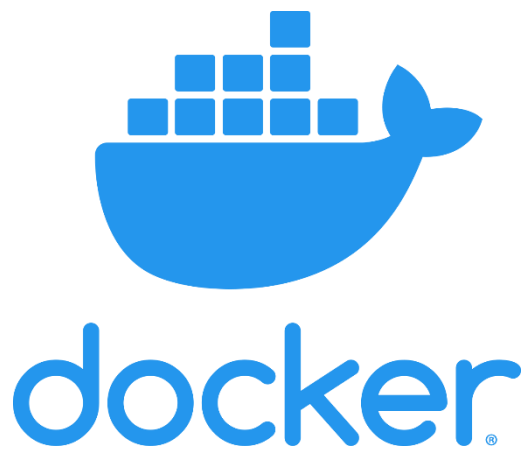


PROXY REVERSO



Pablo Hernández Ruiz

Índice:

Requisitos necesarios:	2
Configuración sitio 1 y sitio 2:	2
Index.html:	2
Docker-compose.yml:	3
Comprobación:	3
Configuración Proxy reverso:	4
Backend-not-found.html:	4
Docker-compose.yml:	4
Dockerfile:	4
Default.conf:	5
Includes:	5
Proxy.conf:	5
Ejecución:	6
Comprobación:	6
Configuración archivo hosts:	6
Comprobación final:	7
Importante:	7

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 sitio 1 y sitio 2:

Lo primero que haremos será crear nuestros directorios donde almacenaremos los archivos que nos ayudaran a la hora de crear nuestro **Proxy reverso**, para ello utilizaremos el comando **"mkdir *Nombre del directorio*"** para crear los directorios, una vez creados accederemos al directorio **"site1"** con el comando **"cd *Directorio*"**:

```
pablohr@pablohr-Virtual-Machine:~/Escritorio/docker3$ mkdir site1
pablohr@pablohr-Virtual-Machine:~/Escritorio/docker3$ mkdir site2
pablohr@pablohr-Virtual-Machine:~/Escritorio/docker3$ mkdir reverse-proxy
```

Ahora lo que haremos tanto en el **site1** como en el **site2** será crear un archivo **.html** donde crearemos una **pequeña** página web y un archivo **.yml** donde crearemos un **contenedor** para esa página web, para crear los archivos utilizaremos el editor de texto con el que estemos más cómodos, yo utilizaré **"nano"**, utilizamos el siguiente comando para crear y editar el archivo **.html**:

"sudo nano *Nombre del archivo*. *Extensión*"

Index.html:

En el archivo **.html** introduciremos las siguientes líneas y lo guardaremos:

```
GNU nano 2.9.3 index.html

<!DOCTYPE html>
<html>
  <head>
    <title>site1.example.com</title>
  </head>
  <body>
    <h1>site1.example.com</h1>
  </body>
</html>
```

Docker-compose.yml:

En el archivo **.yml** introduciremos las siguientes líneas y lo guardaremos:

```
GNU nano 2.9.3 docker-compose.yml
version: '3'
services:
  app:
    image: nginx:1.12
    volumes:
      - ./usr/share/nginx/html/
    expose:
      - "80"
```

Una vez tengamos los archivos listos utilizaremos los siguientes comandos:

-sudo docker-compose build: con este comando crearemos imágenes de los servicios indicados en el **Docker-Compose**.

-sudo docker-compose up -d: con este comando levantaremos el contenedor.

```
pablohr@pablohr-Virtual-Machine:~/Escritorio/docker3/site1$ sudo docker-compose build
app uses an image, skipping
pablohr@pablohr-Virtual-Machine:~/Escritorio/docker3/site1$ sudo docker-compose up -d
Creating network "site1_default" with the default driver
Pulling app (nginx:1.12)...
1.12: Pulling from library/nginx
f2aa67a397c4: Pull complete
e3eaf3d87fe0: Pull complete
38cb13c1e4c9: Pull complete
Digest: sha256:72daaf46f11cc753c4eab981cbf869919bd1fee3d2170a2adeac12400f494728
Status: Downloaded newer image for nginx:1.12
Creating site1_app_1 ...
Creating site1_app_1 ... done
```

Estos pasos los realizaremos también en site2

Comprobación:

Una vez levantados los dos contenedores podremos verlos mediante el siguiente comando:

"sudo docker ps -a"

```
pablohr@pablohr-Virtual-Machine:~/Escritorio/docker3/reverse-proxy$ sudo docker
ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS
PORTS         NAMES
f1a0b2826ef3   nginx:1.12    "nginx -g 'daemon of..." 4 minutes ago  Up 4 minute
s 80/tcp       site2_app_1
4d98a49aa25b   nginx:1.12    "nginx -g 'daemon of..." 9 minutes ago  Up 9 minute
s 80/tcp       site1_app_1
```

Configuración Proxy reverso:

Ahora accederemos al directorio “**reverse-proxy**” y crearemos los siguientes archivos con sus respectivas líneas de código:

Backend-not-found.html:

Aquí tendremos un **.html** el cual aparecerá solo si ocurre algún error en el proxy:

```
GNU nano 2.9.3 backend-not-found.html
<html>
  <head><title>Proxy Backend Not Found</title></head>
  <body >
    <h2>Proxy Backend Not Found</h2>
  </body>
</html>
```

Docker-compose.yml:

Aquí tenemos el archivo que levantará el contenedor del proxy y configure los dos sitios web:

```
GNU nano 2.9.3 docker-compose.yml
version: '3'
services:
  proxy:
    build: ./
    networks:
      - site1
      - site2
    ports:
      - 80:80

networks:
  site1:
    external:
      name: site1_default
  site2:
    external:
      name: site2_default
```

Dockerfile:

Este archivo lleva los comandos que se ejecutaran cuando se active el **docker-compose.yml**:

```
GNU nano 2.9.3 Dockerfile
FROM nginx:1.12

# default conf for proxy service
COPY ./default.conf /etc/nginx/conf.d/default.conf

# NOT FOUND response
COPY ./backend-not-found.html /var/www/html/backend-not-found.html

# Proxy configurations
COPY ./includes/ /etc/nginx/includes/
```

Default.conf:

En este archivo encontraremos la configuración necesaria que utilizará el Dockerfile para poder levantar el contendor:

```

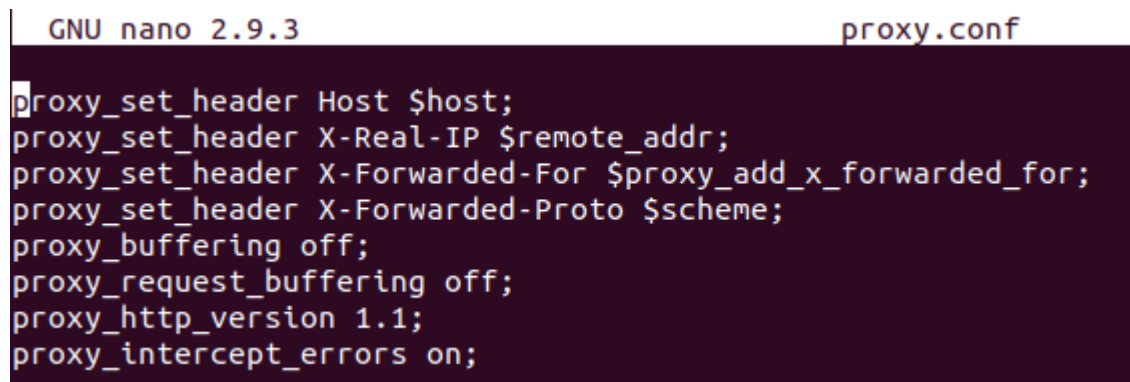
1  # web service1 config.
2  server {
3      listen 80;
4      server_name site1.example.com;
5
6      location / {
7          include /etc/nginx/includes/proxy.conf;
8          proxy_pass http://site1_app_1;
9      }
10
11     access_log off;
12     error_log /var/log/nginx/error.log error;
13 }
14
15 # web service2 config.
16 server {
17     listen 80;
18     server_name site2.example.com;
19
20     location / {
21         include /etc/nginx/includes/proxy.conf;
22         proxy_pass http://site2_app_1;
23     }
24
25     access_log off;
26     error_log /var/log/nginx/error.log error;
27 }
28
29 # Default
30 server {
31     listen 80 default_server;
32
33     server_name _;
34     root /var/www/html;
35
36     charset UTF-8;
37
38     error_page 404 /backend-not-found.html;
39     location = /backend-not-found.html {
40         allow all;
41     }
42     location / {
43         return 404;
44     }
45
46     access_log off;
47     log_not_found off;
48     error_log /var/log/nginx/error.log error;
49 }
```

Includes:

Crearemos una carpeta donde crearemos un archivo con la configuración del proxy.

Proxy.conf:

Aquí encontraremos toda la configuración del proxy:



```

GNU nano 2.9.3 proxy.conf
proxy_set_header Host $host;
proxy_set_header X-Real-IP $remote_addr;
proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
proxy_set_header X-Forwarded-Proto $scheme;
proxy_buffering off;
proxy_request_buffering off;
proxy_http_version 1.1;
proxy_intercept_errors on;
```

Ejecución:

Una vez creados todos los archivos necesarios utilizaremos los siguientes comandos:

-**sudo docker-compose build**: con este comando crearemos imágenes de los servicios indicados en el **Docker-Compose**.

-**sudo docker-compose up -d**: con este comando levantaremos el contenedor.

```
pablohr@pablohr-Virtual-Machine:~/Escritorio/docker3/reverse-proxy$ sudo docker-
compose build
Building proxy
Step 1/4 : FROM nginx:1.12
--> 4037a5562b03
Step 2/4 : COPY ./default.conf /etc/nginx/conf.d/default.conf
--> Using cache
--> 1a7d819c01ae
Step 3/4 : COPY ./backend-not-found.html /var/www/html/backend-not-found.html
--> Using cache
--> e872a3bc4a05
Step 4/4 : COPY ./includes/ /etc/nginx/includes/
--> aa92ac78a6b9
Successfully built aa92ac78a6b9
Successfully tagged reverseproxy_proxy:latest
pablohr@pablohr-Virtual-Machine:~/Escritorio/docker3/reverse-proxy$ sudo docker-
compose up -d
Creating reverseproxy_proxy_1 ...
Creating reverseproxy_proxy_1 ... done
```

Comprobación:

Ahora utilizaremos el comando “**sudo docker ps**” para ver que se ha levantado el nuevo contendor:

```
pablohr@pablohr-Virtual-Machine:~/Escritorio/docker3/reverse-proxy$ sudo docker
ps
CONTAINER ID   IMAGE                                COMMAND                                  CREATED        ST
ATUS          PORTS          NAMES
46c20cc64b03   reverseproxy_proxy                 "nginx -g 'daemon of..."             15 seconds ago Up
13 seconds    0.0.0.0:80->80/tcp reverseproxy_proxy_1
f1a0b2826ef3   nginx:1.12                         "nginx -g 'daemon of..."             10 minutes ago Up
10 minutes    80/tcp         site2_app_1
4d98a49aa25b   nginx:1.12                         "nginx -g 'daemon of..."             15 minutes ago Up
15 minutes    80/tcp         site1_app_1
```

Configuración archivo hosts:

Por último accederemos al archivo hosts para añadir un par de líneas, para ello haremos “**sudo nano /etc/hosts**” y añadiremos lo siguiente:

```
GNU nano 2.9.3 /etc/hosts
127.0.0.1    localhost
127.0.1.1    pablohr-Virtual-Machine
172.26.127.135 site1.example.com
172.26.127.135 site2.example.com

# The following lines are desirable for IPv6 capable hosts
::1        ip6-localhost ip6-loopback
fe00::0    ip6-localnet
ff00::0    ip6-mcastprefix
ff02::1    ip6-allnodes
ff02::2    ip6-allrouters
```

Comprobación final:

Para finalizar nos dirigiremos a nuestro navegador y escribiremos el nombre que le hayamos puesto a la **IP** en el archivo **hosts**, en nuestro caso **site1.example.com** y **site2.example.com**, como podemos comprobar nos aparecen los dos **.html** distintos según la **URL** que empleemos y apuntando a la misma **IP**:



site1.example.com



site2.example.com

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.