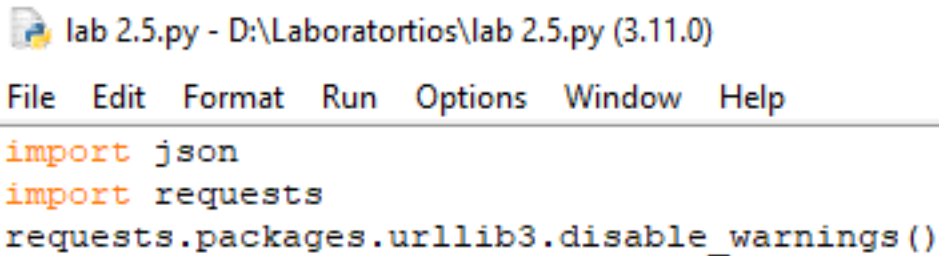


2.5 Lab - RESTCONF with Python.

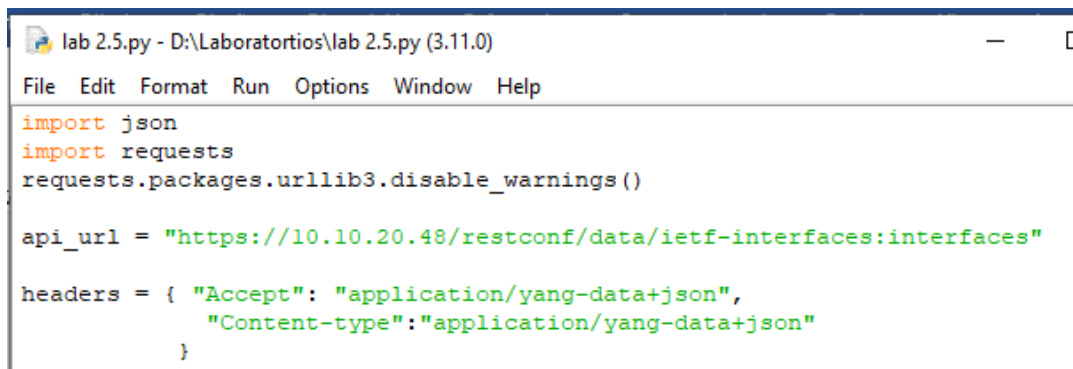
RESTCONF es un protocolo basado en HTTP, RESTCONF se define en el RFC 8040 este es un protocolo y un mecanismo para configuraciones REST, Similar a NETCONF, este usa una base de modelos y comandos definidos por el protocolo NETCONF, Encapsulando esta información en mensajes HTTP.

Importe módulos y deshabilite las advertencias SSL.



```
lab 2.5.py - D:\Laboratortios\lab 2.5.py (3.11.0)
File Edit Format Run Options Window Help
import json
import requests
requests.packages.urllib3.disable_warnings()
```

Cree los componentes de solicitud.

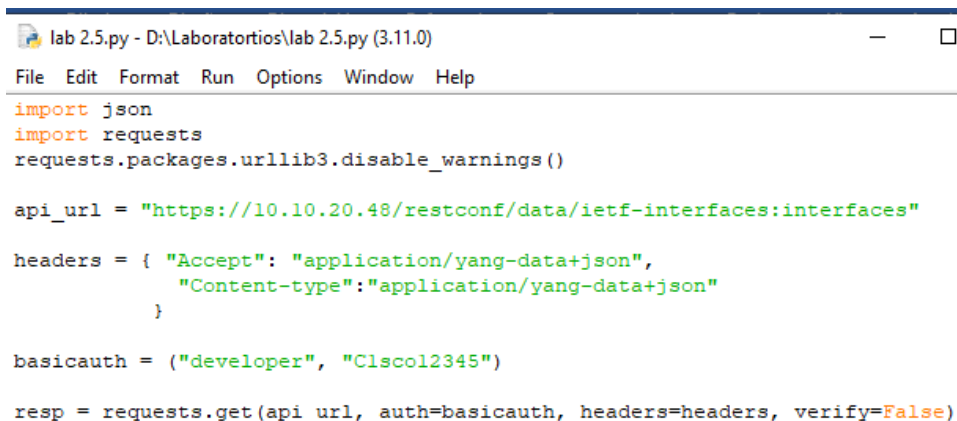


```
lab 2.5.py - D:\Laboratortios\lab 2.5.py (3.11.0)
File Edit Format Run Options Window Help
import json
import requests
requests.packages.urllib3.disable_warnings()

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

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

Envíe la solicitud.



```
lab 2.5.py - D:\Laboratortios\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 = ("developer", "Cisc012345")

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

Evalúe la respuesta.

```
lab 2.5.py - D:\Laboratortios\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 = ("developer", "C1scol2345")

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

response_json = resp.json()

print(response_json)

===== RESTART: D:\Laboratortios\lab 2.5.py =====
{'ietf-interfaces:interfaces': {'interface': [{'name': 'GigabitEthernet1', 'description': 'MANAGEMENT INTERFACE - DON'T TOUCH ME', 'type': 'iana-if-type:ethernetCsmacd', 'enabled': True, 'ietf-ip:ipv4': {'address': [{'ip': '10.10.20.48', 'netmask': '255.255.255.0'}]}, 'ietf-ip:ipv6': {}}, {'name': 'GigabitEthernet2', 'description': 'Network Interface', 'type': 'iana-if-type:ethernetCsmacd', 'enabled': False, 'ietf-ip:ipv4': {}, 'ietf-ip:ipv6': {}}, {'name': 'GigabitEthernet3', 'description': 'Network Interface', 'type': 'iana-if-type:ethernetCsmacd', 'enabled': False, 'ietf-ip:ipv4': {}, 'ietf-ip:ipv6': {}}]}}
```

```
lab 2.5.py - D:\Laboratortios\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 = ("developer", "C1scol2345")

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

response_json = resp.json()

print(json.dumps(response_json, indent=4))
```

```
IDLE Shell 3.11.0
File Edit Shell Debug Options Window Help
AttributeError: module 'json' has no attribute 'volcados'

===== RESTART: D:\Laboratortios\lab 2.5.py =====
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "GigabitEthernet1",
        "description": "MANAGEMENT INTERFACE - DON'T TOUCH ME",
        "type": "iana-if-type:ethernetCsmacd",
        "enabled": true,
        "ietf-ip:ipv4": {
          "address": [
            {
              "ip": "10.10.20.48",
              "netmask": "255.255.255.0"
            }
          ]
        },
        "ietf-ip:ipv6": {}
      },
      {
        "name": "GigabitEthernet2",
        "description": "Network Interface",
        "type": "iana-if-type:ethernetCsmacd",
        "enabled": false,
        "ietf-ip:ipv4": {},
        "ietf-ip:ipv6": {}
      },
      {
        "name": "GigabitEthernet3",
        "description": "Network Interface",
        "type": "iana-if-type:ethernetCsmacd",
        "enabled": false,
        "ietf-ip:ipv4": {},
        "ietf-ip:ipv6": {}
      }
    ]
  }
}
```

Modificar la configuración de la interfaz con RESTCONF en Python

Cree los componentes de solicitud.

```
lab 2.5 part2.py - D:/Laboratorios/lab 2.5 part2.py (3.11.0)
File Edit Format Run Options Window Help
requests.packages.urllib3.disable_warnings()

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

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

basicauth = ("developer", "Cisc012345")

yangConfig = {
    "ietf-interfaces:interface": {
        "name": "Loopback99",
        "description": "WHATEVER99",
        "type": "iana-if-type:softwareLoopback",
        "enabled": True,
        "ietf-ip:ipv4": {
            "address": [
                {
                    "ip": "99.99.99.99",
                    "netmask": "255.255.255.0"
                }
            ]
        },
        "ietf-ip:ipv6": {}
    }
}
```

Envíe la solicitud PUT.

```
lab 2.5 part2.py - D:/Laboratorios/lab 2.5 part2.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/interface

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

basicauth = ("developer", "Cisc012345")

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, header
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()))

#response_json = resp.json()
#print(json.dumps(response_json, indent=4))
```

```
===== RESTART: D:/Laboratorios/lab 2.5 part2.py =====
STATUS OK: 201
```

Conclusión:

El protocolo RESTCONF es similar a otros protocolos REST. Una operación RESTCONF consiste en un método HTTP y el URI del recurso solicitado.

Método: el cliente envía la operación RESTCONF identificada por el método HTTP, y la operación de solicitud actúa sobre el recurso de destino especificado por el URI solicitado. Los métodos son GET, PUT, DELETE, PATCH, OPTIONS.

Entrada: puerto de entrada RESTCONF de hello-world ("/ hello-world / restconf").

Recurso: una expresión de ruta que identifica el recurso al que debe acceder la operación. Si no se proporciona este dominio, el recurso de destino es la API en sí, que se proporciona como "application / vnd.yang.api" como tipo de medio.

Consulta: el conjunto de parámetros asociados con el mensaje RESTCONF. Todos estos tienen esta forma familiar de emparejamiento "nombre = valor". La especificación define un conjunto específico de parámetros, aunque el servidor puede optar por admitir parámetros adicionales que no están definidos en este documento.