ACTIVIDAD 4
MODULARIZACIÓN PROFESIONAL Y ECOSISTEMA GALAXY

Cristóbal Suárez Abad

OPTATIVA - 2º ASIR

Índice:

Actividad 4: Modularización Profesional y Ecosistema Galaxy 2
El Escenario 2
Requerimientos Técnicos 2
Instalar un rol que contenga apache 3
Crea un role nuevo llamado security_hardening 5
Archivos del Rol: 6
Tasks: 6
Handlers: 8
Defaults: Valores por defecto 9
Plantillas Jinja2 10
Archivo "sites.yml" 11
Ejecución del Playbook 12

Actividad 4: Modularización Profesional y Ecosistema Galaxy

El Escenario

TechCorp quiere estandarizar su despliegue. Ya no quieren playbooks "sueltos". Tu misión es crear un sistema donde la seguridad sea un componente (rol propio) y el servidor web sea otro componente (rol de la comunidad).

Requerimientos Técnicos

1. Uso de Ansible Galaxy

- Instala un rol verificado para gestionar el servidor web.
- Cambia el puerto por defecto usando variables en el playbook maestro.

2. Creación del Rol `security_hardening`

Crea un role nuevo llamado security_hardening

- **Tasks:** Migrar la creación de usuarios y la configuración de SSH de la Actividad 3.
- **Handlers:** Migrar el reinicio de SSH a `roles/security_hardening/handlers/main.yml`.
- **Defaults:** Definir el `log_path` en `defaults/main.yml` para que pueda ser sobrescrito fácilmente.

3. Uso de Plantillas Jinja2

En lugar de usar el módulo `copy` para el log, deben usar un **Template**.

- **Tarea:** Crear un archivo `roles/security_hardening/templates/audit_report.j2`.
- **Contenido:** El archivo debe decir: *"Informe generado por Ansible para el host {{ inventory_hostname }}. El uptime actual es: {{ resultado_uptime.stdout }}"*.
- **Acción:** Usar el módulo `template` en las tareas del rol para generar el archivo en los nodos.

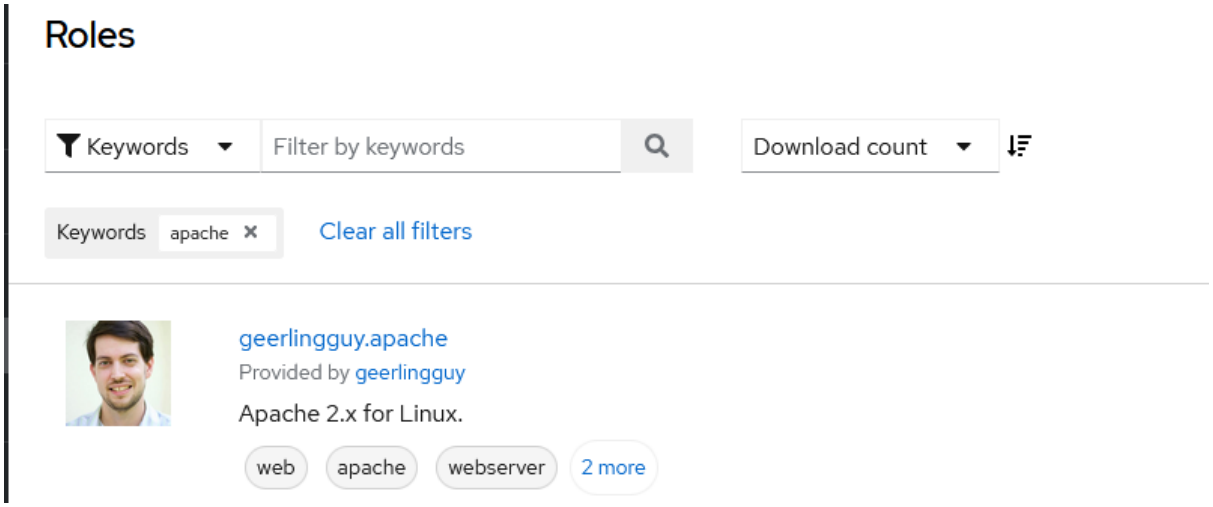
4. Playbook Maestro `site.yml`

- Crear un único punto de entrada que configure todo el centro de datos.
- Realizar ejecución del playbook y explicar la salida

Instalar un rol que contenga apache

Nos vamos a https://galaxy.ansible.com/ui/standalone/roles/

Y nos descargamos uno de los más usados por los usuarios:



ansible-galaxy role install geerlingguy.apache

Lo instalamos en ansible-control

```
root@control:/ansible# ansible-galaxy role install geerlingguy.apache
Starting galaxy role install process
- downloading role 'apache', owned by geerlingguy
- downloading role from https://github.com/geerlingguy/ansible-role-apache/archive/4.2.0.tar.gz
- extracting geerlingguy.apache to /root/.ansible/roles/geerlingguy.apache
- geerlingguy.apache (4.2.0) was installed successfully
root@control:/ansible#
```

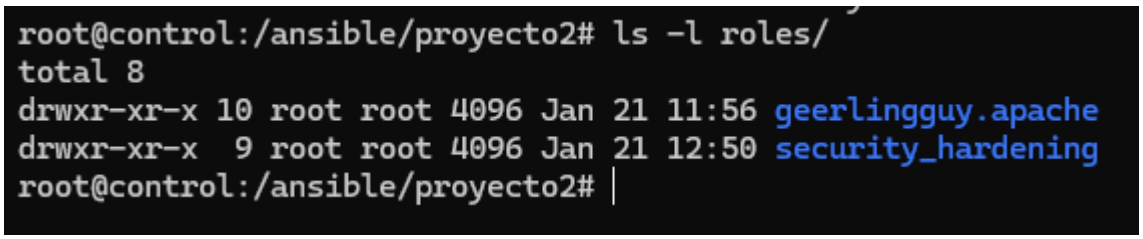
Para ver los roles instalados: ansible-galaxy list

```
- geerlingguy.apache (4.2.0) was installed successfully
root@control:/ansible# ansible-galaxy list
# /root/.ansible/roles
- geerlingguy.apache, 4.2.0
[WARNING]: - the configured path /usr/share/ansible/roles does not exist.
[WARNING]: - the configured path /etc/ansible/roles does not exist.
root@control:/ansible#
```

Por normal general se instalan en el directorio que se ve en la imagen, pero para nuestro caso lo vamos a instalar en un directorio del proyecto2.

Creamos un directorio de roles (“mkdir”) y lo instalamos ahí.

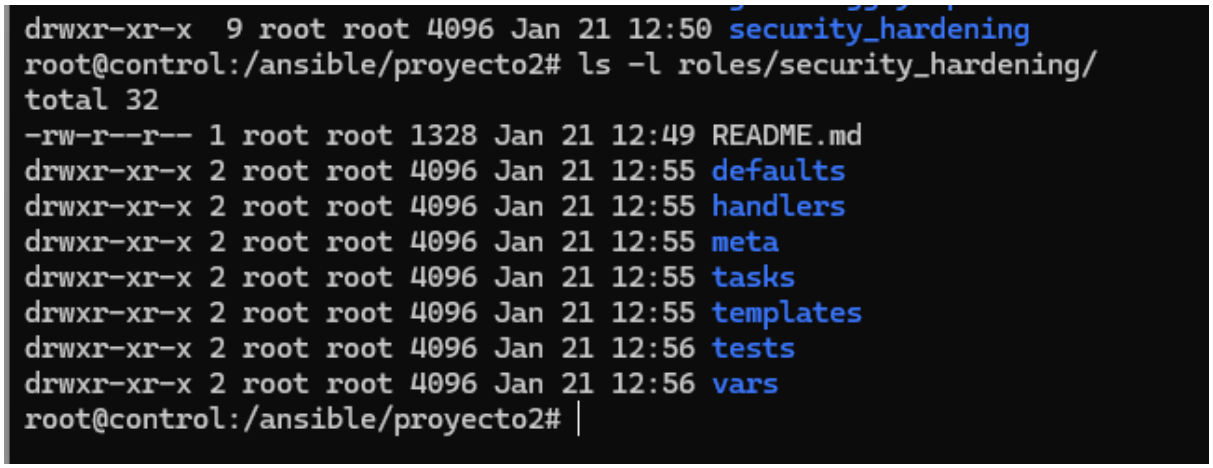
ansible-galaxy install geerlingguy.apache -p ./roles



Crea un role nuevo llamado security_hardening

Este rol también lo creamos en el mismo directorio que el anterior. En este caso nos colocamos

ansible-galaxy role init security_hardening



Como puedes observar, se ve toda la estructura básica del nuevo rol.

Archivos del Rol:

Tasks:

roles/security_hardening/tasks/main.yml

```
---
- name: Registrar uptime del sistema
  command: uptime
  register: resultado_uptime

- name: Crear usuarios de auditoría
  user:
    name: "{{ item }}"
    groups: sudo
    append: yes
  loop: "{{ audit_team }}"

- name: Configurar Hardening SSH
  lineinfile:
    path: /etc/ssh/sshd_config
    regexp: '^PermitRootLogin'
    line: 'PermitRootLogin no'
  notify: Reiniciar SSH

- name: Generar informe de auditoría desde plantilla
  template:
    src: audit_report.j2
    dest: "{{ log_path }}"
```

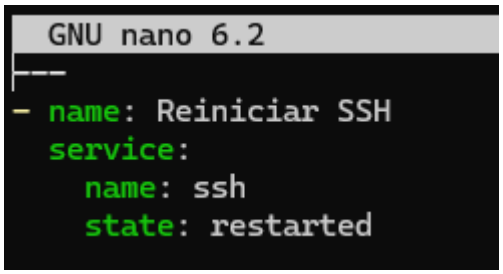
En este caso, se usa “audit_report,j2” como plantilla para generar el informe. Buscará este archivo en la carpeta templates/ del rol.

```
  GNU nano 6.2                                    roles/security_hard
---
- name: Registrar uptime del sistema
  command: uptime
  register: resultado_uptime

- name: Crear usuarios de auditor  a
  user:
    name: "{{ item }}"
    groups: sudo
    append: yes
  loop: "{{ audit_team }}"

- name: Configurar Hardening SSH
  lineinfile:
    path: /etc/ssh/sshd_config
    regexp: '^PermitRootLogin'
    line: 'PermitRootLogin no'
  notify: Reiniciar SSH

- name: Generar informe de auditor  a desde plantilla
  template:
    src: audit_report.j2
    dest: "{{ log_path }}"
```

Handlers:

roles/security_hardening/handlers/main.yml

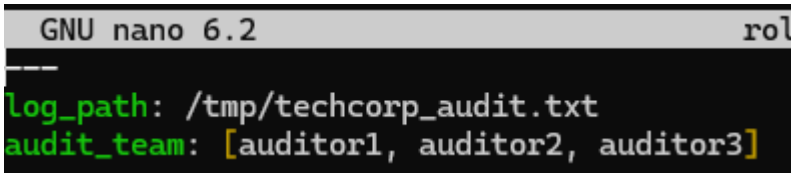
```
---
- name: Reiniciar SSH
  service:
    name: ssh
    state: restarted
```



Defaults: Valores por defecto

roles/security_hardening/defaults/main.yml

---
log_path: /tmp/techcorp_audit.txt
audit_team: [auditor1, auditor2, auditor3]



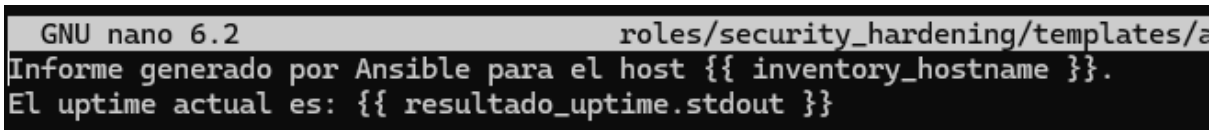
En Ansible, la carpeta “**defaults**” se utiliza para definir variables.

Plantillas Jinja2

Creamos un directorio “templates” dentro del directorio del rol “security_hardening” y luego:

roles/security_hardening/templates/audit_report.j2

Informe generado por Ansible para el host {{ inventory_hostname }}.
El uptime actual es: {{ resultado_uptime.stdout }}



{{ inventory_hostname }}: Contiene el nombre del host (los nodos).

{{ resultado_uptime.stdout }}: El registro del comando “uptime”.

Archivo “sites.yml”

Ahora, en el directorio principal del proyecto, creamos el archivo “**sites.yml**”:

```
root@control:/ansible/proyecto2# ls -l site.yml
-rw-r--r-- 1 root root 283 Jan 21 12:50 site.yml
root@control:/ansible/proyecto2#
```

```
---
- name: Configuración Global de Seguridad
  hosts: all
  become: yes
  roles:
    - security_hardening
```

Se aplica a todos los “hosts”, se usa sudo y se llama al rol “security_hardening”.

```
- name: Despliegue de Servidores Web
  hosts: web_servers
  become: yes
  vars:
    apache_listen_port: 8080
  roles:
    - geerlingguy.apache
```

En este caso solo se aplica a los “**hosts**” del grupo “**web_servers**”. Se añade una variable para que se cambie el puerto que usará Apache. Se llama al rol “**geerlingguy.apache**”.

```
  GNU nano 6.2
---
- name: Configuraci  n Global de Seguridad
  hosts: all
  become: yes
  roles:
    - security_hardening

- name: Despliegue de Servidores Web
  hosts: web_servers
  become: yes
  vars:
    apache_listen_port: 8080
  roles:
    - geerlingguy.apache
```

Ejecución del Playbook

ansible-playbook site.yml

1. **PLAY [Configuración Global de Seguridad]**

Este bloque representa la ejecución del rol "**security_hardening**".

```
root@control:/ansible/proyecto2# ansible-playbook site.yml

PLAY [Configuración Global de Seguridad] ***************************************************************************

TASK [Gathering Facts] *********************************************************************************************
ok: [db_master]
ok: [node2]
ok: [node3]
```

Ansible conecta por SSH a los nodos para recoger información técnica (IPs, sistema operativo, memoria). Es automático.

```
TASK [security_hardening : Registrar uptime del sistema] **
changed: [node2]
changed: [db_master]
changed: [node3]
```

Ejecución del comando **uptime** y guardado en un registro. Aparece como **changed**.

```
TASK [security_hardening : Crear usuarios de auditoría]
ok: [node2] => (item=auditor1)
ok: [db_master] => (item=auditor1)
ok: [node3] => (item=auditor1)
ok: [node2] => (item=auditor2)
ok: [db_master] => (item=auditor2)
ok: [node3] => (item=auditor2)
ok: [node2] => (item=auditor3)
ok: [node3] => (item=auditor3)
ok: [db_master] => (item=auditor3)
```

Aparece como "**ok**". Eso es porque los usuarios auditor1, auditor2 y auditor3 ya existían en los nodos desde la Actividad 3, por lo que Ansible no tuvo que hacer nada.

```
TASK [security_hardening : Configurar Hardening SSH]
ok: [node2]
ok: [node3]
ok: [db_master]
```

Configurar Hardening SSH: También **ok**. La línea de "**PermitRootLogin no**" ya estaba aplicada.

```
TASK [security_hardening : Generar informe de auditoría desde plantilla] ***
changed: [node2]
changed: [node3]
changed: [db_master]
```

Generar informe de auditoría desde plantilla: Aparece como **changed**. Ansible ha tomado la plantilla "**audit_report.j2"**, le ha inyectado el valor del **uptime** actual y ha sobrescrito el archivo en /tmp/techcorp_audit.txt.

2. PLAY [Despliegue de Servidores Web]

Y este bloque corresponde al rol "**geerlingguy.apache**". A partir de aquí se hace todo el proceso de instalación de Apache en los nodos que son servidores web.

El mismo "**Gathering Facts**" que el anterior:

```
PLAY [Despliegue de Servidores Web] ***

TASK [Gathering Facts] ****************
ok: [node3]
ok: [node2]
```

"Include OS-specific variables": Se comprueba la versión del sistema operativo y se cargan las variables adecuadas para esa versión.

```
TASK [geerlingguy.apache : Include OS-specific variables.] *
ok: [node2]
ok: [node3]
```

Un vistazo a los directorios muestra diferentes archivos de variables y tareas para diferentes sistemas operativos:

```
root@control:/ansible# cat proyecto2/roles/geerlingguy.apache/vars/
AmazonLinux.yml  Debian.yml       RedHat.yml       Solaris.yml      Suse.yml         apache-22.yml    apache-24.yml
root@control:/ansible# cat proyecto2/roles/geerlingguy.apache/tasks/
configure-Debian.yml   configure-Solaris.yml  main.yml              setup-RedHat.yml       setup-Suse.yml
configure-RedHat.yml   configure-Suse.yml     setup-Debian.yml      setup-Solaris.yml
```

Como no estamos usando "**Amazon Linux**", este paso se lo salta.

```
TASK [geerlingguy.apache : Include variables for Amazon Linux.] *
skipping: [node2]
skipping: [node3]
```

Se definen los paquetes de **Apache** que se van a usar.
Apache solo se instala en los nodos 2 y 3, que son del grupo "**web_servers**".
Se hace un "**apt update**" antes de la instalación. Luego se instala **Apache**.

```
TASK [geerlingguy.apache : Define apache_packages.] ***************************************
ok: [node2]
ok: [node3]

TASK [geerlingguy.apache : include_tasks] **************************************************
included: /ansible/proyecto2/roles/geerlingguy.apache/tasks/setup-Debian.yml for node2, node3

TASK [geerlingguy.apache : Update apt cache.] **********************************************
changed: [node3]
[WARNING]: Updating cache and auto-installing missing dependency: python3-apt
ok: [node2]

TASK [geerlingguy.apache : Ensure Apache is installed on Debian.] ******************************
changed: [node2]
changed: [node3]

TASK [geerlingguy.apache : Get installed version of Apache.] ***********************************
ok: [node2]
ok: [node3]

TASK [geerlingguy.apache : Create apache_version variable.] ************************************
ok: [node2]
```

Se instala la versión de Apache establecida en la variable. Y se incluye la tarea de configuración de Apache.

```
TASK [geerlingguy.apache : Create apache_version variable.] ************************************
ok: [node2]
ok: [node3]

TASK [geerlingguy.apache : Include Apache 2.2 variables.] **************************************
skipping: [node2]
skipping: [node3]

TASK [geerlingguy.apache : Include Apache 2.4 variables.] **************************************
ok: [node2]
ok: [node3]

TASK [geerlingguy.apache : Configure Apache.] **********************************************
included: /ansible/proyecto2/roles/geerlingguy.apache/tasks/configure-Debian.yml for node2, node3
```

Se cambia el puerto a 8080 y se incluyen dos mods.

```
TASK [geerlingguy.apache : Configure Apache.] **********************************
changed: [node2] => (item={'regexp': '^Listen ', 'line': 'Listen 8080'})
changed: [node3] => (item={'regexp': '^Listen ', 'line': 'Listen 8080'})

TASK [geerlingguy.apache : Enable Apache mods.] ********************************
changed: [node2] => (item=rewrite)
changed: [node3] => (item=rewrite)
changed: [node2] => (item=ssl)
changed: [node3] => (item=ssl)
```

Se termina de hacer la configuración de Apache: poner el archivo "**.conf**" del vhost con un enlace simbólico en el directorio de "**sites-enabled**", quitar el vhost que viene de fábrica, poner **Apache** para que se inicie con el encendido de la máquina y reiniciar Apache para que funcionen los cambios hechos.

```
TASK [geerlingguy.apache : Add vhost symlink in sites-enabled.] ********************
changed: [node2]
changed: [node3]

TASK [geerlingguy.apache : Remove default vhost in sites-enabled.] *****************
skipping: [node2]
skipping: [node3]

TASK [geerlingguy.apache : Ensure Apache has selected state and enabled on boot.] **
changed: [node2]
changed: [node3]

RUNNING HANDLER [geerlingguy.apache : restart apache] ******************************
changed: [node2]
changed: [node3]
```

El "**Play Recap**": Como la mayoría de los cambios son para los "**web_servers**", el "**db_master**" apenas tiene cambios.

```
PLAY RECAP *********************************************************************************************************
db_master                  : ok=5    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
node2                      : ok=21   changed=9    unreachable=0    failed=0    skipped=7    rescued=0    ignored=0
node3                      : ok=21   changed=10   unreachable=0    failed=0    skipped=7    rescued=0    ignored=0
```