# Universidad ORT Uruguay

# Facultad de Ingeniería: Escuela de Tecnología

# Taller Servidores Linux

Andrés Acosta Andrade N.º 241131

Martina De Leon Balbiani N.º 254416

08/08/2025

Administración de servidores Linux

Declaración de auditoría

Nosotros, Andrés Acosta Andrade y Martina De Leon Balbiani, declaramos que el trabajo que se presenta en esa obra es de nuestra propia mano. Podemos asegurar que:

• La obra fue producida en su totalidad mientras cursabamos la materia Administración de Servidores Linux;

• Cuando hemos consultado el trabajo publicado por otros, lo hemos atribuido con claridad;

• Cuando hemos citado obras de otros, hemos indicado las fuentes. Con excepción de estas citas, la obra es enteramente nuestra;

• En la obra, hemos acusado recibo de las ayudas recibidas;

• Cuando la obra se basa en trabajo realizado conjuntamente con otros, hemos explicado claramente qué fue contribuido por otros, y qué fue contribuido por nosotros;

• Ninguna parte de este trabajo ha sido publicada previamente a su entrega, excepto donde se han realizado las aclaraciones correspondientes.

Andres Acosta Andrade

Martina De Leon Balbiani

Administración de servidores Linux

Objetivo

Aplicar los conocimientos básicos de Ansible sobre dos distribuciones Linux:

- CentOS Stream 9
- Ubuntu 24.04

Índice

# 1. Instalación de servidores

## 1.1. bastion

Se crea el servidor bastión usando la distribución CentOS Stream 9 con las siguientes características:

- 2 CPU
- 4 GB de RAM
- 20 GB de disco
- 2 interfaces de red, una en NAT y la otra interna.

La selección de software usada es Server with GUI. Tiene root deshabilitado, y sysadmin como usuario administrador con la password del curso.

Las particiones son como sigue:

| Punto de montaje | Capacidad |
|---|---|
| /boot | 1 |
| / | 10 |
| /var | 5 |
| SWAP | 4 |
| Total | 20 |

```
[sysadmin@bastion taller_linux2025]$ sudo lsblk
NAME             MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda                8:0    0   20G  0 disk
├─sda1             8:1    0    1G  0 part /boot
└─sda2             8:2    0   19G  0 part
  ├─cs_vbox-root 253:0    0   10G  0 lvm  /
  ├─cs_vbox-swap 253:1    0    4G  0 lvm  [SWAP]
  └─cs_vbox-var  253:2    0    5G  0 lvm  /var
sr0               11:0    1 57.4M  0 rom  /run/media/sysadmin/VBox_GAs_7.1.6
```

Se actualiza el servidor con `sudo dnf upgrade` tras la instalación.

Activamos la interfaz interna (enp0s8) y le asignamos la dirección ipv4 192.168.1.1/24

```
sudo nmcli connection add con-name red_interna ifname enp0s8 type ethernet ip4 192.168.1.1/24 ipv6.method disabled
```

## 1.2. centos01

Se crea el servidor centos01 usando la distribución CentOS Stream 9 con las siguientes características:

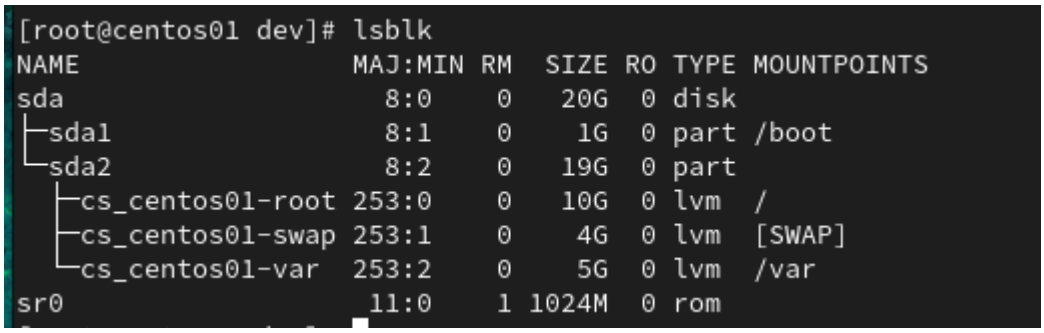
- 1 CPU

- 2 GB de RAM
- 20 GB de disco
- 2 interfaces de red, una en NAT y la otra interna.

La selección de software usada es Minimal, root deshabilitado, y sysadmin como usuario administrador con la password del curso.

Las particiones son idénticas a bastion:

| Punto de montaje | Capacidad |
|---|---|
| /boot | 1 |
| / | 10 |
| /var | 5 |
| SWAP | 4 |
| Total | 20 |




Activamos la interfaz interna (enp0s8) y le asignamos la dirección ipv4 192.168.1.1/24

```
sudo nmcli connection add con-name red_interna ifname enp0s8 type ethernet ip4 192.168.1.2/24 ipv6.method disabled
```

o en caso de que ya exista y esté en dhcp:

```
sudo nmcli connection modify enp0s8 con-name red_interna ifname enp0s8 type ethernet ipv4.method manual ip4 192.168.1.2/24 ipv6.method disabled
```

### 1.3. ubuntu01

Se crea el servidor ubuntu01 usando la distribución Ubuntu 24.04, con las mismas características que centos01:

- 1 CPU
- 2 GB de RAM
- 20 GB de disco
- 2 interfaces de red, una en NAT y la otra interna.

La base elegida para el servidor es Ubuntu Server, root está deshabilitado por defecto, y se crea el usuario administrador sysadmin con la password del curso. Se elije la opción de instalar OpenSSH server.

Las particiones son idénticas a bastion:

| Punto de montaje | Capacidad |
|---|---|
| /boot | 1 |
| / | 10 |
| /var | 5 |
| SWAP | 4 |
| Total | 20 |

```
root@ubuntu01:~# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda      8:0    0   20G  0 disk
├─sda1   8:1    0    1M  0 part
├─sda2   8:2    0    1G  0 part /boot
├─sda3   8:3    0   10G  0 part /
├─sda4   8:4    0    5G  0 part /var
└─sda5   8:5    0    4G  0 part [SWAP]
sr0     11:0    1 1024M  0 rom
root@ubuntu01:~#
```

Le asignamos la dirección de red 192.168.1.3/24

Contenido de `/etc/netplan/50-cloud-init.yaml :`

```
network:
  version: 2
  ethernets:
    enp0s3:
      dhcp4: true
    enp0s8:
      dhcp4: no
      dhcp6: no
      addresses:
        - 192.168.1.3/24
      link-local: [ ipv4 ]
```

Ejecutamos `sudo netplan apply`

### 1.4. Configuración SSH

Generamos las llaves pública/privada para el usuario ansible en el bastión:

```
ssh-keygen
```

```
[sysadmin@bastion ~]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/sysadmin/.ssh/id_rsa):
Created directory '/home/sysadmin/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/sysadmin/.ssh/id_rsa
Your public key has been saved in /home/sysadmin/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:Pfmf7DIs3ksDSM2MVgROr2LExRnS4PsUQWHZx77XHP8 sysadmin@bastion
The key's randomart image is:
+---[RSA 3072]----+
|      o=XOo.     |
|     o *=O. o    |
|      + * =o     |
|     . + = ..  . |
|      + S =  . oo|
|     . +   +. . +|
|        .  .+.  .|
|          .o++ .E|
|         ...o==  |
+----[SHA256]-----+
[sysadmin@bastion ~]$
```

```
ssh-copy-id -i .ssh/id_rsa.pub 192.168.1.2
ssh-copy-id -i .ssh/id_rsa.pub 192.168.1.3
```

```
[sysadmin@bastion ~]$ ssh-copy-id -i .ssh/id_rsa.pub 192.168.1.2
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: ".ssh/id_rsa.pub"
The authenticity of host '192.168.1.2 (192.168.1.2)' can't be established.
ED25519 key fingerprint is SHA256:AJgP37gO52N3ARxpq9PZkwQaPw3jHrw2dWH7ctMImm0.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are
 already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to
install the new keys
sysadmin@192.168.1.2's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh '192.168.1.2'"
and check to make sure that only the key(s) you wanted were added.

[sysadmin@bastion ~]$ ssh-copy-id -i .ssh/id_rsa.pub 192.168.1.3
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: ".ssh/id_rsa.pub"
The authenticity of host '192.168.1.3 (192.168.1.3)' can't be established.
ED25519 key fingerprint is SHA256:K66frisiB7SWl9JNstpH6sHf9pUnboSsQXdaUyZS1K8.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are
 already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to
install the new keys
sysadmin@192.168.1.3's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh '192.168.1.3'"
and check to make sure that only the key(s) you wanted were added.
```

## 2. Configuración de archivo de inventario de Ansible

Instalación de Ansible

```
sudo dnf install ansible-core -y
```

```
mkdir /home/sysadmin/taller_linux2025
```

Contenido de `~/taller_linux2025/ansible.cfg`:

```
[defaults]
inventory = ./inventory.ini  ; archivo con hosts
```

Contenido de `~/taller_linux2025/inventory.ini`:

```
[Centos]
centos01 ansible_host=192.168.1.2

[Ubuntu]
ubuntu01 ansible_host=192.168.1.3

[webserver]
centos01

[Linux:children]
Centos
Ubuntu

[all:vars]
ansible_user=sysadmin
```

Salida de `ansible-inventory -i inventory.ini --list:`

```
[sysadmin@bastion taller_linux2025]$ ansible-inventory -i inventory.ini --list
{
    "Centos": {
        "hosts": [
            "centos01"
        ]
    },
    "Linux": {
        "children": [
            "Centos",
            "Ubuntu"
        ]
    },
    "Ubuntu": {
        "hosts": [
            "ubuntu01"
        ]
    },
    "_meta": {
        "hostvars": {
            "centos01": {
                "ansible_host": "192.168.1.2",
                "ansible_user": "sysadmin"
            },
            "ubuntu01": {
                "ansible_host": "192.168.1.3",
                "ansible_user": "sysadmin"
            }
        }
    },
    "all": {
        "children": [
            "ungrouped",
            "webserver",
            "Linux"
        ]
    },
    "webserver": {
        "hosts": [
            "centos01"
        ]
    }
}
```

Salida de `ansible all -i inventory.ini -m ping`:

```
[sysadmin@bastion taller_linux2025]$ ansible all -i inventory.ini -m ping
ubuntu01 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
centos01 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

## 3. Ejecución de comandos ad-hoc

La ejecución de comandos ad-hoc se realiza con la siguiente sintaxis:

```
$ ansible [pattern] -m [module] -a "[module options]"
```

### 3.1. Listado de usuarios

Para listar todos los usuarios en el servidor Ubuntu usamos el módulo ansible.builtin.shell. Al ya tener el inventario cargado en ansible.cfg ya no es necesario específicar la ubicación del archivo inventory.ini:

```
ansible Ubuntu -m ansible.builtin.shell -a "cat /etc/passwd"
```

```
devuelve el contenido de passwd. Necesitamos que el dígito luego del segundo : sea mayor o igual a 1000, o sea, 4 dígitos, distinto de nobody.
```

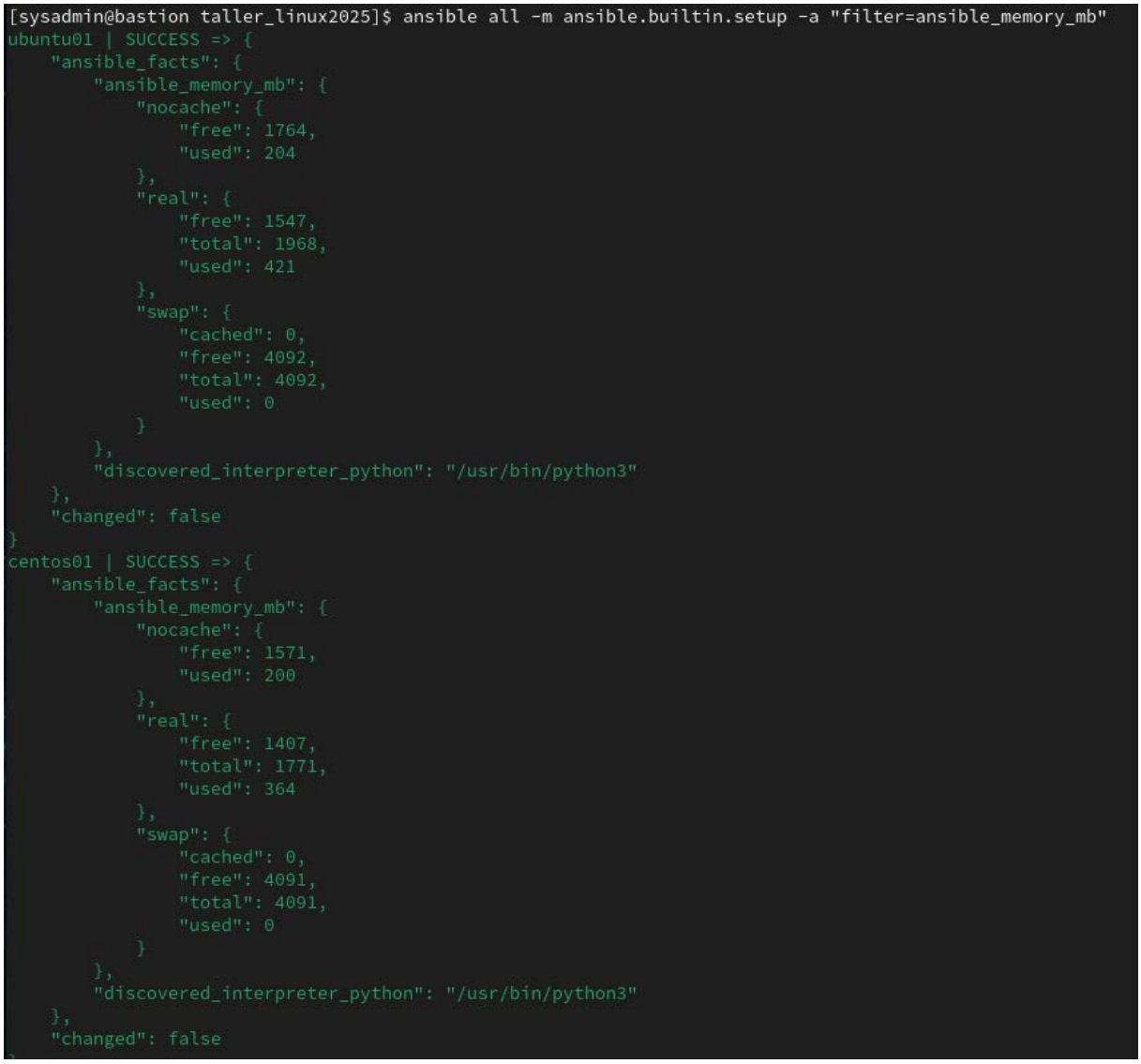
```
ansible Ubuntu -m ansible.builtin.shell -a "cat /etc/passwd" | grep -P "^[^:]*:x:\d\d\d\d" | cut -d ":" -f1 | grep -v "nobody"
```

```
[sysadmin@bastion taller_linux2025]$ ansible Ubuntu -m ansible.builtin.shell -a "cat /etc
/passwd" | grep -P "^[^:]*:x:\d\d\d\d" | cut -d ":" -f1 | grep -v "nobody"
sysadmin
[sysadmin@bastion taller_linux2025]$
```

### 3.2. Uso de memoria

Listamos el uso de memoria RAM en todos los servidores

```
ansible all -m ansible.builtin.setup -a "filter=ansible_memory_mb"
```




## 3.3. Estado del servicio chronyd

Para chequear si un paquete existe podemos usar el módulo ansible.builtin.package_facts.

```
ansible Centos -m ansible.builtin.package_facts | sed '1c\{' | jq
'.ansible_facts.packages.chrony'
```

Alternativamente podemos usar ansible.builtin.package con el parámetro C para chequear sin modificar:

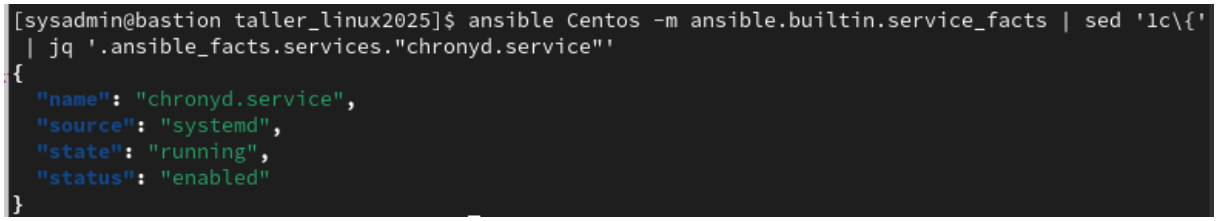
```
ansible Centos -m ansible.builtin.package --become -a "name=chrony
state=present" -C -K
```


Si el objetivo es asegurarnos que el paquete esté instalado, usamos el mismo comando sin el parámetro C:

```
ansible Centos -m ansible.builtin.package --become -a "name=chrony
state=present" -K
```

Usamos el módulo ansible.builtin.service_facts para recopilar datos de los servicios, sed para formatear la salida, y jq para filtrar el json:

```
ansible Centos -m ansible.builtin.service_facts | sed '1c\{' | jq
'.ansible_facts.services."chronyd.service"'
```




Alternativamente, podemos usar ansible.builtin.systemd_service con el parámetro name=chronyd para chequear el estado.

```
ansible Centos -m ansible.builtin.systemd_service -a "name=chronyd"
```

Si el objetivo es modificar el estado del servicio, usamos el mismo módulo con los siguientes parámetros:

```
ansible Centos -m ansible.builtin.systemd_service -a "name=chronyd
enabled=true state=started"
```

```
[sysadmin@bastion taller_linux2025]$ ansible Centos -m ansible.builtin.systemd_service -a "name=c
hronyd enabled=true state=started"
centos01 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "enabled": true,
    "name": "chronyd",
    "state": "started",
```

## 4. Creación y Ejecución de playbooks de Ansible

Se crearán dos playbooks en el directorio ~/playbooks

Contenido de requirements.yml

```
---
collections:
  - name: ansible.posix
    version: "1.5.4"
  - name: community.general
    version: "9.5.10"
```

Ejecutamos `ansible-galaxy collection install -r requirements.yml` para instalar ansible.posix, requerido por el módulo de firewalld

### 4.1. nfs_setup.yml

Esta playbook se va a ejecutar en un CentOS.

Objetivos:

- Paquete nfs-utils instalado
- Puerto 2049 abierto
- /var/nfs_shared con usuario/grupo nobody/nobody, permisos 777
- /var/nfs_shared está compartido por NFS
- si /etc/exports cambia debe ser releído (`exportfs -r`)

Contenido de nfs-setup.yml:

```
---
- name: Configuracion de NFS Server
  hosts: Centos
  become: true

  tasks:
  - name: Asegurarse que nfs-utils esta instalado
```

```
    ansible.builtin.package:
      name: nfs-utils
      state: present

  - name: Iniciar y habilitar nfs
    ansible.builtin.service:
      name: nfs-server
      enabled: true
      state: started

  - name: Abrir puerto 2049
    ansible.posix.firewalld:
      service: nfs
      permanent: true
      immediate: true
      state: enabled

  - name: Crear /var/nfs_shared
    ansible.builtin.file:
      path: /var/nfs_shared
      state: directory
      owner: nobody
      group: nobody
      mode: '777'

  - name: Compartir carpeta por NFS
    ansible.builtin.lineinfile:
      path: /etc/exports
      line: /var/nfs_shared/  *(rw,sync,no_subtree_check)
      state: present
    notify:
      - Releer /etc/exports

  handlers:
    - name: Releer /etc/exports
      ansible.builtin.shell:
        cmd: exportfs -r
```

Chequeamos sintaxis con `ansible-playbook playbooks/nfs_setup.yml --syntax-check`

Ejecutamos `ansible-playbook playbooks/nfs_setup.yml -K`

```
[sysadmin@bastion taller_linux2025]$ ansible-playbook playbooks/nfs_setup.yml -K
BECOME password:

PLAY [Configuracion de NFS Server] *********************************************************

TASK [Gathering Facts] *********************************************************************
ok: [centos01]

TASK [Asegurarse que nfs-utils esta instalado] *********************************************
changed: [centos01]

TASK [Iniciar y habilitar nfs] *************************************************************
changed: [centos01]

TASK [Abrir puerto 2049] *******************************************************************
changed: [centos01]

TASK [Crear /var/nfs_shared] ***************************************************************
changed: [centos01]

TASK [Compartir carpeta por NFS] ***********************************************************
changed: [centos01]

RUNNING HANDLER [Releer /etc/exports] ******************************************************
changed: [centos01]

PLAY RECAP *********************************************************************************
centos01                   : ok=7    changed=6    unreachable=0    failed=0    skipped=0    rescu
ed=0    ignored=0

[sysadmin@bastion taller_linux2025]$
```

Ejecutamos de nuevo para confirmar que la tarea "Releer /etc/exports" no se activa, y no se hacen cambios.

```
[sysadmin@bastion taller_linux2025]$ ansible-playbook playbooks/nfs_setup.yml -K
BECOME password:

PLAY [Configuracion de NFS Server] *********************************************************

TASK [Gathering Facts] *********************************************************************
ok: [centos01]

TASK [Asegurarse que nfs-utils esta instalado] *********************************************
ok: [centos01]

TASK [Iniciar y habilitar nfs] *************************************************************
ok: [centos01]

TASK [Abrir puerto 2049] *******************************************************************
ok: [centos01]

TASK [Crear /var/nfs_shared] ***************************************************************
ok: [centos01]

TASK [Compartir carpeta por NFS] ***********************************************************
ok: [centos01]

PLAY RECAP *********************************************************************************
centos01                   : ok=6    changed=0    unreachable=0    failed=0    skipped=0    rescu
ed=0    ignored=0
```

## 4.2. hardening.yml

Objetivos:

- Actualizar todos los paquetes
- ufw activo, solo permite ssh (22)
- solo se permite login con clave pública, root no puede hacer login
- fail2ban instalado, bloqueando intentos fallidos por SSH. Activo y habilitado
- si se actualizan paquetes, reiniciar
- si cambia configuración, reiniciar ssh

Contenido de ~/taller_linux2025/jail.local

```
[sshd]
enabled = true
```

Contenido de hardening.yml:

```
---
- name: Actualizar y hardenizar Ubuntu
  hosts: Ubuntu
  become: true

  tasks:
  - name: Actualizar paquetes
    ansible.builtin.apt:
      name: "*"
      state: latest
    notify: Reiniciar el equipo

  - name: Configurar Firewall
    community.general.ufw:
      state: enabled
      default: deny
      rule: allow
      port: ssh
      proto: tcp

  - name: Login con clave publica, root no puede login
    ansible.builtin.lineinfile:
      path: /etc/ssh/sshd_config
      search_string: "{{ item.search_string }}"
      state: present
      line: "{{ item.line }}"
    loop:
      - {search_string: "#PasswordAuthentication yes", line:
PasswordAuthentication no }
      - {search_string: "#PermitRootLogin prohibit-password", line:
PermitRootLogin no }
    notify: Reiniciar SSH

  - name: Instalar fail2ban
```

```
    ansible.builtin.package:
      name: fail2ban
      state: present

  - name: Configurar fail2ban
    ansible.builtin.copy:
      src: ./jail.local
      dest: /etc/fail2ban/jail.local

  - name: Iniciar y habilitar fail2ban
    ansible.builtin.service:
      name: fail2ban
      enabled: true
      state: started

  handlers:
    - name: Reiniciar el equipo
      ansible.builtin.reboot:

    - name: Reiniciar SSH
      ansible.builtin.service:
        name: sshd
        state: restarted
```

Probar sintaxis:

```
[sysadmin@bastion taller_linux2025]$ ansible-playbook playbooks/hardening.yml --syntax-check

playbook: playbooks/hardening.yml
```

Ejecutar playbook

```
ansible-playbook playbooks/hardening.yml -K
```

```
[sysadmin@bastion taller_linux2025]$ ansible-playbook playbooks/hardening.yml -K
BECOME password:

PLAY [Actualizar y hardenizar Ubuntu] ******************************************************************

TASK [Gathering Facts] *********************************************************************************
ok: [ubuntu01]

TASK [Actualizar paquetes] *****************************************************************************
ok: [ubuntu01]

TASK [Configurar Firewall] *****************************************************************************
ok: [ubuntu01]

TASK [Login con clave publica, root no puede login] ****************************************************
ok: [ubuntu01] => (item={'search_string': '#PasswordAuthentication yes', 'line': 'PasswordAuthenticatio
n no'})
ok: [ubuntu01] => (item={'search_string': '#PermitRootLogin prohibit-password', 'line': 'PermitRootLogi
n no'})

TASK [Instalar fail2ban] *******************************************************************************
ok: [ubuntu01]

TASK [Configurar fail2ban] *****************************************************************************
ok: [ubuntu01]

TASK [Iniciar y habilitar fail2ban] ********************************************************************
ok: [ubuntu01]

PLAY RECAP *********************************************************************************************
ubuntu01                   : ok=7    changed=0    unreachable=0    failed=0    skipped=0    rescued=0
  ignored=0
```

## 5. Cuestionario

### 5.1. ¿Qué es Ansible?

Ansible es un software utilizado para automatizar la gestión de sistemas remotos y controlar su estado final mediante declaraciones de estado.

### 5.2. ¿Qué es un playbook?

Un playbook es una receta que definir estados y configuraciones, la cual puede ser reutilizada en el mismo u otros sistemas, obteniendo resultados idénticos.

### 5.3. ¿Qué información contiene un inventario de Ansible?

Un inventario es una manera prolija de definir hosts, grupos y variables, para luego utilizar estas definiciones tanto en playbooks como en ejecuciones ad-hoc.

### 5.4. ¿Qué es un módulo de Ansible?

Un módulo es una unidad de código que puede controlar recursos del sistema, o ejecutar comandos. Existen módulos ya creados que permiten interactuar con diferentes partes de un sistema, y realizar tareas específicas, lo cual facilita mucho la creación de playbooks.

### 5.5. ¿Qué ventajas tiene Ansible sobre otros métodos de automatización?

Facilidad de uso, simplicidad, seguridad y confiabilidad. No usa un agente, lo cual lo hace más fácil de mantener y ejecutar.

## 6. Anexo

## 7. Referencias

https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/9/html/configuring_and_managing_networking/configuring-an-ethernet-connection_configuring-and-managing-networking

https://documentation.ubuntu.com/server/explanation/networking/configuring-networks/

https://www.redhat.com/en/blog/configure-ssh-keygen

https://docs.ansible.com/ansible/latest/reference_appendices/config.html

https://docs.ansible.com/ansible/latest/reference_appendices/general_precedence.html

https://docs.ansible.com/ansible/latest/inventory_guide/intro_inventory.html

https://docs.ansible.com/ansible/latest/command_guide/intro_adhoc.html

https://regex101.com/

https://www.gnu.org/software/sed/manual/sed.html

https://hostman.com/tutorials/using-the-jq-command-to-process-json-on-the-command-line/

https://docs.ansible.com/ansible/latest/collections/ansible/builtin/package_module.html

https://docs.ansible.com/ansible/latest/collections/ansible/builtin/package_facts_module.html

https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/8/html/deploying_different_types_of_servers/deploying-an-nfs-server_deploying-different-types-of-servers

https://docs.ansible.com/ansible/latest/collections/ansible/posix/firewalld_module.html

https://galaxy.ansible.com/ui/repo/published/ansible/posix/docs/

https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/9/html/configuring_firewalls_and_packet_filters/using-and-configuring-firewalld_firewall-packet-filters

https://docs.ansible.com/ansible/latest/collections/ansible/builtin/file_module.html

https://docs.ansible.com/ansible/latest/collections/ansible/builtin/lineinfile_module.html

https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/6/html/storage_administration_guide/nfs-serverconfig#nfs-serverconfig-exports

https://docs.ansible.com/ansible/latest/playbook_guide/playbooks_handlers.html

https://docs.ansible.com/ansible/latest/collections/ansible/builtin/shell_module.html

https://docs.ansible.com/ansible/latest/playbook_guide/playbooks_privilege_escalation.html

https://docs.ansible.com/ansible/latest/collections/ansible/builtin/apt_module.html

https://docs.ansible.com/ansible/latest/collections/community/general/ufw_module.html

https://docs.ansible.com/ansible/latest/playbook_guide/playbooks_loops.html

https://help.ubuntu.com/community/Fail2ban

https://github.com/fail2ban/fail2ban/wiki/Proper-fail2ban-configuration

https://docs.ansible.com/ansible/latest/collections/ansible/builtin/copy_module.html

https://www.redhat.com/en/ansible-collaborative/how-ansible-works