
nic.br
Brazilian Network
Information Center
cgi.br
Brazilian Internet
Steering Committee
registro.br
cert.br
cetic.br
ceptro.br
ceweb.br
ix.br

WTR – POP BA
Curso Automação

William Prado
IX.br Engineering

nic.br

Who I am?

William Prado

Engenheiro de Computação – Instituto Nacional de Telecomunicações – INATEL;

12 anos de experiência em Telecomunicações e TI;

Engenheiro de Redes e Automação no IX.br (Engenharia);

Casado e Pai do Miguel e do José;

www.linkedin.com/in/william-prado-20579bb2

Agenda

01 Laboratório

02 Python

03 Netmiko

04 Napalm

05 Scrapli

06 NETCONF

07 NetBox

08 Nornir

09 Desafio

Laboratório

Simulador de Rede: PNETLAB https://www.pnetlab.com/pages/main

Automation Station: Debian12 (https://drive.google.com/drive/folders/1nugVe3ZaO0p6yrMJgHPEgGACOCAl1Ftn)

Netbox Docker: https://github.com/netbox-community/netbox-docker

Python 3.11.2 | Python 3.10 (python3 –version) | Python 3.10

IOSXR (xrv9k-fullk9-x-7.6.1)

Laboratório

Acessando a estação de automação:

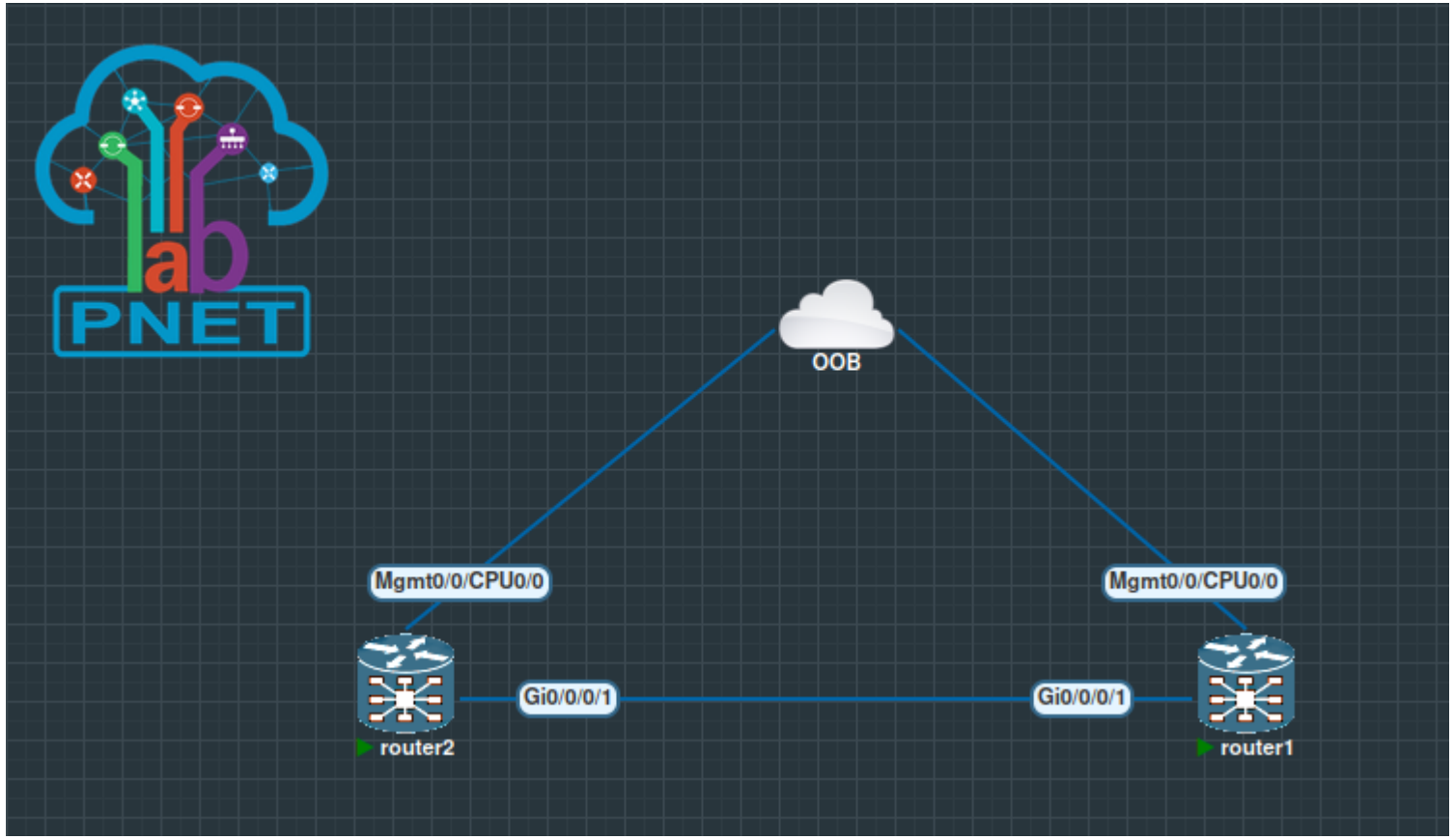
OBS: Verificar qual IP a VM pegou na rede em Bridge

user: wtr
password: wtr
root password: automation

```
root@william:/home/oem# ssh wtr@192.168.1.6
wtr@192.168.1.6's password:
Linux automation 6.1.0-23-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.99-1 (2024-07-15) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Aug  6 14:06:26 2024 from 192.168.1.5
wtr@automation:~$ su -
Senha:
root@automation:~#
```

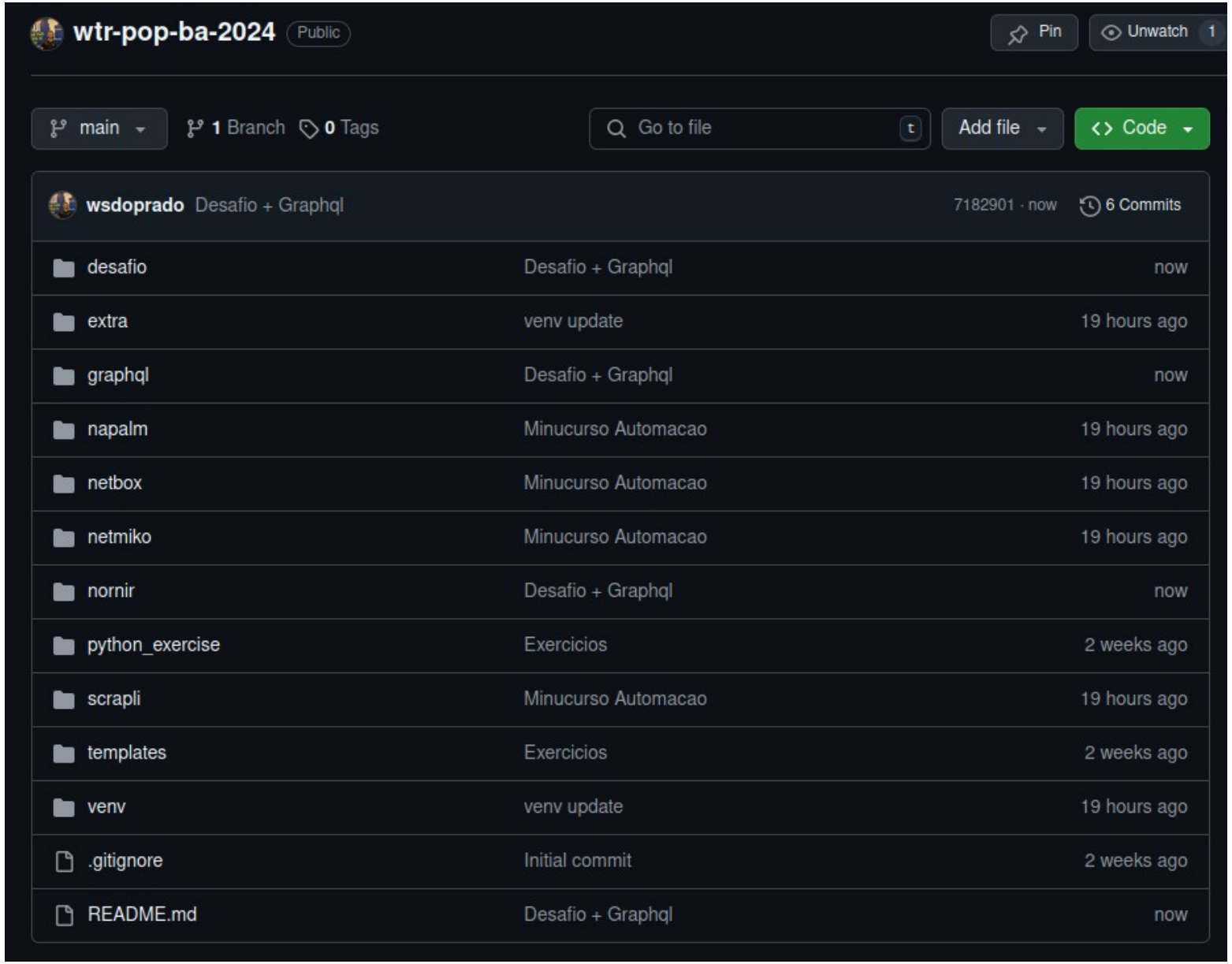
https://github.com/ktbyers/netmiko

Topologia



2x – Cisco IOSXR

Github



https://github.com/wsdoprado/wtr-pop-ba-2024.git

Clonado o Github do Curso:

```
root@william:/opt# git clone https://github.com/wsdoprado/wtr-pop-ba-2024.git
Cloning into 'wtr-pop-ba-2024'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (4/4), done.
```

Acessando o ambiente virtual do Python (venv):

```
root@william:/opt/wtr-pop-ba-2024# source venv/bin/activate
(venv) root@william:/opt/wtr-pop-ba-2024#
```

Agenda

02 Python

03 Netmiko

04 Napalm

05 Scrapli

06 NETCONF

07 NetBox

08 Nornir

09 Desafio

python™

https://www.python.org/about/gettingstarted/

Python – exemplo1.py (Tipo de Dados)

python_exercise > exemplo1.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

#String
nome = "William Stpehan do Prado"

#Inteiro
idade = 32

#Float
altura = 1.82

#Boolean
futebol = True

#mostrar no terminal o conteudo da variavel nome e idade
print(f"Nome: {nome} - Idade: {idade} - Altura: {altura} - Futebol: {futebol}")

#mostrar no terminal o tipo da variavel nome e idade
print(f"type nome: {type(nome)} - type idade: {type(idade)} - type altura: {type(altura)} - type futebol: {type(futebol)}")

```

Python – exemplo2.py (IN/OUT de Dados)

python_exercise > exemplo2.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

#Entrada de Dados
nome = input("Qual seu nome?")

#Entrada de Dados
salario = input("Qual seu Salário?")

#Mostrar Dados no Terminal
print(f"Nome: {nome} - Salário: {salario} R$")
```

Python – exemplo3.py (Lista, Loop e Validação de Dados)

python_exercise > exemplo3.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

aluno1 = "William Prado"
aluno2 = "Miguel Prado"
aluno3 = "Jose Prado"
aluno4 = "Eduardo Reis"
aluno5 = "Carlos Andre"
aluno6 = "Pedro Elias"

#Lista de Strings
lista_alunos = [aluno1,aluno2,aluno3,aluno4,aluno5]
#index              0      1      2      3      4

#Mostrar Lista no Terminal
print(lista_alunos)

#Mostrar Lista no Terminal no index 4
print(lista_alunos[4])

#Loop de Repetição na Lista
for aluno in lista_alunos:
    print(aluno)

#Validação
if aluno6 in lista_alunos:
    print("Aluno Encontrado")
else:
    print(f"Aluno: {aluno6} não encontrado na lista")
```

Python – exemplo4.py (Dicionários e Listas)

python_exercise > exemplo4.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

#Dicionário - dict
dados_aluno1 = {"nome": "William Prado", "idade": 32, "altura": 1.82, "casado": True, "filhos": ["Miguel","José"]}

#Dicionário - dict
dados_aluno2 = {"nome": "Eduardo Pereira", "idade": 40, "altura": 1.75, "casado": False, "filhos": []}

#Mostrar Dados no Terminal
print(dados_aluno1)
print(dados_aluno2)

print(dados_aluno1['filhos'])
print(dados_aluno2['filhos'])

#Lista de Dicionários
lista_alunos = [dados_aluno1,dados_aluno2]

print(lista_alunos)

```

Python – exemplo5.py (Dicionários, Listas e Loop)

python_exercise > exemplo5.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

#Dicionário - dict
filho1_aluno1 = {"nome": "Miguel Guedes", "idade": 5, "altura": 0.80, "casado": False}
filho2_aluno1 = {"nome": "José Guedes", "idade": 2, "altura": 0.60, "casado": False}

#Dicionário - dict + Listas - list
dados_aluno1 = {"nome": "William Prado", "idade": 32, "altura": 1.82, "casado": True, "filhos": [filho1_aluno1,filho2_aluno1]}
dados_aluno2 = {"nome": "Eduardo Pereira", "idade": 40, "altura": 1.75, "casado": False, "filhos": []}

#Lista
lista_alunos = [dados_aluno1,dados_aluno2]

#Mostrar dados no Terminal
print(lista_alunos)

#Loop na Lista alunos
for aluno in lista_alunos:
    print(f"Aluno: {aluno['nome']} - Idade: {aluno['idade']} - Altura: {aluno['altura']}")
    for filho in aluno['filhos']:
        print(f"Nome filho: {filho['nome']} - Idade: {filho['idade']}")


```

Python – exemplo5.py – OUTPUT JSON

```
[
   {
      'nome': 'William Prado',
      'idade': 32, 'altura': 1.82,
      'casado': True,
      'filhos': [
                  {
                     'nome': 'Miguel Guedes',
                     'idade': 5, 'altura': 0.8,
                     'casado': False
                  },
                  {
                     'nome': 'José Guedes',
                     'idade': 2,
                     'altura': 0.6,
                     'casado': False
                  }
               ]
   }
]
```

JSON

Formato de arquivo

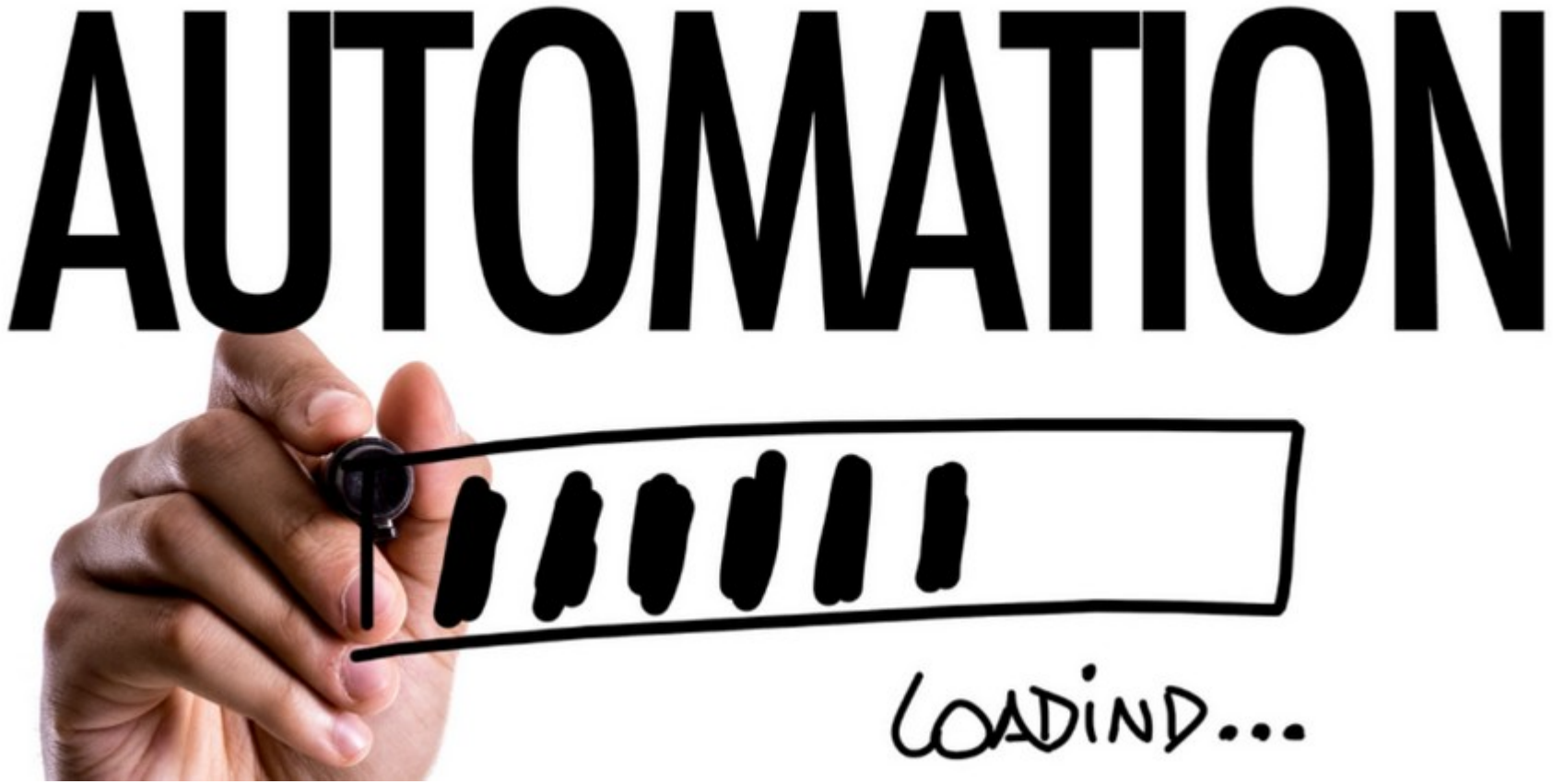
Em computação, JSON, um acrônimo de JavaScript Object Notation, é um formato compacto, de padrão aberto independente, de troca de dados simples e rápida entre sistemas, especificado por Douglas Crockford em 2000, que utiliza texto legível a humanos, no formato atributo-valor. Wikipédia

Lançamento: 2002 (21–22 anos)

Página oficial: json.org/json-pt.html

Última versão: dezembro de 2017; há 6 anos

Variante de: JavaScript


AUTOMATION
LOADING...

Agenda

04 Napalm

07 NetBox

05 Scrapli

08 Nornir

03 Netmiko

06 NETCONF

09 Desafio


NETMIKO

Netmiko - o que é?

Netmiko é uma biblioteca Python que simplifica o processo de conexão a dispositivos de rede multi-vendor usando o protocolo SSH.

Netmiko abstrai muitas complexidades, fornecendo uma biblioteca Python com um conjunto de métodos fáceis de usar:

- **send_command();**
- **send_config_set();**
- **commit();**
- **send_config_from_file();**
- **send_command_timing();**

Alguns casos de uso no qual o Netmiko pode ser útil:

- **Automatizar Backup de equipamentos;**
- **Aplicar um (ou mais) comando e validar o resultado (“versão de software? Modelo de uma placa?”);**
- **Automatizar o processo de troubleshoot aplicando vários comandos;**
- **Transferência de Arquivos via SCP;**

Features:

- **Análise Estruturada** - suporta por meio de bibliotecas de análise TTP, TextFSM e Genie;
- **Suporte Multi-vendor** – aceita vários fornecedores de equipamentos (+ 106 vendors);
- **Configuração de Dispositivo** – fornece métodos para aplicar ou ler configuração nos equipamentos;

https://github.com/ktbyers/netmiko

Netmiko - Instalação

Netmiko pode ser instalado pelo gerenciador de pacotes do python – PIP:

Recomendado usar ambiente virtual do Python:

python3 -m venv venv
source venv/bin/activate
pip3 install netmiko

```
(venv) root@automation:~/course# pip3 install netmiko
Collecting netmiko
  Downloading netmiko-4.4.0-py3-none-any.whl (232 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 232.2/232.2 kB 7.2 MB/s eta 0:00:00
Collecting cffi>=1.17.0rc1
  Downloading cffi-1.17.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (466 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 466.9/466.9 kB 15.5 MB/s eta 0:00:00
Collecting ntc-templates>=3.1.0
  Downloading ntc_templates-6.0.0-py3-none-any.whl (460 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 460.6/460.6 kB 15.3 MB/s eta 0:00:00
Collecting paramiko>=2.9.5
  Downloading paramiko-3.4.0-py3-none-any.whl (225 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 225.9/225.9 kB 16.1 MB/s eta 0:00:00
Collecting pyserial>=3.3
  Downloading pyserial-3.5-py2.py3-none-any.whl (90 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 90.6/90.6 kB 11.5 MB/s eta 0:00:00
Collecting pyyaml>=5.3
  Downloading PyYAML-6.0.2-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (762 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 763.0/763.0 kB 27.4 MB/s eta 0:00:00
Collecting scp>=0.13.6
  Downloading scp-0.15.0-py2.py3-none-any.whl (8.8 kB)
Requirement already satisfied: setuptools>=65.0.0 in ./venv/lib/python3.11/site-packages (from netmiko) (66.1.1)
Collecting textfsm>=1.1.3
  Downloading textfsm-1.1.3-py2.py3-none-any.whl (44 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 44.7/44.7 kB 5.2 MB/s eta 0:00:00
Collecting pycparser
  Downloading pycparser-2.22-py3-none-any.whl (117 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 117.6/117.6 kB 14.7 MB/s eta 0:00:00
Collecting bcrypt>=3.2
  Downloading bcrypt-4.2.0-cp39-abi3-manylinux_2_28_x86_64.whl (273 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 273.8/273.8 kB 25.2 MB/s eta 0:00:00
Collecting cryptography>=3.3
  Downloading cryptography-43.0.0-cp39-abi3-manylinux_2_28_x86_64.whl (4.0 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.0/4.0 MB 51.5 MB/s eta 0:00:00
Collecting pynacl>=1.5
  Downloading PyNaCl-1.5.0-cp36-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.manylinux_2_24_x86_64.whl (856 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 856.7/856.7 kB 60.5 MB/s eta 0:00:00
Collecting six
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Collecting future
  Downloading future-1.0.0-py3-none-any.whl (491 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 491.3/491.3 kB 47.3 MB/s eta 0:00:00
Installing collected packages: pyserial, six, pyyaml, pycparser, future, bcrypt, textfsm, cffi, pynacl, ntc-templates, cryptography, paramiko, scp, netmiko
Successfully installed bcrypt-4.2.0 cffi-1.17.0 cryptography-43.0.0 future-1.0.0 netmiko-4.4.0 ntc-templates-6.0.0 paramiko-3.4.0 pycparser-2.22 pynacl-1.5.0 pyserial-3.5 pyyaml-6.0.2 scp-0.
15.0 six-1.16.0 textfsm-1.1.3
(venv) root@automation:~/course#
```

Netmiko – exercicio1.py (send_command)

netmiko > exercicio1.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

from netmiko import ConnectHandler
from rich import print

#dict com os dados do device para acesso
router1 = {
    "device_type": "cisco_xr",
    "host": "192.168.246.95",
    "username": "wtr",
    "password": "wtr"
}

try:
    #conexão com o router1 e router2
    r1_connection = ConnectHandler(**router1)

    #executar o comando show version no router1 e router2 e mostrar o resultado
    results = r1_connection.send_command('show version')

    print(type(results))

    #finalizar a conexão com o router1 e router2
    r1_connection.disconnect()

except Exception as err:
    print(err)

```

Netmiko – Resultado - exercicio1.py

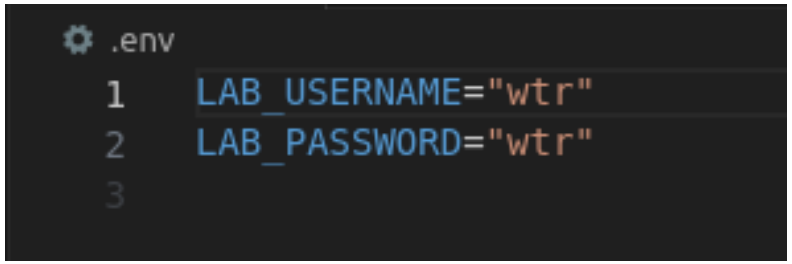
```
Tue Aug 20 17:12:23.565 UTC
Cisco IOS XR Software, Version 7.6.1
Copyright (c) 2013-2022 by Cisco Systems, Inc.

Build Information:
 Built By     : ingunawa
 Built On     : Sat Mar 26 19:10:06 PDT 2022
 Built Host   : iox-ucs-072
 Workspace    : /auto/srcarchive17/prod/7.6.1/xrv9k/ws
 Version      : 7.6.1
 Location     : /opt/cisco/XR/packages/
 Label        : 7.6.1-0

cisco IOS-XRv 9000 () processor
System uptime is 3 days 21 hours 39 minutes


Tue Aug 20 17:12:23.620 UTC
Cisco IOS XR Software, Version 7.6.1
Copyright (c) 2013-2022 by Cisco Systems, Inc.

Build Information:
 Built By     : ingunawa
 Built On     : Sat Mar 26 19:10:06 PDT 2022
 Built Host   : iox-ucs-072
 Workspace    : /auto/srcarchive17/prod/7.6.1/xrv9k/ws
 Version      : 7.6.1
 Location     : /opt/cisco/XR/packages/
 Label        : 7.6.1-0

cisco IOS-XRv 9000 () processor
System uptime is 3 days 21 hours 53 minutes
```

```python
try:
    #conexão com o router1 e router2
    r1_connection = ConnectHandler(**router1)

    #executar o comando show version no router1 e router2 e mostrar o resultado
    results = r1_connection.send_command('show version')

    print(type(results))

    #finalizar a conexão com o router1 e router2
    r1_connection.disconnect()

except Exception as err:
    print(err)
```

```
(venv) root@william:/opt/wtr-pop-ba-2024/netmiko# python3.10 exercicio1.py
<class 'str'>
(venv) root@william:/opt/wtr-pop-ba-2024/netmiko#
(venv) root@william:/opt/wtr-pop-ba-2024/netmiko#
(venv) root@william:/opt/wtr-pop-ba-2024/netmiko#
(venv) root@william:/opt/wtr-pop-ba-2024/netmiko#
```

Netmiko – Usando dotenv (.env) para credenciais:

Recomendado colocar as credenciais de acesso (dados sensíveis) em um arquivo .env:



No desenvolvimento do Script é possível usar as credenciais através das instruções abaixo:

```
from dotenv import load_dotenv

load_dotenv() # load environment variables from .env
```

Netmiko – exercicio2.py (.env)

netmiko > exercicio2.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

import os
from dotenv import load_dotenv
from netmiko import ConnectHandler
from rich import print

#carregar as variáveis definidas no .env
load_dotenv()

#dict com os dados do device para acesso
router1 = {"device_type": "cisco_xr","host": "192.168.246.95","username": os.getenv("LAB_USERNAME"),"password": os.getenv("LAB_PASSWORD")}
router2 = {"device_type": "cisco_xr","host": "192.168.246.96","username": os.getenv("LAB_USERNAME"),"password": os.getenv("LAB_PASSWORD")}

try:
    #conexão com o router1 e router2
    r1_connection = ConnectHandler(**router1)
    r2_connection = ConnectHandler(**router2)

    #executar o comando show version no router1 e router2 e mostrar o resultado
    print(r1_connection.send_command('show version'))
    print(r2_connection.send_command('show version'))

    #finalizar a conexão com o router1 e router2
    r1_connection.disconnect()
    r2_connection.disconnect()

except Exception as err:
    print(err)

```

Netmiko – exercicio3.py (send_config_set)

netmiko > exercicio3.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

import os
from dotenv import load_dotenv
from netmiko import ConnectHandler
from rich import print

load_dotenv()

router1 = {"device_type": "cisco_xr","host": "192.168.246.95","username": os.getenv("LAB_USERNAME"),"password": os.getenv("LAB_PASSWORD")}
router2 = {"device_type": "cisco_xr","host": "192.168.246.96","username": os.getenv("LAB_USERNAME"),"password": os.getenv("LAB_PASSWORD")}

try:
    #conexão com o router1 e router2
    r1_connection = ConnectHandler(**router1)
    r2_connection = ConnectHandler(**router2)

    commands_router1 = ["hostname R1"]
    commands_router2 = ["hostname R2"]

    #enviar a lista de comandos para o router1 e realizar o commit
    print(r1_connection.send_config_set(commands_router1))
    print(r1_connection.commit())

    #enviar a lista de comandos para o router2 e realizar o commit
    print(r2_connection.send_config_set(commands_router2))
    print(r2_connection.commit())

    #finalizar a conexão com o router1 e router2
    r1_connection.disconnect()
    r2_connection.disconnect()

except Exception as err:
    print(err)
```

Netmiko – Resultado - exercicio3.py

```
configure terminal

Tue Aug 20 22:08:39.291 UTC
RP/0/RP0/CPU0:ios(config)#hostname R1

RP/0/RP0/CPU0:ios(config)#
commit

Tue Aug 20 22:08:39.819 UTC
RP/0/RP0/CPU0:R1(config)#
configure terminal

Tue Aug 20 22:08:40.485 UTC
RP/0/RP0/CPU0:ios(config)#hostname R2

RP/0/RP0/CPU0:ios(config)#
commit

Tue Aug 20 22:08:40.978 UTC
RP/0/RP0/CPU0:R2(config)#
```

Netmiko – exercicio4.py (send_config_set)

netmiko > exercicio4.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

import os
from dotenv import load_dotenv
from netmiko import ConnectHandler
from rich import print

load_dotenv()

router1 = {"device_type": "cisco_xr","host": "192.168.246.95","username": os.getenv("LAB_USERNAME"),"password": os.getenv("LAB_PASSWORD")}
router2 = {"device_type": "cisco_xr","host": "192.168.246.96","username": os.getenv("LAB_USERNAME"),"password": os.getenv("LAB_PASSWORD")}

try:
    #conexão com o router1 e router2
    r1_connection = ConnectHandler(**router1)
    r2_connection = ConnectHandler(**router2)

    commands_router1 = ["interface gigabitEthernet 0/0/0/1","ipv4 address 192.168.100.1 255.255.255.252","no shutdown"]
    commands_router2 = ["interface gigabitEthernet 0/0/0/1","ipv4 address 192.168.100.2 255.255.255.252","no shutdown"]

    #enviar a lista de comandos para o router1 e realizar o commit
    print(r1_connection.send_config_set(commands_router1))
    print(r1_connection.commit())

    #enviar a lista de comandos para o router2 e realizar o commit
    print(r2_connection.send_config_set(commands_router2))
    print(r2_connection.commit())

    #finalizar a conexão com o router1 e router2
    r1_connection.disconnect()
    r2_connection.disconnect()

except Exception as err:
    print(err)

```

Netmiko – Resultado - exercicio4.py

```
configure terminal

Tue Aug 20 22:09:58.769 UTC
RP/0/RP0/CPU0:R1(config)#interface gigabitEthernet 0/0/0/1

RP/0/RP0/CPU0:R1(config-if)#ipv4 address 192.168.100.1 255.255.255.252

RP/0/RP0/CPU0:R1(config-if)#no shutdown

RP/0/RP0/CPU0:R1(config-if)#
commit

Tue Aug 20 22:09:59.448 UTC
RP/0/RP0/CPU0:R1(config-if)#
configure terminal

Tue Aug 20 22:09:59.822 UTC
RP/0/RP0/CPU0:R2(config)#interface gigabitEthernet 0/0/0/1

RP/0/RP0/CPU0:R2(config-if)#ipv4 address 192.168.100.2 255.255.255.252

RP/0/RP0/CPU0:R2(config-if)#no shutdown

RP/0/RP0/CPU0:R2(config-if)#
commit

Tue Aug 20 22:10:00.456 UTC
RP/0/RP0/CPU0:R2(config-if)#
```

Netmiko – Parsing: TTP, Genie e TextFSM

Output do comando: **show vlan**

```
VLAN  Name                                Status
----- ------------------------------------ --------
1     default                              active
100   VLAN100                              active
200   VLAN200                              active
300   VLAN300                              active
```

Se armazenar este OUTPUT em uma variável usando Netmiko, o conteúdo é uma string com quebras de linha, espaços e caracteres especiais.

Como pegar estes dados de VLAN_ID, VLAN_NAME e STATUS e gerar uma estrutura de dados organizada?

Netmiko – Parsing: TTP, Genie e TextFSM


```
[
    {
        "vlan_id": "1",
        "vlan_name": "default"
    },
    {
        "vlan_id": "100",
        "vlan_name": "VLAN100"
    },
    {
        "vlan_id": "200",
        "vlan_name": "VLAN200"
    },
    {
        "vlan_id": "300",
        "vlan_name": "VLAN300"
    }
]
```

Netmiko – TTP (Template Text Parser) – exercicio5.py

netmiko > exercicio5.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

import os
from dotenv import load_dotenv
from netmiko import ConnectHandler
from rich import print

load_dotenv()

router1 = {"device_type": "cisco_xr","host": "192.168.246.95","username": os.getenv("LAB_USERNAME"),"password": os.getenv("LAB_PASSWORD")}

try:
    r1_connection = ConnectHandler(**router1)

    print(r1_connection.send_command('show running-config interface'))

    r1_connection.disconnect()

except Exception as err:
    print(err)

```

Netmiko – TTP (Template Text Parser) – exercicio5.py

```
Fri Sep  6 19:37:09.759 UTC
interface MgmtEth0/RP0/CPU0/0
 ipv4 address 192.168.246.95 255.255.255.0
!
interface GigabitEthernet0/0/0/0
 shutdown
!
interface GigabitEthernet0/0/0/1
 ipv4 address 192.168.100.2 255.255.255.252
!
interface GigabitEthernet0/0/0/2
 shutdown
!
interface GigabitEthernet0/0/0/3
 shutdown
!
interface GigabitEthernet0/0/0/4
 shutdown
!
interface GigabitEthernet0/0/0/5
 shutdown
!
interface GigabitEthernet0/0/0/6
 shutdown
!
```

Netmiko – TTP (Template Text Parser) – exercicio6.py

netmiko > exercicio6.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

import os
from dotenv import load_dotenv
from netmiko import ConnectHandler
from rich import print

load_dotenv()

router1 = {"device_type": "cisco_xr","host": "192.168.246.95","username": os.getenv("LAB_USERNAME"),"password": os.getenv("LAB_PASSWORD")}

try:
    r1_connection = ConnectHandler(**router1)

    results = r1_connection.send_command('show running-config interface', use_ttp=True, ttp_template="./template.ttp")

    print(results)

    print(f'type results: {type(results)}')

    #finalizar a conexão com o PE1
    r1_connection.disconnect()

except Exception as err:
    print(err)

```

Netmiko – TTP (Template Text Parser) – template.ttp

Template

```
interface {{ interface }}
 ipv4 address {{ ip }} {{ mask }}
 description {{ description }}
```

Netmiko – TTP (Template Text Parser) – template.ttp

Template

```
interface {{ interface }}
 ipv4 address {{ ip }} {{ mask }}
 description {{ description }}
```

String para realizar o Parse

```
Wed Aug 21 17:37:33.660 UTC
interface MgmtEth0/RP0/CPU0/0
 ipv4 address 192.168.246.96 255.255.255.0
!
interface GigabitEthernet0/0/0/0
 shutdown
!
interface GigabitEthernet0/0/0/1
 ipv4 address 192.168.100.2 255.255.255.252
!
interface GigabitEthernet0/0/0/2
 shutdown
!
interface GigabitEthernet0/0/0/3
 shutdown
!
```

Netmiko – TTP (Template Text Parser) – template.ttp

Template

```
interface {{ interface }}
 ipv4 address {{ ip }} {{ mask }}
 description {{ description }}
```

String para realizar o Parse

```
Wed Aug 21 17:37:33.660 UTC
interface MgmtEth0/RP0/CPU0/0
 ipv4 address 192.168.246.96 255.255.255.0
!
interface GigabitEthernet0/0/0/0
 shutdown
!
interface GigabitEthernet0/0/0/1
 ipv4 address 192.168.100.2 255.255.255.252
!
interface GigabitEthernet0/0/0/2
 shutdown
!
interface GigabitEthernet0/0/0/3
 shutdown
!
```

Netmiko – TTP (Template Text Parser) - Output

```
[
    [
        [
            {'ip': '192.168.246.95', 'mask': '255.255.255.0', 'interface': 'MgmtEth0/RP0/CPU0/0'},
            {'interface': 'GigabitEthernet0/0/0/0'},
            {'ip': '192.168.100.1', 'mask': '255.255.255.252', 'interface': 'GigabitEthernet0/0/0/1'},
            {'interface': 'GigabitEthernet0/0/0/2'},
            {'interface': 'GigabitEthernet0/0/0/3'}
        ]
    ]
]
type results: <class 'list'>
```

Netmiko – TTP (Template Text Parser) - Output

```
[
    [
        [
            {'ip': '192.168.246.95', 'mask': '255.255.255.0', 'interface': 'MgmtEth0/RP0/CPU0/0'},
            {'interface': 'GigabitEthernet0/0/0/0'},
            {'ip': '192.168.100.1', 'mask': '255.255.255.252', 'interface': 'GigabitEthernet0/0/0/1'},
            {'interface': 'GigabitEthernet0/0/0/2'},
            {'interface': 'GigabitEthernet0/0/0/3'}
        ]
    ]
]
type results: <class 'list'>
```

```
Wed Aug 21 17:37:33.660 UTC
interface MgmtEth0/RP0/CPU0/0
 ipv4 address 192.168.246.96 255.255.255.0
!
interface GigabitEthernet0/0/0/0
 shutdown
!
interface GigabitEthernet0/0/0/1
 ipv4 address 192.168.100.2 255.255.255.252
!
interface GigabitEthernet0/0/0/2
 shutdown
!
interface GigabitEthernet0/0/0/3
 shutdown
!
```

Netmiko – TextFSM – exercicio7.py

netmiko > exercicio7.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

import os
from dotenv import load_dotenv
from netmiko import ConnectHandler
from rich import print

load_dotenv()

router1 = {"device_type": "cisco_xr","host": "192.168.246.95","username": os.getenv("LAB_USERNAME"),"password": os.getenv("LAB_PASSWORD")}

try:
    r1_connection = ConnectHandler(**router1)

    results1 = r1_connection.send_command('show ip int brief', use_textfsm=True)
    results2 = r1_connection.send_command('show version', use_textfsm=True)
    results3 = r1_connection.send_command('show arp', use_textfsm=True)
    results4 = r1_connection.send_command('ping 192.168.246.1', use_textfsm=True)

    print(results3)

    r1_connection.disconnect()

except Exception as err:
    print(err)

```

https://github.com/networktocode/ntc-templates/tree/master/ntc_templates/templates

Netmiko – TextFSM – exercicio7.py

netmiko > exercicio7.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

import os
from dotenv import load_dotenv
from netmiko import ConnectHandler
from rich import print

load_dotenv()

router1 = {"device_type": "cisco_xr","host": "192.168.246.95","username": os.getenv("LAB_USERNAME"),"password": os.getenv("LAB_PASSWORD")}

try:
    r1_connection = ConnectHandler(**router1)

    results1 = r1_connection.send_command('show ip int brief', use_textfsm=True)
    results2 = r1_connection.send_command('show version', use_textfsm=True)
    results3 = r1_connection.send_command('show arp', use_textfsm=True)
    results4 = r1_connection.send_command('ping 192.168.246.1', use_textfsm=True)

    print(results3)

    r1_connection.disconnect()

except Exception as err:
    print(err)

```

https://github.com/networktocode/ntc-templates/tree/master/ntc_templates/templates

Netmiko – TextFSM – exercicio7.py - Output

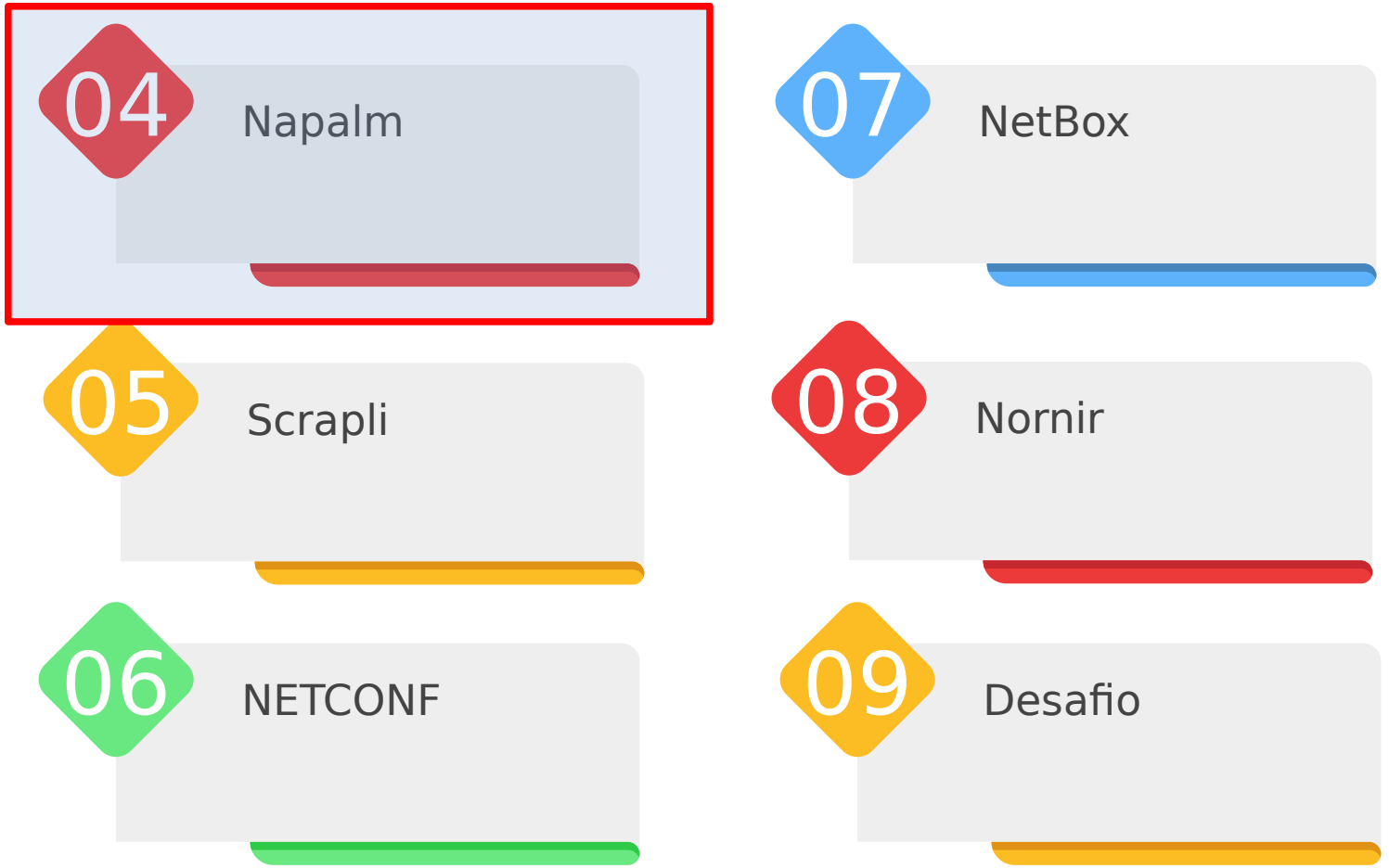
```
[
    {'interface': 'MgmtEth0/RP0/CPU0/0', 'ip_address': '192.168.246.95', 'status': 'Up', 'proto': 'Up', 'vrf': 'default'},
    {'interface': 'GigabitEthernet0/0/0/0', 'ip_address': 'unassigned', 'status': 'Shutdown', 'proto': 'Down', 'vrf': 'default'},
    {'interface': 'GigabitEthernet0/0/0/1', 'ip_address': '192.168.100.1', 'status': 'Up', 'proto': 'Up', 'vrf': 'default'},
    {'interface': 'GigabitEthernet0/0/0/2', 'ip_address': 'unassigned', 'status': 'Shutdown', 'proto': 'Down', 'vrf': 'default'},
    {'interface': 'GigabitEthernet0/0/0/3', 'ip_address': 'unassigned', 'status': 'Shutdown', 'proto': 'Down', 'vrf': 'default'}
]
```

```
[{'version': '7.6.1', 'uptime': '5 days 22 hours 9 minutes', 'location': '/opt/cisco/XR/packages/', 'hardware': 'IOS-XRv 9000', 'build_host': 'iox-ucs-072', 'label': '7.6.1-0'}]
```

```
[
    {'ip_address': '192.168.100.1', 'age': '-', 'mac_address': '50f3.4c00.1404', 'state': 'Interface', 'type': 'ARPA', 'interface': 'GigabitEthernet0/0/0/1', 'cpu': '0/0/CPU0'},
    {'ip_address': '192.168.100.2', 'age': '02:50:12', 'mac_address': '506a.f100.1104', 'state': 'Dynamic', 'type': 'ARPA', 'interface': 'GigabitEthernet0/0/0/1', 'cpu': '0/0/CPU0'},
    {'ip_address': '192.168.246.1', 'age': '00:00:00', 'mac_address': '66ad.b923.09e9', 'state': 'Dynamic', 'type': 'ARPA', 'interface': 'MgmtEth0/RP0/CPU0/0', 'cpu': '0/RP0/CPU0'},
    {'ip_address': '192.168.246.51', 'age': '03:00:48', 'mac_address': '0caf.31ed.f800', 'state': 'Dynamic', 'type': 'ARPA', 'interface': 'MgmtEth0/RP0/CPU0/0', 'cpu': '0/RP0/CPU0'},
    {'ip_address': '192.168.246.71', 'age': '00:00:20', 'mac_address': '0004.96ae.99f8', 'state': 'Dynamic', 'type': 'ARPA', 'interface': 'MgmtEth0/RP0/CPU0/0', 'cpu': '0/RP0/CPU0'},
    {'ip_address': '192.168.246.80', 'age': '00:58:17', 'mac_address': '3e25.2250.6564', 'state': 'Dynamic', 'type': 'ARPA', 'interface': 'MgmtEth0/RP0/CPU0/0', 'cpu': '0/RP0/CPU0'},
    {'ip_address': '192.168.246.95', 'age': '-', 'mac_address': '50f3.4c00.1400', 'state': 'Interface', 'type': 'ARPA', 'interface': 'MgmtEth0/RP0/CPU0/0', 'cpu': '0/RP0/CPU0'},
    {'ip_address': '192.168.246.96', 'age': '02:11:50', 'mac_address': '506a.f100.1100', 'state': 'Dynamic', 'type': 'ARPA', 'interface': 'MgmtEth0/RP0/CPU0/0', 'cpu': '0/RP0/CPU0'},
    {'ip_address': '192.168.246.99', 'age': '00:00:35', 'mac_address': '0004.968f.fe0a', 'state': 'Dynamic', 'type': 'ARPA', 'interface': 'MgmtEth0/RP0/CPU0/0', 'cpu': '0/RP0/CPU0'},
    {'ip_address': '192.168.246.100', 'age': '00:03:44', 'mac_address': '0004.968b.d8aa', 'state': 'Dynamic', 'type': 'ARPA', 'interface': 'MgmtEth0/RP0/CPU0/0', 'cpu': '0/RP0/CPU0'},
    {'ip_address': '192.168.246.101', 'age': '00:04:38', 'mac_address': '0004.9698.6bb7', 'state': 'Dynamic', 'type': 'ARPA', 'interface': 'MgmtEth0/RP0/CPU0/0', 'cpu': '0/RP0/CPU0'},
    {'ip_address': '192.168.246.132', 'age': '02:48:31', 'mac_address': '40f0.78c7.e9f2', 'state': 'Dynamic', 'type': 'ARPA', 'interface': 'MgmtEth0/RP0/CPU0/0', 'cpu': '0/RP0/CPU0'},
    {'ip_address': '192.168.246.133', 'age': '02:41:07', 'mac_address': 'dc77.4c58.b224', 'state': 'Dynamic', 'type': 'ARPA', 'interface': 'MgmtEth0/RP0/CPU0/0', 'cpu': '0/RP0/CPU0'},
    {'ip_address': '192.168.246.135', 'age': '00:00:06', 'mac_address': 'd884.66ea.fe00', 'state': 'Dynamic', 'type': 'ARPA', 'interface': 'MgmtEth0/RP0/CPU0/0', 'cpu': '0/RP0/CPU0'},
    {'ip_address': '192.168.246.137', 'age': '00:06:01', 'mac_address': 'bc2c.e69a.6500', 'state': 'Dynamic', 'type': 'ARPA', 'interface': 'MgmtEth0/RP0/CPU0/0', 'cpu': '0/RP0/CPU0'},
    {'ip_address': '192.168.246.156', 'age': '00:05:59', 'mac_address': '609c.9fde.2f01', 'state': 'Dynamic', 'type': 'ARPA', 'interface': 'MgmtEth0/RP0/CPU0/0', 'cpu': '0/RP0/CPU0'},
    {'ip_address': '192.168.246.180', 'age': '00:00:06', 'mac_address': 'e441.6417.b5fd', 'state': 'Dynamic', 'type': 'ARPA', 'interface': 'MgmtEth0/RP0/CPU0/0', 'cpu': '0/RP0/CPU0'},
    {'ip_address': '192.168.246.181', 'age': '00:00:08', 'mac_address': 'e441.6417.b978', 'state': 'Dynamic', 'type': 'ARPA', 'interface': 'MgmtEth0/RP0/CPU0/0', 'cpu': '0/RP0/CPU0'},
    {'ip_address': '192.168.246.223', 'age': '00:00:01', 'mac_address': '8a4f.a70a.912b', 'state': 'Dynamic', 'type': 'ARPA', 'interface': 'MgmtEth0/RP0/CPU0/0', 'cpu': '0/RP0/CPU0'},
    {'ip_address': '192.168.246.254', 'age': '00:43:50', 'mac_address': '4656.b8e1.7151', 'state': 'Dynamic', 'type': 'ARPA', 'interface': 'MgmtEth0/RP0/CPU0/0', 'cpu': '0/RP0/CPU0'}
]
```

https://github.com/networktocode/ntc-templates/tree/master/ntc_templates/templates

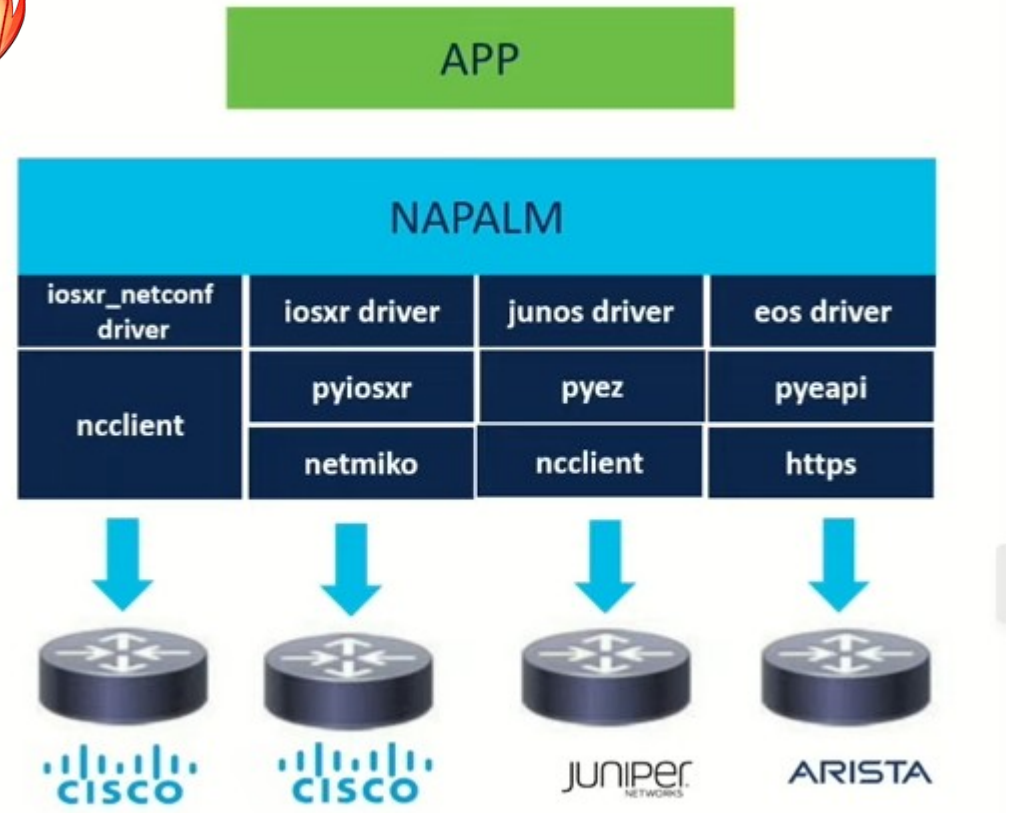
Agenda


04
Napalm
07
NetBox
05
Scrapli
08
Nornir
06
NETCONF
09
Desafio

Napalm – O que é?

- Biblioteca Python que disponibiliza funções para interagir com diferentes equipamentos de rede;
- Cisco IOS-XR, Cisco IOS, Cisco NX-Osm Junos e Arista EOS;
- Open Source;
- Camada de abstração para programação e Automação de Redes;
- Outras plataformas: https://github.com/napalm-automation-community

	EOS	IOS	IOSXR	IOSXR_NETCONF	JUNOS	NXOS	NXOS_SSH
get_arp_table	✓	✓	✗	✗	✗	✗	✓
get_bgp_config	✓	✓	✓	✓	✓	✗	✗
get_bgp_neighbors	✓	✓	✓	✓	✓	✓	✓
get_bgp_neighbors_detail	✓	✓	✓	✓	✓	✗	✗
get_config	✓	✓	✓	✗	✓	✓	✓
get_environment	✓	✓	✓	✓	✓	✓	✓
get_facts	✓	✓	✓	✓	✓	✓	✓
get_firewall_policies	✗	✗	✗	✗	✗	✗	✗
get_interfaces	✓	✓	✓	✓	✓	✓	✓
get_interfaces_counters	✓	✓	✓	✓	✓	✗	✓
get_interfaces_ip	✓	✓	✓	✓	✓	✓	✓
get_ipv6_neighbors_table	✗	✓	✗	✗	✓	✗	✗
get_lldp_neighbors	✓	✓	✓	✓	✓	✓	✓
get_lldp_neighbors_detail	✓	✓	✓	✓	✓	✓	✓
get_mac_address_table	✓	✓	✓	✓	✓	✓	✓
get_network_instances	✓	✓	✗	✗	✓	✓	✓
get_ntp_peers	✗	✓	✓	✓	✓	✓	✓
get_ntp_servers	✓	✓	✓	✓	✓	✓	✓
get_ntp_stats	✓	✓	✓	✓	✓	✓	✗
get_optics	✓	✓	✗	✗	✓	✗	✓
get_probes_config	✗	✓	✓	✓	✓	✗	✗
get_probes_results	✗	✗	✓	✓	✓	✗	✗
get_route_to	✓	✗	✗	✗	✗	✗	✗
get_snmp_information	✓	✓	✓	✓	✓	✓	✓
get_users	✓	✓	✓	✓	✓	✓	✓
get_vlans	✓	✓	✗	✗	✓	✓	✓
is_alive	✓	✓	✓	✓	✓	✓	✓
ping	✓	✓	✗	✗	✓	✓	✓
traceroute	✓	✓	✓	✓	✓	✓	✓



Install: pip install napalm

https://napalm.readthedocs.io/en/latest/

Napalm – Estrutura de Dados

IOS-XR

```
{
  "uptime": 35457914,
  "vendor": "Cisco",
  "hostname": "edge01.tab",
  "fqdn": "edge01.tab01",
  "os_version": "5.3.1",
  "serial_number": "FOX171",
  "model": "ASR-9904-AC”,
  "interface_list": [
      "TenGigE0/0/0/13",
      "TenGigE0/0/0/14",
      "TenGigE0/0/0/24”
   ]
}
```

IOS

```
{
  "uptime": 16676160,
  "vendor": "Cisco",
  "hostname": "NS2903",
  "fqdn": "NS2903-ASW"
  "os_version": "15.0(2)”,
  "serial_number": "FOC1",
  "model": "WS-C2960G",
  "interface_list": [
     "Vlan1",
     "GigabitEthernet0/1”,
     "GigabitEthernet0/5”
   ]
}
```

JUNOS

```
{
  "uptime": 4380,
  "vendor": "Juniper",
  "hostname": "vsrx",
  "fqdn": "vsrx",
  "os_version": "12.1X4",
  "serial_number": "beb91",
  "model": "FIREFLY",
  "interface_list": [
      "ge-0/0/0",
      "ge-0/0/1",
      "ge-0/0/2”
   ]
}
```

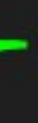
EOS

```
{
  "uptime": 123456,
  "vendor": "Arista”,
"hostname": "localhost",
  "fqdn": "localhost",
  "os_version": "4.15.5M",
  "serial_number": "",
  "model": "vEOS",
  "interface_list": [
    "Ethernet1",
    "Ethernet2",
    "Ethernet3",
    "Management1
  ],
}
```

Napalm – exercicio1.py

napalm > exercicio1.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

from napalm import get_network_driver
import json, os
from dotenv import load_dotenv

load_dotenv()

driver = get_network_driver('iosxr')

router1 = {"hostname": "192.168.246.95","username": os.getenv("LAB_USERNAME"),"password": os.getenv("LAB_PASSWORD")}

try:
    r1_connection = driver(**router1)
    r1_connection.open()
    print(f"Connecting to {router1['hostname']}")
    #output = r1_connection.get_interfaces()
    #output = r1_connection.get_facts()
    #output = r1_connection.get_interfaces_ip()
    output = r1_connection.get_interfaces_counters()
    r1_connection.close()

    output_json = json.dumps(output, indent=4)
    print(output_json)
except Exception as err:
    print(err)

```

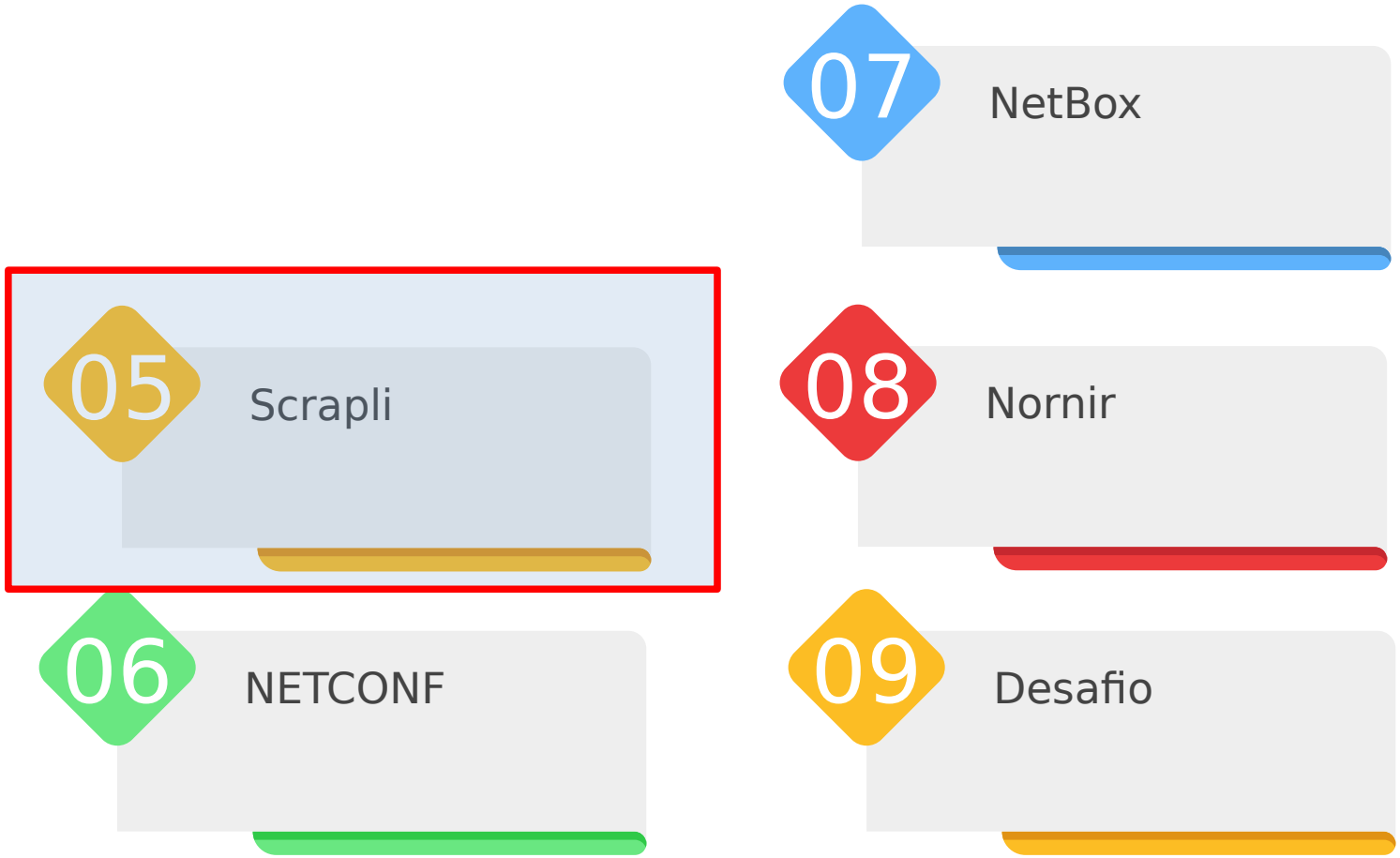


Napalm – exercicio1.py - Output


```
{
    "GigabitEthernet0/0/0/0": {
        "tx_multicast_packets": 0,
        "tx_discards": 0,
        "tx_octets": 0,
        "tx_errors": 0,
        "rx_octets": 0,
        "tx_unicast_packets": 0,
        "rx_errors": 0,
        "tx_broadcast_packets": 0,
        "rx_multicast_packets": 0,
        "rx_broadcast_packets": 0,
        "rx_discards": 0,
        "rx_unicast_packets": 0
    },
    "GigabitEthernet0/0/0/1": {
        "tx_multicast_packets": 0,
        "tx_discards": 0,
        "tx_octets": 0,
        "tx_errors": 0,
        "rx_octets": 0,
        "tx_unicast_packets": 0,
        "rx_errors": 0,
        "tx_broadcast_packets": 0,
        "rx_multicast_packets": 0,
        "rx_broadcast_packets": 0,
        "rx_discards": 0,
        "rx_unicast_packets": 0
    },
    "GigabitEthernet0/0/0/2": {
        "tx_multicast_packets": 0,
        "tx_discards": 0,
        "tx_octets": 0,
        "tx_errors": 0,
        "rx_octets": 0,
        "tx_unicast_packets": 0,
        "rx_errors": 0,
        "tx_broadcast_packets": 0,
        "rx_multicast_packets": 0,
        "rx_broadcast_packets": 0,
        "rx_discards": 0,
        "rx_unicast_packets": 0
    },
```

```
    "GigabitEthernet0/0/0/3": {
        "tx_multicast_packets": 0,
        "tx_discards": 0,
        "tx_octets": 0,
        "tx_errors": 0,
        "rx_octets": 0,
        "tx_unicast_packets": 0,
        "rx_errors": 0,
        "tx_broadcast_packets": 0,
        "rx_multicast_packets": 0,
        "rx_broadcast_packets": 0,
        "rx_discards": 0,
        "rx_unicast_packets": 0
    },
    "GigabitEthernet0/0/0/4": {
        "tx_multicast_packets": 0,
        "tx_discards": 0,
        "tx_octets": 0,
        "tx_errors": 0,
        "rx_octets": 0,
        "tx_unicast_packets": 0,
        "rx_errors": 0,
        "tx_broadcast_packets": 0,
        "rx_multicast_packets": 0,
        "rx_broadcast_packets": 0,
        "rx_discards": 0,
        "rx_unicast_packets": 0
    },
    "GigabitEthernet0/0/0/5": {
        "tx_multicast_packets": 0,
        "tx_discards": 0,
        "tx_octets": 0,
        "tx_errors": 0,
        "rx_octets": 0,
        "tx_unicast_packets": 0,
        "rx_errors": 0,
        "tx_broadcast_packets": 0,
        "rx_multicast_packets": 0,
        "rx_broadcast_packets": 0,
        "rx_discards": 0,
        "rx_unicast_packets": 0
    },
```

Agenda



Scrapli – O que é?

- Biblioteca Python desenvolvida em 2020;
- Foi desenvolvida para abordar limitações encontradas no Netmiko e Paramiko;
- Open Source;
- Desenvolvido para ser mais rápido que o netmiko (paramiko x SSH2);
- Mais opções de protocolo de Transporte que o Netmiko (Telnet, SSH2, paramiko, etc);
- Trabalha com Netconf;
- Tem plugin para Trabalhar com Nornir;
- Implementação para GO - scrapligo

Platform/OS	Scrapli Driver	Scrapli Async Driver	Platform Name
Cisco IOS-XE	IOSXEDriver	AsyncIOSXEDriver	cisco_iosxe
Cisco NX-OS	NXOSDriver	AsyncNXOSDriver	cisco_nxos
Cisco IOS-XR	IOSXRDriver	AsyncIOSXRDriver	cisco_iosxr
Arista EOS	EOSDriver	AsyncEOSDriver	arista_eos
Juniper JunOS	JunosDriver	AsyncJunosDriver	juniper_junos

pip install scrapli[full]

pip install scrapli-netconf

https://github.com/carlmontanari/scrapli

Scrapli

Basic Driver Arguments

Argument	Purpose/Value
host	name/ip of host to connect to
port	port of host to connect to (defaults to port 22)
auth_username	username for authentication
auth_password	password for authentication
auth_secondary	password for secondary authentication (enable password)
auth_private_key	private key for authentication
auth_strict_key	strict key checking -- TRUE by default!
ssh_config_file	True/False or path to ssh config file to use

Scrapli – exercicio1.py

scrapli > exercicio1.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

from scrapli.driver.core import IOSXRDriver
import os, json
from dotenv import load_dotenv

load_dotenv()

router1 = {"host": "192.168.246.95","auth_username": os.getenv("LAB_USERNAME"),"auth_password": os.getenv("LAB_PASSWORD"), "port": 22}

try:
    with IOSXRDriver(**router1) as conn:
        response = conn.send_command("show version")

        results = response.result

        print(results)

        results_textfsm = response.textfsm_parse_output()

        output_json = json.dumps(results_textfsm, indent=4)

        print(output_json)
except Exception as err:
    print(err)

```

Scrapli – exercicio1.py - Output

```
Fri Aug 23 14:04:06.912 UTC
Cisco IOS XR Software, Version 7.6.1
Copyright (c) 2013-2022 by Cisco Systems, Inc.

Build Information:
 Built By     : ingunawa
 Built On     : Sat Mar 26 19:10:06 PDT 2022
 Built Host   : iox-ucs-072
 Workspace    : /auto/srcarchive17/prod/7.6.1/xrv9k/ws
 Version      : 7.6.1
 Location     : /opt/cisco/XR/packages/
 Label        : 7.6.1-0

cisco IOS-XRv 9000 () processor
System uptime is 6 days 18 hours 31 minutes
[
    {
        "version": "7.6.1",
        "uptime": "6 days 18 hours 31 minutes",
        "location": "/opt/cisco/XR/packages/",
        "hardware": "IOS-XRv 9000",
        "build_host": "iox-ucs-072",
        "label": "7.6.1-0"
    }
]
```

Scrapli – exercicio1.py - Output

```
Fri Aug 23 14:04:06.912 UTC
Cisco IOS XR Software, Version 7.6.1
Copyright (c) 2013-2022 by Cisco Systems, Inc.

Build Information:
 Built By     : ingunawa
 Built On     : Sat Mar 26 19:10:06 PDT 2022
 Built Host   : iox-ucs-072
 Workspace    : /auto/srcarchive17/prod/7.6.1/xrv9k/ws
 Version      : 7.6.1
 Location     : /opt/cisco/XR/packages/
 Label        : 7.6.1-0

cisco IOS-XRv 9000 () processor
System uptime is 6 days 18 hours 31 minutes
```

String

```
[
    {
        "version": "7.6.1",
        "uptime": "6 days 18 hours 31 minutes",
        "location": "/opt/cisco/XR/packages/",
        "hardware": "IOS-XRv 9000",
        "build_host": "iox-ucs-072",
        "label": "7.6.1-0"
    }
]
```

Scrapli – exercicio1.py - Output

```
Fri Aug 23 14:04:06.912 UTC
Cisco IOS XR Software, Version 7.6.1
Copyright (c) 2013-2022 by Cisco Systems, Inc.

Build Information:
 Built By     : ingunawa
 Built On     : Sat Mar 26 19:10:06 PDT 2022
 Built Host   : iox-ucs-072
 Workspace    : /auto/srcarchive17/prod/7.6.1/xrv9k/ws
 Version      : 7.6.1
 Location     : /opt/cisco/XR/packages/
 Label        : 7.6.1-0

cisco IOS-XRv 9000 () processor
System uptime is 6 days 18 hours 31 minutes
```

```
[
    {
        "version": "7.6.1",
        "uptime": "6 days 18 hours 31 minutes",
        "location": "/opt/cisco/XR/packages/",
        "hardware": "IOS-XRv 9000",
        "build_host": "iox-ucs-072",
        "label": "7.6.1-0"
    }
]
```

JSON – Dicionário
via TextFSM

Scrapli – exercicio2.py - Telnet

scrapli > exercicio2.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

from scrapli.driver.core import IOSXRDriver
import os
from dotenv import load_dotenv

load_dotenv()

router1 = {"host": "192.168.246.95","auth_username": os.getenv("LAB_USERNAME"),"auth_password": os.getenv("LAB_PASSWORD"), "transport": "telnet"}

try:
    with IOSXRDriver(**router1) as conn:
        response = conn.send_command("show version")
        print(response)
except Exception as err:
    print(err)

```

Scrapli x Netmiko

Scrapli x Netmiko

Transport Library

Underlying transport libraries support:

	Netmiko	Scrapli
paramiko	✓	✓
ssh2 (fast)	✗	✓
asyncssh	✗	✓

Scrapli x Netmiko

Parsing

Raw to structured parsing support

	Netmiko	Scrapli
Genie	✓	✓
TextFSM	✓	✓
TTP	✓	✓

Scrapli x Netmiko

Extensions

Other things that the libraries can do:

	Netmiko	Scrapli
Secure Copy (SCP)	✓	✗
NETCONF	✗	✓
Config Merge/Replace	✗	✓
Nornir Plugin	✓	✓

Agenda

07 NetBox

08 Nornir

06 NETCONF

09 Desafio

NETCONF

- NETCONF (NETwork CONFiguration) provê mecanismos para aplicar, manipular e deletar configurações de dispositivos [RFC6241]
- Fornece uma interface padrão de interação com equipamentos;
- Protocolo de gerenciamento de configuração;
- Armazena estados de configuração (candidate, running, statup, etc);
- Recuperação seletiva de dados com filtros;

YANG

- YANG é uma linguagem de modelagem de dados usada para modelar configuração, dados de estado e chamadas de procedimento remoto. [RFC7950]
- Legível por humanos e fácil de aprender a representação dos dados;
- Modelos de dados de configuração hierárquica

```
list interface {
        key "name";
        unique "type location";

        leaf name {
          type string;
          reference
            "RFC 2863: The Interfaces Group MIB - ifName";
        }

        leaf description {
          type string;
```

NETCONF – Operações

Data Manipulation

- <get>
- <get-config>
- <edit-config>
- <copy-config>
- <delete-config>
- <discard-changes> *(:candidate)*

Session Management

- <close-session>
- <kill-session>

Locking

- <lock>
- <unlock>

Transaction Management

- <commit> *(:candidate, :confirmed)*
- <cancel-commit> *(:confirmed)*

Schema Management

- <get-schema> *(:monitoring)*

RPC Extensions

- <rpc>

Scrapli & Netconf– exercicio3.py

scrapli > exercicio3.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

from scrapli_netconf.driver import NetconfDriver
import json, os, xmltodict
from dotenv import load_dotenv

load_dotenv()

router1 = {"host": "192.168.246.95","auth_username": os.getenv("LAB_USERNAME"),"auth_password": os.getenv("LAB_PASSWORD"), "port": 830}

try:
    with NetconfDriver(**router1) as conn:

        response = conn.get_config(source="running")

        xml_obj = response.result

        print(xml_obj)

        dict_obj = xmltodict.parse(xml_obj)

        print(dict_obj['rpc-reply']['data'])

        output_json = json.dumps(dict_obj['rpc-reply']['data'], indent=4)

        print(output_json)


except Exception as err:
    print(err)

```

Scrapli & Netconf– exercicio3.py

scrapli > exercicio3.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

from scrapli_netconf.driver import NetconfDriver
import json, os, xmltodict
from dotenv import load_dotenv

load_dotenv()

router1 = {"host": "192.168.246.95","auth_username": os.getenv("LAB_USERNAME"),"auth_password": os.getenv("LAB_PASSWORD"), "port": 830}

try:
    with NetconfDriver(**router1) as conn:

        response = conn.get_config(source="running")

        xml_obj = response.result

        print(xml_obj)

        dict_obj = xmltodict.parse(xml_obj)

        print(dict_obj['rpc-reply']['data'])

        output_json = json.dumps(dict_obj['rpc-reply']['data'], indent=4)

        print(output_json)


except Exception as err:
    print(err)

```

Scrapli & Netconf– exercicio3.py

scrapli > exercicio3.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

from scrapli_netconf.driver import NetconfDriver
import json, os, xmltodict
from dotenv import load_dotenv

load_dotenv()

router1 = {"host": "192.168.246.95","auth_username": os.getenv("LAB_USERNAME"),"auth_password": os.getenv("LAB_PASSWORD"), "port": 830}

try:
    with NetconfDriver(**router1) as conn:

        response = conn.get_config(source="running")

        xml_obj = response.result

        print(xml_obj)

        dict_obj = xmltodict.parse(xml_obj)

        print(dict_obj['rpc-reply']['data'])

        output_json = json.dumps(dict_obj['rpc-reply']['data'], indent=4)

        print(output_json)


except Exception as err:
    print(err)

```

Scrapli & Netconf– exercicio3.py

scrapli > exercicio3.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

from scrapli_netconf.driver import NetconfDriver
import json, os, xmltodict
from dotenv import load_dotenv

load_dotenv()

router1 = {"host": "192.168.246.95","auth_username": os.getenv("LAB_USERNAME"),"auth_password": os.getenv("LAB_PASSWORD"), "port": 830}

try:
    with NetconfDriver(**router1) as conn:

        response = conn.get_config(source="running")

        xml_obj = response.result

        print(xml_obj)

        dict_obj = xmltodict.parse(xml_obj)

        print(dict_obj['rpc-reply']['data'])

        output_json = json.dumps(dict_obj['rpc-reply']['data'], indent=4)

        print(output_json)


except Exception as err:
    print(err)

```

Scrapli & Netconf– exercicio3.py

scrapli > exercicio3.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

from scrapli_netconf.driver import NetconfDriver
import json, os, xmltodict
from dotenv import load_dotenv

load_dotenv()

router1 = {"host": "192.168.246.95","auth_username": os.getenv("LAB_USERNAME"),"auth_password": os.getenv("LAB_PASSWORD"), "port": 830}

try:
    with NetconfDriver(**router1) as conn:

        response = conn.get_config(source="running")

        xml_obj = response.result

        print(xml_obj)

        dict_obj = xmltodict.parse(xml_obj)

        print(dict_obj['rpc-reply']['data'])

        output_json = json.dumps(dict_obj['rpc-reply']['data'], indent=4)

        print(output_json)


except Exception as err:
    print(err)

```

Scrapli & Netconf– exercicio3.py

scrapli > exercicio3.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

from scrapli_netconf.driver import NetconfDriver
import json, os, xmltodict
from dotenv import load_dotenv

load_dotenv()

router1 = {"host": "192.168.246.95","auth_username": os.getenv("LAB_USERNAME"),"auth_password": os.getenv("LAB_PASSWORD"), "port": 830}

try:
    with NetconfDriver(**router1) as conn:

        response = conn.get_config(source="running")

        xml_obj = response.result

        print(xml_obj)

        dict_obj = xmltodict.parse(xml_obj)

        print(dict_obj['rpc-reply']['data'])

        output_json = json.dumps(dict_obj['rpc-reply']['data'], indent=4)

        print(output_json)


except Exception as err:
    print(err)

```

Scrapli & Netconf– exercicio3.py - Output


```
"interfaces": [
        {
            "@xmlns": "http://cisco.com/ns/yang/Cisco-IOS-XR-um-interface-cfg",
            "interface": [
                {
                    "interface-name": "MgmtEth0/RP0/CPU0/0",
                    "ipv4": {
                        "addresses": {
                            "@xmlns": "http://cisco.com/ns/yang/Cisco-IOS-XR-um-if-ip-address-cfg",
                            "address": {
                                "address": "192.168.246.96",
                                "netmask": "255.255.255.0"
                            }
                        }
                    }
                },
                {
                    "interface-name": "GigabitEthernet0/0/0/1",
                    "ipv4": {
                        "addresses": {
                            "@xmlns": "http://cisco.com/ns/yang/Cisco-IOS-XR-um-if-ip-address-cfg",
                            "address": {
                                "address": "192.168.100.2",
                                "netmask": "255.255.255.252"
                            }
                        }
                    }
                }]}]
```

Agenda

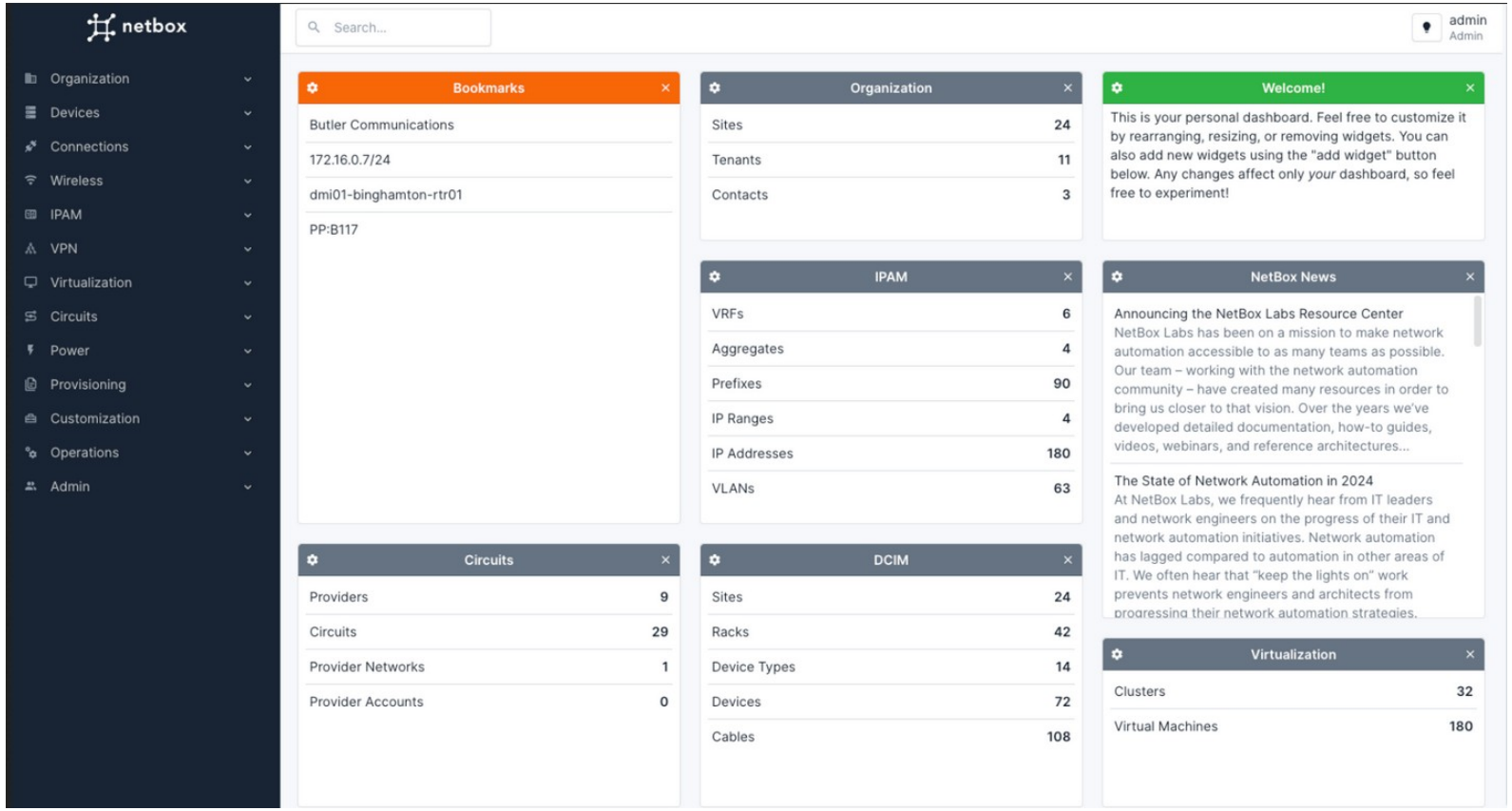
07 NetBox

08 Nornir

09 Desafio

O que é NetBox?

- Open-source IPAM/DCIM;
- Desenvolvido em Python (Django);
- Permite documentar sua infraestrutura;
- Peça chave em Automação de Redes;
- Atua como “Source of Truth” para sua infraestrutura/rede;
- Ideal para todos os times da empresa;
- API Support
  - REST and GraphQL;
  - Webhooks;
  - Pynetbox;



https://netboxlabs.com/docs/netbox/en/stable/
https://docs.netbox.dev/en/stable/
https://github.com/netbox-community/netbox-docker

Instalando o NetBox Docker

```
git clone -b release https://github.com/netbox-community/netbox-docker.git
cd netbox-docker
tee docker-compose.override.yml <<EOF
services:
  netbox:
    ports:
      - 8000:8080
EOF
docker compose pull
docker compose up
```

https://github.com/netbox-community/netbox-docker

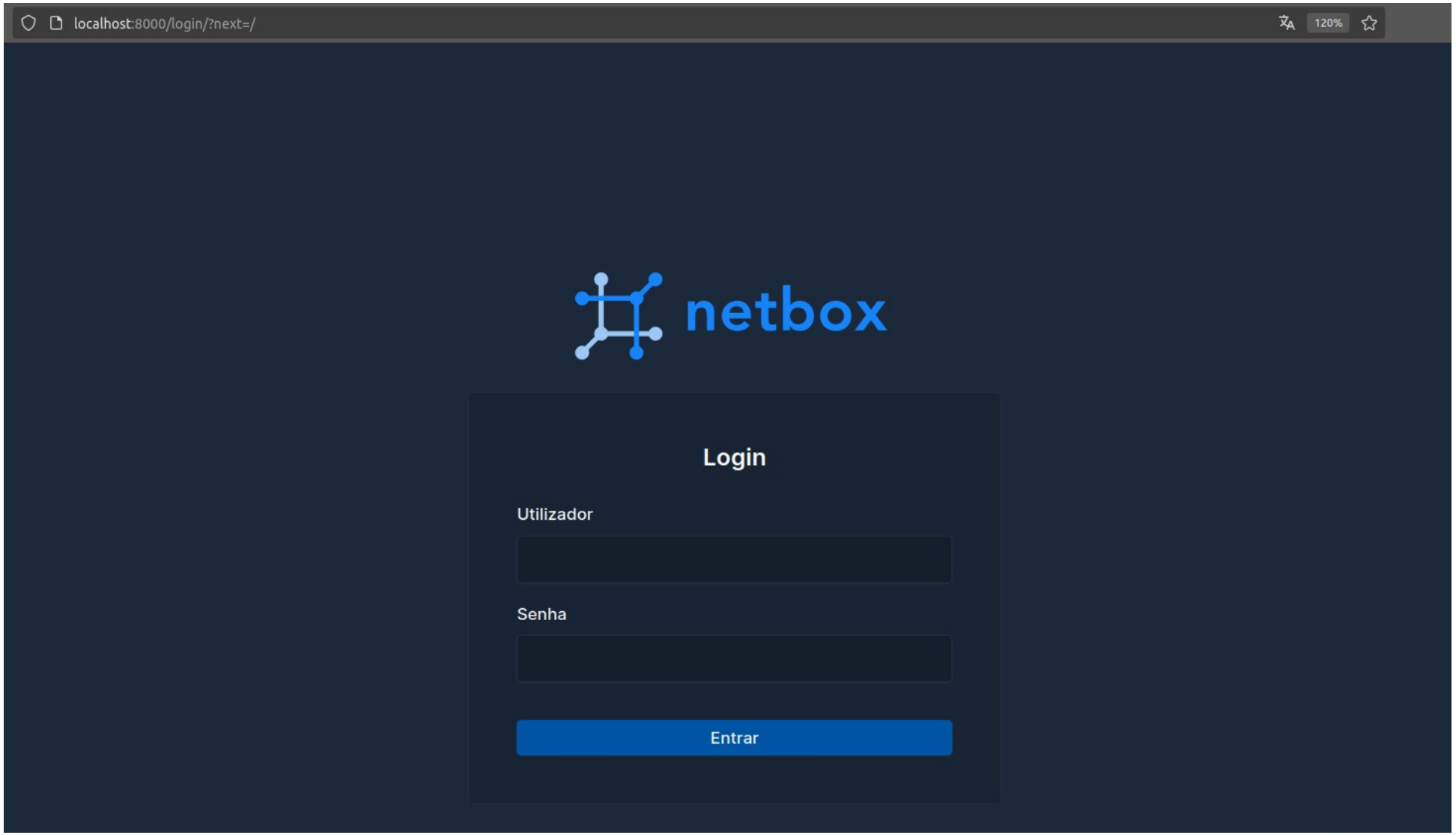
Criando o usuário de acesso - NetBox

To create the first admin user run this command:

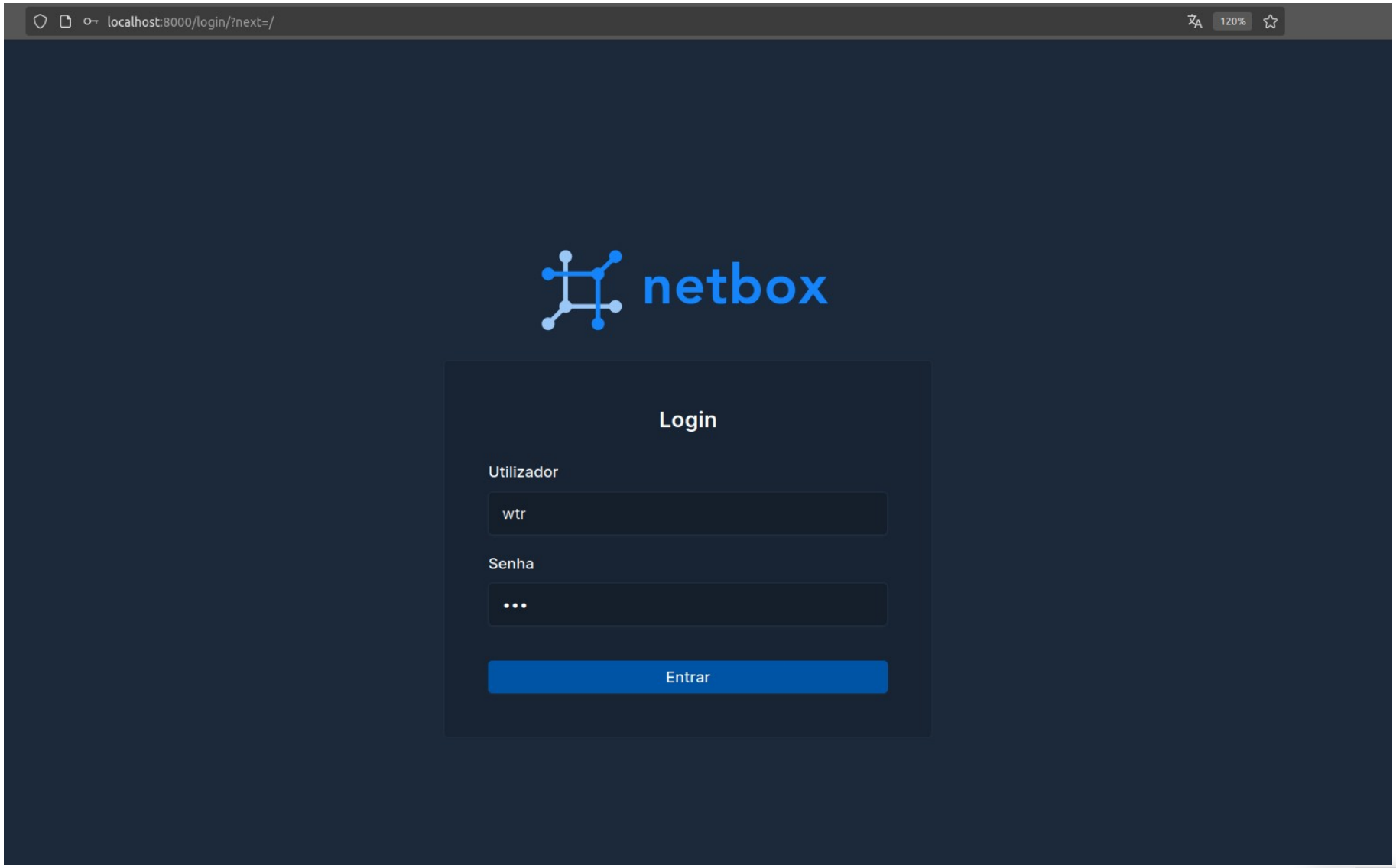
```
docker compose exec netbox /opt/netbox/netbox/manage.py createsuperuser
```

```
root@william:/opt/netbox-docker# docker-compose exec netbox /opt/netbox/netbox/manage.py createsuperuser
 loaded config '/etc/netbox/config/configuration.py'
 loaded config '/etc/netbox/config/extra.py'
 loaded config '/etc/netbox/config/logging.py'
 loaded config '/etc/netbox/config/plugins.py'
Username: wtr
Email address:
Password:
Password (again):
Superuser created successfully.
```

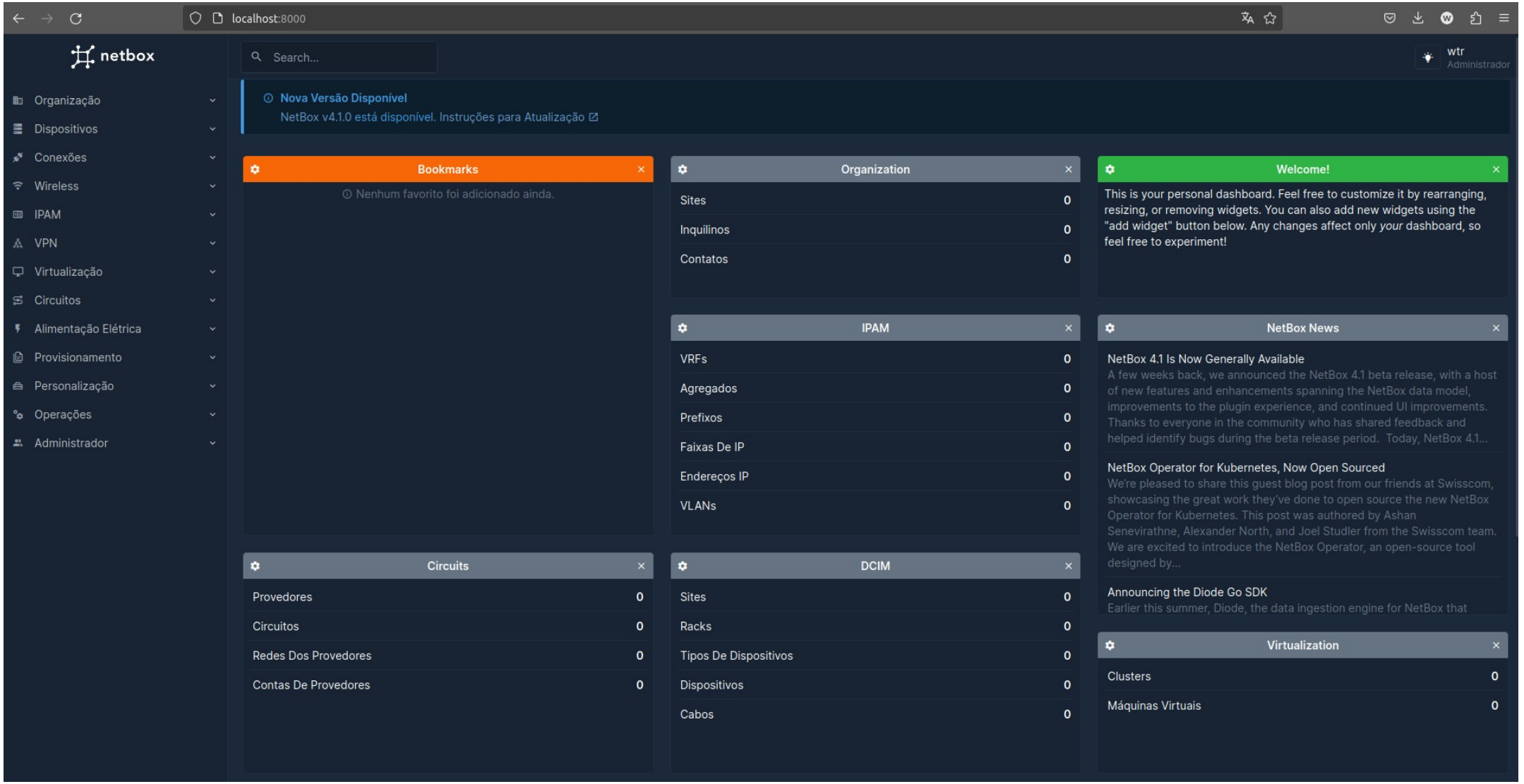
Acessando o NetBox



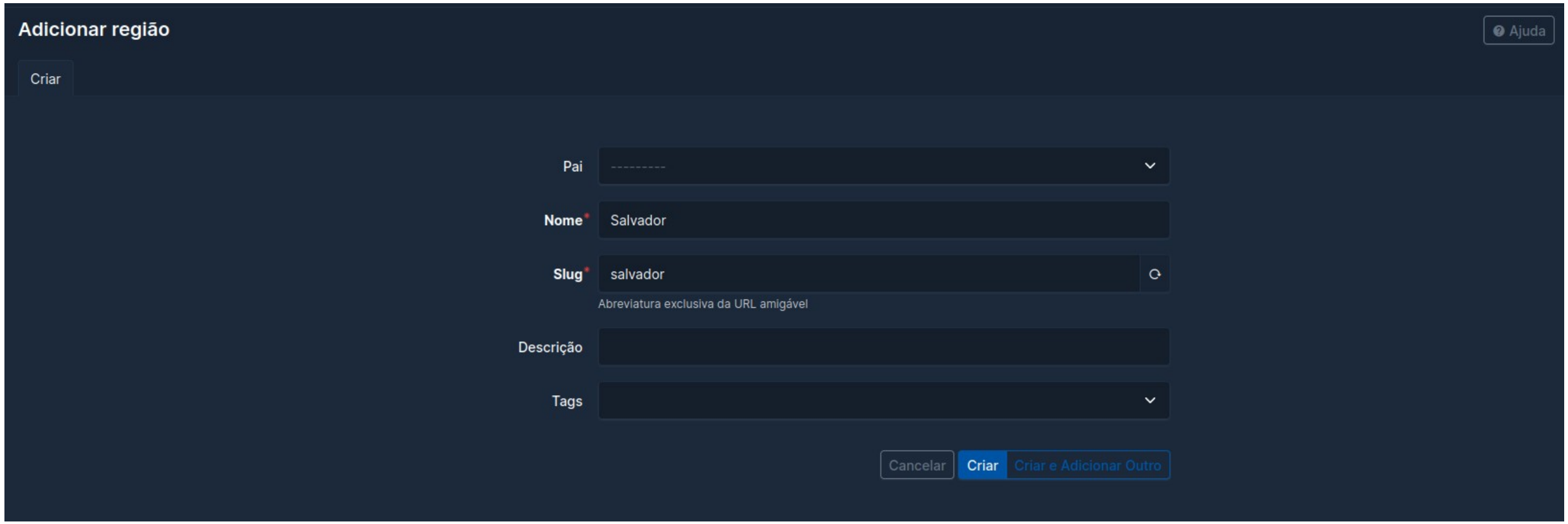
Acessando o NetBox



Acessando o NetBox

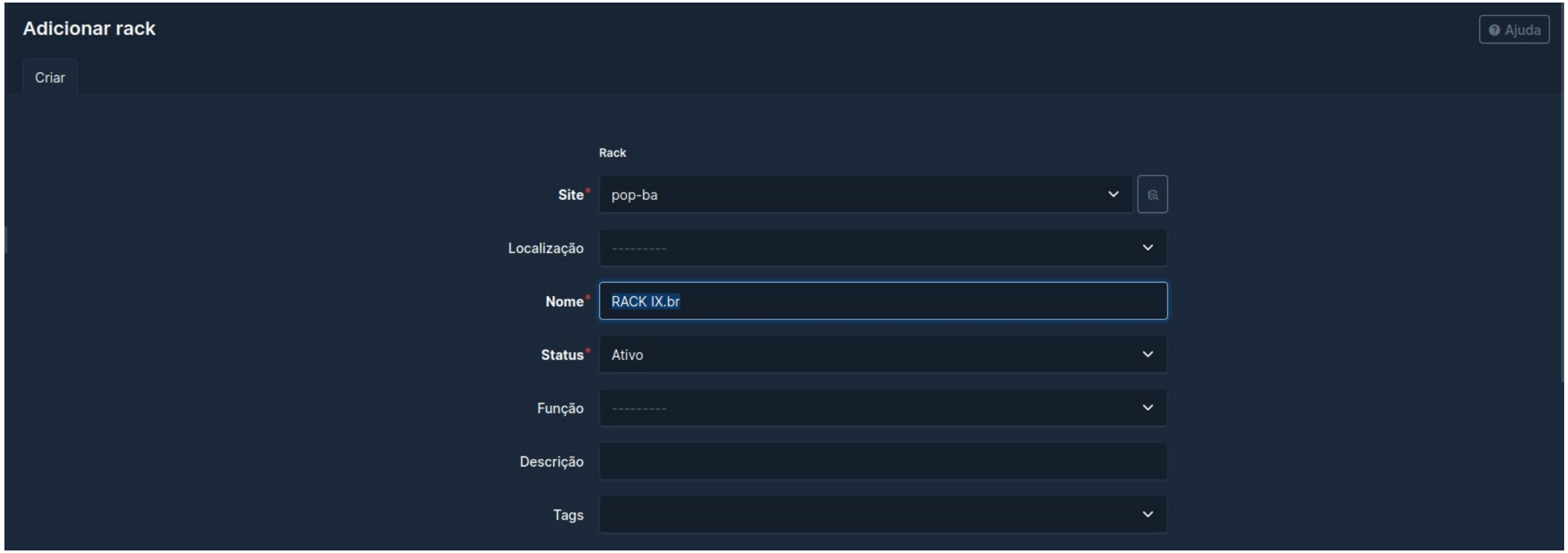


Criando uma Região - Netbox

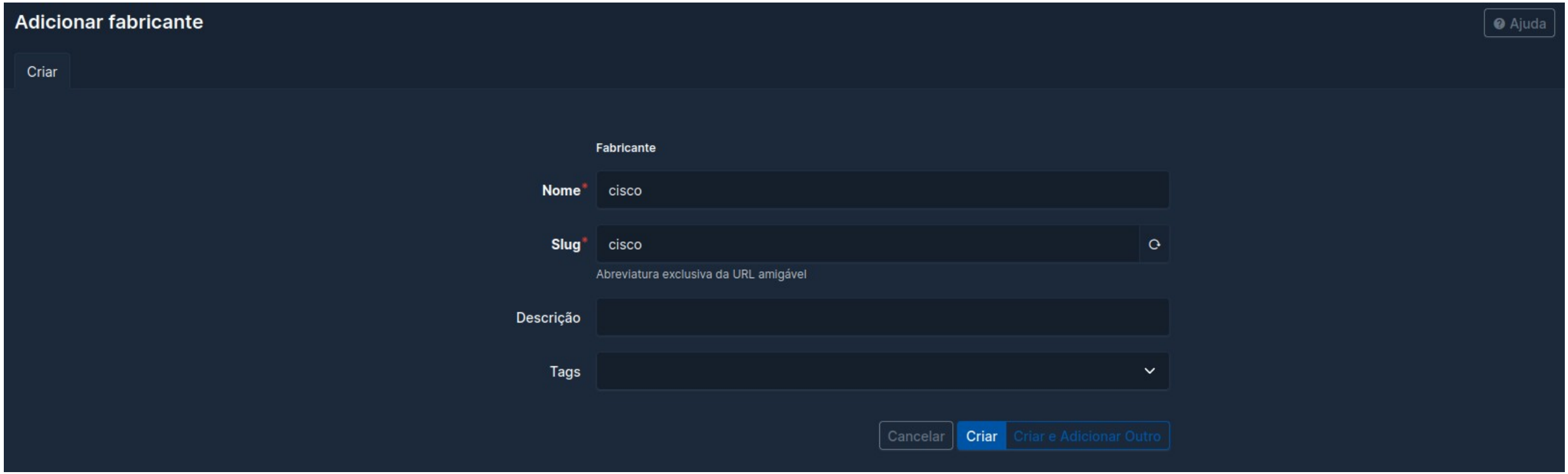


Criando um Site - Netbox

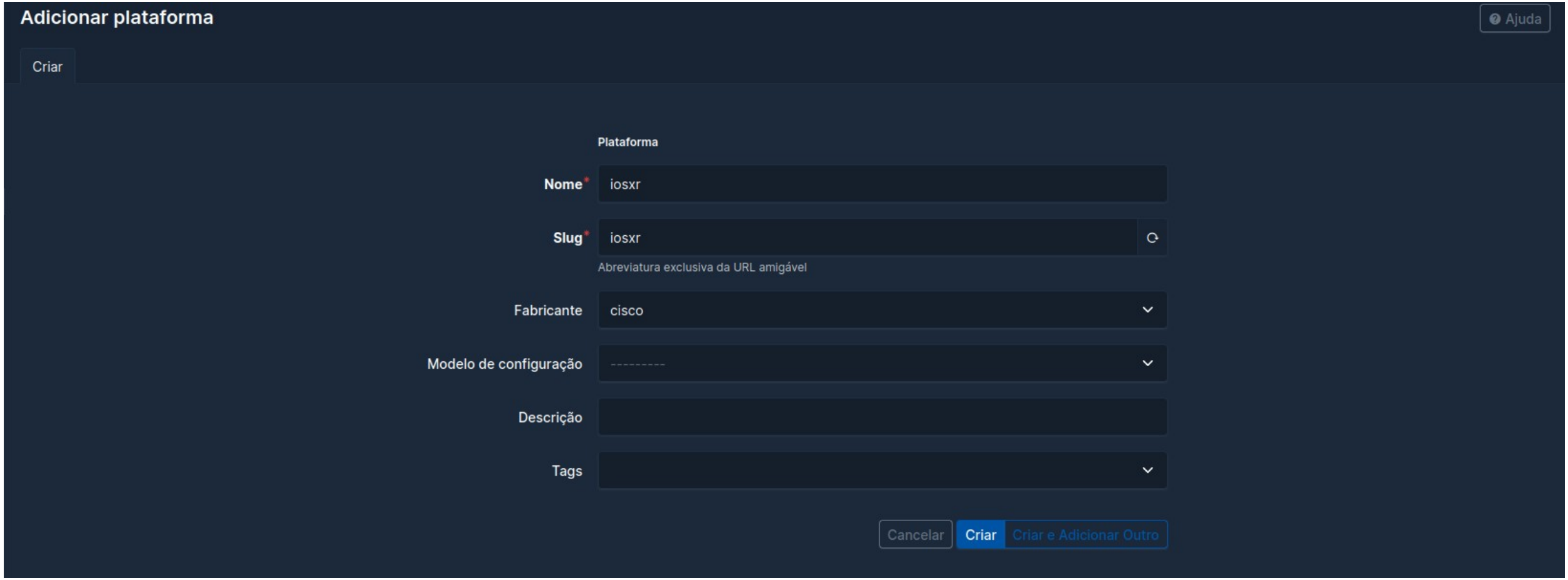
Criando um Rack - Netbox



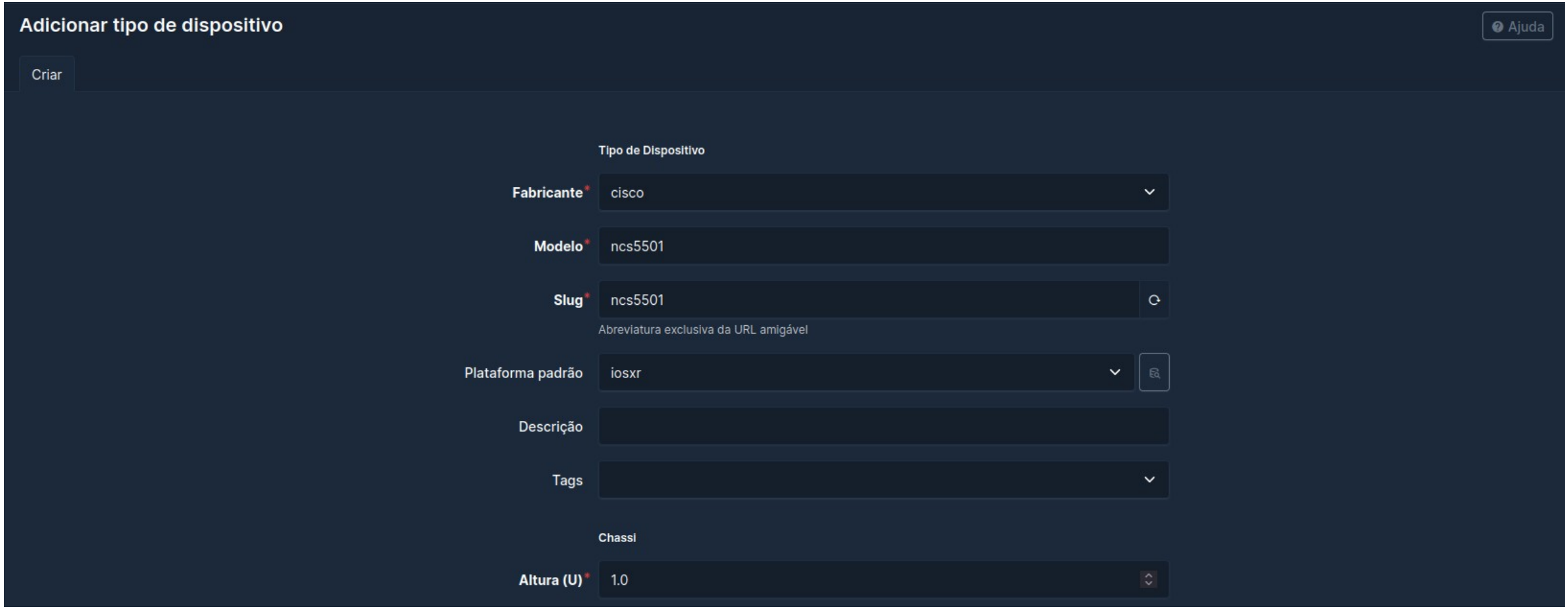
Adicionando um Fabricante - Netbox



Adicionando um Platform - Netbox

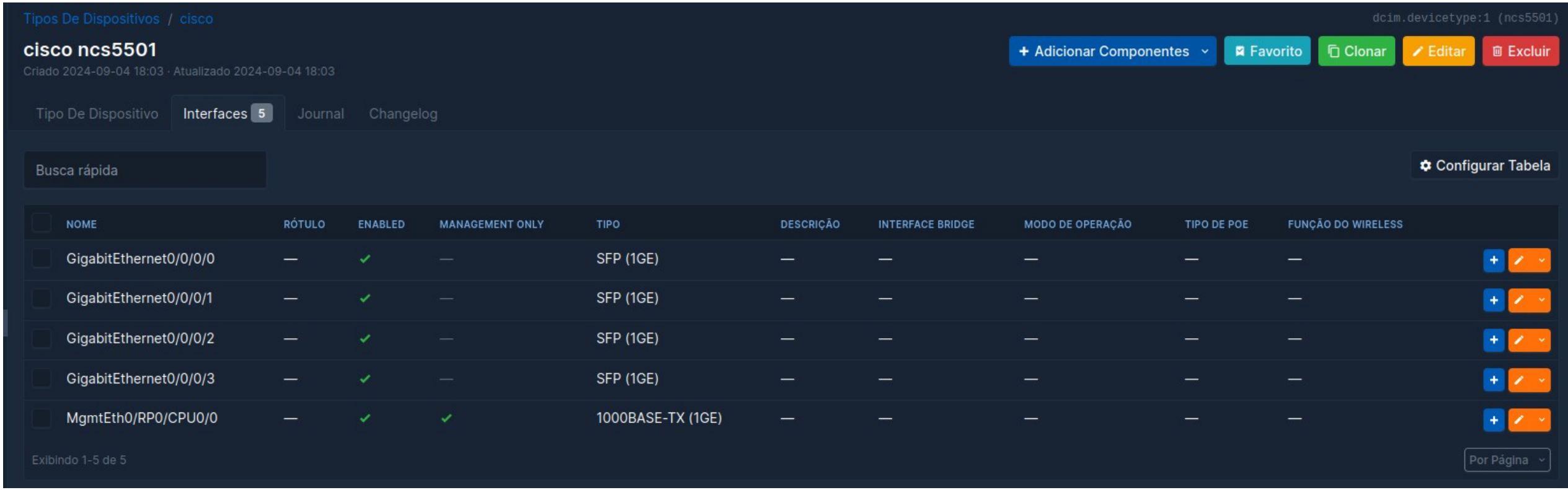


Adicionando um Device Type - Netbox

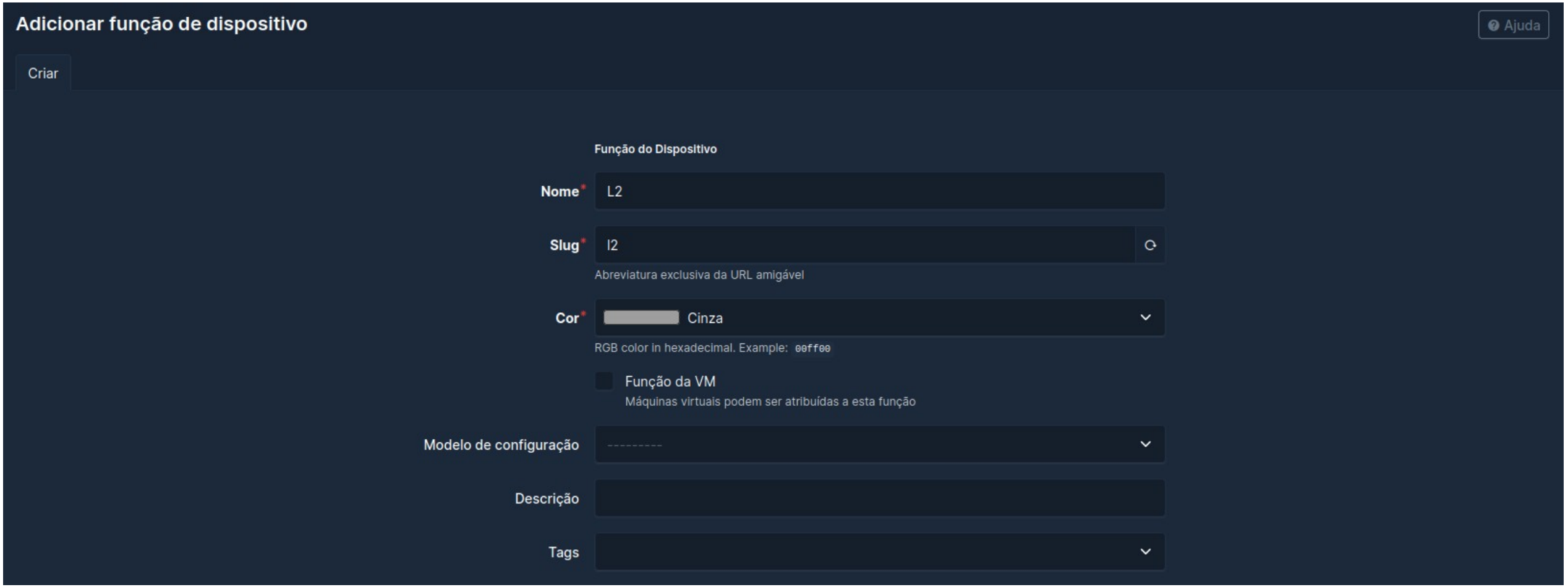


Adicionando Interfaces em um Device Type - Netbox

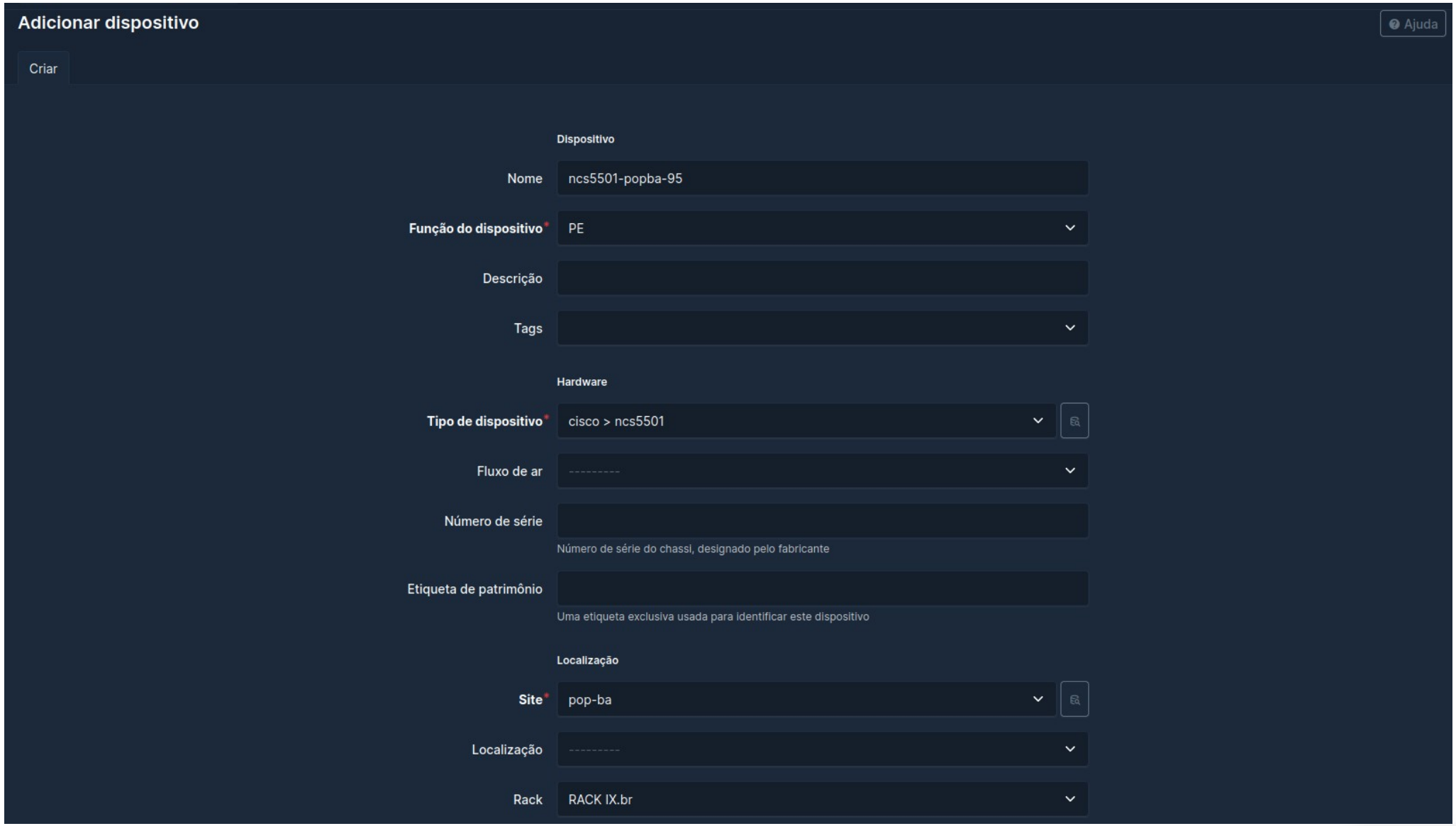
Device Type - Netbox



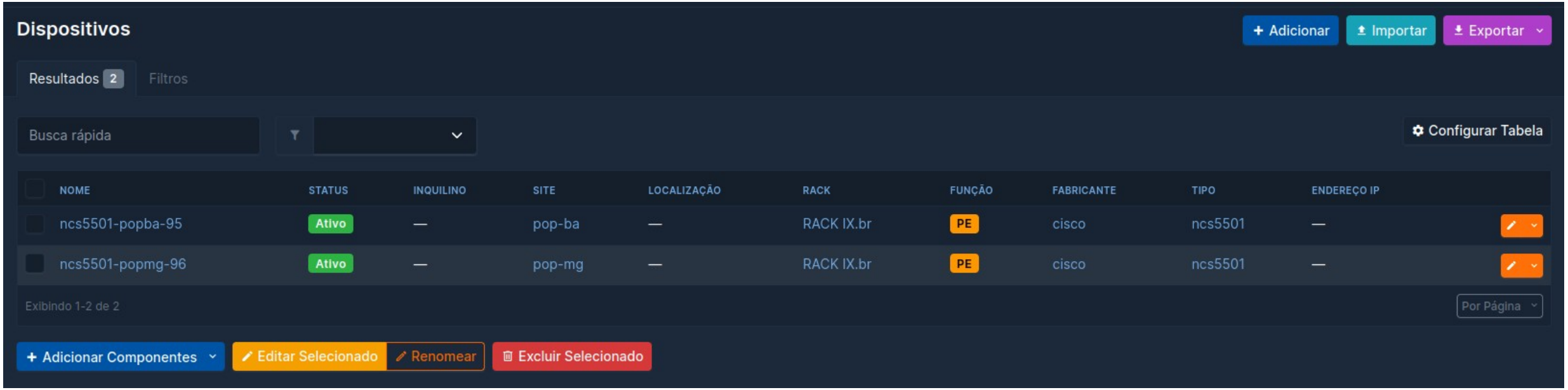
Adicionando Role de Devices - Netbox



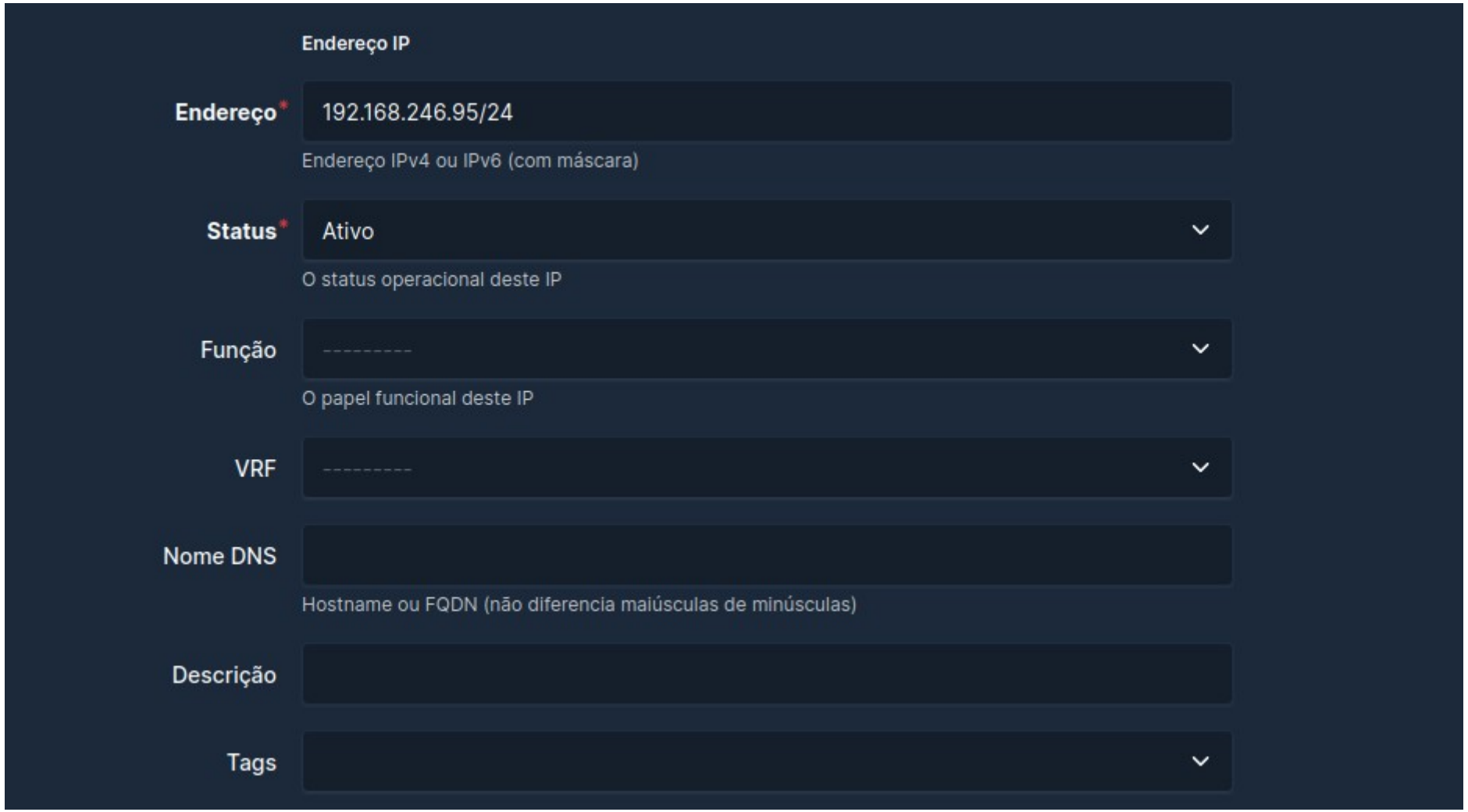
Adicionando um Device - Netbox



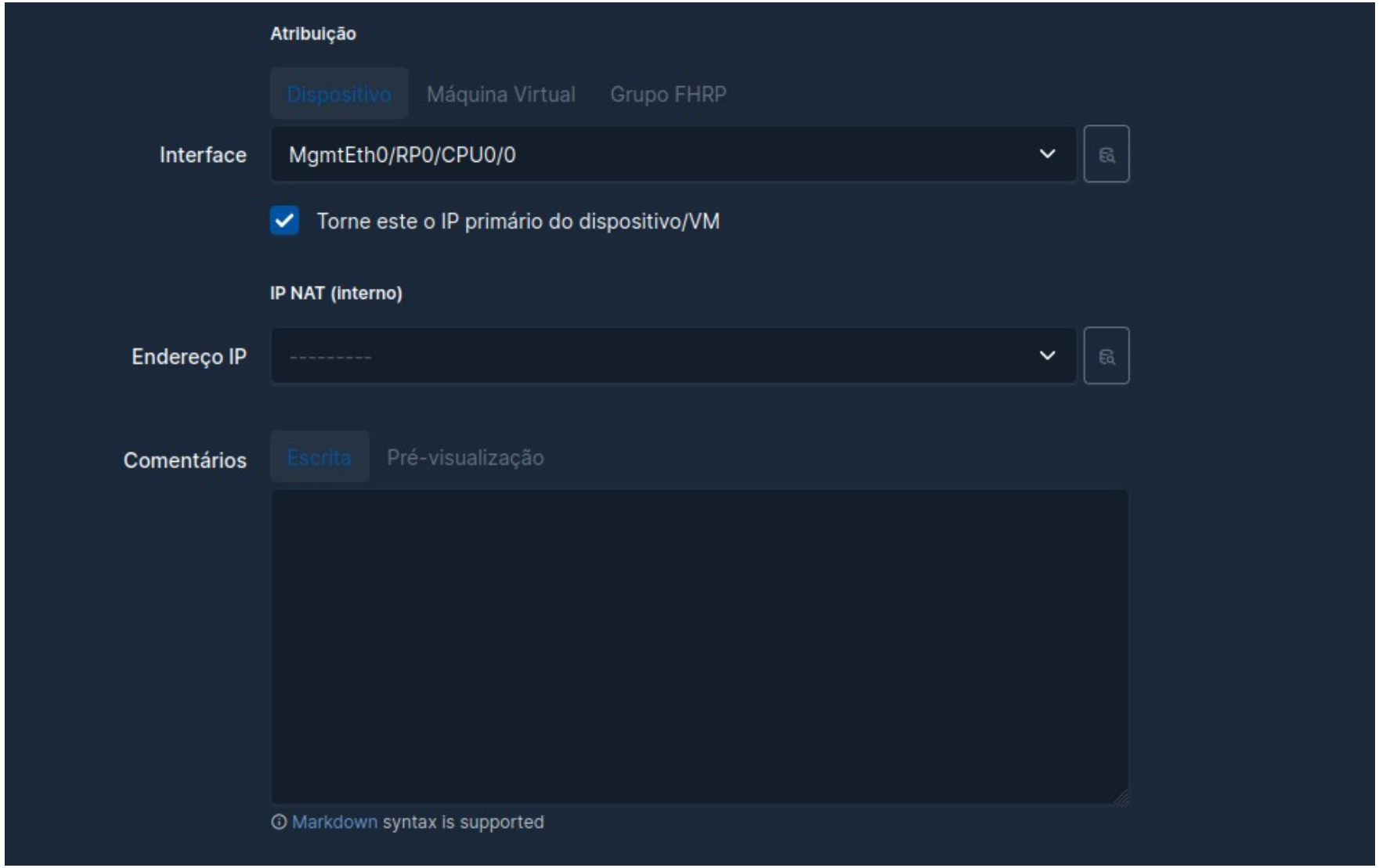
Devices - Netbox



Adicionando IP em Devices - Netbox



Adicionando IP em Devices - Netbox



Adicionando IP em Devices - Netbox

REST API - Sites - Netbox

localhost:8000/api/dcim/sites/

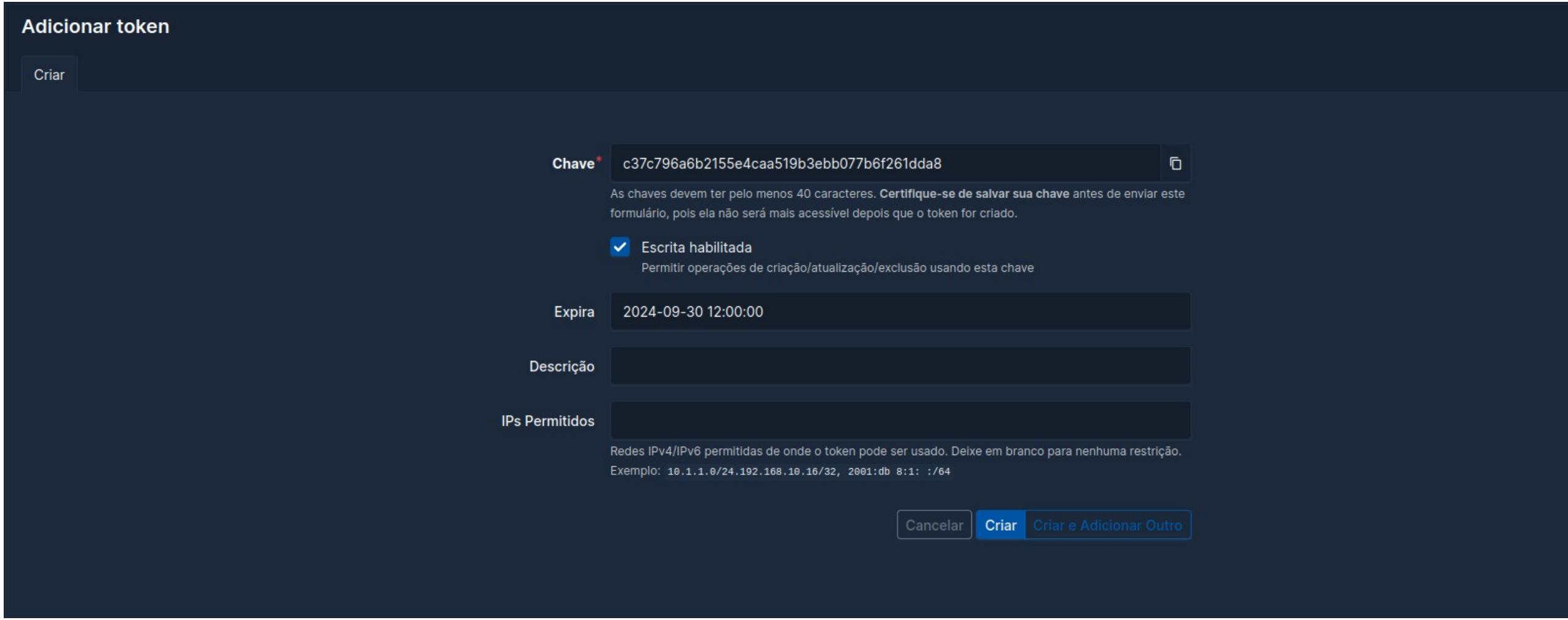

```
{
    "count": 2,
    "next": null,
    "previous": null,
    "results": [
        {
            "id": 1,
            "url": "http://localhost:8000/api/dcim/sites/1/",
            "display": "pop-ba",
            "name": "pop-ba",
            "slug": "pop-ba",
            "status": {
                "value": "active",
                "label": "Ativo"
            },
            "region": {
                "id": 1,
                "url": "http://localhost:8000/api/dcim/regions/1/",
                "display": "Salvador",
                "name": "Salvador",
                "slug": "salvador",
                "description": "",
                "site_count": 0,
                "_depth": 0
            },
            "group": null,
            "tenant": null,
            "facility": "",
            "time_zone": null,
            "description": "",
            "physical_address": "",
            "shipping_address": "",
            "latitude": null,
            "longitude": null,
            "comments": "",
            "asns": [],
            "tags": [],
            "custom_fields": {},
            "created": "2024-09-04T15:21:42.344472Z",
            "last_updated": "2024-09-04T15:21:42.344484Z",
            "circuit_count": 0,
            "device_count": 1,
            "prefix_count": 0,
            "rack_count": 1,
            "virtualmachine_count": 0,
            "vlan_count": 0
        },
```

REST API - Devices - Netbox

localhost:8000/api/dcim/devices/


```
{
    "count": 2,
    "next": null,
    "previous": null,
    "results": [
        {
            "id": 1,
            "url": "http://localhost:8000/api/dcim/devices/1/",
            "display": "ncs5501-popba-95",
            "name": "ncs5501-popba-95",
            "device_type": {
                "id": 1,
                "url": "http://localhost:8000/api/dcim/device-types/1/",
                "display": "ncs5501",
                "manufacturer": {
                    "id": 1,
                    "url": "http://localhost:8000/api/dcim/manufacturers/1/",
                    "display": "cisco",
                    "name": "cisco",
                    "slug": "cisco",
                    "description": ""
                },
                "model": "ncs5501",
                "slug": "ncs5501",
                "description": ""
            },
            "role": {
                "id": 2,
                "url": "http://localhost:8000/api/dcim/device-roles/2/",
                "display": "PE",
                "name": "PE",
                "slug": "pe",
                "description": ""
            },
            "tenant": null,
            "platform": {
                "id": 1,
                "url": "http://localhost:8000/api/dcim/platforms/1/",
                "display": "iosxr",
                "name": "iosxr",
                "slug": "iosxr",
                "description": ""
            },
```

REST API - Token - Netbox



REST API - Netbox – exercicio1.py

netbox > exercicio1.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

import requests

netbox_url_api_devices = "http://localhost:8000/api/dcim/devices/"
token = "c37c796a6b2155e4caa519b3ebb077b6f261dda8"

headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Token '+token
}

response_get_interfaces = requests.request("GET", netbox_url_api_devices, headers=headers, verify=False)


print(response_get_interfaces.headers)
print(response_get_interfaces.status_code)
print(response_get_interfaces.json())
```

REST API - Netbox – exercicio1.py

```
(venv) root@william:/opt/wtr/netbox# python3.10 exercicio1.py
{'Content-Type': 'application/json', 'Vary': 'HX-Request, Accept-Language, Cookie, origin', 'Allow': 'GET, POST, PUT, PATCH, DELETE, HEAD, OPTIONS', 'X-Request-ID': '4489b276-afb7-48c9-a7e5-cb8e083c3508', 'API-Version': '4.0', 'X-Content-
Type-Options': 'nosniff', 'Referrer-Policy': 'same-origin', 'Cross-Origin-Opener-Policy': 'same-origin', 'X-Frame-Options': 'SAMEORIGIN', 'Content-Length': '4227', 'Content-Language': 'en', 'Server': 'Unit/1.32.0', 'Date': 'Wed, 04 Sep 20
24 18:57:01 GMT'}
200
{'count': 2, 'next': None, 'previous': None, 'results': [{'id': 1, 'url': 'http://localhost:8000/api/dcim/devices/1/', 'display': 'ncs5501-popba-95', 'name': 'ncs5501-popba-95', 'device_type': {'id': 1, 'url': 'http://localhost:8000/api/d
cim/device-types/1/', 'display': 'ncs5501', 'manufacturer': {'id': 1, 'url': 'http://localhost:8000/api/dcim/manufacturers/1/', 'display': 'cisco', 'name': 'cisco', 'slug': 'cisco', 'description': ''}, 'model': 'ncs5501', 'slug': 'ncs5501
', 'description': ''}, 'role': {'id': 2, 'url': 'http://localhost:8000/api/dcim/device-roles/2/', 'display': 'PE', 'name': 'PE', 'slug': 'pe', 'description': ''}, 'tenant': None, 'platform': {'id': 1, 'url': 'http://localhost:8000/api/dci
m/platforms/1/', 'display': 'iosxr', 'name': 'iosxr', 'slug': 'iosxr', 'description': ''}, 'serial': '', 'asset_tag': None, 'site': {'id': 1, 'url': 'http://localhost:8000/api/dcim/sites/1/', 'display': 'pop-ba', 'name': 'pop-ba', 'slug':
 'pop-ba', 'description': ''}, 'location': None, 'rack': {'id': 1, 'url': 'http://localhost:8000/api/dcim/racks/1/', 'display': 'RACK IX.br', 'name': 'RACK IX.br', 'description': ''}, 'position': 42.0, 'face': {'value': 'front', 'label':
'Front'}, 'latitude': None, 'longitude': None, 'parent_device': None, 'status': {'value': 'active', 'label': 'Active'}, 'airflow': None, 'primary_ip': {'id': 1, 'url': 'http://localhost:8000/api/ipam/ip-addresses/1/', 'display': '192.168.
246.95/24', 'family': {'value': 4, 'label': 'IPv4'}, 'address': '192.168.246.95/24', 'description': ''}, 'primary_ip4': {'id': 1, 'url': 'http://localhost:8000/api/ipam/ip-addresses/1/', 'display': '192.168.246.95/24', 'family': {'value':
 4, 'label': 'IPv4'}, 'address': '192.168.246.95/24', 'description': ''}, 'primary_ip6': None, 'oob_ip': None, 'cluster': None, 'virtual_chassis': None, 'vc_position': None, 'vc_priority': None, 'description': '', 'comments': '', 'config_
template': None, 'config_context': {}, 'local_context_data': None, 'tags': [], 'custom_fields': {}, 'created': '2024-09-04T18:13:13.076029Z', 'last_updated': '2024-09-04T18:17:10.662861Z', 'console_port_count': 0, 'console_server_port_cou
nt': 0, 'power_port_count': 0, 'power_outlet_count': 0, 'interface_count': 5, 'front_port_count': 0, 'rear_port_count': 0, 'device_bay_count': 0, 'module_bay_count': 0, 'inventory_item_count': 0}, {'id': 2, 'url': 'http://localhost:8000/a
pi/dcim/devices/2/', 'display': 'ncs5501-popmg-96', 'name': 'ncs5501-popmg-96', 'device_type': {'id': 1, 'url': 'http://localhost:8000/api/dcim/device-types/1/', 'display': 'ncs5501', 'manufacturer': {'id': 1, 'url': 'http://localhost:800
0/api/dcim/manufacturers/1/', 'display': 'cisco', 'name': 'cisco', 'slug': 'cisco', 'description': ''}, 'model': 'ncs5501', 'slug': 'ncs5501', 'description': ''}, 'role': {'id': 2, 'url': 'http://localhost:8000/api/dcim/device-roles/2/',
'display': 'PE', 'name': 'PE', 'slug': 'pe', 'description': ''}, 'tenant': None, 'platform': {'id': 1, 'url': 'http://localhost:8000/api/dcim/platforms/1/', 'display': 'iosxr', 'name': 'iosxr', 'slug': 'iosxr', 'description': ''}, 'serial
': '', 'asset_tag': None, 'site': {'id': 2, 'url': 'http://localhost:8000/api/dcim/sites/2/', 'display': 'pop-mg', 'name': 'pop-mg', 'slug': 'pop-mg', 'description': ''}, 'location': None, 'rack': {'id': 2, 'url': 'http://localhost:8000/a
pi/dcim/racks/2/', 'display': 'RACK IX.br', 'name': 'RACK IX.br', 'description': ''}, 'position': 41.0, 'face': {'value': 'front', 'label': 'Front'}, 'latitude': None, 'longitude': None, 'parent_device': None, 'status': {'value': 'active'
, 'label': 'Active'}, 'airflow': None, 'primary_ip': {'id': 2, 'url': 'http://localhost:8000/api/ipam/ip-addresses/2/', 'display': '192.168.246.96/24', 'family': {'value': 4, 'label': 'IPv4'}, 'address': '192.168.246.96/24', 'description'
: ''}, 'primary_ip4': {'id': 2, 'url': 'http://localhost:8000/api/ipam/ip-addresses/2/', 'display': '192.168.246.96/24', 'family': {'value': 4, 'label': 'IPv4'}, 'address': '192.168.246.96/24', 'description': ''}, 'primary_ip6': None, 'oo
b_ip': None, 'cluster': None, 'virtual_chassis': None, 'vc_position': None, 'vc_priority': None, 'description': '', 'comments': '', 'config_template': None, 'config_context': {}, 'local_context_data': None, 'tags': [], 'custom_fields': {}
, 'created': '2024-09-04T18:13:32.655646Z', 'last_updated': '2024-09-04T18:17:49.236462Z', 'console_port_count': 0, 'console_server_port_count': 0, 'power_port_count': 0, 'power_outlet_count': 0, 'interface_count': 5, 'front_port_count':
0, 'rear_port_count': 0, 'device_bay_count': 0, 'module_bay_count': 0, 'inventory_item_count': 0}]}
```

REST API - Netbox – exercicio2.py

netbox > exercicio2.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

import requests

netbox_url_api_devices = "http://localhost:8000/api/dcim/devices/"
token = "c37c796a6b2155e4caa519b3ebb077b6f261dda8"

headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Token '+token
}

response_get_interfaces = requests.request("GET", netbox_url_api_devices, headers=headers, verify=False)


for device in response_get_interfaces.json()['results']:
    #print(device)
    print(f"device: {device['name']} - ip: {device['primary_ip']['address']}")

```

REST API - Netbox – exercicio3.py

netbox > exercicio3.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

import requests

netbox_url_api_devices = "http://localhost:8000/api/dcim/devices/"
token = "c37c796a6b2155e4caa519b3ebb077b6f261dda8"

headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Token '+token
}

response_get_interfaces = requests.request("GET", netbox_url_api_devices, headers=headers, verify=False)


for device in response_get_interfaces.json()['results']:
    ip_mgmt = device['primary_ip']['address']
    ip_mgmt = ip_mgmt.split("/")
    ip_mgmt = ip_mgmt[0]
    print(f"device: {device['name']} - ip: {ip_mgmt}")

```

REST API - Netbox – exercicio4.py

netbox > exercicio4.py > ...

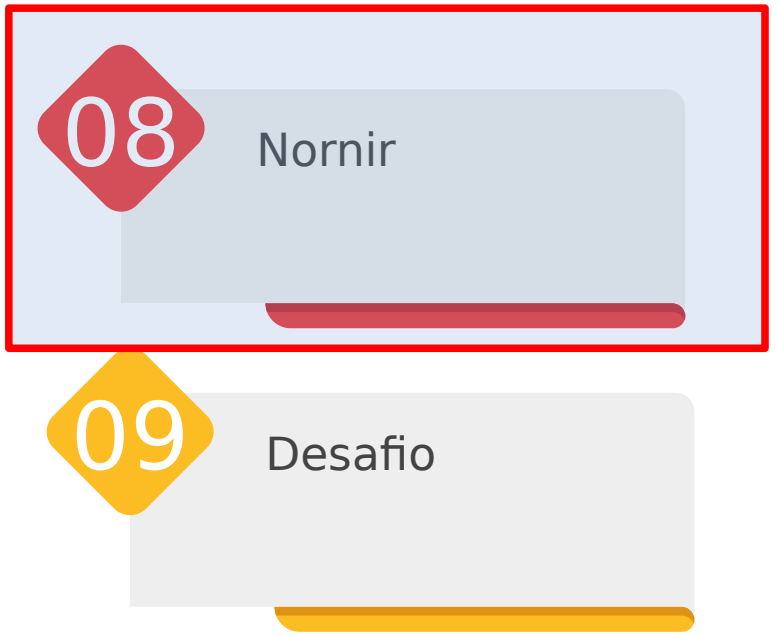
```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

import requests, os
from dotenv import load_dotenv
from netmiko import ConnectHandler
from rich import print

load_dotenv()

netbox_url_api_devices = "http://localhost:8000/api/dcim/devices/"
token = "c37c796a6b2155e4caa519b3ebb077b6f261dda8"

headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Token '+token
}

response_get_interfaces = requests.request("GET", netbox_url_api_devices, headers=headers, verify=False)


for device in response_get_interfaces.json()['results']:
    ip_mgmt = device['primary_ip']['address']
    ip_mgmt = ip_mgmt.split("/")
    ip_mgmt = ip_mgmt[0]

    router = {"device_type": "cisco_xr","host": ip_mgmt,"username": os.getenv("LAB_USERNAME"),"password": os.getenv("LAB_PASSWORD")}

    try:
        print(f"Acessando device: {device['name']}")
        rl_connection = ConnectHandler(**router)

        results = rl_connection.send_command('show ip int brief', use_textfsm=True)

        for interface in results:
            print(interface)

        rl_connection.disconnect()
        print(f"Finalizando o acesso no device: {device['name']}")
    except Exception as err:
        print(err)

```

REST API - Netbox – exercicio5.py

netbox > exercicio5.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

import requests, os
from dotenv import load_dotenv
from napalm import get_network_driver
from rich import print

load_dotenv()

netbox_url_api_devices = "http://localhost:8000/api/dcim/devices/"
token = "c37c796a6b2155e4caa519b3ebb077b6f261dda8"

headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Token '+token
}

response_get_interfaces = requests.request("GET", netbox_url_api_devices, headers=headers, verify=False)


for device in response_get_interfaces.json()['results']:
    ip_mgmt = device['primary_ip']['address']
    ip_mgmt = ip_mgmt.split("/")
    ip_mgmt = ip_mgmt[0]

    router = {"hostname": ip_mgmt,"username": os.getenv("LAB_USERNAME"),"password": os.getenv("LAB_PASSWORD")}
    driver = get_network_driver('iosxr')
    try:
        r_connection = driver(**router)
        r_connection.open()
        output = r_connection.get_facts()
        print(f"device name: {output['hostname']}")
        print(f"device netbox: {device['name']}")

        if str(output['hostname']) != str(device['name']):
            r_connection.load_merge_candidate(config='hostname '+str(device['name']))
            compare = r_connection.compare_config()
            print(compare)
            r_connection.commit_config()
            #r_connection.discard_config()()
        r_connection.close()
    except Exception as err:
        print(err)

```

Agenda



Nornir - o que é?

- Framework de automação 100% em python (debugging);
- Open-source;
- Estrutura multithread;
- Gerenciamento de inventário;
- Suporta YAML e JINJA2 através de plugins;
- Muito rápido;

Inventory: hosts.yaml

nornir > ! hosts.yaml

```yaml
R1:
  hostname: '192.168.246.95'
  port: 22
  username: 'wtr'
  password: 'wtr'
  platform: 'ios'
R2:
  hostname: '192.168.246.96'
  port: 22
  username: 'wtr'
  password: 'wtr'
  platform: 'ios'
```

```python
from nornir import InitNornir
from nornir_rich.functions import print_result
from nornir_netmiko.tasks.netmiko_send_command import netmiko_send_command


nr = InitNornir(
    runner={"plugin": "threaded", "options": {"num_workers": 20}},
    config_file="hosts.yaml")


print_result(nr.run(netmiko_send_command, command_string="show interfaces brief"))

```

Function

Task

https://nornir.readthedocs.io/en/latest/

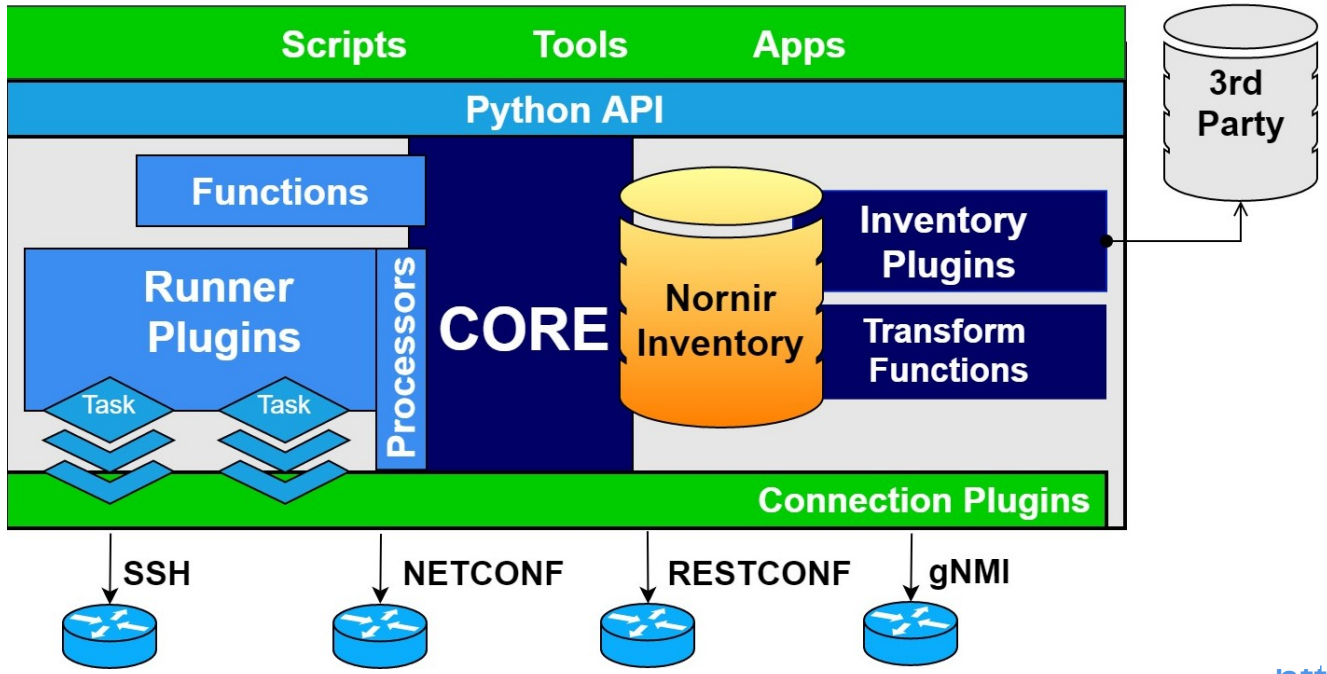
Nornir Plugins

Plugin é um código utilizado para aumentar e melhorar as funcionalidades de um software;

Nornir permite adicionar funcionalidades através de plugins;

Plugins podem ser instalados usando pip:
pip install nornir_netmiko

Nornir Plugins Architecture



https://nornir.tech/nornir/plugins/

Nornir Tasks

- Task define as ações que serão executadas no host;
- Task é uma função python;
- Para executar uma task usamos **run**;
- Tasks são marcadas com failed em caso de exceções na execução;

```python
from nornir import InitNornir
from nornir_rich.functions import print_result
from nornir.core.task import Task, Result

def hello_world(task: Task) -> Result:
    return Result(host=task.host, result=f"{task.host.name} diga olá mundo!")

nr = InitNornir(
    runner={"plugin": "threaded", "options": {"num_workers": 20}},
    config_file="hosts.yaml")

result = nr.run(task=hello_world)

print_result(result)

```

```
hello_world
R1 diga olá mundo!
```

```
hello_world
R2 diga olá mundo!
```

```
hello_world
R3 diga olá mundo!
```

```
hello_world
R4 diga olá mundo!
```

```
hello_world
R5 diga olá mundo!
```

Nornir Tasks

- Task define as ações que serão executadas no host;
- Task é uma função python;
- Para executar uma task usamos **run**;
- Tasks são marcadas com failed em caso de exceções na execução;

```python
from nornir import InitNornir
from nornir_rich.functions import print_result
from nornir.core.task import Task, Result

def hello_world(task: Task) -> Result:
    return Result(host=task.host, result=f"{task.host.name} diga olá mundo!")

nr = InitNornir(
    runner={"plugin": "threaded", "options": {"num_workers": 20}},
    config_file="hosts.yaml")

result = nr.run(task=hello_world)

print_result(result)
```

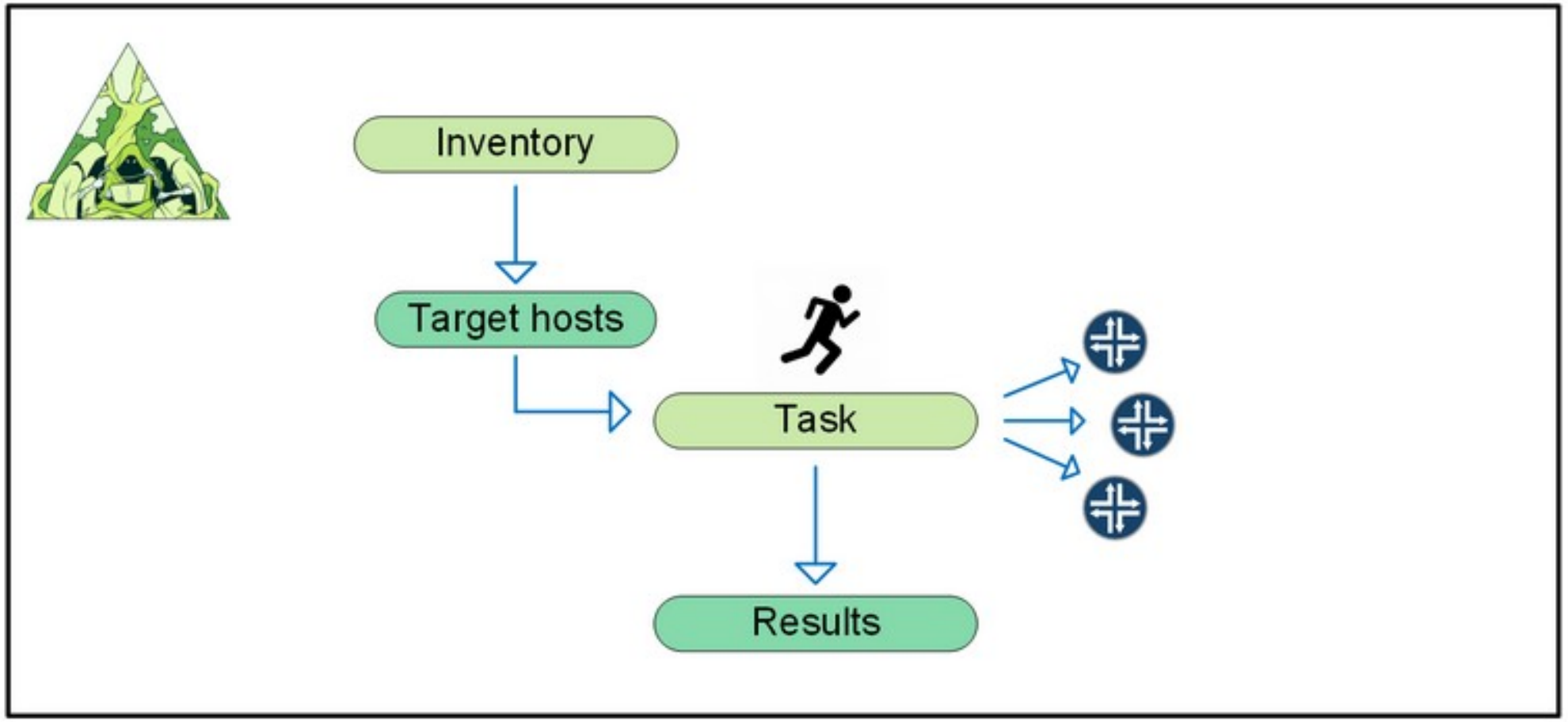
```
hello_world
R1 diga olá mundo!
```

```
hello_world
R2 diga olá mundo!
```

```
hello_world
R3 diga olá mundo!
```

```
hello_world
R4 diga olá mundo!
```

```
hello_world
R5 diga olá mundo!
```


Inventory
Target hosts
Task
Results

Nornir – exemplo1.py

```python
from nornir import InitNornir
from nornir_rich.functions import print_result
from nornir_netmiko.tasks.netmiko_send_command import netmiko_send_command


nr = InitNornir(
    runner={"plugin": "threaded", "options": {"num_workers": 20}},
    config_file="hosts.yaml")


print_result(nr.run(netmiko_send_command, read_timeout=120, command_string="show interfaces brief"))

```

```
netmiko_send_command

Wed Sep  4 20:57:00.230 UTC

              Intf       Intf        LineP              Encap  MTU        BW
              Name       State       State               Type (byte)    (Kbps)
-------------------------------------------------------------------------------
               Nu0          up          up               Null  1500          0
    Mg0/RP0/CPU0/0          up          up               ARPA  1514    1000000
         Gi0/0/0/0  admin-down  admin-down               ARPA  1514    1000000
         Gi0/0/0/1          up          up               ARPA  1514    1000000
         Gi0/0/0/2  admin-down  admin-down               ARPA  1514    1000000
         Gi0/0/0/3  admin-down  admin-down               ARPA  1514    1000000
```

```
netmiko_send_command

Wed Sep  4 20:56:59.982 UTC

              Intf       Intf        LineP              Encap  MTU        BW
              Name       State       State               Type (byte)    (Kbps)
-------------------------------------------------------------------------------
               Nu0          up          up               Null  1500          0
    Mg0/RP0/CPU0/0          up          up               ARPA  1514    1000000
         Gi0/0/0/0  admin-down  admin-down               ARPA  1514    1000000
         Gi0/0/0/1          up          up               ARPA  1514    1000000
         Gi0/0/0/2  admin-down  admin-down               ARPA  1514    1000000
         Gi0/0/0/3  admin-down  admin-down               ARPA  1514    1000000
```

Nornir + Netmiko – exemplo2.py

```python
from nornir import InitNornir
from nornir_rich.functions import print_result
from nornir_netmiko.tasks.netmiko_send_command import netmiko_send_command


nr = InitNornir(
    runner={"plugin": "threaded", "options": {"num_workers": 20}},
    config_file="hosts.yaml")

r1 = nr.filter(name="R1")

print_result(r1.run(netmiko_send_command, read_timeout=120, command_string="show interfaces brief"))

```

```
netmiko_send_command

Wed Sep  4 20:57:00.230 UTC

              Intf       Intf        LineP             Encap  MTU        BW
              Name       State       State              Type (byte)    (Kbps)
--------------------------------------------------------------------------------
               Nu0          up          up              Null  1500          0
     Mg0/RP0/CPU0/0         up          up              ARPA  1514    1000000
          Gi0/0/0/0  admin-down  admin-down              ARPA  1514    1000000
          Gi0/0/0/1         up          up              ARPA  1514    1000000
          Gi0/0/0/2  admin-down  admin-down              ARPA  1514    1000000
          Gi0/0/0/3  admin-down  admin-down              ARPA  1514    1000000
```

Nornir + Netbox + Napalm – exemplo3.py

```python
from nornir import InitNornir
from nornir_napalm.plugins.tasks import napalm_get
from nornir_rich.functions import print_result
import os

nr = InitNornir(
        runner={"plugin": "threaded", "options": {"num_workers": 20}},
        inventory={
            "plugin": "NetBoxInventory2",
            "options": {
                "nb_url": "http://localhost:8000/",
                "nb_token": "c37c796a6b2155e4caa519b3ebb077b6f261dda8",
                "ssl_verify": False}
        })

nr.inventory.defaults.username = "wtr"
nr.inventory.defaults.password = "wtr"
nr.inventory.defaults.port = "22"

print_result(nr.run(task=napalm_get, getters=["get_interfaces"]), vars=["result"])

```

Nornir + Netbox + Inventory – exemplo4.py

```python
from nornir import InitNornir

nr = InitNornir(
        runner={"plugin": "threaded", "options": {"num_workers": 20}},
        inventory={
            "plugin": "NetBoxInventory2",
            "options": {
                "nb_url": "http://localhost:8000/",
                "nb_token": "c37c796a6b2155e4caa519b3ebb077b6f261dda8",
                "ssl_verify": False}
        })

nr.inventory.defaults.username = "wtr"
nr.inventory.defaults.password = "wtr"
nr.inventory.defaults.port = "22"

print(nr.inventory.hosts)

print(nr.inventory.hosts['ncs5501-popba-95'].data)

```

Nornir + Netbox + filters – exemplo5.py

```python
from nornir import InitNornir

nr = InitNornir(
        runner={"plugin": "threaded", "options": {"num_workers": 20}},
        inventory={
            "plugin": "NetBoxInventory2",
            "options": {
                "nb_url": "http://localhost:8000/",
                "nb_token": "c37c796a6b2155e4caa519b3ebb077b6f261dda8",
                "filter_parameters": {"region": ["salvador"],"status": "active", "platform": ["iosxr"]},
                "ssl_verify": False}
        })

nr.inventory.defaults.username = "wtr"
nr.inventory.defaults.password = "wtr"
nr.inventory.defaults.port = "22"

print(nr.inventory.hosts)
```

Nornir + Netbox + Tasks – exemplo6.py

```python
from nornir import InitNornir
from nornir_utils.plugins.functions import print_result
from nornir_netmiko import netmiko_send_config, netmiko_commit

def nornir_task_wtr(task):
    configurations = ["hostname "+str(task.host.name)]
    command = task.run(netmiko_send_config, config_commands=configurations)
    print_result(command)
    commit = task.run(netmiko_commit)
    print_result(commit)


nr = InitNornir(
        runner={"plugin": "threaded", "options": {"num_workers": 20}},
        inventory={
            "plugin": "NetBoxInventory2",
            "options": {
                "nb_url": "http://localhost:8000/",
                "nb_token": "c37c796a6b2155e4caa519b3ebb077b6f261dda8",
                "filter_parameters": {"region": ["salvador"],"status": "active", "platform": ["iosxr"]},
                "ssl_verify": False}
        })

nr.inventory.defaults.username = "wtr"
nr.inventory.defaults.password = "wtr"
nr.inventory.defaults.port = "22"

nr.run(task=nornir_task_wtr)

```

Nornir + Netbox + Inspect – exemplo7.py

nornir > exemplo7.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

from nornir import InitNornir
from nornir_utils.plugins.functions import print_result
from nornir_netmiko import netmiko_send_config, netmiko_commit
from nornir_inspect import nornir_inspect


def nornir_task_wtr(task):
    configurations = ["hostname "+str(task.host.name)]
    command = task.run(netmiko_send_config, config_commands=configurations)
    print_result(command)
    commit = task.run(netmiko_commit)
    print_result(commit)
    print(commit['ncs5501-popba-95'])



nr = InitNornir(
        runner={"plugin": "threaded", "options": {"num_workers": 20}},
        inventory={
            "plugin": "NetBoxInventory2",
            "options": {
                "nb_url": "http://localhost:8000/",
                "nb_token": "c37c796a6b2155e4caa519b3ebb077b6f261dda8",
                "filter_parameters": {"region": ["salvador"],"status": "active", "platform": ["iosxr"]},
                "ssl_verify": False}
        })

nr.inventory.defaults.username = "wtr"
nr.inventory.defaults.password = "wtr"
nr.inventory.defaults.port = "22"

results = nr.run(task=nornir_task_wtr)
nornir_inspect(results)

print(results['ncs5501-popba-95'][1].result)

```

Nornir + Netbox + Netconf – exemplo8.py

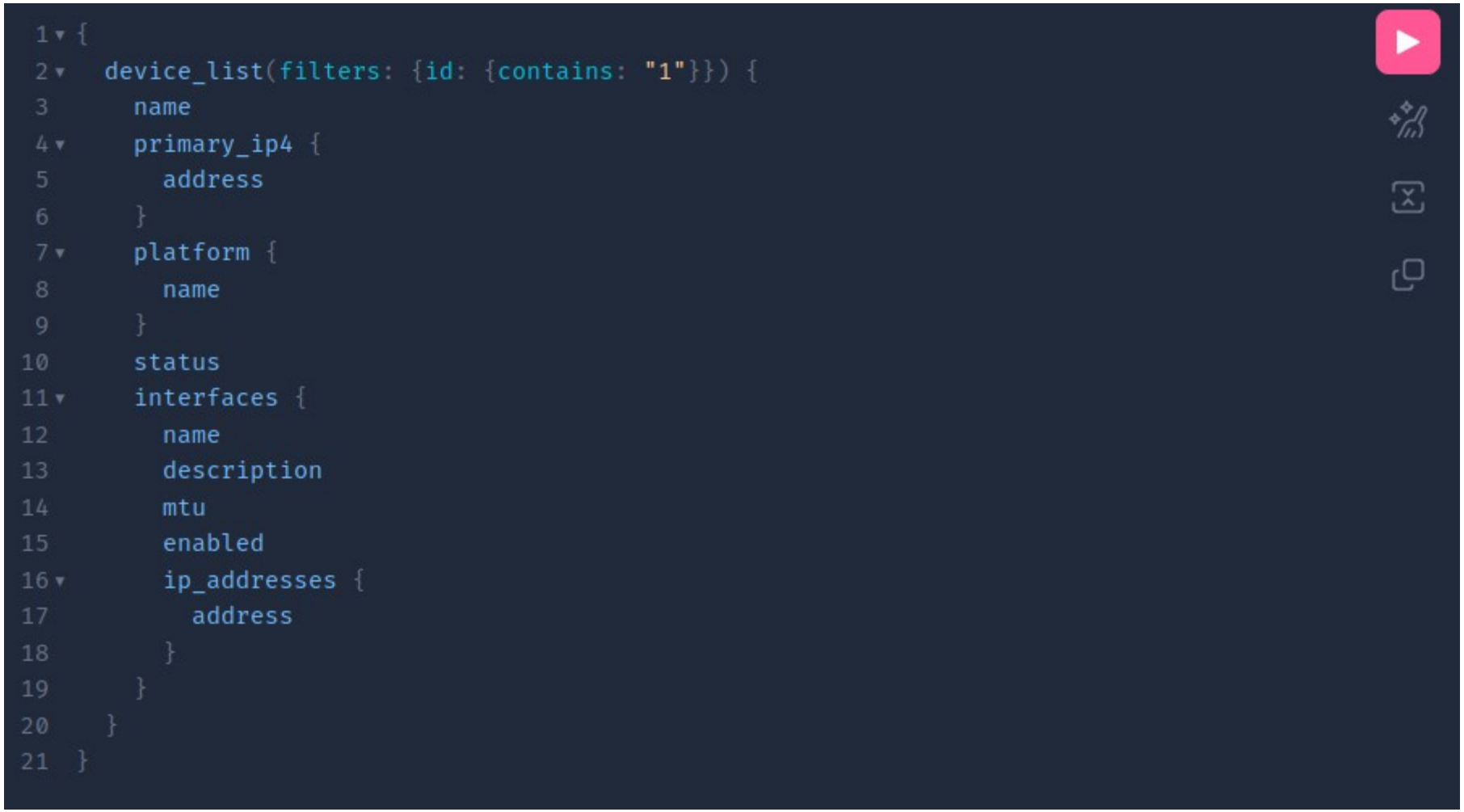
```python
from nornir import InitNornir
from nornir_netconf.plugins.tasks import netconf_get_config
import json,xmltodict

def nornir_task_wtr(task):
    data_interfaces = task.run(netconf_get_config,source="running",filter_type="subtree")
    print(data_interfaces.result.rpc.data_xml)
    obj = xmltodict.parse(data_interfaces.result.rpc.data_xml)
    json_interfaces = json.loads(json.dumps(obj['data']))
    print(json_interfaces)

nr = InitNornir(
        runner={"plugin": "threaded", "options": {"num_workers": 20}},
        inventory={
            "plugin": "NetBoxInventory2",
            "options": {
                "nb_url": "http://localhost:8000/",
                "nb_token": "c37c796a6b2155e4caa519b3ebb077b6f261dda8",
                "filter_parameters": {"region": ["salvador"],"status": "active", "platform": ["iosxr"]},
                "ssl_verify": False}
        })

nr.inventory.defaults.username = "wtr"
nr.inventory.defaults.password = "wtr"
nr.inventory.defaults.port = "830"

nr.run(task=nornir_task_wtr)

```

Nornir + Netbox + Netconf – exemplo9.py

```python
from nornir import InitNornir
from nornir_netconf.plugins.tasks import netconf_get_config
import json,xmltodict

def nornir_task_wtr(task):
    data_interfaces = task.run(
    netconf_get_config,
    source="running",
    path="""
    <interface-configurations xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ifmgr-cfg">
    </interface-configurations>""",
    filter_type="subtree",
    )
    obj = xmltodict.parse(data_interfaces.result.rpc.data_xml)
    json_interfaces = json.loads(json.dumps(obj['data']))

    #print(json_interfaces['interface-configurations']['interface-configuration'])

    for interface in json_interfaces['interface-configurations']['interface-configuration']:
        print(interface)

nr = InitNornir(
        runner={"plugin": "threaded", "options": {"num_workers": 20}},
        inventory={
            "plugin": "NetBoxInventory2",
            "options": {
                "nb_url": "http://localhost:8000/",
                "nb_token": "c37c796a6b2155e4caa519b3ebb077b6f261dda8",
                "filter_parameters": {"region": ["salvador"],"status": "active", "platform": ["iosxr"]},
                "ssl_verify": False}
        })

nr.inventory.defaults.username = "wtr"
nr.inventory.defaults.password = "wtr"
nr.inventory.defaults.port = "830"

nr.run(task=nornir_task_wtr)

```

Netbox - API - Graphql

http://localhost:8000/graphql/



Netbox - API - Graphql


```
{
  "data": {
    "device_list": [
      {
        "name": "ncs5501-popba-95",
        "primary_ip4": {
          "address": "192.168.246.95/24"
        },
        "platform": {
          "name": "iosxr"
        },
        "status": "active",
        "interfaces": [
          {
            "name": "GigabitEthernet0/0/0/0",
            "description": "UPLINK-CUSTOMER-A",
            "mtu": 1535,
            "enabled": true,
            "ip_addresses": []
          },
          {
            "name": "GigabitEthernet0/0/0/1",
            "description": "UPLINK-CUSTOMER-B",
            "mtu": 1535,
            "enabled": true,
            "ip_addresses": []
          },
          {
            "name": "GigabitEthernet0/0/0/2",
            "description": "UPLINK-CUSTOMER-C",
            "mtu": 1535,
            "enabled": true,
            "ip_addresses": []
          },
```

Netbox - API – Graphql – exemplo1.py

graphql > exemplo1.py > ...

```python
#!/opt/wtr-pop-ba-2024/venv/bin/python3.10

import requests

url_graphql = "http://localhost:8000/graphql/"
token = "c37c796a6b2155e4caa519b3ebb077b6f261dda8"

headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Token '+token
}

GRAPHQL_QUERY = {"query": f"""
{{
    device_list(
        filters:
        {{
            id:
            {{
                contains: "1"
            }}
        }}
    )
{{
    name
    primary_ip4
    {{
        address
    }}
    platform
    {{
        name
    }}
    status
    interfaces
    {{
        name
        description
        mtu
        enabled
        ip_addresses
        {{
            address
        }}
    }}
}}
}}"""}

device_interfaces_netbox = requests.post(url=url_graphql, json=GRAPHQL_QUERY, headers=headers, verify=False)

print(device_interfaces_netbox.json())
```

Netbox - API – Graphql – exemplo2.py

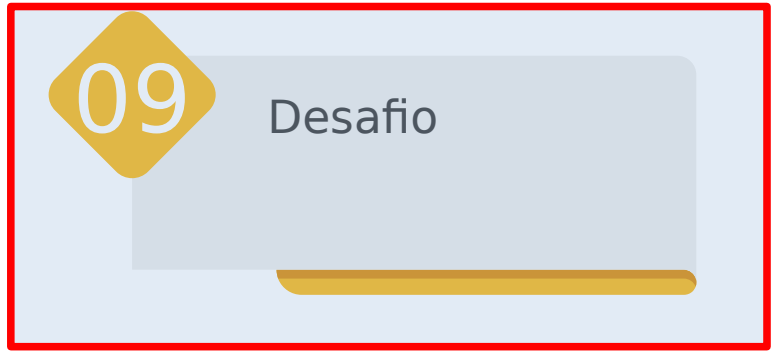
```python
def nornir_task_wtr(task):
    device = nr.inventory.hosts[task.host.name]
    device_id = device['id']
    GRAPHQL_QUERY = {"query": f"""
    {{
        device_list(
            filters:
            {{
                id:
                {{
                    contains: {device_id}
                }}
            }}
        )
    {{
        name
        primary_ip4
        {{
            address
        }}
        platform
        {{
            name
        }}
        status
        interfaces
        {{
            name
            description
            mtu
            enabled
            ip_addresses
            {{
                address
            }}
        }}
    }}
    }}"""}

    device_interfaces_netbox = requests.post(url=url_graphql, json=GRAPHQL_QUERY, headers=headers, verify=False).json()['data']['device_list'][0]['interfaces']
    print(device_interfaces_netbox)
```

Netbox - API – Graphql – exemplo2.py

```python
nr = InitNornir(
        runner={"plugin": "threaded", "options": {"num_workers": 20}},
        inventory={
            "plugin": "NetBoxInventory2",
            "options": {
                "nb_url": "http://localhost:8000/",
                "nb_token": "c37c796a6b2155e4caa519b3ebb077b6f261dda8",
                "filter_parameters": {"region": ["salvador","belohorizonte"],"status": "active", "platform": ["iosxr"]},
                "ssl_verify": False}
        })

nr.inventory.defaults.username = "wtr"
nr.inventory.defaults.password = "wtr"
nr.inventory.defaults.port = "22"

nr.run(task=nornir_task_wtr)
```

```
(venv) root@william:/opt/wtr/extra# python3.10 exemplo2.py
[{'name': 'GigabitEthernet0/0/0/0', 'description': '', 'mtu': None, 'enabled': True, 'ip_addresses': []}, {'name': 'GigabitEthernet0/0/0/1', 'description': '', 'mtu': None, 'enabled': True, 'ip_addresses': []}, {'name': 'GigabitEthernet0/0/0/2', 'description': '', 'mtu': None, 'enabled': True, 'ip_addresses': []}, {'name': 'GigabitEthernet0/0/0/3', 'description': '', 'mtu': None, 'enabled': True, 'ip_addresses': []}, {'name': 'MgmtEth0/RP0/CPU0/0', 'description': '', 'mtu': None, 'enabled': True, 'ip_addresses': [{'address': '192.168.246.96/24'}]}]
[{'name': 'GigabitEthernet0/0/0/0', 'description': 'UPLINK-CUSTOMER-A', 'mtu': 1535, 'enabled': True, 'ip_addresses': []}, {'name': 'GigabitEthernet0/0/0/1', 'description': 'UPLINK-CUSTOMER-B', 'mtu': 1535, 'enabled': True, 'ip_addresses': []}, {'name': 'GigabitEthernet0/0/0/2', 'description': 'UPLINK-CUSTOMER-C', 'mtu': 1535, 'enabled': True, 'ip_addresses': []}, {'name': 'GigabitEthernet0/0/0/3', 'description': 'UPLINK-CUSTOMER-D', 'mtu': 1535, 'enabled': True, 'ip_addresses': []}, {'name': 'MgmtEth0/RP0/CPU0/0', 'description': '', 'mtu': None, 'enabled': True, 'ip_addresses': [{'address': '192.168.246.95/24'}]}]
```

Agenda

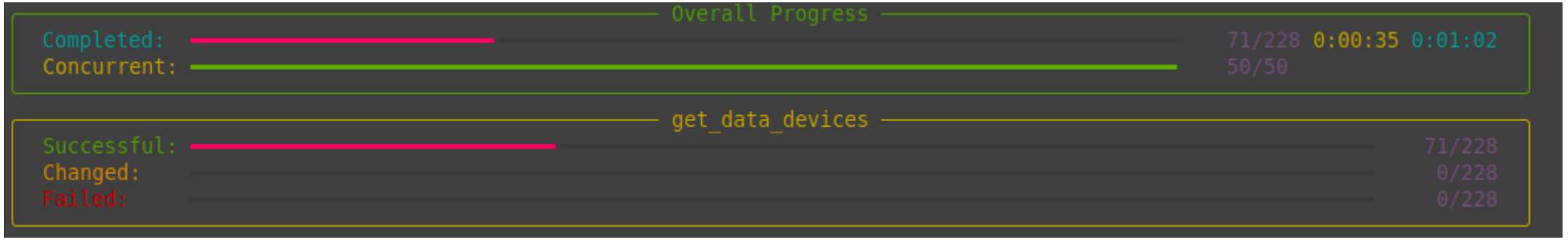


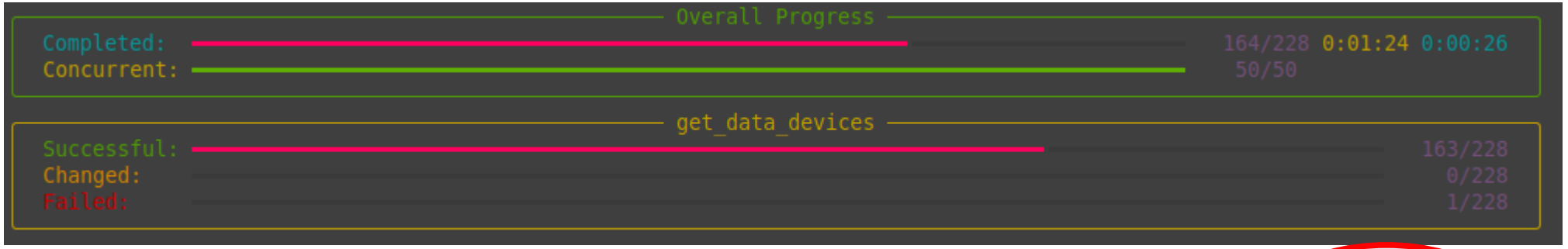
Desafio Automação

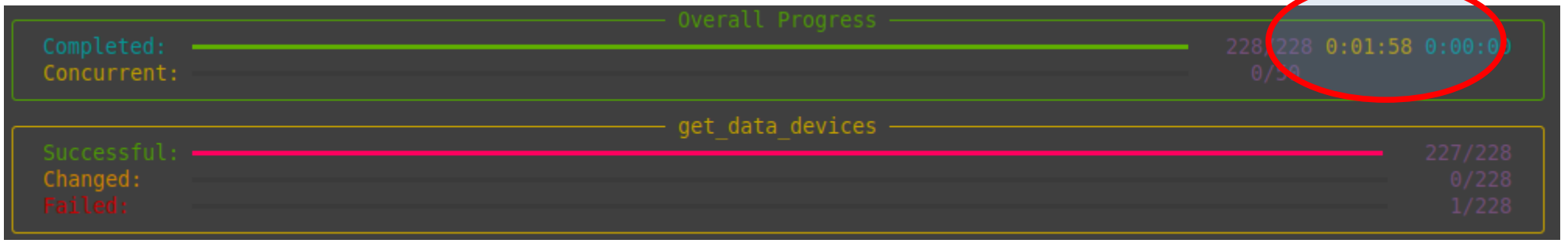
- Criar um Script em Python para ler os dados do netbox e aplicar nos devices de rede se for necessário;

- Requisitos:
  - Nornir
  - Consumir os Dados do Netbox usando GraphQL
  - Aplicar as configurações nos devices usando: Netmiko, Scrapli ou Napalm ou Netconf
  - Dados para validar entre Netbox e Device:
    - Status Operacional: shutdown/ no shutdown
    - MTU
    - Description

Curiosidades: Case de 228 Devices com 50 workers







Obrigado

www.ix.br

wprado@nic.br

11 de setembro de 2024

nic.br cgi.br

www.nic.br | www.cgi.br