Glossaire Ansible

Glossaire Ansible

Glossaire Ansible.

Action

Une action fait partie d'une tâche qui précise les modules à exécuter et les arguments à transmettre à ce module. Chaque tâche ne peut avoir qu'une seule action, mais elle peut aussi avoir d'autres paramètres.

Ad Hoc

Désigne l'exécution d'Ansible pour exécuter une commande rapide, en utilisant `/usr/bin/ansible`, plutôt que le langage d'orchestration, qui est `/usr/bin/ansible-playbook`. Un exemple de commande ad hoc pourrait être le redémarrage de 50 machines dans votre infrastructure. Tout ce que vous pouvez faire ad hoc peut être accompli en écrivant un livre de jeu et les livres de jeu peuvent également cumuler beaucoup d'autres opérations ensemble.

Async

Désigne une tâche qui est configurée pour être exécutée en arrière-plan plutôt que d'attendre l'achèvement. Si vous avez un long processus qui durerait plus longtemps que le délai d'attente SSH, il serait logique de lancer cette tâche en mode asynchrone. Les modes Asynchrones peuvent être interrogés toutes les secondes ou peuvent être configurés pour "tirer et oublier", auquel cas Ansible ne vérifiera même pas à nouveau la tâche ; il la déclenchera simplement et passera aux étapes suivantes. Les modes asynchrones fonctionnent avec `/usr/bin/ansible` et `/usr/bin/ansible-playbook`.

Callback Plugin

Désigne un code écrit par l'utilisateur qui peut intercepter les résultats d'Ansible et en faire quelque chose. Quelques exemples fournis dans le projet GitHub permettent d'effectuer des enregistrements personnalisés, d'envoyer des courriels ou même de jouer des effets sonores.

Check Mode

Désigne l'exécution d'Ansible avec l'option `--check`, qui n'apporte aucune modification sur les systèmes distants, mais n'affiche que les modifications qui peuvent survenir si la commande s'exécute sans ce drapeau. Ceci est analogue aux modes dits "marche à vide" dans d'autres systèmes, mais l'utilisateur doit être averti que ceci ne tient pas compte des défaillances inattendues des commandes ou des effets en cascade (ce qui est vrai pour des modes similaires dans d'autres systèmes). Utilisez ceci pour vous faire une idée de ce qui pourrait arriver, mais ne le remplacez pas par un bon environnement de mise en scène.

Plugin de connexion

Par défaut, Ansible communique avec les machines distantes via des bibliothèques enfichables. Ansible supporte OpenSSH (SSH (Native)) ou une implémentation Python appelée paramiko. OpenSSH est préféré si vous utilisez une version récente, et permet également certaines fonctionnalités comme Kerberos et les hôtes de saut. Ceci est couvert dans la section Mise en route. Il existe aussi d'autres types de connexion comme le mode accéléré, qui doit être amorcé sur l'un des types de connexion basés sur SSH mais qui est très rapide, et le mode local, qui agit sur le système local. Les utilisateurs peuvent également écrire leurs propres plugins de connexion.

Conditionals

Un conditionnel est une expression qui évalue à vrai ou faux qui décide si une tâche donnée est exécutée sur une machine donnée ou non. Les conditionnels d'Ansible sont alimentés par l'instruction '`when:`'.

Déclaratif

Une approche pour réaliser une tâche qui utilise une description de l'état final plutôt qu'une description de la séquence des étapes nécessaires pour atteindre cet état. Pour un exemple du monde réel, une spécification déclarative d'une tâche serait : "mettez-moi en Californie". Selon votre emplacement actuel, la séquence des étapes pour vous rendre en Californie peut varier, et si vous êtes déjà en Californie, rien ne doit être fait du tout. Les Ressources d'Ansible sont déclaratives ; elles déterminent les étapes nécessaires pour atteindre l'état final. Il vous permet également de savoir si des mesures doivent être prises ou non pour arriver à l'état final.

Mode Diff

Un drapeau `--diff` peut être passé à Ansible pour montrer ce qui a changé sur les modules qui le supportent. Vous pouvez le combiner avec `--check` pour obtenir un bon "test". Les diffs de fichiers sont normalement dans un format de diff unifié.

Executor

Un composant logiciel de base d'Ansible qui est la puissance derrière `/usr/bin/ansible` directement - et correspond à l'invocation de chaque tâche dans un playbook. L'exécuteur est quelque chose dont les développeurs d'Ansible peuvent parler, mais ce n'est pas vraiment dans le vocabulaire de l'utilisateur commun.

Facts

Les "facts" sont simplement des choses que l'on découvre au sujet des nœuds distants. Bien qu'ils puissent être utilisés dans des livres de jeu et des modèles tout comme les variables, les "facts" sont des choses qui sont déduites, plutôt que définies. Les "facts" sont automatiquement découverts par Ansible lors de l'exécution d'une lecture de livre de jeu en exécutant le module de configuration interne sur les nœuds distants. Il n'est jamais nécessaire d'appeler le module "setup" explicitement, il s'exécute simplement, mais il peut être désactivé pour gagner du temps s'il n'est pas nécessaire ou vous pouvez dire à ansible de ne collecter qu'un sous-ensemble des faits complets via l'option `gather_subset:`. Pour la commodité des utilisateurs qui passent d'autres systèmes de gestion de configuration, le module factuel tirera également des faits des outils **ohai** et **facter** s'ils sont installés. Ce sont des bibliothèques de "facts" de Chef et Puppet, respectivement. (Ceux-ci peuvent aussi être désactivés via `gather_subset:`)

Plugin de filtrage

Un plugin de filtrage est quelque chose que la plupart des utilisateurs n'auront jamais besoin de comprendre. Ceux-ci permettent la création de nouveaux filtres Jinja2, qui ne sont plus ou moins utiles qu'aux personnes qui savent ce que sont les filtres Jinja2. Si vous en avez besoin, vous pouvez apprendre à les écrire dans la section documentation de l'API.

Forks

Il est possible de dialoguer en parallèle avec des nœuds distants et de définir le niveau de parallélisme soit en passant `--forks` à la commande ansible, soit en éditant la valeur par défaut dans un fichier de configuration. La valeur par défaut est une fourche à cinq (5) forks très conservatrice. Si vous avez beaucoup de RAM, vous pouvez facilement régler cette valeur à une valeur comme 50 pour un parallélisme accru.

Gather Facts (booléen)

Les "facts" ont été mentionnés ci-dessus. Parfois, lors de l'exécution d'un livre de jeu multi-jeux, il est souhaitable d'avoir quelques jeux qui ne collectent pas de "facts" car ils n'en n'ont pas besoin. Définir `gather_facts: False` dans un jeu permet de sauter cette collecte implicite de faits.

Globbing

Le "Globbing" est un moyen de sélectionner un grand nombre d'hôtes en se basant sur des caractères génériques, plutôt que de le faire sur le nom de l'hôte en particulier, ou le nom du groupe dans lequel ils se trouvent. Par exemple, il est possible de sélectionner `ww*` pour faire correspondre tous les hôtes en commençant par `www`. Ce concept est tiré directement de Func, l'un des projets précédents de Michael DeHaan (Ansible Founder).

Groupe d'inventaire

Un groupe d'inventaire se compose de plusieurs hôtes assignés à un pool qui peuvent être facilement ciblés ensemble, ainsi que de variables données qu'ils partagent en commun.

Group Vars

Les dossier `group_vars/` comprend des fichiers qui vivent dans un répertoire à côté d'un fichier d'inventaire, avec un nom de fichier facultatif nommé du nom de chaque groupe. C'est un endroit pratique pour placer les variables qui sont fournies à un groupe donné, en particulier les structures de données complexes, de sorte que ces variables n'ont pas à être intégrées dans le fichier d'inventaire ou dans le livre de jeu.

Handlers

Les "Handlers" sont comme les tâches régulières dans un playbook Ansible (voir Tâches) mais ne sont exécutés que si la tâche contient une directive `notify` et également si elle indique qu'elle a changé quelque chose. Par exemple, si un fichier de configuration est modifié, la tâche faisant référence à l'opération de modélisation du fichier de configuration peut avertir un gestionnaire ("handler") de redémarrage du service. Les gestionnaires peuvent être utilisés à d'autres fins que le redémarrage du service, mais le redémarrage du service est l'utilisation la plus courante.

Hôte (Host)

Un hôte est simplement une machine distante qu'Ansible gère. Des variables individuelles peuvent leur être affectées et elles peuvent également être organisées en groupes. Tous les hôtes ont un nom d'accès (adresse IP ou nom de domaine) et, facultativement, un numéro de port, s'ils ne sont pas accessibles sur le port SSH par défaut.

Host Specifier

Chaque jeu dans Ansible applique une série de tâches (qui définissent le rôle, le but ou les ordres d'un système) à un ensemble de systèmes. `hosts:` est la directive dans chaque jeu qui est souvent appelée le spécificateur hosts et qui configure ce comportement.

Il peut sélectionner un système, plusieurs systèmes, un ou plusieurs groupes, ou même certains hôtes qui sont dans un groupe et explicitement pas dans un autre.

Host Vars

Tout comme les "Group Vars", un répertoire à côté du fichier d'inventaire nommé `host_vars/` peut contenir un fichier nommé d'après chaque nom d'hôte du fichier d'inventaire, au format YAML. Ceci fournit un endroit pratique pour assigner des variables à l'hôte sans avoir à les intégrer dans le

fichier d’inventaire. Le fichier “Host Vars” peut également être utilisé pour définir des structures de données complexes qui ne peuvent pas être représentées dans le fichier d’inventaire.

Idempotence

Une opération est idempotente si le résultat de son exécution unique est exactement le même que le résultat de son exécution répétée sans aucune action intermédiaire.

Includes

L’idée que les fichiers de livre de jeu (qui ne sont rien de plus que des listes de jeux) peuvent inclure d’autres listes de jeux, et les listes de tâches peuvent être externalisées dans des listes de tâches contenues dans d’autres fichiers, et de même avec des “handlers”. Les “Includes” peuvent être paramétrés, ce qui signifie que le fichier chargé en mémoire peut passer des variables. Par exemple, un jeu inclus pour créer un blog WordPress peut prendre un paramètre appelé utilisateur et ce jeu peut être inclus plus d’une fois pour créer un blog pour alice et bob.

Inventaire

Un fichier (par défaut, Ansible utilise un format INI simple) qui décrit les hôtes et les groupes dans Ansible. L’inventaire peut également être fourni via un script d’inventaire (parfois appelé “External Inventory Script”).

Script d’inventaire

Un programme très simple (ou compliqué) qui recherche des hôtes, leur appartenance à un groupe d’hôtes et des informations variables à partir d’une ressource externe - que ce soit une base de données SQL, une solution CMDB, ou quelque chose comme LDAP. Ce concept a été adapté de Puppet (où il est appelé “External Nodes Classifier”) et fonctionne plus ou moins exactement de la même manière.

Jinja2

Jinja2 est le langage de template préféré du module de template d'Ansible. C'est un langage de template Python très simple, généralement lisible et facile à écrire.

JSON

Ansible utilise JSON pour les données de retour des modules distants. Cela permet aux modules d'être écrits dans n'importe quel langage, pas seulement en Python.

Lazy Evaluation

En général, Ansible évalue toutes les variables du contenu du livre de jeu à la dernière seconde, ce qui signifie que si vous définissez une structure de données, cette structure de données elle-même peut définir des valeurs de variables en son sein, et tout "fonctionne" comme vous pouvez vous y attendre. Cela signifie également que les chaînes de caractères variables peuvent inclure d'autres variables à l'intérieur de ces chaînes.

Library

Une collection de modules mis à disposition de `/usr/bin/ansible` ou d'un playbook Ansible.

Limit Groups

En passant `--limit somegroup` aux commandes `ansible` ou `ansible-playbook`, leur exécution peut être limitée à un sous-ensemble d'hôtes. Par exemple, ceci peut être utilisé pour exécuter un livre de jeu qui cible normalement un ensemble entier de serveurs vers un serveur particulier.

Local Action

Une directive `local_action` dans un livre de jeu ciblant des machines distantes signifie que l'étape donnée se produira effectivement sur la machine locale, mais que la variable {{ ansible_hostname }} peut être passée pour référencer le nom d'hôte distant auquel elle fait référence. Ceci peut être utilisé pour déclencher, par exemple, une opération `rsync` ou encore approvisionner des hôtes d'inventaire qui n'existent pas déjà auprès d'un fournisseur en nuage.

Local Connection

En utilisant la `connexion: local` dans un jeu, ou en passant `-c local` à la commande `/usr/bin/ansible`, cela indique que nous gérons l'hôte local et non une machine distante.

Plugin de recherche

Un plugin de recherche est un moyen d'obtenir des données dans Ansible à partir du monde extérieur. Les plugins de recherche sont une extension de Jinja2 et peuvent être appelés à partir des modèles, par exemple, `{{research('file','/path/to/file') }}`.

Loops

Ansible n'est pas un langage de programmation. Il préfère être plus déclaratif, bien que diverses constructions comme la boucle permettent de répéter une tâche particulière pour plusieurs éléments d'une liste. Certains modules, comme `yum` et `apt`, prennent en fait les listes directement, et peuvent installer tous les paquets donnés dans ces listes en une seule transaction, ce qui accélère considérablement le temps total de configuration, de sorte qu'ils peuvent être utilisés sans boucle.

Modules

Les modules sont les unités de travail qu'Ansible expédie aux machines distantes. Les modules sont lancés par `/usr/bin/ansible` ou `/usr/bin/ansible-playbook` (où plusieurs tâches utilisent beaucoup de modules différents en même temps). Les modules peuvent être implémentés dans n'importe quel langage, y compris Perl, Bash, ou Ruby - mais peuvent tirer parti d'un code de bibliothèque communautaire utile si il est écrit en Python. Les modules n'ont qu'à retourner JSON. Une fois que les modules sont exécutés sur des machines distantes, ils sont supprimés, de sorte qu'aucun démon de longue durée n'est utilisé. Ansible fait référence à la collection de modules disponibles en tant que bibliothèque.

Multi-Tier

Multi-Tier est le concept selon lequel les systèmes informatiques ne sont pas gérés un système à la fois, mais par des interactions entre plusieurs systèmes et des groupes de systèmes dans des

ordres bien définis. Par exemple, un serveur Web peut avoir besoin d'être mis à jour avant un serveur de base de données et des éléments sur le serveur Web peuvent avoir besoin d'être mis à jour après. Ansible modélise l'ensemble des topologies et des workflows informatiques plutôt que d'envisager la configuration dans une perspective "un système à la fois".

Notify

L'acte d'une tâche enregistrant un événement de changement et informant un gestionnaire ("handler") de tâche qu'une autre action doit être exécutée à la fin du jeu. Si un "handler" est notifié par plusieurs tâches, il ne sera exécuté qu'une seule fois. Les gestionnaires sont exécutés dans l'ordre dans lequel ils sont listés, et non dans l'ordre dans lequel ils sont notifiés.

Orchestration

De nombreux systèmes logiciels d'automatisation utilisent ce mot pour signifier différentes choses. Ansible l'utilise comme chef d'orchestre pour diriger un orchestre. L'architecture d'un centre de données ou d'un cloud est pleine de nombreux systèmes, jouant de nombreux rôles - serveurs web, serveurs de bases de données, peut-être des équilibreurs de charge, des systèmes de surveillance, systèmes d'intégration continue, etc. Dans l'exécution de tout processus, il est nécessaire de toucher les systèmes avec des commandes particulières, souvent pour simuler des mises-à-jour continues ou pour déployer le logiciel correctement. Certains systèmes peuvent effectuer certaines étapes, puis d'autres d'autres étapes, puis les systèmes précédents déjà traités peuvent avoir besoin d'effectuer d'autres étapes. En cours de route, il peut être nécessaire d'envoyer des courriels ou de communiquer avec des services Web. Une orchestration possible consiste à modéliser ce genre de processus.

Paramiko

Par défaut, Ansible gère les machines via SSH. La bibliothèque qu'Ansible utilise par défaut pour ce faire est une bibliothèque alimentée par Python appelée paramiko. La librairie paramiko est généralement rapide et facile à gérer, bien que les utilisateurs désirant la prise en charge de Kerberos ou de Jump Host puissent passer à un binaire SSH natif tel que OpenSSH en spécifiant le type de connexion dans leurs playbooks, ou en utilisant le drapeau `-c ssh`.

Livres de jeux

Les Livres de jeux (playbooks) sont la langue par laquelle Ansible orchestre, configure, administre ou déploie les systèmes. On les appelle en partie des playbooks parce que c'est une analogie sportive, et c'est censé être amusant de les utiliser. Ce ne sont pas des cahiers d'exercices :)

Jeux (Plays)

Un livre de jeu est une liste de jeux. Un jeu est au minimum une correspondance entre un ensemble d'hôtes sélectionnés par un spécificateur d'hôte (généralement choisis par des groupes mais parfois par des globes de nom d'hôte) et les tâches qui s'exécutent sur ces hôtes pour définir le rôle que ces systèmes vont jouer. Il peut y avoir un ou plusieurs jeu dans un livre de jeu.

Mode Pull (traction)

Par défaut, Ansible fonctionne en mode "push", ce qui lui permet de contrôler très finement le moment pendant lequel il parle à chaque système. Le mode Pull est prévu lorsque vous préférez que les nœuds vérifient un livre de jeu toutes les N minutes sur un horaire particulier. Il utilise un programme appelé ansible-pull et peut également être configuré (ou reconfiguré) à l'aide d'un playbook en mode "push". La plupart des utilisateurs d'Ansible utilisent le mode "push", mais le mode "pull" est inclus pour la variété et le fait d'avoir des alternatives.

ansible-pull fonctionne en vérifiant les commandes de configuration git avec un crontab puis en gérant la machine localement, en utilisant le plugin de connexion local.

Mode Push

Le mode "Push" est le mode par défaut d'Ansible. En fait, ce n'est pas vraiment un mode du tout - c'est juste la manière dont Ansible fonctionne quand on n'y pense pas. Le mode "push" permet à Ansible d'être finement granulé et de conduire les nœuds à travers des processus d'orchestration complexes sans attendre qu'ils s'enregistrent.

Register Variable

Le résultat de l'exécution d'une tâche dans Ansible peut être stocké dans une variable pour être utilisé dans un modèle ou une instruction conditionnelle. Le mot-clé utilisé pour définir la variable est appelé `register`, prenant son nom de l'idée de registres dans la programmation en Assembleur. Il existe un nombre infini de noms de variables que vous pouvez utiliser pour l'enregistrement.

Modèle de ressources

Les modules possibles fonctionnent en termes de ressources. Par exemple, le module `file` sélectionnera un fichier particulier et s'assurera que les attributs de cette ressource correspondent à un modèle particulier. Par exemple, nous pourrions souhaiter changer le propriétaire de `/etc/motd` à `root` s'il n'est pas déjà défini à `root`, ou définir son `mode` à `0644` s'il n'est pas déjà défini à cette valeur. Les modèles de ressources sont idempotents, ce qui signifie que les commandes de changement ne sont exécutées que si nécessaire, et Ansible ramènera le système à l'état désiré quel que soit l'état réel - plutôt que vous ayez à lui dire comment atteindre cet état.

Rôles

Les rôles sont des unités d'organisation dans Ansible. Attribuer un rôle à un groupe d'hôtes (ou un ensemble de groupes, ou des modèles d'hôtes, etc.) implique qu'ils doivent mettre en œuvre un comportement spécifique. Un rôle peut inclure l'application de certaines valeurs de variables, de certaines tâches et de certains gestionnaires ("handlers") - ou simplement une ou plusieurs de ces choses. En raison de la structure de fichier associée à un rôle, les rôles deviennent des unités redistribuables qui vous permettent de partager leur comportement entre les playbooks - ou même avec d'autres utilisateurs.

Mise à jour continue (Rolling Updates)

Le fait d'adresser un certain nombre de nœuds d'un groupe N à la fois pour éviter de les mettre à jour tous en même temps et de mettre le système hors ligne. Par exemple, dans une topologie Web de 500 nœuds traitant un très grand volume, il peut être raisonnable de mettre à jour 10 ou 20 machines à la fois, puis de passer aux 10 ou 20 suivantes une fois terminé. `serial` : le mot-clé dans un Ansible playbooks contrôle la taille du pool de mise à jour mobile. La valeur par défaut est d'adresser la taille du lot d'un seul coup, c'est donc quelque chose que vous devez choisir. La configuration du système d'exploitation (comme s'assurer que les fichiers de configuration sont corrects) n'a généralement pas besoin d'utiliser le modèle de mise à jour continue, mais peut le faire si nécessaire.

Become

Ansible ne nécessite pas de connexion root, et puisqu'il est sans démon, il n'a certainement pas besoin de démons de niveau root (ce qui peut être un problème de sécurité dans les environnements sensibles). Ansible peut se connecter et effectuer de nombreuses opérations enveloppées dans une commande sudo, et peut travailler à la fois avec sudo sans mot de passe et avec sudo avec mot de passe. Certaines opérations qui ne fonctionnent pas normalement avec sudo (comme le transfert de fichiers scp) peuvent être réalisées avec les modules `copy`, `template` et `fetch` d'Ansible en mode sudo.

SSH (Native)

Native OpenSSH en tant que transport Ansible est spécifié avec `-c ssh` (ou dans un fichier de configuration, ou avec une directive dans le playbook) et peut être utile si vous voulez vous connecter via "Kerberized SSH" ou en utilisant les hôtes "SSH jump", etc.

Tags

Ansible permet de baliser les ressources d'un livre de jeu avec des mots-clés arbitraires, puis d'exécuter uniquement les parties du livre de jeu qui correspondent à ces mots-clés. Par exemple, il est possible d'avoir une configuration d'OS complète, et d'avoir certaines étapes appelées ntp, puis d'exécuter uniquement les étapes ntp pour reconfigurer les informations du serveur de temps sur un hôte distant.

Tâche

Les playbooks existent pour exécuter des tâches. Les tâches combinent une action (un module et ses arguments) avec un nom et éventuellement d'autres mots-clés (comme des directives de bouclage). Les gestionnaires ("handlers") sont également des tâches, mais il s'agit d'un type particulier de tâches qui ne s'exécutent que si elles sont notifiées par leur nom lorsqu'une tâche signale un changement sous-jacent sur un système distant.

Modèles Jinja2

Ansible peut facilement transférer des fichiers vers des systèmes distants mais il est souvent souhaitable de substituer des variables dans d'autres fichiers. Les variables peuvent provenir du fichier d'inventaire, des variables d'hôte, des variables de groupe ou des faits. Les modèles utilisent le moteur de modèles Jinja2 et peuvent aussi inclure des constructions logiques comme

des boucles et des instructions if.

Transport

Ansible utilise la terme “Connection Plugins” pour définir les types de transports disponibles. C’est simplement de cette façon qu’Ansible s’adressera aux systèmes gérés. Les transports inclus sont paramiko, ssh (utilisant OpenSSH), et local.

When

Une instruction conditionnelle facultative attachée à une tâche qui est utilisée pour déterminer si la tâche doit être exécutée ou non. Si l’expression suivant le mot-clé `when:` évalue à faux, la tâche sera ignorée.

Vars (Variables)

Contrairement aux “facts”, les variables sont des noms de valeurs (il peut s’agir de simples valeurs scalaires - entiers, booléens, chaînes de caractères) ou des valeurs complexes (dictionnaires/matières, listes) qui peuvent être utilisées dans des modèles et des livres de jeu. Ce sont des choses déclarées, et non des choses qui sont déduites de l’état ou de la nature actuelle du système distant (ce qui est ce que sont les faits).

YAML

Ansible ne veut pas forcer les gens à écrire du code en langage de programmation pour automatiser l’infrastructure, donc Ansible utilise YAML pour définir les langages de configuration des livres de jeu et aussi les fichiers variables. YAML est agréable parce qu’il a un minimum de syntaxe et qu’il est très propre et facile à lire pour les personnes. C’est un bon format de données pour les fichiers de configuration et les humains, mais aussi pour la lecture automatique. L’utilisation de YAML par Ansible provient de la première utilisation de YAML par Michael DeHaan à l’intérieur de Cobbler vers 2006. YAML est assez populaire dans la communauté des langages dynamiques et le format dispose de bibliothèques disponibles pour la sérialisation dans de nombreux langages de programmation (Python, Perl, Ruby, etc.).

Revision #1
Created 3 October 2021 21:20:01 by garfi
Updated 3 October 2021 21:20:29 by garfi