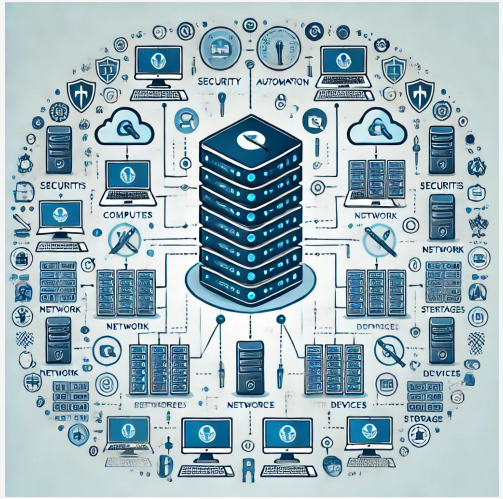
Cloud Camp

Gestión de la configuración

Sesión #19

Presentado por:
Wilmar Neiser Rengifo Güiza

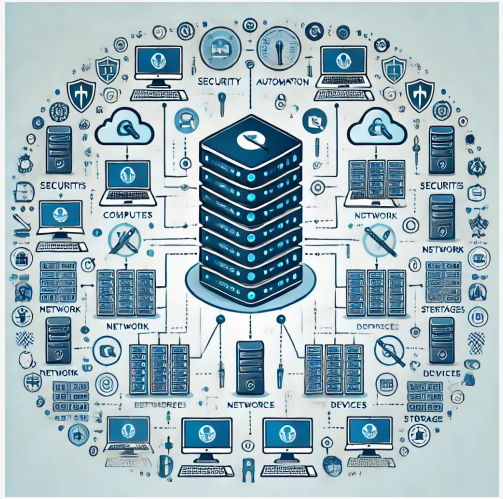
¿Qué es la gestión de la configuración?



La gestión de la configuración es el proceso de mantener sistemas informáticos y de TI en un estado deseado a través de la configuración uniforme, la documentación, y el control de cambios. Permite a los equipos de TI identificar y corregir desajustes en los sistemas para asegurar su rendimiento y seguridad. Su objetivo es centralizar la configuración, prevenir problemas operativos y asegurar el cumplimiento de normativas.

https://www.redhat.com/es/topics/automation/what-is-configuration-management

Ventajas:



- Consistencia y Uniformidad
- Mejora de la Seguridad
- Eficiencia Operativa
- Mejora del Cumplimiento y la Conformidad
- Reducción de Costos

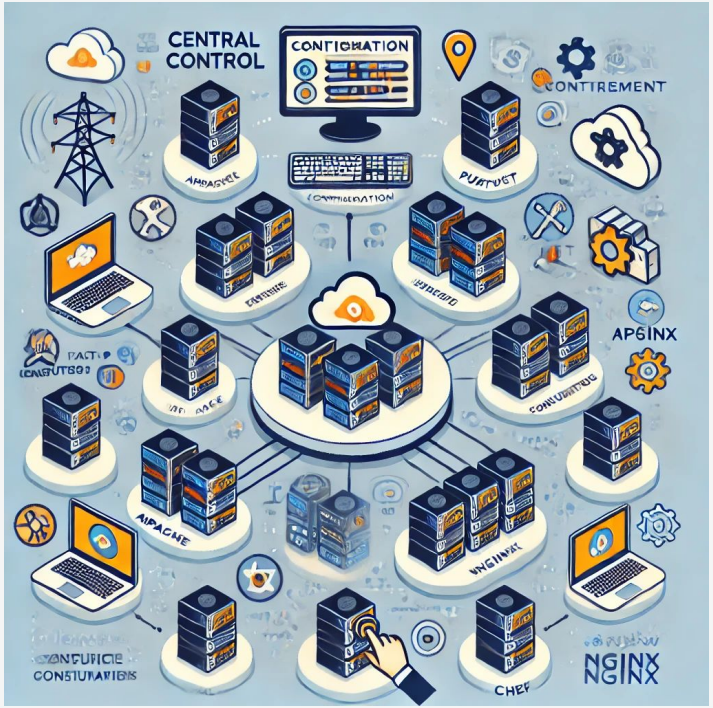
https://www.redhat.com/es/topics/automation/what-is-configuration-management

Caso de uso típico:

Escenario:
Una empresa de tecnología tiene varios servidores Linux que alojan aplicaciones web críticas. Estos servidores necesitan configuraciones consistentes, actualizaciones de seguridad regulares y mantenimiento continuo para asegurar su buen funcionamiento y seguridad.

Objetivos:

- Consistencia: Asegurar que todos los servidores tengan configuraciones uniformes.
- Seguridad: Aplicar parches de seguridad y actualizaciones de manera regular y eficiente.
- Eficiencia Operativa: Reducir el tiempo y esfuerzo necesarios para configurar y mantener los servidores.



Herramientas populares:

Playbooks

Manifests

Recipes

Ansible:

Características:

- Sin Agentes (Agentless)
- Playbooks en YAML
- Arquitectura Simple
- Modularidad y Extensibilidad
- Orquestación y Gestión de Aplicaciones
- Ansible Galaxy

¿Qué es lo minimo para usar Ansible?

- Nodo o máquina de control:
  - Python
  - Pip
  - Ansible
  - hosts.ini
- Máquinas o hosts a configurar:
  - Python
  - Ser accesibles vía SSH
- Un archivo de hosts.ini

Archivo host.ini

```
[webservers]
192.168.1.11
192.168.1.22

[webservers:vars]
ansible_ssh_private_key_file=/path/to/your/private/key.pem
ansible_user=ec2-user
```

Cloud Camp

Vamos a la consola >>>>>

Configurar un nuevo entorno

- Instalar Python
  a. sudo yum install python3
- Crear entorno virtual
  a. python3 -m venv environment
- Activar el entorno:
  a. source environment/bin/activate
- Instalar Ansible dentro del entorno:
  a. pip install ansible
- crear archivo hosts.ini
- crear la llave ssh
- hacer ping a los hosts:
  a. ansible all -m ping -i hosts.in

Comandos nativos (Sin playbook):

- ansible-playbook --syntax-check site.yml
- ansible all -m ping -i hosts
- ansible all -m shell -a 'uptime' -i hosts
- ansible all -m yum -a 'name=git state=present' -i hosts
- ansible all -m copy -a 'src=/path/local/file dest=/path/remote/file' -i hosts
- ansible all -m service -a 'name=httpd state=started' -i hosts
- ansible-inventory --list -i hosts
- ansible all -m setup -i hosts

Playbook

```
---
- name: Ensure Apache is installed and running on Amazon Linux
  hosts: webservers
  become: true
  tasks:
    - name: Install Apache
      ansible.builtin.yum:
        name: httpd
        state: present

    - name: Start and enable Apache
      ansible.builtin.service:
        name: httpd
        state: started
        enabled: true

    - name: Check Apache version
      ansible.builtin.command:
        cmd: httpd -v
      register: apache_version

    - name: Print Apache version
      ansible.builtin.debug:
        var: apache_version.stdout
```

Playbook

name: Una descripción del playbook.
hosts: Especifica el grupo de hosts en el archivo de inventario donde se ejecutarán las tareas (`webservers` en este caso).
become: Indica que las tareas deben ejecutarse con privilegios elevados (usando `sudo`).
tasks: Lista de tareas a ejecutar en los hosts.

- **Install Apache**: Usa el módulo `yum` para instalar Apache (`httpd`) en Amazon Linux.
- **Start and enable Apache**: Usa el módulo `service` para asegurar que Apache (`httpd`) esté iniciado y habilitado para iniciarse al arrancar el sistema.
- **Check Apache version**: Usa el módulo `command` para ejecutar el comando que muestra la versión de Apache y guarda el resultado en la variable `apache_version`.
- **Print Apache version**: Usa el módulo `debug` para imprimir la versión de Apache almacenada en `apache_version.stdout`.

Cloud Camp

¿Preguntas?