

## RINCON ULLOA YAZMIN ELIZABETH

### Lab – NETCONF w/Python: List Capabilities

#### Objectives

**Part 1: Install the ncclient Python module**

**Part 2: Connect to IOS XE's NETCONF service using ncclient**

**Part 3: List the IOS XE's capabilities – supported YANG models**

#### Background / Scenario

Working with NETCONF does not require working with raw NETCONF RPC messages and XML. In this lab you will learn how to use the ncclient Python module to easily interact with network devices using NETCONF. You will learn how to identify which YANG models are supported by the device. This information is helpful when building a production network automation system, that requires specific YANG models to be supported by the given 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: Install the ncclient Python module

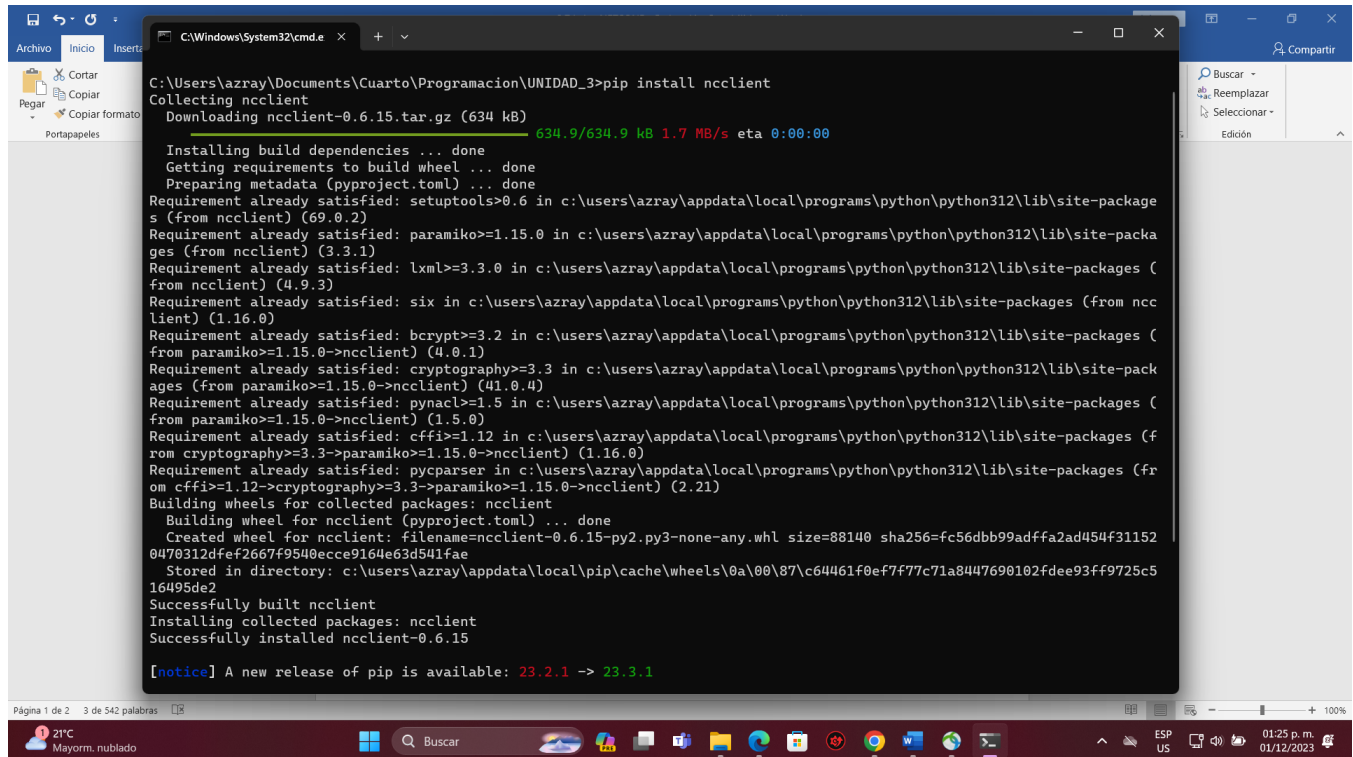
In this part, you will install ncclient module into your Python environment. ncclient is a python module that simplifies NETCONF operations with built in functions that deal with the XML messages and RPC calls.

Explore the ncclient module on the project GitHub repository: <https://github.com/ncclient/ncclient>

##### Step 1: Use pip to install ncclient.

- a. Start a new Windows command prompt (`cmd`).
- b. Install ncclient using pip in the Windows command prompt:

```
pip install ncclient
```



```
C:\Users\azray\Documents\Cuarto\Programacion\UNIDAD_3>pip install ncclient
Collecting ncclient
  Downloading ncclient-0.6.15.tar.gz (634 kB)
    634.9/634.9 kB 1.7 MB/s eta 0:00:00
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Requirement already satisfied: setuptools>0.6 in c:\users\azray\appdata\local\programs\python\python312\lib\site-packages (from ncclient) (69.0.2)
Requirement already satisfied: paramiko>=1.15.0 in c:\users\azray\appdata\local\programs\python\python312\lib\site-packages (from ncclient) (3.3.1)
Requirement already satisfied: lxml>=3.3.0 in c:\users\azray\appdata\local\programs\python\python312\lib\site-packages (from ncclient) (4.9.3)
Requirement already satisfied: six in c:\users\azray\appdata\local\programs\python\python312\lib\site-packages (from ncclient) (1.16.0)
Requirement already satisfied: bcrypt>=3.2 in c:\users\azray\appdata\local\programs\python\python312\lib\site-packages (from paramiko>=1.15.0->ncclient) (4.0.1)
Requirement already satisfied: cryptography>=3.3 in c:\users\azray\appdata\local\programs\python\python312\lib\site-packages (from paramiko>=1.15.0->ncclient) (41.0.4)
Requirement already satisfied: pynacl>=1.5 in c:\users\azray\appdata\local\programs\python\python312\lib\site-packages (from paramiko>=1.15.0->ncclient) (1.5.0)
Requirement already satisfied: cffi>=1.12 in c:\users\azray\appdata\local\programs\python\python312\lib\site-packages (from cryptography>=3.3->paramiko>=1.15.0->ncclient) (1.16.0)
Requirement already satisfied: pycparser in c:\users\azray\appdata\local\programs\python\python312\lib\site-packages (from cffi>=1.12->cryptography>=3.3->paramiko>=1.15.0->ncclient) (2.21)
Building wheels for collected packages: ncclient
  Building wheel for ncclient (pyproject.toml) ... done
  Created wheel for ncclient: filename=ncclient-0.6.15-py3-none-any.whl size=88140 sha256=fc56dbb99adffa2ad454f311520470312dfef2667f9540ecce9164e63d541fae
  Stored in directory: c:\users\azray\appdata\local\pip\cache\wheels\0a\87\c64461f0ef7f77c71a8447690102fdee93ff9725c516495de2
Successfully built ncclient
Installing collected packages: ncclient
Successfully installed ncclient-0.6.15

[notice] A new release of pip is available: 23.2.1 -> 23.3.1
```

- c. Verify that ncclient has been successfully installed. Start Python IDLE and in the interactive shell try to import the ncclient module:

```
import ncclient
```

## Part 2: Connect to IOS XE's NETCONF service using ncclient

### Connect to IOS XE's NETCONF service using ncclient.

The ncclient module provides a “manager” class with “connect ()” function to setup the remote NETCONF connection. After a successful connection, the returned object represents the NETCONF connection to the remote device.

- In Python IDLE, create a new Python script file:
- In the new Python script file editor, import the “manager” class from the ncclient module:
- Setup 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
)
```

The parameters of the manager.connect () function are:

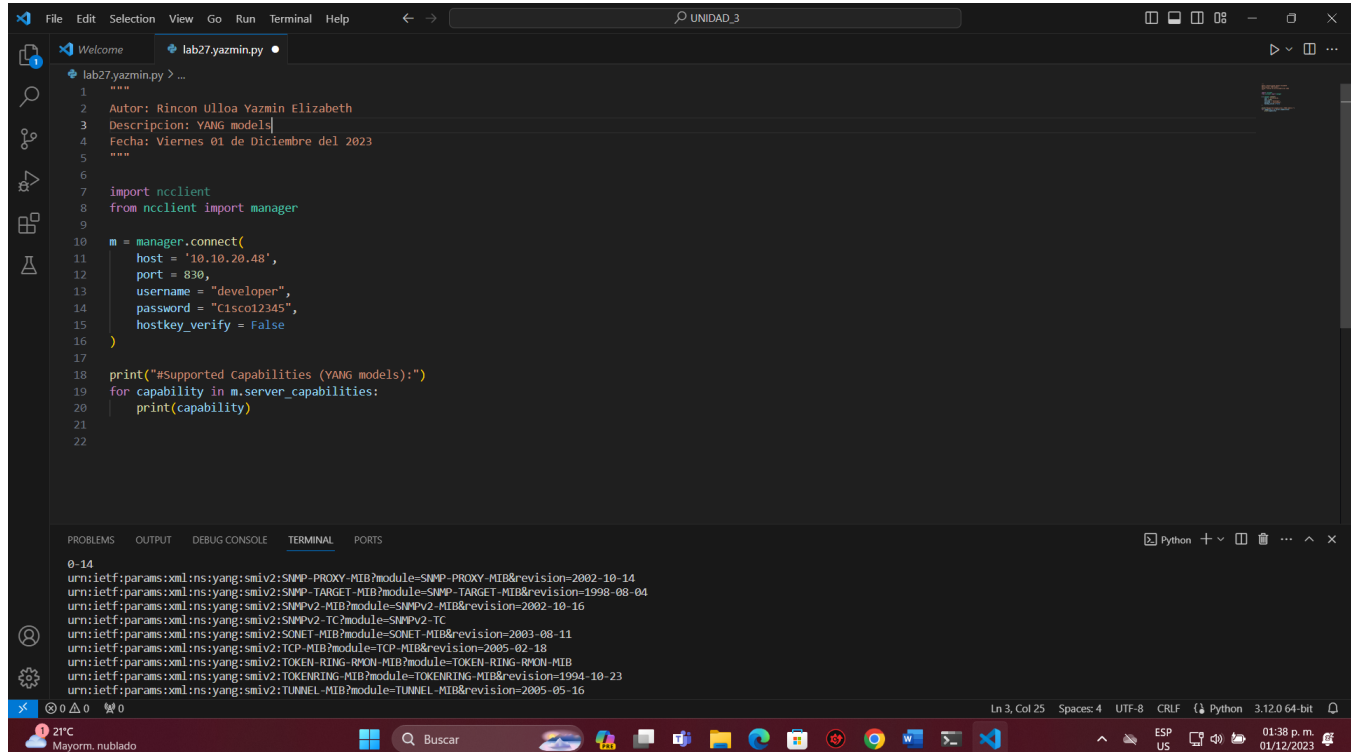
- `host` – the address (host or IP) of the remote device (adjust the IP address to match the router's current address)
- `port` – the remote port of the NETCONF service
- `username` – remote ssh username (in this lab “cisco” for that was setup in the IOS XE VM)
- `password` – remote ssh password (in this lab “cisco123!” for that was setup in the IOS XE VM)
- `hostkey_verify` – whether to verify the ssh fingerprint (in lab it is safe to set to False, in production environments you should always verify the ssh fingerprints)

### Part 3: List the IOS XE's capabilities – supported YANG models

#### Send show commands and display the output

- The `m` object, returned by the `manager.connect()` function that represents the NETCONF remote session. In every NETCONF session, the server first sends its list of capabilities – supported YANG models. With the `ncclient` module, the received list of capabilities is stored in the `m.server_capabilities` list.
- Use a for loop and a print function to print the device capabilities:

```
print("#Supported Capabilities (YANG models):")
for capability in m.server_capabilities:
    print(capability)
```
- Execute the Python script file to see the results.



The screenshot shows a VS Code editor window with a file named `lab27.yazmin.py`. The script imports `ncclient` and `manager` from `ncclient`, then connects to a remote device using the following parameters: `host = '10.10.20.48'`, `port = 830`, `username = "developer"`, `password = "Cisco12345"`, and `hostkey_verify = False`. After connecting, it prints the supported capabilities (YANG models) using a for loop.

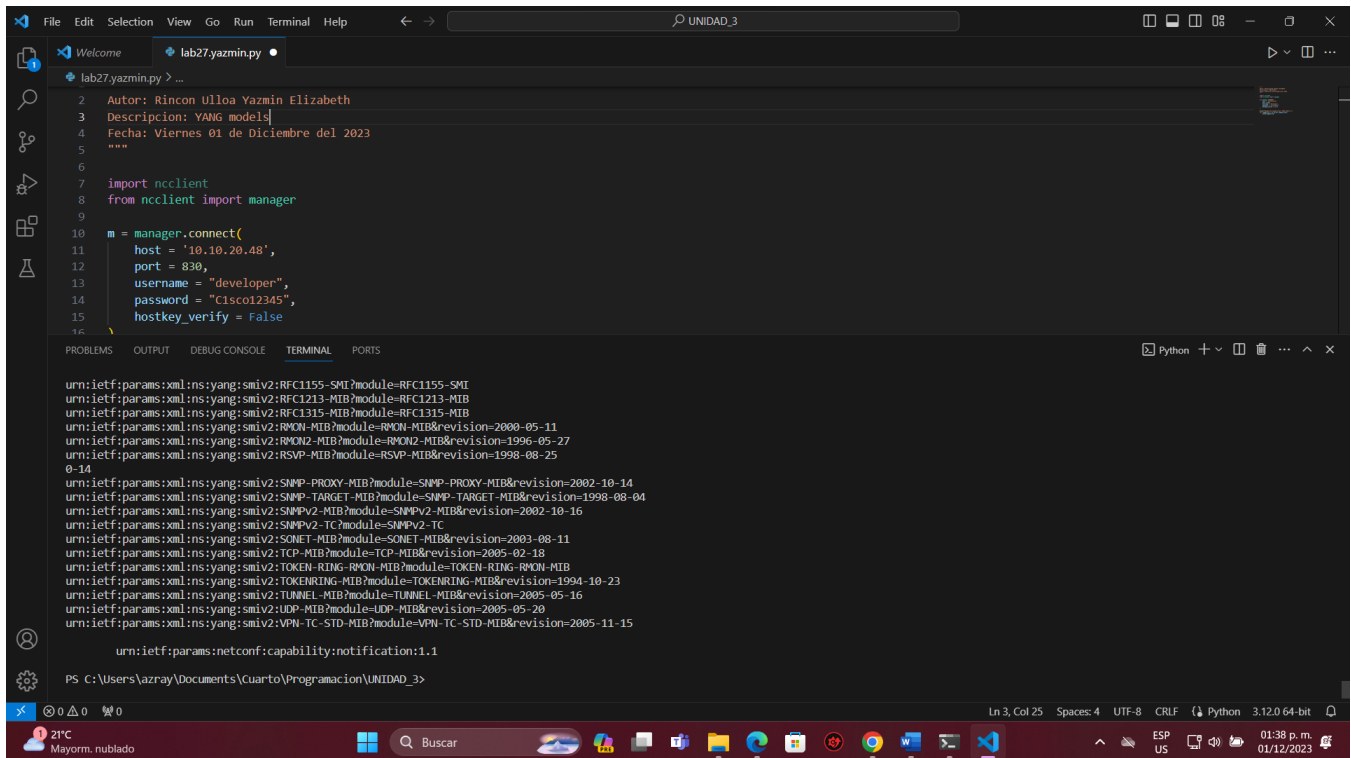
```
1 """
2 Autor: Rincon Ulloa Yazmin Elizabeth
3 Descripcion: YANG models
4 Fecha: Viernes 01 de Diciembre del 2023
5 """
6
7 import ncclient
8 from ncclient import manager
9
10 m = manager.connect(
11     host = '10.10.20.48',
12     port = 830,
13     username = "developer",
14     password = "Cisco12345",
15     hostkey_verify = False
16 )
17
18 print("#Supported capabilities (YANG models):")
19 for capability in m.server_capabilities:
20     print(capability)
21
22
```

The terminal output shows the following capabilities:

```
0-14
urn:ietf:params:xml:ns:yang:smiv2:SNMP-PROXY-MIB?module=SNMP-PROXY-MIB&revision=2002-10-14
urn:ietf:params:xml:ns:yang:smiv2:SNMP-TARGET-MIB?module=SNMP-TARGET-MIB&revision=1998-08-04
urn:ietf:params:xml:ns:yang:smiv2:SNMPv2-MIB?module=SNMPv2-MIB&revision=2002-10-16
urn:ietf:params:xml:ns:yang:smiv2:SNMPv2-TC?module=SNMPv2-TC
urn:ietf:params:xml:ns:yang:smiv2:SONET-MIB?module=SONET-MIB&revision=2003-08-11
urn:ietf:params:xml:ns:yang:smiv2:TCP-MIB?module=TCP-MIB&revision=2005-02-18
urn:ietf:params:xml:ns:yang:smiv2:TOKEN-RING-RMON-MIB?module=TOKEN-RING-RMON-MIB
urn:ietf:params:xml:ns:yang:smiv2:TOKENRING-MIB?module=TOKENRING-MIB&revision=1994-10-23
urn:ietf:params:xml:ns:yang:smiv2:TUNNEL-MIB?module=TUNNEL-MIB&revision=2005-05-16
```

d. Is the Cisco-IOS-XE-cdp YANG model supported by the device?

Si es compatible porque tiene las capacidades para monitorear y configurar el archive de forma remota.



The screenshot shows a code editor with a Python script named `lab27.yazmin.py` and a terminal window displaying the output of the script. The script connects to a device at `10.10.20.48` using the `ncclient` library and lists the supported YANG models (capabilities) from the device.

```
2 Autor: Rincon Ulloa Yazmin Elizabeth
3 Descripcion: YANG models
4 Fecha: Viernes 01 de Diciembre del 2023
5 """
6
7 import ncclient
8 from ncclient import manager
9
10 m = manager.connect(
11     host = '10.10.20.48',
12     port = 830,
13     username = "developer",
14     password = "Cisco12345",
15     hostkey_verify = False
16 )
```

The terminal output lists the following capabilities:

```
urn:ietf:params:xml:ns:yang:smiv2:RFC1155-SMI?module=RFC1155-SMI
urn:ietf:params:xml:ns:yang:smiv2:RFC1213-MIB?module=RFC1213-MIB
urn:ietf:params:xml:ns:yang:smiv2:RFC1315-MIB?module=RFC1315-MIB
urn:ietf:params:xml:ns:yang:smiv2:RMON-MIB?module=RMON-MIB&revision=2000-05-11
urn:ietf:params:xml:ns:yang:smiv2:RMON2-MIB?module=RMON2-MIB&revision=1996-05-27
urn:ietf:params:xml:ns:yang:smiv2:RSVP-MIB?module=RSVP-MIB&revision=1998-08-25
0-14
urn:ietf:params:xml:ns:yang:smiv2:SNMP-PROXY-MIB?module=SNMP-PROXY-MIB&revision=2002-10-14
urn:ietf:params:xml:ns:yang:smiv2:SNMP-TARGET-MIB?module=SNMP-TARGET-MIB&revision=1998-08-04
urn:ietf:params:xml:ns:yang:smiv2:SNMPv2-MIB?module=SNMPv2-MIB&revision=2002-10-16
urn:ietf:params:xml:ns:yang:smiv2:SNMPv2-TC?module=SNMPv2-TC
urn:ietf:params:xml:ns:yang:smiv2:SONET-MIB?module=SONET-MIB&revision=2003-08-11
urn:ietf:params:xml:ns:yang:smiv2:TCP-MIB?module=TCP-MIB&revision=2005-02-18
urn:ietf:params:xml:ns:yang:smiv2:TOKEN-RING-RMON-MIB?module=TOKEN-RING-RMON-MIB
urn:ietf:params:xml:ns:yang:smiv2:TOKENRING-MIB?module=TOKENRING-MIB&revision=1994-10-23
urn:ietf:params:xml:ns:yang:smiv2:TUNNEL-MIB?module=TUNNEL-MIB&revision=2005-05-16
urn:ietf:params:xml:ns:yang:smiv2:UDP-MIB?module=UDP-MIB&revision=2005-05-20
urn:ietf:params:xml:ns:yang:smiv2:VPN-TC-STD-MIB?module=VPN-TC-STD-MIB&revision=2005-11-15

urn:ietf:params:netconf:capability:notification:1.1
```

The terminal prompt is `PS C:\Users\azray\Documents\cuarto\Programacion\UNIDAD_3>`.