Lab 3 – Premier Box

Le moment est venu de créer votre première box. Dans cet atelier, vous adopterez une procédure manuelle: initialisez un nouveau box de base à l'aide de la version vierge d'Ubuntu 14.04 puis installez le logiciel à la main, en exécutant les commandes appropriées. Une fois que l'invité contient les packages et les outils nécessaires, vous l'exportez dans un fichier.

Il s'agit d'une solution basique pour générer un nouveau box de base. Même si cela peut être un processus fastidieux, les possibilités qu'il offre sont tout simplement géniales. Avec votre box à portée de main, vous pouvez introduire Vagrant dans votre flux de travail.

La tâche à accomplir

Supposons que vous êtes responsable de la mise en place de l'environnement de développement pour le prochain projet de votre entreprise veut au lancement: un blog de co - écrit par tout le personnel. Le blogging plate - forme sera être Jekyll, qui est un outil qui fonctionne en mode hors connexion pour traiter un ensemble de texte des fichiers écrits dans MarkDown langue avec certains modèles et la configuration des fichiers dans un complet ensemble de HTML statiques fichiers qui peuvent être servis a ux clients.

Choix d'un box de base et initialisation d'un nouveau projet

Supposons que vous êtes un ingénieur système et votre rôle est de mettre en place l'environnement de développement. Le résultat de votre travail doit être une seule box que les développeurs vont utiliser pour travailler sur un projet.

Lorsque vous avez besoin pour produire un nouveau box, la meilleure solution est

d'étendre l'un des base-boxes existants.

Si vous voulez préparer un environnement développement à base Ubuntu, vous pouvez

choisir l'un des boxes pris en charge par Ubuntu:

- ubuntu/xenial64
- ubuntu/precise32
- ubuntu/focal64

Si vous préférez CentOS, Debian, Fedora ou FreeBSD, vous pouvez utiliser des boxes prises en charge par Chef Software, Inc.:

- chef/centos-6.5
- chef/debian-7.4
- chef/fedora-20
- chef/freebsd-9.2

D'autres boxes peuvent être trouvées sur le site app.vagrantup.com.

Lorsque vous décidez que le base-box que vous souhaitez utiliser, aller à votre répertoire home et créer un nouveau dossier nommé first-box-factory/ :

```
# Host
$ cd
$ mkdir first-box-factory
$ cd first-box-factory
```

Initialisez un nouveau projet en utilisant Vagrant:

```
# Host
$ vagrant init -m ubuntu/focal64
```

Lancez la VM:

```
# Host
$ vagrant up
```

Installation du logiciel nécessaire

Lorsque le système d'exploitation invité est prêt, vous pouvez démarrer la session SSH et installer les logiciels nécessaires. Parce que Jekyll repose sur le Ruby et JavaScript, vous avez à installer les éléments suivants:

- NodeJS
- Rubis
- Jekyll

Vous pourrez également installer Lynx, qui est un navigateur qui fonctionne dans un terminal. Lynx est un outil utile pour tester les sites Web servis par un système d'exploitation invité dans une session SSH . Et parce que vous utilisez Ubuntu, la première chose que vous devez faire est de mettre à jour le système.

```
# Host
$ vagrant ssh
```

Une fois dans le système d'exploitation invité, Nous allons commencer par mettre à jour notre liste de paquets pour nous assurer que nous avons les dernières informations sur les dernières versions des paquets et leurs dépendances:

```
sudo apt-get update
```

Ensuite, nous installerons Ruby et ses bibliothèques de développement ainsi que make et build-essential pour que les bibliothèques de Jekyll se compilent une fois que nous avons installé Jekyll:

```
sudo apt -y install make build-essential ruby ruby-dev
```

Lorsque cela sera terminé, nous passerons à l'ajout de deux lignes à notre fichier .bashrc pour demander au gestionnaire de paquets de Ruby **gem** de placer les gems dans le dossier d'accueil de notre utilisateur. Cela évite les complications pouvant résulter d'installations à l'échelle du système, tout en ajoutant également la commande jekyll à l'utilisateur PATH.

Ouvrez .bashrc en tapant ce qui suit:

```
nano ~/.bashrc
```

Au bas du fichier, ajoutez les lignes suivantes:

```
# Ruby exports
export GEM_HOME=$HOME/gems
export PATH=$HOME/gems/bin:$PATH
```

Enregistrez et fermez le fichier. Pour activer les exportations, exécutez ce qui suit:

```
source ~/.bashrc
```

Lorsque cela sera terminé, nous utiliserons gem pour installer Jekyll lui-même ainsi que Bundler, qui gère les dépendances Gem:

```
gem install jekyll bundler
```

Lorsque l'installation est terminée, vous pouvez vérifier que Jekyll est disponible sur votre système invité:

```
# Guest
$ jekyll --version
```

Ajoutez les règles nécessaires au niveau firwall

```
ufw allow OpenSSH
sudo ufw enable
```

Autorisez le port 4000 de jekyll dans votre firewall

```
sudo ufw allow 4000
```

Vous pouvez fermer la session SSH. La VM est maintenant prête à être mise en box.

Générer un box

Vous avez maintenant le système d'exploitation invité en cours d'exécution, qui comprend tous les logiciels nécessaires. Il est temps de créer le box qui va entraîner exactement la même machine dans chaque boot. Pour ce faire, exécutez cette commande:

```
# Host
$ vagrant package --output first-box-jekyll.box
```

Vagrant propose la sous-commande `vagrant package` pour la boxe. Il doit être exécuté dans un répertoire contenant le système d'exploitation invité (c'est-à-dire dans un répertoire contenant un fichier Vagrantfile).

La commande `vagrant package` peut être utilisée pour une machine virtuelle en cours d'exécution ou arrêtée. Si la machine virtuelle est en cours d'exécution, elle s'arrêtera normalement. La commande produit ensuite un fichier contenant la machine virtuelle en box.

Lorsqu'elle est exécutée sans aucun paramètre, la commande crée un fichier nommé package.box :

```
# host
$ vagrant package
```

Avec le paramètre `--output`, vous pouvez modifier le nom et l'emplacement du fichier.

Dès que `vagrant package` a fini de travail, vous pouvez exécuter `ls` pour vérifier que le fichier nommé **first-box-jekyll.box** a été créé. L'environnement de développement en box est prêt. le box de fichier est prêt, et votre travail comme un ingénieur système est fini.

Liste, installation et suppression des boxes

```
$ vagrant box list
$ vagrant box add
$ vagrant box remove
```

Contrairement aux `vagrant up` et `vagrant package`, ces sous-commandes peuvent être exécutées dans un répertoire arbitraire car elles affectent une installation globale de Vagrant, pas l'instance de VM particulière.

La première commande répertorie les boxes qui ont déjà été installées dans votre système. En fonction de vos actions précédentes , la commande suivante :

```
# Host
$ vagrant box list
```

peut produire les résultats suivants:

```
bentlema/centos-7.2-64 (virtualbox, 1.0.0)
centos/7 (virtualbox, 2004.01)
ubuntu/bionic64 (virtualbox, 20210203.0.0)
ubuntu/focal64 (virtualbox, 20210128.0.0)
```

Les boxes figurant dans la sortie de `vagrant box list` proviennent du répertoire: soit *~/.vagrant.d/boxes/* ou *$VAGRANT_HOME/.vagrant.d/boxes/*, en fonction de la configuration. La liste de l'un de ces répertoires devrait ressembler (dans une certaine mesure) à la sortie de `vagrant box list`:

```
# Host
$ ls -la ~/.vagrant.d/boxes/
$ ls -la $VAGRANT_HOME/.vagrant.d/boxes/
```

Pour ajouter un nouveau box à votre répertoire `.vagrant.d/boxes/`, vous devez utiliser la commande `vagrant box add`. elle prend deux paramètres: le nom du box et l'URL du box , comme suit:

```
# Host
$ vagrant box add [NAME] [ADDRESS]
```

Pour ajouter votre box stockée dans le fichier first-box-jekyll.box sous le nom first-boxjekyll , utilisez cette commande:

```
# Host
$ vagrant box add first-box-jekyll first-box-jekyll.box
```

Lorsque vagrant box add se termine, vous pouvez exécuter à nouveau ce qui `suit:`

```
# Host
$ vagrant box list
```

La sortie doit contenir ceci:

```
first-box-jekyll (virtualbox, 0)
```

Et, bien sûr, le répertoire `.vagrant.d/boxes/` devrait maintenant contenir un autre élément: le sous-répertoire `first-box-jekyll/`.

Pour supprimer un box, vous pouvez utiliser cette commande:

```
# Host
$ vagrant box remove NAME
```

Le paramètre NAME passé à cette commande est le nom de le box tel qu'imprimé par `vagrant box list` .

Étant donné que Vagrant prend en charge plusieurs fournisseurs et la gestion des versions du box, cette commande peut également prendre deux autres paramètres facultatifs :

```
# Host
$ vagrant box remove NAME --provider PROVIDER --box-version VERSION
```

Mais versioning oeuvres que pour les boxes qui sont hébergées à distance, soit dans un Atlas ou serveur HTTP créé avec des informations de versioning.

Pour supprimer le box que vous avez créée dans la section précédente, utilisez ceci:

```
# Host
$ vagrant box remove first-box-jekyll
```

Si vous l'utilisez, pensez à l'ajouter à nouveau car vous avez besoin de le box pour poursuivre la tâche:

```
# Host
$ vagrant box add first-box-jekyll first-box-jekyll.box
```

Using the Box

Si vous fait procéder avec l' exemple jusqu'à à ce moment - là, vous devez avoir le box installée dans le système et prêt à être utilisé. La commande suivante :

```
# Host
$ vagrant box list | grep jekyll
```

devrait produire la ligne unique:

```
first-box-jekyll (virtualbox, 0)
```

Allez à votre répertoire et créer un autre dossier dans lequel le projet sera être stocké:

```
# Host
$ cd
$ mkdir corporate-blog
$ cd corporate-blog
```

Ce projet doit utiliser le box nommée first-box-jekyll . Pour obtenir ce résultat, exécutez cette commande:

```
# Host
$ vagrant init -m first-box-jekyll
```

Vous pouvez maintenant démarrer la VM et ouvrir la session SSH:

```
# Host
$ vagrant up
$ vagrant ssh
```

Une fois dans l'OS invité, accédez au répertoire partagé:

```
# Guest
$ cd /vagrant
```

Générez la première version du blog de l'entreprise:

```
# Guest
$ jekyll new www
```

La précédente commande génère la source de Code du blog de dans Jekyll interne format dans le dossier www sous la forme suivante :

```
$tree www
...
├── 404.html
├── about.markdown
├── _config.yml
├── Gemfile
├── Gemfile.lock
├── index.markdown
├── _posts
│   └── 2021-05-11-welcome-to-jekyll.markdown
└── _site
```

La commande de création du site avec Jekyll doit être exécutée dans le répertoire qui contient les fichiers source Jekyll. Dans ce cas, il est le dossier `/vagrant` de la VM. Les fichiers HTML, Markdown, et les modèles construits sur la base de la configuration sont stockés dans le répertoire `_site/`. Vous pouvez servir les fichiers HTML à partir du répe rtoire `_site/` à l'aide du serveur HTTP intégré de Jekyll en exécutant la commande suivante:

```
# Guest
$ jekyll serve -H 0.0.0.0 --detach
```

Entre autres, la sortie de la commande vous informe comment accéder au site et comment arrêter le serveur: Server address: http://0.0.0.0:4000/ Server detached with pid '2159'. Run 'kill -9 2159' to stop the server.

Par défaut, le serveur HTTP de Jekyll écoute sur le port 4000. Pour visiter le site généré dans votre OS invité, exécutez ceci:

```
# Guest
$ lynx 127.0.0.1:4000
```

Exercice d'application :

1. Préparez un box basé sur Centos/7 contenant les outils d'intégration continu suivants :
   - Java 8
   - Maven 3
   - Jenkins
2. Générez le box « cicd-box.box » basé sur le box créé dans la question 1
3. Téléchargez le box créé dans votre référentiel sur vagrantup.com