Universidad Tecnológica del Norte de Guanajuato

Organismo Público Descentralizado del Gobierno del Estado

"Educación y progreso para la vida"

Tecnologías de la Información y Comunicación

Programa educativo: TSU en Infraestructura de Redes Digitales

Área académica: Programación de Redes

Asignatura:

Unidad 3

Grupo: GIR0441

2.5 Lab - RESTCONF with Python.docx

Alumna Gómez Luna Cinthia Valeria:

Docente: Gabriel Barrón Rodríguez

Dolores Hidalgo, C.I.N., Gto., jueves 24 de noviembre de 2022

Introducción

En el presente documento se podrán observar los pasas que se realizaron para lograr concluir el laboratorio 2.5 así mismo unas preguntas las cuales nos ayudan a conocer mas del tema para que así podamos entender un poco más, hablaremos del modelo restconf para conocer mas de el se dice que es un protocolo basado en http y se puede definir en el RFC 8040 a continuación se presentaran las preguntas. ¿Qué es el modelo RESTCONF? se define en el RFC 8040 este es un protocolo y un mecanismo para configuraciones REST, Similar a NETCONF, este usa una base de modelos y comandos definidos por el protocolo NETCONF, Encapsulando esta información en mensajes HTTP. ¿Como lo puedo implementar con Python? Se dice que para poder trabajar junto con Python es necesario tener el api que se generó en el documento 2.4 en postman

En el siguiente apartado se deberán poner los siguientes comandos de una manera precisa para lograr su funcionalidad se deberán importar el json así como sus requerimientos en la siguiente imagen se lograran ver los pasos realizados se deberán agregar los siguientes comandos.

```
import json

import requests

requests.packages.urllib3.disable_warnings()


api_url = "https://10.10.20.48/restconf/data/ietf-interfaces:interfaces"


headers = { "Accept": "application/yang-data+json",

            "Content-type":"application/yang-data+json"

          }

basicauth = ("developer", "C1sco12345")


resp = requests.get(api_url, auth=basicauth, headers=headers, verify=False)


response_json = resp.json()


print(response_json)
```

lab 2.5.py - C:/Users/valeria/AppData/Local/Programs/Python/Python311/lab 2.5.py (3.11.0)

File Edit Format Run Options Window Help

```
import json
import requests
requests.packages.urllib3.disable_warnings()


api_url = "https://10.10.20.48/restconf/data/ietf-interfaces:interfaces"

 headers = { "Accept": "application/yang-data+json",
             "Content-type":"application/yang-data+json"
           }

basicauth = ("cisco", "cisco123!")

resp = requests.get(api_url, auth=basicauth, headers=headers, verify=False)

response_json = resp.json()

print(response_json)
```

En la siguiente imagen se podrá observar el resultado obtenido cuando se corrió el código que se muestra en la parte de arriba

Se deberá completar el código con los comandos que se indican se deberán agregar los siguientes comandos

```
import
json
        import requests
        requests.packages.urllib3.disable_warnings()

        api_url = "https://10.10.20.48/restconf/data/ietf-
        interfaces:interfaces/interface=Loopback99"

        headers = {
                    "Accept": "application/yang-data+json",
                    "Content-type":"application/yang-data+json"
                  }
        basicauth = ("developer", "C1sco12345")

        yangConfig = {
            "ietf-interfaces:interface": {
                "name": "Loopback99",
                "description": "Laboratorio 2.5 prueba",
                "type": "iana-if-type:softwareLoopback",
                "enabled": True,
                "ietf-ip:ipv4": {
                    "address": [
                        {
                            "ip": "99.99.99.99",
                            "netmask": "255.255.255.0"
                        }
                    ]
                },
                "ietf-ip:ipv6": {}
            }
        }

        resp = requests.put(api_url, data=json.dumps(yangConfig), auth=basicauth,
        headers=headers, verify=False)
        if(resp.status_code >= 200 and resp.status_code <= 299):
            print("STATUS OK: {}".format(resp.status_code))
        else:
            print("Error code {}, reply: {}".format(resp.status_code, resp.json()))
```

Captura de la corrida del código de arriba

Conclusión

En el presente documento se trabajo de una forma muy buena ya que conocimos como se puede trabajar con restconf in Python es una manera muy fácil y practica al momento de realizar los ejercicios se tuvo un gran aprendizaje por que me di cuenta que el restconf es un mecanismo para configuraciones rest similar a netconf este usa una base de modelos y comandos definidos.