

[Docker SecDevOps] Capítulo 5 : Docker Swarm

- gestor de servidores y orquestación embebido en el motor de docker
- managers y workers
- cuando se define un servicio, se define el estado (replicas) y docker swarm se encarga de mantenerlo en ese estado.
- stack: conjunto de servicios desplegados de forma coordinada en swarm
- nodos: (nodo docker) instancia motor docker corriendo en un swarm. Los nodos master distribuyen las tareas entre los nodos workers.
- tareas: definición de una tarea que se ejecuta en un worker.
 - = contenedor
 - servicios globales: servicios que se despliegan en todos los workers del swarm.
 - una vez asignado a un nodo, no se puede mover (a menos que falle y de hecho, se manda una nueva tarea al nuevo nodo)
- balanceador de carga: docker swarm dispone de un balanceador de carga y DNS interno para el reparto de la carga de trabajo

creando un swarm

- `docker swarm init --advertise-addr 192.168.99.100`
- visualizar nodos (desde manager):

```
docker node ls
```

manejando servicios

- `docker service`
 - create
 - inspect
 - logs
 - ls
 - ps
 - rm
 - rollback
 - scale
 - update
- `docker service create --replicas=2 --name webserver nginx:alpine`
- `docker service ls`
- escalar un servicio (subir o bajar el número de réplicas):

```
docker service scale webserver=1
```

- para saber en que nodo se está ejecutando:

```
docker service ps webserver
```

manejando nodos

- `docker node`
 - `demote`
 - `inspect`
 - `ls`
 - `promote`
 - `ps`
 - `rm`
 - `update`
- `demote/promote`: degradar o promocionar un nodo de master→worker o worker→ master (en caso de que falle el master)(queda marcado como **Reachable**)
- se puede marcar un nodo en tres estados para recibir tareas a través de `docker nide update --availability <MODE> <NODE>`:
 - **active**: puede recibir nuevas tareas
 - **pause**: no puede recibir nuevas tareas, pero las ya activas continúan ejecutándose
 - **drain**: no puede recibir nuevas tareas y las actuales son paradas. Para tareas de mantenimiento

stacks

- equivalente **compose** para swarm (de hecho, usan el mismo tipo de configuración desde la v3)
 - se añade **replicas**
- `docker stack`
 - `deploy`
 - `ls`
 - `ps`
 - `rm`
 - `services`
- `docker stack deploy -c docker-compose.yml mystack`
- `docker stack ps mystack`
- [visualizer.yaml](#)

```
visualizer:
  image: dockersamples/visualizer:stable
  ports:
    - "8080:8080"
  volumes:
    - "/var/run/docker.sock:/var/run/docker.sock"
  deploy:
    placement:
      constraints: [node.role == manager]
  networks:
    - webnet
```

- **dockersamples/visualizer:stable** muestra el estado de los nodos del swarm → `docker pull dockersamples/visualizer`

configuraciones y secretos

- configuración: texto plano

- secretos: encriptados y solo disponibles para los servicios asociados.
 - solo disponibles en swarm (se podría hacer con `replica=1`)
 - no pueden superar los 500kb
- `docker config` o `docker secret`
 - `create`
 - `inspect`
 - `ls`
 - `rm`
- `echo «mi secreto» | docker secret create misecreto -`
- `echo «mi configuración» | docker secret create miconfig -`
 - solo aceptan entradas desde **stdin**
- `docker service create --name nynginx --config miconfig --secret misecreto nginx:alpine`
 - las configuraciones se almacenan por defecto como fichero en el *root* del contenedor
 - los secretos se almacenan por defecto como fichero en el **/run/secrets** del contenedor
- `docker service create --name nynginx --config src=miconfig,target=/alternate/my_config --secret src=misecreto,target:/alternate_secret/mysecret nginx:alpine`

limpieza

- nodo worker: `docker swarm leave`
- nodo master: `docker node demote nodo_master && docker swarm leave`
 - `docker swarm leave --force` ← no recomendado

From:

<https://matebcn.eu/> - miguel angel torres egea

Permanent link:

<https://matebcn.eu/info:libros:docker-sec-dev-ops:cap6>

Last update: **06/12/2021 00:38**

