/

**Virtualización de servicios en Docker**

**Coordinador**

**Antonio Sepúlveda Ruiz**

**Grupo de trabajo**

**Luis Valderas Torres**

**Daniel De Tomas Plaza**

**Daniel Palencia Origüel**

Contenido

[Proyecto Docker 4](#_Toc38528899)

[Instalación De Docker 5](#_Toc38528900)

[Instalación De Openldap en docker 8](#_Toc38528901)

[Instalación de Interfaz Web Openldap en docker 9](#_Toc38528902)

[Creación y configuración de Usuario, Grupos y Unidades Organizativas en LDAP en docker 10](#_Toc38528903)

[Para justificar el usar Docker, hemos replicado el mismo proceso en el propio sistema Debian, sin usar Docker, para ver como Docker nos ayuda hacer despliegues mucho más rápido 15](#_Toc38528904)

[Instalación OpenLdap Debian puro 15](#_Toc38528905)

[Creación y configuración de Usuario, Grupos y Unidades Organizativas en LDAP en Debian puro 19](#_Toc38528906)

[Unidades organizativas 21](#_Toc38528907)

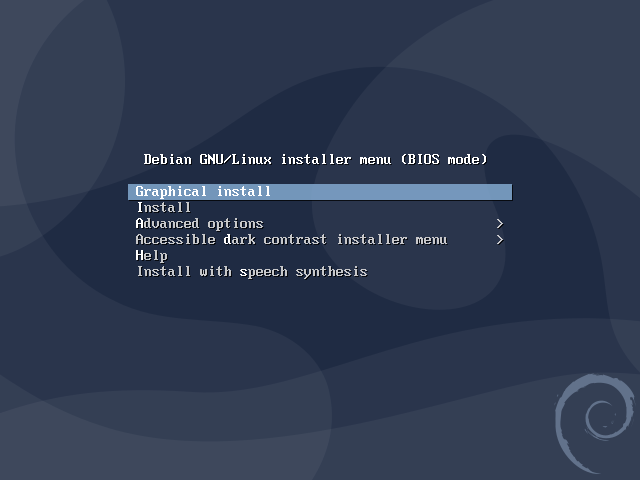
[Grupos 21](#_Toc38528908)

[Usuarios 22](#_Toc38528909)

[Instalación De Interfaz Web Openldap en Debian puro 24](#_Toc38528910)

# Proyecto Docker

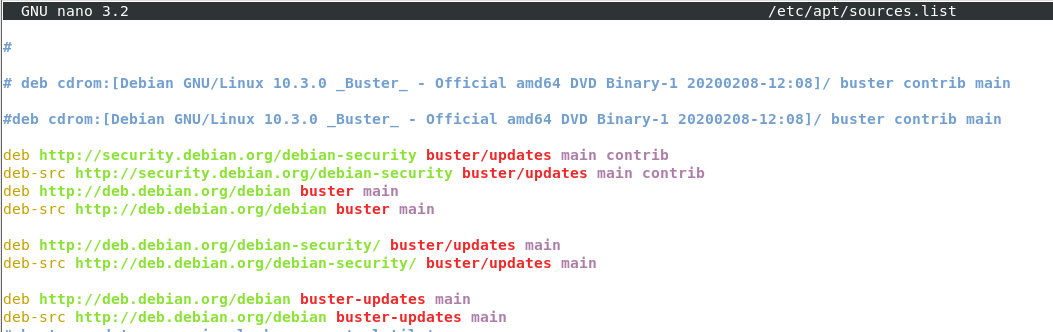
Para empezar el proyecto, lo primero que tenemos que hacer es simular la máquina de la empresa que va a contener Docker y sus servicios, para ello vamos a descargar una ISO de Debian en concreto la versión Debian 10.3.0 de su página web ([www.debian.org](http://www.debian.org)).



Una vez descargada usando VMware vamos a crear una máquina virtual con Debian que simulara ser el equipo de la empresa, vamos a configurar el nombre del equipo “Debian-Docker” y como usuario “Proyecto”.

Una vez instalado procedemos a añadir los repositorios de debian para así poder instalar/actualizar paquetes utilizando el siguiente comando.

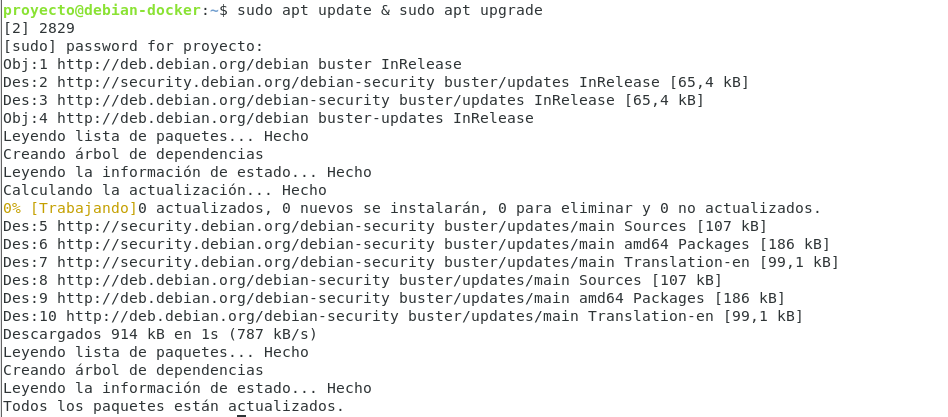
sudo nano /etc/apt/sources.list



En la web de debian podremos obtener la lista con los repositorios correspondientes.

Una vez añadidos los repositorios actualizaremos la lista de paquetes disponibles y actualizaremos los paquetes que no estén actualizados utilizaremos los siguientes comandos.

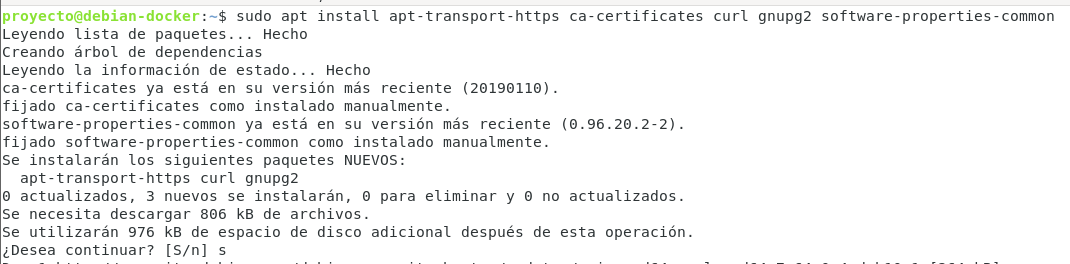
sudo apt update & sudo apt upgrade



## Instalación De Docker

Con el siguiente comando vamos a permitir a apt usar paquetes a través de HTTPS.

sudo apt install apt-transport-https ca-certificates curl gnupg2 software properties-common apt update & sudo apt upgrade



Procedemos a añadir la clave GPG del repositorio oficial de Docker para poder descargar el paquete.

sudo curl -fsSL https://download.docker.com/linux/debian/gpg | sudo apt-key add –



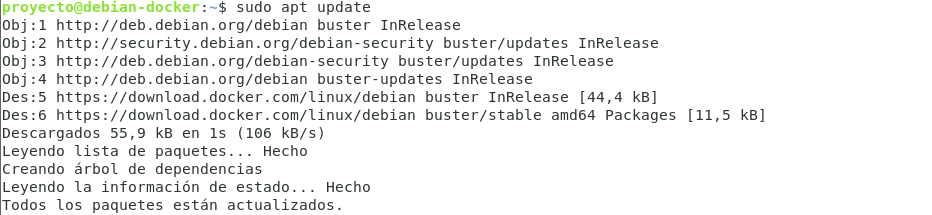
Ahora añadimos el repositorio de Docker alas fuentes del APT.

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/debian $(lsb\_release -cs) stable"



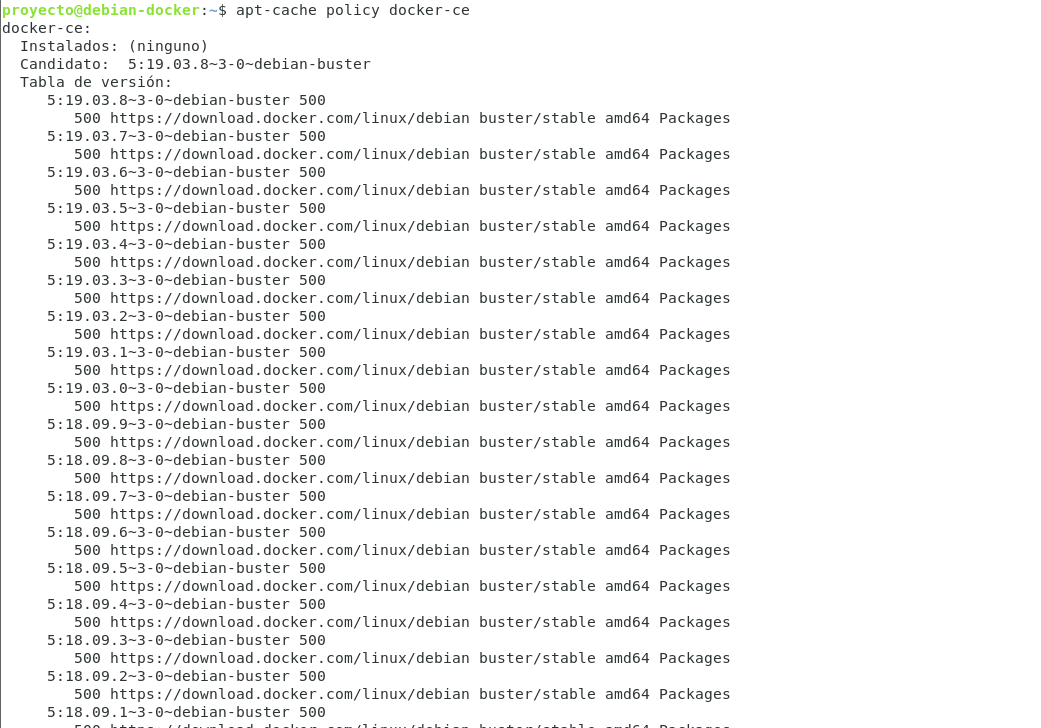
Volvemos a actualizar la base de datos de paquetes para que se añadan los datos de Docker.

sudo apt update



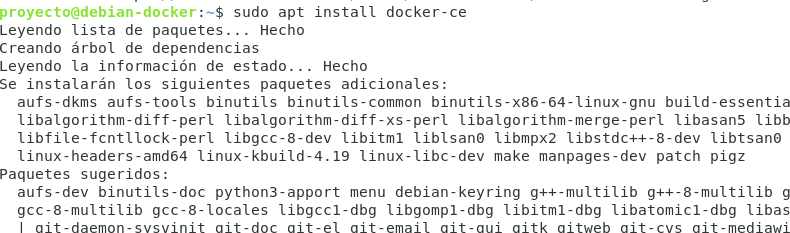
Verificamos que la instalación de Docker se va a realizar desde el repositorio de Docker en lugar del repositorio predeterminado de Debian.

sudo apt-cache policy docker-ce



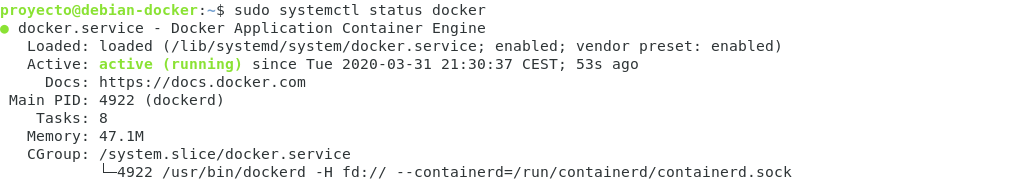
Procedemos a instalar Docker.

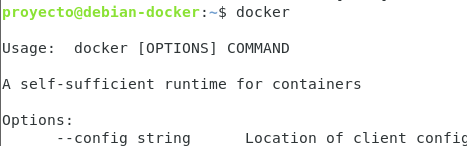
sudo apt install docker-ce



Verificamos que Docker se ha instalado correctamente.

sudo systemctl status Docker

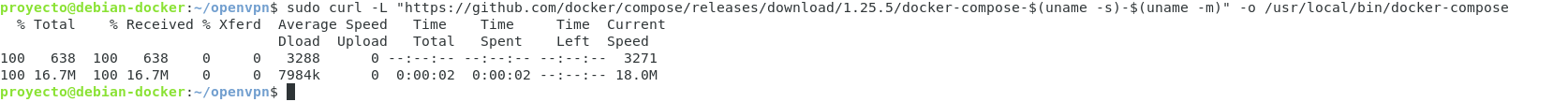




### Instalación Docker-Compose

Vamos a instalar Docker-Compose, que es una herramienta que permite realizar scripts para facilitar la creación de containers y servicios.Para instalar Docker-Compose procedemos a usar el siguiente comando.

sudo curl -L "https://github.com/docker/compose/releases/download/1.25.5/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose



Ahora le tenemos que dar permisos de ejecución para ello ejecutaremos el siguiente comando.

sudo chmod +x /usr/local/bin/docker-compose



Por último, para comprobar que Docker-compose se ha instalado sin problemas lo verificaremos con el siguiente comando.

Docker-compose --version

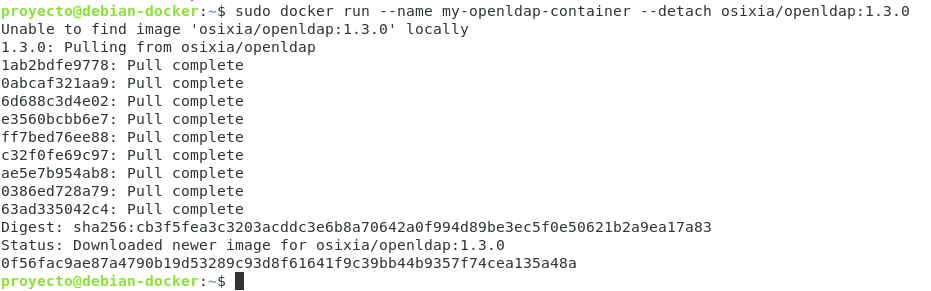


## Instalación De Openldap en docker

Para empezar, descargaremos la imagen de Docker Openldap

Docker run –name my-openldap-container –detach osixia/openldap:1.3.0

Obviamente como no lo tenemos en local, se conectará a buscar la imagen



A continuación, vamos a añadir nuestro dominio y la configuración principal con este comando

docker run -p 389:389 --name ldap-service --hostname ldap-service --env LDAP\_ORGANISATION="Jovellanos" --env LDAP\_DOMAIN="proyecto.docker" \

--env LDAP\_ADMIN\_PASSWORD="1234" --env LDAP\_BASE\_DN="dc=proyecto,dc=docker" --volume /data/slapd/database:/var/lib/ldap \

--volume /data/slapd/config:/etc/ldap/slapd.d --detach osixia/openldap:1.3.0

## Instalación de Interfaz Web Openldap en docker

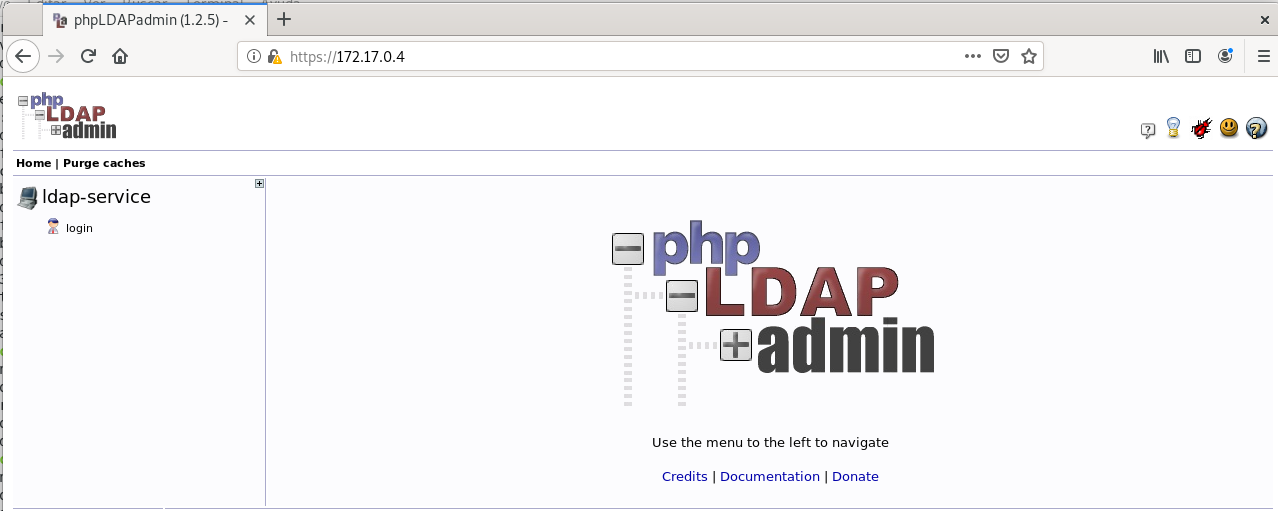
También para mayor comodidad vamos añadir una interfaz web para nuestro openldap

docker run --name phpldapadmin-service --hostname phpldapadmin-service --link ldap-service:ldap-host --env PHPLDAPADMIN\_LDAP\_HOSTS=ldap-service --detach osixia/phpldapadmin:0.9.0

Después con este comando averiguaremos cual es la ip asociada al proceso de ldapadmin

docker inspect -f "{{ .NetworkSettings.IPAddress }}" phpldapadmin-service

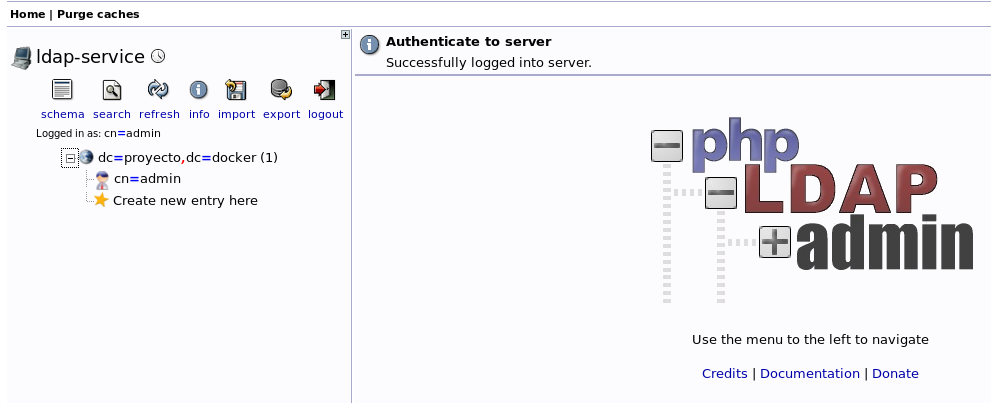
A continuación, si nos conectamos a esa ip accederemos a la interfaz web



Le daremos a la ventana de login poniendo los datos de nuestro dominio anteriormente creado

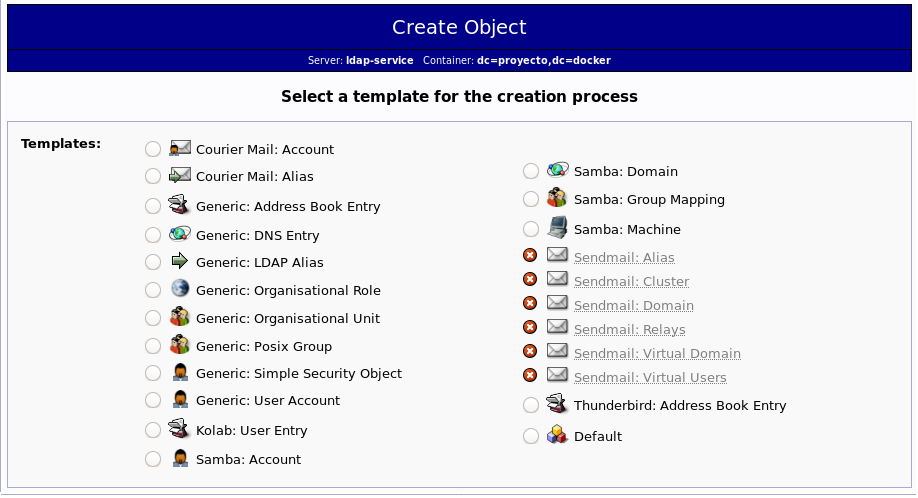


Y como comprobamos podemos acceder perfectamente

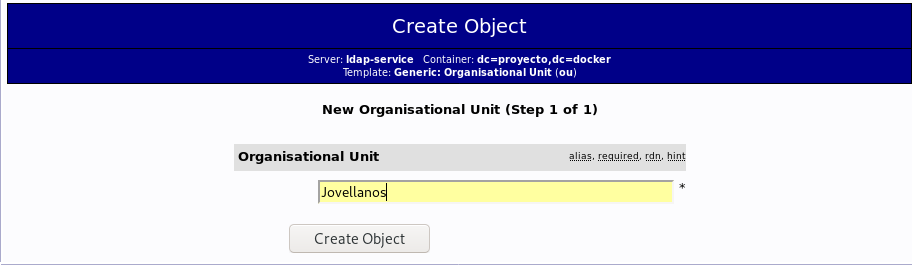


## Creación y configuración de Usuario, Grupos y Unidades Organizativas en LDAP en docker

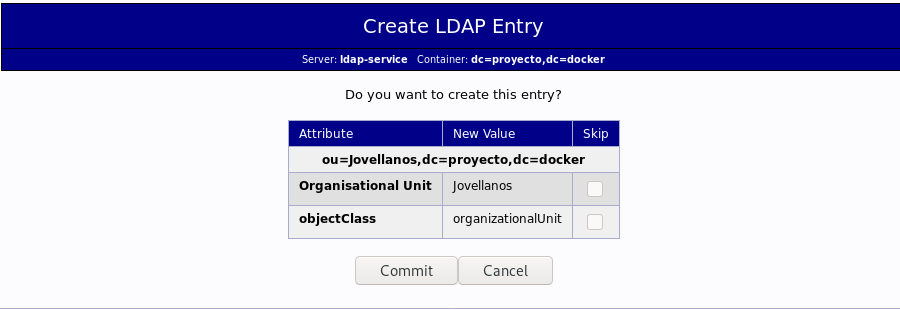
A continuación, creamos un par de entradas para comprobar su funcionalidad



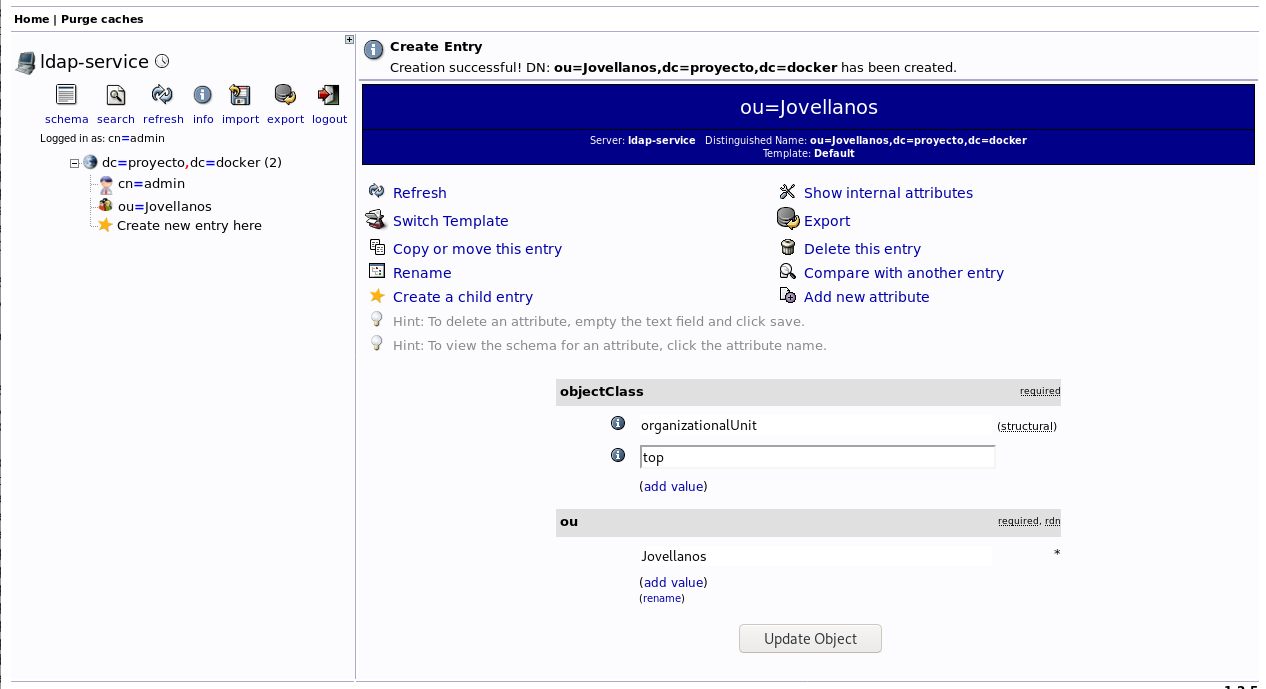
Para empezar, cogeremos crear una UO



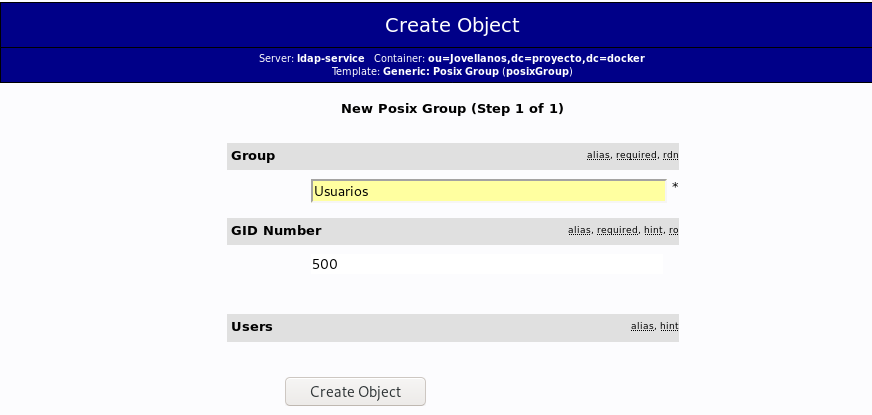
Comprobamos los datos metidos y le damos a commit



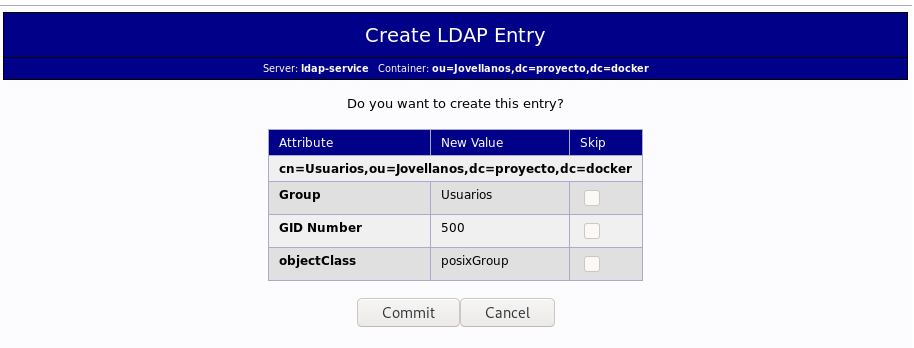
Como vemos, ya aparece a la izquierda



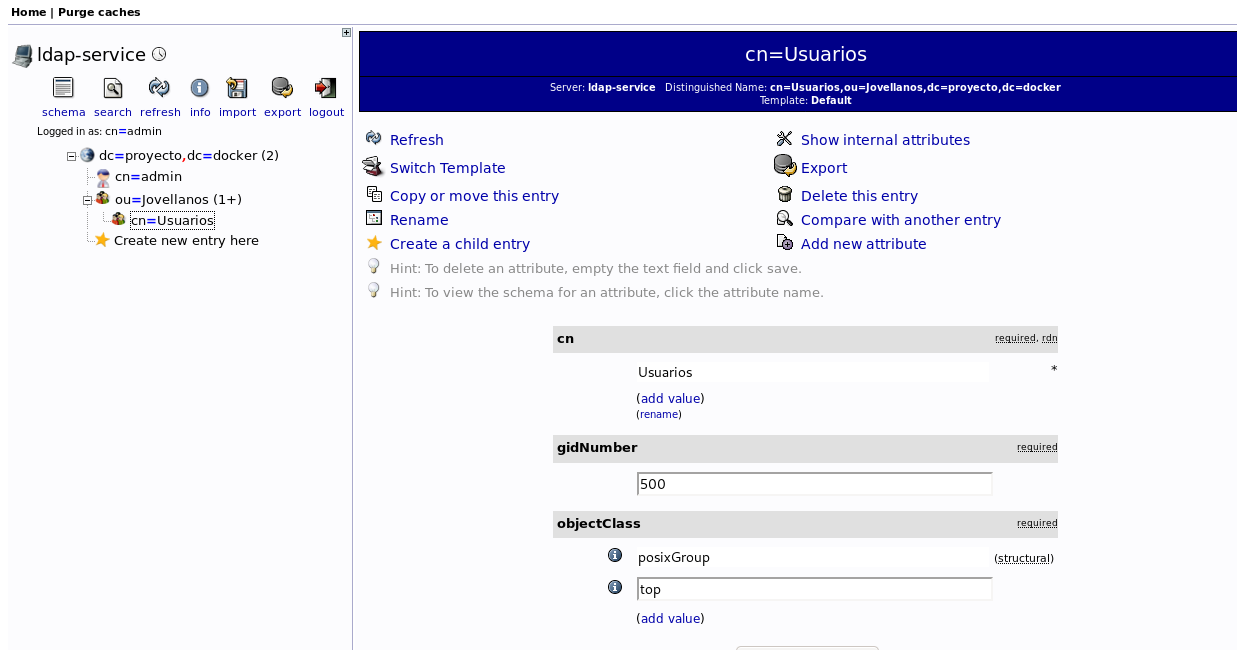
Creamos ahora un grupo llamado usuarios dentro de la UO Jovellanos



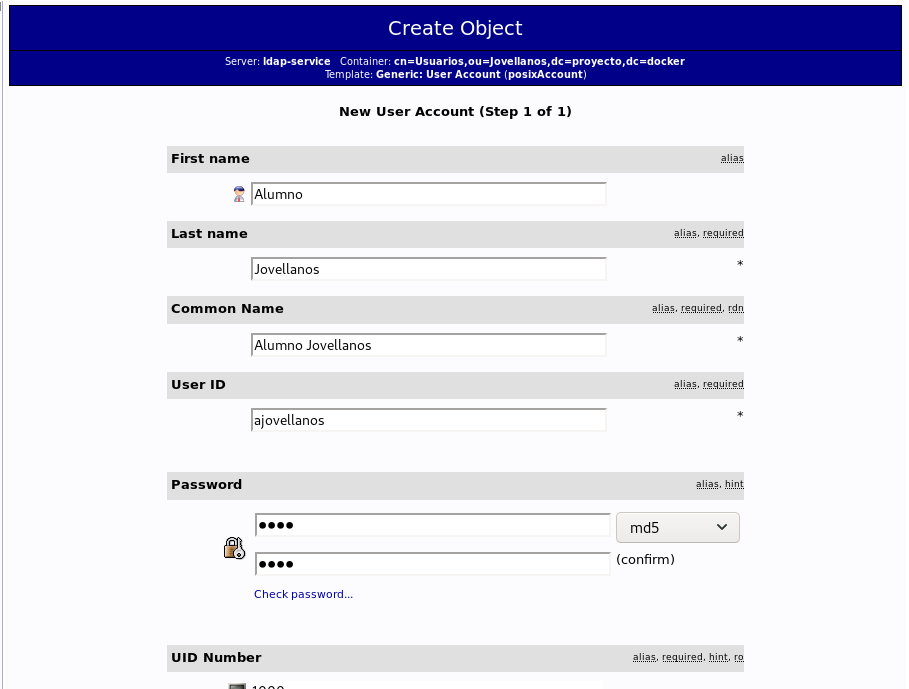
Comprobamos la entrada

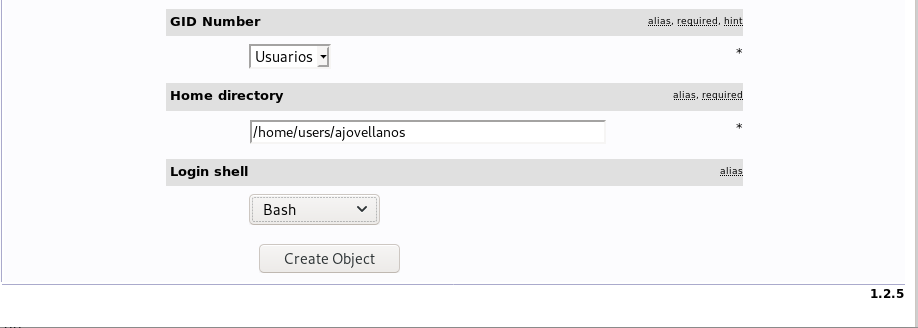


Y ya lo tenemos creado

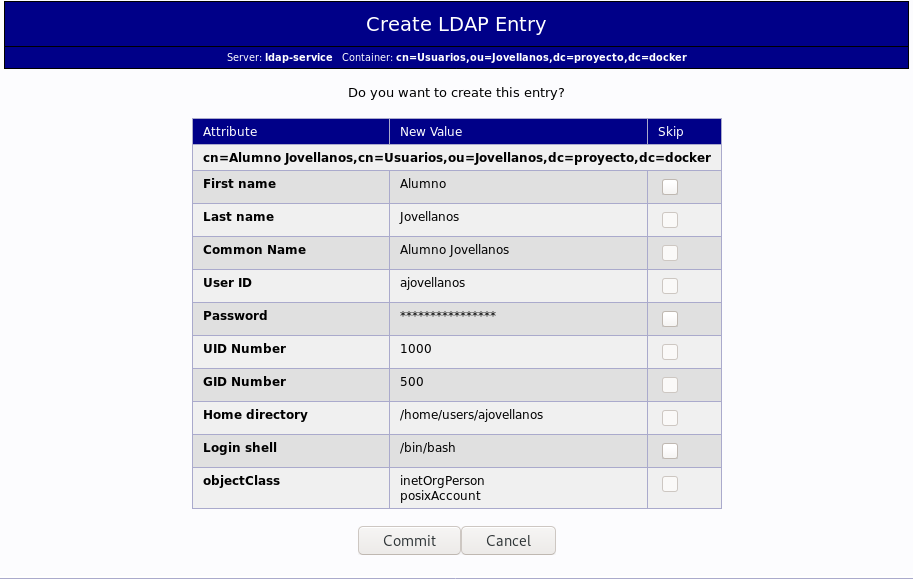


A continuación, creamos un usuario dentro del grupo usuario

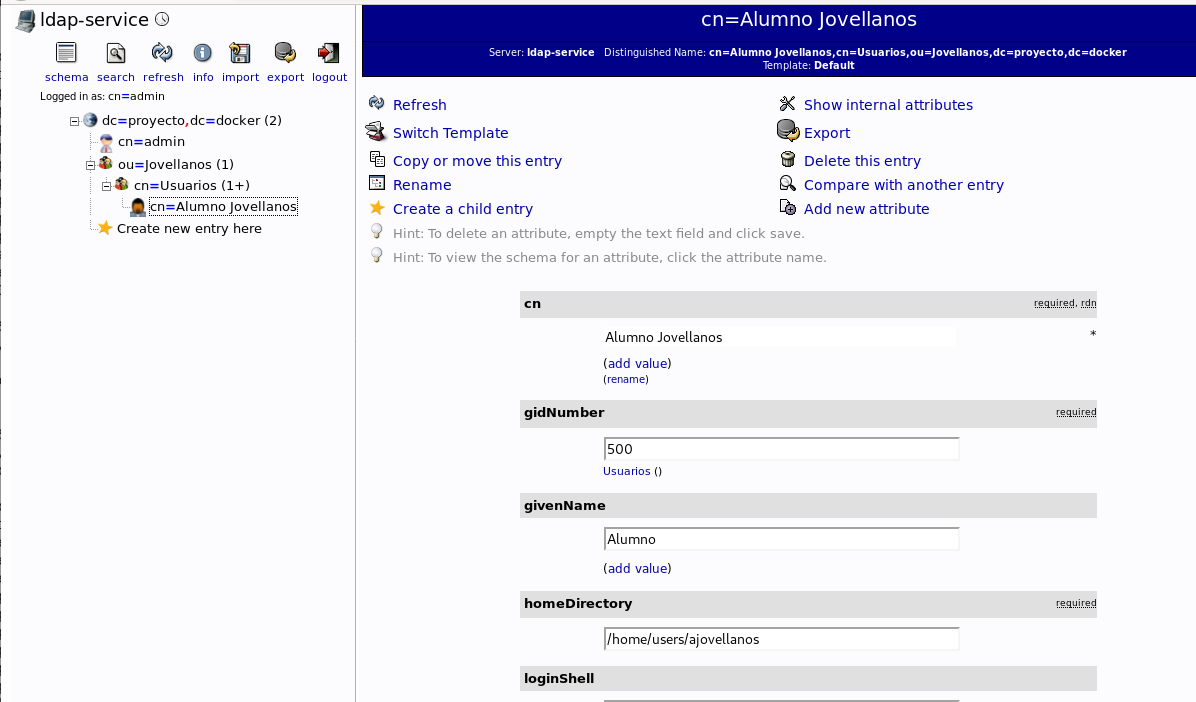




Como vemos podemos configurar múltiples parámetros como password, directorio home, la Shell que usara etc…



Y ya lo tendríamos creado

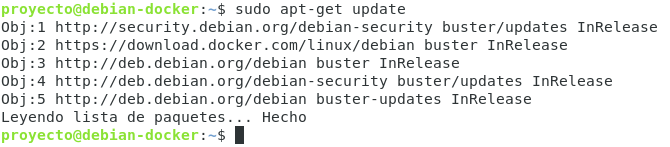


# Para justificar el usar Docker, hemos replicado el mismo proceso en el propio sistema Debian, sin usar Docker, para ver como Docker nos ayuda hacer despliegues mucho más rápido

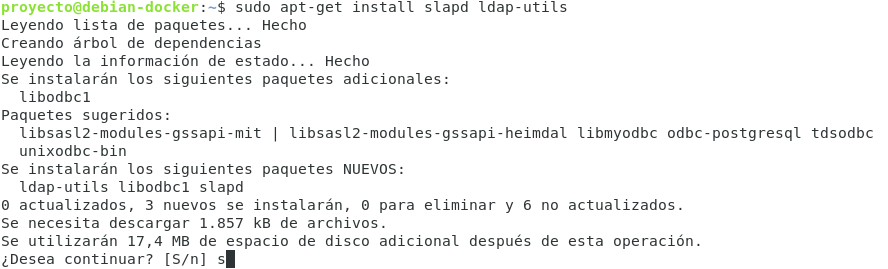
## Instalación OpenLdap Debian puro

Para empezar, realizaremos un

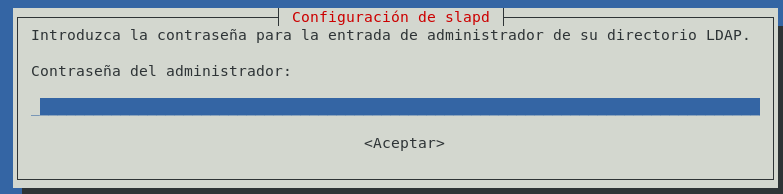
apt-get update



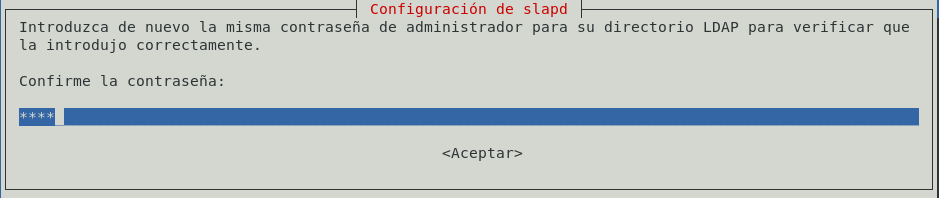
Una vez actualizado los repositorios nos descargaremos el paquete deseado



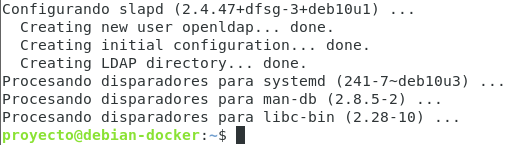
Una vez realizada la instalación del paquete nos pedirá la contraseña de administrador del directorio LDAP



Nos pide de nuevo una confirmación de contraseña

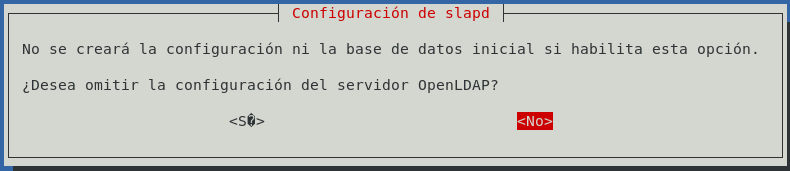


Tras esto, comprobamos que ya ha realizado por si solo una configuración inicial



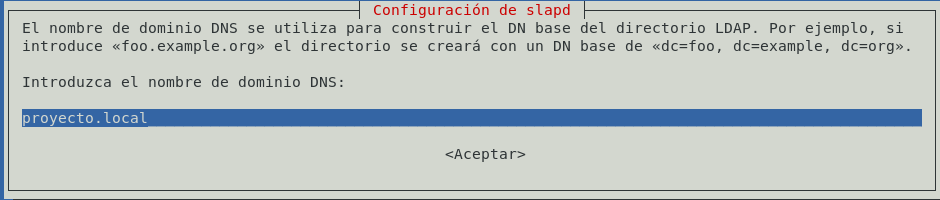
Como nosotros queremos una configuración propia ejecutaremos

dpkg-reconfigure -plow slapd

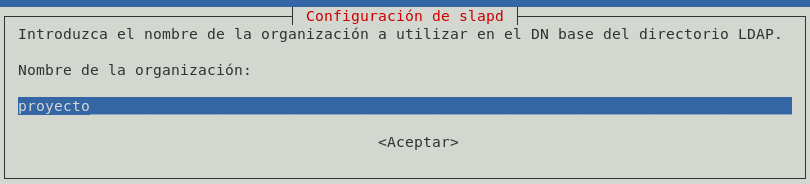


Así que aquí le daremos a la opción No

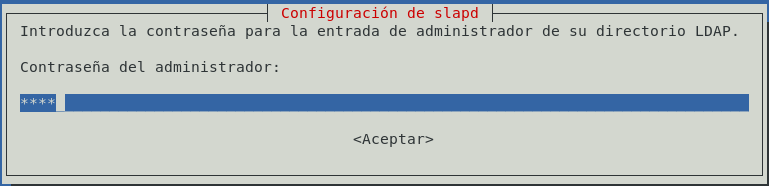
A continuación, nos pide que introduzcamos un nombre de dominio dns en nuestro caso será proyecto.Local



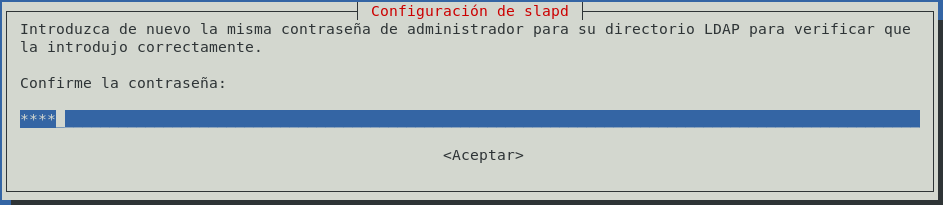
Nos pide un nombre de organización para utilizar en LDAP en nuestro caso será proyecto



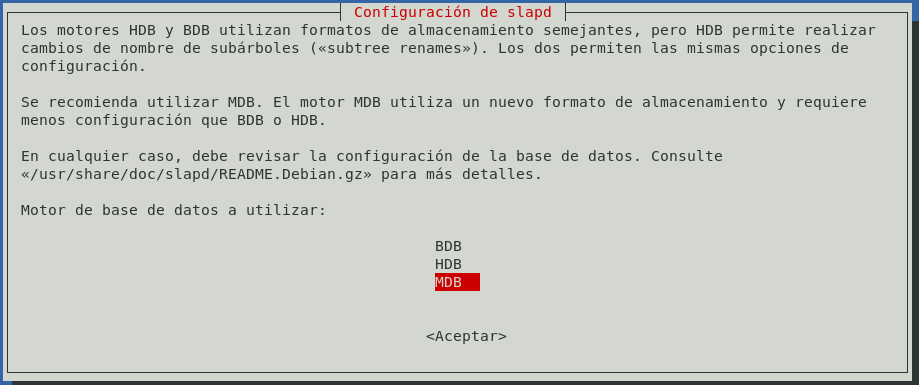
A continuación, nos pide la contraseña del administrador del directorio LDAP



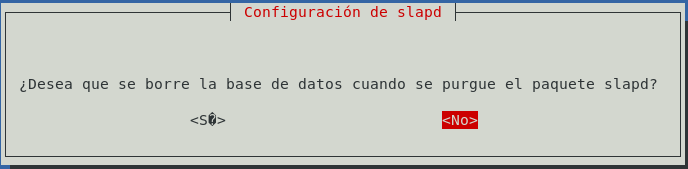
Como antes pide que la confirmemos introduciéndola de nuevo



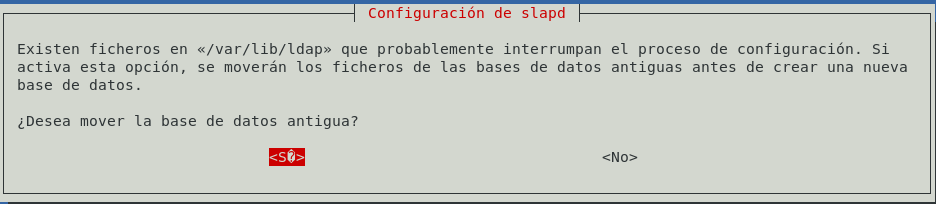
En el siguiente paso nos pide configurar el motor de base de datos que queremos usar, por defecto dejaremos el marcado MDB



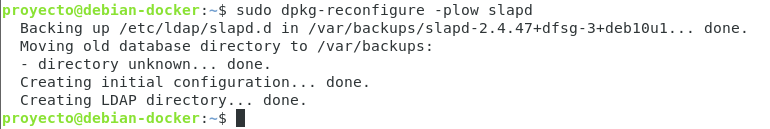
A continuación, nos pregunta que si deseamos que se elimine el directorio slapd de la base de datos cuando slapd sea eliminado, en su defecto dejamos No



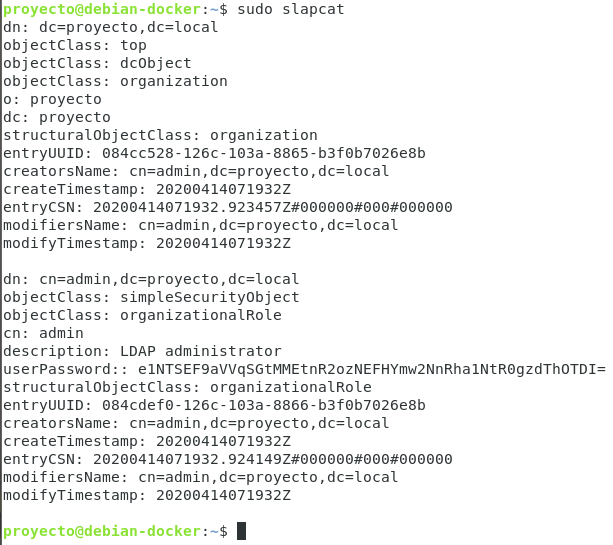
En la siguiente opción nos da a elegir si deseamos que se mueva el antiguo directorio, por defecto dejamos la opción Si



Una vez realizado el proceso nos ha vuelto a realizar la configuración, pero esta vez con nuestros parámetros



Para comprobar que todos los parámetros son correctos usaremos el comando slapcat que nos muestra el siguiente resultado

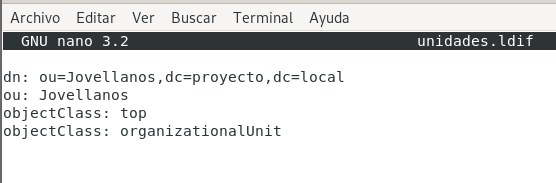


Con esto hemos finalizado la configuración inicial de LDAP, a continuación, pasaremos a crear los grupos, unidades organizativas y usuarios que deseemos

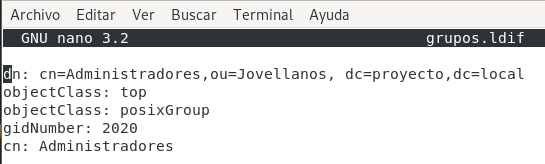
## Creación y configuración de Usuario, Grupos y Unidades Organizativas en LDAP en Debian puro

Para tener una organización fácil y optima vamos a manejar ficheros. ldif que a posterior se cargan en LDAPP a través de un comando, así a su vez podemos tener un fichero. ldif para poder separar usuarios, grupos y UO según nos venga en gana, A continuación, vamos a configurar los ficheros.

Primero crearemos la unidad organizativa Jovellanos

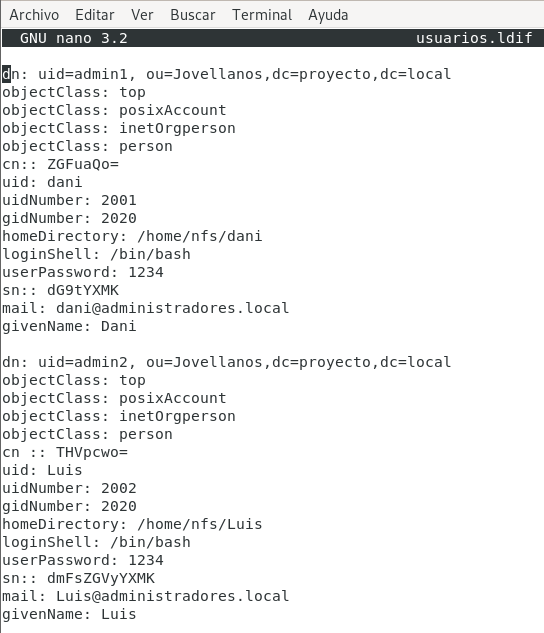


Ahora crearemos el grupo administradores



A continuación, hemos creado dos usuarios dentro de administradores, Tanto el cn como el sn que se requiere para este tipo de cuentas lo hemos realizado en base 64 con el comando

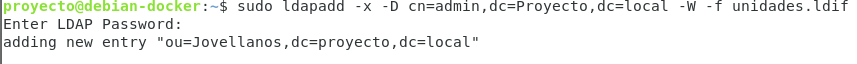
Echo dani | base64 y el resultado es el cn, repetimos el paso con el sn y en este caso ponemos el apellido, así hemos obtenido los valores requeridos



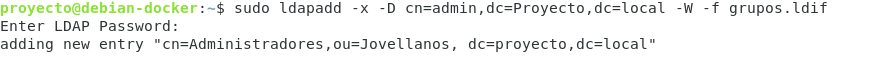
Ahora procederemos añadir a ldap los datos de cada fichero. ldif creado

ldapadd -x -D dc=Jovellanos,dc=local -W -f unidades.ldif

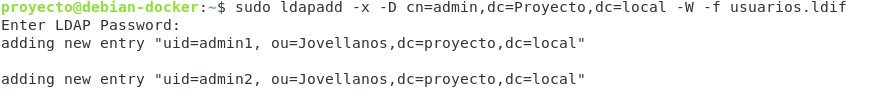
### Unidades organizativas



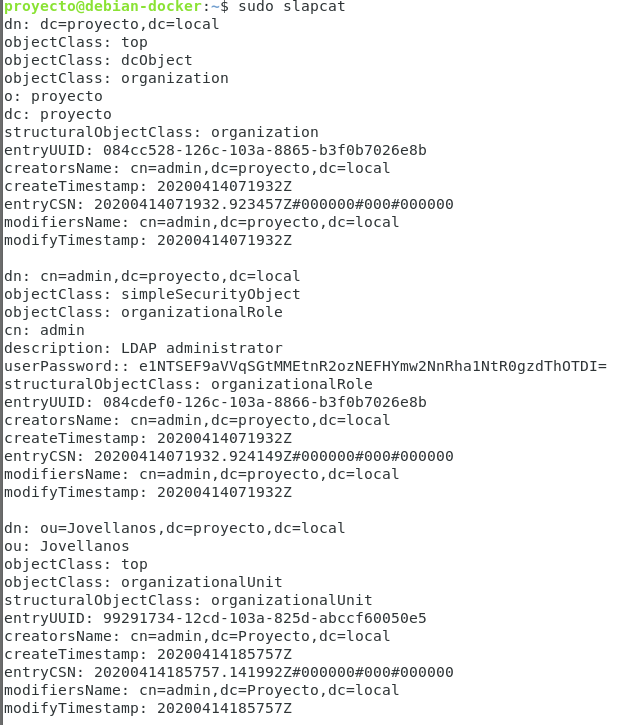
### Grupos

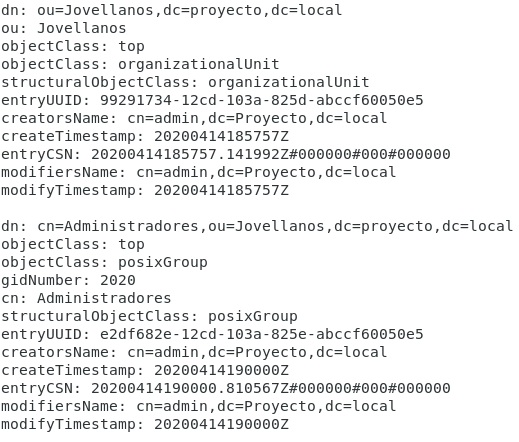


### Usuarios

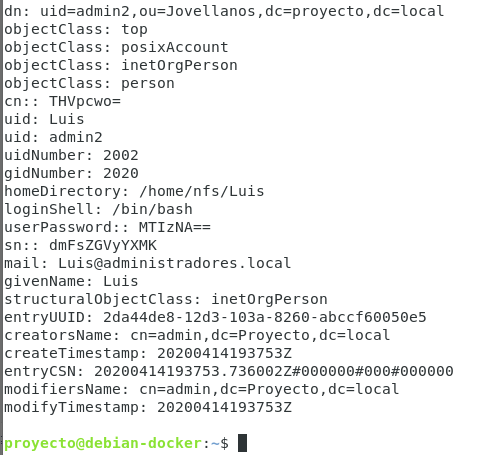


Tras estos procesos podemos usar el comando slapcat para comprobar la información







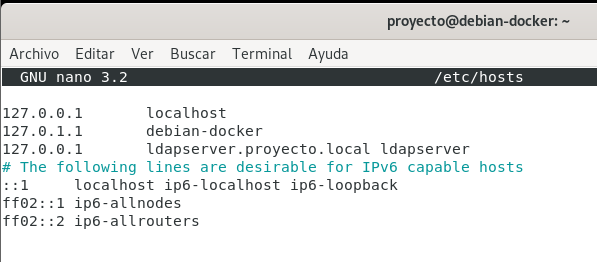


Con esto hemos finalizado la creación de UO, grupos y usuarios deseados para esta demo

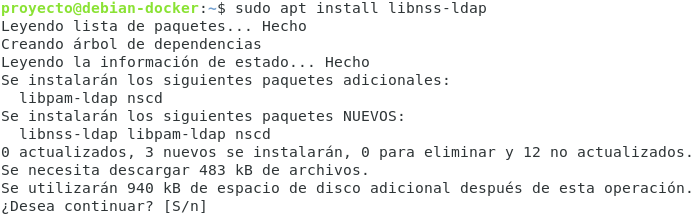
## Instalación De Interfaz Web Openldap en Debian puro

Para un manejo más fácil de Openldap instalaremos una interfaz web

Primero modificaremos el fichero /etc/hosts y añadimos la entrada del server

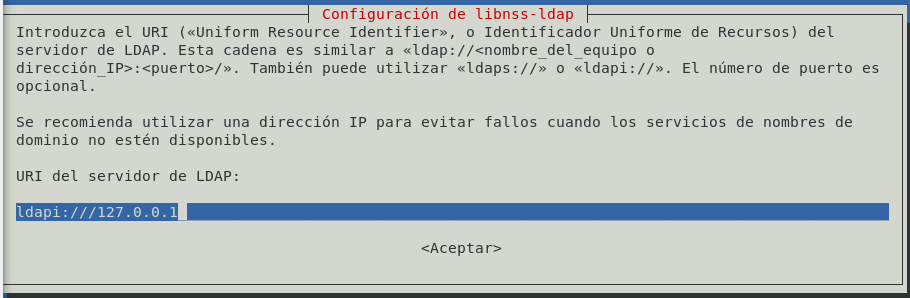


A continuación, procederemos a instalar el paquete libnss-ldap

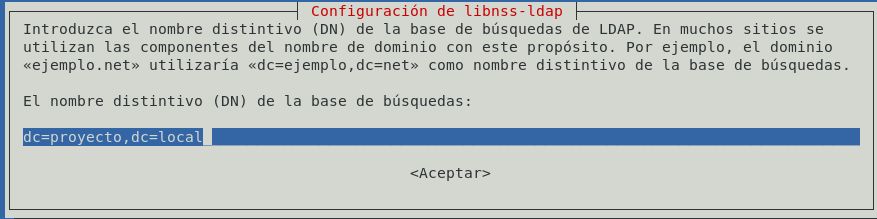


Proseguiremos con la configuración correspondiente

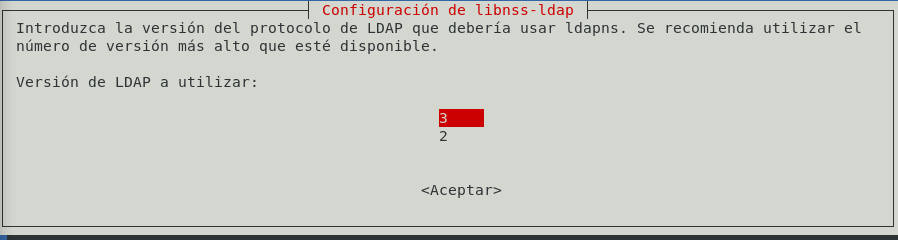
Aquí nos pide la ip que usaremos



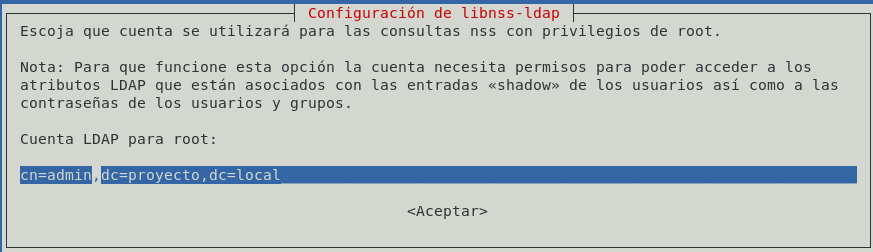
Aquí nos pide rellenar el nombre distintivo



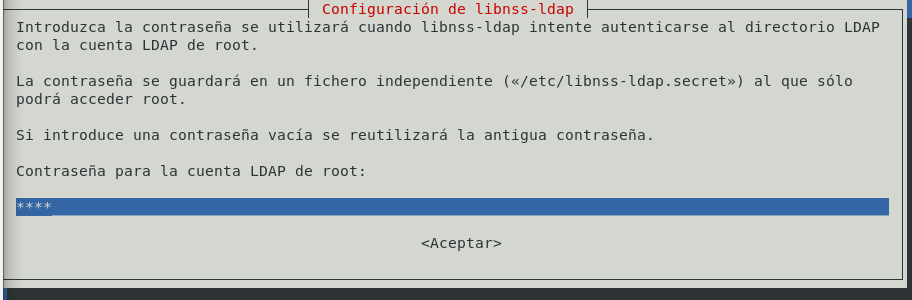
Aquí cogeremos el valor mas alto como indica



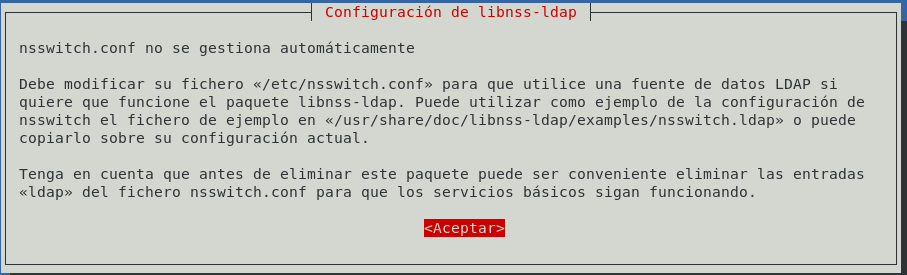
Configurando la cuenta admin para LDAP



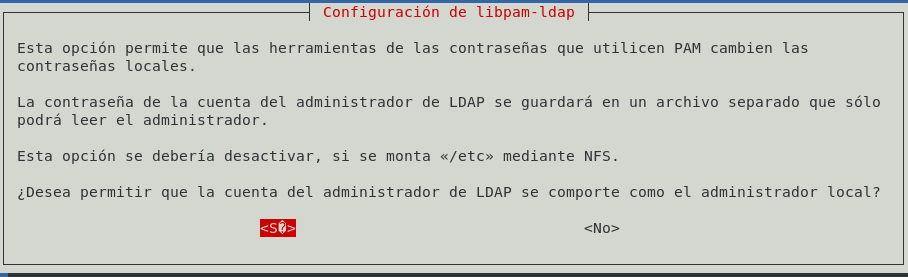
Contraseña admin

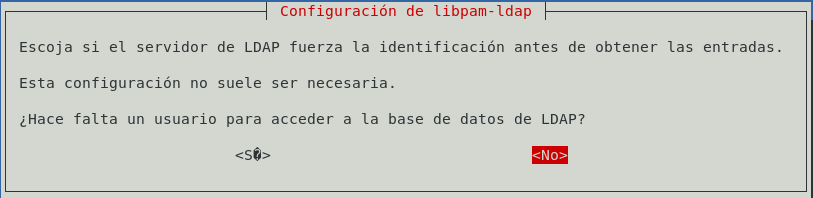


Aviso del fichero nsswitch.conf



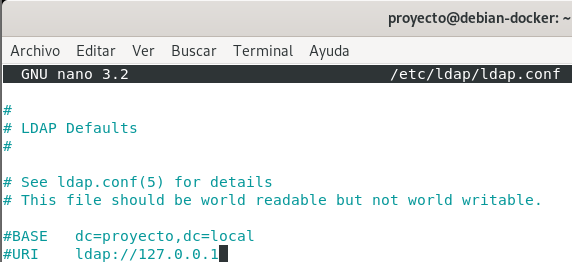
Aquí marcamos la opción por defecto





Vamos a configurar los clientes

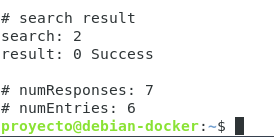




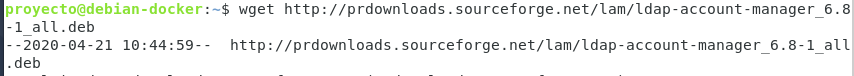
Ejecutaremos el siguiente comando para comprobar que funciona el servidor

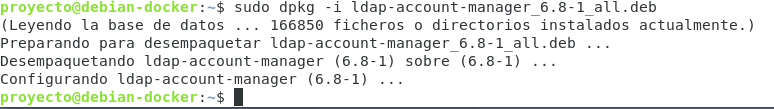


Nos devuelve un 0 así que es correcto



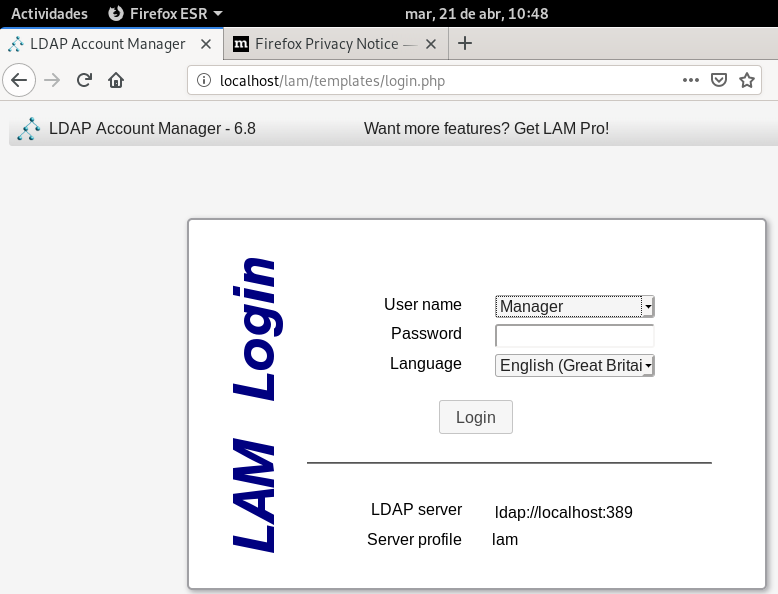
Terminamos bajando la interfaz web deseada



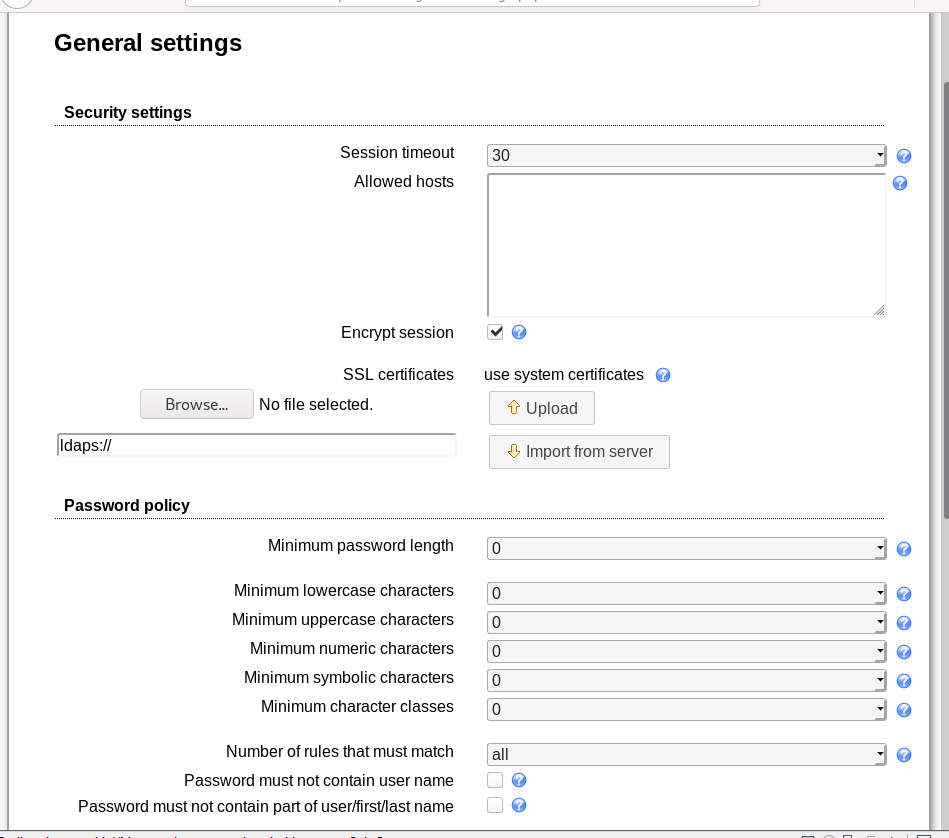


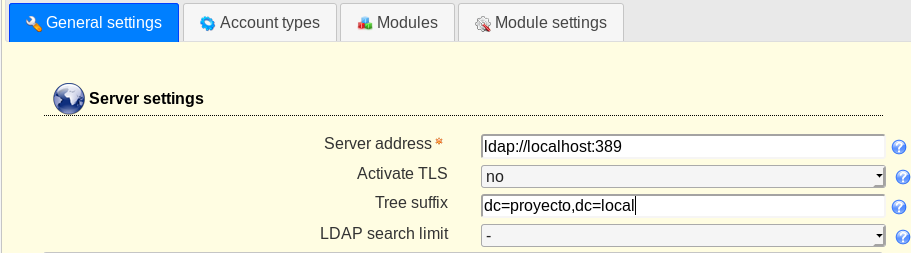
Comprobamos que tras esto podemos acceder

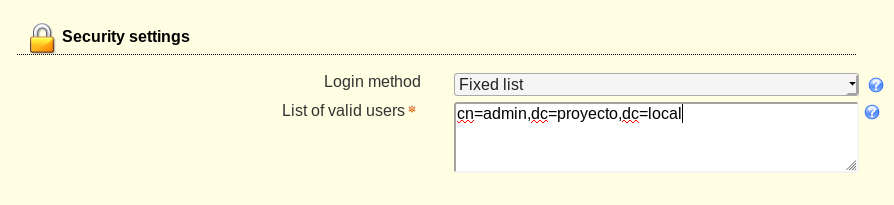
A la interfaz web



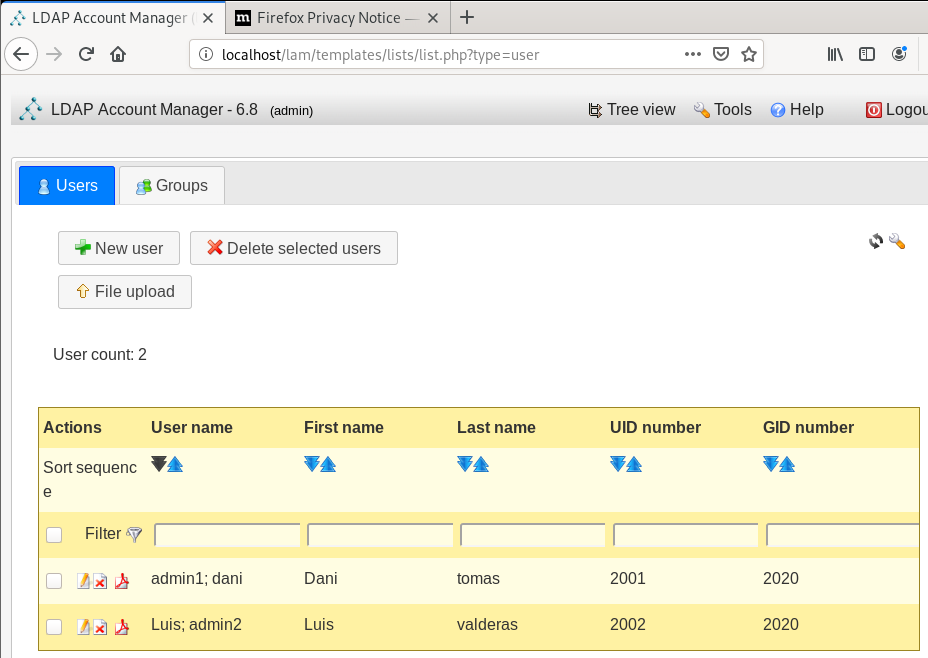
Vamos a proceder a configurarlo







Como vemos después de configurar los parámetros, nos reconoce los ficheros ldif creados

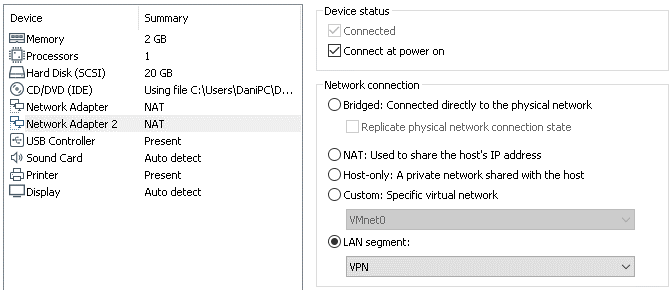


Y hasta aquí como se haría si no usásemos Docker, como vemos es un proceso más tedioso, pero con el mismo resultado (hemos usado otras interfaces web para comprobar e investigar la funcionalidad de cada una)

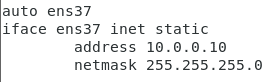
## Instalación OPENVPN DOCKER

### Configuración Máquina Virtual

El primer paso que vamos a realizar va a ser instalar una segunda tarjeta de red que vamos a poner en un segmento llamado VPN para poder realizar posteriormente las pruebas de conexión mediante la VPN.



Dentro de la maquina vamos a asignarle la siguiente dirección de red.



### Docker-Compose Instalación

Para poder instalar OPENVPN vamos a acudir a buscar su imagen, para ello acudimos a <https://hub.docker.com/> , donde dentro del mismo podremos encontrar todas las imágenes publicadas para Docker de distintos servicios, aplicaciones.

Una vez dentro podremos encontrar haciendo una búsqueda la imagen oficial de OPENVPN que corresponde al repositorio **kylemanna/openvpn.**

En el repositorio podemos encontrar una guía de cómo realizar la instalación de OPENVPN en Docker.

En nuestro caso vamos a utilizar la opción de Docker-Compose para realizar la instalación.

Para ello vamos a generar un fichero .yml con el nombre de Docker-compose.



Vamos a rellenar el fichero con los siguientes parámetros para la instalación.



* Version : Corresponde con la versión de Docker-Compose que vamos a utilizar en este caso la 2.
* Services: Le indicamos que servicio vamos a utilizar en este caso OPENVPN.
* Cap\_add: Nos permite añadirle al contenedor privilegios, en este caso con NET\_ADMIN nos permite darle la capacidad de poder manipular cosas como, cambiar la configuración de la Interfaz, administrar el Firewall, modificar la tabla de rutas etc.

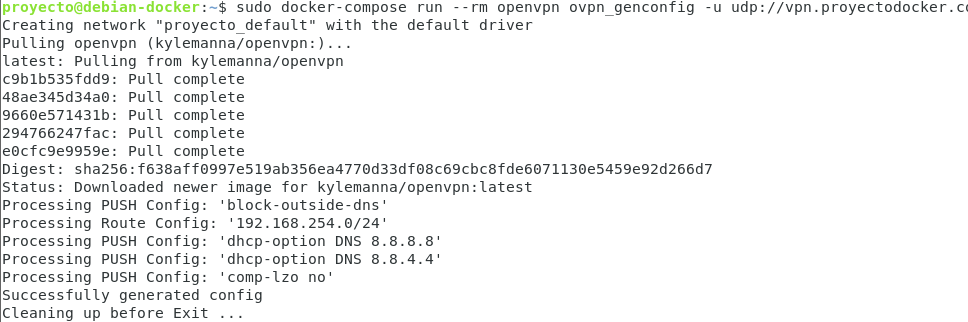
Podemos buscar todas las opciones que nos da el kernel de Linux buscando en google man 7 capabilities, de esta forma podremos ver una lista completa de todas las opciones.

* Image: Le indicamos que imagen queremos usar, en este caso vamos a usar la oficial.
* Container\_name: Indicamos a Docker el nombre que tendrá el contenedor.
* Ports: Le indicamos los puertos que queremos utilizar.
* Restart: Le indicamos que siempre queremos que el contenedor se reinicie en caso de que se detenga.
* Volúmenes: Le indicamos donde queremos que se cree el volumen que va a contener el contenedor, esto es algo muy importante ya que, en caso de no indicarle un punto de montaje, Docker generaría un volumen que en caso de actualizar el contenedor se borraría, en este caso vamos a mapear el volumen a la carpeta /proyecto/openvpn-data/conf. Esto mismo se podría realizar con la opción -v en caso de no hacer la instalación con Docker-compose.

Una vez generado el fichero, vamos a iniciar generando la configuración del VPN, le vamos a indiciar el dominio a resolver para conectarse, también le podríamos poner directamente la IP.

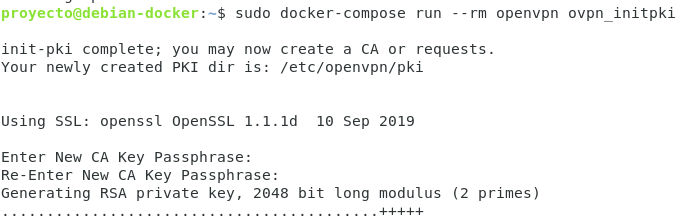
\*Con el modificar –rm conseguimos crear un contenedor temporal una vez acabe de ejecutarse se eliminará.

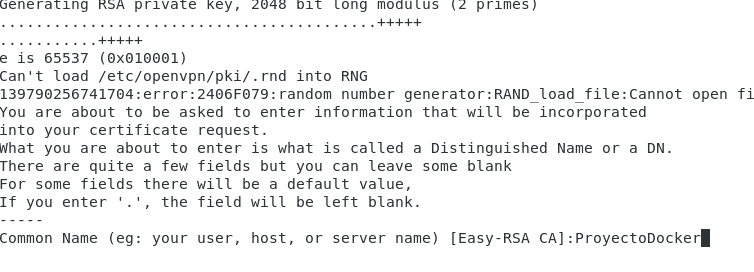
docker-compose run --rm proyectodockervpn ovpn\_genconfig -u udp://vpn.proyectodocker.com

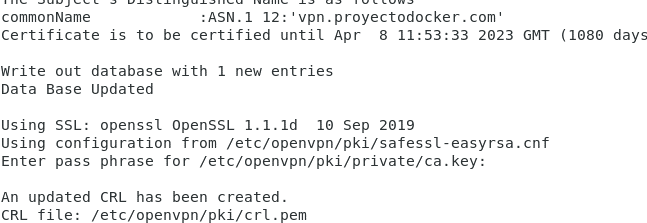


Para hacer funcionar el Servidor VPN tenemos que generar una llave publica, una privada y un certificado. Para realizar esta tarea vamos a utilizar el siguiente comando.

docker-compose run --rm openvpn ovpn\_initpki







Una vez acabada habrá generado las claves publica/privada y el PKI. Una vez generados ya podemos iniciar el contenedor.

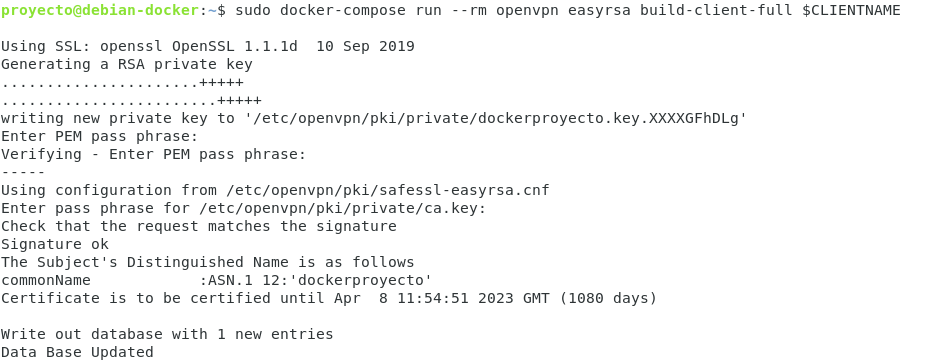
Sudo docker-compose up -d openvpn



Ahora vamos a generar una variable donde almacenaremos el nombre del usuario del vpn, esto no es necesario si le indicáramos directamente el nombre funcionaria de la misma forma.



Vamos a generar al usuario dentro de la base de datos del servidor VPN.



Una vez creado lo único que nos queda por hacer es generar el fichero de configuración que tendrá que usar el usuario para conectarse.

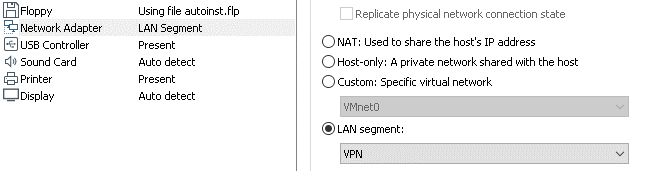
Sudo docker-composer run –rm openvpn ovpn\_getclient $CLIENTNAME > $CLIENTNAME.ovpn





#### Comprobación Funcionamiento VPN

Para comprobar el funcionamiento de la VPN vamos a crear una maquina cliente, en este caso vamos a usar una distribución de Ubuntu, la tarjeta de red la vamos a configurar en el segmento de red de VPN.



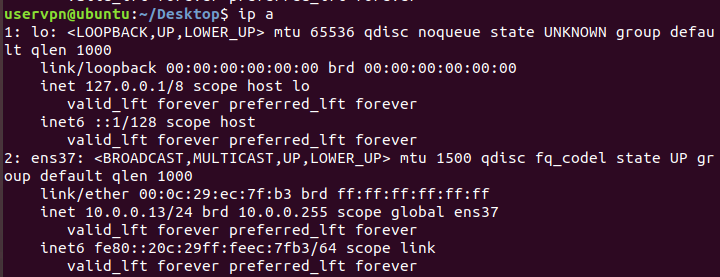
Una vez configurado la interfaz con su correspondiente IP, vamos a realizar varias comprobaciones:

-Revisar la configuración de la interfaz.

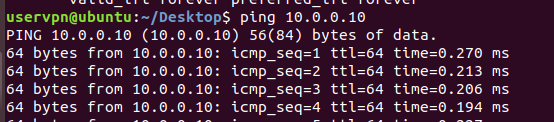
-Tener conectividad con el servidor VPN.

-No tener salida a internet hasta pasar por el VPN.

-Configurar el dns para resolver vpn.proyectodocker.com

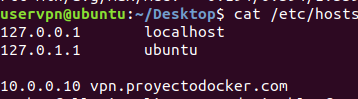


Para comprobar la conectividad con el servidor VPN vamos a realizar ping a la IP 10.0.0.10.



Como podemos comprobar podemos llegar a esa IP.

Ahora vamos a configurar el fichero /etc/hots para que resuelva el dominio y nos llevo a la dirección IP 10.0.0.10



Ya sabemos que podemos llegar a resolver la dirección del vpn, por lo tanto, vamos a ejecutar el perfil del vpn para poder conectarnos.