ACTIVIDAD 2 - GESTIÓN DE INVENTARIOS Y ARCHIVOS DE CONFIGURACIÓN

Cristóbal Suárez Abad

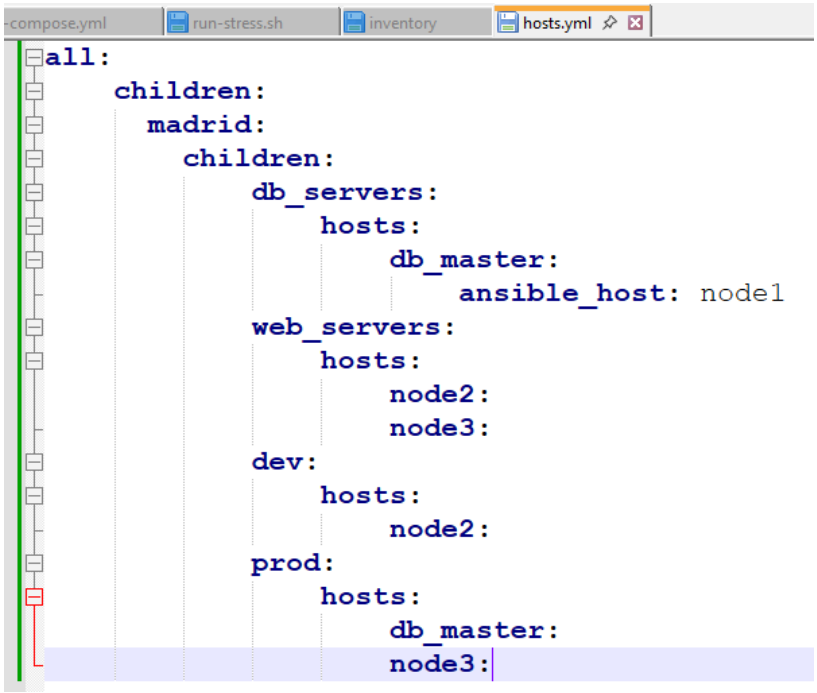
OPTATIVA - 2º ASIR

Contenido

Fase 1: Inventario YAML Estructurado (25 min)

Crea un archivo llamado `hosts.yml`. Debes definir los hosts `node1`, `node2` y `node3` pero asignándoles **alias funcionales** y organizándose en una matriz lógica:

- **Grupos Funcionales:**
  - `db_servers`: Contiene a `node1`.
  - `web_servers`: Contiene a `node2` y `node3`.
- **Grupos de Entorno:**
  - `dev`: Contiene a `node2`.
  - `prod`: Contiene a `node1` y `node3`.
- **Grupo Geográfico:**
  - `madrid`: Contiene a todos los servidores (usando el grupo especial `all`).
- El `node1` debe definirse en el inventario con un alias, por ejemplo `db_master`, de forma que `ansible_host` apunte a `node1`.



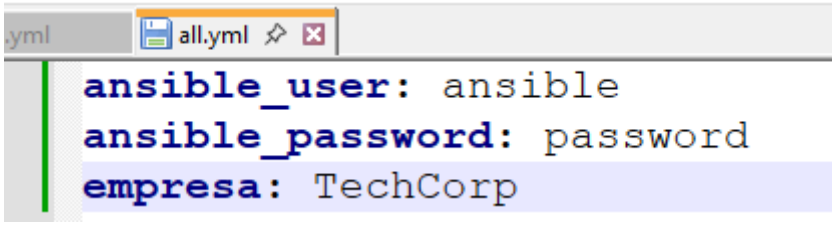
```
all:
  children:
    madrid:
      children:
        db_servers:
          hosts:
            db_master:
              ansible_host: node1
        web_servers:
          hosts:
            node2:
            node3:
        dev:
          hosts:
            node2:
        prod:
          hosts:
            db_master:
            node3:
```

NOTA: Los archivos que se están mostrando se llevaron más adelante a la estructura del contenedor ansible-control. Lo que se muestra en las imágenes es la edición desde el host.

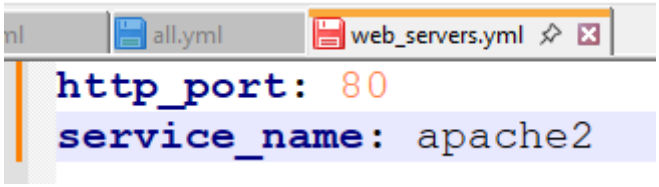
Fase 2: Desacople de Variables

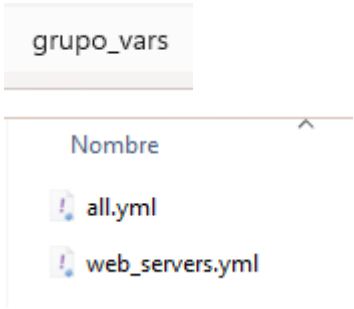
Una buena práctica es no ensuciar el archivo de inventario con variables. Vamos a usar la estructura de directorios nativa de Ansible.

1. **Variables Globales (`group_vars/all.yml`):**
   - Define aquí el usuario de conexión genérico (`ansible`) y la contraseña (`password`).
   - Define una variable `empresa: TechCorp`.

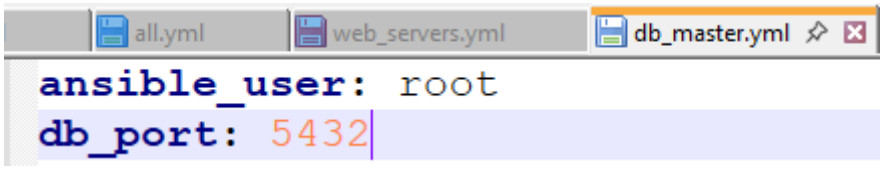


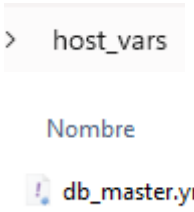
2. **Variables Específicas de Web (`group_vars/web_servers.yml`):**
   - Define `http_port: 80`.
   - Define `service_name: apache2`.





3. **Excepción de Seguridad (`host_vars/db_master.yml`):**
    - Aquí sobreescribiremos la conexión global. Configura este archivo para que la conexión a este host específico se haga con el usuario `root`
    - Define `db_port: 5432`.



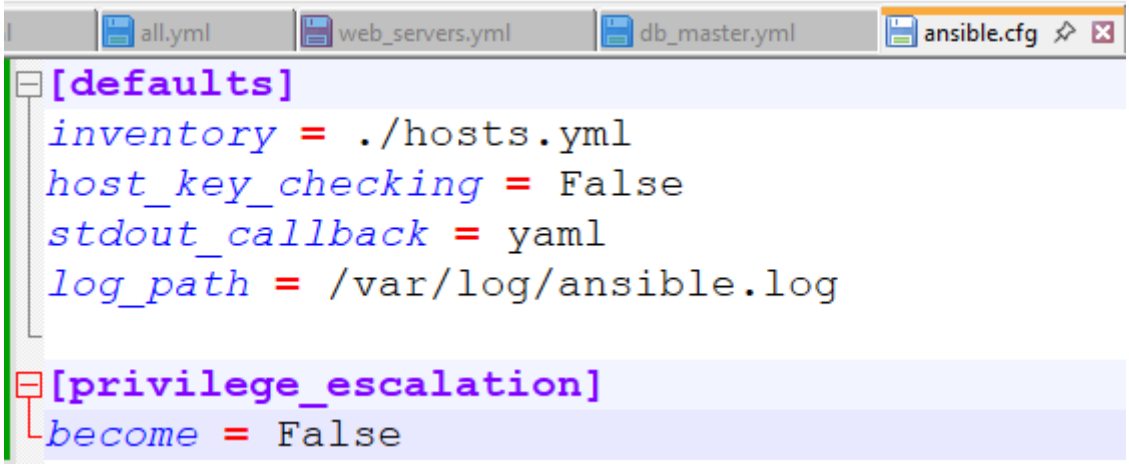


Fase 3: Configuración y Tuning

Configura el archivo `ansible.cfg` que creaste al principio para orquestar todo esto.

Tarea: Edita `ansible.cfg` con las siguientes directivas:

1. Apuntar al inventario `hosts.yml` por defecto.
2. Desactivar `host_key_checking`.
3. Cambiar el formato de salida a `yaml` para leer mejor los resultados.
4. Habilitar el log de las operaciones en `/var/log/ansible.log`



Fase 4: Patrones de Selección y Verificación (20 min)

Ahora probaremos la potencia de los "Patterns" de Ansible para atacar intersecciones de grupos.

Ejecuta y documenta los siguientes comandos:

1. **Prueba de variables:** Usa el módulo debug para ver qué puerto usaría cada servidor y muestra un mensaje con nombre del servicio y el puerto, para los servidores web:

   (Verifica que Node1 NO responde a esto y que Node2/3 muestran los datos correctos).

Desde dentro del contenedor ansible-control:

docker exec -it ansible-control bash

Nos posicionamos en el directorio “proyecto2” y ejecutamos:

ansible web_servers -m debug -a "msg='El servicio {{ service_name }} corre en el puerto {{ http_port }}'"

```
root@control:/ansible/proyecto2# ansible web_servers -m debug -a "msg='
El servicio {{ service_name }} corre en el puerto {{ http_port }}'"
node2 | SUCCESS => {
    "msg": "El servicio apache2 corre en el puerto 80"
}
node3 | SUCCESS => {
    "msg": "El servicio apache2 corre en el puerto 80"
}
root@control:/ansible/proyecto2# cd ..
```

Éxito en nodo 2 y 3. El nodo 1 no responde.

2. **La intersección (AND):** Queremos hacer ping **solo** a los servidores Web que además son de Producción. *Pista: La sintaxis es* `grupo1:&grupo2`.
   Bash
   (Debería responder SOLO el `node3`*).*

ansible "web_servers:&prod" -m ping

```
root@control:/ansible/proyecto2# ansible "web_servers:&prod" -m ping
node3 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

3. **La exclusión (NOT):** Queremos listar las variables de todos los servidores que **NO** son de base de datos.

ansible 'all:!db_servers' -m setup -a "filter=ansible_local"
Dentro de todos (“all”) y luego lo contrario a “db_servers”, por eso usamos la exclamación. IMPORTANTE: usar comillas simples, con las dobles nos da problemas. Nos muestra los nodos 2 y 3, que son nodos web.

```
root@control:/ansible/proyecto2# ansible 'all:!db_servers' -m setup -a
"filter=ansible_local"
node3 | SUCCESS => {
    "ansible_facts": {
        "ansible_local": {},
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false
}
node2 | SUCCESS => {
    "ansible_facts": {
        "ansible_local": {},
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false
}
```

4. **Verificación de privilegios:** Comprueba que node1 está conectando como root y los demás como ansible.

Podemos usar un solo comando para llamar a todos. Pero nos da problemas con el node1 por problemas de SSH:

ansible all -m shell -a "whoami"

```
root@control:/ansible/proyecto2# ansible all -m shell -a "whoami"
node2 | CHANGED | rc=0 >>
ansible
node3 | CHANGED | rc=0 >>
ansible
db_master | UNREACHABLE! => {
    "changed": false,
    "msg": "Invalid/incorrect password: Permission denied, please try again.",
    "unreachable": true
}
```

Podemos usar en su lugar:

ansible db_master -m debug -a "var=ansible_user"

```
root@control:/ansible/proyecto2# ansible db_master -m debug -a "var=ansible_user"
db_master | SUCCESS => {
    "ansible_user": "root"
}
```

```
root@control:/ansible/proyecto2# ansible node2 -m debug -a "var=ansible_user"
node2 | SUCCESS => {
    "ansible_user": "ansible"
}
root@control:/ansible/proyecto2# ansible node3 -m debug -a "var=ansible_user"
node3 | SUCCESS => {
    "ansible_user": "ansible"
}
```

Entregable Final

El alumno debe subir un archivo comprimido con:

1. La estructura de directorios (group_vars, host_vars, ansible.cfg, hosts.yml).

Dentro del contenedor de control creamos el archivo comprimido:

tar -czvf /ansible/proyecto2.tar.gz .

```
root@control:/ansible/proyecto2# tar -czvf /ansible/proyecto2.tar.gz .
./
./inventory
./ansible.cfg
./hosts.yml
./desktop.ini
./host_vars/
./host_vars/db_master.yml
./group_vars/
./group_vars/all.yml
./group_vars/web_servers.yml
root@control:/ansible/proyecto2# ls -l
total 24
-rwxr-xr-x 1 root root  227 Dec 21 16:11 ansible.cfg
-rwxr-xr-x 1 root root  246 Dec 10 17:10 desktop.ini
drwxr-xr-x 2 root root 4096 Dec 21 16:10 group_vars
drwxr-xr-x 2 root root 4096 Dec 21 17:23 host_vars
-rw-r--r-- 1 root root  337 Dec 21 16:13 hosts.yml
-rwxr-xr-x 1 root root  103 Dec 10 18:04 inventory
root@control:/ansible/proyecto2# cd ..
root@control:/ansible# ls -l
total 8
drwxr-xr-x 4 root root 4096 Dec 21 17:23 proyecto2
-rw-r--r-- 1 root root  918 Dec 21 17:36 proyecto2.tar.gz
```

Después desde fuera, lo copiamos al host:

docker cp ansible-control:/ansible/proyecto2.tar.gz ./entregable_final_techcorp.tar.gz

```
D:\2°_ASIR\Optativa\Tema 04\Actividad 1 - Instalación de laboratorio Ansible\LaboratorioAnsible>docker cp ansible-control:/ansible/proyecto2.tar.gz
 ./entregable_final_techcorp.tar.gz
Successfully copied 2.56kB to D:\2°_ASIR\Optativa\Tema 04\Actividad 1 - Instalación de laboratorio Ansible\LaboratorioAnsible\entregable_final_tech
corp.tar.gz
```

2. Un archivo de texto con la demostración que todo lo pedido funciona correctamente