Ansible

DigitalHouse>

Certified Developer

The Ultimate Tech Degre

Índice

1. Introducción
2. Arquitectura
3. Casos de uso

1 | Introducción

¿Qué es Ansible?

Ansible es una herramienta para automatizar el aprovisionamiento de servicios en la nube, la gestión de configuraciones y el despliegue de aplicaciones. Resulta sumamente útil para los DevOps, ya que les permite gestionar sus servidores, configuraciones y aplicaciones fácilmente, de manera robusta y paralela. El proyecto se originó en 2013 y fue comprado por Red Hat en 2015.

¿Cómo trabaja Ansible?

Ansible nos permite controlar y configurar nodos desde un servidor central utilizando SSH para conectarse a los servidores y ejecutar las tareas de configuración. Lo que lo hace diferente de otras herramientas de configuration management es que Ansible utiliza la infraestructura de SSH.

¿Por qué elegir Ansible?

- **No usa agentes:** Solo es necesario que se pueda acceder vía SSH y corra Python el equipo donde aplicaremos la configuración.
- **Idempotente:** Solo se realizarán las configuraciones si son necesarias y se podrán aplicar de manera repetida sin provocar efectos secundarios.
- **Declarativo:** Ansible trabaja por nosotros, dejándonos escribir una descripción del estado que deseamos para un servidor o conjunto de servidores. Luego Ansible se encarga de aplicar dicha descripción de forma idempotente.
- **Fácil de aprender:** Encontrarás una gran comunidad, documentación, repositorios con código listo y muchos tutoriales

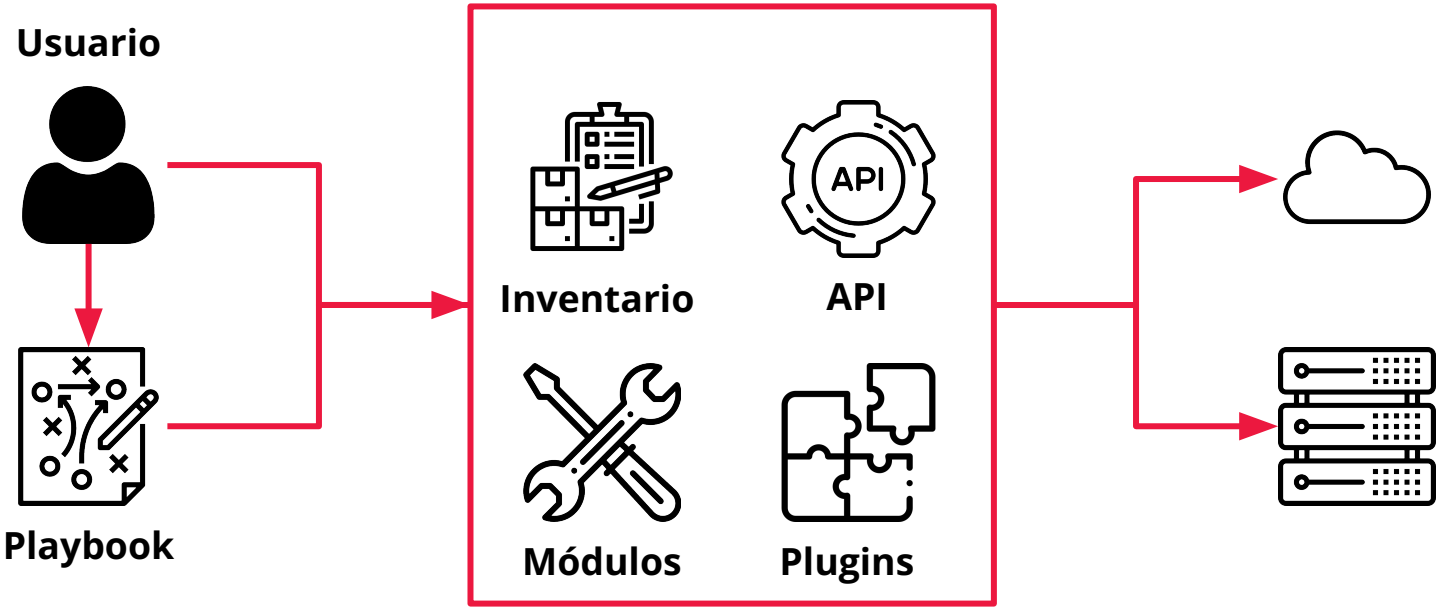
¿Idempotencia?

Wikipedia define a la idempotencia como: “la propiedad para realizar una acción determinada varias veces y aun así conseguir el mismo resultado que se obtendría si se realizase una sola vez”.

En los procesos de infraestructura modernos, en donde las configuraciones se definen en forma de código y muchas veces de forma declarativa, la idempotencia no solo es una herramienta, sino una necesidad para poder implementar procesos de alta predictibilidad.

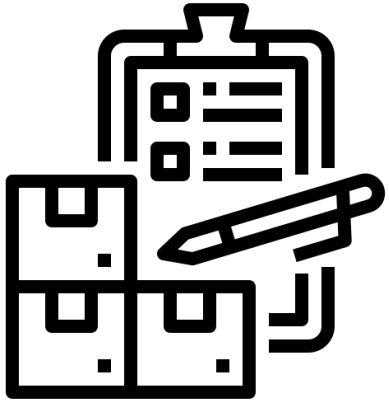
2 | Arquitectura

Arquitectura



Inventario

- El inventario es una lista de los nodos que pueden ser accedidos por Ansible. Por defecto, el inventario está soportado por un archivo de configuration, cuya ubicación es /etc/ansible/hosts. Los nodos pueden estar listados por nombre o IP.
- Cada nodo es asignado a un grupo, como pueden ser “web servers”, “db servers”, entre otros.
- El inventario debe estar escrito en uno de muchos formatos, estos pueden ser YAML, INI, etcétera. YAML es el formato más utilizado en la industria.



Ejemplo de inventario

YAML

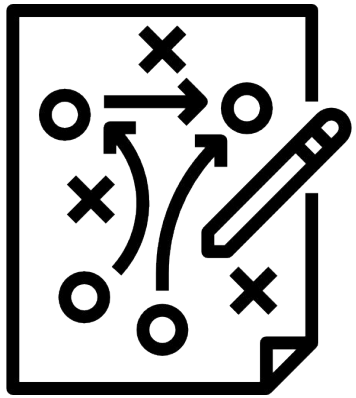
```
mail.example.com

[webservers]
foo.example.com
bar.example.com

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

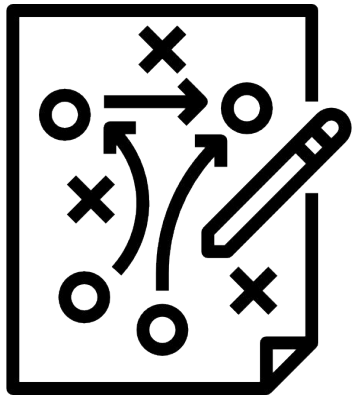
Playbooks

Los **playbooks** son archivos escritos en formato YAML. Estos archivos son la descripción del estado deseado de los sistemas que vamos a configurar. Ansible hace todo el trabajo para llevar los servidores al estado que nosotros hayamos especificado, sin importar el estado en el que estén cuando la configuración se aplique. Los playbooks hacen que las nuevas instalaciones, actualizaciones y la administración del día a día sea repetible, predecible y confiable.



Playbooks

- Los playbooks son simples de escribir y mantener. Se escriben en un lenguaje natural por lo que son muy sencillos de evolucionar y editar.
- Los playbooks contienen Plays (jugadas).
- Las jugadas contienen tareas (en inglés, tasks).
- Las tareas invocan módulos.



Ejemplo de un playbook

Este playbook instala la versión más reciente de Apache y se asegura que esté corriendo en aquellos servidores que estén bajo el grupo **“webservers”** en el inventario:

YAML

```yaml
---
- hosts: webservers
  remote_user: root

  tasks:
  - name: Asegurarse que la ultima version de Apache este instalada
    yum:
      name: httpd
      state: latest

  - name: Asegurarse que Apache este corriendo
    service:
      name: httpd
      state: started
      enabled: yes
```

Módulos

- Hay más de 1000 módulos incluidos con Ansible para automatizar las diferentes partes de nuestro ambiente. Se puede pensar en los módulos como los plugins que hacen el trabajo real de configuración. Cuando se ejecutan las tareas escritas en un playbook, lo que se está ejecutando es en realidad un módulo.
- Cada módulo es independiente (no debería tener dependencia de otros módulos) y se lo puede escribir en cualquiera de los lenguajes de scripting standard de mercado (Python, Perl, Ruby, Bash, etc.). Uno de los principios de diseño de los módulos es la idempotencia.
- Dentro de los módulos más populares podemos encontrar: Service, file, copy, iptables, entre otros.

Ejemplo de invocación de un módulo

Ya vimos que los módulos se incluyen en los playbooks para componer configuraciones complejas o abarcativas. Pero también es posible invocar módulos individualmente desde la línea de comandos una única vez. Ya sea para probar el módulo o realizar una tarea específica.

A continuación, vemos dos comandos, el primero reproduce una de las tareas que ya vimos en un playbook que nos ayuda a asegurarnos que el servicio de Apache está corriendo. Mientras que el segundo invoca el módulo ‘ping’ para hacer un ‘ping’ localmente contra ‘localhost’:

Bash

```
ansible 127.0.0.1 -m service -a "name=httpd state=started"
ansible localhost -m ping
```

La herencia de Python

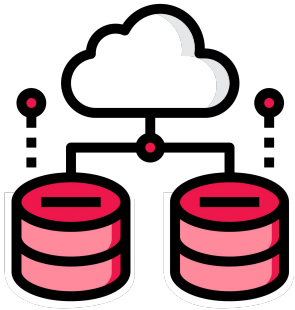
Ansible está desarrollado en Python y, en consecuencia, hereda y/o implementa algunos aspectos del lenguaje. Y si bien, no es necesario ser desarrollador de Python para poder hacer uso de Ansible. Hay algunas cuestiones que es importante conocer:

- **El lenguaje de templating** (Jinja2)
- **Operador ternario:** El operador ternario de Python se puede utilizar dentro de los templates de Jinja para alterar algunos comportamientos de nuestros playbooks en función de ciertas condiciones.
- **Errores:** Muchas veces los errores que encontremos van a estar en un formato que, de estar familiarizados con Python, nos resultará más sencillo de leer.

3 | Casos de uso

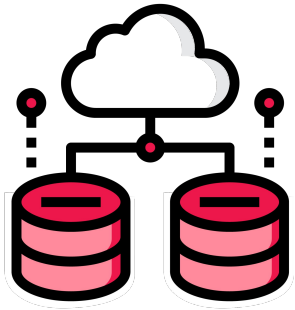
Los casos de uso de Ansible

- **Aprovisionamiento:** utilizar Ansible para instanciar servidores o máquinas virtuales “configurando el sistema de virtualización”.
- **Configuration management:** gestionar y mantener las configuraciones de nuestros servidores.
- **App deployment:** distribuir aplicaciones.



Los casos de uso de Ansible

- **Continuous delivery:** utilizarlo como componente de un proceso de CI/CD para automatizar el despliegue de una aplicación luego de su proceso de compilación.
- **Seguridad y compliance:** la naturaleza idempotente de Ansible hace que podamos utilizarlo para distribuir configuraciones asociadas con la seguridad sin importar la configuración actual.
- **Orchestration:** puede ser utilizado también para orquestar operaciones en la nube o 'configurar' la nube.



Configuration management con Ansible

Ansible es la herramienta más simple para implementar una estrategia de configuration management. Está diseñada para ser minimalista, consistente, segura y altamente confiable. Cualquier desarrollador, tester o administrador de infraestructura puede fácilmente utilizarlo para configurar un conjunto de nodos.

Las configuraciones descritas en **Ansible (playbooks)** son una descripción de la infraestructura (en un lenguaje fácilmente inteligible para el ojo humano) de modo que cualquier persona en el área de IT podría entender el significado de cada tarea en un playbook.

Ansible solo requiere el password o la clave privada del usuario que se utilizará para acceder desde Ansible a los sistemas que se configuraron.

DigitalHouse>