TP 3 - Rédaction du premier playbook

Objectif

Cet exercice couvre les principes de base d'Ansible suivants:

- Comprendre les paramètres du module Ansible
- Comprendre et utiliser les modules suivants
  - module apt
  - module de service
  - module de copie

- Comprendre Idempotence et comment les modules Ansible peuvent être idempotents

Guide

Bien que les commandes Ad-hoc Ansible soient utiles pour des opérations simples, elles ne conviennent pas aux scénarios de gestion de configuration ou d'orchestration complexes. Pour de tels cas d'utilisation, les ***playbooks*** sont la solution.

Les playbooks sont des fichiers qui décrivent les configurations ou étapes souhaitées à implémenter sur les hôtes gérés. Les playbooks peuvent transformer des tâches administratives longues et complexes en routines facilement reproductibles avec des résultats prévisibles et réussis.

Un playbook est l'endroit où vous pouvez prendre certaines de ces commandes Ad-hoc que vous venez d'exécuter et les mettre dans un ensemble répétable de *plays* et *tâches*.

Un playbook peut avoir plusieurs "plays" et un "play" peut avoir une ou plusieurs tâches. Dans une tâche, un *module* est appelé, comme les modules du Lab précédent. Le but d'un *play* est de cartographier un groupe d'hôtes. Le but d'une *tâche* est d'implémenter des modules sur ces hôtes.

Création d’une structure pour le playbook

Dans ce TP, vous créerez un playbook pour configurer un serveur Web Apache en trois étapes:

1. Installer le package apache2
2. Activer/démarrer le service apache2
3. Copier un fichier web.html sur chaque hôte Web

Ce Playbook s’assure que le paquet contenant le serveur Web Apache est installé sur **`node1`**.

- Sur votre hôte de contrôle **ansible**, créez un répertoire appelé `ansible-files dans` votre répertoire personnel, et rentrez dedans.

```
mkdir ansible-files
cd ansible-files/
```

- Ajoutez un fichier nommé ***`apache.yml`*** avec le contenu suivant :

```
---
- name: Apache server installed
  hosts: node1
  become: yes
```

Cela montre l’une des forces d’Ansible: la syntaxe Playbook est facile à lire et à comprendre.

Dans ce Playbook:

- Un nom est donné pour le *play* via ***`name`***
- L’hôte sur lequel sera exécuter le playbook est défini via ***`hosts`***
- On active l’escalade de privilèges utilisateur avec ***`become`***

Astuce

Vous devez évidemment utiliser une élévation de privilèges pour installer un package ou exécuter toute autre tâche nécessitant des autorisations root. Cela se fait dans le Playbook par “become: yes”.

- Maintenant que vous avez défini le play, ajoutez une tâche pour faire quelque chose. Dans cette tâche apt s’assurera que le package Apache est installé dans la dernière version. Modifiez le fichier pour qu’il ressemble à la liste suivante:

```
---
- name: Apache server installed
  hosts: node1
  become: yes
  tasks:
  - name: latest Apache version installed
    apt:
      name: apache2
      state: latest
```

Dans les lignes ajoutées:

- La partie tâches commence avec le mot clé ***tasks***

- Une tâche est nommée et le module de la tâche est référencé. Ici, il utilise le module ***apt***.

- Des paramètres pour le module sont ajoutés:

  - ***name*** : pour identifier le nom du paquet
  - ***state*** : pour définir l'état souhaité du paquet

Astuce

Les paramètres du module sont individuels pour chaque module. En cas de doute, recherchez-les à nouveau avec ***ansible-doc****.*

- Enregistrez votre playbook et quittez votre éditeur.

Exécution du playbook

Les Playbooks Ansible sont exécutés à l'aide de la commande ***ansible-playbook*** sur le nœud de contrôle.

- Avant d'exécuter un nouveau Playbook, il est judicieux de vérifier les erreurs de syntaxe

```
ansible-playbook --syntax-check apache.yml
```

- Vous devriez maintenant être prêt à exécuter votre playbook

```
ansible-playbook apache.yml
```

La sortie ne doit pas signaler d'erreurs mais fournir un aperçu des tâches exécutées et un récapitulatif du *play* résumant ce qui a été fait. Il y a aussi une tâche appelée "*Gathering Facts*" qui y est répertoriée: il s'agit d'une tâche intégrée qui s'exécute automatiquement au début de chaque jeu. Il collecte des informations sur les nœuds gérés.

- Connectez-vous à node1 via SSH pour vous assurer qu'Apache a été installé. Utilisez la commande suivante pour vérifier que apache2 est bien installé :

```
sudo dpkg -l apache2
```

- Déconnectez-vous de node1 avec la commande exit pour revenir sur l'hôte de contrôle et vérifiez via une commande Ad-hoc Ansible que le package apache2 est installé.

```
ansible node1 -m command -a 'dpkg -l apache2'
```

- Exécutez le Playbook une deuxième fois et comparez la sortie: la sortie est passée de «changed» à «ok» et la couleur est passée du jaune au vert. De plus, le "PLAYRECAP" est différent maintenant. Cela permet de repérer facilement ce qu'Ansible a réellement fait.

Ajout de tâche : Démarrage et activation d'Apache

La partie suivante du Playbook Ansible s'assure que le service Apache est activée et démarré sur `node1`.

- Modifiez le playbook ***~/ansible-files/apache.yml*** qui devrait maintenant ressembler à ceci:

```
---
- name: Apache server installed
  hosts: node1
  become: yes
  tasks:
  - name: latest Apache version installed
    apt:
      name: apache2
      state: latest
  - name: Apache enabled and running
    service:
      name: apache2
      enabled: true
      state: started
```

Encore une fois: ce que font ces lignes est facile à comprendre:

- une deuxième tâche est créée et nommée
- un module est spécifié (`service`)
- les paramètres du module sont fournis

- Exécutez votre Playbook étendu:

```
ansible-playbook apache.yml
```

Notez maintenant la sortie: Certaines tâches sont affichées comme “ok” en vert et une est indiquée comme “changed” en jaune.

- Utilisez à nouveau une commande Ad-hoc Ansible pour vous assurer qu’Apache a été activé et démarré, par exemple avec la commande: *systemctl status apache2*.

- Exécutez le Playbook une deuxième fois pour vous habituer au changement de sortie.

Ajout de tâche : Création de fichier html

- Vérifiez que les tâches ont été exécutées correctement et qu’Apache accepte les connexions: effectuez une requête HTTP à l’aide du module *uri* d’Ansible dans une commande Ad-hoc du nœud de contrôle Ansible.

```
ansible localhost -m uri -a "url=http://192.168.201.11"
```

La commande devrait renvoyer une ligne verte “status: 200” entre autres des informations sur la page web par défaut d’apache2.

Nous allons utiliser Ansible pour déployer un simple fichier `web.html`

- Sur l’hôte de contrôle ansible, créez le répertoire ***files***, dans *~/ansible-files/*, pour contenir les ressources de fichiers dans *~/ansible-files/*

- Créez ensuite le fichier *~/ansible-files/files/web.html* sur le nœud de contrôle Ansible et y ajoutez le contenu suivant :

```
<body>
<h1>Apache is running fine</h1>
</body>
```

- Sur le nœud de contrôle, modifiez le fichier *~/ansible-files/apache.yml* et ajoutez une nouvelle tâche en utilisant le module *copy*.

```
---
- name: Apache server installed
  hosts: node1
  become: yes
  tasks:
  - name: latest Apache version installed
    apt:
      name: apache2
      state: latest
  - name: Apache enabled and running
    service:
      name: apache2
      enabled: true
      state: started
  - name: copy web.html
    copy:
      src: web.html
      dest: /var/www/html/index.html
```

- Exécutez votre Playbook étendu:

```
ansible-playbook apache.yml
```

Regardez bien la sortie

- Exécutez à nouveau la commande Ad-hoc à l'aide du module "*uri*" ci-dessus pour tester Apache: La commande devrait maintenant renvoyer une ligne verte "*status: 200*" entre autres informations.

  Testez la page web en utilisant le navigateur :

```
http://192.168.201.11/index.html
```

Application à plusieurs hôtes

Le vrai pouvoir d'Ansible est d'appliquer le même ensemble de tâches de manière fiable à de nombreux hôtes.

- Modifiez le Playbook pour pointer vers le groupe "***web***", puis relancez le playbook.

```
---
- name: Apache server installed
  hosts: web
  become: yes
  tasks:
  - name: latest Apache version installed
...
```

- Après l’exécution du playbook, vérifiez si Apache fonctionne maintenant sur les deux serveurs. Utilisez la commande Ad-hoc avec le module *uri* comme vous avez déjà fait avec le `node1` ci-dessus. Toute sortie doit être verte.