LAB n°3: Exploitation de Docker Machine

Description du thème

Propriétés	Description
Intitulé long	Exploitation de Docker Machine
Formation concernée	M2-Expert Réseaux
Matière	Cloud Computing
Présentation	Les objectif de ce Labo sont : • apprendre à créer, démarrer, inspecter, arrêter, redémarrer, mettre à jour des hôtes Docker depuis le Docker Machine. • utiliser des drivers de plateforme de gestion de virtualisation locale . Ce Labo comporte 09 activités : -Activité 1 : installation de Docker Machine. -Activité 2 : Découverte du driver virtualbox et des commandes Docker Machine
Prérequis	Commandes de base d'administration d'un système Linux. Notions de virtualisation.
Outils	• Un serveur physique ou virtuel avec une distribution Linux 64 bits (ici Centos 7– version stable actuelle). • Docker 18.03 ou supérieur. • Docker machine • Virtual Box Sites officiels : • https://www.docker.com/ • https://registry.hub.docker.com • https://docs.docker.com/ • https://devopssec.fr/
Mots-clés	Docker Machine- Hôte Docker
Auteur	Slim Marghli
Version	V1.0
Date de publication	Septembre 2020

Déployer et gérer vos hôtes docker avec Docker Machine

Introduction

Docker Machine est un outil de provisioning et de gestion des hôtes Docker (hôtes virtuels exécutant le moteur Docker). Vous pouvez utiliser Docker Machine pour créer des hôtes Docker sur votre ordinateur personnel ou sur le datacenter de votre entreprise à l'aide d'un logiciel de virtualisation tel que VirtualBox ou VMWare, vous pouvez aussi déployer vos machines virtuelles chez des fournisseurs de cloud, tels que Azure, AWS, Google Compute Engine,...

À l'aide de la commande **docker-machine**, vous pouvez démarrer, inspecter, arrêter et redémarrer un hôte géré ou mettre à niveau le client et le moteur Docker et configurer un client Docker pour qu'il puisse communiquer avec votre hôte.

Activité n°1: Installation de Docker Machine

Voici la commande qui permet d'installer Docker Machine sous Linux.

```
$ base=https://github.com/docker/machine/releases/download/v0.16.0 &&
    curl -L $base/docker-machine-$(uname -s)-$(uname -m) >/tmp/docker-machine &&
    sudo mv /tmp/docker-machine /usr/local/bin/docker-machine &&
    chmod +x /usr/local/bin/docker-machine
```

Activité n°2: Découverte des drivers et des commandes Docker Machine

Docker machine utilise le concept des drivers. Les drivers vous permettent depuis votre Docker machine de créer un ensemble complet de ressources sur vos machines virtuelles sur des services tiers tels qu'Azure, Amazon, VirtualBox,...

Avant de vous décrire l'utilisation de certains drivers. Voici d'abord la commande qui permet de créer une machine virtuelle depuis votre Docker Machine:

```
$docker-machine create --drive <DRIVER NAME> <MACHINE NAME>
```

La commande **docker-machine create** télécharge une distribution Linux légère nommée **boot2docker** venant avec le moteur Docker installé et crée et démarre la machine virtuelle. Les options de cette commande peuvent différer selon le type de driver que vous utilisez.

Nous allons voir ci-dessous comment créer des hôtes Docker onpremise avec le driver virtualBox.

Découverte du pilote VirtualBox et utilisation des commandes Docker Machine

La configuration requise pour le driver VirtualBox:

- Virtualbox à partir de la version 5
- Le module de noyau vboxdrv

Pour installer le module de noyau **vboxdrv**, il faut au préalable installer le package **kernel-devel**.

Une fois les deux prérequis de configurations satisfaites, vous pouvez dès lors créer votre hôte Docker en lançant la commande create en utilisant le driver virtualbox avec les options par défaut :

```
$docker-machine create --driver virtualbox vbox-test
```

Ensuite, vérifiez la liste des machines Docker disponible en exécutant la commande suivante :

```
$docker-machine ls
```

Résultat

```
NAME        ACTIVE   DRIVER       STATE     URL
vbox-test   -        virtualbox   Running   tcp://192.168.99.100:2376
SWARM   DOCKER      ERRORS
        v19.03.12
```

D'après le résultat, notre hôte vbox-test est bien présent avec l'état Running et possède le moteur docker en version v19.03.12.

Si vous retournez à la fin du résultat de la commande **docker-machine create**, vous remarquerez le message suivant (traduit en français) :"**To see how to connect your Docker Client to the Docker Engine running on this virtual machine, run: docker-machine env vbox-test**". Cette manipulation, va nous permettre de récupérer les variables d'environnements de la nouvelle VM à exporter.

```
$docker-machine env vbox-test
```

Résultat

```
export DOCKER_TLS_VERIFY="1"
export DOCKER_HOST="tcp://192.168.99.100:2376"
export DOCKER_CERT_PATH="/home/user1/.docker/machine/machines/vbox-test"
export DOCKER_MACHINE_NAME="vbox-test"
# Run this command to configure your shell:
# eval $(docker-machine env vbox-test)
```

Le résultat nous indique clairement que si on souhaite utiliser le moteur Docker de la machine virtuelle sur notre shell courant il faut alors utiliser la commande suivante :

```
eval $( docker-machine env vbox-test)
```

En exécutant cette commande sur votre shell courant, alors n'importe quelle commande Docker que vous exécuterez, sera dorénavant directement prise en compte par votre hôte Docker vboxt-test et non plus par votre hôte maître.
Par ailleurs si vous souhaitez vérifier sur quelle hôte Docker se lanceront vos prochaines commandes docker alors soit vous vérifiez si une étoile existe dans la colonne ACTIVE de la commande **docker-machine ls**. Soit plus simple encore, vous lancez la commande suivante :

```
$docker-machine active
Résultat
vbox-test
```

Le résultat nous indique distinctement, que nos futurs commandes docker sur le shell courant s'exécuteront directement sur la machine Docker vbox-test.
Afin de vous prouver que c'est effectivement le cas, je vais télécharger et exécuter l'image nginx :

```
$docker run -d -p 8000:80 --name vbox-test-nginx nginx
```

À présent, ouvrez un nouveau terminal et vérifiez les conteneurs disponibles, vous verrez ainsi que vous ne retrouverez pas le conteneur vbox-test-nginx créé précédemment :

```
$docker ps
Résultat
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS
PORTS               NAMES
```

Connectez vous à la machine Docker à l'aide de la commande suivante :

```
$docker-machine ssh vbox-test
```

Pour vous assurer que le client Docker est automatiquement configuré au début de chaque session de shell, vous pouvez alors intégrer la commande **eval $(docker-machine env vbox-test)** dans votre fichier ~/.bash_profile.
Si vous pensez avoir fini d'utiliser une machine Docker, vous pouvez l'arrêter avec la commande **docker-machine stop** et la redémarrer plus tard avec la commande **docker-machine start**.

```
$docker-machine stop vbox-test
$docker-machine stop vbox-test
```

Vous pouvez surchargez les ressources allouées automatiquement par défaut à hôte Docker en utilisant les options du driver virtualbox.

```
Exemple
Création d’une machine docker avec 30 Go d’espace disque et avec 2Go de ram(1Go par défaut).
$docker-machine create -d virtualbox \
--virtualbox-disk-size "30000" \
--virtualbox-memory "4000" \
```

```
vbox-test-bigger
```

Supprimer vos machines Docker

```
$docker-machine rm <MACHINE NAME>
```

Cette commande aura pour effet de supprimer définitivement la machine Docker de votre plateforme de gestion de virtualisation locale mais aussi de la supprimer de votre fournisseur de cloud, si jamais vous en utilisez un.

Exemple

```
$docker-machine rm  vbox-test-bigger
$docker-machine rm  vbox-test
```

Conclusion

Nous avons utilisé Docker Machine pour créer des hôtes Docker localement mais aussi il est possible de les créer dans le cloud, cela nous montre à quel point il est facile de déployer et des machines Docker n'importe où et de centraliser la gestion de ces VMs depuis une seule machine maître.

Aide-mémoire

```
## Créer une machine Docker
docker-machine create -d <DRIVER NAME> <MACHINE NAME>
    -d ou --driver : choisir un driver

## Rendre une machine Docker active
eval $(docker-machine env <MACHINE NAME>)

# Lister les machines Docker
docker-machine ls

# Vérifier quelle est la machine Docker active dans le shell courant
docker-machine active
## Supprimer un ou plusieurs machine(s) Docker
docker-machine rm <MACHINE NAME>
    -f ou --force : forcer la suppression

## Se connecter en ssh sur une machine Docker
docker-machine ssh <MACHINE NAME>

## Stopper une machine Docker
docker-machine stop <MACHINE NAME>

## Démarrer une machine Docker
docker-machine start <MACHINE NAME>

## Redémarrer une machine Docker
docker-machine restart <MACHINE NAME>
```

```
## Récolter des informations sur une machine Docker
docker-machine inspect <MACHINE NAME>

## Récupérer les variables d'environnements d'une machine Docker
docker-machine env <MACHINE NAME>

## Mettre à niveau une machine Docker vers la dernière version de Docker
docker-machine upgrade <MACHINE NAME>
```