Lab 3- Ad-hoc Commands

Introduction

Il existe plusieurs tâches dans Ansible pour lesquelles vous n'avez pas besoin d'écrire un Playbook Ansible; vous pouvez simplement exécuter une commande ad-hoc pour cette tâche. Il s'agit d'une commande à une seule ligne pour effectuer une seule tâche sur l'hôte cible. Ces commandes sont présentes dans /usr/bin/ansible

Ad-hoc setup

Créons un espace de travail pour nos commandes ad-hoc.

Veuillez noter que certaines des choses que nous allons créer seront abordées plus tard dans ce cours.

Créons maintenant une configuration et un fichier d'inventaire pour ansible

Copiez et collez ce qui suit pour créer un fichier *Ansible Config*. Vous pouvez également le créer vous-même dans un éditeur de fichiers tel que vim sous le répertoire /etc/ansible

```
sudo cp /etc/ansible/ansible.cfg /etc/ansible/ansible.cfg.bk
sudo vi /etc/ansible/ansible.cfg
```

```
[defaults]
inventory = hosts
host_key_checking = False
deprecation_warnings=False
```

Copiez et collez ce qui suit pour créer un fichier d'inventaire.

```
vi /etc/ansible/hosts
```

```
[demo_servers]
centos01 ansible_host= 192.168.208.164
ubuntu01 ansible_host= 192.168.208.165
```

```
[ubuntu]
ubuntu01 ansible_host=192.168.208.165

[centos]
centos01 ansible_host= 192.168.208.164
```

Commandes Ad-hoc

Commandes de base

La commande ad-hoc ci-dessous exécute un module **ping** sur toutes les hôtes du fichier inventaire. Ici **-m** est l'option pour un module.

```
ansible all -m ping
```

Example of output:

```
centos01 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}

ubuntu01 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python"
  },
  "changed": false,
  "ping": "pong"
}
```

La commande mentionnée ci-dessous exécute le module `setup` (Gather facts) sur un groupe d'hôtes, l'option **-a** ou **--args** sert à définir les arguments du module exécuté.

```
ansible all -m setup -a "filter=ansible_distribution*"
```

Exemple d'output:

```
ubuntu01 | SUCCESS => {
```

```
    "ansible_facts": {
        "ansible_distribution": "Ubuntu",
        "ansible_distribution_file_parsed": true,
        "ansible_distribution_file_path": "/etc/os-release",
        "ansible_distribution_file_variety": "Debian",
        "ansible_distribution_major_version": "20",
        "ansible_distribution_release": "focal",
        "ansible_distribution_version": "20.04",
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false
}
centos01 | SUCCESS => {
    "ansible_facts": {
        "ansible_distribution": "CentOS",
        "ansible_distribution_file_parsed": true,
        "ansible_distribution_file_path": "/etc/redhat-release",
        "ansible_distribution_file_variety": "RedHat",
        "ansible_distribution_major_version": "7",
        "ansible_distribution_release": "Core",
        "ansible_distribution_version": "7.8",
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false
}
```


Autres exemples pour la commande de configuration "setup"

Affichez les faits de tous les hôtes et stockez-les indexés par nom d'hôte dans / tmp / facts.

```
ansible all -m setup --tree /tmp/facts
```

Afficher uniquement les faits concernant la mémoire trouvée par ansible sur tous les hôtes et les afficher.

```
ansible all -m setup -a 'filter=ansible_*_mb'
```

Afficher uniquement les faits renvoyés par facter.

```
ansible all -m setup -a 'filter=facter_*'
```

Afficher uniquement des informations sur certaines interfaces.

```
ansible all -m setup -a 'filter=ansible_eth[0-2]'
```

Restreindre les faits collectés supplémentaires au réseau et au virtuel.

```
ansible all -m setup -a 'gather_subset=network,virtual'
```

ne pas traiter puppet facter ou ohai même si présents.

```
ansible all -m setup -a 'gather_subset=!facter,!ohai'
```

Ne collectez que le minimum de faits:

```
ansible all -m setup -a 'gather_subset=!all'
```

Afficher les faits des hôtes Windows avec des faits personnalisés stockés dans C:\\custom_facts.

```
ansible windows -m setup -a "fact_path='c:\\custom_facts'"
```

Authentification par mot de passe

Nous allons maintenant exécuter une commande Ad-Hoc en utilisant l'authentification par mot de passe ssh. (CentOS/RHEL, vous devez installer le paquet supplémentaire appelé 'sshpass' sur les nœuds).

Pour collecter les informations d'un nœud avec un utilisateur différent de l'utilisateur « ansible », il faut lui activer la connexion par mot de passe tout d'abord. (suivez ce guide en cas de besoin : ***https://help.thorntech.com/docs/sftp-gateway-classic/enable-password-login/***)

```
[ansible@server ~]$ ansible ubuntu -m ping -u devops --ask-pass
SSH password:
ubuntu01 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
```

Module shell

La commande ci-dessous vous permet d'exécuter des commandes ad-hoc en tant qu'utilisateur non root avec des privilèges root. L'option **--become** donne les privilèges root et l'option **-K** demande le mot de passe.

```
ansible ubuntu -m shell -a 'fdisk -l' -u devops --become -k -K
```

Exemple d'Output:

```
SSH password:
BECOME password[defaults to SSH password]:
ubuntu01 | CHANGED | rc=0 >>
Disque /dev/loop0 : 55,48 MiB, 58159104 octets, 113592 secteurs
Unités : secteur de 1 × 512 = 512 octets
Taille de secteur (logique / physique) : 512 octets / 512 octets
taille d'E/S (minimale / optimale) : 512 octets / 512 octets
...
```

- **-k** : mot de passe SSH
- **-K** : mot de passe « sudo » pour activer le mode admin avec **--become**

Redémarrer le système

Cette commande ad-hoc est utilisée pour redémarrer le système avec l'option **-f** pour définir le nombre de forks (exécutions concurrentes).

```
ansible all -a "/sbin/reboot" -f 1
```

Exemple d'Output:

```
ubuntu01 | CHANGED | rc=0 >>
Disk /dev/sdb: 10 MiB, 10485760 bytes, 20480 sectors
Units: sectors of 1 \* 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk /dev/sda: 10 GiB, 10737418240 bytes, 20971520 sectors
Units: sectors of 1 \* 512 = 5
```

Transfert de fichiers

La commande ad-hoc ansible ci-dessous est utilisée pour copier un fichier d'une source vers une destination pour un groupe d'hôtes défini dans le fichier d'inventaire. Si la sortie avec le paramètre «changed» est «true», le fichier a été copié vers la destination.

```
ansible all -m copy -a 'src=file1.txt dest=/home/ansible/Desktop/
owner=root mode=0644' –become -K
```

Exécutez la commande ci-dessous pour vérifier si le module copy a fonctionné correctement ou non. Le fichier copié doit arriver à la destination mentionnée dans la commande précédente.

```
ls -l /home/ansible/Desktop
```

le module fetch

La commande ad-hoc ci-dessous est utilisée pour télécharger un fichier à partir d'une hôte définie dans la commande (l'inverse de copy).

```
ansible ubuntu01 -m fetch -a "src=/home/ansible/ubuntu01.txt dest=/tmp/
flat=yes"
```

Exemple d'Output:

```
ubuntu01 | CHANGED > {
"changed": true,
"checksum": "d25c92b5a19de37a9f01205ede54b0b8d60e7cf2",
"dest": "/tmp/ubuntu01.txt",
"md5sum": "e01c22cf59ec1148308e76866c68d125",
"remote_checksum": "d25c92b5a19de37a9f01205ede54b0b8d60e7cf2",
"remote_md5sum": null
}
```

Vérifiez si le fichier a été téléchargé ou non à la destination mentionnée dans la commande.

```
ls /tmp
```