

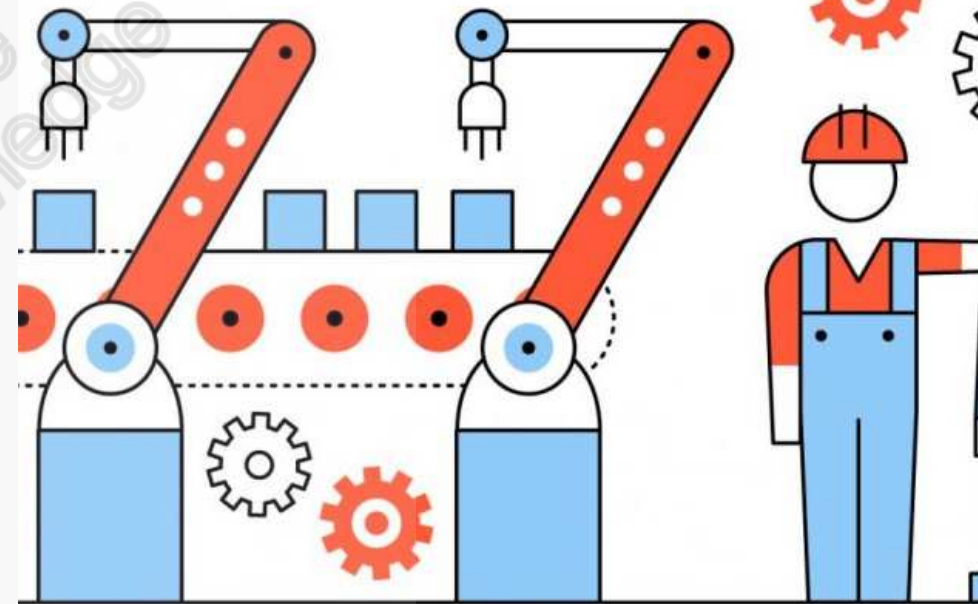


Global Knowledge®

# DEVOPS AUTOMATION

Mohamed Taher BEN BAHRI

AUTOMATION



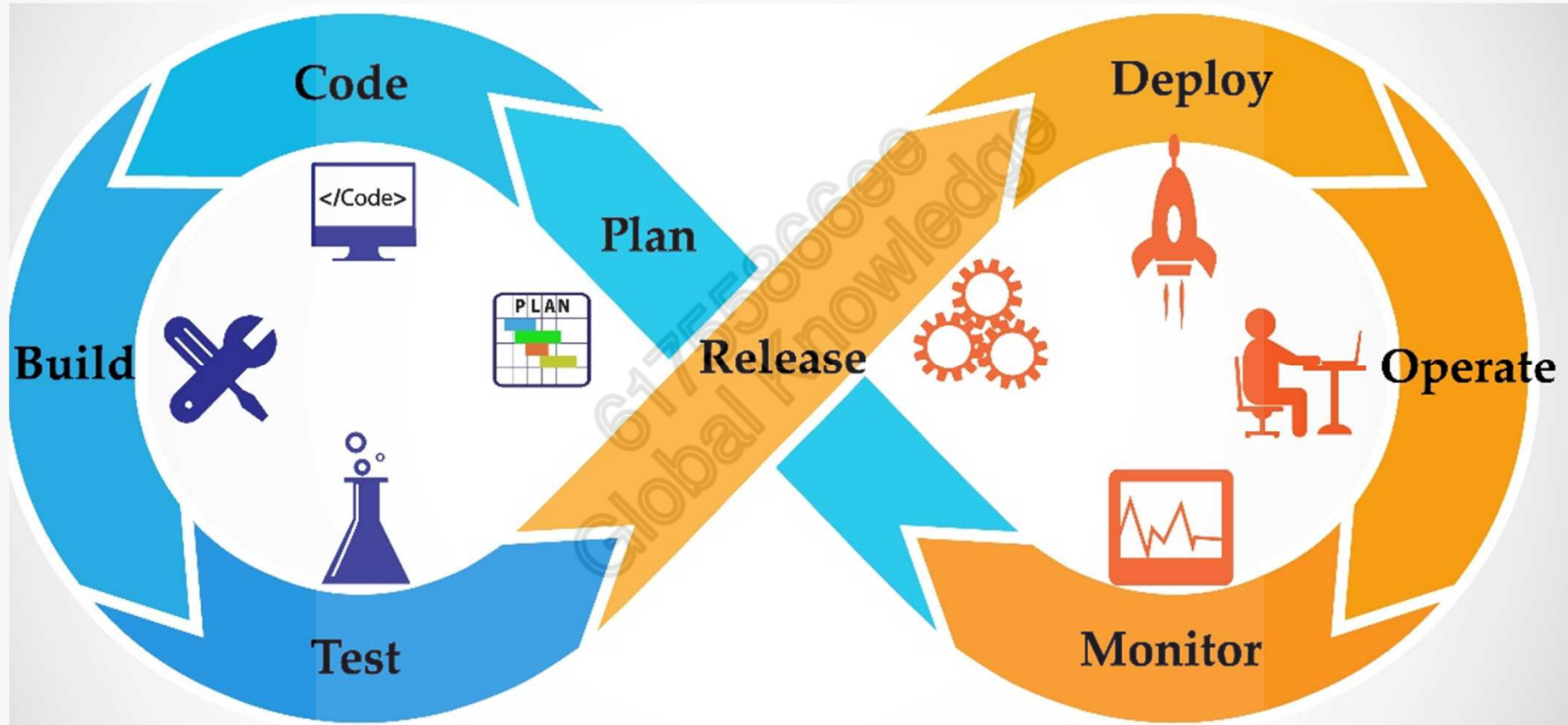
# Plan

---

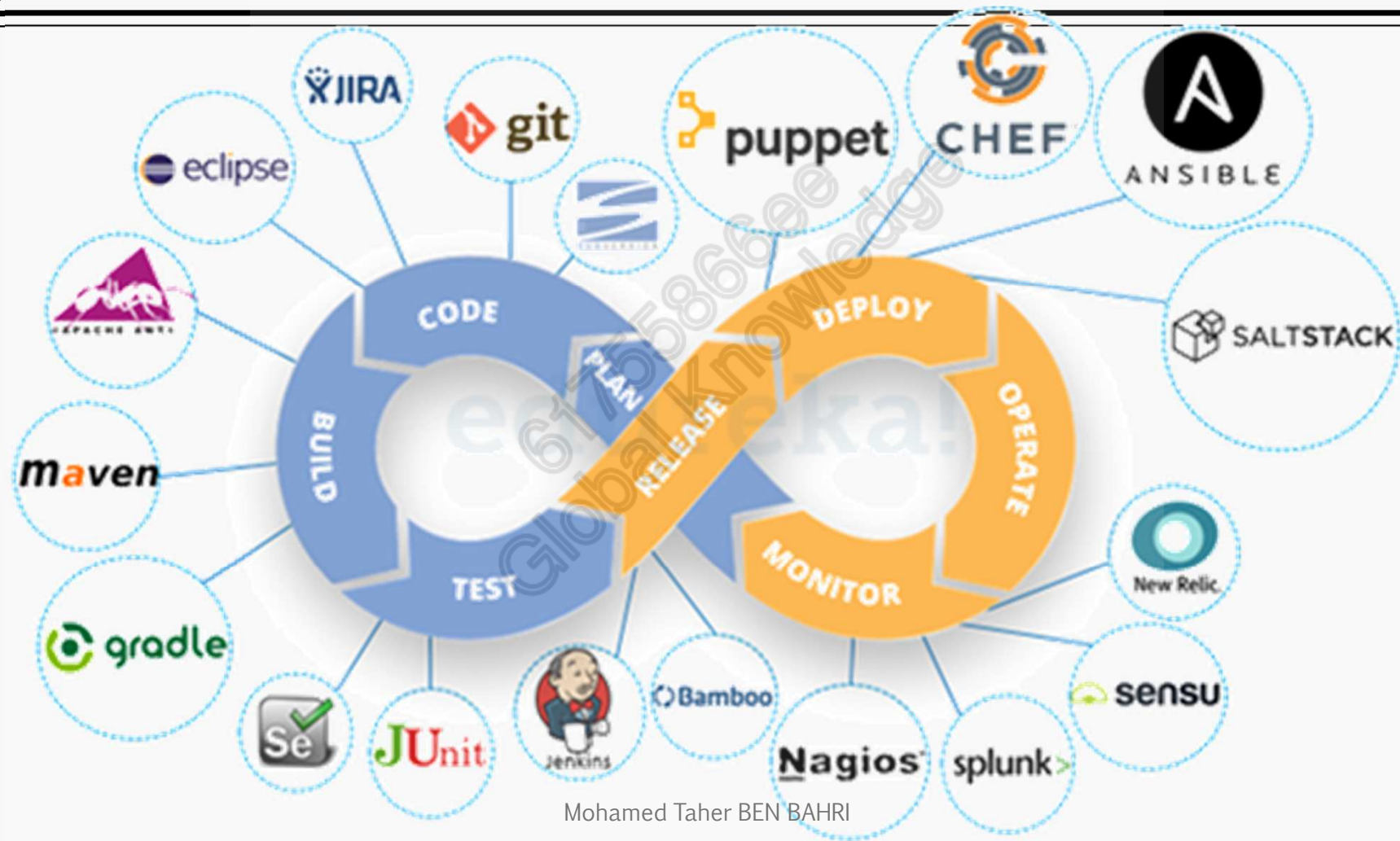
- Outils du cycle de vie DevOps
- What are the stages in DevOps
- How the tools operate in each of the DevOps Stages
- How to use DevOps tools in a real-life scenario

## DevOps : Cycle de vie

---



## Cycle de vie DevOps : Outils

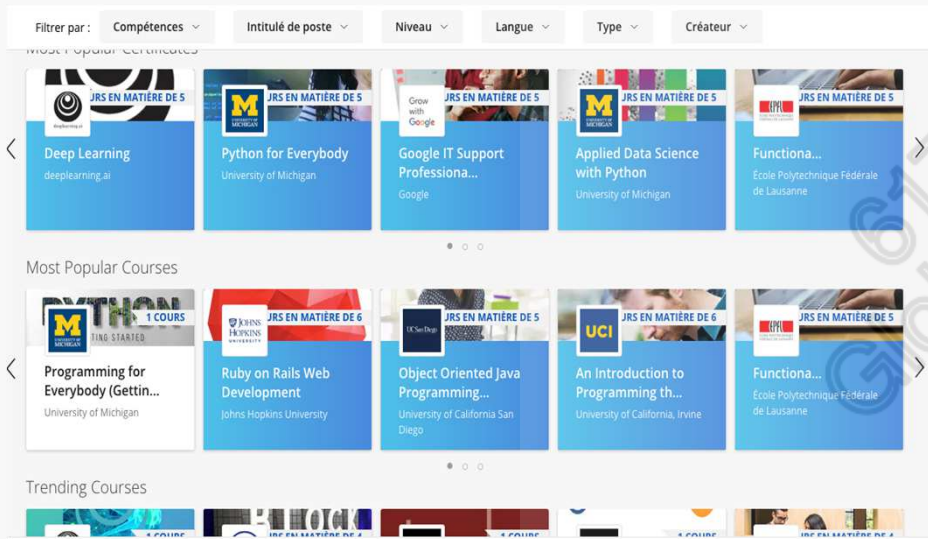


# USE-CASE

617558660e  
Global Knowledge

# Énoncé du problème: Déploiement en masse

Un site Web d'apprentissage regroupe chaque jour plus de 1000 apprenants et 600 cours en parallèle. Son auditoire est réparti sur 5 régions allant de 70 pays. Il a plusieurs serveurs en cours d'exécution à travers le monde.



## Dark Launching scenario

La société propriétaire a créé une nouvelle page d'accueil avec une vue modifiée et plusieurs nouvelles fonctionnalités. Il fonctionne sur CakePHP 3.x et d'autres nouveaux logiciels. Ils veulent lancer cette page d'accueil uniquement pour que le public américain puisse voir les feedbacks du public américain. Nous devons déployer cette page d'accueil sur les serveurs fonctionnant uniquement dans la région américaine. Pour cela, les serveurs situés dans cette région doivent disposer de la dernière version de CakePHP et des logiciels permettant d'exécuter la nouvelle page d'accueil.



# Solution

---



Ils utilisent plusieurs serveurs aux États-Unis.

- ☐ Si nous changeons chaque serveur manuellement, cela prendra beaucoup de temps et ce n'est pas évolutif
- ☐ Si nous exécutons un script Shell pour faire le travail, cela fonctionnera, mais il est complexe de l'exécuter sur plusieurs serveurs, et toute modification apportée au script à l'avenir entraînera une plus grande complexité.
- ✓ Par conséquent, nous avons besoin d'un outil de gestion de la configuration, qui peut effectuer ce travail à la volée automatiquement et s'adapter également à des milliers de serveurs sans aucun problème.

# Scénario 1: Déploiement en masse

S'il est facile de déployer une application sur 5 machines virtuelles, il est impossible pour un humain de déployer une application sur 500 machines virtuelles rapidement, nous avons besoin d'un outil capable de faire ce travail.

Déployer  
l'application sur  
5 VM



## Tâches

- Installer le SE
- Configurer la sécurité
- Installer les bibliothèques requises
- Installer Serveur Apache v2.4
- Installer Serveur BD MySQL 5.5
- Déployer le code de l'application
- Installer la BD
- Importer les Données
- Configurer les serveurs App et DB
- Configurer les Hôtes Virtuelles
- Faire le pointage DNS
- ...

Déployer  
l'application sur  
500 VM





## Scénario 2: Migration du Test au Prod

---

En raison de la différence entre l'environnement de test et de production, des applications fonctionnent en test mais pas en production.  
Il devrait y avoir un outil pour gérer les différences d'environnement



## Scénario 3: Défaillance de l'Application

---


Il n'existe aucun moyen de tracer les modifications faites sur un serveur, mais aussi les méthodes sont complexes pour restauration d'une application défaillante. Il devrait exister un outil qui permet de restaurer automatiquement l'application vers une version stable en cas de défaillance.

## Scénario 3: Etude de Cas

Premier site de réservation de voyages au Royaume-Uni, 1,2 milliard de livres de chiffre d'affaires annuel, plus de 9 millions de visites par mois



Énorme perte d'argent due aux temps d'arrêt!



**NOUS AVONS BESOIN D'OUTILS DE GESTION  
DE LA CONFIGURATION ET DE DÉPLOIEMENT  
POUR L'AUTOMATISATION INFORMATIQUE.**

# INFRASTRUCTURE as CODE (IaC)

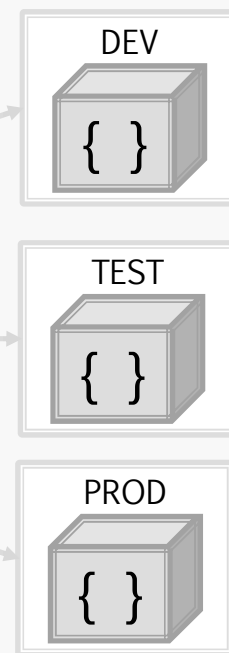
# Infrastructure as Code (IaC)

---

IaC est l'automatisation des opérations informatiques (construction, déploiement, gestion) en fournissant du code, plutôt qu'un processus manuel.



Provisioning des environnements de développement, de test et de production en écrivant du code dans un emplacement centralisé





# Infrastructure as Code (IaC)

## Script Shell

```
echo  
"spock*:*:1010:1010:Spock:/home/sp  
ock:/bin/sh" \ >> /etc/passwd  
(the user Spock is added to  
/etc/passwd)
```

## CM Tool Management

```
user { "spock":  
  ensure => present,  
  gid => "science",  
  home => "/home/spock",  
  shell => "/bin/sh"  
}
```

- Dans un script shell, vous devez écrire un script d'automatisation de toutes pièces, mais dans l'outil de gestion de la configuration (CM), 80% des éléments sont déjà disponibles.
- Dans un script shell, vous devez définir le workflow alors que dans l'outil CM, les workflows sont déjà présents.
- Vous disposez d'une interface utilisateur dans CM pour faciliter votre travail d'automatisation des tâches, mais vous n'avez pas d'interface utilisateur dans un script shell

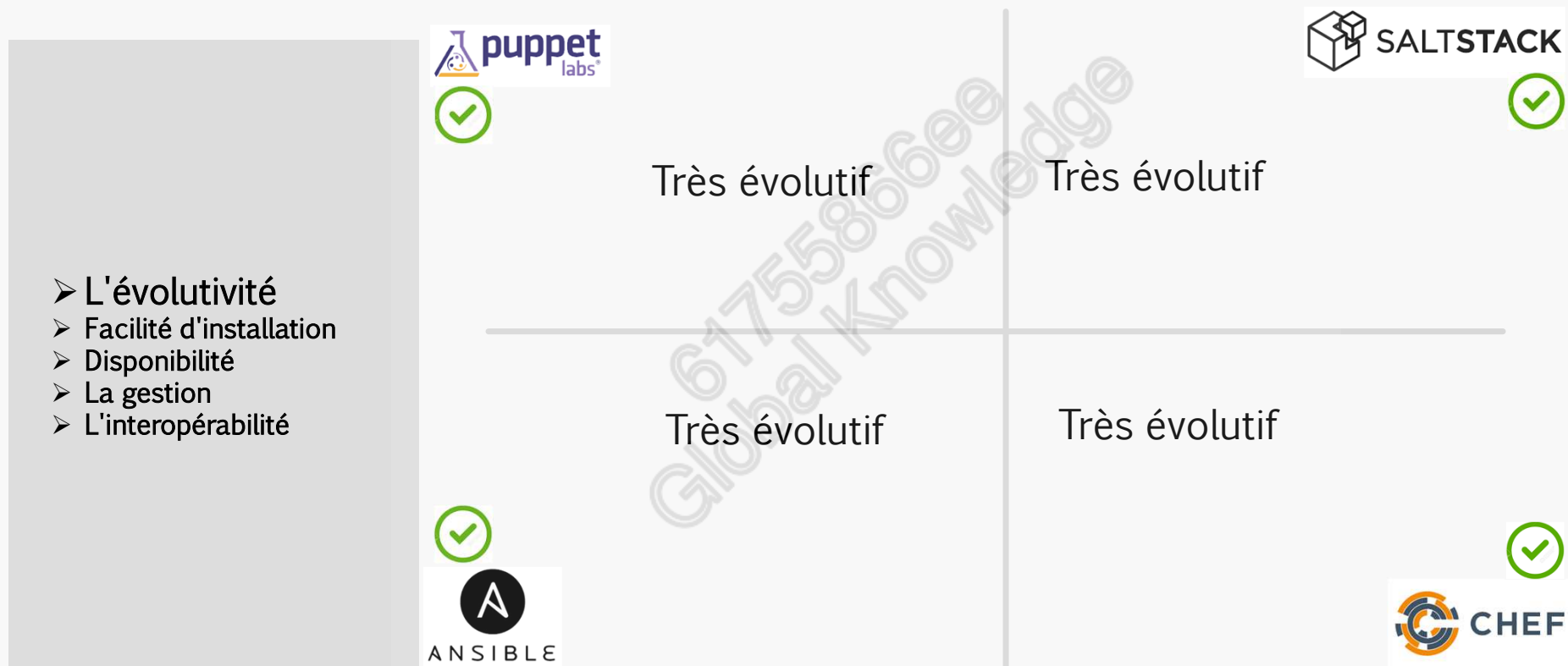




# ETUDE COMPARATIVE PUPPET VS CHEF VS SALTSTACK VS ANSIBLE

Scalabilité - Facilité d'installation - Disponibilité - Management- Interopérabilité

# Scalability



# Facilité d'Installation

- L'évolutivité
- Facilité d'installation
- Disponibilité
- La gestion
- L'interopérabilité



Master - Agent

Le serveur Puppet s'exécute sur la machine principale et les clients Puppet s'exécutent en tant qu'agents sur chaque machine cliente.



Master - Node

Il ne fonctionne que sur le serveur, mais pas d'agents sur les clients; il utilise ssh pour se connecter aux systèmes clients. Les machines clientes (VM) n'ont besoin d'aucune configuration particulière, il est donc plus rapide à configurer



SALTSTACK



Master - Agent

Le serveur s'appelle Master et les clients sont appelés Minions qui s'exécutent en tant qu'agents sur les machines des clients.

Master - Agent

Le serveur Chef s'exécute comme l'ordinateur principal et les clients Chef s'exécutent en tant qu'agents sur chaque ordinateur client.



# Disponibilité

- L'évolutivité
- Facilité d'installation
- **Disponibilité**
- La gestion
- L'interopérabilité



Hautement disponible

Il a une architecture multi-maîtres, si le maître actif tombe en panne, l'autre maître prend la place du maître actif.



ANSIBLE

Hautement disponible

S'exécute avec un seul nœud actif, appelé instance principale, si le principal tombe en panne, une instance secondaire est remplacée



SALTSTACK

Hautement disponible



Plusieurs maîtres peuvent être configurés. Si un maître est en panne, les agents se connectent à l'autre maître de la liste.

Hautement disponible

En cas de défaillance du serveur principal, un serveur de sauvegarde prend la place du serveur principal.



CHEF

# Management

- L'évolutivité
- Facilité d'installation
- Disponibilité
- **Management**
- L'interopérabilité



Pas très facile à apprendre à gérer les configurations car il a son propre langage appelé Puppet DSL. Les clients extraient la configuration du serveur.  
Exécution à distance non immédiate



Facile à apprendre, à gérer les configurations  
Le serveur envoie la configuration à tous les nœuds  
Bon pour les applications en temps réel  
Exécution à distance immédiate



Facile à apprendre à gérer les configurations  
Le serveur fournit des configurations à tous les clients  
Exécution à distance immédiate

Vous devez être programmeur pour gérer les configurations car il offre des configurations en Ruby DSL  
Les clients extraient la configuration du serveur





# Interopérabilité

---

- L'évolutivité
- Facilité d'installation
- Disponibilité
- Management
- L'interopérabilité



Puppet Master ne fonctionne que sous Linux/Unix mais Puppet Agent fonctionne également sous Windows



Ansible Server doit être installé sur une machine Linux/Unix, mais il prend en charge les nœuds Windows.



SALTSTACK



Salt Master fonctionne uniquement sur Linux/Unix mais Salt Minion fonctionne également sur Windows



Chef Master fonctionne uniquement sur Linux/Unix mais le client et Workstation Chef fonctionnent également sur Windows

# PUPPET VS CHEF VS SALTSTACK VS ANSIBLE

Plus de facteurs à considérer:

Config language - Github Activity – Entreprise Cost - Popularity – Success Story

# Langage de Configuration

➤ **Conf. Lang.**

- Github Activity
- Entreprise Cost
- Popularity
- Success Story



Configuration: DSL (Puppet DSL)  
Pas très facile à apprendre,  
orienté administrateur



Configuration: YAML (Python)  
Facile à apprendre, orienté  
administrateur  
Python est intégré à la plupart des  
déploiement Unix et Linux, ce qui rend  
l'outil plus rapide à mettre en oeuvre.

Configuration: YAML (Python)  
Facile à apprendre, orienté  
administrateur  
Python est intégré à la plupart des  
déploiement Unix et Linux, ce qui rend  
l'outil plus rapide à mettre en oeuvre.

Configuration: DSL (Ruby)  
Courbe d'apprentissage abrupte,  
orientée développeur



Mohamed Taher BEN BAHRI



DSL: Domain Specific Language

YAML: Yet Another Markup Language

# GitHub Activity

---

- Conf. Lang.
- **Github Activity**
- Enterprise Cost
- Popularity
- Success Story



Contributors: 355  
Commits: 19595  
Branches: 9  
Releases: 291



Contributors: 1003  
Commits: 13527  
Branches: 33  
Releases: 57



Contributors: 1041  
Commits: 49193  
Branches: 11  
Releases: 82



Contributors: 369  
Commits: 12089  
Branches: 177  
Releases: 231

# Enterprise Cost

---

- Conf. Lang.
- Github Activity
- **Enterprise Cost**
- Popularity
- Success Story



Puppet Enterprise :\$12000



SaltStack Enterprise :\$15000



Ansible Enterprise Tower  
:\$10000

Chef Enterprise :\$7200



# Popularity

- Conf. Lang.
- **Github Activity**
- Enterprise Cost
- Popularity
- Success Story



- ✓ Puppet and Chef sont des anciens joueurs, puppet est plus adopté
- ✓ Saltstack and Ansible sont des nouveaux joueurs, Ansible semble très prometteur avec la tendance croissante



# Success Story

- Conf. Lang.
- Github Activity
- Enterprise Cost
- Popularity
- Success Story



"75% of ICE's (International Exchange) 20000 servers are managed by Puppet Enterprise. They increased from 300 servers per admin to 700 servers per admin. Provisioning dev environment reduced from 1 or 2 days to 21 minutes"



"LinkedIn had about 5000 Salt Minions under management 4 years ago, That number has ballooned to more than 70000 today to manage their infrastructure"



"Using Ansible, we have been able to cut down certain processes from 17 hours to 30 minutes. Ansible is unquantifiable in its benefits"



"Chef has increased the effectiveness and the speed of our development cycle. 30% of Gannett's technology organization used Chef but Gannett has aggressive plans for 100% adoption in this year 2016"

