Práctica 1
Diseño y despliegue de aplicaciones sobre plataformas cloud

Cloud Computing: Servicios y aplicaciones

30 de abril de 2019

Pedro Manuel Gómez-Portillo López

gomezportillo@correo.ugr.es

Índice

1. Descripción del problema

El objetivo de la primera práctica de la asignatura es poner en uso los conocimientos adquiridos en relación con el uso de plataformas IaaS (*Infraestructure as a Service)* y SaaS (*Software as a Service*) de Cloud Computing. Para ello, se desplegarán tres servicios, cada uno de ellos conteniendo los siguientes servicios.

- Alojamiento, sincronización y compartición de archivos
- Base de datos
- Autenticación usando LDAP

Para el primer servicio se ha elegido **Owncloud**[1]. Owncloud es una aplicación *open source* que permite el almacenamiento de archivos en línea. Puede verse como un sustituto de Dropbox que el usuario puede instalar en un servidor propio para mantener él mismo sus archivos. Se usará la versión 9.0.

Para el segundo servicio se utilizará **MySQL**[2]. MySQL es un sistema de gestión de bases de datos relacional desarrollado bajo las licencias pública general y licencia comercial por Oracle Corporation. Actualmente está considerada como la base datos de código abierto más popular del mundo, y la segunda más popular en general por detrás de Oracle[3]. Se usará la versión 8.0.

[1] https://owncloud.org/
[2] https://www.oracle.com/mysql/
[3] https://db-engines.com/en/ranking

Y para el último servicio se utilizará **LDAP**, o *Lightweight Directory Access Protocol*, es un protocolo a nivel de aplicación que permite el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red[4]. Usaremos la implementación *open source* **OpenLDAP**[5] en su versión 2.4.

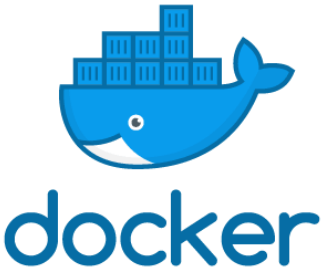
Tanto el código del proyecto como una copia de esta documentación pueden encontrarse en el repositorio en GitHub que se ha utilizado para la práctica.

https://github.com/gomezportillo/cc2

[4] https://es.wikipedia.org/wiki/Protocolo_Ligero_de_Acceso_a_Directorios
[5] https://www.openldap.org/

2. Diseño

Para realizar esta práctica se ha decido hacer uso de **contenedores**, concretamente de **Docker**, que es un proyecto de código abierto que automatiza el despliegue de aplicaciones proporcionando una capa adicional de abstracción y automatización de virtualización de aplicaciones[6].



Por otro lado, como plataforma PaaS se ha elegido **Azure**. Azure es un producto de Microsoft que alquila ordenadores a medida para sus usuarios. Concretamente, en lugar de hacer uso de máquinas virtuales sobre las que instalar Docker y desplegar los contenedores de los servicios, se usarán los contenedores nativos que ofrece Azure[7]. El motivo de esta decisión ha sido que los contenedores son mucho más rápidos y baratos de usar que las máquinas virtuales.

Por tanto, el resultado final serán tres contenedores desplegados en Azure, cada uno de ellos conteniendo un servicio, e intercomunicados correctamente para que el sistema funcione adecuadamente.

El principal objetivo del Cloud Computing es garantizar la escalabilidad de sus aplicaciones, por lo que la principal ventaja de usar este paradigma es poder asegurar que aunque el cantidad de usuarios o el número de archivos que suben al sistema aumente, éste aguantará mientras nuestra tarjeta de crédito también lo haga.

[6] https://es.wikipedia.org/wiki/Docker_(software)
[7] https://azure.microsoft.com/es-es/services/container-instances/

3. Despliegue

En esta sección se describirán los pasos seguidos para realizar el despliegue del sistema. Aunque inicialmente se comenzó a realizar la práctica usando máquinas virtuales sobre las que correr Docker, se ha descubierto que se puede trabajar en Azure usando directamente contenedores, lo que es más rápido y barato. Además, a lo largo del proyecto se utilizará la consola[8] nativa en línea de Azure.

Para estos comandos partimos de haber creado previamente un grupo de recursos llamado *CC2*

3.1. Owncloud

Para desplegar el servidor Owncloud en un contenedor usaremos la imagen oficial ya creada. Para ello, deberemos ejecutar en Azure

```
az container create --resource-group CC2 \
                    --dns-name-label cc2-owncloud \
                    --name owncloud docker \
                    --image owncloud \
                    --ports 80
```

O, alternativamente, ejecutar `make owncloud` en el directorio raíz del proyecto. Tras la ejecución de este comando Azure instanciará un contenedor al que redirigirá el tráfico de red entrante por el puerto 80 al contenedor, que iniciará y nos proporcionará su IP; entrando en esta IP podremos acceder a nuestro servidor Owncloud. Ahora, como aún no tiene base de datos en la que guardar la información, pasaremos a crearla

3.2. MySQL

El problema que se ha encontrado con Azure es que, como las imágenes existentes no están correctamente configuradas y nos interesa conseguir la mayor automatización posible, crearemos nosotros la nuestra.

Para ello se ha creado el siguiente Dockerfile cuya imagen se ha subido al repositorio en DockerHub que puede verse en el siguiente enlace.

https://hub.docker.com/r/pedroma1/docker-mysql

[8] https://shell.azure.com/

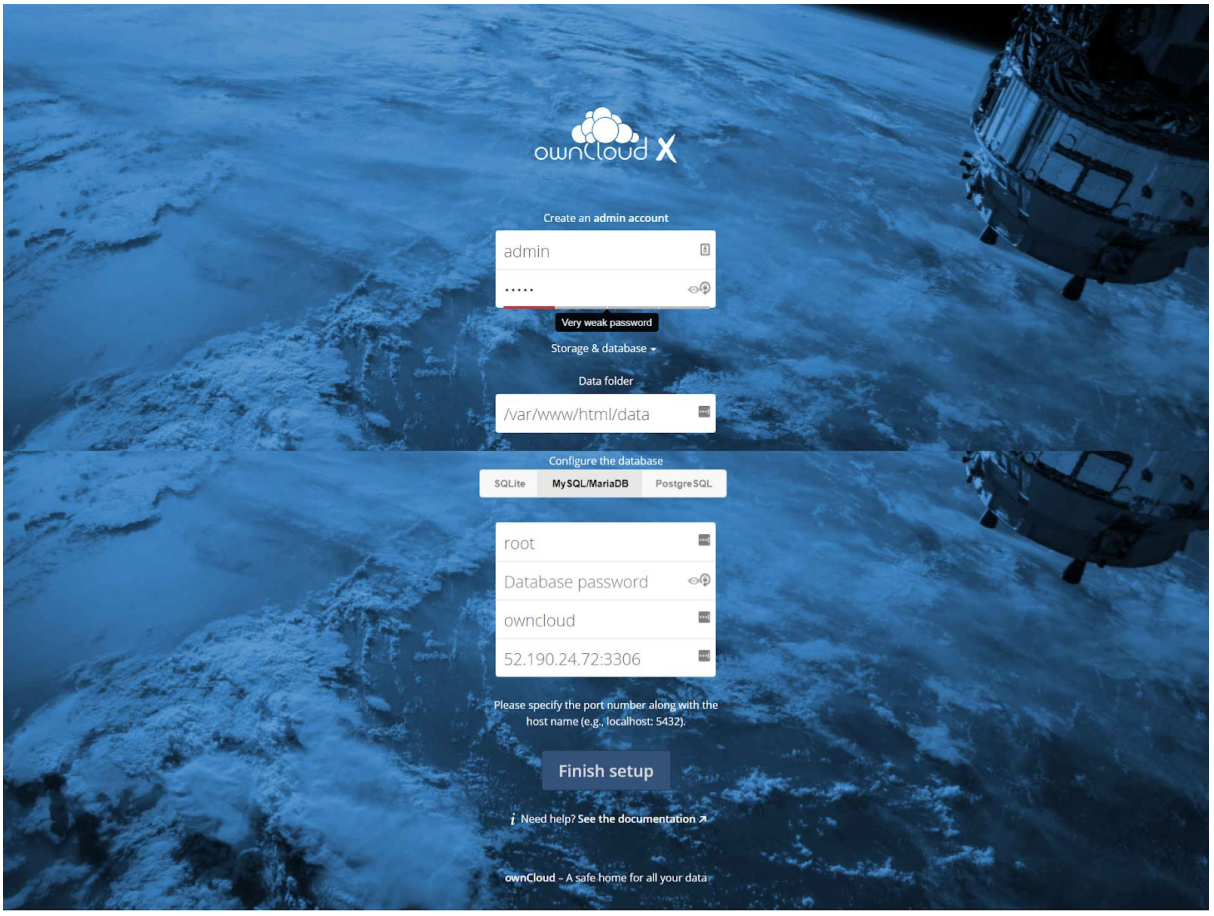
```
FROM ubuntu:latest

WORKDIR /

RUN apt-get update && \
    apt-get install -y python-mysqldb mysql-server && \
    service mysql start && \
    mysql -e "CREATE DATABASE owncloud" && \
    mysql -e "CREATE USER 'root'@'%' IDENTIFIED BY ''" && \
    mysql -e "GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' WITH GRANT
OPTION" && \
   sed -i 's/bind-address/#bind-address/'
           /etc/mysql/mysql.conf.d/mysqld.cnf && \
   service mysql stop && \
   chown -R mysql:mysql /var/run/mysqld

VOLUME ["/var/lib/mysql"]

EXPOSE 3306

CMD ["mysqld_safe"]
```

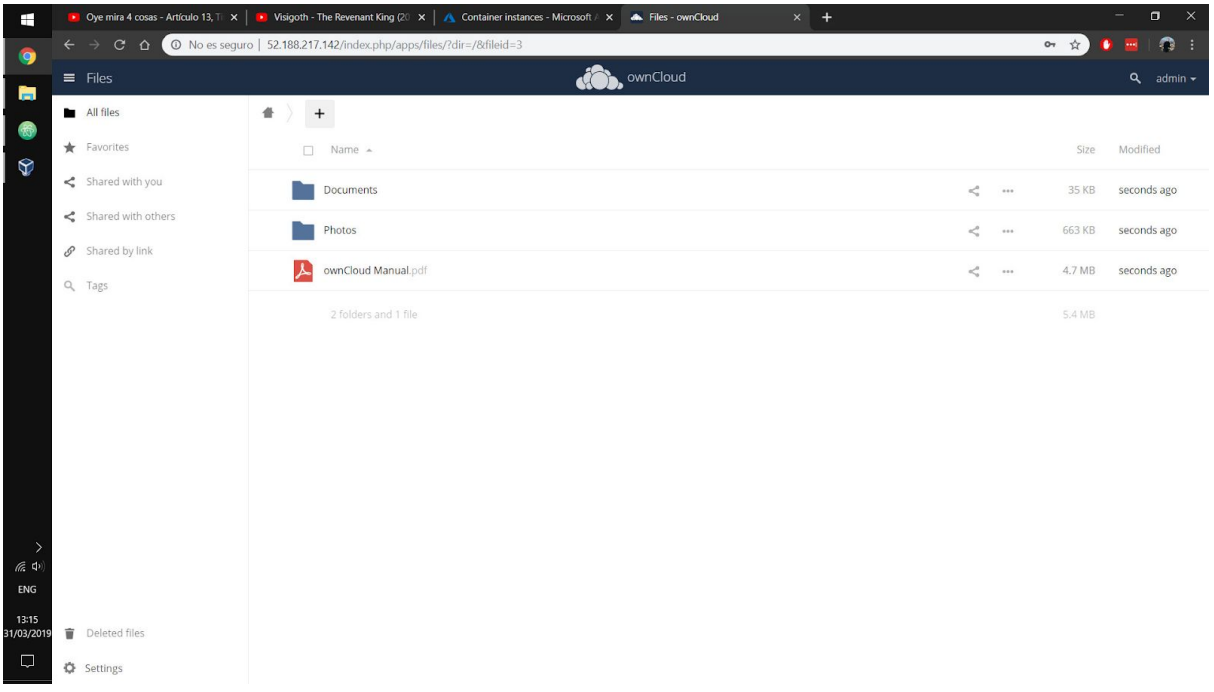
Ahora, ya que tenemos nuestra imagen subida y accesible desde internet podemos usarla para crear directamente un contenedor configurado. Para ello ejecutaremos el siguiente comando en Azure.

```
az container create --resource-group CC2 \
                    --dns-name-label cc2-mysql \
                    --name mysqldocker \
                    --image pedroma1/docker-mysql \
                    --ports 3306
```

O alternativamente podemos ejecutar `make mysql` para ello. Ahora ya podemos acceder al servidor Owncloud y configurarlo con la IP y los datos con los que hemos creado el contenedor de MySQL, además de crear un usuario y contraseña. Una vez hecho, seremos redirigidos a una página similar en la que podremos hacer login con dichos datos y acceder a nuestros archivos (el fondo de la primera imagen sale cortado por la extensión que he utilizado para capturar la pantalla completa)[9].

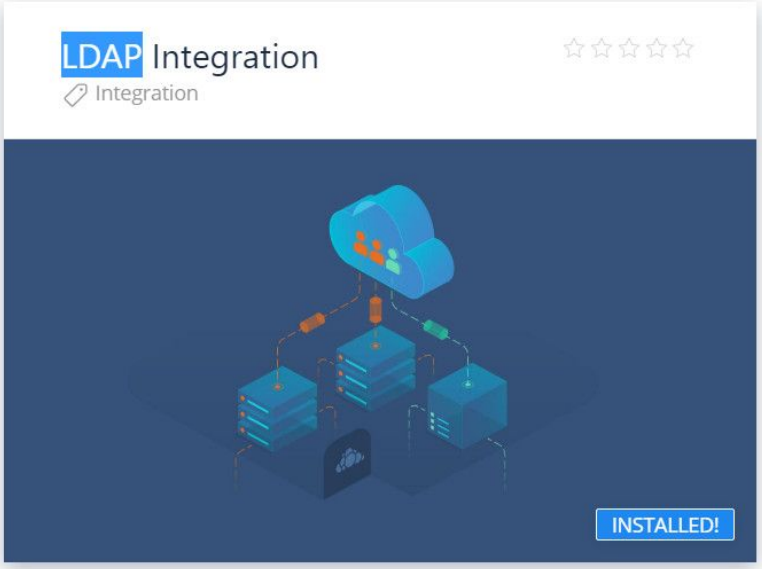
[9] https://chrome.google.com/webstore/detail/fdpohaocaechififmbbbbbknoalclacl


ownCloud X
Create an admin account
admin
Very weak password
Storage & database
Data folder
/var/www/html/data
Configure the database
SQLite
MySQL/MariaDB
PostgreSQL
root
Database password
owncloud
52.190.24.72:3306
Please specify the port number along with the host name (e.g., localhost: 5432).
Finish setup
Need help? See the documentation
ownCloud – A safe home for all your data


Files - ownCloud
No es seguro | 52.188.217.142/index.php/apps/files/?dir=/&fileid=3
Files
ownCloud
admin
All files
Favorites
Shared with you
Shared with others
Shared by link
Tags
Deleted files
Settings
Name
Size
Modified
Documents
35 KB
seconds ago
Photos
663 KB
seconds ago
ownCloud Manual.pdf
4.7 MB
seconds ago
2 folders and 1 file
5.4 MB
ENG
13:15
31/03/2019

3.3. OpenLDAP

Para incluir LDAP en nuestros servicios lo primero que haremos será instalar el plugin de Owncloud; para eso, vamos al mercado de Owncloud, buscamos la aplicación *LDAP Integration* y la instalamos en nuestro servidor.



Ahora llega el momento de crear el contenedor o la máquina virtual de LDAP. Inicialmente, como excusa para desplegarlo de manera diferente al resto de servicios de la práctica, se usaron máquinas virtuales sobre las se instalaron Docker y se ejecutó el contenedor pertinente, pero es un proceso muchísimo más lento y engorroso que usar contenedores directamente, y como en esta práctica se está primando la velocidad y la automatización de los despliegues finalmente se han utilizado los contenedores nativos de Azure e incluso se ha rehecho esta parte de la documentación.

Ahora debemos elegir el contendor; principalmente podemos usar la imagen vista en clase, `larrycai/openldap`, o la de *Osixia*, `osixia/openldap`. Como varios compañeros han tenido problemas con la primera, preventivamente usaremos la segunda. Para ello, lo primero que haremos será crear el contenedor.

```
az container create --resource-group CC2 \
                    --dns-name-label cc2-ldap \
                    --name ldapdocker \
                    --image osixia/openldap \
                    --ports 389
```

Una vez que ya tenemos el contenedor desplegado pasaremos a crear un usuario. Como el contenedor de *osixia* no trae ningún OU, lo primero que haremos será crear uno. El archivo `new_ou.ldif` tiene la información necesaria para ello.

```
dn: ou=Users,dc=example,dc=org
ou: Users
objectClass: top
objectClass: organizationalUnit
description: Parent object of all PEOPLE accounts
```

A continuación ejecutamos el siguiente comando, o `make add-ldap-ou IP=X`.

```
ldapadd -H ldap://$(IP) -x -D cn=admin,dc=example,dc=org -w admin
-c -f new_ou.ldif
```

A continuación crearemos el usuario. Para ello, se ha creado el archivo `new_user.ldif` con la información necesaria para ello.

```
dn: cn=pedro,ou=Users,dc=example,dc=org
objectClass: top
objectClass: account
objectClass: posixAccount
objectClass: shadowAccount
cn: pedro
uid: pedro
uidNumber: 16859
gidNumber: 100
homeDirectory: /home/pedro
loginShell: /bin/bash
gecos: pedro
userPassword: pedro
shadowLastChange: 0
shadowMax: 0
shadowWarning: 0
```

Ahora necesitamos añadir este usuario. Para ello basta con ejecutar el siguiente comando en una máquina con el paquete `ldap-utils` instalado y usando la IP pública del contenedor con OpenLDAP. Con Osixia la contraseña por defecto es *admin*[10].

[10] https://github.com/osixia/docker-openldap#quick-start

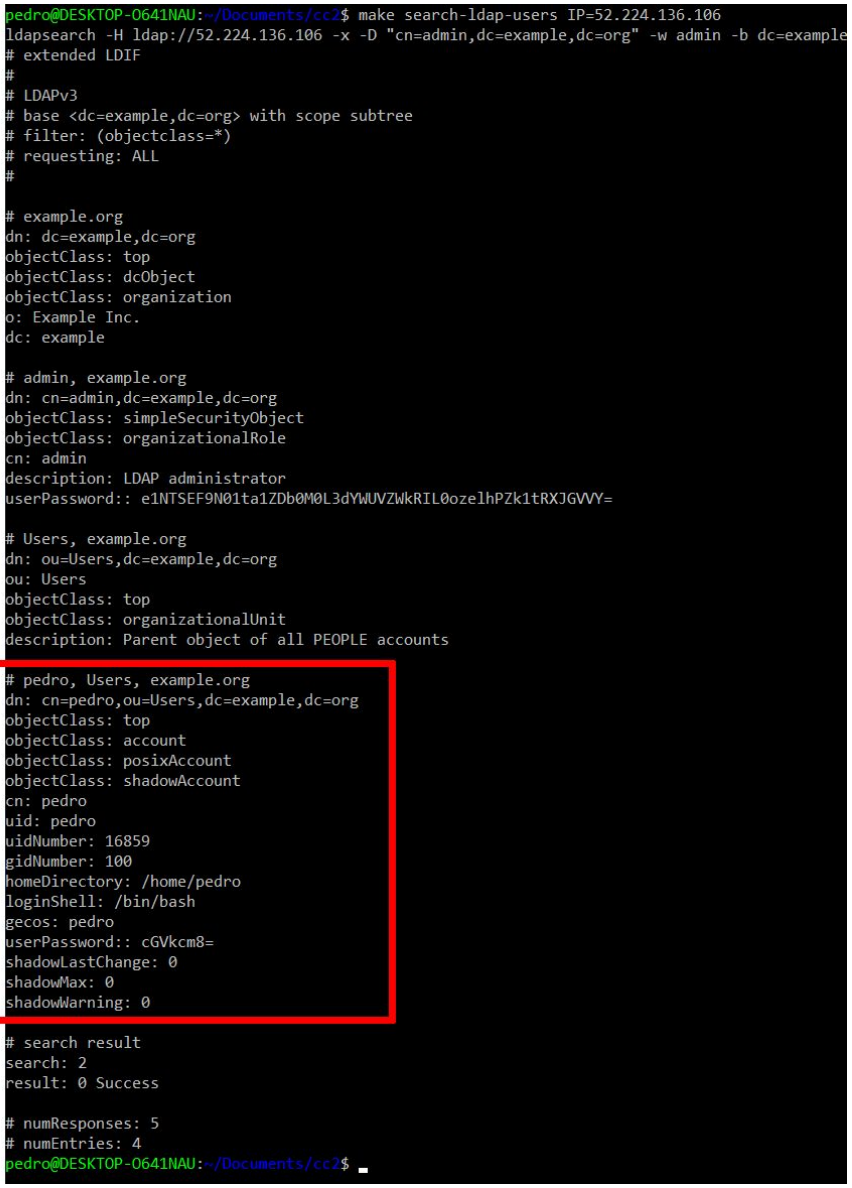
```
ldapadd -H ldap://$(IP) -x -D "cn=admin,dc=example,dc=org" \
     -w admin -c -f new_user.ldif
```

También es posible usar el Makefile para ejecutar `make add-ldap-user IP=X` en el directorio raíz del proyecto para añadir el usuario.

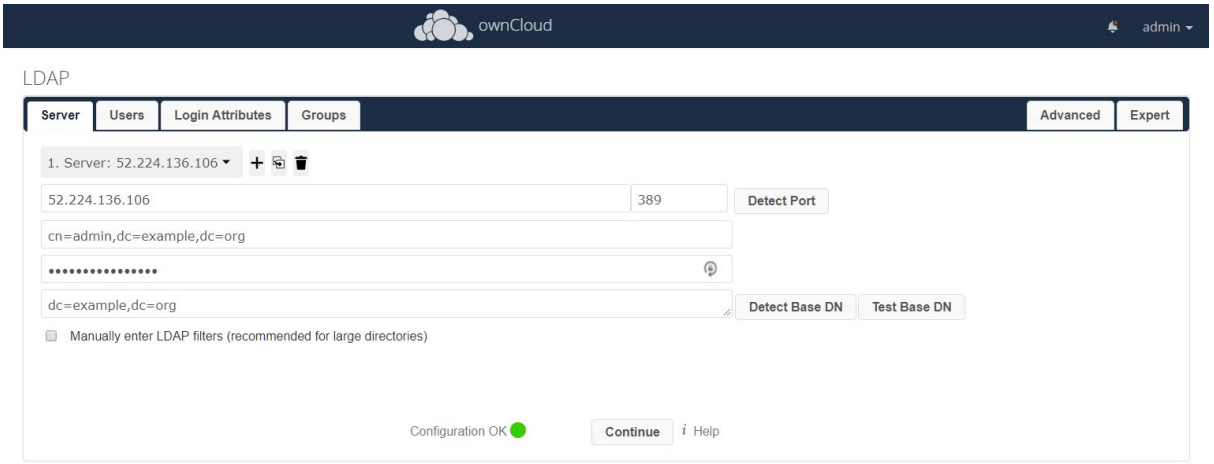
Ahora buscaremos el usuario que acabamos de añadir. Para ello, ejecutamos lo siguiente.

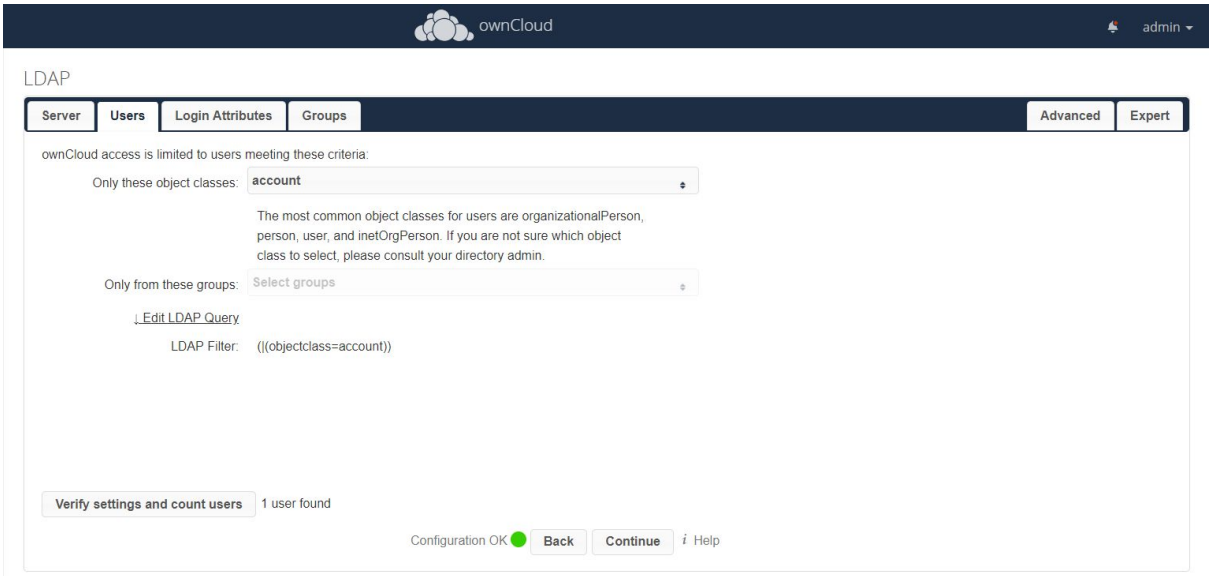
```
ldapsearch -H ldap://$(IP) -x -D "cn=admin,dc=example,dc=org" \
-w admin -b dc=example,dc=org
```

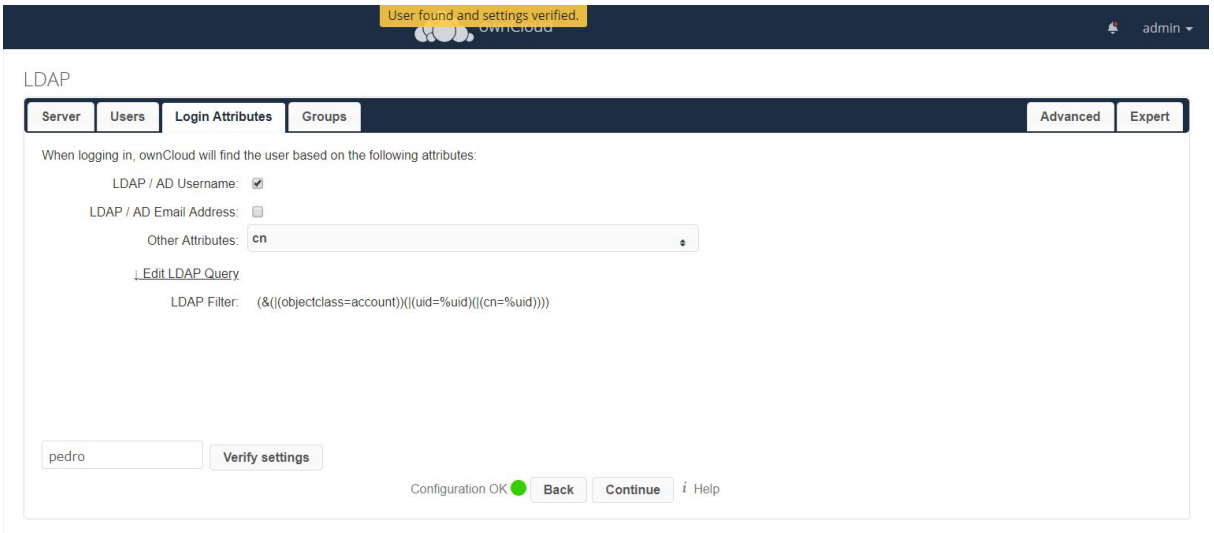
O también podemos ejecutar directamente `make search-ldap-users IP=X`. La ejecución de dicho comando debería dar una salida parecida por pantalla.

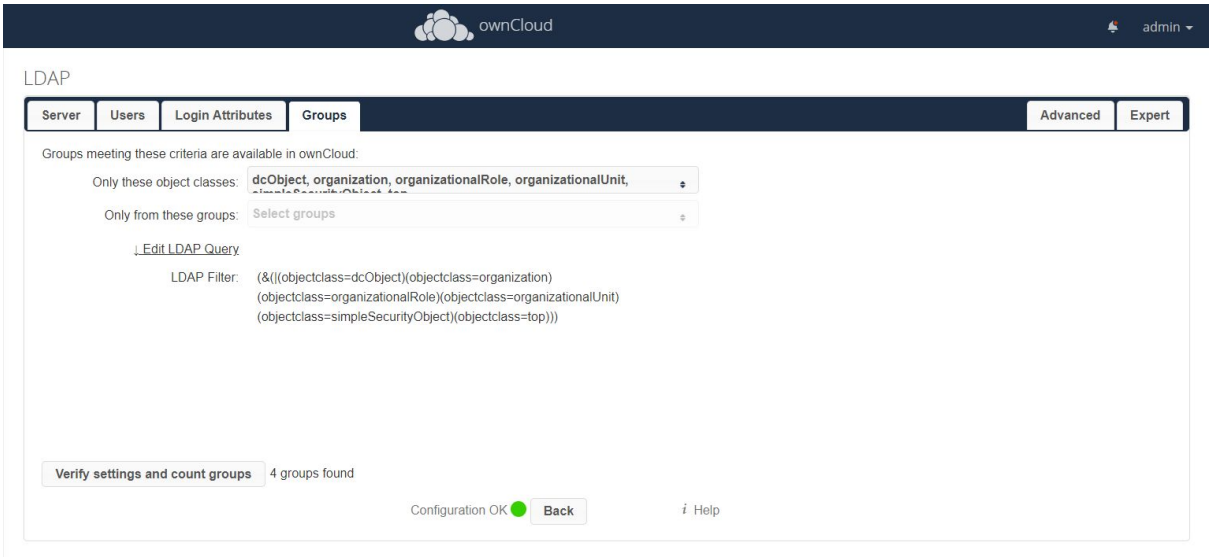


Una vez creado el usuario, lo siguiente que haremos será entrar en nuestro servidor Owncloud y acceder a *Settings>Admin>User authentication*. Una vez allí rellenaremos los campos que aparecen con la información de LDAP necesaria como aparece a continuación.

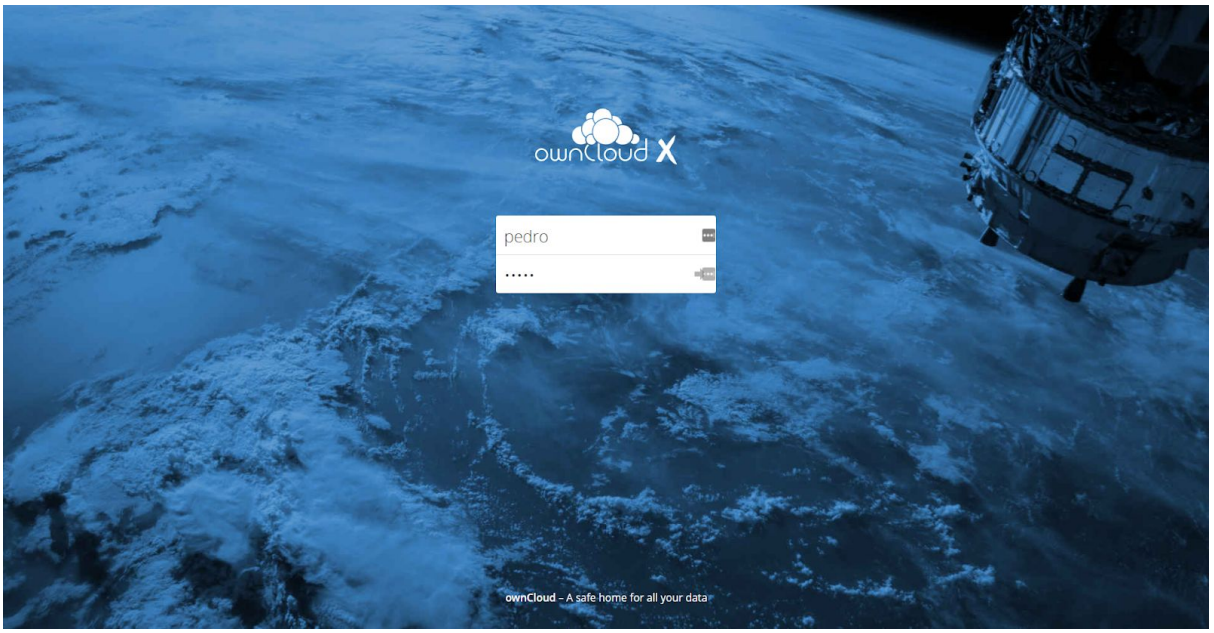


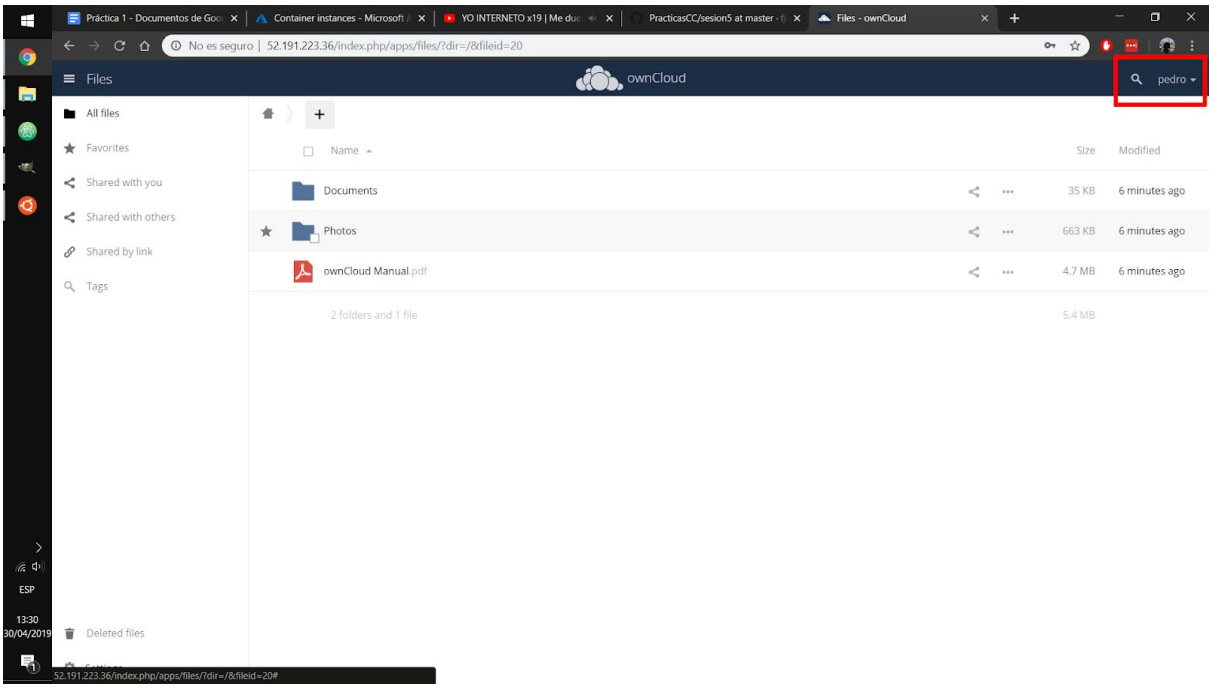




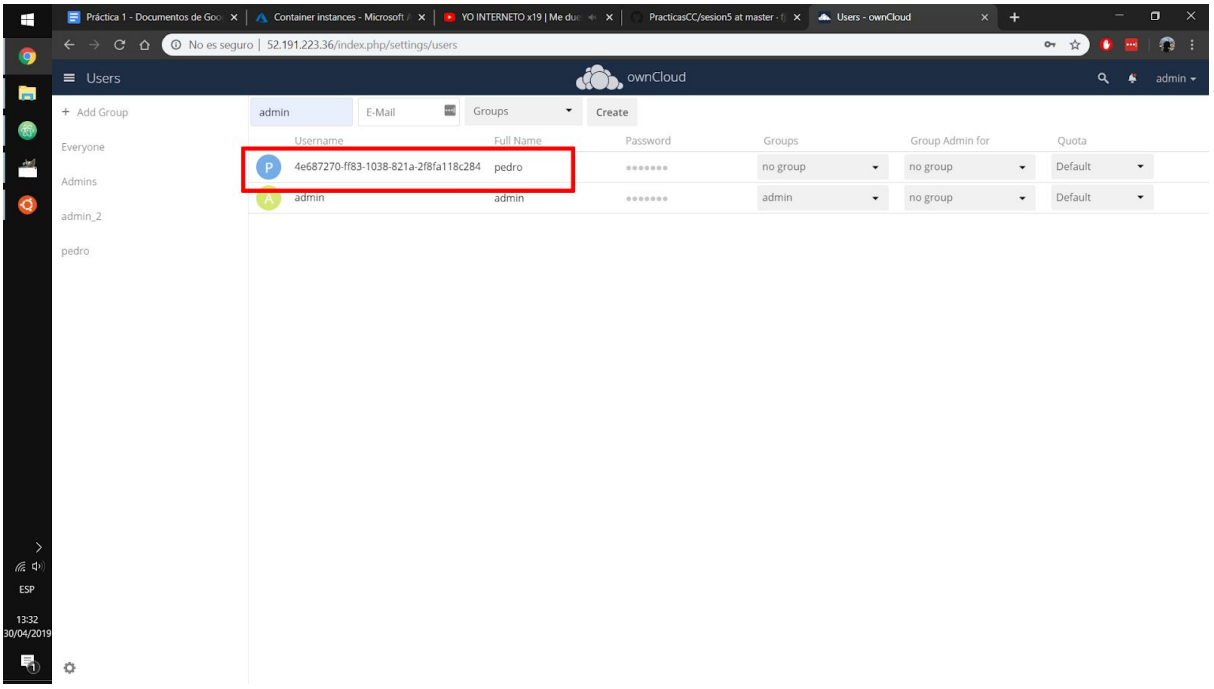


Ahora ya podemos cerrar sesión como administrador para volver a la pantalla de inicio de sesión e iniciarla con el par usuario:contraseña que hemos creado antes, en este caso `pedro:pedro`.





Y ahora si volvemos a hacer login como administrador podemos acceder al menú de usuarios para verlo.



4. Manual de usuario

En esta sección se hará distinción de los dos posibles tipos de usuarios que pueden usar el sistema; por un lado está el técnico que se encarga de desplegar los servicios, y por otro el usuario que usa Owncloud.

4.1. Usuario técnico

Para facilitar al máximo los pasos que debería dar un técnico para desplegar el sistema, aparte de esta documentación, se ha incluido un Makefile para reducir el esfuerzo necesario para realizar el despliegue de los tres servicios. A continuación se explican los pasos necesarios.

1. Acceder al shell de Azure (https://shell.azure.com/).
2. Clonar el repositorio del proyecto (`git clone GITHUB/gomezportillo/CC2`).
3. `cd CC2 && make` para desplegar los tres servicios automáticamente correctamente configurados.
4. Acceder a una terminal con `ldap-utils`, clonamos el repositorio y ejecutamos `make create-ldap-ou && make create-ldap-user`.
5. Acceder a la URL de Owncloud.
    a. Configurar la base de datos con los datos del contenedor de MySQL.
    b. Acceder al mercado e instalar el plugin de LDAP.
    c. Por último, acceder a *Config>ADmin>User Authentication* y configurar LDAP.

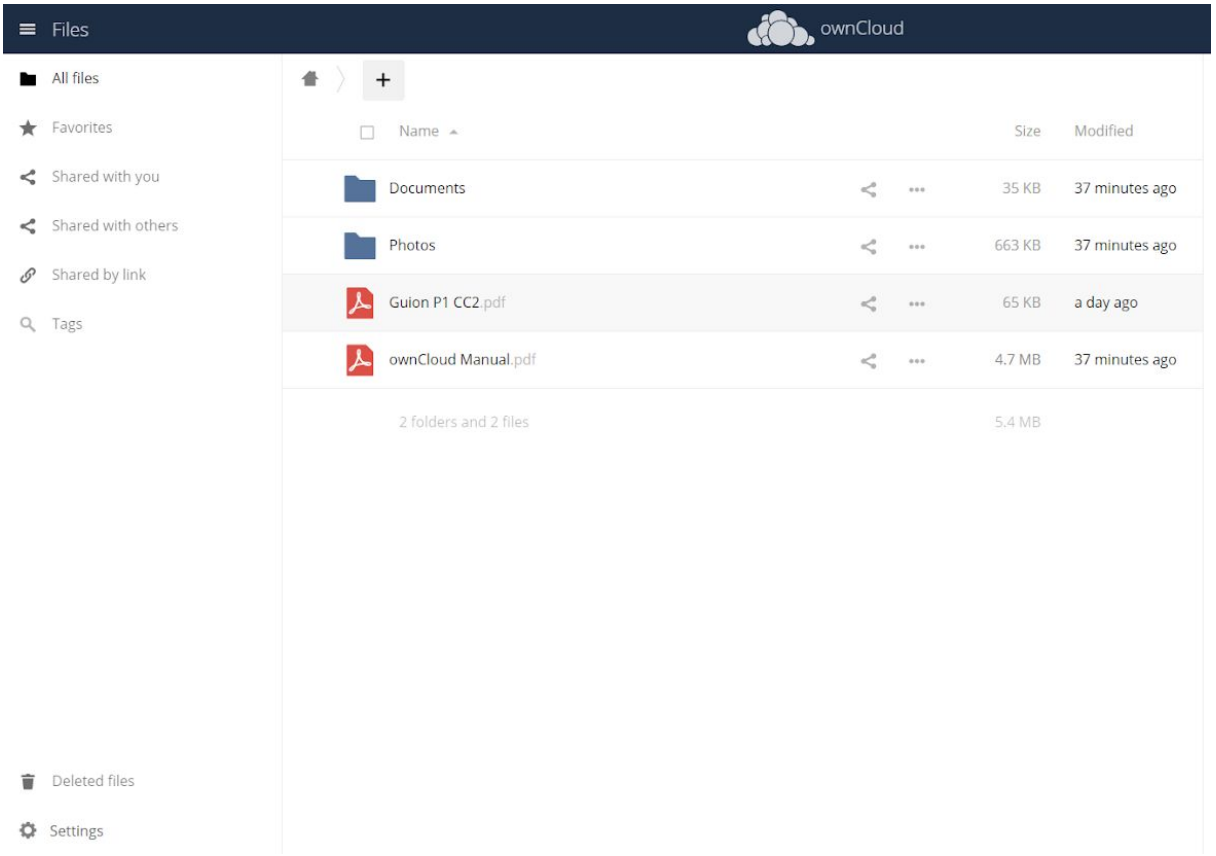
Por otro lado, si solo queremos desplegar uno de los servicios, podemos hacer con `make owncloud`, `make mysql` o `make ldap`. Además, podemos conectarnos a un contenedor con `make connect-to-container NAME=X` o a una máquina virtual con `make connect-to-vm IP=X`.

4.2. Usuario normal

Lo primero que necesitamos será crearlo. Para ello, deberíamos duplicar el archivo `new_user.ldif` y editarlo con los valores que necesitamos. Tras ello, basta con ejecutar `make add-ldap-users`.

Tras esto, usar Owncloud es muy fácil para un usuario. Basta con arrastrar los archivos que queremos subir a la pestaña del navegador y se subirán, y para descargarla basta con hacer doble click sobre ella.

En la captura inferior puede verse el resultado tras subir el guión de esta práctica.



5. Mejoras y conclusiones

A lo largo de la realización del proyecto se ha visto lo repetitivo que puede resultar arrancar una y otra vez las mismas máquinas virtuales con los mismos parámetros, por lo que en un esfuerzo por automatizar lo máximo posible el despliegue de todos los servicios la principal mejora incluida ha sido la un Makefile. Este esfuerzo sería prácticamente obligatorio en un proyecto de más escala, ya que se vuelve mucho más sencillo poder desplegarlo todo con un solo botón.

Además, se ha añadido el comando `make remove-all-containers` para borrarlos cómodamente al terminar de trabajar. A continuación se adjunta el Makefile completo.

```
RG := CC2
owncloud_NAME := owncloud‌docker
mysql_NAME           := mysqldocker
ldap_NAME            := ldapdocker

all: owncloud mysql ldap

owncloud:
     az container create --resource-group $(RG) \
                         --dns-name-label cc2-owncloud \
                         --name $(owncloud_NAME) \
                         --image owncloud \
                         --ports 80


mysql:
     az container create --resource-group $(RG) \
                         --dns-name-label cc2-mysql \
                         --name $(mysql_NAME) \
                         --image pedroma1/docker-mysql \
                         --ports 3306


ldap:
     az container create --resource-group $(RG) \
                         --dns-name-label cc2-ldap \
                         --name $(ldap_NAME) \
                         --image osixia/openldap \
                         --ports 389
```

```
connect-to-container:
	az container exec --resource-group $(RG) \
                  --name $(NAME) \
                  --exec-command "/bin/bash"


connect-to-vm:
	ssh azureuser@$(IP)


add-ldap-ou:
	ldapadd -H ldap://$(IP) -x -D cn=admin,dc=example,dc=org -w
admin -c -f new_ou.ldif


add-ldap-user:
	ldapadd -H ldap://$(IP) -x -D cn=admin,dc=example,dc=org -w
admin -c -f new_user.ldif


search-ldap-users:
	ldapsearch -H ldap://$(IP) -LL -b ou=Users,dc=example,dc=org
-x


remove-all-containers:
	az container delete -n $(owncloud_NAME) --resource-group
$(RG) -y
	az container delete -n $(mysql_NAME) --resource-group $(RG)
-y
	az container delete -n $(ldap_NAME) --resource-group $(RG) -y
```

En un intento extra por automatizar al máximo el despliegue se ha intentado configurar Owncloud para poder mandarle la información necesario de los contenedores de MySQL y OpenLDAP para su configuración, aunque finalmente no ha sido posible.

Por otro lado, se han tenido que solucionar muchos problemas hasta conseguir hacer acceder a Owncloud con los usuarios de LDAP.

En conclusión, esta práctica me ha servido para seguir aprendiendo sobre contenedores, servicios PaaS y automatización de despliegues, lo que sin duda me será necesario en mi futura vida laboral.

6. Bibliografía

- Repositorio de la primera parte de la asignatura
    - https://github.com/gomezportillo/apolo
- Repositorio de la asignatura
    - https://github.com/fjbaldan/PracticasCC
- Documentación de Docker
    - https://docs.docker.com/
- Repositorio de Osixia/ldap
    - https://github.com/osixia/docker-openldap