

Home Lab – Instalación de Aplicaciones en Windows Server con Ansible y Chocolatey

Este laboratorio simula un escenario real de administración de servidores, donde un **Ansible Controller** gestiona servidores Windows de forma remota mediante **WinRM**, aplicando buenas prácticas de automatización y estandarización.

Entorno y software para el LAB:

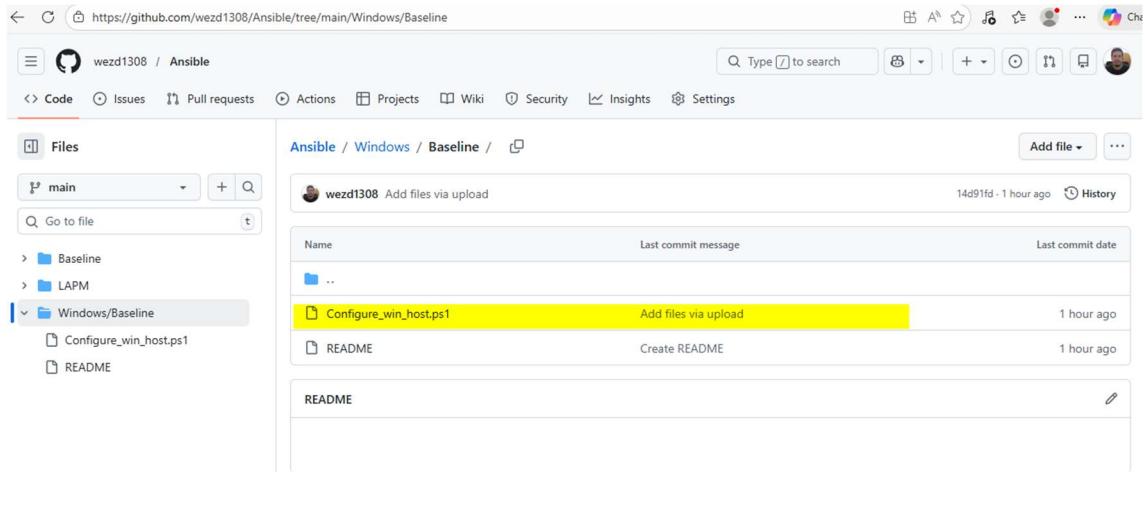
- **VMware Workstation**
- **Ansible Controller:** VM con Linux Ubuntu 24.04.1 LTS con Ansible instalado.
- **Servidores gestionados:** VM con Windows Server 2022
- **Método de conexión:** WinRM sobre HTTP (puerto 5985).
- **Modulos de Ansible para Windows:** [Windows modules — Ansible Documentation](#)

Preparación del servidor Windows Server

Antes de que Ansible pueda gestionar el servidor Windows, es necesario realizar una configuración inicial.

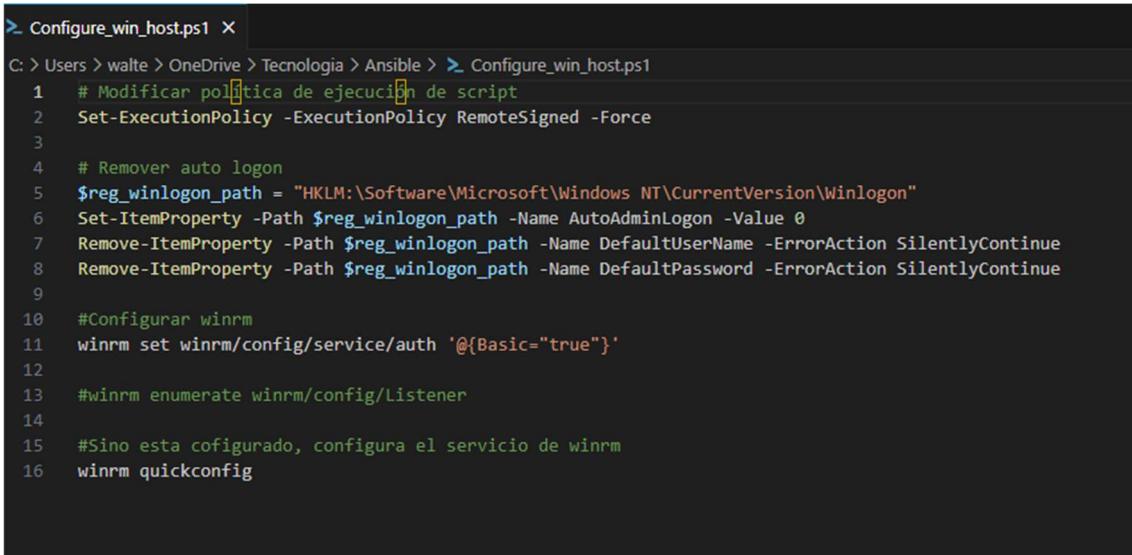
1. Prepara el servidor windows para que pueda ser gestionado por Ansible

Descargar el script `Configure_win_host.ps1` del repositorio [wezd1308/Ansible: Repo para proyectos en Ansible](#),



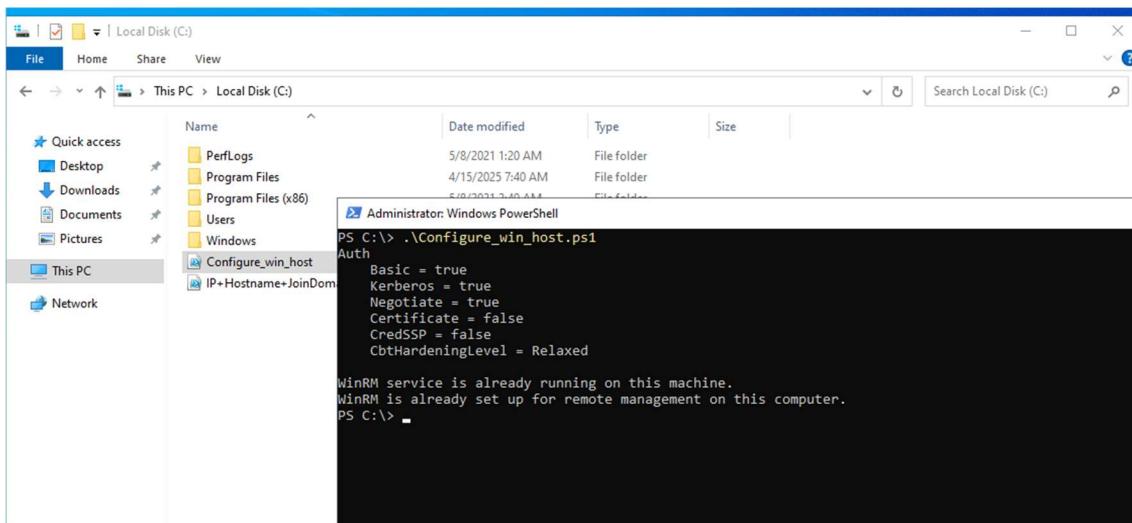
Que hace el script?

- Configura la política de ejecución de powershell para permitir que la ejecución de scripts de manera remota
- Modifica el registro para remover el autologon
- Configura el protocolo winrm de Windows



```
> Configure_win_host.ps1 <
C: > Users > walte > OneDrive > Tecnologia > Ansible > > Configure_win_host.ps1
1 # Modificar politica de ejecucion de script
2 Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Force
3
4 # Remover auto logon
5 $reg_winlogon_path = "HKLM:\Software\Microsoft\Windows NT\CurrentVersion\Winlogon"
6 Set-ItemProperty -Path $reg_winlogon_path -Name AutoAdminLogon -Value 0
7 Remove-ItemProperty -Path $reg_winlogon_path -Name DefaultUserName -ErrorAction SilentlyContinue
8 Remove-ItemProperty -Path $reg_winlogon_path -Name DefaultPassword -ErrorAction SilentlyContinue
9
10 #Configurar winrm
11 winrm set winrm/config/service/auth '@{Basic="true"}'
12
13 #winrm enumerate winrm/config/Listener
14
15 #Sino esta configurado, configura el servicio de winrm
16 winrm quickconfig
```

Ejecutar el script de powershell en el nodo Windows



2. Inventario de Ansible

El inventario define los servidores Linux y Windows gestionados por Ansible, así como las variables que vamos a usar para conectarnos al nodo/nodos Windows. En mi caso uso un archivo llamado inventory.txt, donde usaremos las credenciales definidas en el grupo

[servidores_windows:vars]. El archivo puede ser descargado del repositorio [wezd1308/Ansible: Repo para proyectos en Ansible](https://github.com/wezd1308/Ansible/tree/main/Windows/Baseline)

The screenshot shows a GitHub repository page for 'wezd1308 / Ansible'. The 'Windows' folder contains a 'Baseline' folder. Inside 'Baseline' are files: 'Configure_win_host.ps1', 'README', 'install_pkg_choco_v2.yml', and 'inventory.txt'. A commit titled 'wezd1308 Add files via upload' is visible, showing the addition of 'Configure_win_host.ps1', 'README', 'install_pkg_choco_v2.yml', and 'inventory.txt'. The 'inventory.txt' file is highlighted with a yellow background.

```
ansible@lablansi:~$ ls -ll
total 1676
-rw-rw-r-- 1 wzambrano wzambrano      42 abr 17 2025 ansible.cfg
-rw-rw-r-- 1 ansible   ansible      2967 dic 19 17:04 baseline_linux_debian.yml~
-rw-rw-r-- 1 ansible   ansible      2219 dic 19 18:11 Baseline_LNX_debian_v2.yml
-rw-rw-r-- 1 ansible   ansible      2214 dic 19 18:00 Baseline_LNX_debian_v2.yml.save
-rw-rw-r-- 1 ansible   ansible      2979 dic 19 17:15 baseline_lnx.yaml
-rw-r--r-- 1 root      root        217 ene 12 18:48 comando_shell_win.yml
-rw-rw-r-- 1 ansible   ansible      247 ene 12 18:42 comando_win.yml
-rwxrwxrwx 1 ansible   ansible      30 abr 22 2025 crear_archivo.yml -> /etc/ansible/crear_archivo.yml
drwxr-xr-x 2 root      root        4096 nov 18 13:34 curso
drwxrwxrwx 2 root      root        4096 ene  8 22:35 iic
-rw-rw-r-- 1 ansible   ansible      1698 ene 12 19:13 install_pkg_choco_v2.yml
-rw-rw-r-- 1 ansible   ansible      1188 ene 12 14:41 Install_pkg_choco.yml
-rw-r--r-- 1 root      root        1186 ene  8 19:48 install_tomcat_win.yml
drwxrwxrwx 2 root      root        4096 abr 17 2025 inventario
-rw-rw-r-- 1 ansible   ansible      429 ene 12 15:14 inventory.txt
-rw-rw-r-- 1 ansible   ansible      846 dic 19 21:50 lamp.yaml
-rw-rw-r-- 1 ansible   ansible      1645497 abr 21 2025 modulos.txt
-rw-r--r-- 1 root      root        19 dic 19 22:19 phpinfo.php
drwxrwxrwx 1 root      root        22 abr 21 2025 ping2.yml -> /etc/ansible/ping2.yml
drwxrwxrwx 1 root      root        21 abr 21 2025 ping.yml -> /etc/ansible/ping.yml
-rw-rw-r-- 1 ansible   ansible      795 ene  7 20:39 playbook_win.yml
ansible@lablansi:~$
```

Inventario:

```
ansible@lablansi:~$ cat inventory.txt
[servidores_linux]
#192.168.91.129
#192.168.91.150
#192.168.91.145
#192.168.91.162
192.168.91.160
192.168.91.164

[servidores_windows]
192.168.91.146
192.168.91.147
192.168.91.152

[servidores_windows:vars]
ansible_user=Administrator
ansible_password=*****
ansible_port=5985
ansible_winrm_transport=ntlm
ansible_connection=winrm
ansible_winrm_scheme=http
ansible_winrm_server_cert_validation=ignore
```

```
[servidores_windows]
```

```
192.168.91.146
192.168.91.147
192.168.91.152
```

```
[servidores_windows:vars]
```

```
ansible_user=Administrator
ansible_password=*****
ansible_port=5985
ansible_winrm_transport=ntlm
ansible_connection=winrm
ansible_winrm_scheme=http
ansible_winrm_server_cert_validation=ignore
```

Estas variables permiten a Ansible conectarse correctamente vía WinRM.

3. Playbook de Ansible – Instalación con Chocolatey

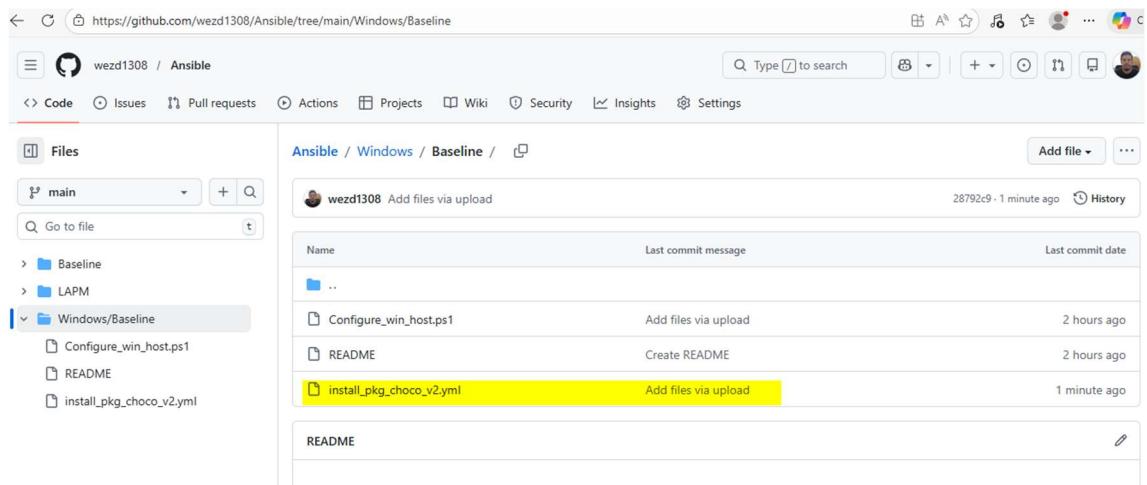
El siguiente playbook realiza:

1. Instalación de Chocolatey.
2. Verificación de su instalación.
3. Instalación de aplicaciones comunes de administración y soporte.

◆ Aplicaciones instaladas

- Sysinternals
- 7-Zip
- Notepad++
- Postman
- TreeSize
- WinSCP
- PowerShell 7

El playbook utiliza los módulos nativos `win_command` y `win_chocolatey`, el archivo de playbook esta en el repositorio [wezd1308/Ansible: Repo para proyectos en Ansible](https://github.com/wezd1308/Ansible/tree/main/Windows/Baseline)



The screenshot shows a GitHub repository interface. The URL in the address bar is <https://github.com/wezd1308/Ansible/tree/main/Windows/Baseline>. The repository name is `wezd1308 / Ansible`. The 'Code' tab is selected. On the left, there's a sidebar with 'Files' and a tree view showing 'main', 'Baseline', 'LAPM', and a 'Windows/Baseline' folder containing 'Configure_win_host.ps1', 'README', and 'install_pkg_choco_v2.yml'. The 'install_pkg_choco_v2.yml' file is highlighted with a yellow background. The main pane displays the contents of the 'Windows/Baseline' folder, including a commit from 'wezd1308' adding files via upload, a 'Configure_win_host.ps1' file, a 'README' file, and the highlighted 'install_pkg_choco_v2.yml' file. The 'install_pkg_choco_v2.yml' file has a timestamp of '1 minute ago'.

4. Ejecución del playbook

Validamos la conexión previamente, en este caso solo lo ejecutare sobre el servidor 192.168.91.152 del grupo servidores_windows de mi inventario

```
ansible servidores_windows -i inventory.txt -m win_ping --limit 192.168.91.152
```

```
ansible@lablansi:~$ ansible servidores_windows -i inventory.txt -m win_ping --limit 192.168.91.152
[WARNING]: Ansible is being run in a world writable directory (/home/ansible), ignoring it as an ansible.cfg source. For more information see
https://docs.ansible.com/ansible-devel/reference_appendices/config.html#cfg-in-world-writable-dir
192.168.91.152 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
ansible@lablansi:~$ |
```

Usamos –limit para ejecutar el playbook únicamente sobre un servidore específico

```
ansible-playbook -i inventory.txt install_pkg_choco_v2.yml --limit 192.168.91.152
```

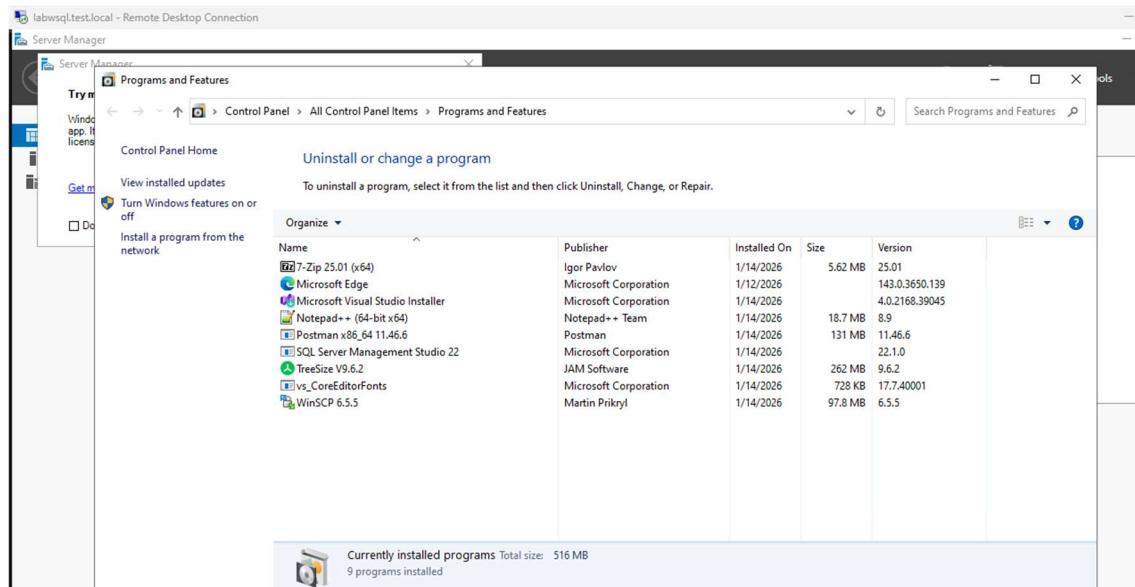
```
ansible@lablansi:~$ ansible-playbook -i inventory.txt install_pkg_choco_v2.yml --limit 192.168.91.152
[WARNING]: Ansible is being run in a world writable directory (/home/ansible), ignoring it as an ansible.cfg source. For more information see
https://docs.ansible.com/ansible-devel/reference_appendices/config.html#cfg-in-world-writable-dir
PLAY [Instalar paquetes con Chocolatey] *****
TASK [Instalar Chocolatey] *****
changed: [192.168.91.152]
TASK [Asegurar que Chocolatey esté instalado] *****
ok: [192.168.91.152]
TASK [Instalar Sysinternals] *****
changed: [192.168.91.152]
TASK [Instalar 7zip] *****
changed: [192.168.91.152]
TASK [Instalar Notepad++] *****
changed: [192.168.91.152]
TASK [Instalar Postman] *****
changed: [192.168.91.152]
TASK [Instalar TeeSize] *****
changed: [192.168.91.152]
TASK [Instalar Winscp] *****
changed: [192.168.91.152]
TASK [Instalar Powershell 7] *****
changed: [192.168.91.152]
```

Nota: para la instalación de chocolatey y todo el software instalado con este módulo, el servidor Windows debe tener salida a internet

Resultado esperado

- Chocolatey instalado correctamente.
- Aplicaciones desplegadas de forma automática.
- Servidor Windows listo para tareas administrativas.
- Proceso repetible y escalable.

Validamos en la maquina Windows que todas las aplicaciones se hayan instalado correctamente



Conclusión

Este laboratorio demuestra cómo Ansible puede ser utilizado eficazmente para administrar servidores Windows, integrándose con Chocolatey para la gestión de software.

El enfoque es aplicable tanto a entornos de laboratorio como a escenarios reales de infraestructura IT.

Autor

Walter Zambrano

Administrador de Sistemas / Infraestructura IT/DevOps

Repo: [Ansible/Windows/Baseline at main · wezd1308/Ansible](#)

#Ansible #WindowsServer #Chocolatey #Automation #DevOps #SysAdmin