Ansible

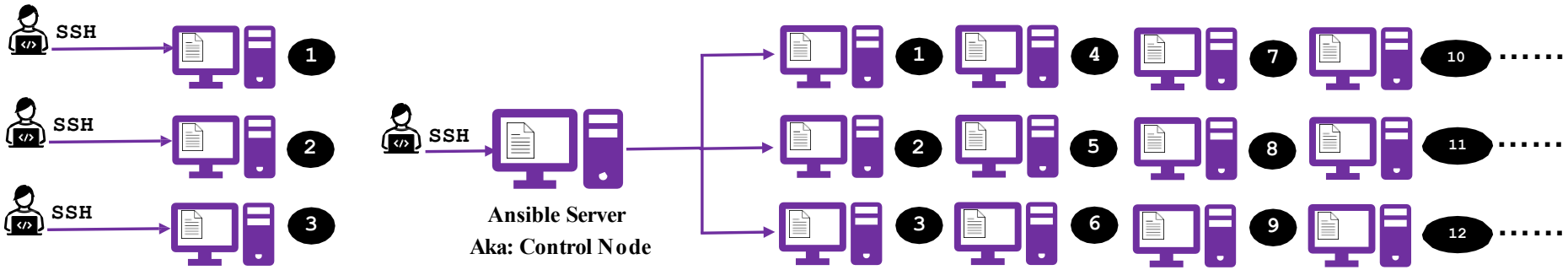
Sommaire

Ansible

Introduction

Qu'est-ce qu'Ansible ?

- Ansible est un outil open source de provisionnement de logiciels, de gestion de configuration et de déploiement d'applications permettant l'infrastructure en tant que code. Il fonctionne sur de nombreux systèmes de type Unix, et peut configurer à la fois des systèmes de type Unix ainsi que Microsoft Windows (Wikipedia).
-
- En termes simples: Ansible est un outil d'automatisation gratuit qui peut automatiser les tâches informatiques sur la machine locale où il est en cours d'exécution et sur les machines distantes



- **<u>Remarque: Ansible est écrit en langage python, mais cela ne signifie pas que vous avez besoin de connaissances python pour utiliser Ansible</u>**

Qu’est-ce qu’Ansible ?

- Ansible peut être utilisé pour:
-

Système d’approvisionnement

Configurer le système

Déployer des applications

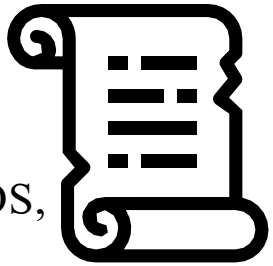
Gérer le système et les applications

Qu'est-ce qu'Ansible ?

- **Example of Tasks**
  - Provisioning
    - Sert du métal nu
    - Systèmes de virtualisation
    - Périphériques réseau
    - Systèmes de stockage
    - Plateforme cloud
  - System Configuration Management
    - Mises à jour ou mises à niveau
    - Installation du package
    - Configuration du service
    - Arrêt| Démarrer| Redémarrage des services
    - Utilisateur ou groupes
    - Attribution d'autorisations aux fichiers et répertoires
  - Déploiement d'applications
  - Sauvegardes
  - Redémarrages hebdomadaires ou mensuels du système
  - Orchestration.

Brève histoire d'Ansible

- Le projet Ansible a été lancé en 2012 par Michael DeHaan
- Il est open source et axé sur la communauté
- Ansible Inc a été racheté par Red Hat en 2015
- Il est disponible pour la plupart des distributions Linux telles que, Red Hat, CentOS, Fedora, Ubuntu, Debian et SUSE
-



- Besoin?
  - Gestion de l'infrastructure (en particulier la virtualisation)
  - Gestion de la configuration (configuration du système ou de l'application)
  - Automatisation des applications multiniveaux (par exemple.app, serveurs Web et de base de données)
  - Point unique d'automatisation (avoir des scripts sur chaque système plutôt qu'un seul) plate-forme de gestion de l'automatisation).

Avantages d'Ansible

- Sans agent!!!
- L'open-source
- Évitez les erreurs humaines
- Gain de temps en automatisant les tâches répétitives ou fastidieuses
- Augmenter la productivité
- Facile à utiliser
- Simple (fichiers texte lisibles par l'homme)
- Souple
- Sécurisé (sur SSH).

Avantages d'Ansible

- Fournit des modules pré-écrits
- Facile à apprendre (tant que vous avez un bon instructeur) ☺
- Excellent produit pour l'orchestration
- Ansible peut être utilisé non seulement pour les systèmes, mais aussi pour le réseau, le stockage, le cloud, etc.
- Fournit environ 1300 modules prêts à l'emploi et environ 4000 modules sur galaxy
- Énormes ressources Ansible en ligne
  -
  - www.ansible.com
  - www.docs.ansible.com
  - www.galaxy.ansible.com
  - www.github.com

- **Un gros plus + pour les demandeurs d'emploi et ceux qui veulent améliorer leur carrière**

Terminologies Ansible

- **Nœud de contrôle ou Ansible Server**
  - Serveur qui exécute l'application Ansible

- **Modules**
  - Module est une commande destinée à être exécutée côté client
  - La plupart des modules de tâches informatiques sont déjà créés et peuvent être trouvés sur le site Web d'Ansible
    - www.docs.ansible.com → Rechercher l'index des modules
    - www.galaxy.ansible.com
  - Exemple de modules :
  -
    1. Install http
    2. Enable http service
    3. Start http service

    } Task

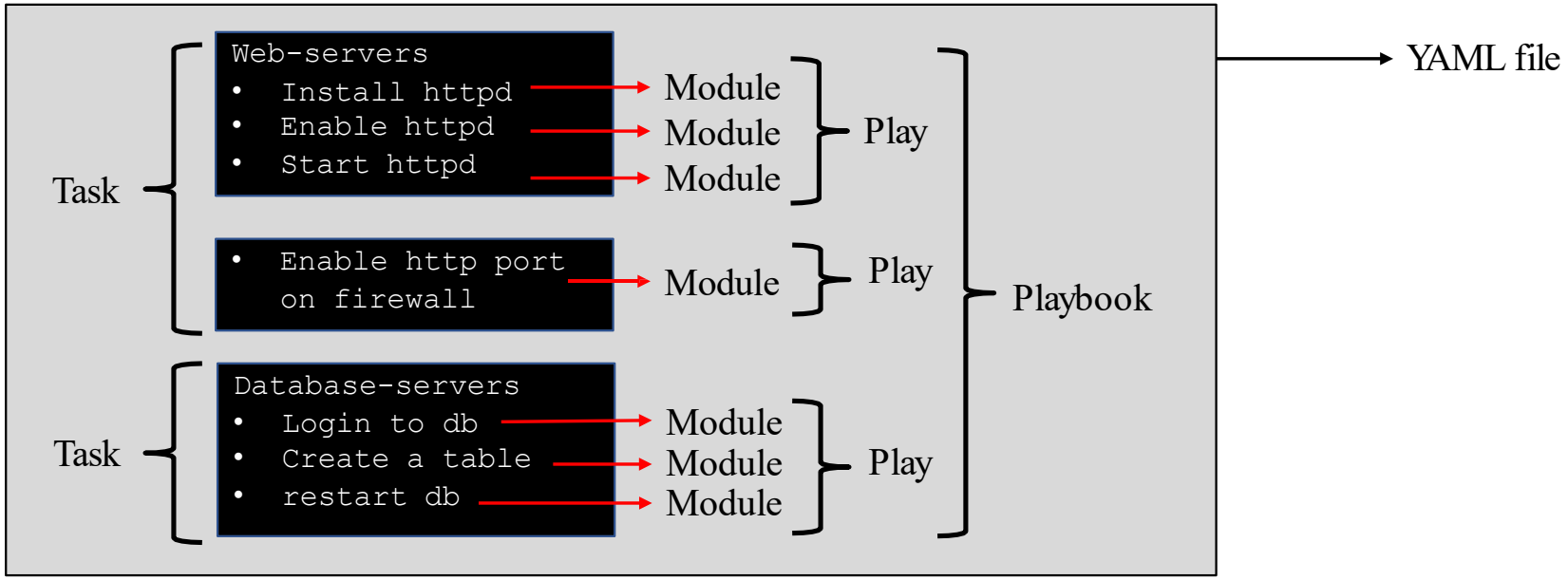
- **Task**
  - Une tâche est une section qui consiste en une seule procédure à effectuer. Une tâche peut avoir plusieurs modules

Terminologies en Ansible

- **Playbook**
  - Fichier d'automatisation avec exécution étape par étape de plusieurs tâches
  -
- **YAML**
  - Un Playbook écrit en langage YAML (Encore un autre langage de balisage)
  -
- **Inventory**
  - Fichier contenant des informations sur les clients distants sur lesquels les tâches sont exécutées
  -
- **Tag**
  - Référence ou alias à une tâche spécifique
  -
- **Variable**
  - Les variables sont comme des conteneurs qui contiennent la valeur définie qui peut être utilisée de manière répétitive
  -
- **Role**
  - Fractionnement du Playbook en petits groupes. Les rôles vous permettent de charger automatiquement des vars, fichiers, tâches, gestionnaires et autres artefacts Ansible associés en fonction d'une structure de fichiers connue. Après avoir regroupé votre contenu en rôles, vous pouvez facilement le réutiliser et le partager avec d'autres utilisateurs.

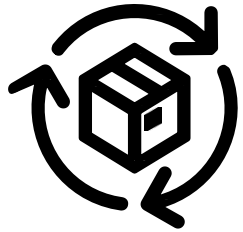
Comment fonctionne Ansible ?

- Chaque tâche spécifique dans Ansible est écrite via un ou plusieurs modules
- Plusieurs modules sont écrits dans l'ordre séquentiel
- Plusieurs modules pour les tâches connexes est appelé un jeu
- Tous les jeux ensemble fait un Playbook
- **Playbook est écrit comme un format de fichier appelé YAML**



Comment fonctionne Ansible?

Exemples de commandes :



To run modules through yaml file:

```
# ansible-playbook example.yml
```

Pour exécuter le module indépendamment

```
# ansible myservers –m ping
```

Fichiers de configuration Ansible:

- `/etc/ansible/ansible.cfg`
- `/etc/ansible/hosts`
- `/etc/ansible/roles`

Autres outils d'automatisation

- **Puppet and Chef**
  - Utilise le langage Ruby qui est plus difficile à apprendre, et leur soutien diminue de jour en jour
  - Ces outils nécessitent l'installation d'agents sur les clients
  - Le processus d'installation est très complexe
  - Manque de documentation
  -

- **Ansible**
  - Utilise YAML simple
  - Sans agent (nécessite uniquement un accès SSH)
  - Installation facile
  - Produit bien documenté

Ansible Open Source contre Red Hat Ansible

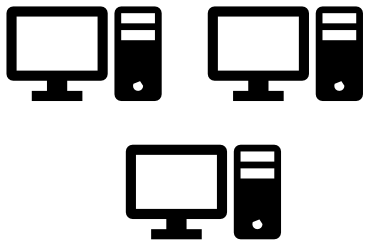
- Ansible est un logiciel open-source
- Il a été acheté par Red Hat en 2015
- Le logiciel Ansible lui-même est gratuit même s'il appartient à Red Hat
- Ansible est le même logiciel sur toutes les plateformes
- La seule différence est que Red Hat fournit des produits supplémentaires Ansible Tower et Conseil ou support technique pour Ansible
- En savoir plus sur Red Hat Ansible sur www.redhat.com

- Red Hat Ansible Tower
  - Red Hat fournit la tour Ansible, un outil basé sur une interface graphique pour gérer l'automatisation Ansible
  - Ansible tower est un produit payant de Red Hat
  - Gère plusieurs serveurs Ansible pour les grandes entreprises

- Ansible AWX
  - Open source
  - Free software.

Ansible

Installation



Installation d'Ansible

IMPORTANT : Prendre un instantané de machine virtuelle

•Le guide d'installation d'Ansible est disponible sur :
https://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html

- **CentOS/Red Hat/Fedora 7**
  - **yum/dnf install epel-release**
  - **yum/dnf install ansible**

- **For version 8:**
  - **yum install epel-release**
  - **yum install python -y** *(should be installed already)*
  - ***yum install ansible ansible-doc***

- **CentOS 8** *(Dans le cas où les étapes ci-dessus ne fonctionnent pas)*
  Habituellement, verion8 viendra avec Python3 déjà installé par défaut. Cependant, si pour une raison quelconque Python3 n'est pas installé, installez-le à l'aide des commandes suivantes
  - **yum/dnf install python3**

- Installation de PIP - Le programme d'installation du package Python
  - **yum/dnf install python3-pip**

- Installer Ansible en tant qu'utilisateur régulier
- **pip3 install ansible ansible-doc**

Installation d'Ansible

- Red Hat 8 (permet au référentiel Ansible Engine d'installer Ansible)
  - `subscription-manager repos --enable ansible-2.8-for-rhel-8-x86_64-rpms`
  - `yum/dnf install ansible`

- Vérifiez la version d'Ansible et exécutez le module ping sans Playbook pour vérifier la stature Ansible
  - `# ansible --version`
  - `# ansible localhost -m ping`

- Ansible config files
  - `/etc/ansible` → **Répertoire par défaut**
  - `/etc/ansible/ansible.cfg`
  - `/etc/ansible/hosts`
  - `/etc/ansible/roles`

À noter :
Si SELinux est activé sur des nœuds distants, vous devez également installer libselinux-python sur eux avant d'utiliser les fonctions liées à la copie/fichier/modèle dans Ansible. Vous pouvez utiliser le module yum ou le module dnf dans Ansible pour installer ce package sur un système distant

Ansible

Ansible avec des playbooks

Syntaxe du fichier YAML

Choses IMPORTANTES à retenir!

- ✓ Toutes les tâches sont exécutées dans l'ordre séquentiel
- ✓ Chaque tâche est traitée une à la fois
- ✓ L'indentation est extrêmement importante
- ✓ Non <tabs> dans le fichier yaml
- ✓ N'utilisez que des espaces
- ✓ Les lignes vides n'ont aucune valeur
- ✓ L'extension de fichier est généralement .yml ou .yaml

Syntaxe du fichier YAML

Choses IMPORTANTES à retenir!

- ✓ Aucune différence entre guillemets doubles ou AUCUN guillemet pour un nom de tâche :
- ✓ Les fichiers YAML Playbook peuvent être placés n'importe où sur le système de fichiers tant qu'ils sont exécutés avec un chemin absolu
- ✓ Lorsqu'un fichier plat est écrit au format YAML pour exécuter des tâches / lectures, il est appelé playbook
- ✓ Il n'est PAS nécessaire de modifier l'autorisation de fichier
  `rwx--x--x`

Exemple de syntaxe de fichier YAML

```
---
  - name: sampleplaybook
    hosts: all or localhost
    become: yes
    become_user: root

    tasks:
    - name: Install Apache httpd
      yum:
      name: httpd
      state: present

    - name: 2nd task
      service:
      name: httpd  state:
      started
```

Nom du playbook

Où courir?

Exécuter en tant qu'utilisateur différent ?

Définir le nom de la tâche

Exécuter le module de tâche yum

Nom du package

Que faire ? --Installer

Déclarer une tâche

- Modules et options Ansible
- `https://docs.ansible.com/ansible/2.5/modules/`

Création du premier Playbook

- Il existe des outils yaml en ligne que vous pouvez utiliser pour créer des Playbooks
- https://onlineyamltools.com/edit-yaml
  - https://codebeautify.org/yaml-editor-online

- Outils à télécharger
  - Notepad++ Windows

- Prendre un instantané après l'installation du logiciel Ansible

```
# su - root
# mkdir /etc/ansible/playbooks
# cd /etc/ansible/playbooks
# vim first.yml

---
- name: "My first playbook"
  hosts: localhost

  tasks:
  - name: "test connectivity"
    ping:
```

- Vérifier la syntaxe du playbook

```
# anisble-playbook --syntax-check first.yml
```

Ou pour faire un essai

```
# anisble-playbook --check first.yml
```

- Exécuter le playbook

```
# anisble-playbook /root/ansible/first.yml
```

Création du premier Playbook

À noter :

- Executer ansible sans playbook

```
# ansible
```

- Exécuter ansible avec un playbook

```
# ansible-playbook
```

Playbook de sortie

Ce playbook imprimera « Hello World » sur localhost

```
# cd /etc/ansible/playbook # vim
helloworld.yml

---
- name: My Second playbook
  hosts: localhost

  tasks:
  - name: Print Hello World
    debug: msg="Hello World"
```

Nom de la pièce ou du playbook

Exécuter sur localhost

Exécutez la tâche suivante

Nom de la tâche

Exécuter le module de débogage qui imprime les instructions pendant l'exécution

Run the playbook

```
# anisble-playbook helloworld.yml
```

Playbook sur plusieurs tâches

Le playbook va envoyer un ping à localhost et imprimer « Hello World »

```
# vim mtask.yml
    ---
    - name: Running 2 tasks
      hosts: localhost

      tasks:
      - name: Test connectivity
        ping:

      - name: Print Hello World
        debug: msg="Hello World"
```

Nom de la pièce

Exécutez-le sur l'hôte local

Exécuter la tâche suivante Nom de la tâche

Exécuter le module ping

Nom de la 2e tâche Exécuter le module de débogage

Run the playbook

```
# anisble-playbook mtask.yml
```

Installation et démarrage d'un package

```
# vim packinstall.yml

---
- name: Installing and Running apache
  hosts: localhost

  tasks:
  - name: Install apache
    yum:
     name: httpd
     state: present

  - name: start httpd
    service:
     name: httpd
     state: started
```

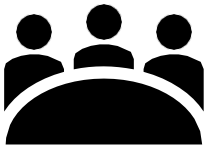
Run the playbook

```
# anisble-playbook packinstall.yml
```

Ansible

Autres fonctionnalités Ansible

Roles



- Rôles simplifie les longs playbooks en regroupant les tâches dans des playbooks plus petits

OU

- Les rôles sont le moyen de diviser un playbook en plusieurs fichiers playbook. Cela simplifie l'écriture de playbooks complexes et les rend plus faciles à réutiliser.

- L'écriture de code ansible pour gérer le même service pour plusieurs environnements crée plus de complexité et il devient difficile de tout gérer dans un seul playbook ansible. De plus, le partage de code entre d'autres équipes devient difficile. C'est là qu'Ansible Role aide à résoudre ces problèmes
-
- Les rôles sont comme des modèles qui sont la plupart du temps statiques et peuvent être appelés par les playbooks

- Les rôles permettent de regrouper l'ensemble de la configuration dans:
  - Tasks
  - Modules
  - Variables
  - Handlers

Roles

Example

Liste des tâches pour east-webservers

```
---
- name: Configurer le serveur web
  hosts: east-webservers
  tasks:
1 - name: Install httpd packages
    yum:
     name: httpd
     state: present

2 - name: Start httpd
    service:
     name: httpd
     state: started

3 - name: Open port http on firewall
    firewalld:
     service: http
     permanent: true
     state: enabled

4 - name: Restart firewalld
    service:
     name: firewalld
     state: reloaded
```

Task list for west-webservers

```
---
- name: Liste des tâches pour
  hosts: west-webservers
  tasks:
1 - name: Install httpd packages
    yum:
     name: httpd
     state: present

2 - name: Start httpd
    service:
     name: httpd
     state: started
```

```
# vim byrole.yml
```

Tâches combinées

```
---
- name: Setup httpd webserver
  hosts: east-webservers
  tasks:
1 - name: Install httpd packages
    yum:
     name: httpd
     state: present

2 - name: Start httpd
    service:
     name: httpd
     state: started

3 - name: Open port http on firewall
    firewalld:
     service: http
     permanent: true
     state: enabled

4 - name: Restart firewalld
    service:
     name: firewalld
     state: restarted

  hosts: west-webservers
  tasks:
5 - name: Install httpd packages
    yum:
     name: httpd
     state: present

6 - name: Start httpd
    service:
     name: httpd
     state: started
```

Roles

```
---
- name: Setup httpd webserver
  hosts: east-webservers
  tasks:
  - name: Install htppd packages
    yum:
      name: httpd
      state: present

  - name: Start httpd
    service:
      name: httpd
      state: started

  - name: Open port http on firewall
    firewalld:
      service: http
      permanent: true
      state: enabled

  - name: Restart firewalld
    service:
      name: firewalld
      state: restarted

  hosts: west-webservers
  tasks:
  - name: Install httpd packages
    yum:
      name: httpd
      state: present

  - name: Start httpd
    service:
      name: httpd
      state: started
```

1 2 3 4

5 6

```
---
- name: Setup full httpd webserver
  tasks:
  - name: Install httpd packages
    yum:
      name: httpd
      state: present

  - name: Start httpd
    service:

  - r                    t 80 for http access
    f          alld:
      service: http
      permanent: true
      state: enabled

  - name: Restart firewalld
    service:
      name: firewalld
      state: restarted
```

Roles Playbook: fullinstall

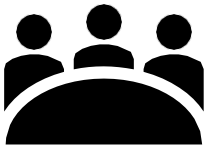
```
---
- name: Setup basic httpd webserv
  tasks:
  - name: Install ap
    yum:
      na

  - n                  pd
    se
      name: httpd
      state: started
```

Roles Playbook: basicinstall

```
---
- name: Full install
  hosts: east-webservers
  roles:
  - fullinstall

- name: Basic install
  hosts: west-webservers
  roles:
  - basicinstall
```

Version très simplifiée du playbook

Roles



- Veuillez noter que les rôles peuvent être regroupés par type de serveurs, type d'applications ou exigences organisationnelles.

- Pour créer des rôles

```
# Go to ControlNode
# cd /etc/ansible/roles
```

Créer un répertoire pour chaque rôle

```
# e.g mkdir [rolenames]  #
mkdir basicinstall
# mkdir fullinstall
```

Vous pouvez également créer des rôles en fonction du type de serveurs:

```
e.g. # mkdir webservers
# mkdir dbservers # mkdir
appservers
```

Créer des tâches de sous-répertoire dans chaque répertoire

```
# mkdir basicinstall/tasks  # mkdir
fullinstall/tasks
```

Créer des fichiers yml dans ces sous-répertoires

```
# touch basicinstall/tasks/main.yml  # touch
fullinstall/tasks/main.yml
```

Roles

 # vim fullinstall/tasks/main.yml

```
---
- name: Install httpd package
  yum:
   name: httpd
   state: present

- name: Start httpd
  service:
   name: httpd
   state: started

- name: Open port for http
  firewalld:
   service: http
   permanent: true
   state: enabled

- name: Restart firewalld
  service:
   name: firewalld
   state: reloaded
```

 # vim basicinstall/tasks/main.yml

```
---
- name: Install httpd package
  yum:
   name: httpd
   state: present

  - name: Start httpd
    service:
     name: httpd
     state: started
```

 # vim /etc/ansible/playbooks/byrole.yml

```
---
- name: Full install
  hosts: all
  roles:
  - fullinstall

- name: Basic install
  hosts: localhosts
  roles:
  - basicinstall
```

Rôles par application

```
---
- name: Install packages
  hosts: all
  tasks:
  - name: Install Apache package
    yum:
     name: httpd
     state: present

  - name: Install Time package
    yum:
     name: ntpd or chrony
     state: present

  - name: Install DNS package
     yum:
     name: named
     state: present
```

1

2

3

- Pour créer des rôles

```
# Go to ControlNode
# cd /etc/ansible/roles
```

Créer un répertoire pour chaque rôle

```
# mkdir apache
# mkdir ntpd
# mkdir named
```

Créer des tâches de sous-répertoire dans chaque répertoire

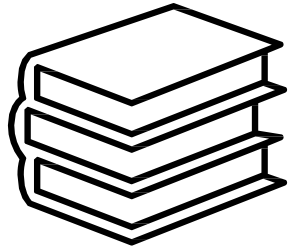
```
# mkdir apache/tasks
# mkdir ntpd/tasks
# mkdir named/tasks
```

Créer des fichiers yml dans ces sous-répertoires

```
# touch apache/tasks/main.yml
# touch ntpd/tasks/main.yml
# touch named/tasks/main.yml
```

```
---
- name: Install packages
  hosts: all
  roles:
  - apache
  - ntpd
  - named
```

Rôles sur Ansible Galaxy



- Vous pouvez trouver une tonne de ressources sur les rôles à travers la galaxie Ansible
- Vous pouvez télécharger des rôles prédéfinis ou pré-écrits à partir de la galaxie Ansible
- www.galaxy.ansible.com

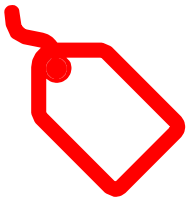
Tags

- Les balises sont la référence ou les alias d'une tâche
- Au lieu d'exécuter un playbook Ansible entier, utilisez des balises pour cibler une tâche spécifique que vous devez exécuter

```
# vim httpbytags.yml
---
- name: Setup Apache server
  hosts: localhost
  tasks:
  - name: Install httpd
    yum:
     name: httpd
     state: present
    tags: i-httpd

  - name: Start httpd
    service:
     name: httpd
     state: started
    tags: s-httpd
```

```
# anisble-playbook httpbytags.yml -t i-httpd
```

```
# anisble-playbook httpbytags.yml -t s-httpd
```



- Pour répertorier toutes les balises d'un playbook
  `# anisble-playbook httpbytags.yml --list-tags`
- Pour exécuter une tâche à l'aide d'une balise
  `# anisble-playbook httpbytags.yml -t i-httpd`
- Pour ignorer une tâche à l'aide d'une balise
  `# anisble-playbook httpbytags.yml --skip-tags i-httpd`
- Nous pouvons utiliser « l'option de tâches » pour démarrer un playbook à une tâche spécifique

```
# anisble-playbook yamlfile.yml --start-at-task 'Task name'
# anisble-playbook http.yml --start-at-task 'Intall httpd'
```

Variables

- Les variables sont comme des conteneurs qui contiennent la valeur définie qui peut être utilisée de manière répétitive

Choses IMPORTANTES à retenir sur les variables!

- Le nom peut inclure des lettres, des chiffres et un trait de soulignement
- Le nom doit toujours commencer par une lettre
- Ne peut pas avoir d'espaces, de points (.) ou de stylo (-) dans le nom de variable
- Les variables peuvent également être définies à l'intérieur des fichiers d'inventaire

Variables

Example

```
---
- name: Install some package
  hosts: all
  vars:
   sespackage: sesquipedalianism

  tasks:
  - name: Package install
    yum:
     name: "{{ sespackage }}"
     state: present

  - name: Start service
    service:
     name: "{{ sespackage }}"
     state: started
```

1

```
---
- name: Package installation
  hosts: all
  vars:
   pack: httpd

  tasks:
  - name: Install package
    yum:
     name: "{{ pack }}"
     state: present

  - name: Start service
    service:
     name: "{{ pack }}"
     state: started
```

Variables

Examples

2

```
---
- name: Copy file to remote clients
  hosts: all
  vars:
   srcfile: /home/iafzal/somefile
  tasks:

  - name: Copying file
    become:  true
    copy:
     src: "{{ srcfile }}"
     dest: /tmp
     owner: iafzal
     group: iafzal
     mode: 0644
```

3

```
---
- name: Create a file
  hosts: localhost
  vars:
   file_name: kramer

  tasks:
  - name: Create file in /tmp
    file:
     state: touch
     path: /tmp"{{ file_name }}".txt
```

4

```
---
 - name: Print Hello world
   hosts: all
   vars:
     say: Hello World!

   tasks:
   - name: Ansible Variable Basic Usage
       debug:
       msg: "{{ say }}"
```

Variables dans le fichier d'inventaire

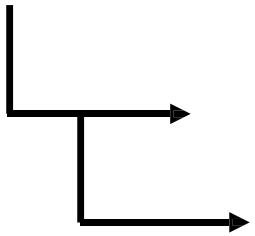
Example

```
[webservers]
client1.xyz.com
client2.xyz.com

[abc:vars]
fooserver=foo.abc.example.com
ntpserver=ntp.abc.example.com
proxyserver=proxy.abc.example.com


server1 ansible_host=201.0.113.111
server2 ansible_host=201.0.113.112
server3 ansible_host=201.0.113.113
server4 ansible_host=abc.example.com
```

Ansible

Autres fonctionnalités Ansible

Handlers



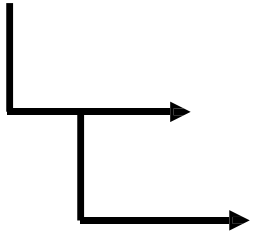
- Handlers sont exécutés à la fin, une fois toutes les tâches terminées. Dans Ansible, les Handlers sont généralement utilisés pour démarrer, recharger, redémarrer et arrêter les services
-
- Parfois, vous souhaitez exécuter une tâche uniquement lorsqu'une modification est apportée à une machine. Par exemple, vous souhaiterez peut-être redémarrer un service si une tâche met à jour la configuration de ce service, mais pas si la configuration est inchangée.
-
- Rappelez-vous le cas où nous avons dû recharger le firewalld parce que nous voulions activer le service http? Oui, c'est un exemple parfait d'utilisation des handlers

- Donc, fondamentalement, les handlers sont des tâches qui ne s'exécutent que lorsqu'elles sont notifiées
-
- Chaque Handler doit avoir un nom global unique
-

Handlers

Example

```
---
- name: Verify apache installation
  hosts: localhost
  tasks:
  - name: Ensure apache is at the latest version
    yum:
     name: httpd
     state: latest

  - name: Copy updated apache config file
    copy:
     src: /tmp/httpd.conf
     dest: /etc/httpd.conf
    notify:
    - Restart apache

  - name: Ensure apache is running
    service:
     name: httpd
     state: started

  handlers:
    - name: Restart apache
      service:
       name: httpd
       state: restarted
```

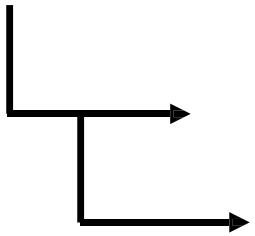


Le service httpd sera redémarré à la fin

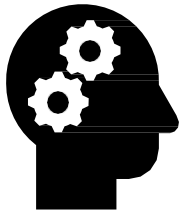
Exemple de pare-feu

Handlers

```
---
- name: Enable service on firewalld
  hosts: localhost
  tasks:

  - name: Open port for http
    firewalld:
     service: http
    permanent: true
    state: enabled
    notify:
    - Reload firewalld

  -   name: Ensure firewalld is
      running service:
       name: firewalld
       state: started

  handlers:
    - name: Reload firewalld
      service:
       name: firewalld
       state: reloaded
```



Conditions



- L'exécution de conditions permet à Ansible de prendre des mesures par lui-même en fonction de certaines conditions
- Sous condition, certaines valeurs doivent être respectées avant l'exécution d'une tâche
- Nous pouvons utiliser l'instruction WHEN pour rendre l'automatisation Ansible plus intelligente

```
- name: Playbook description
  hosts: localhost

  tasks:
  - name: Start a service
    when: A == "B"
    service:
     name: servicename
     state: started
```

Conditions

```
# vim httpbycondition.yml

---
- name: Install Apache WebServer
  hosts: localhost

  tasks:
    - name: Install Apache on Ubuntu Server
      apt-get:
        name: apache2
        state: present
      when: ansible_os_family == "Ubuntu"

    - name: Install Apache on CentOS   Server
      yum:
        name: httpd
        state: present
      when: ansible_os_family == "RedHat"
```

Variable intégrée Ansible

Comment obtenir une liste de toutes les variables intégrées Ansible

Les variables sont recueillies à partir de faits

Rassembler la liste des faits d'un hôte

```
# ansible localhost -m setup
```

Loops

- Une boucle est un outil de programmation puissant qui vous permet d'exécuter un ensemble de commandes à plusieurs reprises.

- Nous pouvons automatiser une tâche spécifique, mais que se passe-t-il si cette tâche elle-même est répétitive?
  - Par exemple, modification des autorisations sur des centaines de fichiers
  - Création de plusieurs utilisateurs à la fois
  - Installation de nombreux paquets sur des centaines de serveurs

- Les boucles peuvent fonctionner main dans la main avec les conditions lorsque nous bouclons certaines tâches jusqu'à ce que cette condition soit remplie

- Lors de la création de boucles, Ansible fournit ces deux directives : loop et with_* mot-clé.

Loops

Exemple d'utilisateurs

- Pour créer plusieurs utilisateurs en ligne de commande Linux, nous utilisons « for loop »
- e.g.

```
# for u in jerry kramer eliane; do useradd $u; done
```

```
vim userloop.yml

---
- name: Create users
  hosts: localhost

  tasks:
1 - name: Create jerry
    user:
      name: jerry

2 - name: Create kramer
    user:
      name: kramer

3 - name: Create eliane
    user:
      name: eliane
```

1 Ajout d'un paramètre de boucle

```
vim userbyloop1.yml

---
- name: Create users thru loop
  hosts: localhost

  tasks:
  - name: Create users
    user:
      name: "{{ item }}""
    loop:
      - jerry
      - kramer
      - eliane
```

2 Ajout d'une variable

```
vim userbyloop2.yml

---
- name: Create users thru loop
  hosts: localhost
  vars:
   users: [jerry,kramer,eliane]

  tasks:
  - name: Create users
    user:
      name: '{{item}}'
    with_items: '{{users}}'
```

Exemple de packages

Loops

- Pour installer plusieurs paquets en ligne de commande Linux, nous utilisons « for loop »
- e.g.

```
# for p in ftp telnet htop; do yum install $p -y; done
```

❶ Ajout de variables et appel de variables via un paramètre d'élément

```
vim installbyloop1.yml

---
- name: Install packages thru loop
  hosts: localhost
  vars:
   packages: [ftp,telnet,htop]

  tasks:
  - name: Install package
    yum
      name: '{{items}}'
      state: present
    with_items: '{{packages}}'
```

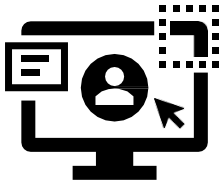
❷ Ajout de variables et appel direct de variables

```
vim installbyloop2.yml

---
- name: Install packages thru loop
  hosts: localhost
  vars:
   packages: [ftp,telnet,htop]

  tasks:
  - name: Install packages
    yum
      name: '{{packages}}'
      state: present
```

Ansible

Ansible
Management Tools

Ansible Management Tools



- L’outil de gestion Ansible fournit une interface web centralisée pour gérer vos playbooks, les inventaires et tout ce que vous faites via une ligne de commande

- Ces outils gèrent également plusieurs nœuds de contrôle Ansible

- Il existe 2 outils de gestion Ansible :
  - Ansible AWX (gratuit)
  - Ansible Tower (produit sous licence)

Ansible AWX

- AWX signifie Ansible WorkX
- AWX est une application Web qui fournit une interface utilisateur, une API REST et un moteur de tâches pour Ansible
- L'AWX vous permet de gérer les playbooks Ansible, les inventaires, d'exécuter des rapports et de planifier des tâches
- Ansible AWX est un outil de gestion open source piloté par la communauté
- C'est le produit en amont, ce qui signifie que toutes les modifications sont effectuées sur AWX avant d'arriver à Ansible Tower.
- Le projet AWX est hébergé sur GitHub et Red Hat accueille les contributions de la communauté.
- Le projet AWX utilise également GitHub pour le suivi des problèmes. Vous pouvez déposer vos problèmes ici:
  - https://github.com/ansible/awx/issues
- Red Hat ne recommande PAS AWX pour les environnements de production.

Ansible AWX

Ansible AWX Avantages et inconvénients

- **Avantages**
  - Caractéristiques d'entreprise complètes et fonctionnalités de Tower
  - Ceci est disponible pour téléchargement gratuit et utilisation
  - Non limité le nombre de nœuds à ajouter
  - Idéal pour les environnements POC/dev/lab ou QA

- **Inconvénients :**
  - Pas de support technique par Red Hat
  - Plusieurs versions en une seule journée sont possibles
  - Red Hat ne recommande pas son utilisation dans des environnements de production.

Ansible AWX

Installation Ansible AWX (Remarque : AWX est déployé à partir des conteneurs docker)

- Mettre à jour le système
  ```
  # yum update -y
  ```
- Redémarrer
  ```
  # reboot
  ```
- Installer le référentiel epel-release
  ```
  # yum/dnf install epel-release
  ```
- Installer des packages supplémentaires
  ```
  # dnf install git gcc gcc-c++ ansible nodejs gettext device-mapper-persistent-data
  lvm2 bzip2 python3-pip
  ```
- Ajouter un référentiel docker
  ```
  # dnf config-manager --add-repo=https://download.docker.com/linux/centos/docker-ce.repo
  ```
- Installer Docker
  ```
  # dnf install docker-ce --nobest -y
  # dnf install docker-ce --nobest --allowerasing
  ```
- Vérifier la version de Docker
  ```
  # docker --version
  ```

Ansible AWX

- Démarrer docker
  `# systemctl start docker`

- Activer docker au démarrage
  `# systemctl enable docker`

- Installer docker-compose
  `# pip3 install docker-compose`

- Vérifier la version docker-compose
  `# docker-compose --version`

- Définir la commande python pour utiliser python3
  `# alternatives --set python /usr/bin/python3`

- Téléchargez la dernière version d'Ansible AWX à partir du référentiel Git Hub
- `# git clone https://github.com/ansible/awx.git`

- Générer une clé secrète pour le chiffrement du fichier d'inventaire et copier la sortie dans un fichier texte
- `# openssl rand -base64 30`

Ansible AWX

- Changer de répertoire et modifier le fichier d'inventaire

```
# awx/installer
OU # awx/tools/docker-compose
# vim inventory
```

- Ajouter ou modifier les paramètres suivants

```
[all:vars]
dockerhub_base=ansible
awx_task_hostname=awx
awx_web_hostname=awxweb
postgres_data_dir="/var/lib/pgdocker"
host_port=80
host_port_ssl=443
docker_compose_dir="/root/awx/tools/docker-compose"
pg_username=awx
pg_password=awxpass
pg_database=awx
pg_port=5432
pg_admin_password=password
rabbitmq_password=awxpass
rabbitmq_erlang_cookie=cookiemonster
admin_user=admin
admin_password=password
create_preload_data=True
secret_key=R+kbcDEUS8DlAftAbfWafVqLZ0lUy+Paqo4fEtgp
awx_official=true
awx_alternate_dns_servers="8.8.8.8,8.8.4.4"
project_data_dir=/var/lib/awx/projects
```

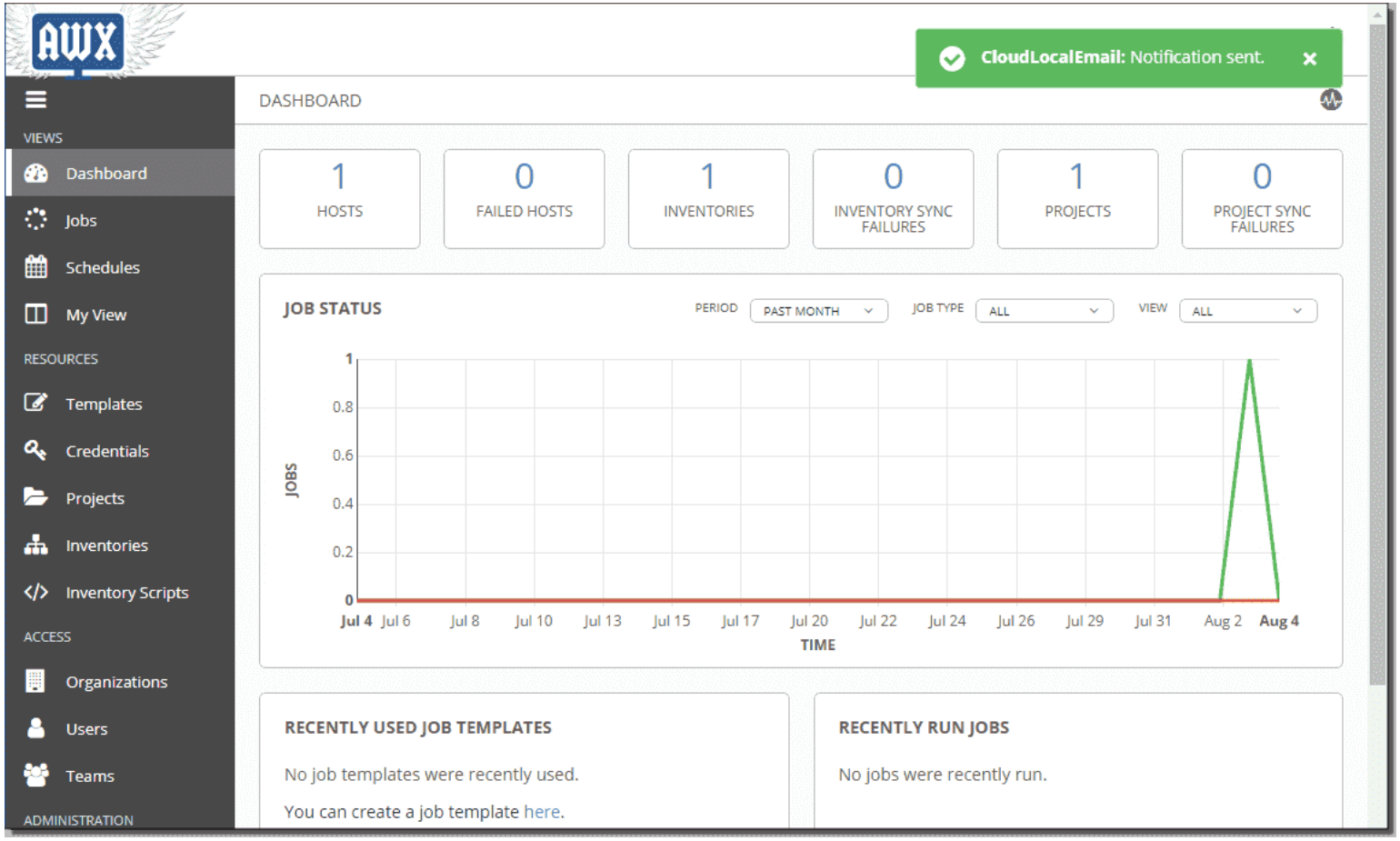
Ansible AWX

- Créer un répertoire pour postgress
  ```
  # mkdir /var/lib/pgdocker
  ```

- Installer AWX
  ```
  # ansible-playbook -i inventory install.yml
  ```

- Ouvrez Firefox sur votre Linux
  ```
  # http://localhost
  ```

Ansible AWX


AWX
CloudLocalEmail: Notification sent.
DASHBOARD
VIEWS
Dashboard
Jobs
Schedules
My View
RESOURCES
Templates
Credentials
Projects
Inventories
Inventory Scripts
ACCESS
Organizations
Users
Teams
ADMINISTRATION
1 HOSTS
0 FAILED HOSTS
1 INVENTORIES
0 INVENTORY SYNC FAILURES
1 PROJECTS
0 PROJECT SYNC FAILURES
JOB STATUS
PERIOD PAST MONTH
JOB TYPE ALL
VIEW ALL
JOBS
1
0.8
0.6
0.4
0.2
0
Jul 4 Jul 6 Jul 8 Jul 10 Jul 13 Jul 15 Jul 17 Jul 20 Jul 22 Jul 24 Jul 26 Jul 29 Jul 31 Aug 2 Aug 4
TIME
RECENTLY USED JOB TEMPLATES
No job templates were recently used.
You can create a job template here.
RECENTLY RUN JOBS
No jobs were recently run.


AWX
Ansible

Ansible Tower

- Ansible Tower est un produit de gestion Ansible commercial de Red Hat qui est également fourni avec le support de Red Hat.

- Red Hat Ansible Tower est disponible en deux éditions qui se différencient par leur support et leurs fonctionnalités. La tarification est basée sur le nombre de nœuds (systèmes, hôtes, instances, machines virtuelles, conteneurs ou périphériques) que vous gérez

    1. Standard
    2. Premium

    https://www.ansible.com/products/pricing

Ansible Tower

Ansible Tower Avantages et inconvénients:

- **Avantages :**
  - 24hrs full technical support
  - Release cycles
  - Fully tested for quality and engineering issues before release
  - Install/upgrades are well-documented and supported

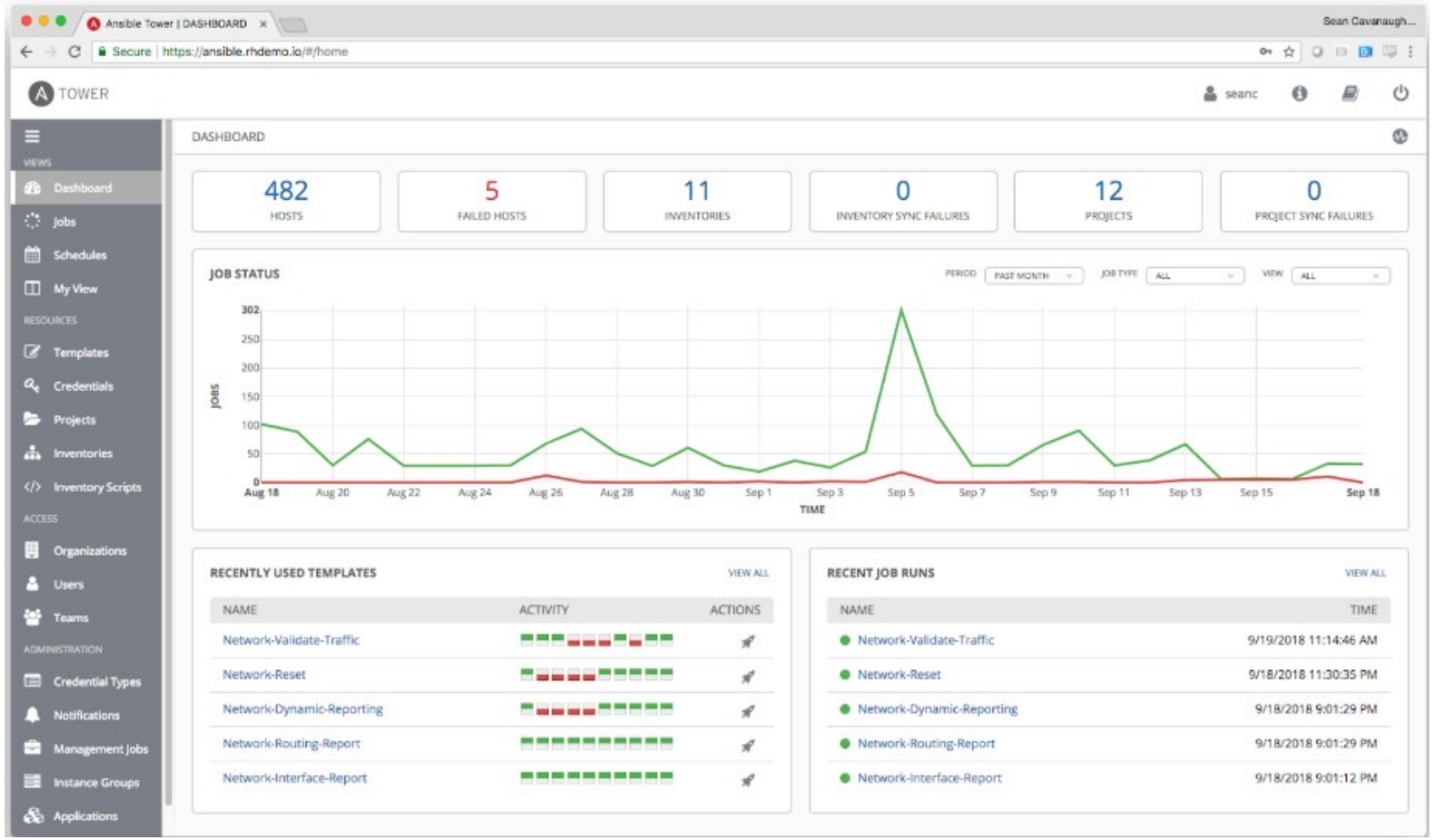
- **Inconvénients:**
  - Expensive
  - You may need more than 10 nodes for a dev/POC environment
  - May be overkill for what you are trying to do.

Ansible Tower

Installation de la tour Ansible

- **https://access.redhat.com/products/ansible-tower-red-hat**

Ansible Tower



RED HAT® ANSIBLE® Tower