



**ANSIBLE - Aprender**

**ANSIBLE - Hacer**

- Documentación oficial ansible
- Excelente tutorial en español

# ANSIBLE - ¿Qué es?

- Es una herramienta de automatización de tareas sobre sistemas de información. Permite configurar sistemas, desplegar software y orquestar tareas de TI sea cual sea su grado de complejidad o número de sistemas afectados.
- Funciona sobre GNU/Linux.
- Dependencias en todos los nodos: Python + OpenSSH

# Ansible - Conceptos

## NODO DE CONTROL

- Cualquier máquina con Ansible instalado.
- Puede ejecutar comandos y playbooks de Ansible invocando el comando ***ansible*** o ***ansible-playbook***,
- Pueden existir varios nodos de control.

# Ansible - Conceptos

## NODO ADMINISTRADO

- Los dispositivos de red y/o servidores que se administran con Ansible.
- Los nodos administrados también se denominan "hosts".
- Ansible no necesita ser instalado en los nodos administrados.
- En los nodos administrados debe estar instalado OpenSSH y Python

# Ansible - Conceptos

## INVENTARIO

- Una lista de nodos administrados.
- Un archivo de INVENTARIO también se denomina "hostfile".
- El INVENTARIO puede especificar información como la dirección IP para cada nodo administrado.
- Un INVENTARIO también puede organizar nodos administrados, creando y anidando grupos para facilitar el escalado.



# Ansible - Conceptos

## COLECCIONES

- Un formato que permite distribuir contenidos Ansible.
- Todo lo que se construye con Ansible se puede empaquetar en COLECCIONES y ponerse a disposición de otros.
- En el interior de una colección puede haber: [playbooks](#), roles, modules y plugins



# Ansible - Conceptos

## MODULOS

- Unidades de código que se ejecutan en Ansible
- Cada módulo tiene un uso particular (p.e. administrar usuarios en un tipo específico de BBDD)
- Se invocan en solitario con una tarea o se pueden invocar varios desde un playbook.

# Ansible - Conceptos

## TAREAS

- Es la unidad de acción en Ansible.
- Se puede ejecutar una tarea una sola vez con un comando ad-hoc (una acción lanzada con un objetivo concreto en un momento concreto sin plan de ser repetida a lo largo del tiempo)

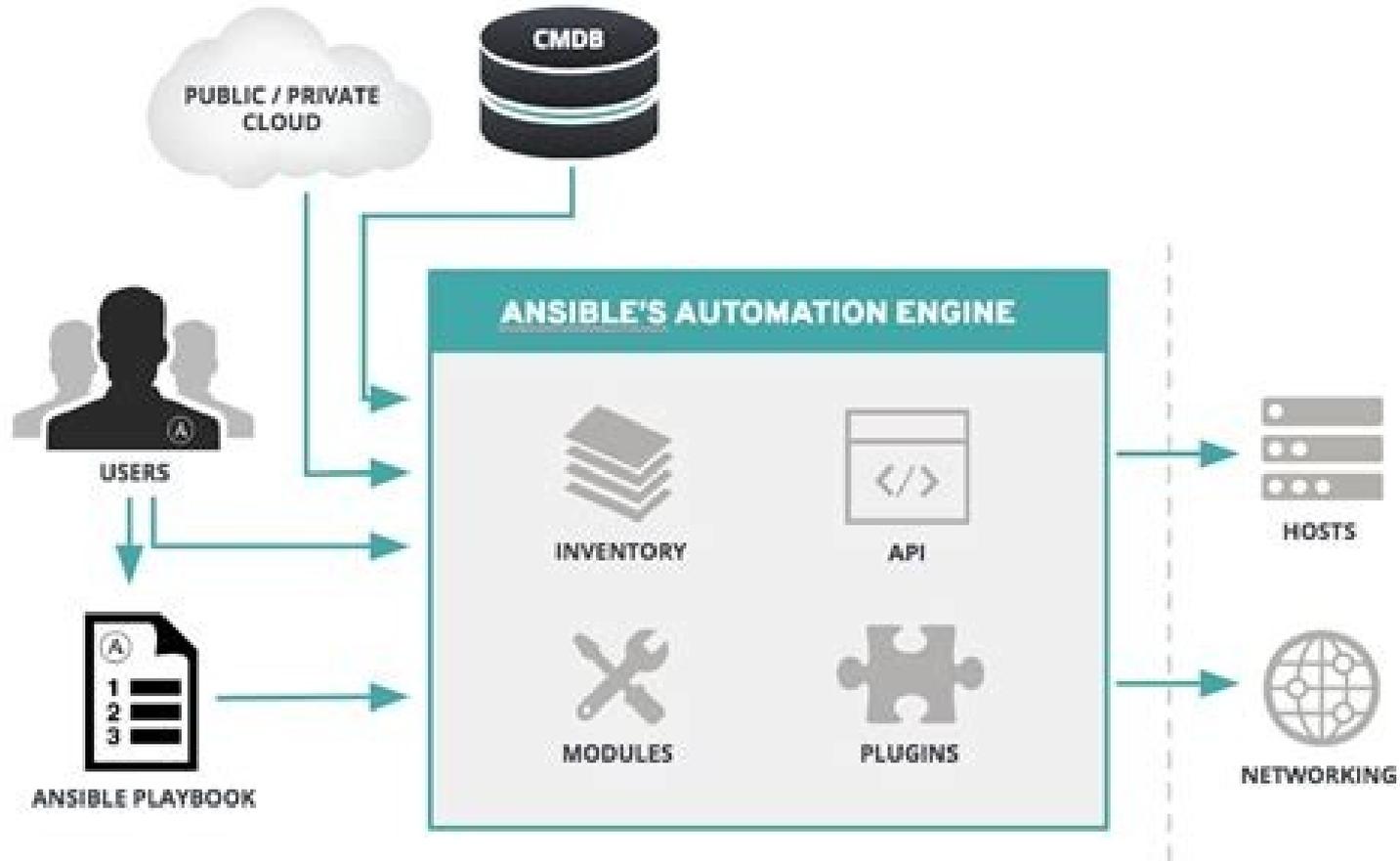


# Ansible - Conceptos

## PLAYBOOKS

- Son listas ordenadas de tareas, guardadas para que pueda ejecutar esas tareas en ese orden repetidamente.
- Los playbooks pueden incluir tanto variables como tareas.
- Los playbooks están escritos en [YAML](#) y son fáciles de leer, escribir, compartir y comprender.

# Ansible - Recapitulación I



## Glosario

CMDB

# ANSIBLE - Infraestructura para el aprendizaje I

Crear la siguiente Infraestructura en *AWS* vía *Learner Lab Foundation Services* con instancias autorizadas en las versiones más recientes:

- Nodo de control: Amazon Linux 2
- Nodo gestionado 1: Amazon Linux 2
- Nodo gestionado 2: Amazon Linux 2
- Par de claves idéntico para los tres nodos.
- Grupo de seguridad único para los tres nodos con la siguiente “rule”:
  - Apertura del tráfico TCP desde y hacia el puerto 22 posicionado sólo en la dirección IPv4

# ANSIBLE - Infraestructura para el aprendizaje II

## Ajustes en los nodos I

- En todos los nodos: `HOSTNAME=<20nn-nn>-nodo-<n>`
- Siendo `root` en `nodo-0` crear par de claves cifradas con el comando `ssh-keygen -t rsa -b 4096 -q -C "nodo-de-control"` y establecer manualmente relación de confianza con los nodos gestionados llevando el contenido de `nodo-0:/root/.ssh/id_rsa.pub` a `nodo-<N>:/root/.ssh/authorized_keys`. **OJO!!** sin machacar lo que ya haya en el destino puesto que ya existe el fichero `/root/.ssh/authorized_keys`.
- En `nodo-0` actualizar `/etc/hosts` con las direcciones IP internas de de los tres nodos.
- Insertar la actualización anterior en los ficheros `/etc/hosts` de los nodos gestionados haciendo uso de la relación de confianza ya establecida. La orden sería la siguiente: `ssh nodo-<n> "echo -e '<Las tres últimas líneas de /etc/hosts de nodo-0 separadas por \n (no por espacios en blanco)' >> /etc/hosts"`

# ANSIBLE - Infraestructura para el aprendizaje III

## Instalación de ansible

- (1) Desde los repositorios de la distro: La mejor elección siempre que se pueda. ->[Instrucciones detalladas](#)<-.
- (2) Desde las fuentes: Sólo si no está disponible en los repositorios de la distro. Ansible está desarrollado en Python así que NO se rige por el procedimiento estandard (`configure`, `make`, `make install`). ->[Instrucciones detalladas](#)<-.
- Por defecto, el sistema elegido en AWS para el nodo-0 no cuenta con ansible en los repositorios, así que procederemos con (2). Tal y como recomienda la documentación de instalación los haremos en modo `--user` (visible sólo por el usuario ordinario desde el cual se instala).

# ANSIBLE - Inventario

- El inventario es mucho más que una relación de hosts sobre los que actuar en algún momento. La lectura del documento enlazado arriba nos da una idea de las enormes capacidades de un *inventario*.
- Ya que estamos en modo aprendizaje, vamos a crear la mínima expresión de un inventario con nuestros nodos gestionados usando [este apartado de la documentación oficial](#).

# ANSIBLE - A trabajar

- Comandos disponibles en la línea de órdenes:

```
ansible;  
ansible-console;  
ansible-inventory;  
ansible-config;  
ansible-doc;  
ansible-playbook;  
ansible-vault;  
ansible-galaxy;  
ansible-pull
```

# ANSIBLE - comandos ad-hoc I

Es la forma más simplificada de interactuar con ansible sobre los nodos gestionados. La sintaxis típica es la siguiente:

```
ansible pattern -m [module] -a "[module options]"
```

**pattern** Obligatorio. Patrón de coincidencias con referencias del **inventario**

**-m [module]** Opcional. El módulo que deseamos ejecutar, si no se indica se usa **ansible.builtin.command** (el de por defecto).

**-a "[module options]"** Opcional si el módulo no requiere opciones. En el caso de **ansible.built.command** necesita estar cumplimentado

# ANSIBLE - comandos ad-hoc II

- `ansible nodos -a "/sbin/reboot"`
- `ansible nodos -a "uname -a"`
- `ansible nodos -a "apt update"`
- `ansible nodos -a "echo $PATH"`

Aplicar en los **nodos** todos los casos de uso publicados en [este enlace](#) y comentar lo particular de la sintáxis cuando no se utiliza el módulo por defecto ([ansible.builtin.command](#))

# ANSIBLE - playbooks

- playbooks es a ansible lo que los shell-scripts son al GNU/Linux.
- Automatizan las tareas largas y complejas que puede hacer ansible.
- Se pueden ejecutar en el tiempo tantas veces como se desee (tareas repetitivas).
- Están escritos en lenguaje **YAML**.
- Admiten bifurcaciones condicionales y bucles iterativos controlados. (características propias de un lenguaje de programación).

# ANSIBLE - playbooks

- Haciendo uso de [este ejemplo](#) incluido en la documentación de Ansible crear playbooks que monten en los nodos gestionados una plataforma LAMP formada por **Apache Web Server + PostgreSQL + PHP**. Todos los paquetes serán traídos desde los repositorios de la distribución de los nodos. Aplicar la segmentación que denota [este repositorio](#) para mayor actividad por parte de los alumnos.
- Antes de invocar su ejecución chequear la validez del código del playbook con la opción **--check de ansible-playbook**

# ANSIBLE - Adquisición de privilegios I

Escenario:

- Sistema: **GNU/Linux**
- IP: **10.10.10.10**
- Usuario ordinario con sesión abierta vía ssh. Nombre del usuario "**operador**".
- Ansible está instalado en la máquina **10.10.10.10** (**apt-get install ansible**)
- El usuario "**operador**" tiene la password del usuario "**root**" en el servidor **10.10.10.20**.

Con este escenario, el usuario "**operador**" puede construir y ejecutar playbooks y/o lanzar comandos directos contra **10.10.10.20** si se dan las siguientes condiciones:

- 1 "**root**" ha generado un par de claves ssh en **10.10.10.20** y ha traído a **operador@10.10.10.10:\$HOME** la parte privada de la clave.
- Los comandos **ansible** y/o **ansible-playbook** deben incorporar los siguientes modificadores al invocarlos:

# ANSIBLE - Adquisición de privilegios II

- **-i** Para indicar el fichero que contiene el inventario.
- **-u** Para indicar el usuario con el que nos identificamos en la/s máquina/s objetivo.
- **--key-file** Para proporcionar el fichero que contiene la clave privada.

Ejemplos:

```
ansible nodo -i /home/operador/inventario -u root --key-file=/home/operador/clave_privada -m apt -a "update_cache=yes"
```

```
ansible-playbook -i /home/operador/inventario -u root --key-file=/home/operador/clave_privada nombre_del_playbook.apb
```

# ANSIBLE - Alternativas



Y muchos más .....