

Lab – NETCONF w/Python: Get Operational Data

Nombre:	Monserrat Orduña basaldua
Num Control:	1222100895

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.

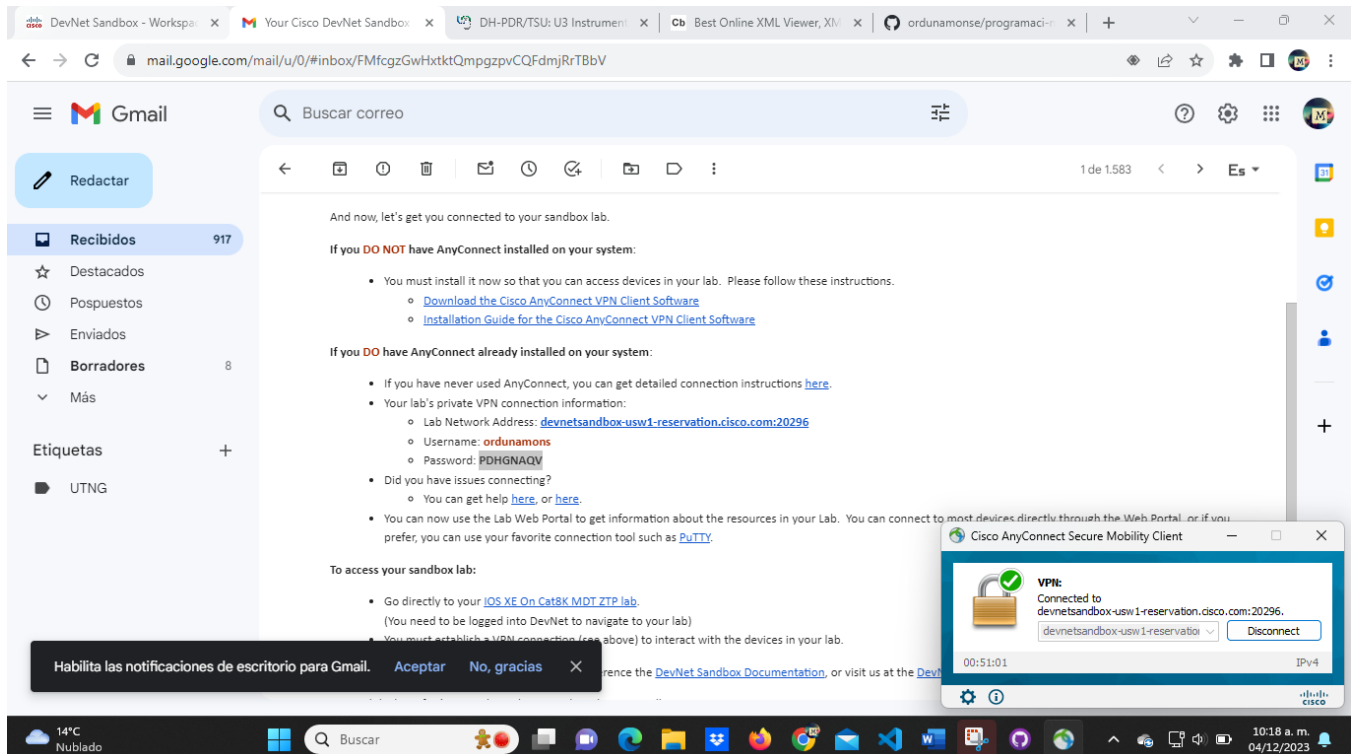
Required Resources

- 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.



The screenshot shows a web browser window with multiple tabs. The active tab is a Gmail inbox. The email content is as follows:

And now, let's get you connected to your sandbox lab.

If you **DO NOT** have AnyConnect installed on your system:

- You must install it now so that you can access devices in your lab. Please follow these instructions.
 - [Download the Cisco AnyConnect VPN Client Software](#)
 - [Installation Guide for the Cisco AnyConnect VPN Client Software](#)

If you **DO** have AnyConnect already installed on your system:

- If you have never used AnyConnect, you can get detailed connection instructions [here](#).
- Your lab's private VPN connection information:
 - Lab Network Address: [devnetsandbox-usw1-reservation.cisco.com:20296](#)
 - Username: **ordunamons**
 - Password: **PDHGNAQV**
- Did you have issues connecting?
 - You can get help [here](#), or [here](#).
- You can now use the Lab Web Portal to get information about the resources in your Lab. You can connect to most devices directly through the Web Portal, or if you prefer, you can use your favorite connection tool such as [PuTTY](#).

To access your sandbox lab:

- Go directly to your [IOS XE On Cat8K MDT ZTP lab](#). (You need to be logged into DevNet to navigate to your lab)
- You must establish a VPN connection (see above) to interact with the devices in your lab.

Below the email content, there is a notification bar that says "Habilita las notificaciones de escritorio para Gmail. Aceptar No, gracias".

In the bottom right corner, there is a Cisco AnyConnect Secure Mobility Client window showing a successful VPN connection to devnetsandbox-usw1-reservation.cisco.com:20296. The window includes a "Disconnect" button and shows the IP address as IPv4.

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

- a. In Python IDLE, create a new Python script file:
- b. 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
```

- c. 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
)
```

- d. 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.

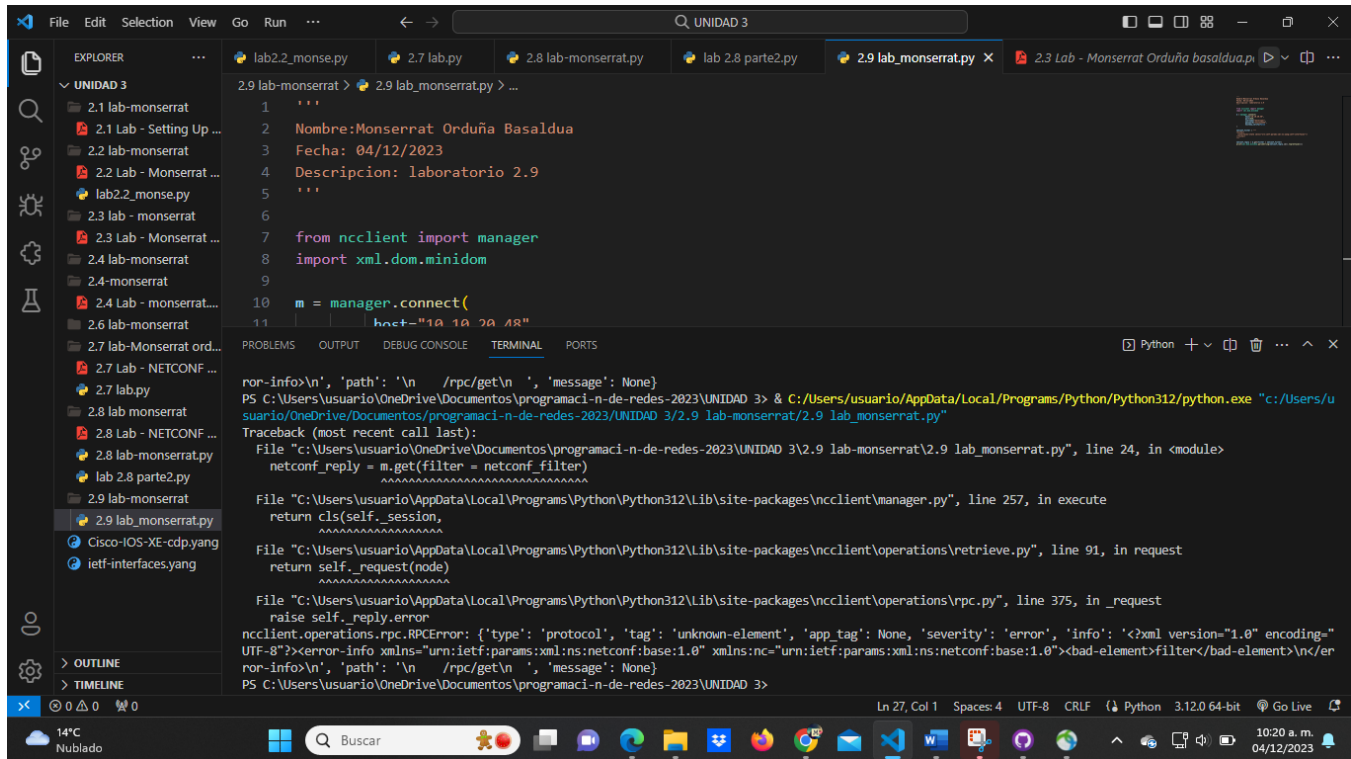
The following filter retrieves the interfaces-state operational data (statistics), as defined in the ietf-interfaces YANG model:

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.



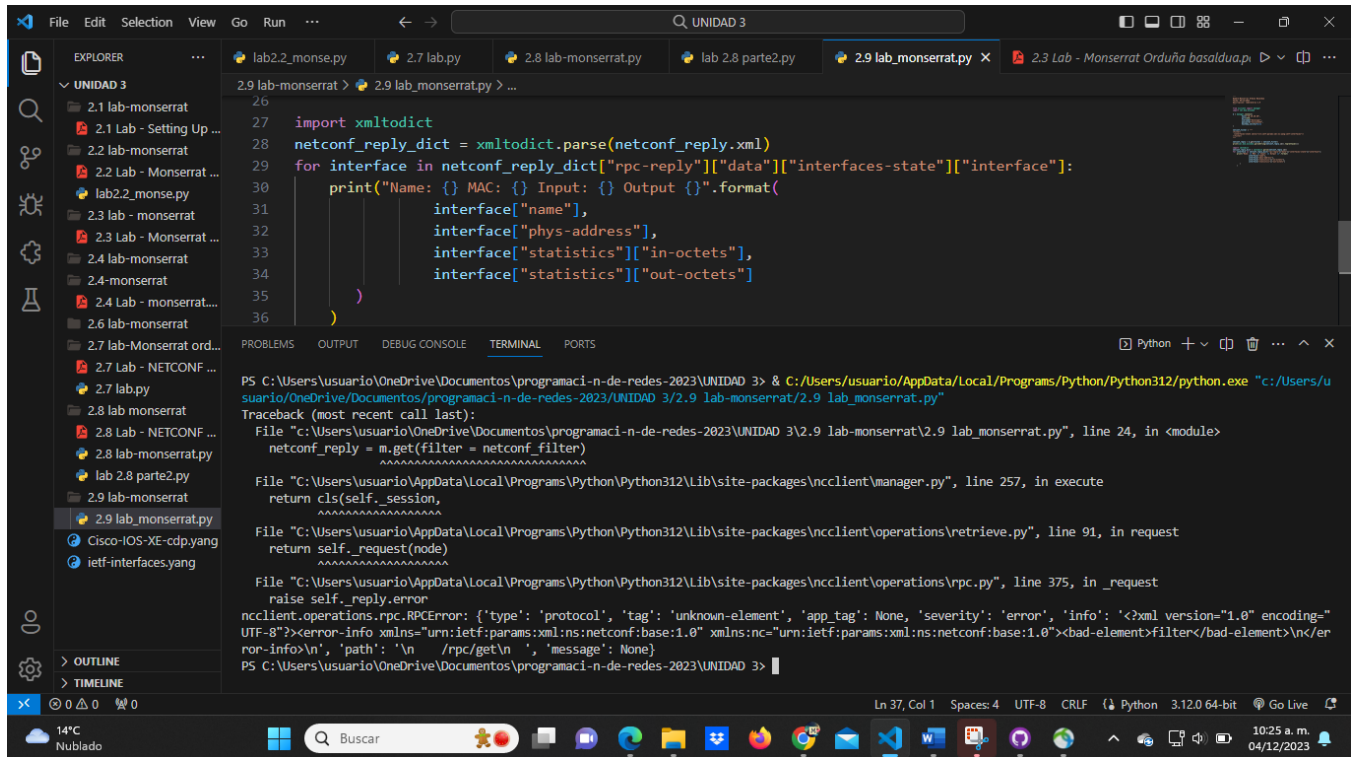
Marca error por que el código de este filtro ya no esta disponibles

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

```
import xmlltodict
netconf_reply_dict = xmlltodict.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 in the main window and a terminal window at the bottom. The script, located in `2.9 lab_monserrat.py`, uses `xmltodict` to parse a NETCONF reply and prints interface details. The terminal shows the command to run the script and a traceback error.

```
26
27 import xmltodict
28 netconf_reply_dict = xmltodict.parse(netconf_reply.xml)
29 for interface in netconf_reply_dict["rpc-reply"]["data"]["interfaces-state"]["interface"]:
30     print("Name: {} MAC: {} Input: {} Output {}".format(
31         interface["name"],
32         interface["phys-address"],
33         interface["statistics"]["in-octets"],
34         interface["statistics"]["out-octets"]
35     ))
36
```

Terminal output:

```
PS C:\Users\usuario\OneDrive\Documentos\programaci-n-de-redes-2023\UNIDAD 3> & C:/Users/usuario/AppData/Local/Programs/Python/Python312/python.exe "c:/Users/usuario/OneDrive/Documentos/programaci-n-de-redes-2023\UNIDAD 3\2.9 lab-monserrat\2.9 lab_monserrat.py"
Traceback (most recent call last):
  File "c:\Users\usuario\OneDrive\Documentos\programaci-n-de-redes-2023\UNIDAD 3\2.9 lab-monserrat\2.9 lab_monserrat.py", line 24, in <module>
    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
ncclient.operations.rpc.RPCError: {'type': 'protocol', 'tag': 'unknown-element', 'app_tag': None, 'severity': 'error', 'info': '<?xml version="1.0" encoding="UTF-8"?><error-info xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"><bad-element>filter</bad-element>\n</error-info>\n', 'path': '\n /rpc/get\n ', 'message': None}
PS C:\Users\usuario\OneDrive\Documentos\programaci-n-de-redes-2023\UNIDAD 3>
```

En general marca errores por que en el link ya aspiro y no esta disponibles