ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELECTRÍCA Y ELECTRÓNICA

ANÁLISIS DE PROTOCOLOS PARA LA GESTIÓN DE REDES

ANÁLISIS DEL PROTOCOLO DE GESTIÓN DE RED RESTCONF

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN TELECOMUNICACIONES

ALEX GEOVANNY REMACHE CHIMBOLEMA

alex.remache@epn.edu.ec

DIRECTOR: ING. XAVIER ALEXANDER CALDERÓN HINOJOSA M.Sc.

xavier.calderon@epn.edu.ec

DMQ, octubre 2022

CERTIFICACIONES

Yo, ALEX GEOVANNY REMACHE CHIMBOLEMA 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.

ALEX GEOVANNY REMACHE CHIMBOLEMA

Certifico que el presente Trabajo de Integración Curricular fue desarrollado por ALEX GEOVANNY REMACHE CHIMBOLEMA, bajo mi supervisión.

ING. XAVIER ALEXANDER CALDERÓN HINOJOSA M.Sc.
DIRECTOR

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.

ALEX GEOVANNY REMACHE CHIMBOLEMA

ING. XAVIER ALEXANDER CALDERÓN HINOJOSA M.Sc.

DEDICATORIA

“Yo te alabaré siempre delante de tus fieles, porque has actuado en mi favor.
Por siempre confiaré en Tu Nombre, porque es bueno confiar en Ti.”
-Libro de los Himnos, sección II, capítulo LII, versículo 9. RVC.

Dedico este Trabajo de Integración Curricular,

A Dios, trino y uno, creador de todo lo visible e invisible, inicio y fin de mis alegrías y tristezas, éxitos y fracasos, fuente de conocimiento, verdad, vida y fe.

A mis amados padres, José Remache Paují y Blanca Chimbolema Chimbolema, a quienes, en este trabajo, presento mi admiración, respeto y amor, pues a pesar de los difíciles momentos y la falta de oportunidades que se presentó en la vida siempre me dieron lo mejor, por eso dedico el cumplimiento de esta meta a ellos.

A mis queridos hermanos, Katherin, Alejandro y Matías Remache Chimbolema.

A mis abuelitos Vicente Remache Guamán (+) y Juana Paují Maji, Manuel Chimbolema Mendoza y Juana Chimbolema Asitimbay (+).

A toda mi familia.

A mis amigos.

A mis hermanos.

Al pueblo Kichwa Puruhá.

Alex G. Remache Ch.

Quito, octubre 2022

AGRADECIMIENTO

“He aquí, Yo estoy contigo, y te guardaré por dondequiera que fueres, y volveré a traerte a esta tierra; porque no te dejaré hasta que haya hecho lo que te he dicho.”
-Libro de los Orígenes, capítulo XXVIII, versículo 15. RVR60.

Gracias a Dios, trino y uno, por la promesa que me dio antes de venir a estudiar a Quito. Gracias por haberme cuidado y darme el entendimiento y las fuerzas suficientes para seguir.

Gracias a mi familia, a mi padre José y a mi madre Blanca, por todo su apoyo, soporte y oraciones. No sé qué sería de mí sin ustedes. Gracias por permitirme venir a estudiar de Ibarra a Quito. A mi hermana Katherin, compañera de toda la vida, a mis hermanos José Alejandro y José Matías, que con su nacimiento en 2019 me han dado una razón para vivir y salir adelante.

Gracias a toda mi familia extendida, porque me han hecho entender que debo llevar mi raíz indígena Kichwa Puruhá con orgullo a donde quiera que vaya. ¡Pagui tucui shunguan!

Gracias a la Escuela Politécnica Nacional por el privilegio de estudiar en sus aulas y por su apoyo económico. Siempre estaré orgullo de ser politécnico.

Gracias a la Facultad de Ingeniería Eléctrica y Electrónica y sus honorables profesores.

Gracias al M.Sc. Xavier Calderón por su guía en el desarrollo de este Trabajo de Integración Curricular.

Gracias al Manchester Team, por haberme enseñado que los verdaderos amigos existen. ¡No me arrepiento del año que estuve con ustedes!

A Jonathan, M. Sebastián y Dayanara, gracias por convertirse en mis amigos y apoyo en la carrera y por todas las anécdotas. A Francisco, L. Sebastián y Hernán. Gracias a mis amigos de la poli, han hecho de mi vida foránea más llevadera. ¡Éxitos, ingenieros!

Gracias a mis amigos de Ibarra.

Gracias a mis hermanos y hermanas, por sus oraciones y el apoyo que me han brindado.

Gracias a todas las personas que han confiado en mí.

Alex G. Remache Ch.

Quito, octubre 2022

ÍNDICE DE CONTENIDO

CERTIFICACIONES II
DECLARACIÓN DE AUTORÍA III
DEDICATORIA IV
AGRADECIMIENTO V
ÍNDICE DE CONTENIDO VI
RESUMEN VIII
ABSTRACT IX
1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO 1
1.1 Objetivo general 2
1.2 Objetivos específicos 2
1.3 Alcance 2
1.4 Marco teórico 3
1.4.1 Protcolo RESTCONF 3
1.4.1.1 Definición del protocolo 3
1.4.1.2 Arquitectura de red 4
1.4.1.3 Lenguaje de Modelado de Datos YANG 5
1.4.1.4 Formato de codificación 5
1.4.1.4.1 XML 5
1.4.1.4.2 JSON 6
1.4.1.5 Mensajes HTTP 6
1.4.1.5.1 Mensajes de solicitud 6
1.4.1.5.2 Mensajes de respuesta 8
1.4.2 Comparación con SNMP 10
1.4.3 Técnicas experimentales en redes 10
1.4.4 Equipos 12
2 METODOLOGÍA 13
2.1 Selección de Herramientas 13
2.2 Temas de las prácticas de laboratorio 15
2.2.1 Práctica 1: Introducción a RESTCONF 15
2.2.2 Práctica 2: Lenguaje de Modelado de datos YANG 16
2.2.3 Práctica 3: Seguridad y Métodos HTTP 16

2.2.4 Práctica 4: Gestión RESTCONF con Postman 18
2.2.5 Práctica 5: Gestion RESTCONF con Python 19
2.3 Estructura de las prácticas de laboratorio 20
2.4 Prácticas de laboratorio 21
2.4.1 Práctica 1 21
2.4.2 Práctica 2 35
2.4.3 Práctica 3 36
2.4.4 Práctica 4 36
2.4.5 Práctica 5 36
2.5 Material complementario 36
3 RESULTADOS, CONCLUSIONES Y RECOMENDACIONES 37
3.1 Resultados 37
3.1.1 Práctica 1: Informe 37
3.1.2 Práctica 2: Informe 39
3.1.3 Práctica 3: Informe 41
3.3.4 Práctica 4: Informe 43
3.1.5 Práctica 5: Informe 43
3.2 Conclusiones 45
3.3 Recomendaciones 47
4 REFERENCIAS BIBLIOGRÁFICAS 48
5 ANEXOS 51

RESUMEN

El desarrollo de las telecomunicaciones ha sido, en los últimos años, exponencial, teniendo, hoy en día, grandes sistemas interconectados a nivel mundial, dando paso a las redes virtuales, de nube, aprendizaje automático, etc. Esta creciente demanda, obliga a los administradores de red, a gestionar de manera eficaz las grandes redes, por lo que, es necesario pasar de los protocolos de gestión tradicionales, como SNMP, a los basados en modelos de datos, como el lenguaje YANG. Esto permite una mayor facilidad de gestión y mantenimiento de redes, mediante programabilidad y automatización.

Uno de los protocolos recientes que utilizan el modelo de datos YANG para gestión de redes, es el protocolo RESTCONF, que surge de la necesidad de aplicar el protocolo NETCONF a sistema de aplicaciones web e interfaces programáticas. Esto ha permitido que RESTCONF sea NETCONF basado en HTTP, con todas las ventajas que esto conlleva, siendo uno de los protocolos de gestión con mayor proyección al futuro.

Este trabajo de integración curricular tiene como objetivo demostrar las propiedades de RESTCONF como protocolo cliente-servidor mediante la combinación de cinco prácticas de laboratorio, emulando redes en un entorno de GNS3, creadas según el formato establecido por la Escuela Politécnica Nacional y el Departamento de Electrónica, Telecomunicaciones y Redes de Información. Cada ejercicio de laboratorio consta de actividades que permiten a los estudiantes de Ingeniería en Telecomunicaciones y profesiones afines comprender, aprender y aplicar la funcionalidad proporcionada por RESTCONF.

PALABRAS CLAVE: GNS3, Laboratorio, Práctica, RESTCONF, YANG.

ABSTRACT

In recent years, the development of telecommunications has been exponential, so today, we have large interconnected systems worldwide, giving way to virtual and cloud networks, machine learning, etc. This growing demand forces network administrators to effectively manage large networks, so it is necessary to move from traditional management protocols, such as SNMP, to those based on data models, such as the YANG language. This allows greater ease of network management and maintenance, through programmability and automation.

One of the recent protocols that uses the YANG data model for network management is the RESTCONF protocol, which arises from the need to apply the NETCONF protocol to web application systems and programmatic interfaces. This has allowed RESTCONF to be NETCONF based on HTTP, with all the advantages that this entails, being one of the management protocols with the greatest future projection.

This curricular integration work aims to demonstrate the properties of RESTCONF as a client-server protocol by combining five laboratory practices, emulating networks in a GNS3 environment, created according to the format established by the National Polytechnic School and the Department of Electronics, Telecommunications and Information Networks. Each laboratory exercise consists of activities that allow students of Telecommunications Engineering and related professions to understand, learn and apply the functionality provided by RESTCONF.

KEYWORDS: GNS3, Laboratory, Practices, RESTCONF, YANG.

1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO

El presente proyecto tiene como finalidad realizar un análisis del protocolo de gestión de redes RESTCONF (*Representational State Transfer Configuration Protocol*); ya que permite aprovechar tecnologías como el Internet de las Cosas, plataformas en la nube, redes definidas por software y principalmente, permite a los dispositivos exponer una API (*Application Programming Information*), la cual es utilizada por las aplicaciones para enviar y recibir configuraciones completas o parciales; con lo cual puede usarse la programación abierta (con Python) para gestionar las plataformas de redes actuales [1], [2], [3], [4], [5].

La gestión de las redes es muy importante en el mundo actual, pues al estar todo interconectado mediante equipos y plataformas de red, es necesario tener protocolos amigables con el usuario para que éste puede operar y gestionar las redes de mejor manera, llegando a la programabilidad y automatización de las redes y de su monitoreo [6]. En la historia de la gestión de redes, se tiene a SNMP (*Simple Network Management Protocol*) como el protocolo más utilizado para este fin, pero debido a la gran escala y complejidad que han llegado a tener los sistemas, SNMP no ha cumplido con la demanda de operación y mantenimiento de redes automatizadas; por lo que, ha sido necesario el desarrollo de un nuevo protocolo de gestión de redes como NETCONF (*Network Configuration*) que proporciona una interfaz de programación de aplicaciones. Sin embargo, NETCONF no ha evolucionado en el desarrollo de redes, pues se requiere interfaces programáticas estándar que admitan el acceso a aplicaciones web y las operaciones de dispositivos de red [7].

Observando esta necesidad, se tiene al protocolo RESTCONF que se desarrolla en base a la combinación de NETCONF y HTTP. RESTCONF proporciona funciones básicas de NETCONF utilizando métodos HTTP. La interfaz programática cumple con el estilo RESTful de la industria de TI (Tecnologías de la Información) ofreciendo a los usuarios la capacidad de desarrollar herramientas eficientes de operación y mantenimiento web [7]. Por esto, para este Trabajo de Integración Curricular se ha propuesto el análisis del protocolo para la Gestión de Redes RESTCONF.

Por lo cual, en este Trabajo de Integración Curricular se realizará una serie de prácticas de laboratorio que permitan demostrar su utilidad y estén dirigidas a los estudiantes de la carrera de Telecomunicaciones y afines, y de esta manera prepararlos para el mundo laboral, pues, como se ha visto, este protocolo será en mediano plazo la herramienta para gestionar, automatizar y programar redes, así como utilizado en muchas áreas de TI.

1.1 Objetivo general

Analizar el protocolo RESTCONF para la gestión de redes utilizando simuladores/emuladores de equipos de red.

1.2 Objetivos específicos

1. Estudiar los fundamentos del protocolo RESTCONF.
2. Desarrollar prácticas de laboratorio con los simuladores/emuladores.
3. Analizar los resultados obtenidos con el uso de simuladores.

1.3 Alcance

El proyecto constará del estudio general del protocolo RESTCONF, fundamentos, aplicaciones y configuración en equipos de red que soporten tal protocolo. Para esto se realizará una comparación con el protocolo de gestión de red SNMP (Simple Network Management Protocol) y una serie de prácticas de laboratorio que nos permitirá conocer las aplicaciones del protocolo. Este proyecto consta de tres fases que son:

Fase de planteamiento

Se estudiarán los fundamentos del protocolo RESTCONF, establecido en el RFC 8040, para la gestión de redes IP que permita a los administradores gestionar los dispositivos de red, supervisar el funcionamiento de la red, diagnosticar y resolver problemas y planificar su crecimiento.

Para esto se realizará una comparativa entre el protocolo RESTCONF con el protocolo de gestión de redes más utilizado en la actualidad, el protocolo SNMP (Simple Network Management Protocol); un breve estudio acerca de los simuladores y emuladores que tienen la capacidad de soportar la configuración de este protocolo y se realizará una compilación de los equipos que soportan RESTCONF dentro de los tres proveedores de equipos de red más importantes Cisco, Juniper y Huawei.

Fase de implementación

Se realizará una serie de 5 prácticas de laboratorio para el estudio de sus aplicaciones y características; que constarán con preparatorios, informes, reactivos, video de implementación, configuraciones y uso de simuladores y/o emuladores que permitan la ejecución de estos protocolos. Las prácticas por realizar son las siguientes:

1. Introducción a RESTCONF.
2. Lenguaje de Modelado de Datos YANG.
3. Seguridad y Métodos HTTP.
4. Gestión RESTCONF con Postman.
5. Gestión RESTCONF con Python.

Fase de análisis de resultados

Al final de una serie de ejercicios de laboratorio, se realizan pruebas y se analizan los resultados obtenidos. El documento final del trabajo de integración curricular se elabora de acuerdo con los lineamientos de la unidad académica. Adicionalmente, las cinco prácticas están escritas en un formato establecido por el Departamento de Electrónica, Telecomunicaciones y Redes de Información. Finalmente, se presenta las conclusiones y recomendaciones.

1.4 Marco teórico

1.4.1 Protocolo RESTCONF

1.4.1.1 Definición del protocolo

RESTCONF (*Representational State Transfer Configuration Protocol*) es un protocolo basado en HTTP (*Hypertext Transfer Protocol*) o HTTPS, que provee interfaces programáticas RESTful y permite al usuario añadir, eliminar, modificar y acceder a los datos de los dispositivos de red, por lo que, es un protocolo para gestión y monitoreo de redes [7].

RESTCONF utiliza los fundamentos del protocolo NETCONF (*Network Configuration*) pero utilizando el estilo arquitectónico REST, pues en los últimos años ha sido ampliamente utilizado como un mecanismo RPC (*Remote Procedure Call*). REST es bastante simple y tiene una amplia gama de herramientas y enlaces para todos los lenguajes de programación populares [8]. Por esto, existe una gran demanda, de tener el protocolo NETCONF sobre REST, haciendo que la IETF (*Internet Engineering Task Force*) defina el protocolo RESTCONF en el RFC 8040 desde junio de 2017 [9].

El protocolo RESTCONF utiliza datos estructurados (XML o JSON) y YANG para proporcionar APIs (*Application Programming Information*) similares a REST, lo que le

permite acceder mediante programación a diferentes dispositivos de red. Las API RESTCONF utilizan métodos HTTP [10].

1.4.1.2 Arquitectura de red

El protocolo RESTCONF tiene una arquitectura de red cliente-servidor RESTCONF como se muestra en la Figura 1.1. Los principales elementos son [7]:

- **Cliente RESTCONF:** El cliente utiliza NETCONF para administrar dispositivos de red. Los datos que un cliente obtiene de un servidor RESTCONF en ejecución incluyen la consulta de datos de configuración y datos de estado.

  El cliente puede modificar y operar los datos de configuración, para que el estado del servidor RESTCONF migre a un estado esperado por el usuario.

  El cliente no puede modificar los datos de estado, estos incluyen el estado de ejecución del servidor RESTCONF y otras estadísticas.

- **Servidor RESTCONF:** El servidor mantiene los dispositivos de red administrados, responde a las solicitudes de los clientes e informa los datos de administración al cliente. Después de recibir una solicitud enviada por el cliente, el servidor analiza la solicitud, la procesa y envía una respuesta al cliente.

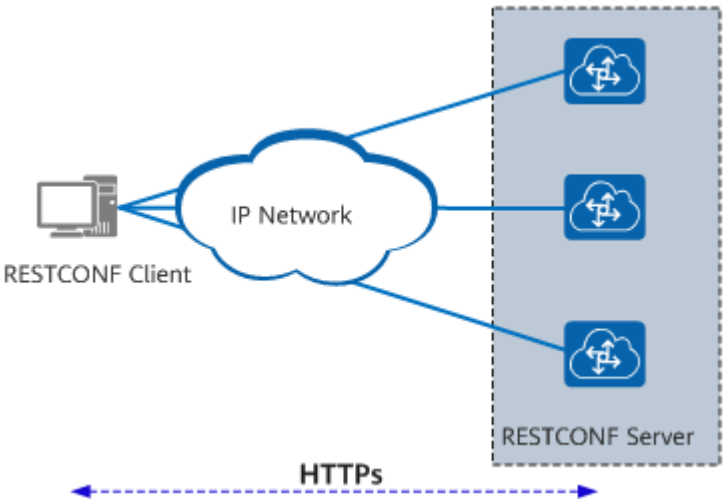

Figura 1.1. Arquitectura de red RESTCONF [7].

El cliente y el servidor RESTCONF ejecutan HTTPS para establecer una sesión segura y orientada a la conexión. El cliente envía una solicitud al servidor. Después de procesar la solicitud del usuario, el servidor responde con una respuesta al cliente. Los mensajes de solicitud y respuesta están codificados en formato XML o JSON [7].

1.4.1.3 Lenguaje de Modelado de Datos YANG

YANG (*Yet Another Next Generation*) es un lenguaje de modelado de datos que precisa una jerarquía sobre los datos que pueden ser utilizados para operaciones base que pueden incluir configuración, estado de datos, llamadas a procedimientos remotos RPC (*Remote Procedure Call*) y notificaciones. Esto admite una completa descripción sobre todos los datos que son enviados entre un cliente y un servidor NETCONF. El lenguaje de modelado de datos YANG fue desarrollado por la IETF (*Internet Engineering Task Force*) y se publicó en 2010 como RFC 6020 [11].

YANG es un lenguaje modular que representa estructuras de datos en un formato de árbol. El lenguaje de modelado de datos contiene una serie de tipos de datos intrínsecos, como cadenas y números enteros. Estos tipos básicos pueden usarse para construir tipos más complejos formando un conjunto muy rico de capacidades. YANG suministra una clara descripción de los nodos y precisa como se interactúa entre ellos estructurando a los modelos en módulos y submódulos [12].

Se debe recordar que el protocolo RESTCONF trabaja sobre los fundamentos de NETCONF sobre HTTPS, por lo que RESTCONF también trabaja con el lenguaje de modelado de datos YANG y con los tipos de datos estructurados XML y JSON [10].

1.4.1.4 Formato de codificación

1.4.1.4.1 XML

Según [13], XML (*Extensible Markup Language*) es un lenguaje de marcas como HTML, pero diseñado para transportar, organizar y almacenar datos, más no mostrarlos, esto permite el uso de un archivo de texto para representar datos jerárquicos complejos. En el lenguaje XML, se definen etiquetas propias y tiene la extensión .xml. El Código 1.1. es una muestra de ejemplo.

```
<?xml version="1.0"?>
<rpc message-id="17">
  <what>ping</what>
  <who>10.20.30.40</who>
  <quick/>
</rpc>
```

Código 1.1. Ejemplo de un documento XML que describe un RPC [8]

1.4.1.4.2 JSON

Según [14], JSON (*JavaScript Object Notation*) es un formato ligero de intercambio de datos textuales. JSON usa la sintaxis de JavaScript para describir objetos de datos, pero sigue siendo independiente del lenguaje y la plataforma. JSON es similar a XML, pero JSON es más corto y fácil de analizar que XML. Código 1.2. es un ejemplo típico.

```
var objeto1 = {
  nombre: "Computadora"
  color: "Gris"
  modelo: "Laptop"
}
```

Código 1.2. Ejemplo de un documento JSON que describe un objeto [7]

1.4.1.5 Mensajes HTTP

HTTP es un protocolo de solicitud-respuesta de texto sin formato. Un cliente envía una solicitud HTTP a un servidor y recibe una respuesta HTTP del servidor.

1.4.1.5.1 Mensajes de solicitud

Un mensaje de solicitud (*Request*) HTTP siempre debe tener tres componentes que son [5]:

- Una línea de solicitud (*Request line*) que especifica el método de solicitud (*Request method*) y el recurso al que se accede, identificado por un identificador de recurso (ID).
- Campos de encabezado (*Request header*) que especifiquen la metainformación que debe seguir el servidor.
- Un cuerpo de solicitud (*Request data*) opcional que está separado de los campos del encabezado de la solicitud por una nueva línea.

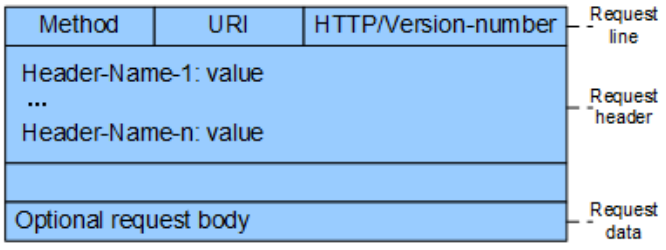

Figura 1.2. Formato de solicitud HTTP [7]

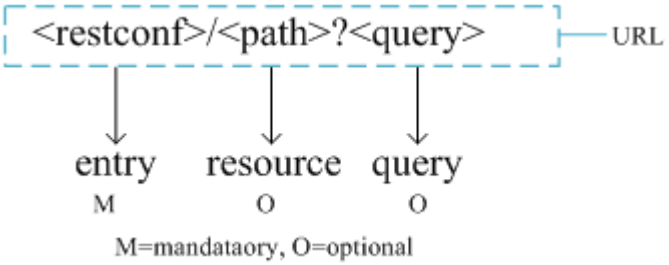

Figura 1.3. Formato URI [7]

Un método de solicitud (*Request method*) es una palabra clave específica que describe el tipo de operación que queremos que realice el servidor al recibir y procesar una solicitud HTTP. Los métodos de solicitud más importantes se pueden observar en la Tabla 1.1.

Tabla 1.1. Métodos de solicitud

Opción	Método
GET	Leer información.
PATCH	Sobrescribir o modificar los recursos.
PUT	Reemplazar el recurso.
POST	Crear o actualizar un recurso.
DELETE	Eliminar el recurso.

Los campos de encabezado (*Request header*) de nuestra solicitud HTTP se utilizan para especificar la metainformación que queremos dar a nuestro servidor al procesar la solicitud. Los tipos más comunes de información enviada dentro de un encabezado se describe en la Tabla 1.2.

Tabla 1.2. Información en el campo de encabezado

Elemento	Información
Content-Type:	Tipo de datos (JSON o XML) que se envían.
Accept:	Tipo de datos (JSON o XML) que se desea recibir.
Authorization:	Proporciona credenciales de autenticación, como nombre de usuario/contraseña o tokens API.
Content-Length:	Tamaño del cuerpo de nuestra solicitud en octetos.
Host:	Especifica el host (y, opcionalmente, el puerto) en el que escucha el servidor web.

El Código 1.3 es un ejemplo de un mensaje de solicitud HTTP, que consiste en un método POST al recurso `/devices`, que envía y pide recibir respuesta en JSON, este se envía a un servidor que se ejecuta en el puerto `8080` del host `data.com`. Este ejemplo, le dice al servidor que cree un recurso con los datos dados en el cuerpo de la solicitud.

```
POST /devices HTTP/1.1
Host: data.com:8080
Content-Type: application/json
Content-Length: 63
Accept: application/json
Authorization: Basic cm9vdDpwYXNzd29yZA==

{
  "name": "my device",
  "mac": "01-23-45-67-89-AB-CD-EF"
}
```

Código 1.3. Ejemplo de un mensaje de solicitud HTTP [5]

1.4.1.5.2 Mensajes de respuesta

Cuando el cliente ha enviado una solicitud, el servidor tiene la obligación de enviar una respuesta. La Figura 1.4. muestra el formato de mensaje de respuesta (*Response*) y la Tabla 1.3. la información de sus componentes.

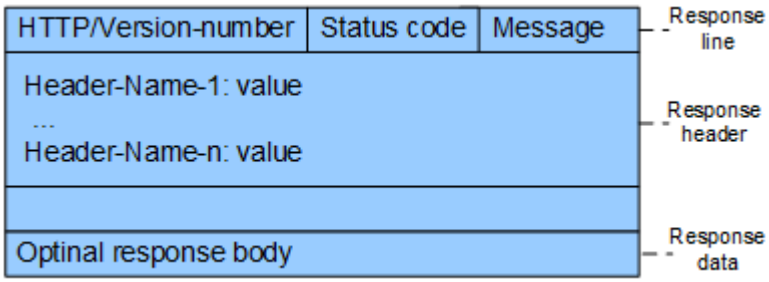

Figura 1.4. Formato de respuesta HTTP [7]

Tabla 1.3. Información del mensaje de respuesta HTTP

Componente	Información
HTTP/Version-number	Versión del protocolo HTTP.
Status code	Código de estado HTTP.
Message	Mensaje de estado HTTP.
Header-Name-n:value	Nombre del campo de encabezado: valor.
Optional request body	(Opcional) Cuerpo del mensaje de respuesta.

El Código 1.4. es un ejemplo de respuesta a la petición hecha por el cliente en el Código 1.3. Esta respuesta le dice que se ha creado con éxito el recurso y le ha enviado la respuesta en JSON, la operación ha sido un ID=25.

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8

{
  "id": 25,
  "name": "my device",
  "mac": "01-23-45-67-89-AB-CD-EF"
}
```

Código 1.4. Ejemplo de un mensaje de respuesta HTTP [5]

Uno de los mensajes importante de la respuesta HTTP, es el código de estado, que nos da información acerca de las respuestas dadas. En la Tabla 1.4. se indica los más importantes [5].

Tabla 1.4. Códigos de estado más habituales

Categorías	Respuestas	Código	Estado
1xx	Informativas		
2xx	Exitosas	200-OK	Información devuelta (GET, PUT, PATCH)
		201-Created	Información creada (POST)
		204-No Content	Información eliminada (DELETE)
3xx	Redirigidas	301-Moved permanently	La solicitud debe ser redirigida.
4xx	Error del cliente	400-Bad request	El servidor no pudo comprender la solicitud (falta de datos o formato erróneo).
		401-Unauthorized	Falta de autenticación.
		403-Forbidden	Sin permisos
		404-Not Found	El recurso solicitado no se pudo encontrar en el servidor.
		405-Method Not Allowed	La solicitud utilizó un método que no está permitido para este recurso específico
5xx	Error del servidor	500-Internal Server error	El servidor ha encontrado algún tipo de error en él.
		502-Bad Gateway	El servidor tuvo problemas para comunicarse con otro servidor ascendente.
		503-Service Unavailable	El servidor actualmente no puede manejar la solicitud.

1.4.2 Comparación con SNMP

SNMP (*Simple Network Management Protocol*) es un protocolo aprobado por la IAB (*Internet Architecture Board*) in 1988 como un protocolo estándar para la gestión de redes IP, llegando a ser uno de los más extendidos, teniendo hasta la actualidad tres versiones.

Consiste en un simple protocolo de solicitud/respuesta que corre sobre UDP (*User Datagram Protocol*). Una de las características que no permite que este protocolo sea utilizado para la gestión de redes masivas y automatizadas, es su alto tiempo de procesamiento, y no configuración de equipos [1]. En la Tabla 1.5. se muestra una comparación entre el protocolo de este estudio RESTCONF, NETCONF y SNMP.

Tabla 1.5. Comparación entre SNMP, NETCONF y RESTCONF

Componente	SNMP v1, v2 y v3	NETCONF	RESTCONF
Estándar	IETF	IETF	IETF
Lenguaje de definición	SMIv2	YANG	YANG
Modelo de información	Módulos MIB	Módulos YANG	Módulos YANG
Lenguaje de transferencia	ASN.1 BER (Basic Encoding Rules)	XML	XML o JSON
Soportado sobre	UDP	SSH/SSL	HTTP/HTTPS/TLS
Seguridad	VACM/USM (v3)	Sí	Sí
Gestión de red	Solo recupera información	Recupera información y configura equipos.	

1.4.3 Técnicas experimentales en redes

En [15], se manifiesta que existen tres tipos de técnicas experimentales usadas para el diseño y validación de nuevas arquitecturas y modelos de redes, esto es de gran ayuda pues permite aprender, experimentar antes de hacerlos en equipos reales, observar el comportamiento de la red y mejorarlos. Estos tres tipos son: la simulación, emulación y pruebas en redes reales. Cada técnica sirve para una necesidad y propósito diferente, no haciendo competencia entre ellas. En la Tabla 1.6. se mostrará las principales diferencias.

Tabla 1.6. Técnicas experimentales en redes

Característica	Simulación	Emulación
Entorno	Repetible y controlado.	No es repetible ni controlado.
Configuración	Elementos sintéticos.	Híbrido (Combina elementos reales con simuladores).
Requerimientos	Fácil de configurar e instalar.	Requiere mayores características para funcionar, configurar e instalar.
Objetivo	Evaluación y diseño.	Evalución y diseño.
Ejecución	Tiempo virtual simulado.	Tiempo real.
Ejemplos	Cisco Packet Tracer.	GNS3.

1.4.3.1 GNS3

GNS3 es un emulador de red gráfico multiplataforma que se ejecuta en Windows, OS X y Linux. Fue desarrollado por la colaboración de Christophe Fillot, Jeremy Grossmann y Julien Duponchelle [16]. GNS3 emula el hardware de un dispositivo y ejecuta las imágenes reales en el dispositivo virtual. Podemos usar el IOS de un enrutador Cisco físico real y ejecutarlo en un enrutador Cisco emulado virtual en GNS3 y emular las características y la funcionalidad de un dispositivo como un enrutador. Además del hardware emulado, GNS3 integra sistemas operativos simulados y se pueden conectar completamente en red a otros dispositivos GNS3 [17]. En Figura 1.5. se visualiza la capacidad el GNS3 en una red simple.

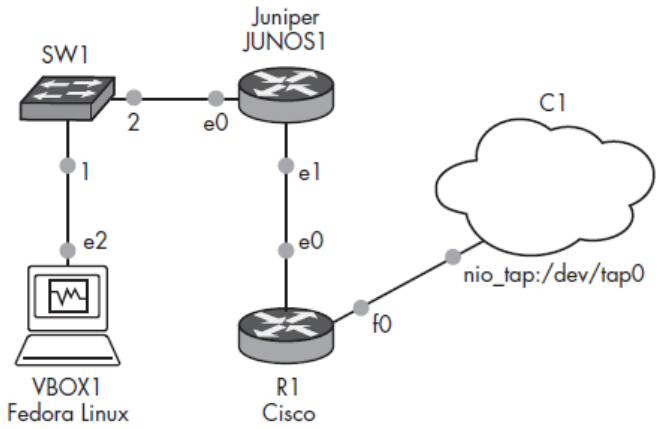

Figura 1.5. Una topología GNS3 que integra Fedora y routers Cisco y Juniper [16]

1.4.4 Equipos

1.4.4.1 Cisco

Según [18], Cisco tiene los siguientes equipos se soportan RESTCONF.

Tabla 1.7. Equipos Cisco

IOS	Equipos	Característica
XE 16.6	ISR 4000 Series	Router de servicios de agregación.
	CSR 1000v	Router de servicios en la nube.
	ASR 1000 Series	Router de servicios de agregación.
	ISRv	Router virtual de servicios integrados.
XE 16.7	ASR 900 Series	Router de servicios de agregación.
	NCS 4200 Series	Sistema de convergencia de red.
XE 16.8	Catalyst 3650, 3850, 9000	Switches
	ISR 1000 Series	Router de servicios integrados.
XE 16.10	IR 1101 Series	Enrutador de servicios integrados resistente.
XE 16.11	IE 3x00 Series	Switches de la serie robusta Catalyst.

1.4.4.2 Juniper

Según [19], [20], [21], Juniper utiliza RESTCONF en los siguientes equipos.

Tabla 1.8. Equipos Juniper

Equipos	Característica
NorthStar Controller	Controladora de redes / Planificador 5.1.0
M Series	Router perimetrales multiservicio
MX Series	Plataformas de enrutamiento universal
T Series	Router de núcleo
PTX Series	Router de transporte de paquetes

1.4.4.3 Huawei

Según [2], [7], [22], [23] Huawei tiene los siguientes equipos de red que trabajan con RESTCONF.

Tabla 1.9. Equipos Huawei

Equipos	Característica
iMaster NCE-Campus	Sistema de Gestión y Control
CloudEngine 12800&16800	Switch de Centro de Datos
NetEngine AR800	Router de acceso
USG6620/6630	Firewall y Gateway VPN

2 METODOLOGÍA

Para el presente Trabajo de Integración Curricular la metodología a emplearse es la investigación explicativa con el método experimental, ya que se utilizará el software GNS3 que simulará/emulará las redes descritas en los siguientes laboratorios; además, VirtualBox y VMware Workstation Pro como softwares de virtualización y DEVASC-LABVM como una máquina virtual Ubuntu de la distribución Linux.

Para el desarrollo de este Trabajo de Integración Curricular se siguió el siguiente enfoque de desarrollo por etapas:

1. **Investigación bibliográfica:** Reúne y selecciona información a través de referencias bibliográficas como textos, artículos, normas y recursos informáticos confiables.
2. **Selección de herramientas:** Pruebas de diferentes herramientas de simulación o emulación, selección de soluciones de red utilizadas en la práctica y elección de equipos de red utilizados.
3. **Selección de temas y objetivos**: Información y métodos para subdividir los temas y objetivos a abordar en la práctica de laboratorio.
4. **Desarrollo del contenido de las prácticas**: Generar los procedimientos a seguir en cada práctica. Se desarrollan enunciados, problemas y ejemplos de aplicación.
5. **Diseño de las prácticas**: Generación de Estructuras de Prácticas de Laboratorio. Se creó cada elemento establecido en el formato de práctica de laboratorio del Departamento de Electrónica, Telecomunicaciones y Redes de Información de la Escuela Politécnica Nacional.
6. **Análisis de resultados**: Pruebas de funcionamiento del procedimiento de cada una de las prácticas y resolución de informes.

2.1 Selección de Herramientas

La topología básica de las prácticas de laboratorio se muestra en la Figura 2.1, que dentro de la arquitectura del protocolo RESTCONF, la máquina virtual hace la función de cliente y el dispositivo de red, en este caso el router, hace la función del servidor. A lo largo, de las prácticas, las redes se volverán más complejas.

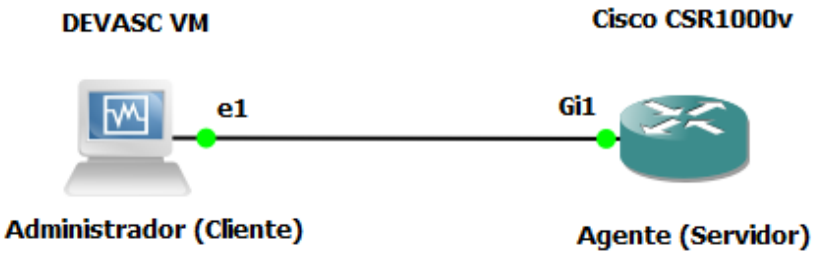

Figura 2.1. Esquema de red propuesto

Debido a que RESTCONF es un protocolo relativamente nuevo, solo es soportado por algunos dispositivos de red, se ha seleccionado el programa GNS3 para realizar las simulaciones y dentro de este ambiente se hizo la tarea de buscar aquellos dispositivos de red en la lista de equipos permitidos en GNS3, también denominados *Appliances*. De la lista se escogió el router Cisco CSR1000v IOS XE 17.3.2 como servidor RESTCONF.

Al utilizar equipos con servicios integrados demanda para su simulación de dispositivos con altos requerimientos de hardware y software. Para la emulación del router Cisco CSR1000v IOS XE 17.3.2 se requiere un archivo de imagen de disco en formato QEMU (Quick Emulator), ya que GNS3 utiliza el emulador QEMU para arrancar el sistema operativo del router Cisco CSR1000v, la extensión actual del archivo de imagen de disco .qcow2, que representa la segunda versión de la copia en escritura de QEMU. De acuerdo con [24], para instalar la imagen Cisco IOS XE 17.3.2 en un entorno Linux/KVM (Máquina virtual basada en kernel), se requiere una máquina virtual y usar archivos de tipo .iso o .qcow2, ya que los requisitos mínimos de instalación son 1 vCPU, en menos 4 GB de RAM y 8 GB de tamaño de disco duro virtual. Para ello, es necesario integrar GNS3 con su máquina virtual, GNS3 VM, para que las imágenes, contenedores y máquinas virtuales puedan ejecutarse de manera más eficiente.

La ventaja de GNS3 es que el programa se puede utilizar para la emulación de máquinas virtuales. Por esta razón, se eligió la máquina virtual Cisco DEVASC como el cliente principal de RESTCONF, ya que proporciona una herramienta útil para la práctica de laboratorio. Según Cisco, el único requisito para usar DEVASC es que la computadora host tenga al menos 4 GB de RAM y 15 GB de espacio libre en el disco duro.

2.2 Temas de las prácticas de laboratorio

Las prácticas de laboratorio tienen temas concernientes a la introducción al protocolo RESTCONF, el modelo de datos YANG, seguridad y programabilidad de la gestión con diferentes herramientas tecnológicas. A continuación, se detalla los temas de las prácticas.

2.2.1 Práctica 1: Introducción a RESTCONF

Esta práctica, al ser la primera, determina y detalla el procedimiento para la implementación de la topología básica, dada en la Figura 2.2., pues tiene como objetivos configurar los dispositivos de red, habilitar el protocolo RESTCONF y la generación de una petición/respuesta en la topología cliente-servidor.

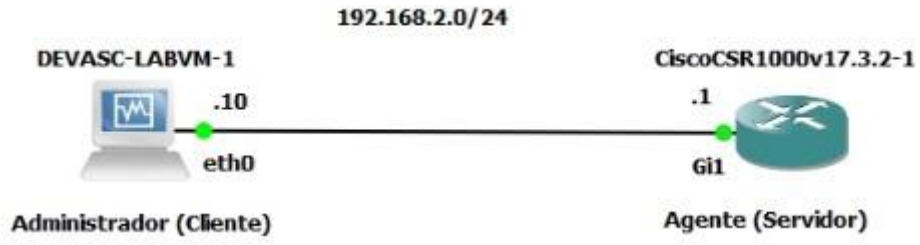

Figura 2.2. Topología de red a implementar en la práctica 1

Para la primera parte de la práctica, se ayuda al estudiante en las cuestiones básicas de la instalación del entorno de trabajo en GNS3, esto resulta de gran utilidad, pues será la base para todas las demás prácticas de laboratorio. En este paso, se tiene como resultado, la conectividad entre equipos e incluso con Internet.

El siguiente procedimiento consiste en la habilitación del protocolo RESTCONF para todas las interfaces del servidor dentro de la red 192.168.2.0/24, esto se configurará con los requerimientos necesarios para que el protocolo funcione de manera correcta.

Con lo realizado y configurado, en los pasos anteriores, en esta parte del laboratorio, se satisface el tercer objetivo de la práctica, al realizar la comprobación de la respuesta a una petición RESTCONF sobre HTTP con la herramienta cURL.

El estudiante antes de realizar la práctica tendrá que realizar un trabajo preparatorio que le ayude a establecer las bases y herramientas necesarias para cumplir con los objetivos de la práctica; así como también, al finalizar el laboratorio, demostrará lo desarrollado, responderá preguntas y dará conclusiones y recomendaciones.

2.2.2 Práctica 2: Lenguaje de Modelado de Datos YANG

El lenguaje de modelado de datos YANG organiza los datos, como información de enrutamiento, interfaces, estadísticas y protocolos, representándolas en una estructura principal de módulos, submódulos y contenedores que son una lista de subnodos relacionados, de esta caracterización de los datos, se basa el protocolo RESTCONF para la gestión de la red, por lo que es necesario tener sumo conocimiento de este Modelo de Datos.

Por lo que, para esta práctica, se tiene como objetivos entender y analizar el modelo de datos YANG, utilizar el comando para sistemas Linux `pyang` para identificar la estructura de datos y ejecutarlas en los formatos XML y JSON.

Antes de iniciar con el laboratorio, el estudiante realizará un trabajo preparatorio para sentar las bases de la práctica y así desarrollar la primera parte del laboratorio, que consiste en el estudio de la estructura y los elementos que tiene el modelo de datos YANG, para esto se ha utilizado los modelos de YANG que soporta el router Cisco con el software XE 17.3.1. Se estudiarán los modelos de datos YANG de tipo IETF y OPENCONFIG.

En la siguiente parte de la práctica, se utilizará la topología de la Figura 2.2. y consiste en entender de una manera más visual estos modelos de datos, desde el cliente se ejecutará el comando `pyang`. Esta herramienta nos ayuda a transformar los modelos de datos en formas más amigables con el usuario, como en la forma de árbol.

Por último, al estudiante se le dará modelos de datos YANG con la estructura de interfaces de un dispositivo de red en formatos XML y JSON, los dos formatos que soporta el protocolo RESTCONF, con el objetivo de establecer comparaciones y analizar cuál es el mejor lenguaje para la gestión con RESTCONF. Además de realizar las conclusiones, recomendaciones y respuestas necesarias para el informe final.

2.2.3 Práctica 3: Seguridad y Métodos HTTP

La tercera práctica corresponde a los niveles de seguridad que se debe tener para un protocolo de gestión y así proteger la red y su configuración mediante el uso de ACLs; además se realizará el uso de la herramienta cURL para estudiar la entrega y recepción de mensajes de solicitud y respuesta mediante los métodos HTTPS.

Los objetivos para este laboratorio son la implementación de las ACLs que permitirán el uso del protocolo a estudiar solo para alguna red o administrador especifico; el estudio del protocolo RESTCONF con los métodos HTTP mediante la herramienta cURL con sus

opciones, verificando la utilidad y eficacia de esta herramienta, y, por último, tener más conocimientos acerca de los métodos HTTPS y así diferenciarlos de acuerdo al requerimiento que se necesite configurar, modificar, establecer u obtener información de módulos o variables de la red.

Como prerrequisito para la realización del laboratorio, el estudiante consultará los comandos necesarios para crear ACLs y agregar al protocolo RESTCONF; además, de entender el funcionamiento de la herramienta cURL y sus principales opciones para utilizarlos en los métodos HTTP con sus encabezados. Para esto es esencial entender cómo crear URLs a partir del conocimiento de los módulos del modelo YANG IETF que el estudiante consultará.

Con estos conocimientos previos, se implementará la topología de la Figura 2.3., donde se observa que el agente tiene dos redes LAN, la red 192.168.2.0/24 y la 192.168.3.0/24 que tiene como hosts máquinas virtuales, configuradas según la topología dada, además, se configuran los puertos GigabitEthernet 1 y GigabitEthernet 2 del agente, como interfaces programáticas RESTCONF. Esta red nos permitirá simular las ACLs necesarias para establecer listas que solo permita la gestión de las redes, interfaces y estadísticas mediante RESTCONF en el administrador principal (192.168.2.10/24) generando mayor seguridad dentro de la red.

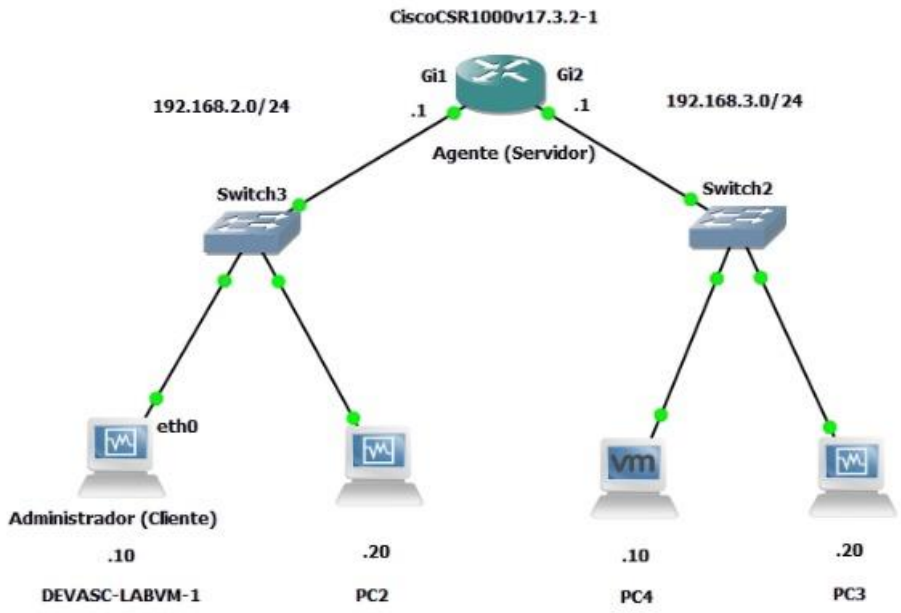

Figura 2.3. Topología de red a implementar en la práctica 3

Como segunda parte de la práctica de laboratorio se estudiarán las URLs que permitan configurar, crear y obtener información a partir de los módulos YANG de tipo IETF, con los

métodos HTTP utilizando cURL. Para estos mensajes de petición, con cURL se configurarán las opciones necesarias para tener los requerimientos necesitados en la práctica.

Con este procedimiento, el estudiante podrá realizar el informe final, estableciendo el nivel de compresión, el cumplimiento de los objetivos, la resolución de las preguntas dadas y sus conclusiones y recomendaciones.

2.2.4 Práctica 4: Gestión RESTCONF con Postman

Con el conocimiento de las prácticas previas, se establece que la gestión de las redes mediante el protocolo RESTCONF, es mejor con herramientas amigables con el usuario, de gran popularidad y fácil entendimiento, por lo que, para la cuarta práctica de laboratorio, se establece la gestión de la red mediante el programa Postman, ampliamente utilizado en el área de networking por el uso de las APIs, que en esencia son los requerimientos cURL con mayores facilidades de configuración.

Por lo que, se establece que los objetivos sean el estudio de Postman y mediante el uso de esa herramienta realizar la configuración de interfaces y rutas estáticas según los requerimientos de la red, así como el diagnosticar el estado de la red mediante APIs con RESTCONF.

Cómo instancia previa a la realización del laboratorio, se necesita que el estudiante conozca las principales áreas de trabajo de Postman, así como los módulos YANG necesarios para el cumplimiento de los objetivos que son: `ietf-restconf-monitoring.yang`, `ietf-interfaces.yang` y `ietf-routing.yang`. Al momento de realizar las peticiones HTTP, el servidor siempre dará un mensaje de respuesta con un código de estado, el estudiante debe saber cuáles son e identificar la respuesta mediante estos códigos, por lo que, se les pide estudiarlos.

En la etapa de procedimiento, el estudiante implementará la topología dada en la Figura 2.4. que consiste en 3 redes que son: LAN 1 (192.168.2.0/24), LAN 2 (192.168.3.0./24) y la red que une los dos routers (10.1.1.0/24). La topología será configurada de acuerdo con las solicitudes dadas en el procedimiento de la Hoja Guía 4. A continuación, con la ayuda del programa Postman, desde el administrador, el cliente podrá configurar la interfaz GigabitEthernet 2 del agente, recolectar información de sus interfaces y tablas de ruteo, crear nuevas interfaces Loopback y rutas estáticas, así como información estadística del estado de las interfaces y el enrutamiento.

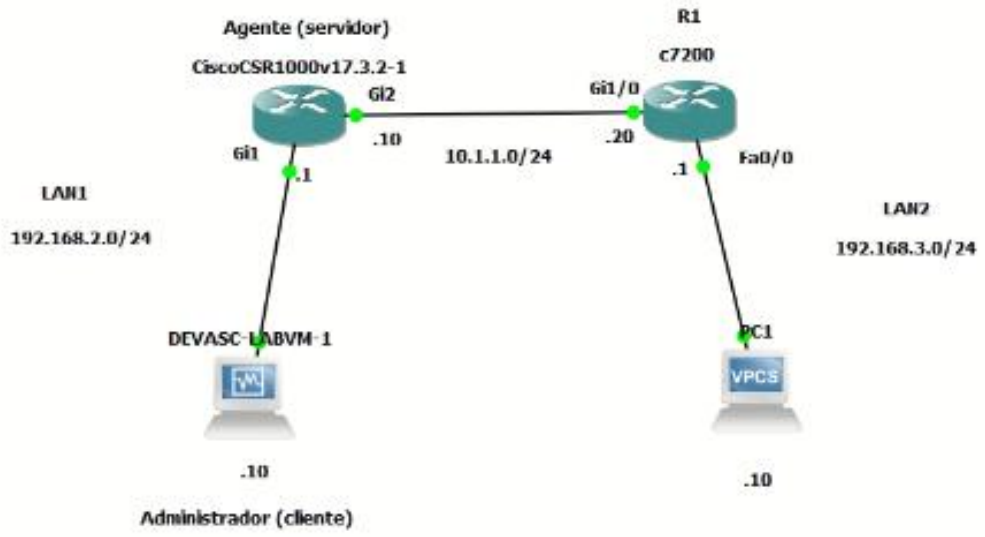

Figura 2.4. Topología de red a implementar en la práctica 4

Al concluir el laboratorio y la correcta implementación del procedimiento, el estudiante responderá a las preguntas dadas en el informe final de práctica, determinará el cumplimiento de los objetivos dados, así como de sus conclusiones y recomendaciones.

2.2.5 Práctica 5: Gestión RESTCONF con Python

Tomando en cuenta que la programación de redes está en apogeo en la actualidad, en la práctica 5 se realizará la gestión de redes mediante el protocolo RESTCONF con el lenguaje de programación más popular, Python. El conocimiento de esta herramienta ayudará a implementar scripts programables y mayor automatización de la gestión de las redes.

Los objetivos para este laboratorio serán los mismos que los de la práctica cuatro, tomando en cuenta que se elaborarán archivos con extensión .py, para analizar el lenguaje Python con el protocolo RESTCONF.

Como antesala a la realización del procedimiento, el estudiante realizará un trabajo preparatorio que consiste en consultar acerca de las librerías: requests, sys y urllib3, que ayudará a la elaboración de los códigos necesarios, así como el manejo de las URLs y métodos, ya estudiados desde la práctica 3; por último, se consultará acerca de las ventajas que ofrece Python dentro de la programación.

En la fase de procedimiento, tomando en cuenta que se realizarán las mismas configuraciones dadas en la práctica 4, esta vez con códigos python, se armará la red de

la Figura 2.4., creando scripts que permitirá configurar interfaces, crear loopbacks y rutas estáticas y obtener información de enrutamiento y estadísticas de la red.

Al final el estudiante, responderá las preguntas dadas en el procedimiento, establecerá una preferencia para una gestión eficaz de la red entre las herramientas dadas y terminará con las conclusiones y recomendaciones para la práctica.

2.3 Estructura de las prácticas de laboratorio

Las prácticas diseñadas fueron escritas siguiendo la estructura de prácticas de laboratorio establecido por el Departamento de Electrónica, Telecomunicaciones y Redes de Información de la Escuela Politécnica Nacional. Cada una de las prácticas diseñadas poseen las ocho partes establecidas en el formato, las cuales son: Tema, Objetivos, Marco teórico, Trabajo preparatorio, Equipos y materiales, Procedimiento, Informe y Referencias.

1. **Tema:** Contiene un nombre que identifica la práctica de laboratorio. Proporciona una idea general de lo que se abordará en la práctica.
2. **Objetivos:** Este apartado contiene tres objetivos, que representan los objetivos a conseguir tras la realización de las diferentes actividades propuestas en la práctica de laboratorio.
3. **Marco Teórico:** Resumir las teorías relacionadas con las disciplinas de la práctica de laboratorio. Su función es brindar la información necesaria para comprender mejor las actividades a realizar durante el desarrollo de la práctica.
4. **Trabajo preparatorio:** Son las actividades previas que los alumnos deben realizar para las prácticas de laboratorio.
5. **Equipos y materiales:** Contiene equipos o materiales necesarios para el desarrollo de la práctica. Cada ejercicio requiere una computadora con el software GNS3.
6. **Procedimiento:** Esta sección detalla el proceso que los estudiantes deben seguir para lograr los objetivos de cada ejercicio. Las actividades propuestas están formuladas con la debida explicación; cómo hacer alguna configuración, analizar y responder algunas preguntas, recuperar o solicitar información, e incluso crear algunos programas.

7. **Informes:** Contienen actividades teóricas o prácticas que los alumnos deben realizar. Estas actividades permiten evaluar los conocimientos adquiridos en las prácticas de laboratorio y reforzarlos.
8. **Referencias:** Contiene fuentes bibliográficas utilizadas para desarrollar la práctica de laboratorio.

2.4 Prácticas de laboratorio

El conjunto de prácticas de laboratorio elaboradas como componente de este Trabajo de Integración Curricular se muestran en esta sección.

2.4.1 Práctica 1

2.4.1.1 Tema Introducción a RESTCONF

2.4.1.2 Objetivos

- Configurar un router y un terminal.
- Configurar el router para habilitar el protocolo RESTCONF.
- Generar una petición/respuesta cliente-servidor mediante RESTCONF.

2.4.1.3 Marco teórico

Protocolo RESTCONF

RESTCONF *(Representational State Transfer Configuration Protocol*) es un protocolo basado en HTTP (*Hypertext Transfer Protocol*) o HTTPS, que provee interfaces programáticas RESTful, esto permite que los clientes que no tienen soporte para servicios web realicen acciones como usuario añadir, eliminar, modificar y acceda a datos de los dispositivos de red mediante solicitudes HTTP, por lo que, es un protocolo para gestión y monitoreo de redes [7]. El protocolo nació como una mejora del protocolo NETCONF (*Network Configuration*) para herramientas web y está definida por IETF (*Internet Engineering Task Force*) en el RFC 8040 desde junio de 2017 [9].

Arquitectura de red

El protocolo RESTCONF tiene una arquitectura de red cliente-servidor como se muestra en la Figura 2.5.

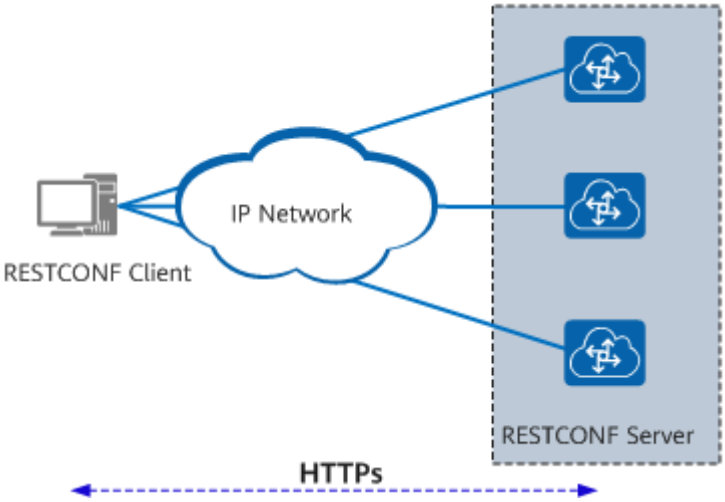

Figura 2.5. Arquitectura de red RESTCONF [7]

El cliente y el servidor RESTCONF ejecutan HTTPS para establecer una sesión segura y orientada a la conexión. El cliente envía una solicitud al servidor. Después de procesar la solicitud del usuario, el servidor responde con una respuesta al cliente. Los mensajes de solicitud y respuesta están codificados en formato XML o JSON [7].

2.4.1.4 Trabajo Preparatorio

a) Describir las características del router Cisco Cloud Services Router CSR1000V.
b) Consultar y describir los comandos necesarios para habilitar RESTCONF en el router Cisco CSR1000V.
c) Consultar mínimo 3 dispositivos que soporten el protocolo RESTCONF.
d) Descargar el archivo OVA de la máquina virtual DEVASC-LABVM.
e) Descargar la imagen GNS3 del router CSR1000V versión 17.3.2 (*csr1000v-universalk9.17.03.02-serial.qcow2*).

2.4.1.5 Equipos y materiales

Hardware

- Computadora.

Software

- VirtualBox.
- VMware Workstation Pro.
- GNS3 y GNS3 VM.

2.4.1.6 Procedimiento

Actividad 1: Implementación del entorno de trabajo

1. **Instalar máquina virtual GNS3.**
   a) Extraer el archivo .zip que fue descargado como parte del preparatorio para obtener el archivo GNS3 VM.ova.
   b) Abrir VMware Workstation Pro, seleccionar *Open a Virtual Machine* y escoger el archivo GNS3 VM.ova.
   c) Configure el nombre de la máquina virtual y hacer clic en *Import*., como se observar en la Figura 2.6.

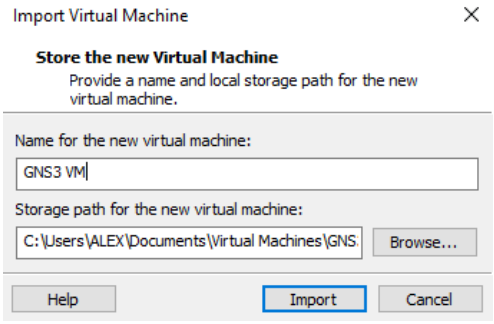

Figura 2.6. Configuración del nombre de la máquina virtual

   d) Añadir un nuevo adaptador de red y configurarlo en modo NAT. Ver Figura 2.7.

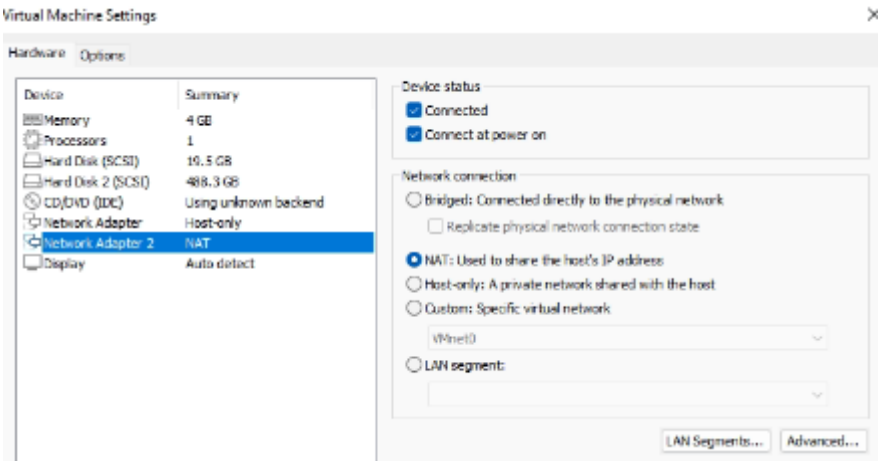

Figura 2.7. Configuración del Adaptador de red en modo NAT

2. **Integrar GNS3 con la máquina virtual GNS3 (GNS3 VM) [25].**
   a) En GNS3, seleccionar *Help > Setup Wizard.*

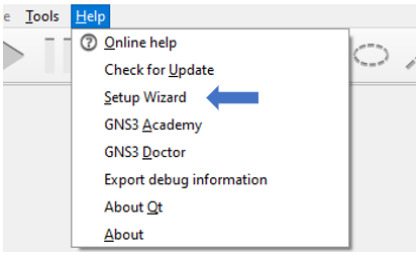

Figura 2.8. Asistente de configuración

b) Escoger la opción *Run appliances in a virtual machine.*

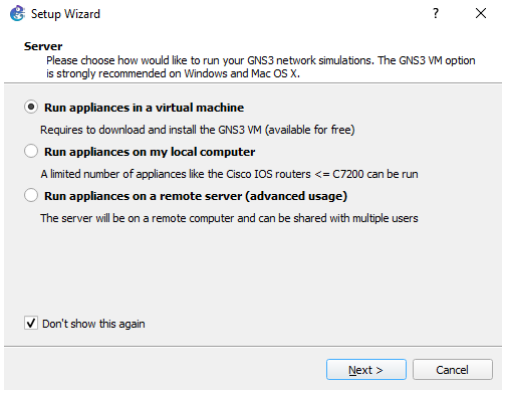

Figura 2.9. Seleccionar el servidor de ejecución de los dispositivos

c) Aunque GNS3 VM ejecutará las máquinas virtuales, imágenes y contenedores se debe configurar el servidor local con los parámetros que se muestran en la Figura 2.10.

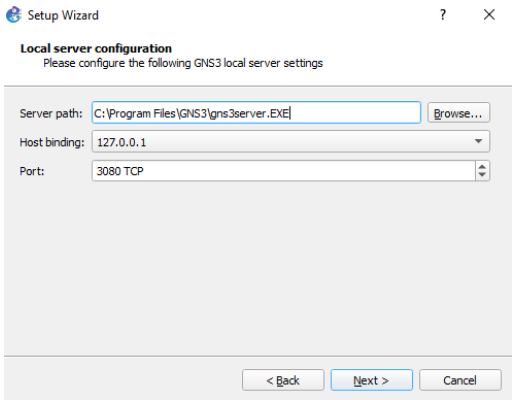

Figura 2.10. Configuración del servidor local

d) Validar la correcta conexión del servidor local y seleccionar *Next*.

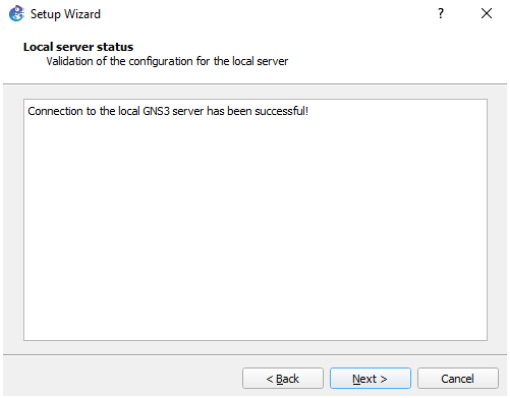

Figura 2.11. Validación de la conexión con el servidor local

e) Haga clic en el botón *Refresh* si la presencia de la máquina virtual WMware Workstation GNS3 no se detecta automáticamente. De lo contrario, configure la cantidad de vCPU y el tamaño de RAM utilizando los parámetros que se muestran en la Figura 2.12.

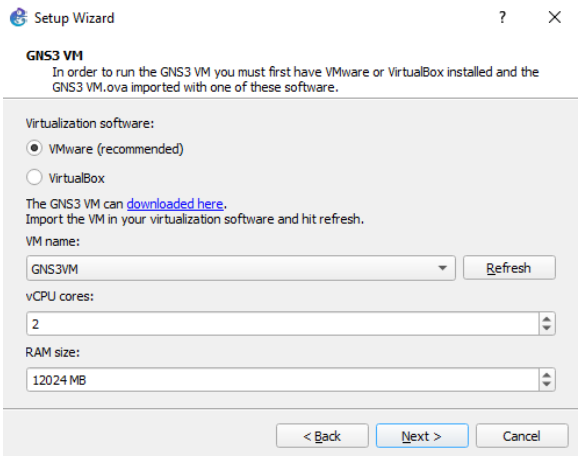

Figura 2.12. Configuración de parámetros de GNS3 VM

f) Se mostrará un resumen de la configuración de la máquina virtual GNS3. Dar clic en *Finish*.

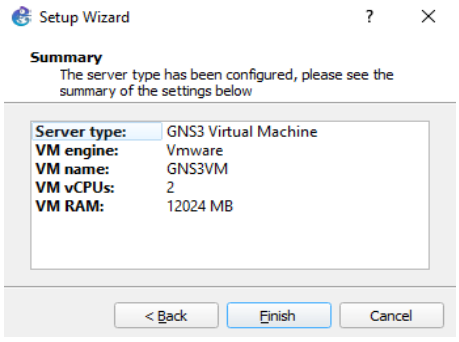

Figura 2.13. Resumen de la configuración de GNS3 VM

3. Importación dBel router Cisco CSR1000v a GNS3

a) Abrir GNS3, en la barra de los dispositivos de red, seleccionar *Routers* > *New Template*.

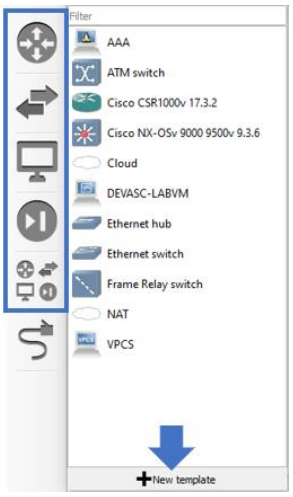

Figura 2.14. Creación de una nueva plantilla

b) Seleccionar *Install an appliance from GNS3 server (recommended)* > *Next*.

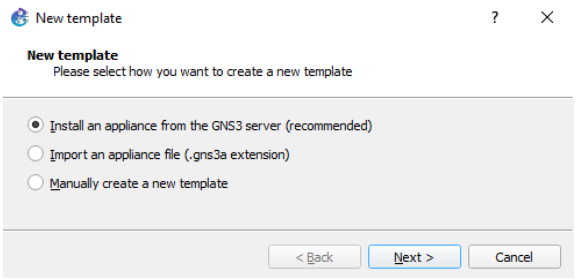

Figura 2.15. Ventana para escoger cómo crear una nueva plantilla.

c) En la ventana *Appliances from server*, seleccionar *Routers* > *Cisco CSR1000v* > *Install,* como se muestra en la Figura 2.14.

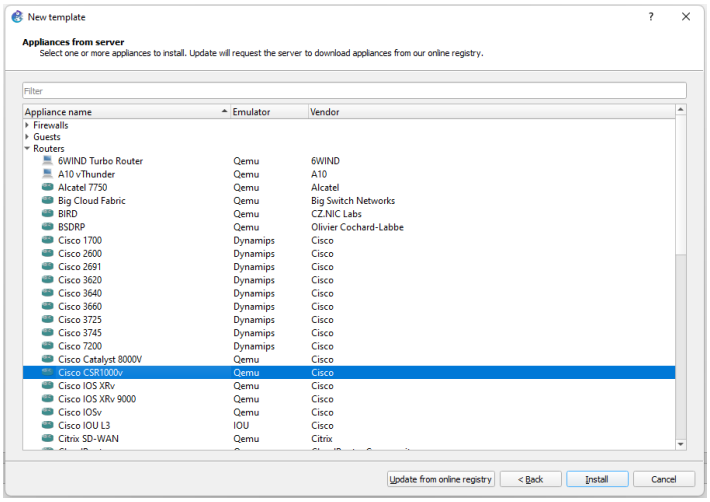

Figura 2.15. Seleccionar el router Cisco CSR1000v

d) En la ventana *Server*, seleccionar *Install the appliance on the GNS3 VM (recommended) > Next*.

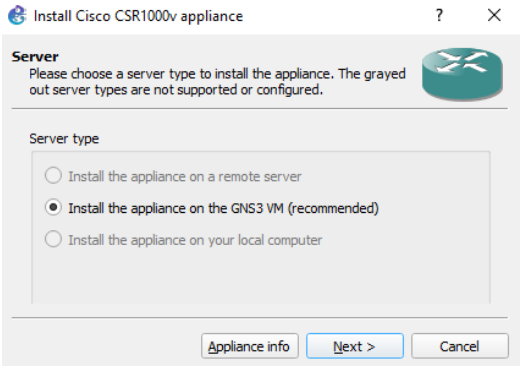

Figura 2.16. Ventana para escoger donde ejecutar el router Cisco CSR1000v

e) En la ventana *Qemu settings,* esperar que GNS3 escoja el mejor quemu, seleccionar *Next.*

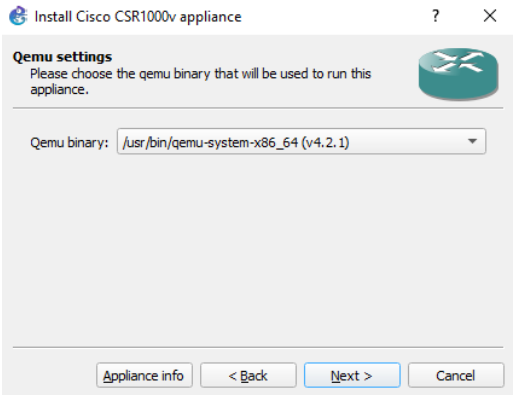

Figura 2.17. Ventana para escoger el Qemu binary

f) Debido a que la versión 17.3.2 del router cisco CSR 1000v no se encuentra en la lista, se debe crear una nueva versión. En la ventana *Required files,* seleccionar *Create a new version.*

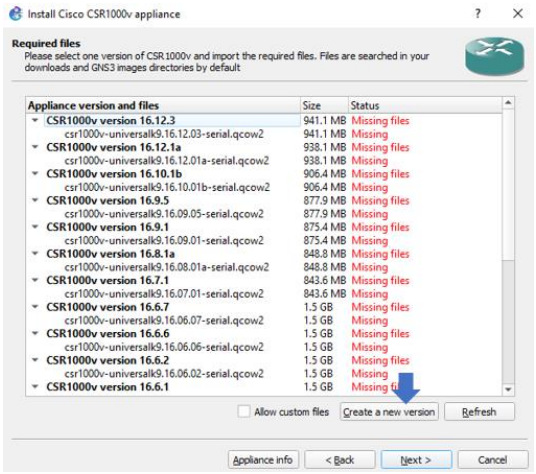

Figura 2.18. Creación de una nueva versión

g) Digitar 17.3.2 como el nombre de la nueva versión, seleccionar *Ok.*

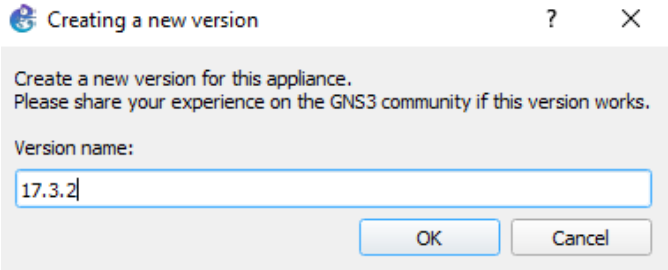

Figura 2.19. Creación de una nueva versión

h) En la ventana *Image,* colocar el nombre del *archivo csr1000v-universalk9.17.03.02-serial.qcow2,* seleccionar *Ok.*

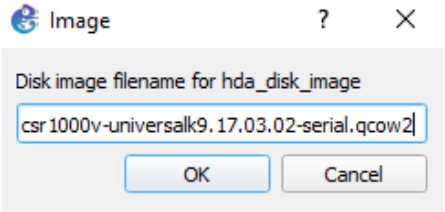

Figura 2.20. Nombre y extensión de la imagen de disco del router Cisco CSR1000v

i) Seleccionar la versión creada e importar la imagen GNS3 que se descargó en el trabajo preparatorio. Una vez listo, seleccionar *Next* > *Yes* > *Ok* > *Finish*.

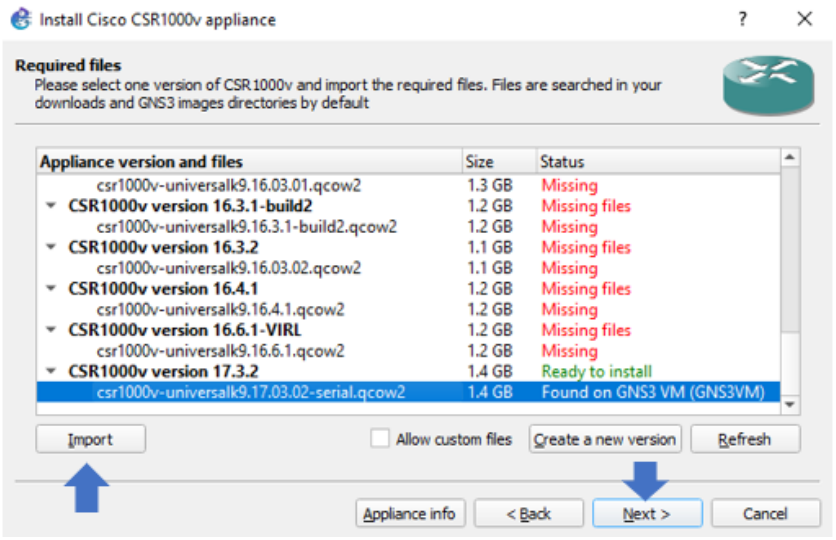

Figura 2.21. Importar la imagen GNS3 del router

j) El router ha sido creado y listo para ser usado.

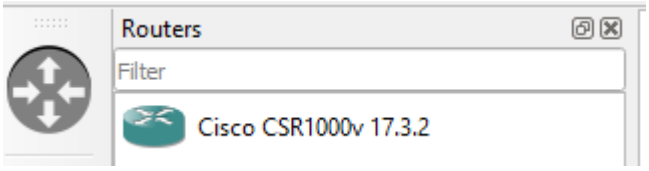

Figura 2.22. Router Cisco CSR1000v 17.3.2

4. Instalación de la máquina virtual DEVASC

a) Abrir VirtualBox, seleccionar *Archivo* > *Importar servicio virtualizado*. En *Archivo* seleccionar el archivo OVA de la máquina virtual DEVASC-LABVM. Seleccionar *Next* > *Importar.*

b) Una vez importada, seleccionar la máquina virtual DEVASC-LABVM, y hacer clic en *Configuración* > *Red* y habilitar *Adaptador 1* y *Adaptador 2.*

c) En *Adaptador 1*, configurar que esté *Conectado a: Controlador genérico*. Clic en *Avanzadas* y habilitar *Cable conectado*.

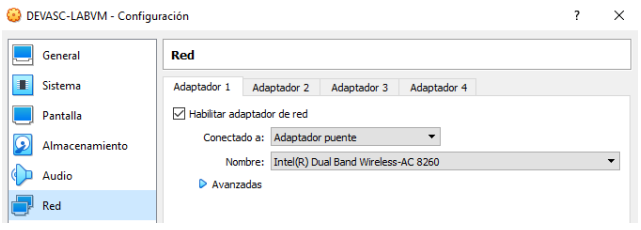

Figura 2.23. Configuración de adaptador de red 1 de DEVASC-LABVM

d) En *Adaptador 2*, configurar que esté *Conectado a: NAT*. Clic en *Avanzadas* y habilitar *Cable conectado*.

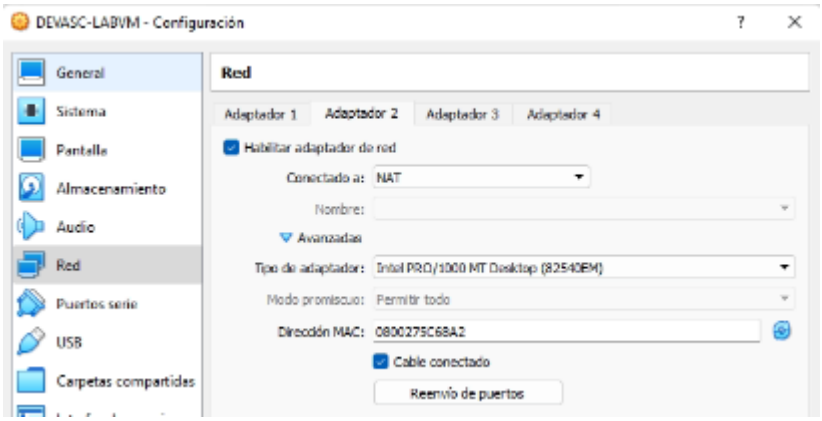

Figura 2.24. Configuración de adaptador de red 2 de DEVASC-LABVM

e) Clic en *Aceptar* y ejecutar la máquina virtual DEVASC-LABVM.

f) DEVASC-LABVM tiene Packet Tracer, por lo que debe aceptar el EULA de Cisco Packet Tracer para continuar con el arranque de la máquina virtual. Cuando vea el acuerdo de licencia, use las teclas de flecha para desplazarse por el texto. Seleccione <*ok*>, presione la barra espaciadora y finalmente seleccione <*I Agree*.

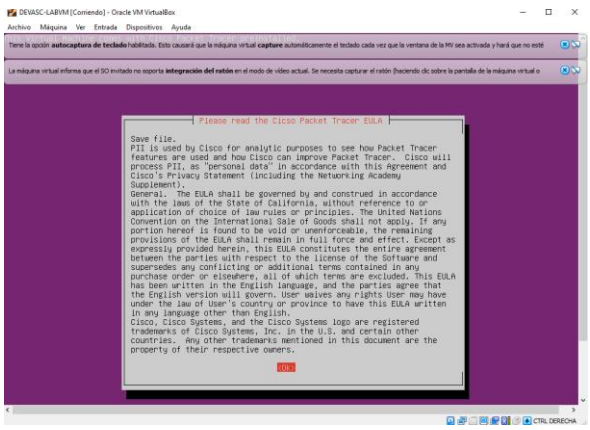

Figura 2.25. EULA de Cisco Packet Tracer.

g) La imagen de Ubuntu se procederá a cargar.

5. Integración de la máquina virtual DEVASC con GNS3

a) En GNS3, seleccionar *Edit > Preferences.*

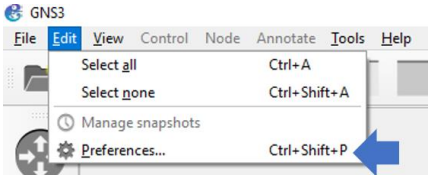

Figura 2.26. Opción para habilitar el cliente

b) En la ventana *Preferences,* seleccionar *VirtualBox VMs > New.*

c) En la ventana *Server,* seleccionar *Run this VirtualBox VM on my local computer > Next.*

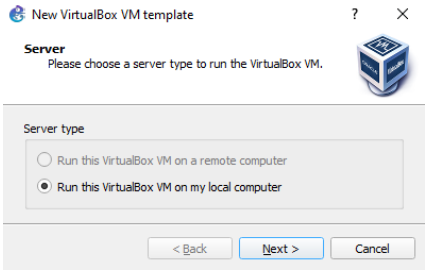

Figura 2.27. Selección del lugar de ejecución de DEVASC

d) En la ventana *VirtualBox Virtual Machine, seleccionar* DEVASC-LABVM y clic en *Finish.*

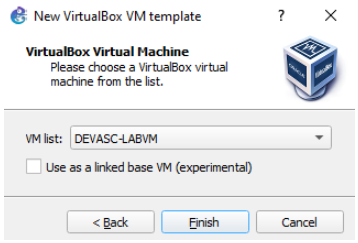

Figura 2.28. Seleccionar la máquina virtual DEVASC-LABVM

e) La máquina virtual ha sido creada y lista para ser usada. Se la puede encontrar en la barra de dispositivos de red > *End Devices.*

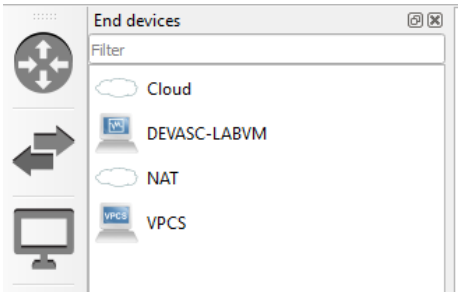

Figura 2.29. Terminal DEVASC-LABVM

Actividad 2: Establecer conectividad entre equipos

a) Abra GNS3 y cree la topología de red que se muestra en la Figura 2.30.

Figura 2.30. Topología de red a implementar

b) Verificar el estado de la interfaz de red `enp0s3` que es el puerto ethernet 0 de la máquina virtual DEVASC-LABVM.

```
devasc@labvm:~$ ifconfig enp0s3
```

c) Configurar la interfaz de red `enp0s3` con la dirección IP de la topología dada en la Figura 7. Para esto, se debe modificar el archivo `etc/network/interfaces`. Con el siguiente comando:

```
devasc@labvm:/$ sudo nano /etc/network/interfaces

#source-directory /etc/network/interfaces.d
auto enp0s8
iface enp0s8 inet static
address 192.168.2.10
netmask 255.255.255.0
gateway 192.168.2.1
```

d) Reiniciar el servicio de red con el comando:

```
sudo /etc/init.d/networking restart
```

e) Verificar la interfaz de red `enp0s3` que tenga la dirección IP configurada.

```
devasc@labvm:~$ ifconfig enp0s3
```

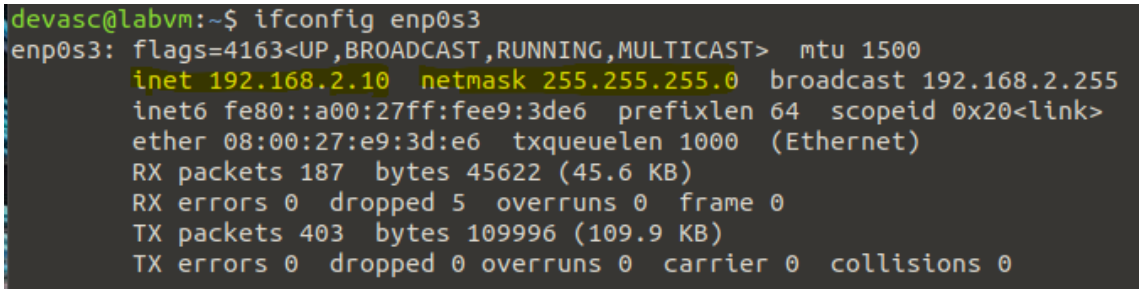

Figura 2.31. Interfaz de red configurada

f) Iniciar el router cisco CSR1000v y configurar la interfaz GigabitEthernet1 con la dirección IP dada en la Figura 2.30.

```
Router>enable
Router#configure terminal
Router (config)#interface G1
Router (config-if)#ip address 192.168.2.1 255.255.255.0
Router (config-if)#no shutdown
Router (config-if)#exit
```

g) Verificar la conectividad entre la máquina virtual DEVASC-LABVM (Administrador) con el router cisco CSR1000v (Agente).

Actividad 3: Configuración de SSH en router cisco CSR1000v

a) Crear un usuario `admin:admin` con privilegio 15 (privileged EXEC).

```
Router(config)#username admin privilege 15 password admin
```

b) Configure el nombre y el dominio del enrutador Cisco CSR1000v.

```
Router(config)#hostname CSR1k
CSR1k(config)#ip domain name labo-restconf.com
```

c) Generar pares de claves RSA con un tamaño de módulo de 2048 bits.

```
CSR1k(config)# crypto key generate rsa modulus 2048
```

d) Habilitar SSH versión 2

```
CSR1k(config)#ip ssh version 2
```

e) Habilitar SSH en líneas VTY y configurar que el acceso a las mismas sea mediante la autenticación de usuarios locales.

```
CSR1k(config)#line vty 0 4
CSR1k(config-line)#transport input ssh
CSR1k(config-line)#login local
CSR1k(config-line)#exit
```

f) Abrir una terminal en la máquina virtual DEVASC y verificar el funcionamiento de SSH.

```
devasc@labvm:~$ ssh admin@192.168.2.1
```

Actividad 4: Habilitar RESTCONF

A partir de ahora y en prácticas posteriores se tratará como Administrador a la máquina virtual DEVASC-LABVM y como Agente al router cisco CSR1000v. Se realizará la siguiente configuración [26].

a) Verificar si los demonios asociados con RESTCONF están habilitados.

```
CSR1k#show platform software yang-management process
```

Pregunta 1. ¿Qué muestra el comando?
Pregunta 2. ¿Qué proceso indica el demonio *nginx*?

b) Habilitar el servicio RESTCONF.

```
CSR1k#configure terminal
CSR1k(config)# ip http secure-server
CSR1k(config)# ip http authentication local
CSR1k(config)# restconf
```

Pregunta 3. ¿Qué indican los comandos ingresados?
Pregunta 4. ¿Cuáles son los logs que se ha mostrado el agente?,¿qué indican?

c) Verificar el servicio RESTCONF.

```
CSR1k#show platform software yang-management process
```

Pregunta 5. ¿Qué muestra el comando?, ¿Cuál es la diferencia entre el resultado de a) y c)?
Pregunta 6. ¿Por qué el proceso `ncsshd` no está corriendo?

Actividad 5: Petición/respuesta cliente-servidor

Se ha habilitado el protocolo RESTCONF en el agente, por lo que, ya es posible realizar peticiones desde el administrador. Esto se realizará con el comando curl y sus opciones [27].

a) Realizar la siguiente petición desde el administrador.

```
devasc@labvm:~$ curl -k https://192.168.2.1:443/restconf/ -u
"admin:admin"
```

Pregunta 7. ¿Qué indica cada opción del comando ingresado?
Pregunta 8. ¿Cuál es la respuesta?, ¿qué indica?
Pregunta 9. ¿El agente ha mostrado algún log?, ¿qué indica?

2.4.1.7 Informe

a) Proporcione la configuración construida en laboratorio con una explicación adecuada de los resultados que se muestran.
b) Responder las preguntas dadas en el procedimiento.
c) Conclusiones y Recomendaciones.
d) Bibliografía

2.4.2 Práctica 2

2.4.2.1 Tema: Lenguaje de Modelado de Datos YANG

La Hoja Guía perteneciente a esta práctica se encuentra en el ANEXO I.

2.4.3 Práctica 3

2.4.3.1 Tema: Seguridad y Métodos HTTP

La Hoja Guía perteneciente a esta práctica se encuentra en el ANEXO II.

2.4.4 Práctica 4

2.4.4.1 Tema: Gestión RESTCONF con Postman

La Hoja Guía perteneciente a esta práctica se encuentra en el ANEXO III.

2.4.5 Práctica 5

2.4.5.1 Tema: Gestión RESTCONF con Python

La Hoja Guía perteneciente a esta práctica se encuentra en el ANEXO IV.

2.5 Material complementario

El propósito de desarrollar material adicional para la guía es mejorar el aprendizaje de los temas tratados en cada práctica de laboratorio. Este material está destinado a estudiantes y docentes que les resulte adecuado utilizar las prácticas presentadas en el trabajo integrado de este curso. Cada uno de estos se describe a continuación.

a) **Resolución de trabajos preparatorios:** El trabajo preparatorio es una actividad que debe ser realizada por los estudiantes como trabajo autónomo. El objetivo de estas es introducir a los alumnos en los temas que se abordarán durante el transcurso de cada práctica. La resolución de preparación es para los maestros. Su finalidad principal es dar una idea general de lo que deben hacer los alumnos en cada preparación. La resolución sobre la labor preparatoria figura en el anexo V.

b) **Reactivos:** Son preguntas de opción múltiple al estilo de Moodle que abordan temas tratados en el marco teórico y guían el trabajo preparatorio. Cada laboratorio tiene un total de 10 reactivos que los profesores pueden usar para evaluar la correcta preparación de los estudiantes antes de realizar las prácticas de laboratorio. El ANEXO VI contiene el enlace de los reactivos recomendados.

c) **Videos:** Los estudiantes y profesores podrán observar la implementación real de cada procedimiento de práctica de laboratorio. Esto ayudará a resolver cualquier duda o inconveniente que pueda surgir al seguir los procedimientos descritos en la guía. Esto también les dará a los estudiantes una mejor comprensión de las configuraciones y actividades, lo que facilitará su aprendizaje. El Anexo VII contiene el enlace a los videos desarrollados.

3 RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

3.1 Resultados

Los resultados de este curso integrando los componentes desarrollados en el trabajo corresponden a la resolución de cada informe de prácticas de laboratorio. El informe incluye tres actividades. La primera actividad requiere mostrar la configuración realizada en el laboratorio y proporcionar evidencia durante el proceso de captura con una interpretación adecuada de los resultados mostrados. La segunda actividad consiste en responder a las preguntas planteadas en el programa, lo que permitirá a los alumnos analizar, reforzar e incrementar los conocimientos adquiridos en el desarrollo práctico. La última actividad corresponde a las conclusiones y recomendaciones. El Anexo VIII contiene resoluciones detalladas para cada informe. A continuación, se presentan los puntos más relevantes para la primera actividad reportada en el anexo anterior.

3.1.1 Práctica 1: Informe

La actividad 1, denominada "Implementación del Ambiente de Trabajo", arrojó un resultado. Al finalizar la configuración expuesta en este evento, se implementó con éxito la instalación de la imagen de disco del router CSR1000v y la integración de la máquina virtual DEVASC con GNS3. Las consideraciones más importantes para lograr este resultado son configurar la VM GNS3 para que se encargue de realizar la creación de imágenes del dispositivo (incluido el enrutador), configurar los parámetros vCPU, RAM y disco duro virtual y, como mínimo, instalar la imagen de disco en caso de que falle el enrutador.

En la actividad 2, “Establecer una conexión entre dispositivos”, se obtuvo la conexión entre la máquina virtual DEVASC y el router CSR1000v. Verifique este resultado obteniendo una prueba de ping exitosa entre los dos dispositivos.

Para la actividad 3 con título “Configuración de SSH en router cisco CSR1000v”, se tiene como resultado una sesión SSH de manera exitosa desde el administrador, para esto se ha creado el usuario admin con su contraseña y privilegio de nivel 15; esto nos permitirá tener acceso directo al router desde el cliente.

La actividad 4 denominada “Habilitar RESTCONF” al cumplirla se tiene la habilitación del protocolo en el router, cambiando su estatus a servidor, esto se muestra con el resultado del comando `show platform software yang-management process`, donde indica que los demonios implicados a los protocolos HTTPS y RESTCONF han sido activados en el router, tal como se muestra en la Figura 3.1. Al momento de habilitar la interfaz RESTCONF, con los comandos dados en la hoja guía, se pudo observar el proceso que se hace internamente para habilitar el servidor RESTCONF. Primero, el proceso *psd* notifica al servidor RESTCONF que debe iniciar, después con el proceso *ndbmand* se activan todos los proveedores de datos, y por último el proceso *dmiauthd* hace que la configuración en ejecución se sincronice con el almacén de datos en ejecución de NETCONF. Se debe recordar que RESTCONF trabaja sobre los mismos fundamentos de NETCONF y su base de datos.

El único demonio que no se ha habilitado es el relacionado con el protocolo NETCONF.

```
CSR1k#show platform software yang-management process
confd            : Running
nesd             : Running
syncfd           : Running
ncsshd           : Not Running
dmiauthd         : Running
nginx            : Running
ndbmand          : Running
pubd             : Running
```

Figura 3.1. Salida del comando ejecutado después de habilitar RESCONF

La implementación de la actividad “Petición/respuesta cliente-servidor” utiliza la herramienta cURL para enviar un mensaje de solicitud HTTP al servidor, teniendo del parte del servidor un mensaje de respuesta de datos conceptuales que está estandarizada en el RFC 8040. Muestra que el protocolo RESCONF es dado por la IETF, en el formato XML, utiliza el modelo YANG y el módulo ietf-restconf. Además, identifica la fecha de revisión del módulo ietf-yang-library que implementa este servidor RESTCONF, en este caso el 21 de junio de 2016 [28]. Esto demuestra que se tiene una conexión cliente-servidor dentro del protocolo RESTCONF, como se observa en la Figura 3.2.

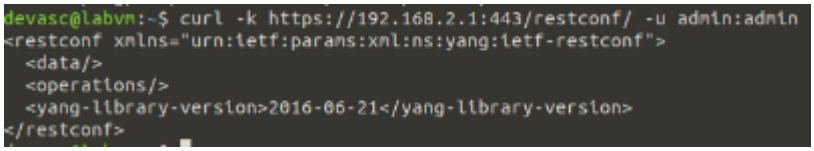

Figura 3.2. Conexión RESTCONF entre el administrador y el agente

3.1.2. Práctica 2: Informe

Para la segunda actividad, nombrada "Explorando modelos YANG en GITHUB", se examinaron dos modelos de datos YANG que tienen la información acerca de las interfaces de dispositivos de red: el modelo *ietf-interfaces* y el modelo *openconfig-interfaces*. Se eligió y describió un ejemplo de nodos del lenguaje YANG (leaf, leaf-list, container y list) de cada modelo de datos. Esta actividad permitió comprender cómo funciona el lenguaje YANG con los modelos de datos. Por ejemplo, fue diseñado como un nodo contenedor (llamado interfaz) utilizado en el modelo de datos de ietf-interfaces para proporcionar una jerarquía de parámetros configurables para la interfaz de un dispositivo.

En la Actividad 3, titulada “Exploración de modelos YANG con Pyang”, el modelo de datos de la Actividad 2 se convirtió a formato de árbol utilizando la herramienta Pyang. El resultado es una estructura de datos completa, ordenada y fácil de entender. Al transformar cada modelo, se obtuvo una vista más amplia de toda la jerarquía de datos disponibles en el modelo. Algo que no fue posible cuando se miró el modelo en el repositorio de GITHUB. Por ejemplo, se reconoce que ietf-interfaces define dos nodos contenedores principales para separar la configuración de la interfaz y los datos de estado. El modelo de datos openconfig-interfaces, por otro lado, utiliza un nodo contenedor principal, con nuevos nodos configurados dentro de él para separar la configuración de la interfaz y los datos de estado.

La Actividad 3, Representación de datos YANG, compara el módulo de ietf-interfaces del modelo YANG IETF para los dos formatos de codificación admitidos por el protocolo RESTCONF que son XML y JSON, observando que XML se usa principalmente para la comunicación entre servidores. y aplicaciones. Esto se debe esencialmente a la capacidad de crear mensajes más complejos que se pueden verificar fácilmente. En cambio, JSON es un formato más simple y fácil de trabajar que los formatos anteriores. Por lo tanto, es la mejor solución para dispositivos que no requieren procesos extensos y es más fácil de usar, ya que presenta el contenido de una manera fácil de entender, como se puede comparar en la Figura 3.3. y la Figura 3.4.

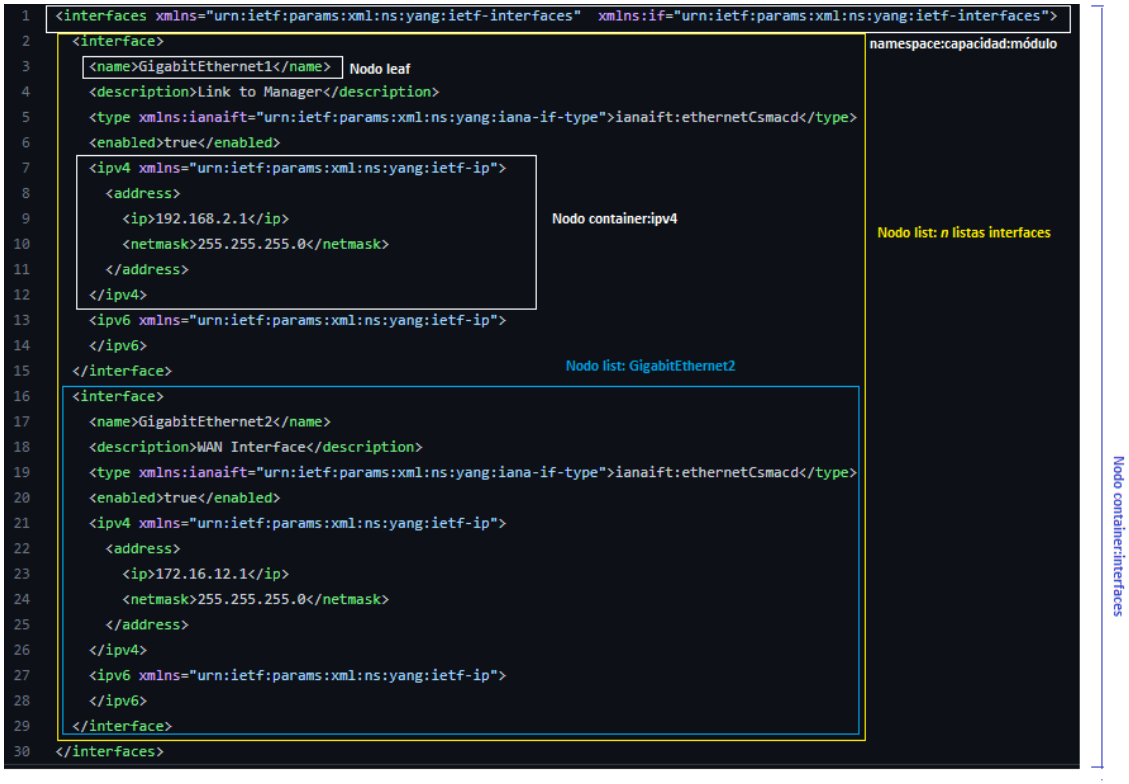

Figura 3.3. Módulo *ietf-interfaces* en formato XML

Figura 3.4. Módulo *ietf-interfaces* en formato JSON

3.1.3. Práctica 3: Informe

La actividad 1 es “Establecer una ACL para RESTCONF”, su implementación tiene al agente como un enrutador de dos redes LANs, pero con la aplicación de una ACL para seguridad y control de flujo de paquetes da como resultado tener acceso al servidor RESTCONF solo de un administrador deseado, en este caso, solo el administrador principal, 192.1682.10/24, tiene acceso a la gestión de la red mediante el protocolo RESTCONF, como se observa en la Figura 3.5.

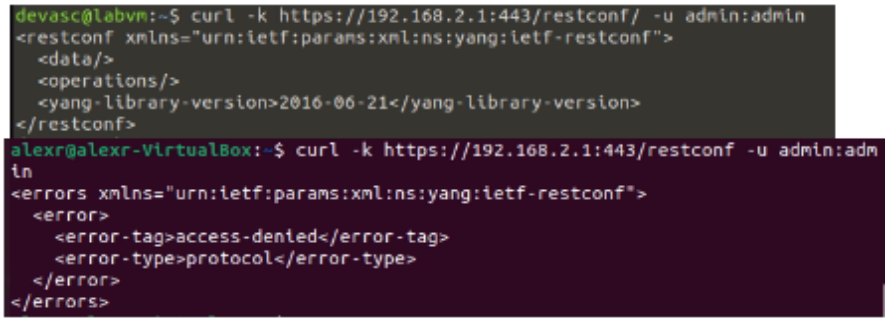

Figura 3.5. Acceso a RESTCONF solo desde el administrador principal

La segunda actividad de la práctica es “Realizar peticiones HTTP con cURL”. Desde la primera práctica se ha utilizado la herramienta cURL, pero para esta actividad se explora mayores opciones y el formato para establecer métodos HTTP, como son: GET, PUT, PATCH y DELETE. Las peticiones dadas por la hoja guía son realizadas con éxito, como la que se hizo con el método GET, configurando el formato de recepción de datos en JSON, a la URL https://192.168.2.1:443/restconf/data/ietf-interfaces:interfaces, como se observa en la Figura 3.6.

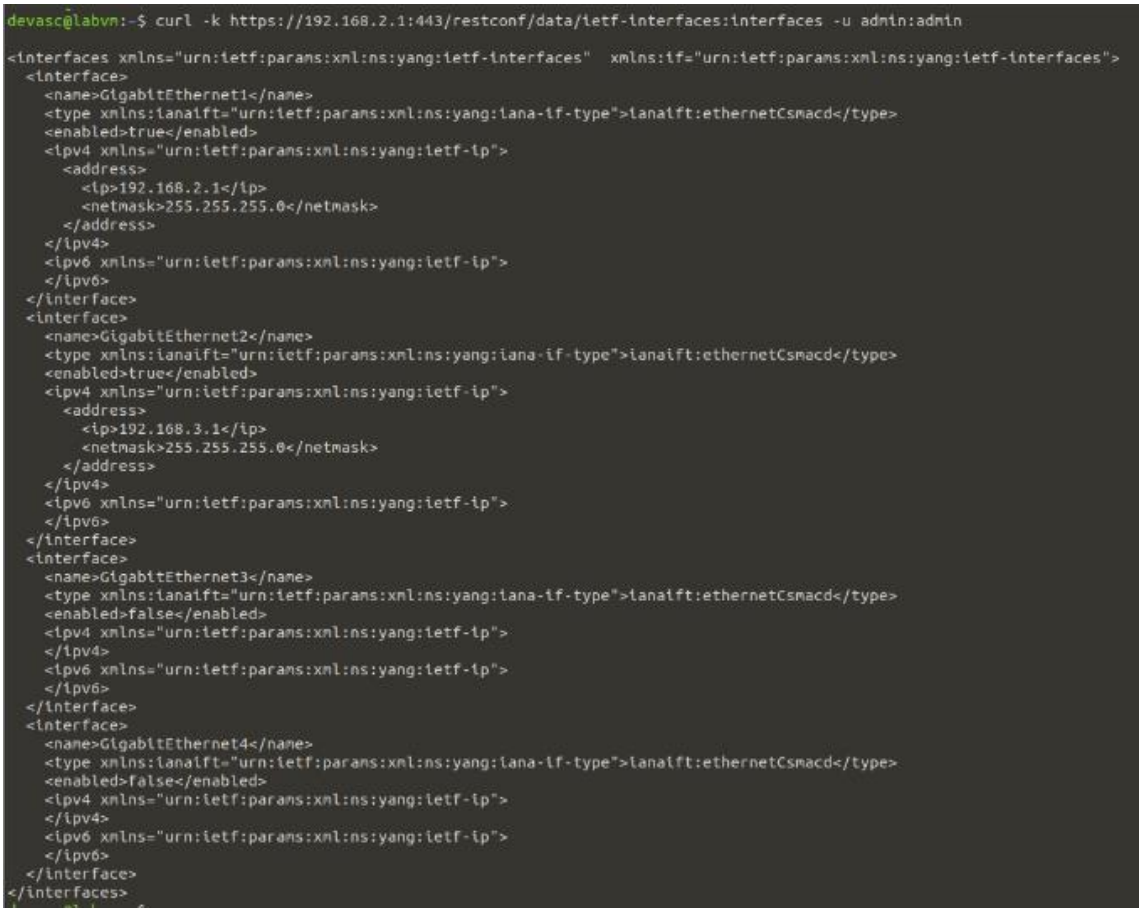

Figura 3.6. Información de las interfaces del agente con cURL

3.1.4. Práctica 4: Informe

El resultado de la primera actividad “Introducción a Postman” es la implementación de la red dada, y una introducción al programa Postman, como la configuración para desactivar el certificado SSL, indispensable en las peticiones HTTP, y la autenticación básica para las sesiones entre el cliente y el servidor. Esto sirvió para todas las APIs creadas posteriormente.

La segunda actividad es “Configuración de interfaces con RESTCONF mediante Postman”. Esta actividad se hizo de manera exitosa, configurando la interfaz GigabitEthernet 2 conforme a las indicaciones de la hoja guía, con el método PATCH; creando y eliminando la interfaz Loopback 1, con los métodos PUT y DELETE respectivamente. Además de conseguir información de cualquier interfaz, conociendo las URLs necesarias, mediante el método GET.

La tercera actividad “Creación de ruta estática con RESTCONF mediante Postman.” Tuvo como resultado la aplicación de las APIs necesarias para tener conectividad en todas las redes. Esto se hizo a partir de la creación de una ruta estática, con el método PUT, que permita al administrador tener conectividad con la red faltante. Y con el método GET, se obtiene la tabla de enrutamiento del agente.

La cuarta actividad consiste en “Análisis de la red con RESTCONF mediante Postman.”, esto permitió conocer el estado del protocolo RESTCONF en la red, así como las estadísticas de entrada y salida de paquetes y la información detallada de las interfaces. Esto es información de vital importancia para tener mayores oportunidades de gestionar la red mediante el protocolo de estudio.

3.1.5. Práctica 5: Informe

El resultado de la primera actividad “Introducción a Python” es la implementación de la red dada, la misma configuración de la práctica 4. Además, se realizaron los códigos básicos y necesarios para la práctica. Se ha creado el script `código info_basica.py`, que permite tener las configuraciones básicas para deshabilitar el certificado SSL, crear una sesión mediante una autenticación HTTP básica, tener las variables seteadas de la URL, como la IP del servidor, puerto y protocolo a utilizar y tener el encabezado configurado para enviar y recibir solicitudes en formato JSON. La configuración del script se puede ver en la Figura 3.7. Además, para tener mayor funcionalidad en los scripts, se crearon cuatro funciones de los métodos HTTP: GET, PUT, PATCH y DELETE, tal como se pide en la hoja guía, esto nos ayudará para los scripts creados posteriormente.

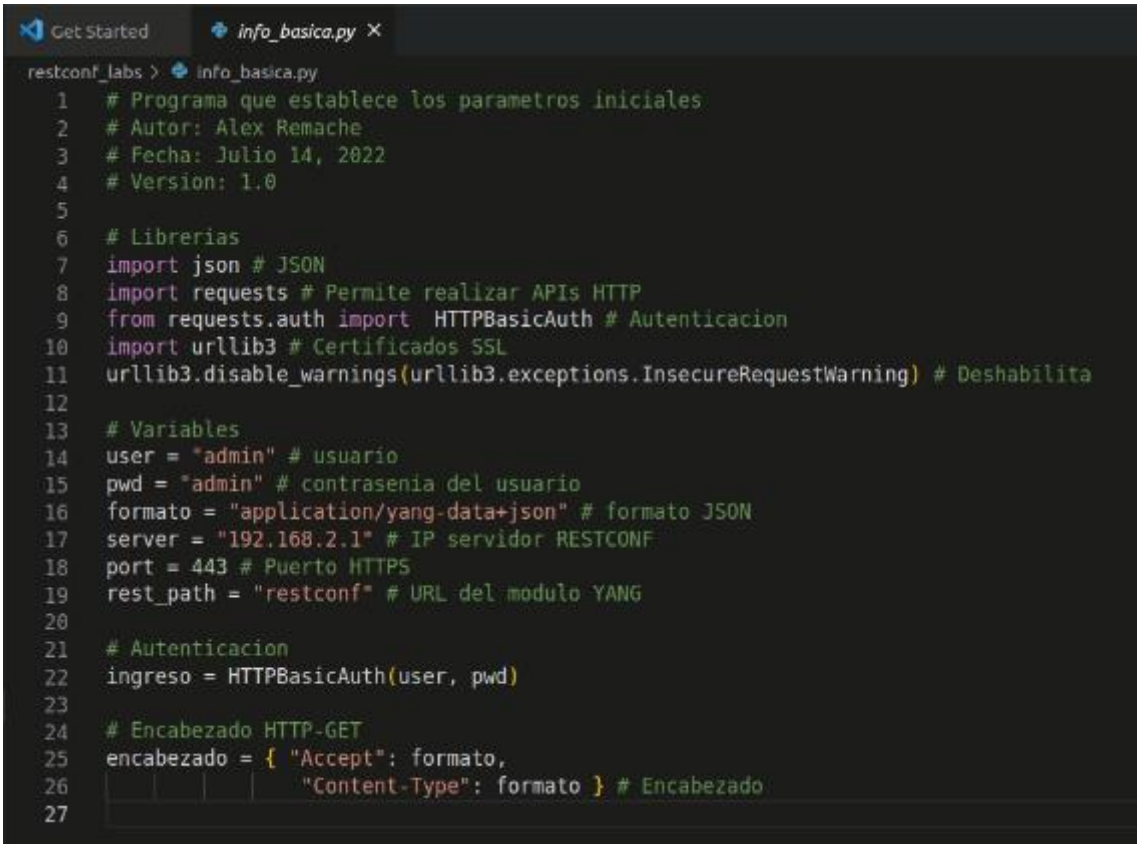

Figura 3.6. Script info_basica.py en el administrador

La segunda actividad es “Configuración de interfaces con RESTCONF mediante Python”. Esta actividad se hizo de manera exitosa, configurando la interfaz GigabitEthernet 2 conforme a las indicaciones de la hoja guía, con el método PATCH; creando y eliminando la interfaz Loopback 1, con los métodos PUT y DELETE respectivamente. Además de conseguir información de cualquier interfaz, conociendo las URLs necesarias, mediante el método GET. Todo esto mediante códigos que permiten realizar las operaciones mencionadas y muestran los resultados y los códigos de estado correspondientes.

La tercera actividad “Creación de ruta estática con RESTCONF mediante Python.” Tuvo como resultado la aplicación del código `put_crear_ruta.py` para la creación de una ruta estática, con el método PUT, que permite al administrador tener conectividad con la red faltante. Y con el método GET, se obtiene la tabla de enrutamiento del agente, esto con el código `get_info_routing.py`.

La cuarta actividad consiste en “Análisis de la red con RESTCONF mediante Python.”, esto permitió conocer el estado del protocolo RESTCONF en la red, así como las estadísticas de entrada y salida de paquetes y la información detallada de las interfaces, como se muestra en la Figura 3.7.

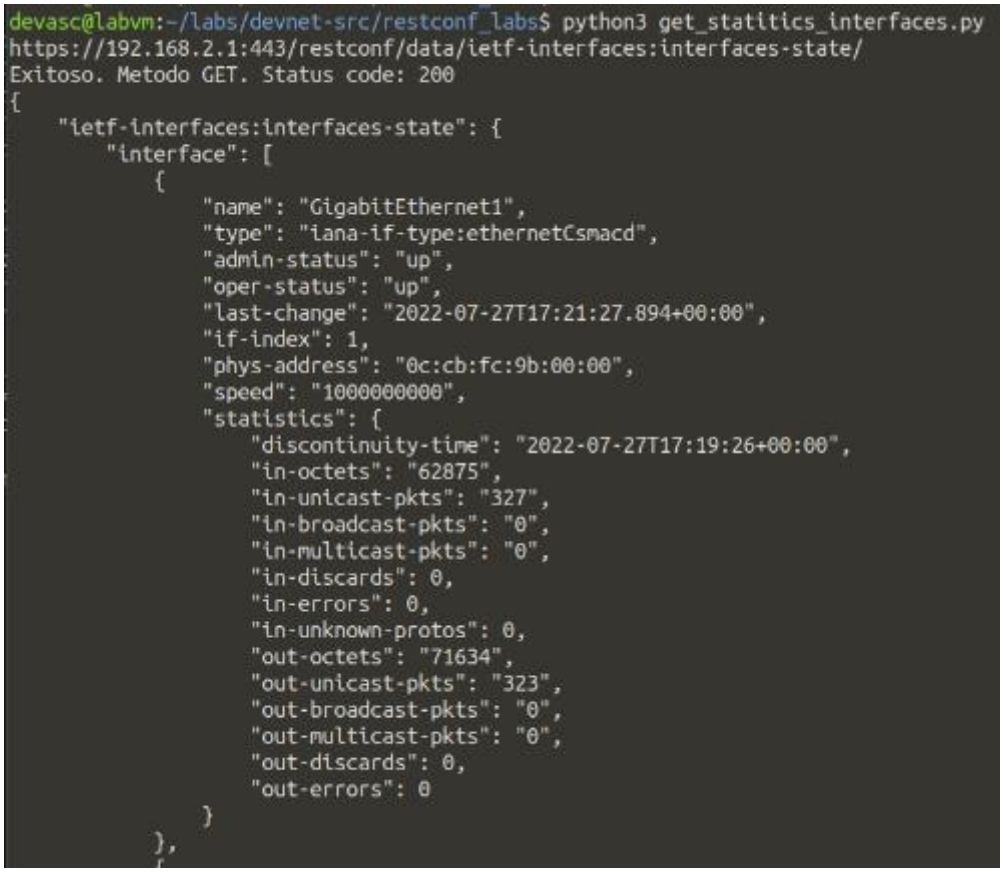

Figura 3.7. Resultado de get_statitics_interfaces.py en el administrador

3.2 Conclusiones

- El protocolo de gestión de red RESTCONF es una opción viable para la administración de la red. RESTCONF aborda de mejor manera las dificultades que tienen otros protocolos de gestión, como SNMP y NETCONF, pues trabaja sobre un protocolo que es conocido, el protocolo HTTP y para una mejor seguridad, RESTCONF trabaja con HTTPS. Mediante los métodos que tiene el HTTP el protocolo ofrece una variedad de peticiones para crear, configurar, eliminar, obtener información, estado de red y de variables, y estadísticas permitiendo al administrador de red, tener las facilidades necesarias para una gestión eficaz. Estas características hacen al protocolo RESTCONF un potencial candidato para que se convierta, en un futuro cercano, en el protocolo de gestión de red simple, amigable con el usuario, cercano a la automatización y programabilidad de las redes.

- El estudio del Lenguaje de Modelado de Datos YANG es esencial para el protocolo RESTCONF. Para la gestión de redes se pasa de un modelo de variables (SNMP) a un modelo de representación de datos en módulos, submódulos y contenedores. Sin embargo, el lenguaje YANG esgrimido en los modelos YANG no enseña la

distribución de los datos en contenedores/listas/nodos, por esto una determinada función de un elemento de red podría representarse con diversos modelos de YANG, como el modelo IETF, OPENCONFIG o nativas de los vendedores y distribuidores de software y equipos y elementos de red delegados de crear, establecer y distribuir diversos de sus modelos de datos. Conocer esta organización y su respectiva conversión a URLs es indispensable para realizar las operaciones con métodos HTTP esenciales para el trabajo con RESTCONF. Al tener un modelo YANG basado en representación de datos implica una mayor facilidad para los administradores la optimización de su gestión, configuración y monitoreo de sus redes.

- El entorno de trabajo para la simulación/emulación de redes ha sido la óptima para la culminación de los objetivos de este Trabajo de Integración Curricular. Al tener que trabajar con equipos de alta gama, como el router utilizado en las prácticas de laboratorio, es esencial tener los equipos hardware y software precisos para su aplicación. Para lo cual, se ha utilizado el programa GNS3 que es una herramienta muy potente y permite emular entornos de red. Además, GNS3 permite la emulación/simulación de equipos de red necesarios para la realización de las prácticas, como los hosts finales basados en máquinas virtuales. Al ser un trabajo experimental, la utilización de GNS3 ayuda a entender de fácil manera las redes y su aplicación es lo más cercana a la realidad.

- Las prácticas de laboratorio desarrolladas en este Trabajo de Integración Curricular permiten al estudiante aprender sobre el protocolo de gestión RESTCONF. A lo largo de todas las prácticas se da un conjunto de herramientas y procedimientos para comprender el protocolo a estudiar, dando topologías sencillas para sentar las bases de una gestión de redes más complejas. Al tener cada práctica de laboratorio, una hoja guía con actividades propuestas como los trabajos preparatorios, procedimiento e informes de manera clara y concisa, permite al estudiante desarrollar sus habilidades en torno al aprendizaje de un nuevo protocolo desarrollando y configurando tareas esenciales para un administrador de red. Además, se tienen videos, reactivos y resoluciones de trabajos preparatorios e informes como material complementario ofrecido para que alumnos, profesores e investigadores tengan una perspectiva más amplia de los contenidos en cada una de las prácticas relacionados con el protocolo a estudiar. Estas hojas guías de laboratorio en su conjunto ayudan a los estudiantes y docentes de la carrera de Telecomunicaciones, y afines, a comprender, instruirse y emplear las funcionalidades que brinda el protocolo RESTCONF, consiguiendo así mejorar sus conocimientos y preparación profesional.

3.3 Recomendaciones

- La utilización de equipos de alta gama es recomendable considerar los requerimientos básicos para su funcionamiento, por lo que, es esencial disponer de un tamaño de memoria RAM de mínimo 4GB y un almacenamiento de disco duro mínimo de 8 GB en la máquina a utilizarse para las prácticas de laboratorio y poder simular/emular sin ningún problema las redes dadas.

- En el capítulo 1, sección Marco Teórico se presentan equipos de red que soportan el protocolo RESTCONF de varias marcas y desarrolladores, por lo que se recomienda explorar esos dispositivos de red u otras versiones del IOS del router Cisco CSR1000v y realizar las prácticas de laboratorio dadas en este Trabajo de Integración Curricular, esto con el objetivo de tener una comparación entre los distintos dispositivos de red y/o versiones, y así escoger el mejor equipo que se adapte a las temáticas planteadas en las prácticas de laboratorio y no requiera una gran cantidad de recursos para su uso.

- El modelo de datos YANG utilizado en las prácticas de laboratorio ha sido el modelo IETF, conociendo que existen dos modelos más del modelo YANG, se recomienda que se estudie y utilicen los modelos OPENCONFIG y nativos, pues dependiendo del modelo que se escoja, se tendrá más información, módulos nuevos y estadísticas que mejorarán la experiencia en el trabajo con RESTCONF.

- Se conoce que la experiencia es mejor tenerla físicamente, pues permite retener más la información y el aprendizaje es óptimo, por lo que, se recomienda que las prácticas de laboratorio dadas en este Trabajo de Integración Curricular sean empleadas en equipos de red (switches o routers) de forma física que soporten el protocolo RESTCONF.

4 REFERENCIAS BIBLIOGRÁFICAS

[1] S. Sinche, D. Raposo, N. Armando, A. Rodrigues, F. Boavida, V. Pereira, and J. S. Silva. “A Survey of IoT Management Protocols and Frameworks,” in *IEEE Communications Surveys & Tutorials,* 2019.

[2] Huawei. “iMaster NCE-Campus V300R020C10 RESTful API Development Guide.” Internet: https://support.huawei.com/enterprise/en/doc/EDOC1100193926, Sep 11, 2021 [May 10, 2022].

[3] J. Domínguez. “Software Defined Networking: The future of Networks.” Master Thesis, Universiad Politécnica de Madrid, Madrid, España, 2021.

[4] R. Vilalta, S. Via, F. Mira, L. Sanabria, R. Martinez, R. Casellas, and J. Alonso-Zarate. “Control and management of a connected car using YANG/RESTCONF and cloud computing,” in *2017 8th International Conference on the Network of the Future (NOF),* 2017

[5] M. Neidinger. *Python Network Programming Techniques: 50 real-world recipes to automate infrastructure networks and overcome networking challenges with Python.* Birmingham, UK: Packt Publishing, 2021.

[6] R. Vilalta, R. Casellas, R. Martínez and R. Muñoz. “Network Programmability and Automation in Optical Networks,” in *23rd IFIP WG 6.10 International Conference ONDM*, 2019.

[7] Huawei. “RESTCONF Configuration - CloudEngine 16800 V200R019C10 Configuration Guide 07.” Internet: https://support.huawei.com/enterprise/en/doc/EDOC1100137878/afa66a1e/restconf-configuration, Jun 11, 2021 [May 10, 2022].

[8] B. Claise, J. Clarke, and J. Lindblad. *Network programmability with YANG: the structure of network automation with YANG, NETCONF, RESTCONF, and gNMI.* Boston, MA: Pearson Education, 2019.

[9] A. Bierman, YumaWorks, M. Bjorklund, Tail-f Systems, K. Watsen and Jupiter Networks. “RESTCONF Protocol.” IETF RFC 8040, Ene 2017.

[10] Cisco. “RESTCONF Programmable Interface.” Internet: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/prog/configuration/166/b_166_programmability_cg/restconf_prog_int.pdf, [May 10, 2022].

[11] M. Bjorklund, Ed. and Tail-f Systems. “YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF).” IETF RFC 6020, Oct 2010.

[12] K. Gray and T. D. Nadeau. (2016). *Network Function Virtualization*. [On-line] Disponible: https://books.google.com.ec/books?id=JwDJBAAAQBAJ&lpg=PA92&dq=data%20modeling%20yang&hl=es&pg=PA92#v=onepage&q=data%20modeling%20yang&f=false [May 30, 2022].

[13] J.J. Tortajada. (2014, May 15). *La guía definitiva de XML: ¡¡XML, JSON y mucho más!!.* [On-line] Disponible: https://books.google.com.ec/books?id=kI-gAwAAQBAJ [May 11, 2022].

[14] S. Aguirre. (2020, Jun 12). *JSON - Vol.1: Primeros pasos - Sintaxis - Tipos de datos.* (2da Edición) [On-line] Disponible: https://books.google.com.ec/books?id=K-fqDwAAQBAJ&lpg=PP1&hl=es&pg=PP1#v=onepage&q&f=false [May 11, 2022].

[15] S. Guruprasad, R. Ricci and J. Lepreau, "Integrated network experimentation using simulation and emulation," in *First International Conference on Testbeds and Research Infrastructures for the DEvelopment of NeTworks and COMmunities*, 2005.

[16] J. C. Neumann. *The Book of GNS3: Built virtual network labs using Cisco, Juniper, and more.* San Francisco, CA: No Starch Press, Inc., 2015.

[17] T. Wangchuk. *Virtualize Network Test-Labs: Using GNS3 and VirtualBox*, 2018

[18] Cisco DevNet. “Model Driven Programmability.” Internet: https://developer.cisco.com/site/standard-network-devices/ [May 12, 2022].

[19] Juniper. “Juniper Northstar API Documentation.” Internet: https://www.juniper.net/documentation/en_US/northstar6.2.0/information-products/api-ref/api-ref-northstar-restconf.html, Nov 25, 2020 [May 13, 2022].

[20] Juniper. “Example: Configuring the REST API | Junos OS | Juniper Networks.” Internet: https://www.juniper.net/documentation/us/en/software/junos/rest-api/topics/example/rest-api-configuring-example.html, Nov 5, 2021 [May 13, 2022].

[21] Juniper OS. “REST API Guide.” Internet: https://www.juniper.net/documentation/us/en/software/junos/rest-api/rest-api.pdf, Dic 15, 2021 [May 13, 2022].

[22] Huawei. “NetEngine AR V600R021C00 Configuration Guide - System Management Configuration.” Internet: https://support.huawei.com/enterprise/en/doc/EDOC1100213454/59f21507/configuring-restconf-for-device-management, Sep 24, 2021 [May 14, 2022].

[23] Huawei. “USG6000E V600R007C20 RESTCONF API Development Guide(doc).” Internet: https://support.huawei.com/enterprise/en/doc/EDOC1100163128, May 20, 2021 [May 14, 2022].

[24] Cisco. “Cisco CSR 1000v and Cisco ISRv Software Configuration Guide.” Internet: https://www.cisco.com/c/en/us/td/docs/routers/csr1000/software/configuration/b_CSR1000v_Configuration_Guide.pdf, Abr 2022 [May 28, 2022]

[25] GNS3. “Setup wizard with the GNS3 VM. (2021)”. Internet: https://docs.gns3.com/docs/getting-started/setup-wizard-gns3-vm/ [May 28, 2022]

[26] Cisco. “Programmability Configuration Guide, Cisco IOS XE Amsterdam 17.3.x” Internet: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/prog/configuration/173/b_173_programmability_cg/restconf_protocol.html#id_70431, Abr 19, 2022 [May 26, 2022].

[27] M. Albrecht. “RESTCONF Tutorial - Everything you need to know about RESTCONF in 2020.” Internet: https://ultraconfig.com.au/blog/restconf-tutorial-everything-you-need-to-know-about-restconf-in-2020/, May 16, 2020 [May 28, 2022].

[28] E. Nilsen-Nygaard. “yang/vendor/cisco/xe/1731/ietf-restconf.yang.” Internet: https://github.com/YangModels/yang/blob/main/vendor/cisco/xe/1731/ietf-restconf.yang, Ago 4, 2020 [Jun 13, 2022].

5 ANEXOS

ANEXO I. Hoja Guía Práctica 2

ANEXO II. Hoja Guía Práctica 3

ANEXO III. Hoja Guía Práctica 4

ANEXO IV. Hoja Guía Práctica 5

ANEXO V. Resolución de Trabajos Preparatorios

ANEXO VI. Enlace de Reactivos

ANEXO VII. Enlace de Videos

ANEXO VIII. Resolución de Informes

ANEXO I

PRÁCTICA N° 2

1 TEMA

LENGUAJE DE MODELADO DE DATOS YANG

2 OBJETIVOS

2.1 Analizar el modelo de datos YANG.
2.2 Identificar el modelo de datos YANG con el comando `pyang`.
2.3 Comparar entre los tipos de datos estructurados XML y JSON.

3 MARCO TEÓRICO

Lenguaje de modelado de datos YANG

YANG (*Yet Another Next Generation*) es un lenguaje de modelado de datos que define una jerarquía sobre los datos que pueden ser usados para operaciones base como incluir configuración, estado de datos, llamadas a procedimientos remotos RPC (*Remote Procedure Call*) y notificaciones. Esto permite una completa descripción sobre todos los datos que son enviados entre un cliente y un servidor NETCONF (*Network Configuration Protocol*). El lenguaje de modelado de datos YANG fue desarrollado por la IETF (*Internet Engineering Task Force*) y se publicó en 2010 como RFC 6020 [1].

YANG es un lenguaje modular que representa estructuras de datos en un formato de árbol. El lenguaje de modelado de datos contiene una serie de tipos de datos intrínsecos, como cadenas y números enteros. Estos tipos básicos pueden usarse para construir tipos más complejos formando un conjunto muy rico de capacidades. YANG proporciona una clara descripción de los nodos y define como se interactúa entre ellos [2].

YANG estructura los modelos en módulos y submódulos. Un módulo puede importar datos de otro módulo externo e incluir datos de submódulos. La jerarquía puede aumentarse, permitiendo que un módulo agregue nodos de datos a la jerarquía definida en otro módulo. Los módulos YANG pueden ser traducidos en su equivalente en XML (*Extensible Markup Language*), permitiendo a las aplicaciones utilizar herramientas que actúen sobre datos en XML que son más comunes [3].

Se debe recordar que el protocolo RESTCONF trabaja sobre los fundamentos de NETCONF sobre HTTPS, por lo que RESTCONF también trabaja con el lenguaje de modelado de datos YANG y con los tipos de datos estructurados XML y JSON [4].

4 TRABAJO PREPARATORIO

a) Describir los modelos de datos YANG de IETF, OPENCONFIG y nativos.
b) Consultar acerca de qué es un `namespace` y qué relación tiene con los modelos de datos.
c) Describir los nodos del modelo de datos YANG.
d) Consultar qué es `pyang` y para qué es empleado.
e) Consultar sobre los formatos XML y JSON, diferencias y ejemplos.

5 EQUIPOS Y MATERIALES

5.1 Hardware

- Computadora.

5.2 Software

- VirtualBox.
- VMware Workstation Pro.
- GNS3 y GNS3 VM.

6 PROCEDIMIENTO

6.1 Topología de red

a) Abra GNS3 y realice la topología de red que se muestra en la Figura 1.

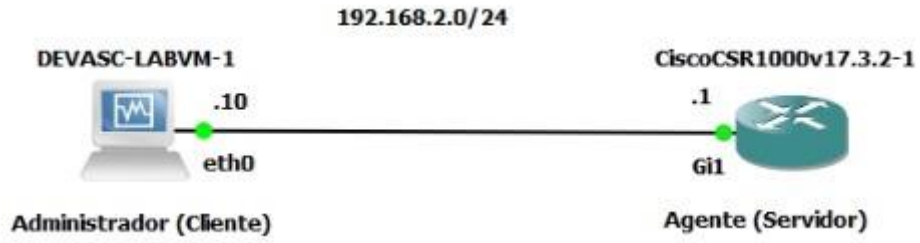

Figura 1. Topología de red a implementar

b) Verificar la conectividad entre el administrador y el agente.
c) Verificar la conectividad del administrador con internet.

Nota: Utilizar el proyecto creado en la práctica 1.

6.2 Explorar modelos YANG en GITHUB

a) En el administrador, abra el navegador Chromium y vaya al siguiente enlace: https://github.com/YangModels/yang
b) En la página modelos de datos YANG de Cisco IOS XE versión 17.3.1 y vaya a las siguientes carpetas: *vendor > cisco > xe > 1731.*
c) Busque el modelo IETF y haga clic en el archivo `ietf-interfaces.yang`. Explore todo el modelo de datos para identificar, seleccionar y describir un nodo `leaf`, un `leaf-list`, un `container` y un `list`.

NOTA: Un módulo es aquel que define un único modelo de datos. En este literal se explorará el modelo de datos "`ietf-interfaces`" que tiene asociado su identificación única por el `namespace`: "`urn:ietf:params:xml:ns:yang:ietf-interfaces`" como se muestra en la Figura 2. Este modelo es establecido por la IETF y muestra una organización jerárquica de los datos que se encuentran en las interfaces de los dispositivos de red.

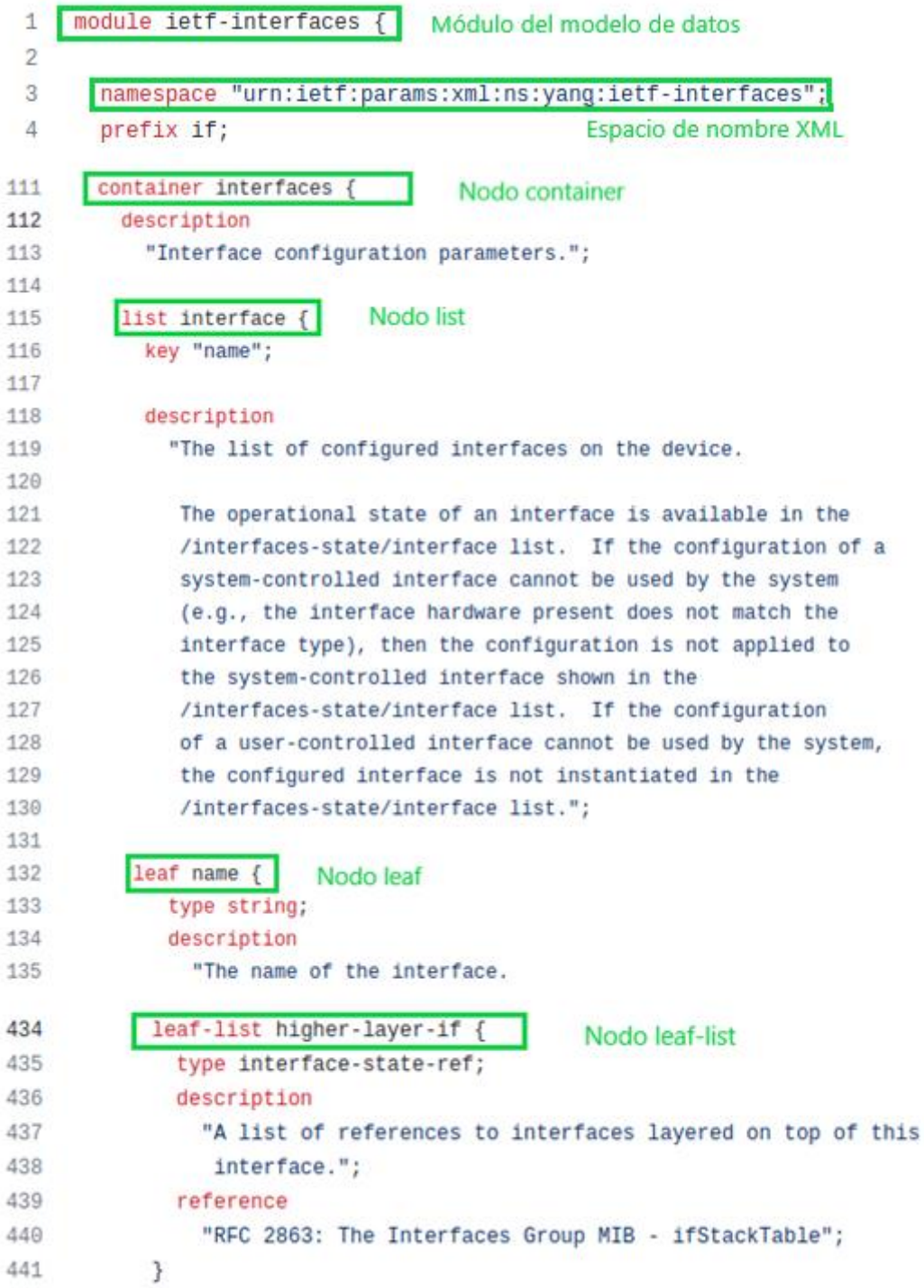

Figura 2. Módulo `ietf-interfaces`

d) En la lista encuentre los modelos OPENCONFIG y dar clic en el `openconfig-interfaces.yang`. Examinar todo el modelo de datos e identificar, escoger y describir un nodo `leaf`, un `leaf-list`, un `container` y un `list`.

Nota: En este literal se explorará un modelo de datos establecido por la organización OPENCONFIG. Al igual que el modelo de IETF se describe

una organización jerárquica de los datos que contienen las interfaces de los dispositivos de red, sin embargo, la forma en cómo se organizan los datos es muy diferente, como se muestra en la Figura 3.

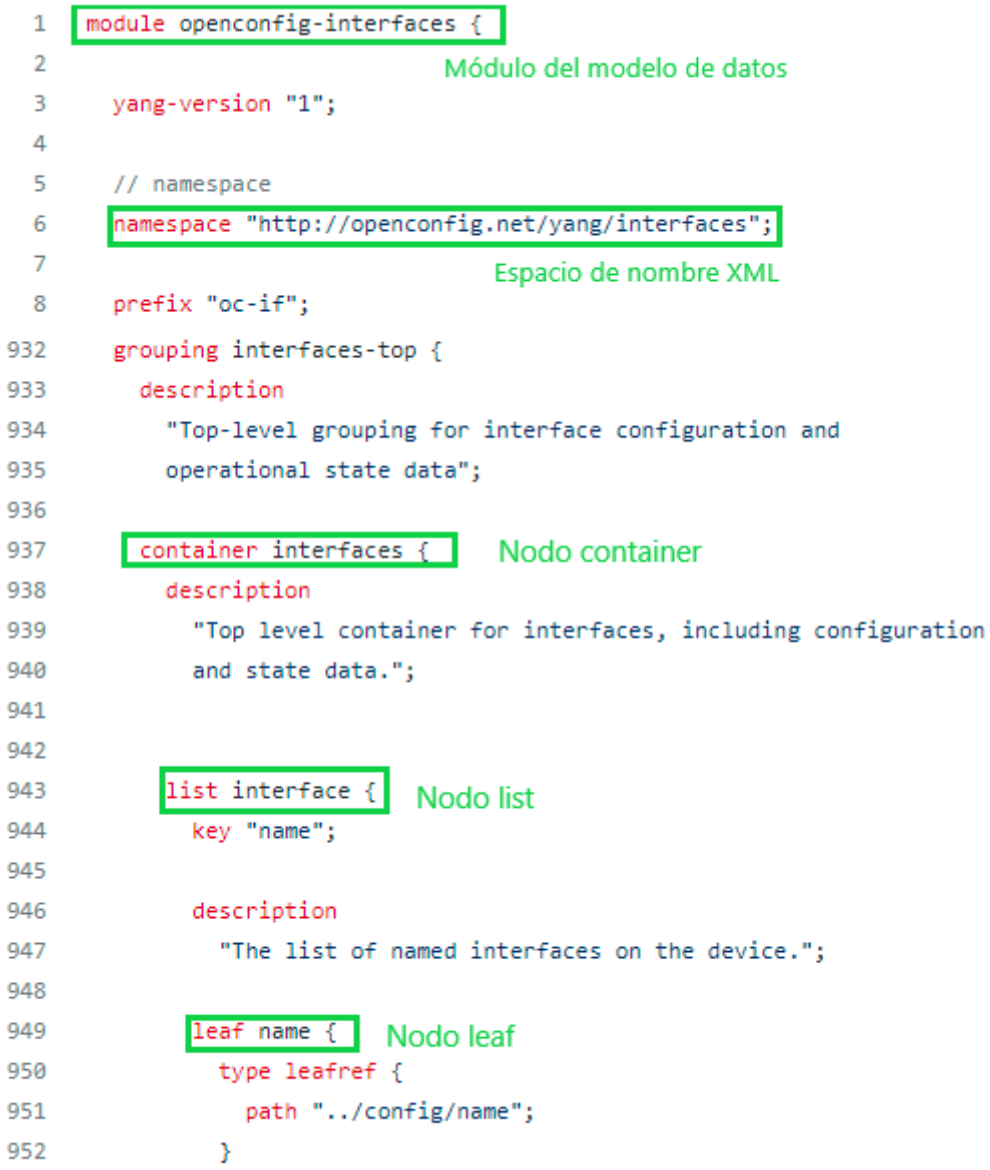

Figura 3. Módulo `openconfig-interfaces`

Se puede notar que este modelo de datos tiene asociado el módulo "`openconfig-interfaces`" con su respectivo `namespace`: `http://openconfig.net/yang/interfaces`.

6.3 Explorar modelos YANG con `pyang`

a) En el administrador, crear en `/labs/devnet-src/` una carpeta llamada `pyang`.

```
devasc@labvm:~$ cd labs/devnet-src/
devasc@labvm:~/labs/devnet-src$ mkdir pyang
```

b) En cada una de las direcciones web de GitHub del literal 6.2., dirigirse al inicio de estos módulos, seleccionar la opción *Raw* y copiar el URL. Luego emplear el comando `wget [URL]` para guardar los modelos en la carpeta `pyang`.

```
devasc@labvm:~/labs/devnet-src$ cd pyang/
devasc@labvm:~/labs/devnet-src/pyang$ wget
https://raw.githubusercontent.com/YangModels/yang/main/vendor
/cisco/xe/1731/ietf-interfaces.yang
devasc@labvm:~/labs/devnet-src/pyang$ wget
https://raw.githubusercontent.com/YangModels/yang/main/vendor
/cisco/xe/1731/openconfig-interfaces.yang
```

c) Verificar si `pyang` está instalado con el comando `pyang -v`

```
devasc@labvm:~/labs/devnet-src/pyang$ pyang -v
pyang 2.2.1
```

d) Visualizar las opciones del comando `pyang` con el comando `pyang -h | more`.

```
devasc@labvm:~/labs/devnet-src/pyang$ pyang -h | more
```

Pregunta 1. ¿A qué estructuras se puede transformar el modelo YANG y con cuál opción?

e) Transformar los modelos de datos en formato de árbol. Seguramente notará que ambos modelos son mucho más fáciles de entender en su totalidad. Analizar la salida del módulo `ietf-interfaces.yang` dada en la Figura 4.

```
devasc@labvm:~/labs/devnet-src/pyang$ pyang -f tree ietf-
interfaces.yang
```

Pregunta 2. ¿Qué propiedades tienen las representaciones gráficas del diagrama del árbol `rw`, `ro`, `x`, `?`, `!` y `*`?
Pregunta 3. ¿Qué comando usaría para transformar a modo árbol el módulo de interfaces de OPENCONFIG?
Pregunta 4. Realizar el análisis hecho en la Figura 4 con la salida de la Pregunta 3.

Figura 4. Módulo `ietf-interfaces` en forma de árbol

6.4 Representación de datos YANG

b) Abrir el archivo *example-ietf-interfaces-data.xml* del siguiente link: https://github.com/alexrch97/Analisis-RESTCONF

c) El archivo XML que es un ejemplo de datos devueltos desde un dispositivo de red que muestra información sobre sus interfaces. Examinar el archivo como se muestra en la Figura 5.

Nota: Al examinar el ejemplo notará que la información mostrada en formato XML sigue la organización jerárquica del modelo de datos “ietf-interfaces”.

Nota: El protocolo RESTCONF, a diferencia de NETCONF, puede utilizar dos tipos de representación de datos, el formato XML y JSON.

Pregunta 5. Realizar los literales a) y b) para el archivo *example-ietf-interfaces-data.json.*

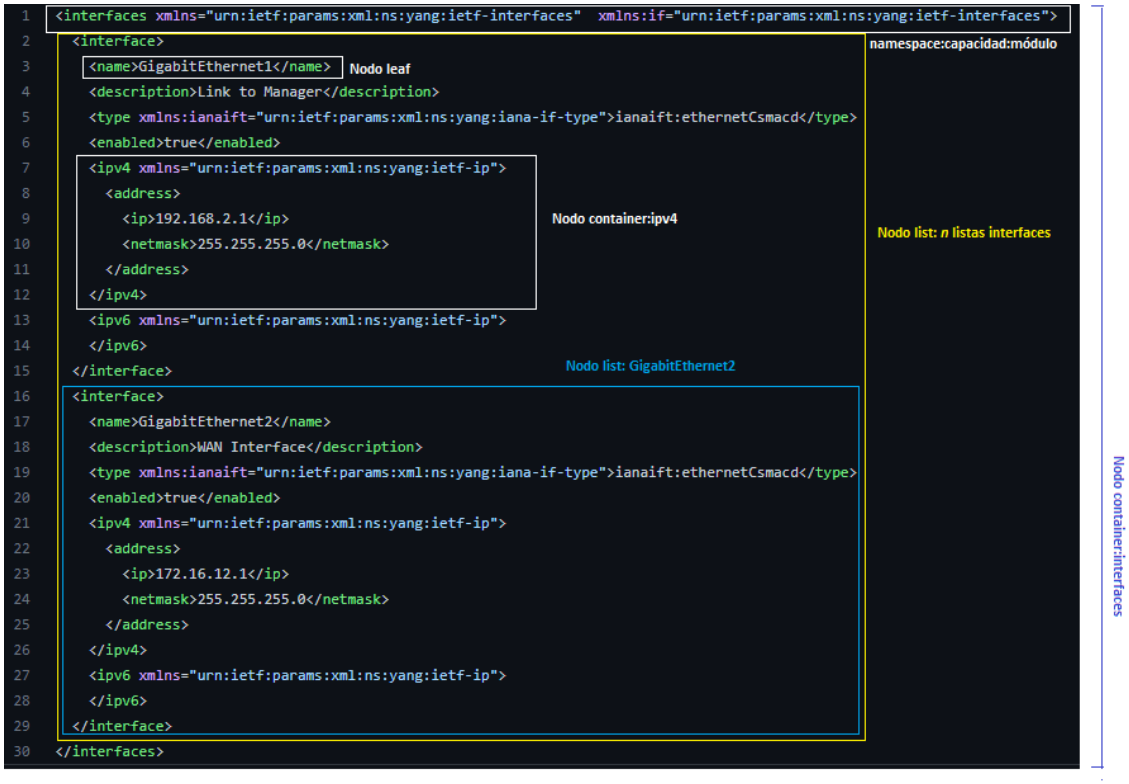

Figura 5. Ejemplo de datos mostrados en formato XML

7 INFORME

7.1 Proporcione la configuración construida en laboratorio con una explicación adecuada de los resultados que se muestran.

7.2 Responder las preguntas dadas en el procedimiento.

7.3 Conclusiones y Recomendaciones

7.4 Bibliografía

8 REFERENCIAS

[1] M. Bjorklund, Ed. and Tail-f Systems. “YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF).” IETF RFC 6020, Oct. 2010.

[2] K. Gray and T. D. Nadeau. (2016). *Network Function Virtualization*. [On-line] Disponible: https://books.google.com.ec/books?id=JwDJBAAAQBAJ&lpg=PA92&dq=data%20modeling%20yang&hl=es&pg=PA92#v=onepage&q=data%20modeling%20yang&f=false [May. 30, 2022].

[3] T. Cucharero. “Control y gestión de sondas de monitorización Ethernet usando NETCONF y modelos de datos YANG.” M.A. tesis, Universidad Autónoma de Madrid, España, 2017.

[4] Cisco. “RESTCONF Programmable Interface.” Internet: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/prog/configuration/166/b_166_programmability_cg/restconf_prog_int.pdf, [May. 10, 2022].

ANEXO II

PRÁCTICA N° 3

1 TEMA

SEGURIDAD Y MÉTODOS HTTP

2 OBJETIVOS

2.4 Implementar ACLs para seguridad.
2.5 Analizar la herramienta cURL para el protocolo RESTCONF.
2.6 Identificar los diferentes métodos RESTCONF.

3 MARCO TEÓRICO

3.1 ACL

Una ACL (*Access-list*) es un conjunto de reglas definidas para controlar el tráfico de red y reducir los ataques, protegiendo la red. Las ACLs son usadas para filtrar y así permitir o negar la entrada o salida de paquetes de la red [1].

3.2 cURL

La herramienta cURL realiza operaciones HTTP y del Protocolo de Transferencia de archivos (FTP). Aunque no está diseñada específicamente para interactuar con las interfaces de programación de aplicaciones (API), funciona bien para probar la funcionalidad REST. cURL obtiene las capacidades de un dispositivo compatible con RESTCONF. De forma predeterminada, cURL realiza una operación HTTP GET en la URL de destino, por lo que no es necesario especificar ningún verbo de acción adicional para que obtenga el contenedor de estados [2].

3.3 Sesiones RESTCONF

A diferencia de NETCONF, RESTCONF no tiene una sesión en la que se realiza un intercambio de capacidades. es decir, no hay ningún mensaje `<hello>` [2]. Para que un cliente RESTCONF determine qué módulos son compatibles con un servidor RESTCONF, ese servidor debe admitir la biblioteca ietf-yang-library (RFC 7895) [3].

3.4 Métodos HTTP

Un método de solicitud (Request method) es una palabra clave específica que describe el tipo de operación que queremos que realice el servidor al recibir y procesar una solicitud HTTP. Los métodos de solicitud más importantes se pueden observar en la Tabla 1 [4].

Tabla 1. Métodos de solicitud

Opción	Método
GET	Leer información.
PATCH	Sobrescribir o modificar los recursos.
PUT	Reemplazar el recurso.
POST	Crear o actualizar un recurso.
DELETE	Eliminar el recurso.

4 TRABAJO PREPARATORIO

a) Consultar el comando en un router Cisco CSR1000V para habilitar ACLs en RESTCONF.
b) Consultar el comando `curl` y describir las opciones que tiene.
c) Consultar los campos de encabezado de una solicitud HTTP.
d) Consultar cómo se crea una URL a partir de un módulo YANG de tipo IETF.

5 EQUIPOS Y MATERIALES

5.1 Hardware

- Computadora.

5.2 Software

- VirtualBox.
- VMware Workstation Pro.
- GNS3 y GNS3 VM.

6 PROCEDIMIENTO

6.1 Establecer una ACL para RESTCONF

a) Abra GNS3 y cree la topología de red que se muestra en la Figura 1.

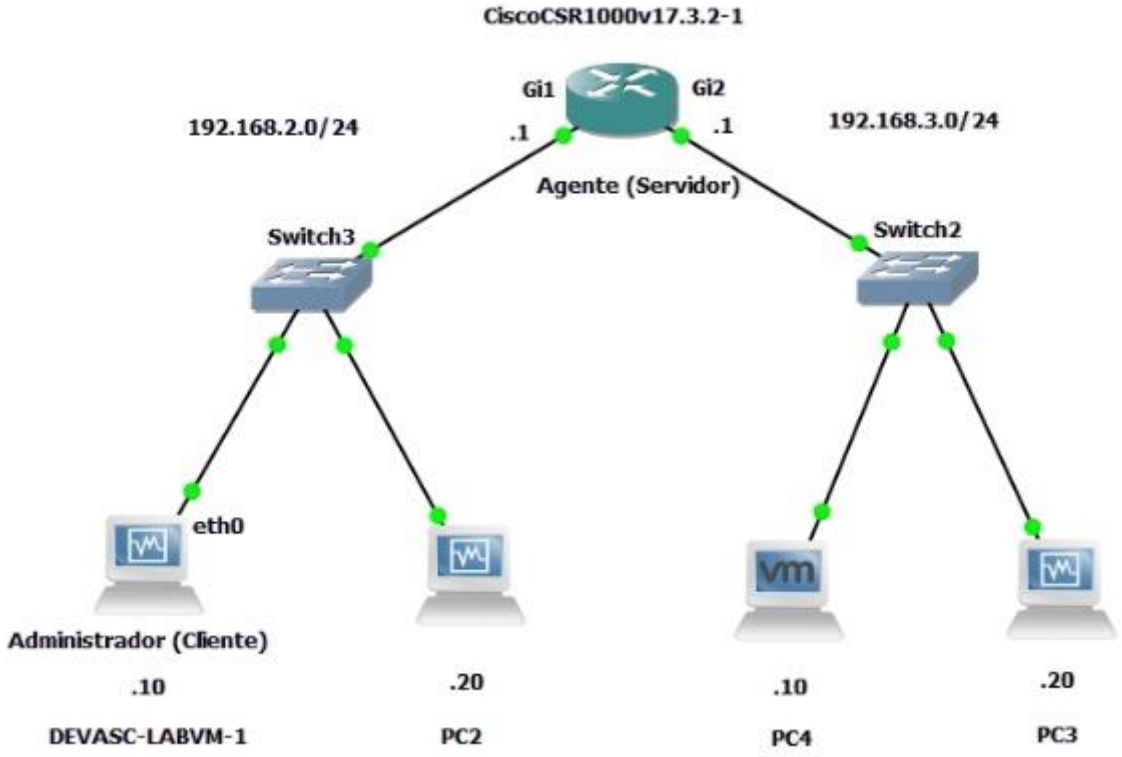

Figura 1. Topología de red a implementar

NOTA: Las PCs a utilizar son máquinas virtuales que se pueden agregar a GNS3 y configurar, tal como en la Práctica 1.

b) Iniciar el router cisco CSR1000v y configurar las interfaces GigabitEthernet1 y GigabitEthernet2 con las direcciones IP dadas en la Figura 1.

c) Verificar la conectividad entre las máquinas virtuales con el router en cada una de sus respectivas redes.

d) Crear un usuario `admin:admin` con privilegio 15 (privileged EXEC).

e) Configure el nombre y el dominio del enrutador Cisco CSR1000v.

f) Habilitar el servicio RESTCONF.

g) Realizar la siguiente petición desde el terminal de los clientes.

```
curl -k https:// {IP servidor}:443/restconf/ -u admin:admin
```

Pregunta 1. ¿Cuál es la IP del servidor en la red 192.168.2.0/24? y ¿en la 192.168.3.0/24? ¿Por qué son las mismas?
Pregunta 2. ¿El agente ha mostrado algún log?, ¿qué indica?

h) Realizar una ACL llamada `permit_REST` en el router que permita ingresar al servidor solo desde la red 192.168.2.0/24.

```
CSR1k>enable
CSR1k#configure terminal
CSR1k(config)#ip access-list standard permit_REST
CSR1k(config-std-nacl)#permit 192.168.2.0 0.0.0.255
CSR1k(config-std-nacl)#end
```

i) Habilitar el protocolo RESTCONF solo para la red de la ACL previa.

```
CSR1k>enable
CSR1k#configure terminal
CSR1k(config)#restconf ipv4 access-list name permit_REST
```

j) Realizar la petición del literal g) para cada cliente.

Pregunta 3. ¿Cuáles son los resultados?, comente.
Pregunta 4. ¿Cuál sería el procedimiento para permitir ingresar a RESTCONF solo desde el administrador DEVASC-LABVM?
Pregunta 5. Describa las ventajas de este tipo de seguridad para el manejo del protocolo RESTCONF.

6.2 Realizar peticiones HTTP con cURL

a) Desde el terminal del administrador DEVASC-LABVM, realizar una petición GET con cURL para tener información de las interfaces del agente.

```
devasc@labvm:~$ curl -k
https://192.168.2.1:443/restconf/data/ietf-
interfaces:interfaces -u admin:admin
```

Pregunta 6. ¿Cuál es el resultado?, ¿qué características muestra?
Pregunta 7. ¿En qué formato muestra el resultado?, ¿cuál es la opción para cambiar a JSON?

b) Bajar la interfaz G2 con una petición PATCH, en formato JSON.

```
devasc@labvm:~$ curl -i -k -X "PATCH"
"https://192.168.2.1:443/restconf/data/ietf-
interfaces:interfaces/interface=GigabitEthernet2" \
> -H 'Content-Type: application/yang-data+json' \
> -H 'Accept: application/yang-data+json' \
> -u 'admin:admin' \
> -d $'{
> "ietf-interfaces:interface": {
> "enabled": false
> }
> }'
```

Pregunta 8. ¿Cuál es el resultado?, ¿qué características muestra?

Pregunta 9. ¿Es eficaz la herramienta cURL para gestionar la red con RESTCONF?

7 INFORME

7.1 Proporcione la configuración construida en laboratorio con una explicación adecuada de los resultados que se muestran.
7.2 Responder las preguntas dadas en el procedimiento.
7.3 Conclusiones y Recomendaciones
7.4 Bibliografía

8 REFERENCIAS

[1] D. Gonzalez. (2022). *The Only Networking Book You Will Need: Networking Explained*. (1st edition) [On-line]. Available: https://books.google.com.ec/books?id=lCVlEAAAQBAJ&lpg=PT148&dq=acl%20networking&hl=es&pg=PP1#v=onepage&q&f=false [June 3, 2022].

[2] B. Claise, J. Clarke, and J. Lindblad. *Network programmability with YANG: the structure of network automation with YANG, NETCONF, RESTCONF, and gNMI.* Boston, MA: Pearson Education, 2019.

[3] A. Bierman, YumaWorks, M. Bjorklund, Tail-f Systems, K. Watsen and Jupiter Networks. “YANG Module Library.” IETF RFC 7895, Jun. 2016.

[4] M. Neidinger. *Python Network Programming Techniques: 50 real-world recipes to automate infrastructure networks and overcome networking challenges with Python.* Birmingham, UK: Packt Publishing, 2021.

ANEXO III

PRÁCTICA N° 4

1 TEMA

GESTIÓN RESTCONF CON POSTMAN

2 OBJETIVOS

2.1 Analizar la herramienta Postman para el protocolo RESTCONF.
2.2 Configurar interfaces y ruta estática mediante RESTCONF.
2.3 Diagnosticar la red mediante RESTCONF.

3 MARCO TEÓRICO

POSTMAN

Según la página oficial [1], Postman es una plataforma para construir y usar APIs (*Application Programming Interfaces*), por lo que, es uno de los programas más populares para trabajar en el desarrollo de estas. Postman simplifica cada paso del ciclo de vida de la API y agiliza la colaboración para que pueda crear mejores APIs y más rápido. Se encuentra disponible para todos los sistemas operativos, como Windows y Linux.

Gracias a Postman podemos guardar todas las *request* que queramos, para tenerlas preparadas y poder ejecutarlas las veces que haga falta. Además, Postman permite trabajar cómodamente con todos los métodos de HTTP, como GET, POST, PUT, PATCH, DELETE [2].

4 TRABAJO PREPARATORIO

a) Identificar las principales áreas de POSTMAN.
b) Consultar las estructuras de los siguientes módulos YANG:
   - ietf-restconf-monitoring.yang
   - ietf-interfaces.yang
   - ietf-routing.yang
c) Consultar sobre los códigos de respuesta a las peticiones HTTP.

5 EQUIPOS Y MATERIALES

5.1 Hardware
- Computadora.

5.2 Software
- VirtualBox.
- VMware Workstation Pro.
- GNS3 y GNS3 VM.

6 PROCEDIMIENTO

6.1 Introducción a Postman

a) Abra GNS3 y cree la topología de red que se muestra en la Figura 1.

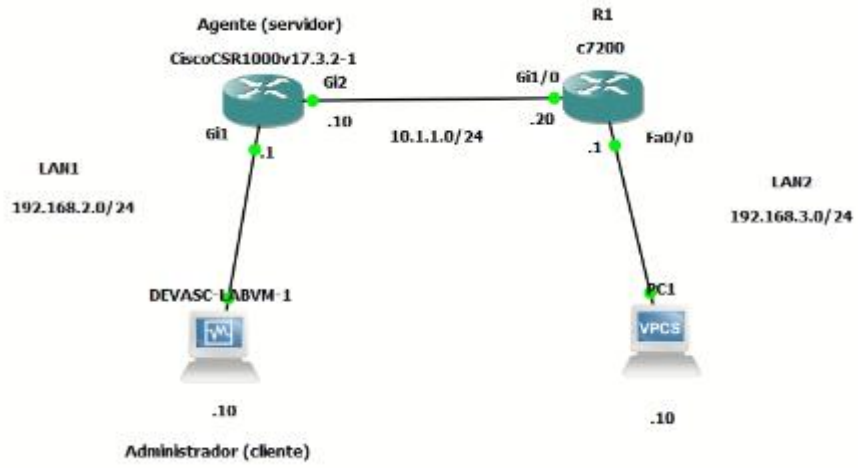

Figura 1. Topología de red a implementar

b) Iniciar el router cisco CSR1000v y configurar la interfaz GigabitEthernet1 con la dirección IP dada en la Figura 1.

c) Iniciar el router cisco c7200 y configurar la interfaz GigabitEthernet1/0 y FastEthernet0/0 con las direcciones IP dadas en la Figura 1.

d) Verificar la conectividad de los hosts con sus respectivos routers.

e) En el agente, crear un usuario `admin:admin` con privilegio 15 (privileged EXEC).

f) Configure el nombre y el dominio del enrutador Cisco CSR1000v.

g) Habilitar el servicio RESTCONF.

h) Realizar la siguiente petición desde el administrador y verificar el funcionamiento de RESTCONF.

```
curl -k https://192.168.2.1:443/restconf/ -u admin:admin
```

i) En el administrador, ejecutar el programa Postman.

j) En Postman, ir a *Files* > *Settings* > *General*, deshabilitar la verificación del certificado SSL y cerrar la ventana.

Pregunta 1. En el comando cURL, ¿por cuál opción se está realizando este paso?

k) En *Collections*, crear una colección llamada Práctica 4 y seleccionar *Add Request.* Se va a crear la primera petición API al servidor RESTCONF, nombrar a esta primera petición *Información RESTCONF*.

l) Para todas las APIs, se debe configurar la autorización, en modo básico con las credenciales del usuario admin del agente. Además, establecer que el

contenido tanto de petición como respuesta, se den en el formato JSON. Esto se puede observar en las Figuras 2 y Figura 3.

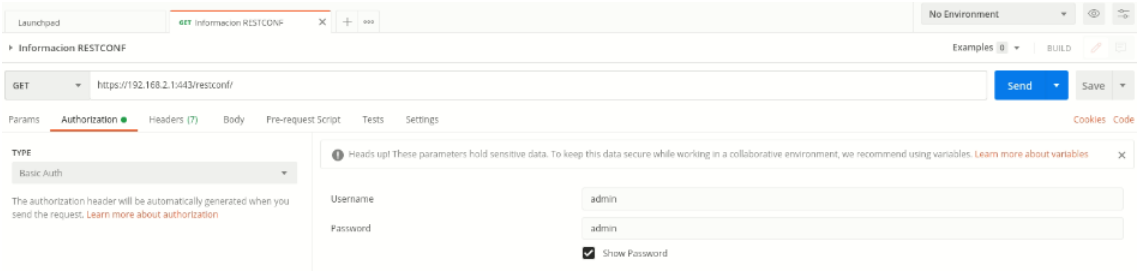

Figura 2. Autorización de tipo básica

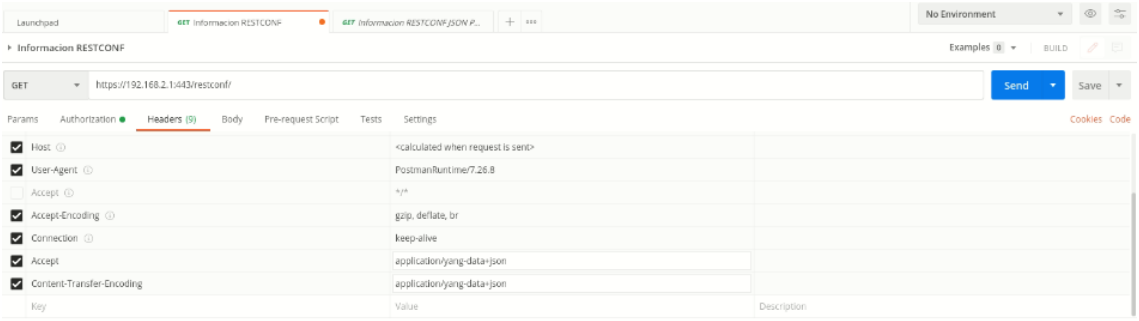

Figura 3. Configuración de encabezados para formato JSON

m) Cómo se muestra en la Figura 2, establecer el API en modo GET, con la misma URL de cURL del literal h) y hacer clic en Send.

Pregunta 2. ¿Cuál es el resultado?, ¿qué código de respuesta dio el servidor?

6.2 Configuración de interfaces con RESTCONF mediante Postman.

a) Crear una API llamada *Información G2*, que muestra la información de la interfaz GigabitEthernet2 del agente.

GET https://192.168.2.1/restconf/data/ietf-interfaces:interfaces/interface=GigabitEthernet2

Pregunta 3. ¿Qué resultado muestra esta API?, ¿la interfaz Gi2, que información tiene?

b) Crear una API llamada *Cambiar G2*, que actualice la configuración de la interfaz, conforme a la topología de la Figura 1. En *Body*, seleccionar *raw* y JSON, ahí incluir el Código 1.

PATCH https://192.168.2.1/restconf/data/ietf-interfaces:interfaces/interface=GigabitEthernet2

```
{
    "ietf-interfaces:interface": {
        "enabled": true,
        "description": "To R1",
```

```
        "ietf-ip:ipv4": {
            "address": [
                {
                    "ip": "10.1.1.10",
                    "netmask": "255.255.255.0"
                }
            ]
        }
    }
}
```

Código 1. Configuración de dirección IP y descripción de Gi2 en JSON

Pregunta 4. ¿Qué resultado muestra esta API?
Pregunta 5. Enviar el API *Información G2,* ¿cuál es el nuevo resultado?
Pregunta 6. En el agente, correr el comando para ver el estado de las interfaces ¿cuál es el resultado?

c) Crear una API llamada *Información LB1*, que muestra la información de la interfaz LoopBack1 del agente.

GET https://192.168.2.1/restconf/data/ietf-interfaces:interfaces/interface=Loopback1

Pregunta 7. ¿Qué resultado muestra esta API?

d) Crear una API llamada *Crear LB1*, que cree la interfaz Loopback1, y la configure con el Código 2.

PUT https://192.168.2.1/restconf/data/ietf-interfaces:interfaces/interface=Loopback1

```
{
 "ietf-interfaces:interface": {
   "name": "Loopback1",
   "description": "Loopback 1",
   "type": "iana-if-type:softwareLoopback",
   "enabled": true,
   "ietf-ip:ipv4": {
   "address": [
      {
        "ip": "20.1.1.1",
        "netmask": "255.255.255.0"
      }
    ]
 },
 "ietf-ip:ipv6": {}
 }
}
```

Código 2. Configuración de dirección IP y descripción de LB1 en JSON

Pregunta 8. ¿Qué resultado muestra esta API?
Pregunta 9. Enviar el API *Información LB1,* ¿cuál es el nuevo resultado?

Pregunta 10. En el agente, correr el comando para ver el estado de las interfaces ¿cuál es el resultado?

e) Crear el API *Borrar LB1*.

Pregunta 11. ¿Cuál es el método y la URL para esta API?
Pregunta 12. Comprobar que la interfaz LB1 ha sido eliminada.

6.3 Creación de ruta estática con RESTCONF mediante Postman.

a) En el router R1, crear la ruta estática para que la LAN2 tenga conectividad con la LAN1.

```
R1# configure terminal
R1(config)# ip route 192.168.2.0 255.255.255.0 10.1.1.10
```

b) En el administrador, crear el API *Información ROUTING* que obtendrá tabla de enrutamiento del agente.

GET https://192.168.2.1/restconf/data/ietf-routing:routing-state

Pregunta 13. ¿Qué resultado muestra esta API?,¿el resultado muestra algún tipo de enrutamiento a la LAN2?

c) Crear el API *Crear ruta estática,* que añadirá la ruta estática para tener conectividad desde la LAN1 hacia la LAN2 con el código 3 como cuerpo.

PUT https://192.168.2.1:443/restconf/data/ietf-routing:routing/routing-instance=default/routing-protocols/routing-protocol=static,1

```
{
    "ietf-routing:routing-protocol": {
        "type": "ietf-routing:static",
        "name": "1",
        "static-routes": {
            "ietf-ipv4-unicast-routing:ipv4": {
                "route": [
                    {
                        "destination-prefix": "192.168.3.0/24",
                        "next-hop": {
                            "next-hop-address": "10.1.1.20"
                        }
                    }
                ]
            }
        }
    }
}
```

Código 3. Configuración de ruta estática hacia la LAN2

Pregunta 14. ¿Qué resultado muestra esta API?
Pregunta 15. Enviar el API *Información ROUTING,* ¿cuál es el nuevo resultado?
Pregunta 16. En el agente, correr el comando `show ip route`, ¿cuál es el resultado?

Pregunta 17. Realizar un ping desde el administrador a la PC1, mostrar el resultado.

6.4 Análisis de la red con RESTCONF mediante Postman.

a) Crear el API *Estadísticas RESTCONF.*

GET https://192.168.2.1/restconf/data/ietf-restconf-monitoring:restconf-state

Pregunta 18. Realizar un análisis de la información de esta API.

b) Crear el API *Estadísticas INTERFACES*

GET https://192.168.2.1/restconf/data/ietf-interfaces:interfaces-state/

Pregunta 19. Realizar un análisis de la información de esta API.

7 INFORME

7.1 Proporcione la configuración construida en laboratorio con una explicación adecuada de los resultados que se muestran.
7.2 Responder las preguntas dadas en el procedimiento.
7.3 Conclusiones y Recomendaciones
7.4 Bibliografía

8 REFERENCIAS

[1] Postman. “What is Postman?” Internet: https://www.postman.com/, 2022 [Jul. 2, 2022].
[2] M. A. Álvarez. “Cómo usar Postman para probar nuestras APIs.” Internet: https://desarrolloweb.com/articulos/como-usar-postman-probar-api, Oct. 19, 2021 [Jul. 2, 2022]

ANEXO IV

PRÁCTICA N° 5

1 TEMA

GESTIÓN RESTCONF CON PYTHON

2 OBJETIVOS

2.1 Analizar la herramienta Python para el uso del protocolo RESTCONF.
2.2 Configurar interfaces y ruta estática mediante RESTCONF.
2.3 Diagnosticar la red mediante RESTCONF.

3 MARCO TEÓRICO

Python

Python es un lenguaje de programación orientado a objetos de alto nivel con semántica dinámica incorporada, principalmente para el desarrollo web. La implementación de Python comenzó en diciembre de 1989 cuando Guido Van Rossum inició un proyecto de lenguaje de programación fácil de usar y fácil de aprender. El nombre "Python" proviene de la afición de Van Rossum por el grupo Monty Python.

Es relativamente simple, por lo que es fácil de aprender. Además, admite el uso de módulos y paquetes, lo que significa que los programas se pueden diseñar de forma modular y el código se puede reutilizar en diferentes proyectos. Una vez que se ha desarrollado un módulo o paquete, puede extenderse para su uso en otros proyectos e importarse o exportarse fácilmente [1].

Desde su desarrollo a principios de la década de 1990, Python se ha convertido en uno de los lenguajes de programación más utilizados en el mundo. Su poderosa sintaxis, facilidad de uso y capacidades multiplataforma, combinadas con un rico ecosistema de funciones preconstruidas, lo han convertido en uno de los lenguajes más utilizados en el campo de la automatización de infraestructura. Desde scripts de automatización simples hasta aplicaciones web complejas, Python puede manejar una amplia variedad de tareas [2].

4 TRABAJO PREPARATORIO

a) Consultar sobre las librerías:
   - `requests`
   - `sys`
   - `urllib`

b) Consultar sobre las URLs para conocer los siguientes métodos e información en módulo YANG de tipo IETF sobre:
   - Información sobre las interfaces.
   - Creación de una interfaz loopback.
   - Creación de una ruta estática.

5 EQUIPOS Y MATERIALES

5.1 Hardware

- Computadora.

5.2 Software

- VirtualBox.
- VMware Workstation Pro.
- GNS3 y GNS3 VM.

6 PROCEDIMIENTO

6.1 Introducción a Python

a) Abra GNS3 y cree la topología de red que se muestra en la Figura 1.

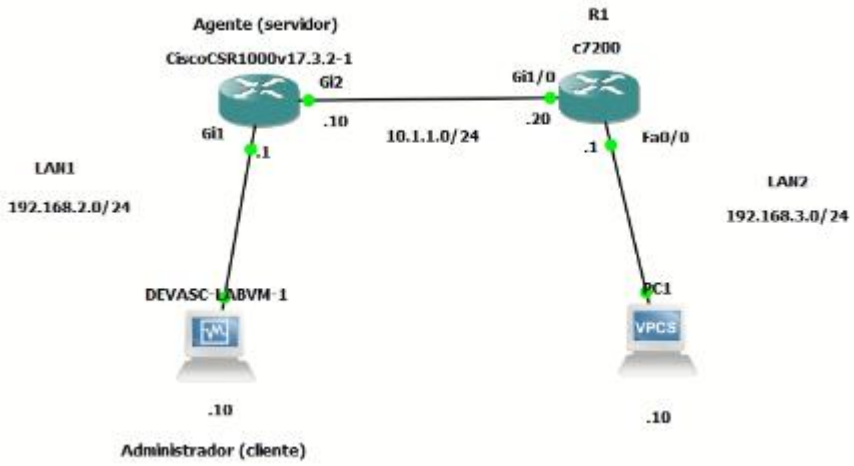

Figura 1. Topología de red a implementar

b) Iniciar el router cisco CSR1000v y configurar la interfaz GigabitEthernet1 con la dirección IP dada en la Figura 1.

c) Iniciar el router cisco c7200 y configurar la interfaz GigabitEthernet1/0 y FastEthernet0/0 con las direcciones IP dadas en la Figura 1.

d) Verificar la conectividad de los hosts con sus respectivos routers.

e) En el agente, crear un usuario `admin:admin` con privilegio 15 (privileged EXEC).

f) Configure el nombre y el dominio del enrutador Cisco CSR1000v.

g) Habilitar el servicio RESTCONF.

h) Realizar la siguiente petición desde el administrador y verificar el funcionamiento de RESTCONF.

```
curl -k https://192.168.2.1:443/restconf/ -u admin:admin
```

i) En el administrador, ejecutar el programa Visual Studio Code.

j) En la carpeta restconf-labs, crear el código `info_basica.py`

```python
# Programa que establece los parametros iniciales
# Autor: Alex Remache
# Fecha: Julio 14, 2022
# Version: 1.0

# Librerias
import json # JSON
import requests # Permite realizar APIs HTTP
from requests.auth import  HTTPBasicAuth # Autenticacion
import urllib3 # Certificados SSL
urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning) # Deshabilita

# Variables
user = "admin" # usuario
pwd = "admin" # contrasenia del usuario
formato = "application/yang-data+json" # formato JSON
server = "192.168.2.1" # IP servidor RESTCONF
port = 443 # Puerto HTTPS
rest_path = "restconf" # URL del modulo YANG

# Autenticacion
ingreso = HTTPBasicAuth(user, pwd)

# Encabezado HTTP-GET
encabezado = { "Accept": formato,
               "Content-Type": formato } # Encabezado
```

Código 1. Script con la información básica

k) Ejecutar el script anterior, en el terminal del administrador.

```
devasc@labvm:~$ cd labs/devnet-src/restconf_labs/
devasc@labvm:~/labs/devnet-src/restconf_labs$ python3
info_basica.py
```

l) Crear la función metodo_get.py.

```python
# Programa que establece la funcion HTTPS GET
# Autor: Alex Remache
# Fecha: Julio 14, 2022
# Version: 1.0

# Librerias
import sys # acceso a variables
sys.path.append("/home/devasc/labs/devnet-src/restconf_labs") # directorio actual
import info_basica as i # info_basica.py
import requests # Permite realizar APIs HTTP
import json # JSON

# Funcion
def metodo_get(url):
    # peticion
    resp = requests.get(url, auth=i.ingreso, headers=i.encabezado, verify=False)
    # Respuesta
    if resp.status_code in [200, 201, 202, 204]:
        print(f"Exitoso. Metodo GET. Status code: {resp.status_code}")
        response_json = resp.json()
        print(json.dumps(response_json, indent=4))
    else:
        print(f"Error al recuperar datos. Status code: {resp.status_code}")
```

Código 2. Método GET en Python

m) Crear la función metodo_put.py.

```python
# Programa que establece la funcion HTTPS PUT
# Autor: Alex Remache
# Fecha: Julio 14, 2022
# Version: 1.0

# Librerias
import sys # acceso a variables
sys.path.append("/home/devasc/labs/devnet-src/restconf_labs") # directorio actual
import info_basica as i # info_basica.py
import requests # Permite realizar APIs HTTP
import json # JSON

# Funcion
def metodo_put(url,yangConfig):
    # peticion
    resp = requests.put(url, auth=i.ingreso, headers=i.encabezado, data=yangConfig, verify=False)
    # Respuesta
    if resp.status_code in [200, 201, 202, 204]:
        print(f"Exitoso. Metodo PUT. Status code: {resp.status_code}")
    else:
        print(f"Error al recuperar datos. Status code: {resp.status_code}")
```

Código 3. Método PUT en Python

n) Crear la función metodo_patch.py.

```python
# Programa que establece la funcion HTTPS PATCH
# Autor: Alex Remache
# Fecha: Julio 14, 2022
# Version: 1.0

# Librerias
import sys # acceso a variables
sys.path.append("/home/devasc/labs/devnet-src/restconf_labs") # directorio actual
import info_basica as i # info_basica.py
import requests # Permite realizar APIs HTTP
import json # JSON

# Funcion
def metodo_patch(url,yangConfig):
    # peticion
    resp = requests.patch(url, auth=i.ingreso, headers=i.encabezado, data=yangConfig, verify=False)
    # Respuesta
    if resp.status_code in [200, 201, 202, 204]:
        print(f"Exitoso. Metodo PATCH. Status code: {resp.status_code}")
    else:
        print(f"Error al recuperar datos. Status code: {resp.status_code}")
```

Código 4. Método PATCH en Python

o) Crear la función metodo_delete.py.

```python
# Programa que establece la funcion HTTPS DELETE
# Autor: Alex Remache
# Fecha: Julio 14, 2022
# Version: 1.0

# Librerias
import sys # acceso a variables
sys.path.append("/home/devasc/labs/devnet-src/restconf_labs") # directorio actual
import info_basica as i # info_basica.py
import requests # Permite realizar APIs HTTP
import json # JSON

# Funcion
def metodo_delete(url):
    # peticion
    resp = requests.delete(url, auth=i.ingreso, headers=i.encabezado, verify=False)
    # Respuesta
    if resp.status_code in [200, 201, 202, 204]:
        print(f"Exitoso. Metodo DELETE. Status code: {resp.status_code}")
    else:
        print(f"Error al recuperar datos. Status code: {resp.status_code}")
```

Código 5. Método DELETE en Python

p) Crear el código `get_info_restconf.py` y ejecutarlo.

```python
# Programa que recolecta la informacion RESTCONF del servidor
# Autor: Alex Remache
# Fecha: Julio 14, 2022
# Version: 1.0
# Metodo HTTPS: GET

# librerias
import sys # acceso a variables
sys.path.append("/home/devasc/labs/devnet-src/restconf_labs") # directorio actual
import info_basica as i # info_basica.py
import metodo_get as get # metodo_get.py

# URL
url = f"https://{i.server}:{i.port}/{i.rest_path}"
print(url)

get.metodo_get(url)
```

Código 6. Script get_info_restconf.py

Pregunta 1. ¿Qué hace el script get_info_restconf.py?
Pregunta 2. Mostrar el resultado de su ejecución y comentarlo.

6.2 Configuración de interfaces con RESTCONF mediante Python.

a) Crear el código `get_info.g2.py`, que muestra la información de la interfaz GigabitEthernet2 del agente.

```python
# Programa que recolecta la informacion de la interfaz
# GigabitEthernet 2 del agente
# Autor: Alex Remache
# Fecha: Julio 14, 2022
# Version: 1.0
# Metodo HTTPS: GET

# Librerias
import sys # acceso a variables
sys.path.append("/home/devasc/labs/devnet-src/restconf_labs") # directorio actual
import info_basica as i # info_basica.py
import metodo_get as get # metodo_get.py

# URL
url = f"https://{i.server}:{i.port}/{i.rest_path}/data/ietf-interfaces:interfaces/interface=GigabitEthernet2"
print(url)

get.metodo_get(url)
```

Código 7. Script get_info_g2.py

Pregunta 3. ¿Qué resultado muestra el código get_info_g2.py?, ¿la interfaz Gi2, que información tiene?

b) Crear el código `patch_interface_g2.py` que actualice la configuración de la interfaz, conforme a la topología de la Figura 1.

```python
import sys # acceso a variables
sys.path.append("/home/devasc/labs/devnet-src/restconf_labs") # directorio actual
import info_basica as i # info_basica.py
import metodo_get as get # metodo_get.py
import metodo_patch as patch # metodo_patch.py

# URL
url = f"https://{i.server}:{i.port}/{i.rest_path}/data/ietf-interfaces:interfaces/interface=GigabitEthernet2"
print(url)

# Payload
yangConfig = """
{
    "ietf-interfaces:interface": {
        "enabled": "true",
        "description": "To R1",
        "ietf-ip:ipv4": {
            "address": [
                {
                    "ip": "10.1.1.10",
                    "netmask": "255.255.255.0"
                }
            ]
        }
    }
}
"""

patch.metodo_patch(url,yangConfig)
get.metodo_get(url)
```

Código 8. Configuración de dirección IP y descripción de Gi2 con Python

Pregunta 4. ¿Qué resultado muestra este código?
Pregunta 5. En el agente, correr el comando para ver el estado de las interfaces ¿cuál es el resultado?

c) En base a la práctica 4 y los códigos y funciones ya creadas, desarrollar un script `get_info_lb1.py` que muestra la información de la interfaz LoopBack1 del agente.

Pregunta 6. Mostrar el código. ¿Qué resultado muestra la ejecución de este código?

d) Crear el código `put_crear_lb1.py` que cree la interfaz Loopback1, y

```python
import sys # acceso a variables
sys.path.append("/home/devasc/labs/devnet-src/restconf_labs") # directorio actual
import info_basica as i # info_basica.py
import metodo_get as get # metodo_get.py
import metodo_put as put # metodo_put.py

# URL
url = f"https://{i.server}:{i.port}/{i.rest_path}/data/ietf-interfaces:interfaces/interface=Loopback1"
print(url)

# Payload
yangConfig = """
{
    "ietf-interfaces:interface": {
        "name": "Loopback1",
        "description": "Loopback 1",
        "type": "iana-if-type:softwareLoopback",
        "enabled": true,
        "ietf-ip:ipv4": {
            "address": [
                {
                    "ip": "20.1.1.1",
                    "netmask": "255.255.255.0"
                }
                ]
            },
        "ietf-ip:ipv6": {}
    }
}
"""

put.metodo_put(url,yangConfig)
get.metodo_get(url)
```

Código 9. Creación de la interfaz Loopback1 con Python

Pregunta 7. ¿Qué resultado muestra la ejecución del Código 9?
Pregunta 8. En el agente, correr el comando para ver el estado de las interfaces ¿cuál es el resultado?

e) Crear el código `delete_int_lb1.py`.

Pregunta 9. Mostrar el código. ¿Qué resultado muestra la ejecución de este código?

6.3 Creación de ruta estática con RESTCONF mediante Python.

a) En el router R1, crear la ruta estática para que la LAN2 tenga conectividad con la LAN1.

```
R1# configure terminal
R1(config)# ip route 192.168.2.0 255.255.255.0 10.1.1.10
```

b) En el administrador, crear el código `get_info_routing.py` que obtendrá tabla de enrutamiento del agente.

```python
# Programa que recolecta la informacion routing del servidor
# Autor: Alex Remache
# Fecha: Julio 14, 2022
# Version: 1.0
# Metodo HTTPS: GET

# librerias
import sys # acceso a variables
sys.path.append("/home/devasc/labs/devnet-src/restconf_labs") # directorio actual
import info_basica as i # info_basica.py
import metodo_get as get # metodo_get.py

# URL
url = f"https://{i.server}:{i.port}/{i.rest_path}/data/ietf-routing:routing-state"
print(url)

get.metodo_get(url)
```

Código 10. Información de enrutamiento del agente con Python

Pregunta 10. ¿Qué resultado muestra la ejecución del Código 10?,¿el resultado muestra algún tipo de enrutamiento a la LAN2?

c) Crear el código `put_crear_ruta.py` que añadirá la ruta estática para tener conectividad desde la LAN1 hacia la LAN2.

```python
import sys # acceso a variables
sys.path.append("/home/devasc/labs/devnet-src/restconf_labs") # directorio actual
import info_basica as i # info_basica.py
import metodo_get as get # metodo_get.py
import metodo_put as put # metodo_put.py

# URL
url = f"https://{i.server}:{i.port}/{i.rest_path}/data/ietf-routing:routing/routing-instance=default/routing-protocols/routing-protocol=static,1"
print(url)

# Payload
yangConfig = """
{
    "ietf-routing:routing-protocol": {
        "type": "ietf-routing:static",
        "name": "1",
        "static-routes": {
            "ietf-ipv4-unicast-routing:ipv4": {
                "route": [
                    {
                        "destination-prefix": "192.168.3.0/24",
                        "next-hop": {
                            "next-hop-address": "10.1.1.20"
                        }
                    }
                ]
            }
        }
    }
}
"""

put.metodo_put(url,yangConfig)
get.metodo_get(url)
```

Código 11. Configuración de ruta estática hacia la LAN2 con Python

Pregunta 11. ¿Qué resultado muestra el Código 11?
Pregunta 12. Ejecutar el Código10., ¿cuál es el nuevo resultado?
Pregunta 13. En el agente, correr el comando `show ip route`, ¿cuál es el resultado?
Pregunta 14. Realizar un ping desde el administrador a la PC1, mostrar el resultado.

6.4 Análisis de la red con RESTCONF mediante Python.

a) Crear el código `get_statitics_restconf.py`

Pregunta 15. Mostrar el código y realizar un análisis de la información de este código.

b) Crear el código `get_statitics_interfaces.py`

Pregunta 16. Mostrar el código y realizar un análisis de la información de este código.

.

Nota: Los códigos de esta práctica se encuentran en el repositorio GitHub: https://github.com/alexrch97/Analisis-RESTCONF

7 INFORME

7.1 Proporcione la configuración construida en laboratorio con una explicación adecuada de los resultados que se muestran.
7.2 Responder las preguntas dadas en el procedimiento.
7.3 Conclusiones y Recomendaciones
7.4 Bibliografía

8 REFERENCIAS

[1] "Python: qué es, para qué sirve y cómo se programa." Internet: https://www.cursosaula21.com/que-es-python/, 2022 [Jul. 12, 2022].
[2] M. Neidinger. *Python Network Programming Techniques: 50 real-world recipes to automate infrastructure networks and overcome networking challenges with Python.* Birmingham, UK: Packt Publishing, 2021.

ANEXO V

RESOLUCIÓN DE TRABAJOS PREPARATORIOS

PRÁCTICA N° 1

1 TEMA

INTRODUCCIÓN A RESTCONF

2 OBJETIVOS

2.1 Configurar un router y un terminal.
2.2 Configurar el router para habilitar el protocolo RESTCONF.
2.3 Generar una petición/respuesta cliente-servidor mediante RESTCONF.

3 TRABAJO PREPARATORIO

3.1 Describir las características del router Cisco Cloud Services Router CSR1000V.

El enrutador de servicios en la nube de Cisco 1000v (CSR 1000v) es un router de forma virtual que ofrece funciones integrales de servicios de red y puerta de enlace WAN en entornos virtuales y de nube [1].

Tabla 1. Características CSR1000V

Característica	Descripción
Versión del Software	Cisco IOS XE Software en formatos ISO, BIN, OVA y QCOW2.
Hipervisores soportados	● VMware ESXi 6.5 ● Citrix XenServer 6.5 ● Red Hat KVM (Red Hat Enterprise Linux 7.5) ● KVM (Ubuntu 14.04 LTS y Suse Linux 12-SP3) ● Microsoft Hyper-V for Windows Server 2016
Nubes públicas soportadas	Amazon Web Services, Microsoft Azure y Google Cloud Platform.
Enrutamiento	BGP, OSPF, EIGRP, Policy-Based Routing (PBR), IPv6, VRF-Lite, Multicast, LISP, GRE y Connectionless Network Services (CLNS).
MPLS	MPLS VPN, VRF y Bidirectional Forwarding Detection (BFD).
Direccionamiento	DHCP, DNS, NAT, 802.1Q VLAN, Ethernet Virtual Connection (EVC) y VXLAN.
Alta disponibilidad	HSRP, Virtual Router Redundancy Protocol (VRRP), Gateway Load Balancing Protocol (GLBP).
Software de seguridad	● VPN: IPsec VPN, DMVPN, Easy VPN, FlexVPN y GetVPN ● Firewall: ZBFW ● Access control: ACL, AAA, RADIUS y TACACS+
Gestión y administración	● Aprovisionamiento y administración: Cisco IOS XE CLI, SSH, Telnet, Cisco Prime Infrastructure, Cisco Prime Network Services Controller y OpenStack Configdrive.

	● Monitoreo y troubleshooting: SNMP, Syslog, NetFlow, IP SLA y Embedded Event Manager (EEM). ● RESTful Application Programming Interfaces (APIs). ● NETCONF y RESTCONF.

3.2 Consultar y describir los comandos necesarios para habilitar RESTCONF en el router Cisco CSR1000V.

El protocolo RESTCONF utiliza los servicios HTTP, por lo que, forma parte de los comandos necesarios para habilitar RESTCONF en el router [2]. A continuación, los principales comandos:

Tabla 2. Comandos para RESTCONF

Paso	Comando	Ejemplo	Descripción
1	enable	`Device> enable`	Habilita el modo privilegiado EXEC.
2	configure terminal	`Device# configure terminal`	Entra el modo global de configuración.
3	restconf	`Device(config)# restconf`	Habilita la interfaz RESTCONF en el equipo de red.
4	ip http secure-server	`Device(config)# ip http secure-server`	Habilita el servidor HTTP seguro (HTTPS).
5	end	`Device(config)# end`	Sale del modo global de configuración y entra al modo privilegiado EXEC.

3.3 Consultar mínimo 3 dispositivos que soporten el protocolo RESTCONF.

Según [3], Cisco tiene los siguientes equipos se soportan RESTCONF.

Tabla 3. Equipos Cisco

IOS	Equipos	Característica
XE 16.6	ISR 4000 Series	Integrated Services Routers
	CSR 1000v	Cloud Services Router
	ASR 1000 Series	Aggregation Services Routers
	ISRv	Integrated Services Virtual Router
XE 16.7	ASR 900 Series	Aggregation Services Routers
	NCS 4200 Series	Network Convergence System
XE 16.8	Catalyst 3650, 3850, 9000	Switches
	ISR 1000 Series	Integrated Services Routers
XE 16.10	IR 1101 Series	Integrated Services Router Rugged
XE 16.11	IE 3x00 Series	Catalyst Rugged Series switches

4 REFERENCIAS

[1] Cisco. “Cisco Cloud Services Router 1000v Data Sheet.” Internet: https://www.cisco.com/c/en/us/products/collateral/routers/cloud-services-router-1000v-series/data_sheet-c78-733443.html, Feb. 28, 2020 [Jun. 13, 2022].

[2] Cisco. “Programmability Configuration Guide, Cisco IOS XE Fuji 16.7.x.” Internet: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/prog/configuration/167/b_167_programmability_cg/restconf_programmable_interface.html, Feb. 5, 2021 [Jun. 13, 2022].

[3] Cisco DevNet. “Model Driven Programmability.” Internet: https://developer.cisco.com/site/standard-network-devices/ [May. 12, 2022].

PRÁCTICA N° 2

1 TEMA

LENGUAJE DE MODELADO DE DATOS YANG

2 OBJETIVOS

2.1 Analizar el modelo de datos YANG.

2.2 Identificar el modelo de datos YANG con el comando `pyang`.

2.3 Comparar entre los tipos de datos estructurados XML y JSON.

3 TRABAJO PREPARATORIO

3.1 Describir los modelos de datos YANG de IETF, OPENCONFIG y nativos.

Hay varias fuentes de muestra YANG disponibles, de las cuales las tres siguientes se consideran principales [1]:

- Modelos nativos/Específicos de Cisco.
- OpenConfig
- IETF

Modelos nativos: También se conocen como modelo específico y son publicados por varios proveedores de dispositivos, incluido Cisco. Por ejemplo, Cisco-IOS-XR-ptp-oper.yang

Modelos de OpenConfig: OpenConfig es un grupo de trabajo informal de operadores de red. OpenConfig define los modelos YANG comunes que todos los proveedores deben admitir para configurar las funciones críticas. p.ej. openconfig-interfaces.yang

Modelos IETF: IETF también define pocos módulos YANG comunes que describan la configuración básica para interfaces, QoS y definan otros tipos de datos comunes (como IPv4, IPv6, etc.). por ejemplo, ietf-syslog-types.yang

3.2 Consultar acerca de qué es un `namespace` y qué relación tiene con los modelos de datos.

Los modelos de datos en YANG se definen en módulos. El módulo incorpora información sobre la versión de YANG utilizada, los datos del creador, información de la versión actual del módulo, descripción del módulo y el `namespace` que será

el identificador del módulo dentro de una comunicación XML [2]. Esta `namespace` será globalmente única, definida, según el tipo de modelo, por su organización permitente [3].

3.3 Describir los nodos del modelo de datos YANG.

Según el RFC 6020 [3], YANG define cuatro tipos de nodos para el modelado de datos: Leaf Node, Leaf-List Node, Container Node y List Node.

Leaf Node contiene datos simples como un número entero o una cadena. Tiene exactamente un valor de un tipo particular y no tiene nodos secundarios.

Leaf-List Node es equivalente a Leaf Node pero elimina la restricción de ocurrencia en su nivel en XML.

Container Node se utiliza para agrupar nodos relacionados. Un contenedor solo tiene nodos secundarios y ningún valor. Un contenedor puede contener cualquier número de nodos secundarios de cualquier tipo.

List Node permite definir una estructura de nodos y permite la existencia de *n* List Node similares.

3.4 Consultar qué es `pyang` y para qué es empleado.

Los módulos YANG pueden volverse complejos rápidamente pues existen identidades, definiciones de tipos y agrupaciones haciendo que sea más difícil comprender la ruta exacta a un elemento de datos que desee extraer o configurar en un dispositivo. Afortunadamente, existen herramientas para ayudar con esto. Una de esas herramientas, es `pyang`. Pyang es una aplicación de código abierto basada en Python y un conjunto de bibliotecas que facilitan mucho el trabajo con módulos YANG al permitirle transformar módulos en diferentes formatos. Además, `pyang` proporciona formas de validar la sintaxis del módulo y verificar las incompatibilidades anteriores entre dos módulos [4].

3.5 Consultar sobre los formatos XML y JSON, diferencias y ejemplos.

XML: El lenguaje de marcado extensible XML (*Extensible Markup Language*) es un lenguaje de marcado que define un conjunto de reglas para describir la estructura y los tipos de datos en un formato que es a la vez humano y legibles por máquina. XML está en un formato basado en texto. Es independiente del idioma y de la plataforma. Los elementos y atributos son los componentes básicos de XML [5].

```
<?xml version="1.0"?>
<rpc message-id="17">
  <what>ping</what>
  <who>10.20.30.40</who>
  <quick/>
</rpc>
```

Código 1. Ejemplo de un documento XML que describe un RPC [4]

JSON: Notación de objetos de JavaScript JSON (*JavaScript Object Notation*). Es un estándar abierto, liviano y autodescriptivo para el intercambio de datos. El formato de datos JSON es legible tanto por humanos como por máquinas. JSON es un subconjunto de JavaScript. Admite los tipos de datos primitivos fundamentales, como cadena, número, objeto, matriz, booleano y nulo. Las otras dos estructuras de datos importantes de JSON son: una colección de pares de nombre/valor y una lista ordenada de valores. Algunas características importantes de JSON incluyen simplicidad, apertura, autodescripción, internacionalización, extensibilidad e interoperabilidad [5].

```
var objeto1 = {
  nombre: "Computadora"
  color: "Gris"
  modelo: "Laptop"
}
```

Código 2. Ejemplo de un documento JSON que describe un objeto [6]

La comparación entre estos lenguajes se muestra en la Tabla 1 [5].

Tabla 1. Comparación entre XML y JSON

Diferencias	XML	JSON
Naturaleza del lenguaje de datos	Es un lenguaje marcado.	No es un lenguaje marcado.
Esquema	Tiene.	No tiene.
Seguridad	Es más seguro	Menos seguro
Tipos de datos	Provee ricos tipos de datos. No soporta un array.	Soporta solo limitados tipos de datos. Soporta array.
Modelo del objeto	No se alinea con modelos de objetos de lenguajes programables.	Se alinea con modelos de objetos de lenguajes programables, haciendo más fácil el trabajo a los desarrolladores.
Similitudes	Son estándares abiertos. Soportan Unicode e internacionalización. Son interoperables. Son lenguajes neutrales.	

4 REFERENCIAS

[1] Cisco. “Informe técnico sobre telemetría basada en el modelo ASR9K.” Internet: https://www.cisco.com/c/es_mx/support/docs/routers/asr-9000-series-aggregation-services-routers/215321-asr9k-model-driven-telemetry-whitepaper.html, Mar. 4, 2020 [Jun. 17, 2022].

[2] M. Merelo. “Estudio de la gestión de configuración a través de NETCONF.” Tesis, Universidad de Sevilla, Sevilla, España, 2017.

[3] M. Bjorklund, Ed. and Tail-f Systems. “YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF).” IETF RFC 6020, Oct. 2010.
[4] B. Claise, J. Clarke, and J. Lindblad. *Network programmability with YANG: the structure of network automation with YANG, NETCONF, RESTCONF, and gNMI.* Boston, MA: Pearson Education, 2019.
[5] C. Surianarayanan, G. Ganapathy, and R. Pethuru. *Essentials of Microservices Architecture: Paradigms, Applications, and Techniques.* Florida, USA: CRC Press ,2019.
[6] Huawei. “RESTCONF Configuration - CloudEngine 16800 V200R019C10 Configuration Guide 07.” Internet: https://support.huawei.com/enterprise/en/doc/EDOC1100137878/afa66a1e/restconf-configuration, Jun. 11, 2021 [Jun. 17, 2022].

PRÁCTICA N° 3

1 TEMA

SEGURIDAD Y MÉTODOS HTTP

2 OBJETIVOS

2.1 Implementar ACLs para seguridad.
2.2 Analizar la herramienta cURL para el protocolo RESTCONF.
2.3 Identificar los diferentes métodos RESTCONF.

3 TRABAJO PREPARATORIO

3.1 Consultar el comando en un router Cisco CSR1000V para habilitar ACLs en RESTCONF.

Según [1], para habilitar ACL para RESTCONF se debe seguir el siguiente proceso:

Tabla 1. Comandos para ACL en RESTCONF

Paso	Comando	Ejemplo	Descripción
1	enable	`Device> enable`	Habilita el modo privilegiado EXEC.
2	configure terminal	`Device# configure terminal`	Entra el modo global de configuración.

3	ip access-list {standard | extended} *access-list-name*
	
	ipv6 access-list *access-list-name*
		`Device(config)# ip access-list standard ipv4_acl1_permit`
		
		`Device(config)# ipv6 access-list ipv6-acl1_permit`
			Especifica una lista de acceso de IP estándar e ingresa al modo de configuración de lista de acceso estándar.
			
			Especifica una lista de acceso IPv6 e ingresa al modo de configuración de la lista de acceso IPv6

4	permit {*host-address* | *host-name* | any} [*wildcard*]	`Device(config-std-nacl)# permit 192.168.255.0 0.0.0.255`	Establece condiciones en una lista de acceso IP/IPv6 que permitirá paquetes.
5	deny {*host-address* | *host-name* | any} [*wildcard*]	`Device(config-std-nacl)# deny any`	Establece condiciones en una lista de acceso IP/IPv6 que denegará paquetes.
6	exit	`Device(config-std-nacl)# exit`	Sale del modo de configuración de lista de acceso estándar y vuelve al modo de configuración global.
7	restconf {ipv4 | ipv6 }access-list name *access-list-name*	`Device(config)# restconf ipv4 access-list name ipv4-acl1_permit`	Configura una ACL para la sesión RESTCONF.
8	end	`Device(config)# end`	Sale del modo global de configuración y entra al modo privilegiado EXEC.

3.2 Consultar el comando curl y describir las opciones que tiene.

El comando `curl` tiene la siguiente sintaxis:

`curl [opciones/URLs]:`

Donde [2]:

`curl [opciones/URLs]`: curl es una herramienta para transferir datos desde o hacia un servidor. Admite entre otros protocolos, estos: FTP, FTPS, HTTP, HTTPS, entre otros. Este comando está diseñado para funcionar sin la interacción del usuario. Además, curl proporciona una variedad de trucos útiles, como compatibilidad con proxy, autenticación de usuarios, cargas FTP, publicaciones HTTP, conexiones SSL, cookies, reanudación de transferencias de archivos y más.

`URLs`: La sintaxis de URL depende del protocolo. Se pueden especificar varias URL o partes de URL.

`-k (--insecure)`: De manera predeterminada, se verifica que cada conexión segura que curl realiza sea segura antes de que se realice la transferencia. Esta opción hace que curl se salte el paso de verificación y continúe sin verificar.

`-u (--user <usuario:contraseña>)`: Especifica el nombre de usuario y la contraseña que se usarán para la autenticación del servidor.

`-H, --header <encabezado>:` (HTTP) Encabezado adicional para incluir en la solicitud HTTP al enviar a un servidor. Puede especificar cualquier número de encabezados adicionales.

`-X, --request <method>:` (HTTP) Especifica un método de solicitud personalizado para usar al comunicarse con el servidor HTTP. Se utilizará el método de solicitud especificado en lugar del método utilizado de otro modo (que por defecto es GET). Las solicitudes HTTP adicionales comunes incluyen PUT y DELETE.

`-d, --data <data>:` (HTTP MQTT) Envía los datos especificados en una solicitud POST al servidor HTTP, de la misma manera que lo hace un navegador cuando un usuario ha completado un formulario HTML y presiona el botón Enviar.

3.3 Consultar los campos de encabezado de una solicitud HTTP.

Los campos de encabezado (*Request header*) de una solicitud HTTP se utilizan para especificar la metainformación que queremos dar a nuestro servidor al procesar la solicitud [3]. Los tipos más comunes de información enviada dentro de un encabezado se describe en la Tabla 2.

Tabla 2. Información en el campo de encabezado

Elemento	Información
Content-Type:	Tipo de datos (JSON o XML) que se envían.
Accept:	Tipo de datos (JSON o XML) que se desea recibir.
Authorization:	Proporciona credenciales de autenticación, como nombre de usuario/contraseña o tokens API.
Content-Length:	Tamaño del cuerpo de nuestra solicitud en octetos.
Host:	Especifica el host (y, opcionalmente, el puerto) en el que escucha el servidor web.

3.4 Consultar cómo se crea una URL a partir de un módulo YANG de tipo IETF.

El protocolo RESTCONF al trabajar sobre HTTPS, necesita que sus solicitudes HTTP sean en formato URL, y al conocer los módulos YANG que trabaja en el router, se pueden crear las URL de la siguiente manera [4]:

`https://<dirección_IP>/<root>/<almacén_de_datos>/<[módulo_YANG:]contenedor>/<nodo_leaf>[?<>opciones]`

Donde:
`dirección_IP:` dirección IP del agente RESTCONF.
`root`: principal punto de entrada para las solicitudes RESTCONF.
`almacén_de_datos`: almacén de datos que se requiere.
`[módulo_YANG:]contenedor:` contenedor del modelo base que se está utilizando. Proporcionar el nombre del módulo es opcional.
`nodo_leaf:` elemento individual (nodo leaf) del contenedor.

`[?<>opciones]:` parámetros opcionales que tienen impacto en la respuesta final.

4 REFERENCIAS

[1] Cisco. “Programmability Configuration Guide, Cisco IOS XE Gibraltar 16.11.x.” Internet: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/prog/configuration/1611/b_1611_programmability_cg/service_level_ACLs_NETCONF_RESTCONF.html, Jun. 1, 2021 [Jun. 27, 2022].

[2] Stenberg. “curl.1 the man page.” Internet: https://curl.se/docs/manpage.html, [Jun. 27, 2022].

[3] M. Neidinger. *Python Network Programming Techniques: 50 real-world recipes to automate infrastructure networks and overcome networking challenges with Python.* Birmingham, UK: Packt Publishing, 2021.

[4] C. Valle. “How to build my own URL for RESTCONF.” Internet: https://community.cisco.com/t5/automation-and-analytics/how-to-build-my-own-url-for-restconf/td-p/4106157, Jun. 19, 2020 [Jun. 27, 2022].

PRÁCTICA N° 4

1 TEMA

GESTIÓN RESTCONF CON POSTMAN

2 OBJETIVOS

2.1 Analizar la herramienta Postman para el protocolo RESTCONF.
2.2 Configurar interfaces y ruta estática mediante RESTCONF.
2.3 Diagnosticar la red mediante RESTCONF.

3 TRABAJO PREPARATORIO

3.1 Identificar las principales áreas de Postman.

Como se puede observar en la Figura 1, el espacio de trabajo de Postman es grande y contiene los siguientes elementos [1]:

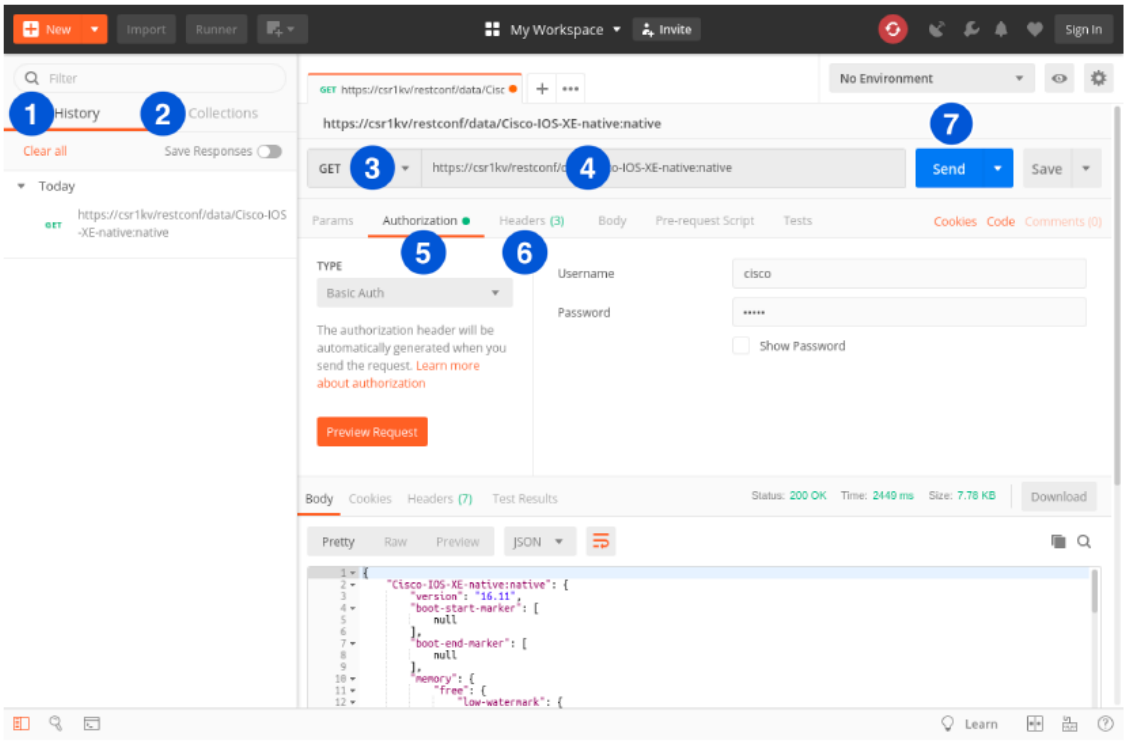

Figura 1. Espacio de trabajo Postman [1]

1. **History:** cada petición a la API que realiza se guarda en Historial.
2. **Collections:** puede crear una colección de elementos de su historial. Esto hace posible crear flujos de trabajo sólidos y guardarlos como una colección antes de comenzar el desarrollo o la codificación.
3. **Método HTTP:** desplegable para que seleccione el método requerido para la API: GET, POST, PATCH, PUT, DELETE, etc.
4. **URL:** aquí es donde ingresa la URL de su punto final de API.
5. **Authorization:** las credenciales de nivel 15 se autenticarán en este espacio.
6. **Headers:** en esta sección se configurará los encabezados de las peticiones HTTP.
7. **Send:** Finalmente, el botón Send emite la llamada API al dispositivo (punto final API)

3.2 Consultar las estructuras de los siguientes módulos YANG:

Los módulos YANG tienen las siguientes estructuras [3]:

- **ietf-restconf-monitoring.yang**
  Como se observa en la Figura 2, la estructura de este módulo, se tiene un contenedor llamado `restconf-state`, que contiene información de monitoreo del protocolo RESTCONF.

```
devasc@labvm:~/labs/devnet-src/pyang$ pyang -f tree ietf-restconf-monitoring.yang
ietf-restconf-monitoring.yang:5: error: module "ietf-yang-types" not found in search path
module: ietf-restconf-monitoring
  +--ro restconf-state
     +--ro capabilities
     |  +--ro capability*   inet:uri
     +--ro streams
        +--ro stream* [name]
           +--ro name                        string
           +--ro description?                string
           +--ro replay-support?             boolean
           +--ro replay-log-creation-time?   yang:date-and-time
           +--ro access* [encoding]
              +--ro encoding    string
              +--ro location    inet:uri
```

Figura 2. Estructura del módulo ietf-restconf-monitoring.yang

- **ietf-interfaces.yang**
  En la Figura 3 se puede observar que la estructura del módulo tiene dos contenedores, el primer contenedor `interfaces` contiene los parámetros de configuración de las interfaces; el segundo `interfaces-state` contiene los nodos de datos para el estado operativo de las interfaces.

```
devasc@labvm:~/labs/devnet-src/pyang$ pyang -f tree ietf-interfaces.yang
ietf-interfaces.yang:6: error: module "ietf-yang-types" not found in search path
module: ietf-interfaces
  +--rw interfaces
  |  +--rw interface* [name]
  |     +--rw name                        string
  |     +--rw description?                string
  |     +--rw type                        identityref
  |     +--rw enabled?                    boolean
  |     +--rw link-up-down-trap-enable?   enumeration {if-mib}?
  +--ro interfaces-state
     +--ro interface* [name]
        +--ro name               string
        +--ro type               identityref
        +--ro admin-status       enumeration {if-mib}?
        +--ro oper-status        enumeration
        +--ro last-change?       yang:date-and-time
        +--ro if-index           int32 {if-mib}?
        +--ro phys-address?      yang:phys-address
        +--ro higher-layer-if*   interface-state-ref
        +--ro lower-layer-if*    interface-state-ref
        +--ro speed?             yang:gauge64
        +--ro statistics
           +--ro discontinuity-time    yang:date-and-time
           +--ro in-octets?            yang:counter64
           +--ro in-unicast-pkts?      yang:counter64
           +--ro in-broadcast-pkts?    yang:counter64
           +--ro in-multicast-pkts?    yang:counter64
           +--ro in-discards?          yang:counter32
           +--ro in-errors?            yang:counter32
           +--ro in-unknown-protos?    yang:counter32
           +--ro out-octets?           yang:counter64
           +--ro out-unicast-pkts?     yang:counter64
           +--ro out-broadcast-pkts?   yang:counter64
           +--ro out-multicast-pkts?   yang:counter64
           +--ro out-discards?         yang:counter32
           +--ro out-errors?           yang:counter32
```

Figura 3. Estructura del módulo ietf-interfaces.yang

- **ietf-routing.yang**
  La Figura 4. muestra el contenido del módulo, se tiene dos contenedores, el primero `routing-state` muestra los datos de estado del subsistema de enrutamiento, el segundo `routing` es un contenedor para la lista de instancias de protocolo de enrutamiento.

Figura 4. Estructura del módulo ietf-routing.yang

3.3 Consultar sobre los códigos de respuesta a las peticiones HTTP.

Uno de los mensajes importante de la respuesta HTTP, es el código de estado, que nos da información acerca de las respuestas dadas. En la Tabla 1. se indica los mensajes más importantes [4].

Tabla 1. Códigos de estado más habituales

Categorías	Respuestas	Código	Estado
1xx	Informativas		
2xx	Exitosas	200-OK	Información devuelta (GET, PUT, PATCH).
		201-Created	Información creada (POST).
		204-No Content	Información eliminada (DELETE).
3xx	Redirigidas	301-Moved permanently	La solicitud debe ser redirigida.
4xx	Error del cliente	400-Bad request	El servidor no pudo comprender la solicitud (falta de datos o formato erróneo).

		401-Unauthorized	Falta de autenticación.
		403-Forbidden	Sin permisos.
		404-Not Found	El recurso solicitado no se pudo encontrar en el servidor.
		405-Method Not Allowed	La solicitud utilizó un método que no está permitido para este recurso específico.
5xx	Error del servidor	500-Internal Server error	El servidor ha encontrado algún tipo de error en él.
		502-Bad Gateway	El servidor tuvo problemas para comunicarse con otro servidor ascendente.
		503-Service Unavailable	El servidor actualmente no puede manejar la solicitud.

4 REFERENCIAS

[1] T. Roman. “Model Driven Network Automation with IOS-XE LTRCRT-2700,” Cisco Live, San Diego, CA, 2019.

[2] M. A. Álvarez. “Cómo usar Postman para probar nuestras APIs.” Internet: https://desarrolloweb.com/articulos/como-usar-postman-probar-api, Oct. 19, 2021 [Jul. 2, 2022]

[3] Apoorvashastry “Cisco-IOS-XE-17.3.1 Release Yang Models (#908).” Internet: https://github.com/YangModels/yang/tree/main/vendor/cisco/xe/1731, Dec. 3, 2020 [Jul. 2, 2022]

[4] M. Neidinger. *Python Network Programming Techniques: 50 real-world recipes to automate infrastructure networks and overcome networking challenges with Python.* Birmingham, UK: Packt Publishing, 2021.

PRÁCTICA N° 5

1 TEMA

GESTIÓN RESTCONF CON PYTHON

2 OBJETIVOS

2.1 Analizar la herramienta Python para el protocolo RESTCONF.
2.2 Configurar interfaces y ruta estática mediante RESTCONF.
2.3 Diagnosticar la red mediante RESTCONF.

3 TRABAJO PREPARATORIO

3.1 Consultar sobre las librerías:

- **requests**
  Esta es una librería de Python que facilita mucho el manejo de solicitudes HTTP. Tiene las funciones `get()`, `post()`, `patch ()`, `put()`, `delete()`, `head()` y `options()` para realizar los métodos de petición: GET, POST, PATCH, PUT, DELETE, HEAD y OPTIONS [1]..

- **sys**
  Este módulo provee acceso a algunas variables usadas o mantenidas por el intérprete y a funciones que interactúan fuertemente con el intérprete. Uno de los objetos más utilizados de la librería sys es sys.path que es una lista de cadenas de caracteres que especifica la ruta de búsqueda de módulos [2].

- **urllib3**
  Es un paquete que reúne varios módulos para trabajar con URLs:
  `urllib.request` para abrir y leer URLs
  `urllib.error` contiene las excepciones propuestas
  `urllib.parse` para parsear URLsLa

3.2 Consultar sobre las URLs para conocer los siguientes métodos e información en módulo YANG de tipo IETF sobre:

- **Información sobre las interfaces**
  GET https://192.168.2.1/restconf/data/ietf-interfaces:interfaces/

- **Creación de una interfaz loopback**
  PUT https://192.168.2.1/restconf/data/ietf-interfaces:interfaces/interface=Loopback1

- **Creación de una ruta estática**
  PUT https://192.168.2.1:443/restconf/data/ietf-routing:routing/routing-instance=default/routing-protocols/routing-protocol=static,1

4 REFERENCIAS

[1] J. J. Lozano. “Python requests. La librería para hacer peticiones http en Python.” Internet: https://j2logo.com/python/python-requests-peticiones-http/, [Jul 13, 2022]

[2] Python. “sys — Parámetros y funciones específicos del sistema.” Internet: https://docs.python.org/es/3/library/sys.html, [Jul 13, 2022]

[3] Python. “urllib — URL módulos de manipulación.” Internet: https://docs.python.org/es/3/library/urllib.html, [Jul 13, 2022]

ANEXO VI

ENLACE DE REACTIVOS

https://epnecuador-my.sharepoint.com/:f:/g/personal/alex_remache_epn_edu_ec/EqWGszuMDfpMts0nbDE1NKABusi-eAFTAV4gWOWvWo83gQ?e=JuTdTg

https://github.com/alexrch97/Analisis-RESTCONF/blob/main/preguntas-232_ITID843_GR1_2022-1-RESTCONF%20(TIC)-20220903-1943.xml

ANEXO VII

ENLACE DE VIDEOS

https://epnecuador-my.sharepoint.com/:f:/g/personal/alex_remache_epn_edu_ec/ElroTtK3mFBMpSfXJcR9tTQBHE-_DodTOUrX5JOm0ucp2A?e=LyH9sw

https://www.youtube.com/channel/UCxDoDHlw9I_6teNU8E4Hapg

ANEXO VIII

RESOLUCIÓN DE INFORMES

PRÁCTICA N° 1

1 TEMA

INTRODUCCIÓN A RESTCONF

2 OBJETIVOS

2.1 Configurar un router y un terminal.
2.2 Configurar el router para habilitar el protocolo RESTCONF.
2.3 Generar una petición/respuesta cliente-servidor mediante RESTCONF.

3 INFORME

3.1 Proporcione la configuración construida en laboratorio con una explicación adecuada de los resultados que se muestran.

Como primera parte del laboratorio se ha armado la topología de la Figura 1., configurando las interfaces tanto del cliente como del servidor, dadas en la topología, como se muestra en la Figura 2 y Figura 3. Para comprobar la conectividad entre los dispositivos de la red se ha hecho una prueba de capa 3, ver Figura 4.

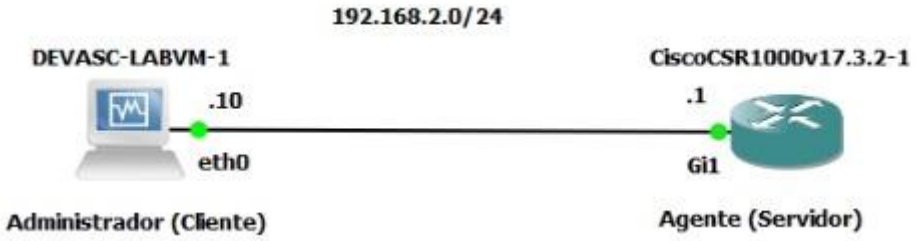

Figura 1. Topología implementada en el laboratorio

```
devasc@labvm:~$ ifconfig enp0s3
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.2.10  netmask 255.255.255.0  broadcast 192.168.2.255
        inet6 fe80::a00:27ff:fee9:3de6  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:e9:3d:e6  txqueuelen 1000  (Ethernet)
        RX packets 187  bytes 45622 (45.6 KB)
        RX errors 0  dropped 5  overruns 0  frame 0
        TX packets 403  bytes 109996 (109.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figura 2. Configuración de la interfaz eth0 del cliente

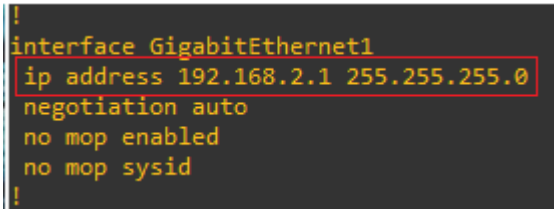

Figura 3. Configuración de la interfaz G1 del servidor

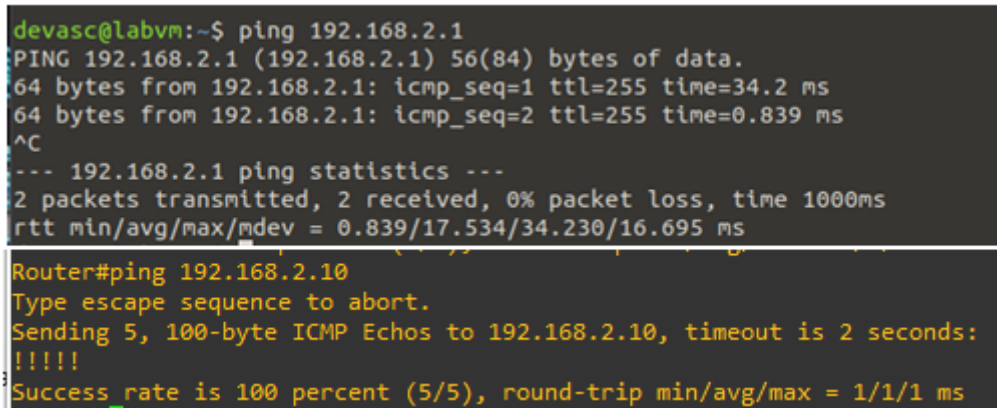

Figura 4. Conectividad cliente-servidor

Como segunda parte del laboratorio, se hizo la configuración básica del router, como cambio de nombre a CSR1K, creación del usuario admin y la habilitación del protocolo SSH para sesiones remotas seguras. Los comandos utilizados se muestran en la Figura 5. En la Figura 6 se prueba que la sesión SSH desde el cliente hasta el servidor fue un éxito.

```
CSR1k#sh running-config
...
!
hostname CSR1k
!
ip domain name labo-restconf.com
!
username admin privilege 15 password 0 admin
!
redundancy
!
ip ssh version 2
!
!
line con 0
 stopbits 1
line vty 0 4
 login local
 transport input ssh
!
...
```

Figura 5. Configuración básica del router

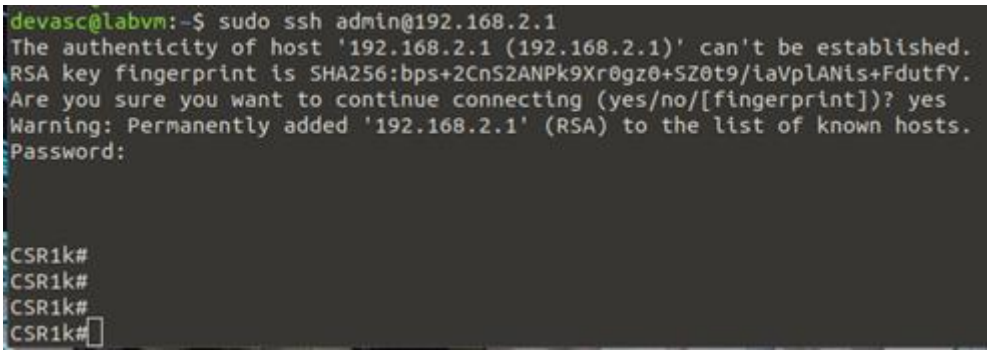

Figura 6. Sesión SSH desde el cliente

Para la tercera parte de la práctica, se ha habilitado el protocolo RESTCONF con los comandos necesarios, tal como se muestra en la Figura 7, y se ha hecho la prueba exitosa de conexión RESTCONF con el comando `curl`, ver Figura 8.

```
CSR1k#sh running-config
...
!
ip forward-protocol nd
ip http server
ip http authentication local
ip http secure-server
!
!
restconf
end
```

Figura 7. Habilitación de la interfaz RESTCONF en el router

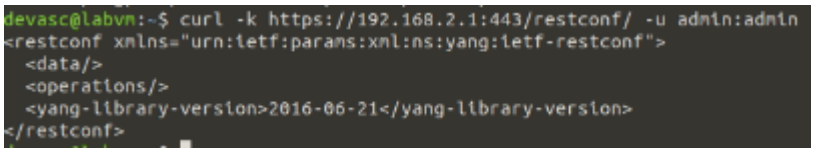

Figura 8. Conexión RESTCONF entre el administrador y el agente

3.2 Responder las preguntas dadas en el procedimiento.

1. **¿Qué muestra el comando?**
   El comando tiene como resultado la Figura 9.

```
CSR1k#show platform software yang-management process
confd            : Not Running
nesd             : Not Running
syncfd           : Not Running
ncsshd           : Not Running
dmiauthd         : Not Running
nginx            : Running
ndbmand          : Not Running
pubd             : Running
```

Figura 9. Salida del comando ejecutado antes de habilitar RESTCONF

2. **¿Qué proceso indica el demonio *nginx*?**
   Cuando un dispositivo arranca con la configuración de inicio, el proceso *nginx* se estará ejecutando. NGINX es un servidor web interno que actúa como un servidor web proxy. Proporciona HTTPS basado en Transport Layer Security (TLS). La solicitud RESTCONF enviada a través de HTTPS es recibida primero por el servidor web proxy NGINX, y la solicitud se transfiere al servidor web *confd* para una verificación adicional de sintaxis/semántica [1].

3. **¿Qué indican los comandos ingresados?**
   Para habilitar la interfaz RESTCONF, primero se ha habilitado HTTPS para que pueda trabajar sobre este protocolo.

4. **¿Cuáles son los logs que ha mostrado el agente?,¿qué indican?**
   A continuación, se muestran los logs del router:

```
*Jun 13 04:03:29.965: %PSD_MOD-5-DMI_NOTIFY_RESTCONF_START:
R0/0: psd: PSD/DMI: restconf server has been notified to start
*Jun 13 04:03:59.314: %NDBMAN-5-ACTIVE: R0/0: ndbmand: All
data providers active.
*Jun 13 04:04:03.492: %DMI-5-NACM_INIT: R0/0: dmiauthd: NACM
configuration has been set to its initial configuration.
*Jun 13 04:04:06.518: %DMI-5-SYNC_COMPLETE: R0/0: dmiauthd:
The running configuration has been synchronized to the NETCONF
running data store.
```

   Al momento de habilitar la interfaz RESTCONF, con los comandos dados en la hoja guía, se pudo observar el proceso que se hace internamente para

habilitar el servidor RESTCONF. Primero, el proceso *psd* notifica al servidor RESTCONF que debe iniciar, después con el proceso *ndbmand* se activan todos los proveedores de datos, y por último el proceso *dmiauthd* hace que la configuración en ejecución se sincronizó con el almacén de datos en ejecución de NETCONF. Se debe recordar que RESTCONF trabaja sobre los mismos fundamentos de NETCONF y su base de datos.

5. **¿Qué muestra el comando?, ¿Cuál es la diferencia entre el resultado de a) y c)?**
   El comando tiene como resultado la Figura 10.

```
CSR1k#show platform software yang-management process
confd            : Running
nesd             : Running
syncfd           : Running
ncsshd           : Not Running
dmiauthd         : Running
nginx            : Running
ndbmand          : Running
pubd             : Running
```

Figura 10. Salida del comando ejecutado después de habilitar RESCONF

En la Figura 10., se muestra todos los demonios que se han habilitado para inicializar el protocolo RESTCONF. Las principales diferencias entre los resultados a) y c) es que se han ejecutado los procesos del yang-management necesarios para RESTCONF, en particular se ejecuta el proceso *dmiauthd* que se vio en los logs del literal 5.

6. **¿Por qué el proceso `ncsshd` no está corriendo?**
   Según [1], `ncssdhd` no está corriendo porque el protocolo NETCONF-YANG no está habilitado en el router.

7. **¿Qué indica cada opción del comando ingresado?**
   El comando que se ingresó en el administrador es:

```
curl -k https://192.168.2.1:443/restconf/ -u "admin:admin"
```

Donde [2]:
`curl [opciones/URLs]`: curl es una herramienta para transferir datos desde o hacia un servidor. Admite entre otros protocolos, estos: FTP, FTPS, HTTP, HTTPS, entre otros. Este comando está diseñado para funcionar sin la interacción del usuario. Además, curl proporciona una variedad de trucos útiles, como compatibilidad con proxy, autenticación de usuarios, cargas FTP, publicaciones HTTP, conexiones SSL, cookies, reanudación de transferencias de archivos y más.

`URLs`: La sintaxis de URL depende del protocolo. Se pueden especificar varias URL o partes de URL, para este comando fue:

https://192.168.2.1:443/restconf/, porque se utiliza el protocolo HTTPS, el puerto :443, el nombre de dominio es la IP del servidor y la ruta al recurso en el servidor web es restconf.

`-k (--insecure)`: De manera predeterminada, se verifica que cada conexión segura que curl realiza sea segura antes de que se realice la transferencia. Esta opción hace que curl se salte el paso de verificación y continúe sin verificar.

`-u (--user <usuario:contraseña>)`: Especifique el nombre de usuario y la contraseña que se usarán para la autenticación del servidor, en el comando son las credenciales para ingresar al router `"admin:admin"`.

8. **¿Cuál es la respuesta?, ¿qué indica?**
   La respuesta se puede visualizar en la Figura 8., esta es una respuesta de datos conceptuales que está estandarizada en el RFC 8040 [3]. Muestra que el protocolo RESCONF es dado por la IETF, en el formato XML, utiliza el modelo YANG y el módulo `ietf-restconf`. Además, identifica la fecha de revisión del módulo `ietf-yang-library` que implementa este servidor RESTCONF, en este caso el 21 de junio de 2016 [4].

9. **¿El agente ha mostrado algún log?, ¿qué indica?**
   El agente ha mostrado el siguiente log:

```
*Jun 13 22:02:37.086: %DMI-5-AUTH_PASSED: R0/0: dmiauthd: User
'admin' authenticated successfully from 192.168.2.10:0 and was
authorized for rest over http. External groups: PRIV15
```

   El log ha mostrado que el usuario `admin` ha sido autentificado exitosamente desde el administrador (192.168.2.10:0), por lo que, el cliente ha sido autorizado para trabajar con REST sobre HTTP (RESTCONF). El usuario `admin` tiene privilegios PRIV15, el nivel más alto para configurar el router.

4 CONCLUSIONES

En esta práctica se ha simulado en GNS3 y configurado una red simple, router-terminal, teniendo conectividad entre los equipos y desde el terminal se puede ingresar mediante SSH al router.

El router CSR1k ha sido configurado con los comandos dados para que sea habilitado el protocolo RESTCONF, esto permitió la creación del cliente (terminal) y el servidor (router).

Mediante el uso de la herramienta CURL, se ha generado una petición desde el administrador al agente, dando una respuesta exitosa, concluyendo que se ha configurado correctamente el protocolo RESTCONF en la red.

5 RECOMENDACIONES

El emulador GNS3 depende mucho de la capacidad de la máquina, por lo que, se recomienda tener una PC con una memoria RAM mayor a los 16 GB, una de las

razones es que el router que se utiliza en la práctica, es un elemento de red complejo, que necesita mucha capacidad para realizar los procesos requeridos.

El funcionamiento de GNS3 puede variar según la máquina que se utilice, por lo que, se recomienda, utilizar el foro de GNS3 para solucionar los problemas que puedan ocurrir.

Es necesario tener conocimiento del protocolo NETCONF, HTTP, HTTPS y el modelo de datos YANG, ya que estos son los fundamentos en los cuales RESTCONF se desarrolla.

6 REFERENCIAS

[1] Cisco. “Programmability Configuration Guide, Cisco IOS XE Fuji 16.7.x.” Internet: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/prog/configuration/167/b_167_programmability_cg/restconf_programmable_interface.html, Feb. 5, 2021 [Jun. 13, 2022].

[2] D. Stenberg. “curl.1 the man page.” Internet: https://curl.se/docs/manpage.html, [Jun. 13, 2022].

[3] A. Bierman, YumaWorks, M. Bjorklund, Tail-f Systems, K. Watsen and Jupiter Networks. “RESTCONF Protocol.” IETF RFC 8040, Ene. 2017.

[4] E. Nilsen-Nygaard. “yang/vendor/cisco/xe/1731/ietf-restconf.yang.” Internet: https://github.com/YangModels/yang/blob/main/vendor/cisco/xe/1731/ietf-restconf.yang, Ago 4, 2020 [Jun 13, 2022].

PRÁCTICA N° 2

1 TEMA

LENGUAJE DE MODELADO DE DATOS YANG

2 OBJETIVOS

2.1 Analizar el modelo de datos YANG.

2.2 Identificar el modelo de datos YANG con el comando `pyang`.

2.3 Comparar entre los tipos de datos estructurados XML y JSON.

3 INFORME

3.1 Proporcione la configuración construida en laboratorio con una explicación adecuada de los resultados que se muestran.

Como primera parte del laboratorio se ha armado la topología de la Figura 1, configurando las interfaces tanto del cliente como del servidor, dadas en la topología, como se muestra en la Figura 2 y Figura 3. Para comprobar la conectividad entre los dispositivos de la red se ha hecho una prueba de capa 3, ver Figura 4. El administrador tiene conexión a internet, como se prueba al hacer un ping al DNS de Google, esto se muestra en la Figura 5.

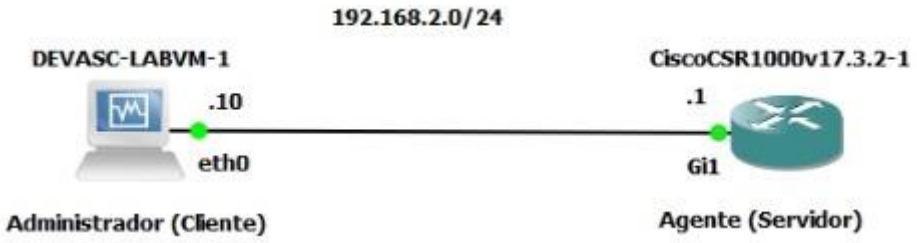

Figura 1. Topología implementada en el laboratorio

```
devasc@labvm:~$ ifconfig enp0s3
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.2.10  netmask 255.255.255.0  broadcast 192.168.2.255
        inet6 fe80::a00:27ff:fee9:3de6  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:e9:3d:e6  txqueuelen 1000  (Ethernet)
        RX packets 187  bytes 45622 (45.6 KB)
        RX errors 0  dropped 5  overruns 0  frame 0
        TX packets 403  bytes 109996 (109.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figura 2. Configuración de la interfaz eth0 del cliente

```
!
interface GigabitEthernet1
 ip address 192.168.2.1 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
```

Figura 3. Configuración de la interfaz G1 del servidor

```
devasc@labvm:~$ ping 192.168.2.1
PING 192.168.2.1 (192.168.2.1) 56(84) bytes of data.
64 bytes from 192.168.2.1: icmp_seq=1 ttl=255 time=34.2 ms
64 bytes from 192.168.2.1: icmp_seq=2 ttl=255 time=0.839 ms
^C
--- 192.168.2.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.839/17.534/34.230/16.695 ms
```

```
Router#ping 192.168.2.10
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.2.10, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
```

Figura 4. Conectividad cliente-servidor

```
devasc@labvm:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=107 time=21.5 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=107 time=20.6 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=107 time=20.9 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=107 time=20.3 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=107 time=20.2 ms
^C
--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4010ms
rtt min/avg/max/mdev = 20.235/20.717/21.517/0.458 ms
```

Figura 5. Conectividad cliente-internet

Después de haber configurado la red a utilizar, se procedió a realizar la segunda parte de estudio de los módulos YANG en el sitio web GitHub. En la Figura 6 se muestran algunos de los modelos YANG que soporta el router Cisco-IOS-XE-17.3.1, que es el que se está ocupando en el laboratorio.

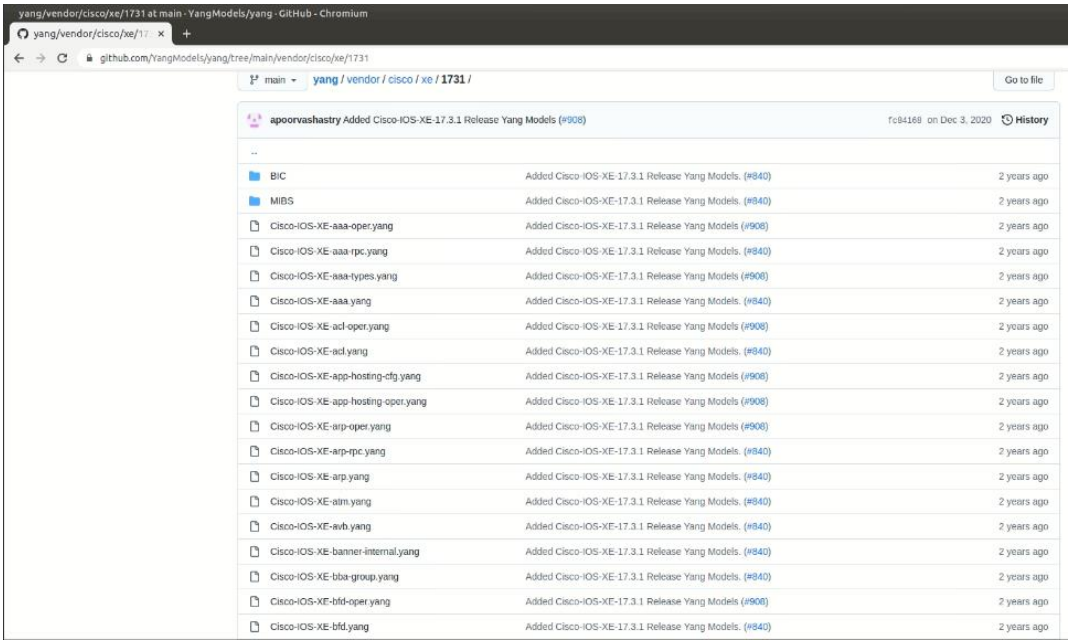

Figura 6. Modelos YANG que soporta el router CSR1k

Se puede observar que el router soporta todos los tipos de modelo YANG que existen como es el nativo de Cisco, OpenConfig e IETF. En la práctica se han comparado dos modelos YANG de datos de interfaces del routers, identificando los nodos `leaf`, `leaf-list`, `container` y `list`. De los modelos `ietf-interfaces.yang` y `openconfig-interfaces.yang` se puede tener bastante información de las interfaces como: nombre, descripción, dirección IP y física, máscara, entrada y salida de paquetes, velocidad, estado de la interfaz, entre otros. OpenConfig tienen características similares ya que este se ha basado en los nodos del modelo YANG de IETF, añadiendo algunos atributos más para realizar la interoperabilidad con otros dispositivos de otras marcas.

En la segunda parte del laboratorio, se pudo observar que la sintaxis de los modelos YANG, es muy compleja y difícil de entender, por lo que, en la tercera parte de la práctica, se puede visualizar de manera más ordenada con la herramienta Python llamada `pyang`. Para esto se han descargado y almacenado los modelos anteriores en la carpeta `pyang` creada en el administrador, como se muestra en la Figura 7.

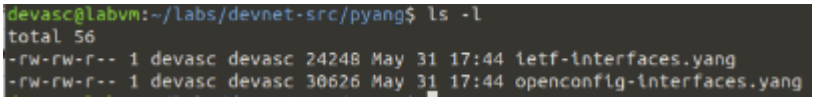

Figura 7. Modelos YANG descargados en la carpeta `pyang`

3.2 Responder las preguntas dadas en el procedimiento.

1. **¿A qué estructuras se puede transformar el modelo YANG y con cuál opción?**

   Según la información dada en el manual de `pyang`, la opción para transformar un modelo de datos a otros formatos es:

   ```
   pyang -f [opción] archivo
   ```

   Las opciones a las que se puede transformar son: *yang, yin, dsdl, identifiers, tree, name, uml, depend, omni, sample-xml-skeleton, capability, jsonxsl, jstree, jtox.*

2. **¿Qué propiedades tienen las representaciones gráficas del diagrama del árbol `rw, ro, x, ?, !` y `*`?**

   En el diagrama del árbol se utiliza una representación gráfica simplificada del modelo de datos [1]. Los significados de los símbolos en estos diagramas son los siguientes:

   - **Abreviaturas antes de los nombres de los nodos de datos:** "`rw`" significa datos de configuración (lectura-escritura), "`ro`" significa datos de estado (solo lectura) y "`x`" significa recurso de operación (ejecutable).
   - **Símbolos después de los nombres de los nodos de datos:** "`?`" significa un nodo opcional, "`!`" significa un contenedor de presencia, y "`*`" denota un nodo list y leaf-list.

3. **¿Qué comando usaría para transformar a modo árbol el módulo de interfaces de OPENCONFIG?**

   Se utilizaría el siguiente comando en el administrador:

   ```
   devasc@labvm:~/labs/devnet-src/pyang$ pyang -f tree openconfig-interfaces.yang
   ```

4. **Realizar el análisis hecho en la Figura 4 de la Hoja Guía 2 con la salida de la Pregunta 3.**

   El diagrama del árbol para el modelo OpenConfig es mucho más complejo, con más atributos que el modelo IETF, como se muestra en la Figura 8.

Figura 8. Módulo `openconfig-interfaces` en forma de árbol [2]

5. **Realizar los literales a) y b) para el archivo `example-ietf-interfaces-data.json`.**

   En el módulo YANG en lenguaje JSON, se tiene la información de interfaces, y estas contienen los elementos de un modelo YANG, como se muestra en la Figura 9.

Figura 9. Ejemplo de datos mostrados en formato JSON

4 CONCLUSIONES

Con la gran cantidad de datos de red que se tiene, se ha necesitado que estos tengan un formato, un lenguaje y una jerarquía definida para ser tratados de manera correcta, esto dio paso al modelo de datos YANG, que mejora la capacidad de manejo de datos en los elementos de red.

Una herramienta útil para la identificación del contenido y jerarquía de un modelo de datos YANG es el comando `pyang` para sistemas Linux, esto nos ha permitido tener mayor visualización y entendimiento sobre el contenido de cada modelo de datos YANG.

Los lenguajes XML y JSON han sido fundamentales para el desarrollo web basado en el modelo de datos YANG, tienen diferencias entre ellos, pero se concluye que depende del desarrollador, la utilización de un lenguaje específico.

5 RECOMENDACIONES

Para tener mayor entendimiento acerca del modelo de datos YANG, es necesario la lectura y estudio del RFC 6020 y RFC 7950 y las aplicaciones de YANG sobre la gestión IP, interfaces o instancias dadas en los RFC 8344, 8343 y 8529, respectivamente.

Es recomendable que para el protocolo RESTCONF se utilice el lenguaje JSON, ya que, el protocolo lo soporta, y es más amigable con el usuario.

6 REFERENCIAS

[1] M. Bjorklund, Ed. and Tail-f Systems. “YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF).” IETF RFC 6020, Oct. 2010.

[2] A. Guamán. “Análisis de protocolos para la gestión de redes: Análisis del protocolo NETCONF.” Tesis, Escuela Politécnica Nacional, Quito, Ecuador, 2022.

[3] O. Blancartet. “JSON vs XML.” Internet: https://www.oscarblancarteblog.com/2014/07/18/json-vs-xml/, Jul. 18, 2014 [Jun. 18, 2022].

PRÁCTICA N° 3

1 TEMA

SEGURIDAD Y MÉTODOS HTTP

2 OBJETIVOS

2.1 Implementar ACLs para seguridad.
2.2 Analizar la herramienta cURL para el protocolo RESTCONF.
2.3 Identificar los diferentes métodos RESTCONF.

3 INFORME

3.1 Proporcione la configuración construida en laboratorio con una explicación adecuada de los resultados que se muestran.

La primera parte del laboratorio consiste en realizar la topología de la Hoja Guía 3, como se muestra en la Figura 1, donde se puede notar que el router tiene dos redes independientes la LAN 1 (192.168.2.0/24) y la LAN 2 (192.168.3.0/24).

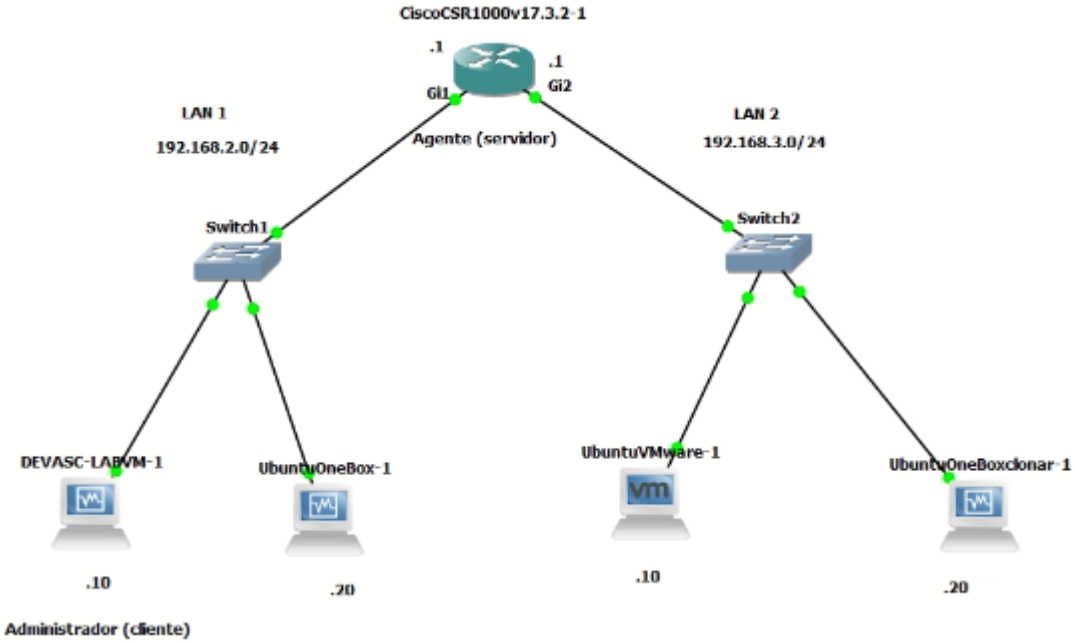

Figura 1. Topología de red implementada

La configuración del router, se puede observar en la Figura 2, tanto la configuración básica, de las interfaces GigabitEthernet1 y GigabitEthernet2 y la habilitación del protocolo RESTCONF.

Además, se han configurado las máquinas virtuales respecto a sus redes, como se muestra en la Figura 3, teniendo conectividad, tanto a internet como en sus propias LANs, ver la Figura 4 y Figura 5.

```
!
hostname CSR1k
!
ip domain name labo-restconf.com
!
username admin privilege 15 password 0 admin
!
interface GigabitEthernet1
 ip address 192.168.2.1 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
interface GigabitEthernet2
 ip address 192.168.3.1 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
ip forward-protocol nd
ip http server
ip http authentication local
ip http secure-server
!
restconf
end
```

Figura 2. Configuración inicial del agente

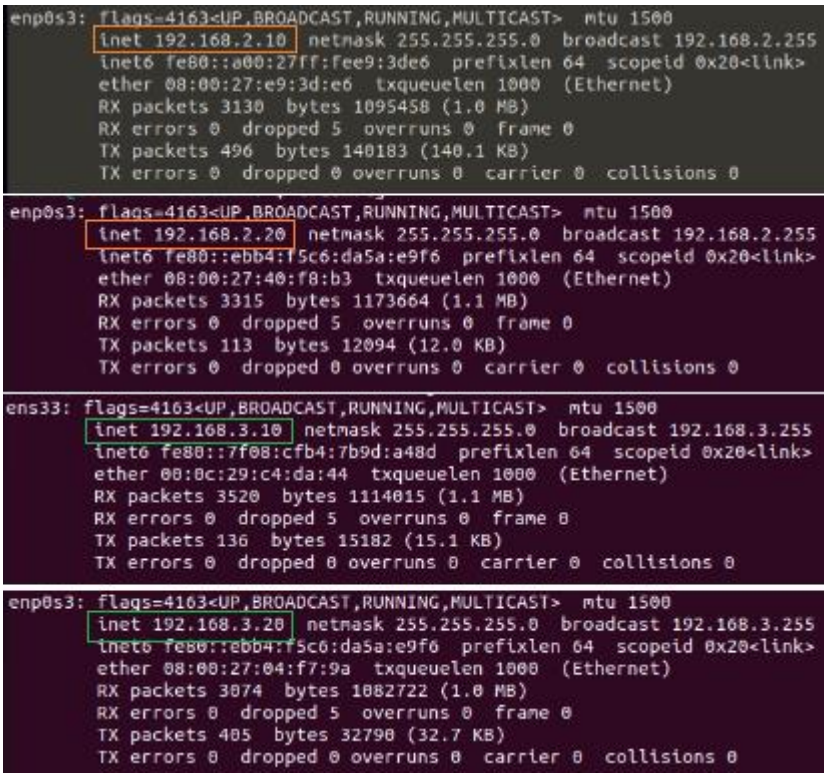

Figura 3. Direcciones IP de los hosts de las redes LAN 1 y LAN 2

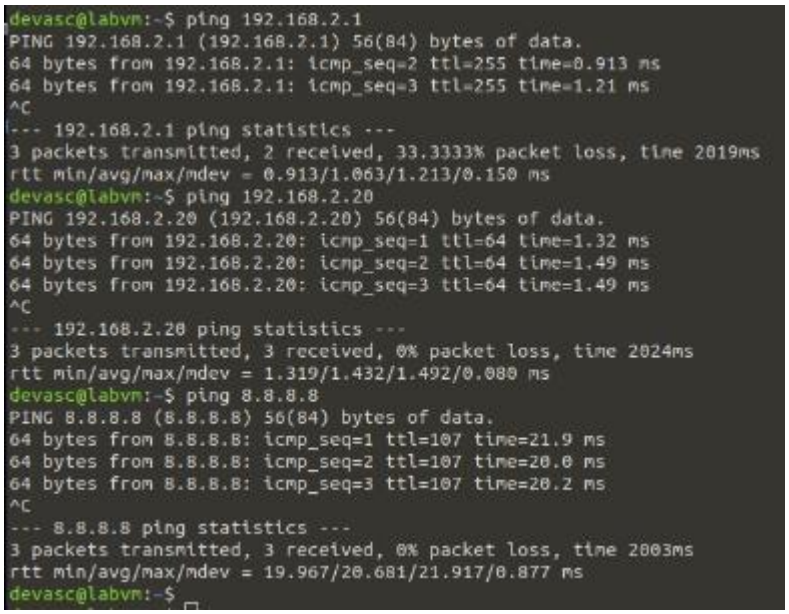

Figura 4. Conectividad en la LAN 1 e internet

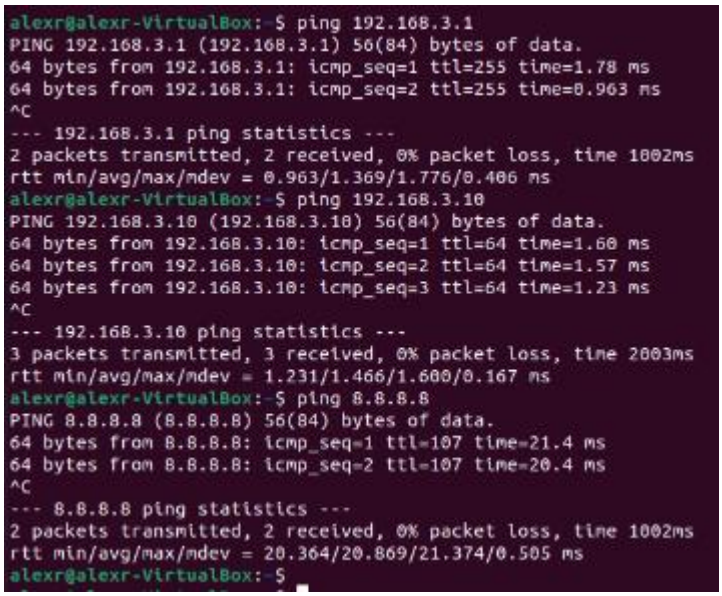

Figura 5. Conectividad en la LAN 2 e internet

Al tener ya habilitado RESTCONF en el agente, se ha hecho la prueba de conexión desde cada uno de los hosts, considerando la interfaz propia del agente como su propia IP de servidor, tal como se muestra en la Figura 6 y Figura 7.

Figura 6. Conexión RESTCONF desde los hosts de la LAN 1

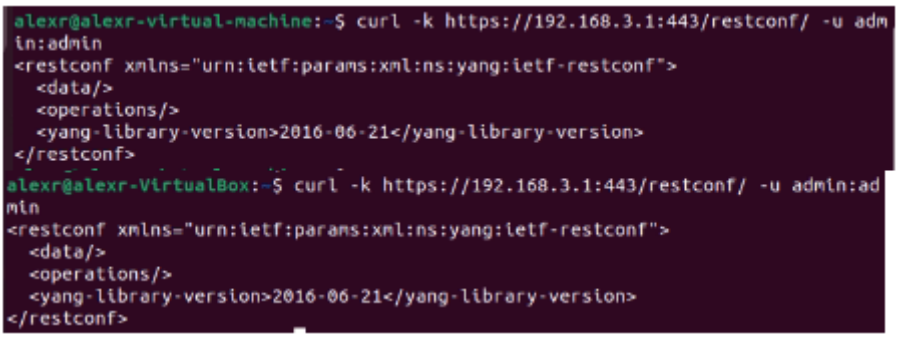

Figura 7. Conexión RESTCONF desde los hosts de la LAN 2

Una sección de esta primera parte de la práctica de laboratorio consiste en la seguridad que se puede tener para que ciertas redes y/o equipos pueden entrar al protocolo RESTCONF utilizando ACLs. La Hoja Guía 3 establece que se cree una ACL llamada `permit_REST` que permita ingresar al servidor solo desde la red 192.168.2.0/24, esta configuración se hizo en el agente y se muestra en la Figura 8.

```
!
ip access-list standard permit_REST
 10 permit 192.168.2.0 0.0.0.255
!
restconf
restconf ipv4 access-list name permit_REST
end
```

Figura 8. Configuración de la ACL para RESTCONF

3.2 Responder las preguntas dadas en el procedimiento.

1. ¿Cuál es la IP del servidor en la red 192.168.2.0/24? y ¿en la 192.168.3.0/24? ¿Por qué son las mismas?

La dirección IP del servidor RESTCONF tanto para la LAN 1 (192.168.2.0/24) como para la LAN 2 (192.168.3.0/4), es cualquier dirección IP de las interfaces del agente, pues para el router, las redes LAN son conocidas, y se puede ingresar al protocolo RESTCONF desde cualquier IP de interfaz, sea de la 192.168.2.1:443 o la 192.168.3.1:443.

2. ¿El agente ha mostrado algún log?, ¿qué indica?

El agente ha mostrado los siguientes logs después de realizar la conexión de todos los hosts al servidor RESTCONF.

```
*Jun 25 03:02:28.269: %DMI-5-AUTH_PASSED: R0/0: dmiauthd: User
'admin' authenticated successfully from 192.168.2.10:0 and was
authorized for rest over http. External groups: PRIV15
*Jun 25 03:03:11.985: %DMI-5-AUTH_PASSED: R0/0: dmiauthd: User
'admin' authenticated successfully from 192.168.2.20:0 and was
authorized for rest over http. External groups: PRIV15
*Jun 25 03:04:22.513: %DMI-5-AUTH_PASSED: R0/0: dmiauthd: User
'admin' authenticated successfully from 192.168.3.20:0 and was
authorized for rest over http. External groups: PRIV15
*Jun 25 03:04:38.672: %DMI-5-AUTH_PASSED: R0/0: dmiauthd: User
'admin' authenticated successfully from 192.168.3.10:0 and was
authorized for rest over http. External groups: PRIV15
```

En estos logs se puede observar que el demonio `dmiauthd` indica qué usuario se ha autenticado exitosamente para tener acceso al protocolo RESTCONF, además, nos indica la dirección IP desde la que se hizo el proceso, observando que se hizo desde las cuatro terminales de la topología dada.

3. ¿Cuáles son los resultados?, comente.

La ACL permite que solo los hosts que estén dentro de la red 192.168.2.0/24 puedan acceder a RESTCONF (ver Figura 9); y las demás redes no (ver Figura 10.), entregando un mensaje de error (acceso denegado).

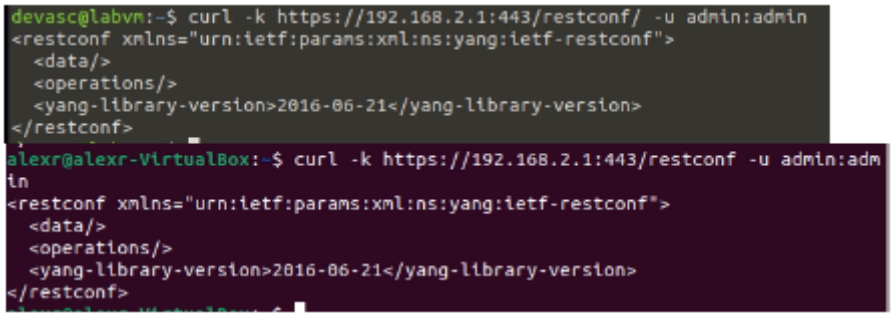

Figura 9. Los terminales de la LAN 1 pueden utilizar RESTCONF

Figura 10. Los terminales de la LAN 2 no pueden utilizar RESTCONF

4. ¿Cuál sería el procedimiento para permitir ingresar a RESTCONF solo desde el administrador DEVASC-LABVM?

Para permitir que solo el administrador 192.168.2.10/824 tenga acceso al servidor, se debería hacer una nueva ACL y anexarle al protocolo RESTCONF, con los siguientes comandos:

```
enable
configure terminal
ip access-list standard permit_ADMIN
permit host 192.168.2.10
end
configure terminal
restconf ipv4 access-list name permit_ADMIN
end
```

En el agente se ha establecido este nuevo ACL y solo se puede acceder a RESTCONF desde el administrador DEVASC-LABVM, como se observa en la Figura 11.

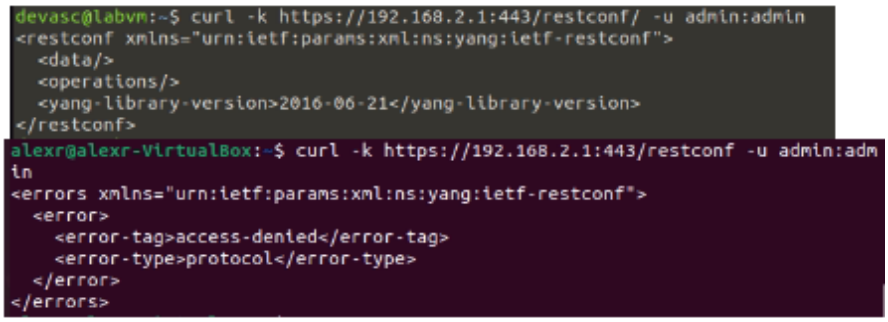

Figura 11. Acceso a RESTCONF solo desde el administrador principal

5. Describa las ventajas de este tipo de seguridad para el manejo del protocolo RESTCONF.

Al ser las ACL filtros de red utilizados por routers o firewalls, estas nos permiten restringir o permitir la circulación de paquetes dentro de la interfaz de red, esto hace que sea absolutamente indispensable en el área de la seguridad de la red [1].

En específico, RESTCONF tiene la gestión de las redes dentro del router, si se deja abierto para que cualquier host puede tener acceso a este protocolo, se tiene una gran posibilidad de tener ataques a la gestión y toda la red, como un ataque de Hombre en Medio (MitM-*Man-in-the-Middle*), haciendo vulnerable la red y todos los datos que pasan por ahí, por lo tanto, tener una ACL bien configurada, con contraseñas seguras de usuarios es indispensable para la seguridad de la red.

6. ¿Cuál es el resultado?, ¿qué características muestra?

Para esta parte de la práctica se utiliza la herramienta cURL, como se muestra en la Figura 12.

Esta muestra el contenido del modelo YANG `ietf-interfaces` y el contenedor `interfaces`, mostrando datos importantes de las cuatro interfaces del agente, como nombre, tipo, dirección IP y si está habilitado o no.

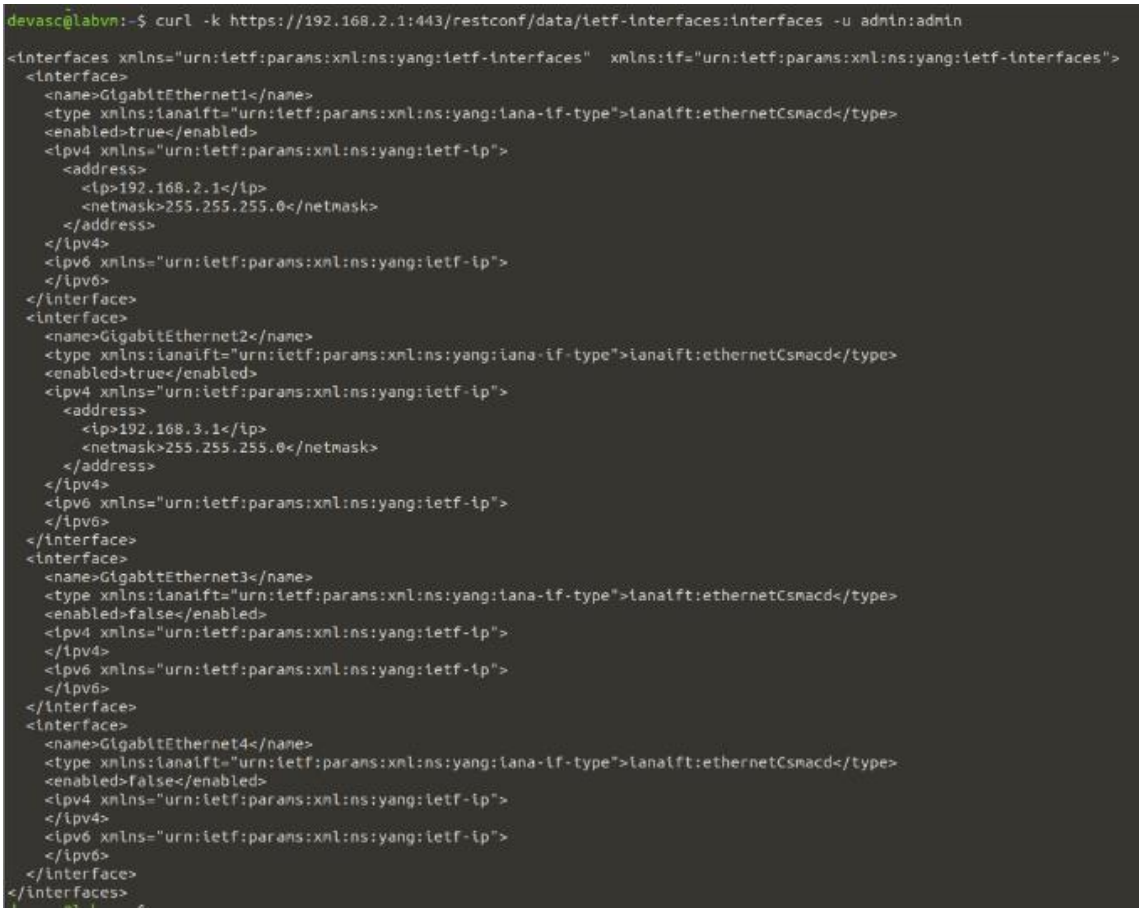

Figura 12. Información de las interfaces del agente con cURL

7. ¿En qué formato muestra el resultado?, ¿cuál es la opción para cambiar a JSON?

La Figura 12. muestra su resultado en formato XML, y para cambiar de formato se debe utilizar la opción -H; como el siguiente comando:

```
curl -k -H 'Accept: application/yang-data+json'
https://192.168.2.1:443/restconf/data/ietf-interfaces:interfaces -u
admin:admin
```

El resultado se muestra en la Figura 13.

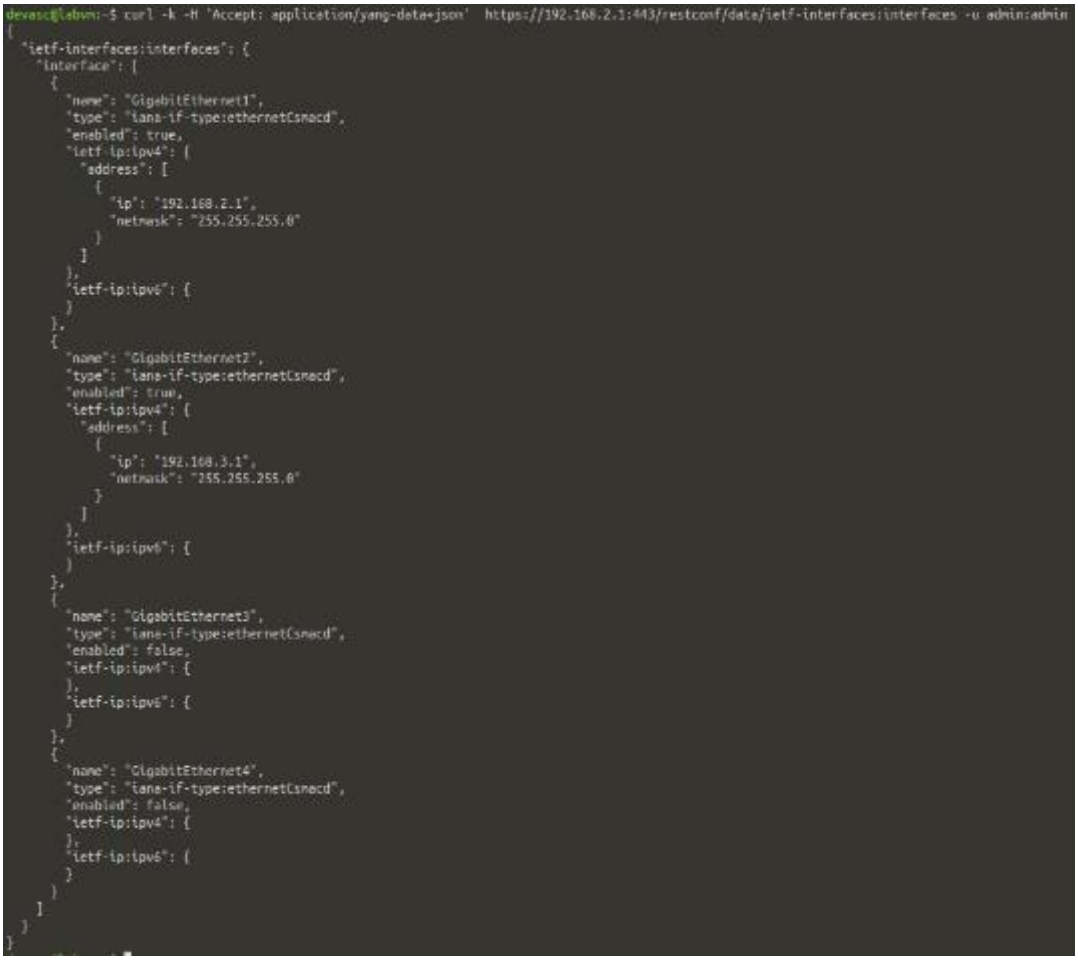

Figura 13. Información de las interfaces del agente en formato JSON

8. ¿Cuál es el resultado?, ¿qué características muestra?

Se ha corrido el comando en el administrador, teniendo el resultado de la Figura 14.

Figura 14. Resultado exitoso para el método PATCH

La URL ejecutada mediante la herramienta cURL era para bajar el interfaz Gi2, utilizando el método PATCH, enviando y recibiendo información en formato JSON. La respuesta a esta petición ha sido un mensaje 204 indicando que es exitosa pero no muestra ninguna información. Para verificar, se ha corrido el comando de la Figura 15 en el agente, teniendo como resultado que efectivamente, se ha bajado la Gi2, utilizando la herramienta cURL en el protocolo RESTCONF.

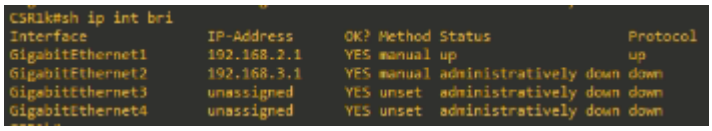

Figura 15. Información resumida sobre las interfaces del agente

9. ¿Es eficaz la herramienta cURL para gestionar la red con RESTCONF?

La herramienta cURL tiene bastantes dificultades al momento de utilizar otros métodos, aparte del GET, pues es más complicado poner la información en los formatos establecidos, se utiliza muchas opciones para una simple petición, por lo que, no es eficaz utilizar cURL dentro de la gestión RESTCONF.

4 CONCLUSIONES

Como se ha visto en la práctica, la necesidad de utilizar ACLs es indispensable para mantener la seguridad de la red, los privilegios de los usuarios, las contraseñas de estos son igualmente importantes para mantener segura la red de ataques externos que pueden poner en peligro la data que se transmite dentro de esta.

El protocolo RESTCONF permite realizar acciones de gestión para las redes, sacar información, actualizar datos, entre otras cosas, por lo que, la herramienta cURL nos permite realizar estos procesos mediante URLs, con opciones, que en ocasiones dificulta explotar la habilidad que tiene RESTCONF, por lo tanto, cURL se debe utilizar para familiarizarse con métodos HTTPS, pero no es eficaz para la gestión de redes mediante RESTCONF.

Los métodos HTTP son indispensables para trabajar sobre RESTCONF, por lo que, su conocimiento es importante para la gestión de rede mediante este protocolo.

5 RECOMENDACIONES

Para mayor seguridad se deben tener en el agente, más usuarios definidos para la gestión de la red, con contraseñas seguras y personales.

Buscar otras herramientas que permitan la automatización y programabilidad de la gestión de red mediante RESTCONF, estas pueden ser POSTMAN o Python.

6 REFERENCIAS

[1] R. Achary. *Cryptography and Network Security: An Introduction*. Dules, VA: Mercury Learning and Information, 2021.

PRÁCTICA N° 4

1 TEMA

GESTIÓN RESTCONF CON POSTMAN

2 OBJETIVOS

2.1 Analizar la herramienta POSTMAN para el protocolo RESTCONF.

2.2 Configurar interfaces y ruta estática mediante RESTCONF.

2.3 Diagnosticar la red mediante RESTCONF.

3 INFORME

3.1 Proporcione la configuración construida en laboratorio con una explicación adecuada de los resultados que se muestran.

Para la primera parte de la práctica se ha implementado la topología dada en la Figura 1, en esta red se puede observar la conexión entre dos redes autónomas, una la red 192.168.2.0/24 y la 192.168.3.0/24. Los routers están dentro de la red 10.1.1.0/24. La configuración de los routers se encuentra en la Figura 2 y Figura 3.

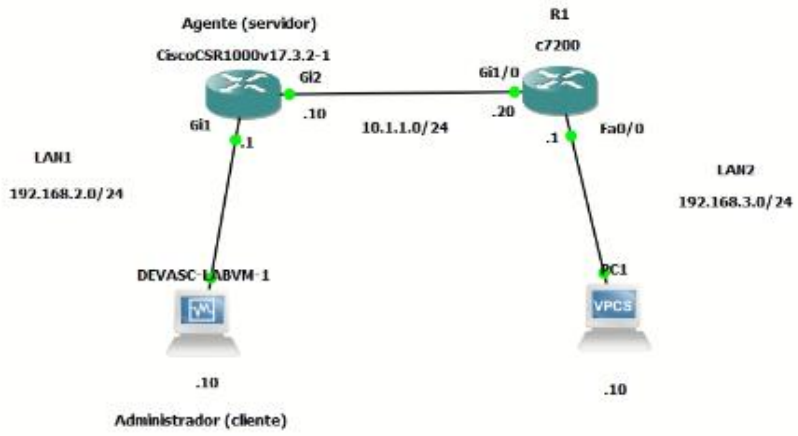

Figura 1. Topología de red implementada

```
!
hostname R1
!
interface FastEthernet0/0
 ip address 192.168.3.1 255.255.255.0
 duplex half
!
interface GigabitEthernet1/0
 ip address 10.1.1.20 255.255.255.0
 negotiation auto
!
end
```

Figura 2. Configuración router R1

```
!
hostname CSR1k
!
ip domain name labo-restconf.com
!
username admin privilege 15 password 0 admin
!
interface GigabitEthernet1
 ip address 192.168.2.1 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
ip forward-protocol nd
ip http server
ip http authentication local
ip http secure-server
!
restconf
end
```

Figura 3. Configuración agente CSR1k

En la Figura 4 se muestra la configuración de las direcciones IP del administrador y de la PC1.

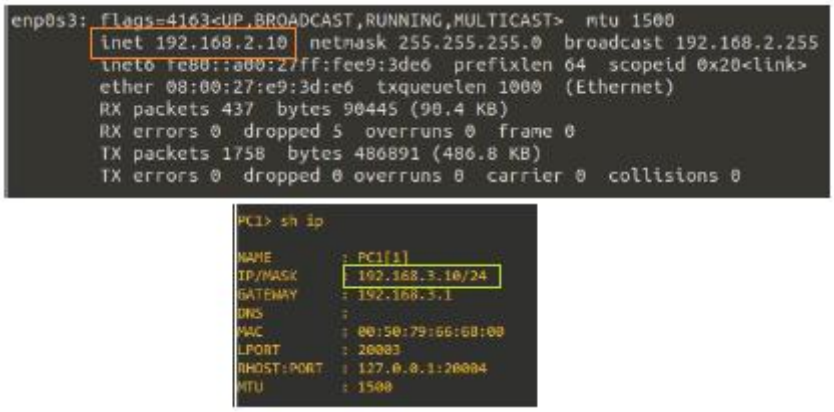

Figura 4. Direcciones IP de los hosts

Dado que, en la topología, en la LAN 1 solo se ha configurado la interfaz GigabitEthert1, el host administrador solo tiene conectividad a esa interfaz y a internet, como se muestra en la Figura 5.

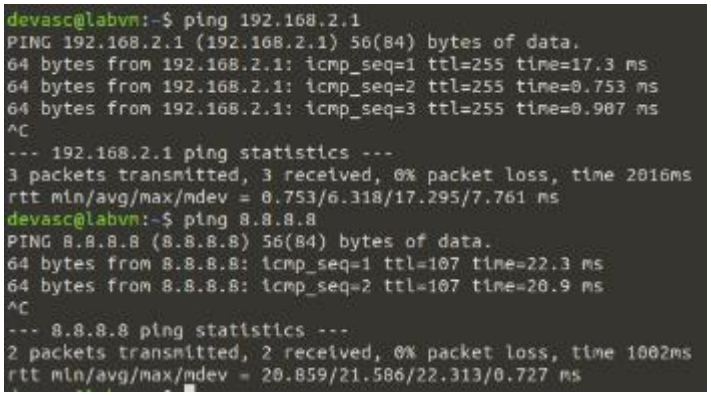

Figura 5. Conectividad del host de la LAN1

En cambio, dado los requerimientos en la Hoja Guía 4, el host PC1, tiene conectividad a las direcciones IP de las dos interfaces configuradas, como se observa en la Figura 6.

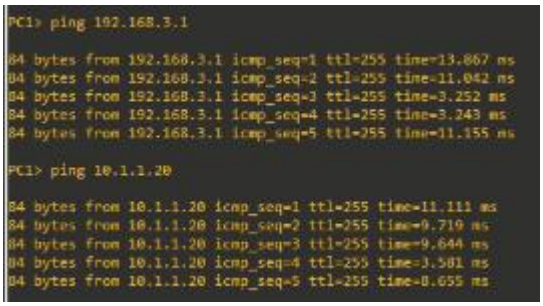

Figura 6. Conectividad del host de la LAN2

Dado que, en la LAN 1, en el agente se ha configurado RESTCONF, se hace la prueba de conectividad con el servidor mediante cURL, ver la Figura 7.

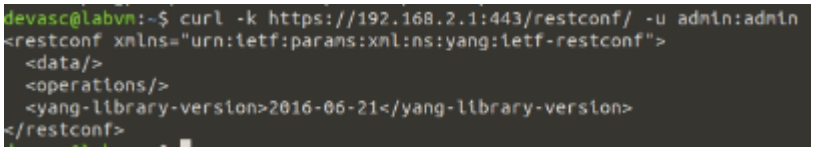

Figura 7. Envío de solicitud cURL al servidor RESTCONF

Ahora se va a trabajar en POSTMAN, por lo que, en el administrador se ejecuta el programa, como se observa en la Figura 8.

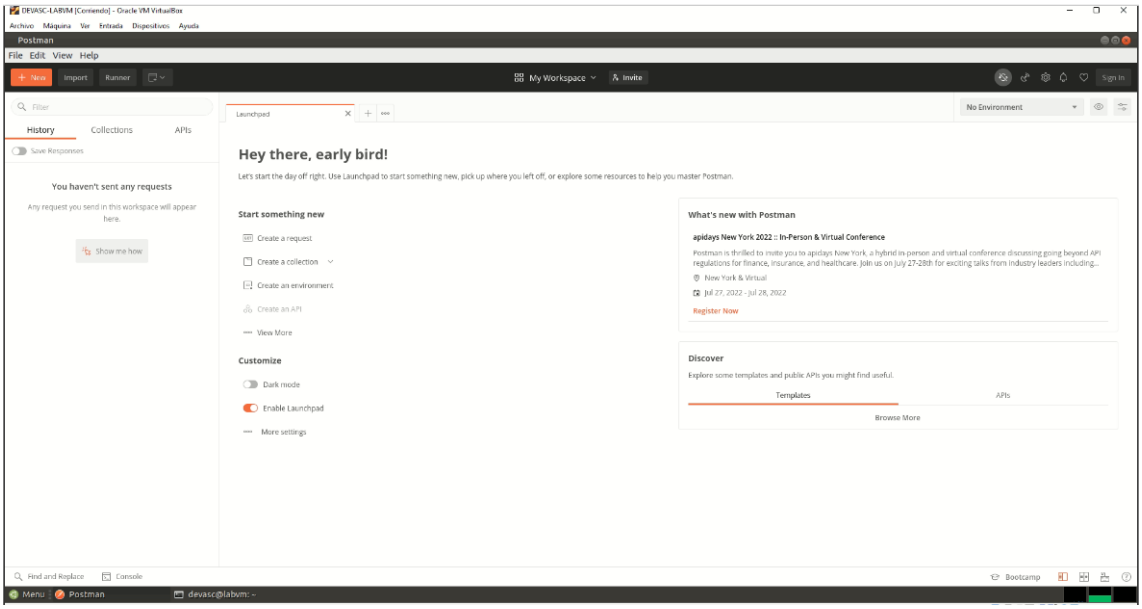

Figura 8. POSTMAN en el administrador

Para evitar problemas con la certificación SSL, se ha deshabilitado esta opción en POSTMAN, como se ve en la Figura 9.

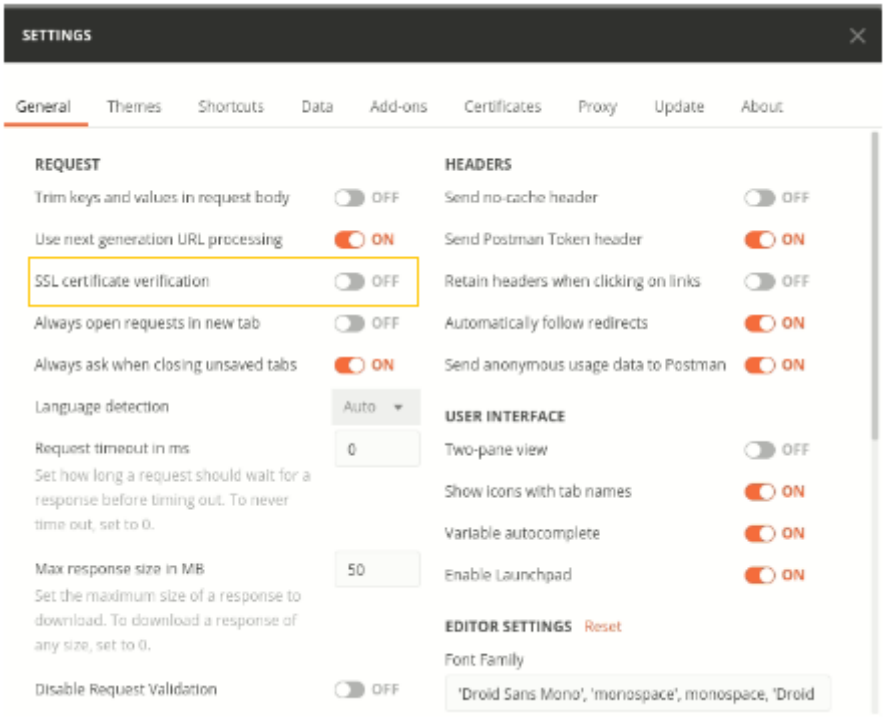

Figura 9. Verificación del certificado SSL en off

Para esta práctica se ha creado una colección llamada Practica 4, para tener todas las APIs realizadas en este laboratorio, ver la Figura 10.

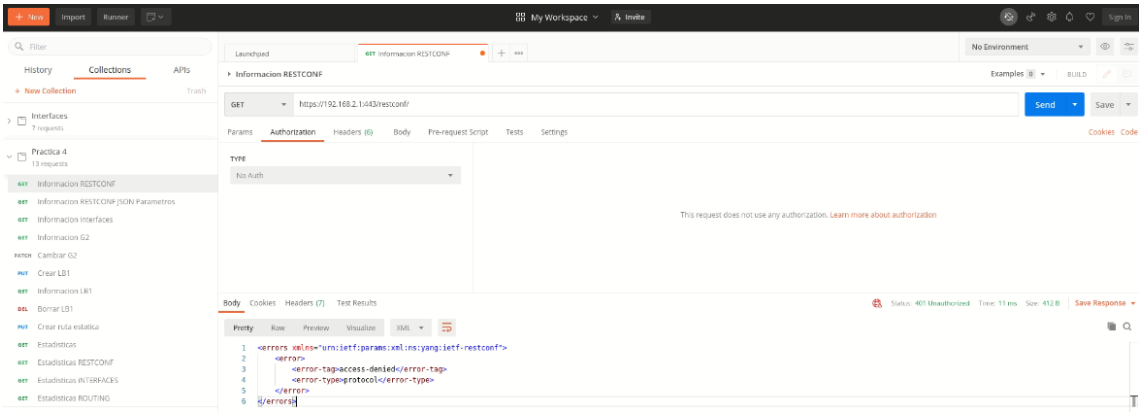

Figura 10. Colección Práctica 4

3.2 Responder las preguntas dadas en el procedimiento.

1. En el comando cURL, ¿por cuál opción se está realizando este paso?

El paso de deshabilitar la verificación del certificado SSL, se hace en reemplazo de la opción `-k (--insecure)`, pues tanto cURL y POSTMAN de manera predeterminada, verifica que cada conexión que realiza sea segura antes de que se realice la transferencia. Esta opción hace que curl se salte el paso de verificación y continúe sin verificar [1].

2. ¿Cuál es el resultado?, ¿qué código de respuesta dio el servidor?

Como resultado, se ha dado la información básica de RESTCONF en formato JSON, por lo que, el código de respuesta ha sido 200 Ok, que indica el éxito de la petición, cómo se muestra en la Figura 11.

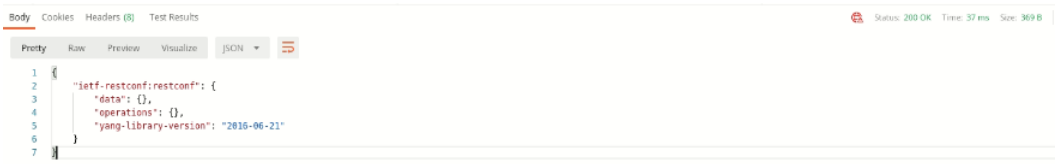

Figura 11. Respuesta a la API Información RESTCONF

3. ¿Qué resultado muestra esta API?, ¿la interfaz Gi2, que información tiene?

Se puede observar, que la interfaz Gi2 no está levantada ni configurada. El código de respuesta es 200 OK.

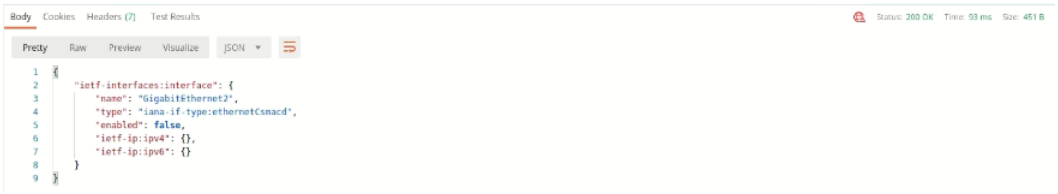

Figura 12. Respuesta a la API Información G2

4. ¿Qué resultado muestra esta API?

La respuesta de la API ha sido un 204 No Content, esto quiere decir que se ha configurado de manera exitosa a la interfaz Gi2, pero no se tiene nada que devolver como información de respuesta.

5. Enviar el API Información G2, ¿cuál es el nuevo resultado?
En la Figura 13 se muestra el resultado de la API con la nueva información de la interfaz Gi2.

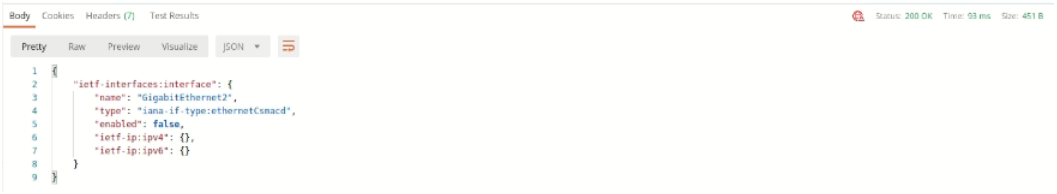

Figura 13. Respuesta a la API Información G2 después de configurar Gi2

6. En el agente, correr el comando para ver el estado de las interfaces ¿cuál es el resultado?
Al ser configurada la interfaz Gi2, desde el administrador a través de RESTCONF, se tiene que la interfaz ha sido levantada y configurada, como se muestra en la Figura 14.

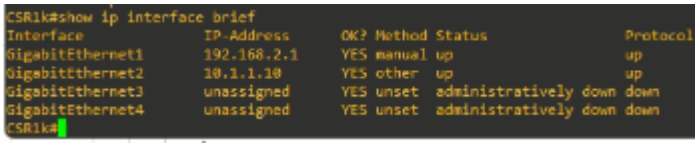

Figura 14. Información de las interfaces del agente

7. ¿Qué resultado muestra esta API?
El resultado a la API *Información LB1* es un 404 Not Found lo que nos dice que el recurso solicitado no se encontró en el servidor. Esto tiene lógica, ya que no existe, al momento, ninguna interfaz loopback creada, como se observa en la Figura 14.

8. ¿Qué resultado muestra esta API?
El resultado a la API *Crear LB1* es un 201 Created que nos indica que el recurso solicitado ha sido creado, es decir que la interfaz Loopback1, ha sido creada con los parámetros dados en el API.

9. Enviar el API Información LB1, ¿cuál es el nuevo resultado?
En la Figura 15. se muestra el resultado de la API con la nueva información de la interfaz LB1.

Figura 15. Respuesta a la API Información LB1 después de crear LB1

10. En el agente, correr el comando para ver el estado de las interfaces ¿cuál es el resultado?

Al ser creada y configurada la interfaz Loopback1, desde el administrador a través de RESTCONF, se tiene que la interfaz ha sido creada, levantada y configurada, como se muestra en la Figura 16.

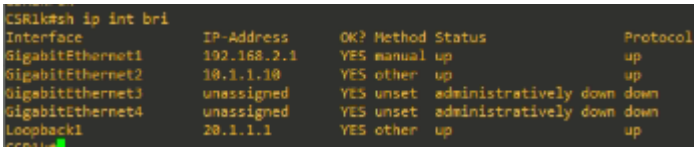

Figura 16. Información de las interfaces del agente incluye LB1

11. ¿Cuál es el método y la URL para esta API?
La API *Borrar LB1* se ha creado con el método mostrado en la Figura 17.

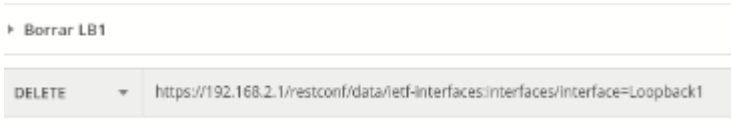

Figura 17. Método y URL para API Borrar LB1

12. Comprobar que la interfaz LB1 ha sido eliminada.
Al ejecutar el API *Borrar LB1* se ha tenido el código 204 No Content, que nos indica que la eliminación ha sido exitosa, por lo que, no se puede mostrar algo que ya no existe. Para corroborar, se ha ejecutado el API *Información LB1* que dio el mensaje 404 Not Found y en el agente se muestra la información de la Figura 14, confirmando que la interfaz Loopback1 ha sido eliminada.

13. ¿Qué resultado muestra esta API?,¿el resultado muestra algún tipo de enrutamiento a la LAN2?
La API *Información ROUTING* muestra las redes a las que está directamente conectada el agente, es decir a las redes 192.168.2.0/24 y 10.1.1.0/24 y las direcciones IP de las interfaces. En el resultado no se muestra ningún tipo de enrutamiento a la red de la LAN2.

14. ¿Qué resultado muestra esta API?
Al ejecutar la API *Crear ruta estática,* el código de respuesta ha sido la 204 No Content, es decir que el método se ha implementado correctamente.

15. Enviar el API Información ROUTING, ¿cuál es el nuevo resultado?
Aparte del resultado del literal 13 esta muestra la ruta estática hacia la red 192.168.3.0/24, creada con el API *Crear ruta estática.* En la Figura 18 se puede observar parte del resultado obtenido.

Figura 18. Respuesta a la API Información ROUTING con la ruta estática creada

16. En el agente, correr el comando show ip route, ¿cuál es el resultado?
Este comando nos indica la tabla de enrutamiento que tiene el agente, de acuerdo con la Figura 19. se confirma que la ruta estática ha sido creada con éxito en el router.

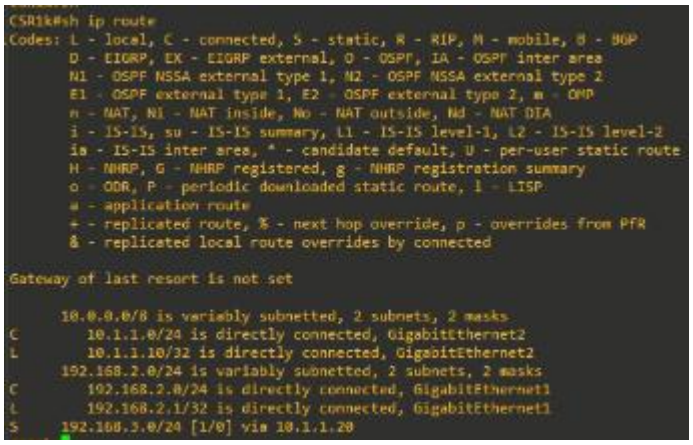

Figura 19. Información de la tabla de enrutamiento del agente

17. Realizar un ping desde el administrador a la PC1, mostrar el resultado.
Al ya tener una ruta estática desde la LAN1 hacia la LAN2, ya es posible tener comunicación entre los dos hosts de la topología, como se muestra en la Figura 20.

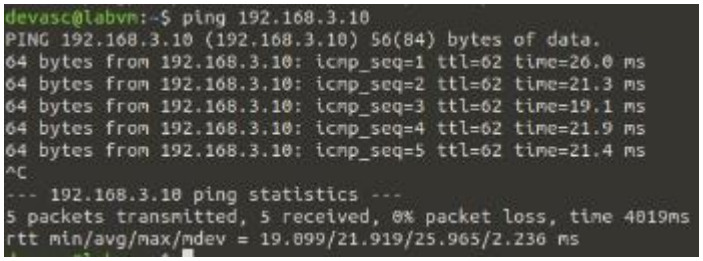

Figura 20. Comunicación entre el administrador y la PC1

18. Realizar un análisis de la información de esta API.
Según [1], el API *Estadísticas RESTCONF* muestra la información del monitoreo del protocolo RESTCONF conteniendo una lista de URI de capacidad de protocolo, además tiene un contenedor que representa los flujos de eventos de notificación admitidos por el servidor.

19. Realizar un análisis de la información de esta API.
Según [2], el API *Estadísticas INTERFACES* muestra la información de los nodos de datos para el estado operativo de las interfaces, dando la siguiente información de todas las interfaces del agente: nombre, tipo de interfaz según el RFC 2863, estado administrativo, estado de operación, fecha del último cambio realizado, dirección física, un estimado del ancho de banda, y las estadísticas que contienen: la discontinuidad de la interfaz y la entrada y salida de paquetes en octetos, de paquetes unicast, broadcast y multicast, de paquetes descartados, con errores o de protocolos desconocidos. En la Figura 21 se puede observar el estado de la interfaz GigabitEthernet1.

Figura 21. Estado y estadísticas de la interfaz Gi1

4 CONCLUSIONES

En comparación con la herramienta anteriormente estudiada, cURL, POSTMAN tiene mayores ventajas, opciones más amigables con el usuario, mejores métodos para realizar las peticiones al servidor, espacio para códigos, visualización de los códigos de respuesta, etc. por lo que, es eficaz para gestionar la red mediante el protocolo RESTCONF.

Mediante el uso de POSTMAN es más fácil tener información de las interfaces y su configuración, en esta práctica se han configurado una interfaz existente y creado otra, la interfaz Loopback, exitosamente. Además, se ha creado la ruta estática faltante para tener conectividad entre todas las redes de la topología y conocer la tabla de enrutamiento del agente.

El protocolo RESTCONF, mediante el módulo YANG de la IETF, puede revisar las estadísticas que nos da, tanto el protocolo como las interfaces. Esta información es muy útil para gestionar la red.

5 RECOMENDACIONES

Estudiar los otros modelos YANG que puede soportar el agente, para tener mayores ventajas de configuración, así como otros tipos de información.

Para la gestión de las redes mediante RESTCONF es recomendable tener herramientas eficaces y que permitan al ingeniero tener ideas claras del estado de la red, por lo que, es mejor el uso de POSTMAN y el formato JSON.

6 REFERENCIAS

[1] Apoorvashastry “ietf-restconf-monitoring.yang.” Internet: https://github.com/YangModels/yang/blob/main/vendor/cisco/xe/1731/ietf-restconf-monitoring.yang, Aug. 4, 2020 [Jul. 7, 2022]

[2] Apoorvashastry “ietf-interfaces.yang” Internet: https://github.com/YangModels/yang/blob/main/vendor/cisco/xe/1731/ietf-interfaces.yang, Aug. 4, 2020 [Jul. 7, 2022]

PRÁCTICA N° 5

1 TEMA

GESTIÓN RESTCONF CON PYTHON

2 OBJETIVOS

2.1 Analizar la herramienta Python para el uso del protocolo RESTCONF.

2.2 Configurar interfaces y ruta estática mediante RESTCONF.

2.3 Diagnosticar la red mediante RESTCONF.

3 INFORME

3.3 Proporcione la configuración construida en laboratorio con una explicación adecuada de los resultados que se muestran.

Para la primera parte de la práctica se ha implementado la topología dada en la Figura 1, en esta red se puede observar la conexión entre dos redes autónomas, una la red 192.168.2.0/24 y la 192.168.3.0/24. Los routers están dentro de la red 10.1.1.0/24. La configuración de los routers se encuentra en la Figura 2 y Figura 3.

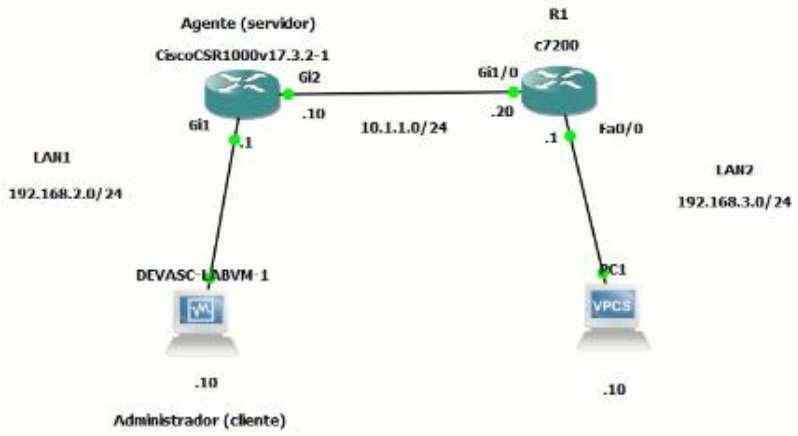

Figura 1. Topología de red implementada

```
!
hostname R1
!
interface FastEthernet0/0
 ip address 192.168.3.1 255.255.255.0
 duplex half
!
interface GigabitEthernet1/0
 ip address 10.1.1.20 255.255.255.0
 negotiation auto
!
end
```

Figura 2. Configuración router R1

```
!
hostname CSR1k
!
ip domain name labo-restconf.com
!
username admin privilege 15 password 0 admin
!
interface GigabitEthernet1
 ip address 192.168.2.1 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
ip forward-protocol nd
ip http server
ip http authentication local
ip http secure-server
!
restconf
end
```

Figura 3. Configuración agente CSR1k

En la Figura 4 se muestra la configuración de las direcciones IP del administrador y de la PC1.

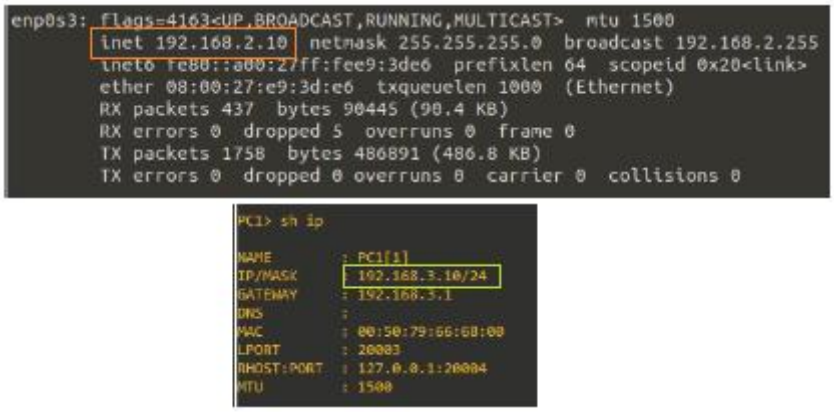

Figura 4. Direcciones IP de los hosts

Dado que, en la topología, en la LAN 1 solo se ha configurado la interfaz GigabitEthert1, el host administrador solo tiene conectividad a esa interfaz y a internet, como se muestra en la Figura 5.

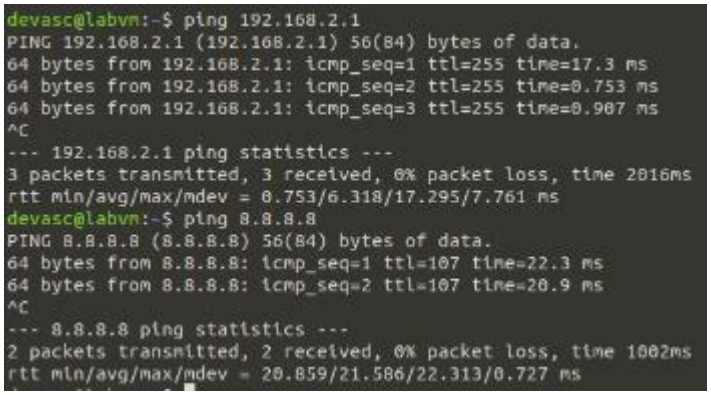

Figura 5. Conectividad del host de la LAN1

En cambio, dado los requerimientos en la Hoja Guía 5, el host PC1, tiene conectividad a las direcciones IP de las dos interfaces configuradas, como se observa en la Figura 6.

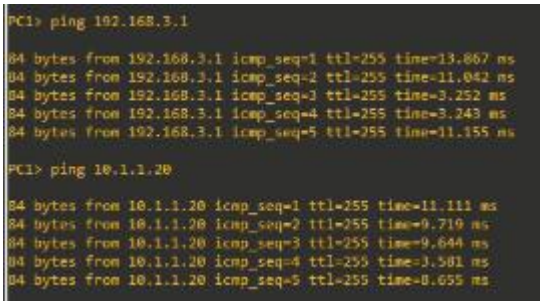

Figura 6. Conectividad del host de la LAN2

Dado que, en la LAN 1, en el agente se ha configurado RESTCONF, se hace la prueba de conectividad con el servidor mediante cURL, ver la Figura 7.

```
devasc@labvm:~$ curl -k https://192.168.2.1:443/restconf/ -u admin:admin
<restconf xmlns="urn:ietf:params:xml:ns:yang:ietf-restconf">
  <data/>
  <operations/>
  <yang-library-version>2016-06-21</yang-library-version>
</restconf>
```

Figura 7. Envío de solicitud cURL al servidor RESTCONF

Ahora se va a trabajar con Python, por lo que, en el administrador se ejecuta el programa Visual Studio Code, en la carpeta restconf-labs, como se observa en la Figura 8.

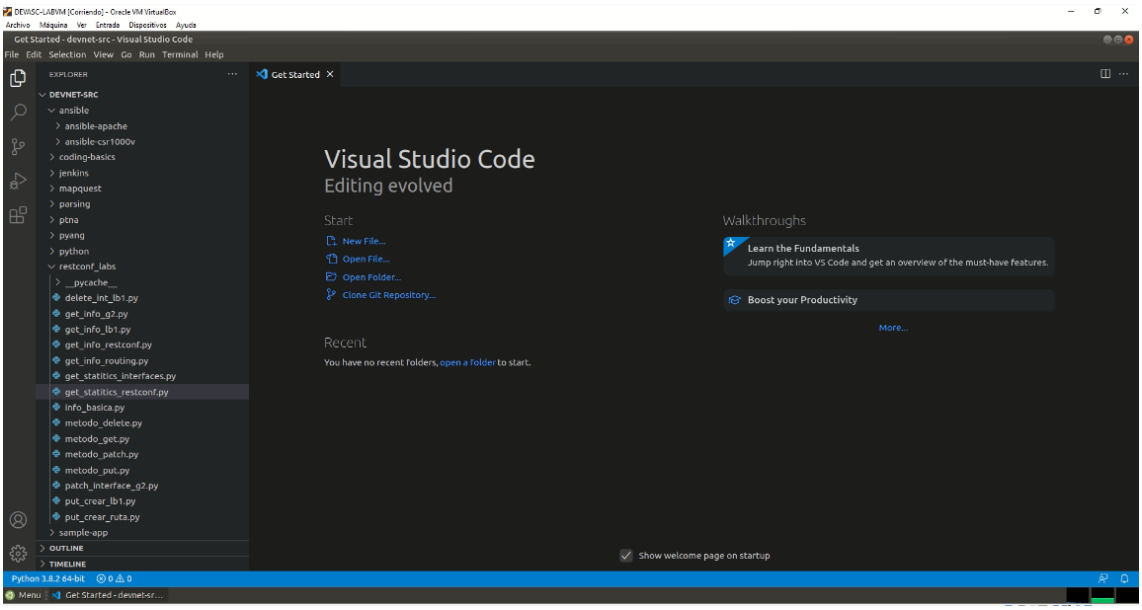

Figura 8. Visual Studio Code en el administrador

A continuación, se crea el código info_basica.py, que permite tener las configuraciones básicas para deshabilitar el certificado SSL, crear una sesión mediante una autenticación HTTP básica, tener las variables seteadas de la URL, como la IP del servidor, puerto y protocolo a utilizar y tener el encabezado configurado para enviar y recibir solicitudes en formato JSON. La configuración del script se puede ver en la Figura 9.

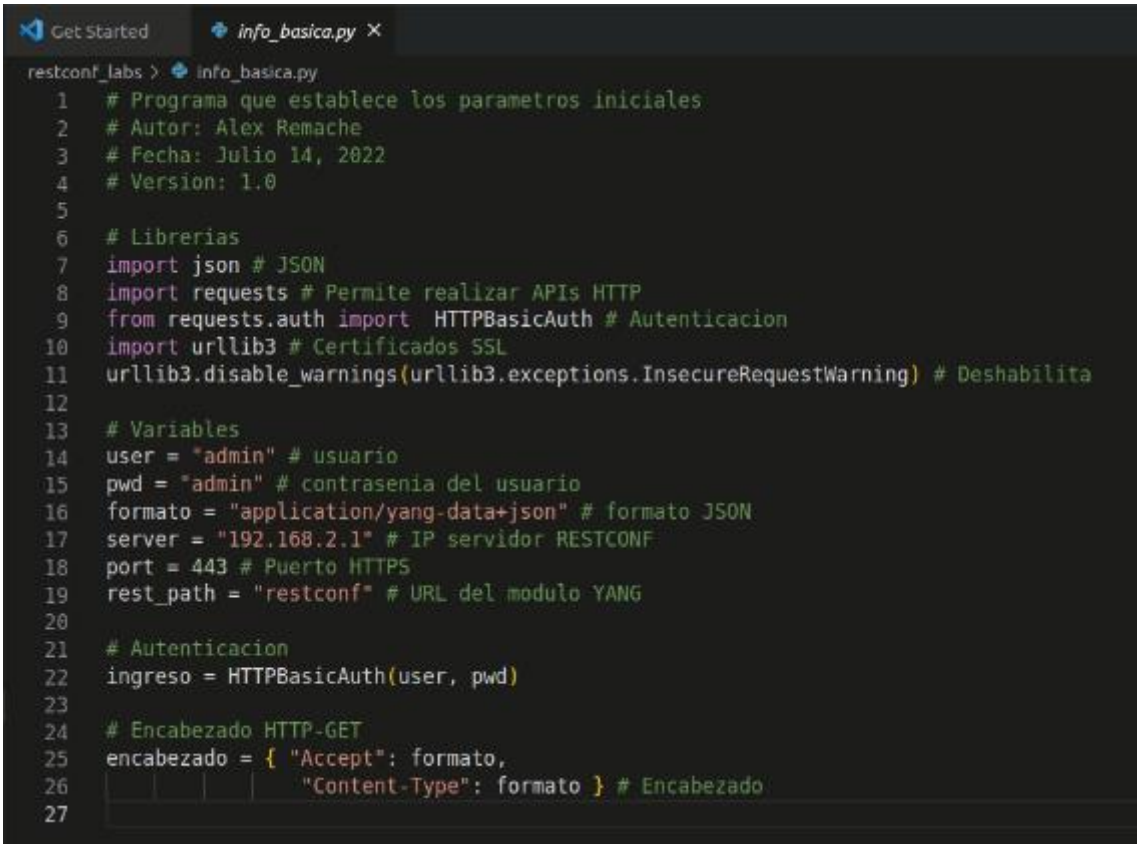

Figura 9. Script info_basica.py en el administrador

Además, para tener mayor funcionalidad en los scripts, se crearon cuatro funciones de los métodos HTTP: GET, PUT, PATCH y DELETE, tal como se pide en la Hoja Guía 5, y se muestra en las siguientes Figuras.

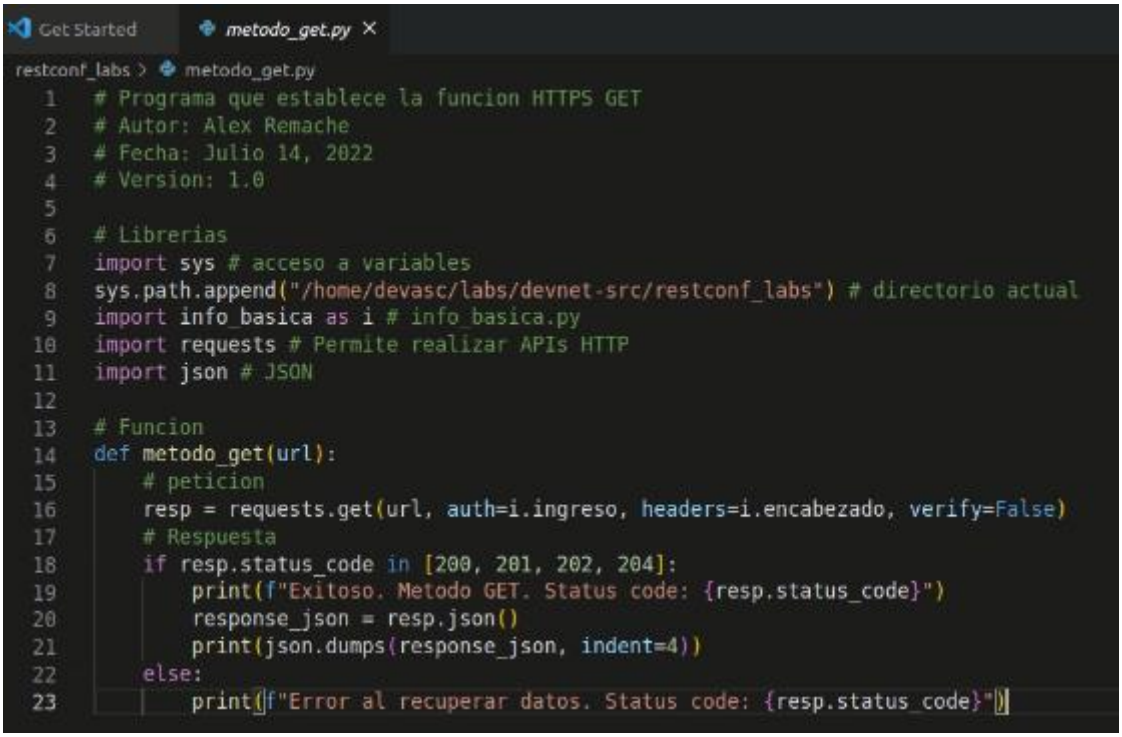

Figura 10. Script método_get.py en el administrador

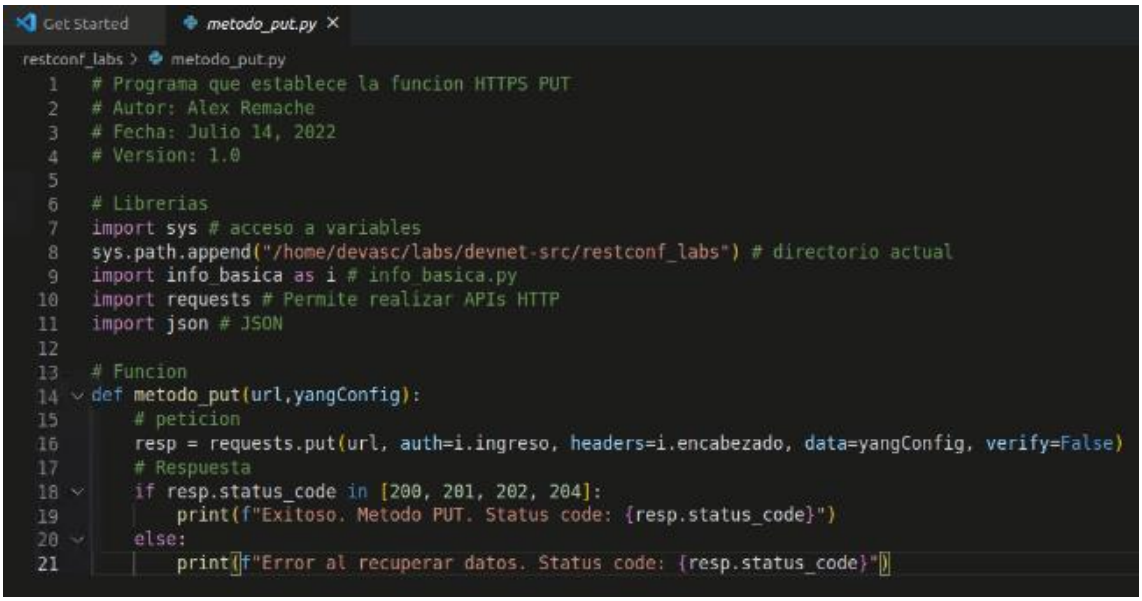

Figura 11. Script método_put.py en el administrador

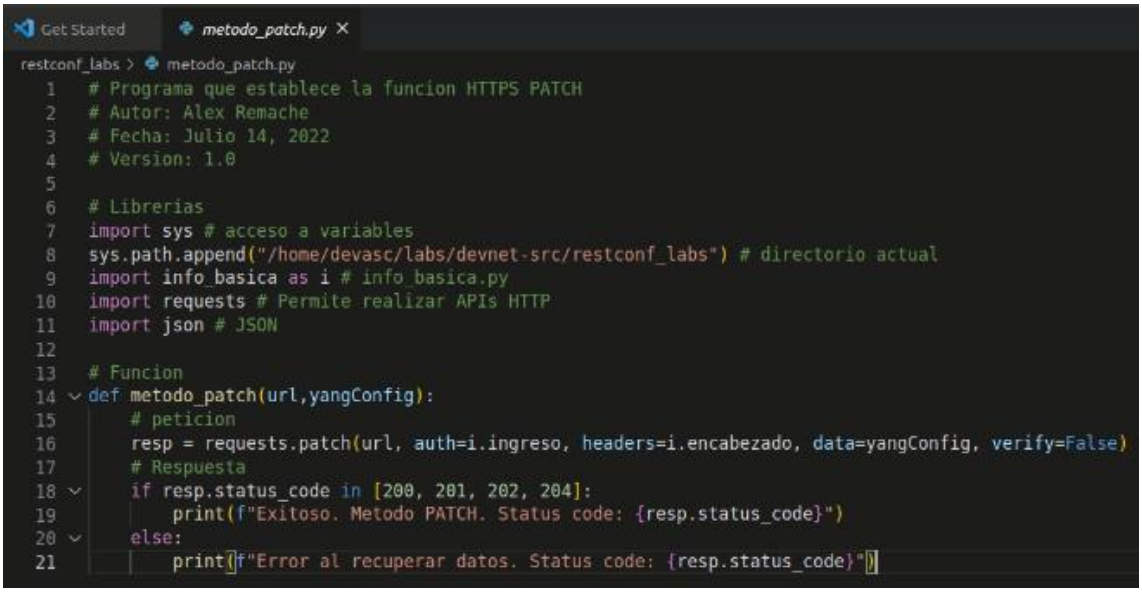

Figura 12. Script método_patch.py en el administrador

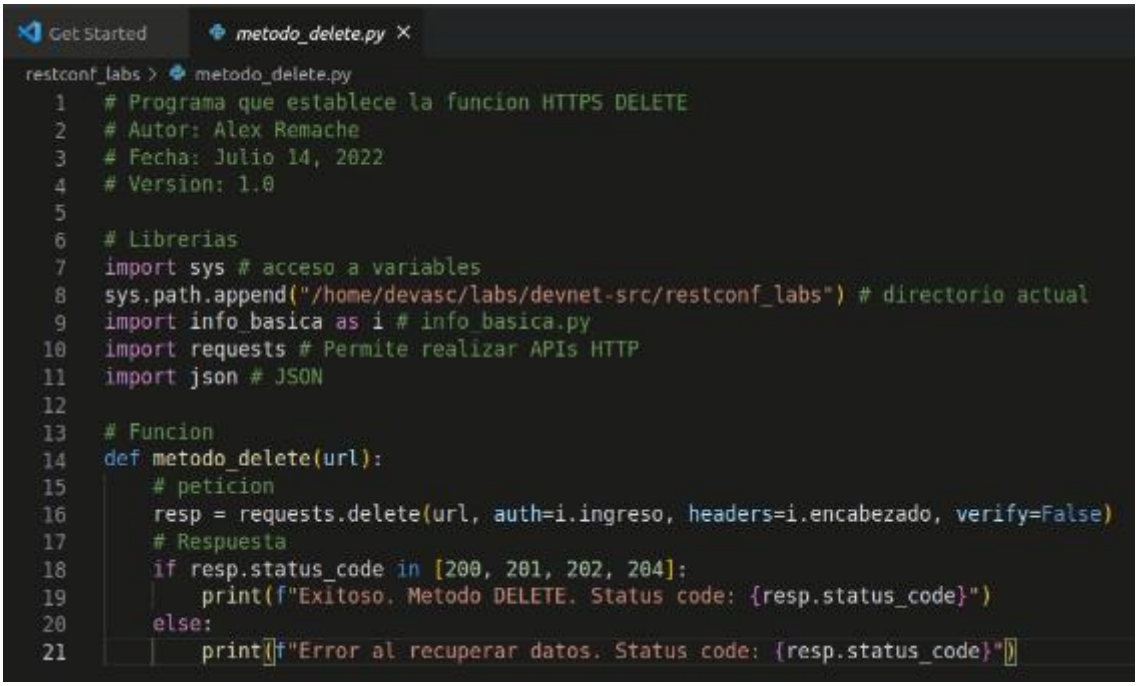

Figura 13. Script método_delete.py en el administrador

Como se puede observar, según sea el método a utilizar, la función tendrá una o dos variables de entrada, y dependiendo el tipo de respuesta, se mostrará el código de estado y/o el cuerpo del mensaje de respuesta. La utilización de estas funciones nos ayudará a crear múltiples mensajes de solicitud, facilitando su ejecución.

Utilizando los scripts ya creados, se procede a crear el código `get_info_restconf.py`, esto es para conocer acerca del establecimiento de la sesión RESTCONF entre el cliente y el servidor, como se observa en la Figura 14.

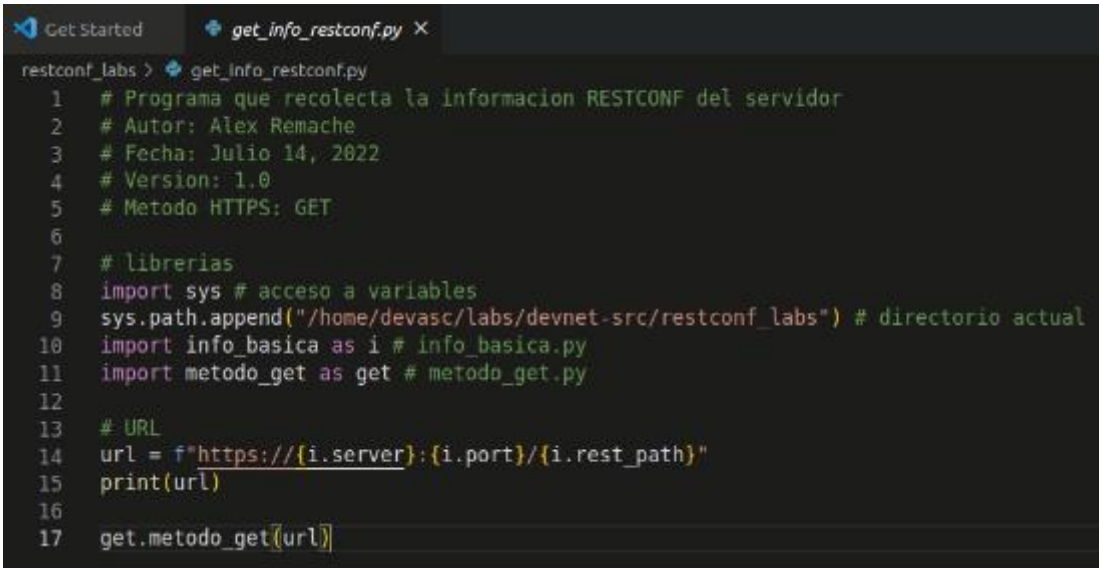

Figura 14. Script get_info_restconf.py en el administrador

3.4 Responder las preguntas dadas en el procedimiento.

1. ¿Qué hace el script get_info_restconf.py?

Como se observa en la Figura 14 el código usa las librerías sys, y así poder estar en el directorio donde se encuentran los códigos `info_basica.py` y `método_get.py` y utilizarlos en el script. En otra sección se recrea la URL a la que se le va a pedir la solicitud, se puede observar que toma las variables del script `info_basica.py`; con la URL creada se utiliza la función `get.metodo_get`, para tener una respuesta a la solicitud enviada.

2. Mostrar el resultado de su ejecución y comentarlo.
En la Figura 15. se observa que al ejecutar el código `get_info_restconf.py`, se tiene el resultado que muestra el código de estado, método utilizado y la información del protocolo creado, esto confirma que los mensajes de solicitud y respuesta entre el servidor y el cliente se da de forma exitosa.

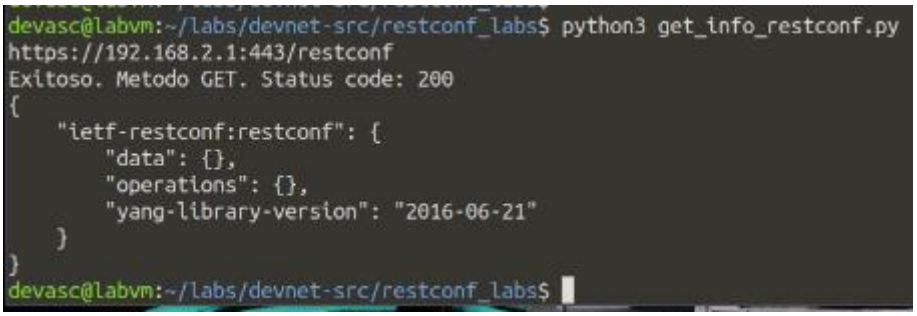

Figura 15. Resultado de get_info_restconf.py en el administrador

3. ¿Qué resultado muestra el código get_info_g2.py?, ¿la interfaz Gi2, que información tiene?
Se puede observar, que la interfaz Gi2 no está levantada ni configurada. El código de respuesta es 200 OK.

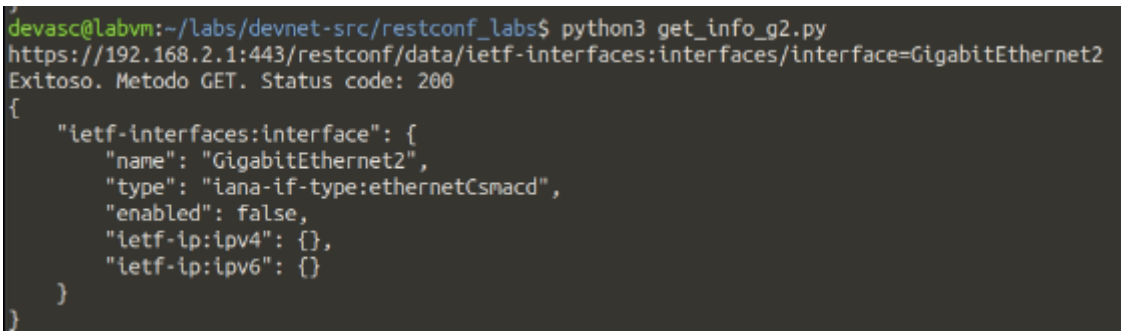

Figura 16. Resultado de get_info_g2.py en el administrador

4. ¿Qué resultado muestra este código?
La respuesta a la ejecución del código `patch_interface_g2.py` ha sido un 204, esto quiere decir que se ha configurado de manera exitosa a la interfaz Gi2, pero no se tiene nada que devolver como información de respuesta, por lo que, se ha pedido la información actualizada de la URL con el método GET, esto se puede observar en la Figura 17.

```
devasc@labvm:~/labs/devnet-src/restconf_labs$ python3 patch_interface_g2.py
https://192.168.2.1:443/restconf/data/ietf-interfaces:interfaces/interface=GigabitEthernet2
Exitoso. Metodo PATCH. Status code: 204
Exitoso. Metodo GET. Status code: 200
{
    "ietf-interfaces:interface": {
        "name": "GigabitEthernet2",
        "description": "To R1",
        "type": "iana-if-type:ethernetCsmacd",
        "enabled": true,
        "ietf-ip:ipv4": {
            "address": [
                {
                    "ip": "10.1.1.10",
                    "netmask": "255.255.255.0"
                }
            ]
        },
        "ietf-ip:ipv6": {}
    }
}
```

Figura 17. Resultado de patch_interface_g2.py en el administrador

5. En el agente, correr el comando para ver el estado de las interfaces ¿cuál es el resultado?

Al ser configurada la interfaz Gi2, desde el administrador a través de RESTCONF, se tiene que la interfaz ha sido levantada y configurada, como se muestra en la Figura 18.

```
CSR1k#show ip interface brief
Interface              IP-Address      OK? Method Status                Protocol
GigabitEthernet1       192.168.2.1     YES manual up                    up
GigabitEthernet2       10.1.1.10       YES other  up                    up
GigabitEthernet3       unassigned      YES unset  administratively down down
GigabitEthernet4       unassigned      YES unset  administratively down down
CSR1k#
```

Figura 18. Información de las interfaces del agente

6. Mostrar el código. ¿Qué resultado muestra la ejecución de este código?

Se ha creado el código get_info_lb1.py, como se muestra en la Figura 19, se ha utilizado la librería `sys`, e importado los códigos `info_basica` y `metodo_get`, creando la URL específica para la interfaz Loopback 1, y utilizándolo en la función get.

Get Started | get_info_lb1.py

restconf_labs > get_info_lb1.py

```
1  # Programa que recolecta la informacion de la interfaz
2  # LoopBack1 del agente
3  # Autor: Alex Remache
4  # Fecha: Julio 14, 2022
5  # Version: 1.0
6  # Metodo HTTPS: GET
7
8  # Librerias
9  import sys # acceso a variables
10 sys.path.append("/home/devasc/labs/devnet-src/restconf_labs") # directorio actual
11 import info_basica as i # info_basica.py
12 import metodo_get as get # metodo_get.py
13
14 # URL
15 url = f"https://{i.server}:{i.port}/{i.rest_path}/data/ietf-interfaces:interfaces/interface=Loopback1"
16 print(url)
17
18 get.metodo_get(url)
19
```

Figura 19. Script get_info_lb1.py en el administrador

Se ha obtenido el resultado de la Figura 20, como se muestra, el código no ha podido obtener datos de esa URL, dándonos un código de estado 404, lo que nos dice que el recurso solicitado no se encontró en el servidor. Esto tiene lógica, ya que no existe, al momento, ninguna interfaz loopback creada, como se observa en la Figura 18.

```
devasc@labvm:~/labs/devnet-src/restconf_labs$ python3 get_info_lb1.py
https://192.168.2.1:443/restconf/data/ietf-interfaces:interfaces/interface=Loopback1
Error al recuperar datos. Status code: 404
devasc@labvm:~/labs/devnet-src/restconf_labs$
```

Figura 20. Resultado de get_info_lb1.py en el administrador

7. ¿Qué resultado muestra la ejecución del Código 9?

El Código 9 es `put_crear_lb1.py` que tiene como objetivo crear y configurar la interfaz Loopback 1, dando el resultado de la Figura 21. que muestra el código de estado 201, esto indica que el recurso solicitado ha sido creado, y se muestra la configuración actual de la interfaz.

```
devasc@labvm:~/labs/devnet-src/restconf_labs$ python3 put_crear_lb1.py
https://192.168.2.1:443/restconf/data/ietf-interfaces:interfaces/interface=Loopback1
Exitoso. Metodo PUT. Status code: 201
Exitoso. Metodo GET. Status code: 200
{
    "ietf-interfaces:interface": {
        "name": "Loopback1",
        "description": "Loopback 1",
        "type": "iana-if-type:softwareLoopback",
        "enabled": true,
        "ietf-ip:ipv4": {
            "address": [
                {
                    "ip": "20.1.1.1",
                    "netmask": "255.255.255.0"
                }
            ]
        },
        "ietf-ip:ipv6": {}
    }
}
devasc@labvm:~/labs/devnet-src/restconf_labs$
```

Figura 21. Resultado de put_crear_lb1.py en el administrador

8. En el agente, correr el comando para ver el estado de las interfaces ¿cuál es el resultado?

Al ser creada y configurada la interfaz Loopback1, desde el administrador a través de RESTCONF, se tiene que la interfaz ha sido creada, levantada y configurada, como se muestra en la Figura 22.

```
CSR1k#sh ip int bri
Interface              IP-Address      OK? Method Status                Protocol
GigabitEthernet1       192.168.2.1     YES manual up                    up
GigabitEthernet2       10.1.1.10       YES other  up                    up
GigabitEthernet3       unassigned      YES unset  administratively down down
GigabitEthernet4       unassigned      YES unset  administratively down down
Loopback1              20.1.1.1        YES other  up                    up
```

Figura 22. Información de las interfaces del agente incluye LB1

9. Mostrar el código. ¿Qué resultado muestra la ejecución de este código?

Se ha creado el código `delete_int_lb1.py`, como se muestra en la Figura 23, se han utilizado la librería `sys`, e importado los códigos `info_basica`, `metodo_delete` y `metodo_get`, creando la URL específica para la interfaz Loopback 1, y utilizándola en la función get. Al ejecutarlo, se tiene la respuesta de la Figura 24, donde se puede observar que la eliminación de la interfaz ha sido exitosa, con un código de estado 204; además, al ya no tener la interfaz, la ejecución del método get nos da un código de estado 404, elemento no encontrado en el servidor.

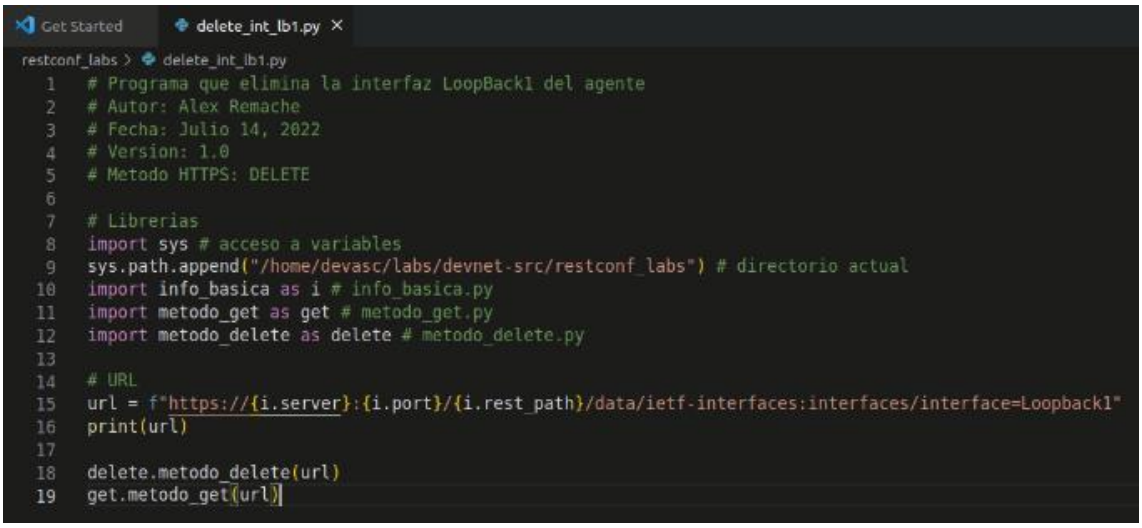

Figura 23. Script delete_int_lb1.py en el administrador

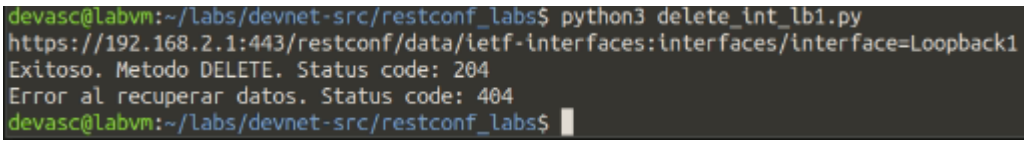

Figura 24. Resultado de delete_int_lb1.py en el administrador

10. ¿Qué resultado muestra la ejecución del Código 10?,¿el resultado muestra algún tipo de enrutamiento a la LAN2?

El Código 10 que es `get_info_routing.py` muestra las redes a las que está directamente conectada el agente, es decir a las redes 192.168.2.0/24 y 10.1.1.0/24 y las direcciones IP de las interfaces. En el resultado no se muestra ningún tipo de enrutamiento a la red de la LAN2.

11. ¿Qué resultado muestra el Código 11?

Al ejecutar el Código 11, `put_crear_ruta.py` el código de respuesta ha sido la 204, es decir que el método se ha implementado correctamente.

12. Ejecutar el Código10., ¿cuál es el nuevo resultado?

Aparte del resultado del literal 10. esta muestra la ruta estática hacia la red 192.168.3.0/24, creada con `put_crear_ruta.py`. En la Figura 25 se puede observar parte del resultado obtenido.

```
                {
                    "destination-prefix": "192.168.3.0/24",
                    "route-preference": 1,
                    "metric": 1,
                    "next-hop": {
                        "outgoing-interface": "",
                        "next-hop-address": "10.1.1.20"
                    },
                    "source-protocol": "ietf-routing:static",
                    "active": [
                        null
                    ],
                    "update-source": "0.0.0.0"
                }
            ]
        }
    },
```


Figura 25. Respuesta a get_info_routing.py con la ruta estática creada

13. En el agente, correr el comando show ip route, ¿cuál es el resultado?
Este comando nos indica la tabla de enrutamiento que tiene el agente, de acuerdo con la Figura 26. se confirma que la ruta estática ha sido creada con éxito en el router.

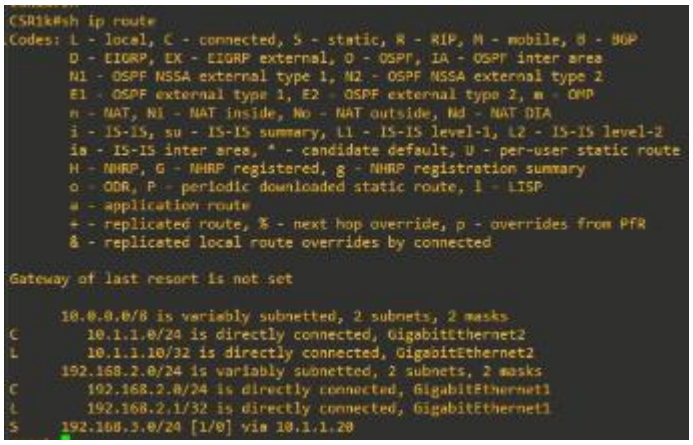

Figura 26. Información de la tabla de enrutamiento del agente

14. Realizar un ping desde el administrador a la PC1, mostrar el resultado.
Al ya tener una ruta estática desde la LAN1 hacia la LAN2, ya es posible tener comunicación entre los dos hosts de la topología, como se muestra en la Figura 27.

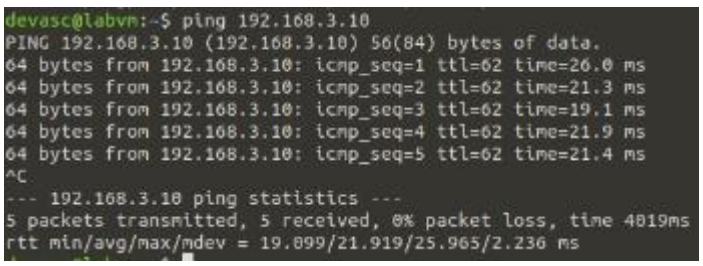

Figura 27. Comunicación entre el administrador y la PC1

15. Mostrar el código y realizar un análisis de la información de este código.
Se ha creado el código `get_statitics_restconf.py`, como se muestra en la Figura 28, se ha utilizado la librería `sys`, e importado los códigos `info_basica` y

`metodo_get`, creado la URL específica para conocer el estado del protocolo y utilizarlo en la función get. Al ejecutarlo, se tiene la respuesta de la Figura 29, que según [1], muestra la información del monitoreo del protocolo RESTCONF conteniendo una lista de URI de capacidad de protocolo, además tiene un contenedor que representa los flujos de eventos de notificación admitidos por el servidor.

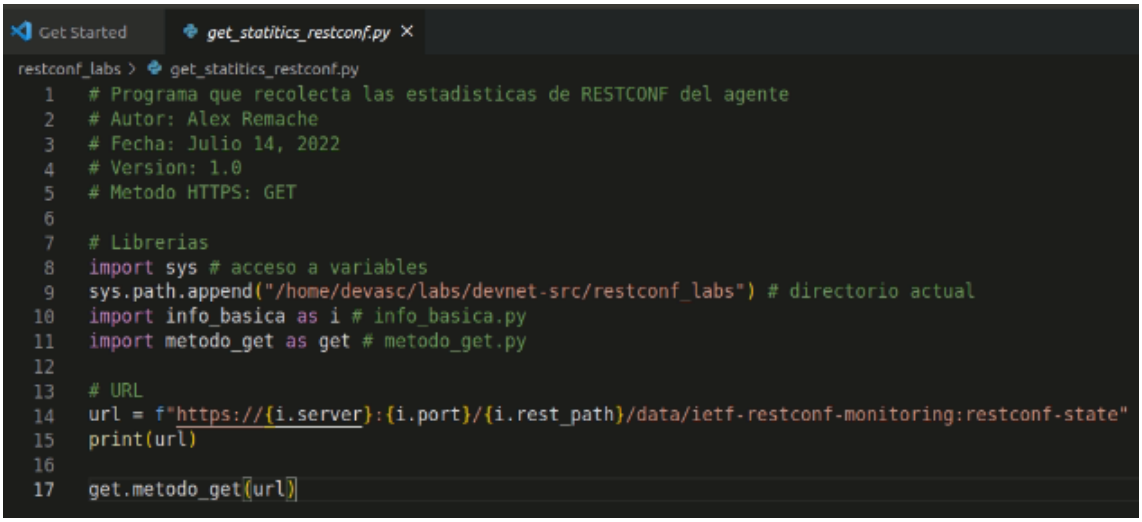

Figura 28. Script get_statitics_restconf.py en el administrador

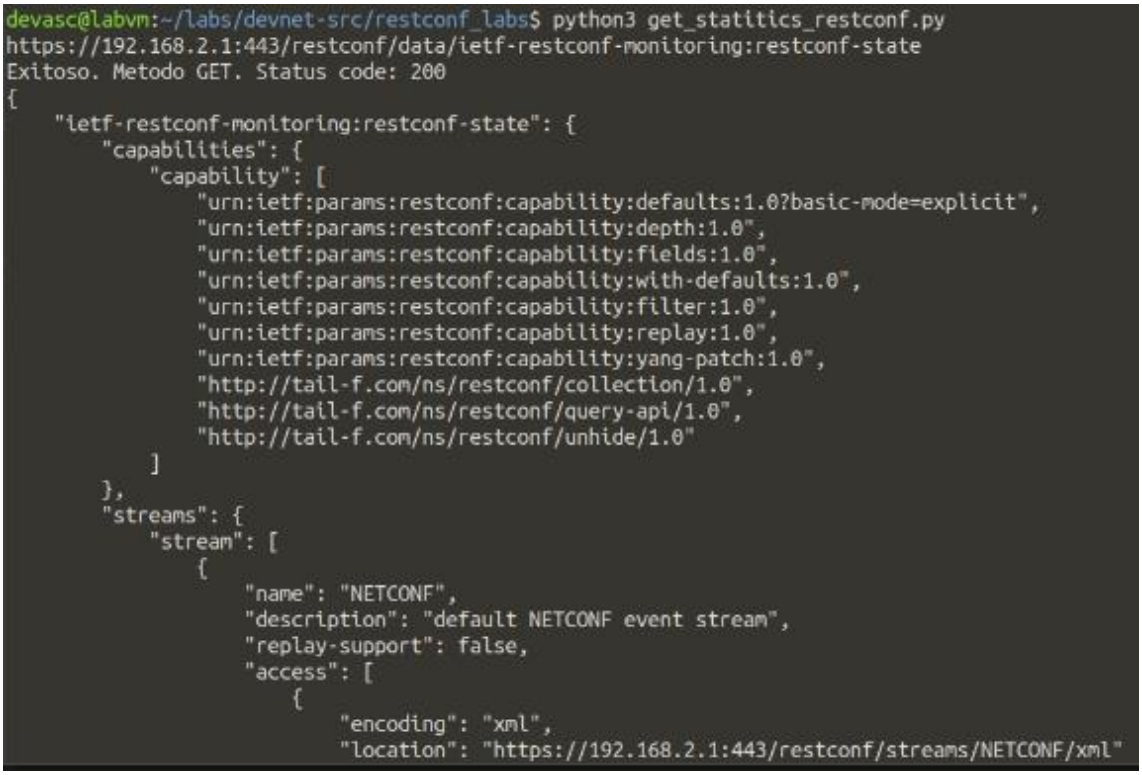

Figura 29. Resultado de get_statitics_restconf.py en el administrador

16. Mostrar el código y realizar un análisis de la información de este código.
Se ha creado el código `get_statitics_interfaces.py`, como se muestra en la Figura 30, se ha utilizado la librería `sys`, e importado los códigos `info_basica`

y `metodo_get`, creando la URL específica para conocer el estado de las interfaces y utilizándolo en la función get. Al ejecutarlo, según [2], muestra la información de los nodos de datos para el estado operativo de las interfaces, dando la siguiente información de todas las interfaces del agente: nombre, tipo de interfaz según el RFC 2863, estado administrativo, estado de operación, fecha del último cambio realizado, dirección física, un estimado del ancho de banda, y las estadísticas que contienen: la discontinuidad de la interfaz y la entrada y salida de paquetes en octetos, de paquetes unicast, broadcast y multicast, de paquetes descartados, con errores o de protocolos desconocidos. En la Figura 31. se puede observar el estado de la interfaz GigabitEthernet1.

Figura 30. Script get_statitics_interfaces.py en el administrador

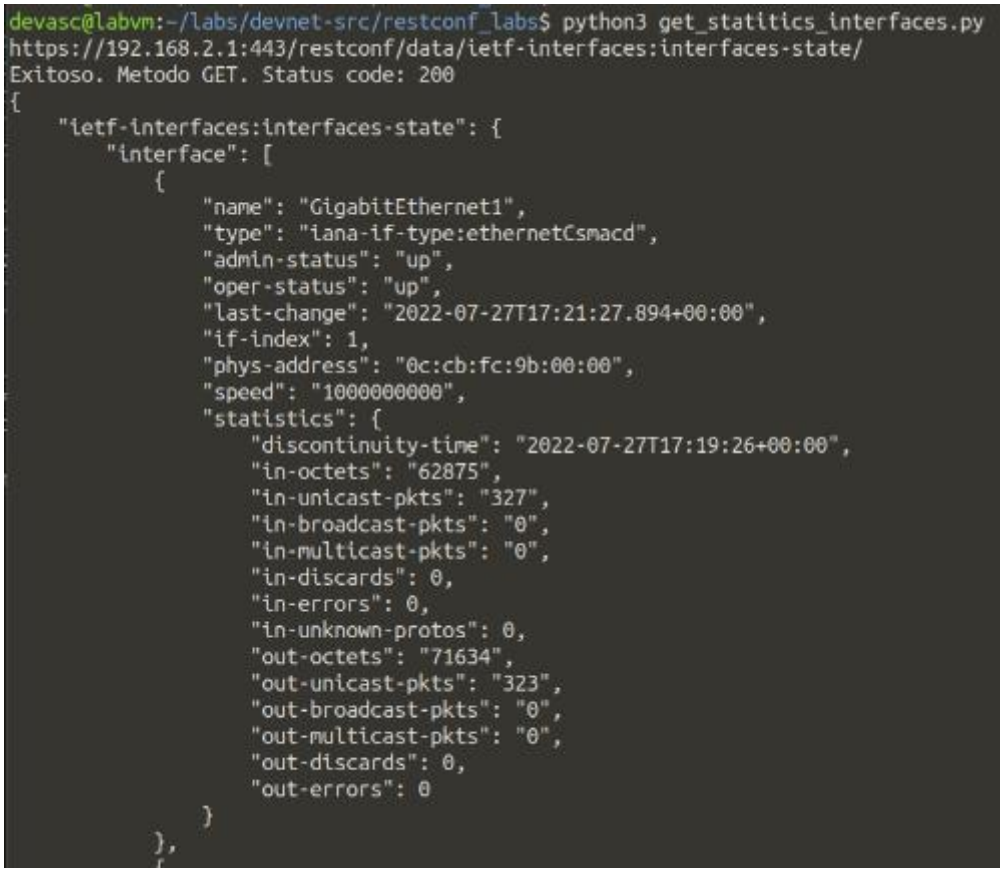

Figura 31. Resultado de get_statitics_interfaces.py en el administrador

4 CONCLUSIONES

La utilización de Python para la gestión de redes mediante el protocolo RESTCONF, se puede observar desde diferentes perspectivas, ya que es mucho mejor en comparación con cURL, pero dependiendo del administrador de red puede estar al mismo nivel de Postman. Al hacer uso de Python se tiene la ventaja de la programabilidad y automatización de los recursos de RESTCONF.

Mediante el uso de Python es más fácil tener información de las interfaces y su configuración, en esta práctica se han configurado una interfaz existente y creado otra, la interfaz Loopback, exitosamente. Además, se ha creado la ruta estática faltante para tener conectividad entre todas las redes de la topología y conocer la tabla de enrutamiento del agente.

5 RECOMENDACIONES

Para la gestión de las redes mediante RESTCONF es recomendable tener herramientas eficaces y que permitan al ingeniero tener ideas claras del estado de la red, por lo que, es recomendable el estudio de Python, sus librerías y variables.

6 REFERENCIAS

[1] Apoorvashastry “ietf-restconf-monitoring.yang.” Internet: https://github.com/YangModels/yang/blob/main/vendor/cisco/xe/1731/ietf-restconf-monitoring.yang, Aug. 4, 2020 [Jul. 13, 2022]

[2] Apoorvashastry “ietf-interfaces.yang” Internet: https://github.com/YangModels/yang/blob/main/vendor/cisco/xe/1731/ietf-interfaces.yang, Aug. 4, 2020 [Jul. 13, 2022]