CLOUD COMPUTING

MEMORIA DE LA PRÁCTICA 1

Adrián de la Torre Rodríguez

Índice

1. Descripción del problema

El objetivo de esta primera práctica es poner en uso los conocimientos adquiridos en relación con el uso de plataformas de cloud computing a nivel de IaaS y SaaS. Para ello, se deberá realizar el diseño y posterior despliegue de una aplicación para gestión de archivos. La aplicación que se desea desplegar precisa de los siguientes servicios:

- Autenticación basada en LDAP.

- Alojamiento, sincronización y compartición de archivos.

- Base de datos

2. Diseño

La plataforma escogida para esta práctica ha sido Azure de Microsoft, se ha elegido por los siguientes motivos:

- Las prácticas de la asignatura se han desarrollado sobre la misma plataforma.

- Ya contaba con bastante conocimiento sobre ella.

- Tenía una suscripción con dinero.

- Por otro lado Azure actualmente es una de las plataformas más competentes y más usadas del mercado.

El despliegue se va a realizar sobre contenedores en Azure, en total se van a lanzar tres contenedores uno por cada servicio y serán conectados entre ellos correctamente para permitir su comunicación.

El software elegido para cumplir los requisitos de la práctica ha sido ownCloud por ser de software libre y permitir el almacenamiento de archivos con una interfaz muy práctica. Tras un estudio de este software he comprendido que es la base central del sistema completo, él se conectará a una base de datos donde se guardarán los archivos y a un sistema de autenticación donde se guardarán los usuarios del sistema. Para la base de datos se ha escogido la imagen de mariadb que está basada en mysql. En principio elegí mysql pero tras numerosos problemas decidí usar mariadb. En cuanto a la autenticación de usuarios se va a usar una imagen de LDAP.

El primer paso será lanzar los contenedores de mariadb y LDAP ya que ownCloud depende de ellos. Una vez desplegados, se lanzará el contenedor de ownCloud y se conectará con los servicios desplegados.

3. Despliegue

En este apartado se van a indicar los pasos seguidos para desplegar todo el sistema y se mostrarán capturas de su correcto funcionamiento. En primer lugar vamos a desplegar un contenedor en Azure con una imagen de MariaDB. Para ello Azure cuenta con la facilidad de desplegar contenedores directamente con un comando sin la necesidad de crear una máquina virtual, instalar docker y luego instalar el contenedor. Para ello es necesario tener instalado el CLI de Azure, haberse logueado y tener una suscripción válida. Primero hay que crear un grupo de recursos donde se alojarán todos los servicios desplegados. Para crear un grupo de recursos hay que escribir el siguiente comando:

```
$ az group create --location westeurope --name myResourceGroup
```

Para lanzar el contenedor basta con escribir el siguiente comando:

```
$ az container create --resource-group myResourceGroup \
                      --dns-name-label dockermariadb \
                      --name dockermariadb \
                      --image mariadb:latest \
                      --ports 3306 \
                      --environment-variables MYSQL_ROOT_PASSWORD=password
```

El comando anterior crea un contenedor en el grupo de recursos *myResourceGroup*, con la etiqueta dns *dockermariadb* (es necesario indicarle este apartado para que cree el contenedor con una IP asociada), su nombre será *dockermariadb*, la imagen es *mariadb:latest*, es decir la última versión, se abre el puerto 3306 ya que es el que usa mariadb y además es necesario pasarle la variable de entorno *MYSQL_ROOT_PASSWORD* con una contraseña ya que sin esa variable el contenedor no podía iniciar.

La siguiente imagen muestra el resultado de escribir el comando anterior:

```
adritake@foreman:~/RepositoriosGit/CC2-18-19$ bash InstallMariaDBDocker.sh
{
  "containers": [
    {
      "command": null,
      "environmentVariables": [
        {
          "name": "MYSQL_ROOT_PASSWORD",
          "secureValue": null,
          "value": "password"
        }
      ],
      "image": "mariadb:latest",
      "instanceView": {
        "currentState": {
          "detailStatus": "",
          "exitCode": null,
          "finishTime": null,
          "startTime": "2019-05-02T09:28:50+00:00",
          "state": "Running"
        },
        "events": [
```

Una vez lanzado el contenedor es necesario comprobar que funciona y además crear una base de datos que pueda usar owncloud posteriormente. Para comprobar si está escuchando en el puerto 3306 basta con usar la herramienta telnet desde la línea de comandos:

```
adritake@foreman:~/RepositoriosGit/CC2-18-19$ telnet 52.137.62.115 3306
Trying 52.137.62.115...
Connected to 52.137.62.115.
Escape character is '^]'.
p
5.5.5-10.3.14-MariaDB-1:10.3.14+maria~bionic◆kM}NO!U◆◆7nJb/d*@F{[mysql_native_passwordConnection closed by foreign host.
```

A continuación hay que conectarse al contenedor y crear una base de datos, para ello usamos el comando de azure el cual ejecuta una terminal en el contenedor indicado:

```
az container exec --resource-group myResourceGroup --name dockermariadb --exec-command "/bin/bash"
```

En la terminal del contenedor ejecutamos `mysql -p` para poder realizar operaciones en el mysql server. Introducimos la contraseña y creamos una base de datos para owncloud con CREATE DATABASE.

```
adritake@foreman:~/RepositoriosGit/CC2-18-19$ az container exec --resource-group myResourceGroup --name dockermariadb --exec-command "/bin/bas
h"
root@wk-caas-9928e0a355fd4dc98e00ec12490c312a-63cde28a09dc146d8a5147:/#
Enter password: 8e0a355fd4dc98e00ec12490c312a-63cde28a09dc146d8a5147:/# mysql -p
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 203
Server version: 10.3.14-MariaDB-1:10.3.14+maria~bionic mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE DATABASE `owncloudb`;
Query OK, 1 row affected (0.000 sec)

MariaDB [(none)]>
```

Una vez creada la base de datos hay que crear el servidor LDAP. Se va a crear de igual manera que el contenedor de mariadb. Escribimos el comando para crearlo:

```
az container create --resource-group myResourceGroup \
                    --dns-name-label dockerldap \
                    --name dockerldap \
                    --image osixia/openldap:1.2.4 \
                    --ports 389 636
```

Escogemos la imagen de osixia ya que en prácticas vimos que era la que funcionaba bien, además abrimos los puertos 389 y 636 que son los usados tal y como indica la documentación de LDAP aunque con el 389 es suficiente para esta práctica.

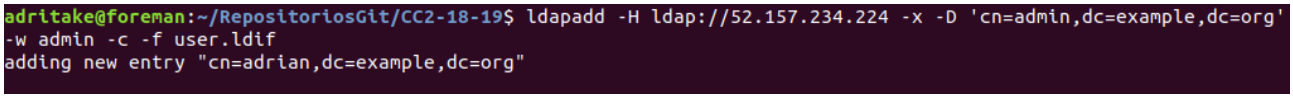
Comprobamos de igual manera que está funcionando y ahora vamos a introducirle un usuario. Para ello necesitamos tener la herramient ldaputils instalada. Creamos un documento llamado *user.lidf* el cual contendrá los datos del usuario. Me he basado en el ejemplo de prácticas y el contenido del documento es el siguiente:

```
dn: cn=adrian,dc=example,dc=org

objectClass: top

objectClass: account

objectClass: posixAccount

objectClass: shadowAccount

cn: adrian

uid: adrian

uidNumber: 16859

gidNumber: 100

homeDirectory: /home/adrian

loginShell: /bin/bash
```

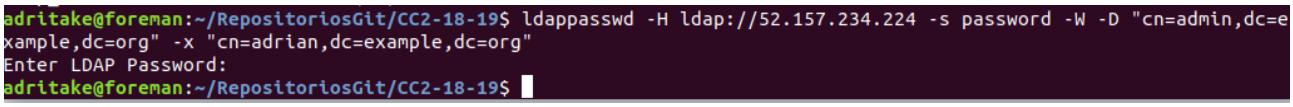
```
gecos: adrian

userPassword: {crypt}x

shadowLastChange: 0

shadowMax: 0

shadowWarning: 0
```

Para añadir ese usuario hay que escribir el comando:

```
ldapadd -H ldap://52.137.30.7 -x -D 'cn=admin,dc=example,dc=org' -w admin -c -f user.ldif
```



Ahora vamos a cambiarle la contraseña con el siguiente comando:



Con esto ya tenemos el servidor LDAP funcionando correctamente y con un usuario creado.

El último paso es desplegar ownCloud, en esta ocasión vamos a realizarlo de distinta manera. Primero creamos una máquina virtual en azure con una imagen de Ubuntu:

```
az vm create \
    --resource-group myResourceGroup \
    --name ownCloudMV \
    --image UbuntuLTS \
    --admin-username adritake \
    --generate-ssh-keys \
    --size Standard_A0 \
    --custom-data installOwnCloudDocker.sh
```

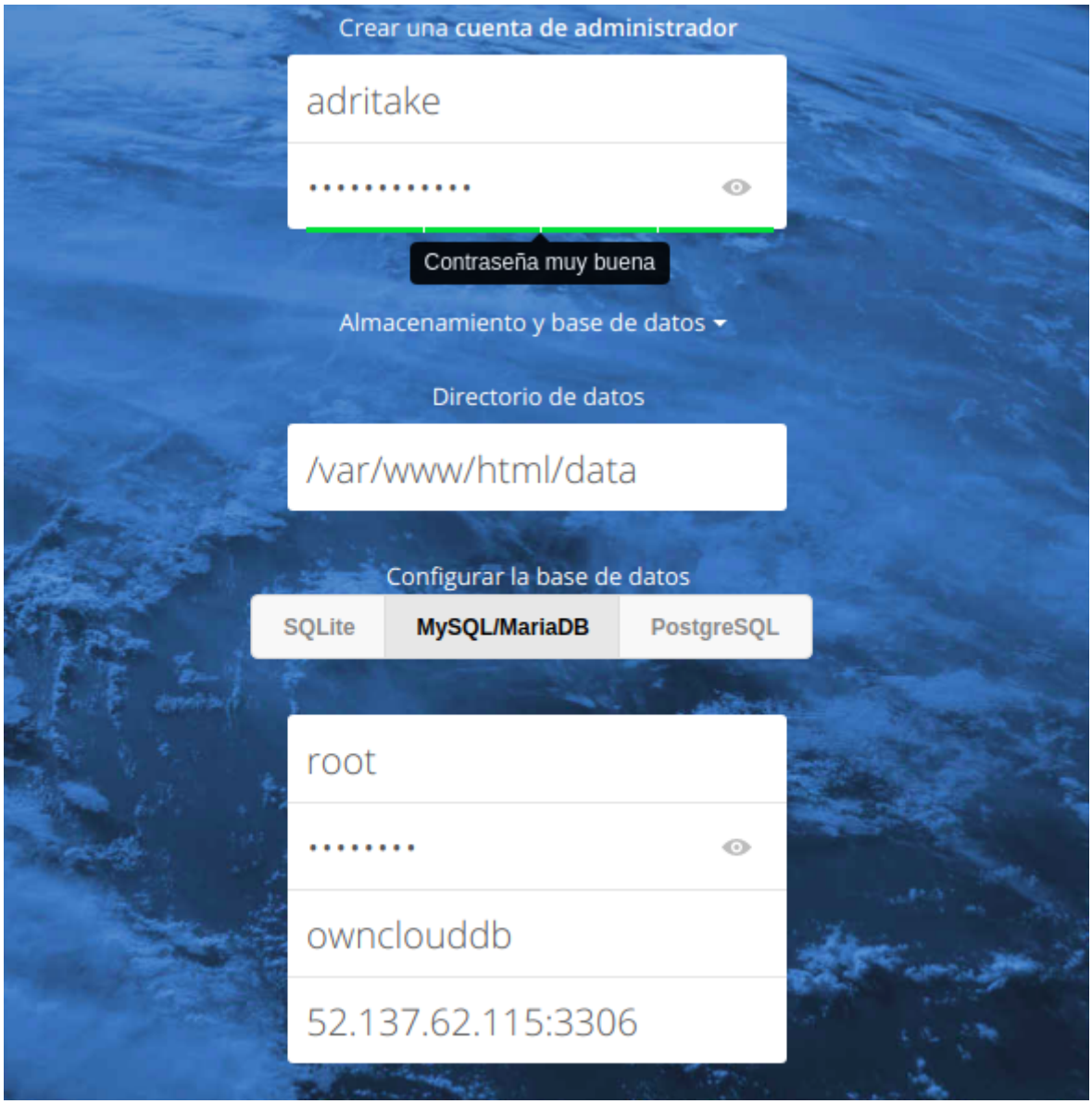
El anterior comando crea la máquina en el grupo de recursos indicado, el nombre de la MV es *ownCloudMV*, su imagen es *UbuntuLTS*, el nombre de usuario es *adritake*, se le indica que copie la clave shh para poder conectarse a ella sin tener que introducir contraseña, el tamaño de disco es

Standard_A0 que es de los más baratos. El argumento *custom-data* indica el script que queremos que se ejecute en la MV una vez ha sido creada. En este caso el contenido del script es el siguiente:

```
#!/bin/bash

# Actualizamos las aplicaciones

sudo apt update

# Agregamos la clave GPG para el repositorio oficial de Docker

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

# Agreamos el repositorio de docker a las fuentes de apt

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $
(lsb_release -cs) stable edge"

# Nos aseguramos de que se instala desde el repositorio de docker en lugar del de ubuntu

apt-cache policy docker-ce

# Instalamos docker

sudo apt-get install -y docker-ce

# Permitimos en el cortafuegos de la MV el puerto 80 ya que es el que usa ownCloud

sudo ufw allow 80

# Activamos el cortafuegos

sudo ufw enable

# Lanzamos el docker de owncloud en el puerto 80

sudo docker run -d -p 80:80 owncloud
```

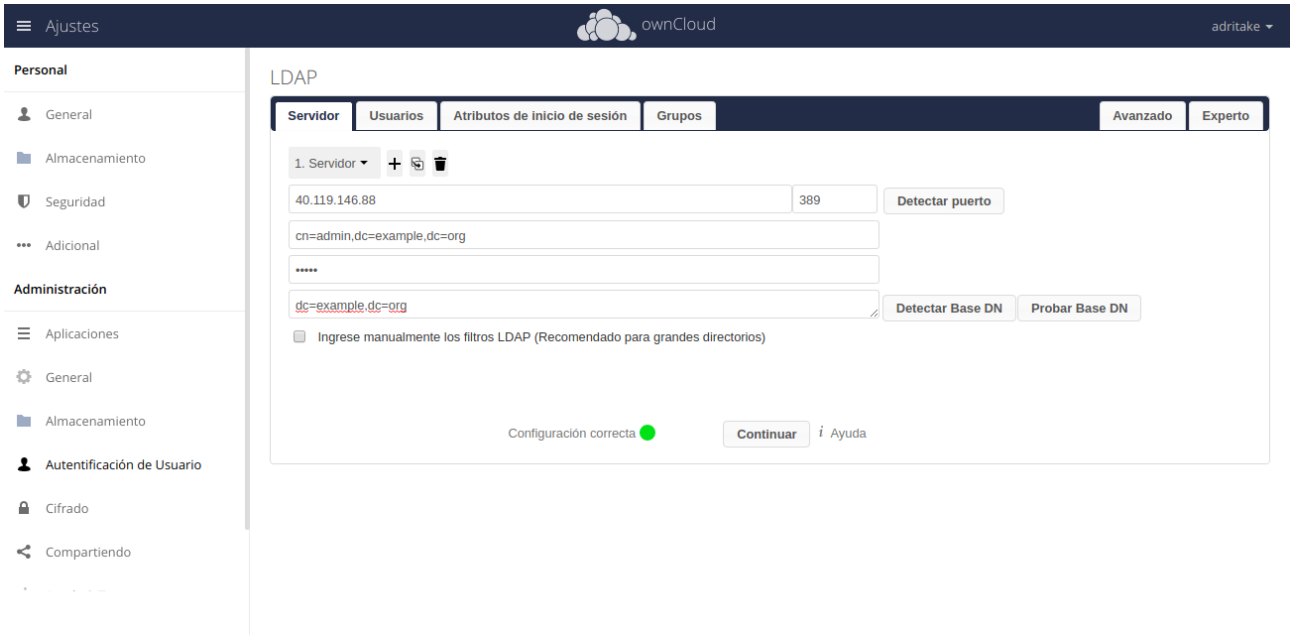
Una vez instalado OwnCloud y ejecutándose, nos conectamos a la IP de la máquina virtual en el puerto 80 y aparece la siguiente pantalla de instalación:

Antes de rellenar los datos de instalación es necesario tener ejecutando el docker de mariadb. Dicho esto, rellenamos los datos de la siguiente manera:



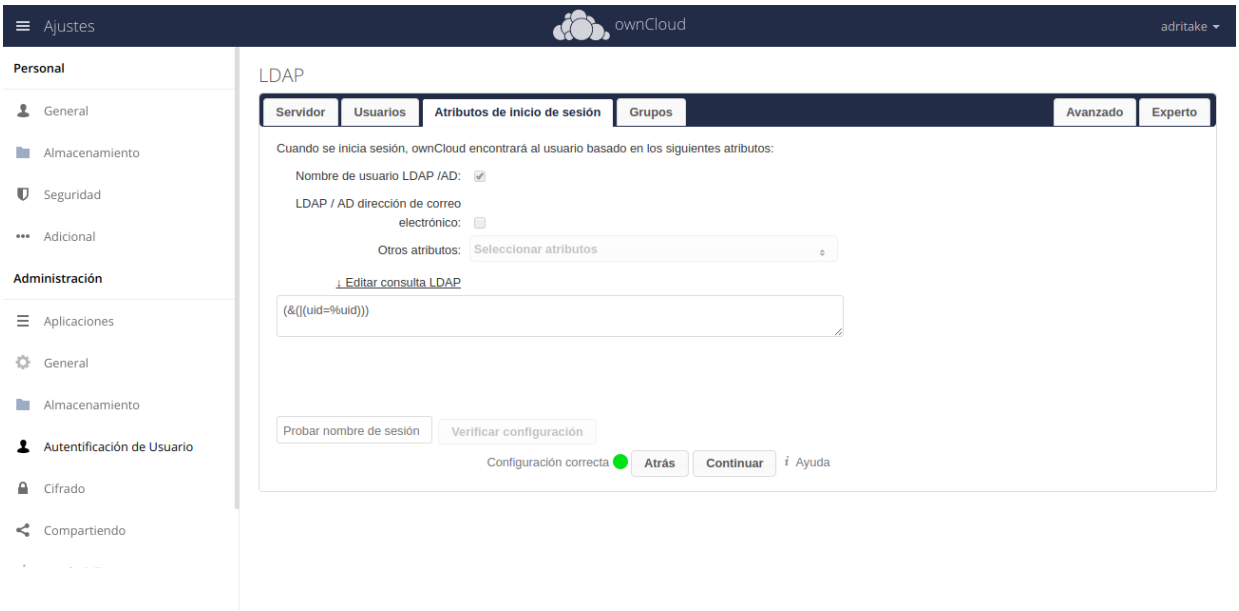
Introducimos un nombre de usuario cualquiera y una contraseña para la cuenta de administrador. Seleccionamos la pestaña de MySQL/MariaDB y le indicamos el usuario *root* que es el que tiene MariaDB por defecto, la contraseña es la introducida en la variable de entorno cuando creamos el docker (en este caso *password*), la base de datos es la que creamos cuando accedimos al contenedor. Por último introducimos la IP del docker de MariaDB y el puerto que usa que es el 3306.

Una vez dentro de OwnCloud accedemos al Market y descargamos la aplicación LDAP Integration la cual delega la gestión de usuarios a un servidor LDAP. Una vez instalada, vamos a *ajustes* > *autentificación de usuario* y allí introducimos los datos del servidor LDAP tal y como aparece en la siguiente imagen (hay que tener el docker de LDAP funcionando previamente):

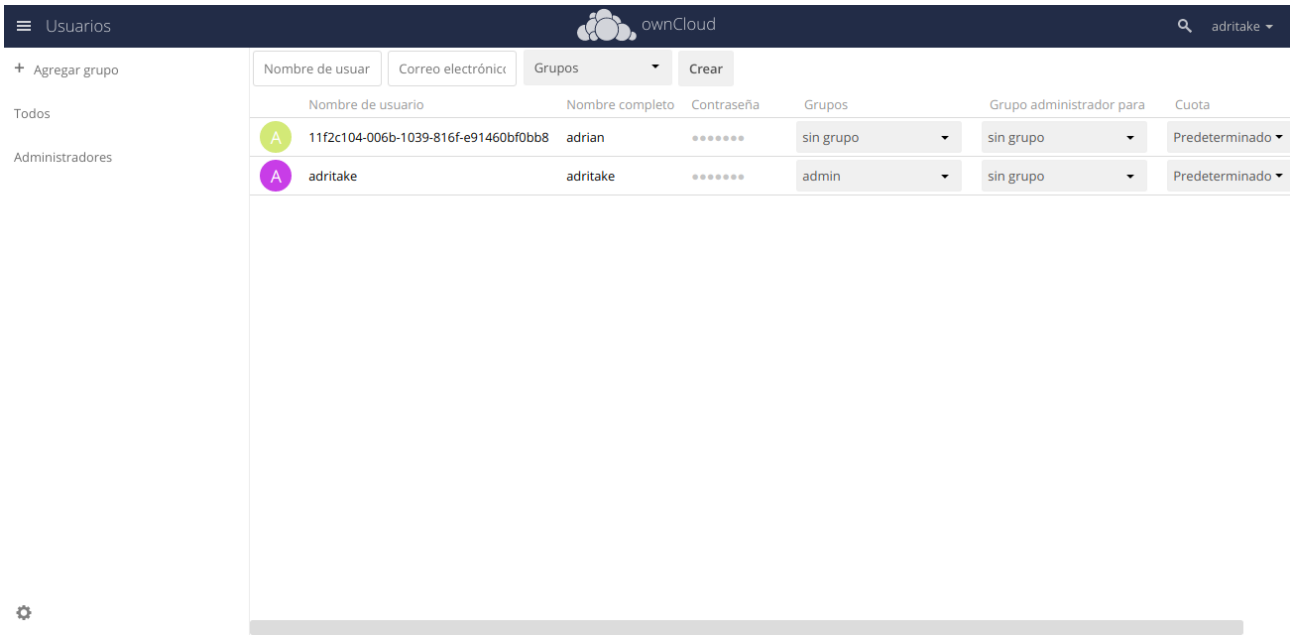


Introducimos la IP del docker de LDAP y el puerto usado, si se pulsa *detectar puerto* debería aparecer el puerto 389 también. Introducimos el DIT *cn=admin,dc=example,dc=org. L*a contraseña por defecto es *admin* y la base DN es *dc=example,dc=org.* Si todo funciona correctamente, abajo debería aparecer un círculo verde e indicar que la configuración es correcta.

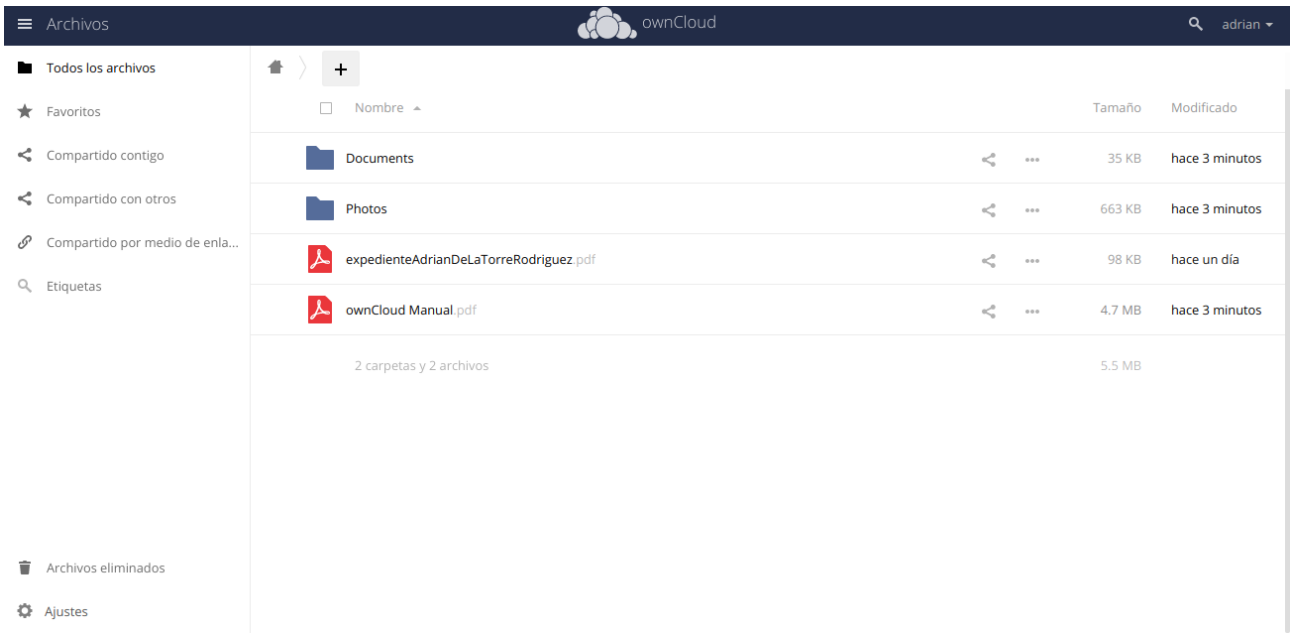
Tuve que cambiar una cosa a mano en la pestaña de Atributos de Inicio de sesión ya que no me aparecía por defecto el *uid* del usuario y tuve que introducir lo siguiente:



Con esto terminamos la configuración de LDAP. Si salimos e iniciamos sesión con el usuario creado con el archivo *user.ldif* y con la contraseña que establecimos con el comando *ldappasswd*, ahora debería aparecer en la pestaña de usuarios el usuario administrador y el usuario creado en LDAP.



Por último vamos a intentar almacenar un archivo cualquiera en nuestros documentos para comprobar el funcionamiento de ownCLoud. Para ello arrastramos el archivo a la interfaz de ownCLoud y se sube automáticamente quedándose almacenado.



4. Manual de usuario

En este apartado se indican los pasos necesarios para reproducir la práctica. Primero hay que clonar el repositorio de la práctica:

```
git clone https://github.com/adritake/CC2-18-19.git
```

Instalar si no se tiene el CLI de Azure y hacer `az login`. A continuación creamos la MV con OwnCloud, para ello, dentro del repositorio ejecutar `bash crearMVOwncloud.sh`. Probar que funciona correctamente accediendo a la IP de la MV desde un explorador pero no se debe tocar nada de la instalación de OwnCloud de momento.

Ejecutamos `bash InstallMariaDBDocker.sh` para lanzar un docker con MariaDB en Azure. Una vez lanzado guardamos la IP y ejecutamos el siguiente comando para ejecutar un *bash* en ese docker:

```
az container exec --resource-group myResourceGroup --name ownCloudMV --exec-command "/bin/bash"
```

En el bash ejecutamos `mysql -p` y nos pedirá una contraseña que en este caso es *password*. Si se lanza mysql correctamente escribimos `CREATE TABLE` `owncloudb` para crear la base de datos de owncloud.

Para crear el docker de LDAP, ejecutar `bash InstallLDAPDocker.sh`, guardamos la IP y ejecutamos el siguiente comando que crea el usuario indicado en el archivo user.ldif:

```
ldapadd -H ldap://<IP> -x -D 'cn=admin,dc=example,dc=org' -w admin -c -f user.ldif
```

Cambiar la contraseña del usuario creado con:

```
sudo ldappasswd -H ldap://<IP> -s password -W -D "cn=admin,dc=example,dc=org" -x "cn=adrian,dc=example,dc=org"
```

Volvemos a acceder a la IP de OwnCloud para completar la instalación. Introducimos un nombre de usuario cualquiera y una contraseña para la cuenta de administrador. Seleccionamos la pestaña de MySQL/MariaDB y le indicamos el usuario *root* que es el que tiene MariaDB por defecto, la contraseña es la introducida en la variable de entorno cuando creamos el docker (en este caso *password*), la base de datos es la que creamos cuando accedimos al contenedor (*owncloudb*). Por último introducimos la IP del docker de MariaDB y el puerto que usa que es el 3306 en el formato <IP>:3306.

Una vez dentro de OwnCloud accedemos al Market y descargamos la aplicación LDAP Integration la cual delega la gestión de usuarios a un servidor LDAP. Una vez instalada, vamos a *ajustes* > *autentificación de usuario* y allí introducimos los datos del servidor LDAP (hay que tener el docker de LDAP funcionando previamente). Introducimos la IP del docker de LDAP y el puerto usado, si se pulsa *detectar puerto* debería aparecer el puerto 389 también. Introducimos el DIT *cn=admin,dc=example,dc=org. L*a contraseña por defecto es *admin* y la base DN es *dc=example,dc=org.* Si todo funciona correctamente, abajo debería aparecer un círculo verde e indicar que la configuración es correcta.

Es posible que haya que introducir en la pestaña de Atributos de Inicio de sesión lo siguiente:

```
(&(|(uid=%uid)))
```

Con esto ya está todo configurado y se debería de poder iniciar sesión con el usuario creado en LDAP y subir y almacenar archivos.

5. Mejoras

Las mejoras introducidas para esta práctica es la automatización de todos los comandos que hay que introducir para crear los contenedores gracias al uso de scripts de shell. Tan solo basta con clonar el repositorio de la práctica y ejecutar todo lo que indica en el README.

https://github.com/adritake/CC2-18-19

6. Bibliografía

[1] https://github.com/fjbaldan/PracticasCC

[2] https://docs.docker.com/

[3] https://hub.docker.com/_/owncloud

[4] https://www.digitalocean.com/community/tutorials/como-instalar-y-usar-docker-en-ubuntu-18-04-1-es

[5] https://hub.docker.com/_/mariadb

[6] https://docs.microsoft.com/en-us/cli/azure/container?view=azure-cli-latest

[7] https://github.com/osixia/docker-openldap