Administration Système

Gérer son environnement de développement avec Vagrant

Lucas Nussbaum

lucas.nussbaum@univ-lorraine.fr

Licence professionnelle ASRALL

Administration de systèmes, réseaux et applications à base de logiciels libres

Vagrant

- Outil pour gérer des environnements de développement ou de test
- Objectif : environnements similaires aux serveurs de production
  - Eviter le « *mais pourtant ça marche chez moi !* »
  - Sans contraintes fortes sur la machine du développeur
    - Linux, Windows, Mac OS X supportés
  - Rapide à mettre en place
  - Permettant de changer d'environnement selon le projet
  - Pour tester des montées de version, des nouvelles configuration
- Solution basée sur la virtualisation
  - Par défaut, utilise VirtualBox
  - D'autres *providers* disponibles, notamment des clouds publics
- Écrit en Ruby

Premiers pas

- Si Debian 10 :
  - VirtualBox n'est pas disponible dans la version *stable* de Debian. Nous allons utiliser un *backport* non-officiel, disponible sur `https://people.debian.org/~lucas/virtualbox-buster/`.
  - Suivre les instructions pour installer VirtualBox sur cette page
- Si Debian 9 :
  - VirtualBox n'est pas disponible dans la version *oldstable* de Debian, mais est disponible dans les *backports*. Modifiez votre fichier `/etc/apt/sources.list` pour ajouter (tout sur la même ligne) :
    `deb http://deb.debian.org/debian/ stretch-backports main contrib non-free`
  - Puis `apt-get update ; apt-get install virtualbox`
- Ensuite : installer Vagrant
  `apt-get update ; apt-get install vagrant`
- Choisir une *box* vagrant (image de machine virtuelle déjà préparée)
  - De très nombreuses déjà disponibles, par exemple sur `https://app.vagrantup.com/boxes/search`

Suite ...

- Démarrer : créez un répertoire (`tp-intro-vagrant`), allez dedans, et ...

```
mkdir tp-intro-vagrant ; cd tp-intro-vagrant
vagrant init debian/contrib-jessie64
```

  Cela crée un fichier `Vagrantfile` contenant une configuration par défaut pour Debian 8 Jessie, en 64 bits.
  L'image "contrib" de Debian inclut le support pour les répertoires synchronisés (voir plus loin).

Commandes principales

- `vagrant up` : récupérer l'image et la démarrer
- `vagrant ssh` : se connecter sur l'image.
  - Par défaut, Vagrant configure un répertoire synchronisé (*synced folder*) pour que le répertoire contenant le `Vagrantfile` soit disponible dans la machine virtuelle, dans le répertoire `/vagrant`
- `vagrant halt` : arrêter la machine virtuelle
- `vagrant reload` : Redémarre (reboote) la machine virtuelle
- `vagrant destroy` : Détruit (supprime) la machine virtuelle
- `vagrant box` : gestion des box (par exemple `vagrant box list`)

Plusieurs projets, plusieurs Vagrantfiles

- Les commandes `vagrant` s'appliquent sur l'environnement défini par le `Vagrantfile` du répertoire courant (ou le premier trouvé quand on remonte la hiérarchie de répertoires)
- Pour avoir plusieurs environnements, le plus simple est d'avoir plusieurs répertoires, avec chacun son `Vagrantfile`

```
~
|-- tp-intro-vagrant
|   `-- Vagrantfile
`-- tp-web
    |-- pages
    |   `-- index.html
    `-- Vagrantfile
```

Travaux pratiques

1. Trouvez et démarrez une image Ubuntu 18.04 LTS
2. Vérifiez que vous arrivez à vous y connecter
3. Parcourez le contenu fichier `Vagrantfile`, qui contient de nombreux exemples de configuration
4. Dans la machine virtuelle, installez un serveur web Apache2 (`apt-get install apache2`)
5. Accédez au contenu exposé par le serveur web de la VM grâce à votre navigateur. Il est nécessaire de configurer un `forwarded_port` dans le `Vagrantfile` (par exemple le port 8080 – le serveur web de la VM sera alors accessible dans votre navigateur sur `http://localhost:8080/`)
6. Ajoutez un répertoire synchronisé (*synced folder*) pour partager un répertoire local (par exemple le sous-répertoire "pages") vers le répertoire /var/www/html/test. Créez un fichier HTML basique. Vérifiez que vous pouvez modifier les fichiers sur votre machine hôte, et voir l'effet des modifications avec votre navigateur en interrogeant le serveur web de la VM.

Travaux pratiques (2)

7. Configurez la machine virtuelle (dans `Vagrantfile`) pour :
   - Démarrer en mode graphique (affichage de la *console* de la machine virtuelle lors du boot)
   - Augmenter le nombre de coeurs de CPU attribués
   - Augmenter la mémoire vive attribuée
8. Configurez l'installation au démarrage des paquets logiciels suivants : `tree`, `multitail`. Ce sont deux outils utiles pour les administrateurs systèmes. Pour cela, utilisez un *provisioner* de type *shell*.
9. Créez maintenant un Vagrantfile minimal (sans lignes inutiles) qui va créer un environnement de développement comprenant l'ensemble des points précédents (Ubuntu, installation d'Apache2, configuration du `forwarded_port`, répertoire synchronisé supplémentaire, modification des coeurs et de la mémoire, interface graphique, etc.)

Travaux pratiques (3)

10. Dans un autre Vagrantfile, décrivez maintenant une configuration multi-machines (voir `https://www.vagrantup.com/docs/multi-machine/`), avec :
    - Une machine sous Debian *jessie*
    - Une machine sous Debian *stretch*
    - Une machine sous Debian *buster*

    Configurez également un réseau privé (*private network*) et vérifiez qu'elles arrivent à communiquer entre elles avec la commande ping, en utilisant leurs adresses IP.
    Puis installez le paquet libnss-mdns dans chaque VM avec un *provisioner*. Regardez quel est le service fourni par ce paquet. Vérifiez que cela fonctionne.

Rendre par mail (à `lucas.nussbaum@univ-lorraine.fr`) un court rapport (format PDF) présentant :

- Le Vagrantfile obtenu à la question 9
- Le Vagrantfile obtenu à la question 10, et un exemple montrant le résultat obtenu avec libnss-mdns à la question 10
- Tout commentaire utile