**ADMINISTRACIÓN DE SISTEMAS OPERATIVOS Y REDES DE COMPUTADORES**

UNIVERSIDAD DE ALICANTE | GRADO EN INGENIERÍA INFORMÁTICA

Francisco Javier Pérez Martínez

74384305m – gRUPO 01

**HITO 2**

Contenido

[1. Licencias 2](#_Toc57047368)

[2. Particionado 3](#_Toc57047369)

[3. Arranque y parada de servicios 5](#_Toc57047370)

[4. CentOS 5](#_Toc57047371)

[4.1 SSH, SFTP, SCP 5](#_Toc57047372)

[4.2 VNC + RDP 7](#_Toc57047373)

[4.3 BD Server 9](#_Toc57047374)

[4.4 Web Server 11](#_Toc57047375)

[4.5 NFS / SAMBA 12](#_Toc57047376)

[4.6 DHCP 15](#_Toc57047377)

[4.7 CUPS 16](#_Toc57047378)

[4.8 FreeNAS + iSCSI 17](#_Toc57047379)

[4.9 Git + OwnCloud 18](#_Toc57047380)

[4.10 DNS 21](#_Toc57047381)

[5. FreeBSD 23](#_Toc57047382)

[5.1 SSH, SFTP, SCP 23](#_Toc57047383)

[5.2 DHCP 25](#_Toc57047384)

[5.3 VNC + RDP 26](#_Toc57047385)

[5.4 CUPS 27](#_Toc57047386)

[5.5 BD Server 28](#_Toc57047387)

[5.6 Git + ownCloud 29](#_Toc57047388)

[5.7 FreeNAS + iSCSI 30](#_Toc57047389)

[6. Windows Server 2019 31](#_Toc57047390)

[6.1 SSH, SFTP, SCP 31](#_Toc57047391)

[6.2 DHCP 32](#_Toc57047392)

[6.3 VNC + RDP 33](#_Toc57047393)

[6.4 NFS / SAMBA 34](#_Toc57047394)

[6.5 FreeNAS + iSCSI 36](#_Toc57047395)

[6.6 DNS 37](#_Toc57047396)

# Licencias

Una **licencia de software** es un contrato entre el licenciante (autor/titular de los derechos de explotación/distribución) y el licenciatario (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 pasaré 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 fileserver.
* 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 compañia 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** (**G**nu **P**ublic **L**icense) con todas sus variantes y la **BSD** (**B**erkeley **S**oftware **D**istribution). 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.

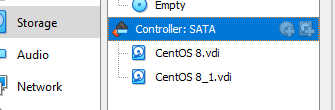
OJO, con esto no digo que la GPL sea mala licencia, considero que es una buena licencia ya que nos da más libertad que las privadas, pero la BSD es el verdadero software libre ya que nos lo permite todo.

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.

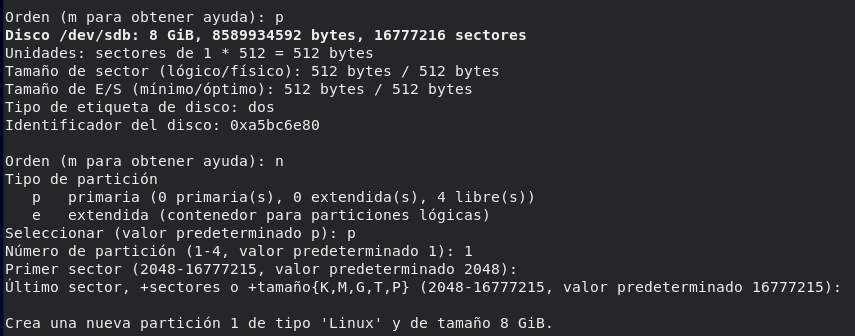
# Particionado

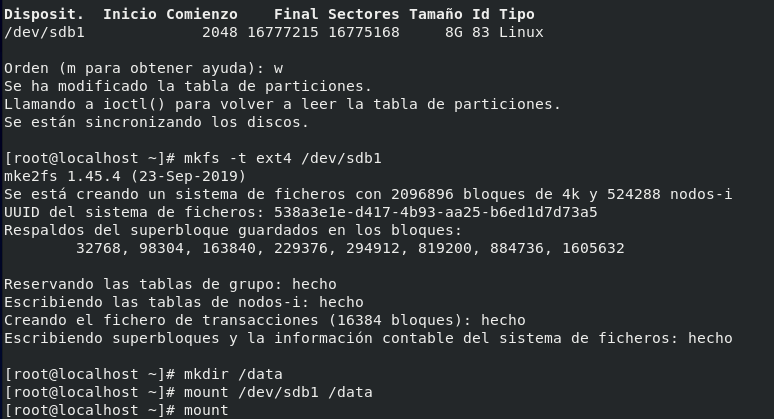
Para crear particiones en cada uno de los sistemas operativos introducimos un nuevo disco duro mediante la configuración de virtual box.



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:



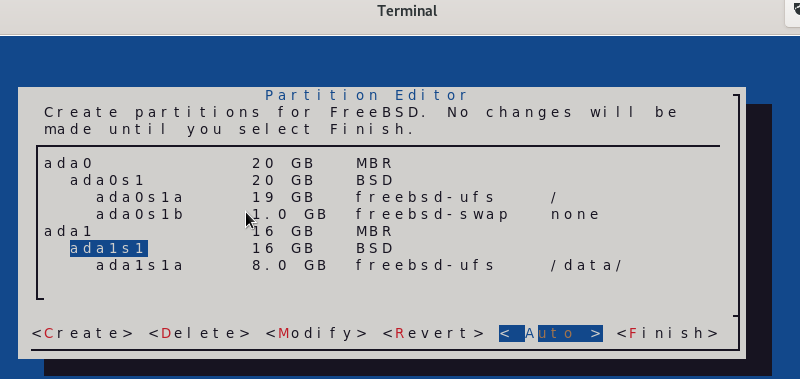


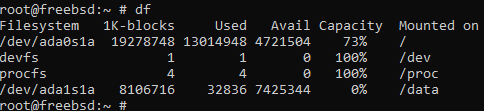


Como podemos observar se ha montado nuestra partición en la carpeta /data  
Para saber que significa cada comando escribir m para obtener ayuda.

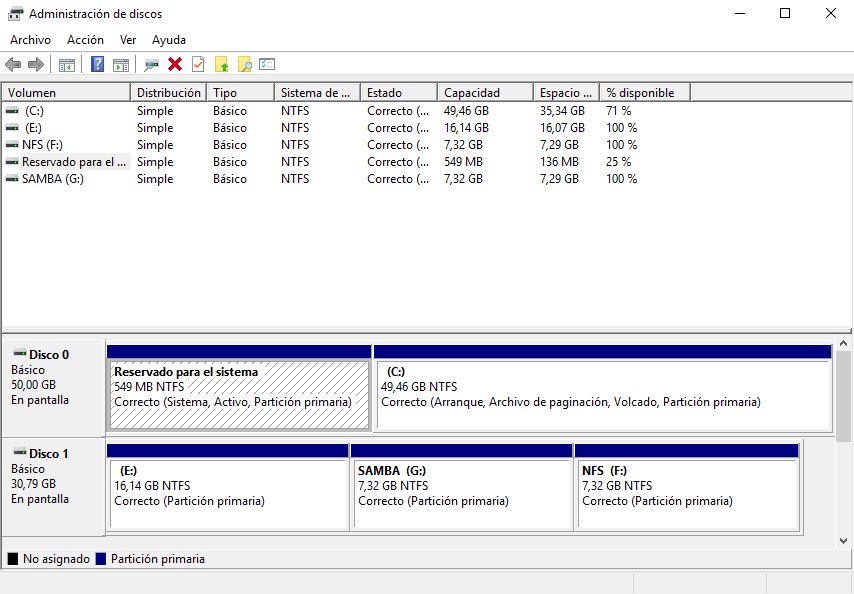
Por último, guardar en archivo **/etc/fstab** la entrada que hace referencia al disco para que se monte durante el arranque de sistema.

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





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



# 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>**

# CentOS

## 

## 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**

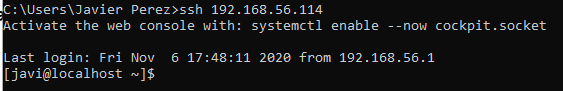
A continuación, genero las claves (pública y privada) en el (host) con   
**ssh-keygen -t rsa**. Una vez generadas, éstas estarán guardadas en la carpeta .ssh/

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"**

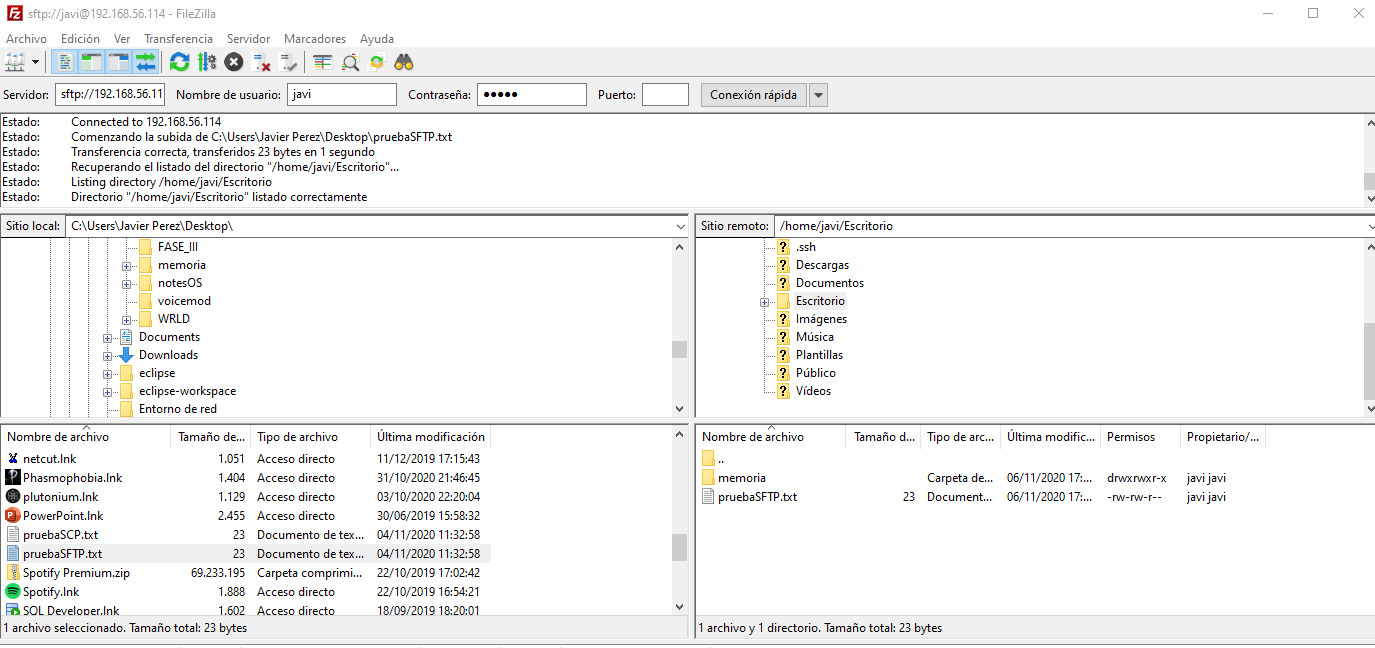
Muy importante darles permisos a las carpetas .ssh/ (chmod 700) y al archivo authorized\_keys (chmod 600) situadas en el servidor.

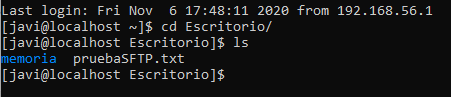
Ahora ya podremos conectarnos sin necesidad de introducir una contraseña gracias a la clave pública generada anteriormente.



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**).

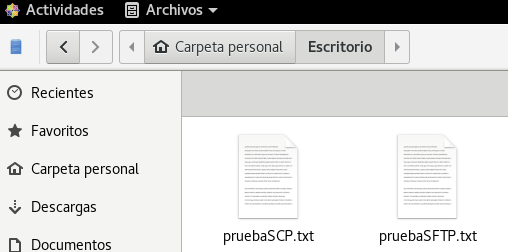
Para comprobar el servicio SFTP he usado el cliente FileZilla, para ello he creado un archivo llamado ‘*pruebaSFTP.txt’ y* lo enviamos al servidor para verificar que todo funciona correctamente.





Para comprobar el servicio SCP simplemente usando el comando siguiente:

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

****



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

## 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:

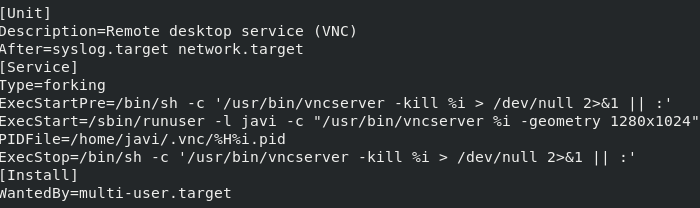
**dnf install tigervnc-server tigervnc-server-module -y**

configuramos la contraseña:

**vncpasswd**

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

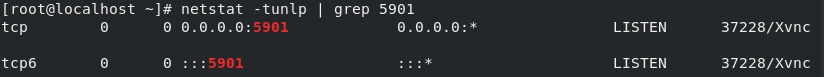
**nano /etc/systemd/system/vncserver@.service**

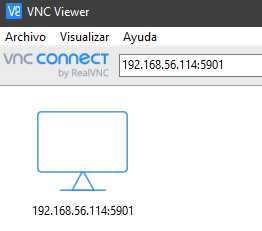
****

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**

****

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

**192.168.56.107:5901**

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

**dnf install epel-release**

**dnf install xrdp**

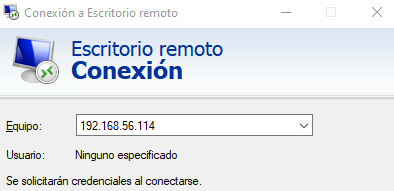
A continuación, habilitamos el puerto 3389 por si no está permitido en el firewall.

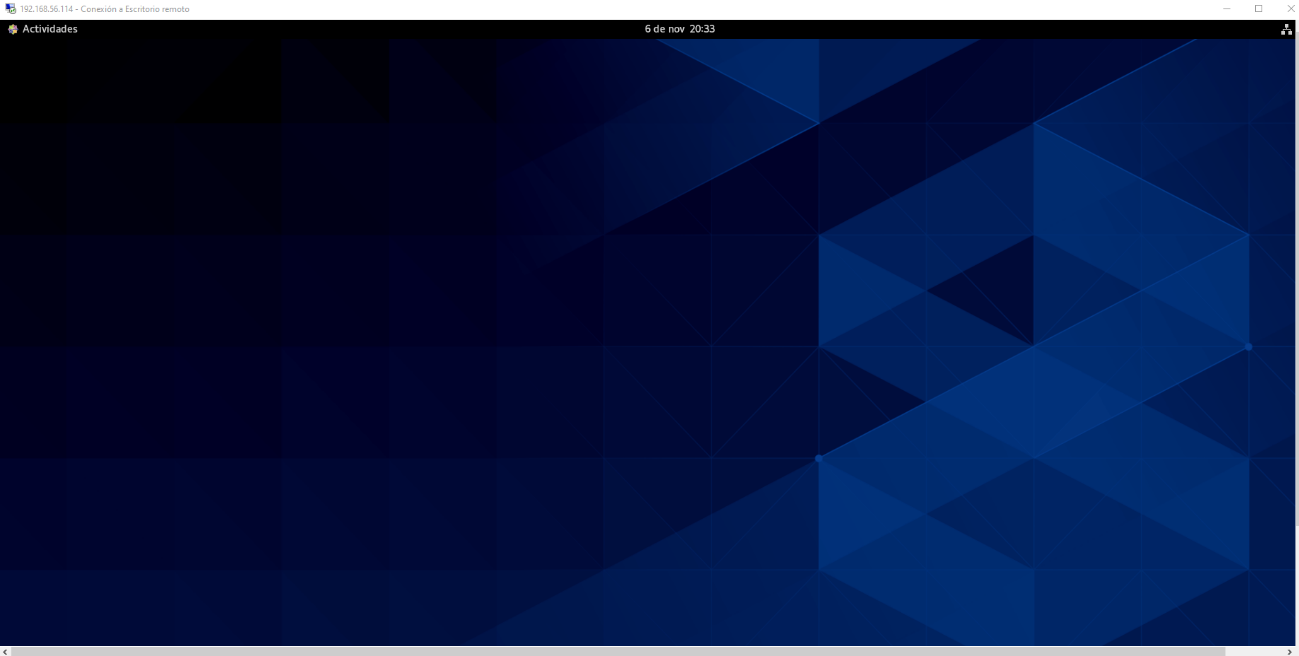
**firewall-cmd –-permanent –-add-port=3389/tcp**

**firewall-cmd --reload**



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





## 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.

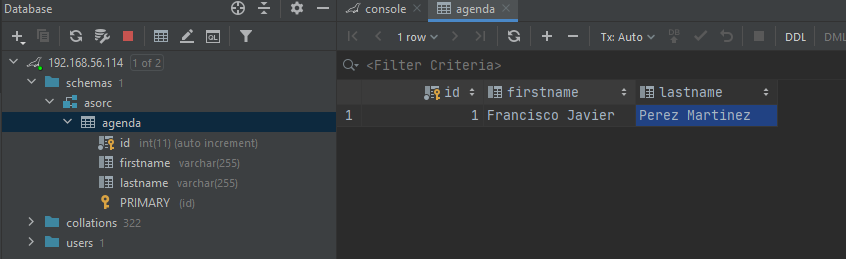
**mysql\_secure\_installation** 🡪 para realizar la instalación de forma segura.

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

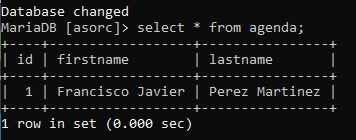
Creamos la base de datos con **create database asorc** y un usuario y contraseña que gestionara la BD con **grant all privileges on asorc.\* to asorcuser@’%’ identified by ‘password’**.

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:



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

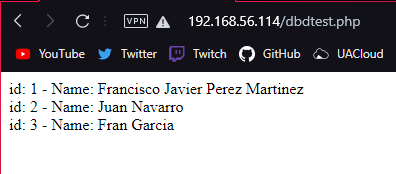
Instalaremos Apache, el cual tiene una característica y es que nos permite validar la configuración antes de ejecutar apache. Para ello, ejecutamos **apachectl configtest**.

Nos saldrá que debemos cambiar el ServerName, entramos en archivo de configuración de httpd **/etc/httpd/conf/httpd.confg** 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:

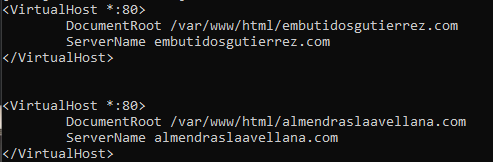
**192.168.56.114/dbdtest.php**



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.

## 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:

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>**

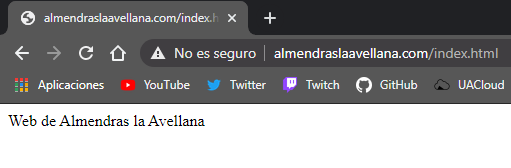
A continuación, tenemos dos opciones utilizar el dns (crear el domino) o utilizar el fichero hosts, en nuestro caso nos dirigimos a dicho fichero situado en la siguiente ruta:

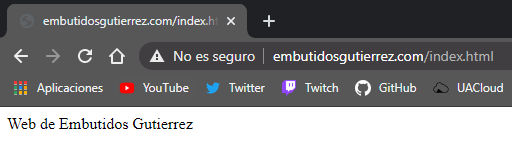
**C:\Windows\System32\drivers\etc\hosts**

E introducimos las entradas siguientes:

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

**systemctl restart httpd**

****

****

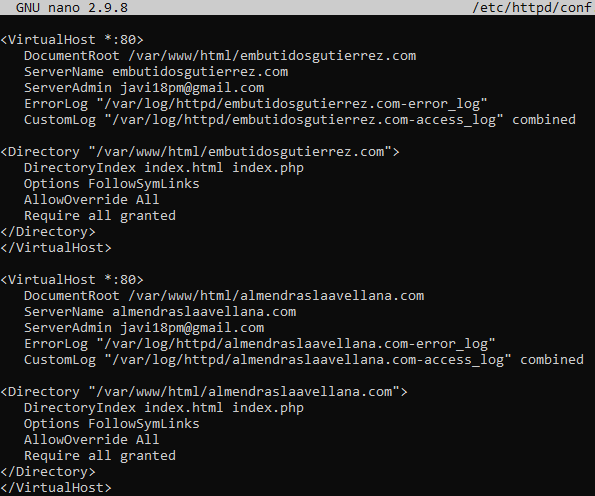
Ahora descargaremos Joomla para que nuestro dominio virtual ejecute un gestor de contenido, en este caso Joomla.

**wget** [**https://downloads.joomla.org/cms/joomla3/3-9-22/Joomla\_3-9-22-Stable-Full\_Package.zip?format=zip**](https://downloads.joomla.org/cms/joomla3/3-9-22/Joomla_3-9-22-Stable-Full_Package.zip?format=zip)

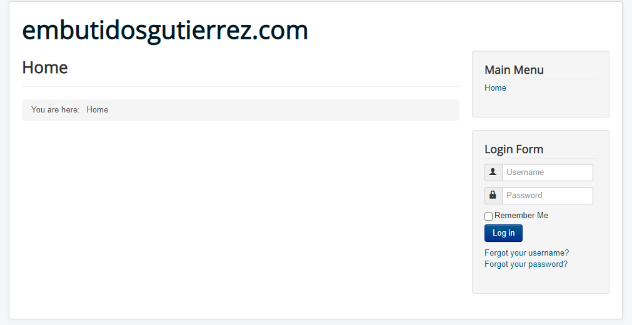
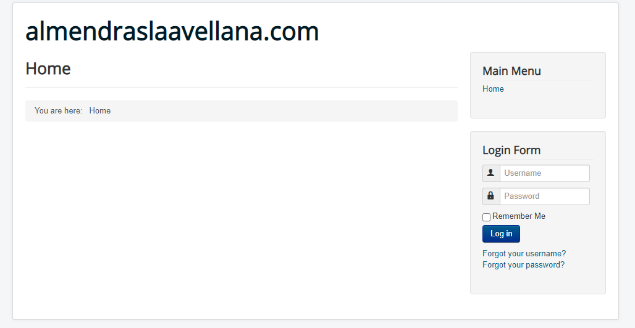
Descomprimimos en la carpeta del dominio y como lo va a ejecutar apache vamos a cambiar el contenido de root a apache con el comando **chown -R apache.apache \*.**

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**



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



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

## 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).

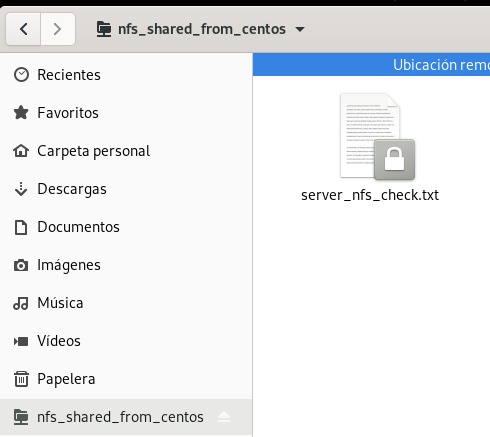
Nos crearemos una carpeta **mkdir nfs\_shared\_from\_centos** y con el comando

**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:

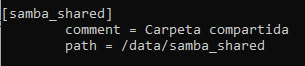
**dnf -y install samba samba-client samba-common**

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:



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 nmb smb.**

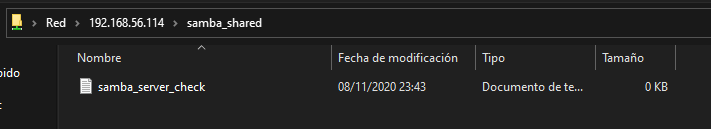
Por último, abrimos los puertos de samba:

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

**$ sudo firewall-cmd --reload**

Con las teclas Win + R escribimos [\\IP-servidor-samba](file:///\\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.

## 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**

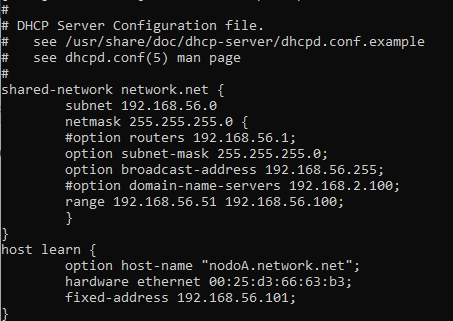
Ahora reiniciamos la red con los comandos**: nmcli con reload** y **nmcli con up enp0s8**

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

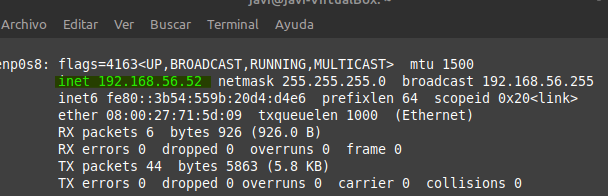
Ahora desactivamos el servidor dhcp de virtualbox y reiniciamos la máquina virtual.

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

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



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.



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

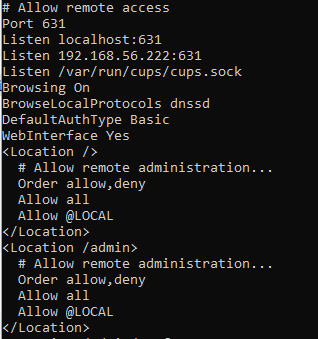
## CUPS

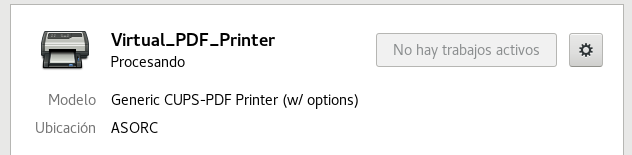
Para instalar el servicio CUPS seguimos los siguientes pasos:



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

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





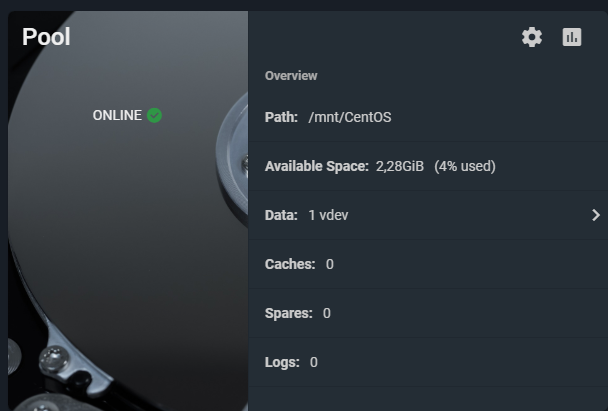
Y desde un navegador web accedemos al servicio de impresoras CUPS y agregamos la impresora PDF en nuestro host.

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

**http://192.168.56.222:631/printers/{nombre\_impresora}**

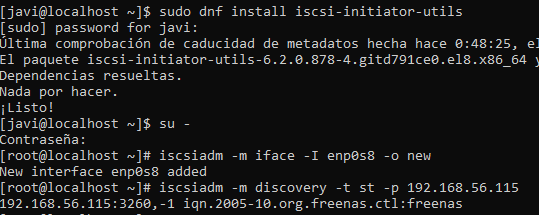
## FreeNAS + iSCSI

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.

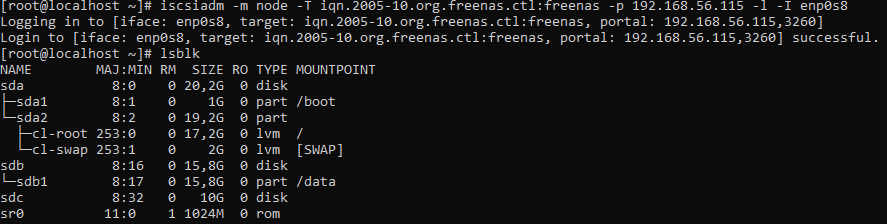


Una vez configurado FreeNAS, procedemos a instalar iSCSI en CentOS:

**$ dnf install iscsi-initiator-utils**

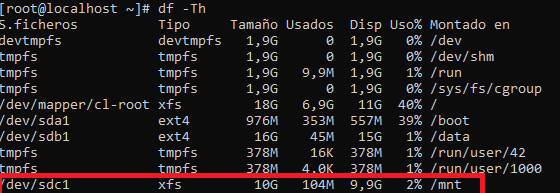
Una vez instalado realizamos los siguientes comandos:

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



Ahora, después de configurar la red con iSCSI, montamos el disco como se ha explicado en el apartado de [particiones.](#_Particionado)

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



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

## 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**

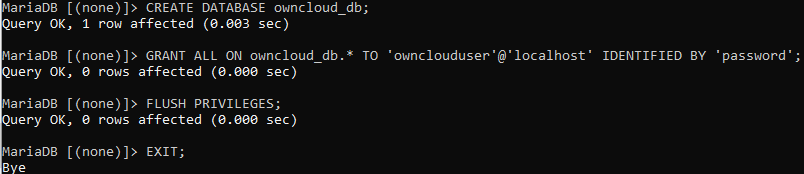
**systemctl start mariadb**

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.

A continuación, descargamos ownCloud en nuestro sistema:

**sudo wget** [**https://download.owncloud.org/community/owncloud-10.3.2.tar.bz2**](https://download.owncloud.org/community/owncloud-10.3.2.tar.bz2)

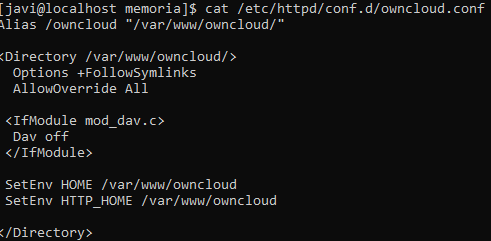
Y lo extraemos en la carpeta **/var/www/**

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

Configuramos los permisos de Apache en dicho directorio:

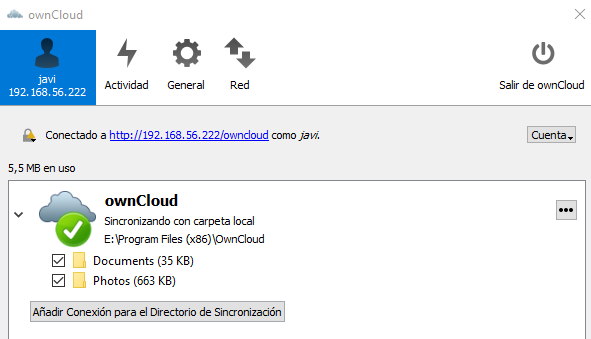
**sudo chown -R apache: /var/www/owncloud**

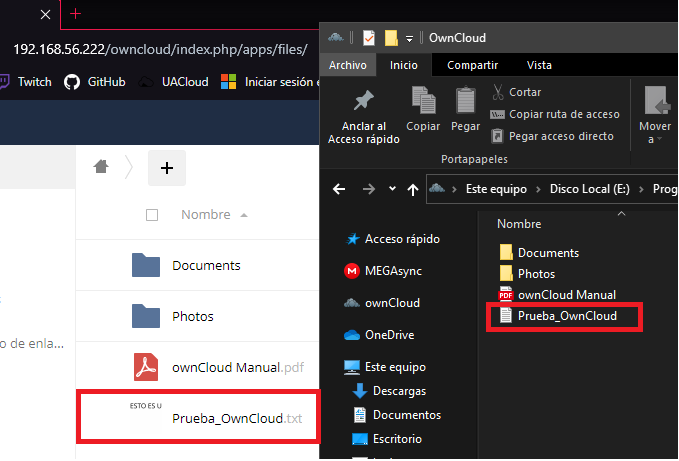
Creamos un fichero de configuración de Apaceh para el acceso de ownCloud: **sudo nano /etc/httpd/conf.d/owncloud.conf** y añadimos las siguientes líneas:

****

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

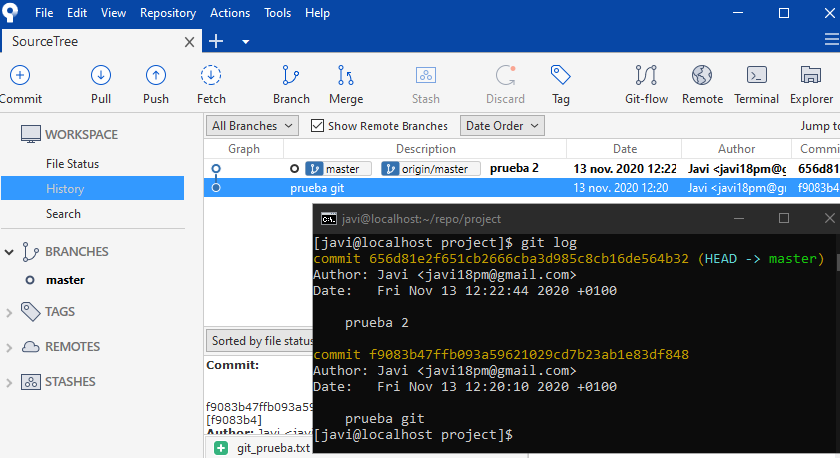
$ git init --bare --shared Project

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:

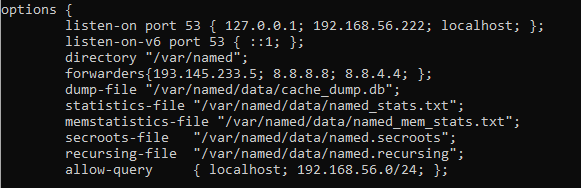
****

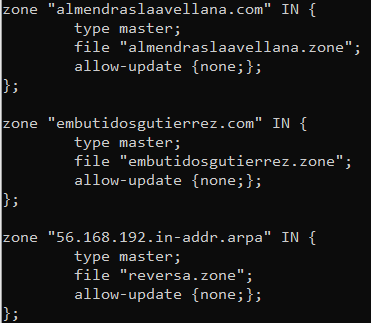
## DNS

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.

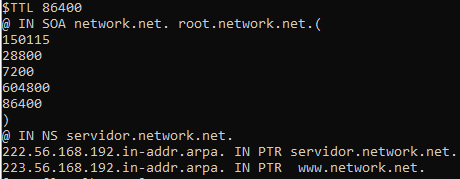




A continuación, creamos los archivos de configuración de las zonas creadas:

**nano /var/named/almendraslaavellana.zone**

**nano /var/named/embutidosgutierrez.zone**



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**

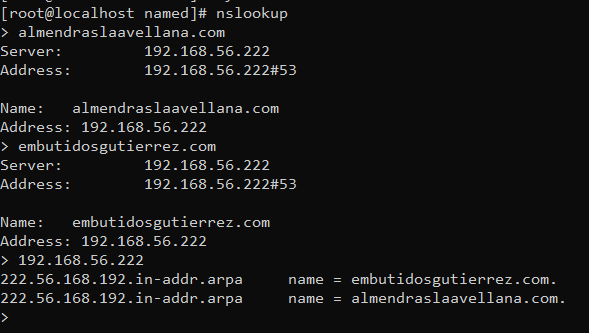
Añadimos el servicio DNS al firewall:

**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**



# FreeBSD

## 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**

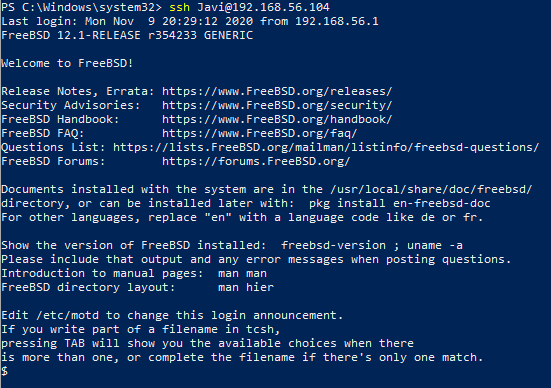
**UsePAM yes**

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:

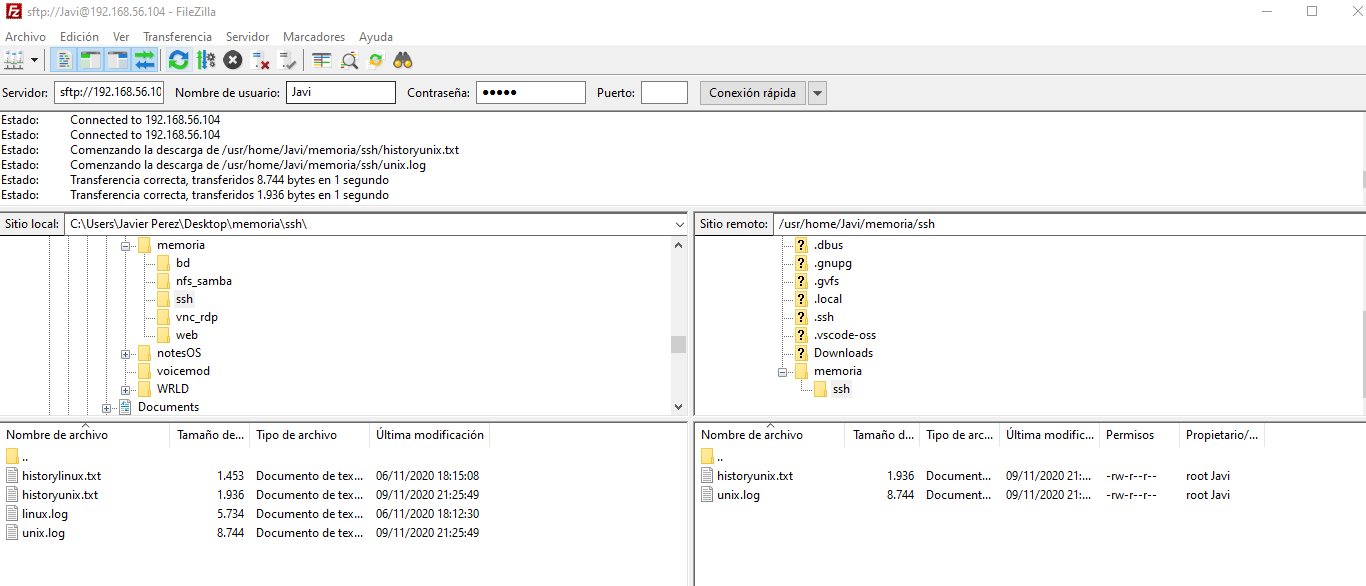
**cat ~/.ssh/id\_rsa.pub | ssh Javi@192.168.56.104 “cat >> ~/.ssh/authorized\_keys"**

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



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**).

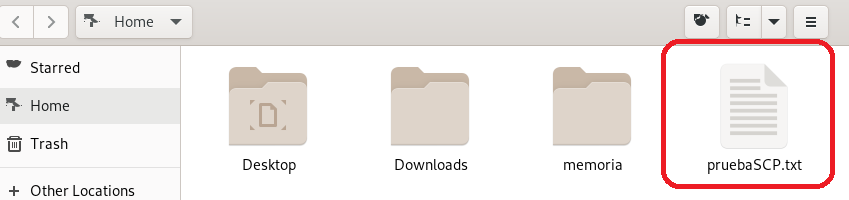
Para comprobar el servicio SFTP he usado el cliente FileZilla, para ello he enviado los logs del servicio de ssh y el history a una carpeta en mi host para verificar que todo funciona correctamente.



Para comprobar el servicio SCP simplemente usando el comando siguiente:

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





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

## DHCP

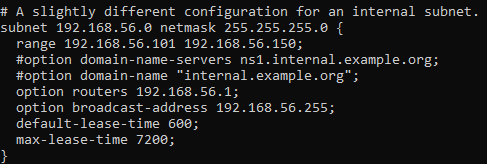
Antes de instalar el servidor dhcp debemos configurar la red para que aparezca como en el enunciado de la práctica. Editamos el archivo **nano /etc/rc.conf** e introducimos:

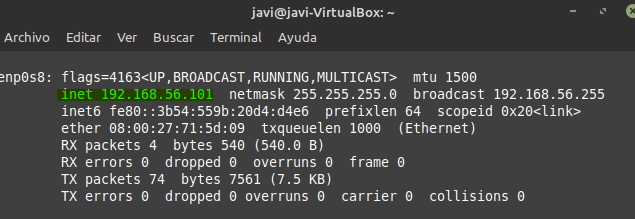
**ifconfig\_em1="inet 192.168.56.221 netmask 255.255.255.0"**

**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**

****

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.

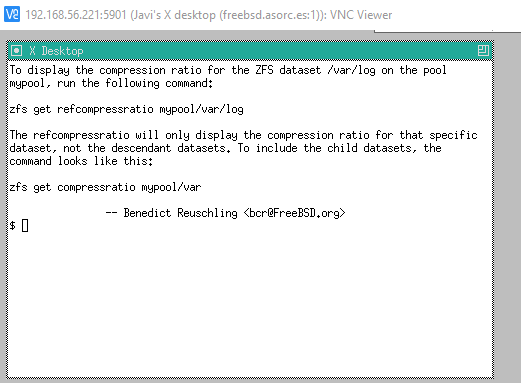
Como se puede observar, el servidor le ha asignado la dirección IP 192.168.56.101, IP disponible dentro del rango configurado.

## VNC + RDP

Para instalar el servicio de VNC utilizamos el siguiente comando:

**pkg install tightvnc**

**vncserver**

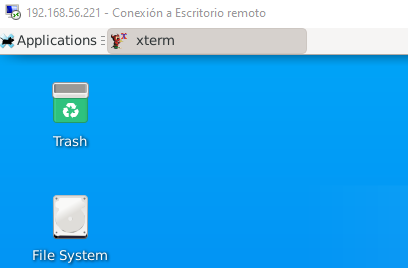
Ahora en nuestro cliente, comprobamos conectándonos al servidor.

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.



Iniciamos el servicio **service xrdp start**

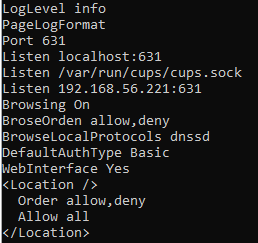
****

## 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:



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

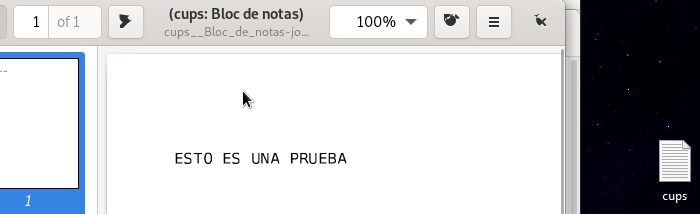


Iniciamos el servicio: **service cupsd start.**

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}**



## BD Server

**PostgreSQL**

Para el servicio de base de datos usaremos PostgreSQL para FreeBSD, instalaremos el paquete de postgre 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:

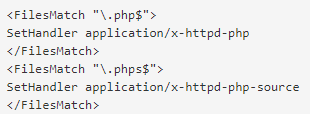
**service postgresql start  
service postgresql status**

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

**sudo -u postgres createuser --interactive -P**

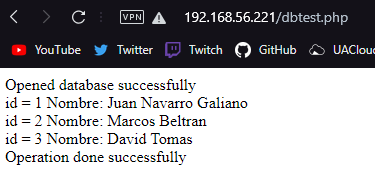
**createdb Javi -O Javi**

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**

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.

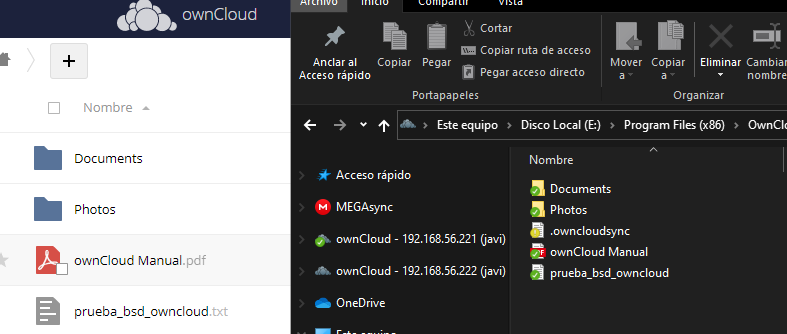


## Git + ownCloud

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

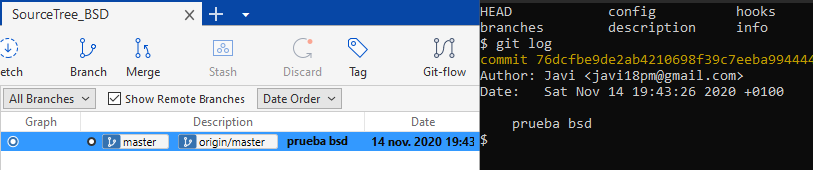
$ git init --bare --shared Project

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/

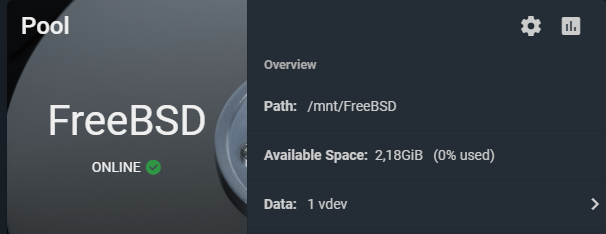
$ cp post-update.sample post-update

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



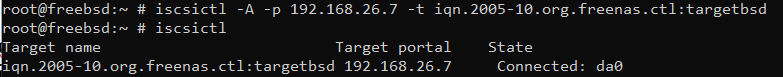
## FreeNAS + iSCSI

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:



**pkg install gpart**

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

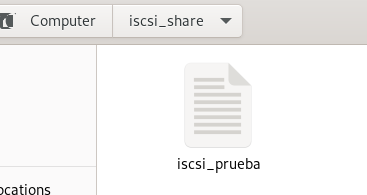
**gpart add -t freebsd-ufs -l 1m /dev/da0**

**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/**



# Windows Server 2019

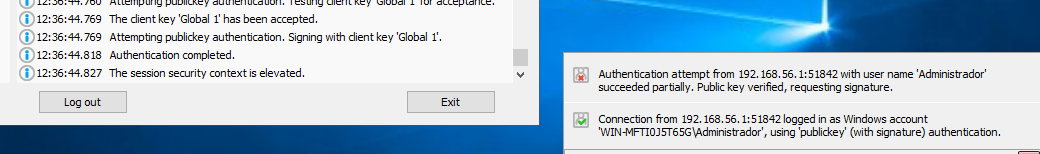
## SSH, SFTP, SCP

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:

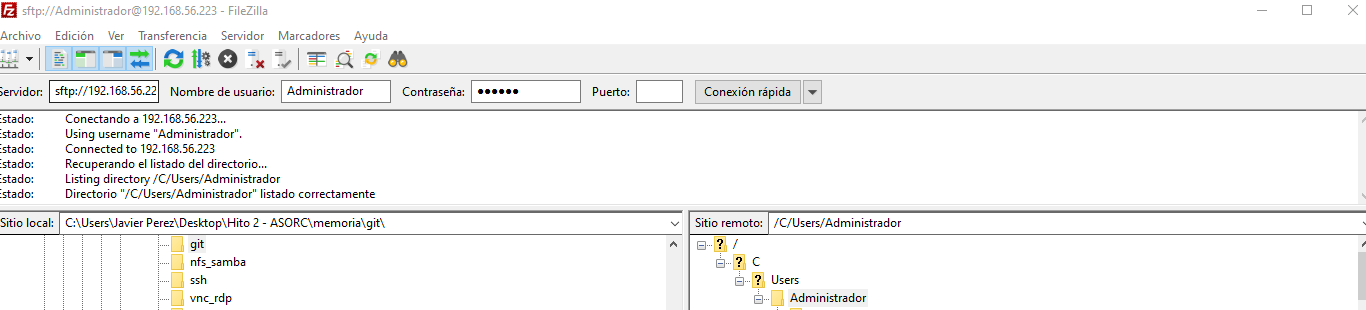
Desde el Client Key Manager añadimos dicha clave y la subimos al servidor.



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**).

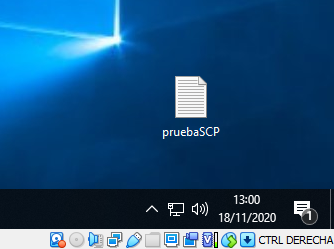
Para comprobar el servicio SFTP he usado el cliente FileZilla:



Para comprobar el servicio SCP simplemente usando el comando siguiente:

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

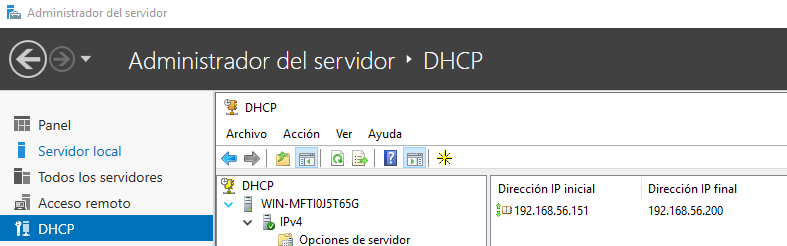




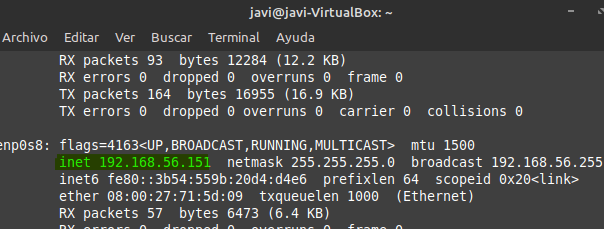
Como vemos se ha copiado correctamente el archivo ‘*pruebaSCP.txt’.*

## DHCP

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



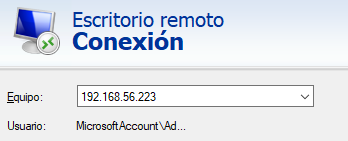
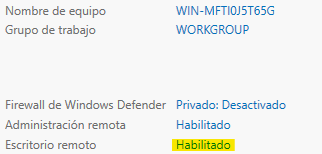
Para comprobar el servicio, iniciaremos otra máquina virtual y comprobaremos que se le ha asignado a dicha máquina una dirección IP dentro del rango de direcciones previamente establecido.

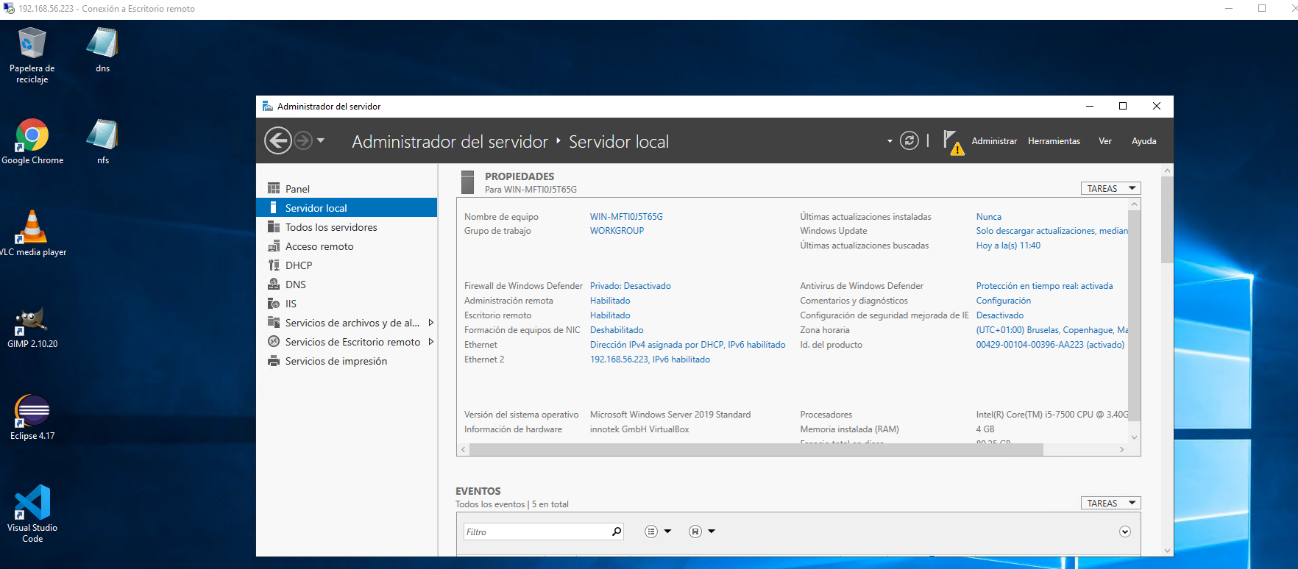


Como se puede observar, se ha establecido una dirección IP dentro del rango de direcciones de nuestro servidor DHCP.

## VNC + RDP

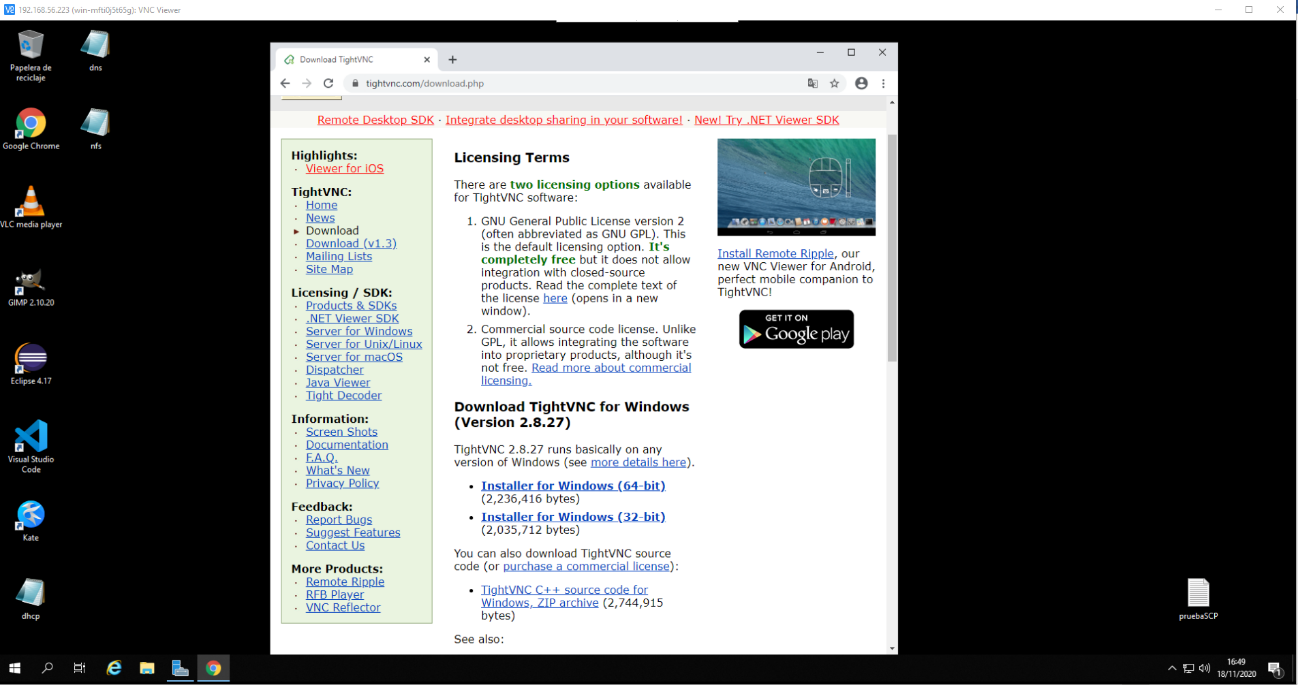
Para instalar RDP, instalamos el servicio desde el Administrador de servidores.



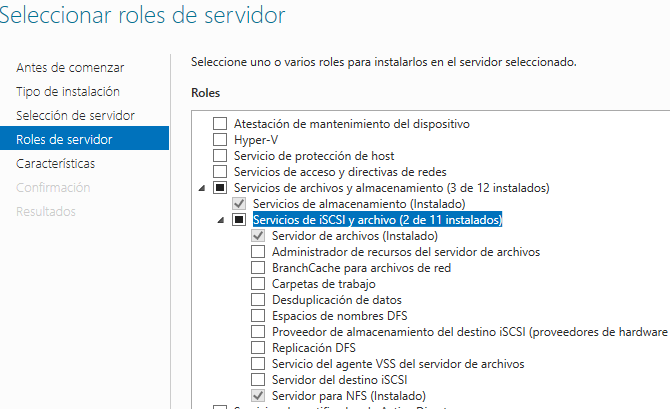


Para instalar VNC, instalaremos un servidor VNC, en nuestro caso TightVNC. La configuración ya viene instalada y funcionando correctamente.

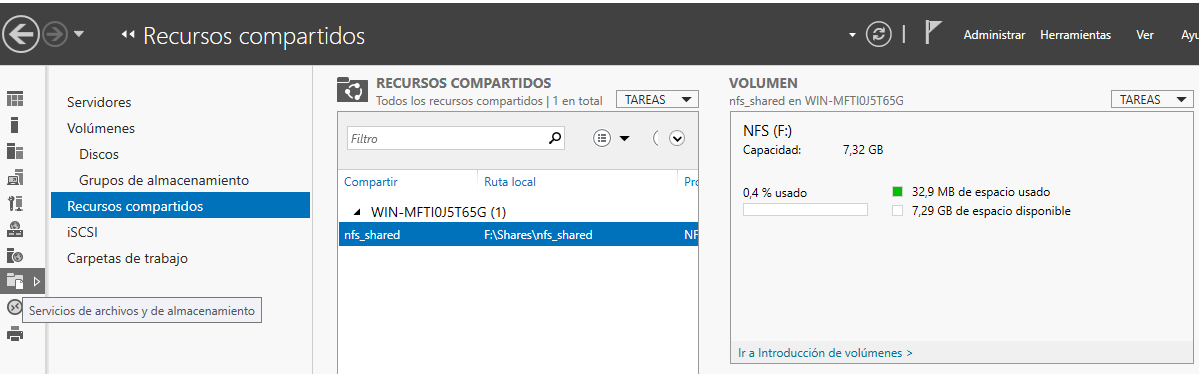
Para probarlo, nos conectamos con nuestro cliente VNC Viewer:



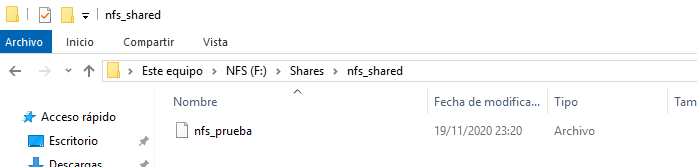
## NFS / SAMBA

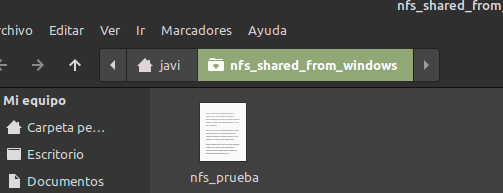
Para instalar el servicio de NFS, lo instalaremos desde el Administrador del servidor.

Una vez instalado, configuramos la carpeta compartida.

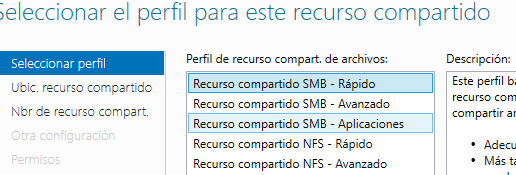


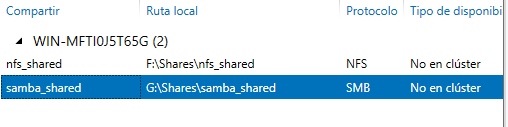
Para comprobarlo, he creado un archivo *nfs\_prueba* en la carpeta compartida.



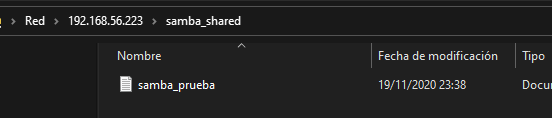
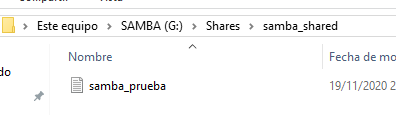


Para instalar el servicio de SAMBA, lo instalaremos igual que NFS.





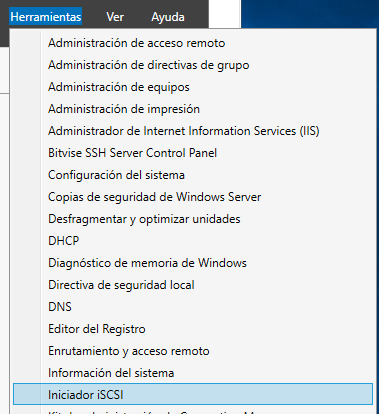
En nuestro cliente, Win + R y escribimos [\\IP-Servidor](file:///\\IP-Servidor).



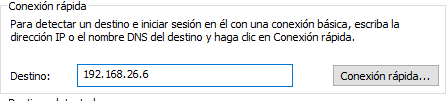
## FreeNAS + iSCSI

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.

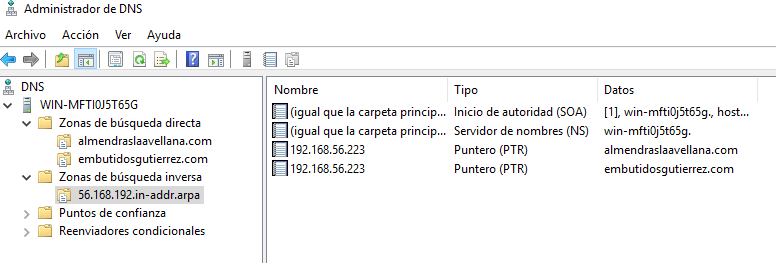


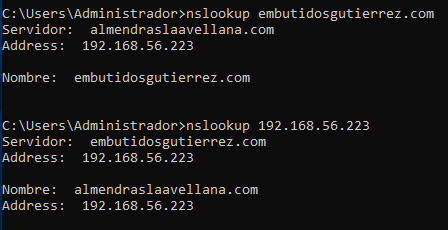
Por último, montamos el disco.



## 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.