Docker-machine

Docker-machine es un sistema que permite crear máquinas remotas con Docker y gestionarlas desde un host centralizado.

Dispone de *drivers* para crear y gestionar máquinas en diferentes orígenes:

- Orígenes propios con virtualizadores: Virtualbox, VMware
- En la nube: Amazon, Azure, Digital Ocean, Google Compute Engine, etc.

El *driver generic* nos permite via SSH conectarnos a máquinas que ya están creadas dónde queremos instalar y gestionar Docker-Engine.

VirtualBox Driver

Si disponemos de VirtualBox instalado, podemos crear una máquina virtual donde lanzar nuestros contenedores Docker.

```
$ docker-machine create --driver=virtualbox dockervm1
```

Se pueden ajustar mas opciones, como el tamaño del disco duro o la RAM, en http://docs.docker.oeynet.com/machine/drivers/virtualbox/#usage

```
$ docker-machine create -d virtualbox --virtualbox-disk-size 50000 dockervm2
```

Podemos ver las variables de entorno para nuestra máquina:

```
$ docker-machine env dockervm1
```

Y si queremos que nuestras operaciones de docker se lanzen sobre una máquina en concreto, lanzamos el comando que nos sugiere la propia instrucción anterior:

```
$ eval $(docker-machine env dockervm1)
```

Por ejemplo, si lanzamos un container busybox ahora, se hará en la VM :

```
$ docker run -d -p 80:80 nginxdemos/hello
```

Para ver la web de NGINX habrá que mirar qué IP tiene la máquina creada:

```
$ docker-machine ssh dockervm1 ip addr | grep global
```

En mi caso es 192.168.99.100 así que con esta accederemos mediante el browser y veremos la pantalla de bienvenida de la demo de NGINX.

Fijar una IP a una máquina creada con docker-machine

Si queremos poder usar una configuración de swarm en producción nos será necesario ajustar adecuadamente las IPs de los diferentes nodos. Aunque docker-machine en VirtualBox local no es la situación mas idónea para el propósito de un swarm, aquí tenemos la solución para resolverlo. Aun y así, tiene la contrapartida de que los certificados creados por docker-machine no serán válidos para IPs que no sean las que éste asignó a la máquina orginalmente.

Seguir las indicaciones al final de este post. La idea es crear el archivo `/var/lib/boot2docker/bootsync.sh` con el siguiente comando que se ejecutará en el arranque:

```
ifconfig eth1 192.168.99.101 netmask 255.255.255.0 broadcast 192.168.99.255 up
```

Generic Driver

El Generic Driver sirve para conectarse a una máquina ya creada y que queremos controlar remotamente: http://docs.docker.oeynet.com/machine/drivers/generic/#example

- Crear 3 VMs con VirtualBox:
  - Se recomienda usar Vagrant con el siguiente Vagrantfile

    ```
    Vagrant.configure("2") do |config|
      config.vm.box = "ubuntu/focal64"
      config.vm.hostname = "ubuntu"
      config.vm.network "public_network", ip: "192.168.1.221"
      config.vm.provider "virtualbox" do |vb|
        vb.name = "ubuntu20"
        vb.gui = false
        vb.memory = "4096"
        vb.cpus = 2
      end
    end
    ```
  - Las IPs deben ser visibles desde la máquina host: usar red «solo anfitrión» con IPs tipo 192.168.10.xxx
  - Instalar Docker mediante apt, snap o bien:

    ```
    $ curl -sSL https://get.docker.com/ | sh
    ```
  - Añadir el usuario principal (vagrant) al grupo docker:

    ```
    $ sudo adduser vagrant docker
    ```
  - Instalar el package net-tools en cada nodo (lo necesita docker-machine):

```
$ sudo apt install net-tools
```

- Instalar Docker-machine en el host.
- En el host, configurar nodo1 mediante el driver generic.
  - Comprobar la IP de la máquina donde vamos a conectarnos:

```
$ vagrant ssh -c "ip add | grep global" default
```

  - Si lo hemos hecho con Vagrant, dispondremos de la *private key* en los archivos de .vagrant , y con el usuario *vagrant* :

```
$ docker-machine create --driver generic \
--generic-ip-address=192.168.10.101 \
--generic-ssh-key .vagrant/machines/node1/virtualbox/private_key \
--generic-ssh-user vagrant node1
```

- Repetir el paso anterior para todos los nodos.
- Comprobar el listado de máquinas

```
$ docker-machine ls
```

- Nos podemos conectar a un nodo mediante:

```
$ docker-machine ssh node1
```

- O bien lanzar un comando concreto:

```
$ docker-machine ssh node1 docker ps -a
```

Crear un swarm

Para crear un swarm seguir los pasos indicados aquí.

En realidad la cuestión es ir al nodo *manager* y ejecutar:

```
$ docker swarm init --advertise-addr <MANAGER-IP>
```

La propia instrucción nos dará el comando que se requiere ejecutar en los nodos *workers*.