TP/TD gestion des configurations : Powerfull Ansible !

2 Présentation d'Ansible

2.3 Organisation de la distribution d'Ansible

```
# Utilisation du module ping pour vérifier la compatibilité de l'accès d'Ansible sur les machin
ansible -m ping debian
```

3 Installation et utilisation d'Ansible

3.1 Obtenir de l'aide

L'utilitaire ansible-navigator permet entre autre d'obtenir de l'aide :

```
# obtenir de l'aide sur le module apt
ansible-navigator doc apt -m stdout #
ou ansible-doc apt
```

Les pages suivantes peuvent vous être fort utiles lors du TP :

- https://docs.ansible.com/ansible/latest/collections/
- https://docs.ansible.com/ansible/latest/collections/index_module.html.
- https://docs.ansible.com/ansible/latest/user_guide/

3.2 Installation de la VM

Nous allons utilisez des containers Debian11 et Rocky 8 (qui est un "clone" de RedHat) comme environnement de formation. Ces containers Docker comprennent systemd. Dans cette configuration il est nécessaire de rétrograder les NameSpaces de votre machine virtuelle en version 1. Pour ce faire lancez la commande:

```
/home/ansible/revert2cgroupv1.sh
```

3.3 Installation et paramétrage de l'environnement du TP.

La version fournis avec la VM d'Ansible est déjà une version plus récente que le l'ont peut trouver dans apt. Pour avoir la dernière version du script aller dans /home/ansible et faire un `git pull` . La commande `./create-cont.sh` , elle génère 5 containers Debian, 5 containers Centos et 3 containers de switchs L3 Ariqta (EOS).

Ces containers sont tous accessibles en ssh.

Le script configure /etc/ansible et /etc/hosts pour que vous puissiez accéder aux containers comme si vous étiez dans un environnement de production.

Les containers Linux sont accessibles directement via ssh et sans mot de passe grâce à la clef générée au début du TP:

```
ssh debian-0 # de 0 à 4
ssh rocky-0
```

Les containers ceos Arista n'ont pas besoin à ce stade du TP d'être accessibles et seront traités ensuite.

4 Prise en main d'Ansible

4.1 Vérification et "debug" basique

1. Test de ping des containers fraîchement crée :

```
#Ping des debian : root@debian:/home/ansible#
ansible -m ping debian debian-2 SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
[...] #Ping des

rocky :

root@debian:/home/ansible# ansible -m ping rocky
rocky-3 | SUCCESS => {     "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python" },
    "changed": false,
    "ping": "pong"
}
[...]
```

2. Utilisation de l'utilitaire `ansible-console` pour lancer `ip a` sur toutes les machines :

```
#Accès à tout les containers : root@debian:/home/ansible#
ansible-console
Welcome to the ansible console. Type help or ? to list commands.

#Execution de la commande dans les 13 containers :

root@all (13)[f:5]$ ip a

[WARNING]: ansible-pylibssh not installed, falling back to paramiko [WARNING]: ansible-pylibssh
not installed, falling back to paramiko ebian-1 | CHANGED | rc=0 >>| 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 57: eth0@if58: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
noqueue state UP group default     link/ether 02:42:ac:1d:00:02 brd ff:ff:ff:ff:ff:ff link-
netnsid 0     inet 172.29.0.2/16 brd 172.29.255.255 scope global eth0         valid_lft forever
preferred_lft forever [...]
```

3. Ansible utilise le protocole réseau SSH.

4. Essaie de `ip a -vvv` : `root@debian:/home/ansible# ansible-console -vvv`

En essayant avec un simple `echo 'hello !'` en lançant la commande sans `-vvv` il n'y a que peu d'informations et la sortie utile de la commande. Avec l'option `-vvv` il y a énormément de verboses, d'informations de débogage. De ce que j'ai compris il y a 4 niveaux de verboses : sans option, 1 -v, 2 vv et 3 -vvv.

5. Création d'un groupe container qui regroupe tous les containers et vérification via les commandes `ansible-inventory --list all` et `ansible-navigator` que le groupe est listé :

   Modification du fichier /etc/ansible/hosts ->

```
[Container:children]
linux arista
```

Existe-t-il un équivalent par défaut ? un groupe listant les nœuds sans groupe ?

Ansible dispose d'un groupe prédéfini appelé ungroupe pour les container sans groupe.

4.2 Installation d'Apache via les modules dnf et apt d'Ansible core

1. Création d'un groupe ou je met les deux hote qui vont acceuilir appache :

```
[webservers]   debian_container
ansible_host=    centos_container
```

`ansible_host=` Code .yaml :

apache.yml :

```
root@debian:/home/ansible# ansible-playbook  apache.yml

PLAY [Install Apache] **********************************************************************

TASK [Gathering Facts] *********************************************************************
ok: [debian-0] ok: [rocky-0]

TASK [Install Apache on Debian] ************************************************************
skipping: [rocky-0] changed: [debian-0]

TASK [Install Apache on CentOS] ************************************************************
skipping: [debian-0] changed: [rocky-0]

PLAY  RECAP  *******************************************************************************
debian-0                   : ok=2    changed=1    unreachable=0    failed=0    skipped=1    re
rocky-0                    : ok=2    changed=1    unreachable=0    failed=0    skipped=1    re
```

2. lfo

3. --check = Vérifie si le playbook vas bien se dérouler --diff = Affiche pourquoi le playbook rate - list-hosts = affiche les hosts touché par les modification du playbook --list-tasks = Affiche les "name" donc les tâches effectuer par le playbook. ansible-lint = Affiche les erreur de configuration de fichier

4. Reload du serveur web :

   - name: Reload le serveur web ansible.builtin.service: name: apache2 state: reloaded tags: relance

5. Le tag never signifie qu'il ne faut pas executer la tache Le tag always sigi=nifie qu'il faut toujours executer la tache.

6.

```
 - name: Lister les facts
hosts: all     gather_facts: true
tasks:       - name: Collecter
les facts
ansible.builtin.setup:
```

7. Récupéré tout les facts comprenant les adresse ip des containers : `ansible all -m setup -a`

```
  'filter=ansible_default_ipv4'
```

8. Voir chatgpt

5 Programmation avec Ansible: conditions et boucles

```
--- hosts:
debian
become: true

  tasks:     - name: Installer Apache et PHP
ansible.builtin.apt:         name: "{{ item
}}"         state: present
update_cache: true       loop:         -
apache2         - libapache2-mod-php7.4
when: ansible_distribution == 'Debian'

-    name: Installer Apache et PHP
ansible.builtin.dnf:         name: "{{
item }}"         state: present
loop:         - httpd         - php
when: ansible_distribution ==
'RockyLinux' --- hosts: debian, centos
become: true

  tasks:     - name: Installer Apache
ansible.builtin.apt:         name: apache2
state: present         update_cache: true
when: ansible_distribution == 'Debian'

-    name: Installer PHP
ansible.builtin.apt:         name:
libapache2-mod-php7.4         state:
present       when: ansible_distribution
== 'Debian'

-    name: Installer Apache
ansible.builtin.dnf:         name: httpd
state: present
      when: ansible_distribution == 'RockyLinux'

-    name: Installer PHP
ansible.builtin.dnf:         name: php
state: present
      when: ansible_distribution == 'RockyLinux'

-    name: Démarrer le service Apache
ansible.builtin.service:
```

```
        name: apache2
  state: started
  enabled: true

  -    name: Copier le fichier phpinfo
  ansible.builtin.copy:         src:
  info.php         dest:
  /var/www/html/index.php         owner:
  www-data         group: www-data
  mode: "0664"
```

5.1 Utilisation d'un rôle Ansible

Commande pour installe : ansible-playboook playbook.yml

```
  ---
  - hosts: all
    become: linux

    roles:     -
  geerlingguy.firewall

    vars:
  firewall_allowed_tcp_ports:
  - 22
  - 80
  - 8080
```

6 Utilisation professionnelle d'Ansible

6.1 Création d'un container via Ansible

```
  --- hosts: localhost
  gather_facts: false
  become: true

    tasks:     - name: Créer le container 203-
  debian-mine
  community.docker.docker_container:
          name: 203-debian-mine         image:
  registry.iutbeziers.fr/debianiut         state:
  started         restart_policy: always
  published_ports:
            - "80:80"
```

6.2 Gestion des routeurs/switch Arista avec Python

6.3 Utilisation d’Ansible pour piloter une machine Windows

7 Tips & tricks

7.1 rolling update