Network Automation Open-Source Tools

Nornir, NAPALM, and Netmiko: Norse Gods, the CLI in flames, and ‘do you support telnet’

NETMIKO

$ whoami

Kirk Byers
Network Engineer:
CCIE #6243 (emeritus)

Programmer:
Netmiko
NAPALM
Nornir

Teach Python, Ansible, Nornir in a Network Automation context

Where am I biased?

1. I have worked a lot on both Netmiko and NAPALM (and to a certain extent on Nornir).
2. I sell courses teaching Python, Ansible, and Nornir (for network automation).
3. I have lot of Python experience.
4. I fall much more into the network engineers should learn programming skills camp.

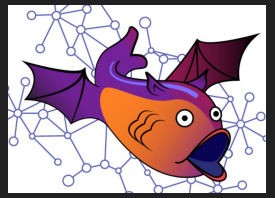
What is the purpose of this presentation?

- To show you some open-source libraries that might be valuable to you in network automation AND what problems those libraries are designed to solve.

- To show you it is not too difficult to get started using these libraries.

But there are lots of other options...

This is in no way intended to be inclusive nor does it provide a lot of context on the trade-offs of using one tool versus another tool.



Trade-offs, trade-offs, trade-offs (commercial vs open-source)

- Cost of software versus cost of your work.
- Do you have the internal skills to be successful?
- Will the open-source library continue to be maintained across time?
- Does it work well enough for your context? How many special changes do you require?

Three core tools: Netmiko, NAPALM, Nornir (N^3)

NETMIKO

NAPALM

Nornir

N^3

NETMIKO

- All written in Python.
- Each have different purposes and can be used together.
- Credit and Blame:

  Netmiko - Kirk Byers (Me)

  NAPALM - Mircea Ulinic, David Barroso, and Kirk Byers

  Nornir - David Barroso

Netmiko

Why does Netmiko exist?

To simplify network automation to legacy devices using SSH.

Relatively simple code.

Low-level CLI interaction is (hopefully) abstracted away.

Reliable, but reliable and CLI are hard to achieve.

*Reasonably good performance (*cough).*

Can be coupled/layered with other tools: NAPALM, Salt, Ansible, Nornir, et cetera

Netmiko Plus

Netmiko + TextFSM (ntc-templates)

Netmiko + PyATS/Genie

Netmiko + Jinja2

Netmiko and Telnet/Serial

Netmiko and Proxy-Servers

Netmiko + NAPALM

Netmiko + Nornir

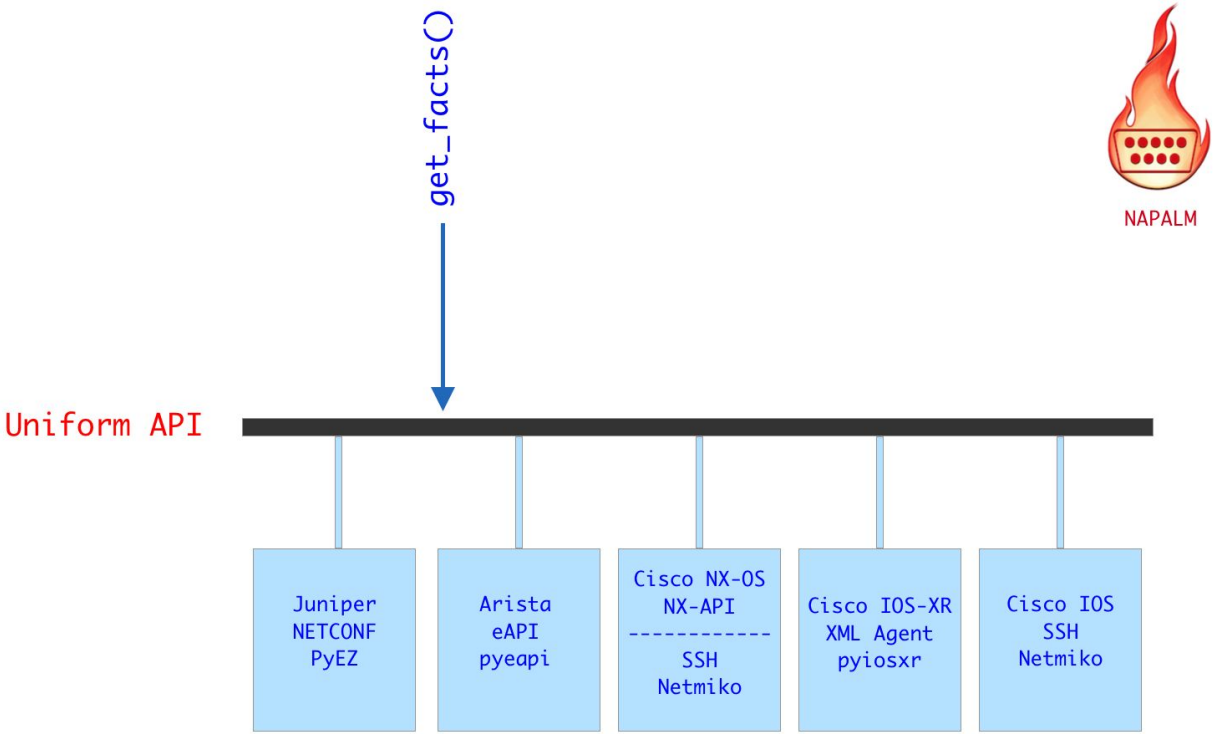
Example

```
from netmiko import ConnectHandler
import os

password = os.environ["NET_PASS"]

my_device = {
    "device_type": "cisco_ios",
    "host": "cisco5.lasthop.io",
    "username": "pyclass",
    "password": password,
}

net_connect = ConnectHandler(**my_device)
out = net_connect.send_command("show ip int brief", use_textfsm=True)
print(out)
```

NAPALM - The CLI in flames

What problem is NAPALM attempting to solve? Different programmatic interfaces for different APIs, but the desire for a uniform programming interface.

Heavily centered around five core platforms: Cisco IOS/IOS-XE, IOS-XR, NX-OS, Juniper Junos, and Arista EOS.


get_facts()
NAPALM
Uniform API
Juniper
NETCONF
PyEZ
Arista
eAPI
pyeapi
Cisco NX-OS
NX-API
SSH
Netmiko
Cisco IOS-XR
XML Agent
pyiosxr
Cisco IOS
SSH
Netmiko

NAPALM - Two main categories of use

1. Config Operations
2. Show Operations (getters)

There are other things, but these are the two main categories.

NAPALM Config Operations - ACID* Transactions

Provide a way to get close to ACID Transactions - Create a candidate configuration and support the following operations:

load_replace_candidate()
load_merge_candidate()
commit_config()
compare_config()
rollback()
discard_config()

Merge - Update part of the configuration.
Replace - Update the entire configuration.

NAPALM Getters

Getters support matrix

Note

The following table is built automatically. Every time there is a release of a supported driver a built is triggered. The result of the tests are aggregated on the following table.

	EOS	IOS	IOSXR	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	✅	✅	✅	✅	✅	✅

NAPALM Getters - One way to solve the “parsing problem”.

Other options:

TextFSM (ntc-templates)
Genie/PyATS
Pipe JSON / Pipe XML
Custom RegEx

NAPALM Bindings to other frameworks

Salt
Ansible
Nornir
Netpalm
StackStorm

Example

```
import os
from pprint import pprint
from napalm import get_network_driver

password = os.environ["NET_PASS"]
driver = get_network_driver("ios")

napalm_conn = driver(
    hostname="cisco5.lasthop.io", username="pyclass", password=password
)
napalm_conn.open()
facts = napalm_conn.get_facts()
pprint(facts)
```

Nornir - Revenge of the Nordic gods (and of the Spaniard).

What problem is Nornir trying to solve? A structured way of running, and managing your network automation but written entirely in a general purpose programming language (Python). No custom DSL.

Systematically manage inventory and handles concurrency.

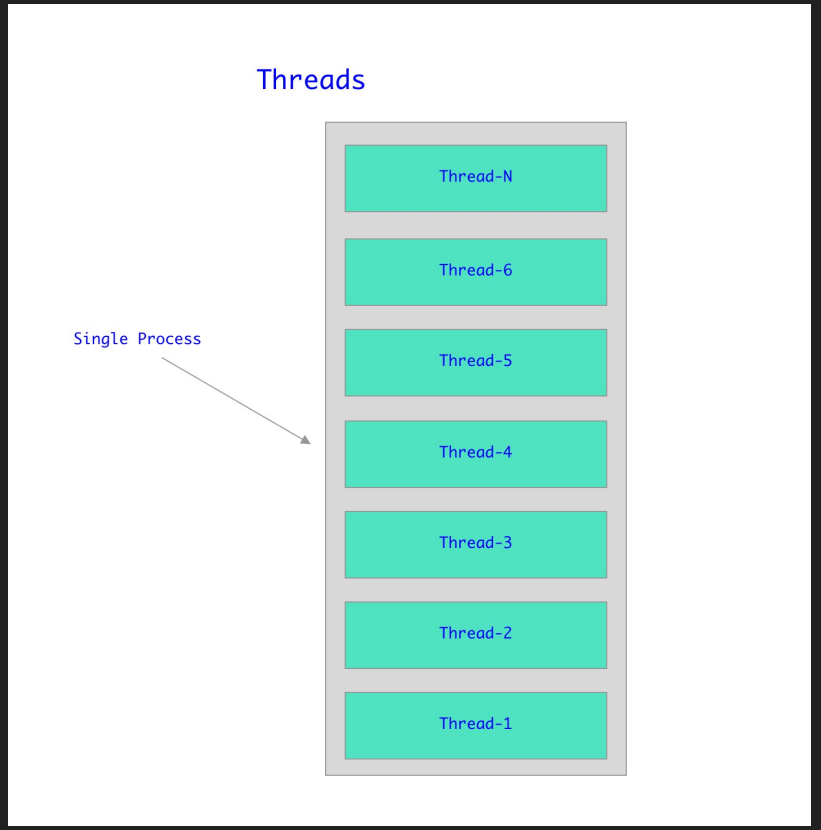
Nornir - Pluggable framework with support for Netmiko and NAPALM Plugins

```
$ tree -C .
.
├── __init__.py
├── napalm_cli.py
├── napalm_configure.py
├── napalm_get.py
├── napalm_ping.py
├── napalm_validate.py
...
├── netmiko_commit.py
├── netmiko_file_transfer.py
├── netmiko_save_config.py
├── netmiko_send_command.py
├── netmiko_send_config.py
```

Nornir - Inventory Plugins

```
$ tree -C .
.
├── ansible.py
├── __init__.py
├── netbox.py
├── nsot.py
└── simple.py
```

Concurrency using threading.



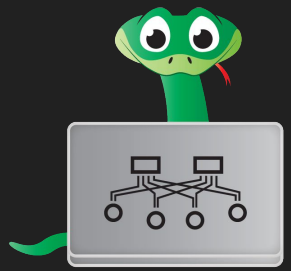
Nornir - Code is written entirely in Python.

```
from nornir import InitNornir
from nornir.core.filter import F
from nornir.plugins.functions.text import print_result  # noqa

def netmiko_direct(task):
    # Manually create Netmiko connection
    net_connect = task.host.get_connection("netmiko", task.nornir.config)
    output = net_connect.send_command("show ip int brief")
    print(output)

if __name__ == "__main__":
    nr = InitNornir(config_file="config.yaml")
    ios_filt = F(groups__contains="ios")
    nr = nr.filter(ios_filt)
    nr.run(task=netmiko_direct)
```

Questions?

ktbyers@twb-tech.com

https://pynet.twb-tech.com



PYTHON

FOR NETWORK ENGINEERS