Cloud Camp

Ansible: Vault y facts

Sesión #22

Presentado por:
Wilmar Rengifo

Uso de Vault

```
# vault.yml
mysql_root_password: "password"
api_key: "your_api_key"
```

Uso de Vault

```
$ANSIBLE_VAULT;1.1;AES256
61626364656667686 96a6b6c6d6e6f707172737475767778797a0a31323334
353637383961626364656667686 96a6b6c6d6e6f707172737475767778797a
0a313233343536373839616263646566768696a6b6c6d6e6f707172737475
767778797a0a313233343536373839616263646566768696a6b6c6d6e6f70
```

Uso de Vault

- ansible-vault encrypt vault.yml
- ansible-playbook playbook.yml --ask-vault-pass
- ansible-playbook playbook.yml --vault-password-file .vault_pass.txt
- ansible-vault decrypt vault.yml
- ansible-vault edit vault.yml

Uso de Vault en AWS Secrets Manager

```
aws secretsmanager create-secret --name nombre_del_secreto --secret-string "palabraclavededecifrado"
```

```
aws secretsmanager get-secret-value --secret-id ansible_vault_password --query SecretString --output text
```

Recuperar secreto desde Ansible

```yaml
tasks:
  - name: Retrieve Vault password from AWS Secrets Manager using CLI
    shell: |
      aws secretsmanager get-secret-value --secret-id ansible_vault_password --query SecretString --output text
    register: vault_password_output
    changed_when: false
```

Ansible Facts

- ansible all -m setup
- ansible all -m setup -a 'filter=ansible_eth0'
- ansible all -m setup -a 'filter=ansible_eth0' | grep "ipv4.address"
- ansible all -m setup -a 'filter=ansible_hostname'

¿Ansible Facts o Hostvars?

Alcance de uso: Los facts están limitados al host actual, mientras que hostvars te permite acceder a las variables (incluyendo facts) de cualquier host definido en el inventario.

Acceso: Los facts se acceden directamente usando ansible_facts o variables predefinidas, mientras que hostvars se accede a través de un diccionario donde las claves son los nombres de los hosts.

Flexibilidad: hostvars es más flexible cuando necesitas cruzar información entre múltiples hosts dentro de un playbook.

Ansible Facts

Descripción: Los facts son variables que Ansible recopila automáticamente desde los hosts gestionados, como información sobre el sistema operativo, red, CPU, memoria, etc.
Obtención: Ansible recopila estos facts automáticamente cuando ejecuta un playbook o un módulo, a menos que se desactive con gather_facts: no.
Ejemplo de uso: Puedes acceder a los facts de un host en un playbook usando ansible_facts (o directamente la variable del fact).

Ansible Facts

```yaml
- hosts: all
  gather_facts: yes
  tasks:
    - name: Mostrar el sistema operativo y la IP
      debug:
        msg: "El sistema operativo es {{ ansible_facts['os_family'] }} y la IP es
```

HostVars

Descripción: Los facts son variables que Ansible recopila automáticamente desde los hosts gestionados, como información sobre el sistema operativo, red, CPU, memoria, etc.
Obtención: Ansible recopila estos facts automáticamente cuando ejecuta un playbook o un módulo, a menos que se desactive con gather_facts: no.
Ejemplo de uso: Puedes acceder a los facts de un host en un playbook usando ansible_facts (o directamente la variable del fact).

HostVars

```
- hosts: web
  gather_facts: yes
  tasks:
    - name: Obtener la IP de la base de datos
      debug:
        msg: "La IP del servidor de base de datos es {{ hostvars['db']['ansible_fac
```

Debug, ansible-inventory y setup

ansible -m debug

✅ Propósito: Se usa dentro de un playbook o directamente con ansible para ver el valor de variables o facts en tiempo de ejecución.

✅ Uso:

- Para inspeccionar variables en tiempo de ejecución.
- Para depurar valores dentro de un playbook.
- Necesita conexión SSH al host.

✅ Ejemplo:

ansible database -m debug -a "var=ansible_default_ipv4.address" -i inventory.yml

Muestra la dirección IP de los hosts en el grupo database.

📌 Características:

- Requiere conexión SSH.
- Funciona solo si ***gather_facts: yes*** está activado.
- Permite ver variables de facts y definidas manualmente.

Debug, ansible-inventory y setup

ansible-inventory

✅ Propósito: Inspecciona el inventario de Ansible antes de ejecutar un playbook.

✅ Uso:

- Para verificar qué hosts están en el inventario.
- Para ver variables definidas en group_vars y host_vars.
- No necesita conexión SSH a los hosts.

✅ Ejemplo:

ansible-inventory --list | jq

Muestra toda la estructura del inventario.

📌 Características:

- No requiere conexión SSH.
- No muestra facts recopilados automáticamente (ansible_default_ipv4.address no aparecerá si no está en host_vars).
- Muestra los hosts, grupos y variables del inventario.

Debug, ansible-inventory y setup

ansible -m setup

✅ Propósito: Recopila y muestra los facts de un host, incluyendo ansible_default_ipv4.address.

✅ Uso:

- Para obtener información detallada sobre un host en tiempo de ejecución.
- Se usa cuando gather_facts: no está desactivado en el playbook.
- Necesita conexión SSH.

✅ Ejemplo:

ansible database -m setup -a "filter=ansible_default_ipv4"

Muestra solo la información relacionada con ansible_default_ipv4.

📌 Características:

- Requiere conexión SSH.
- Muestra facts recopilados en tiempo de ejecución.
- Se puede filtrar la salida con filter= para obtener información específica.

Debug, ansible-inventory y setup

Característica	`ansible -m debug`	`ansible-inventory`	`ansible -m setup`
📌 **Propósito**	Mostrar variables y facts en ejecución	Inspeccionar el inventario antes de ejecutar	Obtener todos los facts del host
🔄 **Necesita conexión SSH?**	✅ Sí	❌ No	✅ Sí
🔍 **Permite ver facts del sistema?**	✅ Sí, si `gather_facts: yes`	❌ No	✅ Sí
🏗️ **Se usa antes de ejecutar un playbook?**	❌ No	✅ Sí	❌ No
🔧 **Se usa dentro de un playbook?**	✅ Sí	❌ No	❌ No
📌 **Muestra los hosts y grupos?**	❌ No	✅ Sí	❌ No