ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

IMPLEMENTACION DE ANSIBLE PARA LA CONFIGURACION DE EQUIPOS DE NETWORKING

IMPLEMENTAR ANSIBLE PARA REALIZAR ENRUTAMIENTO ENTRE VLANs

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE TECNÓLOGO SUPERIOR EN REDES Y TELECOMUNICACIONES

DONOVAN STEVE VACA MUÑOZ

DIRECTOR: ING. GABRIELA KATHERINE CEVALLOS SALAZAR, MSC.

DMQ, agosto 2022

CERTIFICACIONES

Yo, Donovan Steve Vaca Muñoz declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

Donovan Steve Vaca Muñoz

donovan.vaca@epn.edu.ec

donovan_uio@hotmail.com

Certifico que el presente trabajo de integración curricular fue desarrollado por Donovan Steve Vaca Muñoz, bajo mi supervisión.

Ing. Gabriela Katherine Cevallos Salazar
DIRECTOR

gabriela.cevalloss@epn.edu.ec

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

Donovan Vaca

DEDICATORIA

Mi proyecto de titulación va dirigido a mis padres, Franklin Vaca y Angélica Muñoz, que han dedicado cada minuto de su tiempo a guiarme, cuidarme y enseñarme todas sus experiencias. Adicionalmente, agradecerles por cada consejo que me han dado para seguir adelante y no quedarme estancado en una cosa.

A todos mis profesores por brindarme todos sus conocimientos con los que me fui forjando y llegar hasta lo que hoy soy.

A todos los ingenieros de la Escuela de Formación de Tecnólogos (ESFOT), de la Escuela Politécnica Nacional (EPN) por sus consejos que me han servido de guía.

A mis primos, Bryan y Heinert Abad, por enseñarme todo cuanto sé y por enseñarme que con esfuerzo que todo se puede lograr.

Donovan

AGRADECIMIENTO

Primero quiero agradecer a Dios por protegerme siempre y brindarme su fuerza para cumplir mis metas. Adicionalmente, quiero agradecer a la Ingeniera Gabriela Cevallos, mi tutora, quien me ha apoyado en el transcurso de mi proyecto de integración. También quiero agradecer a mis padres, Franklin y Angélica por todo ese apoyo moral, cariño que me brindan a diario y estar conmigo en cada decisión.

Agradezco de igual manera a la Escuela de Formación de Tecnólogos (ESFOT) por darme los conocimientos necesarios a lo largo de mi carrera para llegar a ser un gran profesional.

A mis demás familiares por todo el gran apoyo que me brindaron en mi jornada como estudiante y por haber estado cuento más los necesitaba.

Donovan

ÍNDICE DE CONTENIDOS

RESUMEN

El proyecto en cuestión busca implementar la herramienta de Ansible para la configuración automatizada de equipos de *networking* con el fin de obtener una topología donde se ejecute el enrutamiento entre VLANs. Para ello se usarán los *playbooks* que, mediante una conexión SSH, permitirán configurar interfaces de red, crear VLANs y establecer puertos designados a cada VLAN para la comunicación entre máquinas.

Para la primera parte, se describe al proyecto partiendo de la herramienta Ansible, sus características principales y los requerimientos necesarios para su correcta ejecución, de esta manera se podrá trabajar con Ansible sin ningún problema. Además, se aborda a los simuladores de red realizando una comparativa entre tres simuladores más representativos.

En la siguiente parte se propone la metodología usada para la ejecución del proyecto, describiendo las instrucciones más relevantes alcanzando un buen procedimiento para cada uno de los objetivos planteados.

La tercera parte, describe los pasos de implementación, donde se configura el nodo controlador el cual mediante el protocolo SSH se logra comunicar con los equipos de *networking*. Se configurarán los *playbooks* gracias a la aplicación Ansible instalada, mismos que son desplegados en los nodos administrados. Dentro de esta parte se presenta las pruebas de verificación, observando que se ha realizado la configuración de manera automática y se tiene el enrutamiento entre VLANs.

Para finalizar, en las últimas secciones, la cuarta y la quinta, contienen, respectivamente, las conclusiones obtenidas en el transcurso del proyecto y las recomendaciones para llevar a cabo la misma

PALABRAS CLAVE: Ansible, Protocolo SSH, *Playbooks*, Equipos de *networking*.

ABSTRACT

The project in question seeks to implement the Ansible tool for the automated configuration of networking equipment in order to obtain a topology where routing between VLANs is executed. For this, the playbooks will be used that, through an SSH connection, will allow the teams to raise ports, create VLANs and establish designated ports for communication between machines.

For the first part, the project is described based on the Ansible tool, its main characteristics and the necessary requirements for its correct execution, in this way it will be possible to work with Ansible without any problem. In addition, network simulators are approached by making a comparison between three most representative simulators.

In the next part, the methodology used for the execution of the project is proposed, describing the most relevant instructions, reaching a good procedure for each of the proposed objectives.

The third part describes the implementation steps, where the controller node is configured, which through the SSH protocol is able to communicate with the network equipment. The playbooks will be configured thanks to the installed Ansible application, which are deployed in the managed nodes. Within this part, the verification tests are presented, observing that the configuration has been carried out automatically and that there is routing between VLANs.

Finally, in the last sections, the fourth and fifth, contain, respectively, the conclusions obtained in the course of the project and the recommendations to carry it out.

KEYWORDS: *Ansible, SSH Protocol, Playbooks*, *Networking Equipment.*

1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO

Con el avance tecnológico, estas requieren que se les presta la total atención ya sea para configuraciones o para uso diario. El ser humano se vio en la necesidad de intervenir para que estas tecnologías de *networking* avancen sin problema alguno y sacarles el mayor provecho posible ofreciendo conectividad dentro de sus áreas de trabajo. Sin embargo, muchos de estas configuraciones suelen ser repetitivas para el ser humano que se demora por realizar las mismas implementaciones en cada uno de los equipos de *networking*.

El presente proyecto pretende desplegar una topología de red, automatizando la configuración de los equipos de *networking* mediante el uso de la herramienta de DevOps Ansible. Ansible permite automatizar tareas de administración de los dispositivos de red de manera remota; garantizando rapidez, eficiencia, y fiabilidad; con menos esfuerzo y riesgo de errores humanos.

1.1 Objetivo general

Implementar Ansible para la configuración de equipos de *networking*.

1.2 Objetivos específicos

- Instalar la herramienta de Ansible en el nodo controlador
- Montar los diferentes dispositivos de la red en un simulador de redes
- Implementar las configuraciones mediante la herramienta Ansible
- Realizar pruebas de funcionamiento y verificación de los resultados obtenidos

1.3 Alcance

El presente proyecto conlleva la implementación de una topología de red donde los dispositivos logren tener conectividad configurándolos de forma automatizada mediante la herramienta de Ansible. Para esto se debe establecer una comunicación SSH entre el nodo controlador, máquina principal donde se instala Ansible, y los diferentes dispositivos de red. Con esto se despliega de manera remota las configuraciones necesarias para la implementación de la topología de red establecida.

1.4 Marco Teórico

Herramienta de DevOps Ansible

Ansible es una aplicación con la cual se puede motorizar tareas ya sea para actualizaciones, despliegues y/o administrar diversos servidores [1]. Adicionalmente, con este programa se puede gestionar configuraciones, organizar sistemas, invirtiendo y optimizando mejor el tiempo de quien realice dichas configuraciones.

Además, los equipos que trabajan en una misma topología pueden ejecutar menos tareas repetitivas al momento de requerir un cambio gracias a que se puede establecer comandos en un fichero propio del nodo controlador Ansible. De esta manera Ansible permite realizar de forma más rápida las tareas cotidianas que requieren los dispositivos [2] [3].

Ansible cuenta con dos nodos, un nodo controlador y un nodo administrado; el nodo controlador es una máquina o dispositivo basado en Linux el cual permite la configuración e instalación de Ansible. Por otro lado, se tiene al nodo administrado o remoto, el cual hace referencia a los dispositivos que estén administrados por el nodo maestro o controlador. Para que el nodo controlador y administrado puedan operar correctamente utilizan el protocolo SSH para la comunicación entre ellos [3].

Las ventajas más destacables de Ansible es que su instalación es sencilla, es desarrollado en Phyton; para elaborar las tareas usa el lenguaje YAML para realizar los *playbooks*. Para que Ansible pueda trabajar apropiadamente, puede ser instalado en los sistemas operativos siguientes: Red Hat Enterprise Linux versión 6 de 64-bits o la versión 7 de 64-bits, Ubuntu 12.04 LTS o 14.04 LTS de 64 bits y las versiones más resientes de Ubuntu Server. Adicionalmente, Ansible requiere de 2 (GB) de RAM como mínimo, aunque es recomendable que sea más de 4 (GB), 20 (GB) de disco duro y equipos que soporten versiones de 64 bits [1] [4].

Para la ejecución del presente proyecto se instalará Ansible sobre el sistema operativo Ubuntu Server 20.04; ya que no se presentaron problemas de arranque o demoras en cuento a su instalación y configuración, considerando una memoria de 4 (GB) de RAM y tamaño de disco duro de 10 (GB) para su correcta ejecución. Adicionalmente, en esta versión de Ubuntu server se ha podido instalar la versión *core* de Ansible, una versión la cual permite trabajar de manera más óptima y configurar correctamente los *playbooks.*

Software de simulador de redes

A continuación, se presentará una comparación entre distintos simuladores de red para la ejecución del proyecto presente. Para ello se caracterizó a GNS3, Eve-NG y Packet Tracer describiendo sus especificaciones.

- **GNS3**

Es un programa el cual es empleado para la emulación, configuración, comprobación y soluciones en cuanto a problema de redes ya sean virtuales o reales. Adicionalmente, GNS3 trae consigo ciertos dispositivos por defecto, aunque se puede descargar más de estos mediante imágenes desde la propia aplicación.

Dentro de los requisitos que necesita este programa es una versión reciente del sistema operativo, por ejemplo, una versión de Windows 7 o superior. Además, de un procesador de 4 núcleos, que la máquina cuente con virtualización, una memoria de 16 (GB) de RAM, espacio mínimo de 35 (GB) en disco duro.

Los inconvenientes que ciertas veces se presentan para correr los dispositivos se pueden evitar gracias a que GNS3 cuenta con una máquina virtual en VirtualBox o VMware, la cual permite arrancar la mayor parte de los recursos de los dispositivos que se implementen en GNS3, evitando la sobrecarga de la máquina física que lo contiene [5] [6].

- **Eve-NG**

Al igual que GNS3, Eve-NG es un *software* que permite simular redes mediante un laboratorio donde se emulan los dispositivos de *networking*. Este programa necesita de una máquina virtual la cual está conectada a un servidor *Web* propio de Eve-NG; además, de configuraciones como el implementar una dirección de red estática y cambiar la interfaz de red a NAT que servirá para conectarse al servidor *Web* por SSH [7] [8] [9].

- **Packet Tracer**

Fue desarrollado por CISCO, se pueden realizar simulaciones de redes con usuarios que pueden probar el funcionamiento de internet de las cosas (IoT), ciberseguridad y las redes como tal. Con este programa se puede simular las configuraciones de equipos de *networking* para que los usuarios se familiaricen con el entorno mediante una interfaz de comandos que está simulada desde la propia aplicación. Dentro de los requisitos de la aplicación se tiene en consideración versiones más actuales de sistemas operativos

como lo son Windows, Linux o Mac con 4 (GB) de memora RAM y 1,4 GB de almacenamiento [10] [11] [12].

Para la simulacion y puesta en marcha de la topóplogia de red de este proyecto se ejecutará en el simulador GNS3, ya que con este programa se puede instalar una máquina virtual propia del programa para arrancar los equipos de *networking* y evitar usar todos los recursos que una máquina física pudiese necesitar; permite subir las imágenes de Cisco y una máquina virtual controlador para la realización de los *playbooks*. GNS3 también es un programa muy conocido en el área académica como laboral teniendo una familizarización más estrecha con el mismo.

2 METODOLOGÍA

Para llevar a cabo este proyecto, se hizo uso de la metodología exploratoria para investigar a fondo Ansible. Se analizó las características, ventajas y desventajas, así mismo como su aporte al usarlo dentro de una red. Adicionalmente, se hizo uso de la metodología aplicada por el uso de la herramienta Ansible para desplegar el levantamiento de VLANs en los equipos de *networking* automáticamente. Además, Ansible facilita las tareas complejas y repetitivas erradicando gran parte de los errores causados por el hombre permitiendo el ahorro de tiempo y la automatización de tareas.

A fin de cumplir con el objetivo general, como primera parte se instaló una máquina virtual que cumpla con los requisitos necesarios para la instalación de *Open* SSH y de la herramienta de Ansible, para conformar el nodo controlador.

Además, se agregará a dicha máquina virtual una dirección IP estática la cual será de ayuda para la conexión por medio de SSH con los demás nodos administrados, es decir, los equipos de *networking*.

Mediante el *software* de simulación de red se procederá a colocar los equipos de *networking*, dispositivos finales y el nodo controlador. La topología cuenta con tres *switches* con diferentes VLANs, además de un *router* el cual permitirá realizar el enrutamiento entre VLANs mediante *router-on-a-stick.*

Se establecerá la comunicación entre el nodo controlador y los dispositivos de *networking* mediante SSH. Para ello se generaron las llaves de SSH con las que se obtendrá una conexión remota entre estos equipos con el nodo controlador, así mismo la configuración de los mismos para habilitar la conexión SSH entre los dispositivos.

Se creará el *playbook* del *router* central que contará con subinterfaces para no exceder el uso de las interfaces físicas. También, se creará un *playbook* para el *switch* central el cual tiene puertos troncales para el paso del tráfico de varias VLANs. Finalmente, se contará con tres *playbooks* enfocados a los últimos equipos de *networking*, es decir a los *switches*, que asignará un puerto troncal para la transmisión de las VLANs y los demás puertos se designarán como acceso para una VLAN en específico.

Finalmente, se realizarán las pruebas de funcionamiento ejecutando cada *playbook* en el nodo administrado correspondiente, comprobando que los cambios se hayan realizado exitosamente en el CLI y verificando que los equipos de *networking* se hayan configurado exitosamente mediante Ansible. Además, se verificará el enrutamiento entre VLANs comprobando que cada máquina pueda comunicarse con otra máquina.

3 RESULTADOS

Dentro de este apartado se llevará a cabo el procedimiento de implementación de *playbooks* mediante el uso de Ansible. Primero se crea la máquina virtual en la cual se instalará la herramienta Ansible con *Open* SSH, y otras características que ayudarán a la creación y despliegue de *playbooks*. Posteriormente, se introduce los equipos de red como son *routers* y *switches* que es lo que conforma la topología de red propuesta. Además, se establecerá conexión entre el nodo controlador y los nodos controlados por conexión SSH. Se elabora y presenta la ejecución de los *playbooks* en los equipos de *networking* y la respectiva verificación de la comunicación entre dispositivos de diferentes VLANs.

3.1 Características e instalación de Ansible

Ansible se la instaló en una máquina virtual con el sistema operativo Ubuntu server 20.04 como muestra la Figura 3.1. Previa a la instalación de Ansible se debe primero actualizar los repositorios de la máquina virtual, se utilizó el hipervisor VirtualBox. Para ello se hace uso de los comandos *sudo apt update* para buscar paquetes y se actualiza con el comando *sudo apt upgrade*.

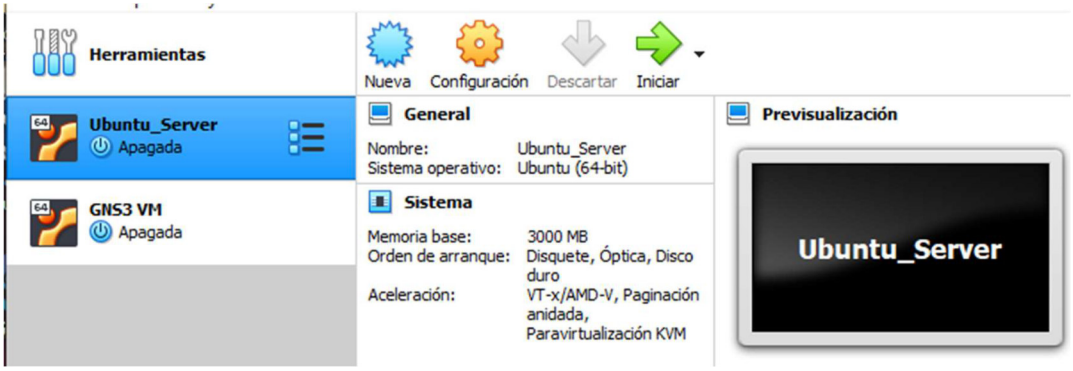


Figura 3.1 Máquina virtual para el nodo controlador

Después que haya finalizado la actualización de los repositorios, se usará el comando *sudo apt install software-properties-common*, que ayudará a añadir repositorios [13], al completar la instalación se usará el comando siguiente: *sudo add-apt-repository --yes --update ppa:ansible/ansible* para que sea más fácil instalar repositorios de terceros [14]. Posteriormente, se aloja las colecciones de Ansible con el comando *ansible-galaxy collection install cisco.ios*.

Ya completada las colecciones, se ejecuta el comando *sudo apt install ansible* para instalar el programa tal como se presenta en la Figura 3.2 . Adicionalmente, para corroborar que se haya instalado correctamente, se usa el comando *ansible --version* el cual despliega la versión más actual de Ansible, ver Figura 3.3 .

```
donovan@ubuntucontrolador:~$ sudo apt install ansible
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  ieee-data libfwupdplugin1 python3-argcomplete python3-dnspython python3-libcloud
  python3-lockfile python3-netaddr python3-selinux
Utilice «sudo apt autoremove» para eliminarlos.
Se instalarán los siguientes paquetes adicionales:
  ansible-core python3-bcrypt python3-packaging python3-paramiko python3-pyparsing
  python3-resolvelib sshpass
Paquetes sugeridos:
  python3-gssapi python-pyparsing-doc
Se instalarán los siguientes paquetes NUEVOS:
  ansible ansible-core python3-bcrypt python3-packaging python3-paramiko python3-pyparsing
  python3-resolvelib sshpass
0 actualizados, 8 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 22,3 MB de archivos.
Se utilizarán 312 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n]
```

Figura 3.2 Instalación Ansible

Figura 3.3 Comprobación de la instalación de Ansible

Seguidamente, se configuró la máquina virtual para que trabaje en una red interna, para ello se dirige a la carpeta *cd /etc/netplan;* mediante el comando *ls* se muestra el único archivo disponible, en este caso *00-instaler-config.yaml* el cual se abrirá con el comando *sudo nano*. Tras ingresar al archivo, se realizó los cambios que se muestran en la Figura 3.4 , los cuales implican deshabilitar DHCP de la máquina para asignar una dirección IP estática: 172.16.0.30 con máscara /16; se guardó el archivo y se ingresa el comando *sudo netplan apply* para aplicar los cambios y cambiar la anterior dirección IP.

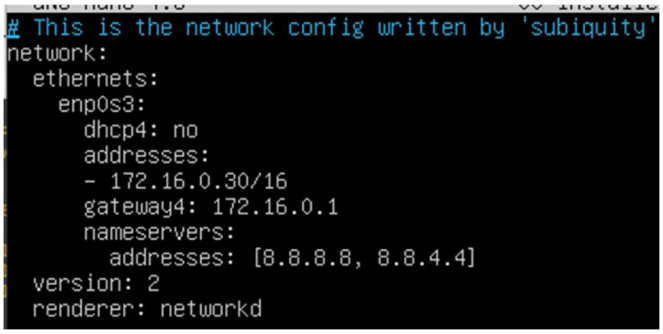


Figura 3.4 Asignación de dirección IP estática en el nodo controlador

3.2 Implementación de la topología en el simulador de red

Primero se procede a instalar GNS3 desde su página oficial junto a la máquina virtual la cual permitirá arrancar los terminales desde esa máquina y no desde la máquina local. Tras haber instalado el programa, se procedió a subir el servidor Ubuntu con el nodo controlador al simulador de red, desde la pestaña de *edits/preferences/VM machines*.

Luego se procedió a instalar los nodos administrados, en este caso un *router* y tres *switches*, en la aplicación de GNS3 con imágenes de Cisco que se encuentran en Internet. Posteriormente se realizó la topología presentada en la Figura 3.5 con los dispositivos finales y el nodo controlador. Esta topología cuenta con tres VLANs diferentes, mismas que se encuentran en un diferente espacio físico. En la Tabla 3.1 se

describe el nombre de los equipos, dirección IP, VLANs designadas entre otros datos de la topología a ser implementada.

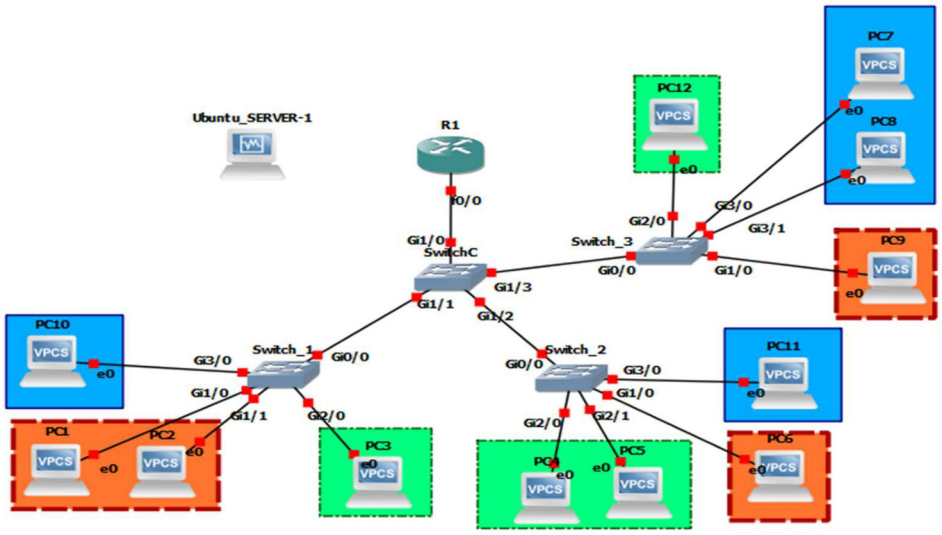


Figura 3.5 Nodo controlador (Máquina Virtual) y Nodos Administrados (*Router* y *Switches*)

Tabla 3.1 Direccionamiento de los equipos de la topología

Nombre Equipo	VLAN	Interfaz	Dirección IP	Máscara	Gateway
Ubuntu_SERVER	N/A	NIC	172.16.0.30	255.255.0.0	172.16.0.1
Router	N/A	Fa0/1	172.16.0.1	255.255.0.0	N/A
		Fa0/0	N/A	N/A	N/A
		Fa0/0.10	192.168.1.1	255.255.255.0	N/A
		Fa0/0.20	192.168.2.1	255.255.255.0	N/A
		Fa0/0.30	192.168.3.1	255.255.255.0	N/A
Switch Central	5	Gi0/1	172.16.0.2	255.255.0.0	N/A
	N/A	Gi1/0-3	N/A	N/A	N/A
Switch 1	5	Gi0/1	172.16.0.3	255.255.0.0	N/A
	N/A	Gi0/0	N/A	N/A	N/A
	10	Gi1/0-3	N/A	N/A	N/A
	20	Gi2/0-3	N/A	N/A	N/A
	30	Gi3/0-3	N/A	N/A	N/A
Switch 2	5	Gi0/1	172.16.0.4	255.255.0.0	N/A
	N/A	Gi0/0	N/A	N/A	N/A
	10	Gi1/0-3	N/A	N/A	N/A
	20	Gi2/0-3	N/A	N/A	N/A
	30	Gi3/0-3	N/A	N/A	N/A

Nombre Equipo	VLAN	Interfaz	Dirección IP	Máscara	Gateway
Switch 3	5	Gi0/1	172.16.0.5	255.255.0.0	N/A
	N/A	Gi0/0	N/A	N/A	N/A
	10	Gi1/0-3	N/A	N/A	N/A
	20	Gi2/0-3	N/A	N/A	N/A
	30	Gi3/0-3	N/A	N/A	N/A
PC1	10	---	192.168.1.10	255.255.255.0	192.168.1.1
PC2	10	---	192.168.1.11	255.255.255.0	192.168.1.1
PC6	10	---	192.168.1.12	255.255.255.0	192.168.1.1
PC9	10	---	192.168.1.13	255.255.255.0	192.168.1.1
PC3	20	---	192.168.2.10	255.255.255.0	192.168.2.1
PC4	20	---	192.168.2.11	255.255.255.0	192.168.2.1
PC5	20	---	192.168.2.12	255.255.255.0	192.168.2.1
PC12	20	---	192.168.2.13	255.255.255.0	192.168.2.1
PC7	30	---	192.168.3.10	255.255.255.0	192.168.3.1
PC8	30	---	192.168.3.11	255.255.255.0	192.168.3.1
PC10	30	---	192.168.3.12	255.255.255.0	192.168.3.1
PC11	30	---	192.168.3.13	255.255.255.0	192.168.3.1

Comunicación SSH entre el nodo controlador y los equipos de red

Para establecer conexión SSH entre el nodo controlador y los nodos administrados se debe instalar *Open* SSH, que se puede hacerlo al instalar la máquina virtual mostrado en la Figura 3.6 .

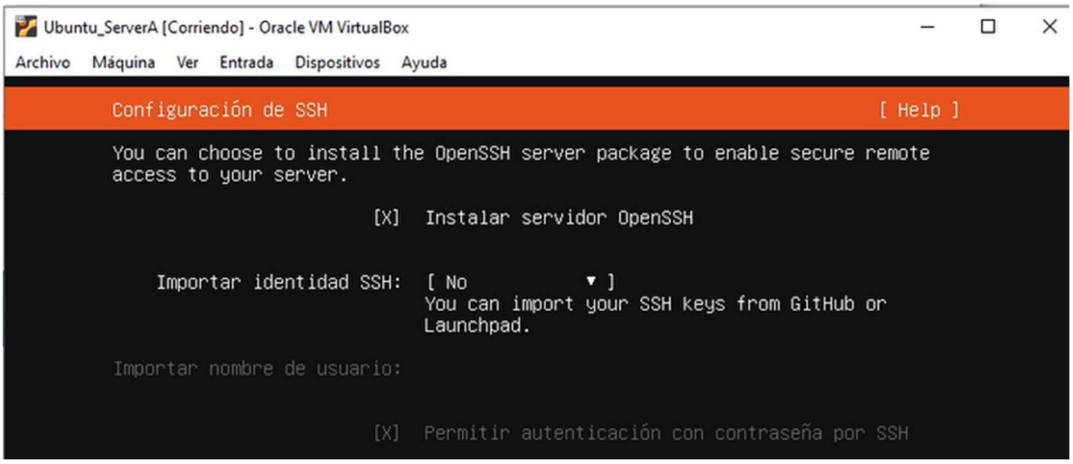


Figura 3.6 Instalación *Open* SSH

Luego se generó las llaves de SSH con el comando *ssh-keygen,* mostrado en la Figura 3.7 ; posteriormente se va a configurar el archivo *ssh_config* con el comando *sudo nano ssh_config,* ver la Figura 3.8 , para el encriptado y para generar llaves por cada conexión realizada.

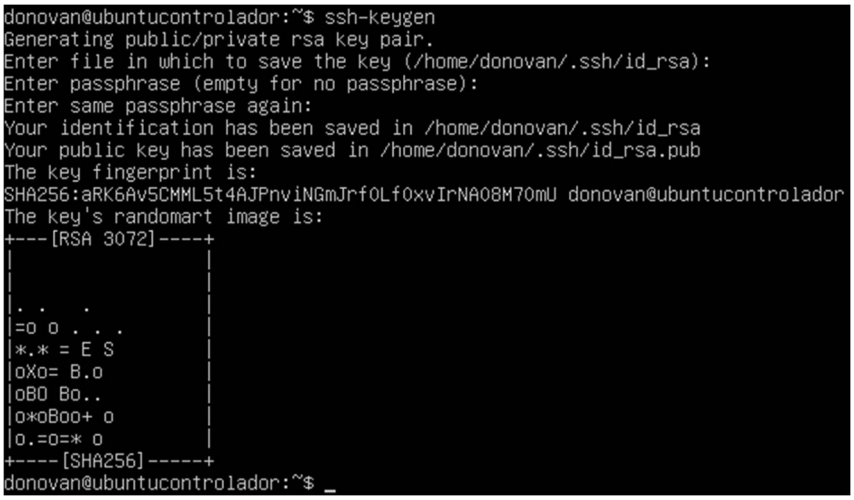


Figura 3.7 Implementación de llaves SSH

Figura 3.8 Dirección al archivo ssh_config

Una vez dentro del archivo, se quitó una línea comentada que es la que empieza con *Ciphers aes128-ctr, aes192-ctr, aes256-ctr, aes128-cbc, 3des-cbc* para el cifrado extremo a extremo y al final del archivo se añadió lo siguiente *KexAlgorithms +diffie-hellman-group1-sha1* para el intercambio de claves generadas por cada conexión empleada, ver Figura 3.9 Al finalizar, se guarda y se cierra el archivo; luego se configurará los quipos de *networking* para establecer comunicación con SSH.

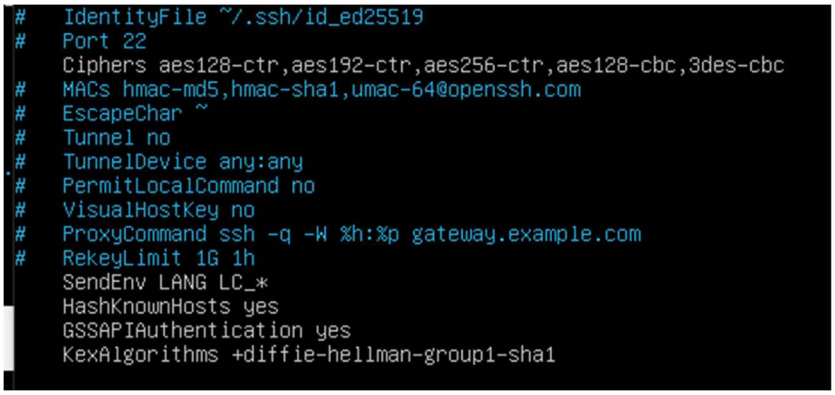


Figura 3.9 Configuración del archivo ssh_config

Como se ve en la Figura 3.10 , se configuró el *router* central, R1, para que este pueda establecer conexión SSH con el nodo controlador. Se debe ingresar a un puerto, en este caso al puerto *Fastethernet 0/1*, configurar su dirección IP y la máscara, que serán en este caso la dirección 172.16.0.1 255.255.0.0, al finalizar la configuración se levantó dicho puerto. Después, se asignó un nombre de dominio, R1, y se generó las llaves de encriptación con el comando *crypto key generate rsa*. Además, se asignó un usuario y contraseña, *RCentral* y 1234 respectivamente para ingresar al equipo.

Ya con esto se ingresó a la línea del terminal virtual, es decir a *line vty*, para realizar un registro de autenticación cada vez que se desee ingresar al equipo y se proporciona acceso remoto al equipo gracias al protocolo *SSH* con el comando *transport input ssh*.

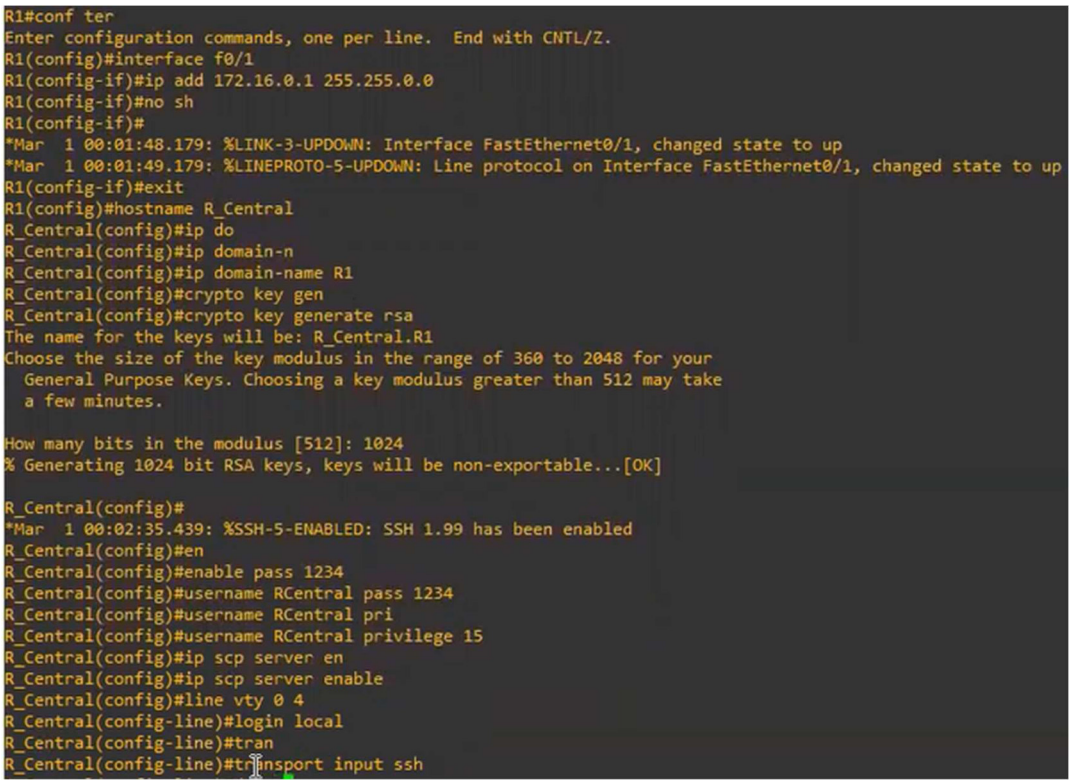


Figura 3.10 Configuración de SSH en el *router* central

Una vez finalizado la configuración en el *router*, se procedió a realizar la copia de la llave que se generó en el nodo controlador con el comando *ssh-copy-id Rcentral@172.16.0.1* dentro del directorio */etc/ssh*. Se da un *enter* y se digitará *yes*, tras hacerlo se mostrará un mensaje con el cual dirá que se podrá ingresar al *router* con el comando *ssh Rcentral@172.16.0.1, c*omo se muestra en la Figura 3.11 . Ya una vez dentro se podrá hacer cualquier cambio o consulta, ver Figura 3.12 , solo requiriendo del acceso remoto.

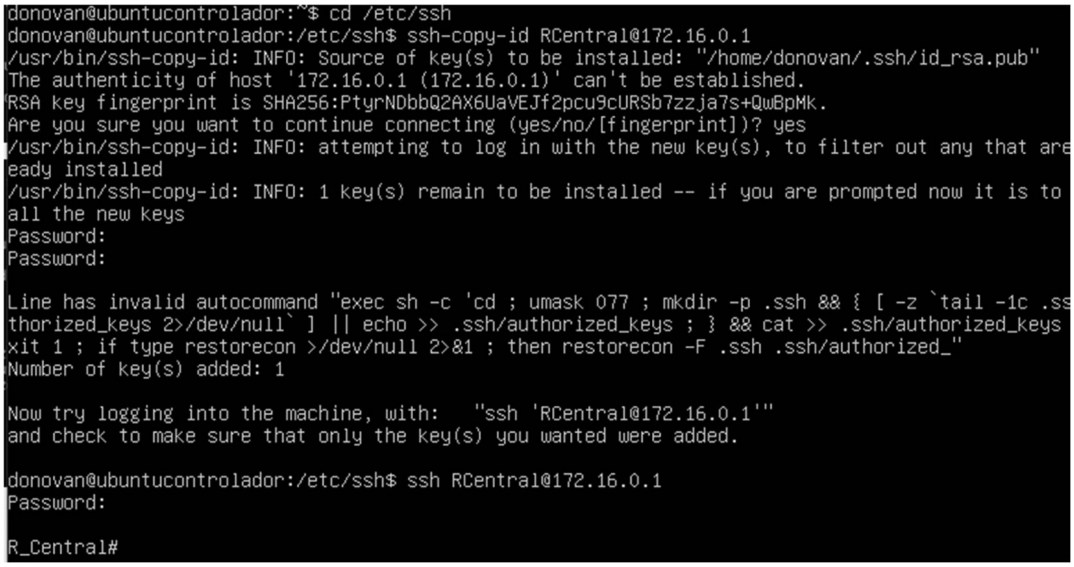


Figura 3.11 Conexión *router* central por SSH desde el nodo controlador

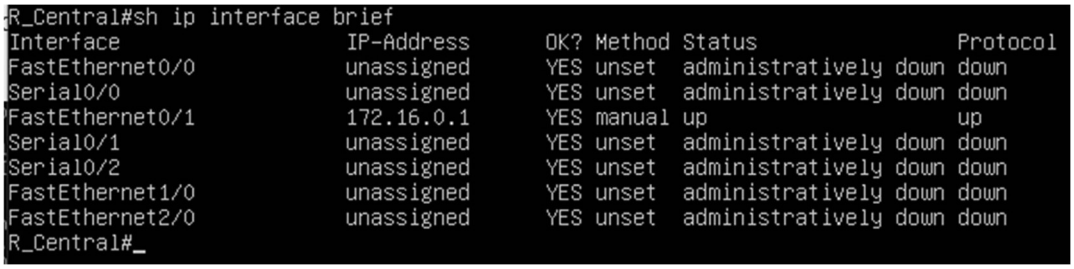


Figura 3.12 Comprobación comunicación SSH nodo controlador - *router* central

Por otro lado, para realizar la comunicación remota en los *switches* se empleó una VLAN, para el proyecto, se usó la VLAN 5, para asignar una dirección IP.

Entonces en el *switch* central se crea la VLAN con el comando *vlan 5*, luego se ingresa a la interfaz usando el comando *interface vlan 5* y se procede a asignar una dirección IP estática, la 172.16.0.2 255.255.0.0, además, con el comando *no shutdown,* se levanta la interfaz. Posteriormente, se asignó un nombre de dominio, SWC, y se generó las llaves de encriptación con *crypto key generate rsa,* también se usará el comando *ip ssh version* 2 para una mejor seguridad.

Luego se ingresó a la línea del terminal virtual, es decir a *line vty*, para realizar un registro de autenticación cada vez que se desee ingresar al equipo y se proporciona acceso remoto al equipo gracias al protocolo *SSH* con el comando *transport input ssh*. Además, se asignó un usuario y contraseña, *SCentral* y cisco1234 respectivamente para ingresar al equipo. Después, se ingresará a la interfaz GigabitEthernet0/1 para usar los

comandos *switchport mode Access* y *switchport Access vlan 5* asignando el puerto a la VLAN 5 para la comunicación SSH presentado en la Figura 3.13 .

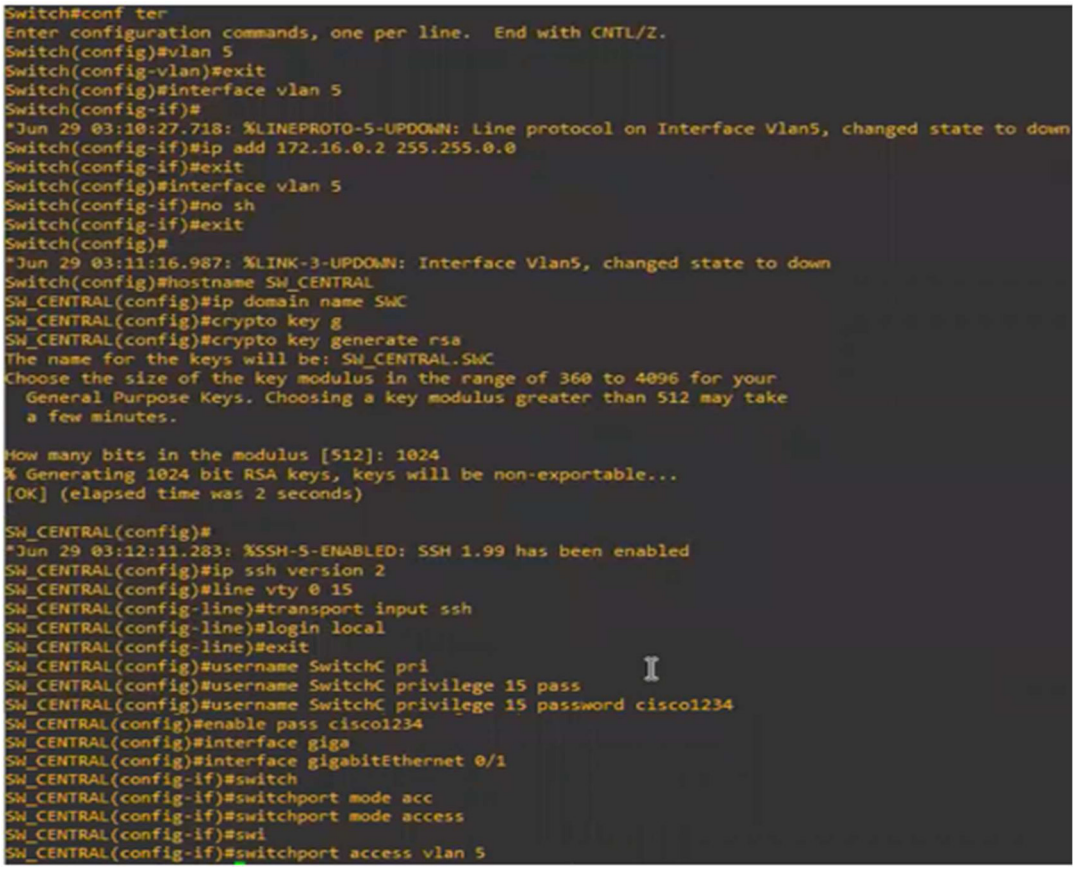


Figura 3.13 Configuración de SSH en el *switch* central

Tras finalizar con las configuraciones del *switch* central, en el nodo controlador se usará el comando *ssh-copy-id SwitchC@172.16.0.2* para otorgarle una llave pública al cual se pueda acceder al *switch* central. Al hacerlo se mostrará un mensaje con el cual dirá que se podrá ingresar al *switch* central con el comando *ssh SwitchC@172.16.0.2,* ver Figura 3.14 .

Figura 3.14 Conexión *Switch* central por SSH

Para la configuración de los demás *switches*, se usaron los mismos pasos aplicados para el *switch* central, pero con algunas diferencias que son la dirección IP, nombre de dominio, usuario y el comando para generar las llaves siendo la dirección 172.16.0.3 para el *switch* 1, 172.16.0.4 para el *switch* 2 y la 172.16.0.5 para el *switch* 3, los tres equipos con máscara 255.255.0.0. Después, el nombre de dominio se les asigna SW1, SW2 y SW3 respectivamente para cada *switch*; el nombre de usuario que se les asignará serán Sw1, Sw2 y Sw3, la contraseña será la misma para cada dispositivo siendo cisco1234. Las configuraciones se pueden observar en la Figura 3.15 Figura 3.17 Figura 3.19

Al finalizar con las configuraciones de cada *switch*, en el nodo controlador se usará el comando *ssh-copy-id (Nombre de usuario del equipo) @ (dirección IP estática del equipo)* para otorgarle una llave pública al cual pueda acceder a cada *switch*. Posteriormente, se usa el comando *ssh (Nombre de usuario del equipo) @ (dirección IP estática del equipo)* para poder acceder al mismo como muestran la Figura 3.16 , Figura 3.18 y Figura 3.20 .

```
Switch#conf ter
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#vlan 5
Switch(config-vlan)#exit
Switch(config)#interface vlan 5
Switch(config-if)#
*Jun 29 03:22:54.747: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan5, changed state to down
Switch(config-if)#ip add 172.16.0.3 255.255.0.0
Switch(config-if)#no sh
Switch(config-if)#exit
Switch(config)#
*Jun 29 03:23:14.553: %LINK-3-UPDOWN: Interface Vlan5, changed state to down
Switch(config)#hostname SW_1
SW_1(config)#ip domain-n
SW_1(config)#ip domain-name SW1
SW_1(config)#cry
SW_1(config)#crypto key
SW_1(config)#crypto key ge
SW_1(config)#crypto key generate rsa
The name for the keys will be: SW_1.SW1
Choose the size of the key modulus in the range of 360 to 4096 for your
  General Purpose Keys. Choosing a key modulus greater than 512 may take
  a few minutes.

How many bits in the modulus [512]: 1024
% Generating 1024 bit RSA keys, keys will be non-exportable...
[OK] (elapsed time was 1 seconds)

SW_1(config)#
*Jun 29 03:24:02.935: %SSH-5-ENABLED: SSH 1.99 has been enabled
SW_1(config)#ip ssh version 2
SW_1(config)#line vty 0 15
SW_1(config-line)#transport input ssh
SW_1(config-line)#login local
SW_1(config-line)#exit
SW_1(config)#username SW1 privilege 15 pass
SW_1(config)#username SW1 privilege 15 password cisco1234
SW_1(config)#enable pass cisco1234
SW_1(config)#interface gi
SW_1(config)#interface gigabitEthernet 0/1
SW_1(config-if)#swi
SW_1(config-if)#switchport mode acc
SW_1(config-if)#switchport mode access
SW_1(config-if)#swi
SW_1(config-if)#switchport access vlan 5
SW_1(config-if)#
```

Figura 3.15 Configuración de *switch* 1 para la comunicación SSH

```
donovan@ubuntucontrolador:~$ ssh Sw1@172.16.0.3

**************************************************************************
* IOSv is strictly limited to use for evaluation, demonstration and IOS  *
* education. IOSv is provided as-is and is not supported by Cisco's      *
* Technical Advisory Center. Any use or disclosure, in whole or in part, *
* of the IOSv Software or Documentation to any third party for any       *
* purposes is expressly prohibited except as otherwise authorized by     *
* Cisco in writing.                                                      *
Password: **************************************************************************

**************************************************************************
* IOSv is strictly limited to use for evaluation, demonstration and IOS  *
* education. IOSv is provided as-is and is not supported by Cisco's      *
* Technical Advisory Center. Any use or disclosure, in whole or in part, *
* of the IOSv Software or Documentation to any third party for any       *
* purposes is expressly prohibited except as otherwise authorized by     *
* Cisco in writing.                                                      *
**************************************************************************
SW_1#
SW_1#
```

Figura 3.16 Conexión *switch* 1 por SSH

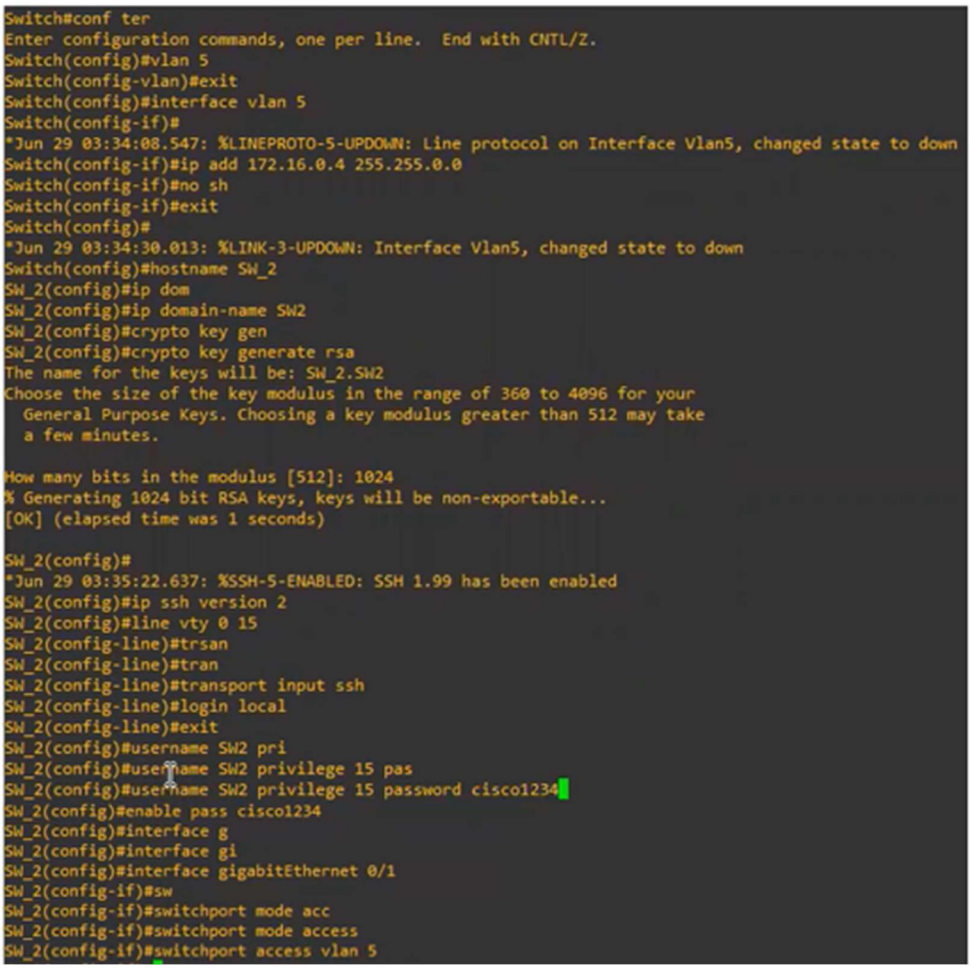


Figura 3.17 Configuración de *switch* 2 para la comunicación SSH

Figura 3.18 Conexión *switch* 2 por SSH

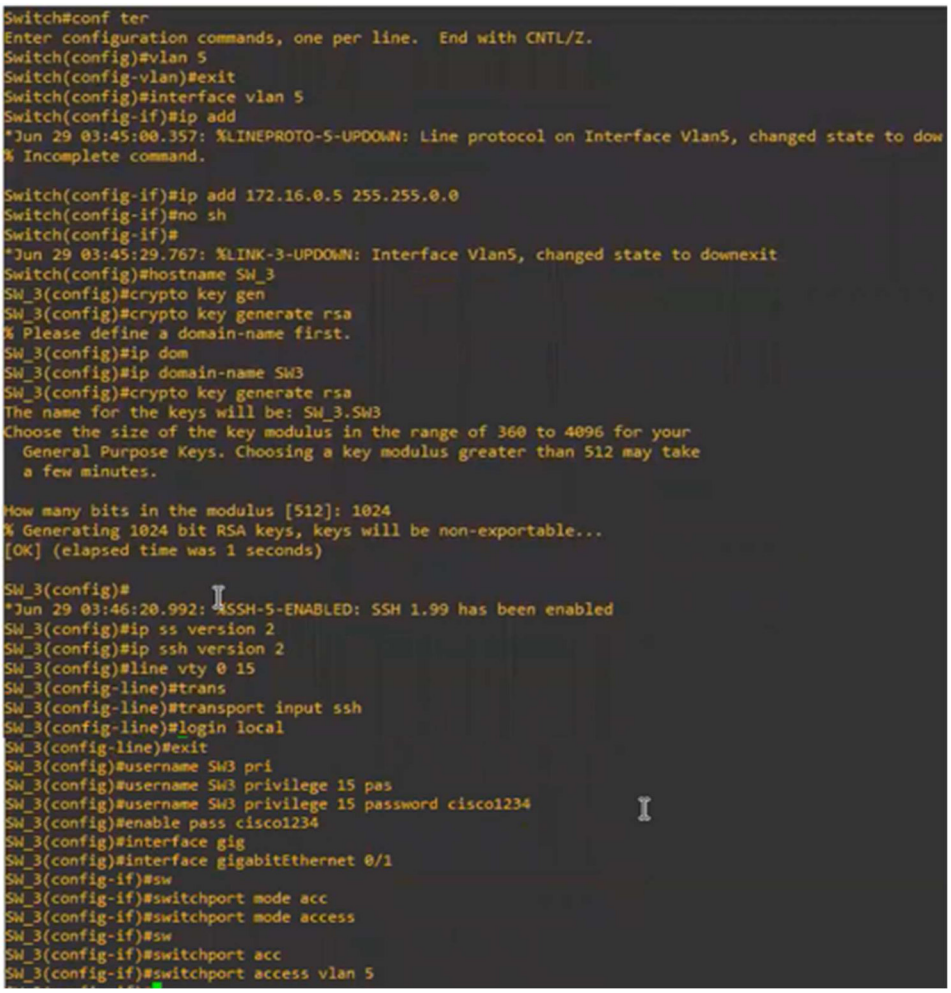


Figura 3.19 Configuración de *switch* 3 para la comunicación SSH

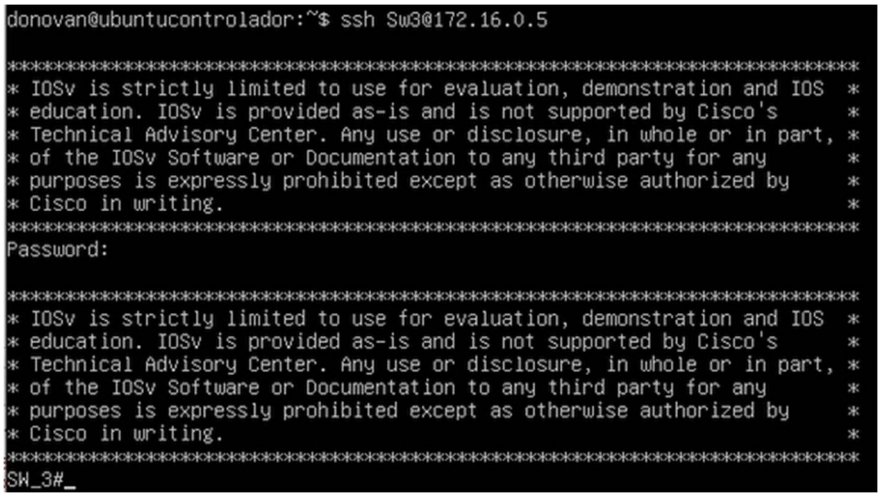


Figura 3.20 Conexión *switch* 3 por SSH

3.3 Implementación de las configuraciones con Ansible

Para empezar la implementación de los *playbooks*, en el nodo controlador, se debe configurar el archivo de *hosts* que se localizará dentro de la dirección *cd /etc/ansible,* en este se añadió un nombre del equipo entre corchetes, seguido de la dirección IP del equipo que se haya configurado, ver Figura 3.21 . Además, se les asignó variables que ayudaron a la conexión de los equipos con el nodo controlador, las cuales cuentan con las credenciales de los equipos: nombre del equipo, la contraseña, el tipo de conexión y el tipo de equipo que Ansible se va a conectar, en este caso un equipo de IOS.

```
  GNU nano 4.8                                                  hosts
[router]
172.16.0.1
[switchc]
172.16.0.2
[switch1]
172.16.0.3
[switch2]
172.16.0.4
[switch3]
172.16.0.5
[router:vars]
ansible_user=RCentral
ansible_password=1234
ansible_connection=network_cli
ansible_network_os=ios
[switchc:vars]
ansible_user=SwitchC
ansible_password=cisco1234
ansible_connection=network_cli
ansible_network_os=ios
[switch1:vars]
ansible_user=SW1
ansible_password=cisco1234
ansible_connection=network_cli
ansible_network_os=ios
[switch2:vars]
ansible_user=SW2
ansible_password=cisco1234
ansible_connection=network_cli
ansible_network_os=ios
[switch3:vars]
ansible_user=SW3
ansible_password=cisco1234
ansible_connection=network_cli
ansible_network_os=ios
```

Figura 3.21 Configuración del archivo inventario *hosts*

Tras ubicar a cada equipo de *networking* en el archivo *hosts*, se empieza a crear el *playbook* para el *router* con el comando *sudo nano RC.yml* dentro de la carpeta de Ansible. Al ingresar al documento creado se procede con el desarrollo del mismo empezando con tres guiones al inicio, luego en la siguiente línea será el objetivo o la tarea general que Ansible realizará.

Seguidamente, se llamó al equipo en cuestión, en este caso se llamó al *host* denominado *router*, después se realizó una lista que definirán las operaciones que realizará Ansible al ejecutar el *playbook* siendo *tasks* el formato para organizar dichas operaciones.

Para cada operación se asignó un nombre a cada tarea, siendo la primera el levantamiento de un puerto, seguido se llamó a un módulo en específico para ejecutar el nodo administrado correctamente, en este caso el módulo *ios_config* que permitirá ingresar comandos que normalmente se usan en un equipo de *networking* pero dentro del *playbook*.

Después se generó una lista donde irán los comandos, siendo en este caso *lines*, dentro de ella se usó el comando *no shutdown* para levantar el puerto, seguido de la línea *parents* que permite especificar el puerto donde se ejecutaron los comandos en la lista de *lines*, en este caso se levantó la interfaz FastEthernet0/0.

En la siguiente tarea se procede con la creación o configuración de las subinterfaces, para ello se vuelve a llamar al módulo y dentro de la lista se creó tres subinterfaces con el comando *interface FasthEthernet0/0.10*, *interface FasthEthernet0/0.20* e *interface FasthEthernet0/0.30* para la VLAN 10, 20 y 30 respectivamente. La siguiente tarea consistió en asignar una dirección IP estática a la subinterfaz FastEthernet0/0.10 con los comandos *encapsulation dot1Q 10* e *ip address 192.168.1.1* con máscara /24. Para la subinterfaz se le asignó una dirección IP 192.168.2.1 con máscara /24 con *encapsulation dot1Q 20* y para la subinterfaz FastEthernet0/0.30 la dirección 192.168.3.1 con máscara 24 *encapsulation dot1Q 30*.

Tras finalizar con el *playbook*, al final de la última tarea de usó el comando *do wr* para guardar los cambios al acabar de ejecutar el *playbook,* presentado en la Figura 3.22 . Se guardó la configuración del *playbook* del *router* y se salió para crear otro.

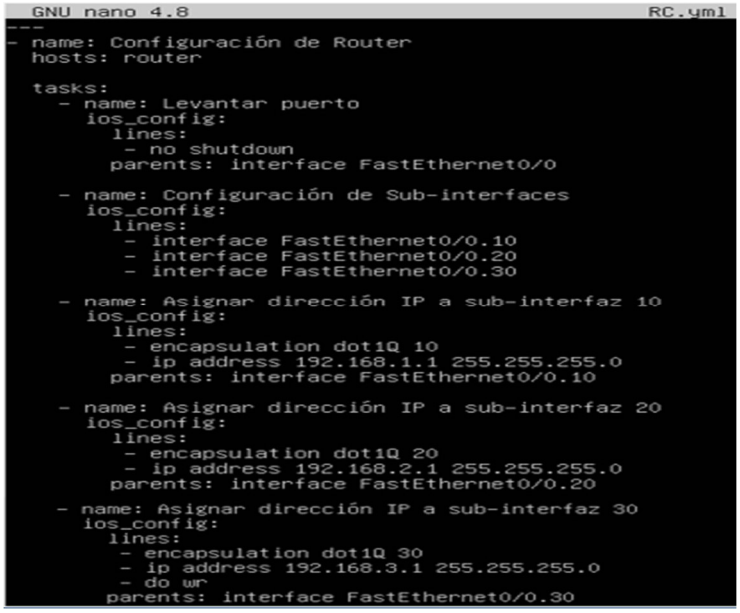


Figura 3.22 *Playbook router* central

Posteriormente, tras configurar el *playbook* del *router* central se creó otro *playbook* con el comando *sudo nano SWC.yml* para el switch central. Se empieza con tres guiones al inicio del archivo y se inicia con la configuración del *switch* central, llamando al equipo con la línea de *hosts*.

Después, se crearon las tareas que va a realizar el *switch*, siendo la primera la de establecer las VLANs 10, 20 y 30 dentro de la lista de la tarea. En la siguiente tarea se estableció un rango de interfaces troncales, para ello se usó el mismo módulo, *ios_config*, y se colocaron los comandos *switchport trunk encapsulation dot1Q*, luego se usó *switchport mode trunk* el cual habilitó el modo troncal y se permite que las VLANs 10, 20 y 30 pasen por estas interfaces con *switchport trunk allowed vlan 10, 20, 30.* Al finalizar con la tarea se usó el comando *do wr* para guardar los cambios tras el despliegue del *playbook*. Todos estos comandos fueron asignados desde la interfaz GigabitEthernet1/0 hasta la GigabitEthernet1/3 con el comando *interface range GigabitEthernet1/0-3* dentro de la línea *parents* mostrado en la Figura 3.23 .

```
 GNU nano 4.8                                  SWC.yml
---
- name: Configuración Switch Central
  hosts: switchc

  tasks:
    - name: Establecer VLANs
      ios_config:
        lines:
          - vlan 10
          - vlan 20
          - vlan 30

    - name: Establecer puertos troncales a las interfaces
      ios_config:
        lines:
          - switchport trunk encapsulation dot1Q
          - switchport mode trunk
          - switchport trunk allowed vlan 10,20,30
          - do wr
        parents: interface range GigabitEthernet1/0-3
```

Figura 3.23 *Playbook switch* central

Para continuar con la configuración de los *playbooks*, se creó un *playbook* para los tres *switches* restantes, ya que comparten las mismas configuraciones, pero solo cambiará el nombre del equipo a llamar. Para ello se usó el comando *sudo nano SW1.yml* y dentro de este se puso los tres guiones al inicio del *playbook*, luego se escribió la tarea principal que realizará Ansible que es la configuración del *switch* 1, después se llamó al equipo con la línea de *hosts*.

Posteriormente, se realizarán las tareas dentro de una lista con *tasks*, la primera tarea en el *switch* 1 será establecer las VLANs, para ello se llamó al comando *ios_config* y dentro de *lines,* se añadió las VLANs que son la 10, 20 y 30. En la siguiente tarea se asignó a la VLAN 10 a un conjunto de interfaces, después se empleó los comandos *switchport mode access* y *switchport access vlan 10*, estos dos comandos se asignarán o alojarán en las interfaces desde la GigabitEthernet1/0 hasta la 1/3, para ello se usó el comando *interface range GigabitEthernet1/0-3* dentro de *parents*.

Para la siguiente tarea, se asignó la VLAN 20, se seguirán los mismos pasos de la VLAN 10, dentro de *lines* se usaron *switchport mode access* y *switchport access vlan 20* junto al comando *interface range GigabitEthernet2/0-3* dentro de *parents*.

Posteriormente se tiene la tarea donde se asigna la VLAN 30, los comandos son *switchport mode access* y *switchport access vlan 30* con el comando *interface range GigabitEthernet3/0-3* dentro de *parents*.

La última tarea consistió en asignar un puerto troncal, interfaz GigabitEthernet0/0, para ello se llamó al mismo comando y dentro de *lines* se usaron los comandos *switchport trunk encapsulation dot1Q*, con *switchport mode trunk* se habilitó el modo troncal y se permite que las VLANs 10, 20 y 30 pasen por esta interfaz con *switchport trunk allowed vlan 10,20,30*. Al final de la tarea se empleó el comando *do wr* para que luego de la ejecución del *playbook* este guarde los cambios de manera automática, ver Figura 3.24 .

Esta configuración del *switch* 1, servirá tanto para el *switch* 2 y 3 pero para que se configuren correctamente, en la línea de *hosts* se cambiará el nombre del equipo, en este caso *switch2* y *switch3* respectivamente para llamar al equipo correspondiente.

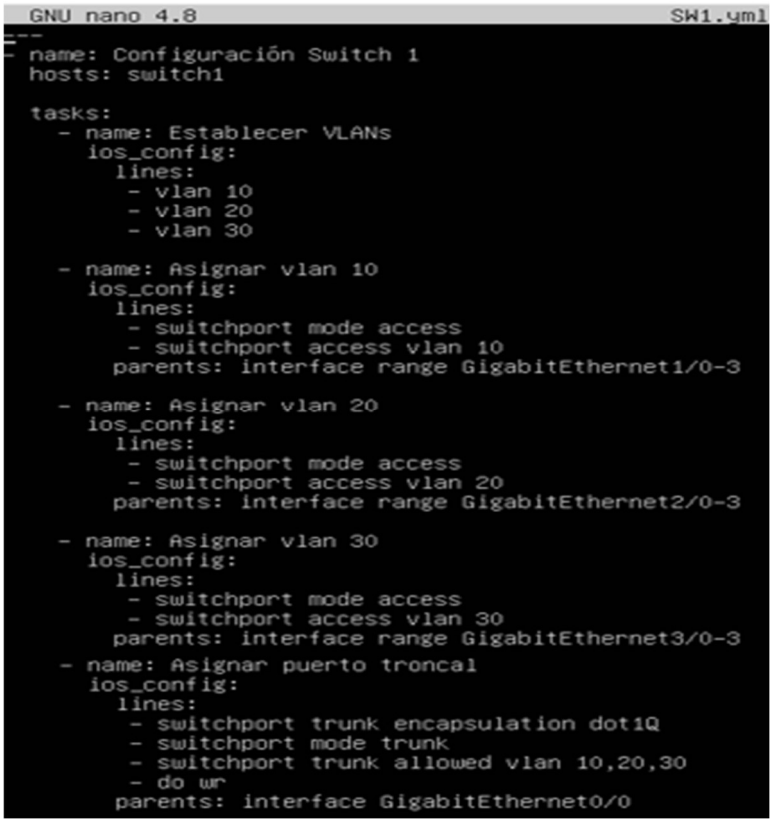


Figura 3.24 *Playbook* para los *Switches*

3.4 Pruebas de funcionamiento y verificación

Router

Para empezar con la ejecución del *playbook*, primero se conecta el nodo controlador con el nodo administrado, en este caso el *router*; después, se comprobó que las interfaces en el *router* no tengan nada configurado, utilizando el comando *show running-config* mostrado en la Figura 3.25 .

Figura 3.25 Puertos *router* central

Tras esta comprobación, se procede a ejecutar el *playbook* del mismo con el comando *ansible-playbook RC.yml.* Ansible inicia la ejecución de las tareas, como levantar el puerto, luego realiza las respectivas configuraciones en las subinterfaces siendo estas la FastEthernet0/0.10, la FastEthernet0/0.20 y la FastEthernet0/0.30. Si no se produjo ningún error, durante la ejecución del *playbook,* todos los mensajes se mostrarán en amarillo con el total de cambios que se hicieron junto a la dirección IP del equipo, ver Figura 3.26 .

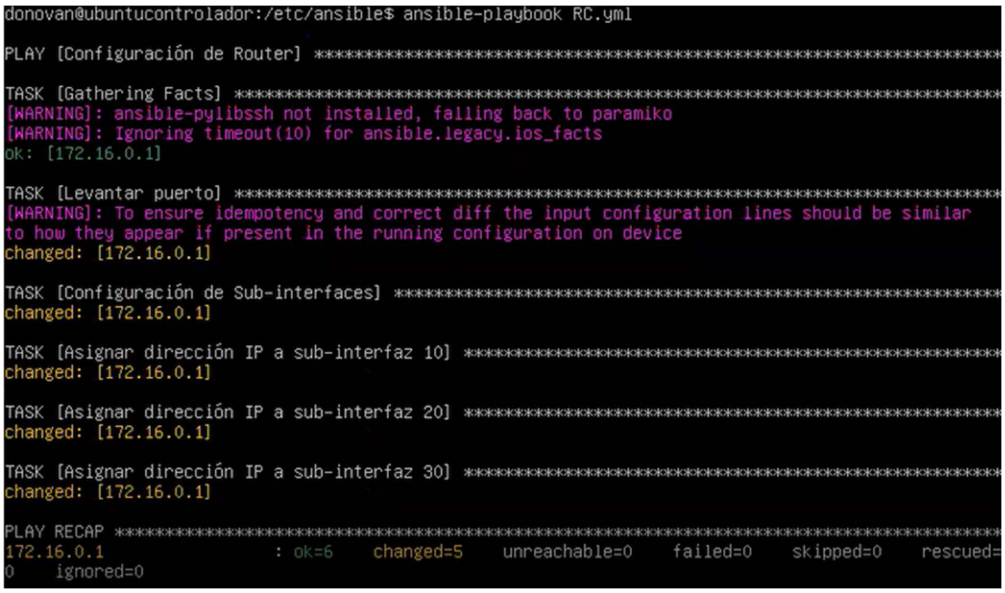


Figura 3.26 Ejecución del *playbook router* central

En el transcurso de la ejecución del *playbook* del *router* central se mostró en el terminal del *router* cómo se van aplicando los cambios uno por uno, mostrado en la

Figura **3.27** .

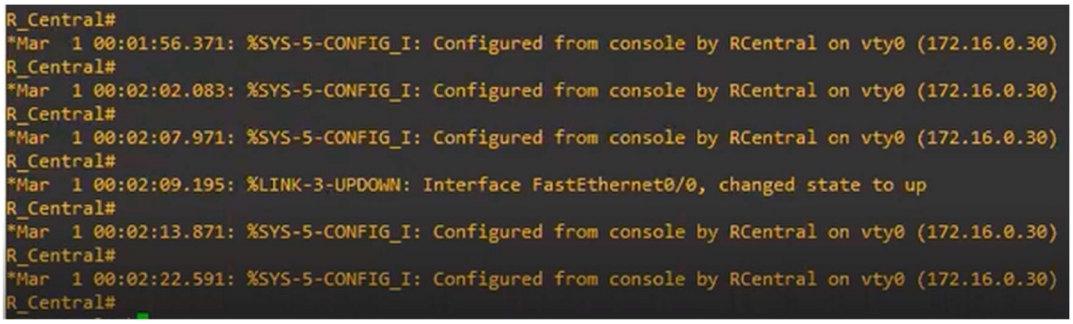


Figura 3.27 Información de cambios en *router* central

Adicionalmente, al finalizar esta ejecución se pudo verificar los cambios realizados en el *router*, se utilizó el comando *show running-config*, ver Figura 3.28 .

Figura 3.28 Cambios realizados en el *router* central

Switch central

Al acabar con el *router* central, se conectó el nodo controlador con el nodo administrado, en este caso el *switch* central. Para empezar con la ejecución del *playbook* del *switch* central, se comprobó que las interfaces del mismo no se encuentren con ninguna configuración realizada, mediante el comando *show running-config* mostrado en la Figura 3.29 .

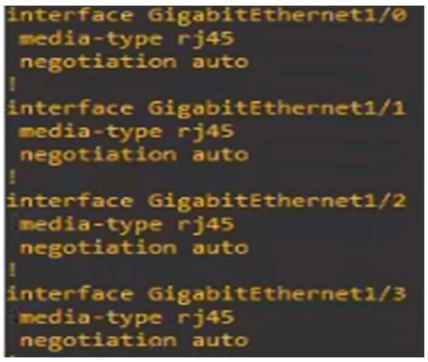


Figura 3.29 Puertos *switch* central

Tras esta verificación inicial, se procede a ejecutar el *playbook* del mismo con el comando *ansible-playbook SWC.yml*, se dio un *enter* y empezó la configuración del *switch.* En primera instancia se establecieron las VLANs, si no se produjo ningún error durante su ejecución, todos los mensajes se mostrarán en amarillo con el total de

cambios que se hicieron junto a la dirección IP del equipo, presentado en la Figura 3.30
.

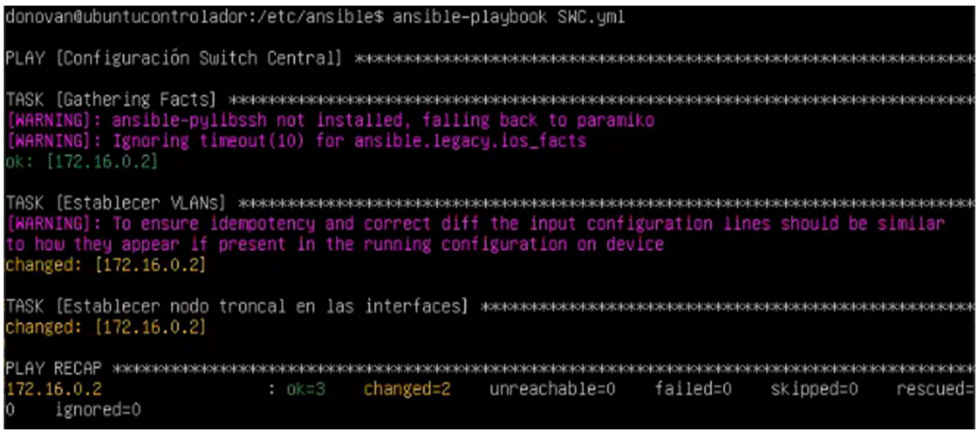


Figura 3.30 Ejecución *playbook switch* central

En el transcurso de la ejecución del *playbook* del *switch* central se irá mostrando en el terminal del mismo cómo se van aplicando los cambios uno por uno, además de que muestra como el *switch* ha guardado los cambios de forma automática, ver Figura 3.31 . Adicionalmente, cuando haya finalizado Ansible con el *playbook*, se podrán verificar los cambios realizados al usar *show running-config*, ver Figura 3.32 .

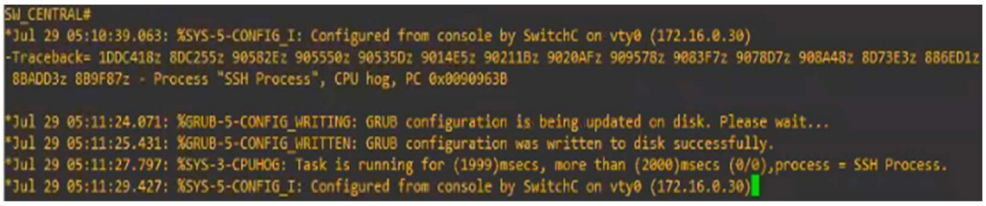


Figura 3.31 Información de cambios en el *switch* central

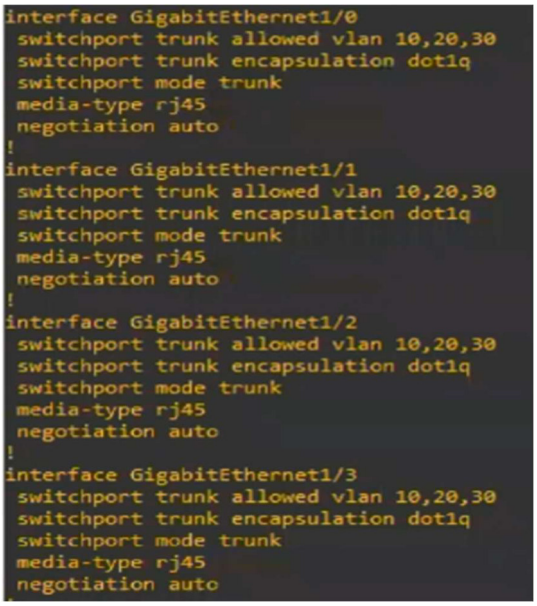


Figura 3.32 Cambios realizados en el *switch* central

Switch 1

Al terminar con el *switch* central, se conectó el nodo controlador con el nodo administrado, en este caso el *switch* 1. Para empezar con la ejecución del *playbook* del *switch* 1, se comprobó que las interfaces del mismo no se encuentren con ninguna configuración realizada, mediante *show running-config* mostrado en la Figura 3.33 Además, se observó que las VLANs 10, 20 y 30 no estuvieran activadas con el comando *show vlan*, mostrado en la Figura 3.34 .

Figura 3.33 Puertos en el *switch* 1

```
VLAN Name                              Status    Ports
---- --------------------------------- --------- -------------------------
1    default                           active    Gi0/0, Gi0/2, Gi0/3, Gi1/0
                                                 Gi1/1, Gi1/2, Gi1/3, Gi2/0
                                                 Gi2/1, Gi2/2, Gi2/3, Gi3/0
                                                 Gi3/1, Gi3/2, Gi3/3
5    VLAN0005                          active    Gi0/1
1002 fddi-default                      act/unsup
1003 token-ring-default                act/unsup
1004 fddinet-default                   act/unsup
1005 trnet-default                     act/unsup
```

Figura 3.34 VLANs en el *switch* 1

Tras esta verificación inicial y que no se hayan asignado interfaces a una VLAN previamente, se procede a ejecutar el *playbook* del equipo con el comando *ansible-playbook SW1.yml*, se dio un *enter* y se empezó con la configuración del *switch* 1. En primera instancia se establecieron las VLANs y luego la asignación de puertos en las mismas. Si no se produjo ningún error durante su ejecución, todos los mensajes se mostrarán en amarillo con el total de cambios que se hicieron junto a la dirección IP del equipo presentado en la Figura 3.35 .

```
donovan@ubuntucontrolador:/etc/ansible$ ansible-playbook SW1.yml

PLAY [Configuración Switch 1] ***************************************************************************

TASK [Gathering Facts] **********************************************************************************
[WARNING]: ansible-pylibssh not installed, falling back to paramiko
[WARNING]: Ignoring timeout(10) for ansible.legacy.ios_facts
ok: [172.16.0.3]

TASK [Establecer VLANs] *********************************************************************************
[WARNING]: To ensure idempotency and correct diff the input configuration lines should be similar
to how they appear if present in the running configuration on device
changed: [172.16.0.3]

TASK [Asignar vlan 10] **********************************************************************************
changed: [172.16.0.3]

TASK [Asignar vlan 20] **********************************************************************************
changed: [172.16.0.3]

TASK [Asignar vlan 30] **********************************************************************************
changed: [172.16.0.3]

TASK [Asignar puerto troncal] ***************************************************************************
changed: [172.16.0.3]

PLAY RECAP **********************************************************************************************
172.16.0.3                 : ok=6    changed=5    unreachable=0    failed=0    skipped=0    rescued=
0    ignored=0
```

Figura 3.35 Ejecución *playbook switch* 1

En el transcurso de la ejecución del *playbook* del *switch* 1, se irá mostrando en el terminal del mismo cómo se van aplicando los cambios uno por uno, además muestra cómo el *switch* ha guardado los cambios de forma automática, ver Figura 3.36 .

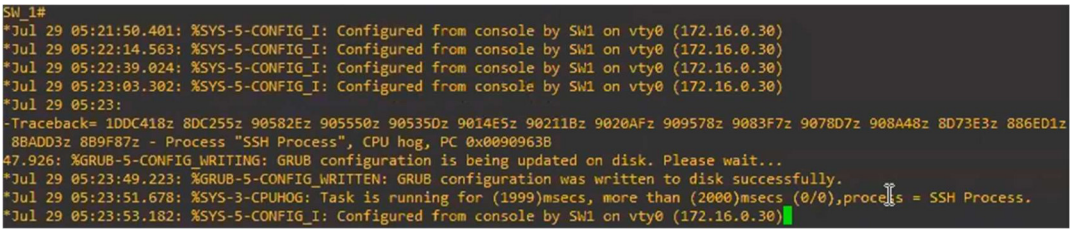


Figura 3.36 Tareas realizadas en el *switch* 1

Adicionalmente, cuando finalizó la ejecución del *playbook*, se observaron los cambios realizados al usar el comando *show running-config*, ver Figura 3.37 . También se puede evidenciar cómo las VLANs se crearon y cómo los puertos fueron asignados a las mismas, ver Figura 3.38 .

Figura 3.37 Cambios realizados en el *switch* 1

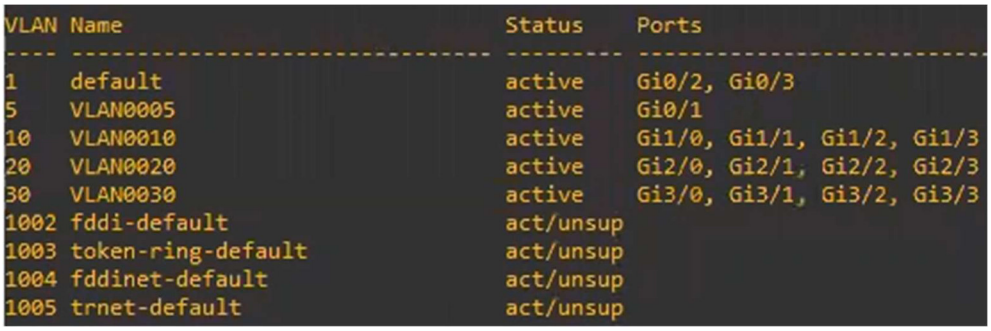


Figura 3.38 VLANs asignadas en el *switch* 1

Switch 2

Al acabar con el *switch* 1, se conectó el nodo controlador al *switch* 2. Antes de iniciar con la ejecución del *playbook* del *switch* 2 se comprobó que las interfaces del mismo no se encuentren con ninguna configuración previa. Se utilizó el comando *show running-config* mostrado en la Figura 3.39 se verificó que las VLANs 10, 20 y 30 no estuvieran establecidas con anterioridad con *show vlan* presentado en la Figura 3.40 .

Figura 3.39 Puertos *switch* 2

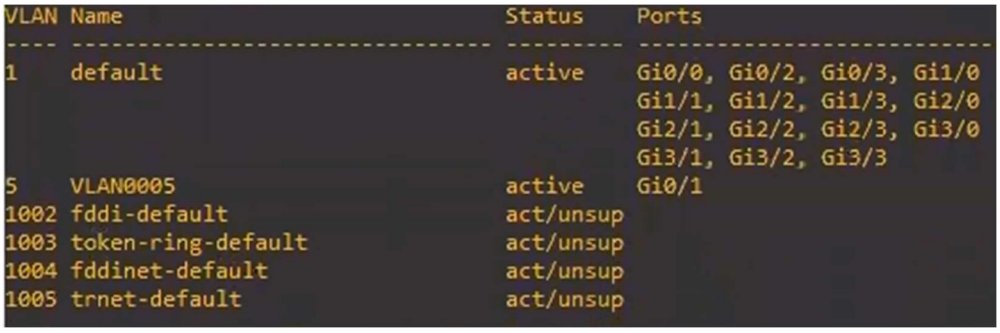


Figura 3.40 VLANs en *switch* 2

Tras esta inspección inicial, se procede a ejecutar el *playbook* del mismo con el comando *ansible-playbook SW2.yml*, se dio un *enter* y empezó con la configuración del *switch.* En primera instancia se establecieron las VLANs y luego la asignación de puertos en las mismas; si no se produjo ningún error durante la ejecución del *playbook* todos los mensajes se mostrarán en amarillo con el total de cambios que se hicieron junto a la dirección IP del equipo presentado en la Figura 3.41 .

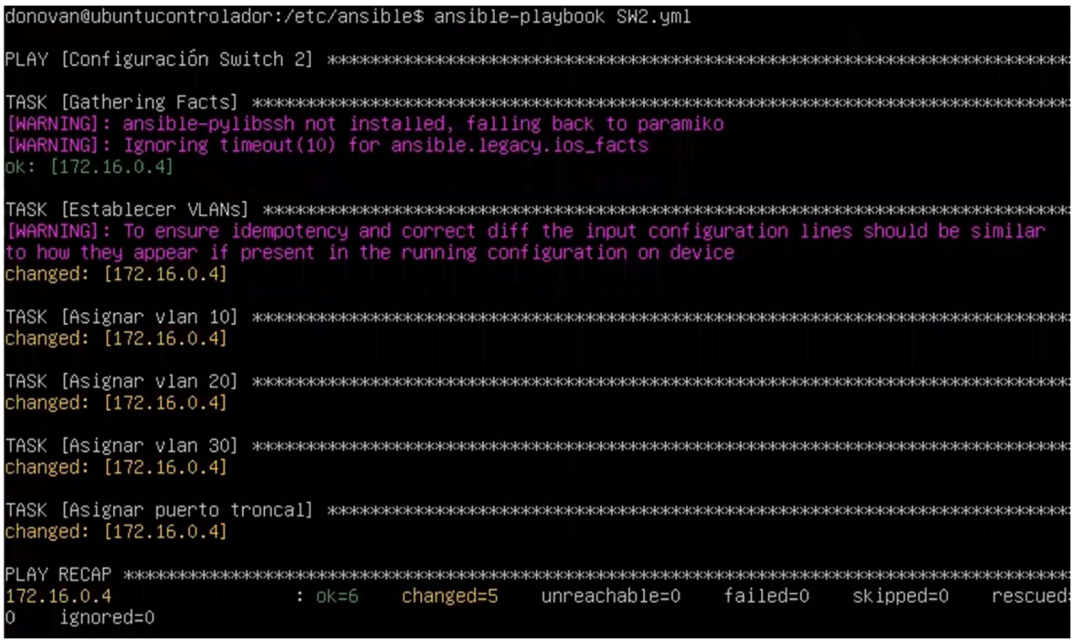


Figura 3.41 Ejecución *playbook switch* 2

En el transcurso de la ejecución del *playbook* del *switch* 2 se irá mostrando en el terminal del mismo cómo se van aplicando los cambios uno por uno, también muestra cómo el *switch* ha guardado los cambios de forma automática, ver Figura 3.42 .

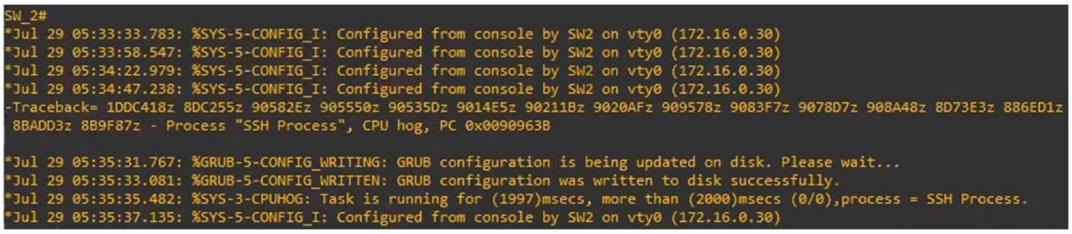
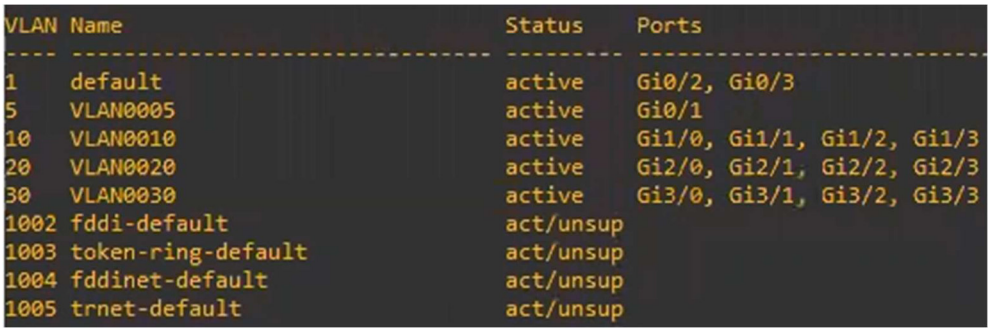


Figura 3.42 Tareas realizadas en el *switch* 2

Adicionalmente, cuando haya finalizado la ejecución del *playbook*, se podrán presenciar los cambios realizados al usar el comando *show running-config*, ver Figura 3.43 . También se puede evidenciar las VLANs creadas y los puertos asignados a las mismas, ver Figura 3.44 .

Figura 3.43 Cambios realizados en el *switch* 2



VLAN	Name	Status	Ports
1	default	active	Gi0/2, Gi0/3
5	VLAN0005	active	Gi0/1
10	VLAN0010	active	Gi1/0, Gi1/1, Gi1/2, Gi1/3
20	VLAN0020	active	Gi2/0, Gi2/1, Gi2/2, Gi2/3
30	VLAN0030	active	Gi3/0, Gi3/1, Gi3/2, Gi3/3
1002	fddi-default	act/unsup	
1003	token-ring-default	act/unsup	
1004	fddinet-default	act/unsup	
1005	trnet-default	act/unsup	

Figura 3.44 VLANs asignadas en el *switch* 2

Switch 3

Al finalizar con el *switch* 2, se conectó el nodo controlador con el *switch* 3. En primera instancia se comprobó que las interfaces del mismo no se encuentren con ninguna configuración previa, por lo que se utilizó el comando *show running-config* mostrado en la Figura 3.45 . Además, se observó con el comando *show vlan* que las VLANs 10, 20 y 30 no estuvieran activadas, ver Figura 3.46 .

Figura 3.45 Puertos del *switch* 3

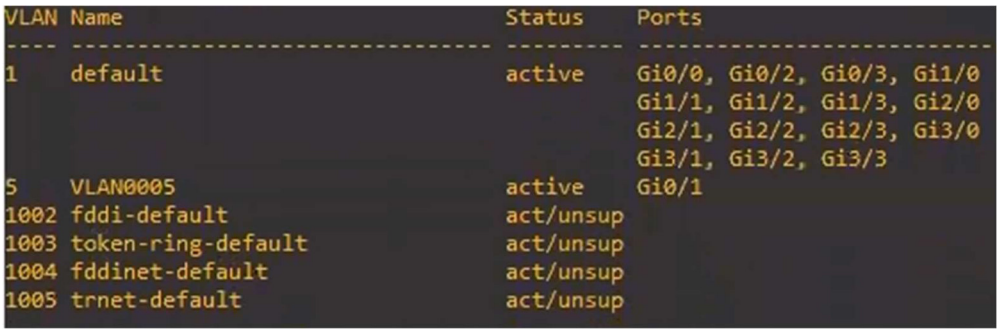


Figura 3.46 VLANs en el *switch* 3

Tras esta verificación inicial y que no se haya asignado ninguna VLAN previamente, se procede a ejecutar el *playbook* del mismo con el comando *ansible-playbook SW3.yml*, se dio un *enter* y empezó con la configuración del *switch.* En primera instancia se establecieron las VLANs y luego la asignación de puertos en las mismas, si no se produjo ningún error durante la ejecución del *playbook* todos los mensajes se mostrarán en amarillo con el total de cambios que se hicieron junto a la dirección del equipo mostrado en la Figura 3.47 .

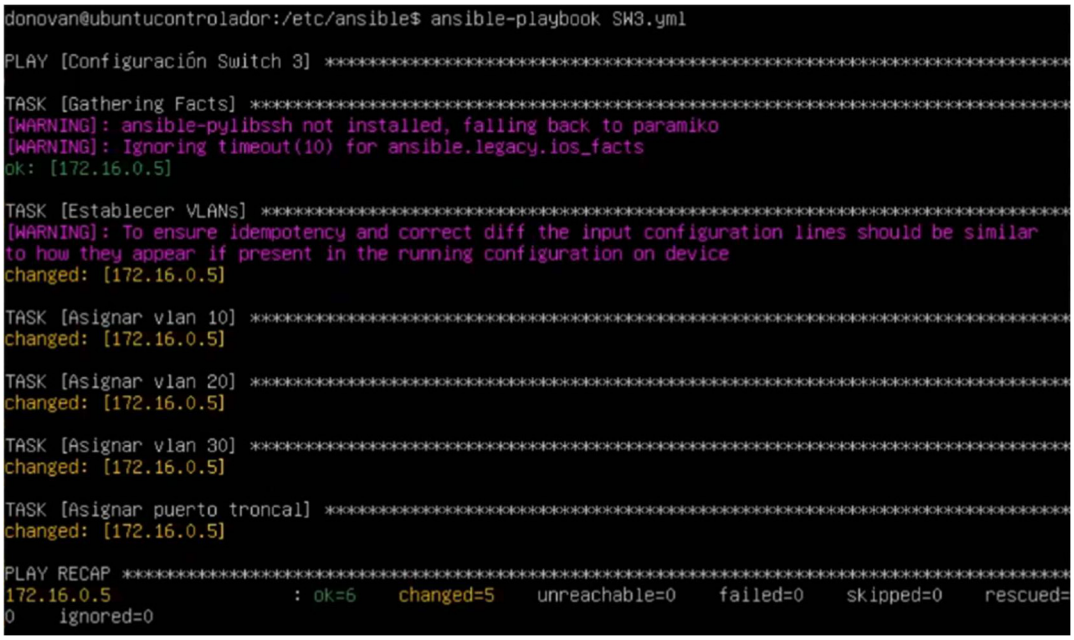


Figura 3.47 Ejecución *playbook switch* 3

En el transcurso de la ejecución del *playbook* del *switch* 3 se irá mostrando en el terminal del mismo cómo se van aplicando los cambios uno por uno, además muestra que el *switch* ha guardado los cambios de forma automática, ver Figura 3.48 .

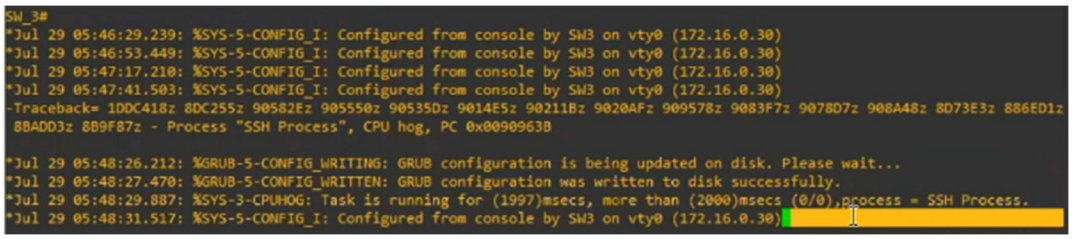


Figura 3.48 Tareas realizadas en el *switch* 3

Cuando haya finalizado Ansible con el *playbook*, se podrán presenciar los cambios realizados al usar *show running-config*, ver Figura 3.49 . También se puede evidenciar las VLANs creadas y cómo los puertos fueron asignados a las mismas, ver Figura 3.50 .

Figura 3.49 Cambios realizados en el *switch* 3

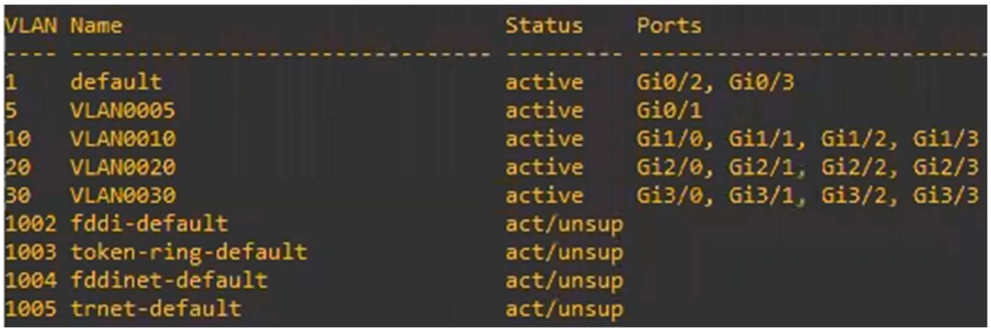


Figura 3.50 VLANs asignadas en el *switch* 3

Comprobación del enrutamiento entre VLANs

Al finalizar con la ejecución de todos los *playbooks*, se abrió las consolas de cada una de las máquinas para empezar con la asignación de la dirección IP mostradas en la Tabla 3.1 . En las máquinas se procede a añadir las direcciones con el comando *ip (dirección IP) 255.255.255.0 (dirección de Gateway)*, en donde la dirección de Gateway será la que se hayan asignado en una de las tres subinterfaces del *router* central, ver Figura 3.51 .

```
PC10> sh ip

NAME        : PC10[1]
IP/MASK     : 0.0.0.0/0
GATEWAY     : 0.0.0.0
DNS         :
MAC         : 00:50:79:66:68:02
LPORT       : 10013
RHOST:PORT  : 127.0.0.1:10014
MTU:        : 1500

PC10> ip 192.168.3.12 255.255.255.0 192.168.3.1
Checking for duplicate address...
PC1 : 192.168.3.12 255.255.255.0 gateway 192.168.3.1

PC10> sh io
Invalid arguments

PC10> sh ip

NAME        : PC10[1]
IP/MASK     : 192.168.3.12/24
GATEWAY     : 192.168.3.1
DNS         :
MAC         : 00:50:79:66:68:02
LPORT       : 10013
RHOST:PORT  : 127.0.0.1:10014
MTU:        : 1500
```

Figura 3.51 Ejemplo de configuración de dispositivos finales

Al finalizar con la configuración de los equipos se procede a hacer un ping entre las máquinas para demostrar que hay conexión. Se escogió una PC de la VLAN 10 y se hizo un ping a las PCs de la misma VLAN, ver Figura 3.52 .

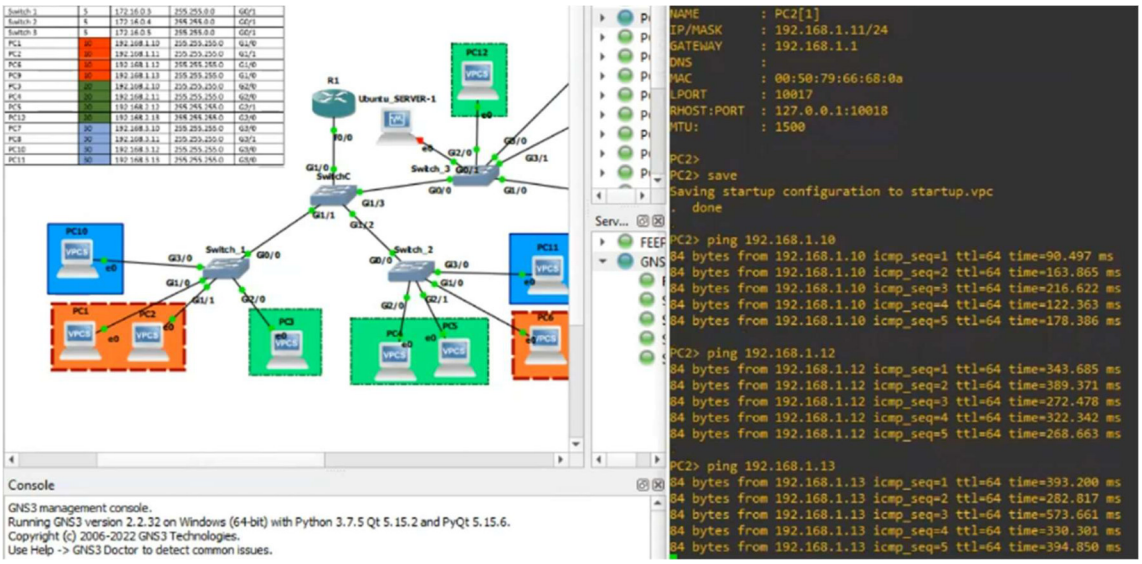


Figura 3.52 Ping entre las PCs de la VLAN 10

Adicionalmente, se realizó un ping desde una PC de la VLAN 10 hacia una PC de la VLAN 30, así como a una de la VLAN 20. Esto se puede evidenciar en la Figura 3.53 .

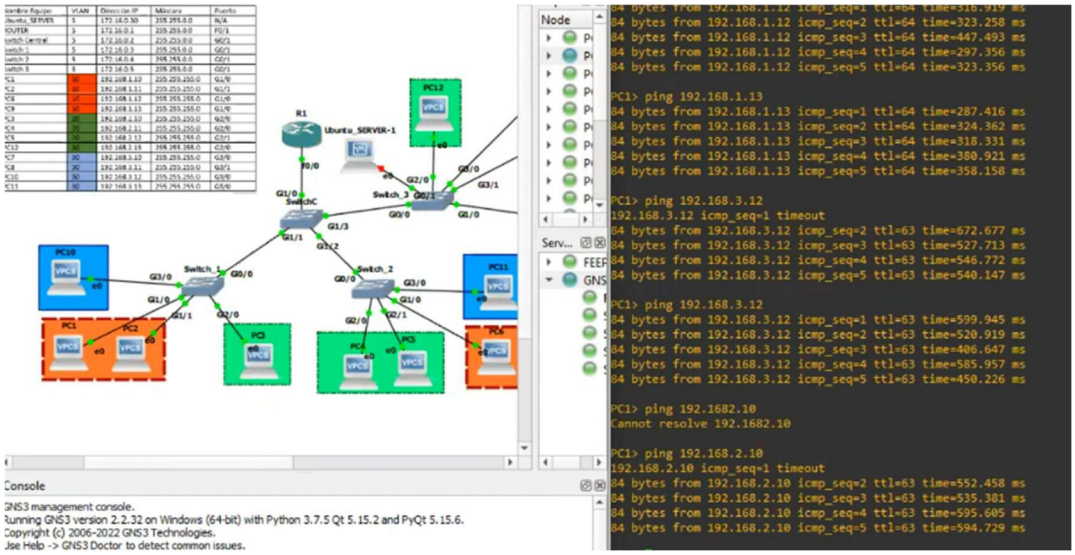


Figura 3.53 Ping entre PC de la VLAN 10 hacia las PCs de la VLAN 20 y VLAN 30

Por otro lado, se realizó, de la misma manera, un ping desde una PC de la VLAN 20 hacia una PC de la VLAN 30, así como a una de la VLAN 10. Adicionalmente, se hizo ping entre las propias máquinas de la misma VLAN. Si no se envían todos los paquetes se vuelve a ejecutar el ping y se enviarán todos los paquetes, ver Figura 3.54 .

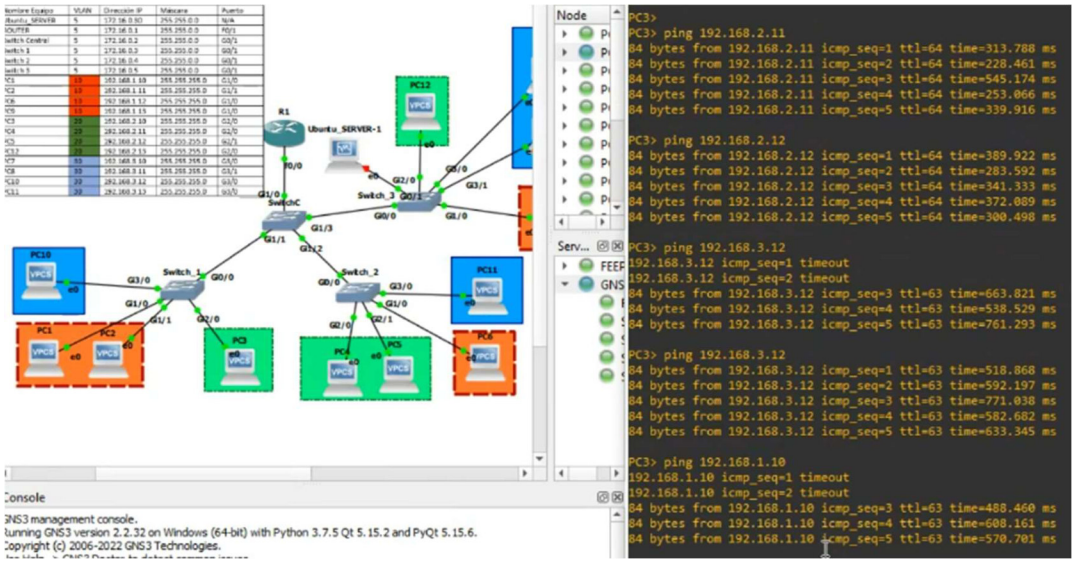


Figura 3.54 Ping entre PC de la VLAN 20 hacia las PCs de la VLAN 10 y VLAN 30

Para finalizar, se realizó los últimos pings entre las PCs de la VLAN 30, observar Figura 3.55 . En la Figura 3.56 se presenta un ping exitoso desde la PC de la VLAN30 hacia otras PCs de las otras VLANs. De esta manera se aprecia que las máquinas pueden establecer comunicación entre ellas y que el enrutamiento entre VLANs mediante

Router-on-a-Stick se ha aplicado. La ejecución de los *playbooks* funcionó y el enrutamiento entre VLANs también.

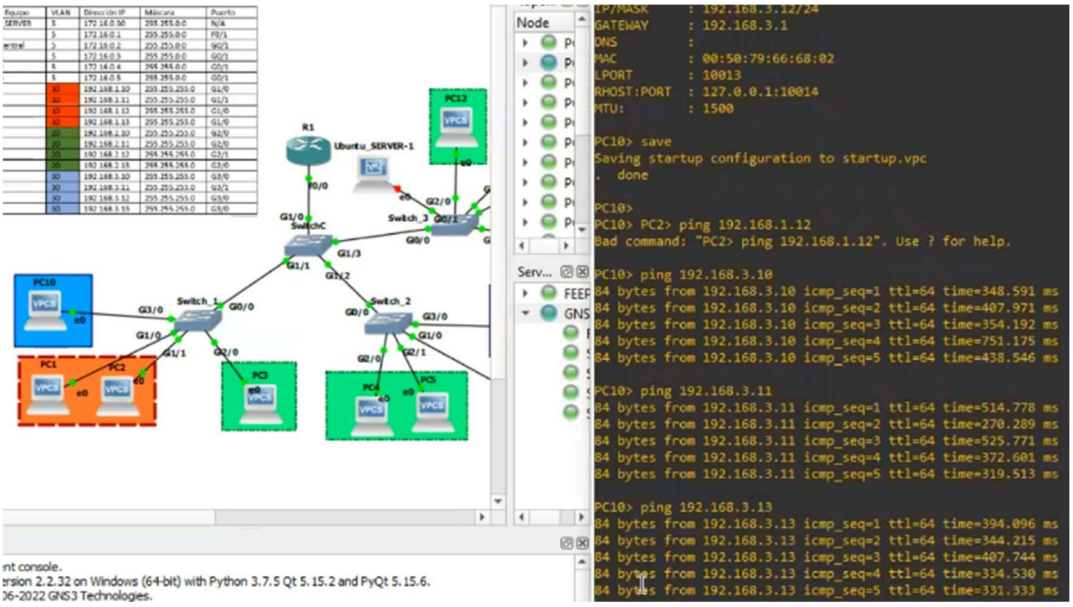


Figura 3.55 Ping entre las PCs de la VLAN 30

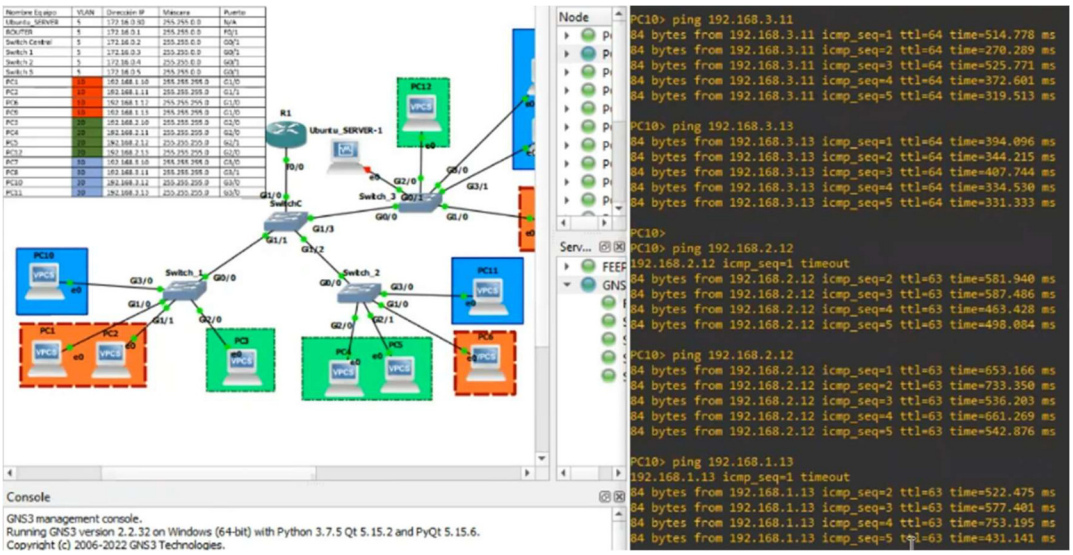


Figura 3.56 Ping entre PC de la VLAN 30 hacia las PCs de la VLAN 10 y VLAN 20

4 CONCLUSIONES

- Para la implementación del proyecto planteado se optó por el sistema operativo servidor de Ubuntu, versión 20.04, ya que se ha trabajo con anterioridad con dicho sistema y resulta conveniente para la instalación de Ansible. Adicionalmente, se usó la versión de este sistema operativo porque al instalar la aplicación cuenta con una versión más actualizada con la que se puede trabajar con los repositorios actualizados, mientras que, en versiones anteriores del sistema operativo mencionado, se instalaba un Ansible más antiguo.
- Ansible cuenta con un archivo denominado *host* en el que se pueden ingresar las credenciales de los equipos, facilitando la comunicación entre la aplicación y los equipos manteniendo una comunicación segura.
- Con Ansible, el trabajo se vuelve automático ya que se puede realizar un solo *playbook* con múltiples tareas. Este *playbook* se puede, con mínimos cambios, implementar a otros equipos de red haciendo que los administradores no empleen mucho tiempo en la configuración; de esta manera se facilitan los procesos de configuración de los equipos de red.
- Ansible-core es una versión más segura para utilizar, ya que al ser instalado cuenta con los repositorios más actualizados, mientras que la versión base de Ansible puede que no cuente con una versión actualizada lo cual sería un inconveniente en la implementación de un *playbook*. Esto también depende de qué versión del sistema operativo será instalado Ansible.
- Para llevar a cabo la implementación de la topología se usó un simulador de red, GNS3, el cual resulta muy útil para simular equipos de red porque estos pueden ser arrancados desde una máquina servidor, evitando el uso excesivo de los recursos de la máquina física. También se puede subir la máquina virtual donde se aloja Ansible que sirvió como nodo controlador y configurar los equipos de red con la aplicación.
- Para una correcta implementación y ejecución de los *playbooks*, se pueden usar distintos módulos que permiten ingresar comandos únicos, en este caso se utilizó *ios_config*, un módulo en el que se pueden usar comandos para configurar equipos de red, como un *router* o *switch,* haciendo que las tareas se ejecuten facilmente. También existen otros módulos con funciones similares, pero se optó por este por su sencillo uso a la hora de ingresar comandos.

- Al correr un *playbook* pueden suceder 3 cosas, la primera es que, al ejecutarlo, todo vaya bien, las tareas se realizan de forma correcta y hará los cambios necesarios en el equipo de red. La otra es que produzca un error sea por espacios o que esté mal digitado una parte del código. El último error se da porque al nodo controlador le cuesta establecer una conexión con los nodos administrados debido a que los equipos puedan consumir todos los recursos de la máquina.
- Al finalizar con las configuraciones, se realizó pruebas de conectividad primero entre los equipos que constituyen una misma VLAN dando respuesta exitosa. Posteriormente, se verificó el enrutamiento entre VLANs lo cual el resultado fue de la misma manera exitoso. Se verifica así la configuración automática desplegada por Ansible de una forma rápida y eficiente.

5 RECOMENDACIONES

- En el *playbook* cuando se configuran varias interfaces de red con los mismos comandos es importante ingresar a las interfaces mediante un rango, ya que si se realiza individualmente puede ser que no se establezca la configuración en uno de estos.
- Debido a que hubo problemas con la versión de Ansible en el sistema operativo de Ubuntu Server 18.04 ya que no se podía usar una versión de Ansible actual (en concreto la 2.9 que permite trabajar con todas las configuraciones requeridas), se optó por un sistema operativo servidor más actual como lo fue Ubuntu 20.04.
- Durante la configuración de un *playbook* es recomendable verificar los espacios que se usan, ya que al ser ejecutado pueden dar un error y el nodo administrado no será configurado.
- Se recomienda verificar la versión de la herramienta Ansible junto al módulo de Python porque Ansible viene conjuntamente con Python para que este pueda trabajar correctamente y dependiendo de la versión puede traer más optimizaciones consigo.
- Es recomendable que, al finalizar con la ejecución de un *playbook*, los equipos se vayan apagando y se ejecute el siguiente ya que puede salir un error de conexión porque los equipos consumen recursos, ya sea de la máquina virtual, del simulador de red o del equipo físico; volviéndolos lentos y que su ejecución sea lenta.

6 REFERENCIAS BIBLIOGRAFÍA

[1] I. Lemus, «Conocimiento Libre,» 5 11 2019. [En línea]. Available: https://conocimientolibre.mx/que-es-ansible/. [Último acceso: 15 5 2022].

[2] R. Hat, «Red Hat,» 2 11 2020. [En línea]. Available: https://www.redhat.com/es/topics/automation/learning-ansible-tutorial. [Último acceso: 15 05 2022].

[3] L. D. CHANNEL, «YouTube,» 21 01 2022. [En línea]. Available: https://www.youtube.com/watch?v=j2t_dU4rDJ0. [Último acceso: 15 05 2022].

[4] Ansible, «Ansible Docs,» [En línea]. Available: https://docs.ansible.com/ansible-tower/2.2.2/html/installandreference/requirements_refguide.html. [Último acceso: 15 05 2022].

[5] TelecTronika, «TelecTronika,» 29 abril 2018. [En línea]. Available: https://www.telectronika.com/articulos/ti/que-es-gns3/. [Último acceso: 16 05 2022].

[6] J. Rejón, «mundotelematico.com,» 14 mayo 2019. [En línea]. Available: https://www.mundotelematico.com/simulador-de-red-gns3/. [Último acceso: 16 05 2022].

[7] Network-tic, «Network-tic,» [En línea]. Available: https://network-tic.com/eve-ng-software-emulacion/#Conociendo-al-personaje-que-es-EVE-NG-y-que-me-puede-aportar. [Último acceso: 16 05 2022].

[8] TekOps, «YouTube,» 10 10 2020. [En línea]. Available: https://www.youtube.com/watch?v=w19Pf9eB22U&t=331s. [Último acceso: 16 05 2022].

[9] César, «Info++,» 31 agosto 2019. [En línea]. Available: https://cesarcabrera.info/que-es-eve-ng-ii-instalacion-paso-a-paso/. [Último acceso: 16 05 2022].

[10] CISCO, «CISCO Networking Academy,» [En línea]. Available: https://www.netacad.com/es/courses/packet-tracer/faq. [Último acceso: 16 05 2022].

[11] T. School, «Tokio,» 13 03 2021. [En línea]. Available: https://www.tokioschool.com/noticias/cisco-packet-tracer/. [Último acceso: 16 05 2022].

[12] ambit, «ambit,» 18 02 2020. [En línea]. Available: https://www.ambit-bst.com/blog/todo-lo-que-debes-saber-de-cisco-packet-tracer. [Último acceso: 16 05 2022].

[13] Yunn, «entrono GNU/Linux,» 08 08 2015. [En línea]. Available: http://entornosgnulinux.com/tag/software-properties-common/#:~:text=Un%20comando%20muy%20habitual%20sin,sus%20llaves%20en%20el%20sistema.. [Último acceso: 18 07 2022].

[14] Ubuntu, «Ubuntu documentation,» [En línea]. Available: https://help.ubuntu.com/stable/ubuntu-help/addremove-ppa.html.es#:~:text=Archivos%20de%20paquetes%20personales%20(PPAs,forma%20que%20puedan%20ser%20probadas.. [Último acceso: 18 07 2022].

7 ANEXOS

ANEXO I: Certificado de Originalidad

CERTIFICADO DE ORIGINALIDAD

Quito, D.M. 29 de agosto de 2022

De mi consideración:

Yo, GABRIELA KATHERINE CEVALLOS SALAZAR, en calidad de Director del Trabajo de Integración Curricular titulado IMPLEMENTAR ANSIBLE PARA REALIZAR ENRUTAMIENTO ENTRE VLANs elaborado por el estudiante DONOVAN STEVE VACA MUÑOZ de la carrera en TECNÓLOGÍA SUPERIOR EN REDES Y TELECOMUNICACIONES, certifico que he empleado la herramienta Turnitin para la revisión de originalidad del documento escrito completo, producto del Trabajo de Integración Curricular indicado.

El documento escrito tiene un índice de similitud del 12%.

Es todo cuanto puedo certificar en honor a la verdad, pudiendo el interesado hacer uso del presente documento para los trámites de titulación.

NOTA: Se adjunta el link del informe generado por la herramienta Turnitin.

https://epnecuador-my.sharepoint.com/:f:/g/personal/gabriela_cevalloss_epn_edu_ec/En5_eJ4LSxROl_RQ3bO3cxMBUhobj3PpXxC-bWzXAMBPvw?e=UNUIXV

Atentamente,

Gabriela Cevallos

Docente

Escuela de Formación de Tecnólogos

ANEXO II: Enlaces

https://www.youtube.com/watch?v=v7GAMzF2vbc

Anexo II.I Código QR de la implementación y pruebas de funcionamiento

ANEXO III: CODIGOS

Playbook router central (RC)

```
---
- name: Configuración de Router
  hosts: router


  tasks:
    - name: Levantar puerto
      ios_config:
        lines:
         - no shutdown
        parents: interface FastEthernet0/0


    - name: Configuración de Sub-interfaces
      ios_config:
        lines:
         - interface FastEthernet0/0.10
         - interface FastEthernet0/0.20
         - interface FastEthernet0/0.30


    - name: Asignar dirección IP a sub-interfaz 10
      ios_config:
        lines:
         - encapsulation dot1Q 10
         - ip address 192.168.1.1 255.255.255.0
        parents: interface FastEthernet0/0.10
```

```
  - name: Asignar dirección IP a sub-interfaz 20
    ios_config:
      lines:
       - encapsulation dot1Q 20
       - ip address 192.168.2.1 255.255.255.0
      parents: interface FastEthernet0/0.20

  - name: Asignar dirección IP a sub-interfaz 30
    ios_config:
      lines:
       - encapsulation dot1Q 30
       - ip address 192.168.3.1 255.255.255.0
       - do wr
      parents: interface FastEthernet0/0.30
```

***Playbook switch* central (SWC)**

```
---
- name: Configuración Switch Central
  hosts: switchc

  tasks:
    - name: Establecer VLANs
      ios_config:
        lines:
         - vlan 10
         - vlan 20
         - vlan 30
```

```
  - name: Establecer nodo troncal en las interfaces
    ios_config:
      lines:
       - switchport trunk encapsulation dot1Q
       - switchport mode trunk
       - switchport trunk allowed vlan 10,20,30
       - do wr
      parents: interface range GigabitEthernet1/0-3
```

***Playbook switch* 1 (SW1)**

```
---
- name: Configuración Switch 1
  hosts: switch1

  tasks:
    - name: Establecer VLANs
      ios_config:
        lines:
         - vlan 10
         - vlan 20
         - vlan 30

    - name: Asignar vlan 10
      ios_config:
        lines:
         - switchport mode access
         - switchport access vlan 10
```

```
      parents: interface range GigabitEthernet1/0-3

  - name: Asignar vlan 20
    ios_config:
      lines:
       - switchport mode access
       - switchport access vlan 20
      parents: interface range GigabitEthernet2/0-3

  - name: Asignar vlan 30
    ios_config:
      lines:
       - switchport mode access
       - switchport access vlan 30
      parents: interface range GigabitEthernet3/0-3

  - name: Asignar puerto troncal
    ios_config:
      lines:
       - switchport trunk encapsulation dot1Q
       - switchport mode trunk
       - switchport trunk allowed vlan 10,20,30
       - do wr
      parents: interface GigabitEthernet0/0
```

***Playbook switch* 2 (SW2)**

```
---
- name: Configuración Switch 2
```

```
hosts: switch2

tasks:
  - name: Establecer VLANs
    ios_config:
      lines:
       - vlan 10
       - vlan 20
       - vlan 30

  - name: Asignar vlan 10
    ios_config:
      lines:
       - switchport mode access
       - switchport access vlan 10
      parents: interface range GigabitEthernet1/0-3

  - name: Asignar vlan 20
    ios_config:
      lines:
       - switchport mode access
       - switchport access vlan 20
      parents: interface range GigabitEthernet2/0-3

  - name: Asignar vlan 30
    ios_config:
      lines:
```

```
        - switchport mode access
        - switchport access vlan 30
        parents: interface range GigabitEthernet3/0-3

    - name: Asignar puerto troncal
      ios_config:
        lines:
        - switchport trunk encapsulation dot1Q
        - switchport mode trunk
        - switchport trunk allowed vlan 10,20,30
        - do wr
        parents: interface GigabitEthernet0/0
```

***Playbook switch* 3 (SW3)**

```
---
- name: Configuración Switch 3
  hosts: switch3

  tasks:
    - name: Establecer VLANs
      ios_config:
        lines:
        - vlan 10
        - vlan 20
        - vlan 30

    - name: Asignar vlan 10
      ios_config:
```

```
    lines:
     - switchport mode access
     - switchport access vlan 10
    parents: interface range GigabitEthernet1/0-3

- name: Asignar vlan 20
  ios_config:
    lines:
     - switchport mode access
     - switchport access vlan 20
    parents: interface range GigabitEthernet2/0-3

- name: Asignar vlan 30
  ios_config:
    lines:
     - switchport mode access
     - switchport access vlan 30
    parents: interface range GigabitEthernet3/0-3

- name: Asignar puerto troncal
  ios_config:
    lines:
     - switchport trunk encapsulation dot1Q
     - switchport mode trunk
     - switchport trunk allowed vlan 10,20,30
     - do wr
    parents: interface GigabitEthernet0/0
```