



# UNIVERSIDAD DE GRANADA

## **Centro de Procesamiento de Datos**

### **Práctica 1 – Contenedores Docker**

---

Arturo Alonso Carbonero

## Apartado VI

Para poder compartir un directorio del host anfitrión con un contenedor, es necesario ejecutar la orden **docker run** con la opción **--volume** o **-v**. Esta opción permite montar un directorio local en un directorio propio del contenedor. En la siguiente imagen se muestra la ejecución de dicho comando para proporcionar el directorio **/var/www/html** local al contenedor y poder mostrar así una página personalizada, añadiendo las opciones a continuación listadas:

- **--name**: Nombre del contenedor.
- **-p**: Permite exponer un puerto del contenedor a un puerto local.
- **-d**: Imagen a utilizar, en este caso *nginx*.

```
alonsoarturo@alonsoarturo-VirtualBox:~$ sudo docker run --name nginx2 -v ~/var/www/html:/usr/share/nginx/html:ro -v ~/var/nginx/conf:/etc/nginx:ro -p8081:80 -p8444:443 -d nginx
a0b9b75d1f15b23c9e87150aaa4752a214c5f228a0586ef5f5d5dd911e3da90b
alonsoarturo@alonsoarturo-VirtualBox:~$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS          NAMES
alonsoarturo@alonsoarturo-VirtualBox:~$ sudo docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS          NAMES
a0b9b75d1f15   nginx    "/docker-entrypoint.    6 seconds ago  Exited (1) 4 s
seconds ago
nginx2
f6adac9158e9   nginx    "/docker-entrypoint.    13 hours ago   Exited (255) 5
minutes ago
0.0.0.0:8080->80/tcp, :::8080->80/tcp, 0.0.0.0:8443->443/tcp, :::8443->443/tcp
nginx1
```

*Resultado inicial*

Como se puede observar, al ejecutar la orden **docker ps** y **docker ps -a**, vemos que el contenedor se ha creado pero se cierra de forma automática tras iniciar. Haciendo uso de la orden **docker logs** y **docker inspect** sobre el contenedor como se muestra en la siguiente imagen, se puede observar que lo que sucede es que el contenedor no es capaz de detectar el fichero de configuración de *nginx*.

```
alonsoarturo@alonsoarturo-VirtualBox:~$ sudo docker logs nginx2
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: /etc/nginx/conf.d/default.conf is not a file or does not exist
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2023/09/26 14:14:05 [emerg] 1#1: open() "/etc/nginx/nginx.conf" failed (2: No such file or directory)
nginx: [emerg] open() "/etc/nginx/nginx.conf" failed (2: No such file or directory)
alonsoarturo@alonsoarturo-VirtualBox:~$ sudo docker inspect nginx2 | grep nginx.conf
"/home/alonsoarturo/var/nginx/conf:/etc/nginx:ro"
"Source": "/home/alonsoarturo/var/nginx/conf",
```

*Detección de error*

Para solucionarlo, he optado por proporcionar de forma directa dicho fichero al contenedor en el momento de su creación. En la imagen siguiente se puede observar que el resultado es el adecuado.

```
alonsoarturo@alonsoarturo-VirtualBox:~$ sudo docker run --name nginx2 -v ~/var/www/html:/usr/share/nginx/html:ro -v ~/etc/nginx/nginx.conf:/etc/nginx.conf:ro -p 8081:80 -p8444:443 -d nginx
d8511efc86fc8a484d9e8a177d74b5f1f01de197d8e09371940926c1d3826811
alonsoarturo@alonsoarturo-VirtualBox:~$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS
PORTS
NAMES
d8511efc86fc   nginx    "/docker-entrypoint...." 12 seconds ago Up 3 seconds
0.0.0.0:8081->80/tcp, :::8081->80/tcp, 0.0.0.0:8444->443/tcp, :::8444->443/tcp
nginx2
```

*Contenedor creado y en ejecución*

```
alonsoarturo@alonsoarturo-VirtualBox:~$ ls /etc/nginx
conf.d      koi-win      nginx.conf   sites-enabled
fastcgi.conf  mime.types  proxy_params snippets
fastcgi_params modules-available scgi_params  uwsgi_params
koi-utf      modules-enabled sites-available win-utf
```

*Fichero de configuración de nginx*

Para demostrar que el acceso personalizado es correcto, he creado un fichero **index.html** en el directorio local **/var/www/html/** cuyo contenido es el siguiente:

```
alonsoarturo@alonsoarturo-VirtualBox:~$ cat /var/www/html/index.html
<html>
  <body>
    <h1>Practica 1 - Nginx y Docker</h1>
    <a>Realido por: Arturo Alonso Carbonero</a>
  </body>
</html>
```

*index.html personalizado*

Para comprobar que el resultado es correcto, basta con acceder desde un navegador a la dirección <http://localhost:8081> o realizar una conexión con **curl** al mismo dominio.



*Resultado en navegador*

```
alonsoarturo@alonsoarturo-VirtualBox:~$ curl -I localhost:8081
HTTP/1.1 200 OK
Server: nginx/1.25.2
Date: Tue, 26 Sep 2023 14:18:43 GMT
Content-Type: text/html
Content-Length: 123
Last-Modified: Tue, 26 Sep 2023 01:25:11 GMT
Connection: keep-alive
ETag: "651232f7-7b"
Accept-Ranges: bytes
```

*Resultado con curl*

## Apartado IX

Si deseamos crear un contenedor de forma interactiva, es necesario ejecutar el comando **docker run** con el parámetro **-it (Foreground/Interactive)**. Esto permite mantener el contenedor en ejecución e interactuar con él. En la siguiente imagen se muestra un ejemplo de ejecución de un contenedor para emplear un *bash* de Ubuntu.

```
alonsoarturo@alonsoarturo-VirtualBox:~$ sudo docker run -it ubuntu bash
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
445a6a12be2b: Pull complete
Digest: sha256:aabed3296a3d45ceded1dc866a24476c4d7e093aa806263c27ddaadbdc3c1054
Status: Downloaded newer image for ubuntu:latest
root@dbb41ec7536a:/#
```

### Ejecución interactiva

Una vez estamos dentro del contenedor, ejecutamos las órdenes **atp update** y **apt upgrade** para actualizar los paquetes e instalamos *nginx* con **apt install nginx**. Salimos del contenedor mediante la orden **exit** y comprobamos con **docker ps -a** si el contenedor se ha creado con éxito.

```
alonsoarturo@alonsoarturo-VirtualBox:~$ sudo docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
dbb41ec7536a	ubuntu	"bash"	2 minutes ago	Exited (0) 7 seconds ago	
d8511efc86fc	nginx	"/docker-entrypoint..."	10 minutes ago	Up 10 minutes	0.0.0.0:8081->80
f6adac9158e9	nginx	"/docker-entrypoint..."	13 hours ago	Exited (255) 18 minutes ago	0.0.0.0:8080->80

### Contenedor creado con éxito

Guardamos la imagen haciendo uso del comando **docker commit** y la ID del contenedor y comprobamos con **docker images** que se ha guardado correctamente.

```
alonsoarturo@alonsoarturo-VirtualBox:~$ sudo docker commit dbb41ec7536a imagenpractica1
sha256:0298768d05daf4a10e06df87d7eb6b0ccf74604c9a2433b29d25ad31d08c00c9
alonsoarturo@alonsoarturo-VirtualBox:~$ sudo docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
imagenpractica1	latest	0298768d05da	5 seconds ago	194MB
nginx	latest	61395b4c586d	5 days ago	187MB
ubuntu	latest	c6b84b685f35	5 weeks ago	77.8MB

### Imagen creada correctamente

Ejecutamos el comando **docker save** con la opción **-o**, la cual permite guardar la salida en un fichero en lugar de la salida estándar. Así, es posible guardar la imagen para compartirla con un tercero. Para ello, se ejecuta dicho comando con el nombre de la imagen y el fichero *.tar* correspondiente. Del otro lado, para obtener la imagen, se ejecuta el comando **docker load** con la opción **-i**, que permite leer un fichero *.tar* como entrada.

```
alonsoarturo@alonsoarturo-VirtualBox:~$ sudo docker save -o ~/Escritorio/CPD/imagenpractica1.tar imagenpractica1
alonsoarturo@alonsoarturo-VirtualBox:~$ ls ~/Escritorio/CPD/
html  imagenpractica1.tar
```

### Imagen guardada

## Anexo I

Para hacer uso de SSHFS con *Alpine*, en primer lugar, es necesario obtener la imagen pertinente.

```
alonsoarturo@alonsoarturo-VirtualBox:~$ sudo docker pull alpine
Using default tag: latest
latest: Pulling from library/alpine
7264a8db6415: Pull complete
Digest: sha256:7144f7bab3d4c2648d7e59409f15ec52a18006a128c733fcff20d3a4a54ba44a
Status: Downloaded newer image for alpine:latest
docker.io/library/alpine:latest
alonsoarturo@alonsoarturo-VirtualBox:~$ sudo docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
imagenpractica1	latest	0298768d05da	10 minutes ago	194MB
nginx	latest	61395b4c586d	5 days ago	187MB
ubuntu	latest	c6b84b685f35	5 weeks ago	77.8MB
alpine	latest	7e01a0d0a1dc	7 weeks ago	7.34MB

*Imagen de Alpine*

Una vez obtenida la imagen, ejecutamos un contenedor de forma interactiva con *Alpine* y ejecutamos el comando **apk add sshfs** para obtener SSHFS dentro del mismo.

```
alonsoarturo@alonsoarturo-VirtualBox:~$ sudo docker run -it alpine /bin/sh
/ # mkdir alonsoarturo_p1
/ # apk add sshfs
fetch https://dl-cdn.alpinelinux.org/alpine/v3.18/main/x86_64/APKINDEX.tar.gz
fetch https://dl-cdn.alpinelinux.org/alpine/v3.18/community/x86_64/APKINDEX.tar.gz
(1/16) Installing openssh-keygen (9.3_p2-r0)
(2/16) Installing ncurses-terminfo-base (6.4_p20230506-r0)
(3/16) Installing libncursesw (6.4_p20230506-r0)
(4/16) Installing libedit (20221030.3.1-r1)
(5/16) Installing openssh-client-common (9.3_p2-r0)
(6/16) Installing openssh-client-default (9.3_p2-r0)
(7/16) Installing fuse-common (3.14.1-r2)
(8/16) Installing fuse3-libs (3.14.1-r2)
(9/16) Installing fuse3 (3.14.1-r2)
(10/16) Installing libffi (3.4.4-r2)
(11/16) Installing libintl (0.21.1-r7)
(12/16) Installing libblkid (2.38.1-r8)
(13/16) Installing libmount (2.38.1-r8)
(14/16) Installing pcre2 (10.42-r1)
(15/16) Installing glib (2.76.4-r0)
(16/16) Installing sshfs (3.7.3-r1)
Executing busybox-1.36.1-r2.trigger
Executing glib-2.76.4-r0.trigger
OK: 18 MiB in 31 packages
```

*Obtención de SSHFS*

A continuación, basta con crear el directorio a montar, mediante **mkdir**, y realizar dicha acción usando **sshfs**. Sin embargo, ocurre un error, el cual se muestra en la imagen siguiente.

```
/ # sshfs alonsoarturo@turing.ugr.es:~ alonsoarturo_p1
fuse: device not found, try 'modprobe fuse' first
/ # modprobe fuse
modprobe: can't change directory to '/lib/modules': No such file or directory
```

*Error encontrado*

El problema es que existe un conflicto con los privilegios. Para solucionar el error y poder continuar, es necesario crear el contenedor con la opción **--privileged=true**.

```
alonsoarturo@alonsoarturo-VirtualBox:~$ sudo docker run --privileged=true -it alpine /bin/sh
```

*Nueva creación*

Tras esto, es posible realizar la acción deseada con el nuevo contenedor.

```
/ # mkdir turing_p1
/ # sshfs alonsoarturo@turing.ugr.es:. turing_p1
The authenticity of host 'turing.ugr.es (150.214.191.90)' can't be established.
ED25519 key fingerprint is SHA256:R6U1/Dp5bVJH1+X94CVnT6AjGU5NoaXxcCEz6XWiUCA.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
alonsoarturo@turing.ugr.es's password:
alonsoarturo@turing.ugr.es's password:
/ # ls
bin          etc          lib          mnt          proc         run          srv          tmp          usr
dev          home         media        opt          root         sbin        sys          turing_p1   var
/ # umount turing_p1
/ #
```

*Resultado final*

## Referencias

### Apartado VI

- <https://soka.gitlab.io/blog/post/2019-07-08-docker-imagenes-y-contenedores/>
- <https://www.nginx.com/blog/deploying-nginx-nginx-plus-docker/>
- <https://forums.docker.com/t/nginx-in-docker-emerg-1-1-open-etc-nginx-nginx-conf-failed-2-no-such-file-or-directory/103325>
- <https://docs.docker.com/engine/reference/commandline/run/>

### Apartado IX

- <https://pandorafms.com/blog/es/docker-run/>

### Anexo I

- <https://stackoverflow.com/questions/55696482/how-to-install-sshfs-inside-alpine-container>
- <https://docs.docker.com/engine/reference/commandline/save/>
- <https://docs.docker.com/engine/reference/commandline/load/>