Travaux pratiques - Utiliser Ansible pour automatiser l'installation d'un serveur Web

Objectifs

Partie 1 : Lancer la DEVASC VM

Partie 2 : Configurer Ansible

Partie 3 : Vérifier les communications avec le serveur Web local

Partie 4 : Créer des Playbooks Ansible pour automatiser l'installation du serveur Web

Partie 5 : Ajouter des options à votre Playbook Ansible pour serveurs Web Apache

Contexte/scénario

Dans ce TP, vous allez d'abord configurer Ansible afin qu'il puisse communiquer avec une application de serveur web. Vous allez ensuite créer un playbook qui automatisera le processus d'installation d'Apache sur le serveur web. Vous allez également créer un playbook personnalisé qui installe Apache avec des instructions spécifiques.

Ressources requises

- 1 PC avec système d'exploitation de votre choix
- Boîte virtuelle ou VMWare
- Machine virtuelle DEVASC

Instructions

Partie 1 : Lancer la machine virtuelle DEVASC

Si vous n'avez pas encore terminé le **TP - Installez le DEVASC-LAB**, faites-le maintenant. Si vous avez déjà terminé ce TP, lancez le DEVASC VM maintenant.

Partie 2 : Configurer Ansible

La machine virtuelle DEVASC est préinstallée avec un certain nombre d'adresses IPv4 factices que vous pouvez utiliser pour différents scénarios et simulations. Dans cette partie, vous allez configurer Ansible pour utiliser l'une des adresses IPv4 factices pour un serveur Web local.

Étape 1: Ouvrez un terminal dans le DEVASC-LABVM.

Étape 2: Activez le serveur SSH.

Le serveur SSH est désactivé dans le DEVASC-LABVM, ainsi que d'autres services qui ne sont généralement pas requis. Démarrez-le avec la commande suivante.

```
devasc@labvm:~$ sudo systemctl start ssh
devasc@labvm:~$
```

Remarque : le serveur SSH et l'utilitaire **sshpass** ont déjà été installés sur votre machine virtuelle. Pour référence, ceux-ci sont installés à l'aide des commandes suivantes :

Installer SSH

```
devasc@labvm:~$ sudo apt-get install openssh-server
```

Installer sshpass

```
devasc@labvm:~$ sudo apt-get install sshpass
```

Étape 3: Ouvrez le répertoire ansible dans VS Code.

a. Ouvrez **VS Code**.

b. Cliquez sur **File > Open Folder...** et naviguez jusqu'au dossier **/labs/devnet-src/ansible**

c. Cliquez sur **OK**.

d. Les deux sous-répertoires pour les laboratoires accessibles sont maintenant chargés dans le volet VS Code **EXPLORER** pour votre commodité. Dans ce TP, vous travaillerez avec le répertoire **ansible-apache**.

Étape 4: Modifier le fichier de stock Ansible

a. Ouvrez le fichier **hosts** dans le répertoire **ansible-apache**.

b. Ajoutez les lignes suivantes au fichier **hosts** et enregistrez.

```
[webservers]
192.0.2.3 ansible_ssh_user=devasc ansible_ssh_pass=Cisco123!
```

c. Les informations d'identification **devasc** et **Cisco123!** sont des informations d'identification d'administrateur pour la machine virtuelle DEVASC. L'adresse IPv4 que vous utiliserez pour ce TP est 192.0.2.3. Il s'agit d'une adresse IPv4 statique sur la machine virtuelle sous l'interface dummy0, comme indiqué dans la sortie de la commande **ip addr**.

```
devasc@labvm:~/labs/devnet-src/ansible$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen
1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group
default qlen 1000
    link/ether 08:00:27:97:ae:11 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s3
       valid_lft 45882sec preferred_lft 45882sec
    inet6 fe80::a00:27ff:fe97:ae11/64 scope link
       valid_lft forever preferred_lft forever
3: dummy0: <BROADCAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/ether a6:44:a7:e8:6a:9e brd ff:ff:ff:ff:ff:ff
    inet 192.0.2.1/32 scope global dummy0
       valid_lft forever preferred_lft forever
    inet 192.0.2.2/32 scope global dummy0
       valid_lft forever preferred_lft forever
    inet 192.0.2.3/32 scope global dummy0
       valid_lft forever preferred_lft forever
    inet 192.0.2.4/32 scope global dummy0
```

```
       valid_lft forever preferred_lft forever
    inet 192.0.2.5/32 scope global dummy0
       valid_lft forever preferred_lft forever
    inet6 fe80::a444:a7ff:fee8:6a9e/64 scope link
       valid_lft forever preferred_lft forever
devasc@labvm:~/labs/devnet-src/ansible$
```

Étape 5: Modifiez le fichier ansible.cfg.

a. Dans le sous-répertoire **ansible-apache**, ouvrez **ansible.cfg**.

b. Vous pouvez supprimer le commentaire. Ajoutez les lignes suivantes au fichier et enregistrez-le. Le fichier **ansible.cfg** indique à Ansible où trouver le fichier d'inventaire et définit certains paramètres par défaut.

```
[defaults]
# Use local hosts file in this folder
inventory=./hosts
# Don't worry about RSA Fingerprints
host_key_checking = False
# Do not create retry files
retry_files_enabled = False
```

Partie 3 : Vérifier les communications avec le serveur Web local

Dans cette partie, vous vérifierez qu'Ansible peut envoyer des commandes au serveur Web local.

Étape 1: Utilisez le module ping pour vérifier qu'Ansible peut effectuer un ping sur le serveur Web.

Utilisez le module Ansible **ping** pour vérifier les communications avec les périphériques répertoriés dans le groupe de **serveurs Web** de votre fichier d'inventaire d' **hôtes**.

```
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$ ansible webservers -m
ping
192.0.2.3 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$
```

Si plusieurs périphériques étaient répertoriés sous le groupe de **serveurs Web** dans votre fichier d'inventaire des **hôtes**, la sortie indiquerait des informations similaires pour chaque périphérique.

Étape 2: Utilisez le module de commande pour vérifier qu'Ansible peut communiquer avec le serveur Web.

Utilisez le module de **commande** Ansible pour vérifier les communications avec les périphériques répertoriés dans le groupe de **serveurs Web** de votre fichier d'inventaire d' **hôtes**. Dans cet exemple, vous envoyez l'argument **-a "/bin/echo hello world"** pour demander au serveur web local de répondre avec "hello world".

```
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$ ansible webservers -m
command -a "/bin/echo hello world"
192.0.2.3 | CHANGED | rc=0 >>
```

```
hello world
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$
```

Partie 4 : Créer des Playbooks Ansible pour automatiser l'installation du serveur Web

Dans cette partie, vous allez créer deux playbooks Ansible. Le premier playbook va automatiser le test d'écho que vous avez fait dans la partie précédente. Imaginez que vous apportez une centaine de serveurs Web en ligne. Le groupe [serveur web] dans le fichier **hosts** répertorie toutes les informations nécessaires pour chaque serveur web. Vous pouvez ensuite utiliser un simple playbook pour vérifier les communications avec chacun d'eux à l'aide d'une seule commande. Dans le second playbook, vous allez créer et automatiser l'installation du logiciel serveur web Apache.

Étape 1: Créez votre playbook Ansible pour tester votre groupe de serveurs Web.

Dans cette étape, vous allez créer un playbook Ansible pour exécuter la même commande **echo**.

a. Dans VS Code, créez un nouveau fichier dans le répertoire **ansible-apache** avec le nom suivant :

test_apache_playbook.yaml

b. Ajoutez les informations suivantes au dossier. Assurez-vous que vous utilisez l'indentation YAML appropriée. Chaque espace et chaque tiret sont significatifs. Vous risquez de perdre une partie du formatage si vous faites un copier-coller.

```
---
- hosts: webservers
  tasks:
    - name: run echo command
      command: /bin/echo hello world
```

Étape 2: Exécutez le playbook Ansible pour tester votre groupe de serveurs Web.

Exécutez le playbook Ansible à l'aide de la commande **ansible-playbook** à l'aide de l'option **-v** verbeuse. Vous devriez voir un résultat similaire à ce qui suit.

```
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$ ansible-playbook -v
test_apache_playbook.yaml
Using /home/devasc/labs/ansible/ansible-apache/ansible.cfg as config file

PLAY [webservers] ************************************************************

TASK [Gathering Facts] *******************************************************
ok: [192.0.2.3]

TASK [run echo command] ******************************************************
changed: [192.0.2.3] => {"changed": true, "cmd": ["/bin/echo", "hello", "world"],
"delta": "0:00:00.002062", "end": "2020-05-20 21:35:32.346595", "rc": 0, "start":
"2020-05-20 21:35:32.344533", "stderr": "", "stderr_lines": [], "stdout": "hello
world", "stdout_lines": ["hello world"]}

PLAY RECAP *******************************************************************
192.0.2.3 : ok=2 changed=1 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0

devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$
```

Étape 3: Créez votre playbook Ansible pour installer Apache.

a. Dans VS Code, créez un nouveau fichier dans le répertoire **ansible-apache** avec le nom suivant :

install_apache_playbook.yaml

b. Ajoutez les informations suivantes au dossier. Assurez-vous que vous utilisez l'indentation YAML appropriée. Chaque espace et chaque tiret sont significatifs. Vous risquez de perdre une partie du formatage si vous faites un copier-coller. Le texte surligné est expliqué à l'étape suivante.

```
---
- hosts: webservers
  become: yes
  tasks:
    - name: INSTALL APACHE2
      apt: name=apache2 update_cache=yes state=latest

    - name: ENABLED MOD_REWRITE
      apache2_module: name=rewrite state=present
      notify:
        - RESTART APACHE2

  handlers:
    - name: RESTART APACHE2
      service: name=apache2 state=restarted
```

Étape 4: Examinez votre playbook Ansible.

Voici une explication de certaines des lignes significatives de votre playbook:

- **`hosts: webservers`** - Ceci fait référence au groupe de périphériques de **serveurs Web** dans votre fichier d'inventaire des **hôtes**. Ce playbook sera exécuté pour tous les appareils avec ce groupe.
- **`become: yes`** - Le mot-clé **become** active l'exécution de la commande **sudo**, ce qui permet de réaliser des tâches telles que l'installation d'applications.
- **`apt:`** - Le module **apt** est utilisé pour gérer les paquets et les installations d'applications sous Linux.
- **`handlers:`** - Les gestionnaires sont similaires à une tâche mais ne sont pas exécutés automatiquement. Ils sont appelés par une tâche. Notez que la tâche **ENALED MOD_REWRITE** appelle le gestionnaire **RESTART APACHE2**.

Étape 5: Exécutez la sauvegarde Ansible pour installer Apache.

Exécutez le playbook Ansible à l'aide de la commande **ansible-playbook** à l'aide de l'option **-v** verbeuse. La première fois qu'Apache est installé sur votre machine virtuelle, la tâche **INSTALL APACHE2** prendra de 30 secondes à quelques minutes en fonction de votre vitesse Internet.

```
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$ ansible-playbook -v
install_apache_playbook.yaml
Using /home/devasc/labs/ansible/ansible-apache/ansible.cfg as config file

PLAY [webservers] ************************************************************

TASK [Gathering Facts] *******************************************************
ok: [192.0.2.3]
```

```
TASK [INSTALL APACHE2] ***********************************************************
ok: [192.0.2.3] => {"cache_update_time": 1590010855, "cache_updated": true, "changed":
false}

TASK [ENABLED MOD_REWRITE] *******************************************************
ok: [192.0.2.3] => {"changed": false, "result": "Module rewrite enabled"}

PLAY RECAP ***********************************************************************
192.0.2.3 : ok=3 changed=0 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
=
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$
```

Le PLAY RECAP doit afficher **ok** et **failed=0** indiquant une exécution réussie du playbook.

Étape 6: Vérifiez que Apache a été installé.

a. Utilisez la commande suivante pour vérifier qu'Apache est maintenant installé. Appuyez sur "q" pour quitter.

```
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$ sudo systemctl status
apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor prese>
     Active: active (running) since Wed 2020-05-20 03:48:49 UTC; 10min ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 8201 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SU>
   Main PID: 8225 (apache2)
      Tasks: 55 (limit: 4654)
     Memory: 5.3M
     CGroup: /system.slice/apache2.service
             ├─8225 /usr/sbin/apache2 -k start
             ├─8229 /usr/sbin/apache2 -k start
             └─8230 /usr/sbin/apache2 -k start
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$
```

b. Ouvrez le navigateur Web Chromium et entrez l'adresse IPv4 de votre nouveau serveur, **192.0.2.3**, pour afficher la page Web Apache2 par défaut.

Partie 5 : Ajouter des options à votre Playbook Ansible pour serveurs Web Apache

Dans un environnement de production, l'installation par défaut d'Apache2 est généralement personnalisée pour les fonctionnalités spécifiques requises par l'organisation. Un playbook Ansible peut également aider à automatiser ces tâches de configuration. Dans cette partie, vous allez personnaliser votre playbook en spécifiant que le serveur Apache utilise un numéro de port différent.

Étape 1: Créez votre playbook Ansible pour installer Apache.

a. Dans VS Code, créez un nouveau fichier dans le répertoire **ansible-apache** avec le nom suivant :

install_apache_options_playbook.yaml

b. Ajoutez les informations suivantes au dossier. Assurez-vous que vous utilisez l'indentation YAML appropriée. Chaque espace et chaque tiret sont significatifs. Vous risquez de perdre une partie du formatage si vous faites un copier-coller.

```
---
- hosts: webservers
  become: yes
  tasks:
   - name: INSTALL APACHE2
     apt: name=apache2 update_cache=yes state=latest

   - name: ENABLED MOD_REWRITE
     apache2_module: name=rewrite state=present
     notify:
       - RESTART APACHE2

   - name: APACHE2 LISTEN ON PORT 8081
     lineinfile: dest=/etc/apache2/ports.conf regexp="^Listen 80"
line="Listen 8081" state=present
     notify:
       - RESTART APACHE2

   - name: APACHE2 VIRTUALHOST ON PORT 8081
     lineinfile: dest=/etc/apache2/sites-available/000-default.conf
regexp="^<VirtualHost \*:80>" line="<VirtualHost *:8081>" state=present
     notify:
       - RESTART APACHE2

  handlers:
   - name: RESTART APACHE2
     service: name=apache2 state=restarted
```

Cette liste de lecture est très similaire à la précédente avec l'ajout de deux tâches que les serveurs Web écoutent sur le port 8081 au lieu du port 80.

Le module **lineinfile** est utilisé pour remplacer les lignes existantes dans les fichiers /etc/apache2/ports.conf et /etc/apache2/sites-available/000-default.conf. Vous pouvez rechercher dans la documentation Ansible pour plus d'informations sur le module **lineinfile**.

Étape 2: Examinez les deux fichiers qui seront modifiés par le playbook.

Affichez les fichiers **/etc/apache2/ports.conf** et **/etc/apache2/sites-available/000-default.conf**. Notez que le serveur Web est actuellement à l'écoute sur le port 80.

```
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$ cat
/etc/apache2/ports.conf
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 80

<IfModule ssl_module>
        Listen 443
<output omitted>
```

```
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$ cat
/etc/apache2/sites-available/000-default.conf
<VirtualHost *:80>
        # The ServerName directive sets the request scheme, hostname and port that
        # the server uses to identify itself. Ceci est utilisé lors de la création
        # redirection URLs. Dans le contexte des hôtes virtuels, le nom du serveur
<output omitted>
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$
```

Étape 3: Exécutez le Playbook Ansible.

a. Exécutez le playbook Ansible à l'aide de la commande **ansible-playbook**.

```
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$ ansible-playbook
install_apache_options_playbook.yaml

PLAY [webservers] ************************************************************

TASK [Gathering Facts] *******************************************************
ok: [192.0.2.3]

TASK [INSTALL APACHE2] *******************************************************
ok: [192.0.2.3]

TASK [ENABLED MOD_REWRITE] ***************************************************
ok: [192.0.2.3]

TASK [APACHE2 LISTEN ON PORT 8081] *******************************************
ok: [192.0.2.3]

TASK [APACHE2 VIRTUALHOST ON PORT 8081] **************************************
ok: [192.0.2.3]

PLAY RECAP *******************************************************************
192.0.2.3 : ok=5 changed=0 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0

devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$
```

Étape 4: Vérifiez qu'Apache a été installé.

a. Affichez à nouveau les fichiers **/etc/apache2/ports.conf** et **/etc/apache2/sites-available/000-default.conf**. Notez que le playbook a modifié ces fichiers pour les écouter sur le port 8081.

```
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$ cat
/etc/apache2/ports.conf
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 8081

<IfModule ssl_module>
```

```
        Listen 443
</IfModule>

<IfModule mod_gnutls.c>
        Listen 443
</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$

devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$ cat
/etc/apache2/sites-available/000-default.conf
<VirtualHost *:8081>
        # The ServerName directive sets the request scheme, hostname and port that
        # the server uses to identify itself. Ceci est utilisé lors de la création
        # redirection URLs. Dans le contexte des hôtes virtuels, le nom du serveur
<output omitted>
devasc@labvm:~/labs/devnet-src/ansible/ansible-apache$
```

b. Ouvrez le navigateur Web Chromium et entrez l'adresse IPv4 de votre nouveau serveur. Mais cette fois, spécifiez 8081 comme numéro de port, **192.0.2. 3:8081**, pour voir la page Web Apache2 par défaut.

Remarque: bien que vous puissiez voir dans le fichier **ports.conf** qu'Apache2 écoute également sur le port 443, il s'agit d'un protocole HTTP sécurisé. Vous n'avez pas encore configuré Apache2 pour un accès sécurisé. Ceci, bien sûr, serait ajouté à votre playbook Ansible, mais dépasse le cadre de ce cours.