

The logo for NIC.br features the text 'nic.br' in a bold, sans-serif font. The 'nic' is in black, and '.br' is in a vibrant green. Below the logo, the full name 'Brazilian Network Information Center' is written in a smaller, black, sans-serif font.

nic.br
Brazilian Network
Information Center

The logo for EGI.br features the text 'egi.br' in a bold, sans-serif font. The 'egi' is in black, and '.br' is in a vibrant green. Below the logo, the full name 'Brazilian Internet Steering Committee' is written in a smaller, black, sans-serif font.

egi.br
Brazilian Internet
Steering Committee

A horizontal row of six logos for various Brazilian internet services. Each logo consists of a name followed by '.br' in a green font. The names are 'registro.br', 'cert.br', 'cetic.br', 'ceptro.br', 'ceweb.br', and 'ix.br'.

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/1nugVe3ZaO0p6yrMJgHPEgGACOCAI1Ftn>)

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

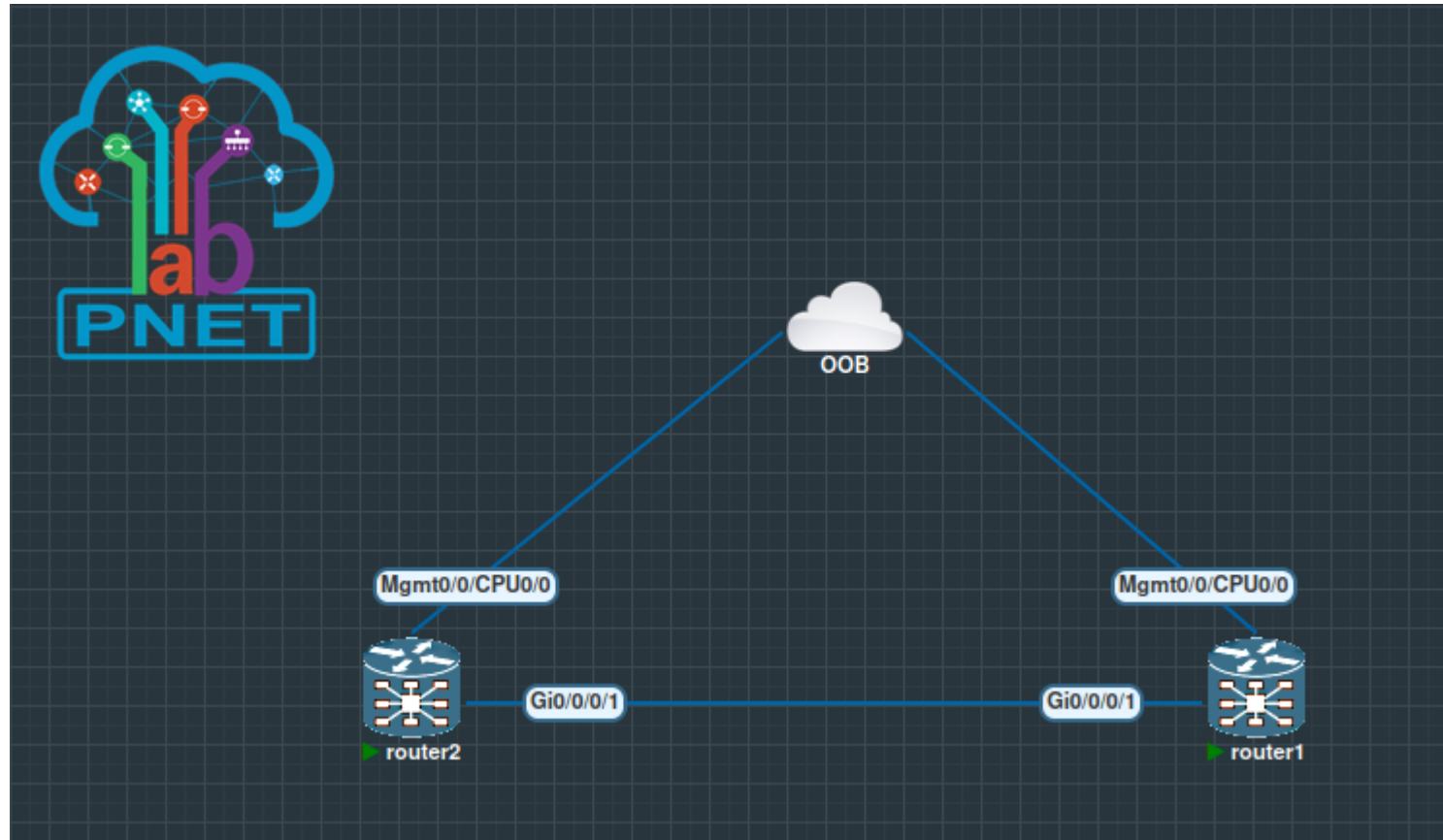
```
root@william:/home/oem# ssh wtr@192.168.1.6
wtr@192.168.1.6's password:
Linux automation 6.1.0-23-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.99-1 (2024-07-15) x86_64

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

The screenshot shows a GitHub repository page for 'wtr-pop-ba-2024' (Public). The repository is on the 'main' branch, has 1 branch and 0 tags. The repository description is 'Desafio + Graphql'. The repository statistics show 7182901 lines of code and 6 commits. The file list includes:

File/Folder	Commit Message	Commit Time
desafio	Desafio + Graphql	now
extra	venv update	19 hours ago
graphql	Desafio + Graphql	now
napalm	Minucurso Automacao	19 hours ago
netbox	Minucurso Automacao	19 hours ago
netmiko	Minucurso Automacao	19 hours ago
nornir	Desafio + Graphql	now
python_exercise	Exercicios	2 weeks ago
scrapli	Minucurso Automacao	19 hours ago
templates	Exercicios	2 weeks ago
venv	venv update	19 hours ago
.gitignore	Initial commit	2 weeks ago
README.md	Desafio + Graphql	now

<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

TM

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

Python – exemplo1.py (Tipo de Dados)

```
python_exercise > exemplo1.py > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  #String
4  nome = "William Stephan do Prado"
5
6  #Inteiro
7  idade = 32
8
9  #Float
10 altura = 1.82
11
12 #Boolean
13 futebol = True
14
15 #mostrar no terminal o conteudo da variavel nome e idade
16 print(f"Nome: {nome} - Idade: {idade} - Altura: {altura} - Futebol: {futebol}")
17
18 #mostrar no terminal o tipo da variavel nome e idade
19 print(f"type nome: {type(nome)} - type idade: {type(idade)} - type altura: {type(altura)} - type futebol: {type(futebol)}")
20 |
```

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

```
python_exercise > exemplo2.py > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  #Entrada de Dados
4  nome = input("Qual seu nome?")
5
6  #Entrada de Dados
7  salario = input("Qual seu Salário?")
8
9  #Mostrar Dados no Terminal
10 print(f"Nome: {nome} - Salário: {salario} R$")
11
12
13
```

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

```
python_exercise > exemplo3.py > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  aluno1 = "William Prado"
4  aluno2 = "Miguel Prado"
5  aluno3 = "Jose Prado"
6  aluno4 = "Eduardo Reis"
7  aluno5 = "Carlos Andre"
8  aluno6 = "Pedro Elias"
9
10 #Lista de Strings
11 lista_alunos = [aluno1,aluno2,aluno3,aluno4,aluno5]
12 #index          0      1      2      3      4
13
14 #Mostrar Lista no Terminal
15 print(lista_alunos)
16
17 #Mostrar Lista no Terminal no index 4
18 print(lista_alunos[4])
19
20 #Loop de Repetição na Lista
21 for aluno in lista_alunos:
22     print(aluno)
23
24 #Validação
25 if aluno6 in lista_alunos:
26     print("Aluno Encontrado")
27 else:
28     print(f"Aluno: {aluno6} não encontrado na lista")
29
```


Python – exemplo4.py (Dicionários e Listas)

```
python_exercise > exemplo4.py > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  #Dicionário - dict
4  dados_aluno1 = {"nome": "William Prado", "idade": 32, "altura": 1.82, "casado": True, "filhos": ["Miguel", "José"]}
5
6  #Dicionário - dict
7  dados_aluno2 = {"nome": "Eduardo Pereira", "idade": 40, "altura": 1.75, "casado": False, "filhos": []}
8
9  #Mostrar Dados no Terminal
10 print(dados_aluno1)
11 print(dados_aluno2)
12
13 print(dados_aluno1['filhos'])
14 print(dados_aluno2['filhos'])
15
16 #Lista de Dicionários
17 lista_alunos = [dados_aluno1, dados_aluno2]
18
19 print(lista_alunos)
20
```

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

```
python_exercise > exemplo5.py > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  #Dicionário - dict
4  filho1_aluno1 = {"nome": "Miguel Guedes", "idade": 5, "altura": 0.80, "casado": False}
5  filho2_aluno1 = {"nome": "José Guedes", "idade": 2, "altura": 0.60, "casado": False}
6
7  #Dicionário - dict + Listas - list
8  dados_aluno1 = {"nome": "William Prado", "idade": 32, "altura": 1.82, "casado": True, "filhos": [filho1_aluno1,filho2_aluno1]}
9  dados_aluno2 = {"nome": "Eduardo Pereira", "idade": 40, "altura": 1.75, "casado": False, "filhos": []}
10
11 #Lista
12 lista_alunos = [dados_aluno1,dados_aluno2]
13
14 #Mostrar dados no Terminal
15 print(lista_alunos)
16
17 #Loop na Lista alunos
18 for aluno in lista_alunos:
19     print(f"Aluno: {aluno['nome']} - Idade: {aluno['idade']} - Altura: {aluno['altura']}")
20     for filho in aluno['filhos']:
21         print(f"Nome filho: {filho['nome']} - Idade: {filho['idade']}")
22
23
```

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



Agenda

03 Netmiko

04 Napalm

05 Scrapli

06 NETCONF

07 NetBox

08 Nornir

09 Desafio

NETM&KO

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 vendedores);
- **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 > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  from netmiko import ConnectHandler
4  from rich import print
5
6  #dict com os dados do device para acesso
7  router1 = {
8      "device_type": "cisco_xr",
9      "host": "192.168.246.95",
10     "username": "wtr",
11     "password": "wtr"
12 }
13
14 try:
15     #conexão com o router1 e router2
16     r1_connection = ConnectHandler(**router1)
17
18     #executar o comando show version no router1 e router2 e mostrar o resultado
19     results = r1_connection.send_command('show version')
20
21     print(type(results))
22
23     #finalizar a conexão com o router1 e router2
24     r1_connection.disconnect()
25
26 except Exception as err:
27     print(err)
28
```

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
```

```
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:

```
⚙️ .env  
1 LAB_USERNAME="wtr"  
2 LAB_PASSWORD="wtr"  
3
```

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 > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  import os
4  from dotenv import load_dotenv
5  from netmiko import ConnectHandler
6  from rich import print
7
8  #carregar as variáveis definidas no .env
9  load_dotenv()
10
11 #dict com os dados do device para acesso
12 router1 = {"device_type": "cisco_xr", "host": "192.168.246.95", "username": os.getenv("LAB_USERNAME"), "password": os.getenv("LAB_PASSWORD")}
13 router2 = {"device_type": "cisco_xr", "host": "192.168.246.96", "username": os.getenv("LAB_USERNAME"), "password": os.getenv("LAB_PASSWORD")}
14
15 try:
16     #conexão com o router1 e router2
17     r1_connection = ConnectHandler(**router1)
18     r2_connection = ConnectHandler(**router2)
19
20     #executar o comando show version no router1 e router2 e mostrar o resultado
21     print(r1_connection.send_command('show version'))
22     print(r2_connection.send_command('show version'))
23
24     #finalizar a conexão com o router1 e router2
25     r1_connection.disconnect()
26     r2_connection.disconnect()
27
28 except Exception as err:
29     print(err)
30
```


Netmiko – exercicio3.py (send_config_set)

```
netmiko > exercicio3.py > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  import os
4  from dotenv import load_dotenv
5  from netmiko import ConnectHandler
6  from rich import print
7
8  load_dotenv()
9
10 router1 = {"device_type": "cisco_xr", "host": "192.168.246.95", "username": os.getenv("LAB_USERNAME"), "password": os.getenv("LAB_PASSWORD")}
11 router2 = {"device_type": "cisco_xr", "host": "192.168.246.96", "username": os.getenv("LAB_USERNAME"), "password": os.getenv("LAB_PASSWORD")}
12
13 try:
14     #conexão com o router1 e router2
15     r1_connection = ConnectHandler(**router1)
16     r2_connection = ConnectHandler(**router2)
17
18     commands_router1 = ["hostname R1"]
19     commands_router2 = ["hostname R2"]
20
21     #enviar a lista de comandos para o router1 e realizar o commit
22     print(r1_connection.send_config_set(commands_router1))
23     print(r1_connection.commit())
24
25     #enviar a lista de comandos para o router2 e realizar o commit
26     print(r2_connection.send_config_set(commands_router2))
27     print(r2_connection.commit())
28
29     #finalizar a conexão com o router1 e router2
30     r1_connection.disconnect()
31     r2_connection.disconnect()
32
33 except Exception as err:
34     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 > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  import os
4  from dotenv import load_dotenv
5  from netmiko import ConnectHandler
6  from rich import print
7
8  load_dotenv()
9
10 router1 = {"device_type": "cisco_xr", "host": "192.168.246.95", "username": os.getenv("LAB_USERNAME"), "password": os.getenv("LAB_PASSWORD")}
11 router2 = {"device_type": "cisco_xr", "host": "192.168.246.96", "username": os.getenv("LAB_USERNAME"), "password": os.getenv("LAB_PASSWORD")}
12
13 try:
14     #conexão com o router1 e router2
15     r1_connection = ConnectHandler(**router1)
16     r2_connection = ConnectHandler(**router2)
17
18     commands_router1 = ["interface gigabitEthernet 0/0/0/1", "ipv4 address 192.168.100.1 255.255.255.252", "no shutdown"]
19     commands_router2 = ["interface gigabitEthernet 0/0/0/1", "ipv4 address 192.168.100.2 255.255.255.252", "no shutdown"]
20
21     #enviar a lista de comandos para o router1 e realizar o commit
22     print(r1_connection.send_config_set(commands_router1))
23     print(r1_connection.commit())
24
25     #enviar a lista de comandos para o router2 e realizar o commit
26     print(r2_connection.send_config_set(commands_router2))
27     print(r2_connection.commit())
28
29     #finalizar a conexão com o router1 e router2
30     r1_connection.disconnect()
31     r2_connection.disconnect()
32
33 except Exception as err:
34     print(err)
35
```

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 > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  import os
4  from dotenv import load_dotenv
5  from netmiko import ConnectHandler
6  from rich import print
7
8  load_dotenv()
9
10 router1 = {"device_type": "cisco_xr", "host": "192.168.246.95", "username": os.getenv("LAB_USERNAME"), "password": os.getenv("LAB_PASSWORD")}
11
12 try:
13     r1_connection = ConnectHandler(**router1)
14
15     print(r1_connection.send_command('show running-config interface'))
16
17     r1_connection.disconnect()
18
19 except Exception as err:
20     print(err)
21
```

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 > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  import os
4  from dotenv import load_dotenv
5  from netmiko import ConnectHandler
6  from rich import print
7
8  load_dotenv()
9
10 router1 = {"device_type": "cisco_xr", "host": "192.168.246.95", "username": os.getenv("LAB_USERNAME"), "password": os.getenv("LAB_PASSWORD")}
11
12 try:
13     r1_connection = ConnectHandler(**router1)
14     results = r1_connection.send_command('show running-config interface', use_ttp=True, ttp_template="./template.ttp")
15     print(results)
16
17     print(f'type results: {type(results)}')
18
19     #finalizar a conexão com o PE1
20     r1_connection.disconnect()
21
22 except Exception as err:
23     print(err)
24
25
26
```

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 > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  import os
4  from dotenv import load_dotenv
5  from netmiko import ConnectHandler
6  from rich import print
7
8  load_dotenv()
9
10 router1 = {"device_type": "cisco_xr", "host": "192.168.246.95", "username": os.getenv("LAB_USERNAME"), "password": os.getenv("LAB_PASSWORD")}
11
12 try:
13     r1_connection = ConnectHandler(**router1)
14
15     results1 = r1_connection.send_command('show ip int brief', use_textfsm=True)
16     results2 = r1_connection.send_command('show version', use_textfsm=True)
17     results3 = r1_connection.send_command('show arp', use_textfsm=True)
18     results4 = r1_connection.send_command('ping 192.168.246.1', use_textfsm=True)
19
20     print(results3)
21
22     r1_connection.disconnect()
23
24 except Exception as err:
25     print(err)
26
```

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

Netmiko – TextFSM – exercicio7.py

```
netmiko > exercicio7.py > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  import os
4  from dotenv import load_dotenv
5  from netmiko import ConnectHandler
6  from rich import print
7
8  load_dotenv()
9
10 router1 = {"device_type": "cisco_xr", "host": "192.168.246.95", "username": os.getenv("LAB_USERNAME"), "password": os.getenv("LAB_PASSWORD")}
11
12 try:
13     r1_connection = ConnectHandler(**router1)
14
15     results1 = r1_connection.send_command('show ip int brief', use_textfsm=True)
16     results2 = r1_connection.send_command('show version', use_textfsm=True)
17     results3 = r1_connection.send_command('show arp', use_textfsm=True)
18     results4 = r1_connection.send_command('ping 192.168.246.1', use_textfsm=True)
19
20     print(results3)
21
22     r1_connection.disconnect()
23
24 except Exception as err:
25     print(err)
26
```

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.3led.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

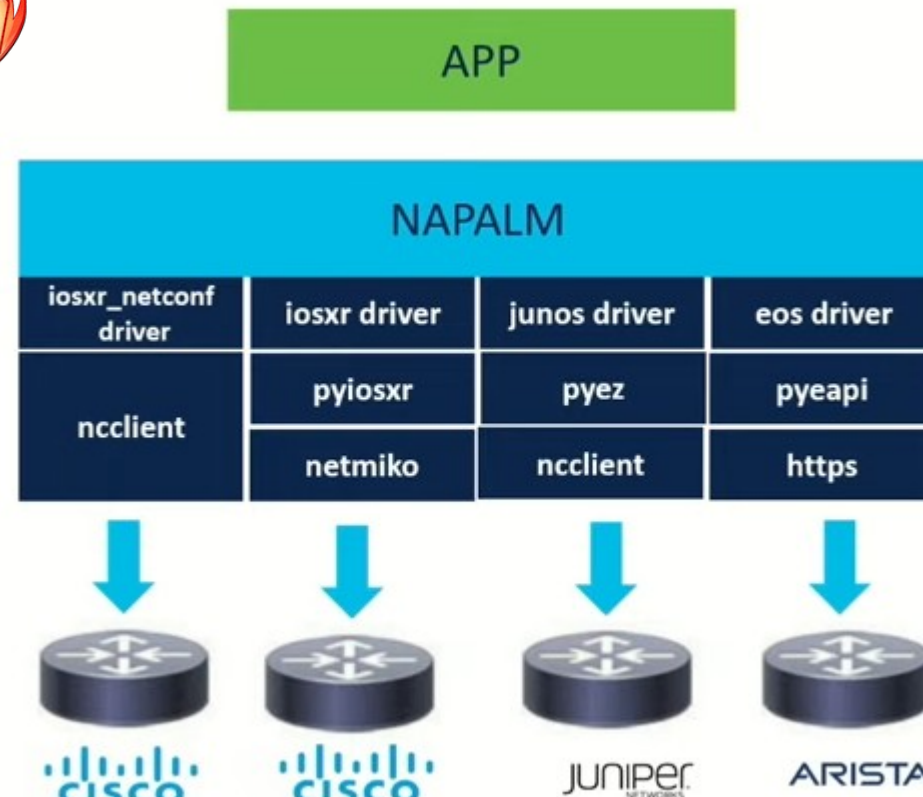
Desafio

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 > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  from napalm import get_network_driver
4  import json, os
5  from dotenv import load_dotenv
6
7  load_dotenv()
8
9  driver = get_network_driver('iosxr')
10
11  router1 = {"hostname": "192.168.246.95", "username": os.getenv("LAB_USERNAME"), "password": os.getenv("LAB_PASSWORD")}
12
13  try:
14      r1_connection = driver(**router1)
15      r1_connection.open()
16      print(f"Connecting to {router1['hostname']}")
17      #output = r1_connection.get_interfaces()
18      #output = r1_connection.get_facts()
19      #output = r1_connection.get_interfaces_ip()
20      output = r1_connection.get_interfaces_counters()
21      r1_connection.close()
22
23      output_json = json.dumps(output, indent=4)
24      print(output_json)
25  except Exception as err:
26      print(err)
27
```



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

05 Scrapli

06 NETCONF

07 NetBox

08 Nornir

09 Desafio

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 > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  from scrapli.driver.core import IOSXRDriver
4  import os, json
5  from dotenv import load_dotenv
6
7  load_dotenv()
8
9  router1 = {"host": "192.168.246.95", "auth_username": os.getenv("LAB_USERNAME"), "auth_password": os.getenv("LAB_PASSWORD"), "port": 22}
10
11  try:
12      with IOSXRDriver(**router1) as conn:
13          response = conn.send_command("show version")
14
15          results = response.result
16
17          print(results)
18
19          results_textfsm = response.textfsm_parse_output()
20
21          output_json = json.dumps(results_textfsm, indent=4)
22
23          print(output_json)
24  except Exception as err:
25      print(err)
26
```

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

[
  {
    "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"
  }
]
```

String

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 > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  from scrapli.driver.core import IOSXRDriver
4  import os
5  from dotenv import load_dotenv
6
7  load_dotenv()
8
9  router1 = {"host": "192.168.246.95", "auth_username": os.getenv("LAB_USERNAME"), "auth_password": os.getenv("LAB_PASSWORD"), "transport": "telnet"}
10
11  try:
12      with IOSXRDriver(**router1) as conn:
13          response = conn.send_command("show version")
14          print(response)
15  except Exception as err:
16      print(err)
17
```


Device Types

Device type support?

- Netmiko: 110+
- Scrapli: 5

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		

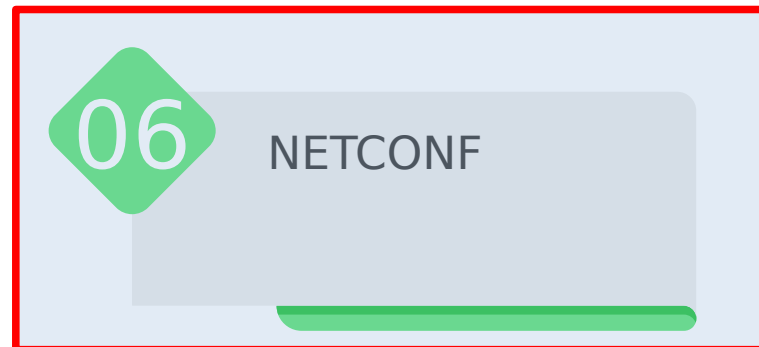
Scrapli x Netmiko

Extensions

Other things that the libraries can do:

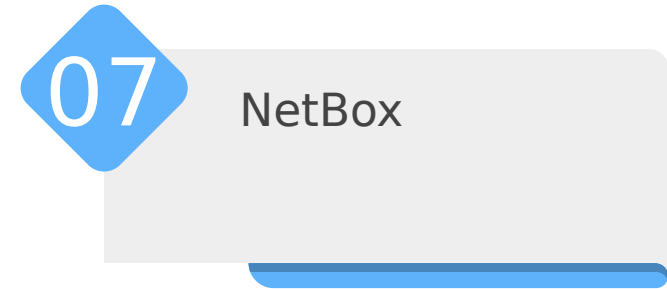
	Netmiko	Scrapli
Secure Copy (SCP)		
NETCONF		
Config Merge/Replace		
Nornir Plugin		

Agenda



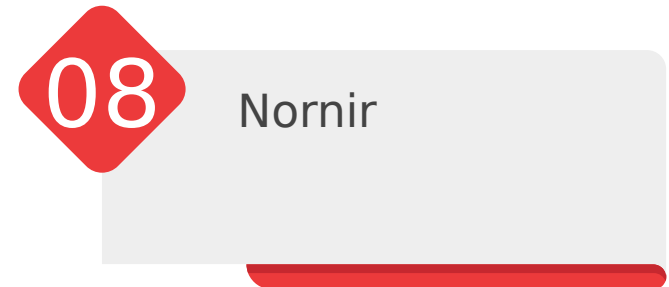
06 NETCONF

This item is highlighted with a red border. It features a green diamond icon with the number 06 and the text NETCONF on a light blue background with a green underline.




07 NetBox

This item features a blue diamond icon with the number 07 and the text NetBox on a light gray background with a blue underline.



08 Nornir

This item features a red diamond icon with the number 08 and the text Nornir on a light gray background with a red underline.



09 Desafio

This item features a yellow diamond icon with the number 09 and the text Desafio on a light gray background with a yellow underline.

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 > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  from scrapli.netconf.driver import NetconfDriver
4  import json, os, xmltodict
5  from dotenv import load_dotenv
6
7  load_dotenv()
8
9  router1 = {"host": "192.168.246.95", "auth_username": os.getenv("LAB_USERNAME"), "auth_password": os.getenv("LAB_PASSWORD"), "port": 830}
10
11  try:
12      with NetconfDriver(**router1) as conn:
13
14          response = conn.get_config(source="running")
15
16          xml_obj = response.result
17
18          print(xml_obj)
19
20          dict_obj = xmltodict.parse(xml_obj)
21
22          print(dict_obj['rpc-reply']['data'])
23
24          output_json = json.dumps(dict_obj['rpc-reply']['data'], indent=4)
25
26          print(output_json)
27
28  except Exception as err:
29      print(err)
30
31
```

Scrapli & Netconf– exercicio3.py

```
scrapli > exercicio3.py > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  from scrapli.netconf.driver import NetconfDriver
4  import json, os, xmltodict
5  from dotenv import load_dotenv
6
7  load_dotenv()
8
9  router1 = {"host": "192.168.246.95", "auth_username": os.getenv("LAB_USERNAME"), "auth_password": os.getenv("LAB_PASSWORD"), "port": 830}
10
11  try:
12      with NetconfDriver(**router1) as conn:
13
14          response = conn.get_config(source="running")
15
16          xml_obj = response.result
17
18          print(xml_obj)
19
20          dict_obj = xmltodict.parse(xml_obj)
21
22          print(dict_obj['rpc-reply']['data'])
23
24          output_json = json.dumps(dict_obj['rpc-reply']['data'], indent=4)
25
26          print(output_json)
27
28  except Exception as err:
29      print(err)
30
31
```

Scrapli & Netconf– exercicio3.py

```
scrapli > exercicio3.py > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  from scrapli_netconf.driver import NetconfDriver
4  import json, os, xmltodict
5  from dotenv import load_dotenv
6
7  load_dotenv()
8
9  router1 = {"host": "192.168.246.95", "auth_username": os.getenv("LAB_USERNAME"), "auth_password": os.getenv("LAB_PASSWORD"), "port": 830}
10
11  try:
12      with NetconfDriver(**router1) as conn:
13          response = conn.get_config(source="running")
14          xml_obj = response.result
15
16          print(xml_obj)
17
18          dict_obj = xmltodict.parse(xml_obj)
19
20          print(dict_obj['rpc-reply']['data'])
21
22          output_json = json.dumps(dict_obj['rpc-reply']['data'], indent=4)
23
24          print(output_json)
25
26  except Exception as err:
27      print(err)
28
29
30
31
```

Scrapli & Netconf– exercicio3.py

```
scrapli > exercicio3.py > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  from scrapli netconf.driver import NetconfDriver
4  import json, os, xmltodict
5  from dotenv import load_dotenv
6
7  load_dotenv()
8
9  router1 = {"host": "192.168.246.95", "auth_username": os.getenv("LAB_USERNAME"), "auth_password": os.getenv("LAB_PASSWORD"), "port": 830}
10
11  try:
12      with NetconfDriver(**router1) as conn:
13
14          response = conn.get_config(source="running")
15
16          xml_obj = response.result
17
18          print(xml_obj)
19
20          dict_obj = xmltodict.parse(xml_obj)
21
22          print(dict_obj['rpc-reply']['data'])
23
24          output_json = json.dumps(dict_obj['rpc-reply']['data'], indent=4)
25
26          print(output_json)
27
28
29  except Exception as err:
30      print(err)
31
```


Scrapli & Netconf– exercicio3.py

```
scrapli > exercicio3.py > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  from scrapli_netconf.driver import NetconfDriver
4  import json, os, xmltodict
5  from dotenv import load_dotenv
6
7  load_dotenv()
8
9  router1 = {"host": "192.168.246.95", "auth_username": os.getenv("LAB_USERNAME"), "auth_password": os.getenv("LAB_PASSWORD"), "port": 830}
10
11  try:
12      with NetconfDriver(**router1) as conn:
13
14          response = conn.get_config(source="running")
15
16          xml_obj = response.result
17
18          print(xml_obj)
19
20          dict_obj = xmltodict.parse(xml_obj)
21
22          print(dict_obj['rpc-reply']['data'])
23
24          output_json = json.dumps(dict_obj['rpc-reply']['data'], indent=4)
25
26          print(output_json)
27
28
29  except Exception as err:
30      print(err)
31
```


Scrapli & Netconf– exercicio3.py

```
scrapli > exercicio3.py > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  from scrapli_netconf.driver import NetconfDriver
4  import json, os, xmltodict
5  from dotenv import load_dotenv
6
7  load_dotenv()
8
9  router1 = {"host": "192.168.246.95", "auth_username": os.getenv("LAB_USERNAME"), "auth_password": os.getenv("LAB_PASSWORD"), "port": 830}
10
11  try:
12      with NetconfDriver(**router1) as conn:
13
14          response = conn.get_config(source="running")
15
16          xml_obj = response.result
17
18          print(xml_obj)
19
20          dict_obj = xmltodict.parse(xml_obj)
21
22          print(dict_obj['rpc-reply']['data'])
23          output_json = json.dumps(dict_obj['rpc-reply']['data'], indent=4)
24
25          print(output_json)
26
27
28
29  except Exception as err:
30      print(err)
31
```

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

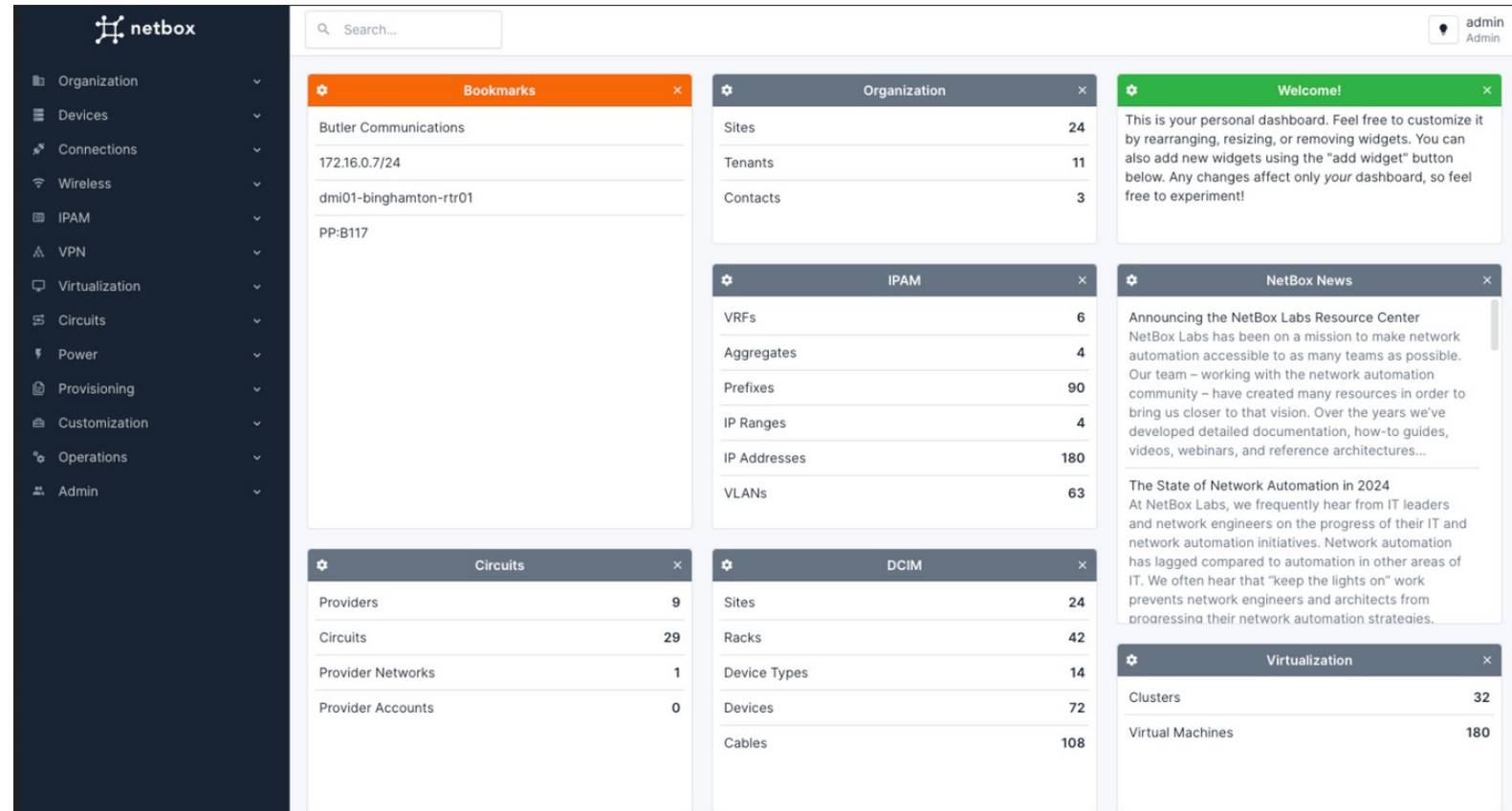
Nornir

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;



The screenshot displays the NetBox web interface. On the left is a dark sidebar with a navigation menu including Organization, Devices, Connections, Wireless, IPAM, VPN, Virtualization, Circuits, Power, Provisioning, Customization, Operations, and Admin. The main content area features a search bar at the top right and a user profile for 'admin Admin'. The dashboard is composed of several widgets:

- Bookmarks:** A list of bookmarks including 'Butler Communications', '172.16.0.7/24', 'dmi01-binghamton-rtr01', and 'PP:B117'.
- Organization:** A table showing counts for Sites (24), Tenants (11), and Contacts (3).
- IPAM:** A table showing counts for VRFs (6), Aggregates (4), Prefixes (90), IP Ranges (4), IP Addresses (180), and VLANs (63).
- DCIM:** A table showing counts for Sites (24), Racks (42), Device Types (14), Devices (72), and Cables (108).
- Circuits:** A table showing counts for Providers (9), Circuits (29), Provider Networks (1), and Provider Accounts (0).
- Welcome!:** A green notification box with text: 'This is your personal dashboard. Feel free to customize it by rearranging, resizing, or removing widgets. You can also add new widgets using the "add widget" button below. Any changes affect only your dashboard, so feel free to experiment!'.
- NetBox News:** A news widget with two articles: 'Announcing the NetBox Labs Resource Center' and 'The State of Network Automation in 2024'.
- Virtualization:** A table showing counts for Clusters (32) and Virtual Machines (180).

<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

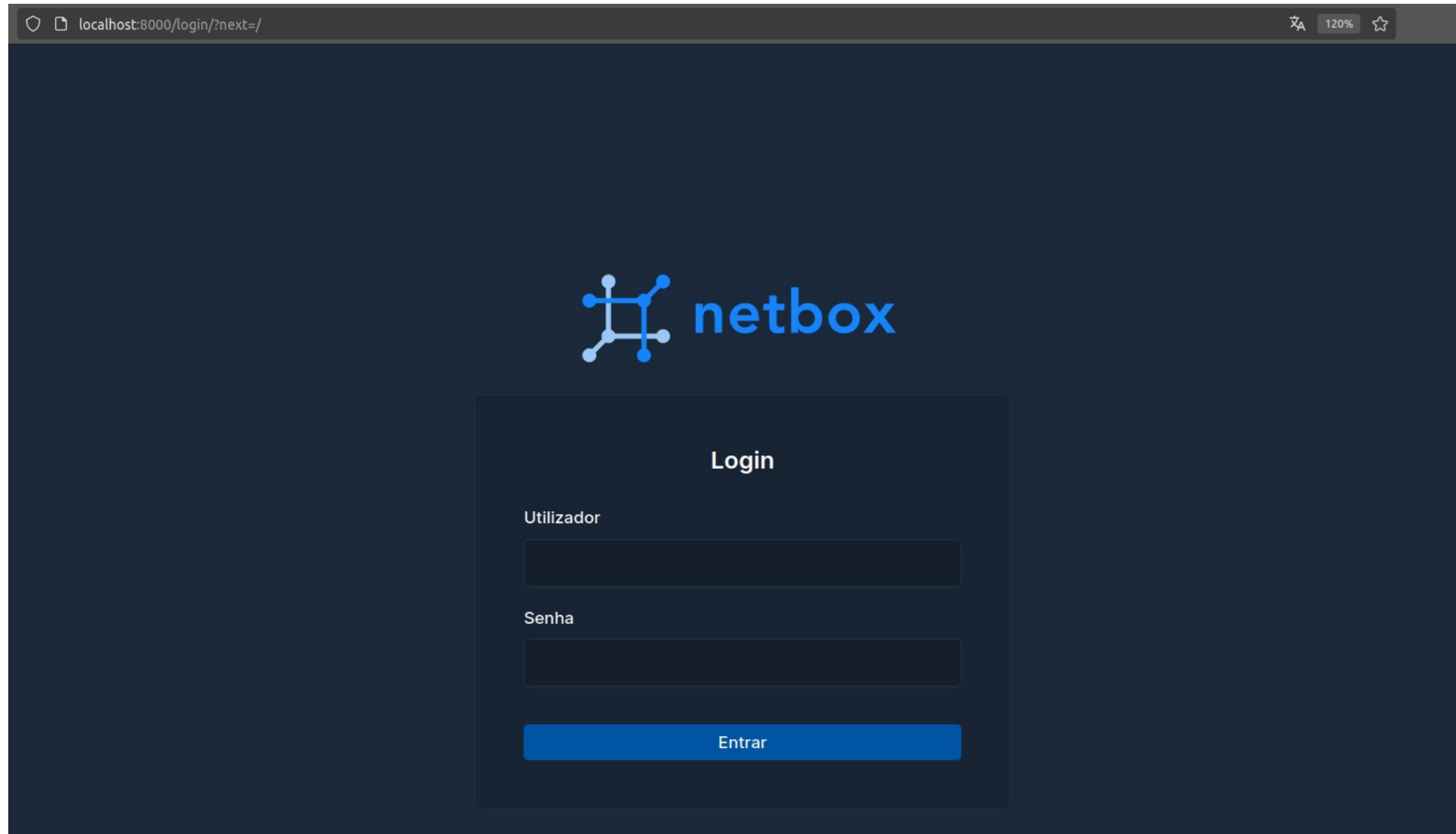
To create the first admin user run this command:

```
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



The image shows a browser window with the address bar displaying "localhost:8000/login/?next=/". The page features the NetBox logo, which consists of a blue network diagram icon and the text "netbox". Below the logo is a "Login" form with two input fields: "Utilizador" (User) and "Senha" (Password). A blue "Entrar" (Login) button is positioned at the bottom of the form.

localhost:8000/login/?next=/

netbox

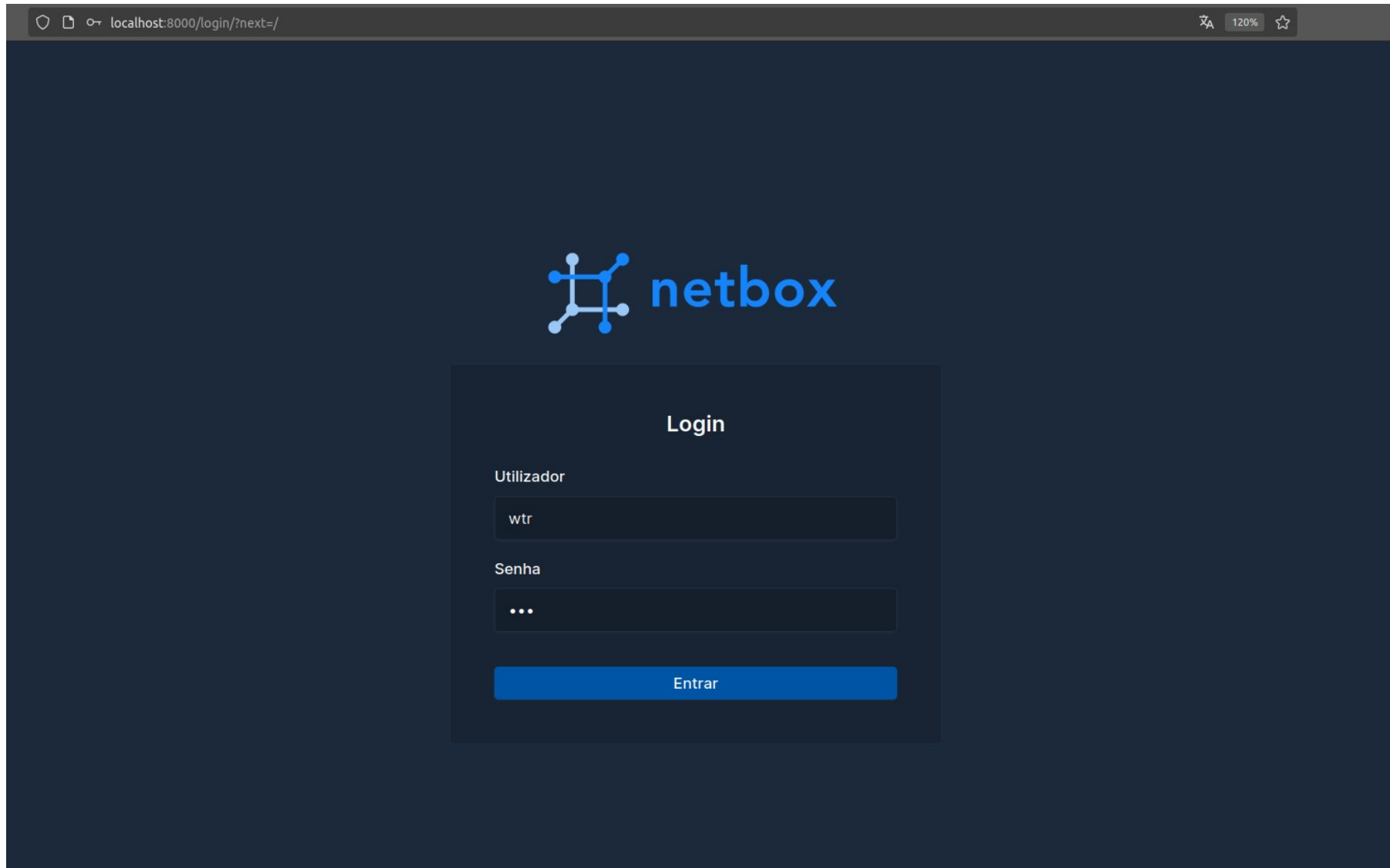
Login

Utilizador

Senha

Entrar

Acessando o NetBox



The screenshot shows a web browser window with the address bar displaying "localhost:8000/login/?next=/". The page features the NetBox logo, which consists of a blue network diagram icon and the text "netbox". Below the logo is a "Login" form with the following fields:

- Utilizador:** A text input field containing the value "wtr".
- Senha:** A password input field represented by three dots.
- Entrar:** A blue button to submit the login information.

Acessando o NetBox

The screenshot displays the NetBox web interface in a browser window at localhost:8000. The interface is dark-themed and features a sidebar on the left with navigation menus for various categories like Organization, Dispositivos, Conexões, etc. The main content area is a dashboard with several widgets:

- Notification:** "Nova Versão Disponível" (New Version Available) for NetBox v4.1.0.
- Bookmarks:** A widget indicating no favorites have been added.
- Organization:** A table showing counts for Sites (0), Inquilinos (0), and Contatos (0).
- IPAM:** A table showing counts for VRFs (0), Agregados (0), Prefixos (0), Faixas De IP (0), Endereços IP (0), and VLANs (0).
- DCIM:** A table showing counts for Sites (0), Racks (0), Tipos De Dispositivos (0), Dispositivos (0), and Cabos (0).
- Circuits:** A table showing counts for Provedores (0), Circuitos (0), Redes Dos Provedores (0), and Contas De Provedores (0).
- Welcome!:** A green widget with a welcome message and instructions on how to customize the dashboard.
- NetBox News:** A widget displaying news articles, including "NetBox 4.1 Is Now Generally Available" and "NetBox Operator for Kubernetes, Now Open Sourced".
- Virtualization:** A widget showing counts for Clusters (0) and Máquinas Virtuais (0).

Criando uma Região - Netbox

Adicionar região Ajuda

Criar

Pai

Nome*

Slug*

Abreviatura exclusiva da URL amigável

Descrição

Tags

Criando um Site - Netbox

Adicionar site

Ajuda

Criar

Site

Nome * pop-ba

Nome completo do site

Slug * pop-ba

Abreviatura exclusiva da URL amigável

Status * Ativo

Região Salvador

Grupo -----

Criando um Rack - Netbox

Adicionar rack Ajuda

[Criar](#)

Rack

Site* pop-ba

Localização -----

Nome* RACK IX.br

Status* Ativo

Função -----

Descrição

Tags -----

Adicionando um Fabricante - Netbox

Adicionar fabricante Ajuda

Criar

Fabricante

Nome*

Slug* ↺

Abreviatura exclusiva da URL amigável

Descrição

Tags

Cancelar Criar Criar e Adicionar Outro

Adicionando um Platform - Netbox

Adicionar plataforma Ajuda

Criar

Plataforma

Nome* iosxr

Slug* iosxr C
Abreviatura exclusiva da URL amigável

Fabricante cisco

Modelo de configuração -----

Descrição

Tags

Cancelar Criar Criar e Adicionar Outro

Adicionando um Device Type - Netbox

Adicionar tipo de dispositivo Ajuda

Criar

Tipo de Dispositivo

Fabricante * cisco

Modelo * ncs5501

Slug * ncs5501 ↻
Abreviatura exclusiva da URL amigável

Plataforma padrão iosxr ↕ 🔍

Descrição

Tags

Chassi

Altura (U) * 1.0 ↕

Adicionando Interfaces em um Device Type - Netbox

Adicionar modelo de interface

Criar

Tipo de dispositivo

Tipo de módulo

Nome*

Intervalos alfanuméricos são suportados para criação em massa. Casos e tipos mistos dentro de um único intervalo não são suportados (exemplo: [ge, xe] -0/0/ [0-9]).

Rótulo

Intervalos alfanuméricos são suportados. (Deve corresponder ao número de objetos a serem criados.)

Tipo*

Ativado

Somente gerenciamento

Device Type - Netbox

Tipos De Dispositivos / cisco dcim.devicetype:1 (ncs5501)

cisco ncs5501

Criado 2024-09-04 18:03 · Atualizado 2024-09-04 18:03

Tipo De Dispositivo **Interfaces 5** Journal Changelog

Busca rápida ⚙️ Configurar Tabela

<input type="checkbox"/>	NOME	RÓTULO	ENABLED	MANAGEMENT ONLY	TIPO	DESCRIÇÃO	INTERFACE BRIDGE	MODO DE OPERAÇÃO	TIPO DE POE	FUNÇÃO DO WIRELESS	
<input type="checkbox"/>	GigabitEthernet0/0/0/0	—	✓	—	SFP (1GE)	—	—	—	—	—	+ ✎ ▾
<input type="checkbox"/>	GigabitEthernet0/0/0/1	—	✓	—	SFP (1GE)	—	—	—	—	—	+ ✎ ▾
<input type="checkbox"/>	GigabitEthernet0/0/0/2	—	✓	—	SFP (1GE)	—	—	—	—	—	+ ✎ ▾
<input type="checkbox"/>	GigabitEthernet0/0/0/3	—	✓	—	SFP (1GE)	—	—	—	—	—	+ ✎ ▾
<input type="checkbox"/>	MgmtEth0/RP0/CPU0/0	—	✓	✓	1000BASE-TX (1GE)	—	—	—	—	—	+ ✎ ▾

Exibindo 1-5 de 5 Por Página ▾

Adicionando Role de Devices - Netbox

Adicionar função de dispositivo

Ajuda

Criar

Função do Dispositivo

Nome *

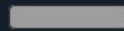
L2

Slug *

l2

Abreviatura exclusiva da URL amigável

Cor *

 Cinza

RGB color in hexadecimal. Example: 00ff00



Função da VM

Máquinas virtuais podem ser atribuídas a esta função

Modelo de configuração

Descrição

Tags

Adicionando um Device - Netbox

Adicionar dispositivo Ajuda

[Criar](#)

Dispositivo

Nome

Função do dispositivo*

Descrição

Tags

Hardware

Tipo de dispositivo*

Fluxo de ar

Número de série

Número de série do chassi, designado pelo fabricante

Etiqueta de patrimônio

Uma etiqueta exclusiva usada para identificar este dispositivo

Localização

Site*

Localização

Rack

Devices - Netbox

Dispositivos + Adicionar Importar Exportar

Resultados **2** Filtros

Busca rápida Configurar Tabela

<input type="checkbox"/>	NOME	STATUS	INQUILINO	SITE	LOCALIZAÇÃO	RACK	FUNÇÃO	FABRICANTE	TIPO	ENDEREÇO IP	
<input type="checkbox"/>	ncs5501-popba-95	Ativo	—	pop-ba	—	RACK IX.br	PE	cisco	ncs5501	—	
<input type="checkbox"/>	ncs5501-popmg-96	Ativo	—	pop-mg	—	RACK IX.br	PE	cisco	ncs5501	—	

Exibindo 1-2 de 2 Por Página

+ Adicionar Componentes Editar Selecionado Renomear Excluir Selecionado

Adicionando IP em Devices - Netbox

Endereço IP

Endereço*

Endereço IPv4 ou IPv6 (com máscara)

Status* ▾

O status operacional deste IP

Função ▾

O papel funcional deste IP

VRF ▾

Nome DNS

Hostname ou FQDN (não diferencia maiúsculas de minúsculas)

Descrição

Tags ▾

Adicionando IP em Devices - Netbox

Atribuição

Dispositivo Máquina Virtual Grupo FHRP

Interface

Torne este o IP primário do dispositivo/VM

IP NAT (Interno)

Endereço IP

Comentários **Escrita** Pré-visualização

© Markdown syntax is supported

Adicionando IP em Devices - Netbox

Dispositivos + Adicionar Importar Exportar

Resultados **2** Filtros

Busca rápida Configurar Tabela

<input type="checkbox"/>	NOME	STATUS	SITE	RACK	FUNÇÃO	FABRICANTE	TIPO	ENDEREÇO IP	
<input type="checkbox"/>	ncs5501-popba-95	Ativo	pop-ba	RACK IX.br	PE	cisco	ncs5501	192.168.246.95/24	✎
<input type="checkbox"/>	ncs5501-popmg-96	Ativo	pop-mg	RACK IX.br	PE	cisco	ncs5501	192.168.246.96/24	✎

Exibindo 1-2 de 2 Por Página

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

Adicionar token

Criar

Chave*



As chaves devem ter pelo menos 40 caracteres. **Certifique-se de salvar sua chave** antes de enviar este formulário, pois ela não será mais acessível depois que o token for criado.

Escrita habilitada

Permitir operações de criação/atualização/exclusão usando esta chave

Expira

Descrição

IPs Permitidos

Redes IPv4/IPv6 permitidas de onde o token pode ser usado. Deixe em branco para nenhuma restrição.
Exemplo: 10.1.1.0/24.192.168.10.16/32, 2001:db 8:1: :/64

Cancelar

Criar

Criar e Adicionar Outro

REST API - Netbox – exercicio1.py

```
netbox > exercicio1.py > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  import requests
4
5  netbox_url_api_devices = "http://localhost:8000/api/dcim/devices/"
6  token = "c37c796a6b2155e4caa519b3ebb077b6f261dda8"
7
8  headers = {
9      'Content-Type': 'application/json',
10     'Authorization': 'Token '+token
11 }
12
13 response_get_interfaces = requests.request("GET", netbox_url_api_devices, headers=headers, verify=False)
14
15
16 print(response_get_interfaces.headers)
17 print(response_get_interfaces.status_code)
18 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 2024 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/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": 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": 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 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}, {"id": 2, "url": "http://localhost:8000/api/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: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/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/api/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, "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": "2024-09-04T18:13:32.655646Z", "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 > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  import requests
4
5  netbox_url_api_devices = "http://localhost:8000/api/dcim/devices/"
6  token = "c37c796a6b2155e4caa519b3ebb077b6f261dda8"
7
8  headers = {
9      'Content-Type': 'application/json',
10     'Authorization': 'Token '+token
11 }
12
13 response_get_interfaces = requests.request("GET", netbox_url_api_devices, headers=headers, verify=False)
14
15
16 for device in response_get_interfaces.json()['results']:
17     #print(device)
18     print(f"device: {device['name']} - ip: {device['primary_ip']['address']}")
19
```

REST API - Netbox – exercicio3.py

```
netbox > exercicio3.py > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  import requests
4
5  netbox_url_api_devices = "http://localhost:8000/api/dcim/devices/"
6  token = "c37c796a6b2155e4caa519b3ebb077b6f261dda8"
7
8  headers = {
9      'Content-Type': 'application/json',
10     'Authorization': 'Token '+token
11 }
12
13 response_get_interfaces = requests.request("GET", netbox_url_api_devices, headers=headers, verify=False)
14
15
16 for device in response_get_interfaces.json()['results']:
17     ip_mgmt = device['primary_ip']['address']
18     ip_mgmt = ip_mgmt.split("/")
19     ip_mgmt = ip_mgmt[0]
20     print(f"device: {device['name']} - ip: {ip_mgmt}")
21
```


REST API - Netbox – exercicio4.py

```
netbox > exercicio4.py > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  import requests, os
4  from dotenv import load_dotenv
5  from netmiko import ConnectHandler
6  from rich import print
7
8  load_dotenv()
9
10 netbox_url_api_devices = "http://localhost:8000/api/dcim/devices/"
11 token = "c37c796a6b2155e4caa519b3ebb077b6f261dda8"
12
13 headers = {
14     'Content-Type': 'application/json',
15     'Authorization': 'Token '+token
16 }
17
18 response_get_interfaces = requests.request("GET", netbox_url_api_devices, headers=headers, verify=False)
19
20
21 for device in response_get_interfaces.json()['results']:
22     ip_mgmt = device['primary_ip']['address']
23     ip_mgmt = ip_mgmt.split("/")
24     ip_mgmt = ip_mgmt[0]
25
26     router = {"device_type": "cisco_xr", "host": ip_mgmt, "username": os.getenv("LAB_USERNAME"), "password": os.getenv("LAB_PASSWORD")}
27
28     try:
29         print(f"Acessando device: {device['name']}")
30         r1_connection = ConnectHandler(**router)
31
32         results = r1_connection.send_command('show ip int brief', use_textfsm=True)
33
34         for interface in results:
35             print(interface)
36
37         r1_connection.disconnect()
38         print(f"Finalizando o acesso no device: {device['name']}")
39     except Exception as err:
40         print(err)
41
```

REST API - Netbox – exercicio5.py

```
netbox > exercicio5.py > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  import requests, os
4  from dotenv import load_dotenv
5  from napalm import get_network_driver
6  from rich import print
7
8  load_dotenv()
9
10 netbox_url_api_devices = "http://localhost:8000/api/dcim/devices/"
11 token = "c37c796a6b2155e4caa519b3ebb077b6f261dda8"
12
13 headers = {
14     'Content-Type': 'application/json',
15     'Authorization': 'Token '+token
16 }
17
18 response_get_interfaces = requests.request("GET", netbox_url_api_devices, headers=headers, verify=False)
19
20
21 for device in response_get_interfaces.json()['results']:
22     ip_mgmt = device['primary_ip']['address']
23     ip_mgmt = ip_mgmt.split("/")
24     ip_mgmt = ip_mgmt[0]
25
26     router = {"hostname": ip_mgmt, "username": os.getenv("LAB_USERNAME"), "password": os.getenv("LAB_PASSWORD")}
27     driver = get_network_driver('iosxr')
28     try:
29         r_connection = driver(**router)
30         r_connection.open()
31         output = r_connection.get_facts()
32         print(f"device name: {output['hostname']}")
33         print(f"device netbox: {device['name']}")
34
35         if str(output['hostname']) != str(device['name']):
36             r_connection.load_merge_candidate(config='hostname '+str(device['name']))
37             compare = r_connection.compare_config()
38             print(compare)
39             r_connection.commit_config()
40             #r_connection.discard_config()
41             r_connection.close()
42     except Exception as err:
43         print(err)
44
```

Agenda

08

Nornir

09

Desafio

Nornir - o que é?

Inventory: hosts.yaml

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

```
nornir > ! hosts.yaml
1  R1:
2  |   hostname: '192.168.246.95'
3  |   port: 22
4  |   username: 'wtr'
5  |   password: 'wtr'
6  |   platform: 'ios'
7  R2:
8  |   hostname: '192.168.246.96'
9  |   port: 22
10 |   username: 'wtr'
11 |   password: 'wtr'
12 |   platform: 'ios'
13
```

```
1  from nornir import InitNornir
2  from nornir_rich.functions import print_result
3  from nornir_netmiko.tasks.netmiko_send_command import netmiko_send_command
4
5
6  nr = InitNornir(
7  |   runner={"plugin": "threaded", "options": {"num_workers": 20}},
8  |   config_file="hosts.yaml")
9
10
11 print_result(nr.run(netmiko_send_command, command_string="show interfaces brief"))
12
```



Function

Task

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

Nornir Plugins

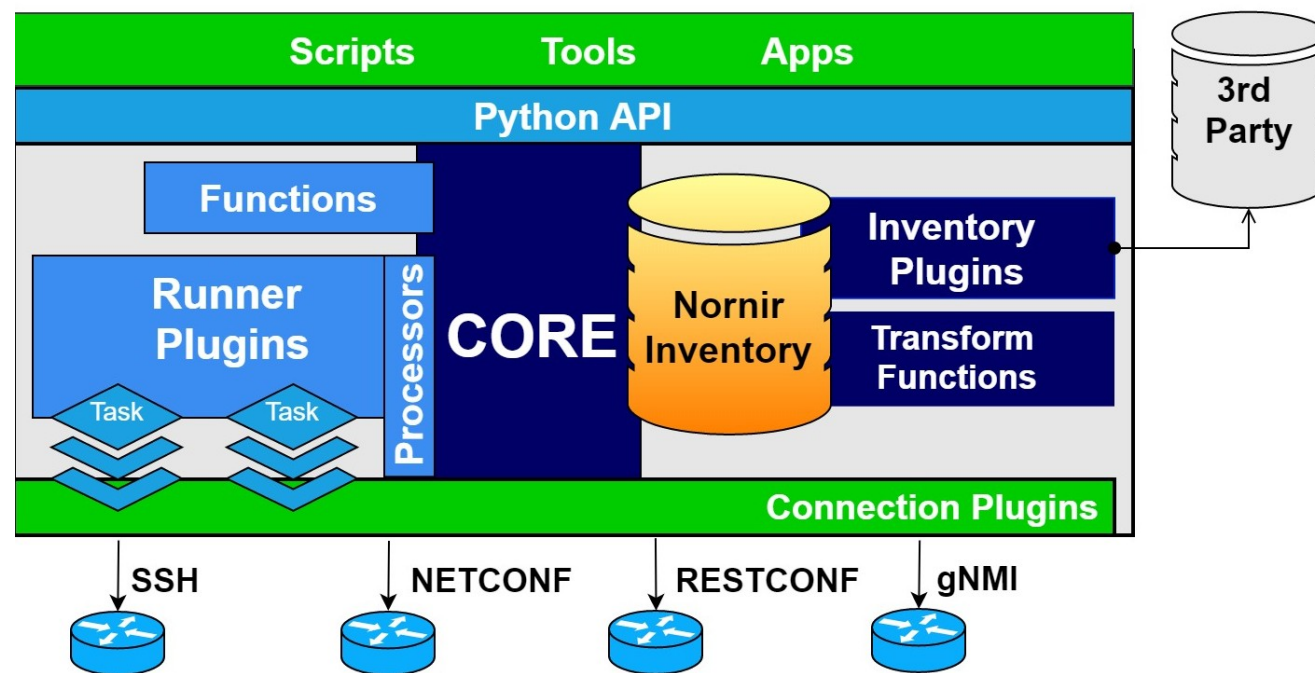
Plugin é um código utilizado para aumentar e melhorar as funcionalidades de um software;

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;

```
1 from nornir import InitNornir
2 from nornir_rich.functions import print_result
3 from nornir.core.task import Task, Result
4
5 def hello_world(task: Task) -> Result:
6     return Result(host=task.host, result=f"{task.host.name} diga olá mundo!")
7
8 nr = InitNornir(
9     runner={"plugin": "threaded", "options": {"num_workers": 20}},
10    config_file="hosts.yaml")
11
12 result = nr.run(task=hello_world)
13
14 print_result(result)
15
```

```
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;

```
1 from nornir import InitNornir
2 from nornir_rich.functions import print_result
3 from nornir.core.task import Task, Result
4
5 def hello_world(task: Task) -> Result:
6     return Result(host=task.host, result=f"{task.host.name} diga olá mundo!")
7
8 nr = InitNornir(
9     runner={"plugin": "threaded", "options": {"num_workers": 20}},
10    config_file="hosts.yaml")
11
12 result = nr.run(task=hello_world)
13
14 print_result(result)
15
```

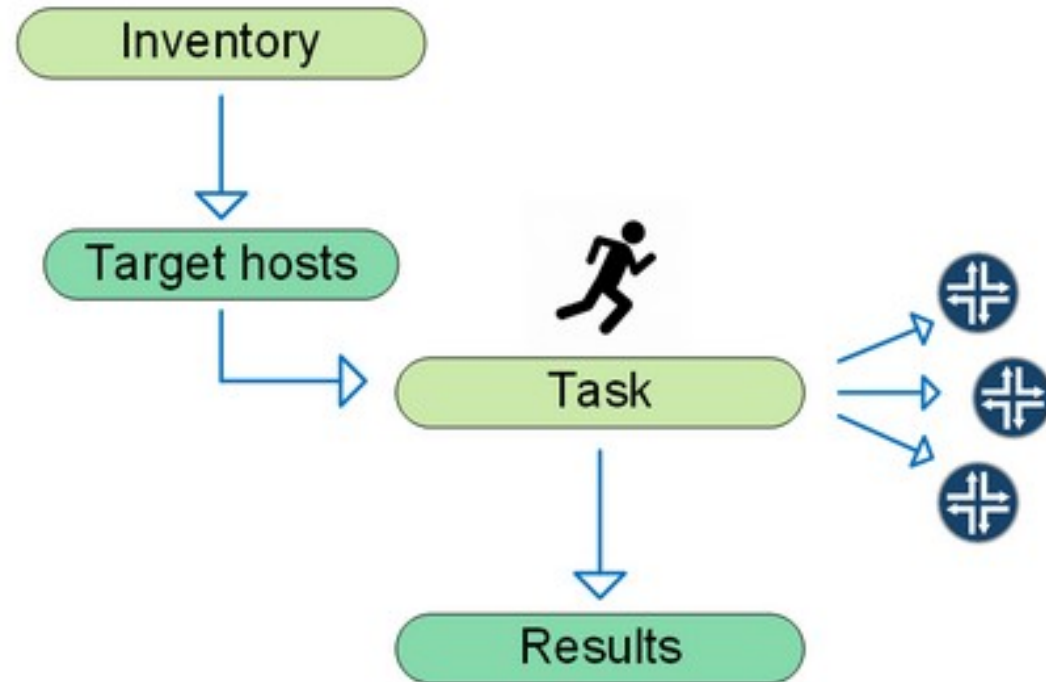
```
hello_world
R1 diga olá mundo!
```

```
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 – exemplo1.py

```
1 from nornir import InitNornir
2 from nornir_rich.functions import print_result
3 from nornir_netmiko.tasks.netmiko_send_command import netmiko_send_command
4
5
6 nr = InitNornir(
7     runner={"plugin": "threaded", "options": {"num_workers": 20}},
8     config_file="hosts.yaml")
9
10
11 print_result(nr.run(netmiko_send_command, read_timeout=120, command_string="show interfaces brief"))
12
```

netmiko_send_command

Wed Sep 4 20:57:00.230 UTC

Intf Name	Intf State	LineP State	Encap Type	MTU (byte)	BW (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 Name	Intf State	LineP State	Encap Type	MTU (byte)	BW (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

```
1 from nornir import InitNornir
2 from nornir rich.functions import print_result
3 from nornir netmiko.tasks.netmiko_send_command import netmiko_send_command
4
5
6 nr = InitNornir(
7     runner={"plugin": "threaded", "options": {"num_workers": 20}},
8     config_file="hosts.yaml")
9
10 r1 = nr.filter(name="R1")
11
12 print_result(r1.run(netmiko_send_command, read_timeout=120, command_string="show interfaces brief"))
13
```

```
netmiko_send_command
Wed Sep  4 20:57:00.230 UTC
```

Intf Name	Intf State	LineP State	Encap Type	MTU (byte)	BW (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

```
1  from nornir import InitNornir
2  from nornir.napalm.plugins.tasks import napalm_get
3  from nornir_rich.functions import print_result
4  import os
5
6  nr = InitNornir(
7      runner={"plugin": "threaded", "options": {"num_workers": 20}},
8      inventory={
9          "plugin": "NetBoxInventory2",
10         "options": {
11             "nb_url": "http://localhost:8000/",
12             "nb_token": "c37c796a6b2155e4caa519b3ebb077b6f261dda8",
13             "ssl_verify": False}
14         })
15
16  nr.inventory.defaults.username = "wtr"
17  nr.inventory.defaults.password = "wtr"
18  nr.inventory.defaults.port = "22"
19
20  print_result(nr.run(task=napalm_get, getters=["get_interfaces"]), vars=["result"])
21
```

Nornir + Netbox + Inventory – exemplo4.py

```
1  from nornir import InitNornir
2
3  nr = InitNornir(
4      runner={"plugin": "threaded", "options": {"num_workers": 20}},
5      inventory={
6          "plugin": "NetBoxInventory2",
7          "options": {
8              "nb_url": "http://localhost:8000/",
9              "nb_token": "c37c796a6b2155e4caa519b3ebb077b6f261dda8",
10             "ssl_verify": False}
11     })
12
13  nr.inventory.defaults.username = "wtr"
14  nr.inventory.defaults.password = "wtr"
15  nr.inventory.defaults.port = "22"
16
17  print(nr.inventory.hosts)
18
19  print(nr.inventory.hosts['ncs5501-popba-95'].data)
20
```


Nornir + Netbox + filters – exemplo5.py

```
1  from nornir import InitNornir
2
3  nr = InitNornir(
4      runner={"plugin": "threaded", "options": {"num_workers": 20}},
5      inventory={
6          "plugin": "NetBoxInventory2",
7          "options": {
8              "nb_url": "http://localhost:8000/",
9              "nb_token": "c37c796a6b2155e4caa519b3ebb077b6f261dda8",
10             "filter_parameters": {"region": ["salvador"], "status": "active", "platform": ["iosxr"]},
11             "ssl_verify": False}
12         })
13
14  nr.inventory.defaults.username = "wtr"
15  nr.inventory.defaults.password = "wtr"
16  nr.inventory.defaults.port = "22"
17
18  print(nr.inventory.hosts)
```

Nornir + Netbox + Tasks – exemplo6.py

```
1  from nornir import InitNornir
2  from nornir_utils.plugins.functions import print_result
3  from nornir_netmiko import netmiko_send_config, netmiko_commit
4
5  def nornir_task_wtr(task):
6      configurations = ["hostname "+str(task.host.name)]
7      command = task.run(netmiko_send_config, config_commands=configurations)
8      print_result(command)
9      commit = task.run(netmiko_commit)
10     print_result(commit)
11
12
13     nr = InitNornir(
14         runner={"plugin": "threaded", "options": {"num_workers": 20}},
15         inventory={
16             "plugin": "NetBoxInventory2",
17             "options": {
18                 "nb_url": "http://localhost:8000/",
19                 "nb_token": "c37c796a6b2155e4caa519b3ebb077b6f261dda8",
20                 "filter_parameters": {"region": ["salvador"], "status": "active", "platform": ["iosxr"]},
21                 "ssl_verify": False}
22         })
23
24     nr.inventory.defaults.username = "wtr"
25     nr.inventory.defaults.password = "wtr"
26     nr.inventory.defaults.port = "22"
27
28     nr.run(task=nornir_task_wtr)
29
```

Nornir + Netbox + Inspect – exemplo7.py

```
nornir > exemplo7.py > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  from nornir import InitNornir
4  from nornir_utils.plugins.functions import print_result
5  from nornir_netmiko import netmiko_send_config, netmiko_commit
6  from nornir_inspect import nornir_inspect
7
8
9  def nornir_task_wtr(task):
10     configurations = ["hostname "+str(task.host.name)]
11     command = task.run(netmiko_send_config, config_commands=configurations)
12     print_result(command)
13     commit = task.run(netmiko_commit)
14     print_result(commit)
15     print(commit['ncs5501-popba-95'])
16
17
18
19  nr = InitNornir(
20     runner={"plugin": "threaded", "options": {"num_workers": 20}},
21     inventory={
22         "plugin": "NetBoxInventory2",
23         "options": {
24             "nb_url": "http://localhost:8000/",
25             "nb_token": "c37c796a6b2155e4caa519b3ebb077b6f261dda8",
26             "filter_parameters": {"region": ["salvador"], "status": "active", "platform": ["iosxr"]},
27             "ssl_verify": False}
28         })
29
30  nr.inventory.defaults.username = "wtr"
31  nr.inventory.defaults.password = "wtr"
32  nr.inventory.defaults.port = "22"
33
34  results = nr.run(task=nornir_task_wtr)
35  nornir_inspect(results)
36
37  print(results['ncs5501-popba-95'][1].result)
38
```

Nornir + Netbox + Netconf – exemplo8.py

```
1  from nornir import InitNornir
2  from nornir netconf.plugins.tasks import netconf_get_config
3  import json,xmltodict
4
5  def nornir_task_wtr(task):
6      data_interfaces = task.run(netconf_get_config,source="running",filter_type="subtree")
7      print(data_interfaces.result.rpc.data_xml)
8      obj = xmltodict.parse(data_interfaces.result.rpc.data_xml)
9      json_interfaces = json.loads(json.dumps(obj['data']))
10     print(json_interfaces)
11
12     nr = InitNornir(
13         runner={"plugin": "threaded", "options": {"num_workers": 20}},
14         inventory={
15             "plugin": "NetBoxInventory2",
16             "options": {
17                 "nb_url": "http://localhost:8000/",
18                 "nb_token": "c37c796a6b2155e4caa519b3ebb077b6f261dda8",
19                 "filter_parameters": {"region": ["salvador"],"status": "active", "platform": ["iosxr"]},
20                 "ssl_verify": False}
21         })
22
23     nr.inventory.defaults.username = "wtr"
24     nr.inventory.defaults.password = "wtr"
25     nr.inventory.defaults.port = "830"
26
27     nr.run(task=nornir_task_wtr)
28
```


Nornir + Netbox + Netconf – exemplo9.py

```
1  from nornir import InitNornir
2  from nornir_netconf.plugins.tasks import netconf_get_config
3  import json,xmltodict
4
5  def nornir_task_wtr(task):
6      data_interfaces = task.run(
7          netconf_get_config,
8          source="running",
9          path=""
10         <interface-configurations xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ifmgr-cfg">
11         </interface-configurations>""",
12         filter_type="subtree",
13     )
14     obj = xmltodict.parse(data_interfaces.result.rpc.data_xml)
15     json_interfaces = json.loads(json.dumps(obj['data']))
16
17     #print(json_interfaces['interface-configurations']['interface-configuration'])
18
19     for interface in json_interfaces['interface-configurations']['interface-configuration']:
20         print(interface)
21
22 nr = InitNornir(
23     runner={"plugin": "threaded", "options": {"num_workers": 20}},
24     inventory={
25         "plugin": "NetBoxInventory2",
26         "options": {
27             "nb_url": "http://localhost:8000/",
28             "nb_token": "c37c796a6b2155e4caa519b3ebb077b6f261dda8",
29             "filter_parameters": {"region": ["salvador"],"status": "active", "platform": ["iosxr"]},
30             "ssl_verify": False}
31     })
32
33 nr.inventory.defaults.username = "wtr"
34 nr.inventory.defaults.password = "wtr"
35 nr.inventory.defaults.port = "830"
36
37 nr.run(task=nornir_task_wtr)
38
```

Netbox - API - GraphQL

<http://localhost:8000/graphql/>

```
1 {
2   device_list(filters: {id: {contains: "1"}}) {
3     name
4     primary_ip4 {
5       address
6     }
7     platform {
8       name
9     }
10    status
11    interfaces {
12      name
13      description
14      mtu
15      enabled
16      ip_addresses {
17        address
18      }
19    }
20  }
21 }
```



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 > ...
1  #!/opt/wtr-pop-ba-2024/venv/bin/python3.10
2
3  import requests
4
5  url_graphql = "http://localhost:8000/graphql/"
6  token = "c37c796a6b2155e4caa519b3ebb077b6f261dda8"
7
8  headers = {
9      'Content-Type': 'application/json',
10     'Authorization': 'Token '+token
11 }
12
13 GRAPHQL_QUERY = {"query": f"""
14 {{
15     device_list(
16         filters:
17             {{
18                 id:
19                     {{
20                         contains: "1"
21                     }}
22             }}
23     )
24 }}
25     name
26     primary_ip4
27     {{
28         address
29     }}
30     platform
31     {{
32         name
33     }}
34     status
35     interfaces
36     {{
37         name
38         description
39         mtu
40         enabled
41         ip_addresses
42         {{
43             address
44         }}
45     }}
46 }}
47 }}"""
48
49 device_interfaces_netbox = requests.post(url=url_graphql, json=GRAPHQL_QUERY, headers=headers, verify=False)
50
51 print(device_interfaces_netbox.json())
```

Netbox - API – GraphQL – exemplo2.py

```
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
        }}
      }}
    }}
    }}"""}

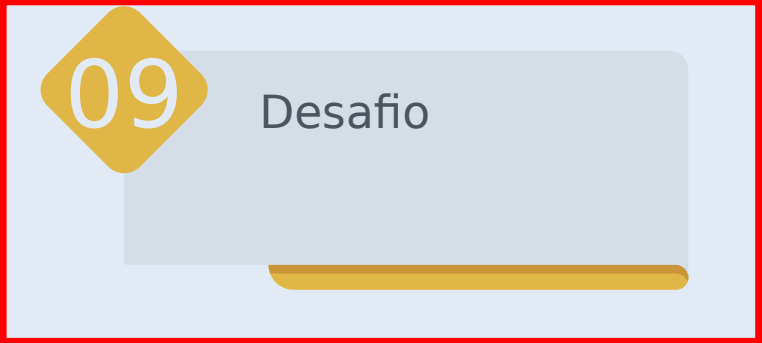
    device_interfaces_netbox = requests.post(url=url_graphql, json=GRAPHQL_QUERY, headers=headers, verify=False).json()['data']['device_list'][0]['interfaces']
    print(device_interfaces_netbox)
```

Netbox - API – Graphql – exemplo2.py

```
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', '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', '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



09 Desafio

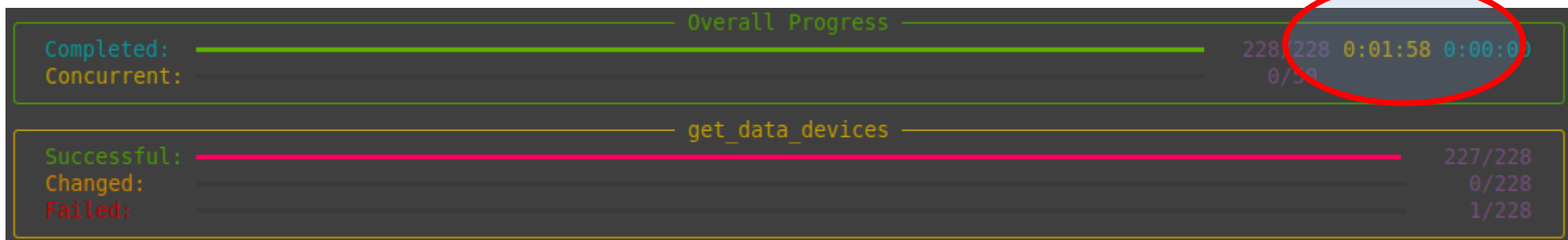
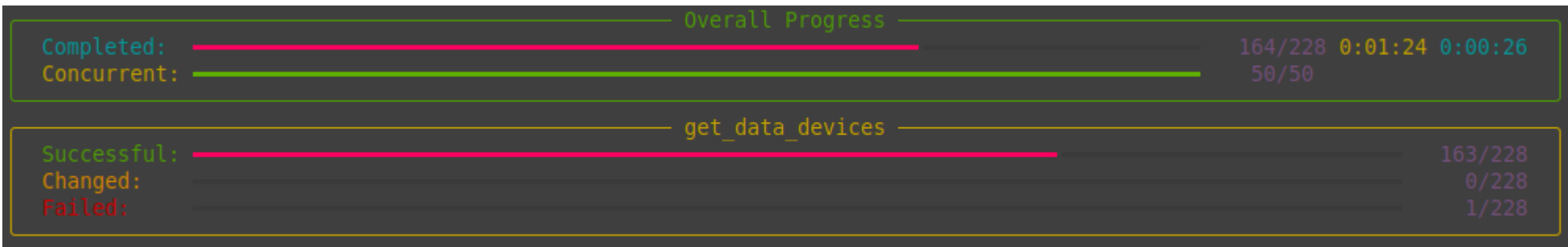
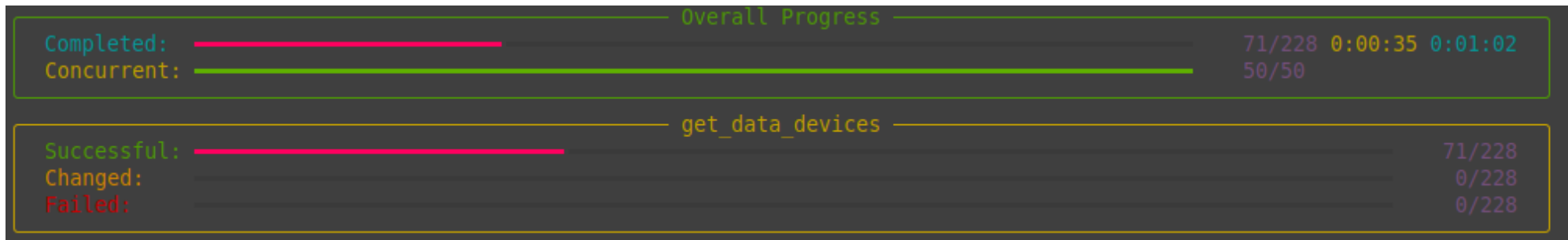
A light blue rectangular box with a red border. Inside the box, on the left, is a yellow diamond shape containing the white number '09'. To the right of the diamond is a light blue rounded rectangle containing the word 'Desafio' in a dark grey font. Below the rounded rectangle is a horizontal yellow bar.

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 **egi.br**

www.nic.br | www.cgi.br