



Universidad Tecnológica
del Norte de Guanajuato
Organismo Público Descentralizado del Gobierno del Estado

Facultad de: *Infraestructura de Redes Digitales*

Nombre del Alumno(a): *Ángel Armando Ramírez Vázquez*

Matrícula: *1221100627*

Materia:

Programación de Redes

Nombre de la Actividad:

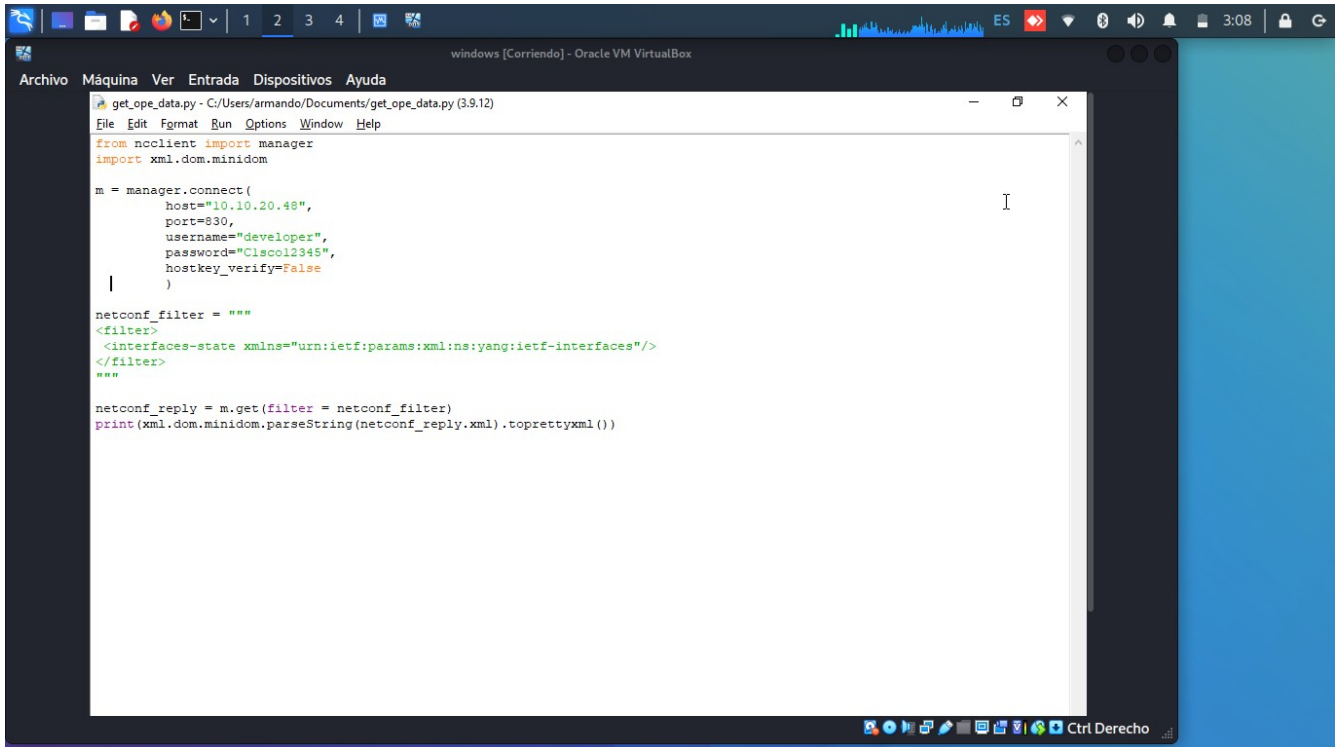
2.9 Lab - NETCONF wPython Get Operational Data

Profesor:

Barron Rodríguez Gabriel

Lugar y Fecha de Presentación: *Dolores Hidalgo C.I.N.*; hoy *6* de *Diciembre* del
2022

Paso 1. Usa el modulo ncclient para devolver información de la configuración.



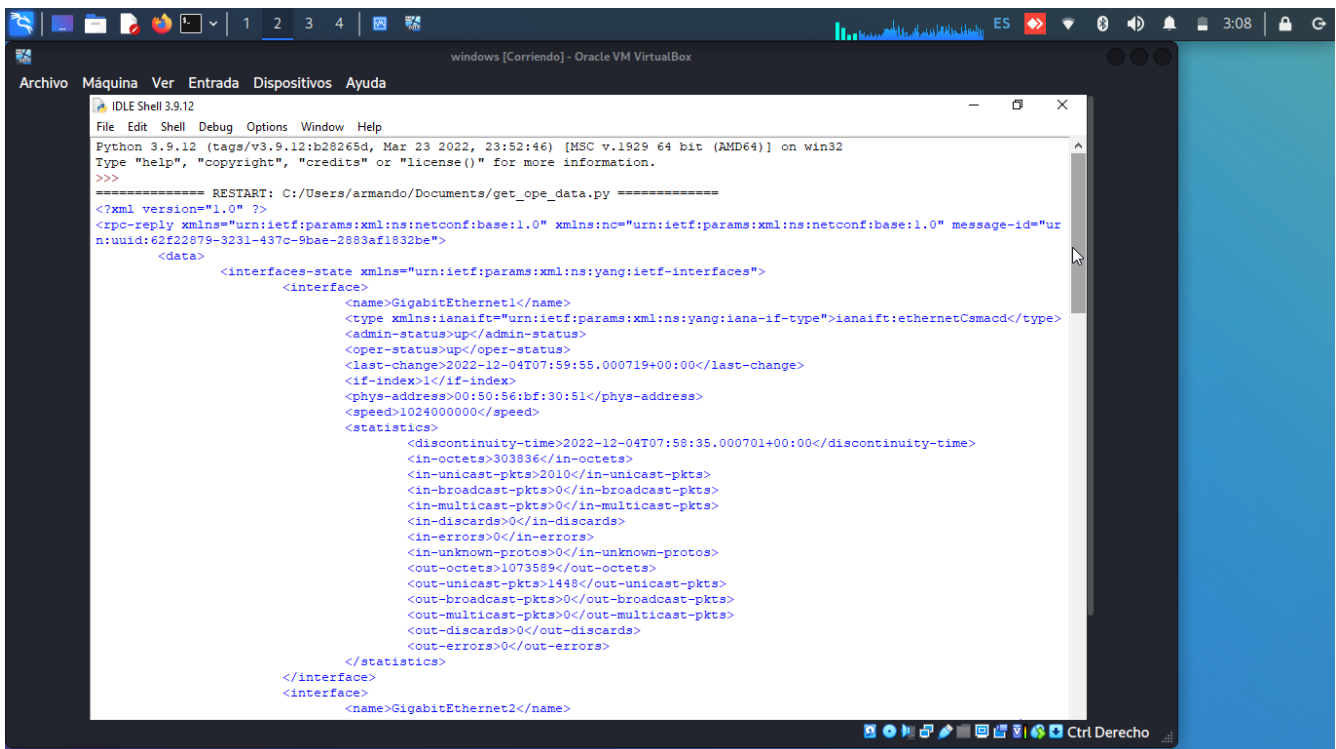
```
get_ope_data.py - C:/Users/armando/Documents/get_ope_data.py (3.9.12)
File Edit Format Run Options Window Help

from ncclient import manager
import xml.dom.minidom

m = manager.connect(
    host="10.10.20.48",
    port=830,
    username="developer",
    password="Cisc0ol2345",
    hostkey_verify=False
)

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

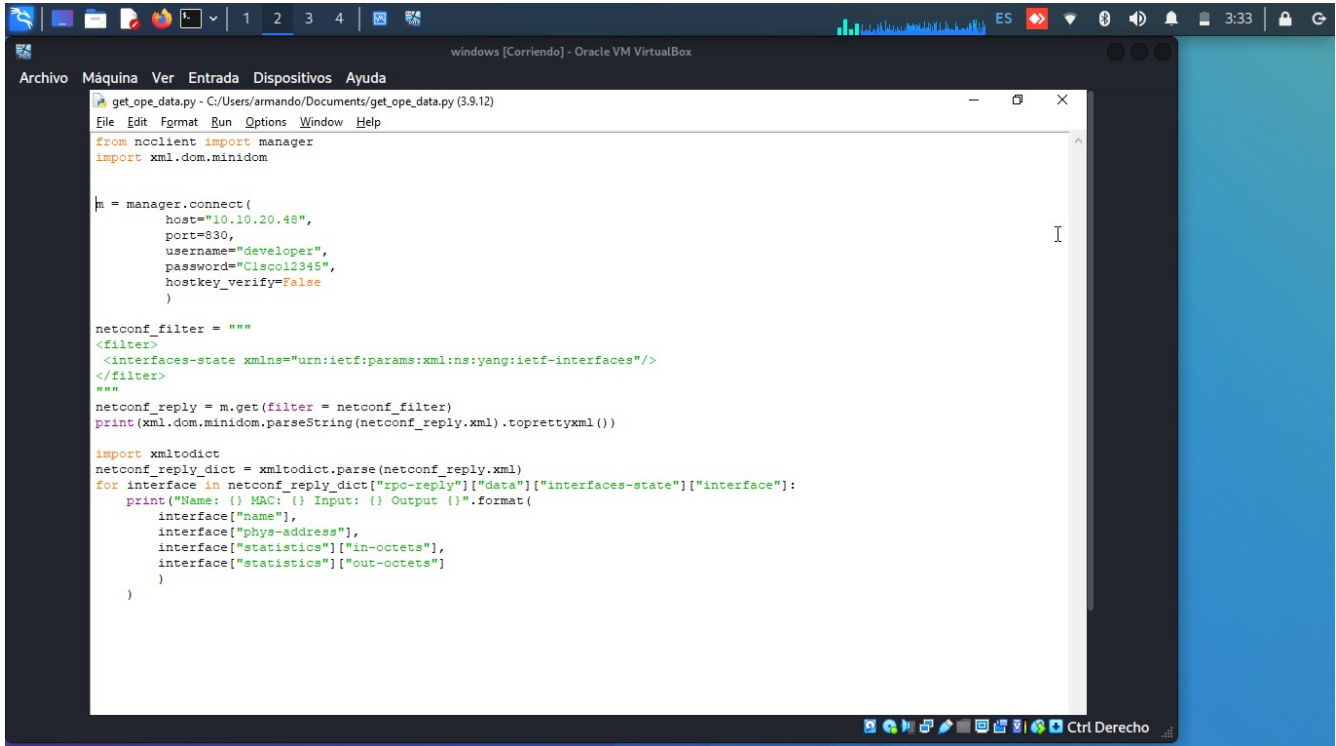
netconf_reply = m.get(filter=netconf_filter)
print(xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml())
```



```
IDLE Shell 3.9.12
File Edit Shell Debug Options Window Help

Python 3.9.12 (tags/v3.9.12:b282654, Mar 23 2022, 23:52:46) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/armando/Documents/get_ope_data.py =====
<?xml version="1.0" ?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="urn:uuid:62f22879-3231-437c-9bae-2883af1832be">
  <data>
    <interfaces-state xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
      <interface>
        <name>GigabitEthernet1</name>
        <type xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">ianaift:ethernetCsmacd</type>
        <admin-status>up</admin-status>
        <oper-status>up</oper-status>
        <last-change>2022-12-04T07:59:55.000719+00:00</last-change>
        <if-index>1</if-index>
        <phys-address>00:50:56:bf:30:51</phys-address>
        <speed>1024000000</speed>
        <statistics>
          <discontinuity-time>2022-12-04T07:58:35.000701+00:00</discontinuity-time>
          <in-octets>303836</in-octets>
          <in-unicast-pkts>2010</in-unicast-pkts>
          <in-broadcast-pkts>0</in-broadcast-pkts>
          <in-multicast-pkts>0</in-multicast-pkts>
          <in-discards>0</in-discards>
          <in-errors>0</in-errors>
          <in-unknown-protos>0</in-unknown-protos>
          <out-octets>1073589</out-octets>
          <out-unicast-pkts>1448</out-unicast-pkts>
          <out-broadcast-pkts>0</out-broadcast-pkts>
          <out-multicast-pkts>0</out-multicast-pkts>
          <out-discards>0</out-discards>
          <out-errors>0</out-errors>
        </statistics>
      </interface>
    </interfaces-state>
  </data>
</rpc-reply>
```

Paso 2. Instalamos y usamos el modulo xmltodict para filtrar información.



The screenshot shows a text editor window titled 'get_ope_data.py - C:/Users/armando/Documents/get_ope_data.py (3.9.12)'. The script uses the 'netclient' library to connect to a device and retrieve configuration data. It then uses 'xmltodict' to parse the XML response and filter for interface information. The code is as follows:

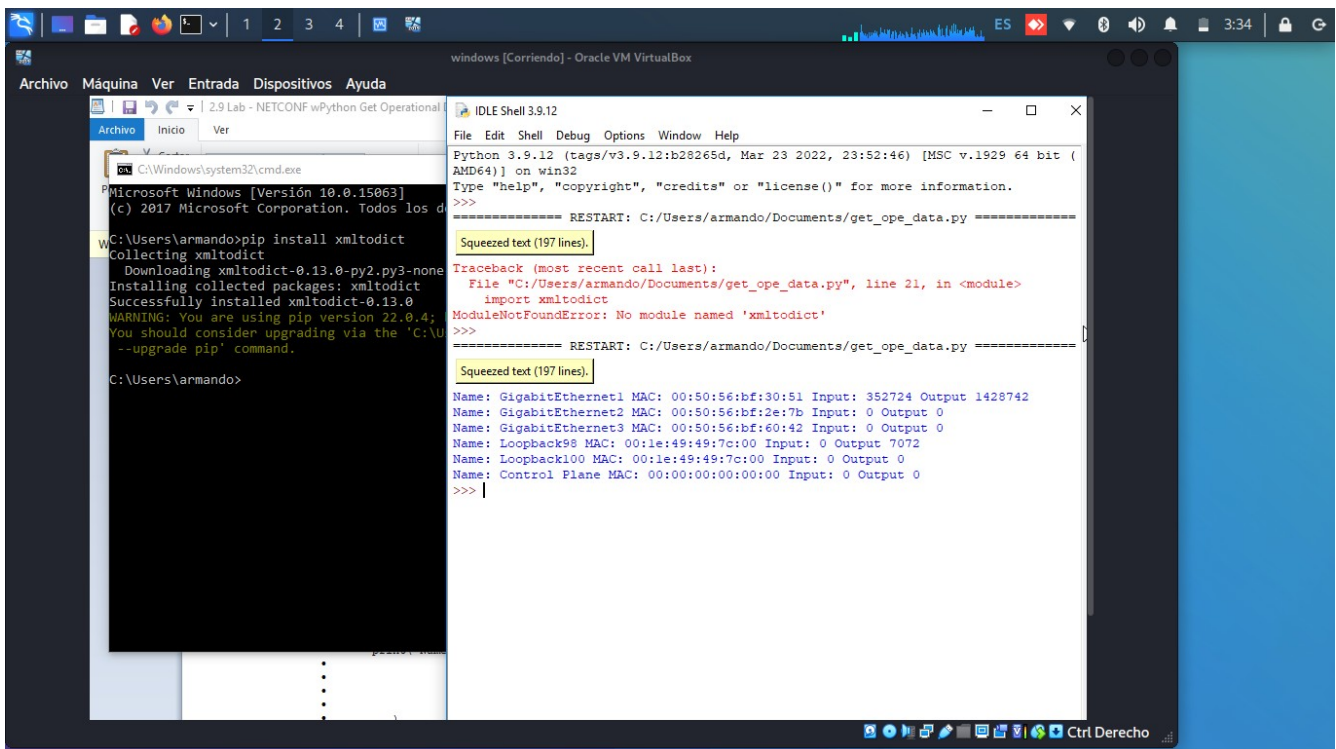
```
from netclient import manager
import xml.dom.minidom

m = manager.connect(
    host="10.10.20.48",
    port=830,
    username="developer",
    password="Cisc0ol2345",
    hostkey_verify=False
)

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())

import xmltodict
netconf_reply_dict = xmltodict.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"]
    ))
```



The screenshot shows a Windows command prompt window titled '2.9 Lab - NETCONF w/Python Get Operational'. The user has installed 'xmltodict' using pip. The output of the installation is as follows:

```
C:\Users\armando>pip install xmltodict
Collecting xmltodict
  Downloading xmltodict-0.13.0-py2.py3-none-any.whl (19 kB)
Installing collected packages: xmltodict
Successfully installed xmltodict-0.13.0
WARNING: You are using pip version 22.0.4; you should consider upgrading via the 'C:\Users\armando>python -m pip --upgrade pip' command.
```

The user then runs the script 'get_ope_data.py'. The output of the script is as follows:

```
Traceback (most recent call last):
  File "C:/Users/armando/Documents/get_ope_data.py", line 21, in <module>
    import xmltodict
ModuleNotFoundError: No module named 'xmltodict'

===== RESTART: C:/Users/armando/Documents/get_ope_data.py =====

Squeezed text (197 lines).

Name: GigabitEthernet1 MAC: 00:50:56:bf:30:51 Input: 352724 Output 1428742
Name: GigabitEthernet2 MAC: 00:50:56:bf:2e:7b Input: 0 Output 0
Name: GigabitEthernet3 MAC: 00:50:56:bf:60:42 Input: 0 Output 0
Name: Loopback98 MAC: 00:1e:49:49:7c:00 Input: 0 Output 7072
Name: Loopback100 MAC: 00:1e:49:49:7c:00 Input: 0 Output 0
Name: Control Plane MAC: 00:00:00:00:00:00 Input: 0 Output 0
```