

ADMINISTRACIÓN DE SISTEMAS OPERATIVOS Y REDES DE COMPUTADORES

FRANCISCO JAVIER PÉREZ MARTÍNEZ
74384305M – GRUPO 01

Contenido

1. Licencias	2
2. Particionado	3
3. Arranque y parada de servicios.....	5
4. CentOS.....	5
4.1 SSH, SFTP, SCP	5
4.2 VNC + RDP	7
4.3 BD Server	9
4.4 Web Server	11
4.5 NFS / SAMBA	12
4.6 DHCP.....	15
4.7 CUPS	16
4.8 FreeNAS + iSCSI	17
4.9 Git + OwnCloud	18
4.10 DNS.....	21
5. FreeBSD	23
5.1 SSH, SFTP, SCP	23
5.2 DHCP.....	25
5.3 VNC + RDP	26
5.4 CUPS	27
5.5 BD Server	28
5.6 Git + ownCloud	29
5.7 FreeNAS + iSCSI	30
6. Windows Server 2019	31
6.1 SSH, SFTP, SCP	31
6.2 DHCP.....	32
6.3 VNC + RDP	33
6.4 NFS / SAMBA	34
6.5 FreeNAS + iSCSI	36
6.6 DNS.....	37

1. Licencias

Una **licencia de software** es un contrato entre el licenciante (autor/titular de los derechos de explotación/distribución) y el licenciario (usuario consumidor, profesional o empresa) del programa informático, para utilizarlo cumpliendo una serie de términos y condiciones establecidas dentro de sus cláusulas, es decir, es un conjunto de permisos que un desarrollador le puede otorgar a un usuario en los que tiene la posibilidad de distribuir, usar o modificar el producto bajo una licencia determinada. Además, se suelen definir los plazos de duración, el territorio donde se aplica la licencia (ya que la licencia se soporta en las leyes particulares de cada país o región), entre otros.

Una vez definido qué es una licencia de software pasará a realizar una pequeña comparación entre EULA, BSD y GPL GNU.

EULA:

- Se prohíbe la copia.
- Puede ser empleado en un único ordenador con un máximo de 2 procesadores.
- No puede ser empleado como webserver o fileserv.
- Registro necesario a los 30 días.
- Puede dejar de funcionar si se efectúan cambios en el hardware.
- Las actualizaciones del sistema pueden modificar la licencia, si la compañía lo desea.
- Solo puede ser transferida una vez a otro usuario.
- Impone limitación sobre la ingeniería inversa.
- Da a Microsoft derecho para en cualquier momento recoger información del sistema y su uso, y también para entregar dicha información a terceros.
- La garantía es por los primeros 90 días.
- Actualizaciones y parches sin garantía.

Dentro de las licencias de software libre nos encontramos la licencia **GPL (Gnu Public License)** con todas sus variantes y la **BSD (Berkeley Software Distribution)**. La GPL no nos permite cerrar el sistema o la aplicación que use cualquiera de las variantes de la GPL y siempre debemos distribuir el código fuente. Esta ley es libre en parte y por lo tanto es hipócrita ya que va gritando a los 4 vientos libertad y luego no nos deja cerrar el sistema.

Por otro lado, la BSD nos permite ver el código y modificarlo, pero también nos permite cerrar el sistema o la aplicación. Lo normal es que esto lo lea alguien que defienda el código abierto y no lo privado pero la libertad también incluye el poder cerrar nuestro sistema. Un ejemplo de ello es el sistema de la manzana, este sistema es un BSD, el kernel Darwin es una mezcla de Mach1 con algo de BSD y es libre, aunque las demás partes del sistema sean de código cerrado.

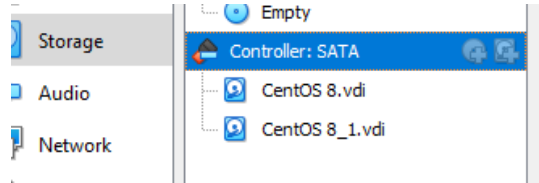
OJO, con esto no digo que la GPL sea mala licencia, considero que es una buena licencia ya que nos da más libertad que las privadas, pero la BSD es el verdadero software libre ya que nos lo permite todo.

A lo largo de su historia esta licencia ha tenido varios cambios: BSD de 4 cláusulas y BSD de 3 cláusulas.

Luego existe una variante, la llamada BSD de 2 cláusulas o simplificada que es la usada por FreeBSD.

2. Particionado

Para crear particiones en cada uno de los sistemas operativos introducimos un nuevo disco duro mediante la configuración de virtual box.



Para crear particiones usamos la herramienta fdisk. Con el comando `$ fdisk -l` comprobamos los discos del sistema disponibles y las particiones que haya creadas.

Pasamos a crear una partición, para ello escribimos el comando `$ fdisk /dev/sdb` y realizamos lo mostrado a continuación en la terminal:

```
Orden (m para obtener ayuda): p
Disco /dev/sdb: 8 GiB, 8589934592 bytes, 16777216 sectores
Unidades: sectores de 1 * 512 = 512 bytes
Tamaño de sector (lógico/físico): 512 bytes / 512 bytes
Tamaño de E/S (mínimo/óptimo): 512 bytes / 512 bytes
Tipo de etiqueta de disco: dos
Identificador del disco: 0xa5bc6e80

Orden (m para obtener ayuda): n
Tipo de partición
  p primaria (0 primaria(s), 0 extendida(s), 4 libre(s))
  e extendida (contenedor para particiones lógicas)
Seleccionar (valor predeterminado p): p
Número de partición (1-4, valor predeterminado 1): 1
Primer sector (2048-16777215, valor predeterminado 2048):
Último sector, +sectores o +tamaño{K,M,G,T,P} (2048-16777215, valor predeterminado 16777215):

Crea una nueva partición 1 de tipo 'Linux' y de tamaño 8 GiB.

Disposit.  Inicio Comienzo  Final Sectores Tamaño Id Tipo
/dev/sdb1          2048 16777215 16775168     8G 83 Linux

Orden (m para obtener ayuda): w
Se ha modificado la tabla de particiones.
Llamando a ioctl() para volver a leer la tabla de particiones.
Se están sincronizando los discos.

[root@localhost ~]# mkfs -t ext4 /dev/sdb1
mke2fs 1.45.4 (23-Sep-2019)
Se está creando un sistema de ficheros con 2096896 bloques de 4k y 524288 nodos-i
UUID del sistema de ficheros: 538a3e1e-d417-4b93-aa25-b6ed1d7d73a5
Respalos del superbloque guardados en los bloques:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Reservando las tablas de grupo: hecho
Escribiendo las tablas de nodos-i: hecho
Creando el fichero de transacciones (16384 bloques): hecho
Escribiendo superbloques y la información contable del sistema de ficheros: hecho

[root@localhost ~]# mkdir /data
[root@localhost ~]# mount /dev/sdb1 /data
[root@localhost ~]# mount

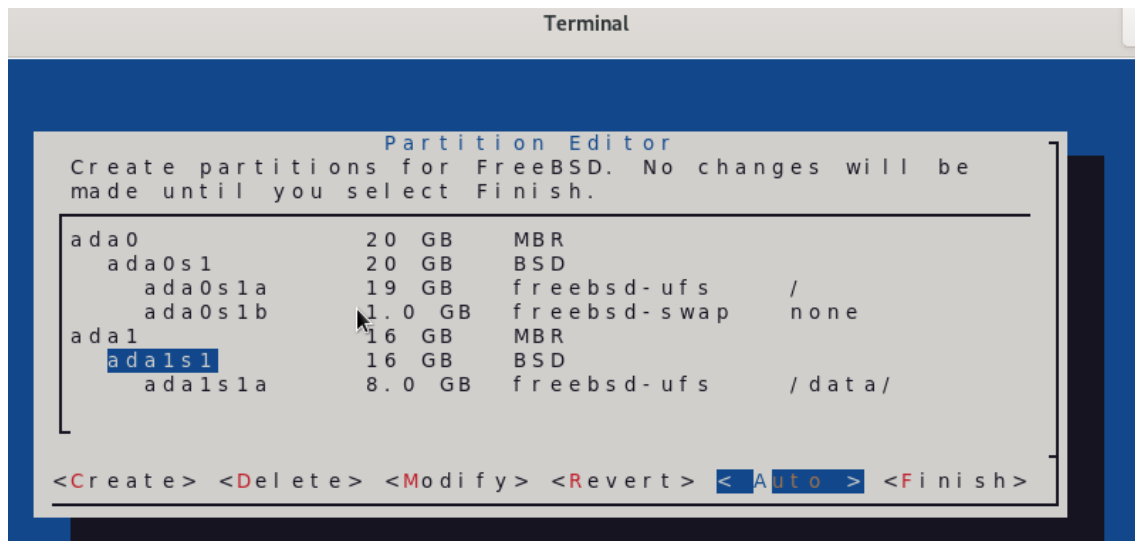
10=1000)
/dev/sdb1 on /data type ext4 (rw,relatime,seclabel)
```

Como podemos observar se ha montado nuestra partición en la carpeta /data

Para saber que significa cada comando escribir m para obtener ayuda.

Por último, guardar en archivo `/etc/fstab` la entrada que hace referencia al disco para que se monte durante el arranque de sistema.

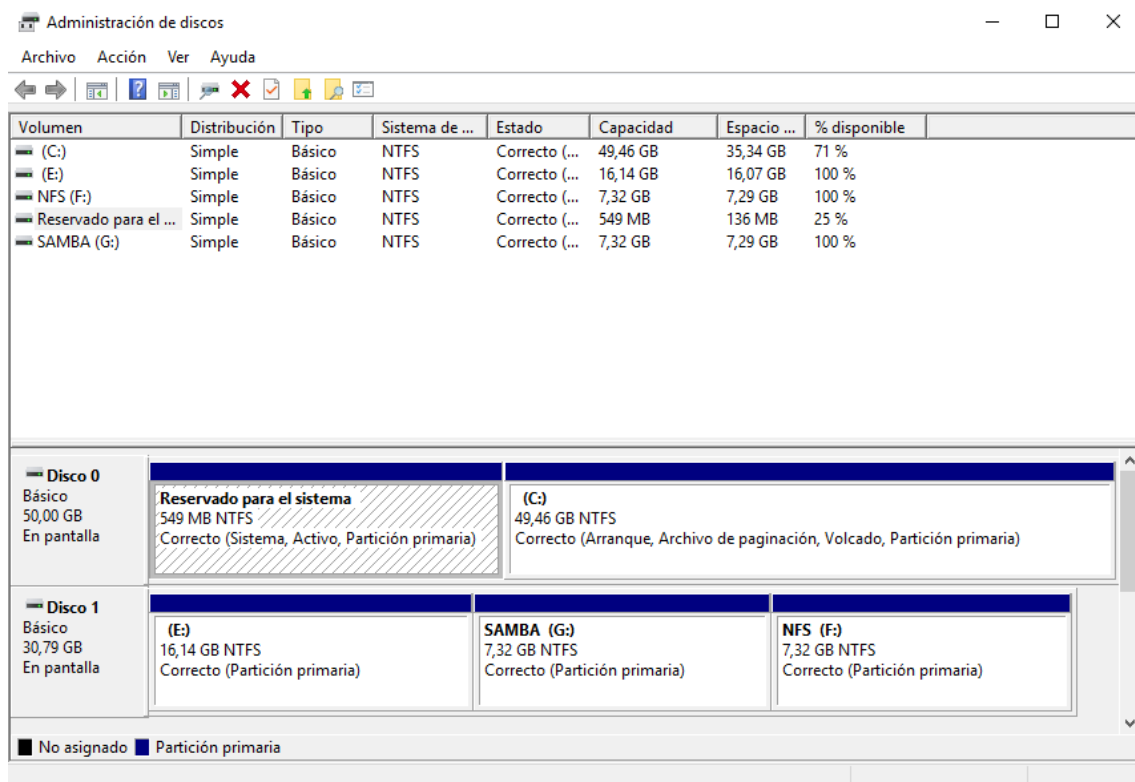
Para crear una partición en FreeBSD, utilizaremos la herramienta de sade:



```

root@freebsd:~ # df
Filesystem 1K-blocks    Used  Avail Capacity  Mounted on
/dev/ada0s1a 19278748 13014948 4721504    73%    /
devfs         1         1         0    100%    /dev
procfs        4         4         0    100%    /proc
/dev/ada1s1a  8106716   32836  7425344     0%    /data
root@freebsd:~ #
  
```

Para particionar en Windows Server, utilizaremos la herramienta de particionado del propio sistema: Administración de discos.



3. Arranque y parada de servicios

Al instalar servicios debemos ponerlos en marcha utilizando órdenes específicas de los diferentes sistemas.

En los sistemas basados en Linux o Unix la herramienta utilizada es “systemctl”. En CentOS utilizamos esta herramienta.

- **Start <servicio>** Inicia el servicio.
- **Stop <servicio>** Detiene el servicio.
- **Enable <servicio>** Activa el servicio iniciarse junto al arranque del sistema.
- **Status <servicio>** Muestra el estado del servicio.

Sin embargo, en FreeBSD utilizamos la orden:

service <servicio> start|stop|enable|status

En el caso de Windows Server, utilizamos la orden **net start|stop <servicio>**

4. CentOS

4.1 SSH, SFTP, SCP

El servicio SSH viene instalado por defecto en CentOS.

En el archivo **/etc/ssh/sshd_config** debemos cambiar y añadir los siguientes puntos:

Port 22

AllowUsers javi

PermitRootLogin no

Para poder conectarnos con una clave pública:

PubkeyAuthentication yes

UsePAM yes

A continuación, genero las claves (pública y privada) en el (host) con **ssh-keygen -t rsa**. Una vez generadas, éstas estarán guardadas en la carpeta .ssh/

Ahora debemos enviar la clave pública al servidor y copiarla en el archivo **authorized_keys** mediante el siguiente comando:

```
cat ~/.ssh/id_rsa.pub | ssh javi@192.168.56.114 "mkdir ~/.ssh; cat >> ~/.ssh/authorized_keys"
```

Muy importante darles permisos a las carpetas .ssh/ (chmod 700) y al archivo **authorized_keys** (chmod 600) situadas en el servidor.

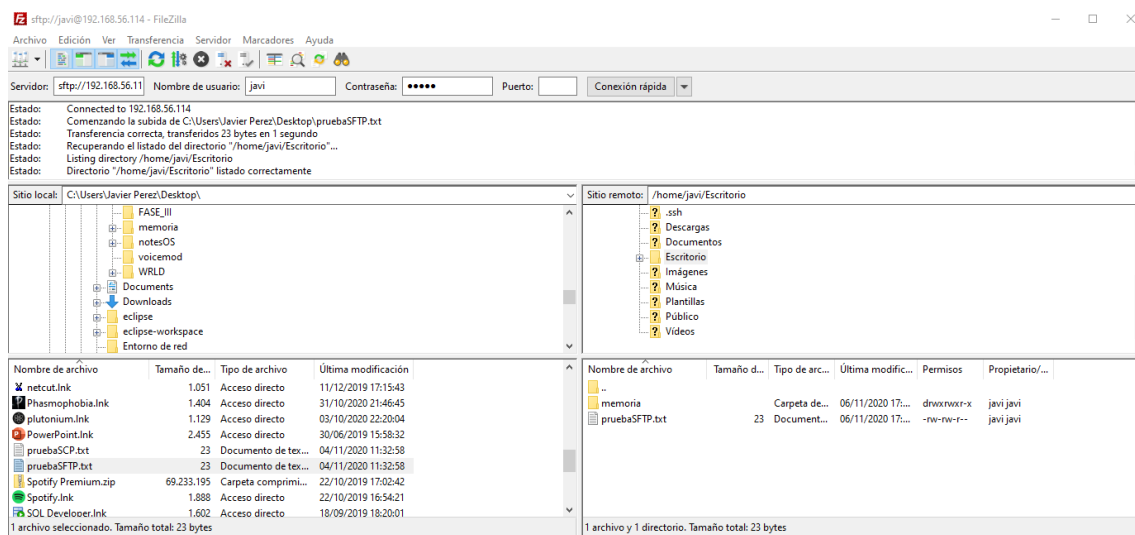
Ahora ya podremos conectarnos sin necesidad de introducir una contraseña gracias a la clave pública generada anteriormente.

```
C:\Users\Javier Perez>ssh 192.168.56.114
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Fri Nov  6 17:48:11 2020 from 192.168.56.1
[javi@localhost ~]$
```

Al ya tener instalado el servicio **SSH**, dicho servicio nos proporciona dos servicios más, uno de acceso a ficheros (**SFTP**) y otro que nos permite copiar archivos entre equipos de forma segura (**SCP**).

Para comprobar el servicio SFTP he usado el cliente FileZilla, para ello he creado un archivo llamado '*pruebaSFTP.txt*' y lo enviamos al servidor para verificar que todo funciona correctamente.



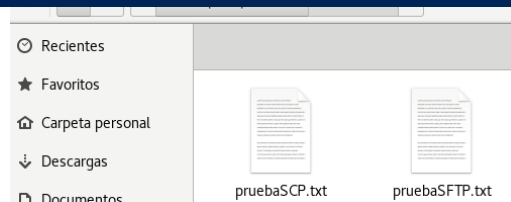
```
Last login: Fri Nov  6 17:48:11 2020 from 192.168.56.1
[javi@localhost ~]$ cd Escritorio/
[javi@localhost Escritorio]$ ls
memoria pruebaSFTP.txt
[javi@localhost Escritorio]$
```

Para comprobar el servicio SCP simplemente usando el comando siguiente:

scp -p <puerto> <ruta origen archivo> user@ip:<ruta destino archivo>

```
PS C:\Users\Javier Perez\Desktop> scp -p 22 .\pruebaSCP.txt javi@192.168.56.114:/home/javi/Escritorio
22: No such file or directory
pruebaSCP.txt 100% 23 7.6KB/s 00:00
PS C:\Users\Javier Perez\Desktop>
```

```
[javi@localhost ~]$ cd Escritorio/
[javi@localhost Escritorio]$ ls
pruebaSCP.txt pruebaSFTP.txt
```



Como vemos se ha copiado correctamente el archivo '*pruebaSCP.txt*'.

4.2 VNC + RDP

VNC (Virtual Network Computing) es un programa de software libre basado en una estructura cliente-servidor el cual nos permite tomar el control del ordenador servidor remotamente a través de un ordenador cliente (Escritorio remoto).

En primer lugar, instalamos VNC en el servidor:

```
dnf install tigervnc-server tigervnc-server-module -y
```

configuramos la contraseña:

```
vncpasswd
```

El siguiente paso es configurar el archivo de configuración del servidor VNC.

```
nano /etc/systemd/system/vncserver@.service
```

```
[Unit]
Description=Remote desktop service (VNC)
After=syslog.target network.target
[Service]
Type=forking
ExecStartPre=/bin/sh -c '/usr/bin/vncserver -kill %i > /dev/null 2>&1 || :'
ExecStart=/sbin/runuser -l javi -c "/usr/bin/vncserver %i -geometry 1280x1024"
PIDFile=/home/javi/.vnc/%H%i.pid
ExecStop=/bin/sh -c '/usr/bin/vncserver -kill %i > /dev/null 2>&1 || :'
[Install]
WantedBy=multi-user.target
```

A continuación, iniciamos el servicio VNC y permitimos el puerto de escucha (5901) en el firewall.

```
systemctl Daemon-reload
```

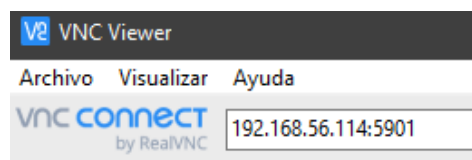
```
systemctl start vncserver@:1.service
```

```
netstat -tunlp | grep 5901
```

```
[root@localhost ~]# netstat -tunlp | grep 5901
tcp        0      0 0.0.0.0:5901        0.0.0.0:*          LISTEN      37228/Xvnc
tcp6       0      0 :::5901            :::*                LISTEN      37228/Xvnc
```

Ahora en nuestro host descargamos el cliente VNC Viewer e introducimos la dirección ip y el puerto.

192.168.56.107:5901



192.168.56.114:5901

Para configurar RDP, debemos instalarlo primero con el siguiente comando:

```
dnf install epel-release
```

```
dnf install xrdp
```

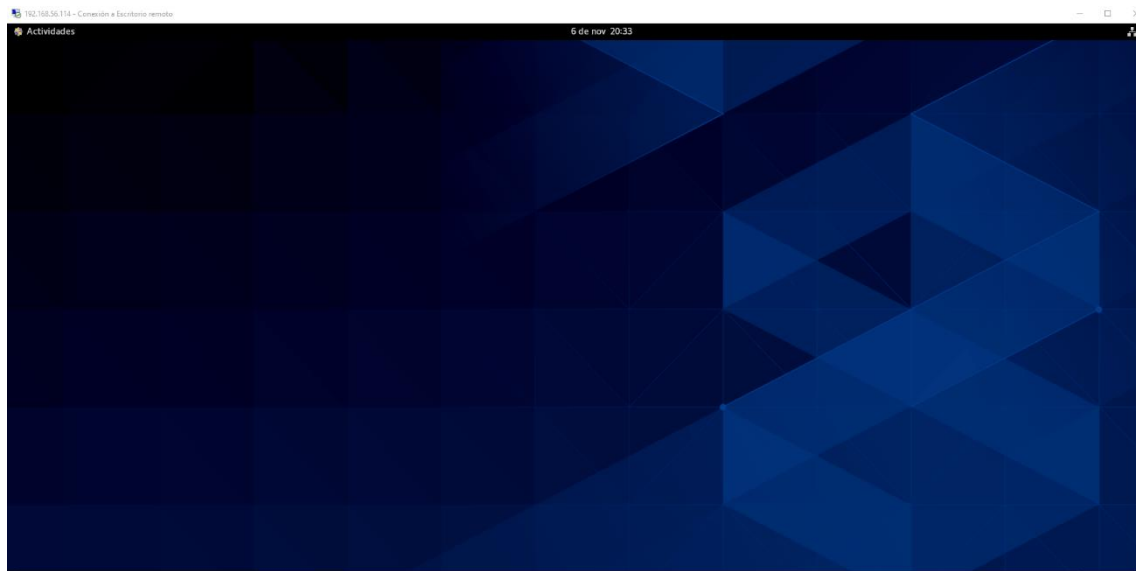
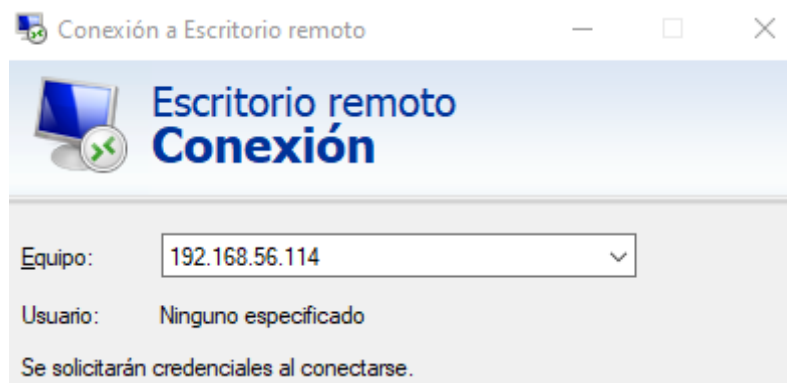
A continuación, habilitamos el puerto 3389 por si no está permitido en el firewall.

```
firewall-cmd --permanent --add-port=3389/tcp
```

```
firewall-cmd --reload
```

```
[root@localhost ~]# netstat -tulnp | grep 3389
tcp        0      0 0.0.0.0:3389        0.0.0.0:*          LISTEN     2376/xrdp
```

Activamos el servicio y mediante conexión a escritorio remoto desde el host nos conectaremos remotamente.



4.3 BD Server

MariaDB

He instalado el servidor de mariadb en CentOS para poder gestionar un sistema de bases de datos siguiendo los siguientes pasos:

En primer lugar, ejecutamos el siguiente comando en el terminal:

```
dnf -y install mariadb-server
```

Abrimos el puerto 3306.

A continuación, ejecutamos los siguientes comandos:

```
systemctl start mariadb → para iniciar el servicio.
```

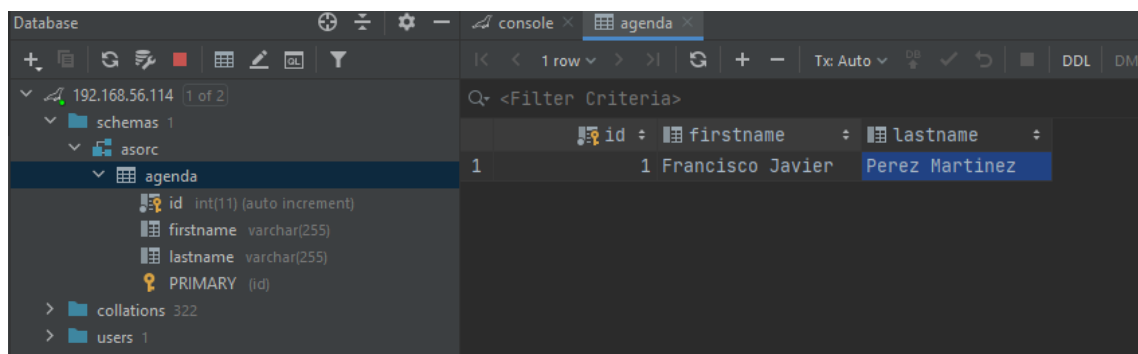
```
mysql_secure_installation → para realizar la instalación de forma segura.
```

```
mysql -u root -p → Accedemos al servicio con el usuario root y con la contraseña previamente establecida durante la instalación de dicho servicio.
```

Creamos la base de datos con **create database asorc** y un usuario y contraseña que gestionara la BD con **grant all privileges on asorc.* to asorcuser@'%' identified by 'password'**.

Ahora nos vamos a un gestor de base de datos, en nuestro caso, **DataGrip** y creamos una nueva conexión con el data source = mariaDB.

He creado una nueva tabla (agenda) con los campos: id, firstname y lastname.



En nuestro servidor mariadb haciendo una select nos mostraría el contenido de dicha tabla:

```
Database changed
MariaDB [asorc]> select * from agenda;
+-----+-----+-----+
| id | firstname      | lastname      |
+-----+-----+-----+
| 1 | Francisco Javier | Perez Martinez |
+-----+-----+-----+
1 row in set (0.000 sec)
```

Para probar que funciona correctamente, vamos a probar con un cliente php que vamos a ejecutar a través de un navegador web.

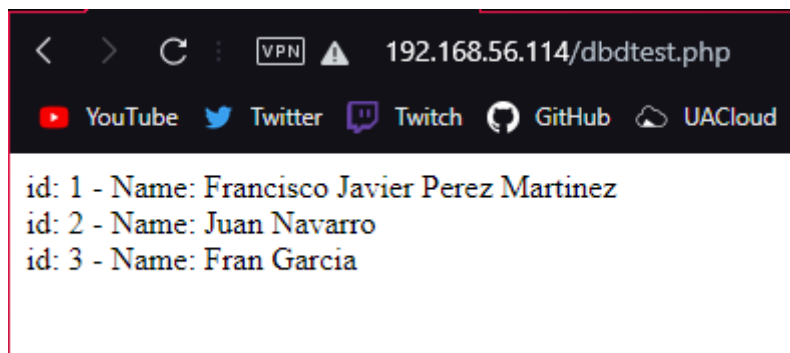
Instalaremos Apache, el cual tiene una característica y es que nos permite validar la configuración antes de ejecutar apache. Para ello, ejecutamos **apachectl configtest**.

Nos saldrá que debemos cambiar el ServerName, entramos en archivo de configuración de httpd **/etc/httpd/conf/httpd.conf** y lo editamos. **ServerName asorc.org:80**

Ahora ya podríamos lanzar httpd con **systemctl start httpd**.

Para terminar, con un programa que he hecho para que lea nuestra base de datos lo añadimos a la carpeta **/var/www/html** y ahora en el navegador escribimos lo siguiente:

192.168.56.114/dbdtest.php



Como se puede observar nos aparecen los campos que hemos creado anteriormente en nuestra base de datos los cuales estamos mostrando gracias a nuestro programa php implementado.

4.4 Web Server

En primer lugar, debemos crear los dominios virtuales en el servidor. Para ello, vamos al fichero de configuración `cd /etc/httpd/conf.d` y creamos el archivo `virtualdomains.conf`, éste archivo se cargará con apache debido al `.conf`.

Dentro del archivo introducimos los dominios virtuales:

```
<VirtualHost *:80>
    DocumentRoot /var/www/html/embutidosgutierrez.com
    ServerName embutidosgutierrez.com
</VirtualHost>

<VirtualHost *:80>
    DocumentRoot /var/www/html/almendraslaavellana.com
    ServerName almendraslaavellana.com
</VirtualHost>
```

Ahora necesitamos crear los directorios `/var/www/html/embutidosgutierrez.com` y `/var/www/html/almendraslaavellana.com`

Una vez creados, el servidor necesita servir un index por lo que lo creamos dentro de ambos directorios. Ejemplo:

```
<html> <head></head>

    <body> Almendras la Avellana </body>

</html>
```

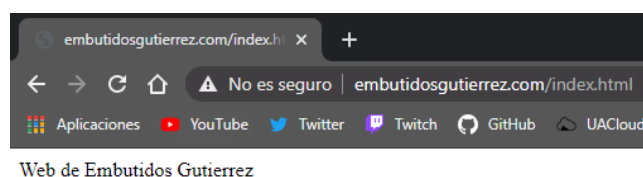
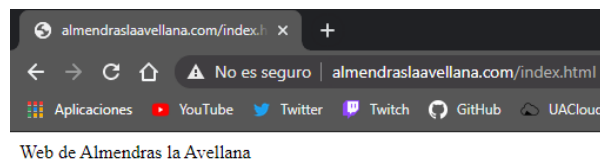
A continuación, tenemos dos opciones utilizar el dns (crear el domino) o utilizar el fichero hosts, en nuestro caso nos dirigimos a dicho fichero situado en la siguiente ruta:

C:\Windows\System32\drivers\etc\hosts

E introducimos las entradas siguientes:

- **192.168.56.114 almendraslaavellana.com**
- **192.168.56.114 embutidosgutierrez.com**

systemctl restart httpd



Ahora descargaremos Joomla para que nuestro dominio virtual ejecute un gestor de contenido, en este caso Joomla.

wget https://downloads.joomla.org/cms/joomla3/3-9-22/Joomla_3-9-22-Stable-Full_Package.zip?format=zip

Descomprimos en la carpeta del dominio y como lo va a ejecutar apache vamos a cambiar el contenido de root a apache con el comando **chown -R apache.apache ***.

Ahora debemos instalar más paquetes de php para poder ejecutar nuestro gestor de contenido:
\$ sudo dnf install php-curl php-xml php-zip php-mysqlnd php-intl php-gd php-json php-ldap php-mbstring php-opcache

Ahora necesitamos que el servidor web sea capaz de utilizar todos estos paquetes para que sea compatible con Joomla. Para ello, editamos el archivo de configuración que anteriormente hemos creado con **nano /etc/httpd/conf.d/virtualdomains.conf**

```
GNU nano 2.9.8 /etc/httpd/conf

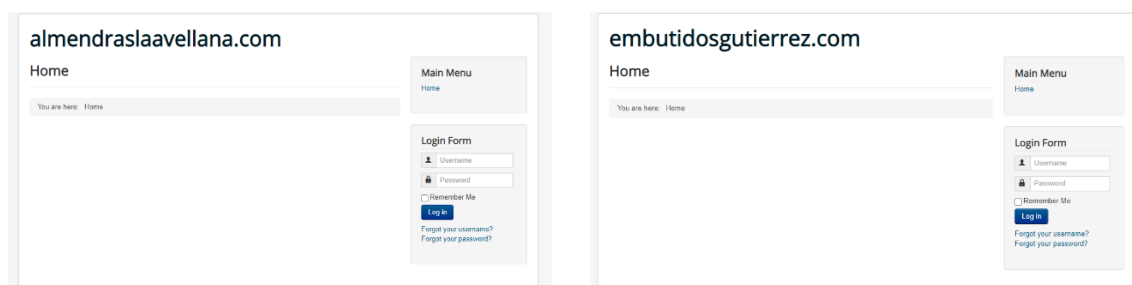
<VirtualHost *:80>
    DocumentRoot /var/www/html/embutidosgutierrez.com
    ServerName embutidosgutierrez.com
    ServerAdmin javi18pm@gmail.com
    ErrorLog "/var/log/httpd/embutidosgutierrez.com-error_log"
    CustomLog "/var/log/httpd/embutidosgutierrez.com-access_log" combined

    <Directory "/var/www/html/embutidosgutierrez.com">
        DirectoryIndex index.html index.php
        Options FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>
</VirtualHost>

<VirtualHost *:80>
    DocumentRoot /var/www/html/almendraslaavellana.com
    ServerName almdndraslaavellana.com
    ServerAdmin javi18pm@gmail.com
    ErrorLog "/var/log/httpd/almendraslaavellana.com-error_log"
    CustomLog "/var/log/httpd/almendraslaavellana.com-access_log" combined

    <Directory "/var/www/html/almendraslaavellana.com">
        DirectoryIndex index.html index.php
        Options FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>
</VirtualHost>
```

El instalador Joomla necesita una cuenta de base datos la cual creamos como se ha explicado anteriormente en el apartado de [servidor de BD](#).



Y por último hacemos un **systemctl restart httpd** para que se apliquen los cambios previamente descritos.

4.5 NFS / SAMBA

Para instalar el servicio de NFS utilizaremos el siguiente comando:

```
dnf -y install nfs-utils
```

A continuación, editamos el fichero `/etc/exports` en el que introducimos que quiero compartir y con quién.

```
/data/nfs_shared 192.168.56.0/24(rw,no_root_squash,sync)
```

En este caso compartimos con toda la red y a continuación iniciamos el servicio con **systemctl start nfs-server**

Una vez instalado, para probarlo lo probaremos con el siguiente comando con el comando **showmount -e 192.168.56.114**

En mi caso he utilizado como cliente Debian ya que en mi host (Windows 10) no disponía del servicio de NFS debido a mi versión del sistema operativo (Windows 10 Home).

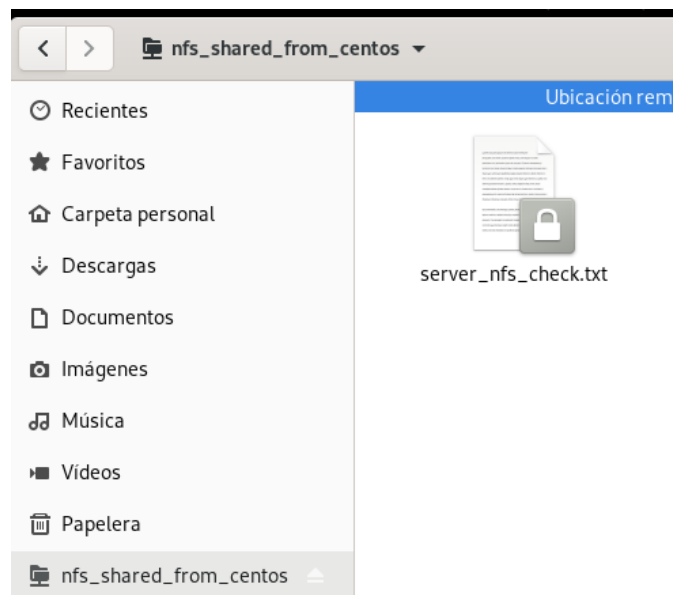
Nos crearemos una carpeta **mkdir nfs_shared_from_centos** y con el comando

```
sudo mount -t nfs -o resvport,rw 192.168.56.114:/data/nfs_shared /home/javi/nfs_shared_fromcentos y lo montamos con mount.
```

En el servidor he creado un fichero para comprobar que todo funciona correctamente.

```
touch /data/nfs_shared/server_nfs_check.txt
```

Y comprobamos que en nuestro cliente aparece la carpeta compartida con el fichero creado en el servidor.



Como se puede observar, el fichero se muestra correctamente en el cliente.

Para instalar el servicio de SAMBA utilizaremos el siguiente comando:

```
dnf -y install samba samba-client samba-common
```

Ahora pasamos al archivo de configuración donde añadimos nuestro grupo de trabajo:

```
nano /etc/samba/smb.conf
```

```
workgroup = WORKGROUP
```

Y ahora crearemos una unidad compartida:

```
[samba_shared]
    comment = Carpeta compartida
    path = /data/samba_shared
```

El directorio añadido al path no existe por lo que lo creamos con el comando siguiente
mkdir -p /data/samba_shared

A continuación, crear el usuario donde se va a conectar y creamos el password de samba.

```
smbpasswd -a javi
```

E iniciamos el servicio con el comando **systemctl start nmbd smb.**

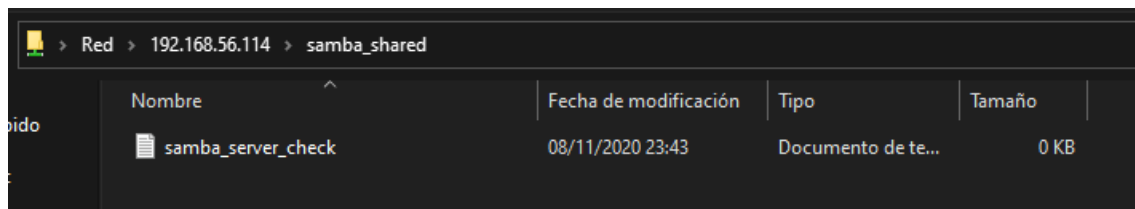
Por último, abrimos los puertos de samba:

```
$ sudo firewall-cmd --add-service=samba --zone=public --permanent
```

```
$ sudo firewall-cmd --reload
```

Con las teclas Win + R escribimos <\\IP-servidor-samba>

En el servidor creamos un fichero: **touch /data/samba_shared/samba_server_check**



Como vemos aparece el fichero en nuestra carpeta compartida con el host.

4.6 DHCP

Antes de instalar el servidor dhcp debemos configurar la red para que aparezca como en el enunciado de la práctica. Editamos el archivo

nano /etc/sysconfig/network-scripts/ifcfg-enp0s8 y añadimos las siguientes líneas:

```
BOOTPROTO=none
IPADDR=192.168.56.222
NETMASK=255.255.255.0
```

Ahora reiniciamos la red con los comandos: **nmcli con reload** y **nmcli con up enp0s8**

Nos tocaría configurar todos los servicios previamente realizados con esta nueva IP-.

Ahora desactivamos el servidor dhcp de virtualbox y reiniciamos la máquina virtual.

Una vez reiniciado, instalamos el servidor dhcp en nuestra máquina virtual:

dnf install dhcp-server y configuramos el archivo **/etc/dhcp/dhcpd.conf**

```
#
# DHCP Server Configuration file.
# see /usr/share/doc/dhcp-server/dhcpd.conf.example
# see dhcpd.conf(5) man page
#
shared-network network.net {
    subnet 192.168.56.0
    netmask 255.255.255.0 {
        #option routers 192.168.56.1;
        option subnet-mask 255.255.255.0;
        option broadcast-address 192.168.56.255;
        #option domain-name-servers 192.168.2.100;
        range 192.168.56.51 192.168.56.100;
    }
}
host learn {
    option host-name "nodoA.network.net";
    hardware ethernet 00:25:d3:66:63:b3;
    fixed-address 192.168.56.101;
}
```

Por último, iniciamos el servicio dhcpd **systemctl start dhcpd** y comprobamos que el servidor DHCP funciona iniciando otra máquina virtual y comprobando el rango de direcciones IP que le hemos dado a nuestro servidor.

```
javi@javi-virtualbox:
Archivo  Editar  Ver  Buscar  Terminal  Ayuda

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.56.52 netmask 255.255.255.0 broadcast 192.168.56.255
    inet6 fe80::3b54:559b:20d4:d4e6 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:71:5d:09 txqueuelen 1000 (Ethernet)
    RX packets 6 bytes 926 (926.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 44 bytes 5863 (5.8 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Como se puede observar, el servidor le ha asignado la dirección IP 192.168.56.52, IP disponible dentro del rango configurado.

4.7 CUPS

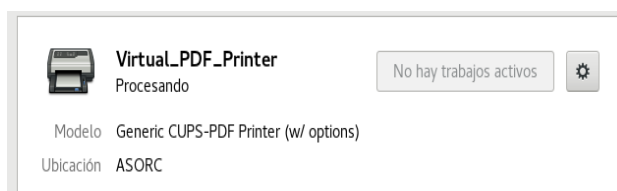
Para instalar el servicio CUPS seguimos los siguientes pasos:

```
yum update
yum -y install cups gcc gcc-c++ cups-devel tar wget
wget https://www.cups-pdf.de/src/cups-pdf_3.0.1.tar.gz
tar -xvf cups-pdf_3.0.1.tar.gz
cd cups-pdf-3.0.1/src/
gcc -O9 -s cups-pdf.c -o cups-pdf -lcups
chmod 700 cups-pdf
cp -p cups-pdf /usr/lib/cups/backend/
cd ../extra
cp cups-pdf.conf /etc/cups/
cp CUPS-PDF_opt.ppd /usr/share/cups/model/
firewall-cmd --zone=public --add-port=631/tcp --permanent
firewall-cmd --reload
sed -i "s/Allow \@LOCAL/Allow all/g" /etc/cups/cupsd.conf
systemctl restart cups
systemctl enable cups
cupsctl --remote-admin
```

A continuación, iniciamos el servicio **systemctl start cups**

Y configuramos el archivo de configuración de CUPS **nano /etc/cups/cupsd.conf**

```
# Allow remote access
Port 631
Listen localhost:631
Listen 192.168.56.222:631
Listen /var/run/cups/cups.sock
Browse On
BrowseLocalProtocols dnssd
DefaultAuthType Basic
WebInterface Yes
<Location />
  # Allow remote administration...
  Order allow,deny
  Allow all
  Allow @LOCAL
</Location>
<Location /admin>
  # Allow remote administration...
  Order allow,deny
  Allow all
  Allow @LOCAL
</Location>
```



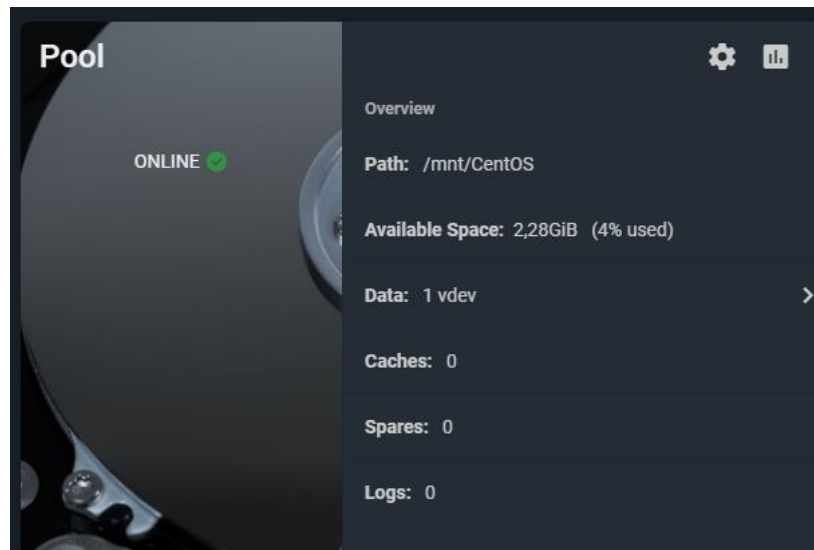
Y desde un navegador web accedemos al servicio de impresoras CUPS y agregamos la impresora PDF en nuestro host.

Para ver si funciona desde un cliente agregamos la impresora en Windows:

`http://192.168.56.222:631/printers/{nombre_impresora}`

4.8 FreeNAS + iSCSI

Para instalar el servicio iSCSI con FreeNAS, debemos instalar primero la máquina virtual FreeNAS y configurar desde el navegador todos los campos para el correcto funcionamiento del recurso compartido de disco con iSCSI.



Una vez configurado FreeNAS, procedemos a instalar iSCSI en CentOS:

```
$ dnf install iscsi-initiator-utils
```

Una vez instalado realizamos los siguientes comandos:

```
[javi@localhost ~]$ sudo dnf install iscsi-initiator-utils
[sudo] password for javi:
Ultima comprobación de caducidad de metadatos hecha hace 0:48:25, el
El paquete iscsi-initiator-utils-6.2.0.878-4.gitd791ce0.el8.x86_64 y
Dependencias resueltas.
Nada por hacer.
¡Listo!
[javi@localhost ~]$ su -
Contraseña:
[root@localhost ~]# iscsiadm -m iface -I enp0s8 -o new
New interface enp0s8 added
[root@localhost ~]# iscsiadm -m discovery -t st -p 192.168.56.115
192.168.56.115:3260,-1 iqn.2005-10.org.freenas.ct1:freenas
```

Usaremos la red enp0s8 como red de compartición de datos con FreeNAS.

```
[root@localhost ~]# iscsiadm -m node -T iqn.2005-10.org.freenas.ct1:freenas -p 192.168.56.115 -l -I enp0s8
Logging in to [iface: enp0s8, target: iqn.2005-10.org.freenas.ct1:freenas, portal: 192.168.56.115,3260]
Login to [iface: enp0s8, target: iqn.2005-10.org.freenas.ct1:freenas, portal: 192.168.56.115,3260] successful.
[root@localhost ~]# lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda           8:0      0 20,2G  0 disk
├─sda1        8:1      0   1G  0 part /boot
├─sda2        8:2      0 19,2G  0 part
├─┌cl-root    253:0    0 17,2G  0 lvm  /
└─┌cl-swap    253:1    0   2G  0 lvm  [SWAP]
sdb           8:16     0 15,8G  0 disk
├─sdb1        8:17     0 15,8G  0 part /data
sdc           8:32     0  10G  0 disk
sr0          11:0     1 1024M  0 rom
```

apartado de [particiones](#).

Con el comando **df -Th** comprobaremos que se ha montado.

```
[root@localhost ~]# df -Th
```

Filesystem	Tipo	Tamaño	Usados	Disp	Uso%	Montado en
devtmpfs	devtmpfs	1,9G	0	1,9G	0%	/dev
tmpfs	tmpfs	1,9G	0	1,9G	0%	/dev/shm
tmpfs	tmpfs	1,9G	9,9M	1,9G	1%	/run
tmpfs	tmpfs	1,9G	0	1,9G	0%	/sys/fs/cgroup
/dev/mapper/cl-root	xfs	18G	6,9G	11G	40%	/
/dev/sda1	ext4	976M	353M	557M	39%	/boot
/dev/sdb1	ext4	16G	45M	15G	1%	/data
tmpfs	tmpfs	378M	16K	378M	1%	/run/user/42
tmpfs	tmpfs	378M	4,0K	378M	1%	/run/user/1000
/dev/sdc1	xfs	10G	104M	9,9G	2%	/mnt

Por último, añadiremos una entrada en **/etc/fstab** para que monte el drive de iSCSI durante el arranque del sistema.

4.9 Git + OwnCloud

Para instalar los servicios de Git y ownCloud se deberán instalar el conjunto LAMP (Linux – Apache – MySQL – PHP).

Una vez instalados, iniciamos los servicios:

```
systemctl start httpd
```

```
systemctl start mariadb
```

Habilitamos el acceso externo al servidor abriendo los puertos:

```
firewall-cmd --permanent --zone=public --add-service=http
```

```
firewall-cmd --permanent --zone=public --add-service=https
```

```
systemctl reload firewalld
```

Creamos una base de datos para poder alojar los archivos en la nube.

```
MariaDB [(none)]> CREATE DATABASE owncloud_db;
Query OK, 1 row affected (0.003 sec)

MariaDB [(none)]> GRANT ALL ON owncloud_db.* TO 'ownclouduser'@'localhost' IDENTIFIED BY 'password';
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> EXIT;
Bye
```

A continuación, descargamos ownCloud en nuestro sistema:

```
sudo wget https://download.owncloud.org/community/owncloud-10.3.2.tar.bz2
```

Y lo extraemos en la carpeta **/var/www/**

```
sudo tar -jxf owncloud-10.3.2.tar.bz2 -C /var/www/
```

Configuramos los permisos de Apache en dicho directorio:

sudo chown -R apache: /var/www/owncloud

Creamos un fichero de configuración de Apache para el acceso de ownCloud: **sudo nano /etc/httpd/conf.d/owncloud.conf** y añadimos las siguientes líneas:

```
[javi@localhost memoria]$ cat /etc/httpd/conf.d/owncloud.conf
Alias /owncloud "/var/www/owncloud/"

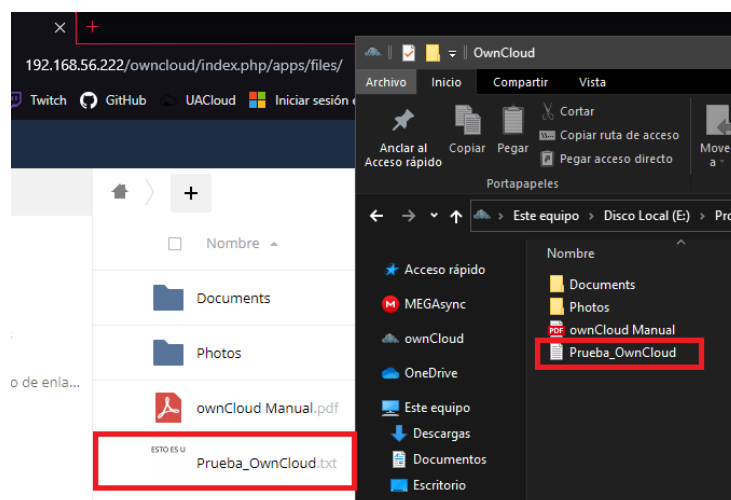
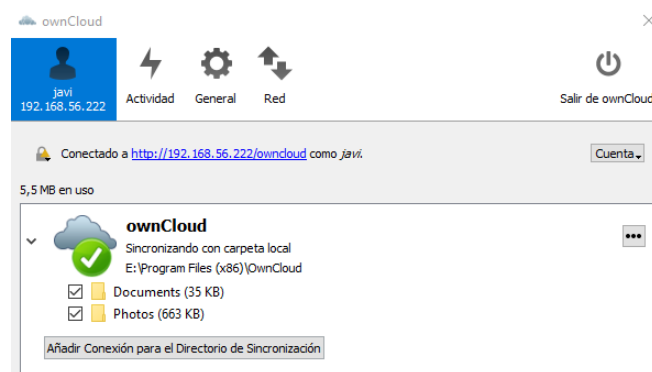
<Directory /var/www/owncloud/>
    Options +FollowSymLinks
    AllowOverride All

    <IfModule mod_dav.c>
        Dav off
    </IfModule>

    SetEnv HOME /var/www/owncloud
    SetEnv HTTP_HOME /var/www/owncloud
</Directory>
```

Para completar la configuración, desde el navegador accederemos a ownCloud mediante la **ip-del-servidor/owncloud** e introduciremos los campos pertinentes.

Por último, desde un cliente en nuestro host podremos compartir/subir archivos y éstos se actualizarán automáticamente en nuestro servidor.



Ahora pasaremos a instalar **git server**, para ello introducimos el siguiente comando:

```
$ sudo yum install git
```

A continuación, crearemos un repositorio local en nuestro servidor.

```
$ mkdir ~/repo
```

```
$ cd ~/repo
```

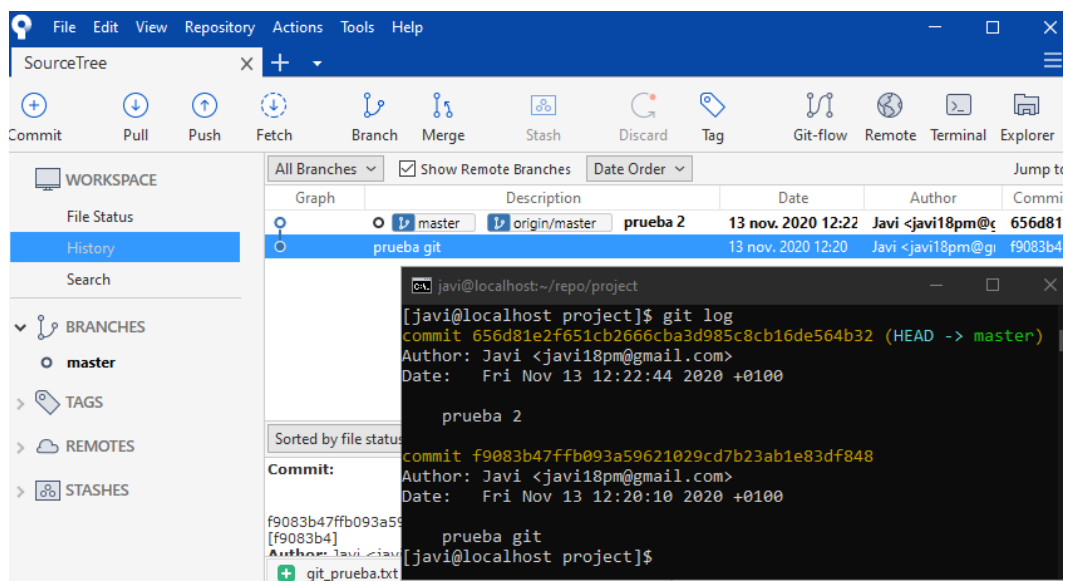
```
$ git init --bare --shared Project
```

Una vez creado el repositorio local, debemos habilitar el post-update hook copiando el fichero sample tal y como aparece a continuación:

```
$ cd ~/repo/project/hooks/
```

```
$ cp post-update.sample post-update
```

Por último, nos descargaremos un cliente para trabajar en nuestro host:



4.10 DNS

Para instalar el servicio de DNS debemos instalar bind:

dnf install bind bind-utils

Editamos el archivo **nano /etc/named.conf** y configuramos nuestras zonas.

```
options {
    listen-on port 53 { 127.0.0.1; 192.168.56.222; localhost; };
    listen-on-v6 port 53 { ::1; };
    directory "/var/named";
    forwarders{193.145.233.5; 8.8.8.8; 8.8.4.4; };
    dump-file "/var/named/data/cache_dump.db";
    statistics-file "/var/named/data/named_stats.txt";
    memstatistics-file "/var/named/data/named_mem_stats.txt";
    secroots-file "/var/named/data/named.secroots";
    recursing-file "/var/named/data/named.recursing";
    allow-query { localhost; 192.168.56.0/24; };
```

```
zone "almendraslaavellana.com" IN {
    type master;
    file "almendraslaavellana.zone";
    allow-update {none;};
};

zone "embutidosgutierrez.com" IN {
    type master;
    file "embutidosgutierrez.zone";
    allow-update {none;};
};

zone "56.168.192.in-addr.arpa" IN {
    type master;
    file "reversa.zone";
    allow-update {none;};
};
```

A continuación, creamos los archivos de configuración de las zonas creadas:

nano /var/named/almendraslaavellana.zone

```
[root@localhost named]# cat almendraslaavellana.zone
$TTL 1D
@      IN SOA  server.almendraslaavellana.com. root.almendraslaavellana.com. (
                                0      ;serial
                                1D     ;refresh
                                1H     ;retry
                                1W     ;expire
                                3H )   ;minimum
@      IN     NS   server.almendraslaavellana.com.
@      IN     A    192.168.56.222
server IN     A    192.168.56.222
host   IN     A    192.168.56.222
```

nano /var/named/embutidosgutierrez.zone

```
$TTL 86400
@ IN SOA network.net. root.network.net.(
150115
28800
7200
604800
86400
)
@ IN NS servidor.network.net.
222.56.168.192.in-addr.arpa. IN PTR servidor.network.net.
223.56.168.192.in-addr.arpa. IN PTR www.network.net.
```

Configuramos los permisos a los dos archivos de configuración creados:

chown named:named /var/named/almendraslaavellana.zone

chown named:named /var/named/embutidosgutierrez.zone

Iniciamos el servicio:

systemctl start named

Añadimos el servicio DNS al firewall:

firewall-cmd --add-service=dns --zone=public --permanent

firewall-cmd -reload

systemctl restart named

Por último, añadimos el **nameserver 192.168.56.222** en **/etc/resolv.conf**

```
[root@localhost named]# nslookup
> almdraslaavellana.com
Server:          192.168.56.222
Address:         192.168.56.222#53

Name:   almdraslaavellana.com
Address: 192.168.56.222
> embutidosgutierrez.com
Server:          192.168.56.222
Address:         192.168.56.222#53

Name:   embutidosgutierrez.com
Address: 192.168.56.222
> 192.168.56.222
222.56.168.192.in-addr.arpa    name = embutidosgutierrez.com.
223.56.168.192.in-addr.arpa    name = almdraslaavellana.com.
>
```

5. FreeBSD

5.1 SSH, SFTP, SCP

El servicio SSH viene instalado por defecto en FreeBSD, pero necesitamos activarlo.

En el archivo **/etc/rc.conf** debemos insertar la siguiente línea:

sshd_enable = "YES"

Ahora editamos el archivo de configuración **nano /etc/ssh/sshd_config** e introducimos las siguientes líneas:

Port 22

AllowUsers Javi

PermitRootLogin no

Para poder conectarnos con una clave pública:

PubkeyAuthentication yes

UsePAM yes

A continuación, genero las claves (pública y privada) en el (host) con **ssh-keygen -t rsa**. Una vez generadas, éstas estarán guardadas en la carpeta **.ssh/**

Ahora debemos enviar la clave pública al servidor y copiarla en el archivo **authorized_keys** mediante el siguiente comando:

```
cat ~/.ssh/id_rsa.pub | ssh Javi@192.168.56.104 "cat >>
~/.ssh/authorized_keys"
```

Ahora ya podremos conectarnos sin necesidad de introducir una contraseña gracias a la clave pública generada anteriormente.

```
PS C:\Windows\system32> ssh Javi@192.168.56.104
Last login: Mon Nov  9 20:29:12 2020 from 192.168.56.1
FreeBSD 12.1-RELEASE r354233 GENERIC

Welcome to FreeBSD!

Release Notes, Errata: https://www.FreeBSD.org/releases/
Security Advisories:  https://www.FreeBSD.org/security/
FreeBSD Handbook:    https://www.FreeBSD.org/handbook/
FreeBSD FAQ:         https://www.FreeBSD.org/faq/
Questions List:      https://lists.FreeBSD.org/mailman/listinfo/freebsd-questions/
FreeBSD Forums:      https://forums.FreeBSD.org/

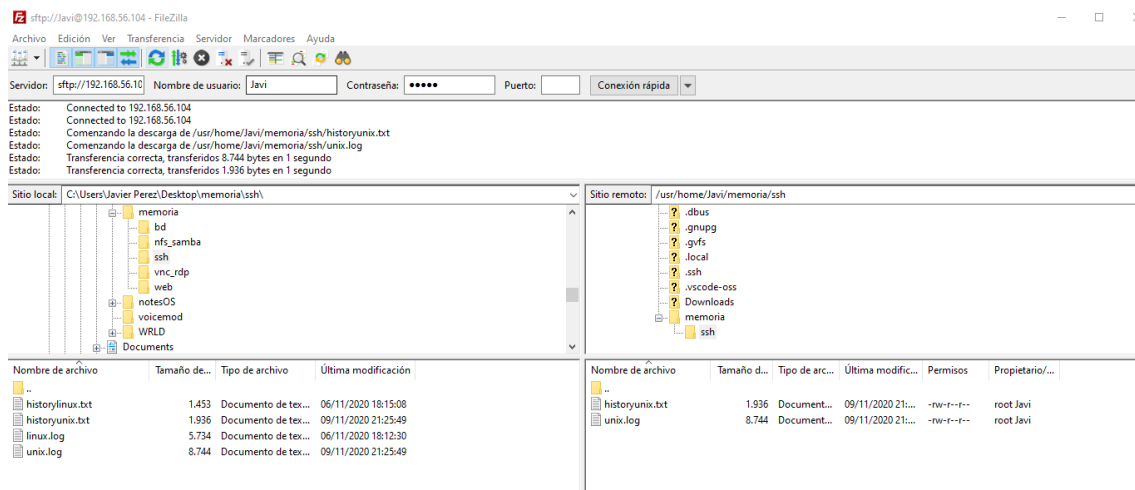
Documents installed with the system are in the /usr/local/share/doc/freebsd/
directory, or can be installed later with:  pkg install en-freebsd-doc
For other languages, replace "en" with a language code like de or fr.

Show the version of FreeBSD installed:  freebsd-version ; uname -a
Please include that output and any error messages when posting questions.
Introduction to manual pages:  man man
FreeBSD directory layout:      man hier

Edit /etc/motd to change this login announcement.
If you write part of a filename in tcsh,
pressing TAB will show you the available choices when there
is more than one, or complete the filename if there's only one match.
$
```


Al ya tener instalado el servicio **SSH**, dicho servicio nos proporciona dos servicios más, uno de acceso a ficheros (**SFTP**) y otro que nos permite copiar archivos entre equipos de forma segura (**SCP**).

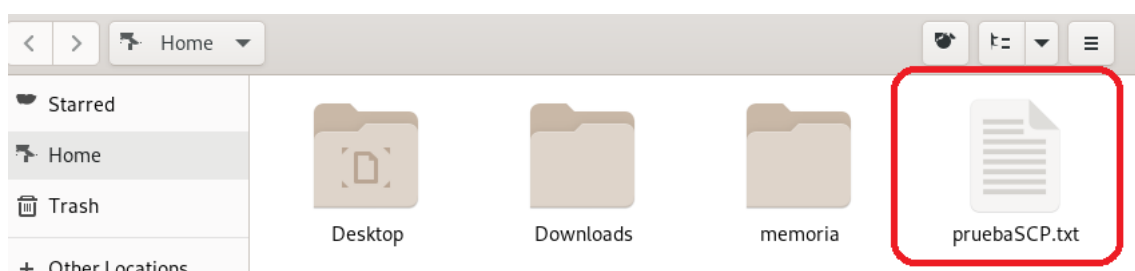
Para comprobar el servicio SFTP he usado el cliente FileZilla, para ello he enviado los logs del servicio de ssh y el history a una carpeta en mi host para verificar que todo funciona correctamente.



Para comprobar el servicio SCP simplemente usando el comando siguiente:

scp -p <puerto> <ruta origen archivo> user@ip:<ruta destino archivo>

```
PS C:\Users\Javier Perez\Desktop> scp -p 22 .\pruebaSCP.txt Javi@192.168.56.221:/home/Javi/
22: No such file or directory
pruebaSCP.txt                                100% 23    11.4KB/s   00:00
PS C:\Users\Javier Perez\Desktop>
```



Como vemos se ha copiado correctamente el archivo 'pruebaSCP.txt'.

5.2 DHCP

Antes de instalar el servidor dhcp debemos configurar la red para que aparezca como en el enunciado de la práctica. Editamos el archivo **nano /etc/rc.conf** e introducimos:

```
ifconfig_em1="inet 192.168.56.221 netmask 255.255.255.0"
```

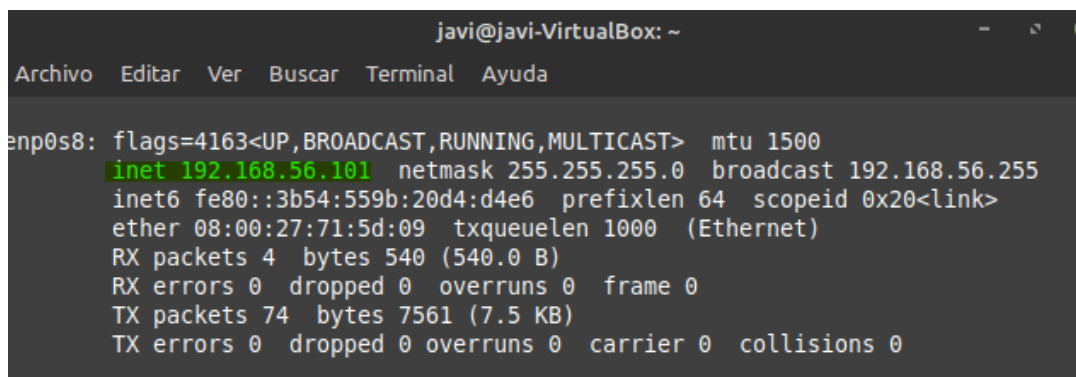
```
defaultrouter="192.168.56.1"
```

Ahora desactivamos el servidor dhcp de virtualbox y reiniciamos la máquina virtual.

Ahora instalamos el servicio: **pkg install isc-dhcp-server** y configuramos el archivo **/etc/rc.conf** añadiendo **dhcpd_enable = "YES"** y en el archivo **/usr/local/etc/dhcpd.conf**

```
# A slightly different configuration for an internal subnet.
subnet 192.168.56.0 netmask 255.255.255.0 {
    range 192.168.56.101 192.168.56.150;
    #option domain-name-servers ns1.internal.example.org;
    #option domain-name "internal.example.org";
    option routers 192.168.56.1;
    option broadcast-address 192.168.56.255;
    default-lease-time 600;
    max-lease-time 7200;
}
```

Por último, reiniciamos el servicio dhcpd **service isc-dhcpd restart** y comprobamos que el servidor DHCP funciona iniciando otra máquina virtual y comprobando el rango de direcciones IP que le hemos dado a nuestro servidor.



```
javi@javi-VirtualBox: ~
Archivo  Editar  Ver  Buscar  Terminal  Ayuda

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 192.168.56.101 netmask 255.255.255.0 broadcast 192.168.56.255
inet6 fe80::3b54:559b:20d4:d4e6 prefixlen 64 scopeid 0x20<link>
ether 08:00:27:71:5d:09 txqueuelen 1000 (Ethernet)
RX packets 4 bytes 540 (540.0 B)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 74 bytes 7561 (7.5 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Como se puede observar, el servidor le ha asignado la dirección IP 192.168.56.101, IP disponible dentro del rango configurado.

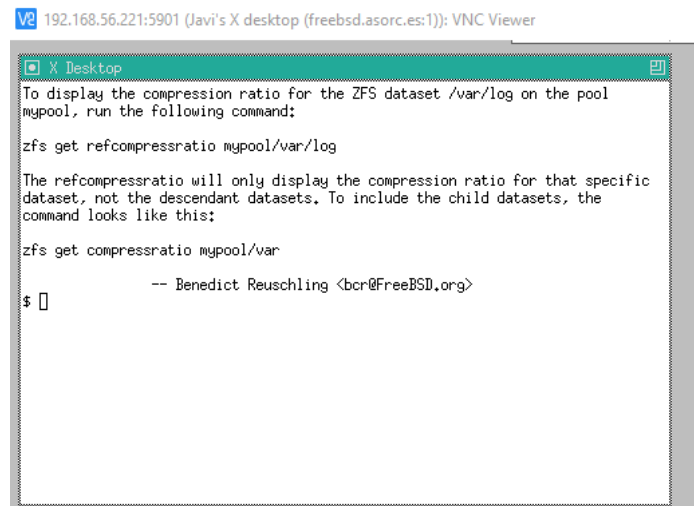
5.3 VNC + RDP

Para instalar el servicio de VNC utilizamos el siguiente comando:

```
pkg install tightvnc
```

```
vncserver
```

Ahora en nuestro cliente, comprobamos conectándonos al servidor.

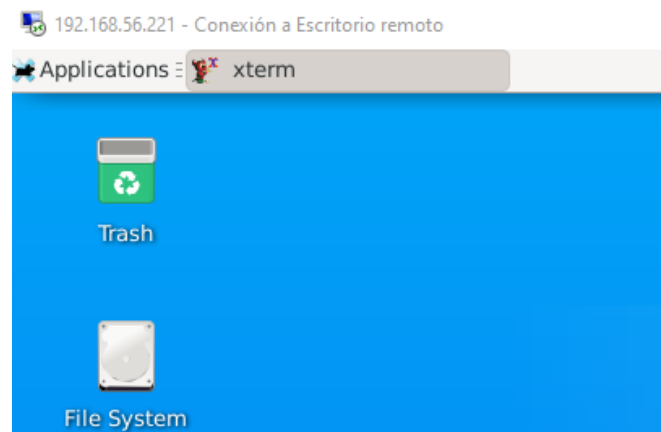


Si queremos quitar el servidor utilizamos el comando **vncserver -kill :1**

Para instalar el servicio de RDP utilizamos el siguiente comando **pkg install xrdp** y configuramos el archivo **/etc/rc.conf** introduciendo las siguientes líneas.

```
#rdp
xrdp_enable="YES"
xrdp_sesman_enable="YES"
```

Iniciamos el servicio **service xrdp start**



5.4 CUPS

Para instalar el servicio de CUPS debemos instalar lo siguiente:

pkg install cups cups-pdf

A continuación, editamos el archivo de configuración **/usr/local/etc/cups/cupsd.conf** y añadimos las siguientes líneas:

```
LogLevel info
PageLogFormat
Port 631
Listen localhost:631
Listen /var/run/cups/cups.sock
Listen 192.168.56.221:631
Browsing On
BrowseOrder allow,deny
BrowseLocalProtocols dnssd
DefaultAuthType Basic
WebInterface Yes
<Location />
    Order allow,deny
    Allow all
</Location>
```

También añadimos al archivo **/etc/rc.conf** las siguientes líneas:

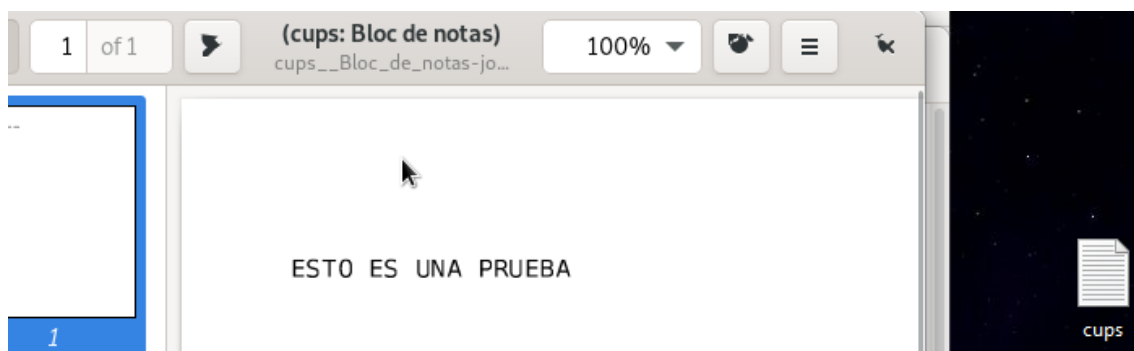
```
#cups
cupsd_enable="YES"
devfs_system_ruleset="system"
```

Iniciamos el servicio: **service cupsd start**.

Y desde un navegador web accedemos al servicio de impresoras CUPS y agregamos la impresora PDF en nuestro host.

Para ver si funciona desde un cliente agregamos la impresora en Windows:

http://192.168.56.221:631/printers/{nombre_impresora}



5.5 BD Server

PostgreSQL

Para el servicio de base de datos usaremos PostgreSQL para FreeBSD, instalaremos el paquete de postgres mediante el siguiente comando:

```
pkg install postgresql12-server
```

Una vez instalado, debemos configurar el archivo de arranque del sistema para que la base de datos se inicie junto al sistema:

```
sysrc postgresql_enable=yes
```

Ahora inicializamos PostgreSQL usando el siguiente comando:

```
/usr/local/etc/rc.d/postgresql initdb
```

E iniciamos el servicio:

```
service postgresql start  
service postgresql status
```

Ahora creamos un rol de PostgreSQL para nuestro usuario de FreeBSD,

```
sudo -u postgres createuser --interactive -P
```

```
createdb Javi -O Javi
```

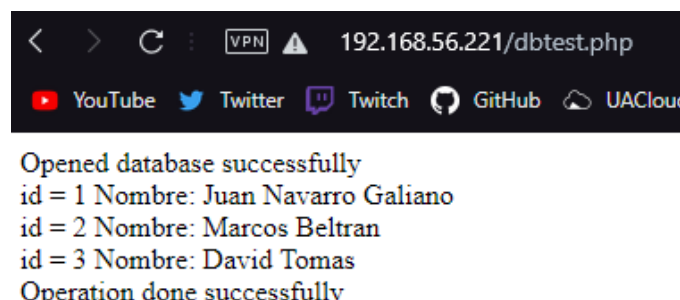
Configuramos e instalamos apache24 y php74.

```
pkg install apache24  
pkg install php74 mod_php74 php74-curl php74-session php74-pgsql  
cd /usr/local/etc/apache24/  
nano -c httpd.conf
```

```
<FilesMatch "\.php$">  
  SetHandler application/x-httpd-php  
</FilesMatch>  
<FilesMatch "\.phps$">  
  SetHandler application/x-httpd-php-source  
</FilesMatch>
```

Por último, añadimos un fichero .php en **/usr/local/www/apache24/data**

Y comprobamos en un navegador web que se muestre correctamente la query a nuestra tabla.



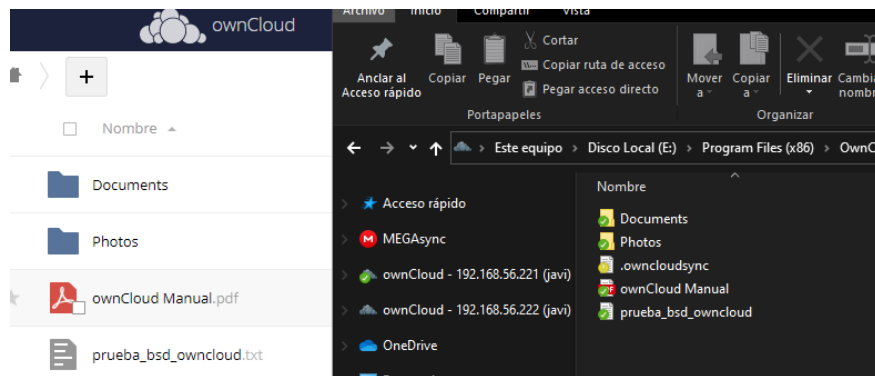
5.6 Git + ownCloud

Para realizar la instalación y configuración de ownCloud seguiremos los siguientes enlaces como guías:

<https://comoinstalar.me/como-instalar-famp-en-freebsd-12/>

<https://comoinstalar.me/como-instalar-owncloud-en-freebsd-12/>

Comprobamos con nuestro cliente:



Ahora pasaremos a instalar **git server**, para ello introducimos el siguiente comando:

```
$ sudo pkg install git
```

A continuación, crearemos un repositorio local en nuestro servidor.

```
$ mkdir ~/repo
```

```
$ cd ~/repo
```

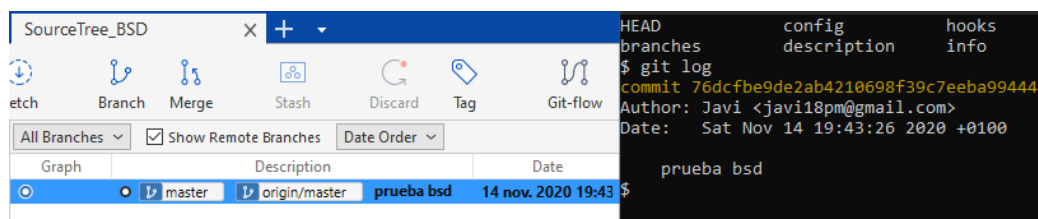
```
$ git init --bare --shared Project
```

Una vez creado el repositorio local, debemos habilitar el post-update hook copiando el fichero sample tal y como aparece a continuación:

```
$ cd ~/repo/project/hooks/
```

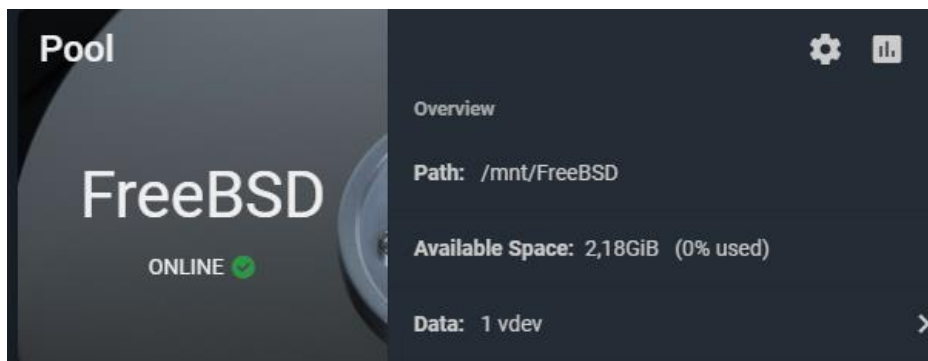
```
$ cp post-update.sample post-update
```

Por último, nos descargaremos un cliente para trabajar en nuestro host:



5.7 FreeNAS + iSCSI

Instalamos y configuramos una maquina FreeNAS.



Una vez instalado, en FreeBSD editamos el archivo rc.conf y añadimos iscsid_enable="YES".

Ahora, introducimos el siguiente comando y comprobamos que esté conectado el target:

```
root@freebsd:~ # iscsictl -A -p 192.168.26.7 -t iqn.2005-10.org.freenas.ctl:targetbsd
root@freebsd:~ # iscsictl
Target name          Target portal      State
iqn.2005-10.org.freenas.ctl:targetbsd 192.168.26.7      Connected: da0
```

```
pkg install gpart
```

```
gpart create -s gpt /dev/da0
```

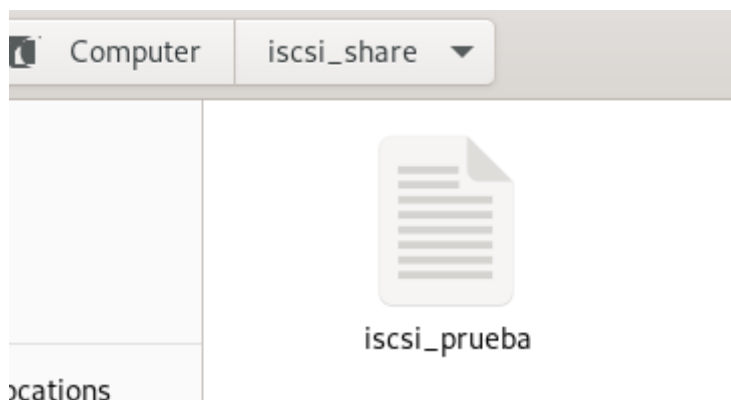
```
gpart add -t freebsd-ufs -l 1m /dev/da0
```

```
newfs -U /dev/da0p1
```

Por último, crearemos una carpeta donde montaremos el disco:

```
mkdir /iscsi_share $mount -t ufs -o rw /dev/da0p1 /iscsi_share
```

```
mount -t ufs -o rw /dev/da0p1 /iscsi_share/
```



6. Windows Server 2019

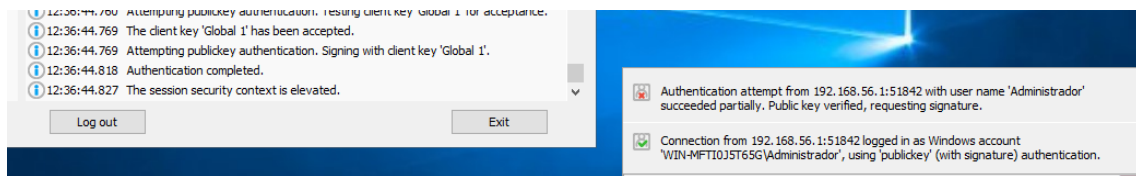
6.1 SSH, SFTP, SCP

Para configurar SSH en nuestro servidor, descargaremos e instalaremos Bitvise Server.

Una vez instalado, desde open easy settings, añadimos a nuestro usuario para poder acceder.

Una vez instalado, en nuestro cliente escribiremos la IP del servidor, el puerto y el método de entrada que en nuestro caso sería mediante clave pública:

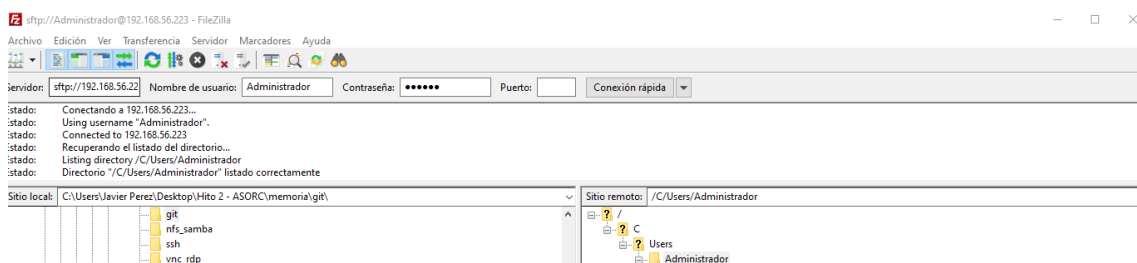
Desde el Client Key Manager añadimos dicha clave y la subimos al servidor.



Ahora ya podríamos entrar sin necesidad de introducir una contraseña.

Al ya tener instalado el servicio **SSH**, dicho servicio nos proporciona dos servicios más, uno de acceso a ficheros (**SFTP**) y otro que nos permite copiar archivos entre equipos de forma segura (**SCP**).

Para comprobar el servicio SFTP he usado el cliente FileZilla:



Para comprobar el servicio SCP simplemente usando el comando siguiente:

scp -p <puerto> <ruta origen archivo> user@ip:<ruta destino archivo>

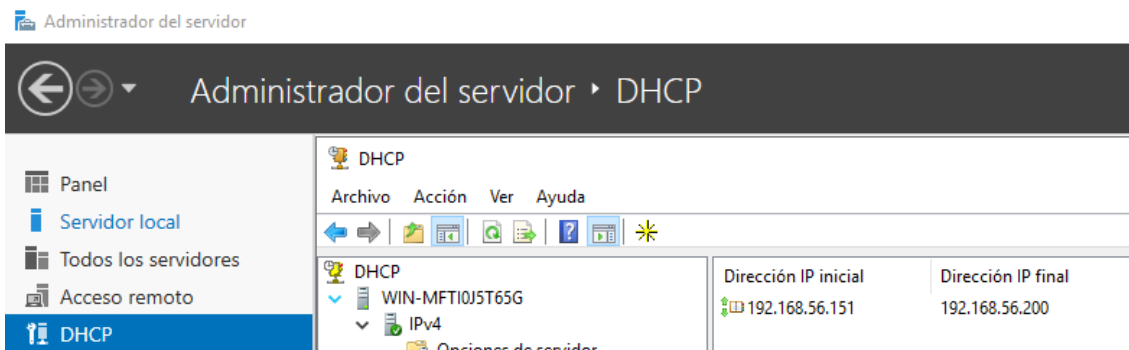
```
PS C:\Users\Javier Perez\Desktop> scp -p .\pruebaSCP.txt Administrador@192.168.56.223:C:\Users\Administrador\Desktop
pruebaSCP.txt
PS C:\Users\Javier Perez\Desktop>
```



Como vemos se ha copiado correctamente el archivo '*pruebaSCP.txt*'.

6.2 DHCP

Para instalar el servicio de DHCP, configuraremos todo el proceso desde la herramienta Administrador del servidor.



Para comprobar el servicio, iniciaremos otra máquina virtual y comprobaremos que se le ha asignado a dicha máquina una dirección IP dentro del rango de direcciones previamente establecido.

```
javi@javi-VirtualBox: ~
Archivo  Editar  Ver  Buscar  Terminal  Ayuda

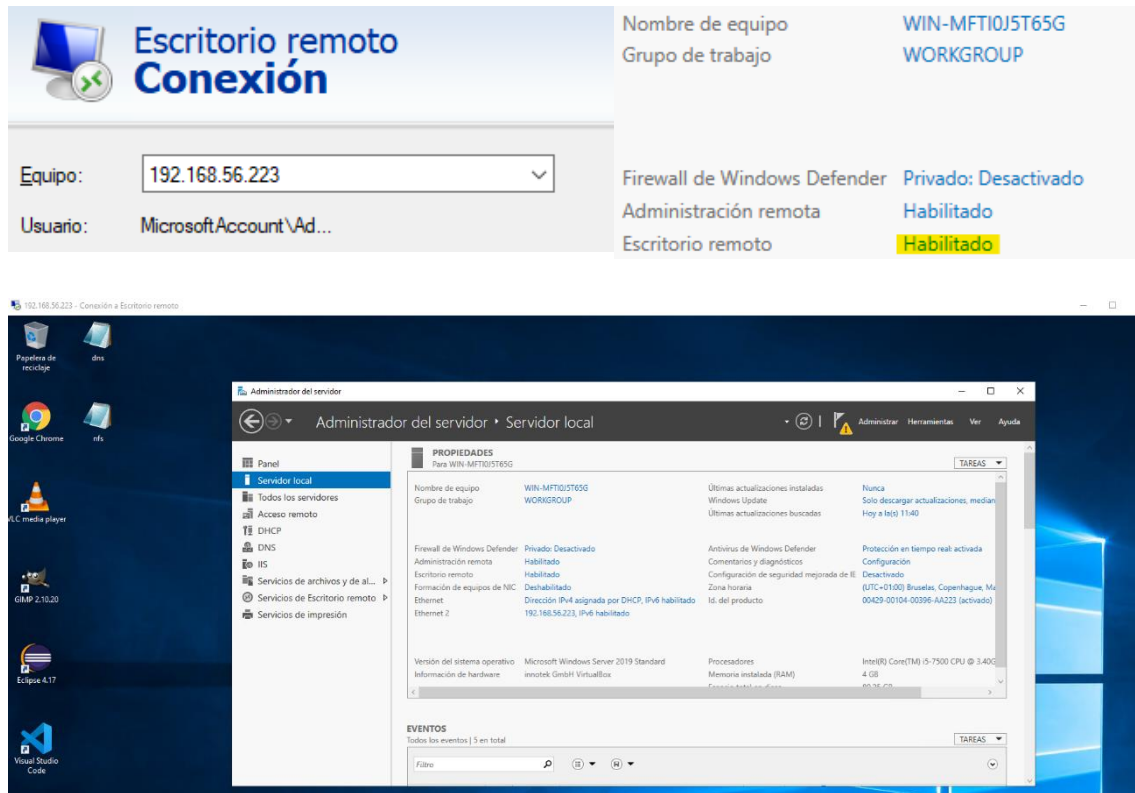
RX packets 93  bytes 12284 (12.2 KB)
RX errors 0  dropped 0  overruns 0  frame 0
TX packets 164  bytes 16955 (16.9 KB)
TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 192.168.56.151  netmask 255.255.255.0  broadcast 192.168.56.255
    inet6 fe80::3b54:559b:20d4:d4e6  prefixlen 64  scopeid 0x20<link>
    ether 08:00:27:71:5d:09  txqueuelen 1000  (Ethernet)
RX packets 57  bytes 6473 (6.4 KB)
RX errors 0  dropped 0  overruns 0  frame 0
```

Como se puede observar, se ha establecido una dirección IP dentro del rango de direcciones de nuestro servidor DHCP.

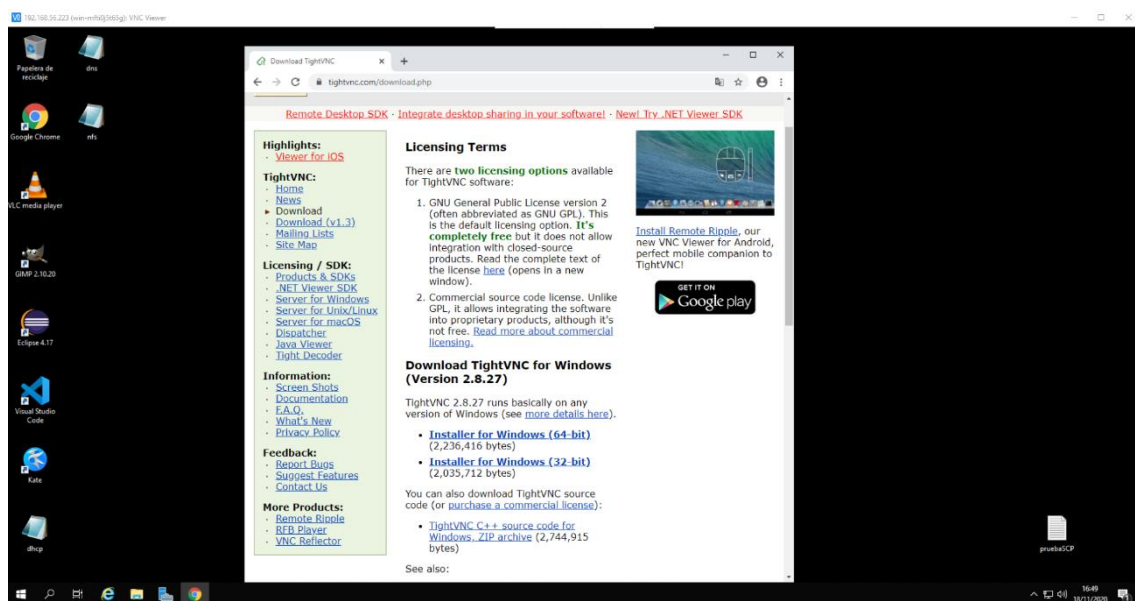
6.3 VNC + RDP

Para instalar RDP, instalamos el servicio desde el Administrador de servidores.



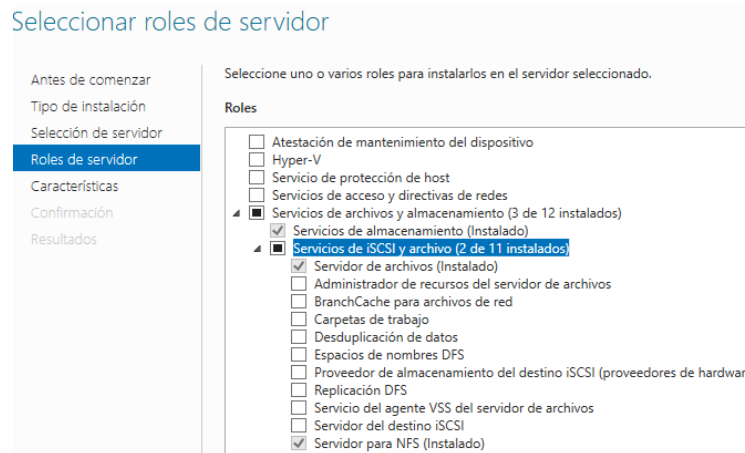
Para instalar VNC, instalaremos un servidor VNC, en nuestro caso TightVNC. La configuración ya viene instalada y funcionando correctamente.

Para probarlo, nos conectamos con nuestro cliente VNC Viewer:

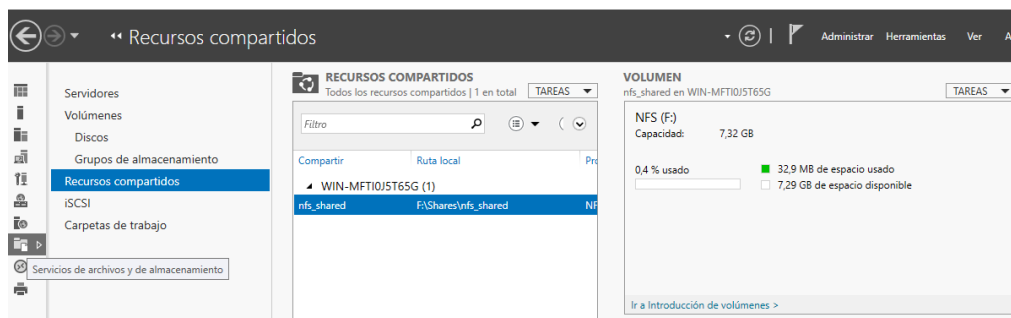


6.4 NFS / SAMBA

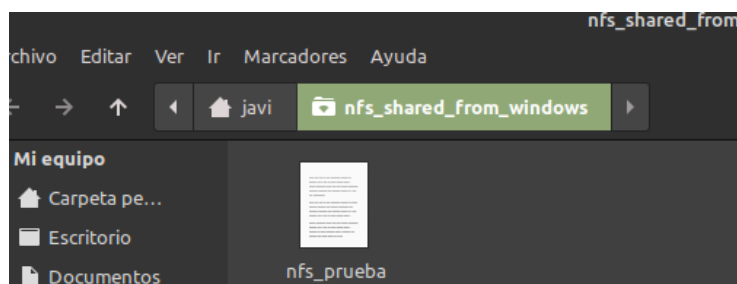
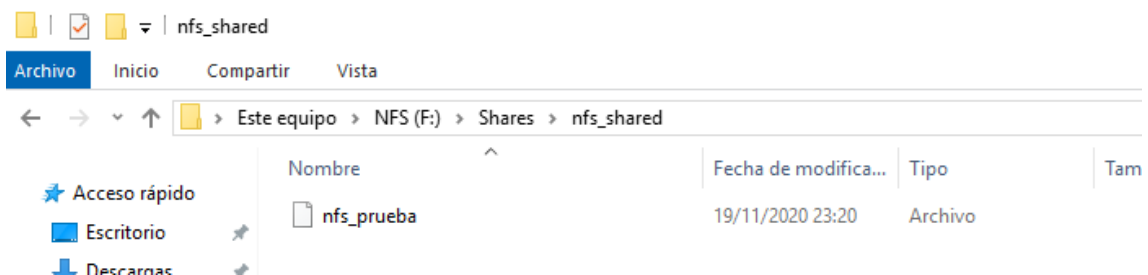
Para instalar el servicio de NFS, lo instalaremos desde el Administrador del servidor.



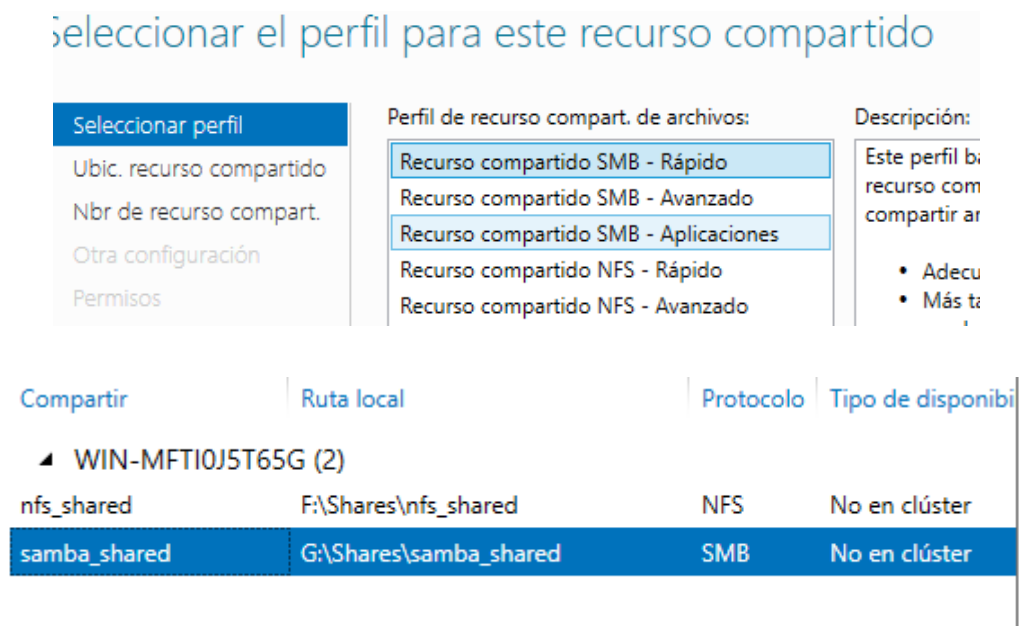
Una vez instalado, configuramos la carpeta compartida.



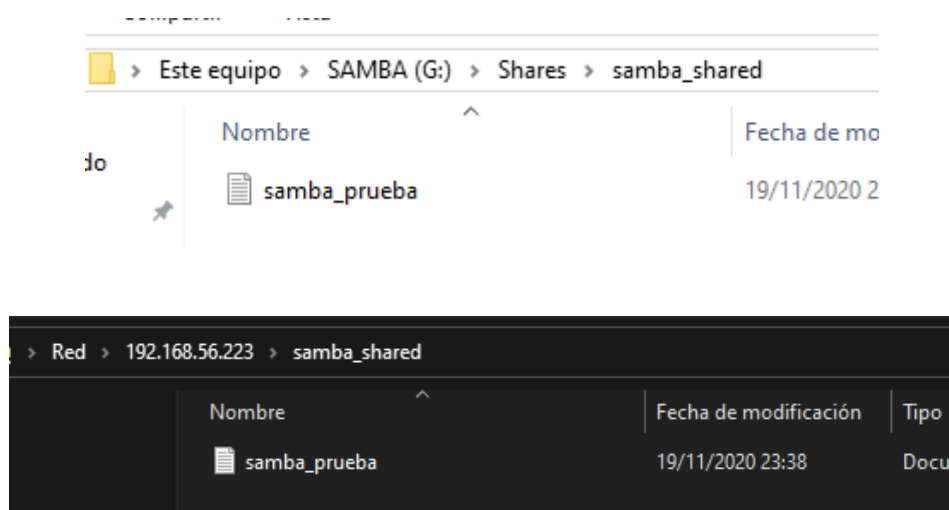
Para comprobarlo, he creado un archivo *nfs_prueba* en la carpeta compartida.



Para instalar el servicio de SAMBA, lo instalaremos igual que NFS.

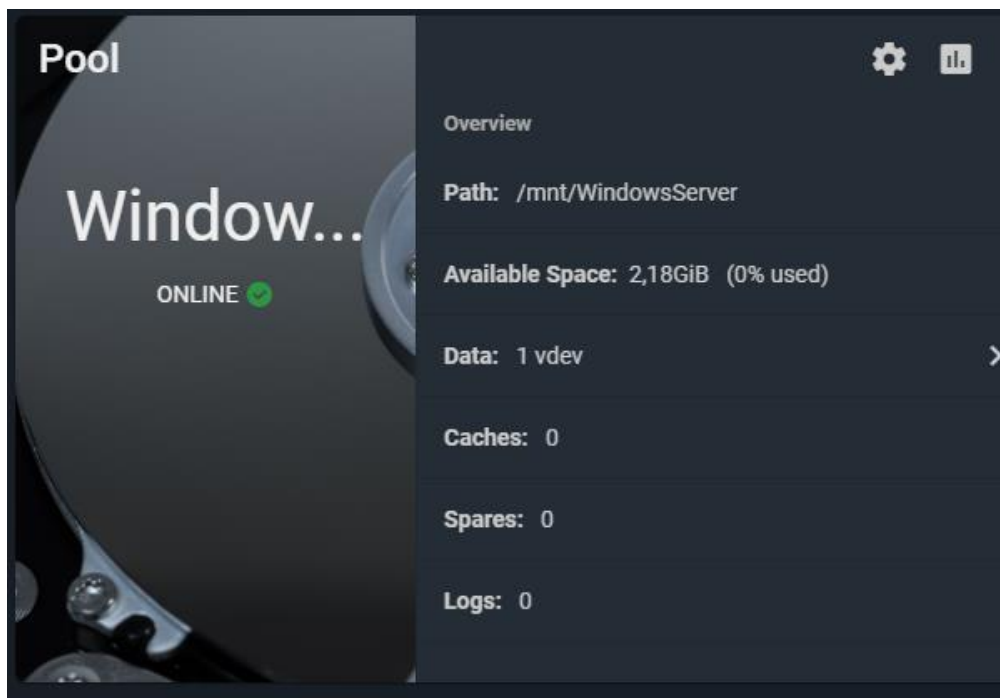


En nuestro cliente, Win + R y escribimos [\\IP-Servidor](#).



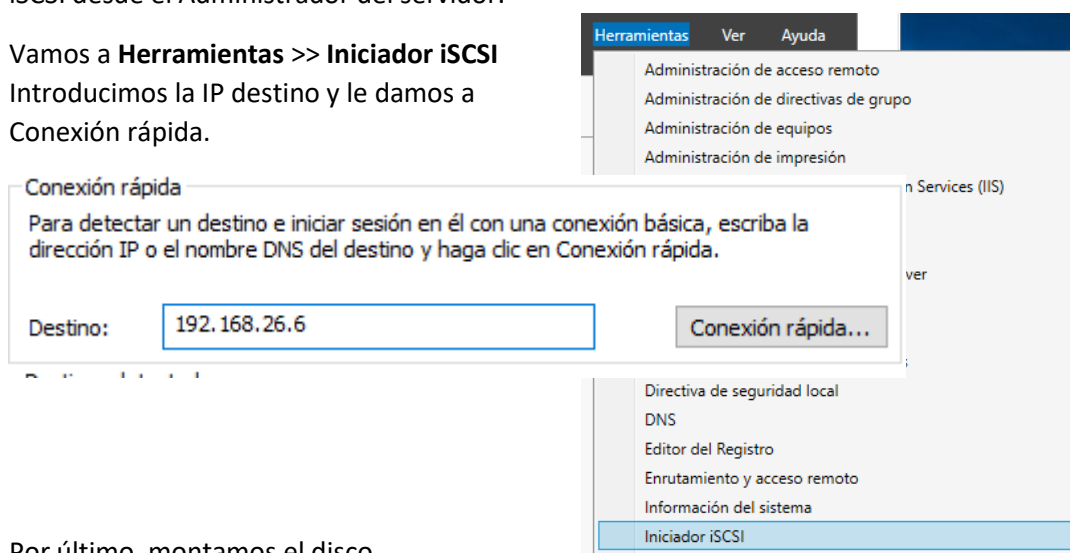
6.5 FreeNAS + iSCSI

Para instalar el servicio iSCSI con FreeNAS, debemos instalar primero la máquina virtual FreeNAS y configurar desde el navegador todos los campos para el correcto funcionamiento del recurso compartido de disco con iSCSI.



Una vez instalado y configurado, en nuestro servidor, Windows Server, pasaremos a configurar iSCSI desde el Administrador del servidor.

Vamos a **Herramientas >> Iniciador iSCSI**
Introducimos la IP destino y le damos a Conexión rápida.

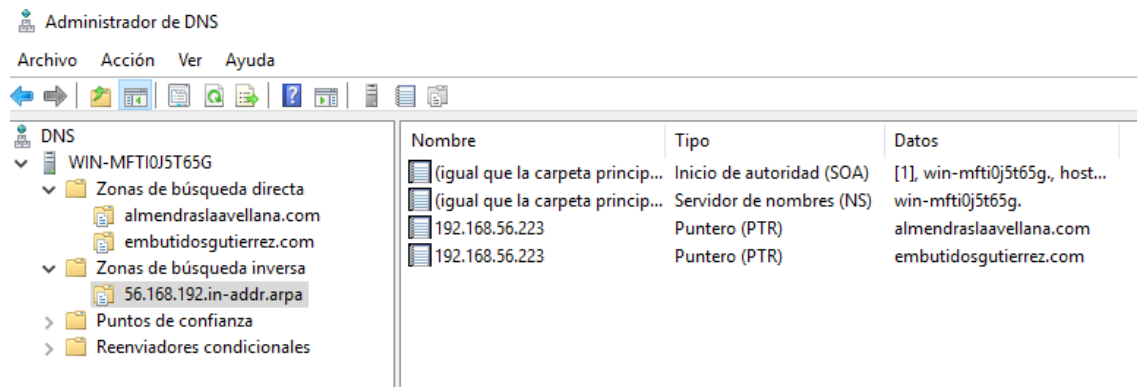


Por último, montamos el disco.



6.6 DNS

Para instalar DNS, instalamos el servicio desde el Administrador del servidor y configuramos las zonas pertinentes.



Ejecutamos nslookup y comprobamos que todo funciona correctamente.

```
C:\Users\Administrador>nslookup embutidosgutierrez.com
Servidor: almendraslaavellana.com
Address: 192.168.56.223

Nombre: embutidosgutierrez.com

C:\Users\Administrador>nslookup 192.168.56.223
Servidor: embutidosgutierrez.com
Address: 192.168.56.223

Nombre: almendraslaavellana.com
Address: 192.168.56.223
```