

Universidad ORT Uruguay

Facultad de Ingeniería: Escuela de Tecnología

“Taller de Servidores Linux”

Autores:

Santiago Albanese N.º 323347

Angel Chirinos N.º 341169

Fecha de Entrega: 12 de Agosto del 2025

Declaración de Autoría

Nosotros, Santiago Albanese y Angel Chirinos declaramos que el trabajo que se presenta a continuación es de nuestra propia mano. Podemos asegurar que:

- La obra fue producida en su totalidad mientras realizábamos la asignatura Taller 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.

Santiago Albanese

12/08/2025

Angel Chirinos

12/08/2025

Índice

1.	Objetivos.....	1
2.	Instalación de Servidores.....	1
2.1.	Instalación de Servidor Ubuntu.....	1
2.2.	Instalación de Servidor Centos	2
3.	Instalación de Git, creación de repositorio y conexión a través de SSH	3
4.	Instalación de Ansible-core.....	5
5.	Ejecución de comandos ad-hoc:.....	7
6.	Creación y ejecución de Playbooks de Ansible.....	10
7.	Investigación	13
8.	Desafíos encontrados.....	19

1. Objetivos

Aplicar los conocimientos básicos de Ansible sobre dos distribuciones Linux: Centos Stream 9 y Ubuntu 24.04.

2. Instalación de Servidores

Se realiza la instalación de los servidores con las siguientes características, desde la instalación:

- 1 CPU
- 2GB RAM
- 2 GB para el filesystem /boot
- 10 GB para el filesystem /
- 5 GB para el filesystem /var
- 4 GB para el SWAP
- 2 interfaces de red. Una conectada a NAT y otra a la red interna.

2.1. Instalación de Servidor Ubuntu

```
Storage configuration

FILE SYSTEM SUMMARY

  MOUNT POINT      SIZE      TYPE      DEVICE TYPE
[ /                10.000G   new ext4   new partition of local disk ▶ ]
[ /boot            2.000G   new ext4   new partition of local disk ▶ ]
[ /var              5.000G   new ext4   new partition of local disk ▶ ]
[ SWAP              4.000G   new swap   new partition of local disk ▶ ]

AVAILABLE DEVICES

  DEVICE                                     TYPE      SIZE
[ VBOX_HARDDISK_VBdc524126-53c571c2         local disk 25.000G ▶ ]
  free space                                3.997G ▶ ]

[ Create software RAID (md) ▶ ]
[ Create volume group (LVM) ▶ ]

USED DEVICES

  DEVICE                                     TYPE      SIZE
[ VBOX_HARDDISK_VBdc524126-53c571c2         local disk 25.000G ▶ ]
  partition 1   new, BIOS grub spacer         1.000M ▶ ]
  partition 2   new, to be formatted as ext4, mounted at /boot 2.000G ▶ ]
  partition 3   new, to be formatted as ext4, mounted at /    10.000G ▶ ]
  partition 4   new, to be formatted as ext4, mounted at /var  5.000G ▶ ]
  partition 5   new, to be formatted as swap    4.000G ▶ ]
```

2.2. Instalación de Servidor Centos

The screenshot shows the 'MANUAL PARTITIONING' screen for 'CENTOS STREAM 9 INSTALLATION'. The interface is divided into several sections:

- Top Bar:** Includes a 'Done' button, a language selector set to 'us', and a 'Help!' button.
- Left Panel:** Titled 'New CentOS Stream 9 Installation', it contains a 'SYSTEM' list with the following items:
 - /boot** (sda1) - 1 GiB
 - /** (sda2) - 10 GiB
 - /var** (sda3) - 5 GiB
 - swap** (sda5) - 4 GiB
- Right Panel:** Configures the selected partition 'sda1'. It includes:
 - Mount Point:** A text field containing '/boot'.
 - Device(s):** A dropdown menu showing 'ATA VBOX HARDDISK (sda)' with a 'Modify...' button.
 - Desired Capacity:** A text field containing '1 GiB'.
 - Device Type:** A dropdown menu set to 'Standard Partition' and an unchecked 'Encrypt' checkbox.
 - File System:** A dropdown menu set to 'ext4' and a checked 'Reformat' checkbox.
 - Label:** An empty text field.
 - Name:** A text field containing 'sda1'.
 - Buttons:** 'Update Settings' and 'Discard All Changes'.
 - Note:** A text block stating: 'Note: The settings you make on this screen will not be applied until you click on the main menu's 'Begin Installation' button.'
- Bottom Bar:** Displays 'AVAILABLE SPACE' as '1.99 MiB' and 'TOTAL SPACE' as '20 GiB'. Below this, it says '1 storage device selected'.

3. Instalación de Git, creación de repositorio y conexión a través de SSH

```
Activities Terminal Aug 6 11:14
~/ansible-obligatorio

Verifying      : perl-TermReadKey-2.38-11.el9.x86_64      6/7
Verifying      : perl-lib-0.65-483.el9.x86_64            7/7

Installed:
git-2.47.3-1.el9.x86_64      git-core-2.47.3-1.el9.x86_64
git-core-doc-2.47.3-1.el9.noarch  perl-Error-1:0.17029-7.el9.noarch
perl-Git-2.47.3-1.el9.noarch  perl-TermReadKey-2.38-11.el9.x86_64
perl-lib-0.65-483.el9.x86_64

Complete!
sysadmin@localhost ~> git config --global user.name "angelch25"
sysadmin@localhost ~> git config --global user.email "angelchppq@gmail.com"
sysadmin@localhost ~> git config list
user.name=angelch25
user.email=angelchppq@gmail.com
sysadmin@localhost ~> mkdir ansible-obligatorio
sysadmin@localhost ~> cd ansible-obligatorio/
sysadmin@localhost ~/ansible-obligatorio> git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/sysadmin/ansible-obligatorio/.git/
sysadmin@localhost ~/ansible-obligatorio (master)> git status
On branch master

no commits yet

nothing to commit (create/copy files and use "git add" to track)
sysadmin@localhost ~/ansible-obligatorio (master)> git log
fatal: your current branch 'master' does not have any commits yet
sysadmin@localhost ~/ansible-obligatorio (master) [128]> ls ~/.ssh/id_rsa
/home/sysadmin/.ssh/id_rsa
sysadmin@localhost ~/ansible-obligatorio (master)> ls ~/.ssh/id_rsa.pub
/home/sysadmin/.ssh/id_rsa.pub
sysadmin@localhost ~/ansible-obligatorio (master)> cat ~/.ssh/id_rsa.pub
```

```
sysadmin@localhost ~/ansible-obligatorio (main)> echo "# Proyecto Ansible Obligatorio" > README.md
sysadmin@localhost ~/ansible-obligatorio (main)> git add README.md
sysadmin@localhost ~/ansible-obligatorio (main)> git commit -m "Primer commit: agregado README.md"
[main (root-commit) 1deac86] Primer commit: agregado README.md
1 file changed, 1 insertion(+)
create mode 100644 README.md
sysadmin@localhost ~/ansible-obligatorio (main)> git push -u origin main
ERROR: Repository not found.
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.
sysadmin@localhost ~/ansible-obligatorio (main) [128]> git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 258 bytes | 64.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:angelch25/ansible-obligatorio.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
sysadmin@localhost ~/ansible-obligatorio (main)>
```

The screenshot shows the GitHub interface for the repository 'ansible-obligatorio' owned by 'angelch25'. The repository is public and currently has 0 stars, 0 forks, and 0 watchers. The main branch is selected, showing a commit '1deac86' from 12 minutes ago with the message 'Primer commit: agregado README.md'. The commit details show a single file 'README.md' added. The README content is visible, starting with the title 'Proyecto Ansible Obligatorio'. The right sidebar contains sections for 'About' (Repository for the obligatory), 'Releases' (No releases published), and 'Packages' (No packages published).

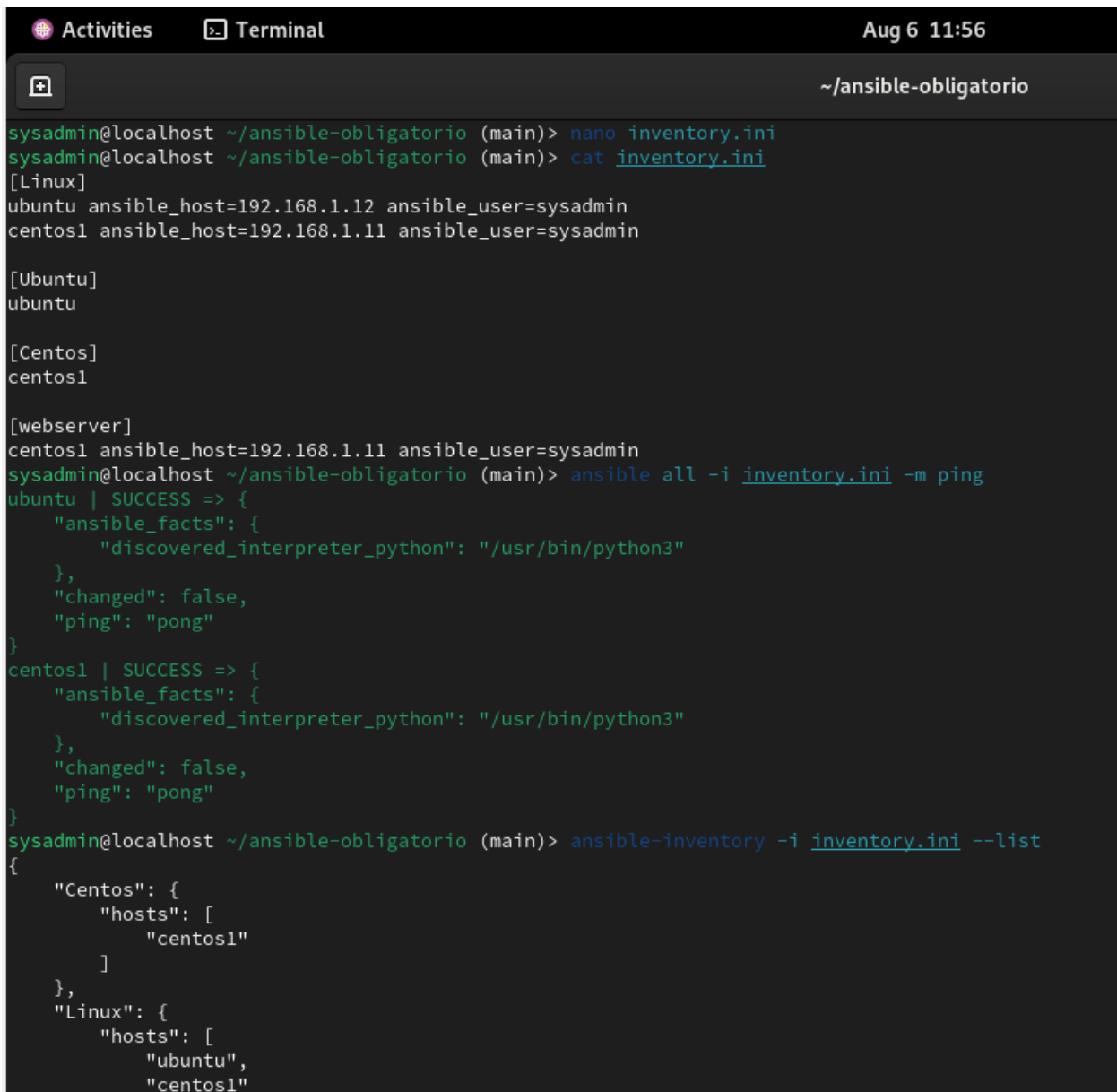
4. Instalación de Ansible-core

Se realiza instalación de Ansible-core y se ajusta para su funcionamiento con fish en lugar de bash como Shell.

```
Activities Terminal Aug 6 11:36
~/ansible-obligatorio

Otherwise pipx is ready to go! 🌟 🌟 🌟
sysadmin@localhost ~/ansible-obligatorio (main)> pipx install ansible-core
  installed package ansible-core 2.15.13, installed using Python 3.9.21
  These apps are now globally available
  - ansible
  - ansible-config
  - ansible-connection
  - ansible-console
  - ansible-doc
  - ansible-galaxy
  - ansible-inventory
  - ansible-playbook
  - ansible-pull
  - ansible-test
  - ansible-vault
done! 🌟 🌟 🌟
sysadmin@localhost ~/ansible-obligatorio (main)> pipx inject ansible-core argcomplete
  injected package argcomplete into venv ansible-core
done! 🌟 🌟 🌟
sysadmin@localhost ~/ansible-obligatorio (main)> pipx inject ansible-core ansible-lint
  injected package ansible-lint into venv ansible-core
done! 🌟 🌟 🌟
sysadmin@localhost ~/ansible-obligatorio (main)> activate-global-python-argcomplete --user
Argcomplete was installed in the user site local directory. Defaulting to user installation.
Adding shellcode to /home/sysadmin/.zshenv...
Added.
Adding shellcode to /home/sysadmin/.bash_completion...
Added.
Please restart your shell or source the installed file to activate it.
sysadmin@localhost ~/ansible-obligatorio (main)> source ~/.bashrc
~/.bashrc (line 11): Unsupported use of '='. In fish, please use 'set PATH "$HOME/.local/bin:$HOME/bin:$PATH"'.
  PATH="$HOME/.local/bin:$HOME/bin:$PATH"
  ^~~~~~
from sourcing file ~/.bashrc
source: Error while reading file '/home/sysadmin/.bashrc'
sysadmin@localhost ~/ansible-obligatorio (main) [1]> nano ~/.config/fish/config.fish
sysadmin@localhost ~/ansible-obligatorio (main)> source ~/.config/fish/config.fish
sysadmin@localhost ~/ansible-obligatorio (main)> ansible --version
ansible [core 2.15.13]
  config file = None
  configured module search path = ['/home/sysadmin/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
```


Creación de archivo de inventario (inventory.ini) y conexión exitosa



```
Activities Terminal Aug 6 11:56
~/ansible-obligatorio

sysadmin@localhost ~/ansible-obligatorio (main)> nano inventory.ini
sysadmin@localhost ~/ansible-obligatorio (main)> cat inventory.ini
[Linux]
ubuntu ansible_host=192.168.1.12 ansible_user=sysadmin
centos1 ansible_host=192.168.1.11 ansible_user=sysadmin

[Ubuntu]
ubuntu

[Centos]
centos1

[webserver]
centos1 ansible_host=192.168.1.11 ansible_user=sysadmin
sysadmin@localhost ~/ansible-obligatorio (main)> ansible all -i inventory.ini -m ping
ubuntu | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
centos1 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
sysadmin@localhost ~/ansible-obligatorio (main)> ansible-inventory -i inventory.ini --list
{
  "Centos": {
    "hosts": [
      "centos1"
    ]
  },
  "Linux": {
    "hosts": [
      "ubuntu",
      "centos1"
    ]
  }
}
```

Se suben cambios mas recientes a Git y Git Hub, a partir de la creación del segundo commit con el archivo inventario.ini como se muestra a continuación:

```

sysadmin@localhost ~/ansible-obligatorio (main)> git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        inventory.ini

nothing added to commit but untracked files present (use "git add" to track)
sysadmin@localhost ~/ansible-obligatorio (main)> git add.
git: 'add.' is not a git command. See 'git --help'.

The most similar command is
    add
sysadmin@localhost ~/ansible-obligatorio (main) [1]> git add .
sysadmin@localhost ~/ansible-obligatorio (main)> git commit -m "Segundo commit: creacion de archivo de inventario"
[main 1faa9ed] Segundo commit: creacion de archivo de inventario
 1 file changed, 12 insertions(+)
 create mode 100644 inventory.ini
sysadmin@localhost ~/ansible-obligatorio (main)> git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 403 bytes | 100.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:angelch25/ansible-obligatorio.git
   1deac86..1faa9ed  main -> main
sysadmin@localhost ~/ansible-obligatorio (main)> git log
commit 1faa9edeb3c7469013d9b6efdf485a264fa6902d (HEAD -> main, origin/main)
Author: angelch25 <angelchpq@gmail.com>
Date:   Wed Aug 6 12:07:22 2025 -0300

    Segundo commit: creacion de archivo de inventario

commit 1deac86122dac637460614a1ba36a3a66ed35147
Author: angelch25 <angelchpq@gmail.com>
Date:   Wed Aug 6 11:03:06 2025 -0300

    Primer commit: agregado README.md
sysadmin@localhost ~/ansible-obligatorio (main)>

```

5. Ejecución de comandos ad-hoc:

5.1. Listar todos los usuarios en servidor Ubuntu

Comando utilizado:

ansible Ubuntu -i inventory.ini -a "cut -d: -f1 /etc/passwd"

Con este comando se le indica a Ansible:

Ubuntu: Es el grupo host al que se le ejecuta el comando (en este caso solo a Ubuntu)

-a: Se coloca para indicar que es un comando del sistema y no un módulo de ansible.

cut -d: -f1 /etc/passwd: Extrae solo el nombre de la lista de usuarios en el sistema.

Se obtiene el siguiente resultado:

```

sysadmin@localhost ~/ansible-obligatorio (main)> ansible Ubuntu -i inventory.ini -a "cut -d: -f1 /etc/passwd"
ubuntu | CHANGED | rc=0 >>
root
daemon
bin
sys
sync
games
man
lp
mail
news
uucp
proxy
www-data
backup
list
irc
_apt
nobody
systemd-network
systemd-timesync
dhcpcd
messagebus
systemd-resolve
pollinate
polkitd
syslog
uidd
tcpdump
tss
landscape
fwupd-refresh
usbmux
sysadmin
sshd

```

5.2. Mostrar el uso de memoria en todos los servidores

Comando utilizado:

ansible all -i inventory.ini -a "free -h"

Con este comando se le indica a Ansible:

all: Muestra el resultado en todos los host indicados en el inventario

-a: Se coloca para indicar que es un comando del sistema y no un módulo de ansible.

Free -h: Muestra el uso de memoria en formato legible.

Se obtiene el siguiente resultado:

```

sysadmin@localhost ~/ansible-obligatorio (main)> ansible all -i inventory.ini -a "free -h"
ubuntu | CHANGED | rc=0 >>
      total        used        free      shared  buff/cache   available
Mem:    1.9Gi       314Mi       1.5Gi       1.1Mi       291Mi       1.6Gi
Swap:    4.0Gi           0B        4.0Gi
centos1 | CHANGED | rc=0 >>
      total        used        free      shared  buff/cache   available
Mem:    1.7Gi       950Mi       292Mi       6.0Mi       690Mi       821Mi
Swap:    4.0Gi       14Mi        4.0Gi

```

5.3. Que el servicio chrony esté instalado y funcionando en servidor Centos

Comando utilizado para verificar la instalación:

ansible Centos -i inventory.ini -b -m dnf -a "name=chrony state=present"

Con este comando se le indica a Ansible:

Centos: Es el grupo host al que se le ejecuta el comando (en este caso solo a Centos)

-a: Se coloca para indicar que es un comando del sistema y no un módulo de ansible.

-b: Ejecuta los comandos con privilegios de sudo

-m dnf: Utiliza el módulo dnf de Ansible para instalar paquetes

Se obtiene el siguiente resultado:

```
sysadmin@localhost ~/ansible-obligatorio (main)> ansible Centos -i inventory.ini -b -m dnf -a "name=chrony state=present"
centos1 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "msg": "Nothing to do",
  "rc": 0,
  "results": []
}
```

Comando utilizado para asegurar que el servicio esta iniciado y habilitado:

ansible Centos -i inventory.ini -b -m service -a "name=chronyd state=started enabled=yes"

Con este comando se le indica a Ansible:

Centos: Es el grupo host al que se le ejecuta el comando (en este caso solo a Centos)

-a: Se coloca para indicar que es un comando del sistema y no un módulo de ansible.

-b: Ejecuta los comandos con privilegios de sudo

-m service: Usa el modulo service para manejar los servicios del sistema

Chronyd: En Centos el servicio se llama chronyd

Se obtiene el siguiente resultado:

```

~/ansible-obligatorio

sysadmin@localhost ~/ansible-obligatorio (main)> ansible Centos -i inventory.ini -b -m service -a "name=chronyd state=started enabled=yes"
centos1 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "enabled": true,
  "name": "chronyd",
  "state": "started",
  "status": {
    "AccessSELinuxContext": "system_u:object_r:chronyd_unit_file_t:s0",
    "ActiveEnterTimestamp": "Sat 2025-08-09 14:58:31 -03",
    "ActiveEnterTimestampMonotonic": "8983118",
    "ActiveExitTimestampMonotonic": "0",
    "ActiveState": "active",
    "After": "tmp.mount sysinit.target ntpd.service ntpdate.service sntp.service basic.target var.mount system.slice -.mount systemd-journald-tmpfiles-setup.service",
    "AllowIsolate": "no",
    "AssertResult": "yes",
    "AssertTimestamp": "Sat 2025-08-09 14:58:31 -03",
    "AssertTimestampMonotonic": "8439584",
    "Before": "shutdown.target multi-user.target",
    "BlockIOAccounting": "no",
    "BlockIOWeight": "[not set]",
    "CPUAffinityFromNUMA": "no",
    "CPUQuotaPerSecUsec": "infinity",
    "CPUQuotaPeriodUsec": "infinity",
    "CPUSchedulingPolicy": "0",
    "CPUSchedulingPriority": "0",
    "CPUSchedulingResetOnFork": "no",
    "CPUShares": "[not set]",
    "CPUUsageNsec": "85241000",
    "CPUWeight": "[not set]",
    "CacheDirectoryMode": "0755",
    "CanFreeze": "yes",
    "CanIsolate": "no",
    "CanReload": "no",
    "CanStart": "yes",
    "CanStop": "yes",
    "CapabilityBoundingSet": "cap_chown cap_dac_override cap_dac_read_search cap_fowner cap_fsetid cap_setgid cap_setuid cap_setpcap cap_

```

6. Creación y ejecución de Playbooks de Ansible

Se crea el playbook `nfs_setup` y se actualiza el archivo inventario. Se realiza commit en git por actualizaciones

```

(use "git add <file>..." to update what will be committed)
(use "git restore <file>..." to discard changes in working directory)
        modified:   inventory.ini

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        nfs_setup.yml

no changes added to commit (use "git add" and/or "git commit -a")
sysadmin@localhost ~/ansible-obligatorio (main)> git add .
sysadmin@localhost ~/ansible-obligatorio (main)> git commit -m "Tercer commit: modificacion de archivo de inventario y creacion de playbook nfs_setup"
[main 808fadb] Tercer commit: modificacion de archivo de inventario y creacion de playbook nfs_setup
 2 files changed, 72 insertions(+), 1 deletion(-)
   create mode 100644 nfs_setup.yml
sysadmin@localhost ~/ansible-obligatorio (main)> git push
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 1019 bytes | 339.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:angelch25/ansible-obligatorio.git
   1faa9ed..808fadb main -> main
sysadmin@localhost ~/ansible-obligatorio (main)> git log
commit 808fadb53ffe928c17ecd429c96a83a14c8e000 (HEAD -> main, origin/main)
Author: angelch25 <angelchpq@gmail.com>
Date:   Wed Aug 6 16:46:12 2025 -0300

    Tercer commit: modificacion de archivo de inventario y creacion de playbook nfs_setup

commit 1faa9edeb3c7469013d9b6efd485a264fa6902d
Author: angelch25 <angelchpq@gmail.com>
Date:   Wed Aug 6 12:07:22 2025 -0300

    Segundo commit: creacion de archivo de inventario

commit 1deac86122dac637460614a1ba36a3a66ed35147
Author: angelch25 <angelchpq@gmail.com>
Date:   Wed Aug 6 11:03:06 2025 -0300

    Primer commit: agregado README.md
sysadmin@localhost ~/ansible-obligatorio (main)>

```

Se realiza corrida del playbook `nfs_setup` en ansible obteniendo resultados ok en todas las tareas ejecutadas.

```

~/ansible-obligatorio

TASK [Gathering Facts] *****
ok: [centos1]
ok: [ubuntu]

TASK [Instalar el paquete NFS en CentOS] *****
skipping: [ubuntu]
ok: [centos1]

TASK [Instalar el paquete NFS en Ubuntu] *****
skipping: [centos1]
ok: [ubuntu]

TASK [Asegurar que el servicio NFS esté iniciado y habilitado en CentOS] *****
skipping: [ubuntu]
ok: [centos1]

TASK [Asegurar que el servicio NFS esté iniciado y habilitado en Ubuntu] *****
skipping: [centos1]
ok: [ubuntu]

TASK [Permitir el servicio NFS en el firewall (firewalld para CentOS)] *****
skipping: [ubuntu]
ok: [centos1]

TASK [Permitir el servicio NFS en el firewall (ufw para Ubuntu)] *****
skipping: [centos1]
changed: [ubuntu]

TASK [Crear el directorio compartido en CentOS] *****
skipping: [ubuntu]
ok: [centos1]

TASK [Crear el directorio compartido en Ubuntu] *****
skipping: [centos1]
ok: [ubuntu]

PLAY RECAP *****
centos1      : ok=5    changed=0    unreachable=0    failed=0    skipped=4    rescued=0    ignored=0
ubuntu      : ok=5    changed=1    unreachable=0    failed=0    skipped=4    rescued=0    ignored=0

sysadmin@localhost ~/ansible-obligatorio (main)

```

Se crea playbook `hardening.yml` y se realiza actualización en git creando commit con el archivo.

```

sysadmin@localhost ~/ansible-obligatorio (main) git log
commit c471d62e1a3e21d6f7eb58a549fde01799322d25 (HEAD -> main, origin/main)
Author: angelch25 <angelchpq@gmail.com>
Date: Wed Aug 6 18:24:36 2025 -0300

    Sexto commit: creacion de playbook hardening.yml

commit 81917bcdfe93db4aa9d24e9abfe7d4b7b4e6bc20
Author: angelch25 <angelchpq@gmail.com>
Date: Wed Aug 6 17:55:35 2025 -0300

    Quinto commit: cambios de sintaxis por error de playbook nfs_setup

commit ed2cb5576f5a6ab886727b0b7df933ac7ca1bc1f
Author: angelch25 <angelchpq@gmail.com>
Date: Wed Aug 6 17:46:20 2025 -0300

    Tercer commit: modificacion de playbook nfs_setup

commit 808fadba53ffe928c17ecd429c96a83a14c8e000
Author: angelch25 <angelchpq@gmail.com>
Date: Wed Aug 6 16:46:12 2025 -0300

    Tercer commit: modificacion de archivo de inventario y creacion de playbook nfs_setup

commit 1faa9edeb3c7469013d9b6efdf485a264fa6902d
Author: angelch25 <angelchpq@gmail.com>
Date: Wed Aug 6 12:07:22 2025 -0300

    Segundo commit: creacion de archivo de inventario

commit 1deac86122dac637460614a1ba36a3a66ed35147
Author: angelch25 <angelchpq@gmail.com>
Date: Wed Aug 6 11:03:06 2025 -0300

    Primer commit: agregado README.md

sysadmin@localhost ~/ansible-obligatorio (main) ansible-playbook -i inventory.ini hardening.yml
[WARNING]: Collection community.general does not support Ansible version 2.15.13

```

Se realiza corrida del playbook hardening.yml en ansible obteniendo resultados ok en todas las tareas ejecutadas.

```
PLAY [Hardening de servidor Ubuntu] *****

TASK [Gathering Facts] *****
ok: [ubuntu]
ok: [centos1]

TASK [Actualizar todos los paquetes del sistema] *****
skipping: [centos1]
ok: [ubuntu]

TASK [Habilitar y configurar ufw] *****
skipping: [centos1]
ok: [ubuntu]

TASK [Permitir conexiones SSH entrantes] *****
skipping: [centos1]
ok: [ubuntu]

TASK [Deshabilitar login de root y autenticación por contraseña] *****
skipping: [centos1]
changed: [ubuntu]

TASK [Deshabilitar autenticación por contraseña] *****
skipping: [centos1]
changed: [ubuntu]

TASK [Instalar fail2ban] *****
skipping: [centos1]
changed: [ubuntu]

TASK [Habilitar y arrancar el servicio de fail2ban] *****
skipping: [centos1]
ok: [ubuntu]

RUNNING HANDLER [reiniciar_ssh] *****
changed: [ubuntu]

PLAY RECAP *****
centos1      : ok=1    changed=0    unreachable=0    failed=0    skipped=7    rescued=0    ignored=0
ubuntu      : ok=9    changed=4    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

sysadmin@localhost ~/ansible-obligatorio (main)>
```

Se verifican los archivos del repositorio git y se descargan en ZIP

The screenshot displays the GitHub interface for the repository 'ansible-obligatorio'. The repository is owned by 'angelch25' and is public. It has 1 branch and 0 tags. The file list shows several files: README.md, ansible_basics.md, hardening.yml, inventory.ini, and nfs_setup.yml. A 'Clone' dropdown menu is open, showing options for Local, Codespaces, and HTTPS. The HTTPS option is selected, showing the URL 'https://github.com/angelch25/ansible-obligatorio'. Below the repository list, the README section is visible, titled 'Proyecto Ansible Obligatorio'.

7. Investigación

¿Qué es Ansible? Mencione dos actividades que se puedan hacer con Ansible

Ansible es una herramienta de automatización de TI de código abierto que permite gestionar la configuración, el aprovisionamiento y la implementación de aplicaciones en múltiples sistemas de forma sencilla y eficiente. Funciona sin necesidad de agentes instalados en los nodos, utilizando SSH para conectarse a los servidores.

Actividades que se pueden hacer con Ansible:

- Automatizar la configuración de servidores: Puedes instalar y configurar servicios como Apache, NGINX, MySQL, etc., en múltiples servidores con un solo comando, asegurando que todos estén configurados de forma idéntica.
- Desplegar aplicaciones: Ansible permite automatizar todo el proceso de despliegue de aplicaciones, desde clonar el repositorio, instalar dependencias, hasta reiniciar servicios o realizar actualizaciones sin intervención manual.

¿Qué es un playbook de Ansible?

Un playbook de Ansible es un archivo escrito en formato YAML que define un conjunto de instrucciones (llamadas plays) que Ansible debe ejecutar en uno o más hosts. Los playbooks son la manera principal de automatizar tareas complejas y repetitivas, como la configuración de sistemas, instalación de software o despliegue de aplicaciones.

Características principales de un playbook:

- Está estructurado en listas y diccionarios YAML.
- Es declarativo: describes el estado que deseas alcanzar, no los pasos exactos para lograrlo.
- Define roles, tareas, variables, módulos y más.
- Permite reutilización de código y organización modular.

¿Qué información contiene un inventario de Ansible?

Un inventario de Ansible contiene la información sobre los hosts (servidores) que Ansible va a gestionar. Es una pieza clave porque le dice a Ansible a qué máquinas conectarse, cómo hacerlo y cómo agruparlas para organizar mejor las tareas.

Tipos de inventario:

Estático: archivo plano (por ejemplo, en formato INI o YAML).

Dinámico: generado automáticamente desde una fuente externa (como AWS, Azure, etc.).

Explique que es un módulo de Ansible y dé un ejemplo.

Un módulo de Ansible es una unidad de trabajo reutilizable que Ansible utiliza para realizar tareas específicas en los hosts gestionados. Los módulos son la forma en que Ansible interactúa con los sistemas: realizan operaciones como copiar archivos, instalar paquetes, gestionar servicios, entre otras.

Características de los módulos:

- Autónomos: cada módulo está diseñado para realizar una tarea específica.
- Idempotentes: esto significa que pueden ejecutarse varias veces sin cambiar el estado del sistema si ya está en el estado deseado.
- Abstracción: muchos módulos proporcionan una interfaz sencilla para tareas complejas sin tener que lidiar con los detalles de bajo nivel.
- Multiplataforma: Ansible tiene módulos para interactuar con sistemas Linux, Windows, redes, nubes, etc.

Ejemplo de módulo de Ansible: apt

El módulo apt se usa para gestionar paquetes en sistemas basados en Debian/Ubuntu.

Uso básico:

yaml

- name: Instalar el paquete 'htop' en todos los servidores

hosts: servidores_linux

tasks:

- name: Instalar htop

apt:

name: htop

state: present

update_cache: yes

En este ejemplo:

El módulo apt se usa para instalar el paquete htop en los servidores que pertenecen al grupo servidores_linux.

state: present asegura que el paquete esté instalado. Si ya está instalado, no hará nada.

update_cache: yes actualiza la caché de paquetes antes de instalar.

Tipos de módulos más comunes en Ansible:

- Gestión de paquetes: apt, yum, pip, dnf.
- Gestión de archivos y directorios: copy, template, file, fetch.
- Gestión de servicios: service, systemd, win_service.
- Gestión de usuarios y grupos: user, group.
- Gestión de sistemas: reboot, hostname, timezone.

¿Qué ventajas tiene Ansible sobre otros métodos de automatización?

1. Simplicidad y Facilidad de Uso

Sin necesidad de agentes: Ansible no requiere que se instalen agentes en los nodos remotos, lo que lo hace mucho más fácil de implementar y mantener. Solo necesitas SSH y Python (preinstalado en la mayoría de sistemas Linux).

Sintaxis sencilla y declarativa: Los playbooks de Ansible están escritos en YAML, un formato legible y sencillo de entender, lo que facilita la escritura y la colaboración entre equipos.

2. Idempotencia

Los módulos de Ansible son idempotentes, lo que significa que puedes ejecutar las mismas tareas múltiples veces y el sistema solo realizará cambios si es necesario. Esto asegura que el estado final sea siempre el mismo, sin importar cuántas veces ejecutes el playbook.

3. Configuración como Código (IaC)

Al usar archivos YAML para definir la infraestructura, Ansible permite definir toda la configuración y gestión de infraestructura como código. Esto facilita la versionabilidad, auditoría y la recuperación de configuraciones.

4. Escalabilidad

Aunque Ansible no requiere agentes, puede gestionar un número muy alto de nodos simultáneamente mediante conexiones SSH, lo que lo hace adecuado para entornos grandes.

Control de inventarios: Ansible permite organizar los hosts en grupos, facilitando la administración de entornos complejos.

5. Sin Dependencias Externas

Ansible es autónomo y no necesita servidores adicionales o bases de datos para funcionar. La única dependencia real es SSH y Python, lo que lo hace muy ligero y fácil de instalar.

6. Automatización de Despliegues y Configuración Completa

Ansible no solo se utiliza para tareas de configuración, sino que también es útil para la orquestación de aplicaciones, despliegues de software y gestión de contenedores (como Docker).

Esto lo convierte en una herramienta versátil que abarca desde la configuración de servidores hasta la implementación continua de aplicaciones.

7. Comunidad y Ecosistema

Gran comunidad: Ansible tiene una comunidad activa y extensa, lo que asegura un flujo constante de actualizaciones y nuevos módulos. Además, Ansible Galaxy ofrece roles y colecciones listas para usar, lo que facilita la reutilización de configuraciones y buenas prácticas.

Soporte de múltiples plataformas: Ansible soporta una gran cantidad de plataformas, desde sistemas operativos (Linux, Windows, macOS) hasta servicios de nube (AWS, Azure, GCP) y herramientas de orquestación de contenedores (Docker, Kubernetes).

8. No es necesario tener conocimientos profundos de programación

Ansible está diseñado para ser accesible a personas que no tienen experiencia en programación. Aunque puedes escribir tareas complejas, no es necesario ser un experto en desarrollo de software para crear playbooks útiles.

9. Gestión de Configuraciones y Orquestación de Flujos

Ansible no solo configura máquinas, también puede orquestar flujos de trabajo complejos entre diferentes máquinas. Esto lo hace útil no solo para configuraciones de infraestructura, sino también para la gestión de procesos entre servidores.

10. Ejecución Remota en Tiempo Real

Los playbooks de Ansible se pueden ejecutar de forma ad-hoc, es decir, puedes hacer cambios o ejecutar tareas en sistemas remotos en tiempo real, sin necesidad de una configuración previa. Esto es útil para tareas rápidas o de emergencia.

Comparación con Otros Métodos de Automatización

Chef y Puppet

Chef/Puppet: Son herramientas potentes y ampliamente utilizadas, pero requieren la instalación de agentes en los nodos. Además, usan lenguajes de programación (Ruby para Chef, Puppet DSL para Puppet), lo que puede ser una barrera para quienes no tienen experiencia en programación.

Ansible: No necesita agentes y usa un lenguaje sencillo (YAML), lo que lo hace mucho más accesible.

SaltStack

SaltStack: También es muy potente y permite la ejecución en tiempo real, pero al igual que Chef y Puppet, depende de agentes (aunque tiene modo sin agente).

Ansible: La ventaja de Ansible es que es más fácil de aprender y tiene un modelo sin agente por defecto.

Scripts Tradicionales (Bash, Python, etc.)

Scripts: Si bien los scripts pueden ser una solución rápida, carecen de la idempotencia, modularidad y escalabilidad que ofrece Ansible. Además, no tienen la misma facilidad de uso ni permiten una gestión estructurada de la infraestructura.

Ansible: Proporciona un enfoque más organizado y escalable, con la ventaja adicional de ser más seguro y menos propenso a errores humanos.

En resumen, Ansible destaca principalmente por su simplicidad, accesibilidad, y por la manera en que permite automatizar tareas de forma rápida y eficaz sin la necesidad de una infraestructura compleja.

Referencias

Ansible Documentation (Última actualización 2023). Recuperado de:

<https://docs.ansible.com/>

Ansible Galaxy, para compartir y reutilizar roles y colecciones. Recuperado de:

<https://galaxy.ansible.com/>

Ansible Best Practices (Última actualización 2023). Recuperado de:

https://docs.ansible.com/ansible/latest/user_guide/playbooks_best_practices.html

Ansible Blog - Artículos y noticias sobre nuevas funcionalidades y casos de uso de Ansible. Recuperado de: <https://www.ansible.com/blog>

8. Desafíos encontrados

Requerimiento de contraseña al usar sudo con Ansible

Descripción del problema:

Durante la ejecución de playbooks en Ansible, se presentó el siguiente error relacionado con el uso de privilegios elevados:

ERROR! sudo: a password is required

Esto ocurrió porque Ansible, al intentar ejecutar tareas con privilegios de sudo, necesita que el usuario remoto tenga configurado el acceso sin requerir contraseña. En este caso, el usuario sysadmin requería ingresar una contraseña para ejecutar comandos con sudo, lo cual interrumpía la automatización.

Solución implementada:

Se modificó la configuración de sudo para permitir que el usuario sysadmin ejecute comandos con sudo sin necesidad de ingresar una contraseña.

Pasos realizados:

1. Acceso como root al servidor o mediante un usuario con privilegios sudo.
2. Se ejecutó el siguiente comando para editar de forma segura el archivo de configuración de sudo:
`sudo visudo`
3. Al final del archivo, se agregó la siguiente línea:
`sysadmin ALL=(ALL) NOPASSWD: ALL`
Esto permite que el usuario sysadmin pueda ejecutar cualquier comando con sudo sin necesidad de autenticarse con una contraseña. Este paso se realizó en ambos servidores tanto en Centos como en Ubuntu.
4. Se modificó la línea correspondiente al grupo de administradores (wheel en CentOS y sudo en Ubuntu) para que todos sus miembros pudieran usar sudo sin contraseña. La línea se ajustó de la siguiente forma:

En CentOS:

```
%wheel  ALL=(ALL)      NOPASSWD: ALL
```

En Ubuntu:

```
%sudo    ALL=(ALL:ALL)  NOPASSWD: ALL
```

Después de realizar estos ajustes y guardar los cambios, se volvió a ejecutar el playbook de Ansible, esta vez sin errores. La automatización se completó con éxito.