

TD 3 : Vagrant

Objectif : Le but de ce TD est de fournir une brève introduction à Vagrant, ses prérequis et un aperçu des commandes de base.

I. *Mise en place et Premier pas avec Vagrant*

Vagrant un outil open-source de gestion de machines virtuelles. Il isole les dépendances et leur configuration dans un environnement unique et cohérent, sans sacrifier aucun des outils existants.

Prérequis

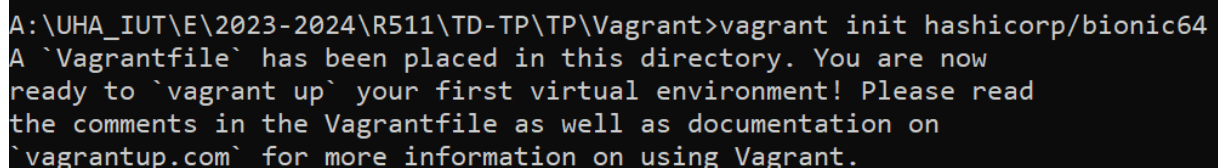
Vous devez tout d'abord installer la dernière version de Vagrant : <https://developer.hashicorp.com/vagrant/install> et de VirtualBox

a) Votre première machine virtuelle

Avec la commande ci-dessous, vous aurez une VM entièrement utilisable avec Ubuntu 18.04 LTS 64 bits. Vous utilisez la fenêtre cmd pour lancer les commandes :

```
vagrant init hashicorp/bionic64
```

Vous devez avoir le résultat suivant :



```
A:\UHA_IUT\E\2023-2024\R511\TD-TP\TP\Vagrant>vagrant init hashicorp/bionic64
A `Vagrantfile` has been placed in this directory. You are now
ready to `vagrant up` your first virtual environment! Please read
the comments in the Vagrantfile as well as documentation on
`vagrantup.com` for more information on using Vagrant.
```

Le nom que vous fournissez après `vagrant init` est le nom de la boîte Vagrant que vous souhaitez utiliser pour votre environnement de développement. Une boîte Vagrant (ou box) est une image préconfigurée de système d'exploitation qui sert de base pour créer des machines virtuelles. Cette notion sera abordée plus tard dans le TD.

Avant de pouvoir passer à l'étape suivante, assurez-vous que Vagrant a créé le fichier Vagrantfile.

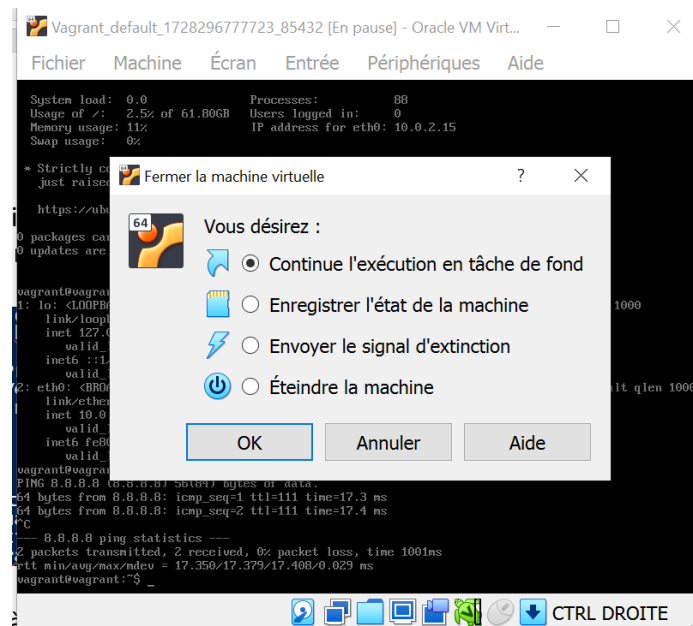
b) Démarrer la machine virtuelle

Maintenant que vous disposez d'un Vagrantfile qui configure votre déploiement, vous pouvez démarrer la machine virtuelle avec la commande : `vagrant up`

Lorsque la machine virtuelle est déployée avec succès, un message indique qu'elle est démarrée et prête :

```
A:\UHA_IUT\E\2023-2024\R511\TD-TP\TP\Vagrant>vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Importing base box 'hashicorp/bionic64'...
==> default: Matching MAC address for NAT networking...
==> default: Checking if box 'hashicorp/bionic64' version '1.0.282' is up to date...
==> default: Setting the name of the VM: Vagrant_default_1728574005775_36413
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
    default: Adapter 1: nat
==> default: Forwarding ports...
    default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
    default: SSH address: 127.0.0.1:2222
    default: SSH username: vagrant
    default: SSH auth method: private key
    default: Warning: Connection reset. Retrying...
    default:
    default: Vagrant insecure key detected. Vagrant will automatically replace
    default: this with a newly generated keypair for better security.
    default:
    default: Inserting generated public key within guest...
    default: Removing insecure key from the guest if it's present...
    default: Key inserted! Disconnecting and reconnecting using new SSH key...
==> default: Machine booted and ready!
==> default: Checking for guest additions in VM...
    default: The guest additions on this VM do not match the installed version of
    default: VirtualBox! In most cases this is fine, but in rare cases it can
    default: prevent things such as shared folders from working properly. If you see
    default: shared folder errors, please make sure the guest additions within the
    default: virtual machine match the version of VirtualBox you have installed on
    default: your host and reload your VM.
    default:
    default: Guest Additions Version: 6.0.10
    default: VirtualBox Version: 7.0
==> default: Mounting shared folders...
    default: /vagrant => A:/UHA_IUT/E/2023-2024/R511/TD-TP/TP/Vagrant
```

1. Dans la liste des informations affichées sur le terminal, à quoi correspondent les forwarding ports ?
2. Toujours selon ces informations, quel sera le nom de votre VM ?
3. Quel est le provider ?
4. Connecter-vous à votre VM en ssh avec la commande `vagrant ssh` vous devez vous connectez à votre VM sans problème
5. Lancer Virtualbox et assurer-vous que votre VM est bien présente dans la liste des VM actives. Faites 'Afficher' la VM et connectez-vous (login & mdp c'est 'vargant').
6. Essayer de fermer votre VM, vous devez avoir le message suivant :



Comme vous pouvez le constater, vous pouvez continuer l'exécution de la VM en tâche de fond.

7. Quelle commande vous devez utiliser pour suspendre puis de defreezer la VM.
8. Quelle commande vous devez utiliser pour arrêter puis redémarrer la VM

c) Détruire la machine virtuelle

Une fois que vous avez terminé, assurez-vous de mettre fin à la machine virtuelle avec la commande : `vagrant destroy`

Confirmez lorsque la CLI vous y invite en saisissant y.

```
A:\UHA_IUT\E\2023-2024\R511\TD-TP\TP\V2>vagrant destroy
default: Are you sure you want to destroy the 'default' VM? [y/N] y
==> default: Forcing shutdown of VM...
==> default: Destroying VM and associated drives...
```

d) Synchronisez les fichiers sur la MH et la VM.

Les machines virtuelles sont pratiques pour le développement, mais peu de personnes souhaitent modifier des fichiers à l'aide d'un simple éditeur basé sur un terminal via SSH. Vagrant synchronise automatiquement les fichiers vers et depuis la machine invitée. De cette façon, vous pouvez modifier des fichiers localement et les exécuter dans votre environnement de développement virtuel.

Par défaut, Vagrant partage votre répertoire de projet (celui contenant le fichier Vagrant) avec le répertoire `/vagrant` de votre VM.

Explorez le dossier synchronisé

1. Sur votre VM, répertoriez les fichiers dans le répertoire vagrant.

- ✚ N.B. Lorsque vous vous connectez via SSH à votre machine, vous êtes dans `/home/vagrant`, qui est un répertoire différent du répertoire `/vagrant` synchronisé.
2. Pour tester la synchronisation entre les deux dossiers, créer un fichier avec un contenu de votre choix sur la VM et vérifier bien qu'il est présent sur le répertoire du projet.

II. *Le Vagrantfile*

Un Vagrantfile est un fichier de configuration utilisé par Vagrant pour définir et gérer les environnements de VM. Il permet de spécifier les paramètres de la VM, les provisionnements, les réseaux, les dossiers synchronisés, etc. Il est écrit en Ruby. L'exécution de la commande `vagrant init` crée automatiquement ce fichier comme vous l'avez pu constater. Cela générera un Vagrantfile par défaut en lien avec la boîte que vous avez utilisée.

Vous pouvez également créer un Vagrantfile manuellement en écrivant le code Ruby nécessaire pour définir la configuration de votre environnement.

Un fichier Vagrant minimal va contenir les lignes suivantes :

```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/bionic64"
end
```

La méthode `Vagrant.configure("2")` définit la version de l'API de configuration utilisée dans le Vagrantfile.

Le bloc `do |config|` est un bloc Ruby qui permet de passer un objet de configuration à la méthode `Vagrant.configure`. `config` est une variable qui représente l'objet de configuration de Vagrant. Cet objet contient toutes les directives et options que vous pouvez définir pour créer votre VM dans le reste du fichier Vagrantfile.

À l'intérieur du bloc, vous pouvez utiliser `config` pour spécifier divers aspects de la VM, comme la boîte à utiliser, la configuration du réseau, le provisionnement, les ressources allouées, etc. Lorsque vous exécutez des commandes Vagrant comme `vagrant up`, Vagrant lit le fichier Vagrantfile et exécute le bloc défini par `Vagrant.configure("2")`. Tout ce qui est défini à l'intérieur de ce bloc sera utilisé pour configurer la VM.

Voici quelques exemples d'instruction qu'on peut appeler à l'intérieur du bloc `Vagrant.configure`

- Définir la boîte à utiliser pour la VM : `config.vm.box = "ubuntu/bionic64"`
- Configurer le réseau pour rediriger le port 8080 de l'hôte vers le port 80 de la VM : `config.vm.network "forwarded_port", guest: 80, host: 8080`

- Synchroniser un dossier local avec la VM : `config.vm.synced_folder`
"./", "/vagrant_data"

III. Les boxes Vagrant

Au lieu de créer une machine virtuelle à partir de zéro, ce qui serait un processus lent et fastidieux, Vagrant utilise une image de base pour cloner rapidement une VM. Ces images de base sont appelées « boîtes » dans Vagrant, et la spécification de la boîte à utiliser pour votre environnement Vagrant est toujours la première étape après la création d'un nouveau Vagrantfile.

Comme vous avez déjà initialisé votre Vagrant avec la commande `vagrant init hashicorp/bionic64`, vous n'avez pas besoin d'ajouter de boîte ; Vagrant en a installé une lorsque vous avez initialisé votre projet. Parfois, il est plus pratique de télécharger des boîtes depuis un dépôt, même si vous ne travaillez pas encore sur un projet Vagrant spécifique. Cela permet de préparer des boîtes pour une utilisation future dans différents projets, sans avoir besoin de les télécharger à nouveau à chaque fois. Pour cela, vous utiliserez la sous-commande `vagrant box add` `'nom_Box'`.

Cela permet de stocker la box localement afin que plusieurs environnements Vagrant puissent la réutiliser.

Vous pouvez avoir la liste des boxes que vous pouvez utiliser sur le lien : <https://portal.cloud.hashicorp.com/vagrant/discover>

a) Gestion des boîtes

1. Récupérer la box de centos/7 avec la commande : `vagrant box add centos/7`
2. Vagrant vous demandera de sélectionner un fournisseur selon ceux proposés par la box. Tapez le chiffre permettant de sélectionner Virtualbox.

Dans la commande précédente, vous remarquerez que le nom des boîtes est composé de deux parties : le nom d'utilisateur et le nom de la boîte, séparés par une barre oblique. Dans l'exemple ci-dessus, le nom d'utilisateur est centos et la boîte est 7. Vous pouvez également spécifier des boîtes via des URL ou des chemins de fichiers locaux.

Les boîtes (ou **boxes**) Vagrant sont stockées localement sur votre machine après leur téléchargement. Le chemin de stockage dépend du système d'exploitation que vous utilisez. Sous Windows le chemin c'est :
C:\Users\VotreNomUtilisateur\.vagrant.d\boxes

3. Vous pouvez avoir la liste des boxes disponibles sur votre machine avec la commande : `vagrant box list`. Combien de boxes vous possédez ?

4. Vous pouvez également supprimer une boîte avec `vagrant box remove <nom_de_la_boîte>` et aussi la mettre à jour avec `vagrant box update`. Supprimer la boîte de hashicorp/bionic64.
5. Vous pouvez ajouter une boîte Vagrant avec un nom personnalisé en utilisant l'option `--name` de la commande `vagrant box add`. Récupérer la boîte `ubuntu/xenial64` en changeant le nom à `my_vBOX`.
6. Ensuite, vous pouvez utiliser ce nouveau nom dans votre Vagrantfile pour référencer la boîte.

Par exemple :

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

b) Utiliser une boîte

Maintenant que vous avez ajouté une boîte à Vagrant, soit en l'initialisant, soit en l'ajoutant explicitement, vous devez configurer votre projet pour l'utiliser comme base. Ouvrez le fichier Vagrantfile et remplacez le contenu par ce qui suit :

```
Vagrant.configure("2") do |config|
  config.vm.box = "centos/7"
end
```

Dans ce cas, le `centos/7` doit correspondre au nom que vous avez utilisé pour ajouter la boîte ci-dessus. C'est ainsi que Vagrant sait quelle boîte utiliser. Si la boîte n'a pas été ajoutée auparavant, Vagrant la téléchargera et l'ajoutera automatiquement lors de son exécution.

Vous pouvez spécifier une version explicite d'une boîte en utilisant `config.vm.box_version` par exemple :

```
Vagrant.configure("2") do |config|
  config.vm.box = "hashicorp/bionic64"
  config.vm.box_version = "1.0.282"
end
```

7. Lancer une VM à partir de votre box et connecter vous en ssh dessus.

✚ Pour que Vagrant prend en considération la nouvelle configuration du fichier Vagrantfile, il faut d'abord supprimer entièrement toute VM créée avec l'ancienne configuration.

c) Personnaliser une boîte

Dans cette partie vous allez personnaliser la boîte que vous avez récupéré en éditant le fichier Vagrantfile.

8. *Pour cela décommenter les lignes de 59 à 65 et 73 à 76*
9. *Décommenter la ligne permettant de rediriger tous flux du port 8080 de votre MH vers le port 80 de votre VM ?*
10. *Lancer une nouvelle VM à partir de votre fichier Vagrantfile modifié. Vérifier bien l'activation du service apache.*
11. *Quel numéro de port il faut utiliser pour accéder à la page web par défaut d'apache de votre VM ?*

d) Créer une nouvelle boîte

Il est possible de rendre disponible votre Box pour d'autres utilisateurs. Pour cela il faut utiliser la commande `vagrant package` pour emballer la machine virtuelle personnalisée en une boîte Vagrant. Assurez-vous que votre VM est éteinte avant de l'emballer. Il est possible également de personnaliser le nom de la boîte à partager : `vagrant package --output my-custom-ubuntu.box`

Cette commande créera un fichier `my-custom-ubuntu.box` contenant votre VM personnalisée au niveau de votre dossier du projet.

12. *Après avoir emballé la boîte, ajoutez-la à votre environnement Vagrant local et Vérifier l'ajout. Attention, ici la commande `vagrant box add` a besoin de deux arguments : le premier est le nom donné à la boîte dans Vagrant. Cela permet de référencer cette boîte dans différents projets ou environnements. Le deuxième argument : Le fichier source à partir duquel la boîte sera ajoutée (le fichier `.box`).*
13. *Créer un nouveau projet Vagrant (un autre dossier) puis initialiser le projet en utilisant votre boîte personnalisée.*
14. *Lancer la VM du 2^e projet, est-ce que le service apache2 est actif ? Est-ce que vous arrivez à accéder à la page web par défaut depuis votre MH ?*
15. *Qu'est-ce que vous remarquez ? Qu'est-ce qu'il faut changer pour rendre accessible la page web depuis votre MH ?*