

Lab – NETCONF w/Python: Get Operational Data

Objectives

Part 1: Retrieve the IOS XE VM's Operational Data - Statistics

Background / Scenario

In this lab, you will learn how to use NETCONF to retrieve operational data from the network device.

Required Resources

- Access to a router with the IOS XE operating system version 16.6 or higher
- Python 3.x environment

Instructions

Part 1: Retrieve Interface Statistics

In this part, you will use the ncclient module to retrieve the device's operational data. The data are returned back in XML form. In the following steps this data will be transformed into a tabular output.

Step 1: Use ncclient to retrieve the device's running configuration.

- In Python IDLE, create a new Python script file:
- In the new Python script file editor, import the "manager" class from the ncclient module and the xml.dom.minidom module:

```
from ncclient import manager
import xml.dom.minidom
```

- Set up an **m** connection object using the `manager.connect()` function to the IOS XE device.

```
m = manager.connect(
    host="192.168.56.101",
    port=830,
    username="cisco",
    password="cisco123!",
    hostkey_verify=False
)
```

- After a successful NETCONF connection, using the "get()" function of the "m" NETCONF session object to retrieve and print the device's operational data. The `get()` function expects a "filter" string parameter that defines the NETCONF filter.

The following filter retrieves the interfaces-state operational data (statistics), as defined in the ietf-interfaces YANG model:

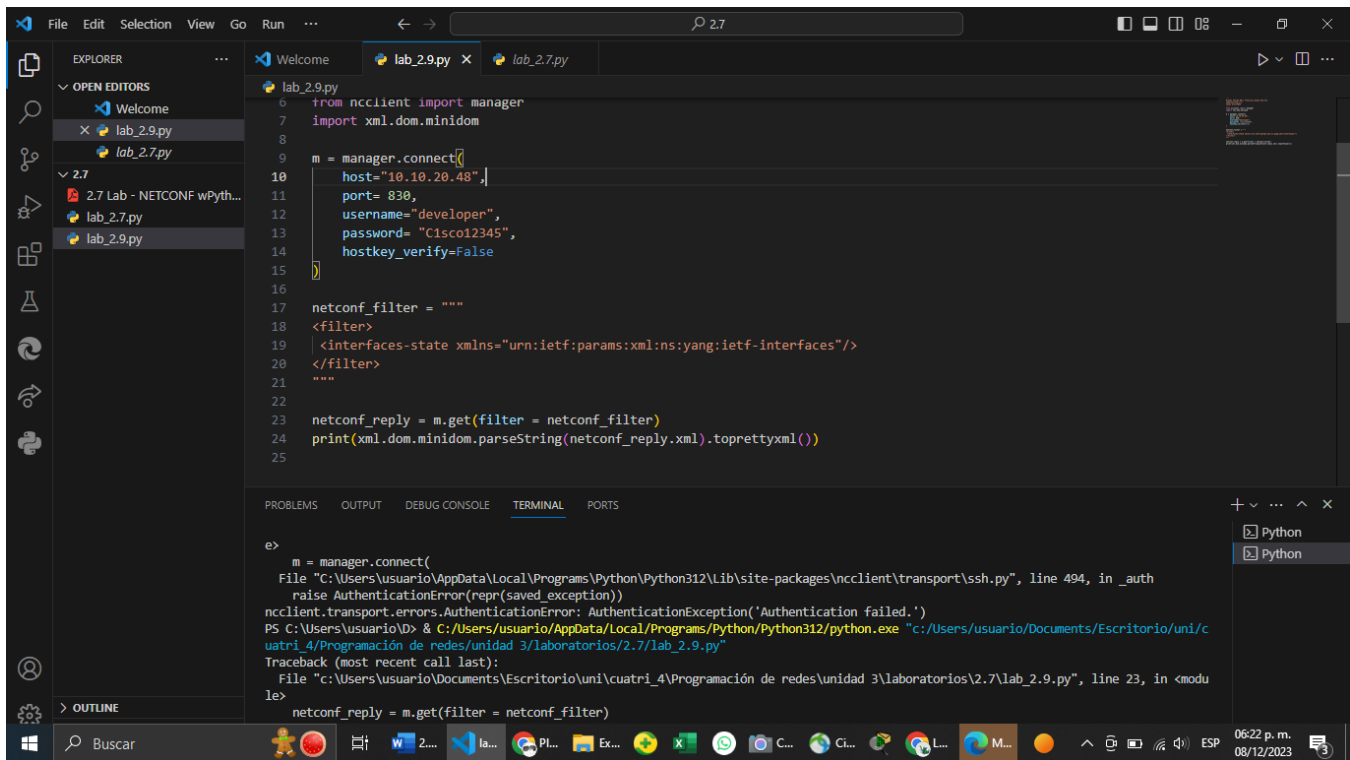
Note: Executing the `get()` function without a filter is similar to execute "debug all".

```
netconf_filter = """
<filter>
```

```
<interfaces-state xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces"/>
</filter>
"""
```

```
netconf_reply = m.get(filter = netconf_filter)
print(xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml())
```

- e. Execute the Python script and explore the output.

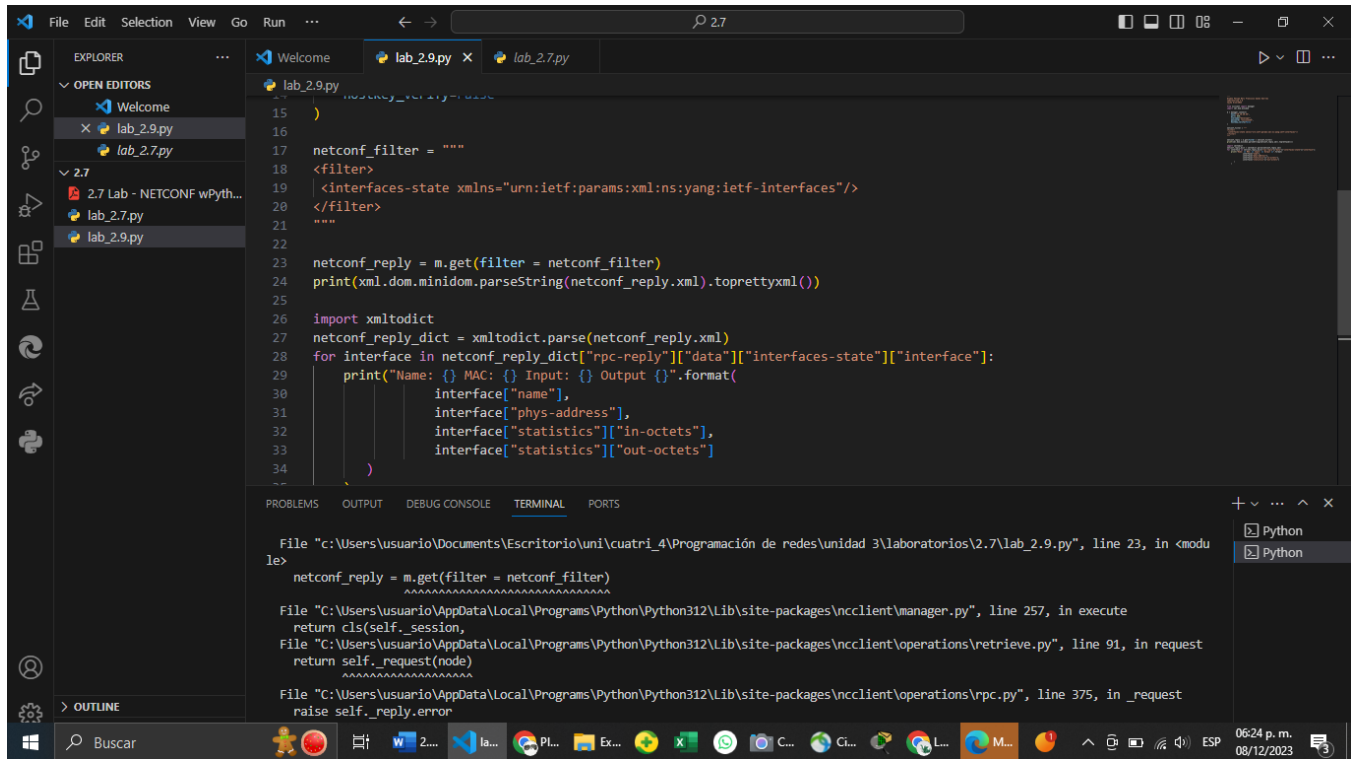


- f. Convert the XML netconf_reply data to a Python dictionary using the “xmldict” module. You can use a simple for loop to print a summary view of the statistical data:

```
import xmldict
netconf_reply_dict = xmldict.parse(netconf_reply.xml)
for interface in netconf_reply_dict["rpc-reply"]["data"]["interfaces-state"]["interface"]:
    print("Name: {} MAC: {} Input: {} Output {}".format(
        interface["name"],
        interface["phys-address"],
        interface["statistics"]["in-octets"],
        interface["statistics"]["out-octets"]
    ))
```

- g. Execute the script and explore the output.

Lab – NETCONF w/Python: Get Operational Data



The screenshot shows a Visual Studio Code editor with a Python script named `lab_2.9.py` open. The script uses the `ncclient` library to connect to a NETCONF server and retrieve interface data. The terminal window at the bottom displays the execution output, showing the XML response from the server and the parsed dictionary structure.

```
15 )
16
17 netconf_filter = """
18 <filter>
19   <interfaces-state xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces"/>
20 </filter>
21 """
22
23 netconf_reply = m.get(filter = netconf_filter)
24 print(xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml())
25
26 import xmltodict
27 netconf_reply_dict = xmltodict.parse(netconf_reply.xml)
28 for interface in netconf_reply_dict["rpc-reply"]["data"]["interfaces-state"]["interface"]:
29     print("Name: {} MAC: {} Input: {} Output {}".format(
30         interface["name"],
31         interface["phys-address"],
32         interface["statistics"]["in-octets"],
33         interface["statistics"]["out-octets"]
34     ))
```

Terminal Output:

```
File "c:\Users\usuario\Documents\Escritorio\uni\cuatri_4\Programación de redes\unidad 3\laboratorios\2.7\lab_2.9.py", line 23, in <module>
le>
netconf_reply = m.get(filter = netconf_filter)
~~~~~
File "C:\Users\usuario\AppData\Local\Programs\Python\Python312\Lib\site-packages\ncclient\manager.py", line 257, in execute
return cls(self.session,
File "C:\Users\usuario\AppData\Local\Programs\Python\Python312\Lib\site-packages\ncclient\operations\retrieve.py", line 91, in request
return self._request(node)
~~~~~
File "C:\Users\usuario\AppData\Local\Programs\Python\Python312\Lib\site-packages\ncclient\operations\rpc.py", line 375, in _request
raise self_reply.error
```

Link de los videos

<https://drive.google.com/drive/folders/1ued0ZQycqiBz-P4meMLWfOUzQJVLcAbv?usp=sharing>