

# introducción

## 1:1 introducción

- herramienta gratuita, open source
- automatización infraestructuras (servidores, cloud, dispositivos)
- similares: chef, puppet, salt
- ventajas ansible:
  - no requiere agente, conexión SSH
  - sintaxis simple y fácil
  - seguro y fácil de mantener
  - rendimiento (al no requerir agentes no consume)
  - no requiere saber programación → YAML
    - otras herramientas requieren Ruby o Python
- desventajas ansible:
  - no es potente como administrador de configuraciones
    - habría que usar GIT como alternativa
  - requiere en grandes entornos configuraciones avanzadas
    - si no es lento
  - resolución lenta de bugs

## 1:2 Instalación

- solo se instala en 1 nodo (server)
- a través del sistema de paquetes
  - CentOS/RedHat: EPEL - `yum install epel-release, yum install ansible`
  - Debian:
    - `sudo apt install software-properties-common` ← añadir otros repositorios vía **PPA**
    - `sudo apt-add-repository ppa:ansible/ansible`
    - o instalar en `/etc/apt/sources.list`:
      - `deb http://ppa.launchpad.net/ansible/ansible/ubuntu trusty main`
      - añadir la clave...
    - `sudo apt update && sudo apt install ansible`
  - `ansible --version`
- los clientes requieren Python 2.4 y conexión SSH

## 1:3 Primeros Pasos

- configuración: `/etc/ansible/hosts`
  - servidores a administrar
  - permite agrupaciones
- para conexiones locales (para ejemplos) añadir al host correspondiente
  - `ansible_connection=local`
- `ansible <servidor|grupos> -m ping` : comprobar conexión a un servidor, a través del módulo correspondiente
- `ansible <servidor|grupos> -a «hostname»` : ejecuta el comando correspondiente
- `ansible <servidor|grupo> -u <usuario> -m ping` : conexiones SSH (con las claves ya puestas en orden) y conectando a través del usuario `<usuario>`
- `ansible all -u <usuario> -a «hostname»`: se conecta con todas las máquinas del fichero

### **/etc/ansible/hosts**

- si hay servidores con diferentes nombres de usuario para conectar, añadir `ansible_user=<usuario>` a cada entrada en **/etc/ansible/hosts**
- `-become` : subir a superusuario
  - `ansible all -m user -a «name=oforte state=present» -become` : necesario para poder crear el usuario en las máquinas remotas (a través del módulo **user**)

## **1:4 Inventario**

- inventario estático
  - formato INI
    - `clave=valor`
    - `[grupos]`
  - `ansible <grupo> -m ping`
  - el mismo dispositivo puede estar en más de un grupo
  - un grupo puede tener subgrupos (un grupo que contiene otros grupos)
    - `[grupo:children]` : incluye la lista de grupos incluidos en esta etiqueta
  - un grupo puede tener variables (children palabra clave)
    - `[grupo:vars]`: usar alguna variable ansible para que se aplique a todo el grupo (vars palabra clave)
    - precedencia de uso de variables:
      1. host (eso incluye los ficheros de variables en directorio **hosts\_vars**)
      2. grupo
      3. grupo padre
      4. por defecto
  - se pueden disgregar los grupos y variables en ficheros adicionales en:
    - `/etc/ansible/group_vars/<GRUPO>`
    - `/etc/ansible/hosts_vars/<SERVIDOR>`
    - estos ficheros usan formato YAML `clave:valor`
  - `-i <fichero>` : usar fichero específico de servidores (en lugar de **/etc/ansible/hosts**)

### **patrones**

- `web[1:5].oforte.net` : expande el nombre a `web1...web5`
- `web[a:f].oforte.net` : idem con letras

### **comandos/parámetros**

- `ansible_connection={ssh|local}`
- SSH:
  - `ansible_host`
  - `ansible_port`
  - `ansible_user`
  - `ansible_ssh_private_key_file`
- `ansible_become={true|false}`
- BECOME:
  - `ansible_become_method={su|sudo}`
  - `ansible_become_user=<USER>` : por defecto, ROOT

## 1:5 Inventario dinámico

- a través de proveedor cloud,. mediante un script
  - AWS
  - GCP
  - DigitalOcean
  - OpenStack
  - Ovirt
  - OpenShift
  - Zabbix
  - ...
  - <https://github.com/ansible/ansible> → contrib/inventory
  - se usa el parámetro **-i** para indicar la ruta al script (este se ejecuta y ofrece la lista al parámetro y por ende a ansible)

## 1:6 ADHOC

- permite realizar acciones de forma simple sin tener que escribir playbooks
- `ansible [opciones] servidores|grupos|all [-m módulo] [-a argumentos/comandos si no se usa módulo]`
  - si no se especifica módulo, por defecto se usa **command**
  - opciones:
    - `-limit | -l <filtro=lista>` : sobre los servidores que queremos aplicar el comando
    - `-user | -u <usuario>`
    - `-become | -b`
    - `-f <n_simultaneo>` : número de servidores simultáneos a los que ejecutar el
    - `-list-hosts` : listar los host a los que afecta la selección
    - `-C` : emulación
    - `-v` : verbose
    - `-vvv` : + verbose

comando

- `-f 1` : ejecución servidor a servidor
- modulos:
- **setup** : información del host (JSON)
- **copy** : copiar ficheros del host-ansible a los clientes
- `-a «src=<origen> dest=<destino>»`
- **yum / apt**
- `-a «name=vim state={present|update|absent}»`
- present : que esté
- update : última versión
- absent : que no esté

## 1:7 configuración

- configuración global
- secciones:
  - valores por defecto generadas
  - configuración de escalar permisos
  - opciones de SSH
  - opciones de SELinux
  - opciones para Ansible Galaxy

- orden de búsqueda:
  1. variable de entorno: ANSIBLE\_CONFIG
  2. ficheros: ansible.cfg / .ansible.cfg
  3. fichero: /etc/ansible/ansible.cfg

## 1:8 windows

- en servidores windows hay que instalar:
  - en ansible **pywinrm** (para remote management)
    - `pip install pywinrm`
    - `easy-install pip` : gestor de paquetes de Python
    - definir **connection** al valor **winrm**
  - en servidor windows:
    - PowerShell v3.0
    - `ConfigureRemotingForAnsible.ps1` : script que hace los ajustes necesarios para la administración remota, desde la web de ansible (apartado windows, Windows System Prep)
    - habilitar puerto 5986
  - `ansible <winserver> -c winrm -k -u alberto -m win_ping`
    - `-c` : forzar el tipo de conexión
    - `-k` : fuerza solicitud contraseña
    - ignorar certificado windows, añadir al fichero de configuración (o usar `-e` para pasar este parámetro por línea de comando): `ansible_winrm_server_cert_validation=ignore`

## 1:9 combinar inventarios

posibilidad de combinar varios inventarios (ya sean estáticos o dinámicos)

- diferentes proveedores cloud
- servidores locales
- etc
- se crea un directorio y añadimos los inventarios (fichero o script.py) y usando la opción `-i` para indicar el directorio
- dentro podemos tener los directorios **group\_vars** y **host\_vars** como en otras situaciones
- es posible referenciar desde un inventario a otro
- se recomienda hacer un `--list-hosts` para ver los servidores afectados por la unión.

## 1:10

From: <https://miguelangel.torresegea.es/wiki/> - miguel angel torres egea

Permanent link: <https://miguelangel.torresegea.es/wiki/info:cursos:udey:ansible:introduccion?rev=1535721237>

Last update: 31/08/2018 06:13

