

# Ansible

CPE-2023

```
setOnLongClickListener(this)
```

```
override fun onLongClick(view: View?): Boolean {  
    if (download != null && downloadName != null)  
        AlertDialog.Builder(context).showQuestion()  
        Toast.makeText(context, R.string.down  
        download2.save(context, downlo
```

# Planning

	Lundi	Mardi	Mercredi	Jeudi
AM	Cours	Cours	Cours	
	Guide TD			
PM	Pratique			Pratique

Alerte Rouge

Lundi 20/02 15h45-17h45
Examen écrit

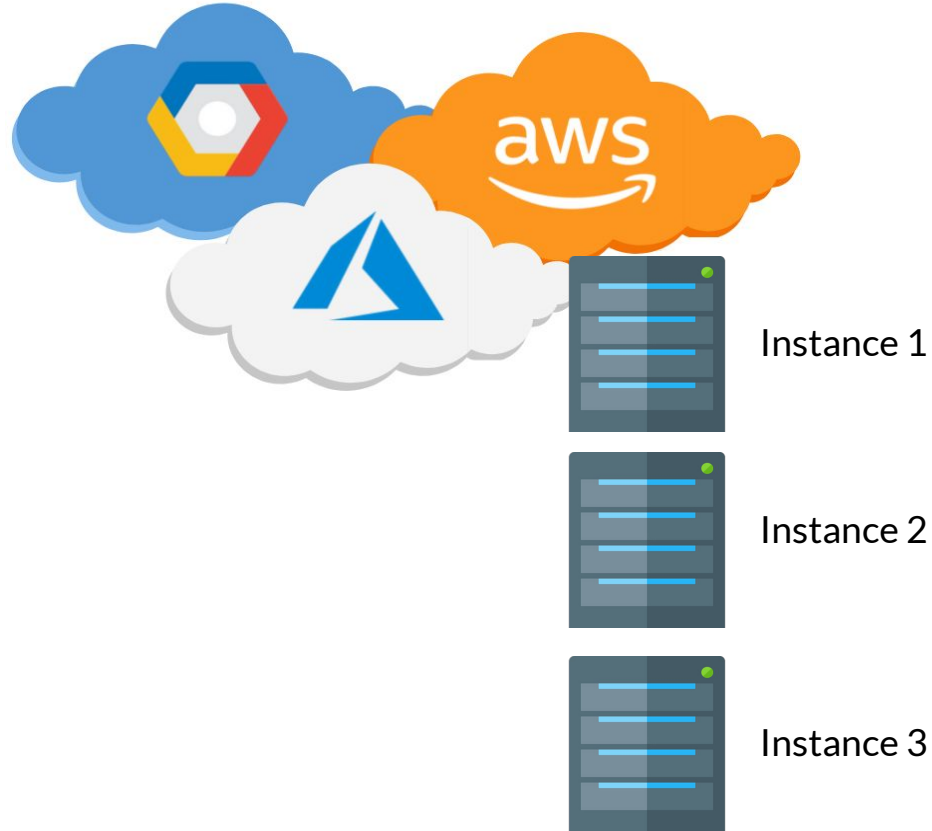


Ansible

# Infrastructure As Code

Manage and provision machines through readable and computable definition files

# Ansible Why ?



# Ansible How ?



An Open Source IT automation engine

### **1. Provisioning**

Set up AUTOMATED INFRASTRUCTURE or  
CLOUD VMs

### **2. Configuration management**

Applications, OS, device.  
Start and stop services, install/update  
applications, implement security policy, ...etc

### **3. Application deployment**

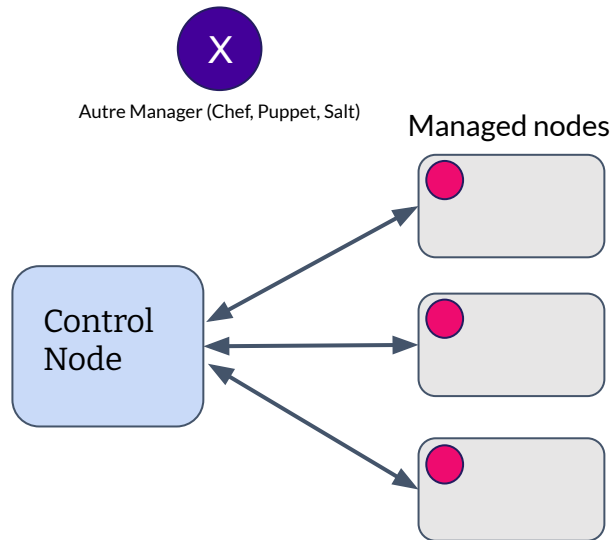
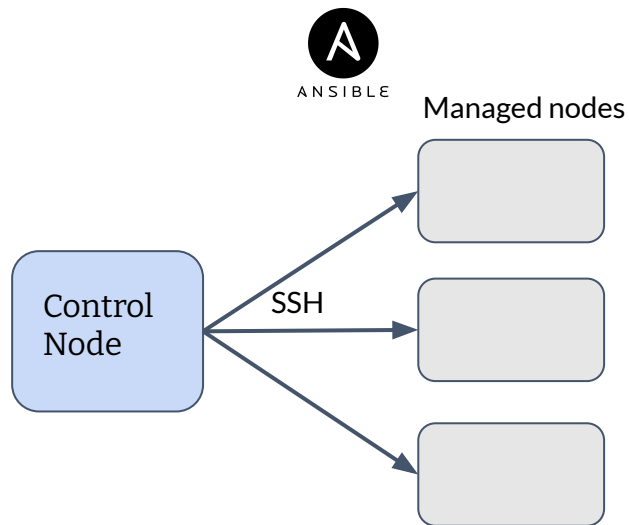
Ansible just requires **Python** and **SSH**



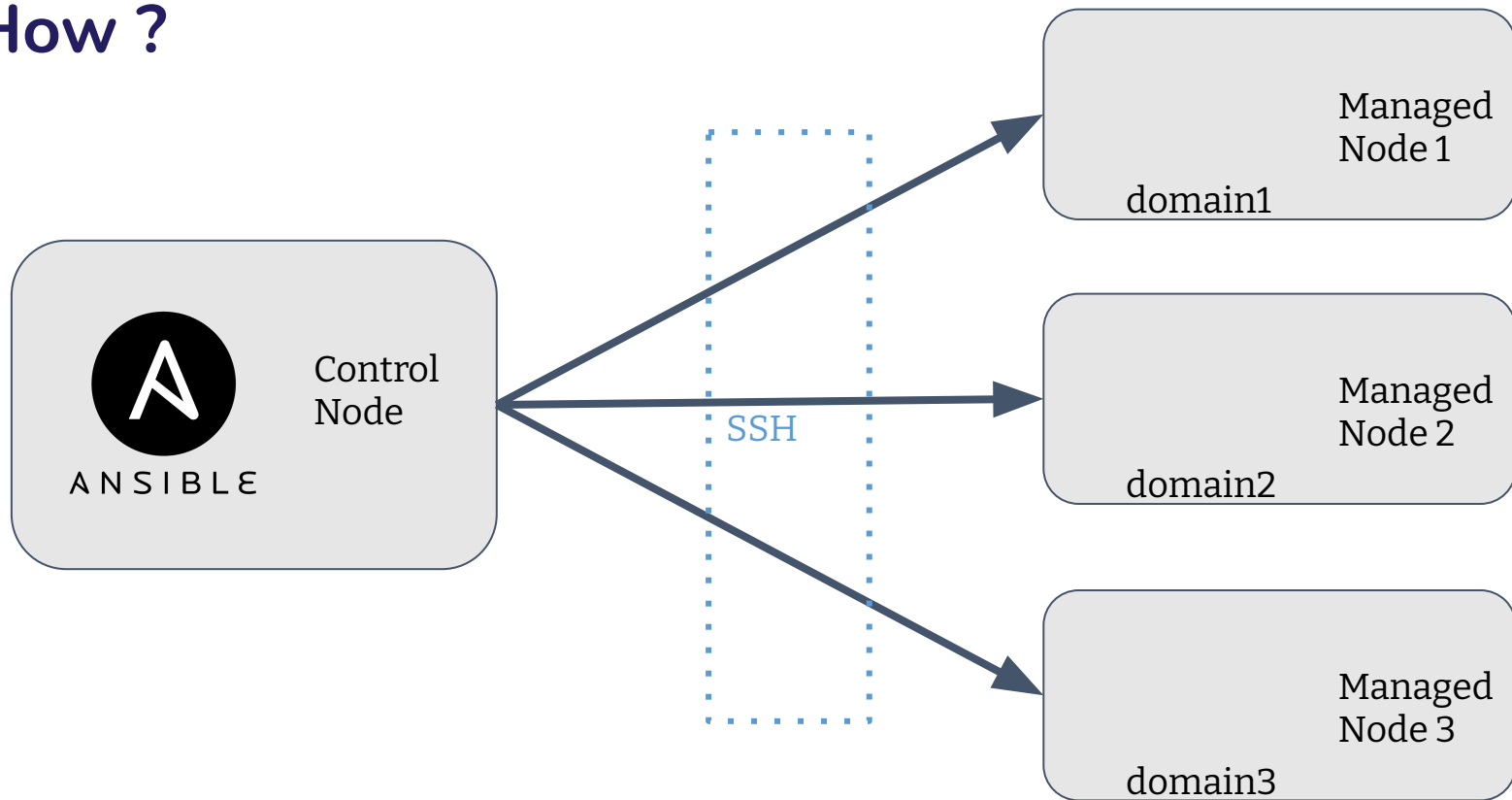
## Ansible is Agentless (SSH)

No need to install agents on nodes

## Declarative vs Imperative



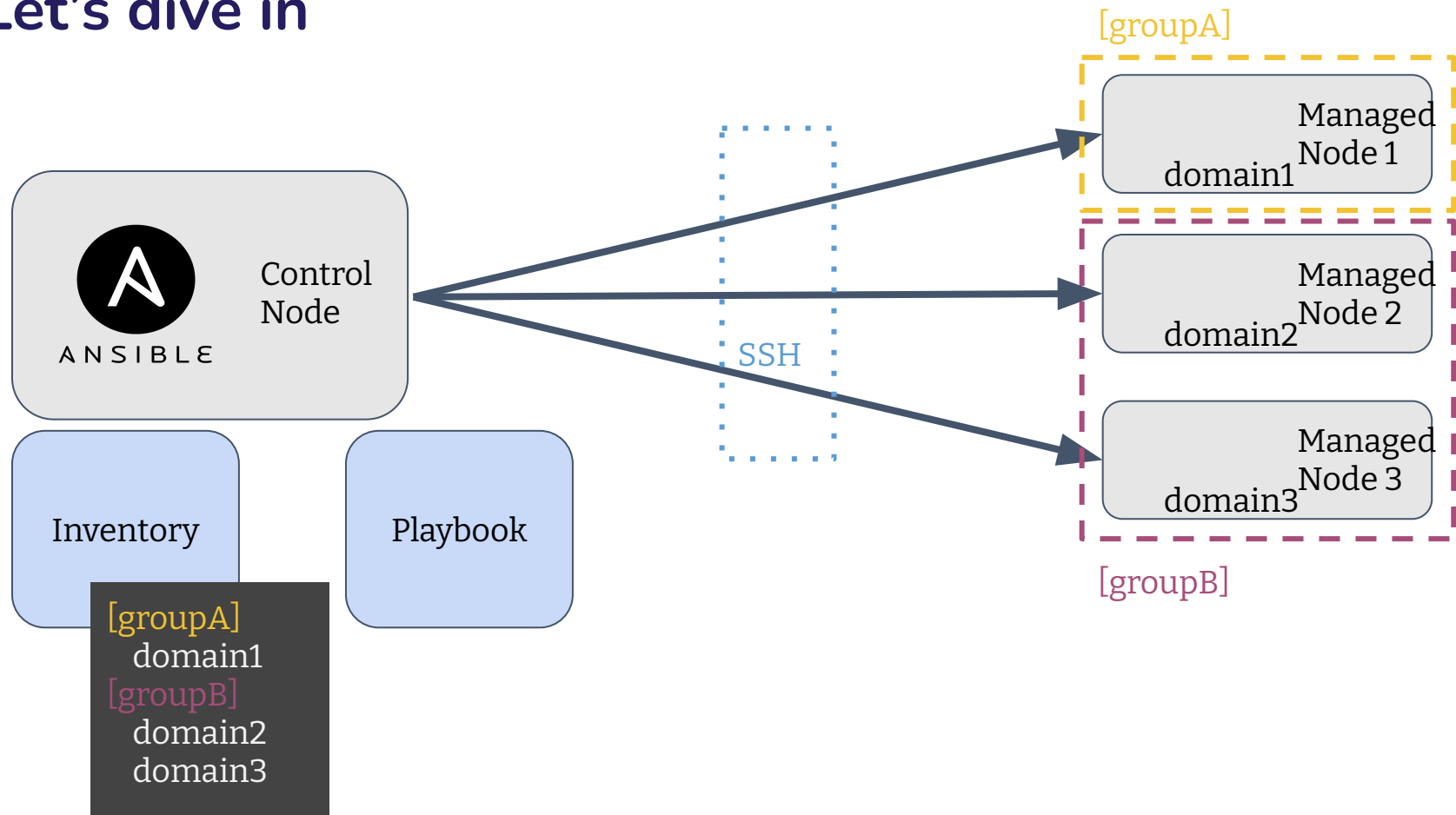
# Ansible How ?





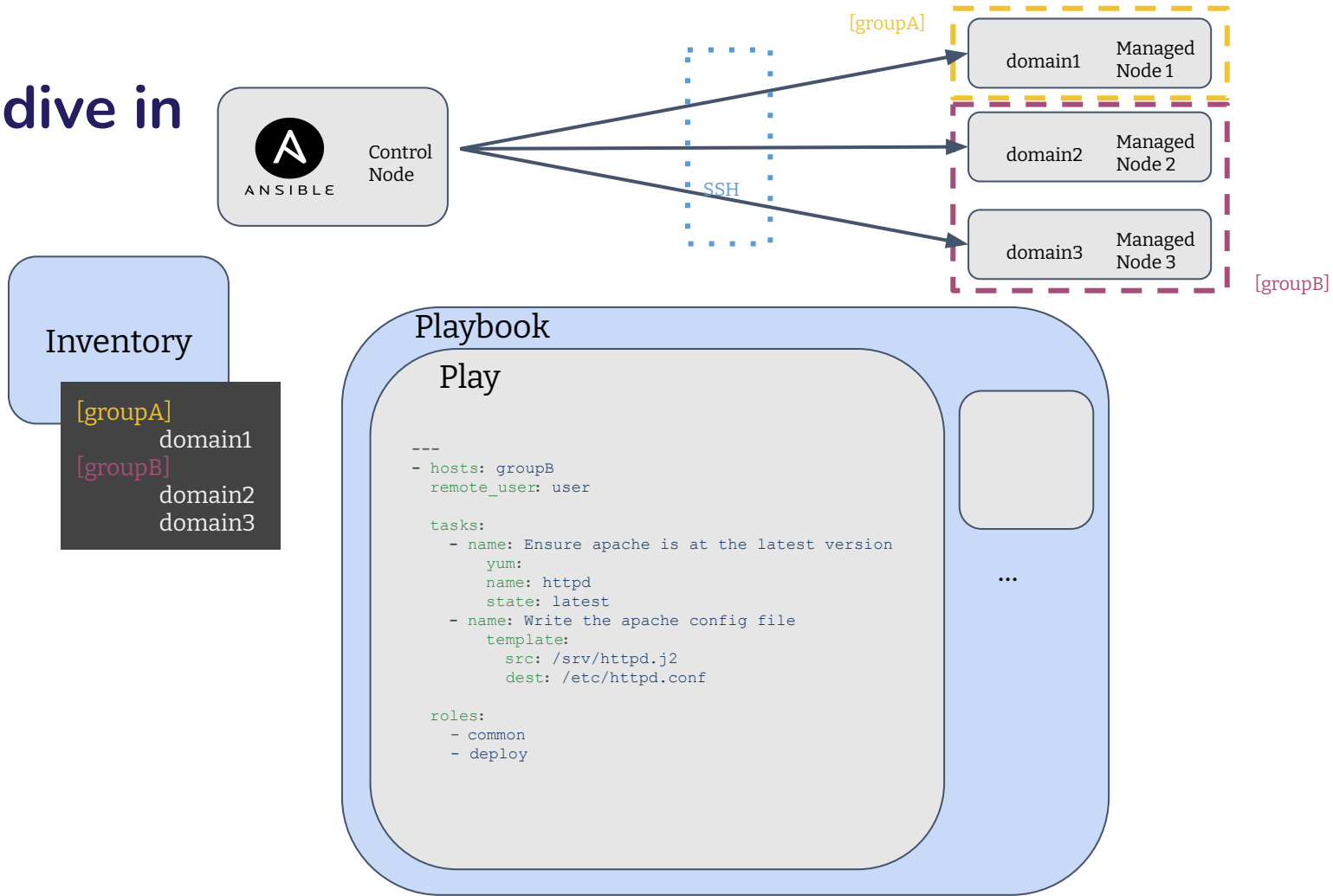
# Ansible

## Let's dive in



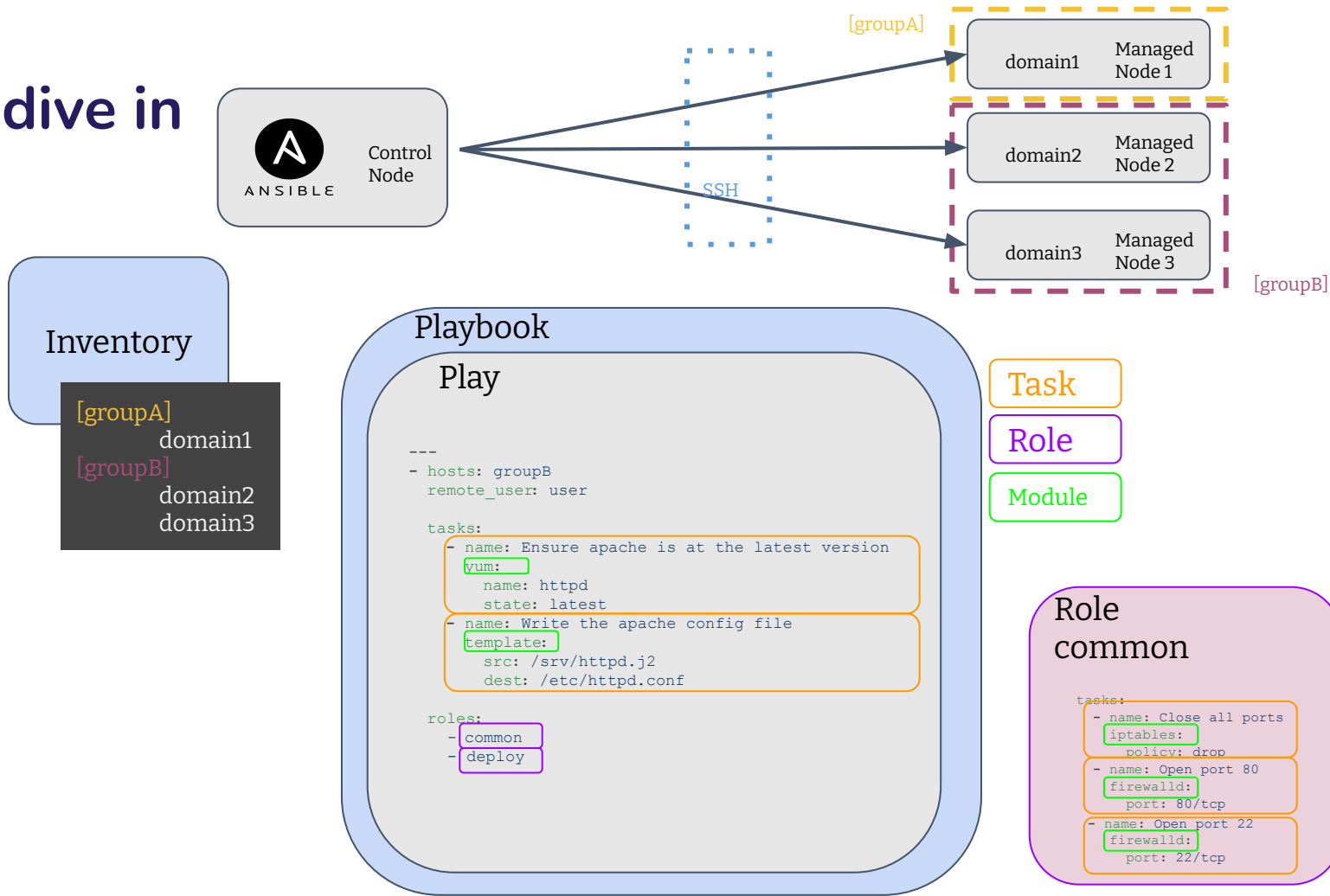
# Ansible

## Let's dive in



# Ansible

## Let's dive in



# Ansible Inventory

- Ansible needs to know where to run the commands
- Ansible's inventory is a list of machines
- Hosts and Groups
- Host/Group Variables
- Tags, SSH keys, Aliases, Login User ...

**Save your whole infrastructure in a file**

/etc/ansible/hosts (exemple au format INI):

```
mail.example.com

[webservers]
foo.example.com
192.168.10.29

[dbservers]
one.example.com
two.example.com
three.example.com
```

équivalent au format **YAML**:

```
all:
  hosts:
    mail.example.com:
  children:
    webservers:
      hosts:
        foo.example.com:
        192.168.10.29:
    dbservers:
      hosts:
        one.example.com:
        two.example.com:
        three.example.com:
```



Ansible

# Inventory/setup.yml

```
--  
all:  
  vars:  
    ansible_user: myuser  
    ansible_ssh_private_key_file: /path/to/my/ssh/key  
  children:  
    prod:  
      hosts: my.dns.takima.io  
...
```

# Ansible Modules

- Lots of build in modules :
  - `setup` : Display all the informations Ansible has on a host
  - `ping` : Ping a host to see if ansible can access it
  - `yum` : Install a package
  - `service` : Start / Stop SystemD daemons
  - `docker_container` : Manage docker containers
- Can add new ones :
  - <https://galaxy.ansible.com/>
  - Write them (in python)

Ansible v2.9  
has about 3,681 modules



Ansible

# playbook.yml

```
--  
- hosts: all  
  become: yes ## Yes I want to become a super user  
  roles:  
    - httpd  
    - firewalld  
...
```



# Tasks/main.yml

```
---  
# Install mysql_db  
-   name: Install mysql  
    yum:  ## I am a module  
        name: mysql_db  
        state: present
```





# Good practice

- Proper file structure
- Keep your plays small (<100 lines)
- Use GIT
- Version & release
- Document -> README file
- Use linting

A couple playbooks and roles file structure :

```
# playbooks
site.yml
webservers.yml
fooservers.yml
roles/
  common/
    tasks/
    handlers/
    library/
    files/
    templates/
    vars/
    defaults/
    meta/
  webservers/
    tasks/
    defaults/
    meta/
```

Enough speaking.  
Take me to the code !

<http://school.pages.takima.io/devops-resources/>