

Lab – NETCONF w/Python: List Capabilities

Nombre:	Montserrat Orduña Basaldua
Num Control:	1222100895

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.

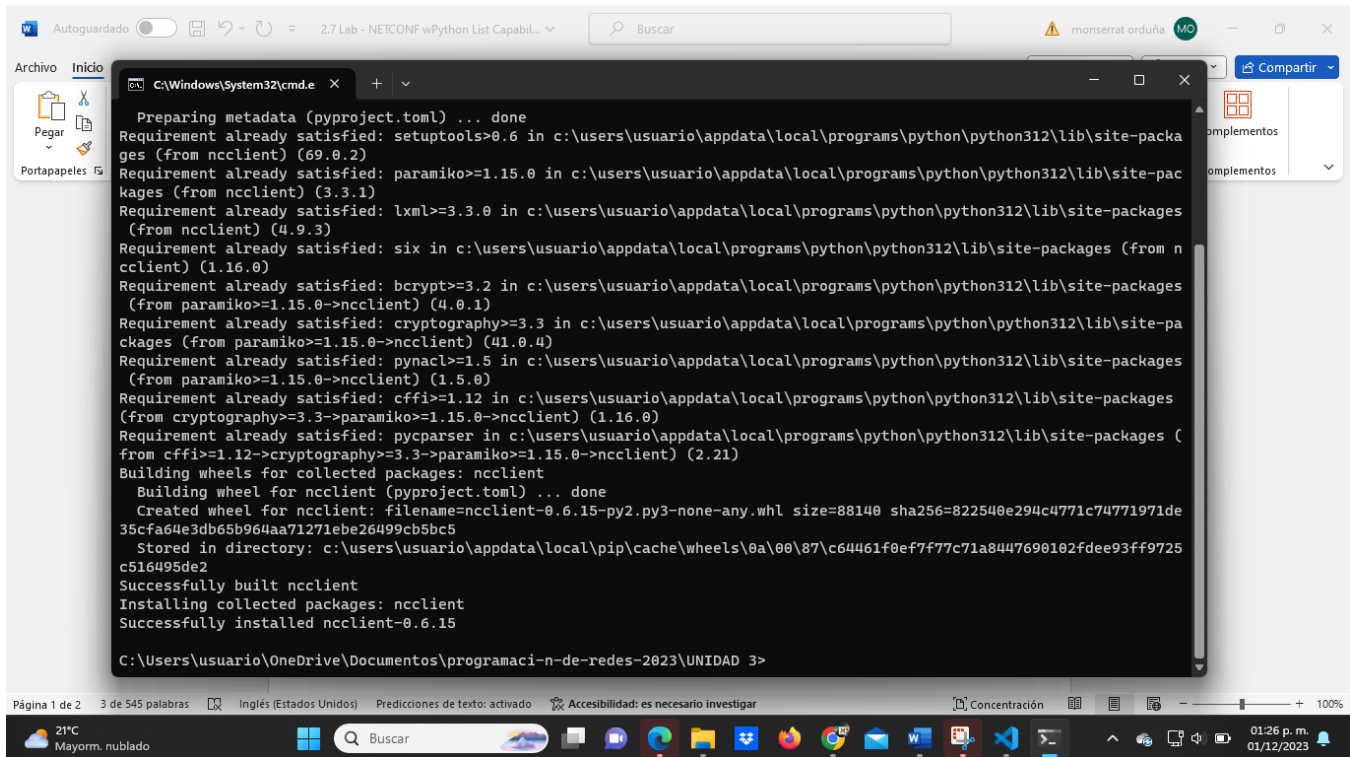
Explore the ncclient module on the project GitHub repository: <https://github.com/ncclient/ncclient>

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
```

Lab – NETCONF w/Python: List Capabilities

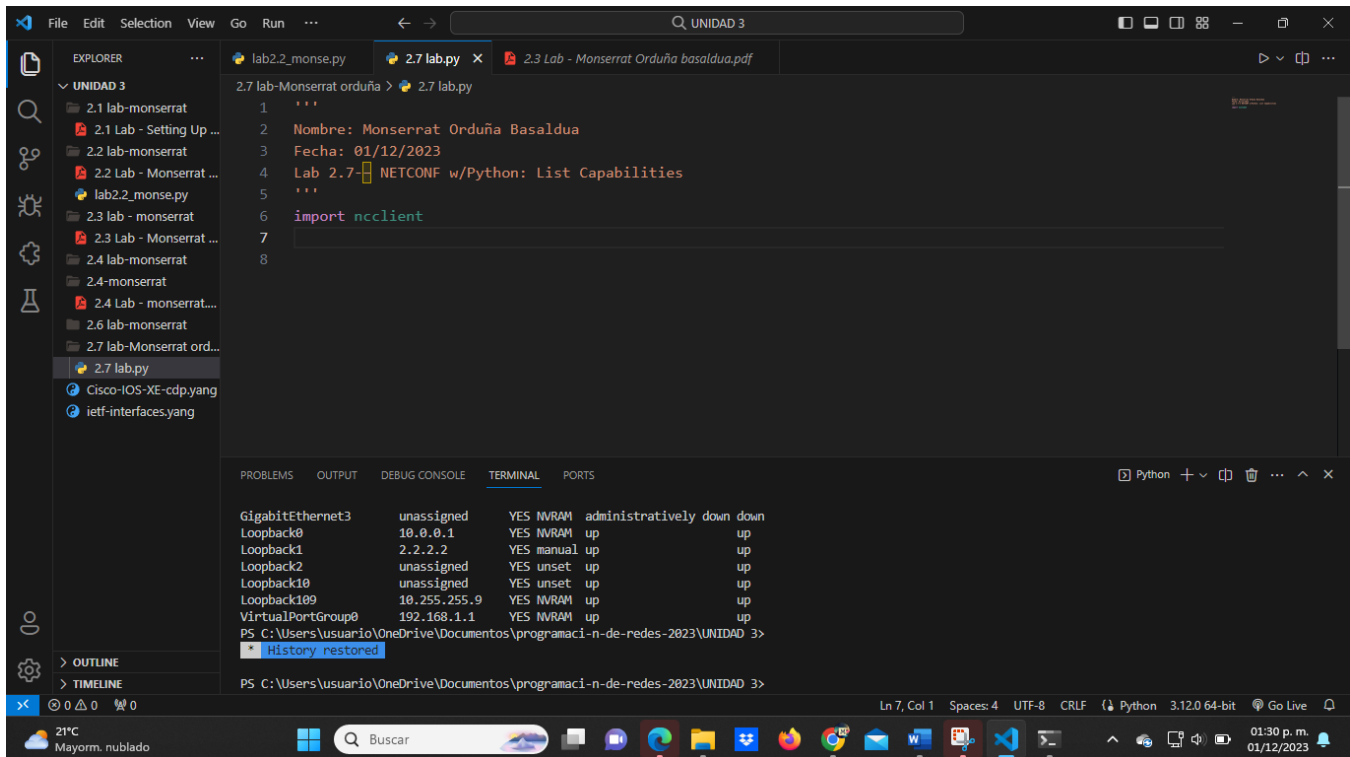


```
C:\Windows\System32\cmd.exe
Preparing metadata (pyproject.toml) ... done
Requirement already satisfied: setuptools>=0.6 in c:\users\usuario\appdata\local\programs\python\python312\lib\site-packages (from ncclient) (69.0.2)
Requirement already satisfied: paramiko>=1.15.0 in c:\users\usuario\appdata\local\programs\python\python312\lib\site-packages (from ncclient) (3.3.1)
Requirement already satisfied: lxml>=3.3.0 in c:\users\usuario\appdata\local\programs\python\python312\lib\site-packages (from ncclient) (4.9.3)
Requirement already satisfied: six in c:\users\usuario\appdata\local\programs\python\python312\lib\site-packages (from ncclient) (1.16.0)
Requirement already satisfied: bcrypt>=3.2 in c:\users\usuario\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\usuario\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\usuario\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\usuario\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\usuario\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-py2.py3-none-any.whl size=88140 sha256=822540e294c4771c74771971de35cf64e3db65b964aa71271ebe26499cb5bc5
Stored in directory: c:\users\usuario\appdata\local\pip\cache\wheels\0a\00\87\c64461f0ef777c71a8447690102fdee93ff9725c516495de2
Successfully built ncclient
Installing collected packages: ncclient
Successfully installed ncclient-0.6.15

C:\Users\usuario\OneDrive\Documentos\programaci-n-de-redes-2023\UNIDAD 3>
```

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

```
import ncclient
```



```
2.7 lab-Monserrat orduña > 2.7 lab.py
1 '''
2 Nombre: Monserrat Orduña Basaldua
3 Fecha: 01/12/2023
4 Lab 2.7- NETCONF w/Python: List Capabilities
5 '''
6 import ncclient
7
8
```

PROBLEMS	OUTPUT	DEBUG CONSOLE	TERMINAL	PORTS
GigabitEthernet3	unassigned	YES NVRAM	administratively down	down
Loopback0	10.0.0.1	YES NVRAM	up	up
Loopback1	2.2.2.2	YES manual	up	up
Loopback2	unassigned	YES unset	up	up
Loopback10	unassigned	YES unset	up	up
Loopback109	10.255.255.9	YES NVRAM	up	up
VirtualPortGroup0	192.168.1.1	YES NVRAM	up	up

```
PS C:\Users\usuario\OneDrive\Documentos\programaci-n-de-redes-2023\UNIDAD 3>
* History restored
PS C:\Users\usuario\OneDrive\Documentos\programaci-n-de-redes-2023\UNIDAD 3>
```

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

Connect to IOS XE's NETCONF service using ncclient.

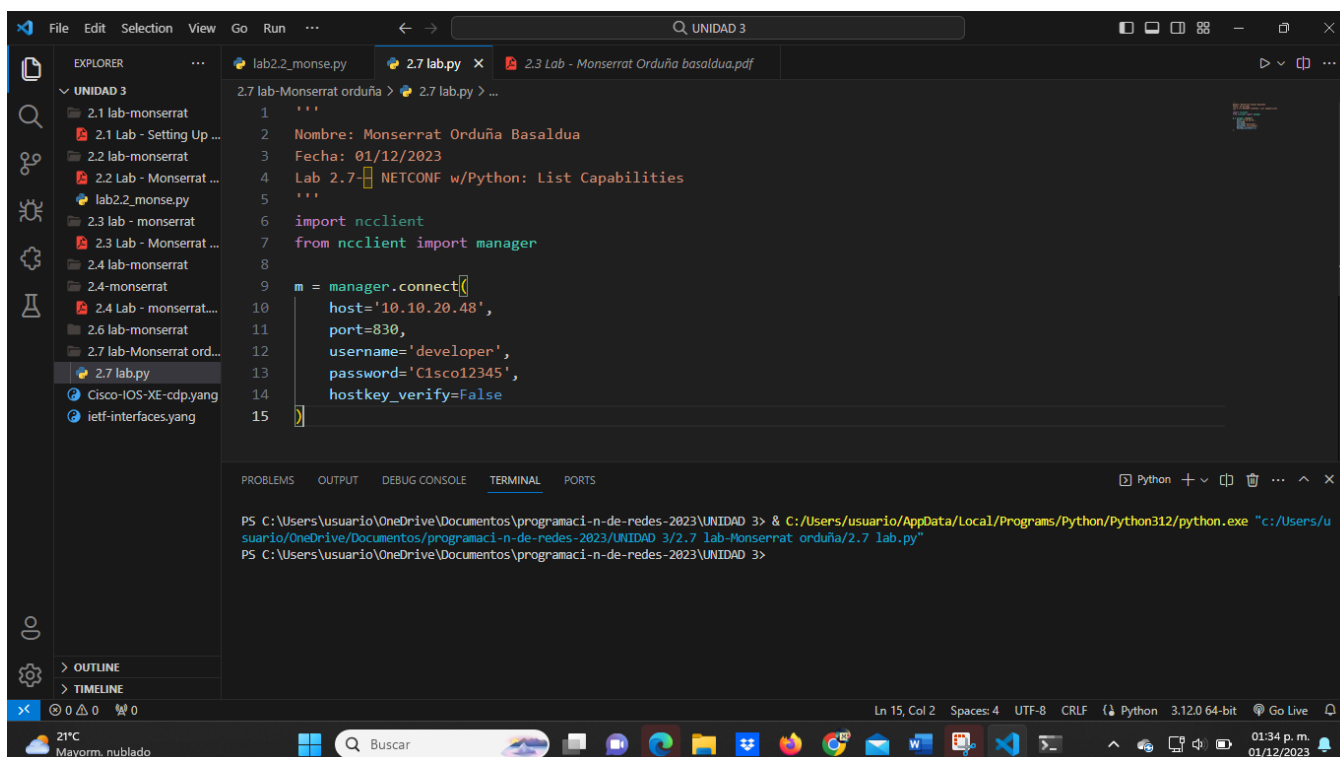
The ncclient module provides a “manager” class with “connect ()” function to setup the remote NETCONF connection. After a successful connection, the returned object represents the NETCONF connection to the remote device.

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

```
from ncclient import manager
```

- 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)

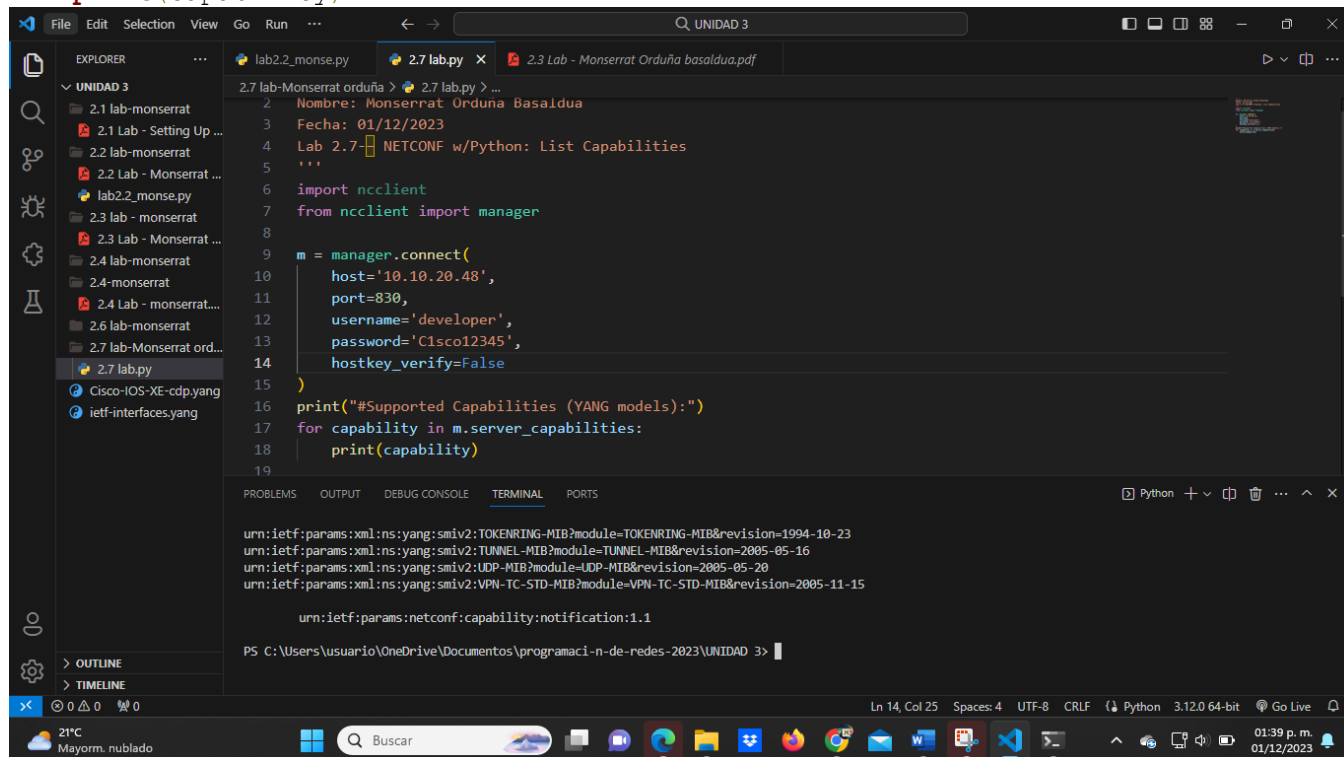
- `password` – remote ssh password (in this lab “cisco123!” for that was setup in the IOS XE VM)
- `hostkey_verify` – whether to verify the ssh fingerprint (in lab it is safe to set to False, in production environments you should always verify the ssh fingerprints)

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)
```



The screenshot shows a VS Code editor window with a file explorer on the left and a terminal at the bottom. The file explorer shows a project named 'UNIDAD 3' with several files, including '2.7 lab.py'. The terminal shows the output of the Python script, which lists the supported YANG models (capabilities) of the Cisco IOS XE device. The output includes several URNs for different YANG models, such as `urn:ietf:params:xml:ns:yang:smiv2:TOKENRING-MIB?module=TOKENRING-MIB&revision=1994-10-23`, and a notification capability `urn:ietf:params:netconf:capability:notification:1.1`.

```
2.7 lab-Monserrat orduña > 2.7 lab.py > ...
2 Nombre: Monserrat Orduña Basaldúa
3 Fecha: 01/12/2023
4 Lab 2.7- NETCONF w/Python: List Capabilities
5 ...
6 import ncclient
7 from ncclient import manager
8
9 m = manager.connect(
10     host='10.10.20.48',
11     port=830,
12     username='developer',
13     password='Cisco12345',
14     hostkey_verify=False
15 )
16 print("#Supported Capabilities (YANG models):")
17 for capability in m.server_capabilities:
18     print(capability)
19
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
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

PS C:\Users\usuario\OneDrive\Documentos\programaci-n-de-redes-2023\UNIDAD 3>
```

- Execute the Python script file to see the results.
- Is the Cisco-IOS-XE-cdp YANG model supported by the device?
Si es compatible ya que es un identificado de capacidad utilizado en el contexto del protocolo NETCONF permite el monitoreo en los dispositivos.