Como root del sistema ejecuta los siguientes commandos:

```
apt-get install nfs-kernel-server
mkdir /export
chmod 777 /export

mount --bind /home/vagrant/info_docker /export
```

En `/etc/fstab` añade la siguiente linea:

```
/home/vagrant/info_docker /export none bind 0 0
```

En `/etc/exports` añade la siguiente linea:

```
/export       127.0.0.1(rw,fsid=0,no_root_squash,insecure,no_subtree_check,async)

service nfs-kernel-server start

mkdir -p /test
mount -t nfs 127.0.0.1:/export /test

exportfs -a

service nfs-kernel-server restart


docker run -tid --name nfs_cli --privileged -v /test:/test busybox

docker run -tid --volumes-from nfs_cli --name=vol_test busybox
```

Prueba como se asignan permisos en volumenes:

```
FROM debian:jessie

RUN useradd hector
VOLUME /misDatos
RUN touch /misDatos/holaMundo && chown -R hector:hector /misDatos
```

Comprueba si holaMundo ha sido creado en /misDatos. ¿No, verdad?

En Docker cualquier cosa después de VOLUME no aplicará los cambios sobre ese volumen, piensa en LAYERS. Lo que pretendemos es que holaMundo se escriba en la imagen del sistema de ficheros pero lo que est'a haciendo es correr en el volumen del contenedor temporal.

Docker es inteligente y copia todos los ficheros que existen en una imagen sobre un volumen y establecen los privilegios necesarios.

Ahora prueba con esto:

```
FROM debian:jessie

RUN useradd hector
RUN mkdir -p /misDatos && touch /misDatos/holaMundo && chown -R hector:hector /misDatos
VOLUME /misDatos
```