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

LIVE INTRA REDE
Ferramentas de automação de redes

William Prado
IX.br Engineering

nic.br


NETMIKO

Netmiko - o que é?

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

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

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

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

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

Features:

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

https://github.com/ktbyers/netmiko

Netmiko - Instalação

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

Recomendado usar ambiente virtual do Python:

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

```
(venv) root@william:/opt/intrarede# pip3 install netmiko
Collecting netmiko
  Using cached netmiko-4.3.0-py3-none-any.whl (219 kB)
Collecting pyyaml>=5.3
  Using cached PyYAML-6.0.1-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (705 kB)
Collecting paramiko>=2.9.5
  Using cached paramiko-3.4.0-py3-none-any.whl (225 kB)
Collecting pyserial>=3.3
  Using cached pyserial-3.5-py2.py3-none-any.whl (90 kB)
Collecting ntc-templates>=2.0.0
  Using cached ntc_templates-4.3.0-py3-none-any.whl (428 kB)
Collecting scp>=0.13.6
  Using cached scp-0.14.5-py2.py3-none-any.whl (8.7 kB)
Collecting textfsm>=1.1.3
  Using cached textfsm-1.1.3-py2.py3-none-any.whl (44 kB)
Collecting bcrypt>=3.2
  Using cached bcrypt-4.1.2-cp39-abi3-manylinux_2_28_x86_64.whl (698 kB)
Collecting pynacl>=1.5
  Using cached PyNaCl-1.5.0-cp36-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.manylinux_2_24_x86_64.whl (856 kB)
Collecting cryptography>=3.3
  Using cached cryptography-42.0.5-cp39-abi3-manylinux_2_28_x86_64.whl (4.6 MB)
Collecting six
  Using cached six-1.16.0-py2.py3-none-any.whl (11 kB)
Collecting future
  Using cached future-1.0.0-py3-none-any.whl (491 kB)
Collecting cffi>=1.12
  Using cached cffi-1.16.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (443 kB)
Collecting pycparser
  Using cached pycparser-2.21-py2.py3-none-any.whl (118 kB)
Installing collected packages: pyserial, six, pyyaml, pycparser, future, bcrypt, textfsm, cffi, pynacl, ntc-templates, cryptography, paramiko, scp, netmiko
Successfully installed bcrypt-4.1.2 cffi-1.16.0 cryptography-42.0.5 future-1.0.0 netmiko-4.3.0 ntc-templates-4.3.0 paramiko-3.4.0 pycparser-2.21 pynacl-1.5.0 pyserial-3.5 pyyaml-6.0.1 scp-0.14.5 six-1.16.0 textfs
m-1.1.3
(venv) root@william:/opt/intrarede#
```

Netmiko – Criando o Script 1

Recomendado colocar as credenciais de acesso em um arquivo .env:

.env

```
LAB_USERNAME="william"
LAB_PASSWORD="intrarede"
```

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 – Criando o Script 1

```python
#!/opt/intrarede/venv/bin/python3.10

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

#carregar as variáveis definidas no .env
load_dotenv()

#dict com os dados do device para acesso
PE3 = {
    "device_type": "cisco_xr",
    "host": "192.168.246.96",
    "username": os.getenv("LAB_USERNAME"),
    "password": os.getenv("LAB_PASSWORD"),
}

try:
    #conexão com o PE1
    pe3_connection = ConnectHandler(**PE3)

    #executar o comando show version no PE1 e mostrar o resultado
    print(pe3_connection.send_command('show version'))

    #finalizar a conexão com o PE1
    pe3_connection.disconnect()

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

Netmiko – Resultado do Script 1

```
(venv) root@william:/opt/intrarede# python3.10 netmiko1.py

Thu Mar 14 17:08:05.595 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 34 minutes

(venv) root@william:/opt/intrarede#
```

Netmiko – Criando o Script 2

```python
#!/opt/intrarede/venv/bin/python3.10

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

load_dotenv()

PE3 = {
    "device_type": "cisco_xr",
    "host": "192.168.246.96",
    "username": os.getenv("LAB_USERNAME"),
    "password": os.getenv("LAB_PASSWORD"),
}

try:
    pe3_connection = ConnectHandler(**PE3)

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

    print(pe3_connection.send_config_set(commands))

    print(pe3_connection.commit())

    pe3_connection.disconnect()

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

Netmiko – Resultado do Script 2

```
(venv) root@william:/opt/intrarede# python3.10 netmiko2.py
configure terminal

Thu Mar 14 17:05:07.291 UTC
RP/0/RP0/CPU0:PE3(config)#interface gigabitEthernet 0/0/0/1

RP/0/RP0/CPU0:PE3(config-if)#ipv4 address 192.168.100.1 255.255.255.252

RP/0/RP0/CPU0:PE3(config-if)#no shutdown

RP/0/RP0/CPU0:PE3(config-if)#
commit

Thu Mar 14 17:05:08.676 UTC
RP/0/RP0/CPU0:PE3(config-if)#
(venv) root@william:/opt/intrarede#
```

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)

```python
#!/opt/intrarede/venv/bin/python3.10

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

load_dotenv()

PE3 = {
    "device_type": "cisco_xr",
    "host": "192.168.246.96",
    "username": os.getenv("LAB_USERNAME"),
    "password": os.getenv("LAB_PASSWORD")
}

try:
    pe3_connection = ConnectHandler(**PE3)

    print(pe3_connection.send_command('show running-config interface'))

    pe3_connection.disconnect()

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

```
Thu Mar 14 17:42:59.828 UTC
interface MgmtEth0/RP0/CPU0/0
 description GERENCIA_OOB
 ipv4 address 192.168.246.96 255.255.255.0
!
interface GigabitEthernet0/0/0/0
 description UPLINK_CUSTOMER_A
 shutdown
!
interface GigabitEthernet0/0/0/1
 description UPLINK_CUSTOMER_B
 ipv4 address 192.168.100.1 255.255.255.252
!
interface GigabitEthernet0/0/0/2
 shutdown
!
interface GigabitEthernet0/0/0/3
 shutdown
!
```

Netmiko – TTP (Template Text Parser)

```python
PE3 = {
    "device_type": "cisco_xr",
    "host": "192.168.246.96",
    "username": os.getenv("LAB_USERNAME"),
    "password": os.getenv("LAB_PASSWORD")
}

try:
    pe3_connection = ConnectHandler(**PE3)

    results = pe3_connection.send_command('show running-config interface')

    ttp_template = """
    interface {{ interface }}
     ipv4 address {{ ip }} {{ mask }}
     description {{ description }}
    """

    parser = ttp(data=results, template=ttp_template)
    parser.parse()

    print(parser.result(format='json')[0])

    pe3_connection.disconnect()

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

```python
#!/opt/intrarede/venv/bin/python3.10

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

load_dotenv()
```

Netmiko – TTP (Template Text Parser)

```
PE3 = {
    "device_type": "cisco_xr",
    "host": "192.168.246.96",
    "username": os.getenv("LAB_USERNAME"),
    "password": os.getenv("LAB_PASSWORD")
}

try:
    pe3_connection = ConnectHandler(**PE3)

    results = pe3_connection.send_command('show running-config interface')

    ttp_template = """
    interface {{ interface }}
     ipv4 address {{ ip }} {{ mask }}
     description {{ description }}
    """

    parser = ttp(data=results, template=ttp_template)
    parser.parse()

    print(parser.result(format='json')[0])

    pe3_connection.disconnect()

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

Netmiko – TTP (Template Text Parser)

```
PE3 = {
    "device_type": "cisco_xr",
    "host": "192.168.246.96",
    "username": os.getenv("LAB_USERNAME"),
    "password": os.getenv("LAB_PASSWORD")
}

try:
    pe3_connection = ConnectHandler(**PE3)

    results = pe3_connection.send_command('show running-config interface')

    ttp_template = """
    interface {{ interface }}
     ipv4 address {{ ip }} {{ mask }}
     description {{ description }}
    """

    parser = ttp(data=results, template=ttp_template)
    parser.parse()

    print(parser.result(format='json')[0])

    pe3_connection.disconnect()

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

Netmiko – TTP (Template Text Parser)

```
Thu Mar 14 17:42:59.828 UTC
interface MgmtEth0/RP0/CPU0/0
 description GERENCIA_OOB
 ipv4 address 192.168.246.96 255.255.255.0
!
interface GigabitEthernet0/0/0/0
 description UPLINK_CUSTOMER_A
 shutdown
!
interface GigabitEthernet0/0/0/1
 description UPLINK_CUSTOMER_B
 ipv4 address 192.168.100.1 255.255.255.252
!
interface GigabitEthernet0/0/0/2
 shutdown
!
interface GigabitEthernet0/0/0/3
 shutdown
!
```

```
[
    [
        {
            "description": "GERENCIA_OOB",
            "interface": "MgmtEth0/RP0/CPU0/0",
            "ip": "192.168.246.96",
            "mask": "255.255.255.0"
        },
        {
            "description": "UPLINK_CUSTOMER_A",
            "interface": "GigabitEthernet0/0/0/0"
        },
        {
            "description": "UPLINK_CUSTOMER_B",
            "interface": "GigabitEthernet0/0/0/1",
            "ip": "192.168.100.1",
            "mask": "255.255.255.252"
        },
        {
            "interface": "GigabitEthernet0/0/0/2"
        },
        {
            "interface": "GigabitEthernet0/0/0/3"
        }
    ]
]
```

Netmiko – TextFSM

```python
#!/opt/intrarede/venv/bin/python3.10

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

load_dotenv()

PE3 = {
    "device_type": "cisco_xr",
    "host": "192.168.246.96",
    "username": os.getenv("LAB_USERNAME"),
    "password": os.getenv("LAB_PASSWORD"),
}

try:
    pe3_connection = ConnectHandler(**PE3)

    print(results = pe3_connection.send_command('show ip int brief', use_textfsm=True))

    pe3_connection.disconnect()

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

Netmiko – TextFSM

```python
#!/opt/intrarede/venv/bin/python3.10

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

load_dotenv()

PE3 = {
    "device_type": "cisco_xr",
    "host": "192.168.246.96",
    "username": os.getenv("LAB_USERNAME"),
    "password": os.getenv("LAB_PASSWORD"),
}

try:
    pe3_connection = ConnectHandler(**PE3)

    print(results = pe3_connection.send_command('show ip int brief', use_textfsm=True))

    pe3_connection.disconnect()

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

Netmiko – TextFSM

```python
#!/opt/intrarede/venv/bin/python3.10

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

load_dotenv()

PE3 = {
    "device_type": "cisco_xr",
    "host": "192.168.246.96",
    "username": os.getenv("LAB_USERNAME"),
    "password": os.getenv("LAB_PASSWORD"),
}

try:
    pe3_connection = ConnectHandler(**PE3)

    print(results = pe3_connection.send_command('show ip int brief', use_textfsm=True))

    pe3_connection.disconnect()

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

```
[
    {
        'interface': 'MgmtEth0/RP0/CPU0/0',
        'ip_address': '192.168.246.96',
        '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': 'Down',
        'proto': 'Down',
        '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'
    }
]
```

Nornir - o que é?

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

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


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


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

```

Function

Task

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

Inventory: hosts.yaml

```yaml
R1:
  hostname: '192.168.246.94'
  port: 22
  username: 'ixforum'
  password: 'ixforum'
  platform: 'ios'
R2:
  hostname: '192.168.246.95'
  port: 22
  username: 'ixforum'
  password: 'ixforum'
  platform: 'ios'
R3:
  hostname: '192.168.246.96'
  port: 22
  username: 'ixforum'
  password: 'ixforum'
  platform: 'ios'
R4:
  hostname: '192.168.246.102'
  port: 22
  username: 'ixforum'
  password: 'ixforum'
  platform: 'ios'
R5:
  hostname: '192.168.246.103'
  port: 22
  username: 'ixforum'
  password: 'ixforum'
  platform: 'ios'

```

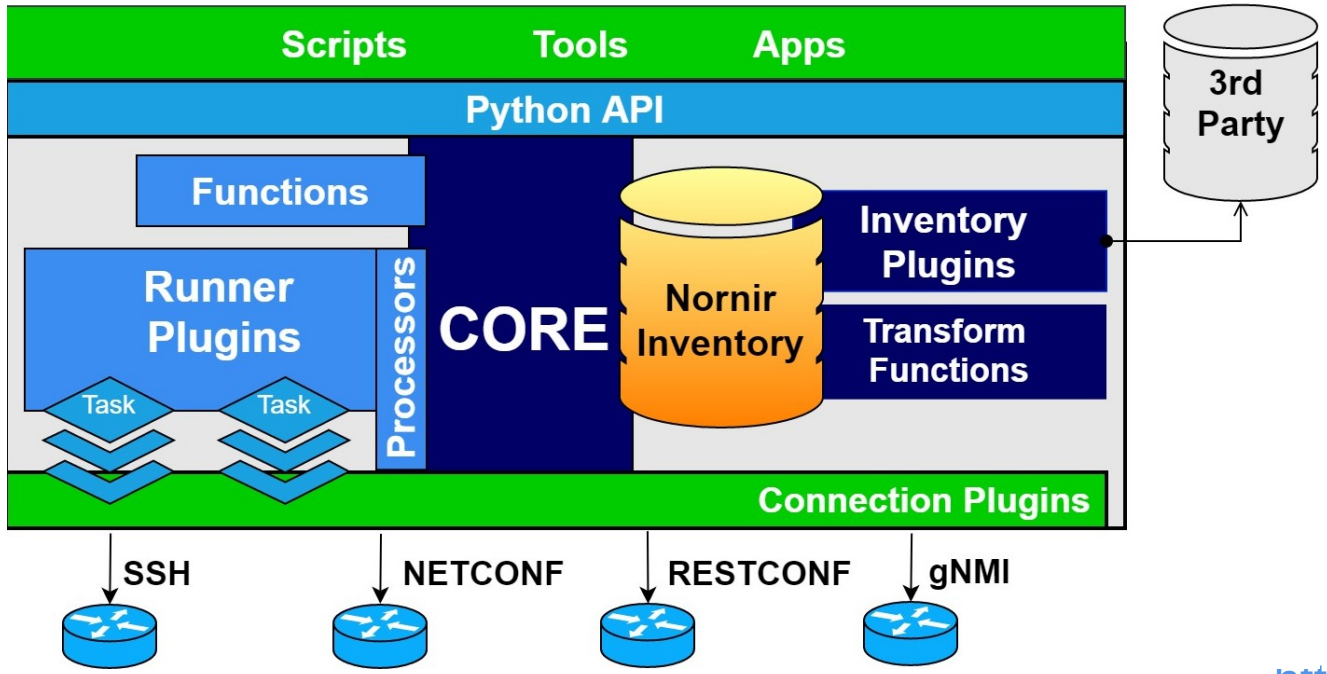
Nornir Plugins

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

Nornir permite adicionar funcionalidades através de plugins;

Plugins podem ser instalados usando pip:
pip install nornir_netmiko

Nornir Plugins Architecture



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

Nornir Tasks

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

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

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

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

result = nr.run(task=hello_world)

print_result(result)

```

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

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

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

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

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

Nornir Tasks

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

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

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

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

result = nr.run(task=hello_world)

print_result(result)
```

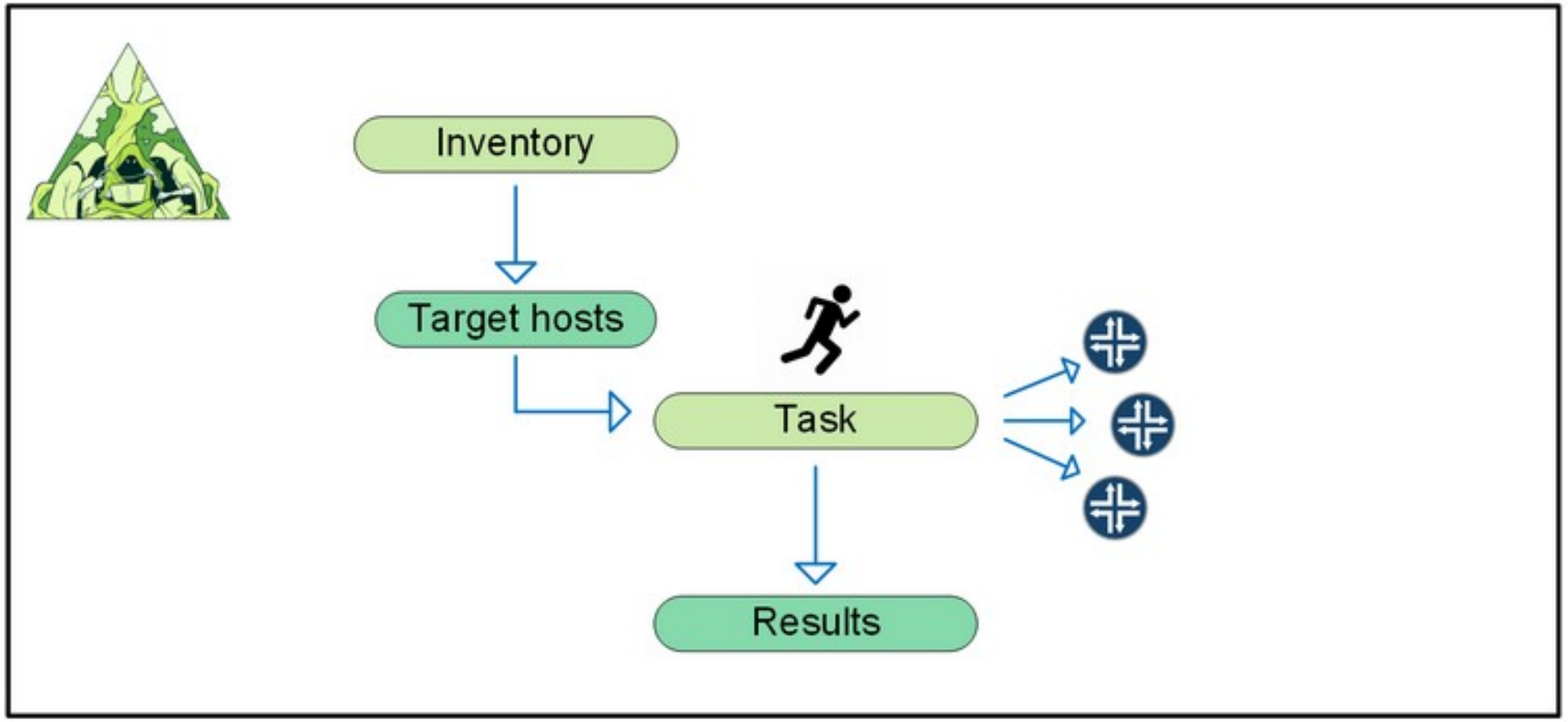
```
hello_world
R1 diga olá mundo!
```

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

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

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

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


Inventory
Target hosts
Task
Results

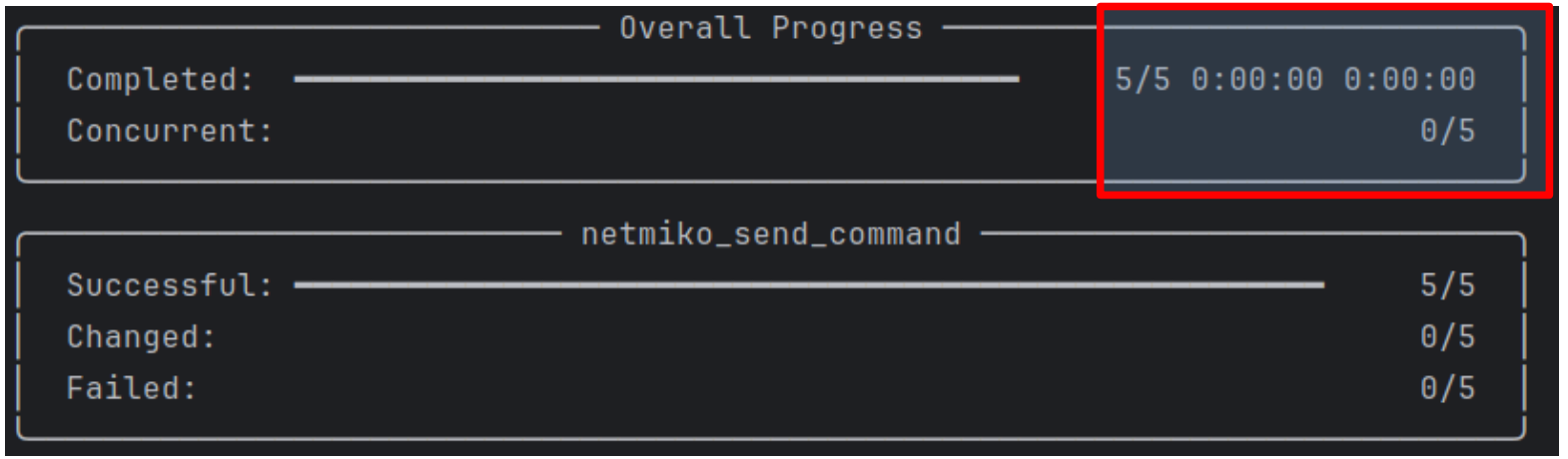
Nornir - Task 1: Netmiko

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


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


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

```
Overall Progress
Completed:   ━━━━━━━━━━━━━━━━━━━━━━━━━   5/5 0:00:00 0:00:00
Concurrent:                                               0/5

netmiko_send_command
Successful:  ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━    5/5
Changed:                                                  0/5
Failed:                                                   0/5
```

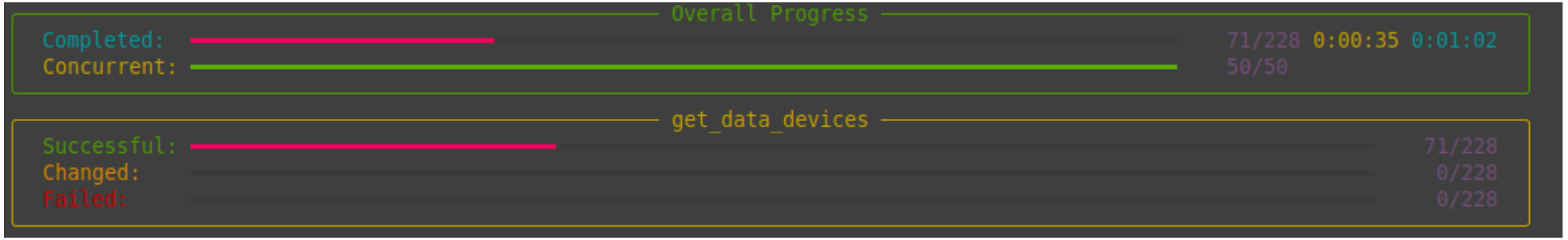
Nornir - Task 1: Netmiko

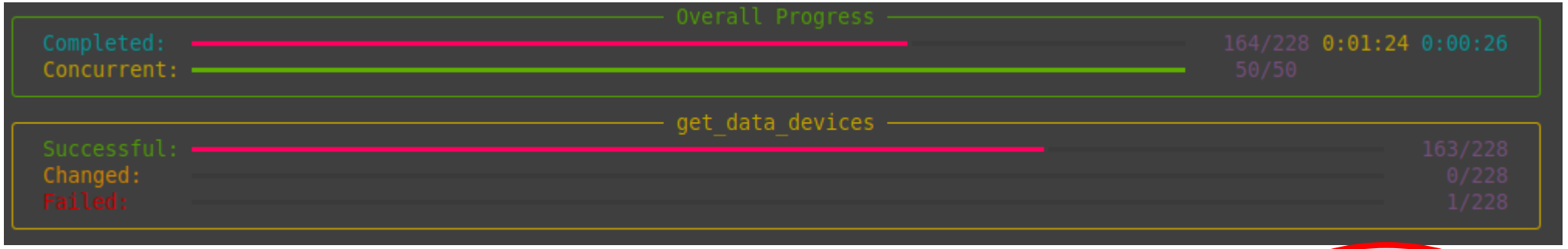
```python
from nornir import InitNornir
from nornir_rich.functions import print_result
from nornir_netmiko.tasks.netmiko_send_command import netmiko_send_command


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


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

```

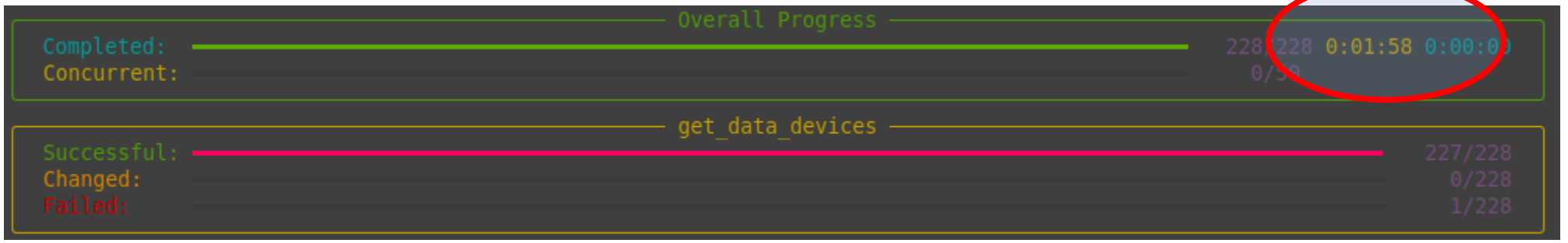


Aproximadamente 1s.

Curiosidades: Case de 228 Devices com 50 workers







Obrigado

www.ix.br

wprado@nic.br

20 de março de 2024

nic.br cgi.br

www.nic.br | www.cgi.br