Formation Ansible – Python

Ihab ABADI / UTOPIOS

SOMMAIRE - Ansible

SOMMAIRE - Ansible

Ansible - Introduction

Ansible est un projet récent (2012) qui a été développé entièrement en python et qui répond à des besoins de remplacement, de déploiement ou de changements de serveur. Ces actions peuvent être mené sur l'ensemble ou une partie des serveurs.

C'est un système agentless capable de piloter des systèmes Windows, Linux, Unix et aussi des équipements réseau tel que Cisco ou Juniper.

Pour fonctionner ansible n'a besoin que d'un accès ssh et de python ou des APIs. Il n'y a pas de serveur central, tout ordinateur ayant Ansible peut commander les autres.

- <u>Autres produits ?</u>

  Les autres produits qui permettent l'orchestration de serveur :
  - Puppet (client/serveur)
    C'est projet qui a été créé en 2005 et qui est écrit en RUBY.
  - Chef (client/serveur)
    Il a été créé en 2009 par des ex-employé de Puppetlabs. Il a été écrit en RUBY puis ré-écrit en Erlang.
  - Saltstack (client/serveur ou agentless)
    Cette plateforme a vu le jour en 2011. Elle est écrite en python.

Ansible – Mise en oeuvre

2. Mise en œuvre de ANSIBLE

- Installation
  Ansible s'installe facilement et uniquement sur l'ordinateur de management.
  Elle peut se faire via les paquets de la distribution (UBUNTU/DEBIAN) ou via des dépôts tiers EPEL (REDHAT/CENTOS). Elle peut se faire aussi via PIP.

  Exemple : pip install ansible

  - Via pip on peut installer une version spécifique d'ansible
  - Python 2.5 au minimum doit être installé sur les machines cibles
  - Pour les machines utilisant SELINUX il faudra installer le paquet python-selinux

- Définition de l'inventaire

  Le fichier inventaire " *hosts* " se trouve dans *"/etc/ansible* " par défaut. Il définit la liste des machines qui pourrait répondre aux requêtes d'Ansible. Les ordinateurs définis dans le fichier " *hosts* "peuvent être
  organisés en groupe. On peut même créer des groupes de groupe. Ce fichier est modifiable dans la configuration
  d'ansible ou même spécifiable à l'execution des comma,des ansible.

Exemple :

```
[Ordonnanceur]
10.2.2.5
[file_slow]

[File_fast]
10.2.2.10
```

Ansible – Mise en oeuvre

- Les variables

  On peut définir des variables arbitraires qui peuvent être associées à une machine ou un groupe de machines. Elles permettent de modifier le comportement d'Ansible (info d'authentification). Les variables se définissent soit sur la ligne correspondante à la machine soit dans une section dédiée.

  - Il est commun d'utiliser la variable *"type"* qui sera utile non pas à ansible mais qui sera utilisée par la suite dans un rôle ou un playbook (ensemble de tâches à effectuer).

[file_slow]

nlame1.example.com type=master
nlame2.example.com type=slave

- Un autre exemple de variable est ansible_ssh_user. Elle est utilisé pour établir des connexions SSH avec le bon utilisateur.

  Exemple :

  [file_slow]
  nlame1.example.com ansible_ssh_user=root

- Enfin il est possible de définir des variables dans les dossiers group_vars et host_vars (Format YML)

  Exemple :
  [file_slow]
  nlame1.example.com type=master
  nlame2.example.com type=slave

  Contenu de : /etc/ansible/group_vars/file_slow
  ansible_ssh_user : ansible

Ansible – Mise en oeuvre

Contenu de : /etc/ansible/host_vars/
nlame1.example.com
type : master

Contenu de : /etc/ansible/host_vars/
nlame2.example.com
type : slave

- Les outils / les accès

  Les outils :

  ansible fournit plusieurs outils en ligne de commande :

Outil	Description
ansible	Execution d'une tâche
ansible-playbook	Execution de playbook (ensemble de tâches à effectuer)
ansible-doc	Accès au listing + documentation des serveurs
ansible-vault	Gestion de fichiers chiffrés (stockage variable - mot de passe)
ansible-galaxy	Accès au dépôt des rôles d'ansible

Le module **ping** permet de valider les accès d'un inventaire. Les machines cibles répondront par pong.

Exemple :
ansible –i hosts file_slow –m ping

nota :

- L'option -i spécifie le chemin vers l'inventaire
- L'option –m spécifie le module ansible à utiliser
- L'option all mis à la place de file_slow aurait impacté toutes les machines présentent dans l'inventaire

Ansible - Mise en œuvre

Atelier

- En utilisant deux machines virtuelles, exécuter le module ping d’ansible.

Ansible – Mise en oeuvre

Les accès :

L'accès SSH aux hôtes peut être explicité de plusieurs manières :

- En utilisant les informations de connexion dans l'inventaire
- En précisant toutes les informations sur la ligne de commande ansible
- En configurant la machine de management pour une connexion transparente

En pratique la méthode *"configurer la machine de management"* est la plus pratique à utiliser.
Pour cela il faudra :

- Créer une paire de clé SSH (ssh-keygen)
- Déployer la clé publique sur chacun des serveurs (ssh-copy-id)
- Créer une configuration sudo sans mot de passe sur les hôtes distants ou un login en tant que root. Le login utilisateur pour chaque hôte peut être spécifié dans l'inventaire.

Exemple :

```
$ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key
(/home/user/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
...
$ssh-copy-id ansible@nlame1.example.com
```

Nota: sudo peut être configuré pour ne pas demander de mot de passe lors de son utilisation

```
$cat /etc/sudoers.d/ansible
ansible ALL=(ALL) NOPASSWD : ALL
```

Ansible - Mise en œuvre

Atelier

- Relancer le module ping après avoir générer des clés ssh.

Ansible – Utilisation des modules

3. Utilisation des modules

- Définition
  Les modules sont la base des actions exécutées par Ansible. Chaque module est lié à une action spécifique. Les modules acceptent des arguments. La liste des modules proposé par l'application ansible est accessible grâce à la commande *"ansible-doc --list"*. On peut retrouver la liste des modules à cette adresse : ***http://docs.ansible.com/ansible/modules_by_category.html***

  La syntaxe est :
  ansible (hote/groupe/all) –m MODULE [-a *"arg1=val1"*]

- Exemple :
  $ ansible all –m yum –a *"name=httpd state=latest"*

Cette commande permet de vérifier si la dernière version d'apache est installée, si non, alors elle installe la dernière version.

Voici une liste de modules couramment utilisés :

- *ping* : validation de l'inventaire
- *setup* : retourne une liste d'informations matériels de l'hôte
- *shell* et *command* : permettent d'exécuter des commandes sur les hôtes
- *user* : permet de gérer des utilisateurs sur les hôtes
- *file* : permet de gérer des droits sur des fichiers
- *service* : permet de gérer les services systèmes tel que : arrêt/démarrage/redémarrage ou activation et désactivation au boot
- yum/apt/zypper : permet de gérer l'installation, la mise à jour et la suppression de paquets

Ansible - Mise en œuvre

Atelier

- À l’aide du module command, installer python sur les deux machines.
- À l’aide du module copy copier un script python vers les deux machines.
- À l’aide du module command, exécuter votre script python sur les deux machines.

Ansible – Playbook

4. Les Playbooks :

    1. Définition :
    Les playbooks permettent d'exécuter un ensemble de tâches à effectuer sur les machines hôtes de manière séquentielle sur tout ou partie de l'inventaire.

    Chaque tâche utilise un module de Ansible. On utilise le langage YAML pour écrire un playbook. Un playbook est constitué au minimum :
    - D'une variable *hosts* qui désigne les machines cibles
    - D'une variable *task* qui définit une action à accomplir.

    Les playbooks peuvent aussi :
    - Emettre des notifications, qui seront utiles pour déclencher une action dans certaines conditions
    - Effectuer des actions conditionnelles selon leur valeur (par exemple le type de distribution du système)
    - Utiliser des templates pour créer ou modifier des fichiers

    On trouve aussi dans les playbooks :
    - Des rôles à utiliser
    - Des handlers (tâches spécifique se déclenchant grâce à une notification)
    - Des éléments de configuration pour Ansible (par exemple l'utilisation de sudo)
    - Des variables spécifiques qui ne sont pas relié à l'inventaire.

Ansible – Playbook

Exemple :

```
- hosts: file_slow
  tasks:
  - name: Install apache
    yum:
      name: httpd
      state: present
      tags:
        -install_httpd
```

L'exécution du playbook install_apache.yml se fait grâce à la commande ansible-playbook :

```
$ansible-playbook install_apache.yml
```

Par défaut, un playbook s'exécute sur tous les hôtes concernés. Il est possible de de cibler ces actions grâce aux options :

- Tags : cette option se définie dans le playbook et on l'appelle --tags nom_du_tag ou –skiptags
- L'option –l qui permet de cibler une machine en particulier

Exemple :

```
$ansible-playbook install_apache.yml --tags install_httpd -l nlame1.example.com
```

La syntaxe YAML (Yet Another Markup Language)

C'est un langage facile à lire et à écrire. On peut y inclure une syntaxe JSON. On peut y manipuler du texte, des listes et des dictionnaires. L'indentation est très strict (2 espaces / indentation). On démarre un playbook toujours avec --- puis on va à la ligne. Enfin les commentaires sont défini grâce au #

Ansible - Playbook

Atelier

- Créer un premier playbook qui permet de:
- Installer Python
- Copier les scripts python d’une application.
- Exécuter le script python