

Tiempo para aprender juntos

Dirección de Servicios de
Infraestructura y Operaciones

Marzo 2021

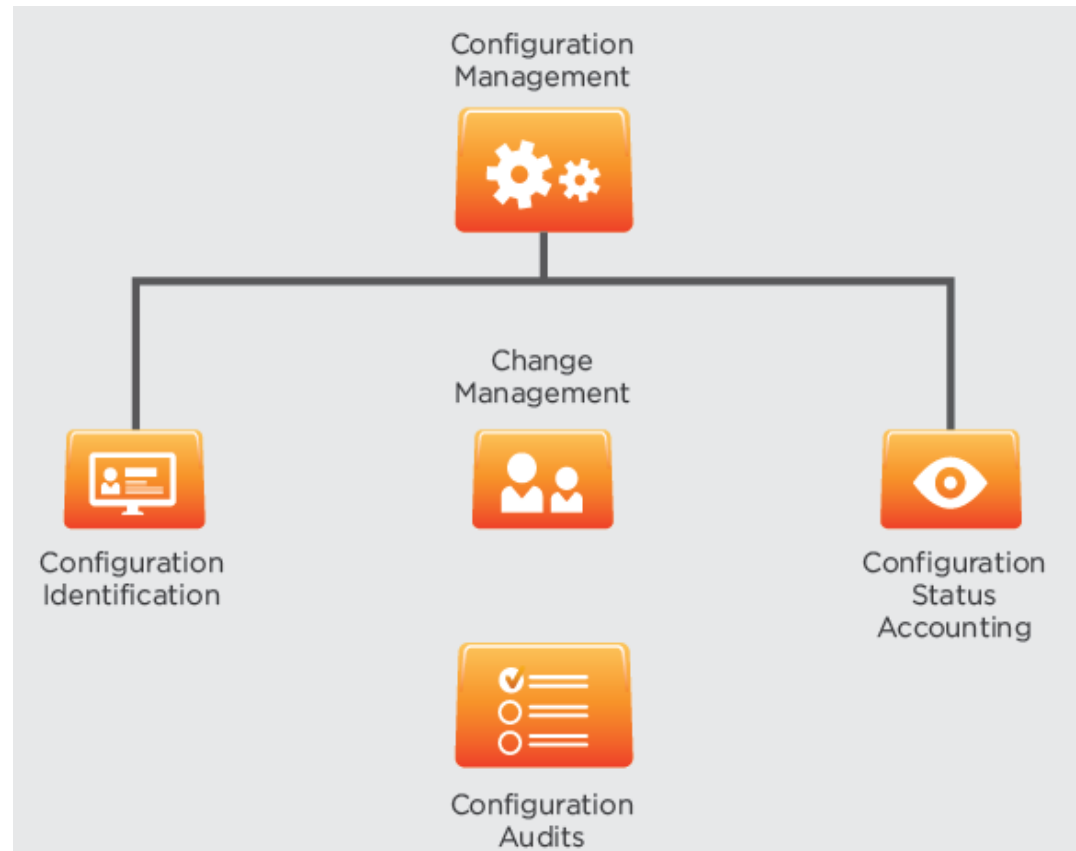
Ansible



- ¿Qué es Configuration Management?
- ¿Qué es Ansible?
- ¿Cómo aprender Ansible?
- ¿En que áreas se usa Ansible?
- ¿Cómo funciona Ansible?
- Laboratorios

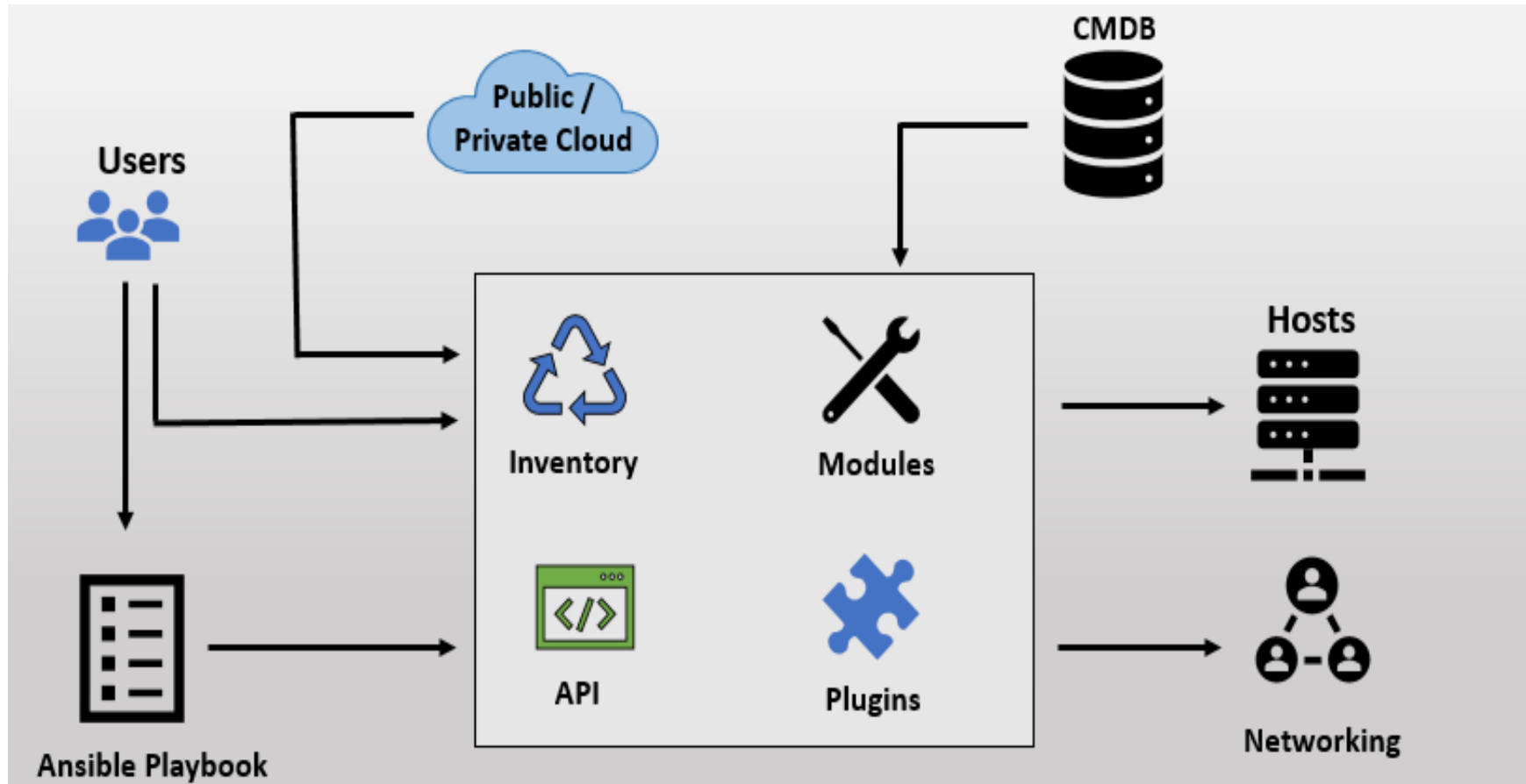
¿Qué es Configuration Management?

- Es el **proceso** de **manejar cambios** en un **sistema** de una manera que asegure la **integridad** a lo **largo** del **tiempo**.
- Involucra el uso de **herramientas** y **procesos** que faciliten la **automatización** y la **observabilidad**.



¿ Qué es Ansible?

- Es un software **open-source** desarrollado en **Python** de gestión de la configuración, que permite **centralizar** (a través de **gestión SSH**) la **configuración** de **servidores, dispositivos de red, dispositivos de seguridad, proveedores de Nube** de una forma **automatizada** usando **YAML**.



```
#Web Upgrade Playbook
#Web Upgrade Playbook
1
2
3  name: Upgrade Web1
4  hosts: web1
5  tasks:
6  -
7    name: stop web1
8    command: /sbin/service httpd stop
9  -
10   name: Execute upgrade script
11   script: upgrade_script.py
12  -
13   name: Restart web1
14   command: /sbin/shutdown -r now
15  -
16  name: Upgrade Web2
17  hosts: web2
18  tasks:
19  -
20   name: stop web1
21   command: /sbin/service httpd stop
22  -
23   name: Execute upgrade script
24   script: upgrade_script.py
25  -
26   name: Restart web1
27   command: /sbin/shutdown -r now
28
```

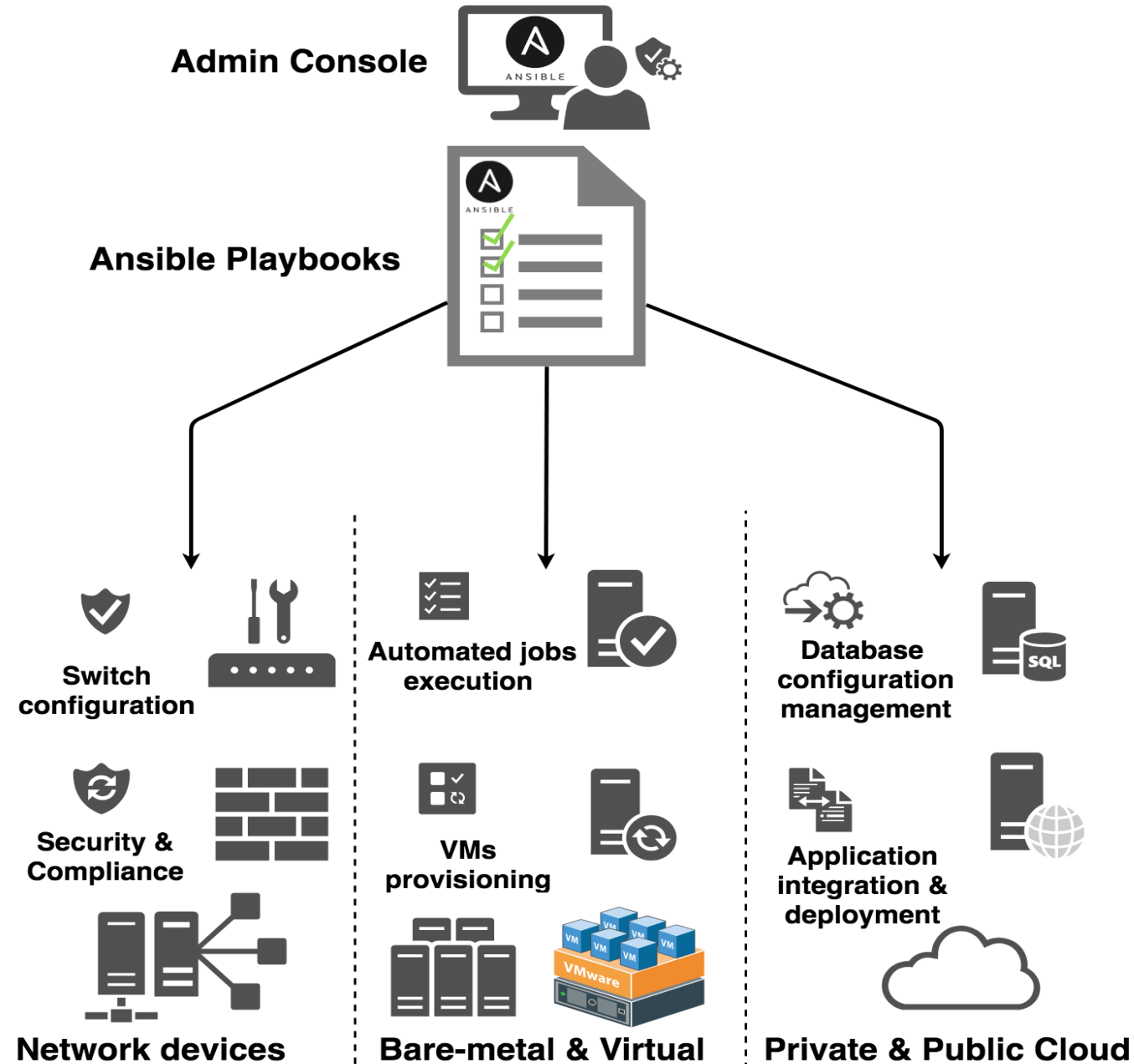
- Sus **ventajas**:
 - **Gratis**: Es open-source.
 - **Simple de configurar y usar**: Conocimientos elementales de programación son requeridos.
 - **Poderoso**: Permite modelar complejos flujos.
 - **Flexible**: Puedes orquestar un ambiente ya sea en premisas o nube.
 - **Sin agente**: No necesitas instalar agentes.
 - **Eficiente**: No consume tantos recursos del dispositivo en cuestión.

- Sus **componentes**:
 - **Nodo**: Objeto a administrar: servidor, firewall, switch, balanceador, etc.
 - **Comandos Ah-Hoc**: Herramienta en línea de comandos para mecanizar una única tarea en uno o más nodos administrados.
 - **Play**: Lista de tareas a realizar en los clientes especificados.
 - **Playbooks**: Describen la configuración que se desea se apliquen en los equipos gestionados.
 - **Tareas**: Definición de una acción a realizar.
 - **Módulos**: También conocidos como library plugins, que se pueden usar desde línea de comando o en un tarea de un Playbook (PowerShell en Windows y Python en Linux).
 - **Facts**: Información útil de los clientes.
 - **Inventario**: Incluye información estática, dinámica de los clientes administrados.
 - **Roles**: Es una agrupación de tareas, archivos y plantillas que puede ser reutilizadas.

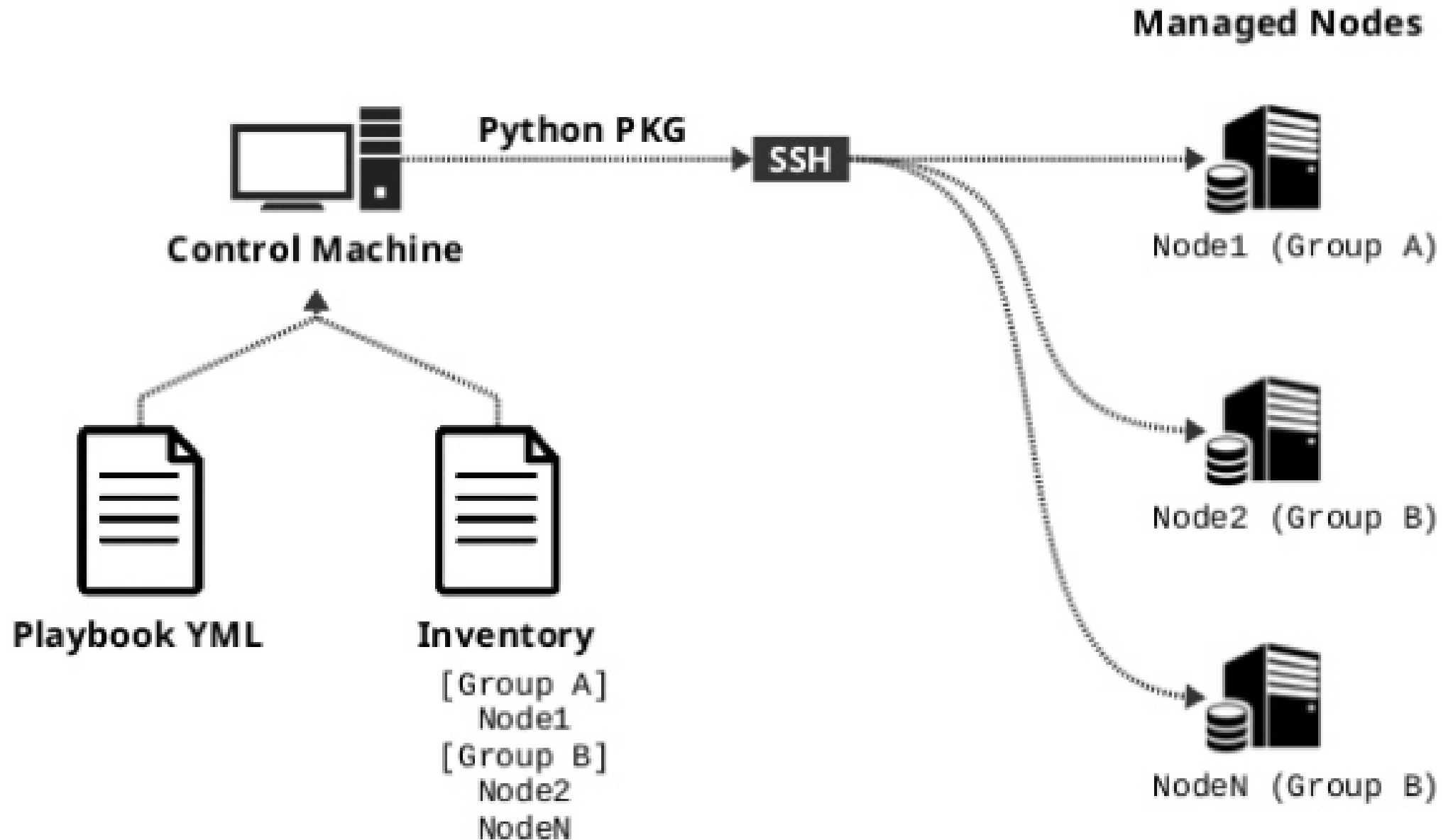
¿Cómo aprender Ansible?

- En el sitio de Ansible.com (<https://www.ansible.com/>)
- Sobre los módulos:
 - https://docs.ansible.com/ansible/2.8/modules/list_of_all_modules.html
- En **Internet** hay buenos cursos de **Ansible**.
 - **Udemy**
 - **Coursera**
- **Buscando** en foros (<https://stackoverflow.com/>) o sitios (<https://medium.com/>).
- **Aplicarlo** a una **situación** en **2021** que consideras puedas resolver a través de **scripting**.

¿En qué áreas se usa Ansible?



¿Cómo funciona Ansible?





- En **Sharepoint** se encuentra el archivo con todo el inventario de VMs creadas, selecciona **2 VMs** (registra en el Excel tu nombre).
- Ve al siguiente repositorio:
 - https://github.com/HugoAquinoNavarrete/ansible_scripting
- Haz un “**Fork**” desde GitHub.
- **Clónalo** en solo **una VM**.
 - `git clone https://github.com/<tu_usuario>/ansible_scripting.git`
- Descarga la llave (archivo **.pem**) en el directorio “**ansible_scripting**” de la VM desde donde clonaste el repositorio.



- Ve al directorio “**ansible_scripting**” y edita el archivo “**ansible_inventario.txt**” para registrar la 2 IPs de las VMs.
 - `cd ansible_scripting`
 - `nano ansible_inventario.txt`

- El archivo debe quedar así:

```
[servers]  
servidor_1 ansible_host=1.2.3.4  
servidor_2 ansible_host=5.6.7.8
```

- Instala ansible **solo** en “servidor_1” de la siguiente manera:
 - `sudo apt update`
 - `sudo apt install -y ansible`
 - `ansible --version`



- Ve al directorio “**playbooks**” y revisa el contenido de:
 - `cd playbooks`
 - `cat ping.yml`
- Ahora en el directorio “**bin**”, revisa y ejecuta el script “**01-ping.sh**”.
 - `cd ../bin`
 - `./01-ping.sh`
- ¿Qué observas de la salida de la ejecución?



- Siguiendo en directorio “**bin**” revisa el contenido del script:
 - “**02-mensaje.sh**”
- Antes de ejecutar el script, ve al directorio “**playbooks**” y revisa el archivo “**mensaje.yml**” y posteriormente ejecuta el script.
 - `cat ../playbooks/mensaje.yml`
 - `./02-mensaje.sh`
- ¿Qué observas de la salida de la ejecución?



- **Reto 1 - Completa el script “reto-1.sh” para que haga lo siguiente:**
 - Ejecuta el playbook contenido en el archivo “**reto1.yml**”, el cual hará lo siguiente:
 - Imprimir el nombre completo del mes.
 - Imprimir el día del mes.
- **Reto 2 - Completa el script “reto-2.sh” para que haga lo siguiente:**
 - Ejecuta el playbook contenido en el archivo “**reto2.yml**”, el cual hará lo siguiente:
 - Instalar la versión más reciente de ansible.



- Prueba desde una ventana del navegador cargando la dirección IP de cualquiera de las 2 VMs o desde la línea de comando ejecuta:
 - `curl <ip_publica_vm>`
- Siguiendo en directorio “**bin**” revisa el contenido de los scripts:
 - “03-nginx_instala.sh”
 - “04-nginx_sube.sh”
 - “05-nginx_baja.sh”
 - “06-nginx_desinstala.sh”



- Antes de ejecutar el script, ve al directorio “**playbooks**” y revisa los archivos “**nginx_instala.yml**”, “**nginx_sube.yml**”, “**nginx_baja.yml**” y “**nginx_desinstala.yml**”; posteriormente ejecuta el script.
 - `cat ../playbooks/nginx_instala.yml`
 - `cat ../playbooks/nginx_sube.yml`
 - `cat ../playbooks/nginx_baja.yml`
 - `cat ../playbooks/nginx_desinstala.yml`

 - `./03-nginx_instala.sh`
 - `./04-nginx_sube.sh`
 - `./05-nginx_baja.sh`
 - `./06-nginx_desinstala.sh`



- Siguiendo en directorio “**bin**” revisa el contenido del script:
 - “**07-nginx_despliega.sh**”
- Antes de ejecutar el script, ve al directorio “**playbooks**” y revisa el archivo “**nginx_despliega.yml**”, posteriormente ejecuta el script.
 - `cat ../playbooks/nginx_despliega.yml`
 - `./nginx_despliega.sh`
- Baja el servicio nginx (**./05-nginx_baja.sh**) y vuélvelo a subir (**./06-nginx_sube.sh**) para que los cambios surtan efecto.
- Prueba desde una ventana del navegador cargando la dirección IP de cualquiera de las 2 VMs o desde la línea de comando ejecuta:
 - `curl <ip_publica_vm>`



- Siguiendo en directorio “**bin**” revisa el contenido de los scripts:
 - “**08-directorios_crea.sh**”
 - “**09-directorios_borra.sh**”
- Antes de ejecutar los scripts, ve al directorio “**playbooks**” y revisa los archivos “**directorios_crea.yml**” y “**directorios_borra.yml**”; posteriormente ejecuta los scripts.
 - `cat ../playbooks/directorios_crea.yml`
 - `cat ../playbooks/directorios_borra.yml`
 - `./08-directorios_crea.sh`
 - `./09-directorios_borra.sh`
- ¿Qué observas de la salida de la ejecución?



- **Reto 3 - Completa el script “reto-3.sh” para que haga lo siguiente:**
 - Ejecuta el playbook contenido en el archivo “**reto3.yml**”, el cual hará lo siguiente:
 - Crear un directorio que se llame “retos” bajo \$HOME
 - Crear 5 directorios (con los nombres que tu prefieras) dentro del directorio “retos”.
- **Reto 4 - Completa el script “reto-4.sh” para que haga lo siguiente:**
 - Ejecuta el playbook contenido en el archivo “**reto4.yml**”, el cual hará lo siguiente:
 - Borrará todos los directorios creados en el reto 3.



- Siguiendo en directorio “**bin**” revisa el contenido del script:
 - “**10-git_clona.sh**”
- Antes de ejecutar el script, ve al directorio “**playbooks**” y revisa el archivo “**git_clona.yml**” y posteriormente ejecuta el script.
 - `cat ../playbooks/git_clona.yml`
 - `./10-git_clona.sh`
- ¿Qué observas de la salida de la ejecución?



- Siguiendo en directorio “**bin**” revisa el contenido del script:
 - “**11-permisos.sh**”
- Antes de ejecutar el script, ve al directorio “**playbooks**” y revisa el archivo “**permisos.yml**” y posteriormente ejecuta el script.
 - `cat ../playbooks/permisos.yml`
 - `./11-permisos.sh`
- ¿Qué observas de la salida de la ejecución?



- **Reto 5 - Completa el script “reto-5.sh” para que haga lo siguiente:**
 - Ejecuta el playbook contenido en el archivo “**reto5.yml**”, el cual hará lo siguiente:
 - Clonar uno de tus repositorios de GitHub o uno que encuentres.
- **Reto 6 - Completa el script “reto-6.sh” para que haga lo siguiente:**
 - Ejecuta el playbook contenido en el archivo “**reto6.yml**”, el cual hará lo siguiente:
 - Cambiar permisos para que solamente el usuario “ubuntu” pueda leer los archivos.



- Siguiendo en directorio “**bin**” revisa el contenido del script:
 - “**12-archivos_lectura.sh**”
- Antes de ejecutar el script, ve al directorio “**playbooks**” y revisa el archivo “**archivos_lectura.yml**” y posteriormente ejecuta el script.
 - `cat ../playbooks/archivos_lectura.yml`
 - `./12-archivos_lectura.sh`
- ¿Qué observas de la salida de la ejecución?



- Siguiendo en directorio “**bin**” revisa el contenido del script:
 - “**13-archivos_copia.sh**”
- Antes de ejecutar el script, ve al directorio “**playbooks**” y revisa el archivo “**archivos_copia.yml**” y posteriormente ejecuta el script.
 - `cat ../playbooks/archivos_copia.yml`
 - `./13-archivos_copia.sh`
- ¿Qué observas de la salida de la ejecución?



- **Reto 7 - Completa el script “reto-7.sh” para que haga lo siguiente:**
 - Ejecuta el playbook contenido en el archivo “**reto7.yml**”, el cual hará lo siguiente:
 - Cargar el contenido del archivo “retos.txt” que se encuentra en el directorio “archivos”.
 - Crear directorio que se llame “retos” bajo \$HOME.
 - Crear un directorio por cada línea del archivo “retos.txt” bajo el directorio “retos”.
- **Reto 8 - Completa el script “reto-8.sh” para que haga lo siguiente:**
 - Ejecuta el playbook contenido en el archivo “**reto8.yml**”, el cual hará lo siguiente:
 - Cambiar permisos para que solamente el usuario “ubuntu” pueda leer y escribir los archivos.



- Siguiendo en directorio “**bin**” revisa el contenido del script:
 - “**14-archivos_comprime.sh**”
- Antes de ejecutar el script, ve al directorio “**playbooks**” y revisa el archivo “**archivos_comprime.yml**” y posteriormente ejecuta el script.
 - `cat ../playbooks/archivos_comprime.yml`
 - `./14-archivos_comprime.sh`
- ¿Qué observas de la salida de la ejecución?



- Siguiendo en directorio “**bin**” revisa el contenido del script:
 - “**15-archivos_descomprimir.sh**”
- Antes de ejecutar el script, ve al directorio “**playbooks**” y revisa el archivo “**archivos_descomprime.yml**” y posteriormente ejecuta el script.
 - `cat ../playbooks/archivos_descomprime.yml`
 - `./15-archivos_descomprime.sh`
- ¿Qué observas de la salida de la ejecución?



- Siguiendo en directorio “**bin**” revisa el contenido de los scripts:
 - “**16-variables_1.sh**”
 - “**17-variables_2.sh**”
 - “**18-variables_3.sh**”
- Antes de ejecutar los scripts, ve al directorio “**playbooks**” y revisa los archivos “**variables_1.yml**”, “**variables_2.yml**” y “**variables_3.yml**”; posteriormente ejecuta los scripts.
 - `cat ../playbooks/variables_1.yml`
 - `cat ../playbooks/variables_2.yml`
 - `cat ../playbooks/variables_3.yml`
 - `./16-variables_1.sh`
 - `./17-variables_2.sh`
 - `./18-variables_3.sh`
- ¿Qué observas de la salida de la ejecución?



- Siguiendo en directorio “**bin**” revisa el contenido del script:
 - “**19-condicionales.sh**”
- Antes de ejecutar el script, ve al directorio “**playbooks**” y revisa el archivo “**condicionales_1.yml**” y posteriormente ejecuta el script.
 - `cat ../playbooks/condicionales_1.yml`
 - `./19-condicionales.sh`
- ¿Qué observas de la salida de la ejecución?



- Siguiendo en directorio “**bin**” revisa el contenido del script:
 - “**20-condicionales.sh**”
- Antes de ejecutar el script, ve al directorio “**playbooks**” y revisa el archivo “**condicionales_2.yml**” y posteriormente ejecuta el script.
 - `cat ../playbooks/condicionales_2.yml`
 - `./20-condicionales.sh`
- ¿Qué observas de la salida de la ejecución?



- Siguiendo en directorio “**bin**” revisa el contenido del script:
 - “**21-archivos_crea.sh**”
- Antes de ejecutar el script, ve al directorio “**playbooks**” y revisa el archivo “**archivos_crea.yml**” y posteriormente ejecuta el script.
 - `cat ../playbooks/archivos_crea.yml`
 - `./21-archivos_crea.sh`
- ¿Qué observas de la salida de la ejecución?



- **Reto 9 - Completa el script “reto-9.sh” para que haga lo siguiente:**
 - Ejecuta el playbook contenido en el archivo “**reto9.yml**”, el cual hará lo siguiente:
 - Cargar el contenido del archivo “retos.txt” que se encuentra en el directorio “archivos”.
 - Crear directorio que se llame “desafios” bajo \$HOME.
 - Crear un directorio por cada línea del archivo “retos.txt” bajo el directorio “retos”.
 - Crea en cada directorio los siguientes archivos:
 - Archivo “informacion.txt” que contenga 5 líneas.
 - Archivo “informacion.sh” que sea un bash Shell que imprima un texto de 5 palabras así como la fecha.

- Tiempo para repasar jugando



Gracias

