
VAGRANT

- Automatiser la création de machines virtuelles
- Partager des environnements de VM avec d'autres utilisateurs.
- Basé sur l'utilisation de la CLI dans les environnements des machines virtuelles
- Objectif de créer des environnements qui sont similaires ou identiques tant que possible avec les serveurs de production
- Répliquez facilement la production sur une Dev **Box**.
- Multiplateforme (Windows, Linux, OS X).
- Pas besoin de configurer encore et encore des éléments d’environnement

Getting started with Vagrant

```
$ vagrant box add hashicorp/precise32
```

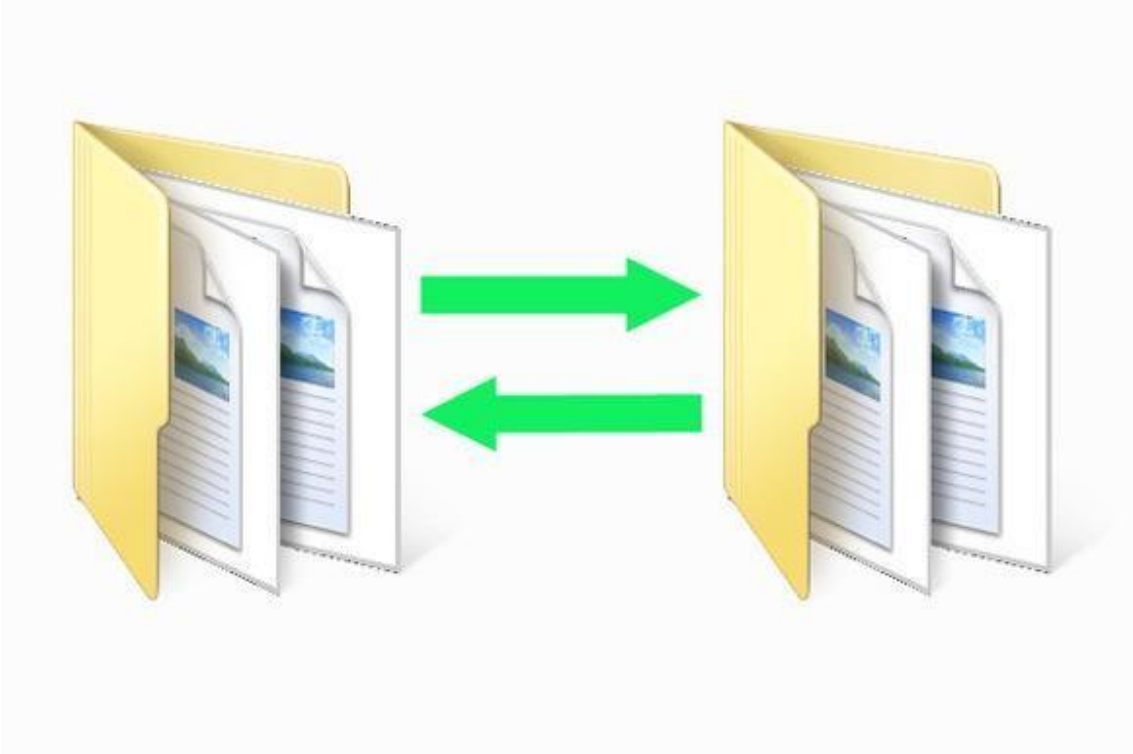
```
$ vagrant init hashicorp/precise32
$ vagrant up
```

```
$ vagrant ssh
```

Vagrantfile

```
Vagrant.configure("2") do |config|
  # Configuration de la box
  config.vm.box = "ubuntu/precise64"

  # Configuration des ressources
  config.vm.provider "virtualbox" do |vb|
    vb.memory = "1024" # 1 GB de RAM
    vb.cpus = 2        # 2 CPUs
  end

  # Configuration du réseau
  config.vm.network "forwarded_port", guest: 80, host: 8080

  # Configuration du dossier partagé
  config.vm.synced_folder "./vagrant_data", "/vagrant_data"

  # Configuration de la provision avec Ansible
  config.vm.provision "ansible" do |ansible|
    ansible.playbook = "provision.yml"
  end
end
```

Synced folder



Providers

- ❏ Boot d’instances EC2 ou VPC.
- ❏ SSH vers les instances.
- ❏ Définir des configurations spécifiques par region afin que Vagrant gère les machines dans de multiple régions.
- ❏ Packager des instances en boxes compatibles vagrant-aws

Vagrant AWS Provider

Vagrant AWS Provider

```
Vagrant.configure("2") do |config|
  config.vm.box = "dummy"

  config.vm.provider :aws do |aws, override|
    aws.access_key_id = "YOUR KEY"
    aws.secret_access_key = "YOUR SECRET KEY"
    aws.session_token = "SESSION TOKEN"
    aws.keypair_name = "KEYPAIR NAME"

    aws.ami = "ami-7747d01e"

    override.ssh.username = "ubuntu"
    override.ssh.private_key_path = "PATH TO YOUR PRIVATE KEY"
  end
end
```


```
$ vagrant plugin install vagrant-aws
...
$ vagrant up --provider=aws
...
```

Provisioners

Provisioning

```
#!/usr/bin/env bash

apt-get update
apt-get install -y apache2
if ! [ -L /var/www ]; then
  rm -rf /var/www
  ln -fs /vagrant /var/www
fi
```

```
Vagrant.configure("2") do |config|
  config.vm.box = "hashicorp/precise32"
  config.vm.provision :shell, path: "bootstrap.sh"
end
```

Multi-Machine

- Vagrant est capable de définir et de contrôler plusieurs machines invitées par Vagrantfile.
- Ces machines sont généralement capables de travailler ensemble ou sont d’une manière ou d’une autre associées les unes aux autres
- Modélisation précise d'une topologie de production multi-serveurs, telle que la séparation d'un serveur Web et d'un serveur de base de données.
- Modélisation d'un système distribué et de la manière dont ils interagissent les uns avec les autres.
- Test d'une interface, telle qu'une API vers un composant de service.

Multi-Machine

```
Vagrant.configure("2") do |config|
  config.vm.provision "shell", inline: "echo Hello"

  config.vm.define "web" do |web|
    web.vm.box = "apache"
  end

  config.vm.define "db" do |db|
    db.vm.box = "mysql"
  end
end
```

Multi-Machine

```
Vagrant.configure("2") do |config|
    config.vm.provision :shell, inline: 'echo A'
    config.vm.define :testing do |test|
        test.vm.provision :shell, inline: 'echo B'
    end
    config.vm.provision :shell, inline: 'echo C'
end
```

vagrant up web - vagrant ssh web
vagrant up db - vagrant ssh db


VAGRANT