UNIVERSITÉ CAEN NORMANDIE	BUT Réseaux et Télécommunications - Ifs	2024/2025
	R405	Clément Brisacier
	TP 2 - Vagrant et Ansible	

Généralités :

- se connecter en adminetu sous Debian
- de préférence, utiliser VSCode avec l'extension YAML
- allez télécharger sur Ecampus l'archive « starter kit »
- penser à garder votre code d'une séance à l'autre (Unicloud, ou encore mieux : GitHub !)
- on a besoin de pipenv pour travailler :

```
sudo apt update && sudo apt -y install python3-pip pipenv
```

1. Vagrant

Documentation utile :

- Cheat Sheet Vagrant : https://gist.github.com/jmdilly/497c749ed890c9d5c270315db7335988
- Doc officielle « get started » de Vagrant : https://developer.hashicorp.com/vagrant/tutorials/get-started

1.1. Installation

Commencez par extraire l'archive « Starter kit ». Tout au long de ce TP, les commandes seront à lancer systématiquement depuis ce dossier. Pensez donc bien à utiliser la commande cd pour vous déplacer votre session shell dans le bon dossier !

Ensuite, installez Vagrant à partir du site officiel en suivant la procédure d'installation : https://www.vagrantup.com/downloads.html

1.2. Test de déploiement

Pour ce TP, une première version du `Vagrantfile` vous est proposée. Celui contient la définition de la VM `lb-1` et d'un nombre variable de serveurs DNS et de serveurs Web. Allez jeter un coup d'oeil à ce fichier en essayant de comprendre la syntaxe !

Les variables `$num_dns` et `$num_web` en début de fichier vous permettront d'augmenter le nombre de VM dans votre maquette. Cependant, au début du TP, pour des questions de rapidité de mise en place de l'environnement de maquette, il est préférable, et suffisant, de laisser ces variables à 1.

1. À l'aide de la commande `vagrant up` , démarrer l'ensemble des VM du TP. Soyez patients, ça prends du temps… Vous remarquerez sûrement que Vagrant va télécharger la box correspondante !
2. Connectez-vous à l'une des VM à l'aide de la commande vagrant `ssh <NOM VM>`
3. Détruisez la VM `lb-1` à l'aide de la commande `vagrant destroy <NOM VM>`

Vous voyez, grâce à Vagrant, nous avons automatisé complètement la mise en place et l'installation initiale des VM de notre maquette !

Durant ce TP, en cas de problèmes sur une ou plusieurs VM ou simplement pour tester nos playbooks Ansible entièrement, il suffira d'utiliser les commandes Vagrant pour détruire et redéployer tout ou partie de notre maquette.

Cela étant dit, il s'agit de VMs et non de containers : tout détruire et tout recréer prend quelques minutes. Ainsi, quand vous le pouvez, préférez recréer les VM unitairement.

2. Installation d'Ansible

Documentation utile :

- https://docs.python.org/fr/3/library/venv.html
- https://pipenv.pypa.io/en/latest/

Nous allons utiliser `pipenv` pour installer Ansible sans gérer les problèmes de dépendances. Placez-vous dans le répertoire de travail où se situe le `Vagrantfile`. Vérifiez qu'il n'y a bien pas de `Pipfile`, puis le générer automatiquement avec la commande :

```
pipenv install ansible==11.0.0
```

« ==11.0.0 » indique une contrainte sur la version. Ici, pour indiquer que l'on souhaite spécifiquement la version 11.0.0. Il est également possible d'indiquer une version minimum, une version maximum, voir les deux !

Après l'installation, vous devriez voir un fichier `Pipfile` apparaître dans votre répertoire courant. En effet, lorsque que l'on utilise la commande `pipenv install` avec un argument (le nom du package à installer), celui-ci l'ajoute automatiquement au fichier `Pipfile` du répertoire et s'il n'existe pas, le crée pour vous !

Il ne vous reste plus qu'à utiliser la commande `pipenv shell` pour rentrer dans votre environnement virtuel où Ansible est installé ! N'oubliez pas que ces manipulations seront à refaire à chaque TP !

3. Premiers pas avec Ansible

Je vous conseille d'ouvrir votre répertoire de travail directement dans VSCode pour avoir l'arborescence complète à disposition sur une seule interface : File > Open Folder, et aller chercher votre répertoire Starter Kit !

3.1. Ajout des hôtes dans l'inventaire

Ouvrez le fichier `inventory/hosts.ini`

Celui-ci contient déjà un groupe avec l'hôte `lb-1` déclaré. Des paramètres spécifiques à l'hôte ont été configurés : son IP, l'utilisateur avec lequel Ansible devra se connecter en ssh, ainsi que la clef SSH à utiliser lors de la phase d'authentification sans mot de passe.

Vagrant génère une clef SSH différente par VM, il est donc nécessaire d'indiquer à Ansible quelle clef utiliser. Les clefs SSH sont stockées dans les dossiers de chaque VM :

`.vagrant/machines/<nom VM>/virtualbox/private_key`

En utilisant l'hôte déjà existant dans l'inventaire comme modèle et la documentation, ajoutez deux groupes `dns` et `web` contenant respectivement `dns-1` et `web-1`, avec les bons paramètres d'IP, d'utilisateur et de clé SSH.

3.2. Validation de la connexion aux hôtes

À l'aide de la commande suivante, testez qu'Ansible est en mesure de se connecter correctement à tous les hôtes de l'inventaire :

```
ansible -i inventory/hosts.ini -m ansible.builtin.ping all
```

Cette commande exécute le module Ansible `ansible.builtin.ping` sur tous les hôtes contenus dans le groupe `all` (c'est à dire tous les hôtes) de l'inventaire indiqué via l'option `-i`.

NB : dans le cas précis de notre TP, l'inventaire `inventory/hosts.ini` a été spécifié comme inventaire par défaut dans le fichier `ansible.cfg` présent dans le dossier du projet. Ainsi, sans précision, ce sera bien cet inventaire qui sera utilisé par Ansible.

Relancez la commande précédente sans l'option préciser l'inventaire : retrouvez-vous bien le même résultat ?

3.3. Variables globales

Dans l'inventaire du « starter kit », un fichier de variables pour le groupe `all` existe déjà au format YAML : Indiquez le chemin de ce fichier !

Si on souhaite créer un fichier de variables qui s'appliquerait à tous les hôtes du groupe `dns`, quelle serait son nom et son chemin ?

Même question pour un fichier de variables qui s'appliquerait uniquement à l'hôte dns-1 ?

Ce fichier contient deux variables :
- `ansible_become` positionné à yes
- `ansible_ssh_extra_args` sur lequel nous avons ajouté des options qui serviront lors de l'établissement des connexions SSH. Ces options désactivent la vérification de l'authentification des hôtes car les clefs d'authentifications des VMs changeront à chaque fois que Vagrant les recréera. Cela vous évitera de devoir nettoyer les fichier `~/.ssh/known_hosts` à chaque fois. Ces options ne sont surtout pas à utiliser en production car elles seraient une faille de sécurité !

À quoi sert l'option `ansible_become` ? Pourquoi avons-nous du l'activer ?

4. Tâches Ansible

Documentation utile :

- https://docs.ansible.com/ansible/latest/collections/ansible/builtin/apt_module.html
- https://docs.ansible.com/ansible/latest/collections/ansible/builtin/shell_module.html
- https://docs.ansible.com/ansible/latest/collections/ansible/builtin/service_module.html
- https://docs.ansible.com/ansible/latest/inventory_guide/intro_patterns.html

Dans cette partie, nous allons lancer des tâches (aussi appelées modules) unitaires avec la commande `ansible`. Cela va déjà nous donner un aperçu de la puissance d'Ansible ! D'ailleurs, vous l'avez déjà fait dans la partie 3.2, en exécutant une tâche ping (`ansible.builtin.ping`)

En utilisant la commande `ansible --help` indiquez à quoi correspondent les différents paramètres de la commande tapée précédemment :

```
ansible -i inventory/hosts.ini -m ansible.builtin.ping all
```

A quoi correspondent ?
- `ansible`
- `-i inventory/hosts.ini`
- `-m ansible.builtin.ping`
- `all`

4.1. Mettre à jour et installer des packages

Testez la commande :

```
ansible -m ansible.builtin.apt -a "update_cache=yes upgrade=yes" lb-1
```

- D'après la documentation du module `ansible.builtin.apt` citée au début de la partie 4, qu'a réalisé cette commande ?
- Quel a été le résultat de cette tâche sur `lb-1` ?
- Pourquoi n'avons-nous pas eu besoin de spécifier l'inventaire à utiliser ?

Maintenant, modifiez la commande pour l'exécuter sur tous les hôtes de votre inventaire et relancez la.
- Quel a été le résultat de cette tâche sur `lb-1` ? Sur les autres ?
- Pourquoi cette différence ?

Maintenant, à vous de jouer ! D'après la documentation et les exemples précédents, installez le paquet `apache2` sur tous les hôtes.

Que se passe-t-il lorsque vous lancer plusieurs fois cette commande ? `apache2` est-il installé plusieurs fois ?

4.2. Utiliser les patterns d'hôtes et de groupes

Au temps pour moi, nous n'aurons besoin du paquet `apache2` que sur les hôtes du groupe `web`. Après avoir lu la documentation d'introduction sur les patterns d'hôtes de groupes, utilisez une commande Ansible pour désinstaller `apache2` sur tous les hôtes sauf ceux du groupe `web`

NB : je *spoile* un peu mais en `bash` le caractère « `!` » a une signification particulière, donc il faut mettre votre pattern entre simple quotes : « `'` »

5. Débuter avec les playbooks

Documentation utile :

- https://docs.ansible.com/ansible/latest/user_guide/index.html#writing-tasks-plays-and-playbooks
- https://docs.ansible.com/ansible/latest/user_guide/playbooks_intro.html#playbook-syntax

Dans la partie 4, nous avons vu comment utiliser individuellement des tâches Ansible pour réaliser des actions sur les hôtes gérés à l'aide de cet outil. Nous allons maintenant voir comment réaliser des scénarios plus élaborés utilisant plusieurs tâches à l'aide des playbook.

Le but de cette partie sera de réaliser les différentes tâches de la partie 4 en une seule invocation d'Ansible. Ainsi, avez de commencer cette partie, réinitialisez vos VMs à l'aide de Vagrant.

5.1. Premier play du premier playbook

Dans le dossier de travail, créez un fichier `playbook_tp2.yaml` et commencez par ajouter trois tirets sur la première ligne.

À l'aide des documentations données au débuts des parties 4 et 5 et des exemples fournis dedans, ajouter un premier *play* qui s'exécutera sur le groupe `all` et réalisera la mise à jour des dépôts (`update`, pas `upgrade` !)

Utilisez ensuite la commande `ansible-playbook` pour exécuter ce playbook :

```
ansible-playbook playbook_tp2.yaml
```

Que se passe-t-il lorsque vous lancez plusieurs fois de suite votre playbook ?

5.2. Plusieurs plays dans un playbook

Ensuite, ajoutez un 2e play à votre playbook qui s'exécute sur les hôtes du groupe `web` et qui installe `apache2` dessus. Exécutez-le et vérifiez qu'`apache2` est bien installé sur `web-1`.