Ansible

Infra as Code, DevOps for real !

Ansible

Un logiciel pour configurer des machines

Ansible

Un logiciel pour configurer des machines

- Installer des logiciels (apt install)

Ansible

Un logiciel pour configurer des machines

- Installer des logiciels (apt install)
- Modifier des fichier de configuration (vim /etc)

Ansible

Un logiciel pour configurer des machines

- Installer des logiciels (apt install)
- Modifier des fichier de configuration (vim /etc)
- Contrôler les services qui tournent (systemctl...)

Ansible

Un logiciel pour configurer des machines

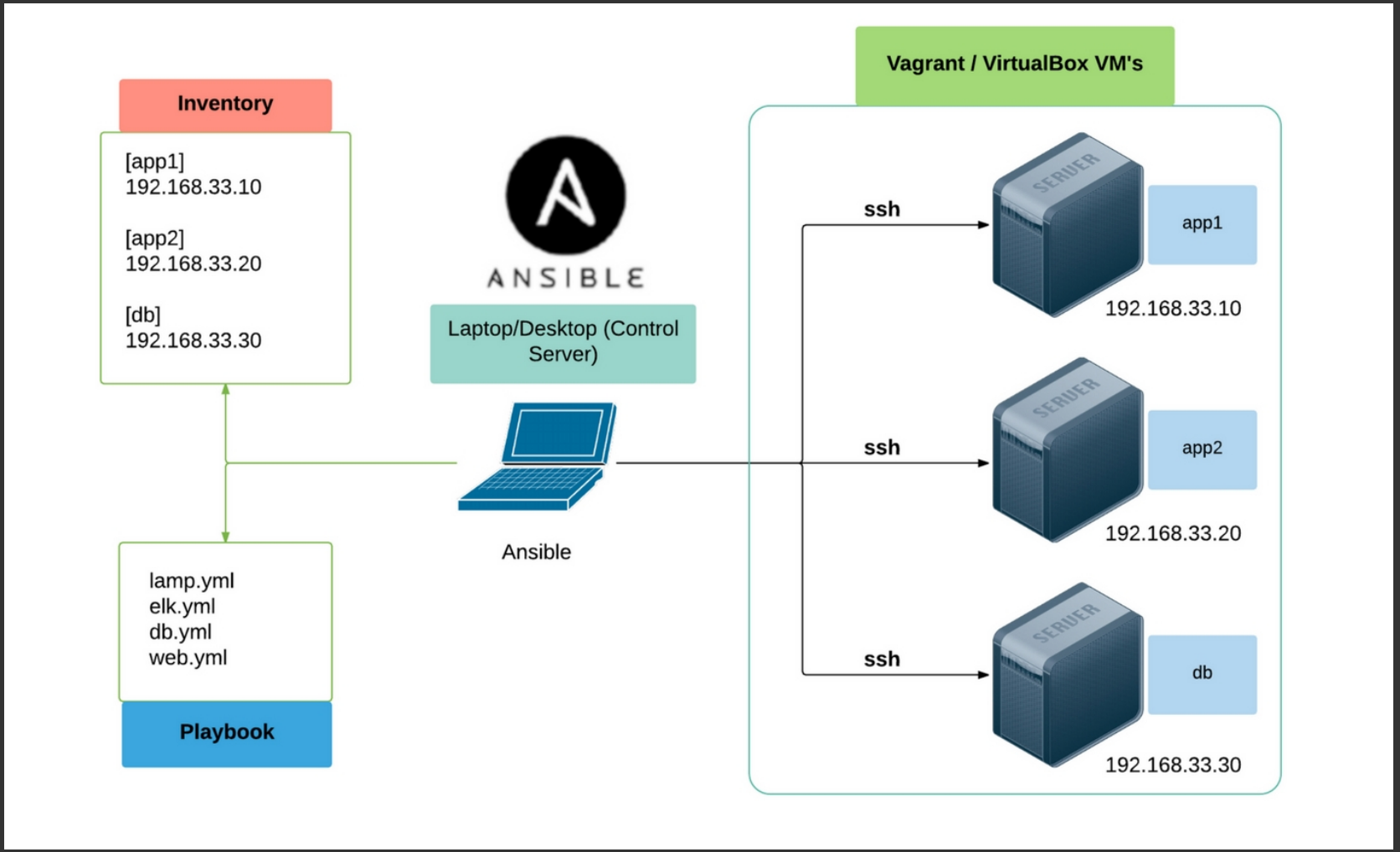
- Installer des logiciels (apt install)
- Modifier des fichier de configuration (vim /etc)
- Contrôler les services qui tournent (systemctl...)
- Gérer les utilisateurs et les permissions sur les fichiers

Ansible

Un logiciel pour configurer des machines

- Installer des logiciels (apt install)
- Modifier des fichier de configuration (vim /etc)
- Contrôler les services qui tournent (systemctl...)
- Gérer les utilisateurs et les permissions sur les fichiers
- Etc.

Ansible en image



Instrastructure As Code

Du code qui contrôle l'état d'un serveur

Un peu comme un script bash mais:

Instrastructure As Code

Du code qui contrôle l'état d'un serveur

Un peu comme un script bash mais:

- **Descriptif** : on peut lire facilement l'**état actuel** de l'infra

Instrastructure As Code

Du code qui contrôle l'état d'un serveur

Un peu comme un script bash mais:

- **Descriptif** : on peut lire facilement l'**état actuel** de l'infra
- **Idempotent** : on peut rejouer le playbook **plusieurs fois** pour s'assurer de l'état

Instrastructure As Code

Du code qui contrôle l'état d'un serveur

Un peu comme un script bash mais:

- **Descriptif** : on peut lire facilement l'**état actuel** de l'infra
- **Idempotent** : on peut rejouer le playbook **plusieurs fois** pour s'assurer de l'état
- Du coup: playbook == état actuel de l'infra

Instrastructure As Code

Du code qui contrôle l'état d'un serveur

Un peu comme un script bash mais:

- **Descriptif** : on peut lire facilement l'**état actuel** de l'infra
- **Idempotent** : on peut rejouer le playbook **plusieurs fois** pour s'assurer de l'état
- Du coup: playbook == état actuel de l'infra
- On contrôle ce qui se passe

Instrastructure As Code

Du code qui contrôle l'état d'un serveur

Un peu comme un script bash mais:

- **Descriptif** : on peut lire facilement l'**état actuel** de l'infra
- **Idempotent** : on peut rejouer le playbook **plusieurs fois** pour s'assurer de l'état
- Du coup: playbook == état actuel de l'infra
- On contrôle ce qui se passe
- Assez différent de l'administration système "adhoc" (= improvisation)

Infrastructure As Code

Avantages

- On peut multiplier les machines (une machine ou 100 machines identiques c'est pas beaucoup plus de travail).

Infrastructure As Code

Avantages

- On peut multiplier les machines (une machine ou 100 machines identiques c'est pas beaucoup plus de travail).
- Git ! gérer les version de l'infrastructure et collaborer facilement comme avec du code.

Infrastructure As Code

Avantages

- On peut multiplier les machines (une machine ou 100 machines identiques c'est pas beaucoup plus de travail).
- Git ! gérer les version de l'infrastructure et collaborer facilement comme avec du code.
- Tests fonctionnels (pour éviter les régressions/bugs)

Infrastructure As Code

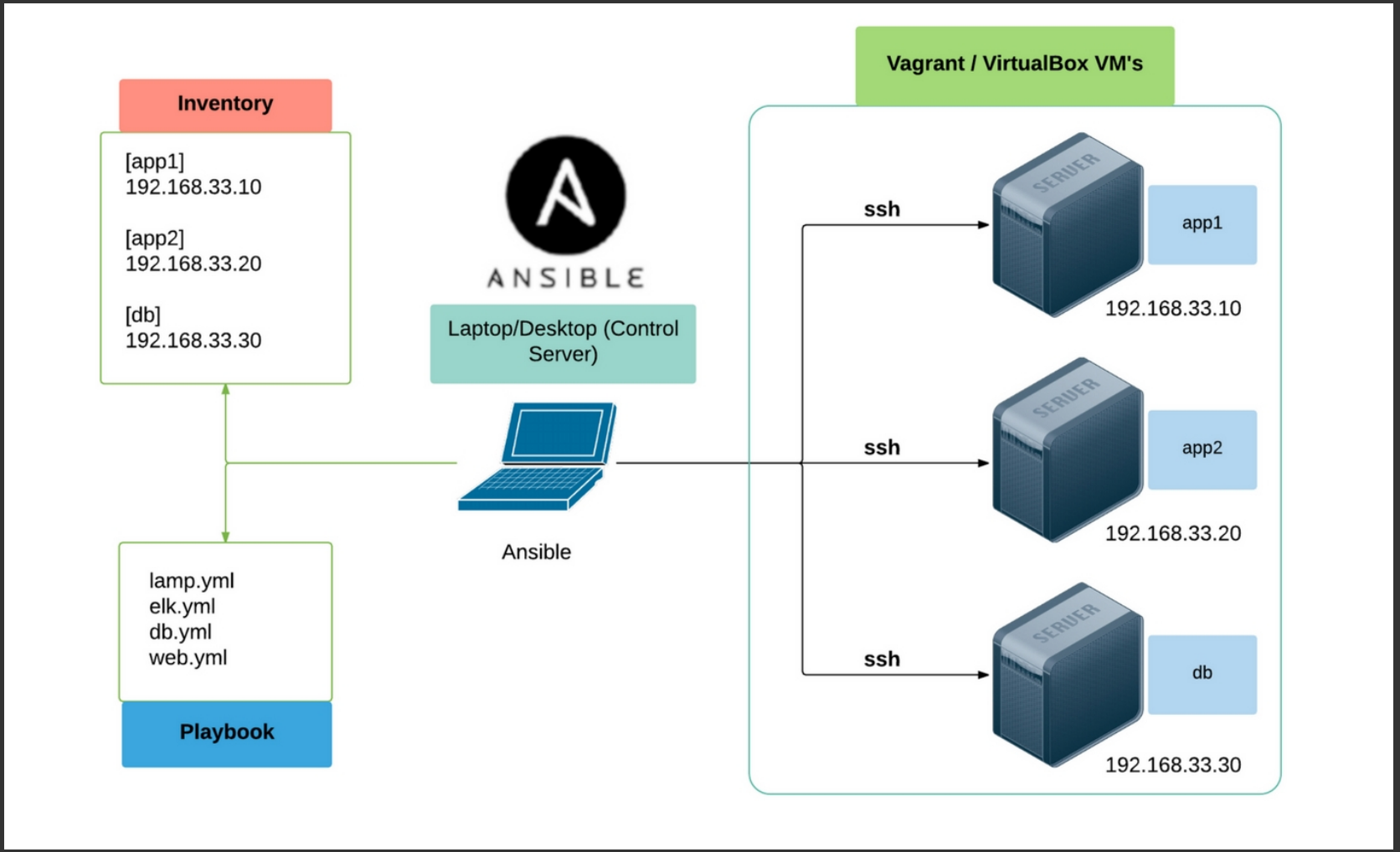
Avantages

- On peut multiplier les machines (une machine ou 100 machines identiques c'est pas beaucoup plus de travail).
- Git ! gérer les version de l'infrastructure et collaborer facilement comme avec du code.
- Tests fonctionnels (pour éviter les régressions/bugs)
- Pas de surprise = possibilité d'agrandir les clusters sans soucis !

Prérequis pour utiliser Ansible (minimal)

1. Pouvoir se connecter en **ssh** sur la machine : **obligatoire** pour démarrer !!!
2. **Python** disponible sur la machine à configurer : **facultatif** car on peut l'installer avec ansible

Ansible en image



L'inventaire des machines

- Une liste de toutes les machines
- Classées par catégories
- Avec des variables spécifiques pour leur configuration

```
[nodes]
node1 ansible_host=10.0.2.4 node_number=1
node2 ansible_host=10.0.2.5 node_number=2
node3 ansible_host=10.0.2.7 node_number=3
node4 ansible_host=10.0.2.8 node_number=4

[nodes:vars]
ansible_user=elk-admin
ansible_password=el4stic
ansible_become_password=el4stic%
```

```yaml
- marché:
    lieu: Crimée Curial
    jour: dimanche
    horaire:
      unité: "heure"
      min: 9

      max: 14
    fruits:
      - nom: pomme
        couleur: "verte"
        pesticide: avec

      - nom: poires
        couleur: jaune
        pesticide: sans
    légumes:
      - courgettes
      - salade
      - potiron
```

Syntaxe: les playbooks ansible sont en YAML

```yaml
- marché:
    lieu: Crimée Curial
    jour: dimanche
    horaire:
      unité: "heure"
      min: 9

      max: 14
    fruits:
      - nom: pomme
        couleur: "verte"
        pesticide: avec

      - nom: poires
        couleur: jaune
        pesticide: sans
    légumes:
      - courgettes
      - salade
      - potiron
```

Syntaxe: les playbooks ansible sont en YAML

- Alignement ! (**2 espaces** !!)

```yaml
- marché:
    lieu: Crimée Curial
    jour: dimanche
    horaire:
      unité: "heure"
      min: 9

      max: 14
    fruits:
      - nom: pomme
        couleur: "verte"
        pesticide: avec

      - nom: poires
        couleur: jaune
        pesticide: sans
    légumes:
      - courgettes
      - salade
      - potiron
```

Syntaxe: les playbooks ansible sont en YAML

- Alignement ! (**2 espaces** !!)
- ALIGNEMENT !! (comme en python)

```yaml
- marché:
    lieu: Crimée Curial
    jour: dimanche
    horaire:
      unité: "heure"
      min: 9

      max: 14
    fruits:
      - nom: pomme
        couleur: "verte"
        pesticide: avec

      - nom: poires
        couleur: jaune
        pesticide: sans
    légumes:
      - courgettes
      - salade
      - potiron
```

Syntaxe: les playbooks ansible sont en YAML

- Alignement ! (**2 espaces** !!)
- ALIGNEMENT !! (comme en python)
- **ALIGNEMENT !!!** (vous allez quand même vous planter :p)

```yaml
- marché:
    lieu: Crimée Curial
    jour: dimanche
    horaire:
      unité: "heure"
      min: 9

      max: 14
    fruits:
      - nom: pomme
        couleur: "verte"
        pesticide: avec

      - nom: poires
        couleur: jaune
        pesticide: sans
    légumes:
      - courgettes
      - salade
      - potiron
```

Syntaxe: les playbooks ansible sont en YAML

- Alignement ! (**2 espaces** !!)
- ALIGNEMENT !! (comme en python)
- **ALIGNEMENT !!!** (vous allez quand même vous planter :p)
- des listes (tirets)

```yaml
- marché:
    lieu: Crimée Curial
    jour: dimanche
    horaire:
      unité: "heure"
      min: 9

      max: 14
    fruits:
      - nom: pomme
        couleur: "verte"
        pesticide: avec

      - nom: poires
        couleur: jaune
        pesticide: sans
    légumes:
      - courgettes
      - salade
      - potiron
```

Syntaxe: les playbooks ansible sont en YAML

- Alignement ! (**2 espaces** !!)
- ALIGNEMENT !! (comme en python)
- **ALIGNEMENT !!!** (vous allez quand même vous planter :p)
- des listes (tirets)
- des paires **clé: valeur**

```yaml
- marché:
    lieu: Crimée Curial
    jour: dimanche
    horaire:
      unité: "heure"
      min: 9

      max: 14
    fruits:
      - nom: pomme
        couleur: "verte"
        pesticide: avec

      - nom: poires
        couleur: jaune
        pesticide: sans
    légumes:
      - courgettes
      - salade
      - potiron
```

Syntaxe: les playbooks ansible sont en YAML

- Alignement ! (**2 espaces** !!)
- ALIGNEMENT !! (comme en python)
- **ALIGNEMENT !!!** (vous allez quand même vous planter :p)
- des listes (tirets)
- des paires **clé: valeur**
- Un peu comme du JSON

A quoi ça ressemble un playbook ?

```
---
- hosts: serveur_web
  vars:
    - logfile_path: "/var/log/monlog"
  tasks:
    - name: installer le serveur nginx
      apt:
        name: nginx
        state: present
    - name: créer un fichier de log
      file:
        path: {{ logfile_path }}
        mode: 755
  handlers:
    - systemd:
        name: nginx
        state: "reloaded"
```

Quatre parties (ou plus)

A quoi ça ressemble un playbook ?

```
---
- hosts: serveur_web
  vars:
    - logfile_path: "/var/log/monlog"
  tasks:
    - name: installer le serveur nginx
      apt:
        name: nginx
        state: present
    - name: créer un fichier de log
      file:
        path: {{ logfile_path }}
        mode: 755
  handlers:
    - systemd:
        name: nginx
        state: "reloaded"
```

Quatre parties (ou plus)

- `hosts:` sur quelle machine on applique la configuration

A quoi ça ressemble un playbook ?

```
---
- hosts: serveur_web
  vars:
    - logfile_path: "/var/log/monlog"
  tasks:
    - name: installer le serveur nginx
      apt:
        name: nginx
        state: present
    - name: créer un fichier de log
      file:
        path: {{ logfile_path }}
        mode: 755
  handlers:
    - systemd:
        name: nginx
        state: "reloaded"
```

Quatre parties (ou plus)

- `hosts:` sur quelle machine on applique la configuration
- `vars:` des valeurs pour la configuration

A quoi ça ressemble un playbook ?

```yaml
---
- hosts: serveur_web
  vars:
    - logfile_path: "/var/log/monlog"
  tasks:
    - name: installer le serveur nginx
      apt:
        name: nginx
        state: present
    - name: créer un fichier de log
      file:
        path: {{ logfile_path }}
        mode: 755
  handlers:
    - systemd:
        name: nginx
        state: "reloaded"
```

Quatre parties (ou plus)

- `hosts:` sur quelle machine on applique la configuration
- `vars:` des valeurs pour la configuration
- `tasks:` les taches de configuration

A quoi ça ressemble un playbook ?

```yaml
---
- hosts: serveur_web
  vars:
    - logfile_path: "/var/log/monlog"
  tasks:
    - name: installer le serveur nginx
      apt:
        name: nginx
        state: present
    - name: créer un fichier de log
      file:
        path: {{ logfile_path }}
        mode: 755
  handlers:
    - systemd:
        name: nginx
        state: "reloaded"
```

Quatre parties (ou plus)

- `hosts:` sur quelle machine on applique la configuration
- `vars:` des valeurs pour la configuration
- `tasks:` les taches de configuration
- `handlers:` des postraitements

A quoi ça ressemble un playbook ?

```yaml
---
- hosts: serveur_web
  vars:
    - logfile_path: "/var/log/monlog"
  tasks:
    - name: installer le serveur nginx
      apt:
        name: nginx
        state: present
    - name: créer un fichier de log
      file:
        path: {{ logfile_path }}
        mode: 755
  handlers:
    - systemd:
        name: nginx
        state: "reloaded"
```

Quatre parties (ou plus)

- `hosts:` sur quelle machine on applique la configuration
- `vars:` des valeurs pour la configuration
- `tasks:` les taches de configuration
- `handlers:` des postraitements
- Des **modules python** !
  - *file*, *apt*, *systemd*

TP1 !

Installer un cluster Elasticsearch et Kibana avec Ansible