ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

IMPLEMENTACIÓN DE SERVICIOS DE NETWORKING MEDIANTE DEVOPS

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

BYRON PATRICIO ENCALADA MONTUFAR

DIRECTOR: Ing. FERNANDO BECERRA, Msc.

DMQ, febrero 2024

CERTIFICACIONES

Yo, BYRON PATRICIO ENCALADA MONTÚFAR 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.

BYRON PATRICIO ENCALADA MONTUFAR

byron.encalada@epn.edu.ec

byron094@hotmail.com

Certifico que el presente trabajo de integración curricular fue desarrollado por BYRON PATRICIO ENCALADA MONTUFAR, bajo mi supervisión.

FERNANDO BECERRA
DIRECTOR

fernando.becerrac@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.

BYRON PATRICIO ENCALADA MONTUFAR

DEDICATORIA

Con profundo agradecimiento y un cálido amor, dedico este proyecto de titulación a mi querida familia: mis padres, hermana y primo, quienes han sido mi sólido pilar a lo largo de esta travesía. Además, mi hija Luciana ha sido la chispa que avivó mi motivación para alcanzar la meta de convertirme en un profesional.

Byron Patricio

AGRADECIMIENTOS

Quiero expresar desde lo más profundo de mi corazón mi agradecimiento sincero a mi tutor de tesis. Él ha sido una fuente constante de inspiración para mí, destacándose como uno de los maestros más excepcionales en mi carrera y sirviendo como un modelo ejemplar de lo que significa ser un verdadero profesional. A mis queridos compañeros de clase y amigos, les agradezco de todo corazón por su apoyo inquebrantable durante los momentos más desafiantes. Su responsabilidad y dedicación han sido un faro luminoso, guiándonos hacia la construcción de un futuro lleno de promesas y éxitos.

Byron Patricio

ÍNDICE DE CONTENIDOS

RESUMEN

El presente proyecto, IMPLEMENTACIÓN DE SERVICIOS DE *NETWORKING* MEDIANTE *DEVOPS*, permite la configuración remota de *OSPF*. La automatización mediante *Ansible* facilita la aplicación de configuraciones en equipos de *networking* y reducción de errores.

En la primera sección se describe el proyecto, donde detalla las características de la herramienta de automatización *Ansible*, los requerimientos para su configuración y adicionales los equipos utilizados en el proyecto. Se definen los objetivos, generales y específicos y el alcance del proyecto.

En la sección dos se profundiza en la metodología empleada para la obtención de cada objetivo empezando en la creación de máquinas virtuales, la configuración en *GNS3*, y en la imagen de *Ubuntu server*, que contiene la herramienta *Ansible* instalada y se ejecuta la operación remota para configurar los equipos en la topología.

En la sección tres se explica detalladamente las configuraciones hechas en los equipos para lograr el objetivo general en base a los objetivos específicos, al final se tiene los resultados obtenidos después de las pruebas de funcionamiento para permitir la configuración de enrutamientos *OSPF* en los equipos de *networking*.

En la cuarta y quinta sección se tienen las conclusiones y recomendaciones obtenidas tras el proyecto, para concentrar los aspectos relevantes del proyecto, también cómo valer de guía en proyectos de investigación relacionados con este tema y/o el área de estudios.

PALABRAS CLAVE: *Ansible*, *Playbook*, *OSPF*, *Linux*, Automatización, *DevOps*.

ABSTRACT

The present project, IMPLEMENTATION OF NETWORKING SERVICES THROUGH DEVOPS, enables the remote configuration of OSPF. Automation through Ansible facilitates the application of configurations on networking equipment and reduces errors.

In the first section, the project is described, detailing the features of the Ansible automation tool, the requirements for its configuration, and additional information about the equipment used in the project. The project's general and specific objectives are defined, along with the project's scope.

Section two delves into the methodology used to achieve each objective, starting with the creation of virtual machines, configuration in GNS3, and the Ubuntu server image containing the installed Ansible tool. Remote operations are executed to configure the equipment in the topology.

Section three provides a detailed explanation of the configurations made on the equipment to achieve the general objective based on the specific objectives. Finally, the obtained results after the operational tests are presented to enable OSPF routing configuration on the networking equipment.

In the fourth and fifth sections, the conclusions and recommendations obtained from the project are outlined, focusing on the relevant aspects of the project. Additionally, it serves as a guide for research projects related to this topic and/or the study area

KEYWORDS: *Ansible, Playbook, OSPF, Linux, Automatization, DevOps.*

1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO

El objetivo de este proyecto es automatizar la configuración de protocolos de enrutamiento usando la herramienta de *DevOps*, *Ansible*. *Ansible* posibilita la automatización de tareas administrativas repetitivas en equipos de red, servidores o almacenamiento de forma remota. Esta metodología garantiza una implementación eficiente, rápida y confiable, al mismo tiempo que previene posibles errores humanos.

Los dispositivos utilizados en la ejecución de este proyecto comprenden un conjunto computacional donde implementaremos la virtualización del simulador gráfico de red *GNS3*. En este entorno virtual, llevaremos a cabo la instalación de imágenes de equipos de *networking*, específicamente de la marca *CISCO* (*cisco.ios*). Además, incorporaremos una imagen virtualizada de un servidor con *UBUNTU SERVER*, que funcionará como gestor de *Ansible* dirigido a los *routers*. Dichos *routers* serán configurados utilizando el protocolo *OSPF*.

Este servidor local, actuando como la base central de configuración entre los equipos de *networking*, posibilitará la implementación de enrutamientos basados en el protocolo *OSPF*. El servidor central *Ansible* instalado generará archivos en formato *YAML* de acuerdo con las necesidades específicas de las diversas configuraciones requeridas por la topología para los equipos de *networking*.

Los archivos que *Ansible* utilizará, conocidos como *playbooks*, están redactados en *YAML*, como mencionamos anteriormente. Esta elección de lenguaje proporciona una estructura clara y facilita su comprensión. Estos *playbooks* esencialmente contienen las instrucciones detalladas para la configuración de los protocolos *OSPF*. En términos prácticos, son estos archivos los que delinean las redes que deben establecer adyacencia entre los diversos equipos de *networking*.

Durante el proyecto se creó el archivo *playbook* de *Ansible* para la configuración de *host*, cada variable y tarea a ejecutarse en los equipos. Este *host* se relaciona a un archivo de inventario donde se detalla cada equipo que se configurará y las credenciales de dichos equipos. Las variables permitirán listar cada campo a configurar en los equipos *router*.

Ansible se emplea para administrar configuraciones, permitiendo una gestión centralizada de los enrutamientos. Esto simplifica la gestión y el mantenimiento de la red, ya que cualquier cambio en la configuración de un enrutamiento se realiza de manera rápida durante la ejecución del *playbook*.

1.1 Objetivo general

Implementación de servicios de *networking* mediante *DevOps*.

1.2 Objetivos específicos

- Analizar *DevOps* la metodología *DevOps* puede abordar y mejorar cada uno de los servicios de *networking*.
- Diseñar soluciones específicas para cada servicio de *networking*, aprovechando herramientas y prácticas de *DevOps*.
- Implementar las soluciones mediante *DevOps* para el despliegue de los servicios de *networking*.
- Realizar pruebas de cada uno de los servicios de *networking* implementados bajo el enfoque de *DevOps*.

1.3 Alcance

El presente proyecto pretende que los estudiantes utilicen las prácticas de *DevOps* y en la implementación de soluciones tanto en entornos de *host* como en infraestructuras de *networking*. Para alcanzar esta meta, se requiere de un conjunto de herramientas de *DevOps* capaces de automatizar de manera efectiva la implementación de los servicios propuestos, además de facilitar la realización de pruebas exhaustivas para garantizar la calidad de las soluciones. En este proyecto, se tiene previsto la implementación de tres componentes:

• Implementación de una red virtualizada para obtener los diferentes tipos de *SLA* de *OSPF* con *ansible*.

1.4 Marco Teórico

DevOps

DevOps se define como una cultura y conjunto de prácticas que tiene como objetivo integrar de manera colaborativa el desarrollo (*Dev*) y las operaciones (*Ops*) en el ciclo de vida del *software*. La expresión "*DevOps*" surge de la combinación de las palabras "*development*" y "*operations*", resaltando la importancia de la colaboración entre los equipos de desarrollo de *software* y los profesionales de operaciones de tecnologías de la información.

La meta primordial de *DevOps* es acelerar la entrega de *software* y mejorar su calidad mediante la automatización de procesos, la promoción de la colaboración y el reparto de responsabilidades entre los equipos de desarrollo y operaciones. Esto implica la adopción de prácticas ágiles, la automatización de tareas, la utilización de herramientas de integración y entrega continuas (*CI/CD*), junto con la asunción de una mentalidad de mejora continua.

En resumen, *DevOps* aspira a superar las barreras tradicionales entre desarrollo y operaciones, fomentando una colaboración más estrecha y promoviendo la entrega eficiente y rápida de *software* de alta calidad.

OSPF

El protocolo *Open Shortest Path First (OSPF)* el cual es definido en el *RFC* 2328 es un *internal Gateway protocol*, el cual utiliza la información de las rutas en un solo sistema autónomo. *OSPF* aprende, anuncia información de los enrutamientos de las subredes *IP* de los equipos vecinos, el cual calcula la mejor ruta por la métrica. En algún momento si la topología de red cambia anuncia que las rutas han fallado y elije la nueva mejor ruta, a esto se denomina convergencia. [1] [2] [3] [4]

LSA

Los *LSA* son los paquetes que el protocolo *OSPF* utiliza en el intercambio de información sobre los estados del enlace. Hay varios *LSA* en el cual cada uno transmite información específica sobre la topología que se presenta en la red. [2] [5] [6] [7] [8] [9]

Los tipos de *LSAs* de *OSPF* son los siguientes:

- *LSA* tipo 1: contiene toda la información de los estados del enlace de cada *router*.
- *LSA* tipo 2: este *LSA* proporciona la información sobre los equipos *routers* conectados en la red de multiacceso.
- *LSA* tipo 3: este resume la información fuera de redes de un área *OSPF* y lo redistribuye a través de las áreas.
- *LSA* tipo 4: este informa a los demás *routers* en un área sobre la existencia de *routers* de frontera entre sistemas autónomos.
- *LSA* tipo 5: este es generado por un *ASBR* que describe rutas externas al sistema autónomo de *OSPF.*
- *LSA* tipo 6: Es un multicast *OSPF*, reservado para un uso futuro.
- *LSA* tipo 7: es utilizado en áreas de tipo *Not-So-Stubby (NSSA)* es similar al área tipo 5, pero adaptado a las áreas *NSSA*.

GNS3

GNS3 (Graphical Network Simulator-3) se presenta como un entorno de simulación de redes de código abierto que capacita a los usuarios para emular entornos informáticos. Este *software* brinda una plataforma gráfica que posibilita el diseño, la configuración y la evaluación de redes virtuales en un entorno seguro y controlado. [10] [11]

Entre las características destacadas de *GNS3* se encuentran:

- Emulación de Dispositivos de Red: *GNS3* permite emular una variedad de dispositivos de red, como *routers*, *switches* y *firewalls*. Los usuarios tienen la flexibilidad de incorporar imágenes de *software* provenientes de distintos fabricantes para simular configuraciones de equipos reales.
- Interfaz Gráfica de Usuario (*GUI*): Proporciona una interfaz gráfica intuitiva que facilita a los usuarios el proceso de arrastrar y soltar dispositivos en un lienzo, así como la interconexión y configuración de parámetros.
- Soporte para Imágenes de *IOS* y Otros Sistemas Operativos: *GNS3* es compatible con imágenes de sistemas operativos de red, como *Cisco IOS (Internetwork Operating System),* permitiendo a los usuarios emular equipos *Cisco* dentro de sus topologías.
- Integración con *Dynamips* y *QEMU*: *GNS3* utiliza motores de emulación como *Dynamips* y *QEMU* (*Quick Emulator*) para ejecutar imágenes de *router* y sistemas operativos en máquinas virtuales.
- Simulación de Conectividad de Red: Facilita a los usuarios la simulación y prueba de la conectividad de red antes de implementar configuraciones en entornos de producción. Esta funcionalidad resulta especialmente valiosa para estudiantes, ingenieros de red y profesionales de *TI* que desean experimentar con configuraciones sin afectar redes reales.
- Apoyo a la Comunidad: *GNS3* cuenta con una comunidad activa de usuarios que comparten topologías, configuraciones y experiencias, fomentando el aprendizaje colaborativo.

En resumen, *GNS3* emerge como una herramienta valiosa tanto para estudiantes que buscan practicar configuraciones de red como para profesionales de redes que necesitan evaluar cambios antes de implementarlos en entornos de producción.

2 METODOLOGÍA

De acuerdo con los requerimientos del presente proyecto, la metodología empleada en su desarrollo es de tipo exploratoria, por las investigaciones de la herramienta de *DevOps Ansible*. Adicionalmente se empleó la metodología aplicada, al aplicar la implementación de configuraciones *OSPF* en los *routers* de la topología que muestra la Figura 2.1 a continuación:

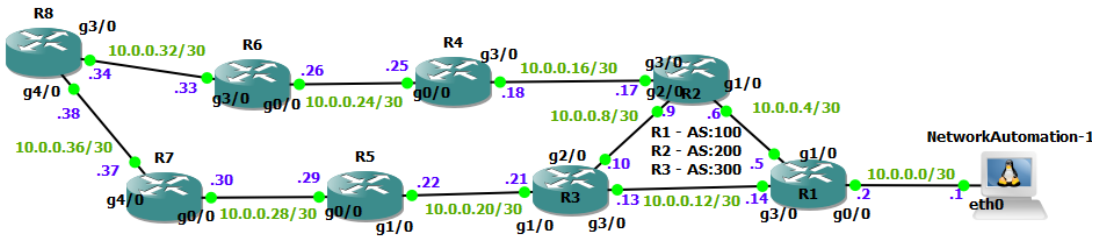


Figura 2.1 Topología de red

2.1 Objetivo 1

Analizar la herramienta de *DevOps* que soluciona cada componente del proyecto de titulación. En primera instancia se realizará una investigación sobre el manejo de *DevOps, SLA* y *OSPF* estas herramientas se procederá a desarrollar el trabajo de titulación.

2.2 Objetivo 2

Diseñar la solución para cada servicio de *networking* mediante herramientas de *DevOps*.

Una vez que se tiene claro el panorama de como implementar *DevOps* se empezará a elaborar el código de la aplicación necesario para poder implementar mediante *ansible* los *SLA* de *OSPF*.

2.3 Objetivo 3

Implementar las soluciones mediante herramientas de *DevOps* para el despliegue de los servicios de *networking*. Se implementará el *playbook* que contiene los *LSA* y *OSPF*.

2.4 Objetivo 4

Verificar el funcionamiento de cada servicio de *networking* implementado mediante *DevOps*. Finalmente se realizará pruebas del contenedor y se verificará su buen funcionamiento.

3 RESULTADOS

A continuación, se describe el procedimiento para desplegar la solución de implementación de servicios de *networking* a través de *DevOps*. En primera instancia, se procederá con la instalación del sistema de simulación de entorno, en el cual configuraremos los equipos de red, específicamente los *routers*.

3.1 Instalar la herramienta de *Ansible* en el servidor

La instalación de la herramienta *Ansible* se realiza una vez se tenga la imagen del *Server Ubuntu*, y se debe tener en cuenta que la versión de *ansible* sea compatible con la imagen *QEMU* del *GNS3* con la cual se realizará la implementación de las configuraciones respectivas, para la instalación son los mostrados en la Figura 3.1 siguiente:

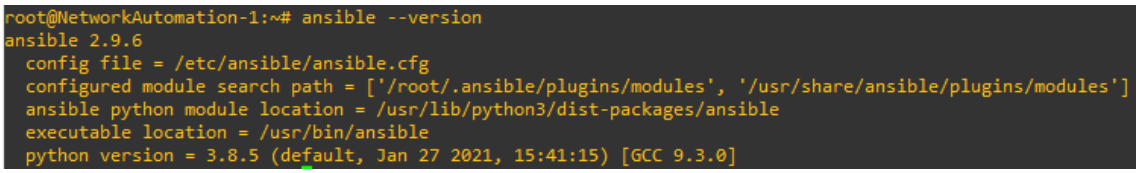


Figura 3.1 Implementación *ansible*

Después de completar la instalación de la herramienta, se generarán archivos en el directorio */etc/ansible*. En este mismo directorio, será necesario crear los archivos correspondientes al proyecto, los cuales serán detallados en secciones posteriores.

Ansible

Ansible es una plataforma de gestión de configuración la cual nos permite automatizar las redes, servidores y almacenamientos.

Ansible es un *software* en código abierto el cual automatiza la administración de la configuración y la ejecución en la automatización de servicios en gestión de los equipos en redes, seguridad y almacenamiento. Esto lo gestiona a través del protocolo *SSH* y uno de los requerimientos es que el *host* deba tener instalado *Python* para que pueda ejecutar y se pueda utilizar. Utiliza archivos en lenguaje *YAML* para detallar las tareas a ejecutar y las disposiciones a los diferentes equipos. [12] [13] [14] [15]

Ansible trabaja realizando una conexión con los equipos e incrustando pequeños programas llamados módulos, los cuales sirven para realizar las indicaciones de los módulos a través del protocolo *SSH*. Una vez realizada culminada la implementación de instrucciones. [12] [13] [16] [17] [18]

Inventario

En *Ansible* se debe realizar un archivo el cual va a ser nuestro contenedor de información sobre los equipos dentro de la topología, este archivo gestiona los equipos routers dentro de la red que forman la infraestructura donde se realizará las operaciones de *Ansible*. Es en este archivo de *Ansible* es donde se determina el equipo y la ejecución de las tareas a realizar. [13] [12] [19]

Playbook

Playbook es la configuración o las tareas que se va a ejecutar en los equipos *router* de manera remota y desde el inventario de *Ansible*. El *playbook* para las configuraciones de los *routers* es donde se realiza las configuraciones de enrutamiento de *OSPF*. Las implementaciones se ejecutan en secuencia, describiendo el estado del sistema en un archivo *YAML*. [12] [13]

YAML

YAML es un formato de serialización de datos que sea de fácil escritura y legible por el usuario. Es utilizado para realizar las configuraciones, archivos de datos y también puede ser utilizado con otros tipos de lenguajes de programación, también se puede crear estructuras más complejas que en una línea de comandos. *Yaml* una de sus principales funcionalidades es la flexibilidad y accesibilidad a las personas. [13] [20]

YAML utiliza las siguientes etiquetas:

- Nombre: nombre de la tarea.
- *Host*: lista o grupo de dispositivos a los que se va a implementar o configurar.
- *Vars*: las variables a emplearse.
- *Task*: las tareas a ejecutarse.

3.2 Diseño de los servicios de *networking* con *ansible*.

En el sistema operativo *Ubuntu Server* del servidor debemos revisar la versión del *ansible* que ya tiene por defecto si lo hubiera, y verificar en sus características si esta versión es compatible con el *QEMU* instalado de imagen de los *routers* en nuestro sistema *GNS3*. El comando para ejecutar es ***ansible –version*** como se muestra en la Figura 3.2 siguiente:

```
root@NetworkAutomation-1:~# ansible --version
ansible 2.9.6
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  executable location = /usr/bin/ansible
  python version = 3.8.5 (default, Jan 27 2021, 15:41:15) [GCC 9.3.0]
```

Figura 3.2 Versión de *ansible*

Una vez verificado el funcionamiento y compatibilidad de versiones, continuamos con la configuración y conexión *SSH* en nuestro *Ubuntu Server* hacia los equipos.

En el sistema *GNS3* se realizó las topologías para implementar en el proyecto de titulación el cual tendrá el protocolo de enrutamiento de estado del enlace *OSPF* como se muestra en la Figura 3.3.

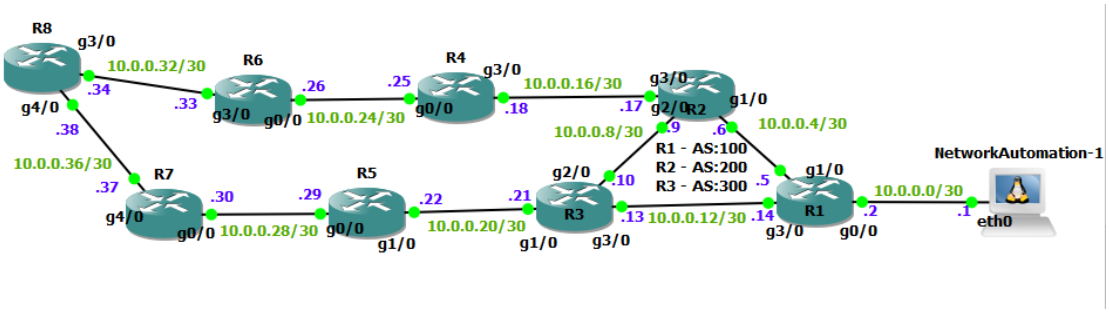


Figura 3.3 Topología de red

Después de completar la construcción de la topología de red, procedemos a llevar a cabo la configuración de las interfaces de los *routers*. En este proceso, implementamos una configuración estática de enrutamiento para garantizar la conectividad completa entre los equipos. Además de esto, se establecen el nombre del dispositivo (*hostname*), el dominio al que pertenece y las credenciales de acceso para cada *router*. También se configuran sesiones *SSH* entre los equipos, utilizando el servidor *Ubuntu* de *Ansible* como intermediario.

Este conjunto de acciones contribuye a establecer un entorno de red funcional y seguro para el intercambio de datos entre los dispositivos como se muestra en las Figura 3.4 y Figura 3.5.

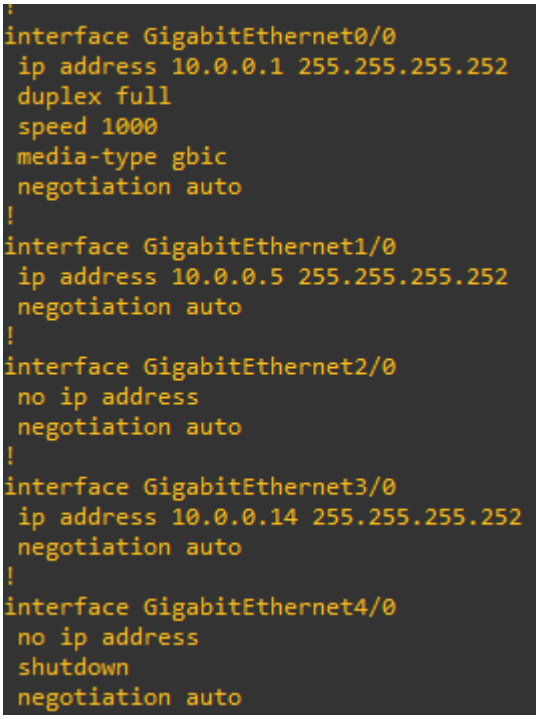


Figura 3.4 Configuración interfaces R1

Estos comandos configuran las interfaces *GigabitEthernet* de un dispositivo de red. Aquí está el detalle de cada comando:

- *interface GigabitEthernet0/0*: Indica que se está configurando la interfaz *GigabitEthernet0/0*.
- *ip address* 10.0.0.1 255.255.255.252: Asigna la dirección *IP* 10.0.0.1 a la interfaz con una máscara de subred de 255.255.255.252.
- *duplex full*: Configura la interfaz para funcionar en modo dúplex completo, lo que permite la transmisión y recepción simultánea de datos.
- *speed* 1000: Configura la velocidad de la interfaz a 1000 *Mbps* (1 *Gbps*).
- *media-type gbic*: Especifica el tipo de módulo *GBIC (Gigabit Interface Converter)* que se está utilizando en la interfaz.
- *negotiation auto*: Habilita la negociación automática para la configuración de velocidad y *dúplex*.
- !: Este símbolo indica un comentario, y en este caso, se utiliza para indicar el final de la configuración de la interfaz *GigabitEthernet0/0*.

Se repiten comandos similares para otras interfaces:

- *interface GigabitEthernet1/0*: Configura la interfaz con la dirección *IP* 10.0.0.5 y la máscara de subred correspondiente. Se habilita la negociación automática.

- *interface GigabitEthernet2/0*: No se asigna una dirección *IP* a esta interfaz, pero se habilita la negociación automática.
- *interface GigabitEthernet3/0*: Configura la interfaz con la dirección *IP* 10.0.0.14 y la máscara de subred correspondiente. Se habilita la negociación automática.
- *interface GigabitEthernet4/0*: No se asigna una dirección *IP* a esta interfaz, pero se apaga la interfaz (*shutdown*) y se habilita la negociación automática.

En resumen, estos comandos están configurando varias interfaces *GigabitEthernet* en un dispositivo de red, asignando direcciones *IP*, configurando velocidades, *dúplex* y negociación automática según sea necesario. En la Figura 3.5 se puede observar lo antes mencionado.

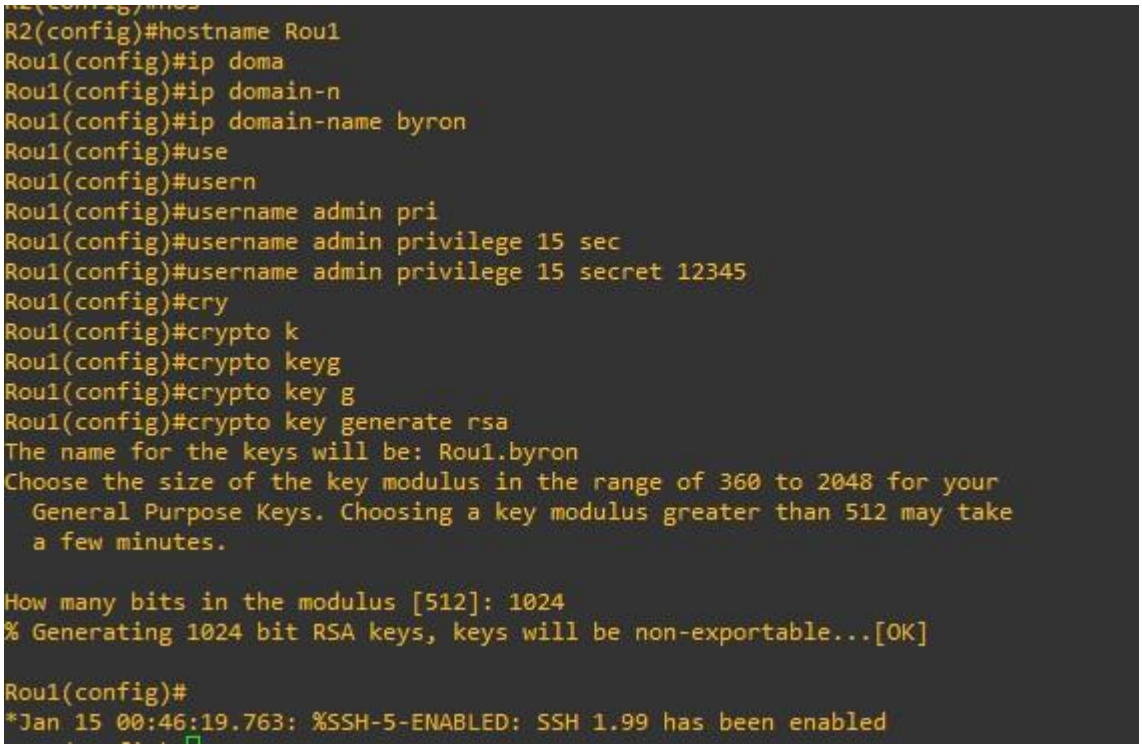


Figura 3.5 Configuración de protocolo *SSH* versión 2

Este conjunto de comandos está configurando la seguridad en un dispositivo de red *Cisco*. Aquí está el detalle de cada comando:

- Roul(config)#*ip domain-name* byron: Establece el nombre de dominio del dispositivo como "byron". Esto es importante para la generación de claves criptográficas.
- *Roul*(*config*)#*username admin privilege* 15 *secret* 12345: Crea un usuario llamado "*admin*" con privilegios máximos (nivel 15) y asigna una contraseña cifrada ("12345") a ese usuario.

- *Roul*(*config*)#*crypto key generate rsa*: Inicia el proceso de generación de claves *RSA* para su uso en servicios criptográficos, como *SSH*.
- *The name for the keys will be*: Rou1.byron: Asigna un nombre a las claves generadas. En este caso, las claves se nombrarán "*Rou1.byron*".
- *Choose the size of the key modulus in the range of 360 to 2048 for your General Purpose Keys*.: Solicita al usuario que elija el tamaño del módulo de clave *RSA* en el rango de 360 a 2048 *bits*. En este caso, se selecciona un tamaño de 1024 *bits*.
- *Generating 1024 bit RSA keys, keys will be non-exportable...*[*OK*]: Genera claves *RSA* de 1024 *bits*. Las claves generadas serán no exportables y se informa que el proceso ha sido exitoso.

En resumen, estos comandos están configurando la seguridad en el dispositivo, estableciendo el nombre de dominio, creando un usuario con privilegios máximos y una contraseña cifrada, y generando claves *RSA* para mejorar la seguridad en servicios criptográficos como *SSH*.

Al general las claves criptográficas *RSA* para que pueda emitir una emisión de certificados en la conexión y se habilita la versión 2 del protocolo *SSH* para el acceso de terminal virtual (*VTY*) como se muestra en la Figura 3.6.

```
Rou1(config)#ip ssh version 2
Rou1(config)#en
Rou1(config)#ena
Rou1(config)#enable se
Rou1(config)#enable secret 12345
Rou1(config)#li
Rou1(config)#lin
Rou1(config)#line vt
Rou1(config)#line vty 0 4
Rou1(config-line)#tra
Rou1(config-line)#transport ss
Rou1(config-line)#transport in
Rou1(config-line)#transport input ss
Rou1(config-line)#transport input ssh
Rou1(config-line)#lo
Rou1(config-line)#log
Rou1(config-line)#logi
Rou1(config-line)#login lo
Rou1(config-line)#login local
Rou1(config-line)#do wr
Building configuration...
[OK]
Rou1(config-line)#
```

Figura 3.6 Configuración protocolo *SSH*

El código es una secuencia de comandos de configuración para un dispositivo de red *Cisco* en modo de configuración. A continuación, se detalla lo que cada línea del código hará:

- *Roul(config)#ip ssh version* 2: Habilita la versión 2 del protocolo *SSH* en el dispositivo. *SSH* (*Secure Shell*) es un protocolo de red que proporciona una forma segura de acceder y gestionar dispositivos remotos.
- *Roul(config)#enable*, este comando permite acceder al modo de configuración de privilegios elevados.
- *Roul*(*config*)#*enable secret* 12345: Configura una contraseña cifrada para el modo privilegiado (modo *enable*). En este caso, la contraseña es "12345". El comando *enable secret* proporciona un nivel de seguridad más alto que el comando *enable password*.
- Roul(config)#*line vty* 0 4: Selecciona las líneas de consola virtual (*virtual terminal lines*) de 0 a 4 para su configuración. Estas líneas son utilizadas para acceder al dispositivo a través de *SSH*.
- Roul(config-line)#*transport input ssh*: Configura las líneas virtuales para aceptar conexiones *SSH*. Esto significa que los usuarios podrán conectarse al dispositivo a través del protocolo *SSH* en las líneas virtuales seleccionadas.
- *Roul*(*config-line*)#*login local*: Habilita la autenticación local para las conexiones *SSH*. Los usuarios deberán ingresar sus credenciales locales (nombre de usuario y contraseña configurados en el dispositivo) al conectarse a través de *SSH*.
- Roul(*config-line*)#*do wr*: Este comando ejecuta el equivalente a *write memory* o *copy running-config startup-config*, que guarda la configuración actual en la memoria persistente del dispositivo.

En resumen, este conjunto de comandos configura un dispositivo *Cisco* para utilizar la versión 2 del protocolo *SSH*, establece una contraseña cifrada para el modo privilegiado, configura líneas virtuales para aceptar conexiones *SSH*, habilita la autenticación local para *SSH* y guarda la configuración.

Se realizó la configuración del servidor *Ubuntu* con *ansible* en el cual ingresamos a la configuración de la interfaz con el comando ***nano /etc/network/interfaces*** y en este archivo editamos la *IP* estática que necesitamos para el proyecto como se muestra en la Figura 3.7.

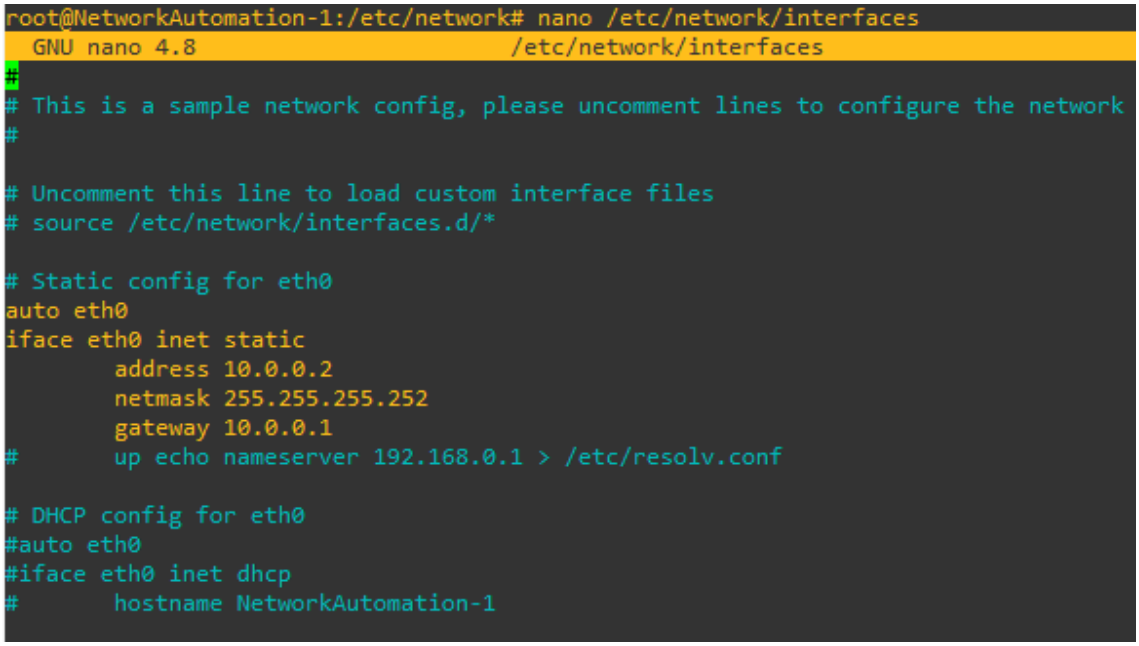


Figura 3.7 Configuración interfaz *Ubuntu server*

Este fragmento de código es un archivo de configuración de red para sistemas basados en *Debian*, utilizando el sistema de administración de red "*interfaces*". Aquí está el código detallado:

- *auto ethe*: Indica que la interfaz "*ethe*" se configurará automáticamente al arrancar.
- *iface ethe inet static*: Establece la configuración de red de la interfaz "*ethe*" como estática.
- *address* 10.0.0.2: Asigna la dirección *IP* 10.0.0.2 a la interfaz "*ethe*".
- *netmask* 255.255.255.252: Configura la máscara de subred de la interfaz "*ethe*".
- *gateway* 10.0.0.1: Especifica la puerta de enlace para la interfaz "*ethe*".
- *up echo nameserver* 192.168.0.1 > /etc/resolv.conf: Configura el servidor de nombres *DNS* en /etc/resolv.conf utilizando el comando echo.
- *auto ethe*: Indica que la interfaz "*ethe*" no se configurará automáticamente al arrancar el *DHCP*.
- *#iface ethe inet dhcp*: Esta línea está comentada, lo que significa que está desactivada. Si deseas utilizar *DHCP* en lugar de una configuración estática, debes descomentar esta línea y comentar las líneas de configuración estática.

En resumen, este código proporciona una configuración de red de muestra, permitiendo la configuración estática o *DHCP* para la interfaz "*ethe*". Se debe descomentar las líneas correspondientes según la preferencia de configuración y ajustar el nombre del *host* según sea necesario.

En este archivo se realizó las configuraciones de dirección *IP* 10.0.0.2/30, dirección de *Gateway* 10.0.0.1 y en este mismo archivo se deshabilita el *DHCP* de la interfaz del servidor. Una vez realizado esta configuración podemos verificar que esté en ejecución con el comando ***ifconfig*** como se muestra en la Figura 3.8.

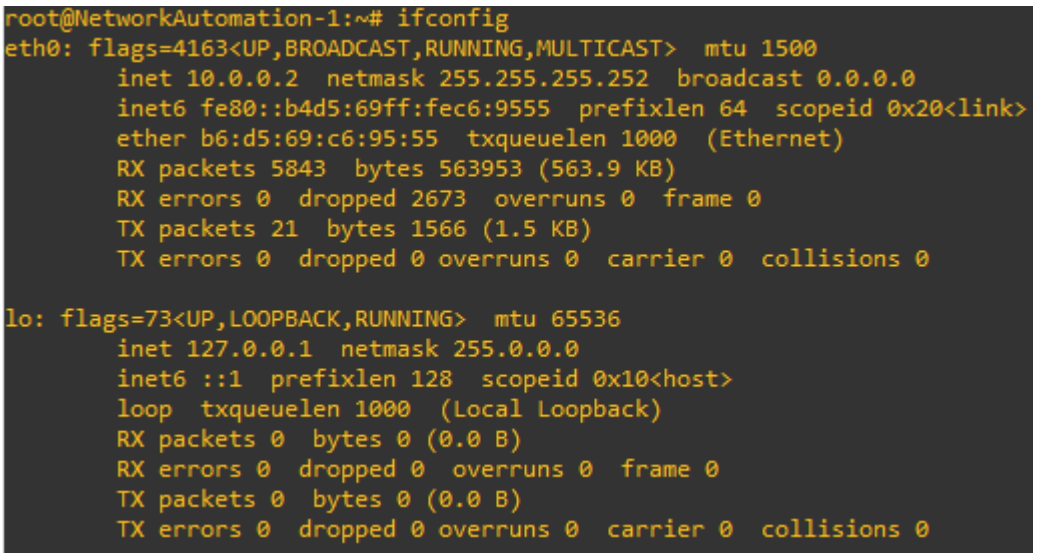


Figura 3.8 Configuración interfaz servidor

El código proporciona información sobre las interfaces de red en un sistema basado en *Linux* utilizando el comando *ifconfig*. Aquí está el detalle de la información proporcionada:

- eth0 (interfaz *Ethernet*):
    - *flags*=4163<*UP, BROADCAST, RUNNING, MULTICAST*>: Muestra las características de la interfaz, indicando que está activa (*UP*), permite difusión (*BROADCAST*), está en funcionamiento (*RUNNING*), y es una interfaz de multidifusión (*MULTICAST*).
    - *mtu* 1500: Indica la Unidad Máxima de Transmisión (*Maximum Transmission Unit*) configurada para 1500 *bytes*.
    - *inet* 10.0.0.2 *netmask* 255.255.255.252 broadcast 0.0.0.0: Muestra la configuración de dirección *IP* (10.0.0.2), máscara de subred (255.255.255.252) y difusión.
    - inet6 fe80::b4d5:69ff:fec6:9555 *prefixlen* 64 *scopeid* 0x20<*link*>: Muestra la dirección *IP*v6 y otros detalles.
    - ether b6:d5:69:c6:95:55: Muestra la dirección física (*MAC*) de la interfaz *Ethernet*.

    - *RX packets* 5843 *bytes* 563953 (563.9 KB): Estadísticas de recepción, indicando el número de paquetes recibidos y la cantidad de *bytes* recibidos.
    - *RX errors dropped* 2673 *overruns* 0 *frame* 0: Estadísticas de errores de recepción.
    - *TX packets* 21 *bytes* 1566 (1.5 *KB*): Estadísticas de transmisión, indicando el número de paquetes transmitidos y la cantidad de *bytes* transmitidos.
    - *TX errors dropped overruns carrier Ø collisions* 0: Estadísticas de errores de transmisión.
- lo (interfaz de bucle local, *loopback*):
    - *flags*=73<*UP, LOOPBACK, RUNNING*>: Muestra las características de la interfaz de bucle local, indicando que está activa (*UP*), es de bucle local (*LOOPBACK*) y está en funcionamiento (*RUNNING*).
    - *mtu* 65536: Indica la Unidad Máxima de Transmisión configurada para 65536 *bytes*.
    - inet 127.0.0.1 *netmask* 255.0.0.0: Muestra la configuración de dirección *IP* para la interfaz de bucle local (127.0.0.1).
    - inet6 ::1 *prefixlen* 128 *scopeid* 0x10<*host*>: Muestra la dirección *IP*v6 y otros detalles.
    - *loop txqueuelen* 1000 (Local *Loopback*): Indica que es una interfaz de bucle local.
    - *RX packets* 0 *bytes* 0 (0.0 B): Estadísticas de recepción para la interfaz de bucle local.
    - *TX packets* 0 *bytes* 0 (0.0 B): Estadísticas de transmisión para la interfaz de bucle local.

En resumen, la salida del comando ifconfig proporciona información detallada sobre las interfaces de red presentes en el sistema, incluyendo la dirección *IP*, la dirección MAC, estadísticas de recepción y transmisión, entre otros detalles

Una vez realizado estos pasos, se procede a realizar la verificación de conexión *SSH* entre el servidor *Ubuntu ansible* con los equipos *routers* de le topología, usando el comando ***ssh* admin@10.0.0.X**, en el cual la “X” se debe cambiar según la dirección del equipo que se necesita acceder como se muestra en la Figura 3.9.

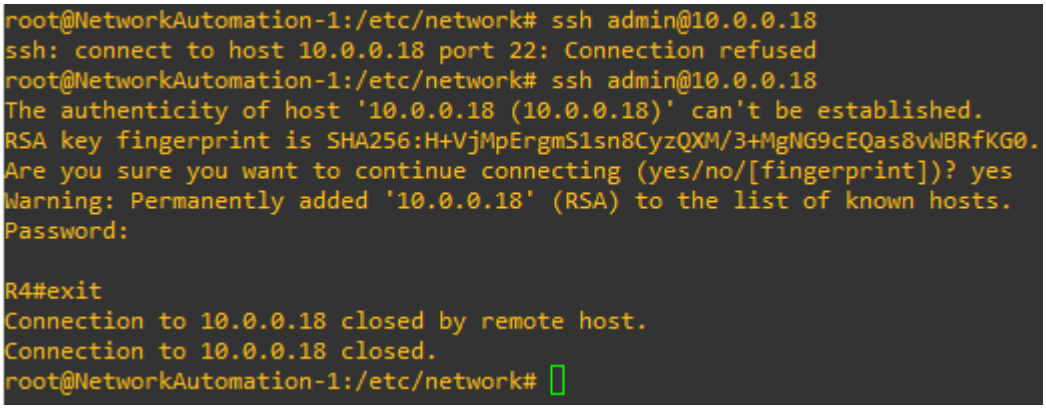


Figura 3.9 Conexión *SSH* entre servidor y equipo *router*

Este fragmento de código muestra una sesión de conexión *SSH* desde el *host* denominado "*NetworkAutomation*-1" hacia la dirección *IP* 10.0.0.18 utilizando el nombre de usuario "admin". Aquí está el detalle de cada paso:

- *ssh* admin@10.0.0.18: Inicia una conexión *SSH* al dispositivo con la dirección *IP* 10.0.0.18 utilizando el nombre de usuario "admin". *SSH* (*Secure Shell*) es un protocolo de red seguro utilizado para acceder y gestionar dispositivos remotos.
- *The authenticity of host* '10.0.0.18 (10.0.0.18) *can't be established*. *RSA key fingerprint is* SHA256:H+VjMpErgmSisnBCyzQXM/3+MgNG9cEQas8vWBRfKGO.: Advierte al usuario que la autenticidad del *host* no ha sido establecida y muestra la huella digital (*fingerprint*) de la clave *RSA* del *host*. Esto es una medida de seguridad para asegurarse de que la conexión *SSH* sea a un *host* legítimo.
- *Are you sure you want to continue connecting* (*yes*/*no*/[*fingerprint*])? *yes*: Pregunta al usuario si desea continuar con la conexión. En este caso, el usuario responde "*yes*", indicando que confía en la autenticidad del *host*.
- *Warning: Permanently added* '10.0.0.18' (*RSA*) *to the list of known hosts*.: Advierte al usuario que la dirección *IP* del *host* y su clave *RSA* han sido agregadas permanentemente a la lista de hosts conocidos. Esto evita que el sistema muestre advertencias similares en futuras conexiones a este *host*.
- *Password*: Solicita al usuario que ingrese la contraseña asociada al nombre de usuario "*admin*" para autenticarse en el *host* remoto.
- R4#*exit*: Una vez autenticado, el *prompt* cambia a "R4#", indicando que se ha establecido una sesión de *shell* en el dispositivo remoto (en este caso, un *router* con nombre "R4"). Luego, el usuario ejecuta el comando *exit* para salir de la sesión remota.

- *Connection to* 10.0.0.18 *closed*: Informa que la conexión al *host* con la dirección *IP* 10.0.0.18 ha sido cerrada.

En resumen, este código representa una conexión *SSH* exitosa desde el *host* "*NetworkAutomation*-1" al dispositivo con la dirección *IP* 10.0.0.18, utilizando el nombre de usuario "*admin*". El usuario ha confirmado la autenticidad del *host* y ha ingresado la contraseña para autenticarse en el dispositivo remoto (*router* con nombre "R4"). La sesión *SSH* se cierra después de ejecutar el comando *exit* en el dispositivo remoto

3.3 Implementación de las configuraciones mediante *Ansible*.

Se realiza la estructuración del inventario de los equipos a los cuales *ansible* se conectará e implementará las configuraciones respectivas y las tareas a realizar.

Para está estructuración ingresamos al archivo con el comando ***nano /etc/ansiblelhosts*** en el cual ingresaremos todos los equipos en la topología como se muestra en la Figura 3.10.

```
  GNU nano 4.8                    /etc/ansible/hosts                    Modified
[Routers]
# Indica el grupo de hosts
R1 ansible_host=10.0.0.1
R2 ansible_host=10.0.0.6
R3 ansible_host=10.0.0.10
R4 ansible_host=10.0.0.18
R5 ansible_host=10.0.0.22
R6 ansible_host=10.0.0.26
R7 ansible_host=10.0.0.30
R8 ansible_host=10.0.0.34

[Routers:vars]
ansible_network_os=cisco.ios.ios
ansible_user=admin
ansible_password=12345
ansible_become=true
ansible_become_method=ansible.netcommon.enable
ansible_become_pass=12345
ansible_connection=ansible.netcommon.network_cli
```

Figura 3.10 Inventario *hosts*

Este código *Ansible* es un archivo de inventario que define grupos de *hosts* y variables asociadas a esos grupos. Aquí está una descripción detallada de lo que hace cada parte del código:

- Encabezado [*Routers*]: Indica la creación de un grupo de *hosts* llamado "*Routers*". Los grupos de hosts son utilizados para organizar y agrupar máquinas en *Ansible*.
- Definición de *hosts* en el grupo "*Routers*":

    - R1 *ansible_host*=10.0.0.1: Define el *host* R1 con la dirección *IP* 10.0.0.1.
    - R2 *ansible_host*=10.0.0.6: Define el *host* R2 con la dirección *IP* 10.0.0.6.
    - R3 *ansible_host*=10.0.0.10: Define el *host* R3 con la dirección *IP* 10.0.0.10.
    - R4 *ansible_host*=10.0.0.18: Define el *host* R4 con la dirección *IP* 10.0.0.18.
    - R5 *ansible_host*=10.0.0.22: Define el *host* R5 con la dirección *IP* 10.0.0.22.
    - R6 *ansible_host*=10.0.0.26: Define el *host* R6 con la dirección *IP* 10.0.0.26.
    - R7 *ansible_host*=10.0.0.30: Define el *host* R7 con la dirección *IP* 10.0.0.30.
    - R8 *ansible_host*=10.0.0.34: Define el *host* R8 con la dirección *IP* 10.0.0.34.
- Encabezado [*Routers:vars*]: Indica la definición de variables para el grupo "*Routers*". Las variables especificadas aquí se aplicarán a todos los hosts dentro de este grupo.
- Variables definidas para el grupo "*Routers*":
    - *ansible_network_os cisco.ios.ios*: Establece el sistema operativo de red (*network OS*) para los hosts en el grupo como "*cisco.ios.ios*". Esto se utiliza para adaptar las tareas de *Ansible* según el sistema operativo específico.
    - *ansible_user*=admin: Especifica el nombre de usuario que *Ansible* utilizará para conectarse a los hosts.
    - *ansible_password*=12345: Define la contraseña que se utilizará para la autenticación.
    - *Ansible_become=true*: Habilita el privilegio de "*become*", que permite ejecutar comandos con privilegios elevados.
    - *ansible_become_method=ansible.netcommon.enable*: Indica el método que se utilizará para elevar privilegios, en este caso, utilizando el módulo *ansible.netcommon.enable*.
    - *ansible_become_pass*=12345: Especifica la contraseña para el proceso de elevación de privilegios.
    - *ansible_connection=ansible.netcommon.network_cli*: Establece el método de conexión como *ansible.netcommon.network_cli*, lo que indica

que *Ansible* utilizará el módulo *network_cli* para interactuar con los dispositivos de red.

En resumen, este código *Ansible* define un grupo de hosts llamado "*Routers*", asigna direcciones *IP* a cada router y establece variables específicas para el grupo, el sistema operativo de red, credenciales de acceso y configuraciones de privilegios

Al culminar el inventario de equipos que se declara en el archivo *hosts* con los cuales *ansible* interactuará, verifica la conectividad entre *ansible* con todos los equipos declarados como se muestra en las Figura 3.11 y Figura 3.12.

```
root@NetworkAutomation-1:/etc/ansible# ansible all -m ping
[WARNING]: ansible-pylibssh not installed, falling back to paramiko
[WARNING]: ansible-pylibssh not installed, falling back to paramiko
[WARNING]: ansible-pylibssh not installed, falling back to paramiko
[WARNING]: ansible-pylibssh not installed, falling back to paramiko
[WARNING]: ansible-pylibssh not installed, falling back to paramiko
R3 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
R1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
R4 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
R5 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
R2 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
```

Figura 3.11 *ping pong ansible* hacia equipos *routers*

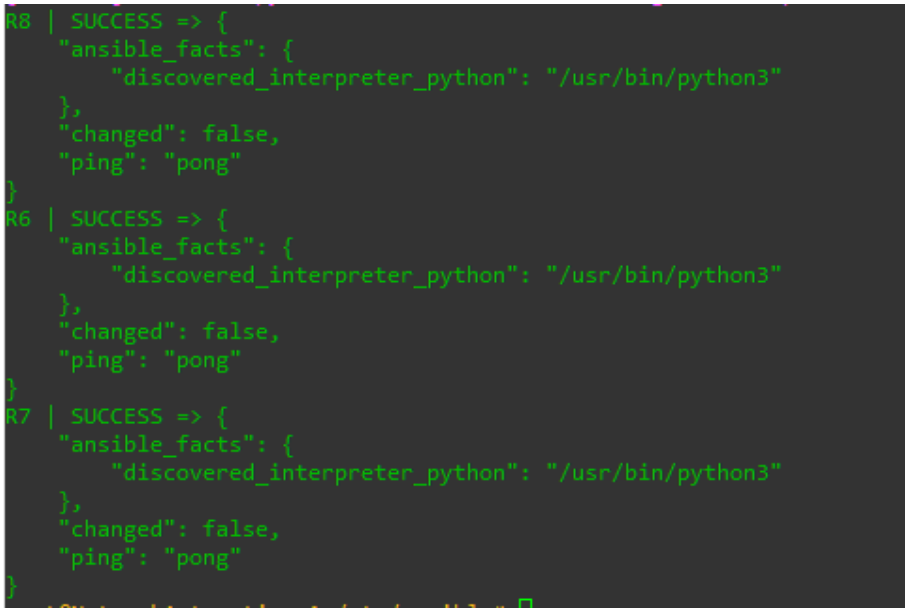


Figura 3.12 *ping pong ansible* hacia equipos *routers*

Posteriormente a la verificación de la conexión hacia los equipos *routers*, procedemos a realizar las configuraciones respectivas las cuales se enviará para cada uno de los equipos, crearemos los *playbooks* para cada realización de las tareas respectivas.

Este archivo se creará con el comando ***nano /etc/ansible/R1_1C.yml*** y en este archivo detallar las tareas que realizará para la implementación en la cual se comenzará a realizar los ruteos respectivos desde el primer equipo hacia los demás.

Las tareas para hacer en primera instancia se muestran en la Figura 3.13.

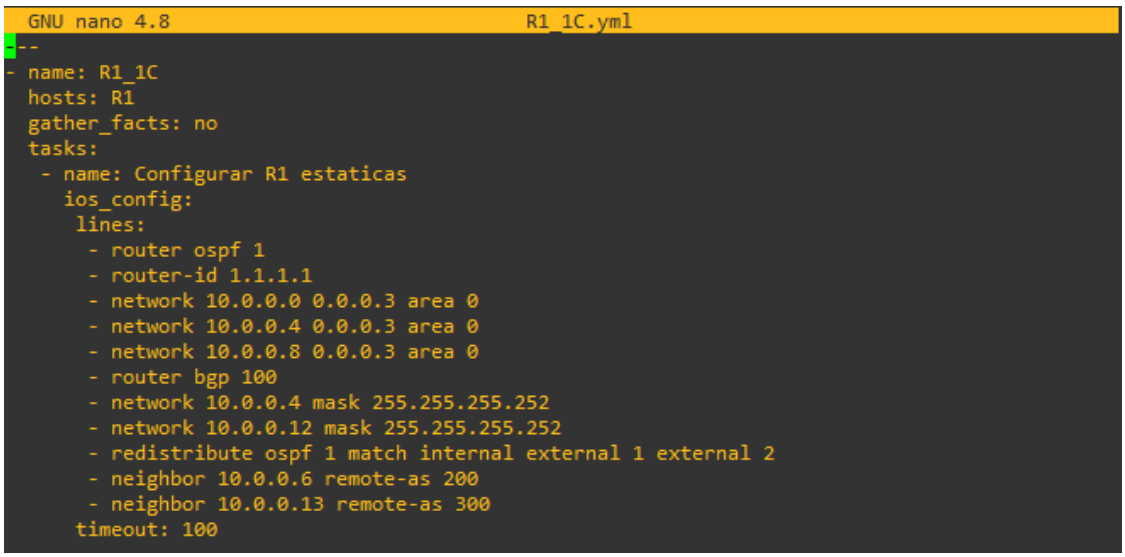


Figura 3.13 Primera configuración de equipos

Este código de *Ansible* está destinado a configurar un dispositivo de red identificado como "R1" con algunas configuraciones específicas utilizando el módulo *ios_config*. Aquí hay una descripción de lo que hará el código:

- Nombre de la tarea y *hosts*: El nombre de la tarea es "Configurar R1 estaticas". Se ejecutará en los hosts que pertenezcan al grupo "R1".
- *gather_facts*: La recopilación de hechos (*gather_facts*: no) está desactivada. Esto significa que *Ansible* no recopilará información sobre el sistema operativo de los *hosts* antes de ejecutar las tareas. Es útil desactivar la recopilación de hechos cuando no se necesita información previa.
- Tareas: La tarea principal utiliza el módulo *ios_config* para aplicar configuraciones específicas al dispositivo. Aquí están las configuraciones que se aplicarán:
    - Configuración *OSPF*: Se establece un proceso *OSPF* con *ID* 1, un *router* ID de 1.1.1.1 y se configuran varias redes con sus respectivas máscaras y áreas *OSPF*.
    - Configuración *BGP*: Se configura un proceso *BGP* con el número de autonomía (*AS*) 100. Se especifican algunas redes y se realiza la redistribución de rutas entre *OSPF* y *BGP*. También se configuran vecinos *BGP* con las direcciones *IP* 10.0.0.6 y 10.0.0.13, cada uno en su propio *AS* (200 y 300, respectivamente).
    - *timeout*: Se establece un tiempo de espera para la ejecución de las configuraciones en 100 segundos.

En resumen, el código configura el dispositivo R1 con configuraciones *OSPF* y *BGP* específicas, establece vecinos *BGP* y aplica un tiempo de espera de 100 segundos para la ejecución de las configuraciones. Es importante tener en cuenta que este código asume que el dispositivo es compatible con el módulo *ios_config* y que las configuraciones proporcionadas son apropiadas para la topología de red y los requisitos específicos

La configuración en el equipo se visualiza en la Figura 3.14.

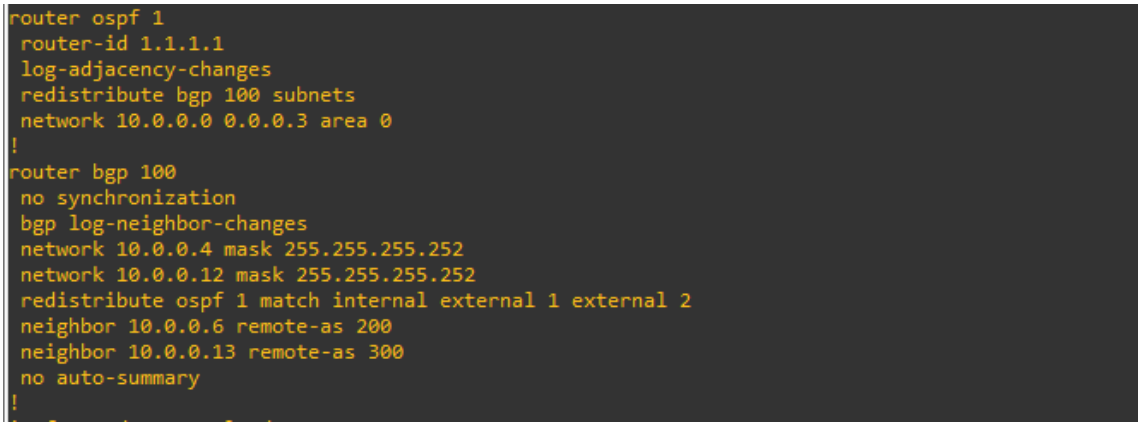


Figura 3.14 Visualización configuración en equipo

Este fragmento de código configura un *router* para utilizar *OSPF* (*Open Shortest Path First*) y *BGP* (*Border Gateway Protocol*) en un entorno de red. Aquí está el detalle de cada comando:

- *router ospf* 1: Configura el router para utilizar el protocolo *OSPF* con el identificador de proceso 1.
- *router-id* 1.1.1.1: Establece la identificación del *router OSPF* como 1.1.1.1. El *ID* del *router OSPF* es un identificador único que se utiliza para distinguir este *router* en la red *OSPF*.
- *log-adjacency-changes*: Habilita el registro de cambios en las adyacencias *OSPF*. Este comando registrará eventos cuando las relaciones de adyacencia *OSPF* cambien.
- *redistribute bgp* 100 *subnets*: Configura la redistribución de rutas aprendidas a través del protocolo *BGP* (con el número de autonomía 100) en *OSPF*, incluyendo información de subredes.
- *network* 10.0.0.0 0.0.0.3 area 0: Anuncia la red 10.0.0.0/30 en el área *OSPF* 0.
- *router bgp* 100: Configura el *router* para utilizar el protocolo *BGP* con el número de autonomía 100.
- *no synchronization*: Desactiva la sincronización de *BGP* con el protocolo de enrutamiento interior. Esto es comúnmente utilizado cuando se está redistribuyendo entre *BGP* y otro protocolo de enrutamiento.
- *bgp log-neighbor-changes*: Habilita el registro de cambios en los vecinos *BGP*. Este comando registrará eventos cuando cambien las relaciones con los vecinos *BGP*.
- *network* 10.0.0.4 *mask* 255.255.255.252: Anuncia la red 10.0.0.4/30 en *BGP*.
- *network* 10.0.0.12 *mask* 255.255.255.252: Anuncia la red 10.0.0.12/30 en *BGP*.

- *redistribute ospf* 1 *match internal external* 1 *external* 2: Configura la redistribución de rutas aprendidas a través de *OSPF* (con el identificador de proceso 1) en *BGP*. Se especifica la correspondencia de rutas internas y externas.
- *neighbor* 10.0.0.6 *remote-as* 200: Configura una vecindad *BGP* con el *router* en la dirección *IP* 10.0.0.6 y el número de autonomía (*AS*) remoto 200.
- *neighbor* 10.0.0.13 *remote-as* 300: Configura otra vecindad *BGP* con el *router* en la dirección *IP* 10.0.0.13 y el número de autonomía remoto 300.

En resumen, este código establece configuraciones para *OSPF* y *BGP* en un *router*, incluyendo la redistribución de rutas entre estos protocolos y la configuración de vecinos *BGP*.

Una vez configurado el primer *router* procedemos a realizar la implementación de cada equipo en la topología de manera remota, cabe mencionar que como en primera instancia se configuro ruteos estáticos, posteriormente se implementará un *playbook* para negar esa configuración de los equipos, y así tengan únicamente implementado los protocolos necesarios para el cumplimiento de objetivos del presente proyecto.

Las tareas por enviar se muestran en la Figura 3.15.

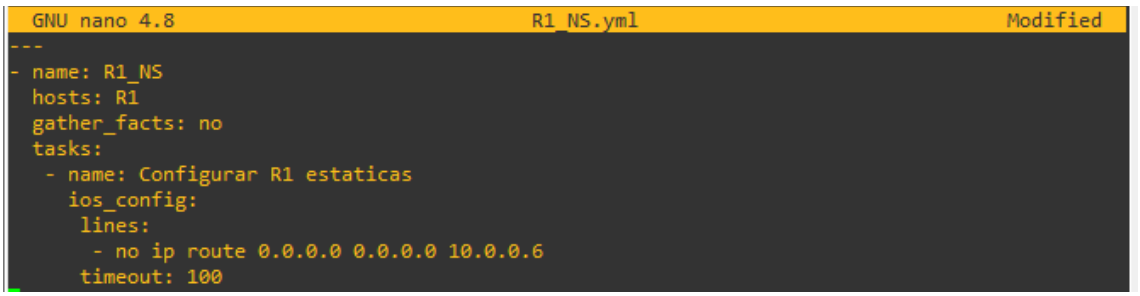


Figura 3.15 Negación ruteo estático

Este código de *Ansible* está diseñado para realizar una tarea específica en un dispositivo de red llamado "R1" utilizando el módulo *ios_config*. Aquí hay una descripción detallada de lo que hará el código:

- Nombre de la tarea y *hosts*: El nombre de la tarea es "Configurar R1 estaticas". Se ejecutará en los *hosts* que pertenezcan al grupo "R1".
- *gather_facts*: La recopilación de hechos (*gather_facts*: no) está desactivada. Esto significa que *Ansible* no recopilará información sobre el sistema operativo de los *hosts* antes de ejecutar las tareas. La desactivación de la recopilación de hechos es útil cuando no se necesita información previa.

- Tareas: La tarea principal utiliza el módulo *ios_config* para aplicar una configuración específica al dispositivo. Aquí está la configuración que se aplicará:
    - "no *IP route* 0.0.0.0 0.0.0.0 10.0.0.6": Se elimina la ruta estática predeterminada que apunta a la dirección *IP* 10.0.0.6. En otras palabras, se elimina la ruta predeterminada hacia el próximo salto 10.0.0.6 de la tabla de enrutamiento del dispositivo R1.
    - *timeout*: Se establece un tiempo de espera para la ejecución de la configuración en 100 segundos.

En resumen, el código elimina una ruta estática predeterminada en el dispositivo R1 que apunta a la dirección *IP* 10.0.0.6. Este tipo de cambio puede ser útil en situaciones donde se necesite ajustar la configuración de enrutamiento en el dispositivo para reflejar cambios en la red o en la topología. Es importante tener en cuenta que este código asume que el dispositivo es compatible con el módulo *ios_config* y que la configuración proporcionada es apropiada para los requisitos específicos de la red.

3.4 Implementación de las configuraciones mediante Ansible.

Una vez culminada la implementación de todos los equipos con los protocolos deseados, se realiza la visualización de los *LSA* en los equipos, como se muestra en la Figura 3.16.

```
R8#sh ip ospf database

            OSPF Router with ID (8.8.8.8) (Process ID 1)

                Router Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum Link count
6.6.6.6         6.6.6.6         1751        0x80000018 0x00D3BE 1
7.7.7.7         7.7.7.7         1811        0x80000018 0x00E79A 1
8.8.8.8         8.8.8.8         1500        0x80000018 0x0022F0 2

                Net Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
10.0.0.34       8.8.8.8         1500        0x80000017 0x00D6CB
10.0.0.38       8.8.8.8         1500        0x80000017 0x00E0B9

                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
10.0.0.0        6.6.6.6         1751        0x80000017 0x00C131 100
10.0.0.4        6.6.6.6         1751        0x80000017 0x008AC8 0
10.0.0.8        6.6.6.6         1751        0x80000017 0x0062EC 0
10.0.0.12       7.7.7.7         1811        0x80000017 0x001C2B 0
10.0.0.16       6.6.6.6         1751        0x80000017 0x001235 0
10.0.0.20       7.7.7.7         1811        0x80000017 0x00CB73 0
10.0.0.24       6.6.6.6         1751        0x80000017 0x00C17D 0
10.0.0.28       7.7.7.7         1811        0x80000017 0x007BBB 0
```

Figura 3.16 *LSA* tipo 1, tipo 2 y tipo 5

El código proporciona información sobre el estado de la base de datos *OSPF* en un *router*. Aquí está el detalle de la salida del comando *show ip* ospf database:

- Encabezado de Información del *Router OSPF*:
  Indica que el *router* está ejecutando *OSPF* con un *ID* de *router* de 8.8.8.8 y pertenece al proceso *OSPF* con el *ID* 1.
- *Router Link States* (Área 2):
  - *Link ID:* Identificador de enlace.
  - *ADV Router: Router* que ha originado el anuncio.
  - *Age:* Edad del anuncio en segundos.
  - *Seq:* Número de secuencia del anuncio.
  - *Checksum:* Valor de suma de comprobación del anuncio.
  - *Link count:* Número de enlaces asociados al anuncio.
- *Net Link States (Área 2):*
  Muestra información sobre enlaces de red en el área 2.
- *Type-5 AS External Link States:*
  Muestra información sobre enlaces externos del Tipo 5.

En resumen, la salida del comando proporciona detalles sobre el estado de la base de datos *OSPF* en el *router*, incluyendo información sobre los enlaces del *router*, enlaces de red y enlaces externos del Tipo 5. La información incluye detalles como *ID* de enlace, *router* anunciante, edad, número de secuencia, suma de comprobación y etiquetas (en el caso de enlaces externos).

Se puede evidenciar que en el *router* 1 los resultados de los *routers* de borde nos muestran los *LSA* completos que son el tipo 1, tipo 2 y tipo 5 como se observa en la Figura 3.17.

```
R1#show ip ospf database

            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Router Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum Link count
1.1.1.1         1.1.1.1         87          0x80000001 0x00D050 1

                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
10.0.0.4        1.1.1.1         88          0x80000001 0x004E56 0
10.0.0.8        1.1.1.1         6           0x80000001 0x004493 200
10.0.0.12       1.1.1.1         88          0x80000001 0x00FD9E 0
10.0.0.16       1.1.1.1         6           0x80000001 0x00F3DB 200
10.0.0.20       1.1.1.1         6           0x80000001 0x00D98D 300
10.0.0.24       1.1.1.1         6           0x80000001 0x00A324 200
10.0.0.28       1.1.1.1         6           0x80000001 0x0089D5 300
10.0.0.32       1.1.1.1         6           0x80000001 0x00536C 200
10.0.0.36       1.1.1.1         6           0x80000001 0x002B90 200
```

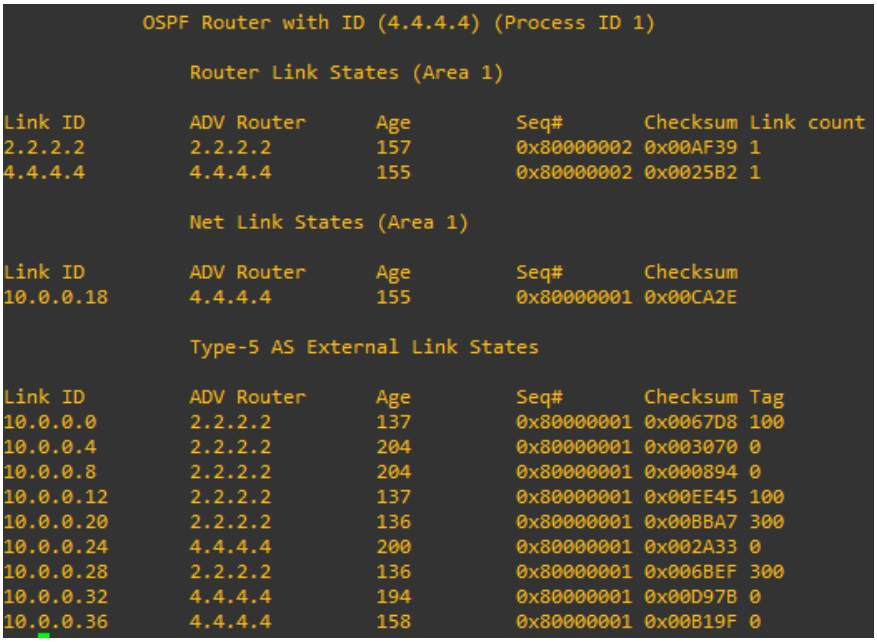
Figura 3.17 *LSA* tipo 1, tipo 2 y tipo 5

Se puede evidenciar que en *router* 2, de la misma forma nos muestra los *LSA* tipo 1, tipo 2 y tipo 5, ya que se encuentra como un router de borde y tiene diferentes sistemas autónomos de OSPF, como se observa en la Figura 3.18.

```
            OSPF Router with ID (2.2.2.2) (Process ID 1)

                Router Link States (Area 1)

Link ID         ADV Router      Age         Seq#       Checksum Link count
2.2.2.2         2.2.2.2         80          0x80000002 0x00AF39 1
4.4.4.4         4.4.4.4         80          0x80000002 0x0025B2 1

                Net Link States (Area 1)

Link ID         ADV Router      Age         Seq#       Checksum
10.0.0.18       4.4.4.4         80          0x80000001 0x00CA2E

                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
10.0.0.0        2.2.2.2         52          0x80000001 0x0067D8 100
10.0.0.4        2.2.2.2         128         0x80000001 0x003070 0
10.0.0.8        2.2.2.2         128         0x80000001 0x000894 0
10.0.0.12       2.2.2.2         52          0x80000001 0x00EE45 100
10.0.0.20       2.2.2.2         52          0x80000001 0x00BBA7 300
10.0.0.24       4.4.4.4         125         0x80000001 0x002A33 0
10.0.0.28       2.2.2.2         52          0x80000001 0x006BEF 300
10.0.0.32       4.4.4.4         173         0x80000001 0x00D97B 0
10.0.0.36       4.4.4.4         137         0x80000001 0x00B19F 0
```

Figura 3.18 *LSA* tipo 1, tipo 2 y tipo 5

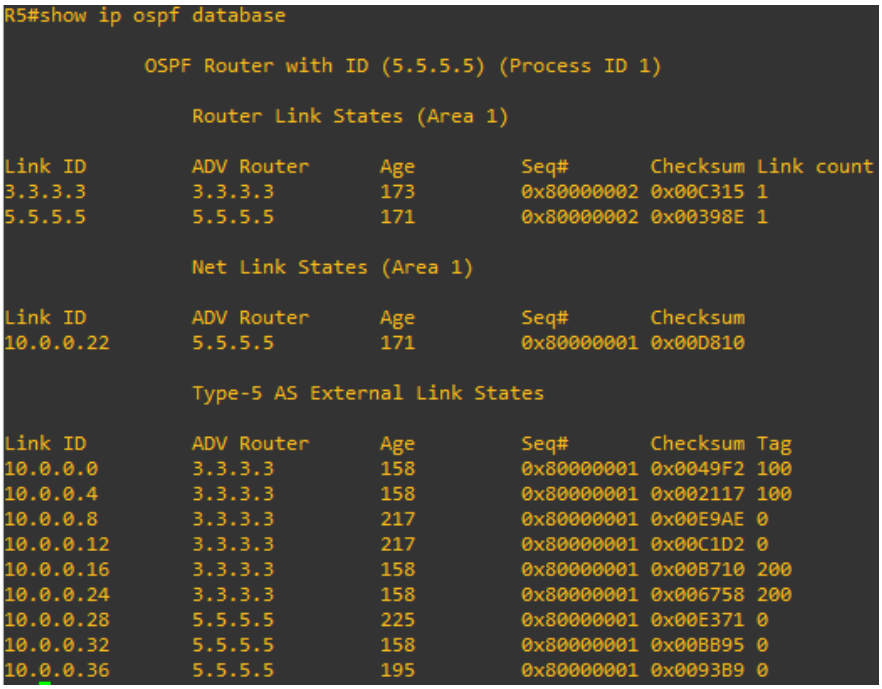
Podemos observar que en el *router* 4 de la misma manera se verifica que los *LSA* tipo 1 son entre mismas áreas de *OSPF* y los *LSA* de diferente tipo son los de área diferentes, como se observa en la Figura 3.19.



```
          OSPF Router with ID (4.4.4.4) (Process ID 1)

              Router Link States (Area 1)

Link ID         ADV Router      Age         Seq#       Checksum Link count
2.2.2.2         2.2.2.2         157         0x80000002 0x00AF39 1
4.4.4.4         4.4.4.4         155         0x80000002 0x0025B2 1

              Net Link States (Area 1)

Link ID         ADV Router      Age         Seq#       Checksum
10.0.0.18       4.4.4.4         155         0x80000001 0x00CA2E

              Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
10.0.0.0        2.2.2.2         137         0x80000001 0x0067D8 100
10.0.0.4        2.2.2.2         204         0x80000001 0x003070 0
10.0.0.8        2.2.2.2         204         0x80000001 0x000894 0
10.0.0.12       2.2.2.2         137         0x80000001 0x00EE45 100
10.0.0.20       2.2.2.2         136         0x80000001 0x00BBA7 300
10.0.0.24       4.4.4.4         200         0x80000001 0x002A33 0
10.0.0.28       2.2.2.2         136         0x80000001 0x006BEF 300
10.0.0.32       4.4.4.4         194         0x80000001 0x00D97B 0
10.0.0.36       4.4.4.4         158         0x80000001 0x00B19F 0
```

Figura 3.19 *LSA* tipo 1, tipo 2 y tipo 5

En el caso del *router* 5, se puede notar que la verificación de los *LSA* tipo 1 se realiza dentro de las mismas áreas de *OSPF*, mientras que los *LSA* de diferentes tipos corresponden a áreas distintas, tal como se evidencia en la Figura 3.20.



```
R5#show ip ospf database

          OSPF Router with ID (5.5.5.5) (Process ID 1)

              Router Link States (Area 1)

Link ID         ADV Router      Age         Seq#       Checksum Link count
3.3.3.3         3.3.3.3         173         0x80000002 0x00C315 1
5.5.5.5         5.5.5.5         171         0x80000002 0x00398E 1

              Net Link States (Area 1)

Link ID         ADV Router      Age         Seq#       Checksum
10.0.0.22       5.5.5.5         171         0x80000001 0x00D810

              Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
10.0.0.0        3.3.3.3         158         0x80000001 0x0049F2 100
10.0.0.4        3.3.3.3         158         0x80000001 0x002117 100
10.0.0.8        3.3.3.3         217         0x80000001 0x00E9AE 0
10.0.0.12       3.3.3.3         217         0x80000001 0x00C1D2 0
10.0.0.16       3.3.3.3         158         0x80000001 0x00B710 200
10.0.0.24       3.3.3.3         158         0x80000001 0x006758 200
10.0.0.28       5.5.5.5         225         0x80000001 0x00E371 0
10.0.0.32       5.5.5.5         158         0x80000001 0x00BB95 0
10.0.0.36       5.5.5.5         195         0x80000001 0x0093B9 0
```

Figura 3.20 *LSA* tipo 1, tipo 2 y tipo 5

Observando el *router* 8, se aprecia que la confirmación de los *LSA* tipo 1 tiene lugar dentro de las áreas OSPF correspondientes, mientras que los *LSA* de tipos diferentes se asocian a áreas distintas, como se puede notar en la Figura 3.21.

```
R8#show ip ospf database

            OSPF Router with ID (8.8.8.8) (Process ID 1)

                Router Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum Link count
6.6.6.6         6.6.6.6         219         0x80000002 0x00FFA8 1
7.7.7.7         7.7.7.7         186         0x80000002 0x001484 1
8.8.8.8         8.8.8.8         185         0x80000003 0x004CDB 2

                Net Link States (Area 2)

Link ID         ADV Router      Age         Seq#       Checksum
10.0.0.34       8.8.8.8         218         0x80000001 0x0003B5
10.0.0.38       8.8.8.8         185         0x80000001 0x000DA3

                Type-5 AS External Link States

Link ID         ADV Router      Age         Seq#       Checksum Tag
10.0.0.0        6.6.6.6         181         0x80000001 0x00ED1B 100
10.0.0.4        6.6.6.6         199         0x80000001 0x00B6B2 0
10.0.0.8        6.6.6.6         199         0x80000001 0x008ED6 0
10.0.0.12       6.6.6.6         181         0x80000001 0x007587 100
10.0.0.16       6.6.6.6         251         0x80000001 0x003E1F 0
10.0.0.20       6.6.6.6         182         0x80000001 0x0042E9 300
10.0.0.24       6.6.6.6         267         0x80000001 0x00ED67 0
10.0.0.28       7.7.7.7         229         0x80000001 0x00A7A5 0
```

Figura 3.21 *LSA* tipo 1, tipo 2 y tipo 5

El enrutador con la *IP* 1.1.1.1 se anuncia a sí mismo como un Enrutador de Límite de Sistema Autónomo (*ASBR*) con un enlace conectado a una Red de Acceso Restringido (*Stub Network*) (10.0.0.0/30). La antigüedad del Estado del Enlace es 287 e incluye información sobre el enlace, la red y las métricas de Tipo de Servicio (*TOS*) como se observa en la Figura 3.22.

```
R1#show ip ospf database router

            OSPF Router with ID (1.1.1.1) (Process ID 1)

                Router Link States (Area 0)

  LS age: 287
  Options: (No TOS-capability, DC)
  LS Type: Router Links
  Link State ID: 1.1.1.1
  Advertising Router: 1.1.1.1
  LS Seq Number: 80000001
  Checksum: 0xD050
  Length: 36
  AS Boundary Router
  Number of Links: 1

    Link connected to: a Stub Network
     (Link ID) Network/subnet number: 10.0.0.0
     (Link Data) Network Mask: 255.255.255.252
      Number of TOS metrics: 0
       TOS 0 Metrics: 1
```

Figura 3.22 *LSA* tipo 1

A continuación, vemos dos *routers* involucrados, identificados por las direcciones *IP* 3.3.3.3 y 5.5.5.5, ambos desempeñando el rol de Enrutadores de Límite de Sistema Autónomo (*ASBR*). Cada uno de ellos está conectado a una Red de Tránsito mediante un enlace, y proporciona información detallada sobre la dirección del Enrutador Designado y la dirección de interfaz del enrutador respectivo. Ambos anuncios presentan una antigüedad de 264 y métricas *TOS* especificadas. Es relevante destacar que, en el segundo anuncio, el bit de enrutamiento está activado en este *LSA* como se observa en la Figura 3.23.

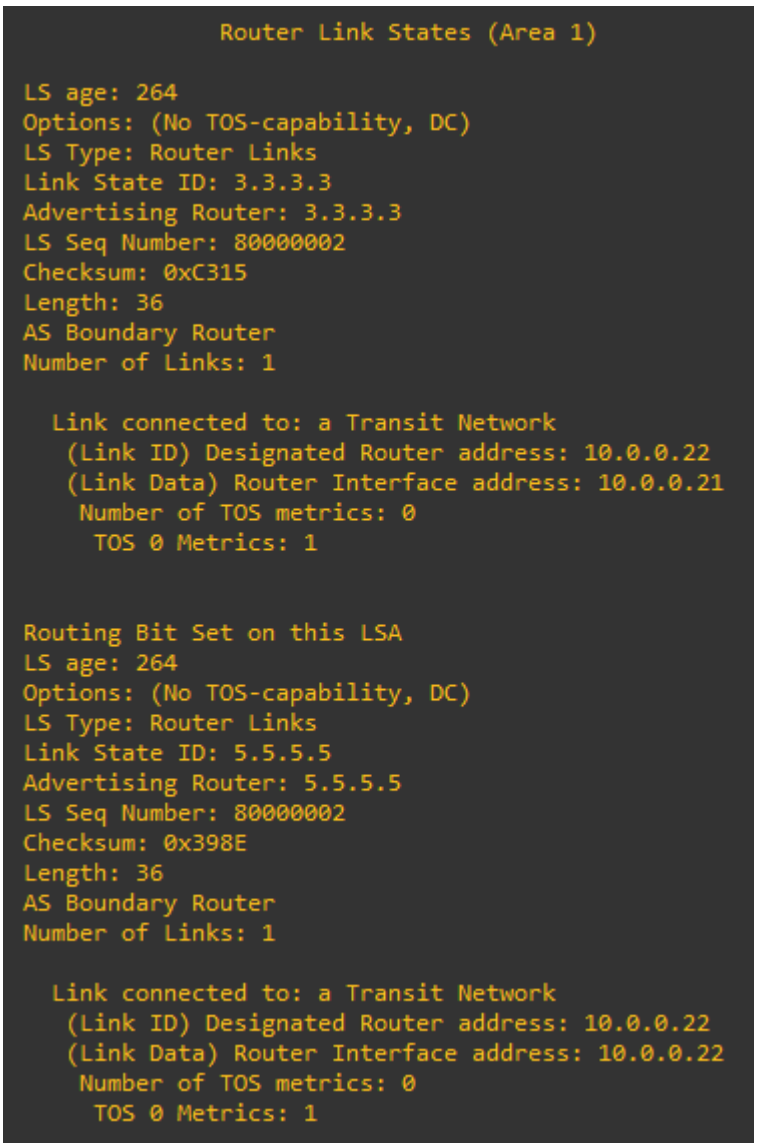


Figura 3.23 *LSA* tipo 1

El *router* con *ID* 5.5.5.5 y un Proceso *ID* de 1 en el Área 1. El resumen muestra la cantidad de *LSAs* por tipo, incluyendo *routers*, redes, resúmenes de red, *ASBR* tipo-7, extensiones tipo-7. También se proporcionan conteos para la redistribución de prefijos

en tipo-7 y tipo-5, así como *LSAs* de área opaca y *AS* opaco. En total, hay 12 *LSAs* en la base de datos del proceso 1 como se muestra en la Figura 3. 24.

```
           OSPF Router with ID (5.5.5.5) (Process ID 1)

Area 1 database summary
  LSA Type      Count    Delete   Maxage
  Router        2        0        0
  Network       1        0        0
  Summary Net   0        0        0
  Summary ASBR  0        0        0
  Type-7 Ext    0        0        0
    Prefixes redistributed in Type-7  0
  Opaque Link   0        0        0
  Opaque Area   0        0        0
  Subtotal      3        0        0

Process 1 database summary
  LSA Type      Count    Delete   Maxage
  Router        2        0        0
  Network       1        0        0
  Summary Net   0        0        0
  Summary ASBR  0        0        0
  Type-7 Ext    0        0        0
  Opaque Link   0        0        0
  Opaque Area   0        0        0
  Type-5 Ext    9        0        0
      Prefixes redistributed in Type-5  3
  Opaque AS     0        0        0
  Non-self      7
  Total         12       0        0
```

Figura 3. 24 *LSA* tipo 5

4 CONCLUSIONES

- La implementación de automatización de *DevOps* en *networking* nos ayuda a optimizar el manejo de equipamiento de manera remota y detallada únicamente enviando tareas específicas.
- Automatización y Orquestación, *DevOps* fomenta la automatización y orquestación de tareas en la gestión de redes. La automatización ayuda a reducir errores, mejora la consistencia y agiliza el despliegue de servicios.
- Infraestructura como Código (*IaC*), la implementación de servicios de *networking* se beneficia de la adopción de *IaC*, donde la infraestructura se define y gestionada mediante código. Esto permite una gestión más eficiente, trazabilidad y escalabilidad.
- Entregas Continuas, *DevOps* facilita las entregas continuas en el contexto de *networking*, lo que significa que las actualizaciones y cambios en la

infraestructura de red pueden ser implementados de manera rápida y confiable, reduciendo los tiempos de inactividad.

- Monitoreo y Análisis, la implementación de servicios de *networking* en un entorno *DevOps* enfatiza el monitoreo continuo y el análisis de rendimiento. Esto permite la identificación proactiva de problemas, la optimización de recursos y una mejora constante en la calidad del servicio.
- Colaboración entre Equipos, *DevOps* fomenta la colaboración entre equipos de desarrollo y operaciones, lo que en el contexto de *networking* implica una comunicación estrecha entre los profesionales de redes y los desarrolladores para garantizar la alineación de los objetivos y la resolución rápida de problemas.
- Seguridad Integrada, la seguridad es una parte integral de la implementación de servicios de *networking* en un entorno *DevOps*. Las prácticas de seguridad deben integrarse desde el principio del ciclo de vida del desarrollo y despliegue, asegurando la protección de la infraestructura de red contra amenazas.
- Flexibilidad y Escalabilidad, *DevOps* proporciona la flexibilidad necesaria para adaptarse a cambios rápidos en los requisitos y escalar la infraestructura según sea necesario. Esto es crucial en entornos de *networking* donde la demanda puede variar.
- Resiliencia y Recuperación, la implementación de servicios de *networking* mediante *DevOps* incluye la planificación para la resiliencia y la recuperación. Se deben establecer prácticas y mecanismos para garantizar la continuidad de los servicios en caso de fallas.
- Ciclo de Retroalimentación Continua, la implementación de servicios de *networking* en un entorno *DevOps* se beneficia de un ciclo de retroalimentación continua. La retroalimentación constante y la mejora continua son fundamentales para adaptarse a los cambios en las necesidades y tecnologías de red
- Se concluye que al utilizar *DevOps* para la ejecución de tareas de *networking*, se debe aclarar el alcance de las necesidades a realizar, y con esto se puede realizar de una manera más clara y fácil las implementaciones adecuadas.
- Establecer los parámetros a implementar, para evidenciar el detalle de *LSA* en la topología implementada, y verificar el funcionamiento correcto del mismo.
- El uso de ansible para *networking* conlleva a una manera sustancial de manejo de redes de alto alcance, mismo que nos permite realizar tareas de una manera óptima.

- Considerando las tareas realizadas en la implementación de servicios de *networking* mediante *ansible*, es viable en ejecución y en funcionamiento, abarcando el cumplimiento de los objetivos propuestos de manera satisfactoria.

5 RECOMENDACIONES

- Se sugiere explorar posibles escenarios de aplicación para esta solución, la cual, mediante un enfoque investigativo, podría resultar beneficiosa en entornos con redes extensas, permitiendo así un análisis más exhaustivo de la infraestructura empresarial.
- Dado que *Ansible* se destaca como una de las herramientas más reconocidas, demostrando eficacia y ofreciendo beneficios significativos, es recomendable explorar otras herramientas de automatización de procesos específicas para escenarios particulares. Esto permitirá elegir la más adecuada para satisfacer cada necesidad de manera óptima.
- Automatiza tanto como sea posible, desde la configuración de red hasta la gestión de políticas y la monitorización. Utiliza herramientas de automatización como *Ansible*, *Puppet* o *Terraform* para definir y gestionar la infraestructura como código.
- Aplica el concepto de versionado de configuración utilizando sistemas de control de versiones como *Git*. Esto permite realizar un seguimiento de los cambios, revertir a versiones anteriores y coordinar colaborativamente con otros equipos.
- Fomenta la colaboración continua entre los equipos de desarrollo y operaciones (*DevOps*). Establece una comunicación fluida y la integración de procesos para lograr objetivos comunes y resolver problemas de manera eficiente.
- Implementa pruebas automatizadas para verificar la configuración de red y la conectividad. Esto ayuda a identificar problemas antes de la implementación en producción, mejorando la calidad y la confiabilidad del servicio.
- Adopta prácticas de entregas continuas y despliegues pequeños y frecuentes. Esto reduce el riesgo de implementación y permite correcciones rápidas en caso de problemas.
- Establece una sólida estrategia de monitoreo y análisis para identificar problemas de rendimiento y seguridad en tiempo real. Utiliza herramientas de monitoreo como *Nagios*, *Prometheus* o *Grafana*.
- Incorpora prácticas de seguridad desde el inicio del desarrollo hasta la implementación. Asegúrate de aplicar principios de seguridad como la

segmentación de red, cifrado y gestión de identidad para proteger la infraestructura de *networking*.

- Diseña la infraestructura de red de manera escalable y modular para adaptarse a cambios en la demanda. Utiliza patrones de diseño como microservicios para facilitar la escalabilidad y la gestión.
- Proporciona capacitación continua para los equipos involucrados en la implementación de servicios de *networking* mediante *DevOps*. Mantente al tanto de las nuevas tecnologías y metodologías para garantizar la eficacia a largo plazo.
- Mantiene una documentación clara y actualizada de la configuración de red, procesos de implementación y procedimientos de resolución de problemas. Esto facilita la transferencia de conocimientos y la resolución eficiente de problemas.
- Evalúa la posibilidad de utilizar servicios en la nube y soluciones gestionadas para simplificar la administración y escalabilidad de la infraestructura de red.
- Implementa estrategias de respaldo y recuperación para garantizar la continuidad del servicio en caso de fallos o interrupciones.

- Después de analizar las necesidades de implementación en diversas topologías, es esencial considerar cuidadosamente la tarea a ejecutar, asegurándose de que no afecte el funcionamiento de la red.
- La documentación detallada de las tareas nos brindará una visión integral de la implementación y las posibles áreas de mejora.

6 REFERENCIAS BIBLIOGRÁFICAS

BIBLIOGRAFÍA

[1] CISCO, «CISCO,» CISCO, 28 Agosto 2023. [En línea]. Available: https://www.cisco.com/c/es_mx/support/docs/ip/open-shortest-path-first-ospf/7039-1.html. [Último acceso: 2 Febrero 2024].

[2] C. D. CERO, «CCNA DESDE CERO,» CCNA DESDE CERO, 26 Noviembre 2020. [En línea]. Available: https://ccnadesdecero.com/curso/como-configurar-ospf/. [Último acceso: 2 Febrero 2024].

[3] OSPF, «IBM,» IBM, 08 MARZO 2021. [En línea]. Available: https://www.ibm.com/docs/es/i/7.1?topic=routing-open-shortest-path-first. [Último acceso: 02 FEBRERO 2024].

[4] OSPF, «SAPALOMERA,» SAPALOMERA, 01 ENERO 2024. [En línea]. Available: https://www.sapalomera.cat/moodlecf/RS/2/course/module8/index.html#8.0.1.1. [Último acceso: 2 FEBRERO 2024].

[5] C. D. CERO, «CCNA DESDE CERO,» CCNA DESDE CERO, 11 Agosto 2020. [En línea]. Available: https://ccnadesdecero.com/curso/ospf-intercambio-lsbd/. [Último acceso: 2 Febrero 2024].

[6] CISCO, «CISCO,» CISCO, 18 OCTUBRE 2023. [En línea]. Available: https://www.cisco.com/c/es_mx/support/docs/ip/open-shortest-path-first-ospf/13703-8.html. [Último acceso: 02 FEBRERO 2024].

[7] SAPALOMERA, «SAPALOMERA,» SAPALOMERA, 1 ENERO 2024. [En línea]. Available: https://www.sapalomera.cat/moodlecf/RS/3/course/module6/6.1.2.1/6.1.2.1.html. [Último acceso: 2 FEBRERO 2024].

[8] EDUANGI, «EDUANGI,» EDUANGI, 4 MAYO 2009. [En línea]. Available: https://www.eduangi.org/node131.html. [Último acceso: 2 FEBRERO 2024].

[9] SAPALOMERA, «SAPALOMERA,» SAPALOMERA, 01 ENERO 2024. [En línea]. Available: https://www.sapalomera.cat/moodlecf/RS/3/course/module6/6.1.2.4/6.1.2.4.html. [Último acceso: 2 FEBRERO 2024].

[10] GNS3, «GNS3,» GNS3, 01 Enero 2024. [En línea]. Available: https://www.gns3.com/software. [Último acceso: 02 Febrero 2024].

[11] G. G. Ramirez, «GNS3 como herramienta para el diseño y,» Universidad de Valladolid. Escuela de Ingeniería Informática de Valladolid., Valladolid, 2022.

[12] RedHat, «redhat,» RedHat, 21 Junio 2022. [En línea]. Available: https://www.redhat.com/es/topics/automation/learning-ansible-tutorial. [Último acceso: 1 Febrero 2024].

[13] R. Hat, «ANSIBLE,» Red Hat, 1 enero 2024. [En línea]. Available: https://docs.ansible.com/. [Último acceso: 2 Febrero 2024].

[14] ANSIBLE, «ANSIBLE,» ANSIBLE, 01 ENERO 2024. [En línea]. Available: https://docs.ansible.com/ansible/latest/collections/cisco/ios/index.html. [Último acceso: 02 FEBRERO 2024].

[15] ANSIBLE, «ANSIBLE,» ANSIBLE, 01 ENERO 2024. [En línea]. Available: https://docs.ansible.com/ansible/latest/collections/cisco/ios/ios_ospf_interfaces_module.html#ansible-collections-cisco-ios-ios-ospf-interfaces-module. [Último acceso: 02 FEBRERO 2024].

[16] ANSIBLE, «ANSIBLE,» ANSIBLE, 01 enero 2024. [En línea]. Available: https://docs.ansible.com/ansible/latest/collections/cisco/ios/ios_cliconf.html#ansible-collections-cisco-ios-ios-cliconf. [Último acceso: 2 febrero 2024].

[17] ANSIBLE, «ANSIBLE,» ANSIBLE, 01 ENERO 2024. [En línea]. Available: https://docs.ansible.com/ansible/2.9/. [Último acceso: 02 FEBRERO 2024].

[18] ANSIBLE, «ANSIBLE,» ANSIBLE, 1 ENERO 2024. [En línea]. Available: https://docs.ansible.com/ansible/latest/reference_appendices/release_and_maintenance.html. [Último acceso: 02 FEBRERO 2024].

[19] ANDIBLE, «ANSIBLE,» ANSIBLE, 01 ENERO 2024. [En línea]. Available: https://docs.ansible.com/ansible/latest/collections/cisco/ios/ios_cliconf.html#ansible-collections-cisco-ios-ios-cliconf. [Último acceso: 02 FEBRERO 2024].

[20] R. Hat, «Red Hat,» Red Hat, 3 Marzo 2023. [En línea]. Available: https://www.redhat.com/es/topics/automation/what-is-yaml. [Último acceso: 1 Febrero 2024].

7 ANEXOS

ANEXO I: Certificado de Originalidad

CERTIFICADO DE ORIGINALIDAD

Quito, D.M. XX de XXXX de 2022

De mi consideración:

Yo, FERNANDO VINICIO BECERRA CAMACHO, en calidad de Director del Trabajo de Integración Curricular titulado IMPLEMENTACIÓN DE SERVICIOS DE NETWORKING MEDIANTE DEVOPS asociado al TELEMÁTICA APLICADA elaborado por el estudiante BYRON PATRICIO ENCALADA MONTUFAR de la carrera en Tecnología 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 xxx%.

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.

LINK

Atentamente,

FERNANDO VINICIO BECERRA CAMACHO

Docente

Escuela de Formación de Tecnólogos

ANEXO II: Enlaces

ByronEncalada.mp4

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

ANEXO III: Contenido de archivos de configuración

Archivo configuraciones R1_1C.yml

```
---
- name: R1_1C
  hosts: R1
  gather_facts: no
  tasks:
   - name: Configurar R1 estaticas
     ios_config:
      lines:
       - router ospf 1
       - router-id 1.1.1.1
       - network 10.0.0.0 0.0.0.3 area 0
       - network 10.0.0.4 0.0.0.3 area 0
       - network 10.0.0.8 0.0.0.3 area 0
       - router bgp 100
       - network 10.0.0.4 mask 255.255.255.252
       - network 10.0.0.12 mask 255.255.255.252
       - redistribute ospf 1 match internal external 1 external 2
       - neighbor 10.0.0.6 remote-as 200
       - neighbor 10.0.0.13 remote-as 300
      timeout: 100
```

Archivo configuraciones R2_2C.yml

```
---
- name: R2_2C
  hosts: R2
```

```
    gather_facts: no
    tasks:
     - name: Configurar R2 estaticas
       ios_config:
        lines:
         - router ospf 1
         - router-id 2.2.2.2
         - network 10.0.0.16 0.0.0.3 area 0
         - router bgp 200
         - network 10.0.0.4 mask 255.255.255.252
         - network 10.0.0.8 mask 255.255.255.252
         - redistribute ospf 1 match internal external 1 external 2
         - neighbor 10.0.0.5 remote-as 100
         - neighbor 10.0.0.10 remote-as 300
        timeout: 100
```

Archivo configuraciones R3_3C.yml

```
---
- name: R3_3C
  hosts: R3
  gather_facts: no
  tasks:
   - name: Configurar R3 estaticas
     ios_config:
      lines:
       - router ospf 1
       - router-id 3.3.3.3
       - network 10.0.0.20 0.0.0.3 area 1
```

```
      - router bgp 300
      - network 10.0.0.8 mask 255.255.255.252
      - network 10.0.0.12 mask 255.255.255.252
      - redistribute ospf 1 match internal external 1 external 2
      - neighbor 10.0.0.9 remote-as 200
      - neighbor 10.0.0.14 remote-as 100
     timeout: 100
```

Archivo configuraciones R4_4C.yml

```
---
- name: R4_4C
  hosts: R4
  gather_facts: no
  tasks:
   - name: Configurar R4 estaticas
     ios_config:
      lines:
       - router eigrp 1
       - redistribute ospf 1 metric 100000 65000 255 255 10000
       - network 10.0.0.24 0.0.0.3
       - no auto-summary
       - router ospf 1
       - router-id 4.4.4.4
       - redistribute eigrp 1 metric 100000 subnets
       - network 10.0.0.16 0.0.0.3 area 1
      timeout: 100
```

Archivo configuraciones R5_5C.yml

```
---
- name: R5_5C
  hosts: R5
  gather_facts: no
  tasks:
   - name: Configurar R5 estaticas
     ios_config:
      lines:
       - router eigrp 1
       - redistribute ospf 1 metric 100000 65000 255 255 10000
       - network 10.0.0.28 0.0.0.3
       - no auto-summary
       - router ospf 1
       - router-id 5.5.5.5
       - redistribute eigrp 1 metric 100000 subnets
       - network 10.0.0.20 0.0.0.3 area 1
      timeout: 100
```

Archivo configuraciones R6_6C.yml

```
---
- name: R6_6C
  hosts: R6
  gather_facts: no
  tasks:
   - name: Configurar R6 estaticas
     ios_config:
      lines:
```

```
      - router eigrp 1
      - redistribute ospf 1 metric 100000 65000 255 255 10000
      - network 10.0.0.24 0.0.0.3
      - no auto-summary
      - router ospf 1
      - router-id 6.6.6.6
      - redistribute eigrp 1 metric 100000 subnets
      - network 10.0.0.32 0.0.0.3 area 2
     timeout: 100
```

Archivo configuraciones R7_7C.yml

```
---
- name: R7_7C
  hosts: R7
  gather_facts: no
  tasks:
   - name: Configurar R7 estaticas
     ios_config:
      lines:
       - router eigrp 1
       - redistribute ospf 1 metric 100000 65000 255 255 10000
       - network 10.0.0.28 0.0.0.3
       - no auto-summary
       - router ospf 1
       - router-id 7.7.7.7
       - redistribute eigrp 1 metric 100000 subnets
       - network 10.0.0.36 0.0.0.3 area 2
      timeout: 100
```

Archivo configuraciones R8_8C.yml

```
---
- name: R8_8C
  hosts: R8
  gather_facts: no
  tasks:
   - name: Configurar R8 estaticas
     ios_config:
      lines:
       - router-id 8.8.8.8
       - network 10.0.0.32 0.0.0.3 area 2
       - network 10.0.0.36 0.0.0.3 area 2
      timeout: 100
```

Archivo configuraciones R1_NS.yml

```
1|
```

Archivo configuraciones R2_NS.yml

```
---
- name: R2_NS
  hosts: R2
  gather_facts: no
  tasks:
   - name: Configurar R2 estaticas
     ios_config:
      lines:
       - no ip route 0.0.0.0 0.0.0.0 10.0.0.10
      timeout: 100
```

Archivo configuraciones R3_NS.yml

```
---
- name: R3_NS
  hosts: R3
  gather_facts: no
  tasks:
   - name: Configurar R3 estaticas
     ios_config:
      lines:
       - no ip route 0.0.0.0 0.0.0.0 10.0.0.13
      timeout: 100
```

Archivo configuraciones R4_NS.yml

```
---
- name: R4_NS
  hosts: R4
  gather_facts: no
  tasks:
   - name: Configurar R4 estaticas
     ios_config:
      lines:
       - no ip route 0.0.0.0 0.0.0.0 10.0.0.26
      timeout: 100
```

Archivo configuraciones R5_NS.yml

```
---
- name: R5_NS
  hosts: R5
  gather_facts: no
  tasks:
```

```
  - name: Configurar R5 estaticas
    ios_config:
     lines:
      - no ip route 0.0.0.0 0.0.0.0 10.0.0.21
     timeout: 100
```

Archivo configuraciones R6_NS.yml

```
---
- name: R6_NS
  hosts: R6
  gather_facts: no
  tasks:
   - name: Configurar R6 estaticas
     ios_config:
      lines:
       - no ip route 0.0.0.0 0.0.0.0 10.0.0.34
      timeout: 100
```

Archivo configuraciones R7_NS.yml

```
---
- name: R7_NS
  hosts: R7
  gather_facts: no
  tasks:
   - name: Configurar R7 estaticas
     ios_config:
      lines:
       - no ip route 0.0.0.0 0.0.0.0 10.0.0.29
      timeout: 100
```

Archivo configuraciones R8_NS.yml

```
---
- name: R8_NS
  hosts: R8
  gather_facts: no
  tasks:
   - name: Configurar R8 estaticas
     ios_config:
      lines:
       - no ip route 0.0.0.0 0.0.0.0 10.0.0.37
      timeout: 100
```