Ansible automatise l'approvisionnement, la gestion de la configuration, le déploiement des applications, l'orchestration intra-service, la livraison continue et de nombreux autres processus informatiques. Il utilise un langage simple appelé YAML pour gérer les tâches de configuration, de déploiement et d'orchestration et ne nécessite aucun agent. Ansible Tower est une interface utilisateur basée sur le Web pour Ansible.

Ansible fonctionne en se connectant à vos nœuds et en exécutant de petits programmes, appelés modules Ansible, pour configurer les ressources de votre système. Ansible exécute ces modules sur Secure Shell (SSH) par défaut et les supprime une fois terminé.

Le rapprochement entre Dev et Ops, ou DevOps, a donné naissance au concept d’Infrastructure As Code : il s’agit notamment d’automatiser l’infrastructure, et à cet effet on a besoin d’outils dont la fonction primaire est la gestion des configurations systèmes.

Aujourd’hui, les plus connus sont Puppet, Chef, Salt et Ansible.

Depuis plus de 10 ans, les technologies d'orchestration et d'automatisation ont permis aux équipes en charge des opérations d'améliorer la qualité de service, mais également de réduire les délais de mise en œuvre, ainsi que les coûts.

Ces technologies se sont adaptées au fur et à mesure des évolutions technologiques, telle que la virtualisation, et sont encore à l’heure actuelle le fer de lance de nombreuses Directions des Opérations. Elles sont pour l’essentiel proposées par des éditeurs de logiciel sous forme d’une licence commerciale, le code source de ces outils n’étant pas disponible.

En parallèle, une nouvelle forme de consommation de l’informatique est apparue, le « Cloud Computing », avec des outils mettant à disposition le code source sous licence publique. Ainsi autour d’acteurs tels que Amazon ou Google, un écosystème est apparu. Parmi tous ces systèmes, certains ont pour vocation de proposer des services d’orchestration et d’automatisation de ce nouveau monde de consommation.

Enfin, les technologies de type Cloud sont apparues peu à peu au sein des Datacenters des entreprises (OpenStack, CI/CD, PaaS, etc.) et cette tendance ne fait que s’accentuer.

Aujourd’hui le challenge des Directions des opérations est de maintenir leurs anciens systèmes (legacy), tout en adoptant les nouveaux aux modes opératoires déjà en place.

Dans cette optique, nous allons examiner dans cet article les outils Ansible et Puppet qui sont deux produits très utilisés aujourd’hui.

Avec le rapprochement entre Dev et Ops et la forte augmentation du nombre de serveurs due à la prolifération de la virtualisation et à la disponibilité du cloud computing, le concept d’Infrastructure As Code a vu le jour : il s’agit notamment d’automatiser l’infrastructure, autrement dit nous devons gérer des serveurs au sein et en dehors de l’entreprise. C’est dans cette perspective qu’interviennent les outils d’orchestration de Datacenter et de gestion de configuration pour automatiser le processus de déploiement des services IT. Ainsi Chef, Puppet, Ansible et SaltStack ont été conçu pour simplifier la configuration et la maintenance des dizaines, voire des milliers de serveurs. Mais cela ne veut pas dire que les petits magasins ne vont pas bénéficier de ces outils car l’automatisation et l’orchestration facilitent généralement la vie dans une infrastructure de toute taille.

* Puppet, c’est le plus complet en termes d’actions disponibles, de modules et d’interfaces utilisateur. Il englobe tous les systèmes d’exploitation et offre des outils approfondis pour ces derniers. Sa configuration initiale est relativement simple mais il nécessite l’installation d’un serveur maitre et d’agents clients sur chaque système à gérer.

Puppet est initialement un outil de gestion de configurations, aidant les administrateurs à maintenir leur infrastructure, la provisionner et la configurer.  
Avec Puppet, il est possible d'automatiser des tâches répétitives, déployer des applications et être proactif sur les changements d'une infrastructure.

● **Facter** : Variables d'environnements Ruby  
● **PuppetDB** : Permet de mettre en cache des données générées par Puppet afin d'accélérer ce dernier  
● **Mcollective** : Framework d'orchestration pour gérer simultanément un parc de machine  
● **Hiera** : outil de configuration clé/valeur pour faciliter l'écriture des configurations Puppet (intégré à Puppet depuis la version 3.0)   
● **Dashboard** : Interface web de management pour analyser les rapports

**Introduction**

Avec la généralisation des solutions faisant intervenir le cloud et le besoin dynamique dans les architectures mise en œuvre, les outils d'orchestration sont rapidement devenus indispensables. Là où auparavant, un site web était hébergé sur un unique serveur facilement administrable, la plupart des hébergeurs proposent maintenant des solutions intégrant de la répartition de charge (load balancing) selon les besoins dynamiques du site. Ces sociétés ont donc un fort besoin d'orchestration: installation d'une nouvelle machine, installation des services associés, intégration avec le service en cours d’exécution, entre autres.

Tout administrateur système, quelle que soit la taille du parc informatique qu’il a sous sa responsabilité; va chercher à automatiser au possible les tâches pénibles ou répétitives. On pourrait bien entendu imaginer dans un premier temps une connexion distante réalisée manuellement par l’administrateur : une connexion ssh suivie d’un script bash que l’on exécute. Cependant, cette solution est très limitée, il faut connaitre les mots de passe de chaque machine auxquelles on veut se connecter, ce qui devient très rapidement une perte de temps majeure dans une infrastructure conséquente. De plus, l’aspect répétitif reste encore très présent et pour peu que les machines aient quelques différences de configuration, le script qui fonctionne sur une ne fonctionnera pas nécessairement sur l’autre. Il est très fréquent dans les parcs informatiques assez anciens d’avoir plusieurs versions de différents logiciels en cours d’utilisation suivant la date d’installation du service. Les mises à jour deviennent très rapidement un casse-tête, et ne sont très souvent jamais réalisées (bien que d’autres problématiques interviennent pour cet exemple, telles que la compatibilité de l’application courante). Sans pour autant vouloir assurer une mise à jour, la modification d’un paramètre commun à l’ensemble des machines devient ingérable. De manière générale, un parc informatique va très rapidement être très hétérogène, ce qui tend à complexifier l’administration.

En tant qu’administrateur système, on va vouloir conserver autant que possible une vue générale de notre parc informatique, en décrivant au travers d’outils de haut niveau la configuration souhaitée, sans se préoccuper de sa réalisation effective. Autrement dit, se concentrer que le « quoi » plus que sur le « comment ». Ce paradigme est dit « déclaratif », et plusieurs outils présentés par la suite vont choisir cette approche. D’autres auront une approche un peu plus mixte, se rapprochant du paradigme « impératif » et de la réalisation concrète des actions (le « comment » évoqué ci-dessus). Avec les deux paradigmes évoqués, nous touchons directement un concept qui résume parfaitement l’idée des outils d’orchestration : « infrastructure as code ». Quel que soit l’outil utilisé, nous allons chercher à décrire notre infrastructure, les configurations souhaitées sous forme de « code » plus ou moins haut-niveau. Ces outils intègrent également les notions de push et pull, présentes dans les outils de versioning, tel que Git.

Un point reste à mentionner, avant de commencer cette étude : la différence entre outils d’automatisation et outils d’orchestration, entre « tache » et « processus ». L’optimisation d’un processus ne peut pas être réalisée simplement en automatisant des tâches. L’automatisation va être autour d’une seule tâche, l’orchestration va être sur l’ensemble du workflow, sur le processus global. Par exemple, on va pouvoir automatiser l’installation des différents composants d’une architecture trois-tiers. L’outil d’orchestration ne va pas se limiter à installer ces composants, mais s’assurer qu’ils sont déployés dans le bon ordre, en optimisant le processus global.

Puppet est un outil de gestion de configuration logicielle open source, inspiré du langage de Ruby pour la description des configurations. Autrement dit, une bonne connaissance en Ruby est recommandée.

Il se base sur une approche client-serveur avec un ou des serveurs maitres qui vont centraliser la description des états voulus par les clients.

Il est multiplateforme c’est-à-dire il prend en charge Linux, Windows, AIX, etc.