Práctica 4. Dockerizar un servidor FTP

Si buscamos *vsftpd* en Docker Hub encontraremos que la imagen más utilizada es fauria/vsftpd

En la página de Docker Hub tenemos toda la información necesaria para usar este contenedor. Tendrás que consultarla así que échale un primer vistazo.

Intentar usar un contenedor usando todas sus opciones desde el principio es garantía de no conseguirlo. Vamos a ir poco a poco. Nuestro objetivo final será crear un contenedor dockerizado con las mismas características que el contenedor de la práctica 2, que recordamos era:

1. Deja que sólo puedan conectarse los usuarios locales.
2. Habilita la carga de archivos.
3. Haz que los archivos se suban con umask 022.
4. Habilita chroot con la opción 2, es decir, que cada usuario del sistema se conecte a `/home/$USER/ftp`
5. Restringe para que solo `userftp` pueda conectarse por FTP
6. Agrega un archivo `pruebaftp.txt` en `/home/userftp/ftp/upload/` para usar en las pruebas.

> **Info**
>
> La imagen docker `fauria/vsftpd` está configurada para usar "usuarios virtuales" en lugar de usuarios locales. Pero como la configuración ya está hecha en el contenedor no nos preocupará. Simplemente hemos de saberlo para cuando veamos ciertas configuraciones en vsftpd.conf que no conozcamos.

Vamos a empezar por intentar correr un contentedor que permita un acceso FTP activo simple. Viendo las opciones que nos da la página de "fauria/vsftpd" probaremos esto:

```
docker run \
    --rm \
    -p 21:21 -p 20:20 \
    -d \
    --name pruebavsftpd \
    fauria/vsftpd
```

La opción -rm borrará el contenedor al pararlo. Como estamos en pruebas nos interesa esa opción. Corre el contenedor y luego, siguiendo las instrucciones del contenedor, ejecuta lo siguiente para obtener el usuario y contraseña de acceso:

```
$ docker logs pruebavsftpd
    **************************************************
    *                                                *
    *    Docker image: fauria/vsftpd                 *
    *    https://github.com/fauria/docker-vsftpd     *
    *                                                *
    **************************************************

    SERVER SETTINGS
    ---------------
    · FTP User: admin
    · FTP Password: nfcf4ZoxNRsV2c3L
    · Log file: /var/log/vsftpd/vsftpd.log
    · Redirect vsftpd log to STDOUT: No.
```

Ahora abre Filezilla y establece un conexión con el contenedor. Recuerda que si está corriendo en una EC2 de AWS habrás de tener abiertos los puertos 20 y 21. Y en Filezilla tendrás que establecer una conexión FTP (no FTPS ni sFTP) y en opciones de transferencia establecer el modo "Activo".

Una vez establecida la conexión, ¿esta activo el chroot (puedes moverte por todos los directorios del servidor)? ¿Puedes subir archivos al servidor?

Bueno, ya tenemos algunas cosas claras. Pero se trata de un servidor FTP. No nos interesa que los archivos a descargar o subir estén dentro del contenedor y se borren cuando este se elimine. Así que vamos a pararlo pero montando el directorio de datos en un directorio de nuestro host. Crearemos un directorio `datosftp` y lo montaremos como dicen las instrucciones del contenedor.

```
mkdir datosftp
docker run \
    --rm \
    -p 21:21 -p 20:20 \
    -d \
    --name pruebavsftpd \
    -v /home/admin/datosftp:/home/vsftpd \
    fauria/vsftpd
```

Vuelve a conectarte con Filezilla. Recuerda recuperar la nueva contraseña de admin. Observa en la máquina host como se crea un directorio `/home/admin/datosftpd/admin` que alojará los datos. Con filezilla envía un fichero al directorio raíz donde te has conectado. Comprueba en el host que aparece dicho fichero en `/home/admin/datosftpd/admin`. Ya podemos parar nuevamente el contenedor.

Vamos ahora con una configuración más completa en la que crearemos un usuario ftp y activaremos el modo pasivo:

```
docker run \
    --rm \
    -e FTP_USER=userftp \
    -e FTP_PASS=ieselcaminas \
    -e PASV_MIN_PORT=21100 \
    -e PASV_MAX_PORT=21110 \
    -p 21:21 -p 21100-21110:21100-21110 \
    -d \
    --name pruebavsftpd \
    -v /home/admin/datosftp:/home/vsftpd \
    fauria/vsftpd
```

Atención

Antes de probarlo, fíjate que hemos abierto en el servido los puertos 21100 a 21110. Por tanto, si nuestro docker corre en una EC2 de AWS deberemos modificar el grupo de seguridad para permitir el acceso a dicho rango de puertos.

Prueba ahora a conectarte con `userftp` password `ieselcaminas` y modo pasivo desde Filezilla. Si todo va bien, prueba a enviar un fichero y después comprueba en el host que existe dicho fichero en `/home/admin/datosftp/userftp/` . Comprueba sus permisos. ¿Qué permisos tiene?

Con esto ya tenemos "casi" todo lo que dijimos en un principio. Y digo casi, porque no hemos tenido en cuenta los permisos con los que se suben los archivos. En la pregunta anterior comprobaríamos que los permisos del fichero subido son -rw------- . Veamos por qué. Si revisamos la documentación del contenedor vemos que hay 2 variables para definir los permisos de los archivos subidos:

- **Variable name: FILE_OPEN_MODE**
- Default value: 0666
- Accepted values: File system permissions.
- Description: The permissions with which uploaded files are created. Umasks are applied on top of this value. You may wish to change to 0777 if you want uploaded files to be executable.
- **Variable name: LOCAL_UMASK**
- Default value: 077
- Accepted values: File system permissions.
- Description: The value that the umask for file creation is set to for local users. NOTE! If you want to specify octal values, remember the "0" prefix otherwise the value will be

treated as a base 10 integer!

Es decir, el fichero subido se crea con permisos 0666 ó rw-rw-rw. Y luego se hace un AND con la LOCAL_UMASK negada. Como LOCAL_UMASK=077, la negada es 700 ó 111000000. Por tanto, el resultado de rw-rw-rw AND 11100000000 = rw-------

Pero nosotros queríamos que nuestra UMASK fuera 022, así que pasaremos la siguiente variable:

```
docker run \
    -e FTP_USER=userftp \
    -e FTP_PASS=ieselcaminas \
    -e PASV_MIN_PORT=21100 \
    -e PASV_MAX_PORT=21110 \
    -e LOCAL_UMASK=022 \
    -p 21:21 -p 21100-21110:21100-21110 \
    -d \
    --name vsftpdsincifrado \
    -v /home/admin/datosftp:/home/vsftpd \
    fauria/vsftpd
```

Comprueba ahora que si subes un fichero sus permisos serán rw-r--r-- y con esto ya tenemos todo lo que hicimos en el servidor vsftp sin cifrado pero dockerizado. Fíjate que le he quitado el `-rm` y le he cambiado el nombre al contenedor por `vsftpdsincifrado`.

Warning

En esta práctica hemos creado un usuario en el docker run usando variables de entorno. Pero podría ocurrir que quisiéramos añadir más usuarios una vez el contenedor ya creado. En la página del contenedor fauria/vsftpd en Docker Hub tenéis la forma de hacerlo usando un comando `docker exec`

Dockerizando servidor vsftpd con CIFRADO

Si recordamos de la "Práctica 3. Configuración de servidor FTP con Cifrado" para activar FTPS en vsftpd debíamos hacer 2 cosas:

- Generar un certificado autofirmado que se guarda en un fichero .pem (le llamaremos vsftpd.pem)
- Modificar el fichero de configuración `vsftpd.conf` con las directivas que activan el cifrado SSL

Para hacer esto en un contenedor podríamos plantearnos varias estrategias partiendo de que tenemos los ficheros vsftpd.pem y vsftpd.conf en el host y los queremos copiar al interior del contenedor

1. Crear un contenedor como el anterior. Después copiar en su interior los 2 ficheros con comandos `docker cp vsftpd.conf vsftpdconcifrado:/etc/vsftpd/vsftpd.conf` y `docker cp vsftpd.pem vsftpdconcifrado:/etc/vsftpd/vsftpd.pem`
2. Crear un Dockerfile que genere una imagen copiando esos 2 ficheros a su interior. Después crear el contenedor, con los mismos parámetros a partir de la imagen creada

> **Warning**
>
> Aunque ambas soluciones parecen válidas podéis comprobar que la primera no funciona. Una vez copiados los archivos al interior el contenedor no arranca correctamente. No es la primera vez que comprobamos que algunas imágenes docker solo nos permiten hacer ciertas cosas si antes creamos nuestra propia imagen con un `docker build`. Así que ahora y en lo sucesivo usaremos esa estrategia.

Por tanto, vamos a crear primero una imagen propia a partir de `fauria/vsftpd` copiando en su interior los ficheros que necesitemos. Crearemos un directorio `practicavsftpd` y entraremos dentro.

Primero crearemos el fichero vsftpd.pem con el certificado:

```
openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout vsftpd.pem -out
vsftpd.pem
```

Luego copiaremos el vsftpd.conf del contenedor sin cifrado en el host para después modificarlo:

```
docker cp vsftpdsincifrado:etc/vsftpd/vsftpd.conf .
sudo nano vsftpd.conf
```

Añadimos las líneas que vimos en la "Práctica 3. Configuración de servidor FTP con Cifrado" que habilitaban el cifrado

```
rsa_cert_file=/etc/vsftpd/vsftpd.pem
rsa_private_key_file=/etc/vsftpd/vsftpd.pem
ssl_enable=YES

allow_anon_ssl=NO
force_local_data_ssl=YES
force_local_logins_ssl=YES
ssl_tlsv1=YES
ssl_sslv2=NO
ssl_sslv3=NO
require_ssl_reuse=NO
ssl_ciphers=HIGH
```

Fíjate que las vamos a copiar en `/etc/vsftpd/` en lugar de la ruta donde la copiábamos en la Práctica3. Da igual dónde siempre que el COPY que hagamos en el Dockerfile sea coherente.

Ahora creamos el Dockerfile con este contenido

```
FROM fauria/vsftpd

# Copia tu archivo vsftpd.conf y los certificados
COPY vsftpd.conf /etc/vsftpd/vsftpd.conf
COPY --chown=root:root vsftpd.pem /etc/vsftpd/vsftpd.pem
```

Fíjate que hemos usado una opción en COPY para que vsftpd.pem se copie al interior del contenedor siendo el propietario root del grupo root. Esto es necesario para el certificado.

Creamos nuestra imagen

```
docker build -t mivsftpd .
```

Y creamos nuestro contenedor a partir de la imagen creada:

```
docker run \
    -e FTP_USER=userftp \
    -e FTP_PASS=ieselcaminas \
    -e PASV_MIN_PORT=21100 \
    -e PASV_MAX_PORT=21110 \
    -e LOCAL_UMASK=022 \
    -p 21:21 -p 21100-21110:21100-21110 \
    -d \
    --name vsftpdconcifrado \
    -v /home/admin/datosftp:/home/vsftpd \
    mivsftpd
```

Comprueba ahora el acceso con Filezilla. Recuerda que tendrás que usar "Cifrado: Requiere FTP explícito sobre TLS" y el "Modo de transferencia: Pasivo"