



Instalación automatizada

SMX Sistemes Operatius en Xarxa

Harjodh Singh Kaur, Jose Manuel Perez, Mireia García Gómez
2 Cicle Mitjà

SMX2C



Esta obra está sujeta a una licencia de

[Reconocimiento-NoComercial-SinObraDerivada 3.0 España de Creative Commons](#)

Resumen del proyecto:

Nuestro proyecto automatiza la configuración de un servidor **Ubuntu Server 24.04** mediante scripts lanzados desde una máquina remota **Ubuntu Desktop 24.04**.

El primer Script prepara tanto la máquina remota desde la que se lanza como la de destino: actualiza paquetes, instala herramientas como **Ansible** y **sshpass**, configura la red, cambia contraseñas, habilita el acceso **SSH** y despliega un controlador de dominio **Samba AD DC** mediante **Playbooks**.

El segundo Script, ejecutado desde el servidor, configura una máquina **cliente**: establece el hostname, habilita **SSH** y aplica la configuración necesaria mediante otro **Playbook**.

Con ambos **Scripts**, se consigue una implementación rápida y con apenas intervención manual, reduciendo errores y ahorrando tiempo.

Palabras clave:

UBUNTU
SAMBA AD DC
DHCP
KERBEROS
OPEN SOURCE
ANSIBLE
SSH

Abstract:

This project focuses on automating the configuration of an **Ubuntu Server 24.04** through the use of scripts executed from a remote **Ubuntu Desktop 24.04** system.

The first Script is responsible for preparing both the remote workstation and the target server. It updates system packages, installs essential tools such as **Ansible** and **sshpass**, configures the network settings, updates root passwords, enables **SSH** access, and deploys a **Samba AD DC** via **Playbooks**.

The second Script, executed from the server, handles the configuration of a **client** machine. It sets the hostname, enables SSH access, and applies the required configurations through an additional **Playbook**.

Together, these scripts enable a streamlined, efficient deployment process that minimizes manual intervention, reduces the likelihood of configuration errors, and saves valuable time.

Keywords:

UBUNTU
SAMBA AD DC
DHCP
KERBEROS
OPEN SOURCE
ANSIBLE
SSH

ÍNDICE

1 Introducción	5
1.1 Contexto y justificación	5
1.2 Objetivos	6
1.2.1 Objetivo general	6
1.2.2 Objetivos específicos	6
1.3 Estrategia y planificación del proyecto	7
1.4 Metodología de trabajo	8
1.5 Estudio económico y presupuestario	8
1.5.1 Tareas y componentes del proyecto	8
1.5.2 Costos de desarrollo	8
1.5.3 Costes de mantenimiento	9
1.5.4 Oportunidades de beneficio	9
1.5.5 Decisión sobre la continuidad del proyecto	9
2 Descripción del proyecto	10
2.1 Análisis de requisitos	10
2.1.1 Requisitos funcionales	11
2.1.2 Requisitos no funcionales	11
2.2 Previsión de tareas de investigación	13
2.3 Tecnologías	13
2.3.1 Comparativa de las tecnologías valoradas	14
2.3.2 Tecnologías escogidas	14
2.4 Estructura del proyecto	15
2.5 Descripción de los componentes:	15
2.5.1 Componente 1	15
2.5.3 Componente 3	15
2.5.4 Componente 4	15
2.5.4 Componente 5	15
2.6 Definición de las tareas [proyecto de investigación].	16
2.7 Definición de las funcionalidades.	17
2.7.1. Configuración automática del servidor Ubuntu 24.04	17
2.7.2. Automatización de tareas con Ansible	17
2.7.3. Configuración automática de un cliente Ubuntu	17
2.7.4. Conexión de una máquina Windows 10 al dominio	17
3 Otros capítulos	18
4 Comandos Y Soluciones	18
4.1 Comandos	18
4.2 Consecución de los objetivos	22
4.3 Valoración de la metodología y planificación	22
4.4 Visión de futuro	22
5. Glossario	23
6. Bibliografía	24
7. Conclusión	25

1 Introducción

1.1 Contexto y justificación

El proyecto trata sobre la automatización de la configuración de un servidor **Ubuntu Server 24.04**, mediante un script desde una máquina **Ubuntu Desktop 24.04**, la cual lanza el script sobre la máquina server para llevar a cabo todas las instrucciones introducidas de manera automática sin apenas interacción del recurso humano.

El primer Script se encarga de preparar tanto la máquina local como el servidor remoto. Desde la máquina Desktop, se actualizan los repositorios y se instalan herramientas necesarias como **sshpass** y **Ansible**. Luego, se conecta al servidor para aplicar una configuración personalizada de red, establecer el hostname como “**Server**”, habilitar el acceso SSH para root y cambiar la contraseña del mismo. A continuación, se genera una clave SSH (si no existe) y se copia al servidor para facilitar futuras conexiones sin contraseña. El script también lanza varios *playbooks* de Ansible que automatizan la instalación y configuración de un controlador de dominio Samba AD DC, dejando listo el servidor con todos los servicios establecidos.

El segundo Script complementa el proceso conectándose primero al servidor (ya configurado) y desde allí a una máquina cliente. En esta, también se cambia la contraseña de root, se asigna el hostname “**Cliente1**” y se habilita el acceso SSH. Luego, se copia la clave SSH al cliente y se lanza un *playbook* de Ansible desde el servidor, aplicando todas las configuraciones diseñadas.

En conjunto, ambos Scripts permiten automatizar la configuración completa de la red, reduciendo errores y ahorrando tiempo.

1.2 Objetivos

El objetivo principal es **automatizar tareas repetitivas y críticas de administración de sistemas** mediante **scripts Bash y Ansible**, reduciendo así la intervención manual, los errores humanos y el tiempo de despliegue.

1.2.1 Objetivo general

Diseñar y desarrollar una solución automatizada para configurar servidores y clientes en una red local, utilizando tecnologías open source que permitan una implementación rápida, segura y eficiente.

1.2.2 Objetivos específicos

Desarrollar scripts automatizados que准备en y configuren tanto el servidor como las máquinas cliente, minimizando la intervención manual.

Instalar y configurar Ansible y herramientas auxiliares como sshpass en la máquina remota para facilitar la gestión automatizada de los sistemas.

Establecer una configuración de red personalizada en el servidor Ubuntu Server 24.04, incluyendo hostname, IP estática, acceso SSH y contraseñas seguras.

Implementar un controlador de dominio Samba AD DC de forma automatizada mediante playbooks de Ansible.

Automatizar la configuración básica de una máquina cliente, incluyendo hostname, acceso SSH y despliegue de configuraciones desde el servidor.

Garantizar la conexión segura entre máquinas usando claves SSH para permitir comunicaciones sin contraseñas entre los nodos.

Reducir el tiempo de despliegue y los errores humanos durante la configuración inicial de servidores y clientes en entornos de red.

1.3 Estrategia y planificación del proyecto

Para llevar a cabo este proyecto, hay varias estrategias posibles:

- **Desarrollar una infraestructura totalmente nueva**, diseñada desde cero según las necesidades de la empresa
- **Adaptar una infraestructura existente**, aprovechando la que ya hay y haciendo las mejoras necesarias.
- **Implementar una solución híbrida**, combinando elementos nuevos con otros ya disponibles para optimizar costes y recursos.

Estrategia elegida

La estrategia escogida es desarrollar una infraestructura nueva, puesto que permite garantizar que la red cumpla con todos los requisitos de rendimiento, seguridad y escalabilidad. Esta opción nos da la flexibilidad de crear una red muy estructurada, con una planificación eficiente del cableado, los dispositivos y la gestión de los recursos.

¿Por qué es la mejor estrategia?

Después de analizar la viabilidad del proyecto, hemos llegado a la conclusión que crear una red desde cero es la mejor opción por varios motivos:

- **Eficiencia y rendimiento:** Una infraestructura nueva permite optimizar la velocidad y la capacidad de la red desde el principio, evitando problemas derivados de elementos antiguos u obsoletos.
- **Seguridad:** Se pueden implementar las últimas medidas de seguridad, evitando riesgos que puedan existir en sistemas antiguos.
- **Escalabilidad:** La red se diseñará pensando en el futuro, para que pueda adaptarse al crecimiento de la empresa sin necesidad de grandes reformas.
- **Optimización de costes a largo plazo:** A pesar de que la inversión inicial puede ser más alta, a la larga se reducirán los costes de mantenimiento y se evitarán problemas derivados de una infraestructura desactualizada.

Con esta estrategia, garantizamos que la empresa tendrá una red robusta, segura y preparada para el futuro.

1.4 Metodología de trabajo

La metodología seleccionada es PMI (Project Management Institute), un tipo de metodología estándar. La cual la hemos adaptado a nuestro proyecto de la siguiente manera:

- **Inicio:** Definición del proyecto y obtención de la aprobación por parte del cliente.
- **Planificación:** Establecimiento de objetivos, cronogramas, costes y riesgos.
- **Ejecución:** Desarrollo del proyecto según lo planificado.
- **Monitorización y control:** Supervisión de los avances y ajustes según sea necesario.

1.5 Estudio económico y presupuestario

1.5.1 Tareas y componentes del proyecto

Instalación/preparación de sistemas operativos:

- Ubuntu Desktop 24.04 (remota)
- Ubuntu Server 24.04
- Ubuntu Desktop 24.04
- Windows 10.

Configuración de red básica en las máquinas.

- Remota
 - enp1s0: PuigCastellar1
- Server
 - enp1s0: PuigCastellar1
 - enp2s0: Personal1
- UBDesk
 - enp1s0: Personal1
- W10
 - enp1s0: Personal1

Desarrollo de los scripts Bash automatizados para gestionar el servidor y el cliente.

Despliegue de servicios mediante Ansible configuración de:

- Server
 - Configuración de red
 - configuración de SSH
 - Servicio DHCP
 - Samba AD DC
- Cliente UB
 - Instalaciones:
- Cliente w10
 - Acceso a Samba ADDC

1.5.2 Costos de desenvolupament

Coste /Hora de desarrollador: 25 €

Recurso	Descripción	Coste estimado
Equipos físicos/virtuales	3 máquinas (Ubuntu Desktop, Server y Win10)	0 € (entorno ya disponible)
Sistemas operativos	Ubuntu (open source), Windows 10 (licencia existente)	0 €
Herramientas	Ansible, sshpass, software de red (open source)	0 €
Mano de obra	Desarrollo, pruebas, documentación	25 €/h x 150h = 3.750 €
	Total estimado	3.750 €

1.5.3 Costes de mantenimiento

Elemento	Frecuencia	Coste anual estimado
Actualizaciones del sistema (Ubuntu)	Periódicas	0 €
Actualizaciones de Windows	Automáticas (licencia activa)	0 €
Revisión de scripts/playbooks	Anual (adaptaciones o mejoras)	40 € (4 h x 10 €/h)
	Total anual estimado	40 €

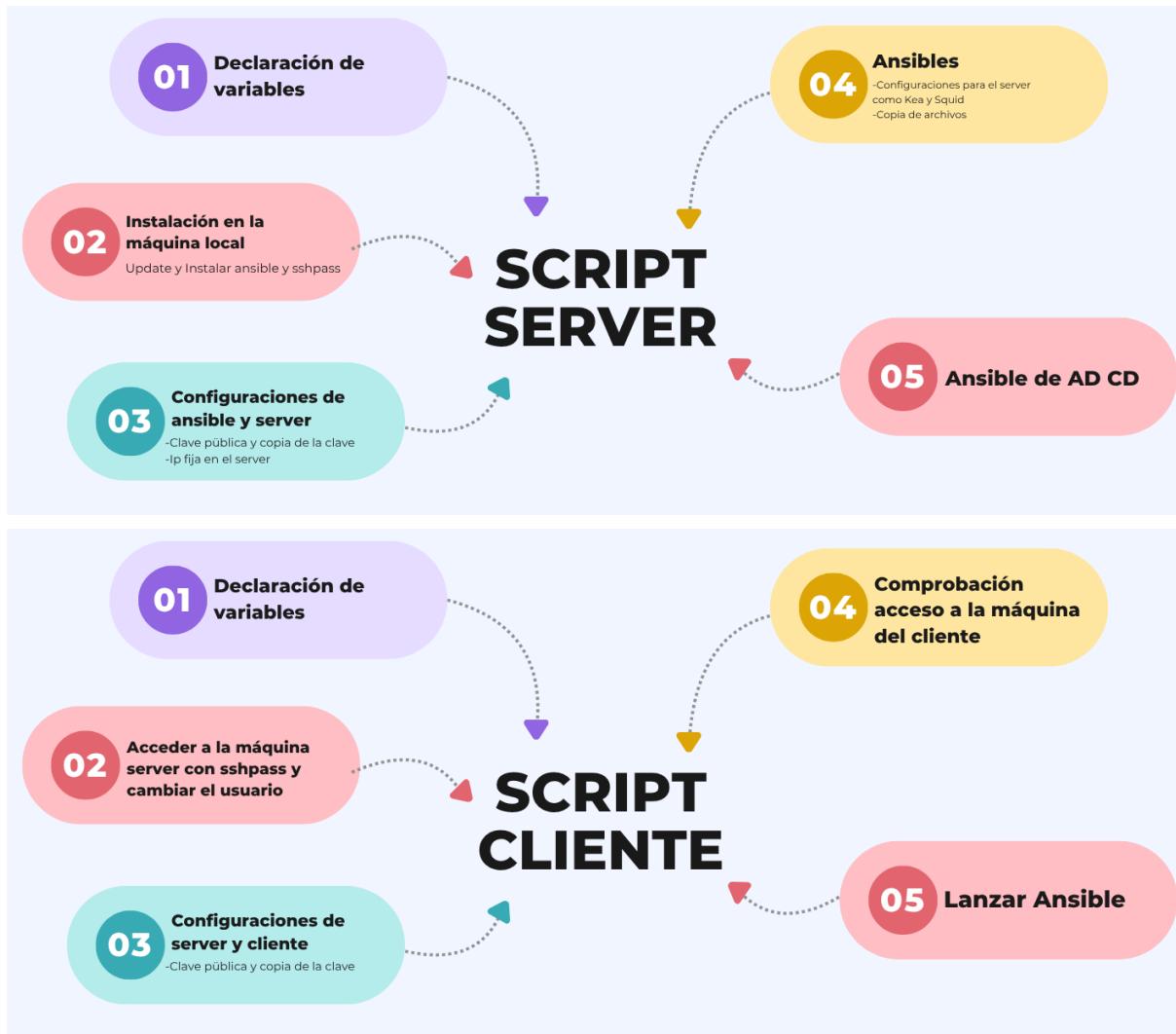
1.5.4 Oportunidades de beneficio

- Ahorro de tiempo en la configuración y despliegue de sistemas.
- Reducción de errores humanos mediante la automatización.
- Compatibilidad multiplataforma, demostrando que Samba permite integrar clientes Windows en un entorno Linux.
- Ampliación sencilla a más clientes o servicios.
- Base reutilizable para otros entornos educativos o empresariales.
- Formación técnica sólida en scripting, automatización y administración de dominios

1.5.5 Decisión sobre la continuidad del proyecto

Dado su bajo coste, su utilidad práctica y su versatilidad al incluir clientes Linux y Windows en un entorno automatizado, **la continuidad del proyecto se considera viable**. Además, su escalabilidad y adaptabilidad lo convierten en una gran solución para entornos educativos o pequeñas empresas que deseen automatizar su infraestructura TI sin grandes inversiones.

2 Descripción del proyecto



2.1 Anàlisi de requisits

Para lograr una automatización eficiente y fiable, se requiere que el sistema cumpla con una serie de funcionalidades específicas, así como condiciones técnicas que aseguren su rendimiento, escalabilidad y facilidad de uso. A continuación, se detallan los requisitos funcionales.

2.1.1 Requisitos funcionales

Automatizar la preparación del servidor:

El sistema debe ser capaz de configurar automáticamente un servidor Ubuntu Server 24.04, incluyendo red, hostname, acceso SSH y credenciales.

Automatizar la preparación de la máquina cliente:

El sistema debe aplicar configuraciones básicas (hostname, contraseña, SSH) a una máquina cliente conectada al servidor.

Ejecutar scripts desde una máquina remota:

Los scripts deben poder lanzarse desde una máquina Ubuntu Desktop 24.04 y realizar la configuración sin intervención directa del usuario.

Instalar y utilizar Ansible para la automatización:

El sistema debe instalar Ansible y utilizarlo para ejecutar playbooks que configuren servicios específicos en las máquinas.

Desplegar un controlador de dominio Samba AD DC:

El sistema debe instalar y configurar automáticamente un servidor Samba como controlador de dominio Active Directory.

Gestionar el acceso mediante claves SSH:

El sistema debe generar e intercambiar claves SSH para establecer conexiones seguras sin necesidad de introducir contraseñas.

Permitir una configuración reproducible:

Todo el proceso debe poder repetirse en otras máquinas con las mismas características, garantizando resultados coherentes y fiables.

2.1.2 Requisitos no funcionales

Usabilidad

El sistema debe ser sencillo de ejecutar desde la máquina remota, requiriendo mínima intervención manual durante la ejecución de los scripts.

Automatización completa

El proceso debe desarrollarse sin necesidad de intervención humana durante la ejecución, salvo para la confirmación inicial o errores críticos.

Seguridad

Las conexiones SSH deben ser seguras mediante el uso de claves públicas y privadas, evitando el uso de contraseñas en las conexiones remotas posteriores a la configuración inicial.

Portabilidad

Los scripts y playbooks deben ser compatibles con las versiones específicas de Ubuntu 24.04 Desktop y Server, facilitando su reutilización en entornos similares.

Mantenibilidad

La estructura del proyecto debe ser clara y modular, usando Ansible con roles bien definidos para facilitar futuras modificaciones o ampliaciones.

Robustez y fiabilidad

Los scripts deben incluir comprobaciones básicas para garantizar que los comandos se ejecutan correctamente y que el sistema objetivo responde adecuadamente, con manejo de errores simple.

Rendimiento

El tiempo de configuración debe ser razonable, minimizando esperas innecesarias para permitir un despliegue eficiente.

Documentación

El sistema debe incluir comentarios y documentación básica para facilitar su comprensión y uso por parte de otros usuarios o administradores.

2.2 Previsió de tasques d'investigació

Durante el desarrollo del proyecto, ha sido necesario realizar una serie de tareas de investigación para comprender en profundidad los componentes involucrados y asegurar una correcta automatización de los procesos. A continuación, se detallan las investigaciones realizadas:

- Búsqueda de alternativas para conectarnos vía remota reduciendo la menor intervención humana posible. Lo que nos llevó a **SshPass** que con un par de parámetros nos permite indicar la contraseña y evitar la pregunta de la fingerprint que nos realiza el ssh por primera vez.
- Dado que con el script no nos dejaba realizar los comandos remotamente encontramos el operador **EOF** el cual nos permite realizar bloques de comandos dentro de la misma sesión de ssh.
- También aprendimos el uso de las tuberías en más profundidad cómo utilizar `echo "root:usuario" | chpasswd`. Esto nos permite lanzar el comando `chpasswd` e indicar el usuario y contraseña.
- Aprendimos que el usuario root no viene con contraseña establecida por lo que hay que crearla si queremos conectarnos de manera remota.
- Para sustituir una línea de parámetro en el documento ssh utilizamos el siguiente comando.
`sed -i "s/^#\!?PermitRootLogin .*/PermitRootLogin yes/" /etc/ssh/sshd_config`

- **sed -i** modifica el archivo directamente sin necesidad de crear uno nuevo.
 - **s/** es la sintaxis de sustitución del comando sed: s/patrón/nuevo_valor
 - **^** indica que es al principio de la línea
 - **#\?** Descomenta si está comentado
- Investigación de samba AD DC para conectar una máquina windows al dominio.

2.3 Tecnologías

2.3.1 Comparativa de las tecnologías valoradas

Isard / VirtualBox

Para hacer pruebas en máquinas virtuales. Es fácil de usar, pero puede ralentizar el sistema si no tiene mucha potencia.

Visual Studio Code

Editor para escribir código rápido y con muchas funciones. Requiere algo de configuración inicial.

GitHub

Guarda y organiza los archivos para que varias personas trabajen juntas. Puede ser difícil al principio para quienes no lo conocen.

Google Drive

Para compartir y editar documentos en línea fácilmente. Necesita internet y tiene espacio limitado.

InfinityFree

Hosting gratuito para subir proyectos web. Fácil, pero no apto para proyectos grandes o con mucho tráfico.

2.3.2 Tecnologías escogidas

Para este proyecto hemos elegido varias tecnologías que nos han facilitado el desarrollo y la colaboración.

Usamos **Isard** para crear máquinas virtuales, lo que nos permitió hacer pruebas en un entorno seguro sin afectar nuestras computadoras principales.

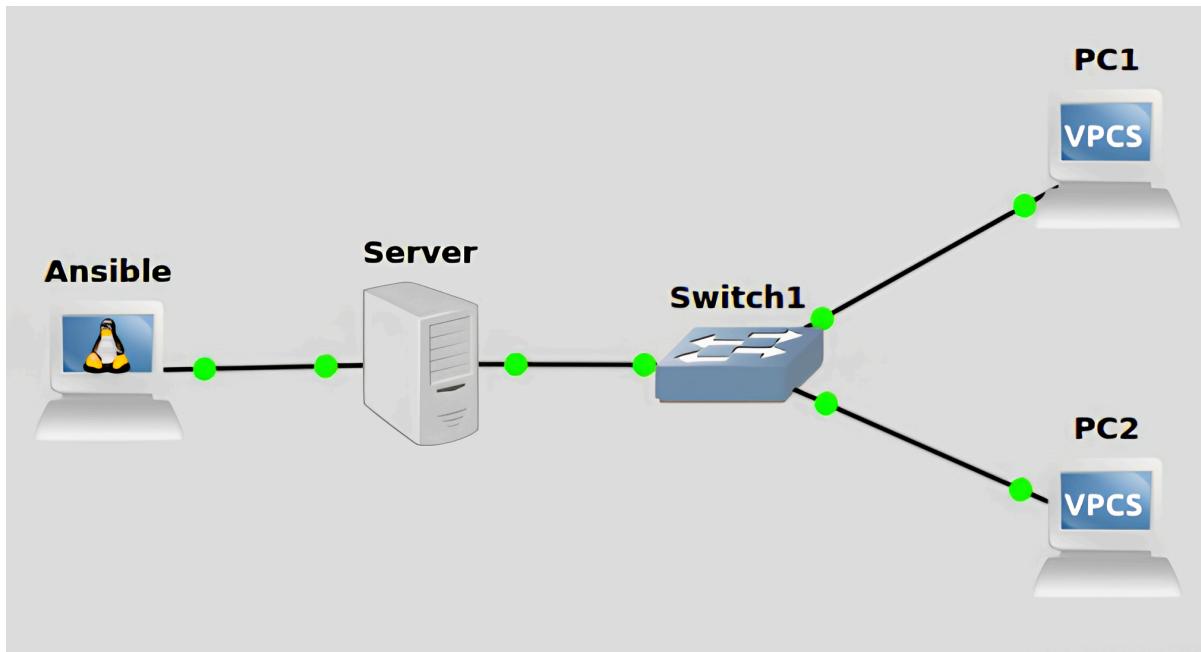
Para escribir y editar el código, utilizamos **Visual Studio Code**, ya que es un programa rápido y práctico con muchas herramientas útiles para programar.

GitHub fue nuestra plataforma para guardar y controlar las versiones de los archivos, lo que nos ayudó a trabajar en equipo sin perder ningún cambio.

También usamos **Google Drive** para compartir documentos y archivos, facilitando que todos pudiéramos acceder y colaborar desde cualquier lugar.

Por último, **InfinityFree** nos sirvió como hosting gratuito para alojar y mostrar los proyectos web de forma sencilla.

2.4 Estructura del proyecto



2.5 Descripción de los componentes:

2.5.1 Componente 1

Maquina remota - Ansible:

Es una herramienta de automatización que permite gestionar configuraciones, implementar aplicaciones y orquestar tareas complejas de forma sencilla, eficiente y sin necesidad de instalar agentes en los sistemas gestionados.

2.5.2 Componente 2

Servidor:

Será el que reciba la instalación y configuración para proporcionar, servicios, recursos, programas y/o datos a los otros dispositivos.

2.5.3 Componente 3

Switch (opcional):

Facilitar la transferencia de datos entre dispositivos conectados en una red local

2.5.4 Componente 4

PC Ubuntu:

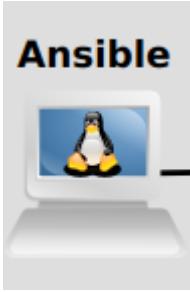
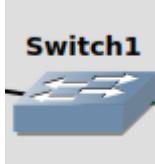
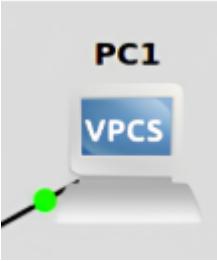
Recibirá las instalaciones y configuraciones.

2.5.4 Componente 5

PC Windows:

Recibirá las configuraciones y el acceso a AD DC

2.6 Definición de las tareas.

 Ansible	Configuracion de server y clientes
 Server	Servidor DHCP que da internet a los clientes y reparte IP's
 Switch1	Conecta el server con los clientes en una misma red local
 PC1 VPCS	Dispositivo que configuraremos para los clientes

2.7 Definición de las funcionalidades.

El proyecto contempla la automatización completa del proceso de configuración de una infraestructura básica de red compuesta por un servidor, una máquina cliente Linux y una máquina cliente Windows 10.

Las funcionalidades principales del sistema son las siguientes:

2.7.1. Configuración automática del servidor Ubuntu 24.04

- Actualización del sistema y repositorios.
- Instalación de paquetes esenciales (openssh-server, sshpass, ansible, etc.).
- Configuración de red estática, hostname y acceso SSH para el usuario root.
- Generación e instalación de claves SSH para acceso sin contraseña.
- Instalación y despliegue de un servidor Samba configurado como controlador de dominio (AD DC).

2.7.2. Automatización de tareas con Ansible

- Ejecución de playbooks desde la máquina Desktop hacia el servidor para automatizar su configuración.
- Ejecución de playbooks desde el servidor hacia un cliente Linux para replicar ciertas configuraciones básicas.
- Uso de inventarios y variables personalizadas para gestionar diferentes máquinas.

2.7.3. Configuración automática de un cliente Ubuntu

- Asignación de hostname personalizado.
- Habilitación del acceso SSH.
- Unión a la infraestructura automatizada del dominio.

2.7.4. Conexión de una máquina Windows 10 al dominio

- Configuración manual para la unión al dominio Samba.
- Validación de inicio de sesión con credenciales del dominio.
- Pruebas de conectividad y autenticación con el servidor.

3 Otros capítulos

Nuestro centro dispone de infraestructura virtual con **IsardVDI**, es una herramienta de virtualización libre bajo licencia **AGPL3** (GNU Affero General Public License). Es una variante de la Licencia Pública General de **GNU** (GPL), diseñada específicamente para el uso de software a través de una red, como en servicios web.

En **IsardVDI** el inconveniente más frecuente que nos hemos enfrentado ha sido que la red de isard no funciona correctamente con total normalidad

El **VirtualBox** es un software para virtualización, que se utiliza para virtualizar sistemas operativos dentro de nuestro ordenador anfitrión, creando lo que se conoce como máquina virtual. Es muy parecido al isard solo que en vez de tener una cuota de recursos, está limitado por los recursos del PC anfitrión ya que depende de ellos.

El **Github** es una plataforma basada en la nube donde puedes almacenar, compartir y trabajar junto con otros usuarios para escribir código.

El **ChatGPT** es un asistente virtual que comprende y genera texto en lenguaje humano, capaz de responder preguntas, redactar textos, traducir, programar, resumir información.

Utilizando la aplicación de **ChatGPT** el inconveniente más común al cual nos hemos enfrentado es que hay un cierto límite a la hora de utilizarlo gratuitamente y en el caso de no pagar tendrías que comenzar un nuevo chat el cual le hace perder la información que obtuviste anteriormente.

4 Comandos y soluciones

4.1 Comandos

Actualización de sistema y herramientas necesarias

```
apt update && apt upgrade -y  
apt install -y sshpass ansible
```

Gestión de contraseñas y acceso ssh

```
echo "root:contraseña" | chpasswd  
sed -i 's/PermitRootLogin prohibit-password/PermitRootLogin yes/' /etc/ssh/sshd_config  
systemctl restart ssh
```

Creación de claves de seguridad.

```
ssh-keygen -t rsa -b 4096 -N "" -f ~/.ssh/id_rsa  
ssh-copy-id root@IP_DEL_SERVIDOR
```

Configuración de red

```
nano /etc/netplan/01-netcfg.yaml  
netplan apply
```

Configuración de hostname

```
hostnamectl set-hostname Server  
hostnamectl set-hostname Cliente1
```

Comprobación de conectividad

```
ping -c 4 IP_DESTINO
```

Instalación y despliegue playbooks Ansible

Ej. Playbook	Ej. Inventory/hosts
<pre>- name: Install Samba packages apt: name: - samba - samba-ad-dc state: present</pre>	<pre>[Server] Server1 ansible_host=root@192.168.237.2 ansible_user=root</pre>

Ejecución de Ansible

```
ansible-playbook playbook_samba.yml
```

Ejecución de script

```
bash script_configuracion.sh
```

4.2 Solución

SCRIPT1: configurar_server.sh

```
#!/bin/bash
# Script de configuración desde la máquina local hacia la máquina Server
# Cargar variables
source ./vars.sh
start=$(date +%s)
SVIP0=$1 # IP actual del servidor (se pasa como argumento al ejecutar el script)

# -----
# Parte local
# -----

echo "[+] Actualizando repositorios localmente..."
echo "$PASSsv1" | sudo -S apt update

echo "[+] Instalando sshpass y ansible..."
echo "$PASSsv1" | sudo -S apt install -y sshpass ansible

# -----
# Parte remota (Server)
# -----

echo "[+] Generando configuración de red..."
NETPLAN_CONFIG=$(cat <<EOF_NETPLAN
network:
  version: 2
  ethernets:
    enp1s0:
      dhcp4: false
      addresses:
        - ${SVIP1}/22
      routes:
        - to: default
          via: 192.168.236.1
      nameservers:
        addresses: [8.8.8.8, 1.1.1.1]
    enp2s0:
      dhcp4: false
      addresses:
        - ${SVIP2}/24
EOF_NETPLAN
)

echo "[+] Conectando y configurando la máquina Server ($SVIP0)...

sshpass -p "$PASSsv1" ssh -o StrictHostKeyChecking=no "$USERsv1@$SVIP0" bash <<
EOF
echo "$PASSsv1" | sudo -S bash -c '
  echo "[+] Cambiando contraseña de root..."
  echo "root:$PASSsv2" | chpasswd
```

```
echo "[+] Cambiando hostname a Server..."  
hostnamectl set-hostname Server  
  
echo "[+] Habilitando acceso root por SSH..."  
sed -i "s/#\\?PermitRootLogin .*/PermitRootLogin yes/" /etc/ssh/sshd_config  
systemctl restart ssh  
  
if [ ! -f /etc/netplan/50-cloud-init.yaml.bkup ]; then  
    echo "[+] Haciendo copia de seguridad de Netplan..."  
    cp /etc/netplan/50-cloud-init.yaml /etc/netplan/50-cloud-init.yaml.bkup  
else  
    echo "[i] Backup de Netplan ya existe, no se sobrescribe."  
fi  
  
echo "[+] Reescribiendo Netplan..."  
cat > /etc/netplan/50-cloud-init.yaml << 'EONET'  
$NETPLAN_CONFIG  
EONET  
  
echo "[+] Aplicando nueva configuración de red en segundo plano..."  
nohup bash -c "sleep 2 && netplan apply" > /dev/null 2>&1 &  
  
echo "[✓] Configuración aplicada. Cerrando sesión."  
exit  
'  
EOF  
  
# -----  
# Postconfiguración  
# -----  
  
echo "[✓] Script ejecutado correctamente. La máquina Server ya está configurada."  
  
# Clave pública SSH  
if [[ -f ~/.ssh/id_rsa.pub ]]; then  
    read -p "[?] Ya existe una clave SSH en ~/.ssh/id_rsa.pub. ¿Deseas sobrescribirla? (s/n): " RESP  
    if [[ "$RESP" == "s" || "$RESP" == "S" ]]; then  
        echo "[+] Eliminando clave SSH antigua..."  
        rm -f ~/.ssh/id_rsa ~/.ssh/id_rsa.pub  
        echo "[+] Generando nueva clave SSH..."  
        ssh-keygen -t rsa -b 4096 -f ~/.ssh/id_rsa -N ""  
    else  
        echo "[i] Se usará la clave SSH existente."  
    fi  
else  
    echo "[+] Generando clave SSH..."  
    ssh-keygen -t rsa -b 4096 -f ~/.ssh/id_rsa -N ""  
fi  
  
echo "[...] Esperando a que la máquina Server esté disponible en $SVIP1..."
```

```
for i in {1..10}; do
    ping -c 1 "$SVIP1" > /dev/null 2>&1 && break
    echo "Esperando... ($SVIP1)"
    sleep 3
done

echo "[+] Copiando clave SSH al root del servidor..."
sshpass -p "$PASSsv2" ssh-copy-id root@"$SVIP1"

for PLAYBOOK in "${PLAYBOOK[@]}"; do
    echo "Ejecutando $PLAYBOOK..."
    ansible-playbook -i "$INVENTORY" "$DIR2/$PLAYBOOK"

done

echo "Creando estructura de carpetas Ansible para $PROYECTO..."

# Crear estructura básica
mkdir -p $PROYECTO/{inventory,roles/samba_ad_dc/{tasks,vars,files}}


# Archivo de inventario
cat > $PROYECTO/inventory/hosts <<EOF
[server]
192.168.237.2 ansible_user=root ansible_ssh_pass=melon ansible_become=true
EOF

# Playbook principal
cat > $PROYECTO/playbook.yml <<EOF
---
- name: Desplegar servidor Samba AD DC
  hosts: server
  become: yes
  roles:
    - samba_ad_dc
EOF

# Variables del rol
cat > $PROYECTO/roles/samba_ad_dc/vars/main.yml <<EOF
hostname: dc
ip_address: 192.168.1.2
dns_forwarder: 8.8.8.8
fqdn: dc.hjm.local
domain_name: hjm.local
realm: HJM.LOCAL
domain: hjm
net_prefix: 192.168.1.0/24
admin_password: usuario1234*
EOF

# Tareas del rol
cat > $PROYECTO/roles/samba_ad_dc/tasks/main.yml <<'EOF'
---
- name: Establecer hostname
```

```
ansible.builtin.hostname:
  name: "{{ hostname }}"

- name: Añadir FQDN a /etc/hosts
  ansible.builtin.lineinfile:
    path: /etc/hosts
    line: "{{ ip_address }} {{ fqdn }} {{ hostname }}"
    create: yes

- name: Desactivar y detener systemd-resolved
  ansible.builtin.systemd:
    name: systemd-resolved
    enabled: no
    state: stopped

- name: Crear /etc/resolv.conf
  ansible.builtin.copy:
    dest: /etc/resolv.conf
    content: |
      nameserver {{ ip_address }}
      nameserver {{ dns_forwarder }}
      search {{ domain_name }}

- name: Verificar si resolv.conf es immutable
  ansible.builtin.shell: lsattr /etc/resolv.conf | grep '\-i\|'
  register: resolv_conf_attr
  changed_when: false
  failed_when: false

- name: Establecer el atributo immutable si no está
  ansible.builtin.shell: chattr +i /etc/resolv.conf
  when: resolv_conf_attr.rc != 0

- name: Instalar paquetes necesarios
  ansible.builtin.apt:
    name:
      - acl
      - attr
      - samba
      - samba-dsdb-modules
      - samba-vfs-modules
      - smbclient
      - winbind
      - libpam-winbind
      - libnss-winbind
      - libpam-krb5
      - krb5-config
      - krb5-user
      - dnsutils
      - chrony
      - net-tools
    state: present
    update_cache: yes
```

```
- name: Deshabilitar servicios innecesarios
ansible.builtin.systemd:
  name: "{{ item }}"
  enabled: no
  state: stopped
loop:
- smbd
- nmbd
- winbind

- name: Habilitar samba-ad-dc
ansible.builtin.systemd:
  name: samba-ad-dc
  enabled: yes
  masked: no

- name: Backup smb.conf si existe
ansible.builtin.command: mv /etc/samba/smb.conf /etc/samba/smb.conf.orig
args:
  removes: /etc/samba/smb.conf

- name: Provisionar dominio Samba
ansible.builtin.command: >
  samba-tool domain provision
  --realm={{ realm }}
  --domain={{ domain }}
  --server-role=dc
  --dns-backend=SAMBA_INTERNAL
  --adminpass='{{ admin_password }}'
register: provision_result
changed_when: "Administrator password" in provision_result.stdout

- name: Sustituir krb5.conf
ansible.builtin.copy:
  remote_src: yes
  src: /var/lib/samba/private/krb5.conf
  dest: /etc/krb5.conf
  force: yes

- name: Iniciar servicio samba-ad-dc
ansible.builtin.systemd:
  name: samba-ad-dc
  state: started

- name: Crear usuario hjmer en el dominio
ansible.builtin.command: >
  samba-tool user create hjmer usuario123*
register: create_user_result
changed_when: "Created user" in create_user_result.stdout

- name: Establecer permisos en ntp_signd
ansible.builtin.file:
```

```
path: /var/lib/samba/ntp_signd/
owner: root
group: _chrony
mode: '0750'

- name: Configurar chrony
  ansible.builtin.blockinfile:
    path: /etc/chrony/chrony.conf
    block: |
      bindcmdaddress {{ ip_address }}
      allow {{ net_prefix }}
      ntpsigndsocket /var/lib/samba/ntp_signd

- name: Reiniciar y habilitar chronyd
  ansible.builtin.systemd:
    name: chronyd
    enabled: yes
    state: restarted

EOF

echo "[✓] Estructura del proyecto creada correctamente en ./PROYECTO"
echo " Ejecutando ansible..."
sleep 2
cd PROYECTO/
ansible-playbook -i inventory/hosts playbook.yml

echo "Ansible terminado"
end=$(date +%s)
runtime=$((end - start))

echo "Tiempo de ejecución: $runtime segundos"
```

SCRIPT 2: configurar_cliente.sh

```
#!/bin/bash
# Cargar variables
source ./vars.sh

# Comprobar que todas las variables necesarias están definidas
if [[ -z "$SVIP1" || -z "$PASSsv1" || -z "$USERsv1" || -z "$PCIP3" || -z "$PASSsv2" || -z
"$directorio_ansible" ]]; then
    echo "[!] Faltan variables necesarias en vars.sh"
    exit 1
fi

echo "[+] Conectando con el servidor intermedio ($SVIP1)..."
sshpass -p "$PASSsv2" ssh -o StrictHostKeyChecking=no "root@$SVIP1" bash << EOF1
echo "[+] Instalando sshpass si es necesario..."
apt-get update && apt-get install -y sshpass

echo "[+] Conectando con el cliente ($PCIP3)..."
sshpass -p "$PASSsv1" ssh -o StrictHostKeyChecking=no "$USERsv1@$PCIP3" bash
<< EOF2
echo "[+] Elevando privilegios para tareas administrativas..."
echo "$PASSsv1" | sudo -S bash -c '
    echo "[+] Cambiando contraseña del root..."
    echo "root:$PASSsv2" | chpasswd

    echo "[+] Estableciendo hostname a Cliente1..."
    hostnamectl set-hostname Cliente1

    echo "[+] Habilitando acceso SSH para root..."
    sed -i "s/^#\!.*PermitRootLogin .*/PermitRootLogin yes/" /etc/ssh/sshd_config
    systemctl restart ssh
'

EOF2

echo "[+] Comprobando clave SSH local en el servidor..."
if [[ ! -f ~/.ssh/id_rsa.pub ]]; then
    echo "[+] Generando nueva clave SSH..."
    ssh-keygen -t rsa -b 4096 -f ~/.ssh/id_rsa -N ""
else
    echo "[i] Clave SSH ya existe. Usando la existente."
fi

echo "[+] Esperando disponibilidad del cliente ($PCIP3)..."
for i in {1..10}; do
    ping -c 1 "$PCIP3" > /dev/null 2>&1 && break
    echo "Esperando... ($PCIP3)"
    sleep 3
done

echo "[+] Copiando clave SSH al cliente (root@$PCIP3)..."
sshpass -p "$PASSsv2" ssh-copy-id -o StrictHostKeyChecking=no root@"$PCIP3"
```

```
echo "[+] Ejecutando playbook Ansible en el servidor..."  
su - usuario -c "  
if [ -d '$directorio_ansible' ]; then  
    cd '$directorio_ansible'  
    ansible-playbook -i hosts playbook.yml  
else  
    echo '[!] El directorio Ansible no se encontró: $directorio_ansible'  
    exit 1  
fi  
"  
EOF1  
  
echo "[✓] Script terminado correctamente"
```

Archivo variables.

```
#!/bin/bash  
# Variables de configuración para el script principal  
  
USERsv1="usuario"  
PASSsv1="usuario"  
PASSsv2="melon"  
SVIP1="192.168.237.2" # IP fija para enp1s0  
SVIP2="192.168.1.2" # IP fija para enp2s0  
PCIP3="192.168.1.3"  
DIR1="/home/usuario/ansibles/hosts"  
DIR2="/home/usuario/ansibles/playbooks"  
INVENTORY="$DIR1/host_A-S"  
PROYECTO="ADDC-HJM"  
directorio_ansible="/home/usuario/ansible_cliente" # La ruta donde se encuentra  
ansible_cliente  
PLAYBOOK=(  
    "Playbook_instalaciones.yml"  
    "Playbook_KEA.yml"  
    "Playbook_squid.yml"  
    "Playbooks_cliente.yml"  
)
```

4.4 Visión de futuro

Este proyecto es una primera fase, pero con gran potencial de crecimiento, manteniendo la filosofía de automatización, eficiencia y facilidad de mantenimiento.

Nos gustaría mejorar el script añadiendo algunas funcionalidades:

Ampliación del dominio Samba AD DC

Con la integración de más máquinas cliente, tanto Linux como Windows, simplificando la administración de usuarios y recursos compartidos.

Incorporación de servicios adicionales

Cómo un servidor DNS.

Grafana para la monitorización y gestión del estado de la red y sus servicios.

Mejora de la seguridad

Integración de firewall, UFW o iptables.

Copias de seguridad automatizada con crontab y rsync u otro programa/servicio.

Interfaz gráfica

Nos gustaría incluir una interfaz gráfica para poder modificar y/o personalizar las variables del script de manera cómoda dinámica de esta manera usuarios no técnicos podrán configurarlo de manera sencilla.

5. Glossario

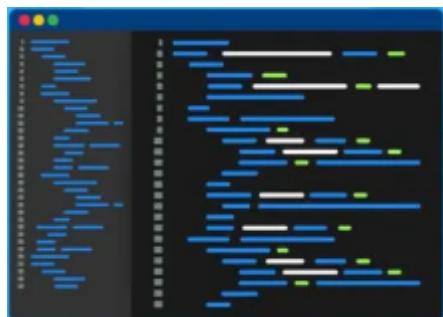
-Servidor



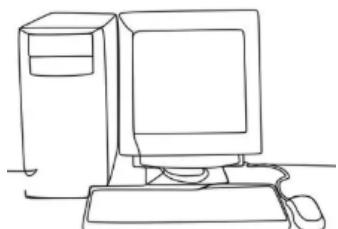
-Ansible



-Script



-Cliente



-AD DC



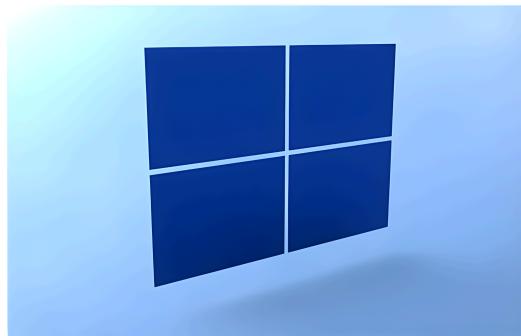
-DHCP



-Redes



-Windows



6. Bibliografía

W3schools:

Es una página web pensada para ayudarte a aprender a crear sitios web, paso a paso y de forma sencilla. Está llena de ejemplos claros y ejercicios para que puedas practicar mientras aprendes. No necesitas saber nada previo: te guía desde lo más básico en lenguajes de programación. Es gratis, fácil de usar y perfecto si tienes curiosidad por el mundo del desarrollo web.

<https://www.w3schools.com/html/default.asp>



InfinityFree:

InfinityFree es un proveedor de alojamiento web gratuito que permite a los usuarios crear y mantener sitios web sin costo alguno.

<https://www.infinityfree.com/> // web creada: <https://hjm.rf.gd/?i=1>



Instalación de Samba AD DC de clockwork:

Lo utilizamos para llevar a cabo la instalación de Samba y fue de gran ayuda ya que está muy bien redactado y explicado.

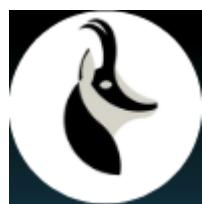
<https://drive.google.com/file/d/1hvDscD-TzPgHwvkA8eQNYsrPSMOWi9L6/view>



El IsardVDI:

Lo utilizamos para hacer máquinas virtuales las cuales llevarían nuestro proyecto, aunque estuvimos limitados a la hora de abrir máquinas al mismo tiempo ya que si abrías más de 3 colapsaron las máquinas.

<https://elmeuescriptori.gestioeducativa.gencat.cat/desktops>



EL Moodle:

Lo utilizamos para sacar información desde las prácticas que hemos realizado a lo largo del curso o de los pdf de teorías.

<https://moodle.elpuig.xeill.net/course/view.php?id=581>



7. Conclusión

Instalación de Samba AD DC de clockwork:

Nuestra conclusión es que a pesar de que iniciamos con una idea diferente, de instalaciones de cableado y diferentes dispositivos informáticos, pero dado a los breves plazos de entrega de los que disponíamos cambiamos a la idea ya que nos dimos cuenta que podríamos automatizar casi todo mediante un script, el cual nos ayudó mucho para profundizar el tema de Ansible y aprender más sobre ello.

Este proyecto ha sido todo un reto para nosotros, ha sido una experiencia de aprendizaje real, en la que hemos podido aplicar todo lo que hemos aprendido durante el ciclo para crear una red informática, segura y funcional desde cero.

Empezamos con una idea clara: diseñar una infraestructura que no solo funcionara, sino que estuviera preparada para crecer junto con la empresa. A lo largo del camino, nos enfrentamos con retos técnicos, tuvimos que tomar decisiones importantes sobre qué tecnologías usar y aprendimos a trabajar en equipo para superar los obstáculos. Herramientas como Ansible nos ayudaron a automatizar procesos y ahorrar mucho tiempo, se convirtió en una parte clave del proyecto.

Aunque no todo salió perfecto, supimos adaptarnos ante la adversidad de los diferentes imprevistos que surgían además, también aprendimos que ser flexibles y saber reaccionar ante los imprevistos es igual de importante.

Nos vamos con la satisfacción de haber construido una red completa y funcional, pero sobre todo, con el orgullo de haberlo hecho nosotros mismos. Este proyecto nos ha preparado para el mundo real, y deja la puerta abierta para seguir mejorando en el futuro, con nuevas funciones y más personalización.