Cisco Automation Framework Documentation

Release 0.5.9+dev

Kyle Kowalczyk

Jun 06, 2019

Contents

1 Requirements: 3

2 Contents: 5

2.1 Project Goals 5
2.2 Install Instructions 5
2.3 Usage Examples 5
2.4 CiscoAutomationFramework Code Documentation 7

3 Indices and tables 9

The Cisco Automation Framework is intended to provide a network administrator or network engineer a single tool to allow them to write scripts that when run against any Cisco device return consistent output so the author does not have to account for changes in command syntax or command output.

> **Warning:** This documentation is a work in progress and is in no way complete or comprehensive at this time, it is being actively developed and more information will come in time.

CHAPTER 1

Requirements:

Paramiko

CHAPTER 2

Contents:

2.1 Project Goals

Create a framework that is able to issue commands to a Cisco device regardless of device type or firmware level (IOS, NXOS, IOSXE, ASA) and return the data in a format that a script writer would find the most useful. In addition to that these commands should be able to be issued via any transport medium and return the same results in the same format weather is SSH, serial, or telnet.

For example: when getting the MAC address table it is best to return a list of lists with the appropriate columns in each list so then the script creator can iterate over the mac address table and pluck out each column they need to use vs haveing to find a way to parse that data theirselves

2.2 Install Instructions

Install main branch from git repo

```
pip install git+http://git.smallguysit.com/CiscoAutomationFramework.git
```

Install a specific branch from git repo

```
pip install git+http://git.smallguysit.com/CiscoAutomationFramework.
git@BRANCHNAME
```

2.3 Usage Examples

Below are some example scripts showing how to use the framework to interact with remote devices

2.3.1 Display running config using SSH

```
from CiscoAutomationFramework import CAF

# credentials for remote device
ip = '192.168.1.1'
username = 'user4'
password = 'password1'
enable_password = 'myenablePassword1'

# log into device and capture the running config
with CAF('ssh' ip, username, password, enable_password) as ssh:
    running_config = ssh.show_run()

# print running config
print(running_config)
```

2.3.2 List the number of ports on a device using SSH

```
from CiscoAutomationFramework import CAF

# credentials for remote device
ip = '192.168.1.1'
username = 'user4'
password = 'password1'
enable_password = 'myenablePassword1'

# log into device and capture the running config
with CAF('ssh' ip, username, password, enable_password) as ssh:
    hostname = ssh.hostname
    port_inv = ssh.physical_port_inventory_longname()

# print running config
print('Device {} has a total of {} ports'.format(hostname, len(port_inv)))
```

2.3.3 List running config of a device using serial interface

```
from CiscoAutomationFramework import CAF

# credentials for remote device
interface = 'COM4'
username = 'user4'
password = 'password1'
enable_password = 'myenablePassword1'

# log into device and capture the running config
with CAF('serial' interface, username, password, enable_password) as serial:
    running_config = serial.show_run()

# print running config
print(running_config)
```

2.4 CiscoAutomationFramework Code Documentation

2.4.1 Framework.py

2.4.2 Util.py

CHAPTER 3

Indices and tables

- genindex
- modindex
- search