

Environnement de développement avec Vagrant et Docker

Maxence Bothorel, Thibaut Crouvezier

Licence professionnelle ASRALL,
IUT Nancy Charlemagne,
Nancy

22/01/2015

Sommaire

- 1 Introduction
- 2 Vagrant
- 3 Docker

Intro

Sommaire

1 Introduction

2 Vagrant

- Introduction
- Utilisation basique
- Configuration
- Utilisation avancée

3 Docker

Vagrant



- Développer en Ruby sous licence MIT
- Créé en 2010 par Mitchell Hashimoto

Vagrant



- Développer en Ruby sous licence MIT
- Créé en 2010 par Mitchell Hashimoto
- Utilise des machines virtuelles
- Public cible : Les développeurs

Utilisation

- vagrant init hashicorp/trusty64
- vagrant up
- ssh

Utilisation

- `vagrant init hashicorp/trusty64`
- `vagrant up`
- `ssh`

Vagrantfile

Les Vagrantfile sont les fichiers de configurations. Vagrant va chercher le Vagrantfile en remontant l'arborescence.

Vagrantfile

Il existe plusieurs Vagrantfiles. Ils sont tous lus dans un certain ordre :

- 1 Le Vagrantfile téléchargé avec la machine.
- 2 Un Vagrantfile dans le dossier `/home/user/.vagrant.d`.
- 3 Le Vagrantfile créé lors de la commande `vagrant init`.
- 4 Si le dernier n'existe pas : un Vagrantfile qui concerne plusieurs machines.
- 5 Si le dernier n'existe pas non plus : un Vagrantfile qui concerne l'hyperviseur.

Quelques exemples

Il existe 4 types de configurations :

- `config.vm` : la machine virtuelle en général
- `config.ssh` : la configuration SSH de la machine hôte
- `config.winrm` : la connexion à une machine Windows
- `config.vagrant` : concerne l'hôte

```
Vagrant.configure(2) do |config|  
  config.vm.network "forwarded_port", guest : 80, host : 8080  
  config.vm.network "private_network", type : "dhcp"  
  config.ssh.username  
  config.ssh.private_key_path  
  config.winrm.host  
end
```

Partage de dossiers

Il existe plusieurs type de partage de dossiers :

- Partage via Samba, uniquement pour Windows
- Partage via l'hyperviseur, par défaut
- Partage via rsync avec rsync-auto
- Partage via NFS, recommandé

Un exemple ?

```
config.vm.synced\_folder "src/", "/srv/website",  
  create: true, disabled: false, type: nfs
```

Il existe plus d'options pour gérer les partages NFS :

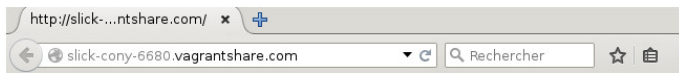
- `nfs_export` : S'il est à "false", vagrant ne modifiera pas le fichier `/etc/exports`
- `nfs_udp` : Le protocole de transport à utiliser
- `nfs_version` : détermine le protocole de transport à utilisé

Partage sur le réseau

On peut aussi partager toute la machine à travers Internet :

- `vagrant login`
- `vagrant share`
- `vagrant share [--ssh] [--http]`
- `vagrant connect [--ssh]`

Partage sur le réseau



Je suis Apache sur la box de Vagrant !

```
maxence@maxence-desktop ubuntu % vagrant share
=> default: Detecting network information for machine...
default: Local machine address: 127.0.0.1
default:
default: Note: With the local address (127.0.0.1), Vagrant Share can only
default: share any ports you have forwarded. Assign an IP or address to your
default: machine to expose all TCP ports. Consult the documentation
default: for your provider ('virtualbox') for more information.
default:
default: Local HTTP port: 8080
default: Local HTTPS port: disabled
default: Port: 2222
default: Port: 8080
=> default: Checking authentication and authorization...
=> default: Creating Vagrant Share session...
default: Share will be at: slick-cony-6680
=> default: Your Vagrant Share is running! Name: slick-cony-6680
=> default: URL: http://slick-cony-6680.vagrantshare.com
=> default:
=> default: You're sharing your Vagrant machine in "restricted" mode. This
=> default: means that only the ports listed above will be accessible by
=> default: other users (either via the web URL or using `vagrant connect`).
[]
```

Partage sur le réseau

```
maxence@maxence-laptop ~ % vagrant share --ssh
=> default: Detecting network information for machine...
default: Local machine address: 127.0.0.1
default:
default: Note: With the local address (127.0.0.1), Vagrant Share can only
default: share any ports you have forwarded. Assign an IP or address to your
default: machine to expose all TCP ports. Consult the documentation
default: for your provider ('virtualbox') for more information.
default:
default: An HTTP port couldn't be detected! Since SSH is enabled, this is
default: not an error. If you want to share both SSH and HTTP, please set
default: an HTTP port with '--http'.
default:
default: Local HTTP port: disabled
default: Local HTTPS port: disabled
default: SSH Port: 2222
default: Port: 2222
=> default: Generating new SSH key...
default: Please enter a password to encrypt the key:
default: Repeat the password to confirm:
default: Inserting generated SSH key into machine...
=> default: Checking authentication and authorization...
=> default: Creating Vagrant Share session...
default: Share will be at: exquisite-reindeer-4943
=> default: Your Vagrant Share is running! Name: exquisite-reindeer-4943
=> default:
=> default: You're sharing your Vagrant machine in "restricted" mode. This
=> default: means that only the ports listed above will be accessible by
=> default: other users (either via the web URL or using 'vagrant connect').
=> default:
=> default: You're sharing with SSH access. This means that another user
=> default: simply has to run 'vagrant connect --ssh exquisite-reindeer-4943'
=> default: to SSH to your Vagrant machine.
=> default:
=> default: Because you encrypted your SSH private key with a password,
=> default: the other user will be prompted for this password when they
=> default: run 'vagrant connect --ssh'. Please share this password with them
=> default: in some secure way.
```

Partage sur le réseau

```
maxence@maxence-laptop ~ % vagrant connect --ssh exquisite-reindeer-4943
Loading share 'exquisite-reindeer-4943'...
The SSH key to connect to this share is encrypted. You will require
the password entered when creating the share to decrypt it. Verify you
access to this password before continuing.

Press enter to continue, or Ctrl-C to exit now.
Password for the private key:
Executing SSH...
Welcome to Ubuntu 12.04.5 LTS (GNU/Linux 3.2.0-72-virtual i686)

 * Documentation:  https://help.ubuntu.com/

System information as of Tue Jan 20 00:13:22 UTC 2015

System load:  0.05               Processes:            71
Usage of /:   2.4% of 39.37GB    Users logged in:     0
Memory usage: 7%                IP address for eth0: 10.0.2.15
Swap usage:   0%

Graph this data and manage this system at:
https://landscape.canonical.com/

Get cloud support with Ubuntu Advantage Cloud Guest:
http://www.ubuntu.com/business/services/cloud

New release '14.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Tue Jan 20 00:08:33 2015 from 10.0.2.2
vagrant@vagrant-ubuntu-precise-32:~$
```


L'environnement multi machine

Le but de cette fonctionnalité est de regrouper la configuration de plusieurs machines dans un seul Vagrantfile :

```
Vagrant.configure(2) do |config|  
  config.vm.define "precise" do |precise|  
    precise.vm.box = "hashicorp/precise32"  
  end  
  config.vm.define "trusty", autostart: false do |trusty|  
    trusty.vm.box = "hashicorp/trusty64"  
  end  
end
```

Attention

Certaines commandes devront être complétées du nom de l'environnement, comme *vagrant ssh precise*

Les provisions shell

Les provisions permettent d'automatiser des installations ou scripts lors de la commande `vagrant up` :

```
config.vm.provision "apache", inline: <--SHELL
  sudo apt-get update
  sudo apt-get install -y apache2
SHELL
```

- `vagrant provision`
- `vagrant provision --provision-with apache sql`
- `vagrant up --no-provision`

Les provisions shell

Il est possible de combiner les provisions avec un environnement multi-machine :

```
config.vm.provision "mysql", type: "shell",  
  inline: "apt-get install mysql-server"  
  
config.vm.define "web" do |web|  
  web.vm.provision "php",  
    inline: "apt-get install php5"  
  web.vm.provision "apache2", type: "shell",  
    inline: "apt-get install apache2"  
end
```

Les provisions file

La provision nommée "file" permet d'envoyer un fichier vers le système invité :

```
config.vm.provision "file", source: ".zshrc", destination "/home/user/.
```

Les provisions Docker

Les provisions Docker permettent de créer, déployer, configurer et démarrer des conteneurs :

Sommaire

- 1 Introduction
- 2 Vagrant
- 3 Docker
 - Introduction
 - Installation
 - Utilisation des Conteneurs

- Développé en Go, sous licence Apache 2.0
- Créer par Salomon Hykes en Mars 2013
- Architecture légère
-

- Installation rapide depuis le dépôt Debian 8
- Installation sous OS X et Windows via une machine virtuelle
- Récupération d'une image sur le dépôt Docker

- Qu'est-ce qu'un conteneur ?
- Lancer un conteneur à partir d'une image
- Déployer des outils avec le conteneur