

Uso de Docker Swarm

Creando un Swarm Docker

Ejecute los siguientes comando en una consola y explique que es lo que hacen usando la documentación de Docker Swarm:

Crear máquinas para el cluster

Para crear las máquinas utilizar como nodos con los comandos:

```
docker-machine create -d virtualbox node-1
```

```
docker-machine create -d virtualbox node-2
```

```
docker-machine create -d virtualbox node-3
```

Podemos observar las máquinas recién creadas mediante:

```
docker-machine ls
```

Tome nota de las direcciones IP de cada una de las machine.

Ingresamos a la machine node-1 para inicializar un cluster swarm

```
eval $(docker-machine env node-1)
```

```
docker swarm init \  
--advertise-addr $(docker-machine ip node-1) \  
--listen-addr $(docker-machine ip node-1):2377
```

La salida del comando (algo como esto):

To add a worker to this swarm, run the following command:

```
docker swarm join \  
--token SWMTKN-1-3z26o3av9pq9f6cwalp26efqw727e4tebc5e4ie7jc7bm14r \  
192.168.99.100:2377
```

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.

Anótelo ya que lo necesitará para incorporar las otras machines al cluster.

Ejecutamos la salida del comando anterior en cada uno de los otros nodos (reemplazarlo por la IP y el token que tenga la salida en su caso)

En el node-2

```
eval $(docker-machine env node-2)
```

```
docker swarm join \  
--token SWMTKN-1-3z26o3av9pq9f6cwalp26efqw727e4tebc5e4ie7jc7bm14r \  
192.168.99.100:2377
```

En la machine node-3

```
eval $(docker-machine env node-3)
```

```
docker swarm join \  
--token SWMTKN-1-3z26o3av9pq9f6cwalp26efqw727e4tebc5e4ie7jc7bm14r \  
192.168.99.100:2377
```

Si en algún momento necesitamos saber el token para unirse al cluster ejecutar el siguiente comando

```
docker swarm join-token -q worker
```

Regresamos al nodo node-1 y vemos el estado del cluster con:

```
eval $(docker-machine env node-1)
```

```
docker node ls
```

Analizar y explicar la salida del comando.

Agregando una red virtual para utilizar en el cluster

Para agregar una red para ser utilizada en el cluster ejecutar:

```
eval $(docker-machine env node-1)  
docker network create --driver overlay go-demo
```

Vemos la información de la red creada con:

```
docker network ls
```

Para la máquina creada anteriormente utilizar el comando:

```
docker-machine restart machine1
```

Agregando servicios (containers) al cluster

Para agregar servicio al cluster (un servicio consta de una o más instancias de un container) usamos el siguiente comando:

```
eval $(docker-machine env node-1)  
docker service create --name go-demo-db \  
-p 27017 \  
--network go-demo \  
--detach=false \  
mongo
```

En este caso un container MongoDB. Explique las opciones del comando anterior.

Para ver los servicios del cluster usar el comando:

```
docker service ls
```

Explique la salida del comando anterior.

Para ver los detalles del servicio utilizar:

```
docker service inspect go-demo-db
```

Ahora que tenemos la base de datos agreguemos un container go-demo:

```
eval $(docker-machine env node-1)  
docker service create --name go-demo \  
  -p 8080 \  
  -e DB=go-demo-db \  
  --network go-demo \  
  --detach=false \  
  vfarcic/go-demo
```

Volvamos a analizar la salida de:

```
docker service ls
```

Escalando servicios en el cluster

Para cambiar el número de réplicas de un servicio usar:

```
docker service update --replicas 5 --detach=false go-demo
```

Explique que hace el comando anterior y su salida.

Volvamos a analizar la salida de:

```
docker service ls
```

Y además si deseamos más información sobre el servicio utilizamos:

```
docker service ps go-demo
```

Probando el failover del cluster

Primero analicemos el estado del servicio go-demo con el comando:

```
docker service ps go-demo
```

Ahora simularemos la caída del nodo node-3 con el comando:

```
docker-machine rm -f node-3
```

Volvamos a analizar el estado del servicio go-demo con el comando:

```
docker service ps go-demo
```

Qué ocurrió?

Siga ejecutando el comando cada 10 o 15 segundos y vea si hay cambios.

Qué pasó con los containers que estaban en el nodo node-3?