
PrestaShop con Docker y Docker Compose

Implantación de Aplicaciones Web

Curso 2025/2026

Índice general

1	Despliegue de PrestaShop usando contenedores Docker y Docker Compose	1
1.1	Tareas a realizar	1
1.2	Requisitos de la instancia EC2 en AWS	1
1.3	Requisitos del archivo docker-compose.yml	2
1.3.1	Imágenes	2
1.3.2	Networks	2
1.3.3	Docker restart policies	3
1.3.4	Variables de entorno	3
1.3.5	Orden en el que se inician los servicios	4
1.4	Ejemplo de archivo docker-compose.yml utilizando la imagen prestashop/prestashop	4
1.5	Posibles errores que le pueden aparecer	6
1.5.1	Error: mbind: Operation not permitted	6
1.6	Entregables	6
2	Referencias	7
3	Licencia	8

1 Despliegue de PrestaShop usando contenedores Docker y Docker Compose

En esta práctica tendremos que realizar la implantación de un sitio [PrestaShop](#) en [Amazon Web Services \(AWS\)](#) haciendo uso de contenedores [Docker](#) y la herramienta [Docker Compose](#).

1.1. Tareas a realizar

A continuación se describen **muy brevemente** algunas de las tareas que tendrá que realizar.

1. Crear una máquina virtual [Amazon EC2](#). Para evitar posibles problemas con el instalador de PrestaShop asegúrese que su máquina virtual tiene al menos **2 GB de memoria RAM** y **20 GB de almacenamiento EBS**.
2. Instalar y configurar [Docker](#) y [Docker compose](#) en la máquina virtual.
3. Crear un archivo **docker-compose.yml** para poder desplegar los servicios de **PrestaShop**, **MySQL** y **phpMyAdmin**. Deberá utilizar las imágenes oficiales de [Docker Hub](#).
4. Buscar cuál es la dirección IP pública de su instancia en AWS y comprobar que puede acceder a los servicios de **PrestaShop** y **phpMyAdmin** desde una navegador web.
5. Una vez que tenga el archivo **docker-compose.yml** configurado de forma correcta, se propone añadir el servicio **https-portal** para poder acceder al servicio de **PrestaShop** a través de un nombre de dominio por **HTTPS**.

1.2. Requisitos de la instancia EC2 en AWS

Se recomienda que la instancia EC2 de AWS tenga como mínimo las siguientes características:

- **Tipo de instancia:** t2.small
- **Almacenamiento EBS:** 20 GB

1.3. Requisitos del archivo `docker-compose.yml`

1.3.1. Imágenes

Deberá utilizar las imágenes de [Docker Hub](#) que se indican para cada uno de los servicios:

- Servicio: `prestashop`: [bitnami/prestashop](#)
- Servicio: `mysql`: [mysql](#)
- Servicio: `phpmyadmin`: [phpmyadmin/phpmyadmin](#)
- Servicio: `https-portal`: [stevelt/https-portal](#)

1.3.2. Networks

Los servicios definidos en el archivo `docker-compose.yml` deberán usar dos redes:

- `frontend-network`
- `backend-network`

En la red `frontend-network` estarán los servicios:

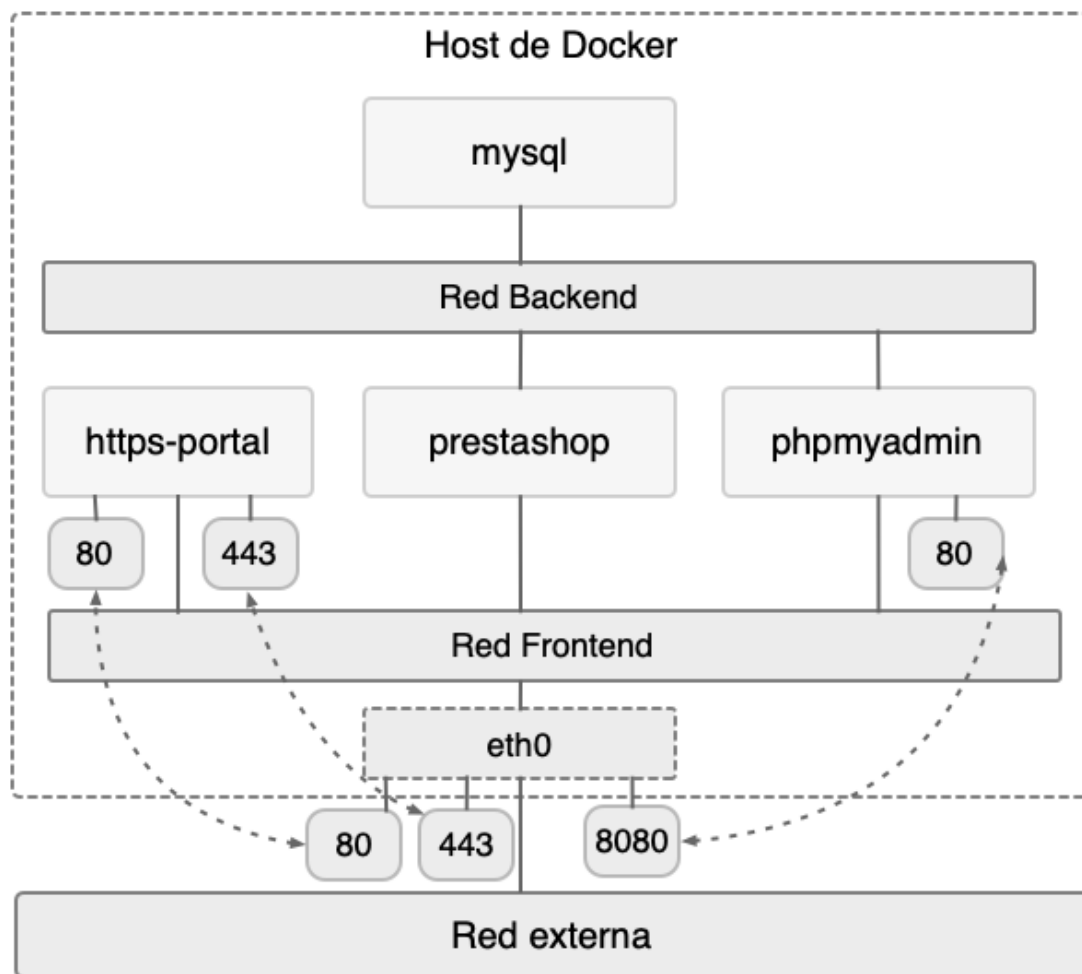
- **`https-portal`**
- **`prestashop`**
- **`phpmyadmin`**

Y en la red `backend-network` sólo estará el servicio:

- **`mysql`**

Sólo los servicios que están en la red `frontend-network` expondrán sus puertos en el host. Por lo tanto, el servicio **`mysql`** no deberá estar accesible desde el host.

A continuación se muestra un diagrama con las redes y los servicios que tiene que crear:



1.3.3. Docker restart policies

Deberá utilizar alguna política de reinicio para que los contenedores se reinicien cada vez que se detengan de forma inesperada.

Se recomienda consultar la [documentación oficial de la opción restart](#).

1.3.4. Variables de entorno

Deberá hacer uso de un archivo **.env** para almacenar todas las variables de entorno que necesite en el archivo **docker-compose.yml**.

En la [documentación oficial](#) puede encontrar más información sobre cómo hacer uso de variables de entorno en el archivo **docker-compose.yml**.

1.3.5. Orden en el que se inician los servicios

Deberá indicar el orden en el que se deben iniciar los servicios con la opción `depends_on`. Se recomienda la lectura del artículo [Control startup and shutdown order in Compose](#)

Para garantizar que el servicio de **MySQL** está listo para aceptar conexiones, deberá utilizar la opción `healthcheck` del archivo `docker-compose.yml`. Se recomienda la lectura del artículo [Healthcheck. Compose file version 3 reference](#).

1.4. Ejemplo de archivo `docker-compose.yml` utilizando la imagen `prestashop/prestashop`

A continuación se muestra una posible solución de la práctica utilizando la imagen de **PrestaShop** `prestashop/prestashop`.

Este ejemplo también está disponible en siguiente repositorio en GitHub:

- <https://github.com/josejuansanchez/iaw-prestashop>

Recuerde que tendrá que modificar el archivo `docker-compose.yml` de ejemplo para utilizar la imagen de PrestaShop de **Bitnami** (`bitnami/prestashop`).

Contenido del archivo `docker-compose.yml`:

```
1 version: '3.3'
2
3 services:
4   prestashop:
5     image: prestashop/prestashop:1.7
6     environment:
7       - PS_INSTALL_AUTO=${PS_INSTALL_AUTO:-1}
8       - DB_SERVER=${DB_SERVER:-mysql}
9       - DB_USER=${MYSQL_USER}
10      - DB_PASSWD=${MYSQL_PASSWORD}
11      - DB_NAME=${MYSQL_DATABASE}
12      - PS_DOMAIN=${PS_DOMAIN}
13      - PS_ENABLE_SSL=${PS_ENABLE_SSL:-1}
14      - PS_FOLDER_INSTALL=${PS_FOLDER_INSTALL:-install-dev}
15      - PS_FOLDER_ADMIN=${PS_FOLDER_ADMIN:-admin-dev}
16      - PS_COUNTRY=${PS_COUNTRY:-es}
17      - PS_LANGUAGE=${PS_LANGUAGE:-es}
18      - ADMIN_MAIL=${ADMIN_MAIL}
19      - ADMIN_PASSWD=${ADMIN_PASSWD}
20     restart: always
21     volumes:
22       - prestashop_data:/var/www/html
23     depends_on:
24       - mysql
25     networks:
26       - frontend_network
27       - backend_network
28
```

```
29  mysql:
30    image: mysql:8
31    environment:
32      - MYSQL_ROOT_PASSWORD=${MYSQL_ROOT_PASSWORD}
33      - MYSQL_DATABASE=${MYSQL_DATABASE}
34      - MYSQL_USER=${MYSQL_USER}
35      - MYSQL_PASSWORD=${MYSQL_PASSWORD}
36    restart: always
37    volumes:
38      - mysql_data:/var/lib/mysql
39    networks:
40      - backend_network
41    security_opt:
42      - seccomp:unconfined
43
44  phpmyadmin:
45    image: phpmyadmin:5
46    restart: always
47    ports:
48      - 8080:80
49    environment:
50      - PMA_HOST=mysql
51    depends_on:
52      - mysql
53    networks:
54      - frontend_network
55      - backend_network
56
57  https-portal:
58    image: steveltn/https-portal:1
59    ports:
60      - 80:80
61      - 443:443
62    environment:
63      DOMAINS: 'localhost -> http://prestashop:80 #local'
64      #DOMAINS: '${PS_DOMAIN} -> http://prestashop:80 #production'
65    volumes:
66      - ssl_certs_data:/var/lib/https-portal
67    depends_on:
68      - prestashop
69    restart: always
70    networks:
71      - frontend_network
72
73  volumes:
74    prestashop_data:
75    mysql_data:
76    ssl_certs_data:
77
78  networks:
79    frontend_network:
80    backend_network:
```

Contenido del archivo .env:

```
1 # Configuración de acceso a la base de datos
2 MYSQL_ROOT_PASSWORD=password
```

```
3 MYSQL_DATABASE=prestashop_db
4 MYSQL_USER=prestashop_user
5 MYSQL_PASSWORD=prestashop_password
6
7 # Configuración de PrestaShop
8 PS_DOMAIN=localhost
9 #PS_DOMAIN=iaw-test.ddns.net
10 ADMIN_MAIL=admin@mail.es
11 ADMIN_PASSWD=admin_password
```

1.5. Posibles errores que le pueden aparecer

1.5.1. Error: mbind: Operation not permitted

Si utiliza la imagen de MySQL etiquetada como **mysql:8**, puede ser que obtenga el siguiente mensaje de error: `mbind: Operation not permitted`.

Para solucionarlo tiene que ejecutar el contenedor sin el perfil `seccomp` (*Secure Computing mode*) predeterminado.

`seccomp` es una característica del kernel de Linux que permite limitar las llamadas al sistema que puede realizar un proceso.

Solución:

```
1  mysql:
2    image: mysql:8
3    ...
4    security_opt:
5      - seccomp:unconfined
```

Referencias:

- [Run without the default seccomp profile.](#)
- [Issue: docker logs mbind: Operation not permitted #303.](#)

1.6. Entregables

En esta práctica habrá que entregar un **documento técnico** con la descripción de los pasos que se han llevado a cabo durante todo el proceso.

El documento debe incluir **como mínimo** lo siguientes contenidos:

- URL del repositorio de GitHub donde se ha alojado el documento técnico escrito en [Markdown](#).
- Descripción de la configuración del archivo `docker-compose.yml` que se ha utilizado en esta práctica.
- Descripción de las acciones que ha realizado durante durante la puesta en producción
- URL del sitio web con HTTPS habilitado.

2 Referencias

- [Repositorio oficial en GitHub de PrestaShop.](#)
- [Archivo docker-compose.yml de ejemplo del repositorio oficial de PrestaShop](#)
- [Guía de usuario de PrestaShop](#)
- [Docker](#)
- [Docker Compose](#)
- [Control startup and shutdown order in Compose](#)

3 Licencia

Esta página forma parte del curso Implantación de Aplicaciones Web de José Juan Sánchez Hernández y su contenido se distribuye bajo una licencia Creative Commons Reconocimiento-NoComercial-CompartirIgual 4.0 Internacional.