

ANSIBLE ADVANCED



SOMMAIRE

Ansible

- 1.Les facts
- 2.Les variables magiques
- 3.Un peu plus loin dans les variables
- 4.Les rôles
- 5.Sécurité: Ansible vault
- 6.Projet: Déploiement d'une plateforme complète



Ansible - Les Facts

1.1 Définitions

Les facts sont des variables d'un type particulier

Ce sont des variables contenant des informations qui proviennent directement des serveurs distants

On peut utiliser ce type de variable pour par exemple connaître le système d'exploitation de chacun de vos serveurs et faire lancer certaines tâches en fonction



Ansible - Les Facts

1.2 Utilisation

Il a plusieurs façons d'obtenir les facts
via un playbook

```
- name: Print all available facts
```

```
ansible.builtin.debug:
```

```
var: ansible_facts
```



Ansible - Les Facts

1.2 Utilisation

via la ligne de commande

```
ansible <hostname> -m ansible.builtin.setup
```



Ansible - Les Facts

1.2 Utilisation

Le résultat de ces commandes ressemble à ça:

```
{  
  "ansible_all_ipv4_addresses": [  
    "REDACTED IP ADDRESS"  
  ],  
  "ansible_all_ipv6_addresses": [  
    "REDACTED IPV6 ADDRESS"  
  ],  
}
```



Ansible - Les Facts

1.2 Utilisation

Un exemple d'utilisation

```
- name: install apache and php last version for (Debian os family)
  apt:
    name: ['apache2', 'php', 'php-mysql']
    state: present
    update_cache: yes
  when: ansible_facts['os_family'] == "Debian"
```



Ansible - Les Facts

1.2 Utilisation

Un exemple d'utilisation

```
- name: install apache and php last version for (RedHat os family)
yum:
  name: ['httpd', 'php', 'php-mysqlnd']
  state: present
  update_cache: yes
when: ansible_facts['os_family'] == "RedHat"
```



Ansible - Les Facts

1.2 Utilisation

Par défaut les facts sont gardées en mémoire, cependant il est possible de les rendre persistants afin d'y accéder de manière répétée , ça veut dire quoi ?

Depuis un playbook vous pouvez accéder aux facts de tous vos hosts

```
{{ hostvars['asdf.example.com']['ansible_facts']['os_family'] }}
```



Ansible - Les Facts

1.2 Utilisation

Il est également possible de désactiver les facts principalement pour des raisons de performances

```
- hosts: whatever  
  gather_facts: no
```



Ansible - Les Facts

1.2 Utilisation

Bon à savoir par défaut ansible rassemble les facts en début de lancement de playbook



Ansible - Les variables magiques

2.1 Définition

Les variables magiques sont l'ensemble des variables liés aux opérations d'ansible

Les noms de ces variables sont réservés et ne peuvent donc pas être utilisée par d'autres variables

```
{{ hostvars['test.example.com']['ansible_facts']['distribution'] }}
```



Ansible - Les variables magiques

2.2 Utilisation

Comme pour tout le reste

```
{% for host in groups['app_servers'] %}  
    {{ hostvars[host]['ansible_facts']['eth0']['ipv4']['address'] }}  
{% endfor %}
```

ansible_play_hosts, role_path, ansible_check_mode



Ansible - Les variables advanced

3.1 Utilisation

On a déjà vu comment fonctionne les variables et comment les utiliser , on va maintenant voir que les variables peuvent s'utiliser à plusieurs endroits différents

on imagine que l'on a la variable `ntp_server`

- si l'on veut que cette variable soit disponible pour tous nos hosts

```
# file: /etc/ansible/group_vars/all
```

```
# this is the site wide default
```

```
ntp_server: default-time.example.com
```



Ansible - Les variables advanced

3.1 Utilisation

- si l'on veut que cette variable soit disponible pour tous les hosts de type webservers

```
# file: /etc/ansible/group_vars/webrowsers
```

```
ntp_server: boston-time.example.com
```



Ansible - Les variables advanced

3.1 Utilisation

- si l'on veut que cette variable soit disponible pour un seul host

```
# file: /etc/ansible/host_vars/xyz.boston.example.com
```

```
ntp_server: override.example.com
```



Ansible - Les variables advanced

3.2 En pratique

Choisissez un package à installer (wget , telnet, tcpdump, ldapsearch etc ...) et créer un playbook qui installe ce package sur les OS de type “Debian” et “RedHat”

Tester votre playbook sur les 2 types d’OS si c’est possible



Ansible - Les variables advanced

3.2 En pratique / Bonus

Créer votre propre fact(dynamique) et remonté le via un playbook

```
ansible.builtin.set_fact
```



Ansible - Les rôles

4.1 Définition

Les rôles vous permettent de charger automatiquement des variables, des fichiers, des tâches, des gestionnaires et d'autres artefacts Ansible associés en fonction d'une structure de fichiers connue. Après avoir regroupé votre contenu dans des rôles, vous pouvez facilement les réutiliser et les partager avec d'autres utilisateurs.

En résumé les rôles vous permettent de réutiliser votre code dans des playbooks différents



Ansible - Les rôles

4.2 Utilisation

Un rôle ansible à une structure bien définie, pour que le rôle existe il faut au moins qu'il possède un sous-répertoire



Ansible - Les rôles

4.2 Utilisation

```
# playbooks  
  
site.yml  
  
webservers.yml  
  
fooservers.yml  
  
roles/  
  
    common/  
  
        tasks/  
  
        handlers/  
  
        library/  
  
        files/  
  
        templates/  
  
        vars/  
  
        defaults/  
  
        meta/  
  
webservers/  
  
    tasks/
```



Ansible - Les rôles

4.2 Utilisation

Les répertoires

- tasks: l'ensemble des tâches disponible pour un host
- defaults: les variables par défaut pour le rôle
- vars: les autres variables du rôle (celle que l'on ne pourra pas surcharger facilement)
- templates: les templates à déployer pour le rôle



Ansible - Les rôles

4.2 Utilisation

Les tâches

- par défaut ansible prend le fichier main.yml se trouvant dans ce répertoire mais il est possible de le splitter en plusieurs fichiers



Ansible - Les rôles

4.2 Utilisation

```
# roles/example/tasks/main.yml
```

```
- name: Install the correct web server for RHEL
  import_tasks: redhat.yml
  when: ansible_facts['os_family']|lower == 'redhat'
```

```
- name: Install the correct web server for Debian
  import_tasks: debian.yml
  when: ansible_facts['os_family']|lower == 'debian'
```

```
# roles/example/tasks/redhat.yml
```

```
- name: Install web server
  ansible.builtin.yum:
    name: "httpd"
    state: present
```

```
# roles/example/tasks/debian.yml
```

```
- name: Install web server
  ansible.builtin.apt:
    name: "apache2"
    state: present
```



Ansible - Les rôles

4.2 Utilisation

Comment on utilise un rôle dans un playbook

```
---  
- hosts: webservers  
  roles:  
    - common  
    - webservers
```



Ansible - Les rôles

4.2 Pratique

Créer un rôle qui va créer un utilisateur



Ansible - Les rôles

4.2 Pratique/Bonus

Créer un deuxième rôle qui supprime l'utilisateur si il existe si et seulement si le hostname du serveur match avec “front”



Ansible - Ansible vault

5.1 Définition

Un vault est un coffre-fort, c'est un espace de stockage chiffré où l'on conserve les données sensibles comme les mots de passes, les certificats ssl etc

En règle générale, on aime bien versionner nos fichiers (donc nos playbooks) et comme les playbooks contiennent des identifiants il est impératif de les chiffrer, ansible vault est fait pour ça



Ansible - Ansible vault

5.2 Utilisation

via une variable chiffrée dans un playbook

```
ansible-vault encrypt string 'secret-text' --name 'secret'
```



Ansible - Ansible vault

5.2 Utilisation

Résultat :

```
secret: !vault |
  $ANSIBLE_VAULT;1.1;AES256
  33616163316438306663303330376334363862343430363432343536313835323132353939376438
  34393333730353532346630626133666434363932303133650a343137353735363138646536633432
  39303765306633643830353238646433356230623537363466373438323931353763666537386163
  3062303334343830370a3339393466323461343130633631363630383562663038353633331376631
  3664
```



Ansible - Ansible vault

5.2 Utilisation

Ansible demandera ensuite le mdp pour chiffrer , vous pouvez le mettre directement dans votre var

```
---
- hosts: localhost
  vars:
    - secret: !vault |
      $ANSIBLE_VAULT;1.1;AES256
      33616163316438306663303330376334363862343430363432343536313835323132353939376438
      3439333730353532346630626133666434363932303133650a343137353735363138646536633432
      39303765306633643830353238646433356230623537363466373438323931353763666537386163
      3062303334343830370a333939346632346134313063363136363038356266303835363331376631
      3664

  tasks:
    - debug: var=secret
    - fail:
      when: 1 == 1
```

Copier



Ansible - Ansible vault

5.2 Utilisation

Ensuite vous devez exécuter votre playbook avec l'option `--ask-vault-pass`

```
ansible-playbook playbook.yml --ask-vault-pass
```



Ansible - Ansible vault

5.3 Pratique

utilisez ansible vault pour chiffrer une de vos variable et lancer votre playbook



Ansible - Ansible vault

5.3 Pratique/Bonus

utilisez ansible vault pour chiffrer plusieurs de vos variable et trouver un moyen de ne pas demander le mot de passe a chaque lancement du playbook



Ansible - Déploiement d'une plateforme

6

Le but de cet exercice est de reprendre tout ce que l'on a vu jusqu'à présent et de l'utiliser pour déployer une plateforme complète lamp

Vous pouvez utiliser les rôles ou non , les variables ou non , le but n'est pas de faire quelque chose de "beau", mais de faire quelque chose qui fonctionne et qui fonctionne sur 2 types d'OS différents (redhat et debian par exemple)



Ansible - Déploiement d'une plateforme

6

Le cahier des charges:

- Mettre la bonne timezone Europe/Bruxels ou Europe/Paris
- Créer un group gasr
- Créer un utilisateur asr dans le group sudoer , asr
- Modifier le bashrc/profil de l'utilisateur asr pour qu'il exécute la commande "ll" qui est un alias de ls -ailh
- Installer apache



Ansible - Déploiement d'une plateforme

6

Le cahier des charges:

- Installer php et php pour apache (ainsi que php-mysql)
- Installer mariadb
- Créer une base de donnée asr_blog avec le user asr et une table user (id, nom, mail)
- Remplissez la table user avec des datas fictives
- créer un répertoire de log asr_blog dans /var/www/asr_blog
- créer un répertoire blog dans /home/asr/blog



Ansible - Déploiement d'une plateforme

6

Le cahier des charges:

- copier le fichier index.php dans /home/asr/blog
- ajouter l'utilisateur asr dans le groupe www-data
- créer un template de vhost apache pour qu'il pointe sur le /home/asr/blog
- BONUS: tester via une tâche ansible que le site répond bien (status http = 200) si c'est le cas on ne fait rien si ce n'est pas le cas on attend 30 secondes on recheck le status et on arrête quelque soit le résultat

