Servidores FTP y Samba

Para empezar hemos añadido al docker-compose un nuevo contenedor de debian para alojar el servidor de Samba y de FTP:

```
# Servidor Samba y FTP para documentación
doc_server:
  build: ./doc_server
  container_name: dev-doc-server
  restart: always
  networks:
    development:
      ipv4_address: 172.40.0.10
  volumes:
    - doc_data:/var/doc_server
    - ./doc_server/smb.conf:/etc/samba/smb.conf
    - ./doc_server/vsftpd.conf:/etc/vsftpd.conf
    - ./doc_server/entrypoint.sh:/entrypoint.sh
  cap_add:
    - NET_ADMIN
  dns:
    - 172.20.0.100
  sysctls:
    - net.ipv4.ip_forward=1
  extra_hosts:
    - "dns_server:172.20.0.100"
  entrypoint: ["/bin/sh", "/entrypoint.sh"]
  depends_on:
    - dns_server
```

Para su correcto funcionamiento hemos creado varios archivos para completar su configuración, todos en la carpeta “doc_server” del proyecto. En el Dockerfile instalamos todos los paquetes necesarios, así como los usuarios que serán necesarios para Samba. Luego en el entrypoint creamos algunos archivos de ejemplo y nos aseguramos de que los permisos sean correctos. Aparte hay tres archivos más para configuración del funcionamiento de los servicios, smb.conf, supervisord.conf y vsftpd.conf.

La estructura de archivos es:

```
/var/doc_server/
├── desarrollo/
│   ├── SW1/ # Exclusivo para empleado1
│   ├── SW2/ # Exclusivo para empleado2
│   ├── SW3/ # Exclusivo para empleado3
│   ├── SW4/ # Exclusivo para empleado4
│   └── SW5/ # Exclusivo para empleado5
├── revision/
│   ├── SW1/ # Empleado1 y revisor tienen acceso
│   ├── SW2/ # Empleado2 y revisor tienen acceso
│   ├── SW3/ # Empleado3 y revisor tienen acceso
│   ├── SW4/ # Empleado4 y revisor tienen acceso
│   └── SW5/ # Empleado5 y revisor tienen acceso
└── publico/
    ├── SW1/ # Solo revisor puede modificar, todos pueden leer
    ├── SW2/ # Solo revisor puede modificar, todos pueden leer
    ├── SW3/ # Solo revisor puede modificar, todos pueden leer
    ├── SW4/ # Solo revisor puede modificar, todos pueden leer
    └── SW5/ # Solo revisor puede modificar, todos pueden leer
```

Podremos acceder a Samba y a ftp desde la consola del host, ya que al ser una red docker el host tiene acceso a los contenedores.

Para Samba la conexión se hará de la siguiente forma: “smbclient //172.40.0.10/carpeta -U usuario%contraseña”

Por ejemplo, probaremos a simular el flujo de trabajo, primero un usuario subirá un archivo, un revisor lo verificará y lo subirá a la carpeta “público”, donde se podrá ver por cualquier usuario.

```
javi@javi-Aspire-A315-41:/tmp$ echo "Documento listo para revisión" > /tmp/revision.txt
javi@javi-Aspire-A315-41:/tmp$ smbclient //172.40.0.10/revision_SW1 -U empleado1%password1
Try "help" to get a list of possible commands.
smb: \> lcd /tmp
smb: \> put revision.txt
putting file revision.txt as \revision.txt (10.1 kb/s) (average 10.1 kb/s)
smb: \> ls
  .                                   D        0  Fri May  2 00:06:23 2025
  ..                                  D        0  Thu May  1 23:09:28 2025
  revision.txt                        A       31  Fri May  2 00:06:23 2025

                130331724 blocks of size 1024. 83974860 blocks available
```

```
javi@javi-Aspire-A315-41:/tmp$ smbclient //172.40.0.10/revision_SW1 -U revisor%revisorpass
Try "help" to get a list of possible commands.
smb: \> get revision.txt /tmp/revision.txt
getting file \revision.txt of size 31 as /tmp/revision.txt (15.1 KiloBytes/sec) (average 15.1 KiloBytes/sec)
smb: \> exit
javi@javi-Aspire-A315-41:/tmp$ smbclient //172.40.0.10/publico -U revisor%revisorpass
Try "help" to get a list of possible commands.
smb: \> cd SW1
smb: \SW1\> put revision.txt
putting file revision.txt as \SW1\revision.txt (15.1 kb/s) (average 15.1 kb/s)
smb: \SW1\> ls
  .                                   D        0  Fri May  2 00:36:08 2025
  ..                                  D        0  Fri May  2 00:32:55 2025
  leeme.txt                           N       47  Fri May  2 00:33:05 2025
  documentacion.txt                   N       26  Fri May  2 00:33:05 2025
  manual.txt                          N       25  Fri May  2 00:33:05 2025
  revision.txt                        A       31  Fri May  2 00:36:08 2025

                130331724 blocks of size 1024. 83972096 blocks available
smb: \SW1\> exit
```

```
javi@javi-Aspire-A315-41:/tmp$ lftp -u anonymous, 172.40.0.10
lftp anonymous@172.40.0.10:~> ls
drwxr-xr-x    2 ftp      ftp          4096 May 01 22:36 SW1
drwxr-xr-x    2 ftp      ftp          4096 May 01 22:33 SW2
drwxr-xr-x    2 ftp      ftp          4096 May 01 22:33 SW3
drwxr-xr-x    2 ftp      ftp          4096 May 01 22:33 SW4
drwxr-xr-x    2 ftp      ftp          4096 May 01 22:33 SW5
lftp anonymous@172.40.0.10:/> cd SW1
lftp anonymous@172.40.0.10:/SW1> ls
-rw-r--r--    1 ftp      ftp            26 May 01 22:33 documentacion.txt
-rw-r--r--    1 ftp      ftp            47 May 01 22:33 leeme.txt
-rw-r--r--    1 ftp      ftp            25 May 01 22:33 manual.txt
-rwxr--r--    1 ftp      ftp            31 May 01 22:36 revision.txt
lftp anonymous@172.40.0.10:/SW1>
```

Como podemos ver, el usuario anónimo puede acceder a través de FTP al contenido de las carpetas públicas.
Por otro lado, también demostramos por ejemplo que Empleado2 no puede acceder al directorio privado de desarrollo de Empleado1:

```
javi@javi-Aspire-A315-41:/tmp$ smbclient //172.40.0.10/desarrollo_SW1 -U empleado2%password2
tree connect failed: NT_STATUS_ACCESS_DENIED
javi@javi-Aspire-A315-41:/tmp$
```

O que Empleado1 no puede subir archivos a la carpeta pública.
Para consultar los permisos en mayor profundidad, acceder a Dockerfile y a entrypoint.sh de la carpeta /doc_server. (Algunos permisos están repetidos entre los dos porque no siempre se veían reflejados al levantar el contenedor, así que se repiten para asegurar el correcto funcionamiento).

Para el servidor sftp proxy lo primero que hemos hecho ha sido crear un par de claves sin passphrase para una conexión segura con “ ssh-keygen -t rsa -b 2048 -f /root/.ssh/test_key -N "" ”. De este par hemos copiado la clave pública en el servidor para que nos permita conectarnos por ssh. Este servidor utiliza un script de python para sincronizar periódicamente el contenido de la carpeta pública del contenedor doc_server, así el usuario nunca accede directamente al servidor FTP.
Para probar su funcionamiento hemos creado un contenedor adicional fuera de la red que hemos creado para garantizar que los contenedores no están conectados. Para demostrarlo, desde el contenedor externo intentamos conectarnos al servidor ftp. Para ello lo hacemos usando la ip del host de los contenedores, la conexión será rechazada porque el contenedor router no escucha en el puerto 22:

```
lftp anonymous@172.40.0.10:~> lftp -u anonymous, 192.168.68.102
lftp anonymous@192.168.68.102:~> ls
`ls' at 0 [Connection refused]
```

De igual manera, si intentamos conectar con el mismo servidor ni siquiera lo encontrará:

```
lftp anonymous@192.168.68.102:~> lftp -u anonymous, 172.40.0.10
lftp anonymous@172.40.0.10:~> ls
`ls' at 0 [Connecting...]
```

Para conectarnos al servidor ftp desde fuera de red lo haremos de la siguiente manera:

Como se mencionó previamente, aquí estamos conectándonos a sftp usando la ip del host, donde el contenedor router tiene el puerto 2222 expuesto. La petición de ssh se redirige al servidor sftp proxy que se comunica con el servidor ftp.
De los contenedores que estamos usando, ninguno tiene los puertos expuestos, es el router el que maneja todo el intercambio de información.
Igual que en el caso anterior, para consultar más en detalle la configuración interna acceder a los archivos de la carpeta sftp_proxy.

Javier Rábago Montero
Miguel Álvarez Garcés
Antonio González de los Santos