3.1 Portainer. Gestión de contenedores

portainer-logo.png

Imagen obtenida de https://www.portainer.io/

Esta herramienta sirve para...

Gestionar los distintos contenedores, imágenes, stacks,... que tengamos en nuestra Raspberry Pi a través de un entorno gráfico en lugar de hacerlo a través del terminal del sistema operativo. Cuenta con una versión BE (Business Edition) y otra CE (Community Edition), usaremos la 2ª.

Web de proyecto y otros enlaces de interés

Página web oficial: https://www.portainer.io/

Repositorio de la versión CE en github: https://github.com/portainer/portainer

Documentación del proyecto: https://docs.portainer.io/

Despliegue

> Si crees que instalar portainer del modo que a continuación se explica es complicado puedes instalarlo a través del método que explicamos en el capítulo 3.3 Linux Media Delivery System (LMDS). No te librará de utilizar la terminal pero quizás te resulte mas amigable.

En la propia documentación podemos encontrar como desplegar Portainer (https://docs.portainer.io/start/install-ce/server/docker/linux). Vamos a recopilar aquí qué hay que hacer y explicar los comandos

```
docker volume create portainer_data
docker run -d -p 8000:8000 -p 9443:9443 --name portainer --restart=always -v
/var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer-
ce:latest
```

En la primera línea creamos un volumen llamado portainer_data

En la segunda línea lanzamos, desplegamos un contenedor:

- -d (--detach): Ejecuta un contenedor en segundo plano.
- -p (--expose): Nos permite indicar qué puerto del contenedor se corresponde con qué puerto de la máquina anfitriona.
- --name: Nos permite establecer el nombre del contenedor.
- --restart: Nos permite establecer qué queremos que ocurra en caso de que el contenedor falle. En este caso establecemos que se reunicie siempre.
- -v (--volume): Nos permite mapear rutas del contenedor con rutas de la máquina anfitriona.
- El último parámetro que aparece en la ruta `portainer/portainer-ce:latest` es la imagen que se va a ejecutar.

> Visto y explicado cómo realizar la instalación según indica la documentación oficial, nosotros/as vamos a hacerlo de otro modo.

La forma que hemos visto con anterioridad funciona. Podéis usarla sin ningún problema. Ahora bien, dado que en este curso desconozco el nivel de partida de cada compañero/a que lo cursa voy a optar por utilizar un modo de despliegue semejante para cada servicio y, por ello, voy a hacer uso de docker-compose. Vamos allá:

Para ello accedemos al terminal y escribimos lo siguiente:

```
cd $HOME
mkdir portainer
cd portainer
nano docker-compose.yml
```

Dentro del fichero escribimos el siguiente contenido:

```
version: '2'
services:

  portainer:
    container_name: portainer
    image: portainer/portainer-ce
    restart: unless-stopped
    ports:
      - 9000:9000
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
      - ./volumes/data:/data
```

Para salir del fichero pulsaremos `control + x` y guardaremos los cambios. Posteriormente ponemos en marcha los contenedores con `docker compose up -d` Aparecerá en pantalla algo similar a

portainer-deploy.png

Elaboración propia

Si queremos comprobar que el contenedor está en marcha podemos ejecutar `docker ps --all` lo que nos mostrará todos los contenedor que hay en la máquina. Si queremos ver si, concretamente, está disponible el que acabamos de crear podemos ejecutar docker `ps --all | grep portainer`. Obteniendo unos resultados similares a los siguientes:

docker-ps-grep-portainer.png

Elaboración propia

También podemos tratar de acceder a la interface gráfica a través de un navegador web. Para ello accedemos a través del navegador la Raspberry Pi y al servicio Portainer del siguiente modo `http://<IP>:puerto` en mi caso tengo configurada la raspberry Pi con la IP `192.168.0.201` y portainer con el puerto `9000` por lo que escribo `http://192.168.0.201:9000` y así accedo a la interface web de portainer.

portainer-index.png

Elaboración propia

Funcionamiento

> En el resto del curso el despliegue de los distintos servicios lo haré siempre a través de comandos pero debes saber que con esta herramienta puedes hacer lo mismo en un entorno gráfico.

Como ya indicamos en la introducción esta herramienta es una interface web para docker lo cual facilitará enormemente la vida a aquellas personas que no estén acostumbradas a trabajar desde una interface de texto (terminal).

Nada mas acceder veremos una imagen como la que hemos visto un par de párrafos más arriba en ella podemos gestionar los usuarios, entornos, logs, parámetros de configuración... de este primer menú lateral no voy a comentaros nada. Si pinchamos en local veremos una imagen como la siguiente:

portainer-local.png

Elaboración propia

En la misma os he remarcado 4 zonas:

- zona 1 (verde): Tenemos acceso a los diferentes contenedores, imágenes, redes y volúmenes de nuestro entorno local.
- zona 2 (azul claro): Acceso a las mismas secciones que teníamos antes de entrar en el entorno local.
- zona 3 (rojo): Información del entorno.
- zona 4 (morado): Similar a la zona 1 con algo de información adicional.

Si accedemos a, por ejemplo, `Containers` tendremos un listado de todos los contenedores descargados:

portainer-containers.png

Elaboración propia

Dónde podemos ver si hay contenedores detenidos, fallando, sanos,... también podemos acceder al detalle de cualquiera de ellos y dentro del detalle detenerlo, reiniciarlo, pausarlo, borralo,... ver los logs, acceder al terminar del contenedor,... todo ello sin necesidad de conocer los comandos

docker que hay por detrás:

portainer-container-details.png

Elaboración propia

portainer-container-details-logs.png

Elaboración propia

Es bastante probable que nunca tengáis que recurrir a estas opciones pero si en alguna ocasión las necesitáis no tendréis que buscar que con `docker logs nombre_del_contenedor --follow` podéis hacer lo mismo que haciendo 4 clicks.

Acceder a las secciones de `Images` o `Volumes` nos puede resultar muy útil para de un vistazo ver, respectivamente, que imágenes o volúmenes no están en uso y así borrarlas para que dejen de ocupar espacio en nuestro disco duro.

portainer-image.png

Elaboración propia

Además, por si fuera poco, nos agrupa los contenedores por stacks de modo que nos facilita ver si los contenedores asociados a un determinado servicio (hay servicios que pueden requerir el funcionamiento de mas de 1 contenedor) están operativos o no.

portainer-stacks.png

Elaboración propia

portainer-stack-details.png

Elaboración propia

Y, ya para terminar pero no por ello menos importante, nos ofrece también la posibilidad de crear contenedores a partir de plantillas de aplicaciones preexistentes. Esta funcionalidad puede permitirnos desplegar servicios en segundos sin conocer ni un solo comando de docker (si bien no es lo deseable):

portainer-templates.png

Elaboración propia

Como nos ocurrirá en servicios posteriores, esta herramienta podría dar por si misma para un curso dedicado. Os presento aquellas cuestiones que me parecen mas relevantes a fin de que seáis

capaces de continuar vosotros/as desde este punto de partida.

Revision #17
Created 2023-02-02 16:01:51 CET by Pablo Ruiz
Updated 2023-07-20 17:08:56 CEST by Pablo Ruiz