Cours sur Ansible

1. Introduction à la gestion de configuration

La gestion de configuration est un concept essentiel en DevOps qui permet d'administrer et de gérer l'infrastructure informatique de manière automatisée et cohérente. Les ingénieurs DevOps utilisent la gestion de configuration pour assurer l'installation, la mise à jour, la sécurité et la maintenance des serveurs à grande échelle.

Pourquoi avons-nous besoin de la gestion de configuration ?

- **Administration efficace** : Un administrateur système gérant un parc de 100 serveurs (50 Linux, 25 CentOS, 25 Ubuntu) doit automatiser les mises à jour, les correctifs de sécurité et les installations logicielles.
- **Scalabilité** : Dans le cloud, où les microservices peuvent multiplier les serveurs par 10, gérer des milliers de machines devient impossible manuellement.
- **Standardisation** : La gestion de configuration permet d'avoir des configurations homogènes sur tous les serveurs.

2. Outils de gestion de configuration

Des outils tels que Puppet, Chef, SaltStack et Ansible ont été développés pour répondre à ces besoins. Parmi eux, **Ansible** est l'outil le plus populaire car il est simple et puissant.

Pourquoi Ansible ?

Ansible est développé par **Red Hat** et est préféré par de nombreux ingénieurs DevOps pour plusieurs raisons :

- **Agentless** : Contrairement à Puppet qui nécessite une architecture maître-esclave, Ansible n'a besoin que d'un accès SSH aux serveurs.
- **Simplicité** : Utilise YAML pour écrire ses configurations sous forme de **playbooks**.
- **Flexible** : Prend en charge aussi bien Linux que Windows (via SSH pour Linux et WinRM pour Windows).
- **Gestion dynamique** : Peut détecter et gérer automatiquement de nouveaux serveurs grâce à l'inventaire dynamique.

3. Ansible vs Puppet

Critère	Ansible	Puppet
Mode de fonctionnement	Push (pousse les configurations sur les serveurs)	Pull (les serveurs récupèrent leur configuration depuis un serveur maître)
Agent	Agentless	Requiert un agent
Langage de configuration	YAML	DSL spécifique à Puppet

Critère	Ansible	Puppet
Facilité d'utilisation	Simple à apprendre	Plus complexe
Gestion Windows	Supporté mais plus compliqué	Meilleur support
Exécution parallèle	Peut être limité sur de grandes infrastructures	Plus performant

4. Fonctionnement de Ansible

Ansible fonctionne en utilisant un **fichier d'inventaire** qui liste les serveurs à gérer et en exécutant des **playbooks** contenant les tâches d'automatisation.

Inventaire

L'inventaire est un fichier qui contient les adresses IP des serveurs à gérer. Exemple de fichier `inventory.ini` :

```
[webservers]
192.168.1.10
192.168.1.11

[dbservers]
192.168.1.20
```

Playbook

Un **playbook** est un fichier YAML qui décrit les tâches à exécuter sur les serveurs. Exemple de playbook `install_apache.yml` :

```
- hosts: webservers
  become: yes
  tasks:
    - name: Installer Apache
      apt:
        name: apache2
        state: present
```

Ce playbook installe Apache sur les serveurs définis dans `[webservers]`.

5. Déploiement avec Ansible

Scénario : Déploiement sur 10 serveurs EC2 sur AWS.

1. **Créer un inventaire dynamique** pour récupérer les IP des instances AWS.
2. **Écrire un playbook** pour installer les dépendances et configurer les serveurs.
3. **Exécuter le playbook** avec la commande `ansible-playbook`.

6. Ansible Galaxy

Ansible permet de partager et réutiliser des rôles grâce à **Ansible Galaxy**, un dépôt public de rôles prêts à l'emploi. Exemple :

```
ansible-galaxy install geerlingguy.apache
```

7. Limitations d'Ansible

- **Gestion Windows** : Plus compliquée que Linux.
- **Performance** : Moins efficace que Puppet sur des milliers de serveurs.
- **Debugging** : L’exécution parallèle peut être difficile à déboguer.

8. Questions d'entretien

1. **Quel langage de programmation est utilisé pour développer Ansible ?**
   - Python.
2. **Ansible supporte-t-il Linux et Windows ? Si oui, quel protocole est utilisé ?**
   - Oui, via SSH pour Linux et WinRM pour Windows.
3. **Quelle est la différence entre Ansible et Puppet ?**
   - Ansible est un outil **push**, sans agent, tandis que Puppet est un outil **pull**, nécessitant un serveur maître.
4. **Pourquoi choisir Ansible ?**
   - Simplicité, agentless, utilisation de YAML.
5. **Ansible est-il un outil de type push ou pull ?**
   - Push.
6. **Quels clouds sont supportés par Ansible ?**
   - AWS, Azure, GCP, etc.

9. Prochain cours

Demain, nous mettrons Ansible en pratique en écrivant des playbooks et en les exécutant sur des instances EC2 sur AWS.

Fin du cours.