Infra - Ansible

- Ansible
- Mise en place & Get Started

Ansible

Ansible est un outil d'automatisation IT orienté **configuration management**, **déploiement** et **orchestration**.

> tu décris *l'état souhaité* de ton infra… et Ansible s'occupe de le rendre réel.

Cas d'usage concrets

- Installer/configurer des serveurs (nginx, users, firewall…)
- Maintenir la cohérence entre environnements (dev/staging/prod)

Alternative à Puppet / Chef.

3. Orchestration multi-serveurs

Exemple :

1. Provisionner VM
2. Installer DB
3. Déployer backend
4. Configurer load balancer

Tout dans un seul workflow.

4. Infrastructure as Code (IaC)

- Décrire ton infra en YAML
- Versionner dans Git
- Reproductibilité totale

Concepts clés

- **Inventory** → liste des serveurs
- **Playbooks** → fichiers YAML (le cœur)
- **Modules** → actions (apt, service, copy…)
- **Roles** → structure modulaire propre

Mise en place & Get Started

Prérequis :

- Avoir un poste Linux
- Avoir Docker et Docker Compose d'installé

Installation Ansible

Pour installer Ansible sur linux il convient d'exécuter la commande d'installation suivante :

```
sudo apt install ansible python3-pip -y
```

Et pour avoir le lint d'aide :

```
pip install ansible-lint
```

Créer des “faux serveurs” avec Docker

Pour ce faire nous allons écrire un `docker-compose` contenant 4 services tirant les images ubuntu :

```
version: "3"

services:
  marseille:
    image: ubuntu:22.04
    container_name: marseille
    command: sleep infinity

  nantes:
    image: ubuntu:22.04
    container_name: nantes
    command: sleep infinity

  paris:
    image: ubuntu:22.04
    container_name: paris
```

```
    command: sleep infinity

  vps:
    image: ubuntu:22.04
    container_name: vps
    command: sleep infinity
```

Puis créer les conteneurs :

```
docker compose up -d
```

> **Préparer les containers (IMPORTANT) ->** Ansible a besoin de Python minimum.

```
docker exec -it marseille apt update
docker exec -it marseille apt install -y python3

docker exec -it nantes apt update && docker exec -it nantes apt install -y python3
docker exec -it paris apt update && docker exec -it paris apt install -y python3
docker exec -it vps apt update && docker exec -it vps apt install -y python3
```

Créer un inventaire

Créer le fichier suivant `inventory.yml` :

```
all:
  children:
    main:
      hosts:
        marseille:

    backup:
      hosts:
        nantes:
        paris:

    brain:
      hosts:
        vps:
```

```
  vars:
    ansible_connection: docker
    ansible_python_interpreter: /usr/bin/python3
```

Ici on utilise la connexion Docker (pas SSH).

Test de base de l'inventaire : `ansible all -i inventory.yml -m ping`

Créer un playbook

Par exemple nous créer le fichier `helloworl.yml` et y mettre la configuration suivante :

```
- hosts: all
  become: true
  tasks:
    - name: Install curl
      apt:
        name: curl
        state: present
        update_cache: yes

    - name: Create test file
      file:
        path: /tmp/ansible_test
        state: touch
```

Puis exéctuer le playbook sur les "serveurs" : `ansible-playbook -i inventory.yml helloworl.yml`

Pour vérifier le bon déroulement du playbook nous allons voir si le fichier `ansible_test` a bien été créé dans les conteneurs :

```
docker exec -it marseille ls /tmp
```

> Une fois un playbook exécuté, les conteneurs vont garder leurs états et la configuration. Pour nettoyer les serveurs et repartir de zéro, il faut exécuter la commande suivante : `docker compose down -v`