Automatisation avec Ansible

Installer Ansible

Automne 2024

Table des matières

1 Installer Ansible

Dans le premier module, on a identifié la façon de s'approprier l'installation d'un wiki et comment aborder l'automatisation de son déploiement : en décomposant les étapes **d'architecture**, **d'installation** et de **configuration**. En finissant par faire le choix d'un outil d'automatisation adapté à vos besoins : **Ansible**.

Dans ce module, on va faire connaissance avec l'outil de gestion **Ansible,** on va comprendre comment déployer automatiquement MediaWiki sur 2 serveurs avec Ansible, et on va apprendre à installer Ansible dans un environnement de travail virtuel.

1.1 Découvrir Ansible

Ansible est un outil d'automatisation informatique écrit en Python. Il peut **configurer** des systèmes, **déployer** des logiciels et **orchestrer des tâches** informatiques avancées, telles que des déploiements continus.

Son créateur s'appelle **Michael DeHaan**; la première version de Ansible date de **2012**. Depuis, Ansible s'enrichit constamment et une version majeure est proposée approximativement tous les deux mois.

Le nom Ansible est tiré d'un roman de science-fiction écrit par Ursula Le Guin, et qui désigne un moyen de communication plus rapide que la lumière.

Entretemps, Ansible a été racheté en **2015** par **Red Hat**; la communauté compte plus de **3500 contributeurs**.

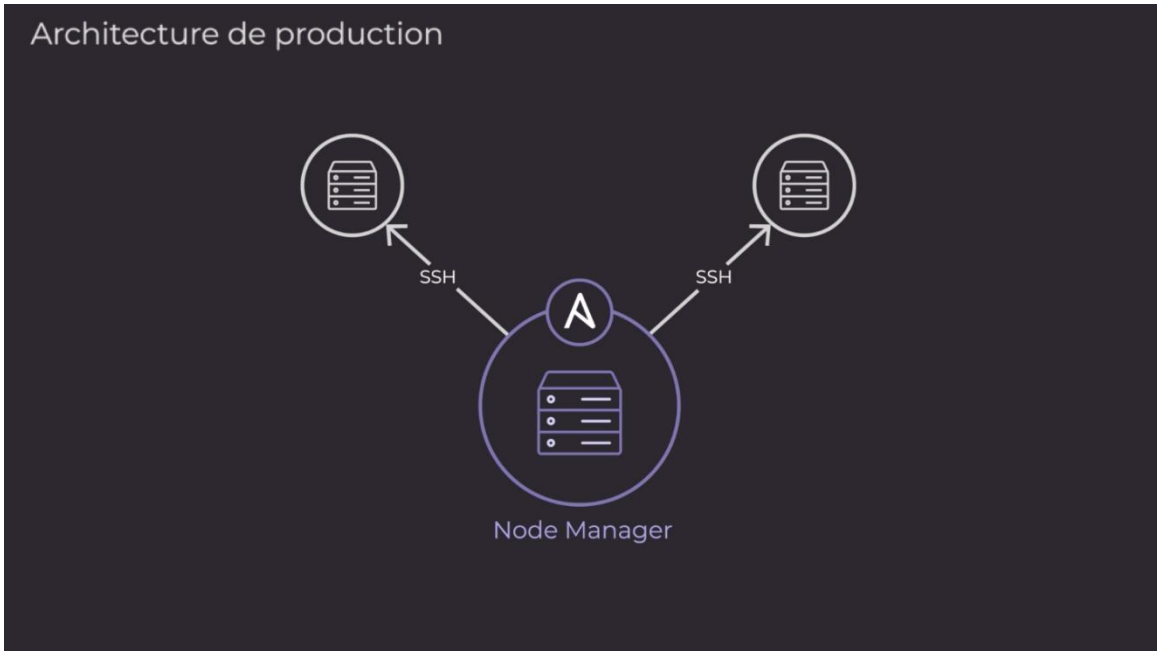
Red Hat a été racheté par **IBM** en **2018**. Donc, Ansible appartient désormais à IBM.

1.1.1 Compléter votre architecture technique pour Ansible

Votre architecture technique est pour le moment constituée de 2 serveurs. On va ajouter un nouveau serveur qui deviendra votre **tour de contrôle Ansible**.

Votre architecture ressemble maintenant au schéma suivant :

- deux serveurs (Apache et MySQL); dans le jargon Ansible, ces serveurs sont appelés des ***nodes***;
- un serveur de contrôle, appelé ***node manager***. C'est le serveur sur lequel seront installés les outils **Ansible** et depuis lequel, les opérations de configuration seront lancées à distance sur les nodes :



Node

Un **node** (ou ***managed node***, ou ***host***) est un poste connecté au node manager en SSH, et sur lequel Ansible viendra pousser les tâches d'automatisation. Ansible n'est pas installé sur les nodes.

Node manager

Un **node manager**, ou ***control node***, est un poste qui contrôle les nodes grâce à sa connexion SSH. Il a Ansible d'installé pour leur pousser les tâches d'automatisation grâce aux commandes ansible et ansible-playbook. Ça peut être n'importe quelle machine Linux, mais pas Windows.

Ansible est un outil **agentless**, c'est-à-dire qu'il n'installe pas d'agent sur les nodes. Il travaille donc en mode **push** : il *pousse* les installations sur les nodes. Pour cela, il n'utilise que les outils déjà présents sur la plupart des systèmes Linux : SSH et Python.

L'inverse du mode **push** est le mode **pull**. Par exemple, une marketplace d'applications comme le Play Store ou l'AppStore d'Apple sont des systèmes en mode **pull**: le client (le smartphone) *tire* les applications ou les mises à jour vers lui.

Il est conseillé d'avoir un serveur de référence sur lequel on configure un environnement d'automatisation toujours opérationnel et sécurisé, capable de se connecter aux nodes de notre infrastructure.

1.1.2 Gérer les configurations avec le Node Manager

Préparer les trois machines virtuelles suivantes :

Machine virtuelle	Système d'exploitation	Réseau
node manager	Debian	NAT
serveur 1	AlmaLinux	NAT
serveur 2	AlmaLinux	NAT

Le **node manager** va être notre **tour de contrôle**. On va y installer Ansible et tous ses outils, pour déployer automatiquement MediaWiki sur les nodes.

Le **node manager** et les **nodes** peuvent avoir des **systèmes d'exploitation différents**. Il n'y a pas de corrélation entre le système du node manager et le système des nodes.

Sur le **node manager**, on trouve les **outils Ansible** et les **scripts d'automatisation**. Tous les scripts seront lancés depuis le node manager. Ce qui aura pour effet d'exécuter des opérations de configuration à distance sur les nodes.

1.2 Installer Ansible

1.2.1 Installer Ansible sur le node manager

On va installer Ansible sur le node manager. Il y a plusieurs façons d'installer Ansible :

- via les **packages logiciels** sur un système Linux ;
- via **pip** de **Python** dans un **virtualenv** ou pas ;
- via les **sources officielles** (Archives ou Git) maintenues par Red Hat.

Ansible peut être installé avec ces trois méthodes nativement sur des systèmes Linux.

Pour installer Ansible sur Windows, il faudra passer par un émulateur Unix de type Cygwin.

1.2.2 Installer les prérequis

L'installation de Ansible sur le node manager se fera avec la méthode **pip** de **Python** dans un **virtualenv**.

Un **virtualenv** est un **outil Python** qui permet de créer des environnements de travail virtuels isolés. virtualenv crée un dossier qui contient les fichiers exécutables **Python**, et une copie de la bibliothèque **pip**.

pip *est un système de gestion de paquets utilisé pour installer et gérer les paquets logiciels écrits en* ***Python****.*

Avec cette méthode, on crée un **environnement de travail virtuel cloisonné**, dans lequel on peut installer la version Ansible de notre choix (basée sur le release repository).

À chaque sortie d'une **nouvelle version de Ansible**, il y a des nouveautés et des dépréciations; il est donc important de tester la compatibilité de vos scripts avant de mettre à jour les outils Ansible. Avec **virtualenv** c'est très pratique de tester la compatibilité.

Sur Debian, le paquet disponible pour installer Ansible n'est pas la toute dernière version (Debian privilégie la sécurité et la stabilité plutôt que la nouveauté). C'est parfois un inconvénient quand on attendait une fonctionnalité tout juste sortie mais non disponible sur Debian. La méthode pip permet de s'en affranchir et de disposer de la dernière version de Ansible.

Se connecter sur le **node manager** en **root** et installer le paquet **python-virtualenv**, ce qui permettra de créer des environnements de travail virtuel : **virtualenv**.

```
root@debian:~# apt install python3-virtualenv
```

Installer aussi le paquet **sshpass** qui servira ultérieurement pour se connecter en **SSH**.

```
root@debian:~# apt install  sshpass
```

1.2.3 Créer un simple utilisateur

Pour ne pas travailler en **root** sur le **node manage**r (ce n'est vraiment pas recommandé, le compte root peut tout faire sans aucune limite), on va créer un simple utilisateur **user-ansible** :

```
root@debian:~# adduser user-ansible
Adding user `user-ansible' ...
Adding new group `user-ansible' (1001) ...
Adding new user `user-ansible' (1001) with group `user-ansible' ...
Creating home directory `/home/user-ansible' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for user-ansible
Enter the new value, or press ENTER for the default
        Full Name []: ansible
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n]
Adding new user `user-ansible' to supplemental / extra groups `users' ...
Adding user `user-ansible' to group `users' ...
```

Entrer un mot de passe et compléter les informations demandées.

Maintenant que l'utilisateur est créé, on peut l'utiliser avec la commande suivante :

```
root@debian:~# su - user-ansible
user-ansible@debian:~$
```

On travaille maintenant avec l'utilisateur **user-ansible** sur le **node-manager**.

1.3 Créer un environnement de travail virtuel

Comme indiqué plus haut, On utilisera un **environnement de travail virtuel** pour cloisonner l'installation et l'exécution d'Ansible. Ceci permettra de gérer les dépendances avec la version de Python et d'installer une version particulière de Ansible.

On va installer la dernière version disponible en date du 17 octobre2024 : **version 11.0.0a2**.

On crée alors un environnement de travail virtuel nommé **ansible11.0.0a2**. Le nom est arbitraire. On peut mettre ce qu'on veut.

Sur le **node manager**, lancer la commande suivante :

```
user-ansible@debian:~$ virtualenv ansible11.0.0a2
created virtual environment CPython3.11.2.final.0-64 in 187ms
  creator CPython3Posix(dest=/home/user-ansible/ansible11.0.0a2, clear=False,
no_vcs_ignore=False, global=False)
  seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle,
via=copy, app_data_dir=/home/user-ansible/.local/share/virtualenv)
    added seed packages: pip==23.0.1, setuptools==66.1.1, wheel==0.38.4
  activators
BashActivator,CShellActivator,FishActivator,NushellActivator,PowerShellActivator,Pyt
honActivator
```

On utilise la commande **virtualenv** pour créer l'environnement **ansible11.0.0a2**, dans lequel les outils, les ressources et le gestionnaire de paquets sont installés.

On va aussi créer un deuxième environnement pour la version 10.5.0 :

```
user-ansible@debian:~$ virtualenv ansible10.5.0
created virtual environment CPython3.11.2.final.0-64 in 214ms
  creator CPython3Posix(dest=/home/user-ansible/ansible10.5.0, clear=False,
no_vcs_ignore=False, global=False)
  seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle,
via=copy, app_data_dir=/home/user-ansible/.local/share/virtualenv)
    added seed packages: pip==23.0.1, setuptools==66.1.1, wheel==0.38.4
  activators
BashActivator,CShellActivator,FishActivator,NushellActivator,PowerShellActivator,Pyt
honActivator
```

Pour afficher les environnements virtuels :

```
user-ansible@debian:~$ ls -l

total 8
drwxr-xr-x 4 user-ansible user-ansible 4096 Oct 16 23:23 ansible10.5.0
drwxr-xr-x 4 user-ansible user-ansible 4096 Oct 16 23:16 ansible11.0.0a2
```

Pour **activer** l'environnement virtuel de la version **11.0.0a2** :

```
user-ansible@debian:~$ source ansible11.0.0a2/bin/activate
(ansible11.0.0a2) user-ansible@debian:~$
```

Le prompt a changé, ce qui signifie qu'on est dans l'environnement de travail virtuel ansible11.0.0a2

On peut changer d'environnement en **activant** l'environnement virtuel de la version 10.0.5 :

```
user-ansible@debian:~$ source ansible10.5.0/bin/activate
(ansible10.5.0) user-ansible@debian:~$
```

1.4 Installer Ansible dans votre environnement virtuel

Installer **Ansible** dans l'environnement virtuel **11.0.0a2** avec **pip**

Il faut activer en premier l'environnement virtuel 11.0.0a2:

```
user-ansible@debian:~$ source ansible11.0.0a2/bin/activate
(ansible11.0.0a2) user-ansible@debian:~$
```

Ensuite, lancer l'installation de Ansible version **11.0.0a2** dans l'environnement virtuel **11.0.0a2**:

```
(ansible11.0.0a2) user-ansible@debian:~$ pip install ansible==11.0.0a2
Collecting ansible==11.0.0a2
  Downloading ansible-11.0.0a2-py3-none-any.whl (48.7 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 48.7/48.7 MB 8.9 MB/s eta 0:00:00
Collecting ansible-core~=2.18.0rc1
  Downloading ansible_core-2.18.0rc1-py3-none-any.whl (2.2 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.2/2.2 MB 11.6 MB/s eta 0:00:00
Collecting jinja2>=3.0.0
  Downloading jinja2-3.1.4-py3-none-any.whl (133 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 133.3/133.3 kB 6.9 MB/s eta 0:00:00
Collecting PyYAML>=5.1
  Downloading PyYAML-6.0.2-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_6           4.whl (762 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 763.0/763.0 kB 12.9 MB/s eta 0:00:00
Collecting cryptography
  Downloading cryptography-43.0.1-cp39-abi3-manylinux_2_28_x86_64.whl (4.0 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.0/4.0 MB 11.1 MB/s eta 0:00:00
Collecting packaging
  Downloading packaging-24.1-py3-none-any.whl (53 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 54.0/54.0 kB 8.3 MB/s eta 0:00:00
Collecting resolvelib<1.1.0,>=0.5.3
  Downloading resolvelib-1.0.1-py2.py3-none-any.whl (17 kB)
Collecting MarkupSafe>=2.0
  Downloading MarkupSafe-3.0.1-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x        86_64.whl (23 kB)
Collecting cffi>=1.12
  Downloading cffi-1.17.1-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64            .whl (467 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 467.2/467.2 kB 10.8 MB/s eta 0:00:00
Collecting pycparser
  Downloading pycparser-2.22-py3-none-any.whl (117 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 117.6/117.6 kB 12.6 MB/s eta 0:00:00
Installing collected packages: resolvelib, PyYAML, pycparser, packaging, MarkupS              afe,
jinja2, cffi, cryptography, ansible-core, ansible
Successfully installed MarkupSafe-3.0.1 PyYAML-6.0.2 ansible-11.0.0a2 ansible-core-2.18.0rc1 cffi-1.17.1
cryptography-43.0.1 jinja2-3.1.4 packaging-24.1 pycparser-2.22 resolvelib-1.0.1
```

Finalement, Vérifier la **version de Ansible** :

```
(ansible11.0.0a2) user-ansible@debian:~$ ansible --version
ansible [core 2.18.0rc1]
  config file = None
  configured module search path = ['/home/user-ansible/.ansible/plugins/modules',
'/usr/share/ansible/plugins/modules']
  ansible python module location = /home/user-ansible/ansible11.0.0a2/lib/python3.11/site-
packages/ansible
  ansible collection location = /home/user-
ansible/.ansible/collections:/usr/share/ansible/collections
  executable location = /home/user-ansible/ansible11.0.0a2/bin/ansible
  python version = 3.11.2 (main, Aug 26 2024, 07:20:54) [GCC 12.2.0] (/home/user-
ansible/ansible11.0.0a2/bin/python)
  jinja version = 3.1.4
  libyaml = True
```

Installer **Ansible version 10.5.0** dans l'environnement virtuel **ansible10.5.0** avec **pip**

Il faut activer en premier l'environnement virtuel 10.5.0 :

```
user-ansible@debian:~$ source ansible10.5.0/bin/activate
(ansible10.5.0) user-ansible@debian:~$
```

Installer Ansible version **10.0.5** dans l'environnement virtuel **10.0.5**:

```
(ansible10.5.0) user-ansible@debian:~$ pip install ansible==10.5.0
Collecting ansible==10.5.0
  Downloading ansible-10.5.0-py3-none-any.whl (49.0 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.0/49.0 MB 5.6 MB/s eta 0:00:00
Collecting ansible-core~=2.17.5
  Downloading ansible_core-2.17.5-py3-none-any.whl (2.2 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.2/2.2 MB 12.3 MB/s eta 0:00:00
Collecting jinja2>=3.0.0
  Using cached jinja2-3.1.4-py3-none-any.whl (133 kB)
Collecting PyYAML>=5.1
  Using cached PyYAML-6.0.2-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (762
kB)
Collecting cryptography
  Using cached cryptography-43.0.1-cp39-abi3-manylinux_2_28_x86_64.whl (4.0 MB)
Collecting packaging
  Using cached packaging-24.1-py3-none-any.whl (53 kB)
Collecting resolvelib<1.1.0,>=0.5.3
  Using cached resolvelib-1.0.1-py2.py3-none-any.whl (17 kB)
Collecting MarkupSafe>=2.0
  Using cached MarkupSafe-3.0.1-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
(23 kB)
Collecting cffi>=1.12
  Using cached cffi-1.17.1-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (467
kB)
Collecting pycparser
  Using cached pycparser-2.22-py3-none-any.whl (117 kB)
Installing collected packages: resolvelib, PyYAML, pycparser, packaging, MarkupSafe, jinja2,
cffi, cryptography, ansible-core, ansible
Successfully installed MarkupSafe-3.0.1 PyYAML-6.0.2 ansible-10.5.0 ansible-core-2.17.5 cffi-
1.17.1 cryptography-43.0.1 jinja2-3.1.4 packaging-24.1 pycparser-2.22 resolvelib-1.0.1
```

Vérifier la **version de Ansible :**

```
(ansible10.5.0) user-ansible@debian:~$ ansible --version
ansible [core 2.17.5]
  config file = None
  configured module search path = ['/home/user-ansible/.ansible/plugins/modules',
'/usr/share/ansible/plugins/modules']
  ansible python module location = /home/user-ansible/ansible10.5.0/lib/python3.11/site-
packages/ansible
  ansible collection location = /home/user-
ansible/.ansible/collections:/usr/share/ansible/collections
  executable location = /home/user-ansible/ansible10.5.0/bin/ansible
  python version = 3.11.2 (main, Aug 26 2024, 07:20:54) [GCC 12.2.0] (/home/user-
ansible/ansible10.5.0/bin/python)
  jinja version = 3.1.4
  libyaml = True
```

1.5 Vérifier votre installation

Dans le répertoire **bin** de votre environnement virtuel, on peut constater que plusieurs commandes Ansible sont installées :

```
(ansible10.5.0) user-ansible@debian:~$ ls ansible10.5.0/bin/ansible*

ansible10.5.0/bin/ansible                ansible10.5.0/bin/ansible-galaxy
ansible10.5.0/bin/ansible-community      ansible10.5.0/bin/ansible-inventory
ansible10.5.0/bin/ansible-config         ansible10.5.0/bin/ansible-playbook
ansible10.5.0/bin/ansible-connection     ansible10.5.0/bin/ansible-pull
ansible10.5.0/bin/ansible-console        ansible10.5.0/bin/ansible-test
ansible10.5.0/bin/ansible-doc            ansible10.5.0/bin/ansible-vault
```

Regardons de plus près quelques outils :

ansible	Cette commande permet de lancer des actions Ansible en mode ad-hoc (en ligne de commande) ;
ansible-config	Cette commande permet de gérer la configuration de Ansible. Si On lance la commande **$ ansible-config list**, on va lister la configuration de Ansible. Toutes ces variables sont contenues dans **./lib/python2.7/site-packages/ansible/constants.py**;
ansible-doc	Cette commande permet d'obtenir de l'aide pour utiliser Ansible. La documentation est très bien faite, c'est plutôt pratique pour se guider quand on commence, surtout que on peut y trouver des exemples concrets.