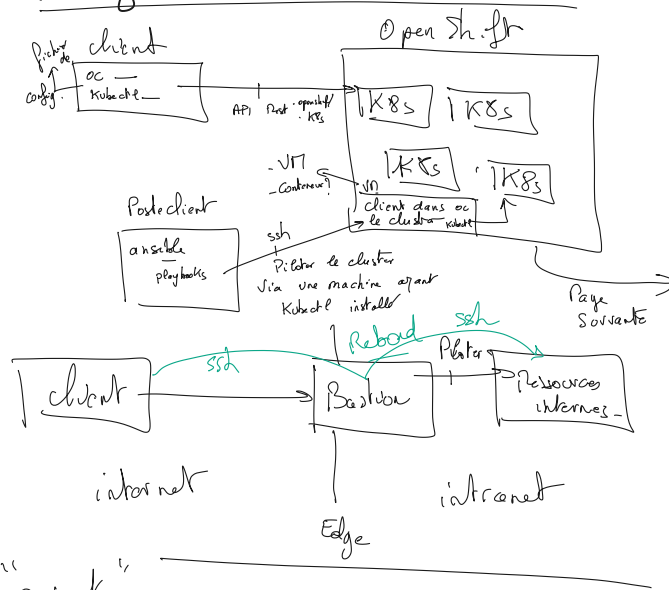
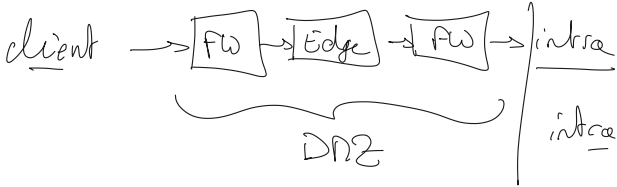
Bonjour tout le monde



"avant"



Paas = VM = n Services dont SSH Server

Paas = Conteneur = 1 Appli / 1 Process / 1 Service

Par Conteneur

↳ Pas de SSH en Conteneurs

---

1. Modèle "Impératif"

→ suite d'instructions → Code / Scripts

opérations
tests
boucles ....

→ Php, Python, java, C#

bash, pwsh (script), groovy

→ Pilote de Machines en impératif

→ ssh → sed / awk

→ cp mv mkdir systemctl etc...

→ sh ⇒ la base en impératif

Problèmes

- incompatibilité
- droits
- syntaxe de commandes

- en cas d'erreur ?
- dépendances
- versions d'OS ?

→ en cas de modif ?

↓ idempotence

mkdir
cp
systemctl
apt .....

Idempotence =

n fois un groupe d'actions doit mener au même résultat

ex: cp fic1 fic2 ⇒ idempotent ?

ex2: echo "ok" > fic3 ⇒ idempotent
↳ écrase

ex3: echo "ok" >> fic4 ⇒ idempotent ?
non !

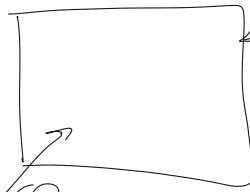
Modifications

C U D

- Create → 1er lancement
- Update → n lancements suivants

} idempotent sous Ansible

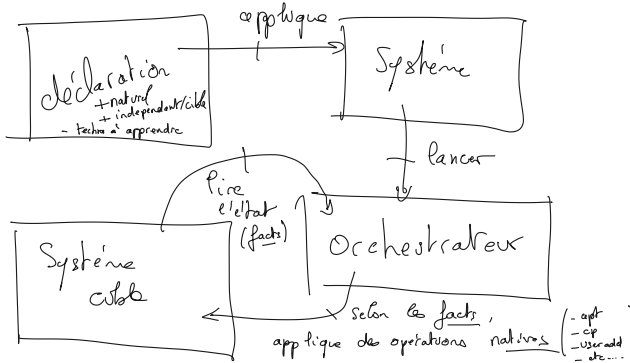
- Delete → Pas fourni
  ↳ autre playbook à créer vous même



Mode déclaratif

→ Est une bonne pratique pour avoir un état souhaité

→ Système idempotent par nature



"Etat désiré"

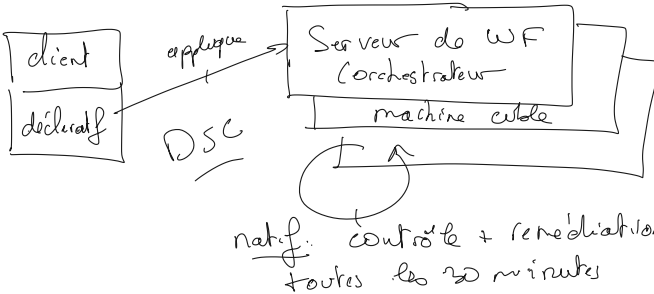
Microsoft DSC → peu de modules

Desired State Configuration → concurrent d'ansible

↳ powershell

↓

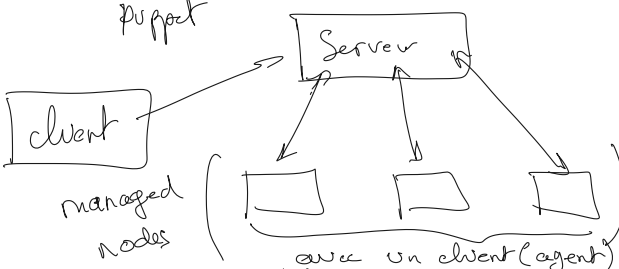
environnement de module

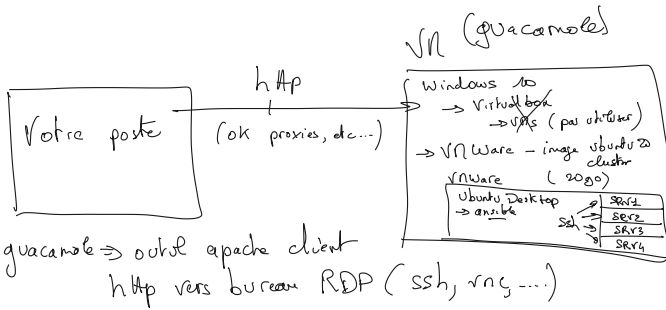


---

chef
puppet

Server





srv1 ansible_user=stagiaire ansible_password=Pa$$w0rd

apt install sshpass

stagiaire@master:~/ansible$ ssh stagiaire@srv2
The authenticity of host 'srv2 (10.0.0.12)' can't be established.
ECDSA key fingerprint is SHA256:LGq+0PVuZVc21S7YIjhVolK/G4SYu6Osrpjys5pEo4E.
Are you sure you want to continue connecting (yes/no/[fingerprint])? ^C
stagiaire@master:~/ansible$ ssh stagiaire@srv2 **-o StrictHostKeyChecking=no**
Warning: Permanently added 'srv2,10.0.0.12' (ECDSA) to the list of known hosts.
stagiaire@srv2's password:

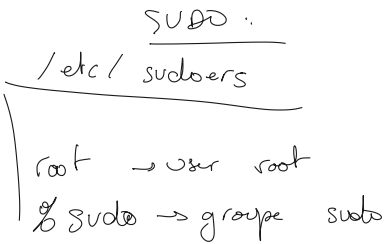
```
[vars]
ansible_ssh_common_args='-o StrictHostKeyChecking=no'

[machines]
srv1
srv2
srv3
srvmachin ansible_host=srv4

[machines:vars]
ansible_user=stagiaire
ansible_password=Pa$$w0rd
```

```
all:
  hosts:
    srv1:
      ansible_user: stagiaire
      ansible_password: Pa$$w0rd
    srv2:
    srv3:
    srv4:
machines:
  hosts:
    srv1:
    srv2:
```

Premier playbook :

```
---
- name: Premier pb qui met à jour les systèmes
  hosts: others   # le filtre dans l'inventaire est ici
  tasks:      # cette série de commande n'est qu'un exemple, il peut y en avoir bien d'autres
   - name: Valider que la machine est bien disponible ( inutile dans la pratique )
     ping:
```

ansible-playbook -i inventory.ini pb1.yaml



Autoriser le passwordless sudo :

%sudo ALL=(ALL:ALL) **NOPASSWD**: ALL

Limiter le nombre d'executions simultanées :
https://docs.ansible.com/ansible/latest/user_guide/playbooks_strategies.html

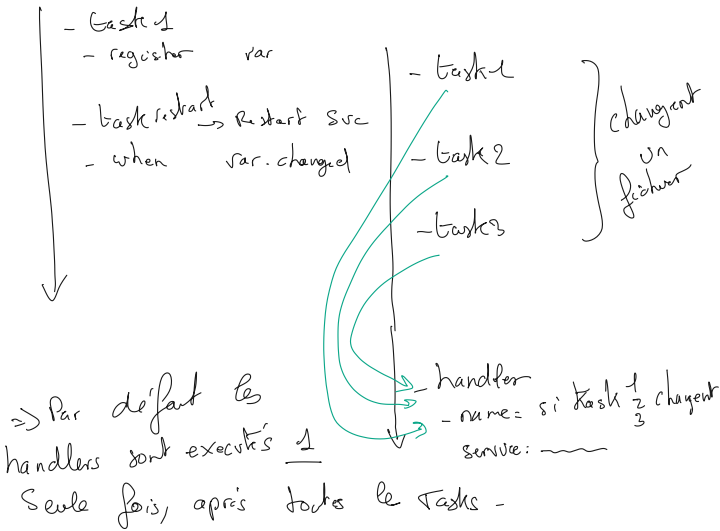
Limiter l'execution d'un PB à un sous ensemble plus petit de machines :
ansible-playbook -i inventory.ini pb1.yaml -l srv1

```
 - name: Mettre à jour le système
    become: yes    # become pour cette tache
    apt:
      update_cache: yes
      upgrade: yes
   - name: Mettre à jour le système v2
    become: yes    # become pour cette tache
    apt: update_cache=yes upgrade=yes
```

```
---
- name: Premier pb qui met à jour les systèmes
  hosts: all   # le filtre dans l'inventaire est ici
  serial: 5
  become: no    # become pour l'intégralité du PB
  tasks:      # cette série de commande n'est qu'un exemple, il peut y en avoir bien d'autres
  - name: Lancer une commande
    command: echo "Bonjour tout le monde"
    register: mycommand
  - debug:
      var: mycommand.stdout_lines
  - debug:
      msg: "Le résultat de ma commande est : {{ mycommand.stdout_lines[0] }}"
```

Essayer d'executer une task plusieurs fois :

```
 - name: purposely fail
    shell: /bin/false
    register: task_register_var
    until: task_register_var is not failed
    retries: 5
    ignore_errors: yes
```



```
---
- name: Modèle de handlers
  hosts: all   # le filtre dans l'inventaire est ici
  become: no    # become pour l'intégralité du PB
  tasks:
  - name: Premier message
    debug:
      msg: Hello no handlers
  - name: Debug qui appelle un handler si changed ( debug ne sera jamais changed,
donc le handler ne sera jamais appelé )
    debug:
      msg: Hello no handlers
    notify: handler1

  - name: Copie un fichier ( ne s'execute pas si il existe déjà )
    become: yes
    copy:
      src: files/my.ini
      dest: /etc/my.ini
    notify: my.ini_created



  - name: Tache qui change quelque chose systématiquement
    become: yes
    file:
      path: /etc/my.touched
      state: touch
    notify: my.touched_changed

  - name: Flush handlers ( nom de tache... )
    meta: flush_handlers

  - name: Tache qui change quelque chose systématiquement - second appel
    become: yes
    file:
      path: /etc/my.touched2
      state: touch
    notify: my.touched_changed

  - name: Flush handlers ( nom de tache... )
    meta: flush_handlers

  handlers:
  - name: my.touched_changed
    debug:
      msg: Le fichier /etc/my.touched a été créé
  - name: my.ini_created
    debug:
      msg: Le fichier /etc/my.ini a été copié
```

U**tilisation des blocs :**

```
tasks:
  - name: Install, configure, and start Apache
    block:
      - name: Install httpd and memcached
        ansible.builtin.yum:
          name:
          - httpd
          - memcached
          state: present

      - name: Apply the foo config template
        ansible.builtin.template:
          src: templates/src.j2
          dest: /etc/foo.conf

      - name: Start service bar and enable it
        ansible.builtin.service:
          name: bar
          state: started
          enabled: True
    when: ansible_facts['distribution'] == 'CentOS'
    become: true
    become_user: root
    ignore_errors: yes
```

Pour utiliser les rôles :
1) chercher un role sur ansible.galaxy.com selon vos besoins
2) télécharger ce role sur votre machine :
ex : ansible-galaxy role install geerlingguy.java
(qui est un role développé par geerlingguy qui fait des choses autour de java)
3) ajouter une référence à ce role dans un playbook :

```
---
- hosts: all
  become: yes
  tasks:
  - name: Les taches s'executent avant ou après les roles ?
    debug:
      msg: Task debug
  roles:
    - geerlingguy.git
```

4) puis exécution du playbook qui exécutera le role :
ansible-playbook -i inventory.ini pb.yaml -l srv1

Exemple : configurer les serveurs SSH

1) télécharger dans le sous dossier roles :
ansible-galaxy role install -p roles willshersystems.sshd

2) l'implémenter dans un playbook:

```
---
- hosts: all
  vars:
    sshd_skip_defaults: true
    sshd:
      Compression: true
      ListenAddress:
        - "0.0.0.0"
        - "::"
      GSSAPIAuthentication: no
      Match:
        - Condition: "Group user"
          GSSAPIAuthentication: yes
    sshd_UsePrivilegeSeparation: no
    sshd_match:
        - Condition: "Group xusers"
          X11Forwarding: yes
  roles:
    - role: willshersystems.sshd
```

Créer nos propres roles :
1) initialisation d'un skelete de rôle :
ansible-galaxy role init --init-path roles nboost.mydebug

2) Contrôle :

```
tree roles/nboost.mydebug/
roles/nboost.mydebug/
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── meta
│   └── main.yml
├── README.md
├── tasks
│   └── main.yml
├── templates
├── tests
│   ├── inventory
│   └── test.yml
└── vars
    └── main.yml

8 directories, 8 files
```

3) implémenter le rôle
exemple : mettre un message Hello World ! :

cat roles/nboost.mydebug/tasks/main.yaml :

```
---
# tasks file for nboost.mydebug
- name: Hello depuis mon role
  debug:
    msg: Hello depuis le role nboost.mydebug
```

4) Appeler le role depuis un playbook :
cat pb-mydebug.yaml

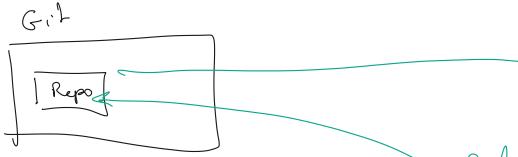
```
---
- hosts: all
  tasks:
  - name: Les taches s'executent avant ou après les roles ?
    debug:
      msg: Un message depuis le playbook principal
  roles:
    - nboost.mydebug
```

5) exécuter le tout :

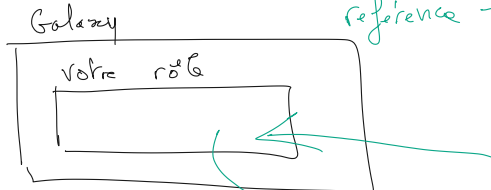
```
ansible-playbook -i inventory.ini pb-mydebug.yaml -l srv1
```

Pour Publier Vos Rôle sur Galaxy:

1. créer un repo Git sur internet (github, etc....)



2. Créer un Rôle local
   ansible-galaxy init compte.role
3. l'implémenter....
4. Le pousser dans le Git
   git add ; commit ; push
5. Créer depuis l'UI galaxy, un Compte, et Rôle
6. Configurer le Rôle avec votre git



7. ansible-galaxy install <compte.role> → téléchargé

Dockerfile

FROM ubuntu

RUN Git clone → Refuse un PB avec ses refs