Networking Academy

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.

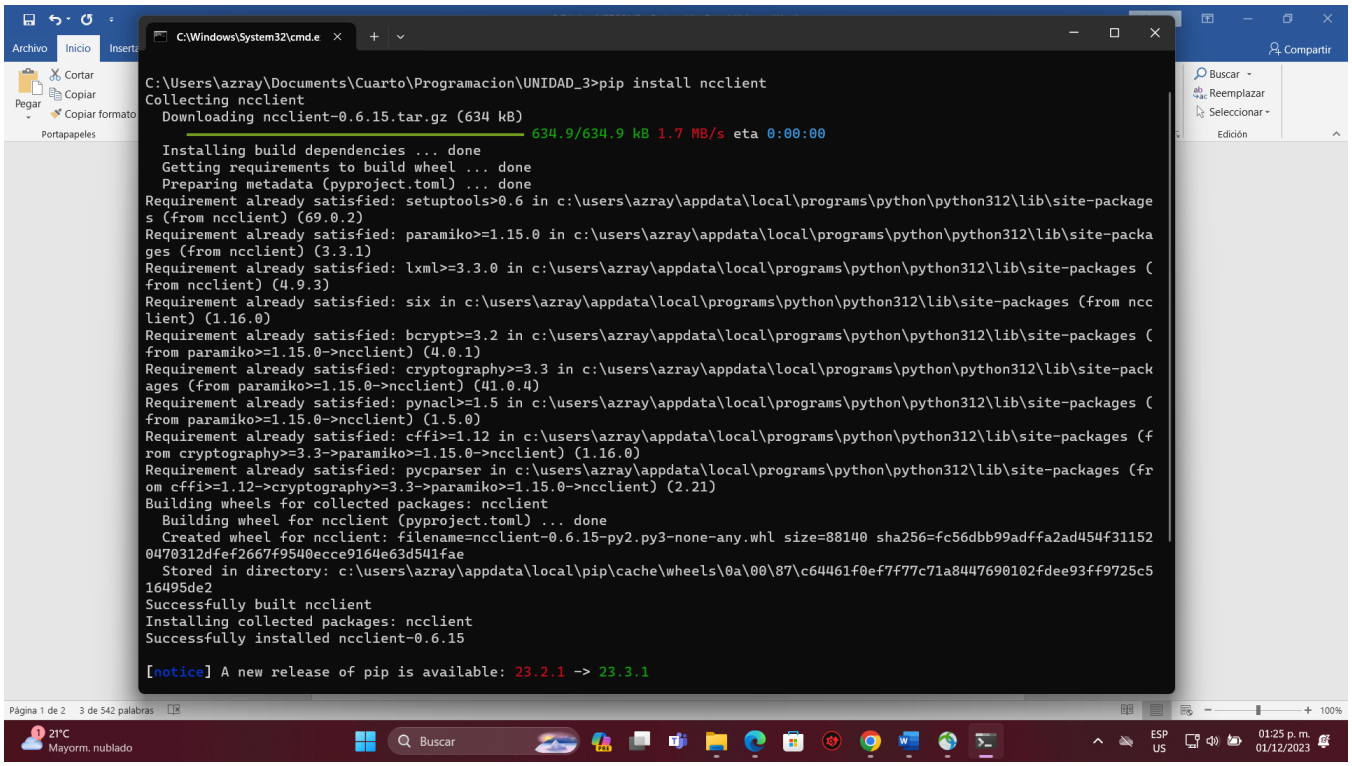
Explore the ncclient module on the project GitHub repository: https://github.com/ncclient/ncclient

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. 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.

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:

```
from ncclient import manager
```

c. 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

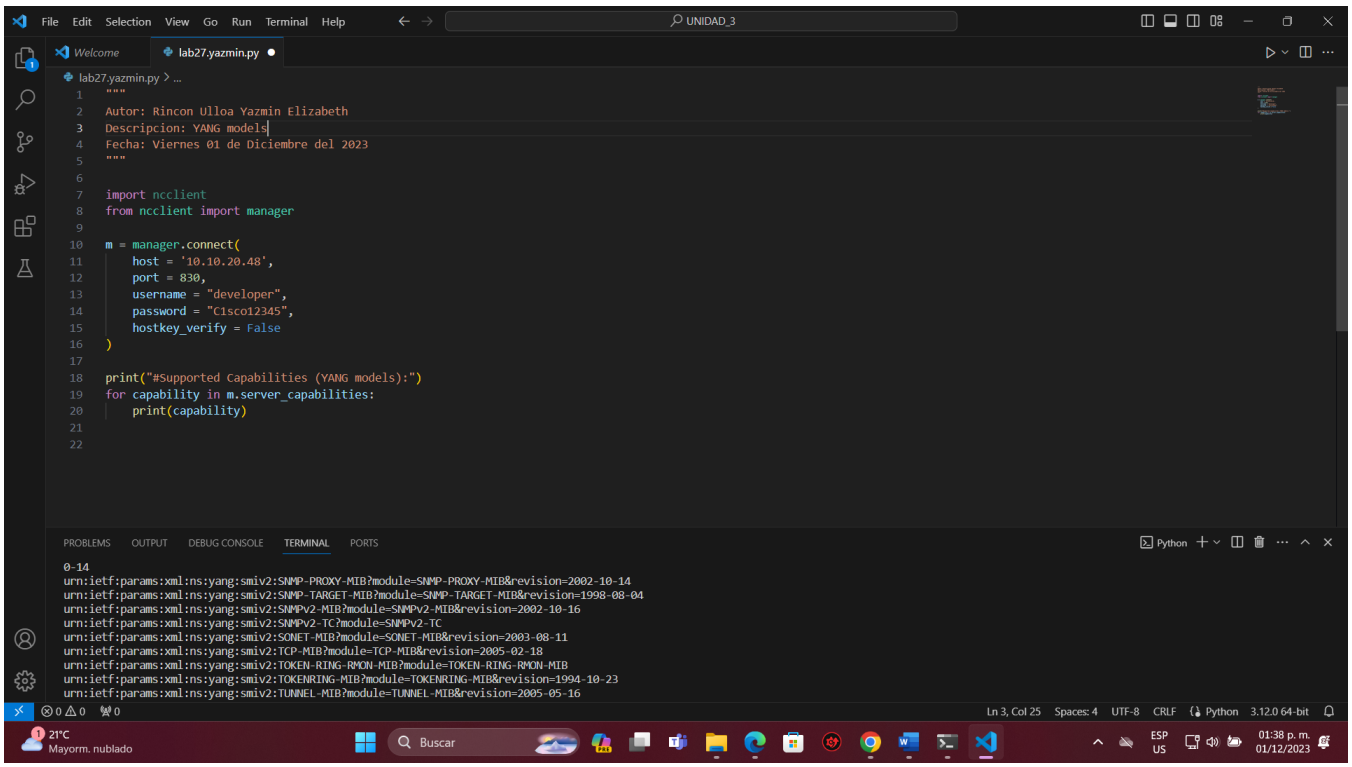
Send show commands and display the output

a. 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.

b. 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)
```

c. Execute the Python script file to see the results.



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.