



Administración de servidores

con Ansible

Martín Gil Blanco

Introducción

La infraestructura TI se refiere al conjunto de hardware, software, recursos de red y servicios necesarios para la existencia, operación y gestión de un entorno de TI empresarial. Permite a una organización ofrecer soluciones y servicios de TI a sus empleados, socios y/o clientes, y normalmente es interna a la organización y se despliega dentro de instalaciones propias.



- Automatización

Despliega tu infraestructura de TI (instalación/configuración de software) sin intervención humana (sysops).

¿Por qué evoluciona la forma de gestionar la infraestructura TI?

Mejoras tecnológicas

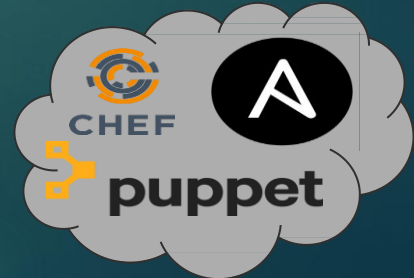
- La computación en la nube, los servidores dedicados y los servidores privados virtuales (VPS) más baratos y mejores
- Evolución de las tecnologías de virtualización y de contenedores
- Evolución de las tecnologías y productos de programación

Aumento de la demanda de servicios de TI (empleados, socios y clientes)

- Servicios de computación en la nube (almacenamiento, software, etc.)
- Diversificación empresarial

Diseño de servicios

- Escalado de servicios (evitar morir de éxito): Balanceo de carga
- Alta disponibilidad



¿Qué es Ansible?

- Ansible es una solución de software libre diseñada para administrar la configuración, desplegar aplicaciones, aprovisionar recursos y automatizar la infraestructura de TI.
- Destaca por ser una herramienta confiable, consistente y capaz de escalar fácilmente.
- Su configuración es sencilla y su uso resulta bastante intuitivo.
- A diferencia de otras alternativas, no requiere instalar agentes, ya que se apoya principalmente en el protocolo SSH.

¿Qué es Ansible?

- Su uso resulta sencillo porque se basa en el enfoque de *Infrastructure as Code (IaC)*.
- Esta metodología permite definir archivos de configuración para administrar y aprovisionar la infraestructura sin depender de tareas manuales repetitivas.
- En Ansible, la automatización se lleva a cabo mediante los llamados *Playbooks*, que son los documentos donde se describe y ejecuta todo el proceso.

Características Ansible

- **Preparación:** facilita la instalación y configuración de la infraestructura tecnológica, incluyendo servidores, redes, usuarios y servicios.
- **Gestión de configuración:** permite definir y aplicar configuraciones de forma ágil y sencilla, manteniendo los sistemas estables y en el estado deseado.
- **Despliegue de aplicaciones:** con Ansible es posible instalar y distribuir aplicaciones en varios sistemas de manera simultánea y coordinada.
- **Orquestación:** ofrece la capacidad de organizar y coordinar los distintos componentes de la infraestructura para que trabajen de forma integrada y adecuada

¿Cómo funcionan los productos de automatización?

Receta:

```
web_servers:
  install packages
    - apache2
    - libapache2-mod-php7
    - php7-mysql
  copy ./website to /var/www/html/

db_server1:
  install packages
    - mysql5-server
    - mysql5-client
  update config mysql5-server bind-address=0.0.0.0
  copy db.sql to /root
  run mysql -u root < \
    /root/db.sql
```

Inventario:

```
[web_servers]
  www1 192.168.1.2
  www2 192.168.1.3
  www3 192.168.1.4

[db_servers]
  db_server1 192.168.1.7
```


Ventajas de la automatización

Rol de administrador de sistemas más enfocado en resolver problemas específicos

- + Productividad

(Re)desplegar nuevos servicios rápidamente

- + Resiliencia de los servicios

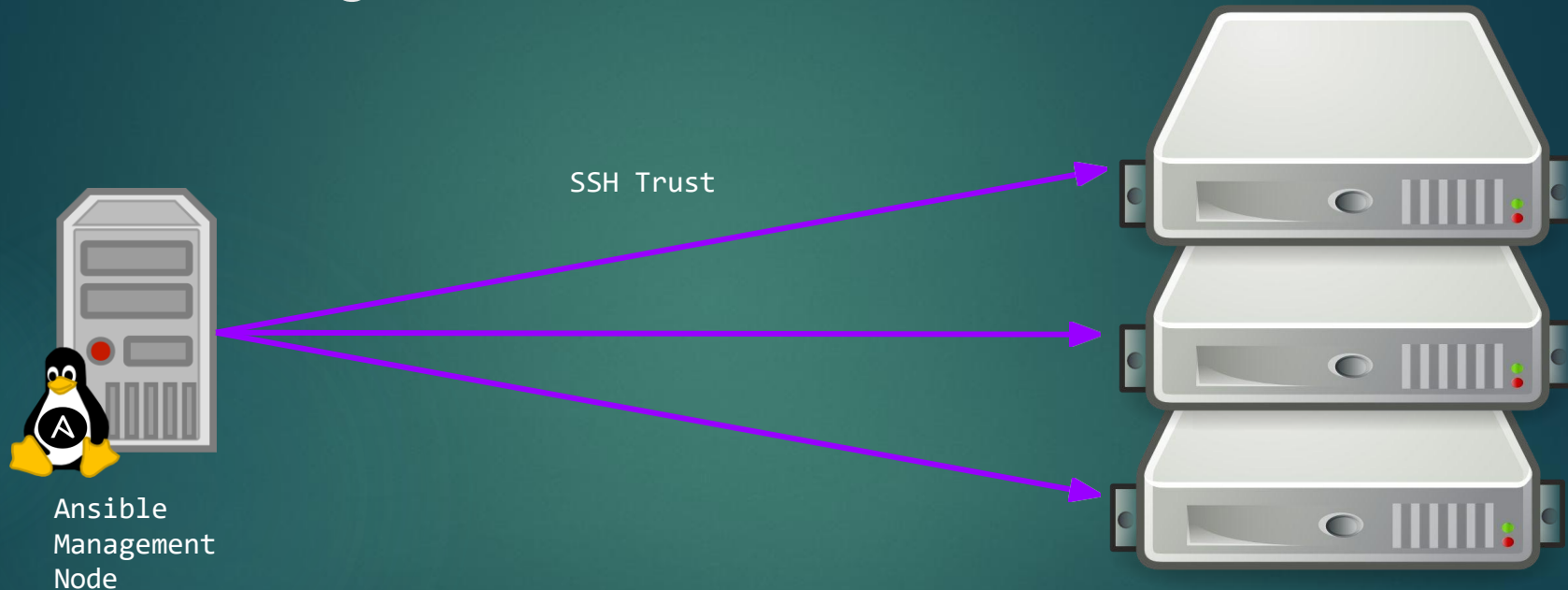
(Autogenerar) “documentación” sobre el despliegue de servicios

- + Reutilización

Administrar tanto centros de datos como entornos en la nube

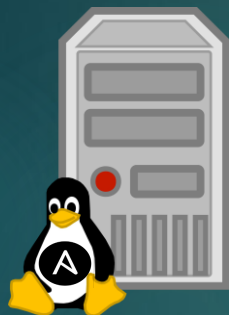
Aprovechar la oferta de diferentes proveedores de VPS y servidores dedicados

Ansible: ¿cómo funciona? Instalación



```
# apt-get install ansible
# ssh-keygen
# for i in n1 n2 n3 ; do ssh-copy-id root@$i; done
```

Ansible: ¿cómo funciona?: Creación de inventario y primeras configuraciones



Ansible
Management
Node

```
# vi /etc/ansible/hosts
[webservers]
n1 ansible_ssh_host=10.0.2.4
n2 ansible_ssh_host=10.0.2.6
[db]
n3 ansible_ssh_host=10.0.2.5
```

Si el acceso de root
está deshabilitado

Grupo de nodos

ansible_user=adm ansible_become=yes

```
# ansible all -m ping
```

```
# ansible all -m apt -a "name=zip state=latest"
```

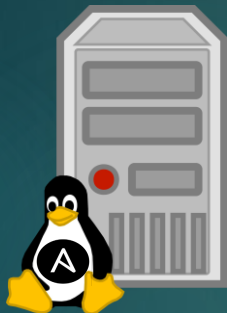
Donde

Módulo

Argumentos

ansible-doc -l
ansible-doc apt

Ansible: ¿cómo funciona?: Creación de inventario y primeras configuraciones – Playbooks (I)



Ansible
Management
Node

```
# vi mypb.yml
```

```
---
```

```
- hosts: webservers
```

```
vars:
```

```
  MyMessage: "Hello World"
```

```
tasks:
```

```
- name: ensure apache is in the latest version
```

```
  apt: name=apache2 state=latest
```

```
- name: Install website
```

```
  template: src=index.html.j2 dest=/var/www/html/index.html
```

Lenguaje YAML

```
# vi index.html.j2
```

```
<html><head><title>{{MyMessage}}</title>
```

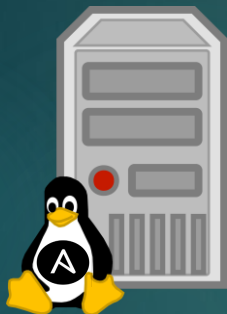
```
</head><body><p>{{MyMessage}}</p></body></html>
```

Plantilla Jinja2

```
# ansible-playbook mypb.yml
```

```
# wget -O- http://n2/
```

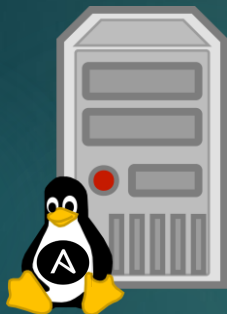
Ansible: ¿cómo funciona?: Creación de inventario y primeras configuraciones – Playbooks (II)



Ansible
Management
Node

```
# vi mypb2.yml ---
- hosts: webservers
  vars:
    vhosts:
      - { servername: "www.mydomain.com" , docroot: "/var/www/html" }
      - { servername: "www.mydomain2.com" , docroot: "/var/www/html" }
    tasks:
      - name: install a LAPP (Linux/Apache/PostgreSQL/PHP) environment      apt:
        name={{item}} state=latest update_cache=yes upgrade=yes with_items: [
          apache2, libapache2-mod-php, php-pgsql ]
      - name: Define Virtual Hosts
        template: src=default.j2 dest=/etc/apache2/sites-available/default2.conf
      - name: Install websites
        copy: src=index.html dest=/var/www/html/index.html
      - name: Enable websites
        shell: a2dissite -q 000-default && a2ensite -q default2 notify:
          restart apache
    handlers:
      - name: restart apache
        service: name=apache2 state=restarted
```

Ansible: ¿cómo funciona?: Creación de inventario y primeras configuraciones – Playbooks (III)



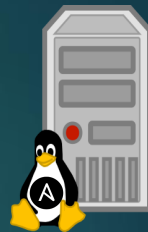
Ansible
Management
Node

```
# vi default2.j2
{% for vhost in vhosts %}
<VirtualHost *:80>
  ServerName {{ vhost.servername }}
  DocumentRoot {{ vhost.docroot }}
  {% if vhost.serveradmin is defined %}
    ServerAdmin {{ vhost.serveradmin }}
  {% endif %}
  <Directory "{{ vhost.docroot }}">
    AllowOverride All
    Options -Indexes +FollowSymLinks
    Order allow,deny
    Allow from all
  </Directory>
</VirtualHost>
{% endfor %}

# ansible-playbook mypb2.yml

# wget --header="Host: www.mydomain.com" -O- -q http://n2/
```

Ansible: ¿cómo funciona?: Creación de inventario y primeras configuraciones – Roles (I)



```
# vi install.yml
---
- name: Apply basic config to all systems
  hosts: all
  roles:
    - common

- name: Install PostgreSQL on db nodes

  hosts: db
  roles:
    - database

- name: Install Apache2 on webserver hosts:
  webserver
  roles:
    - webserver
```

Roles

```
install.yml
├── roles
│   ├── common
│   │   └── tasks
│   │       └── main.yml
│   ├── database
│   │   └── tasks
│   │       └── main.yml
│   └── webserver
│       ├── files
│       │   └── index.html
│       ├── handlers
│       │   └── main.yml
│       ├── tasks
│       │   └── main.yml
│       ├── templates
│       │   └── default2.j2
│       └── vars
│           └── main.yml
```

Ansible: ¿cómo funciona?: Creación de inventario y primeras configuraciones – Roles (II)

```
install.yml
roles
├── common
│   └── tasks
│       └── main.yml
├── database
│   └── tasks
│       └── main.yml
├── webserver
│   ├── files
│   │   └── index.html
│   ├── handlers
│   │   └── main.yml
│   ├── tasks
│   │   └── main.yml
│   ├── templates
│   │   └── default2.j2
│   └── vars
│       └── main.yml
```

```
# vi roles/common/tasks/main.yml
- name: update apt databases
  apt: update_cache=yes
- name: upgrade software
  apt: upgrade=dist
- name: Install base packages
  apt: name={{item}} state=latest
  with_items:
    - git
    - zip
```



Ansible: ¿cómo funciona?: Creación de inventario y primeras configuraciones – Roles (III)

```
install.yml
roles
├── common
│   └── tasks
│       └── main.yml
├── database
│   └── tasks
│       └── main.yml
├── webserver
│   ├── files
│   │   └── index.html
│   ├── handlers
│   │   └── main.yml
│   ├── tasks
│   │   └── main.yml
│   ├── templates
│   │   └── default2.j2
│   └── vars
│       └── main.yml
```

```
# vi roles/database/tasks/main.yml
- name: Install PostgreSQL
  apt: name={{item}} state=present install_recommends=no
  with_items:
    - postgresql
    - postgresql-client
```



Ansible: ¿cómo funciona?: Creación de inventario y primeras configuraciones – Roles (IV)

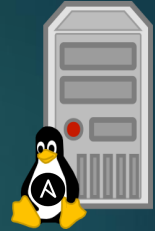


```
# vi roles/webserver/files/index.html
<html><head><title>
Hello World</title>
</head><body><p>Hello World</p></body></html>
```

```
├── install.yml
├── roles
│   ├── common
│   │   └── tasks
│   │       └── main.yml
│   ├── database
│   │   └── tasks
│   │       └── main.yml
│   └── webserver
│       ├── files
│       │   └── index.html
│       ├── handlers
│       │   └── main.yml
│       ├── tasks
│       │   └── main.yml
│       ├── templates
│       │   └── default2.j2
│       └── vars
│           └── main.yml
```

Ansible: ¿cómo funciona?: Creación de inventario y primeras configuraciones – Roles (V)

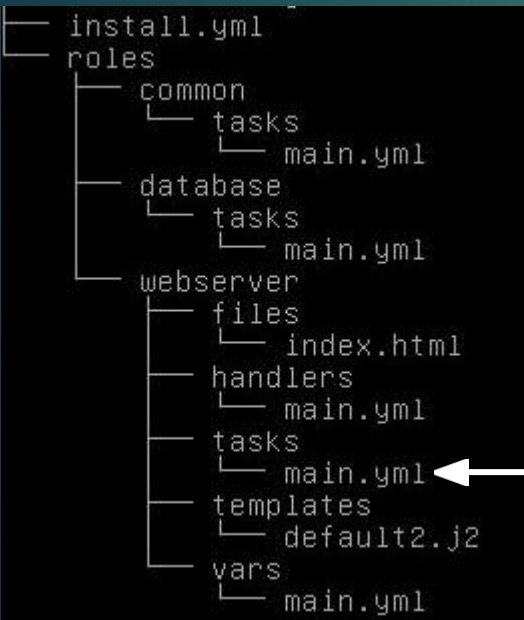
```
# vi roles/webserver/handlers/main.yml  
- name: restart apache  
  service: name=apache2 state=restarted
```



```
├── install.yml  
├── roles  
│   ├── common  
│   │   └── tasks  
│   │       └── main.yml  
│   ├── database  
│   │   └── tasks  
│   │       └── main.yml  
│   └── webserver  
│       ├── files  
│       │   └── index.html  
│       ├── handlers  
│       │   └── main.yml  
│       ├── tasks  
│       │   └── main.yml  
│       ├── templates  
│       │   └── default2.j2  
│       └── vars  
│           └── main.yml
```

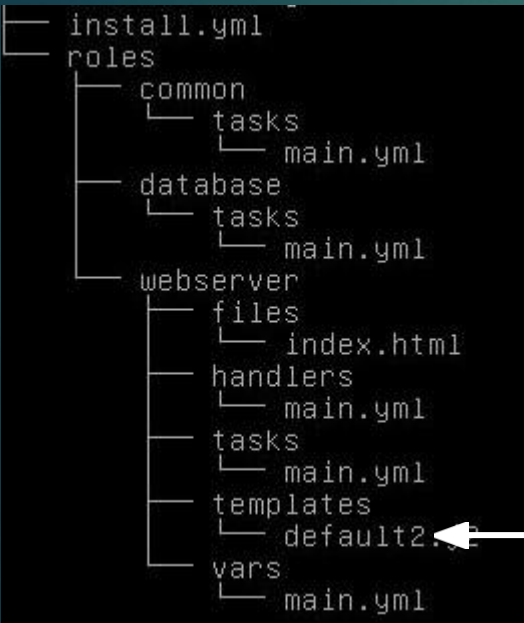
Ansible: ¿cómo funciona?: Creación de inventario y primeras configuraciones – Roles (VI)

```
# vi roles/webserver/tasks/main.yml
- name: install LAMP environment
  apt: name={{item}} state=latest update_cache=yes
  with_items: [ apache2, libapache2-mod-php, php-pgsql ]
- name: Define Virtual Hosts
  template: src=default2.j2
  dest=/etc/apache2/sites-available/default2.conf
- name: Install websites
  copy: src=../files/index.html dest=/var/www/html/index.html
- name: Enable websites
  shell: a2dissite 000-default && a2ensite default2
  notify: restart apache
```

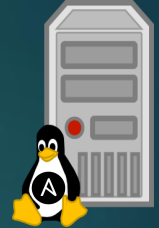


Ansible: ¿cómo funciona?: Creación de inventario y primeras configuraciones – Roles (VII)

```
# vi roles/webserver/templates/default.j2
{% for vhost in vhosts %}
<VirtualHost *:80>
  ServerName {{ vhost.servername }}
  DocumentRoot {{ vhost.docroot }}
  {% if vhost.serveradmin is defined %}
  ServerAdmin {{ vhost.serveradmin }}
  {% endif %}
  <Directory "{{ vhost.docroot }}">
    AllowOverride All
    Options -Indexes +FollowSymLinks
    Order allow,deny
    Allow from all
  </Directory>
</VirtualHost>
{% endfor %}
```



Ansible: ¿cómo funciona?: Creación de inventario y primeras configuraciones – Roles (VIII)



```
install.yml
roles
├── common
│   └── tasks
│       └── main.yml
├── database
│   └── tasks
│       └── main.yml
└── webserver
    ├── files
    │   └── index.html
    ├── handlers
    │   └── main.yml
    ├── tasks
    │   └── main.yml
    ├── templates
    │   └── default2.j2
    └── vars
        └── main.yml
```

▶ # vi roles/webserver/vars/main.yml vhosts:

```
↵ {servername: "www.mydomain.com", docroot: "/var/www/html"}
↵ {servername: "www.mydomain2.com", docroot: "/var/www/html" }
```

▶ # ansible-playbook install.yml

Ansible: ¿cómo funciona?: Creación de inventario y primeras configuraciones – Roles (IX) – Ansible Galaxy

```
# ansible-galaxy install bpresles.apache bpresles.php
```

```
# vi install2.yml
```

```
---
```

```
- name: Install webservers
  hosts: webservers
  roles:
    - bpresles.apache
    - bpresles.php
```

```
# ansible-playbook install2.yml
```

