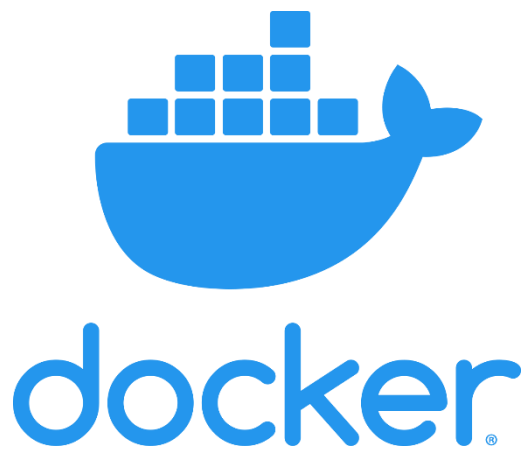


PROXY REVERSO



Pablo Hernández Ruiz

Índice:

Requisitos necesarios:.....	2
Configuración contenedores:.....	2
Comprobación:.....	3
Configuración páginas web:.....	3
Configuración servidor proxy:.....	4
IP:	4
Configuración:.....	4
Comprobar funcionamiento del contenedor:.....	5
Comprobación final:.....	5
Importante:	5

Requisitos necesarios:

Para la realización de esta práctica necesitaremos tener previamente instalado las siguientes herramientas:

-**Docker**: es un conjunto de productos de **plataforma como servicio (PaaS)** que utilizan la virtualización a nivel de sistema operativo para entregar software en paquetes llamados **contenedores**.

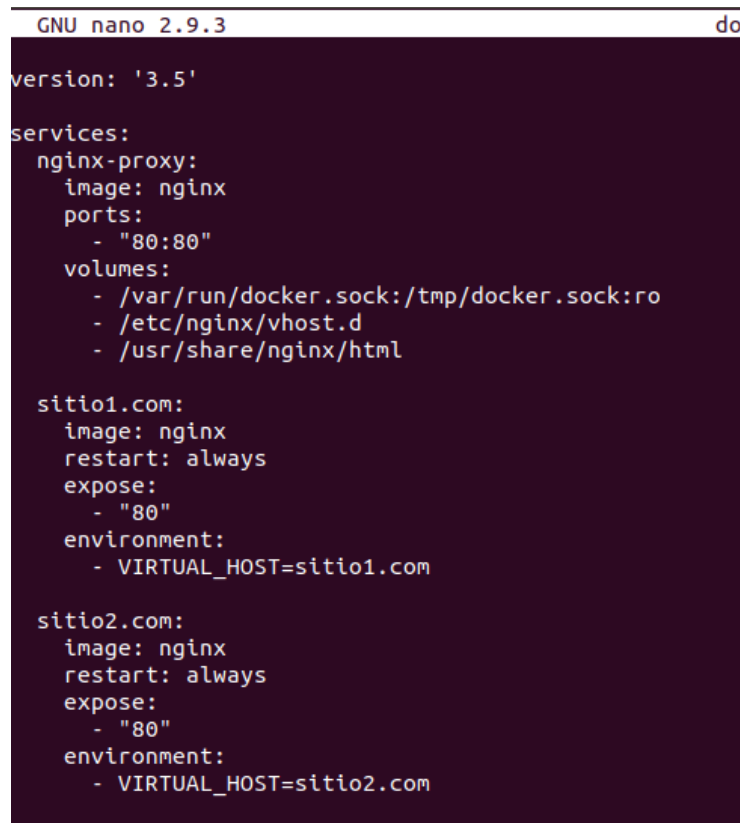
-**Docker-Compose**: es una herramienta para definir y ejecutar aplicaciones **Docker** de varios contenedores. Con **Docker-Compose**, usa un archivo **YAML** para configurar los servicios de su aplicación. Luego, con un solo comando, crea e inicia todos los servicios desde su configuración.

Para instalar las herramientas solo tendremos que utilizar la siguiente sentencia:

"sudo apt install *Nombre de la herramienta*"

Configuración contenedores:

Lo primero que haremos será crear un archivo **.yml** donde le daremos forma a lo que será el **servidor** y las **dos páginas web**:



```
GNU nano 2.9.3 do
version: '3.5'
services:
  nginx-proxy:
    image: nginx
    ports:
      - "80:80"
    volumes:
      - /var/run/docker.sock:/tmp/docker.sock:ro
      - /etc/nginx/vhost.d
      - /usr/share/nginx/html

  sitio1.com:
    image: nginx
    restart: always
    expose:
      - "80"
    environment:
      - VIRTUAL_HOST=sitio1.com

  sitio2.com:
    image: nginx
    restart: always
    expose:
      - "80"
    environment:
      - VIRTUAL_HOST=sitio2.com
```

Ahora ejecutaremos el comando **"sudo docker-compose up -d"** para levantar los contenedores.

Comprobación:

Ahora utilizaremos el comando **"sudo docker ps -a"** para ver que se ha levantado los contenedores:

```
pablohr@pablohr-Virtual-Machine:~/Escritorio/docker4$ sudo docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
96afee9d0bdf	nginx	"/docker-entrypoint..."	About an hour ago	Up 59 minutes	0.0.0.0:80->80/tcp	docker4_n
glnx-proxy_1						
9671666d8366	nginx	"/docker-entrypoint..."	About an hour ago	Up About an hour	80/tcp	docker4_s
itio2.com_1						
133b1c0aed3e	nginx	"/docker-entrypoint..."	About an hour ago	Up About an hour	80/tcp	docker4_s
itio1.com_1						

Configuración páginas web:

Nos dirigiremos dentro del contenedor de cada una de las páginas web mediante el siguiente comando:

"sudo docker exec -it *ID contenedor* bin/bash"

Una vez dentro tendremos que actualizar los paquetes de herramientas (en cada contenedor en el que nos metemos), para ello utilizaremos los siguientes comandos:

"apt update"

"apt upgrade"

También instalaremos un **editor de texto** para poder editar los archivos, en mí caso utilizaré nano por lo que ejecutaré el siguiente comando:

"apt install nano"

Una vez actualizados los paquetes lo que haremos será **borrar** el archivo **index.html** mediante el comando **"rm -rf *directorio donde se ubica el archivo*"**:

```
pablohr@pablohr-Virtual-Machine:~$ sudo docker exec -it 133 bin/bash
root@133b1c0aed3e:/# rm -rf /usr/share/nginx/html/index.html
root@133b1c0aed3e:/# nano /usr/share/nginx/html/index.html
```

Y ahora crearemos uno nuevo en el que agregaremos unas pocas líneas para hacer una **página web personalizada**:

```
GNU nano 3.2 /usr/share/nginx/html/index.html

<html>
<title>sitio 1</title>
<body>
sitio1
</body>
</html>
```

Esto lo haremos también con el contenedor de la otra página web

Configuración servidor proxy:

IP:

Lo primero que haremos será saber **las IP** de los contenedores que contienen nuestras **páginas web**, para ello lo que haremos será utilizar el siguiente comando:

“sudo docker inspect *ID contenedor* | grep “IPAddress””

```
pablohr@pablohr-Virtual-Machine:~/Escritorio/docker4$ sudo docker inspect 967 |
grep "IPAddress"
    "SecondaryIPAddresses": null,
    "IPAddress": "",
    "IPAddress": "172.26.0.3",
pablohr@pablohr-Virtual-Machine:~/Escritorio/docker4$ sudo docker inspect 133 |
grep "IPAddress"
    "SecondaryIPAddresses": null,
    "IPAddress": "",
    "IPAddress": "172.26.0.2",
pablohr@pablohr-Virtual-Machine:~/Escritorio/docker4$ sudo docker inspect 96a |
grep "IPAddress"
[sudo] contraseña para pablohr:
    "SecondaryIPAddresses": null,
    "IPAddress": "",
    "IPAddress": "172.26.0.4",
```

Configuración:

Una vez conozcamos **las IP**, nos adentraremos en el contenedor del servidor y crearemos un nuevo archivo el cual llamaremos **“load-balancing.conf”**, en el añadiremos las siguientes líneas:

```
GNU nano 3.2 /etc/nginx/conf.d/load-balancing.conf

upstream backend {
    server 172.26.0.2;
    server 172.26.0.3;
}

server {
    listen 80;
    server_name loadbalancing.example.com;

    location / {
        proxy_redirect off;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header Host $http_host;
        proxy_pass http://backend;
    }
}
```

En la parte de **“server *IP*”** cambiaremos la **IP** que tenga nuestro **contenedor** de la página web.

Lo siguiente que haremos será borrar el archivo “**default.conf**” para que utilice el archivo que hemos creado antes, y por último reiniciaremos el servidor, para ello utilizaremos el siguiente comando:

“**/etc/init.d/nginx restart**”

```
pablohr@pablohr-Virtual-Machine:~$ sudo docker exec -it 96a bin/bash
root@96afee9d0bdf:/# nano /etc/nginx/conf.d/load-balancing.conf
root@96afee9d0bdf:/# cd /etc/nginx/conf.d/
root@96afee9d0bdf:/etc/nginx/conf.d# ls
default.conf  load-balancing.conf
root@96afee9d0bdf:/etc/nginx/conf.d# rm -rf default.conf
root@96afee9d0bdf:/etc/nginx/conf.d# /etc/init.d/nginx restart
```

Comprobar funcionamiento del contenedor:

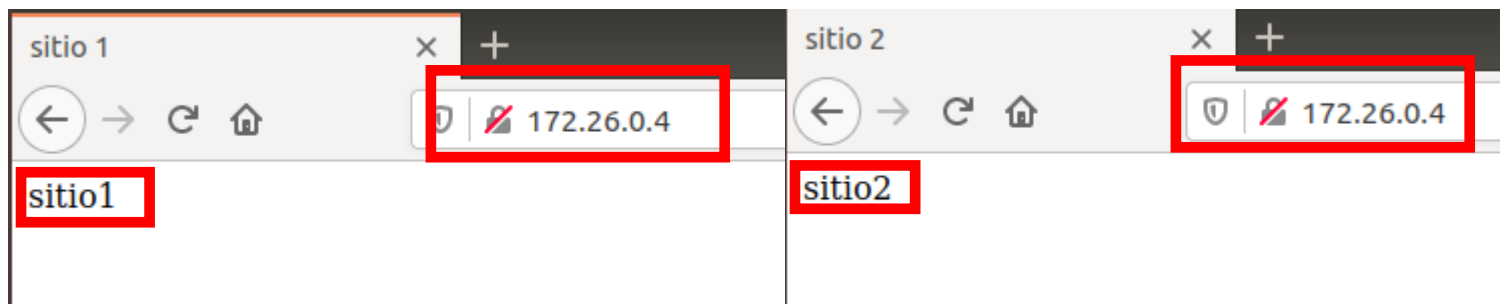
Para comprobar que **todas las modificaciones son correctas** lo que haremos será utilizar la siguiente sentencia:

“**nginx -t**”

```
root@96afee9d0bdf:/# nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

Comprobación final:

Ahora lo que haremos será dirigirnos al navegador y escribir la **IP** de nuestro servidor y **recargar** la página para comprobar que cambia a la otra página web:



Importante:

- A la hora de agregar los archivos **.yml** hay que tener especial cuidado a la hora de escribirlo ya que si escribimos un espacio de más o de menos pues hacer que el archivo no se ejecute correctamente, también cuidado a la hora de escribir una palabra mal.
- Cuidado con las rutas de los directorios, un fallo a la hora de escribir donde se sitúa nuestro archivo puede darnos un error el cual luego no pensemos que pueda ser ese el error que nos esté generando el fallo.