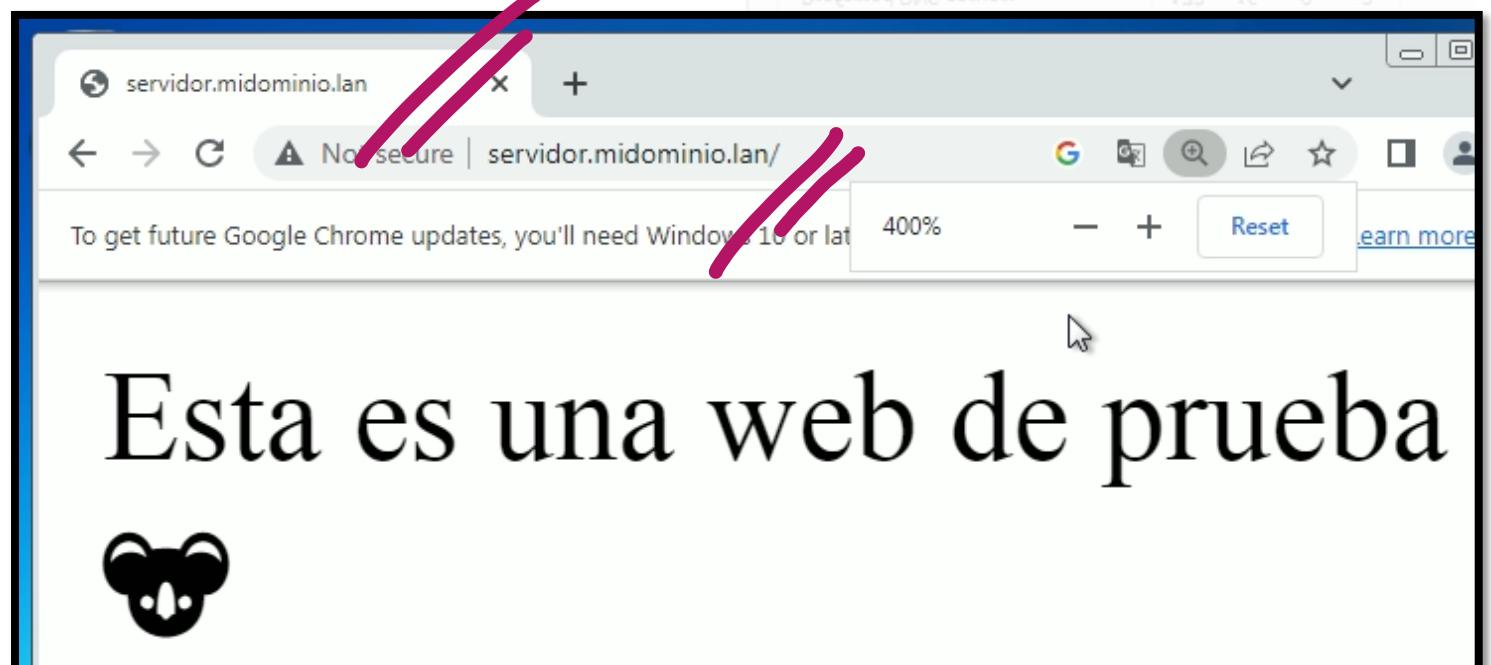
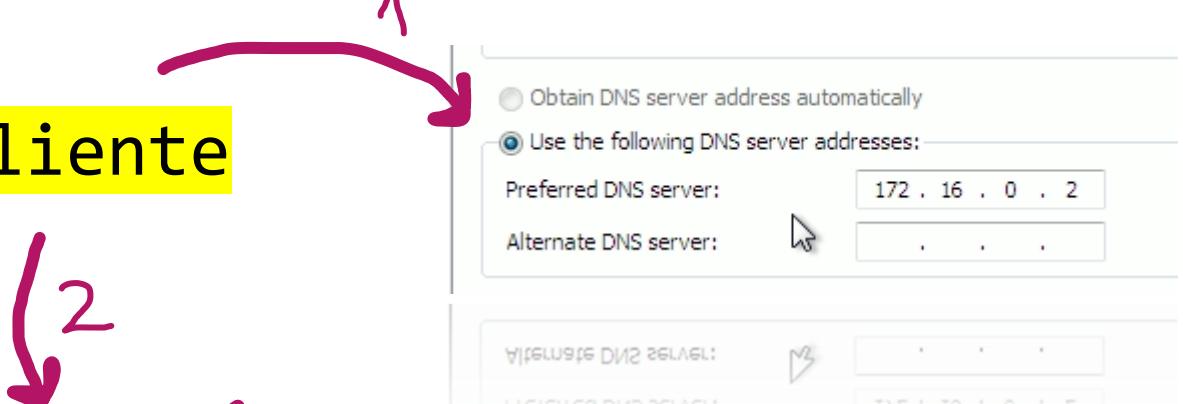
Los archivos de zona contienen los **registros DNS**. Aquí se busca la información correspondiente al dominio. Existen registros tipo **A** (asocian dirección ipv4 a nombre de dominio), **AAAA** (asocia una dirección ipv6 a un nombre), **NS** (indica el nombre del servidor), **CNAME** (permite definir un alias), **MX** (apunta a un servidor de correo), entre otros tipos de registro.

Comandos como **nslookup** o **dig** se utilizan para realizar resoluciones de nombres y pueden ser útiles para comprobar el funcionamiento del servidor DNS.

```
[root@fedoraserver named]# nslookup ubuntu.midominio.lan
Server:      172.16.0.2
Address:     172.16.0.2#53

Name:  ubuntu.midominio.lan
Address: 172.16.0.5
```

En el cliente



APACHE -> Servidor web (80/443)

```
dnf install httpd  
firewall-cmd --add-service=http  
systemctl start httpd
```

```
# Change this to Listen on a specific IP address  
# httpd.service is enabled to run at boot time  
# available when the service starts. See the  
# page for more information.  
#  
#Listen 12.34.56.78:80  
Listen 80
```

directorio raíz del servidor

```
<Directory "/var/www/html">  
    #  
    # Possible values for the Options  
    # or any combination of:  
    #   Indexes Includes FollowSymLinks  
    #  
    # Note that "MultiViews" must be  
    # doesn't give it to you.  
    #  
    # The Options directive is both c  
    # http://httpd.apache.org/docs/2.  
    # for more information.  
    #  
    # Options Indexes FollowSymLinks  
    Options -Indexes
```

En /etc/httpd/conf/httpd.conf

Se establece la configuración básica.

Es interesante eliminar la navegación por el directorio raíz de apache y **eliminar la firma del servidor** cuando se produce un error.

ServerSignature Off

Not Found

The requested URL was not found on this server.

Apache/2.4.57 (Fedora Linux) OpenSSL/3.0.9 Server at 192.168.1.13 Port 80

Por defecto apache responderá las solicitudes hacia el puerto 80, mostrando el código html que se encuentra en el directorio raíz, para que el navegador pueda interpretarlo como un sitio web.

servidor

```
[root@fedoraserver html]# pwd  
/var/www/html  
[root@fedoraserver html]# ls  
index.html  
[root@fedoraserver html]# cat index.html  
<html>  
  <meta charset='UTF-8'>  
  <body>  
    Esta es una web de prueba &#128040;  
  </body>  
</html>  
[root@fedoraserver html]#
```

Esta es una web de prueba



cliente

HTTPS

Proporciona una capa adicional de seguridad al **cifrar la información transmitida entre el navegador web del usuario y el servidor web que aloja el sitio**. Esto ayuda a proteger los datos confidenciales, como contraseñas, información de tarjetas de crédito y otros datos sensibles, contra posibles ataques o interceptaciones maliciosas.

El protocolo HTTPS utiliza certificados TLS.

Un **certificado TLS** es un archivo electrónico que sirve para autenticar la identidad del sitio web y cifrar la comunicación entre el servidor y el navegador del usuario. Estos certificados son emitidos por Autoridades de Certificación (CA) confiables.

Se puede generar y [autofirmar un certificado TLS de manera gratuita](#)

Si bien el protocolo https puede utilizarse con un certificado autofirmado, aparecerá una advertencia en el navegador. Esto puede ser sospechoso o generar una mala imagen para los usuarios que accedan a tu sitio desde internet.



The screenshot shows a browser window with the following details:

- Address Bar:** ▲ No es seguro | <https://192.168.1.13>
- Title Bar:** Visor de certificados: fedoraserver
- General Tab:** Selected tab.
- Enviado a:**

Nombre común (CN)	fedoraserver
Organización (O)	profesantiago
Unidad organizativa (OU)	<No incluido en el certificado>
- Emitido por:**

Nombre común (CN)	fedoraserver
Organización (O)	profesantiago
Unidad organizativa (OU)	<No incluido en el certificado>
- Details Tab:** Unavailable or not visible in the screenshot.

SQUID -> Proxy web (:3128)

Un **proxy** actúa como intermediario entre la comunicación cliente/servidor. En este caso se utiliza como **filtro de contenido** para restringir la navegación web (*puede actuar como caché, brindar anonimato y balanceo de carga, entre otras funciones*).

Luego de instalar squid (**dnf install squid**) se pueden definir listas de acceso (**acl**) y tomar decisiones sobre la navegación web (**http_access**). 

Se pueden crear listas basadas en diferentes elementos y luego definir como se relacionan en /etc/squid/squid.conf

algunos tipos de ACL

src: direcciones IP de origen.

dstdomain: dominios de destino.

url_regex: palabras claves dentro de la url.

time: horarios o días.

Los dominios destino o palabras clave pueden colocarse en archivos de texto y referenciarlos en el squid.conf

```
acl localnet src 0.0.0.1-0.255.255.255    # RFC 1122 "this" network
acl localnet src 10.0.0.0/8                   # RFC 1918 local private n
acl localnet src 100.64.0.0/10                 # RFC 6598 shared address
acl localnet src 169.254.0.0/16                # RFC 3927 link-local (dir
acl localnet src 172.16.0.0/12                # RFC 1918 local private n
acl localnet src 192.168.0.0/16               # RFC 1918 local private n
acl localnet src fc00::/7                      # RFC 4193 local private n
acl localnet src fe80::/10                     # RFC 4291 link-local (dir
```

Defino redes/equipos como ACL de origen

```
acl win11 src 192.168.1.11                  # Mi equipo con Windows 11
acl win7 src 192.168.1.12                     # PC virtual con Windows 7
```

```
#####
acl SSL_ports port 443
acl Safe_ports port 80                         # http
acl Safe_ports port 21                         # ftp
```

```
acl sitiosbloqueados dstdomain "/etc/squid/block"
http_access deny win11 sitiosbloqueados
```

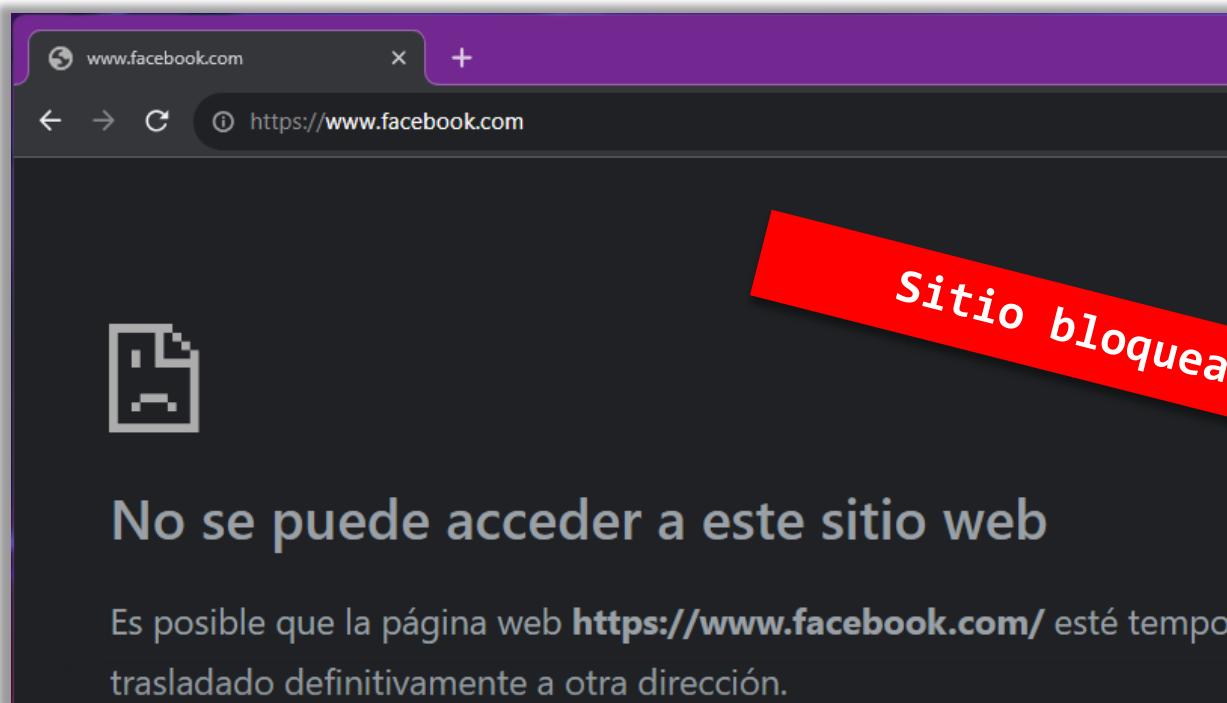
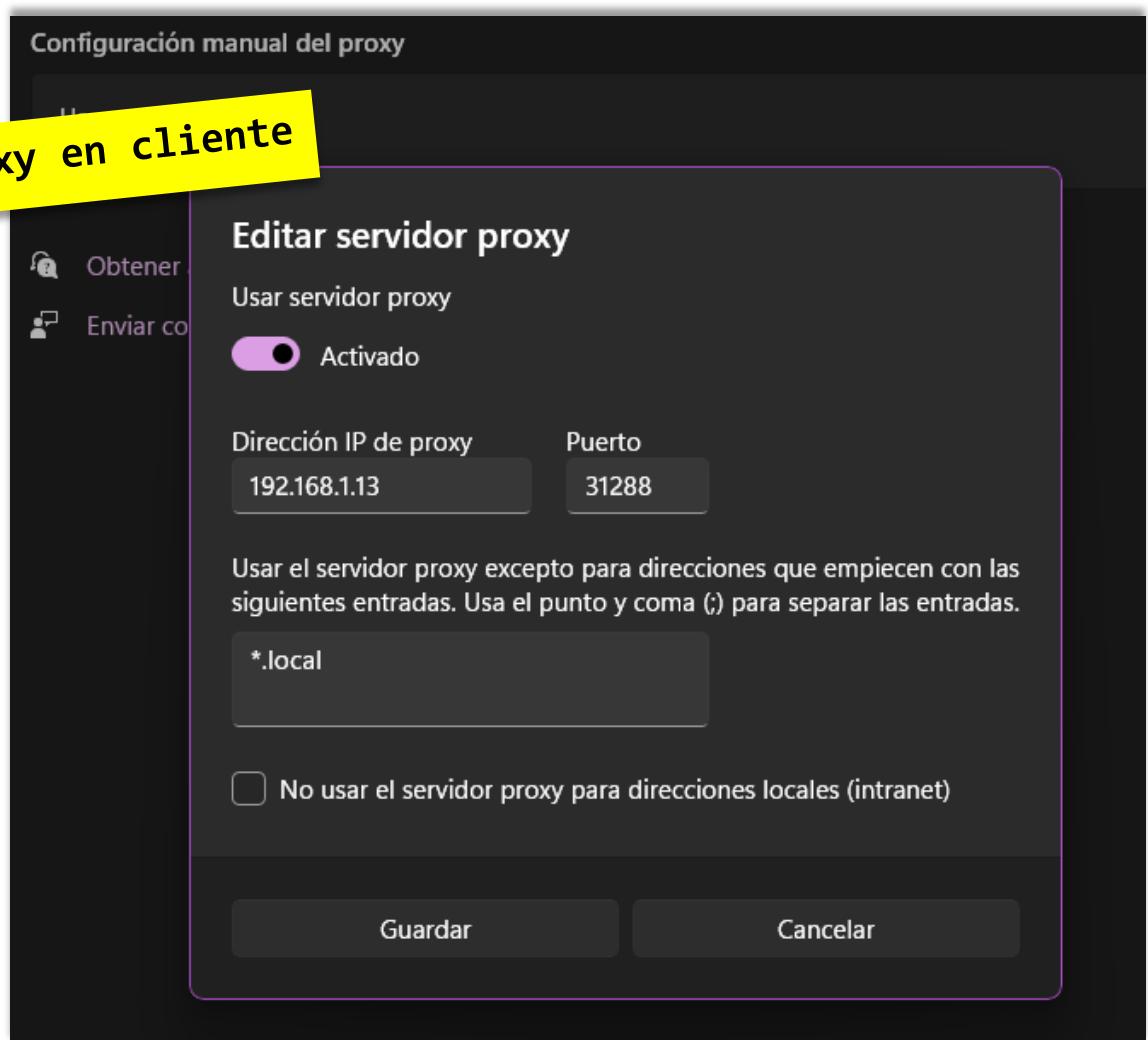
```
acl urlpatron url_regex juegos musica pelis
http_access allow win11 !urlpatron
```

```
# For example, to allow access from your local networks, you may
# following rule (and/or add rules that match your definition of
http_access deny all
```

Permitir la comunicación en el firewall

```
[root@fedoraserver ~]# firewall-cmd --add-service=squid  
success  
[root@fedoraserver ~]# _
```

Configurar proxy en cliente

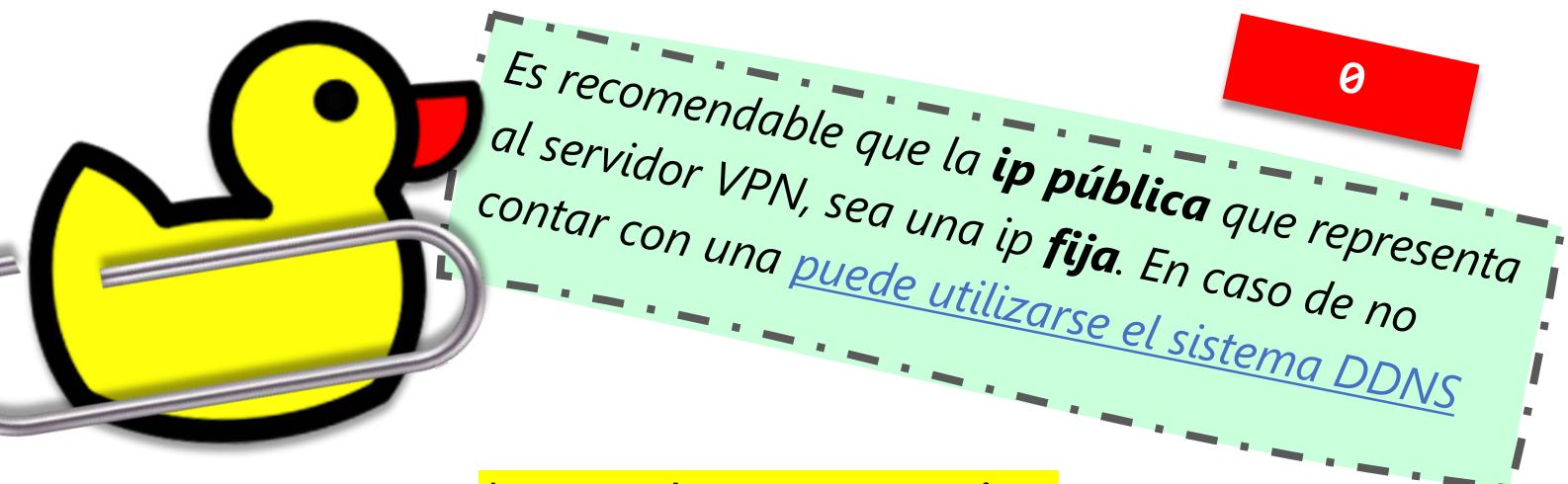


VPN (OPENVPN server - paso a paso)

Una VPN (Red Privada Virtual) es una tecnología que permite establecer una **conexión segura y cifrada** entre cliente y servidor.

Esta conexión segura se establece a través de una red pública (**internet**) y crea un **túnel** encriptado a través del cual fluye el tráfico de datos.

Las VPN permiten a los usuarios **acceder** a recursos en una red local (como en una empresa o institución) **desde ubicaciones remotas como si estuvieran físicamente presentes en esa red**. Esto es útil para el trabajo remoto o el acceso a archivos y recursos internos de forma segura.



*Comandos necesarios

1

```
dnf install openvpn easy-rsa
```

Copio el directorio por defecto para que certificados y archivos no se reemplacen (por seguridad).

```
cp -r /usr/share/easy-rsa /etc/openvpn/  
cd /etc/openvpn/easy-rsa/3.1.5/
```

**algunos comandos o rutas pueden variar dependiendo de la versión del servicio o la distribución gnu-linux.*

```
[root@Fedora 3.1.5]# ls
easyrsa  openssl-easyrsa.cnf  pki  x509-types
[root@Fedora 3.1.5]# _
```

Dentro del directorio easy-rsa se incluye un **script**, el cual utilizaremos para crear una infraestructura de clave pública (**PKI**) y la autoridad certificadora (**CA**) para firmar digitalmente y autenticar las claves públicas.

./easyrsa init-pki

./easyrsa build-ca ##### Creo una entidad certificadora (puede estar en otro equipo).

Se crea directorio PKI

pki/private (allí se encuentra clave privada de la CA -> **ca.crt**).

Creo par de claves para servidor

./easyrsa gen-req servidor-fedora nopass

se crea la privada y el request para la pública (Hay que firmarla).

./easyrsa sign-req server servidor-fedora

#Se firma la clave.

Copio todo a un mismo directorio.

/etc/openvpn/server

ca.crt

servidor-fedora.crt

servidor-fedora.key

CLAVE TLS_CRYPT dentro del mismo directorio.

openvpn --genkey secret ta.key

3

Dentro de `easy-rsa` se encuentra el script que utilizamos y dentro de `server` deberían estar los siguientes archivos:

`ca.crt`
`servidor-fedora.crt`
`servidor-fedora.key`
`ta.key`

```
[root@Fedora openvpn]# pwd
/etc/openvpn
[root@Fedora openvpn]# ls
client easy-rsa server
```

CREO DIRECTORIO (si no existe) para almacenar claves del cliente

```
mkdir /etc/openvpn/client/keys
```

```
chmod -R 700 /etc/openvpn/client
```

```
cd /etc/openvpn/easy-rsa/3.1.5
```

```
./easyrsa gen-req cliente1 nopass
```

```
./easyrsa sign-req client cliente1
```

Copio todo al directorio `/etc/openvpn/client/keys`

`ca.crt`

`cliente1.crt`

`cliente1.key`

`ta.key`

Configuración de cliente y servidor

```
[root@Fedora private]# cd /usr/share/doc/openvpn/sample/sample-config-files/
[root@Fedora sample-config-files]# ls
client.conf   home.up          loopback-server  openvpn-shutdown.sh  README
firewall.sh   loopback-client  off ice.up       openvpn-startup.sh  roadwarrior-client.conf
[root@Fedora sample-config-files]# roadwarrior-server.conf
```

Existen plantillas de los archivos `client.conf` y `server.conf`

```
##copiar server.conf a /etc/openvpn/server
```

```
##copiar client.conf a /etc/openvpn/client
```

/etc/openvpn/server/server.conf

```
# OpenVPN also supports
# single-machine <-> single-machine
# configurations (See the Examples page
# on the web site for more info).
#
# This config should work on Windows
# or Linux/BSD systems. Remember on
# Windows to quote pathnames and use
# double backslashes, e.g.:
# "C:\\Program Files\\OpenVPN\\config\\foo.key"
#
# Comments are preceded with '#' or ';'
#####
#
# Which local IP address should OpenVPN
# listen on? (optional)
:local a.b.c.d
#
# Which TCP/UDP port should OpenVPN listen on?
# If you want to run multiple OpenVPN instances
# on the same machine, use a different port
# number for each one. You will need to
# open up this port on your firewall.
port 1194
#
# TCP or UDP server?
:proto tcp
proto udp
#
# "dev tun" will create a routed IP tunnel,
# "dev tap" will create an ethernet tunnel,
```

Config utilizada

```
#nombres correctos de .key y
.crt
dh none
push "redirect-gateway def1
bypass-dhcp" ##### Clientes
puedan salir a internet a traves
de la VPN
tls-auth # comentar y agregar
tls-crypt ta.key
cipher AES-256-GCM
auth SHA512
user nobody
group nobody
# el resto de parametros
normalmente pueden quedar con
los valores por defecto.
```

/etc/openvpn/client/client.conf

##debe coincidir con el
server.conf

nombre/ip del servidor +
puerto

Comento .key y .cert
ya que voy a utilizar un
archivo .ovpn (más
sencillo de gestionar).

```
# Are we connecting to a TCP or
# UDP server? Use the same setting as
# on the server.
:proto tcp
proto udp
#
# The hostname/IP and port of the server.
# You can have multiple remote entries
# to load balance between the servers.
remote santiagointernet.duckdns.org 1194
:remote my-server-2 1194
```

```
# file can be used for all clients.
:ca ca.crt
:cert client.crt
:key client.key
```

5

Gestionar firewall y activar ip_forward

```
systemctl start openvpn-server@server
```

```
sysctl -w net.ipv4.ip_forward=1
echo 1 > /proc/sys/net/ipv4/ip_forward
```

El servidor debe permitir el tráfico entre la interfaz virtual (utilizada por el túnel vpn) y la red interna.

```
firewall-cmd --add-service=openvpn
#IPTABLES
iptables -t nat -I POSTROUTING 1 -s 10.8.0.0/24 -o enp0s3 -j MASQUERADE
iptables -I INPUT 1 -i tun0 -j ACCEPT
iptables -I FORWARD 1 -i enp0s3 -o tun0 -j ACCEPT
iptables -I FORWARD 1 -i tun0 -o enp0s3 -j ACCEPT
iptables -I INPUT 1 -i enp0s3 -p udp --dport 1194 -j ACCEPT
```

```
root@Fedora ~]# iptables -L -v
Chain INPUT (policy ACCEPT 2 packets, 432 bytes)
pkts bytes target     prot opt in     out     source               destination
    0    0 ACCEPT      udp   --  enp0s3  any    anywhere            anywhere
    0    0 ACCEPT      all   --  tun0   any    anywhere            anywhere
                                         destination      udp dpt:openvpn

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target     prot opt in     out     source               destination
    0    0 ACCEPT      all   --  tun0   enp0s3 anywhere            anywhere
    0    0 ACCEPT      all   --  enp0s3 tun0   anywhere            anywhere

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

*Además de esto, el router debe direccionar las peticiones externas que llegan al puerto 1194, hacia el servidor VPN dentro de la lan.

Crear archivo .ovpn

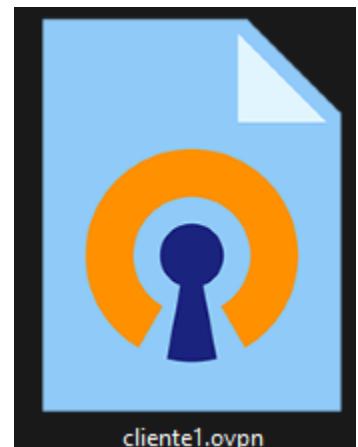
6

Un fichero .ovpn incluye todo lo necesario para el cliente pueda conectarse (*archivo .conf, ca.crt, cliente1.crt, cliente1.key y ta.key*)

```
# Set log file verbosity.
verb 3

# Silence repeating messages
;mute 20
<ca>
-----BEGIN CERTIFICATE-----
MIIDSzCCAjOgAwIBAgIUTZ1dn2F1LKoDi
BQAwFjEUMBIGA1UEAwLc2FudGlhZ28tQ
NzI5MTYxOTQ3WjAWMRQwEgYDVQQDDAtzY
AQEBBQADggEPADCCAQoCggEBAMLoDhpZ
dF8wFmmWM6qAHec87wouhCFvIEpE4PLoo
Ht3noblxYkTaTUl+pA4vmR70JqQC+4OXs
/KcQdyKqztFQARPO6quyajJsWor7hXwuD
gaSucM7wilJokHMKWDNnOqcMILcPqnNgd
BsZeUxBL+gdZ0sF14azBHfSDNb7lPV4ii
BgNVHRMEBTADAQH/MB0GA1UdDgQWBBSR0
HSMESjBIgBSR0XOFK2xiTakdtgHGCeut
dGlhZ28tQ0GCFE2dXZ9hYiyqa4vF+GAOF
hkiG9w0BAQsFAAOCAQEAYtqwoND6aDr38
2UktjHFv91cOHh0A6JpSdm3RrC//qLrq+
V7Pq2wJd01MBZJm5Wpa5y62XDLImkmNNT
dAFyLr/XEjZ5JuA/xGrLv0mfvh/rUJ/x
BYP/RdyALhTDuYS3Bmx5MHumiXvcmrICk
16c0rh6s6MkWsNL0Dh6Bb3J/BU4SCJIgb
-----END CERTIFICATE-----
</ca>
<cert>
Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number:
      b1:c7:d0:89:c5:59:1c:
    Signature Algorithm: sha256WithRSAEncryption
    Subject: CN=santiago-CA
    Validity
      Not Before: Aug 1 17
      Not After : Nov 3 17
    Subject: CN=cliente1
    Subject Public Key Info:
      Public Key Algorithm:
        Public-Key: (2048

```



contenido cliente.conf

<ca>

contenido ca.crt

</ca>

<cert>

contenido cliente1.crt

</cert>

<key>

contenido cliente.key

</key>

<tls-crypt>

contenido ta.key

</tls-crypt>

Puede crearse de forma manual o automatizarlo mediante un script.

script para crear archivo .ovpn

```
#!/bin/bash  
  
# First argument: Client identifier
```

Este script debe ejecutarse con el nombre de los certificados del cliente como parámetro

```
KEY_DIR=~/client-configs/keys  
OUTPUT_DIR=~/client-configs/files  
BASE_CONFIG=~/client-configs/base.conf  
  
cat ${BASE_CONFIG} \  
  <(echo -e '<ca>') \  
  ${KEY_DIR}/ca.crt \  
  <(echo -e '</ca>\n<cert>') \  
  ${KEY_DIR}/${1}.crt \  
  <(echo -e '</cert>\n<key>') \  
  ${KEY_DIR}/${1}.key \  
  <(echo -e '</key>\n<tls-crypt>') \  
  ${KEY_DIR}/ta.key \  
  <(echo -e '</tls-crypt>') \  
  > ${OUTPUT_DIR}/${1}.ovpn
```

Modificar las rutas con la ubicación de los archivos y opcionalmente el directorio de salida.

Recordar que, si el script se crea desde Windows, al ejecutarlo en Linux puede haber problemas con la implementación de los saltos de línea.

Un cambio de formato con el comando **dos2unix** puede ser necesario.

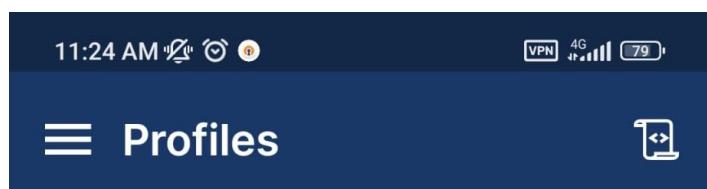
Conexión desde el cliente

Si el servicio está activo y todo correctamente configurado, solo resta colocar el **.ovpn** en el cliente y descargar la **aplicación de cliente de Open VPN**.

Solo deberíamos indicar la ubicación de archivo .ovpn y la conexión debería realizarse.

A partir de este momento el cliente puede trabajar en los equipos de la red local donde se encuentra el servidor, sin estar físicamente en ella y a través de internet

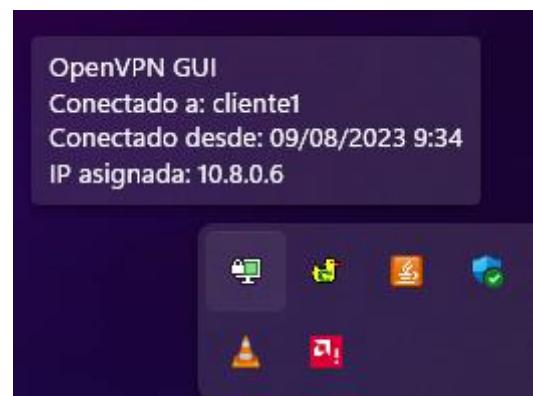
cliente Android



CONNECTED

OpenVPN Profile
santiagointernet.duckdns.org
[cliente1]

cliente Windows



desde dispositivo móvil

Estoy comunicándome con una ip privada (equipo dentro de mi lan).



11:25 AM

Estoy físicamente fuera de mi lan. Utilizando datos móviles y conectado al servidor desde internet.



ESTE SITIO WEB SE ENCUENTRA EN MI CASA

AUTOMATIZACIÓN → (.bashrc .bash_profile y CRON)

- Existe un archivo .bashrc y un .bash_profile por cada usuario.
- Se encuentran ocultos en su directorio personal.

.bash_profile: Se ejecuta cuando el usuario inicia sesión. Se pueden especificar las configuraciones y **variables de entorno** que estarán disponibles o la **ejecución de un comando o script al inicio de la sesión.**

```
----- BIENVENIDO root -----  
Hoy es 10-08  
  
El firewall se restaurado...  
[root@fedoraserver ~]#
```

```
# .bash_profile  
# Get the aliases and functions  
if [ -f ~/.bashrc ]; then  
    . ~/.bashrc  
fi  
  
# User specific environment and startup programs  
./inicio.sh
```

.bashrc: Este archivo se ejecuta cada vez que se inicia una nueva instancia de Bash, como al abrir una terminal o una ventana de línea de comandos.

Se puede utilizar para **setear alias de comandos**, entre otros usos.

```
# User specific aliases and functions  
  
alias rm='rm -i'  
alias cp='cp -i'  
alias mv='mv -i'  
alias vertodo='ls -lart'  
alias chau='shutdown -h now'
```

Cron (tareas programadas)

El demonio Cron permite a los usuarios **programar comandos o scripts para que se ejecuten automáticamente en momentos específicos o en intervalos regulares**, sin necesidad de intervención manual.

```
[root@fedoraserver ~]# systemctl status crond.service
● crond.service - Command Scheduler
  Loaded: loaded (/usr/lib/systemd/system/crond.service; enabled; preset: enabled)
  Drop-In: /usr/lib/systemd/system/service.d
            └─10-timeout-abort.conf
    Active: active (running) since Thu 2023-08-10 12:02:54 -03: 1h 21min ago
      Main PID: 868 (crond)
        Tasks: 2 (limit: 2252)
```

El usuario **root** puede programar tareas editando el archivo **/etc/crontab** luego cada usuario tiene su crontab particular al que accede mediante **crontab -e**.

```
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root

# For details see man 4 crontabs

# Example of job definition:
# ----- minute (0 - 59)
# | ----- hour (0 - 23)
# | | ----- day of month (1 - 31)
# | | | ----- month (1 - 12) OR jan,feb,mar,apr ...
# | | | | ----- day of week (0 - 6) (Sunday=0 or ?) OR sun,mon,tue,wed,thu,fri,sat
# | | | | |
# * * * * * user-name command to be executed
0 15 * * 1 root /root/respaldo.sh
```

 El script `respaldo.sh` se ejecutará automáticamente a las 15 horas, 0 minutos, todos los días del mes (*), todos los meses (*), los días Lunes(1).

Este archivo cuenta con 5 campos que permiten definir: **minutos, horas, día del mes, mes y día de la semana**.

Cada campo espera un número o caracteres válidos (un asterisco representa todos los posibles valores de ese campo).

Finalmente se puede seleccionar el usuario que ejecutará la tarea (en caso de ser root) y el comando o script a ejecutar.

Ese servicio es especialmente útil para automatizar **respaldos o tareas periódicas**.