!SLIDE

Ansible

Ansible est un outil d'automatisation neutre et prévisible.

Sa courbe d'apprentissage est douce, et il est pertinant de l'utiliser à partir de 1 serveur.

Il n'utilise pas d'agents, et ne requiert qu'un accés SSH et un python sur le serveur.

Largement soutenu par la communauté, il fait partie du top 10 des applications Python sur Github.

!SLIDE

Inventaire

L'inventaire contient la liste des groupes, des machines à répartir dedans, et des paramètres correspondant à cet inventaire.

L'inventaire peut être statique, ou dynamique (cloud et clusters).

On utilise différents inventaires pour les différents environnements : test, préprod, prod...

!SLIDE

Scénario

Le scénario regroupe des tâches à effectuer par des groupes de machines.

Le DSL utilisé n'en est pas un, c'est un mélange de YAML et de Jinja2.

La lisibilité des recettes est une priorité.

L'organisation des scénarios est souple.
Un scénario peut être dédié à une action complexe, comme l'ajout d'un noeud dans un cluster, ou une mise à jour roulante.

!SLIDE

Taches

Un scénario regroupe des taches.

Les taches font des actions simples, sans paraphraser les choses que l'on connait déjà.

La plupart des taches concernent la création de fichiers de configurations à
partir de gabarits, et de lignes de commandes traditionnelles.

La commande Shell, bien que déconseillée, est l'arme ultime qui garantit que l'on ne sera jamais coincé.

!SLIDE

Séquentielles

L'exécution des tâches s'effectue de manière séquentielle, pour limiter les surprises.
Ansible ne croit pas à la gestion de dépendances et à la parallélisation.

Les actions sur un ensemble de machines sont, elles, parallélisées, par lots de tailles raisonnables.

!SLIDE

Idempotance

Une action peut être rejouée plusieurs fois.

Une tache vérifie que son action va servir à quelque chose avant de se lancer.

On demande un résultat, pas la méthode pour y arriver.

Normalement, un scénario peut être rejoué sans risques, mais il est plus sage de les travailler dans une VM.

!SLIDE
Variables

L'utilisation de la syntaxe Jinja dans les divers fichiers YAML, et l'approche
rigoureuse "Tout est explicite" permet un usage efficace et sans surprise des variables.

Chaque lecture de scénario commence par "La collecte des faits", et permet d'accéder à différentes informations des machines présentent dans l'inventaire (nom de l'OS, adresse IP des différentes interfaces...)

Le résultat d'une action peut être enregistré.

!SLIDE

Gabarits

Jinja2 est le format de gabarit utilisé par Ansible.
Standard de fait dans le monde Python, il ne surprend personne
et ressemble à beaucoup de choses déja vu dans d'autres technologies.

Son système de remplacement de variables et de filtre est aussi utilisé dans les fichiers YAML.

!SLIDE

Boucles

Ansible a clairement une approche déclarative, orthogonale à la notion de script.

Il est quand même possible d'effectuer des itérations simples, comme installer une liste de paquets.

!SLIDE
Clauses

Il est possible de rendre certaines parties du flot ansible conditionnel.
Les taches, évidemment, mais aussi les "includes" ou les rôles.

!SLIDE
Rôles

Un rôle regroupe un ensemble de taches, et tout ce qui va avec (gabarits, fichiers...).

C'est l'unité de base pour capitaliser le travail entre différents projets.

Un rôle peut dépendre d'un autre rôle.

!SLIDE
Notifications

Pour éviter de devoir faire une action systématiquement, il est possible d'utiliser une notification.
La notification sera envoyée (à la fin), si la tache a effectué une modification.

Son usage classique et de lier un fichier de conf au redémarrage du service correspondant.

!SLIDE
Démonstration