Montar Servidor FTP con volumen(backup) en Contenedor Docker

Acosta Rosales Jair Sebastián

26 de julio de 2020

1. Creando dockerfile de proftpd

El servidor ftp elegido para el despliegue en docker ha sido proftpd, el cual contiene una basta documentación de su implementación.

Los elementos necesarios para la implementación del dockerfile son los siguientes:

- Archivo .deb de proftp, el cual se puede conseguir en: `https://debian.pkgs.org/10/debian-main-amd64/proftpd-basic_1.3.6-4+deb10u5_amd64.deb.html`
- **proftpd.conf**: Archivo de configuración de proftpd el cuál contiene las configuraciones deseadas de nuestro servidor ftp.
- **tls.conf**: Archivo de configuración para conexiones con FTPS
- **proftpd.key.pem**: Llave de certificación de seguridad tls
- **proftpd.cert.pem**: Certificado de seguridad tls

  Una vez que tenemos estos elementos procedemos a construir el dockerfile de la siguiente forma

```
FROM debian:10

#Instala todas las dependencias del servidor proftpd en modo no interactivo(automatico)

RUN apt-get update && \
    DEBIAN_FRONTEND=noninteractive apt-get install -q -y --no-install-recommends \
    adduser \
    debianutils \
    libacl1 \
    libattr1 \
    libc6 \
    libcap2 \
    libhiredis0.14 \
    libmemcached11 \
    libmemcachedutil2 \
    libncursesw6 \
    libpam-runtime \
    libpam0g \
    libpcre3 \
    libssl1.1 \
    libtinfo6 \
    libwrap0 \
    lsb-base \
    netbase \
    sed \
    ucf \
    zlib1g \
    openssl \
    ca-certificates \
    ssl-cert \
    nano \
    net-tools

#Copia el .deb para instalar el servidor
COPY ./proftpd.deb /tmp/proftpd.deb
RUN dpkg -i /tmp/proftpd.deb

#Creando carpetas de subida y descarga de archivos, además agregando una shell falsa que

RUN mkdir /home/ftp/

#Copia
COPY ftpdirectory /home/ftp/
COPY proftpd.cert.pem /etc/ssl/certs/
COPY proftpd.key.pem /etc/ssl/private/
COPY proftpd.conf /etc/proftpd/
COPY tls.conf /etc/proftpd/
```

En esta imagen se puede apreciar como se divide la creación de todo el ambiente para el servidor en distintos puntos:

1. El SO usado como base para montar el servidor es Debian 10.
2. Se instalan todas las dependencias para este servidor, las cuales se pueden encontrar en: `https://debian.pkgs.org/10/debian-main-amd64/proftpd-basic_1.3.6-4+deb10u5_amd64.deb.html` en el apartado **Requires**.
3. Copia el archivo .deb de proftpd y procede a instalar el paquete.
4. Crea el directorio ftp donde los usuarios se conectarán.
5. Copia todos los archivos de configuración para proftpd que contienen nuestra configuración deseada los cuales sustituyen a los archivos por default que se crearon al instalar proftpd y también copia las carpetas **download** y **upload** que tendrán archivos para descarga y subida respectivamente.

```
RUN chmod 777 /home/ftp/upload && chmod 755 /home/ftp/download && \
 echo "/bin/false" >> /etc/shells && \
 addgroup ftp && \
 useradd --group ftp --home /home/ftp/ -s /bin/false laptop && \
 useradd --group ftp --home /home/ftp/ -s /bin/false motorola && \
 useradd --home /home/ftp/ -s /bin/false xiaomi


# echo "AllowForeignAddress             on" >> /etc/proftpd/proftpd.conf    #Permite las conexiones de ip's foraneas

EXPOSE 21

#Inicia el servidor en modo foreground
CMD ["proftpd","-n"]
```

En esta imagen se muestran las últimas instrucciones para la creación del dockerfile

1. Se cambian los permisos de las carpetas download y upload que vayan acorde a sus respectivas funciones.
2. Se crea un shell falso para deshabilitar la línea de comandos en los usuarios.
3. Se crea el grupo ftp.
4. Se crean tres usuarios que serán miembros del grupo creado y además se les especifica el directorio home que tendrán por default el cual es el directorio que contiene a ambas carpetas de descarga y subida de archivos, además se indica que tendrán al shell falso.
5. Se expone el puerto 21 que es el que maneja FTP por default para recibir peticiones.
6. Se establece el comando CMD que ejecutara el contenedor apenas iniciarse, este comando es importante dado que ayuda a permitir la vida del contenedor, pues permite que el servidor ftp se ejecute en primer plano y de esta forma el contenedor viva el tiempo que el servicio vive.

Nota: Existe una instrucción en nuestro Dockerfile que modifica la configuración de proftpd la cual tiene escrito "AllowForeignAddres", esta configuración permite al servidor ftp aceptar conexiones de un dispositivo que no pertenezca a la misma red, algo que puede resultar peligroso y por ende se comentó.

2. Creando la imagen del servidor

En este apartado se procederá a crear la imagen correspondiente que alojará el servidor ftp con las configuraciones correspondientes deseadas. La siguiente imagen muestra el comando ejecutado para crear la imagen.

```
sudo docker build -t proftpd:v5 proftp
```

Aquí podemos observar las partes de las que está conformado el comando:

1. **build:** Indica que el comando es para crear una imagen.
2. **-t:** Bandera que indica que se agregara un valor de "tag" para la imagen.
3. **Nombre:tag** Indica el nombre de la imagen y el nombre del tag, de tal forma que podemos tener una imagen con el mismo nombre pero diferente tag, indicando así varias versiones de un mismo servidor.
4. **Carpeta:** Indica el nombre de la carpeta donde está ubicado el archivo Dockerfile previamente creado.

Finalmente obtendremos como resultado del proceso de cración de la imagen será este:

```
Successfully built 4c06665813b7
Successfully tagged proftpd:v5
```

3. Creación de contenedor

Una vez creada la imagen procederemos a usarla para crear el contenedor de docker que tendrá el servidor ftp que recibirá las peticiones.

3.1. Creación de volumen para contenedor

La creación de un volumen permite tener archivos un backup de archivos del contenedor, los cuales persisten a pesar de que el contenedor se eliminen, de este modo podemos mantener archivos importantes sin preocuparnos de perderlos en el contenedor.

La siguiente imagen muestra la creación del volumen:

```
jsebastian-ar@jSebastian-AR:~$ docker volume create ftp_users
ftp_users
```

Donde:

1. **volume:** Indica que será una instrucción para volumenes
2. **create:** Indica que se creará un volumen
3. **nombre_vol:** Es el nombre que le asignaremos al volumen

La siguiente imagen muestra el comando ejecutado para crear el contenedor:

```
jsebastian-ar@jSebastian-AR:~/Documentos/git-repos/ASR/Practica_3/docker$ docker run -d --name ftp_users --network host -v ftp_users:/home/ftp/ proftpd:v5
7182c7c7d55b17b00bc33aece62c2160ae979402d4313bea4cf2da011055134b
jsebastian-ar@jSebastian-AR:~/Documentos/git-repos/ASR/Practica_3/docker$
```

Donde:

1. **run:** Indica la creación y ejecución de un nuevo contenedor.

2. **-d:** Ejecuta el contenedor en modo detach, es decir como un daemon.

3. –**name:** Bandera que permite asignar el nombre al contenedor, seguido del mismo nombre.

4. –**network:** Permite asignar la red a la que estará conectado el contenedor, en este caso será la red "host" pues está permitira que el contenedor forme parte de la misma red de dispositivos que se conectarán al servidor y de esta forma evitaremos la línea comentada "AllowForeignAddresses" en el Dockerfile.

5. **-v:** Indica el volumen usado como backup para el contenedor, seguido de la full_path dentro del contenedor a la que estará ligado el backup del host, gracias a esta full_path podremos ser capaces de acceder al backup desde adentro del contenedor y modificar su contenido.

   Para este caso la carpeta ligada al volumen es la misma carpeta donde se almacenan la carpeta "download" y "upload" para el server ftp.

6. **nombre:tag** Indicamos el nombre y el tag de la imagen usada para crear el contenedor