NOMBRE: **Isidro Lara Lopez**
CARRERA: **INFRAESTRUCTURA DE REDES DIGITALES**
DOCENTE: **BARRON RODRIGUEZ GABRIEL**
NO. CONTROL: **1221100381**
FECHA: **14 DE DICIEMBRE DEL 2022**
GRUPO: **GIR0441**

Lab – RESTCONF with Postman

Objectives

Part 1: Setup HTTP Headers in Postman

Background / Scenario

RESTCONF is a RESTful API interface that provides a programmatic interface for accessing data defined in YANG device models.

In this lab, you will learn how to interact with the RESTCONF interface using the Postman application to retrieve the device's configuration, update and create new interfaces and retrieve operation data.

Required Resources

- Postman
- Access to a router with the IOS XE operating system version 16.6 or higher.

Instructions

Part 1: Setup HTTP headers and Authentication in Postman

In this part, you will setup the HTTP Headers in Postman to work with RESTCONF REST API interfaces.

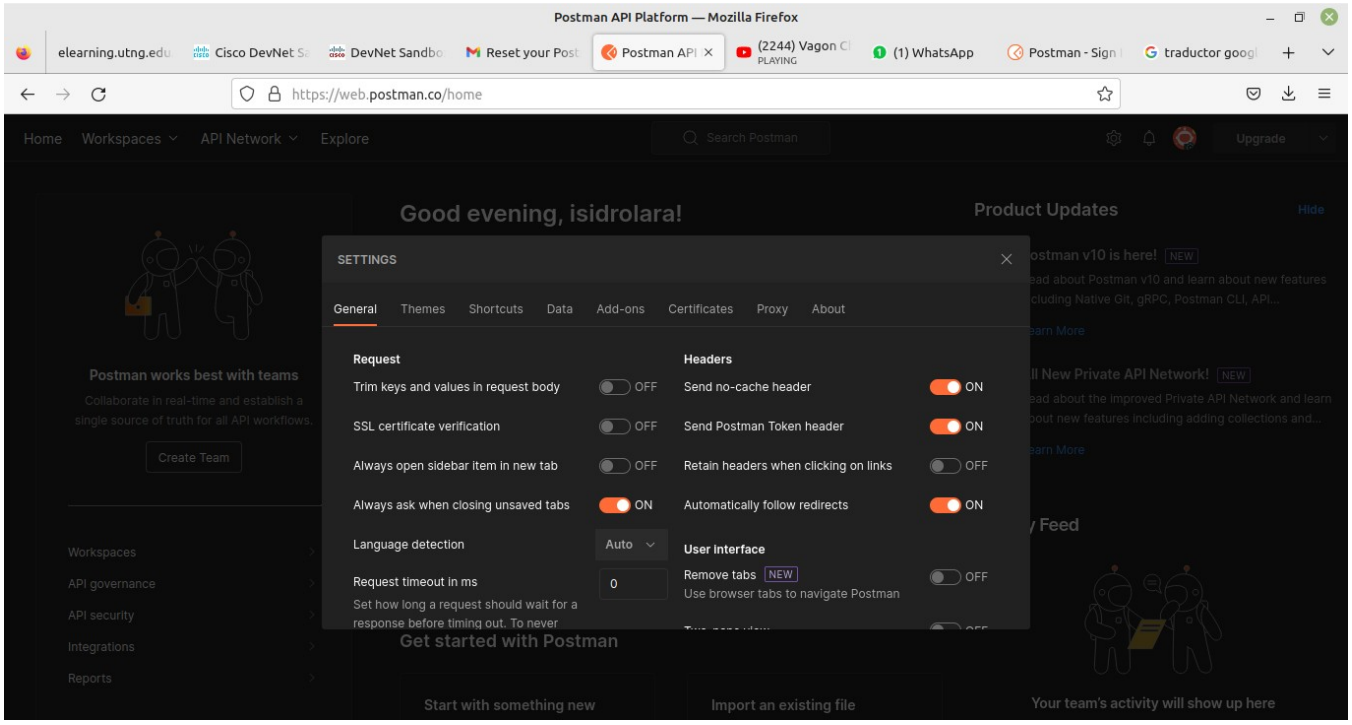
Step 1: Configure Postman.

Configure a Postman setting.

a. Open Postman and create a new tab if necessary.

b. Click **File > Settings**.

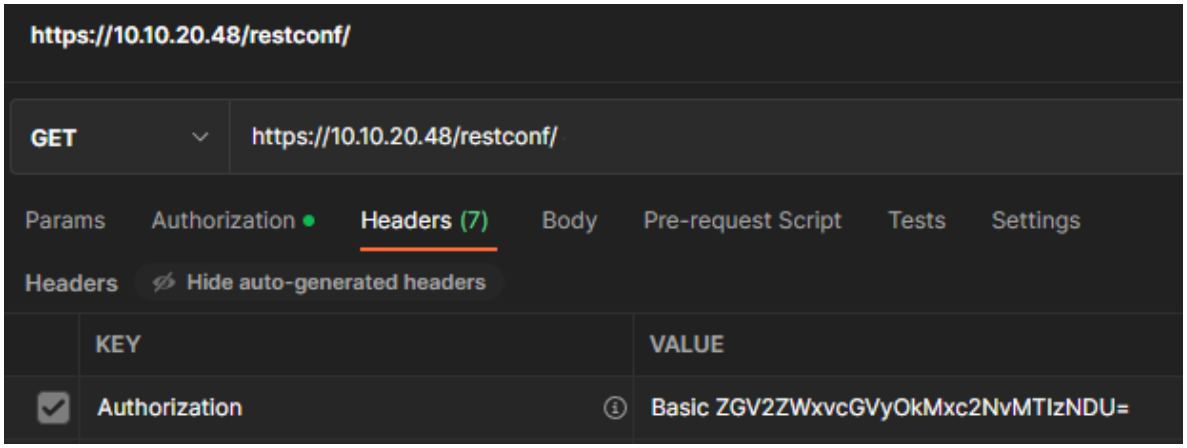
c. Under the **General** tab, set the **SSL certificate verification** to **OFF**. Close the **Settings** dialog box.



Step 2: Select the method and enter the required URL.

a. Next to the URL box, select the request method to be **GET**.

b. Enter the URL for API endpoint: https://192.168.56.101/restconf/ (adjust the IP address to match the router's current address)

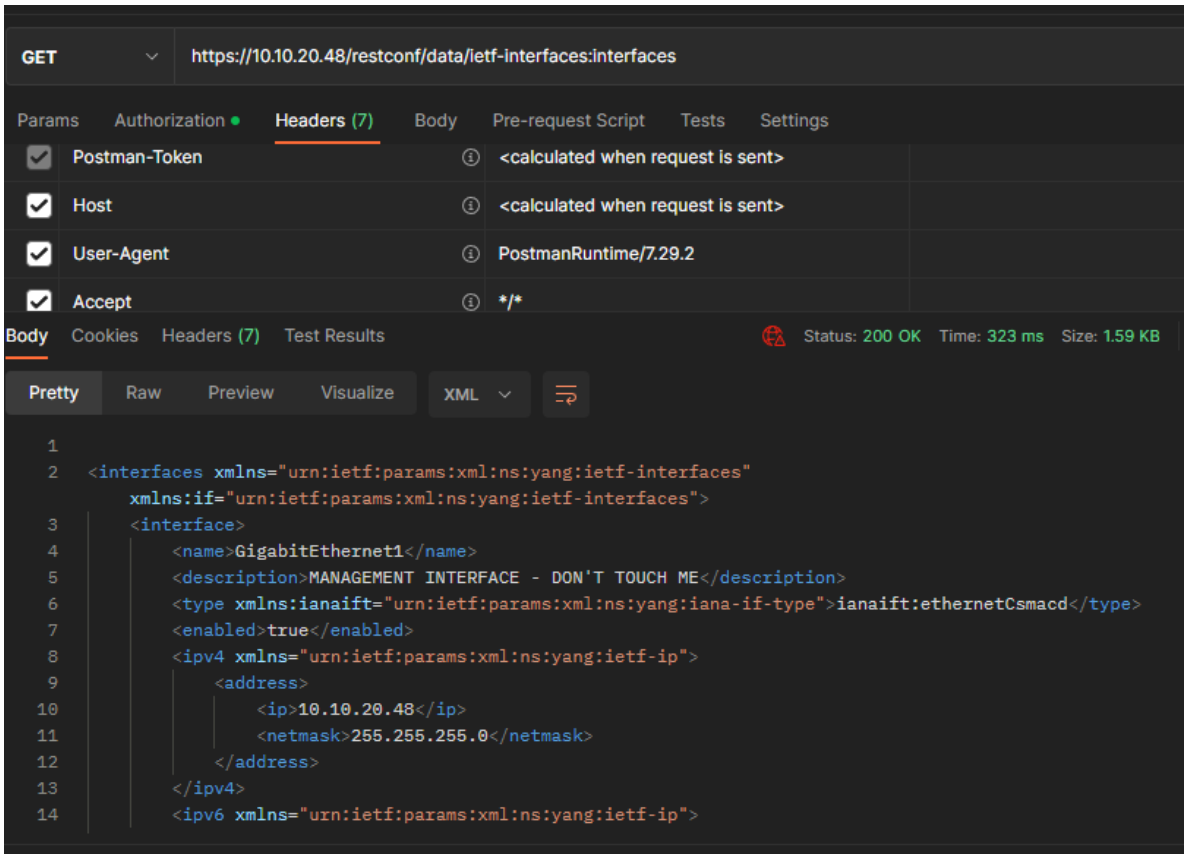


Step 3: Enter the authentication header information.

a. Below the URL input field, select **Authorization**.

b. Select the type **Basic Auth** and enter the username and password for authenticating to the RESTCONF API service:
`cisco`
`cisco123!`

c. Click on the **Preview Request** to add the Authentication header to the Headers.



Step 4: Enter the additional Headers.

a. Next, below the URL input field, select **Headers**.

b. Under **Headers**, click the **New key** field under the **Key** column and enter **Content-Type** to define what is the encoding of data that are sent with Postman in the REST API request.

c. Next again under **Headers**, click the **New key** field under the **Key** column and enter **Accept** to define in what encoding you wish to receive back the data replies from the REST API request.

d. In the value columns, enter `application/yang-data+json` when sending or receiving data that are in JSON encoding. When sending or receiving data in XML encoding (default), use `application/yang-data+xml`.

Step 5: Send the request.

a. Click **Send**.

b. After a brief delay, you should see the response appear below the request information. Scroll down, if necessary to view the response data.

c. When the **Accept** header was set to `application/yang-data+json` the response consists of JSON encoded data.

d. Uncheck the **Accept** header, or change it to `application/yang-data+xml` (default behavior when no Accept header is defined) and click on **Send**. Now the response consists of XML encoded data:

Note: If there is an error, check the Status of the request. Check the meaning of the status code. 200 means success. A 404 error may mean that the URL was entered incorrectly. A 401 or 403 error could indicate a problem with the authentication, so check that the credentials are entered correctly.

Part 2: Retrieve Interface Configuration

Step 1: Retrieve YANG data in XML.

a. Duplicate the existing REST API request in Postman by right clicking on the request and selecting Duplicate Tab to create a new request with prefilled existing Headers:

b. In the URL for the API endpoint, enter after the host and port portion: `/restconf/data/ietf-interfaces:interfaces` to retrieve data defined in the **ietf-interfaces** YANG model in the **interfaces** container.

c. Click **Send** to send the RESTCONF API request. With unchecked **Accept** header or the **Accept** header set to `application/yang-data+xml`, the result YANG data are returned in XML encoding.

d. Check the Accept header and set it to `application/yang-data+json` to receive YANG data formatted in JSON encoding.

Step 2: Retrieve information about a single interface

a. In the URL for the API endpoint, enter after the host and port portion: `/restconf/data/ietf-interfaces:interfaces/interface=Loopback1` to retrieve data for the **Loopback1** interface only.

b. Click **Send** …

```
<interface xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces"  xmlns:if="urn:ietf:params:xml:ns:yang:ietf-interfaces">
    <name>Loopback1</name>
    <description>LAb 2.2</description>
    <type xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">ianaift:softwareLoopback</type>
    <enabled>true</enabled>
    <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
        <address>
            <ip>192.168.20.105</ip>
            <netmask>255.255.255.0</netmask>
        </address>
    </ipv4>
    <ipv6 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
  </ipv6>
</interface>
```

Step 3: Retrieve operation data

a. Change the RESTCONF API Endpoint to retrieve operation and statistical data about the interfaces. You can use the "pyang" tool to discover what container in the **ietf-interfaces** YANG model includes the operational and stats data.

b. What is the new URL?

Part 3: Update Interface Configuration

a. Duplicate the existing REST API request in Postman by right clicking on the request and selecting Duplicate Tab to create a new request with prefilled existing Headers.

b. In the URL for the API endpoint, change the method from **GET** to **PUT** and enter after the host and port portion: `/restconf/data/ietf-interfaces:interfaces/interface=Loopback99` to create a new interface **Loopback99**.

```
<interface xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces"  xmlns:if="urn:ietf:params:xml:ns:yang:ietf-interfaces">
    <name>Loopback99</name>
    <description>LAB 2.5 Python</description>
    <type xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">ianaift:softwareLoopback</type>
    <enabled>true</enabled>
    <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
        <address>
            <ip>192.168.10.101</ip>
            <netmask>255.255.255.0</netmask>
        </address>
    </ipv4>
    <ipv6 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
  </ipv6>
</interface>
```

c. In the Body part of the request, enter the YANG data you want to apply to the device's configuration (copy, paste and customize the JSON data from the previous GET request on Loopback1 – change the interface name and IP address).

d. Click **Send** to create a new interface Loopback99 with the IP address 99.99.99.99/24.

e. In the Output (scroll down), you should see the HTTP Error code **201 Created** indicating that the new interface has been created:

f. Verify using the IOS CLI that the new Loopback99 interface has been created (sh ip int brief).

g. Update the RESTCONF API request to delete the interface **Loopback99**.

h. What changes were needed to be made in the RESTCONF request to delete the interface?

Se creo la interfaz loopBack99 con la api

El objetivo de utilizer postman en este laboratorio fue crear la solicitud de una manera en un entorno mas gráfico para que fuera mas facil RESTCONF es un protocol basado en http, un protocolo y un mecanismo para configuraciones REST

¿Qué es postman?

Postman es una aplicación que nos permite testear APIs a través de una interfaz gráfica de usuario. Entre las ventajas que tiene Postman encontramos la capacidad de crear colecciones y distintos ambientes de pruebas. Postman es una herramienta fácil de usar que nos ayuda a optimizar el tiempo de ejecución de pruebas.

Es una de las herramientas más populares para hacer pruebas eficaces a nuestra API con postman podremos hacer uso de los métodos GET, POST, PUT y PATCH en este laboratorio veremos los métodos GET y PUT.

Postman sirve para diversas labores en las cuales destacaremos en esta posibilidad las próximas:

•Testear colecciones o catálogos de APIs como para Frontend como para Backend.

•Acomodar en carpetas, funciones y módulos los servicios web.

• Posibilita gestionar el periodo de vida (conceptualización y definición, desarrollo, monitoreo y mantenimiento) de nuestra API.

•Producir documentación de nuestras propias APIs.

•Laborar con ámbitos (calidad, desarrollo, producción) y de esta manera es viable compartir por medio de un ámbito cloud la información con lo demás del equipo involucrado en el desarrollo.

Postman cuenta con una secuencia de procedimientos que nos permiten tomar acción frente a nuestras propias pedidos:

GET: Obtener información

POST: Añadir información

PUT: Suplir la información

PATCH: Actualizar alguna información

DELETE: Borrar información

¿Qué se puede hacer en el modelo YANG?

RESTCONF usa data estructurada (XML, JSON) y YANG que provee API's estilo REST, habilitando acceso programable a equipos. Comandos HTTP como GET, POTS,PUT,PATCH y DELETE son redireccionados a una RESTCONF API para acceder a los recursos de data representados por modelos YANG.

¿Para que sirve `application/yang-data+xml` y `application/yang-data+json`?

Debe especificar las cabeceras Content-Type y Accept correctamente para que coincidan con el tipo de contenido que proporcione y se acepte como respuesta. Dentro de estas se debe colocar **`application/yang-data+xml`** y el header funciona como metadatos adicionales que se mandan a la API para ayudar al servidor a comprender qué tipo de solicitud se está mandando.

¿Que significa `ietf-interfaces:interfaces`?

/interface al final no es estrictamente necesario, pero al ser el único elemento dentro del contenedor (ver output de pyang) podemos usarlo para reducir un nivel la profundidad de la respuesta.

XML es el acrónimo de Extensible Markup Language, o sea, es un lenguaje de marcado que define un grupo de normas para la codificación de documentos.
El diseño XML se concentra en la simplicidad, la generalidad y la facilidad de uso y, por consiguiente, se usa para diversos servicios web. Tanto es de esta forma que hay sistemas con el propósito de beneficiar en la definición de idiomas basados en XML, así como APIs que ayudan en el procesamiento de datos XML - que no tienen que confundirse con HTML.
YANG se convirtió en un lenguaje de modelado de datos por cierto. Es un lenguaje con base en estándares que se usa para producir demandas de configuración de dispositivos o demandas de datos operativos (como tienen la posibilidad de ser los comandos show). Tiene un formato estructurado parecido a un programa de PC que es legible por humanos. Hay algunas aplicaciones accesibles que tienen la posibilidad de realizar en una plataforma de gestión centralizada para producir estas demandas de configuración y datos operativos.
Lenguaje de modelado determinado en el RFC 6020.