Network Automation Foundations

Presented By CBTS & Arista Networks

8-6-2019
Craig Bruenderman
CCIE #46341

Rhetorical Questions

- Why are you here?
- What is your problem statement?
- Do you know how your business makes money?
  - What is your Unit of Value?
  - Knowing that will help you connect to your business
- What are your use cases?

Network Automation

- What?
  - “Network automation is the process of automating the configuration, management, testing, deployment, and operations of physical and virtual devices within a network. Every day network tasks and functions are performed automatically.” Juniper Networks
- Why?
  - Speed? Maybe.
  - Multiplex yourself
  - Predictability/Determinism
  - Drive heterogeneous environments with same toolkit
  - Reduce the mundane and get to workflows that move your career, and your business forward

Basic Toolkit

- Python Programming Language
- Fundamental Data Structures
- IDE
  - Combines a text editor with add-ons such as code completion, debugging, build tools, version control integration, integrated terminal, etc.
- API Concepts
- SDK Concepts

Python

- General-purpose object-oriented programming language developed since the 1980's by Guido van Rossum
- Open source, itself written in C
- Interpreted
  - Internally, your source code is translated into machine code
  - Byte code then runs in a platform virtual machine, similar to Java
- High-level language, so approachable, easy(ish) to read by humans
- Huge ecosystem of extension libraries
  - Text processing, Bioinformatics, Network Automation, Machine Learning…

Python Versions

- Python 2
  - Lots of code written for Python 2.7
  - 2.7 will be deprecated on Jan 1 2020
    - https://pythonclock.org/
- **Python 3**
  - Start here
  - Certainly differences, but minimal for our purposes when starting out
    - `print “Hello World” becomes print(“Hello World”)`

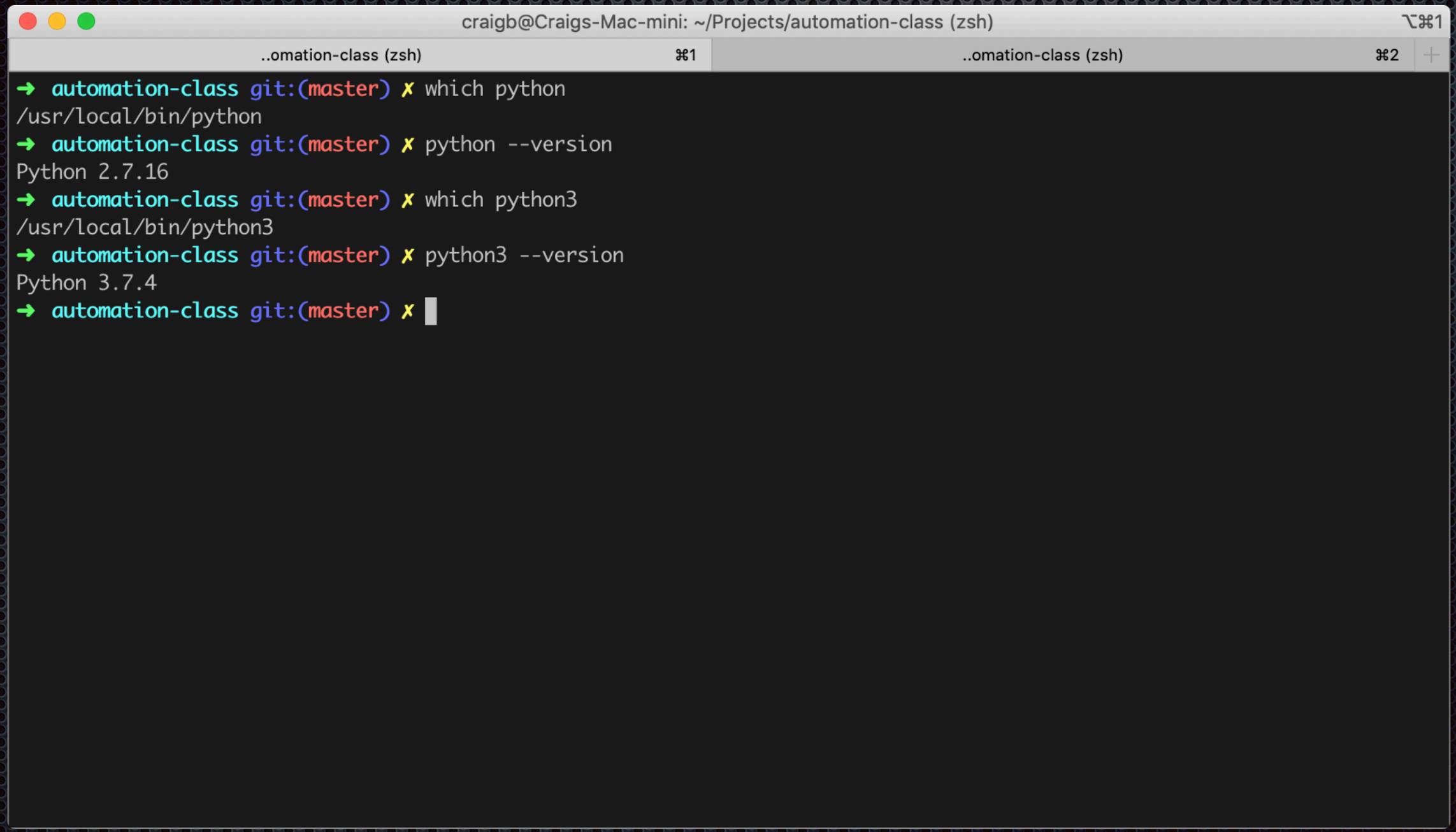
Python Run Modes

- Interactive
  - Run python from CLI
  - Do prototyping here
  - Can’t do this with compiled languages
- Scripted
  - Capture your code in .py files

Python Modules

- These are Python libraries you can include to avoid reinventing the wheel
- PIP - Package manager to install/remove/upgrade Modules
- Netmiko
  - Module / library for interacting with network equipment

Lab 1.1 - Python Paths and Versions



Lab 1.2 - Pip Package List

craigb@Craigs-Mac-mini: ~/Projects/automation-class (zsh) ⌥⌘1

..omation-class (zsh) ⌘1 | ..omation-class (zsh) ⌘2

```
➜ automation-class git:(master) ✗ pip list
DEPRECATION: Python 2.7 will reach the end of its life on January 1st, 2020. Please upgrade your Python as Python 2.7 won't b
e maintained after that date. A future version of pip will drop support for Python 2.7.
Package                       Version
----------------------------- --------
asn1crypto                    0.24.0
astroid                       1.6.5
backports.functools-lru-cache 1.5
bcrypt                        3.1.7
certifi                       2019.3.9
cffi                          1.12.3
Click                         7.0
configparser                  3.7.4
cryptography                  2.7
docutils                      0.14
enum34                        1.1.6
futures                       3.2.0
ipaddress                     1.0.22
isort                         4.3.16
lazy-object-proxy             1.3.1
mccabe                        0.6.1
meraki                        0.34
netmiko                       2.4.0
paramiko                      2.6.0
pathspec                      0.5.9
pip                           19.0.3
pip-tools                     3.8.0
pycparser                     2.19
pylint                        1.9.4
PyNaCl                        1.3.0
pyparsing                     2.4.0
pyserial                      3.4
python-dateutil               2.8.0
PyYAML                        5.1.1
scp                           0.13.2
setuptools                    40.8.0
singledispatch                3.4.0.3
six                           1.12.0
textfsm                       0.4.1
ucsmsdk                       0.9.8
```

Lab 1.3 - Interactive Python

craigb@Craigs-Mac-mini: ~/Projects/automation-class (zsh) — 120×30 ⌥⌘1

..omation-class (zsh) ⌘1 | ..omation-class (zsh) ⌘2

```
➜  automation-class git:(master) ✗ python3
Python 3.7.4 (default, Jul  9 2019, 18:13:23)
[Clang 10.0.1 (clang-1001.0.46.4)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> d = {'name': 'craig', 'employer': 'cbts', 'spirit animal': 'sloth'}
>>> d.keys()
dict_keys(['name', 'employer', 'spirit animal'])
>>> d.values()
dict_values(['craig', 'cbts', 'sloth'])
>>> d.items()
dict_items([('name', 'craig'), ('employer', 'cbts'), ('spirit animal', 'sloth')])
>>> exit()
➜  automation-class git:(master) ✗
```

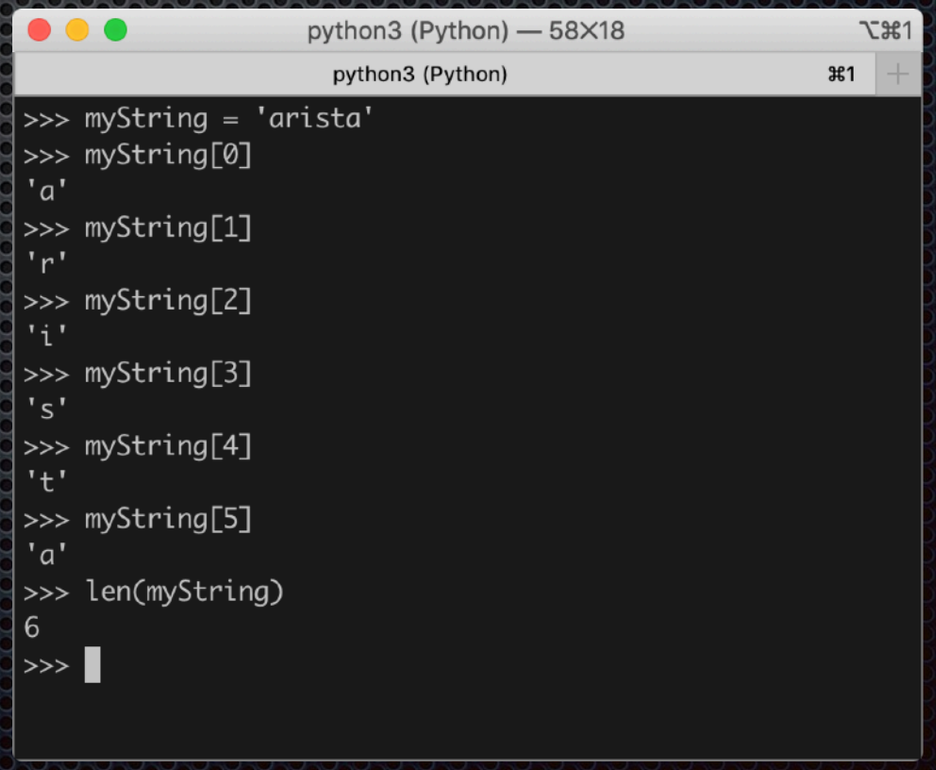
Data Objects

- Everything in Python is an Object
  - Objects are just pieces of memory with values and sets of associated operations
- Python has several built-in data object types
  - Numbers, Strings, booleans
  - Lists, Dictionaries, Tuples
- More to this topic, but that will get us started

Lists

- Positionally ordered sequence of data objects
- Entries are indexed (referenced) by their integer index, which starts at 0
- In Python, a string is technically a list of single-character strings

- Examples
  - In Python
  - In JSON
  - In YAML



Dictionaries

- Unordered mappings of key - value pairs
- Entries are indexed by their key, which can be multiple types, but is often a string
- Examples
  - In Python
  - In JSON
  - In YAML

Data Structure

- **Unstructured** - No predefined data model
  - Text files, media files, screen scrapes
- **Structured** - Pre-defined data types with predictable characteristics; easy to parse; usually text-based
  - Dates, IP addresses, E.164 telephone numbering plan
- **Semi-Structured** - Somewhere in between; uses internal tags and/or hierarchy to establish parent/child relationships
  - XML, JSON (Javascript Object Notation), YAML (YAML Ain't Markup Language)

Data Exchange Formats

- Designed to be exchanged/consumed by software
- Several to choose from
  - XML — Heavier weight
  - JSON — Lighter weight, maps closely to native data structures in programming languages
  - YAML — Easiest to read
- Largely used for storing and exchanging:
  - Lists (Arrays)
  - Dictionaries (Associative Arrays)

Lab 2.1 - Lab Prep

- http://bit.ly/cbautomation

```
craigb@Craigs-Mac-mini: ~ (zsh)                                    ⌥⌘1
~ (zsh)                ⌘1      ..omation-class (zsh)          ⌘2
➜  ~ git clone https://github.com/craigbruenderman/automation-class.git
Cloning into 'automation-class'...
remote: Enumerating objects: 115, done.
remote: Total 115 (delta 0), reused 0 (delta 0), pack-reused 115
Receiving objects: 100% (115/115), 49.82 MiB | 26.01 MiB/s, done.
Resolving deltas: 100% (64/64), done.
➜  ~ tree automation-class
automation-class
├── Lab_Guide.md
├── Lab_Guide.pdf
├── Network_Foundations_Preso.key
├── Network_Foundations_Preso.pdf
├── README.md
└── code
    ├── group_vars
    │   ├── all.yml
    │   └── webservers.yml
    ├── host_vars
    │   └── db1.yml
    ├── hosts
    ├── lab2.2.py
    ├── lab2.3.py
    ├── lab4.1.py
    ├── lab7.1.yml
    ├── lab7.2.yml
    ├── lab8.1.yml
    ├── ports.json
    ├── ports.yml
    └── reports
        ├── facts.j2
        └── master.md

4 directories, 19 files
➜  ~
```

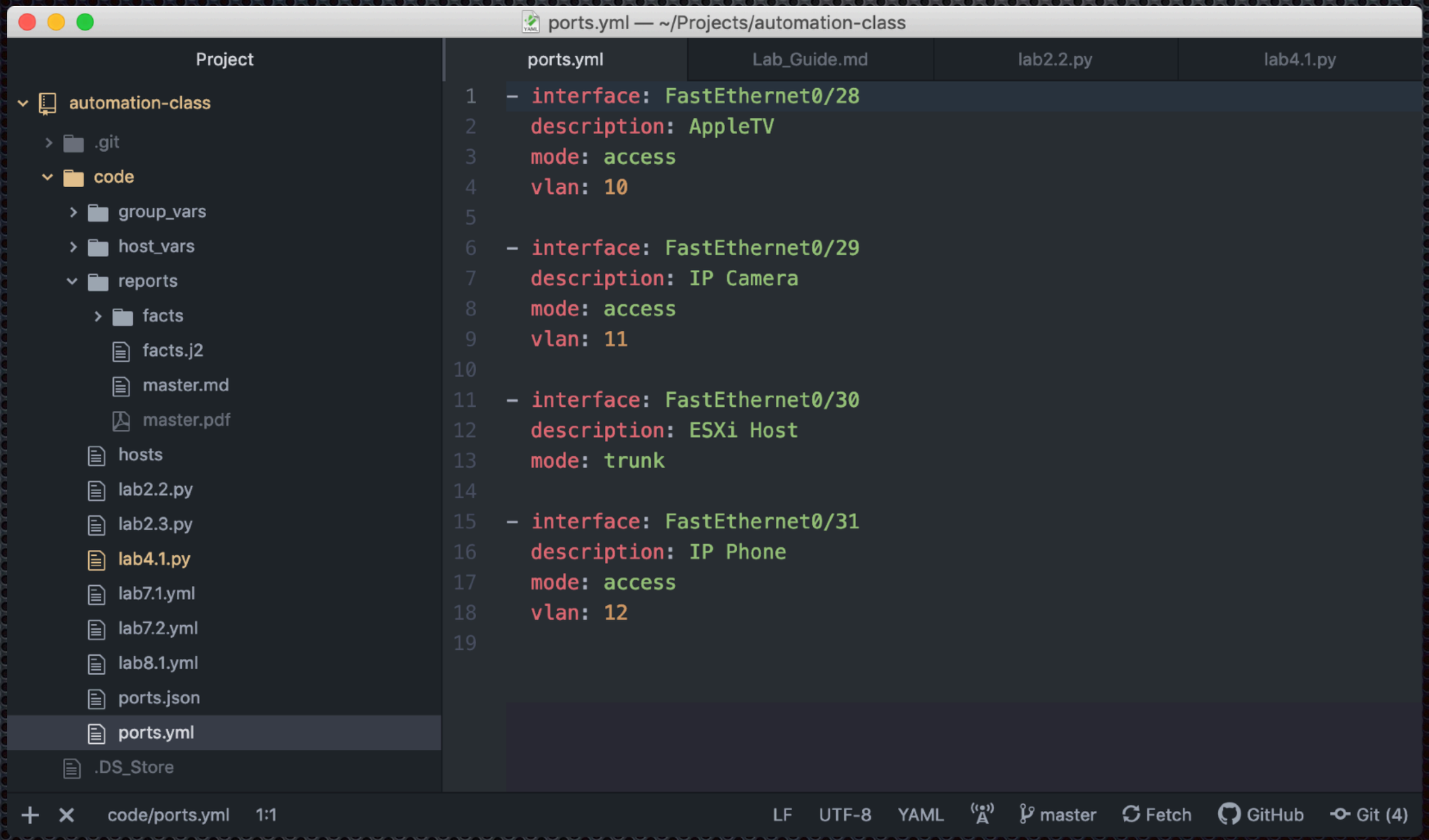
Lab 2.2 - Data Structures

python -i lab2.2.py (Python) — 156×47 ⌥⌘1

python (Python) ⌘1

```
➜ code git:(master) ✗ python -i lab2.2.py
>>> type(dns_servers)
<type 'list'>
>>> dns_servers
['8.8.8.8', '8.8.4.4']
>>> core1
{'vlans': [{'name': 'Data', 'id': 10}, {'name': 'Voice', 'id': 50}], 'hostname': 'core1.cbts.com', 'vendor': 'arista', 'users': {'admin': 15, 'craigb': 0},
'dns': ['8.8.8.8', '8.8.4.4']}
>>> as_python()
------ As Python Native List ------
<type 'list'>
['8.8.8.8', '8.8.4.4']


------ As Python Native Dictionary ------
<type 'dict'>
{'dns': ['8.8.8.8', '8.8.4.4'],
 'hostname': 'core1.cbts.com',
 'users': {'admin': 15, 'craigb': 0},
 'vendor': 'arista',
 'vlans': [{'id': 10, 'name': 'Data'}, {'id': 50, 'name': 'Voice'}]}


>>> as_yaml(core1)
------ As YAML ------
dns:
- 8.8.8.8
- 8.8.4.4
hostname: core1.cbts.com
users:
  admin: 15
  craigb: 0
vendor: arista
vlans:
- id: 10
  name: Data
- id: 50
  name: Voice

>>> as_json(core1)
------ As JSON ------
{"vlans": [{"name": "Data", "id": 10}, {"name": "Voice", "id": 50}], "hostname": "core1.cbts.com", "vendor": "arista", "users": {"admin": 15, "craigb": 0},
"dns": ["8.8.8.8", "8.8.4.4"]}


>>>
```

Lab 2.3 - Data Structures



Lab 2.3 - Convert Between Data Structures

craigb@Craigs-Mac-mini: ~/Projects/automation-class/code (zsh) ⌥⌘1

..on-class/code (zsh) ⌘1 | ..omation-class (zsh) ⌘2

```
➜  code git:(master) ✗ ls
group_vars hosts      lab2.3.py  lab7.1.yml lab8.1.yml ports.yml
host_vars  lab2.2.py  lab4.1.py  lab7.2.yml ports.json reports
➜  code git:(master) ✗ python3 -i lab2.3.py
<class 'list'> 4

<class 'dict'>
<class 'dict'>
<class 'dict'>
<class 'dict'>


[{'description': 'AppleTV',
  'interface': 'FastEthernet0/28',
  'mode': 'access',
  'vlan': 10},
 {'description': 'IP Camera',
  'interface': 'FastEthernet0/29',
  'mode': 'access',
  'vlan': 11},
 {'description': 'ESXi Host', 'interface': 'FastEthernet0/30', 'mode': 'trunk'},
 {'description': 'IP Phone',
  'interface': 'FastEthernet0/31',
  'mode': 'access',
  'vlan': 12}]
>>>
➜  code git:(master) ✗ ls
group_vars hosts      lab2.3.py  lab7.1.yml lab8.1.yml ports.yml
host_vars  lab2.2.py  lab4.1.py  lab7.2.yml ports.json reports
➜  code git:(master) ✗
```

IDE Overview

- Integrated Development Environment
- Distinction between IDE and Code Editor not always clear
- Integrate several tools
  - Source code editor with syntax highlighting, auto-completion
  - Shortcuts
  - Plugin System
  - Version Control System Integration
  - Build, execution, debugging tools

IDE Options

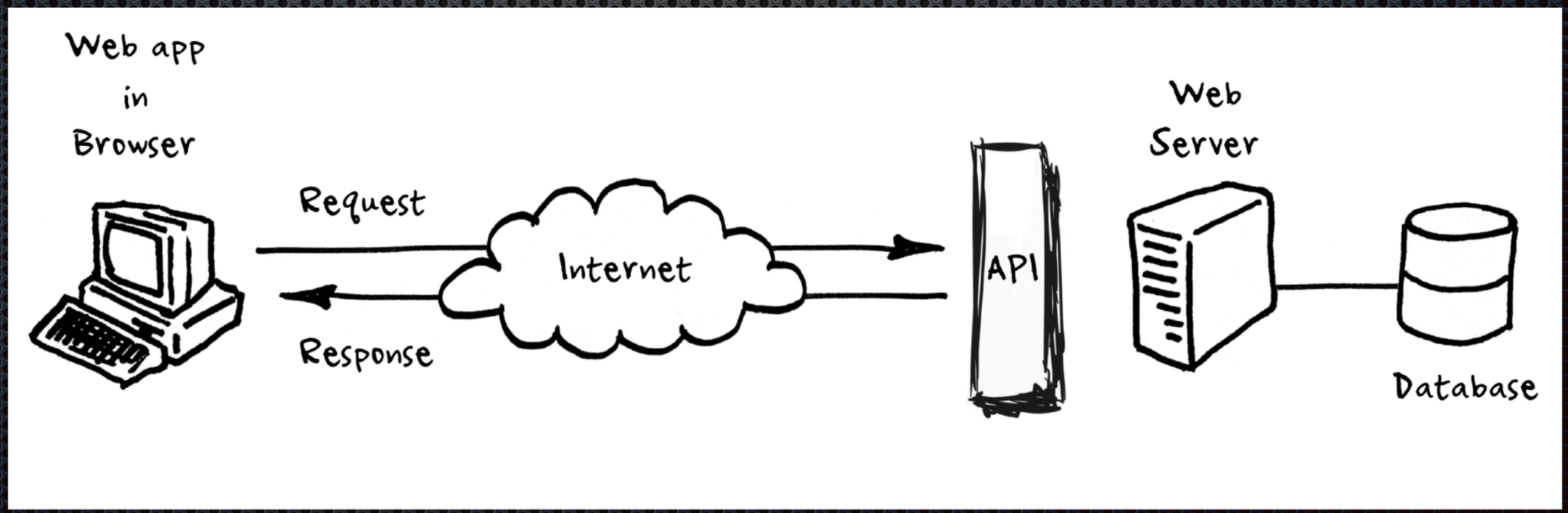
- Visual Studio Code
  - Multi-platform, Open Source, developed by Microsoft
- Atom
  - Multi-platform, Open Source, developed by Google
- PyCharm
  - Multi-platform, Freemium, developed by JetBrains
- Sublime Text
  - Multi-platform, Commercial

API Overview

- Application Programming Interface
- In basic terms, APIs just allow applications to communicate with one another.
- When people speak of “an API”, they sometimes generalize and actually mean “a publicly available web-based API that returns data, likely in JSON or XML”
- The API is not the database or even the server, it is the code that governs the access point(s) for the server

Web Services API

- Data exchange format is XML or JSON
- This lets us represent complex data structures
- Web Services APIs exchange data using HTTP(s) as transport



REST Overview

- An architectural style defined by a set of principles developed by Roy Fielding in his PhD work
  - Style, not a standard or a protocol
- Focuses on resources (that is, things, rather than actions) and ways to access the resources
- Resources are different types of information accessed through URLs
- Just like going to a URL in your browser retrieves an information resource
- The URLs are accompanied by an HTTP method that specifies how you want to interact with the resource

Database	HTTP
Create	POST
Read	GET
Update	PUT
Delete	DELETE

REST Overview

- The base path (or base URL or host) refers to the common path for the API
- The endpoint refers to the end path of the endpoint
- The query string contains parameters for the endpoint

$$\underbrace{http://apiserver.com}_{\text{base path}}\ \underbrace{/homes}_{\text{endpoint}}\ \underbrace{?limit=5\&format=json}_{\text{query string}}$$

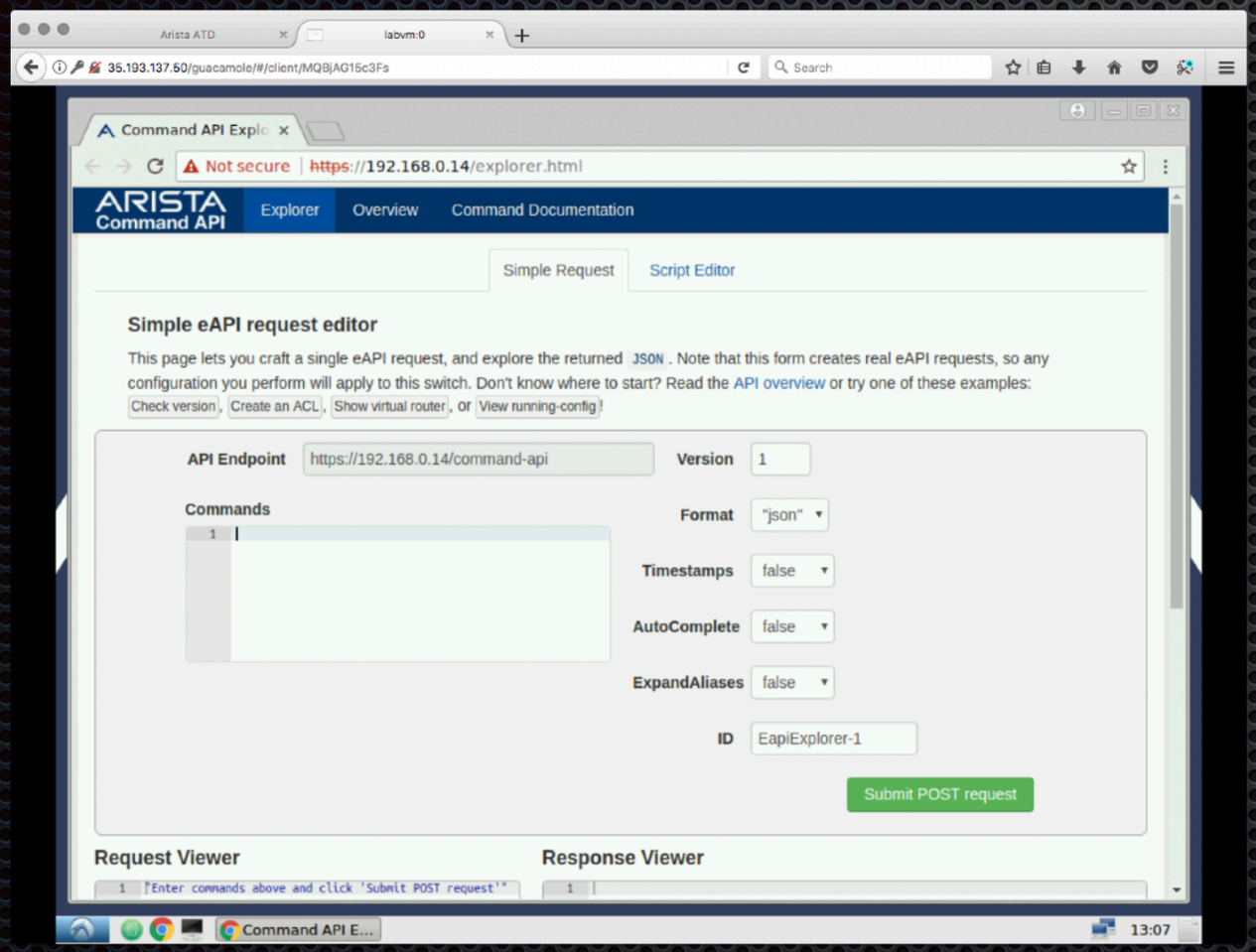
Web Service API - Arista eAPI

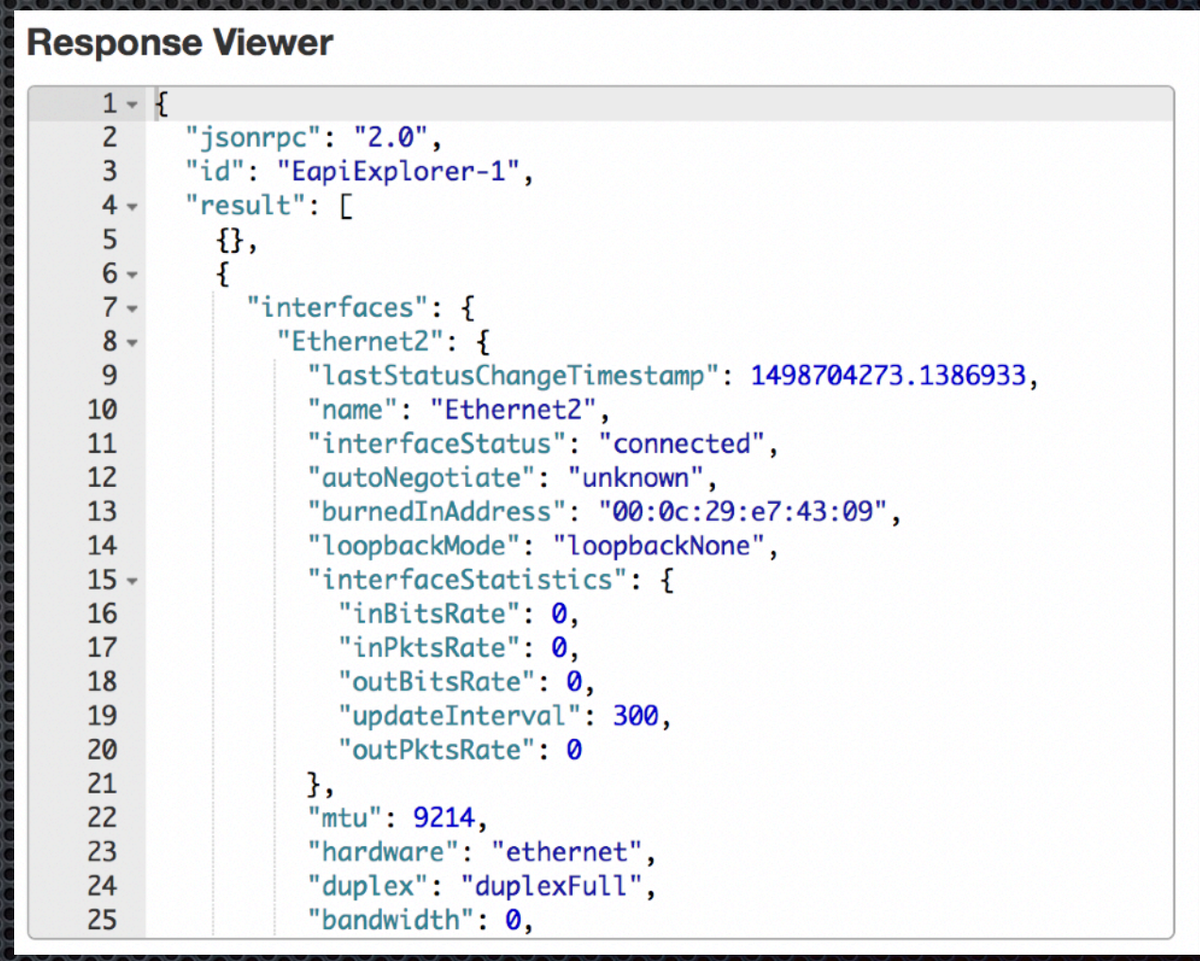
- eAPI is Arista's Web Service API
  - Provides a method for simply interacting with Arista gear without CLI interaction
- HTTP(s) transport
- JSON Request and Response
- eAPI allows CLI Commands to be issued remotely
- eAPI returns output as JSON, generally in key-value pairs

Navigating Web Service APIs

- Browser-based interaction
- Postman interaction
- Python REST interaction with requests library
- Python specific client library interaction (like pyeapi)
- Ansible interaction

Lab 3 - eAPI Command Explorer





Lab 4.1 - Add a user with pyeapi

arista@ip-10-33-5-52: ~/automation-class/code (ssh) ⌥⌘1

arista@ip-10-33-5-52: ~/automation-class/code (ssh) ⌘1 | ..omation-class (zsh) ⌘2

```
arista@ip-10-33-5-52:~/automation-class/code$ python lab4.1.py
Users on the device

{'admin': {'format': '5',
           'nopassword': False,
           'privilege': '15',
           'role': 'network-admin',
           'secret': '$1$5O85YVVn$HrXcfOivJEnISTMb6xrJc.',
           'sshkey': ''},
 'arista': {'format': '5',
            'nopassword': False,
            'privilege': '15',
            'role': '',
            'secret': '$1$4VjIjfd1$XkUVulbNDESHFzcxDU.Tk1',
            'sshkey': 'ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQC6bJB3TkBEQZ9jNyO1kbdU0P20gZ1D72CvsPNZ5S4bbciBNTT/MHX8GwyLmM9k+iha
HK2JtRhWFcdsm9MojRgjAuzw4wn/6pa92y/93GvaYL//dOBXrHctZsX3PX7TZFL9VVBVA8aFp5iXxEM8uyKWhxnBo/D0eR25Jed4gHVHQMi6Hyh7eKRpE3E6kvRlSkh
NikZ5EwdoM7lg2i6rjf7+o3G6isGtxliMZD98N6qWW79U6euS072qkK/q3dfgyHdd8a8MD5VLWbYR9ikhKwpXAmxcFn5aRllqXJ++QAW0NO78noI91ICRxpAuQSzgrn
tdwXdyFWiqyiD3AxK28qWZ arista@labaccess'},
 'testuser': {'format': 'sha512',
              'nopassword': False,
              'privilege': '15',
              'role': 'network-admin',
              'secret': '$6$/ZIghqg7TS/LDYOJ$yz/RMPf29PBfv71mSkQjcx0gmoolmchTLtEdX1dio1.HubQEM090JwbcQ6kcxhquEHzRQzszMlybCnaMEA
oxf/',
              'sshkey': ''}}


arista@ip-10-33-5-52:~/automation-class/code$
```

Lab 4.2 - Get VLAN data with pyeapi

arista@ip-10-33-5-52: ~/automation-class/code (ssh) ⌥⌘1

arista@ip-10-33-5-52: ~/automation-class/code (ssh) ⌘1 | ..omation-class (zsh) ⌘2

```
d is a dictionary of VLANs

VLAN0012
>>>
>>>
>>> type(d)
<type 'dict'>
>>> d
{'1': {'state': 'active', 'name': 'default', 'vlan_id': '1', 'trunk_groups': []}, '12': {'state': 'active', 'name': 'V
LAN0012', 'vlan_id': '12', 'trunk_groups': []}, '34': {'state': 'active', 'name': 'VLAN0034', 'vlan_id': '34', 'trunk_
groups': []}}
>>> d.keys()
['1', '12', '34']
>>> d['12']['name']
'VLAN0012'
>>> d['12']['state']
'active'
>>>
```

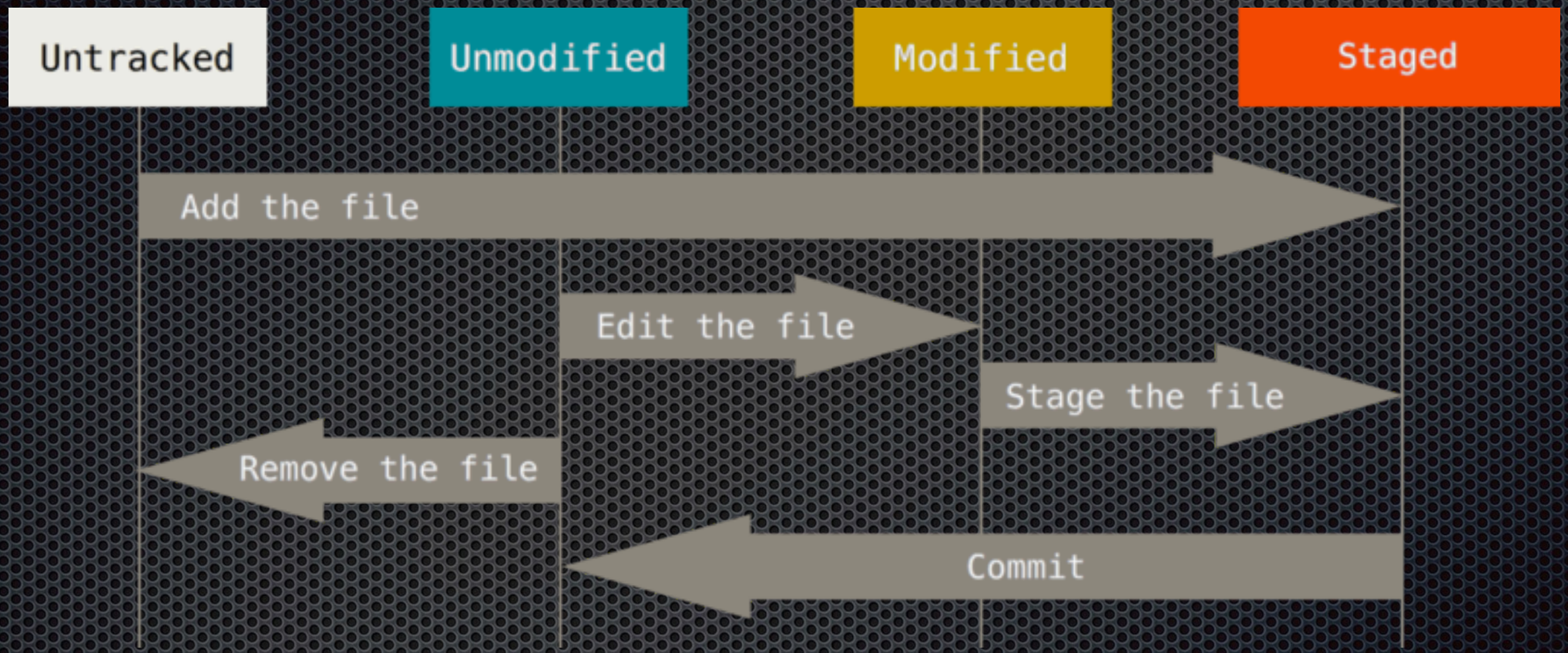
Expanded Toolkit

- Git
- Github
- Ansible

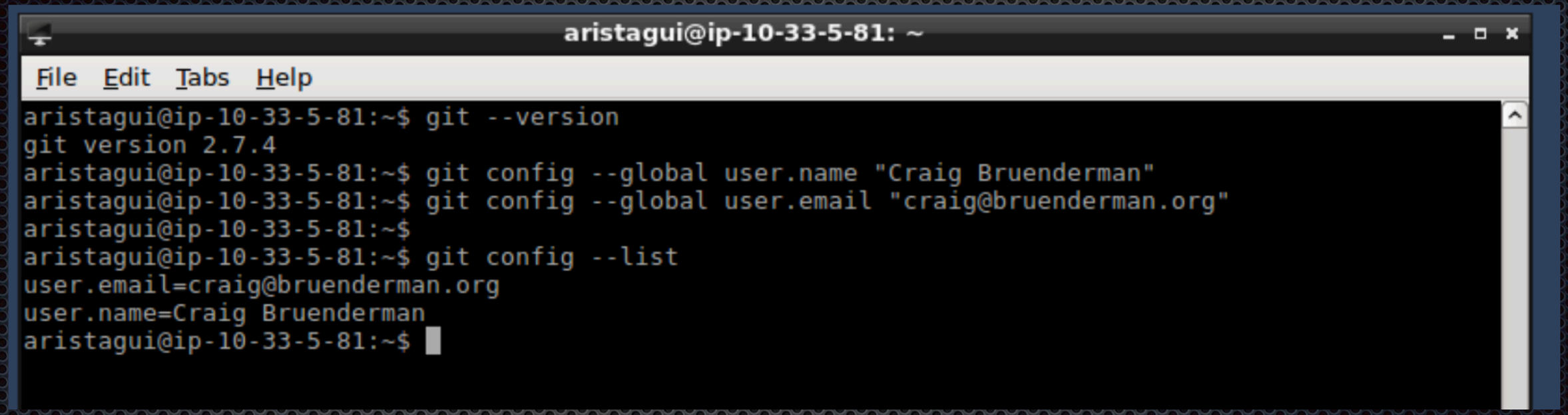
Git Overview

- **Version Control System -** System in which we store and maintain versioned repository of source code
  - Enables collaboration between multiple contributors
- Git is one such free source/version control system that is distributed by design
- Credit (blame) log
- Excellent free book https://git-scm.com/book/en/v2

Git Workflow



Lab 5.1 - Git Preferences Setup



Lab 5.2 - Git Create Your Own Repository

aristagui@ip-10-33-5-81: ~/muh-bad-codez

File Edit Tabs Help

```
aristagui@ip-10-33-5-81:~$ mkdir muh-bad-codez
aristagui@ip-10-33-5-81:~$ cd muh-bad-codez/
aristagui@ip-10-33-5-81:~/muh-bad-codez$ git init
Initialized empty Git repository in /home/aristagui/muh-bad-codez/.git/
aristagui@ip-10-33-5-81:~/muh-bad-codez$ echo "Info here" >> README.md
aristagui@ip-10-33-5-81:~/muh-bad-codez$ git add README.md
aristagui@ip-10-33-5-81:~/muh-bad-codez$ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   README.md

aristagui@ip-10-33-5-81:~/muh-bad-codez$ git commit -m "my first commit"
[master (root-commit) 6c2aa7d] my first commit
 1 file changed, 1 insertion(+)
 create mode 100644 README.md
aristagui@ip-10-33-5-81:~/muh-bad-codez$ git status
On branch master
nothing to commit, working directory clean
aristagui@ip-10-33-5-81:~/muh-bad-codez$
```

Git Branches

- Branching means you diverge from the main line of development and continue to do work without messing with that main line
- These are useful for playing with features, testing, multi-user
- You probably don’t want to commit incomplete or broken code, but you still want to keep it safe in the repo
- Git keeps a special pointer called HEAD that points to the current branch you’re on

Lab 5.3 - Branch and Merge

- This is a bit of an advanced topic and often confusing
- Very optional

Github

- Since Git is a distributed VCS, where is the central source of truth for the repo?
  - Run your own central Git server
  - Use a SaaS
- Popular hosting service for storing Git source repositories
- Allows public and private repos
- 100MM+ repos and growing
- Uses “freemium” model
- Bought by Microsoft June 2018

Lab 5.4 - View Code Diffs in Github

craigbruenderman committed 1 hour ago — 1 parent 672424d — commit f999e6fae0b729e77e7953039608b2cb6887e699

Showing **3 changed files** with **9 additions** and **2 deletions**. Unified | Split

BIN -33.1 KB (88%) Lab_Guide.pdf

Binary file not shown.

11 code/ansible/hosts

```
@@ -1,2 +1,9 @@
-[veos]
-192.168.0.14
+[webservers]
+web1 ansible_host=10.0.0.1
+web2 ansible_host=10.0.0.2
+
+[database_servers]
+db1 ansible_host=172.16.99.1
+
+#[veos]
+#switch1 ansible_host=192.168.0.14
```

0 code/ansible/install-jq.yml → code/ansible/lab8.1.yml

File renamed without changes.

0 comments on commit f999e6f

Ansible Overview

- **Ansible** is an open source project for software provisioning, configuration management, and application deployment
  - Infrastructure as Code Tool sponsored by Red Hat (now owned by IBM, fingers crossed)
- **AWX** - community project that enables users to better control their Ansible project use in IT environments
  - AWX is the upstream project from which the commercial Red Hat Ansible Tower offering is ultimately derived
- **Ansible Tower** - is produced by taking selected releases of AWX, hardening them for long-term supportability, and making them available to customers as the Ansible Tower offering

Ansible

- Simple, yet extremely powerful automation
- Agent-less and self-contained
- Uses SSH or Web Service as transport
- Capable of automating network devices, servers, cloud providers, VMware, many OEMs and SaaS services
- Python based, so extendable
- YAML driven, for easier digestion
- **Idempotent** — ability to perform same action repeatedly while producing the same result, without harm; making multiple identical requests has the same effect as making a single request

Ansible Terminology

- **Tasks** – smallest unit of work in Ansible; combines a module action with a task name
- **Play** – mapping of task instructions to the set of hosts they will run against
- **Playbook** — collection of at least one play by which Ansible orchestrates, configures, administers, or deploys systems
- **Inventory** — file that defines hosts and groups in Ansible
- **Modules** — task plugins that control particular resources

Ansible Modules

- 1200+ built-in modules including:
  - apt, yum, copy, command, cron, dns, docker, easy_install, ec2 (amazon modules), file, filesystem, find, git, known_hosts, mysql, mongodb, nagios, npm, openstack, rax (rackspace), pip, shell, snmp_facts…
- Typical modules for me
  - vmware, eos, ios, nxos, ucsm
- Modules differ in completeness

Lab 6.1 - Ansible Ad Hoc Command

craigb@Craigs-MacBook-Pro: ~/GoogleDrive/CBTS/Automation Foundations/repo/code/ansible (zsh) ⌥⌘1

..ndations/repo (zsh) ⌘1 | ../code/ansible (zsh) ⌘2 | ..e/code/python (zsh) ⌘3

```
➜  ansible git:(master) ✗ ansible localhost -m raw -a "ping -c 3 127.0.0.1"
 [WARNING]: No inventory was parsed, only implicit localhost is available

localhost | CHANGED | rc=0 >>
PING 127.0.0.1 (127.0.0.1): 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.050 ms
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.086 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.077 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.050/0.071/0.086/0.015 ms


➜  ansible git:(master) ✗
```

Lab 6.2 - Ad Hoc Package Installation

arista@ip-10-33-5-52: ~/automation-class/code (ssh) ⌥⌘1

arista@ip-10-33-5-52: ~/automation-class/code (ssh) ⌘1 | ..omation-class (zsh) ⌘2

```
arista@ip-10-33-5-52:~/automation-class/code$ ansible localhost -m apt -a "name=yamllint" --become -k -c local
SSH password:
 [WARNING]: Unable to parse /etc/ansible/hosts as an inventory source

 [WARNING]: No inventory was parsed, only implicit localhost is available

 [WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

localhost | CHANGED => {
    "cache_update_time": 1553007139,
    "cache_updated": false,
    "changed": true,
    "stderr": "",
    "stderr_lines": [],
    "stdout": "Reading package lists...\nBuilding dependency tree...\nReading state information...\nThe following NEW packages
will be installed:\n  yamllint\n0 upgraded, 1 newly installed, 0 to remove and 65 not upgraded.\nNeed to get 0 B/23.4 kB of arc
hives.\nAfter this operation, 124 kB of additional disk space will be used.\nSelecting previously unselected package yamllint.\
r\n(Reading database ... \r(Reading database ... 5%\r(Reading database ... 10%\r(Reading database ... 15%\r(Reading database ..
. 20%\r(Reading database ... 25%\r(Reading database ... 30%\r(Reading database ... 35%\r(Reading database ... 40%\r(Reading dat
abase ... 45%\r(Reading database ... 50%\r(Reading database ... 55%\r(Reading database ... 60%\r(Reading database ... 65%\r(Rea
ding database ... 70%\r(Reading database ... 75%\r(Reading database ... 80%\r(Reading database ... 85%\r(Reading database ... 9
0%\r(Reading database ... 95%\r(Reading database ... 100%\r(Reading database ... 179755 files and directories currently install
ed.)\r\nPreparing to unpack .../yamllint_1.2.1-1_amd64.deb ...\r\nUnpacking yamllint (1.2.1-1) ...\r\nProcessing triggers for m
an-db (2.7.5-1) ...\r\nSetting up yamllint (1.2.1-1) ...\r\n",
    "stdout_lines": [
        "Reading package lists...",
        "Building dependency tree...",
        "Reading state information...",
        "The following NEW packages will be installed:",
        "  yamllint",
        "0 upgraded, 1 newly installed, 0 to remove and 65 not upgraded.",
        "Need to get 0 B/23.4 kB of archives.",
        "After this operation, 124 kB of additional disk space will be used.",
        "Selecting previously unselected package yamllint.",
        "(Reading database ... ",
        "(Reading database ... 5%",
        "(Reading database ... 10%",
        "(Reading database ... 15%",
        "(Reading database ... 20%",
        "(Reading database ... 25%",
        "(Reading database ... 30%",
        "(Reading database ... 35%",
        "(Reading database ... 40%",
```

Lab 6.3 - Ansible ad hoc Against Inventory

```
arista@ip-10-33-5-52: ~/automation-class/code (ssh)                    ⌥⌘1
arista@ip-10-33-5-52: ~/automation-class/code (ssh)  ⌘1   ..omation-class (zsh)  ⌘2
arista@ip-10-33-5-52:~/automation-class/code$ ansible veos -i hosts -m raw -a "show version" -u arista -k
SSH password:
192.168.0.14 | CHANGED | rc=0 >>
Arista vEOS
Hardware version:
Serial number:       leaf1
System MAC address:  2cc2.60fd.3d06

Software image version: 4.21.2F
Architecture:           i386
Internal build version: 4.21.2F-10430819.4212F
Internal build ID:      7b26fbef-3d08-4910-bb95-df08faaa5b13

Uptime:                 0 weeks, 0 days, 0 hours and 29 minutes
Total memory:           3977260 kB
Free memory:            3197824 kB

Warning: Permanently added '192.168.0.14' (ECDSA) to the list of known hosts.
Shared connection to 192.168.0.14 closed.


192.168.0.11 | CHANGED | rc=0 >>
Arista vEOS
Hardware version:
Serial number:       spine2
System MAC address:  2cc2.6094.d76c

Software image version: 4.21.2F
Architecture:           i386
Internal build version: 4.21.2F-10430819.4212F
Internal build ID:      7b26fbef-3d08-4910-bb95-df08faaa5b13

Uptime:                 0 weeks, 0 days, 0 hours and 29 minutes
Total memory:           3977260 kB
Free memory:            3196868 kB

Warning: Permanently added '192.168.0.11' (ECDSA) to the list of known hosts.
Shared connection to 192.168.0.11 closed.


192.168.0.10 | CHANGED | rc=0 >>
Arista vEOS
Hardware version:
```

Ansible Inventory

- Contains a list of hosts which you want Ansible to interact with
- Can be as basic as containing a list of hostnames
- Can be more complex, containing groups, groups of groups, host variables, etc.
- General advice to keep inventory files minimal and put variables elsewhere

Ansible Variables

- Often we want to set the same parameter to a different value on a per-device or group basis
- DRY principle
- Multiple ways to define vars, depending on what we need
  - Directly within playbooks
  - Within inventory
  - In separate host variables and group variables files
- There is an order of precedence when vars are defined in more than one place, but try to avoid this

Ansible Group Variables

- Stored in group_vars directory
- Apply to all of the hosts which reside inside a group
- To configure group variables simply create a YAML file which matches the name of the group
- Also an **all** group variables file which specifies variables for all groups.

Ansible Host Variables

- Stored in host_vars directory
- Host variables work the same way as Group variables
- Instead of the filename matching group name, it needs to match the host name

Ansible Variables Example Layout

craigb@Craigs-Mac-mini: ~/Projects/automation-class/code (zsh) — 122×27 ⌥⌘1

..on-class/code (zsh) ⌘1 | ~ (zsh) ⌘2

```
➜  code git:(master) ✗ tree
.
├── group_vars
│   ├── all.yml
│   └── webservers.yml
├── host_vars
│   └── db1.yml
├── hosts
├── lab2.2.py
├── lab2.3.py
├── lab4.1.py
├── lab7.1.yml
├── lab7.2.yml
├── lab8.1.yml
├── ports.json
├── ports.yml
└── reports
    ├── facts
    │   ├── 192.168.10.2.md
    │   └── 192.168.10.3.md
    ├── facts.j2
    ├── master.md
    └── master.pdf

4 directories, 17 files
➜  code git:(master) ✗
```

Anatomy of an Ansible Playbook

```yaml
---
# Name of the playbook
- name: Our first Ansible playbook
  # List of hosts or host group to run against
  hosts: localhost
  # Toggle to run with root privileges
  become: true

  # Variables we define to use within this playbook
  vars:
    # Password to use when running command as root
    ansible_become_password: "@rista123"
    # Name of the package we'll install
    package_name: "jq"

  # List of tasks to be executed
  tasks:
    # Name of one task
    - name: Install jq
      # Action of this task
      # In this case, the apt package manager module
      apt:
        # Actual name of the package to be installed
        # Referencing the defined variable 'package_name'
        name: "{{ package_name }}"

```

Ansible Challenges

- Very whitespace sensitive
- Documentation sometimes needs its own documentation
- Multiple connection methods aren’t always clear

Lab 7.1 - Use Ansible Playbook to Install Packages

```
arista@ip-10-33-5-52: ~/automation-class/code (ssh)                    ⌥⌘1
arista@ip-10-33-5-52: ~/automation-class/code (ssh)  ⌘1   ..omation-class (zsh)  ⌘2
arista@ip-10-33-5-52:~/automation-class/code$ more ports.json
[{"interface": "FastEthernet0/28", "vlan": 10, "description": "AppleTV", "mode": "access"}, {"interface": "FastEthernet0/29", "
vlan": 11, "description": "IP Camera", "mode": "access"}, {"interface": "FastEthernet0/30", "description": "ESXi Host", "mode":
 "trunk"}, {"interface": "FastEthernet0/31", "vlan": 12, "description": "IP Phone", "mode": "access"}]
arista@ip-10-33-5-52:~/automation-class/code$
arista@ip-10-33-5-52:~/automation-class/code$ ansible-playbook -i hosts lab7.1.yml

PLAY [Our first Ansible playbook] **********************************************************************************************

TASK [Gathering Facts] *********************************************************************************************************
ok: [localhost]

TASK [Install some packages] ***************************************************************************************************
changed: [localhost] => (item=[u'pandoc', u'jq'])

PLAY RECAP *********************************************************************************************************************
localhost                  : ok=2    changed=1    unreachable=0    failed=0

arista@ip-10-33-5-52:~/automation-class/code$ jq . ports.json
[
  {
    "interface": "FastEthernet0/28",
    "vlan": 10,
    "description": "AppleTV",
    "mode": "access"
  },
  {
    "interface": "FastEthernet0/29",
    "vlan": 11,
    "description": "IP Camera",
    "mode": "access"
  },
  {
    "interface": "FastEthernet0/30",
    "description": "ESXi Host",
    "mode": "trunk"
  },
  {
    "interface": "FastEthernet0/31",
    "vlan": 12,
    "description": "IP Phone",
    "mode": "access"
  }
]
arista@ip-10-33-5-52:~/automation-class/code$
```

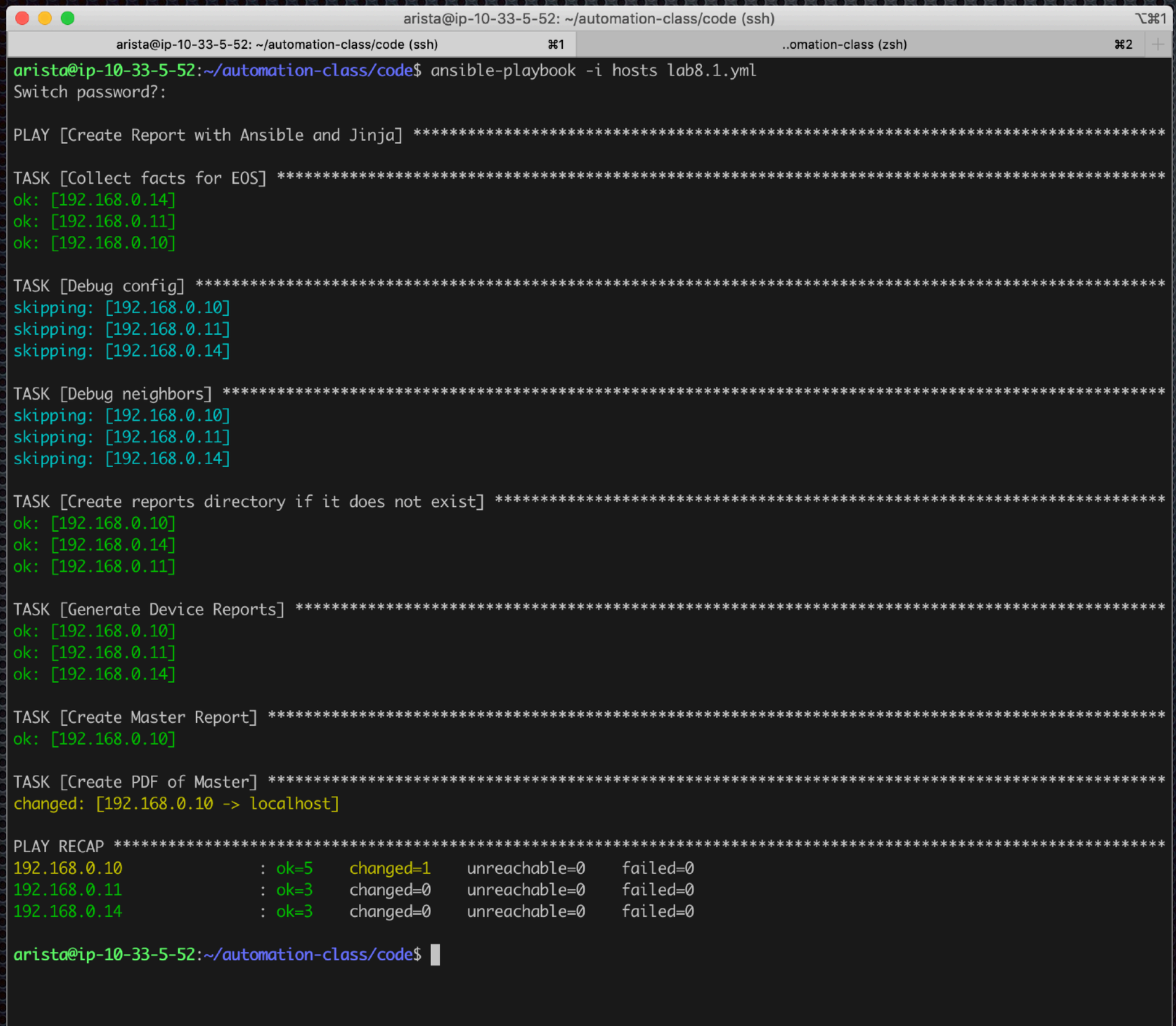
Lab 7.2 - Use Ansible to interact with the Twilio REST API

- **Prize Alert**
- First one to use Ansible to send me an MMS with their email address, switch serial, and an Ansible heart wins a prize

Jinja Templating Engine

- Jinja is a Python-based templating engine that works with Ansible out of the box
- Jinja templates take variables from Ansible and then output text
- It's like mail merge for configuration
- Writing Ansible playbooks that issue sequential commands to configure all 48+ interfaces on a switch is inefficient
- Jinja is also present in other parts of Ansible, such as filters and tests
- Jinja2 is the actual name of the Python library

Lab 8.1 - Jinja Templating



Use Cases

Use Case: Bulk Config Audit

meraki-dns-check.py

```python
#!/usr/local/bin/python3

from meraki import meraki
from tabulate import tabulate
from itertools import islice
import csv
import os

apikey = os.environ['MERAKI_API_KEY']
myOrgs = meraki.myorgaccess(apikey)
#print(myOrgs)

for i, entry in enumerate(myOrgs):
    if entry['name'] == "CustomerName":
        orgid = entry['id']

outputList = []
myNetworks = meraki.getnetworklist(apikey, orgid, suppressprint=True)
# Limiter for debugging
#limit = 3
limit = len(myNetworks)
print("%s Networks found" % limit)

for network in islice(myNetworks, limit):
    deviceList = meraki.getnetworkdevices(apikey, network["id"], suppressprint=True)
    for device in deviceList:
        deviceUplinks = meraki.getdeviceuplink(apikey, network["id"], device["serial"], suppressprint=True)
        for uplink in deviceUplinks:
            outputList.append({'Network Name': network.get("name", None), "Device Name" : device["name"], "Device Model" : device["model"], "Interface" : uplink.get("interface" ,
            None), "DNS" : uplink.get("dns", None), "Static IP" : uplink.get("usingStaticIp", None)})

print(tabulate(outputList, headers="keys"))

with open("pjs-devices.csv", "w") as csvfile:
    fields = outputList[0].keys()
    writer = csv.DictWriter(csvfile, fieldnames=fields)
    writer.writeheader()
    for row in outputList:
        writer.writerow(row)
    csvfile.close()

```

LF UTF-8 Python master Fetch GitHub Git (1)

Use Case: Bulk Config Audit

craigb@Craigs-Mac-mini: ~/Projects/meraki-tools (zsh) ⌥⌘1

..ons/repo/code (zsh) ⌘1 | ..ndations/repo (zsh) ⌘2 | ../meraki-tools (zsh) ⌘3

```
➜ meraki-tools git:(master) ✗ python3 meraki-dns-check.py
Organization Operation Successful - See returned data for results

707 Networks found
```

Network Name	Device Name	Device Model	Interface	DNS	Static IP
000000-HUB1-LOU-PN	LOU-HQ-SEC-MX450-P01	MX450	WAN 1	208.67.222.222, 208.67.220.220	True
000000-HUB1-LOU-PN	LOU-HQ-SEC-MX450-P01	MX450	WAN 2	208.67.222.222, 208.67.220.220	True
000000-HUB1-LOU-PN	LOU-HQ-PRI-MX450-P01	MX450	WAN 1	208.67.222.222, 208.67.220.220	True
000000-HUB1-LOU-PN	LOU-HQ-PRI-MX450-P01	MX450	WAN 2	208.67.222.222, 208.67.220.220	True
000000-HUB1-NSH-PN	NSH-PT-SEC-MX450-P01	MX450	WAN 1	208.67.222.222, 208.67.220.220	True
000000-HUB1-NSH-PN	NSH-PT-SEC-MX450-P01	MX450	WAN 2		
000000-HUB1-NSH-PN	NSH-PT-PRI-MX450-P01	MX450	WAN 1	208.67.222.222, 208.67.220.220	True
000000-HUB1-NSH-PN	NSH-PT-PRI-MX450-P01	MX450	WAN 2	208.67.222.222, 208.67.220.220	True
008670-KY-LOU-PN	008670-KY-LOU-PN	MX65W	WAN 1	209.18.47.63, 209.18.47.62	False
008670-KY-LOU-PN	008670-KY-LOU-PN	MX65W	WAN 2	208.67.222.222, 208.67.220.220	True
008670-KY-LOU-PN	008670-KY-LOU-SW-P01	MS120-48LP	WAN 1	208.67.222.222, 208.67.220.220	False
008670-KY-LOU-PN	008670-KY-LOU-SW-P02	MS225-24P	WAN 1		
004857-TX-AUS-PN	004857-TX-AUS-MX-P01	MX65W	WAN 1	209.18.47.63, 209.18.47.61	False
004857-TX-AUS-PN	004857-TX-AUS-MX-P01	MX65W	WAN 2	208.67.222.222, 208.67.220.220	True
004857-TX-AUS-PN	004857-TX-AUS-SW-P01	MS120-48LP	WAN 1	208.67.222.222, 208.67.220.220	False
000000-CBTS-TEST-PN	000000-OH-CIN-MX-P01	MX65W	WAN 1	192.168.250.1	False
000000-CBTS-TEST-PN	000000-OH-CIN-MX-P01	MX65W	WAN 2	208.67.222.222, 208.67.220.220	True
000000-CBTS-TEST-PN	000000-CBTS-TEST-SW-P01	MS120-48LP	WAN 1	208.67.222.222, 208.67.220.220	False
004849-TX-COR-PN	004849-TX-COR-MX-P01	MX65W	WAN 1	71.10.216.1, 71.10.216.2	False
004849-TX-COR-PN	004849-TX-COR-MX-P01	MX65W	WAN 2	208.67.222.222, 208.67.220.220	True
004849-TX-COR-PN	004849-TX-COR-SW-P01	MS120-48LP	WAN 1	208.67.222.222, 208.67.220.220	False
000017-KY-LOU-PN	000017-KY-LOU-MX-P01	MX65W	WAN 1	209.18.47.61, 209.18.47.63	False
000017-KY-LOU-PN	000017-KY-LOU-MX-P01	MX65W	WAN 2	208.67.222.222, 208.67.220.220	True
000017-KY-LOU-PN	000017-KY-LOU-SW-P01	MS120-48LP	WAN 1	208.67.222.222, 208.67.220.220	False
000036-KY-LOU-PN	000036-KY-LOU-MX-P01	MX65W	WAN 1	209.18.47.61, 209.18.47.63	False
000036-KY-LOU-PN	000036-KY-LOU-MX-P01	MX65W	WAN 2	208.67.222.222, 208.67.220.220	True
000036-KY-LOU-PN	000036-KY-LOU-SW-P01	MS120-48LP	WAN 1	208.67.222.222, 208.67.220.220	False
004914-SC-ROC-PN	004914-SC-ROC-MX-P01	MX65W	WAN 1	8.8.8.8, 208.104.2.36	False
004914-SC-ROC-PN	004914-SC-ROC-MX-P01	MX65W	WAN 2	208.67.222.222, 208.67.220.220	True
004914-SC-ROC-PN	004914-SC-ROC-SW-P01	MS120-48LP	WAN 1	208.67.222.222, 208.67.220.220	False
004876-CA-RIV-PN	004876-CA-RIV-MX-P01	MX65W	WAN 1	71.10.216.1, 71.10.216.2	False
004876-CA-RIV-PN	004876-CA-RIV-MX-P01	MX65W	WAN 2	208.67.222.222, 208.67.220.220	True
004876-CA-RIV-PN	004876-CA-RIV-SW-P01	MS120-48LP	WAN 1	208.67.222.222, 208.67.220.220	False
000004-KY-LOU-PN	000004-KY-LOU-MX-P01	MX65W	WAN 1	209.18.47.62, 209.18.47.61	False
000004-KY-LOU-PN	000004-KY-LOU-MX-P01	MX65W	WAN 2	208.67.222.222, 208.67.220.220	True
000004-KY-LOU-PN	000004-KY-LOU-MS-P01	MS120-48LP	WAN 1	208.67.222.222, 208.67.220.220	False

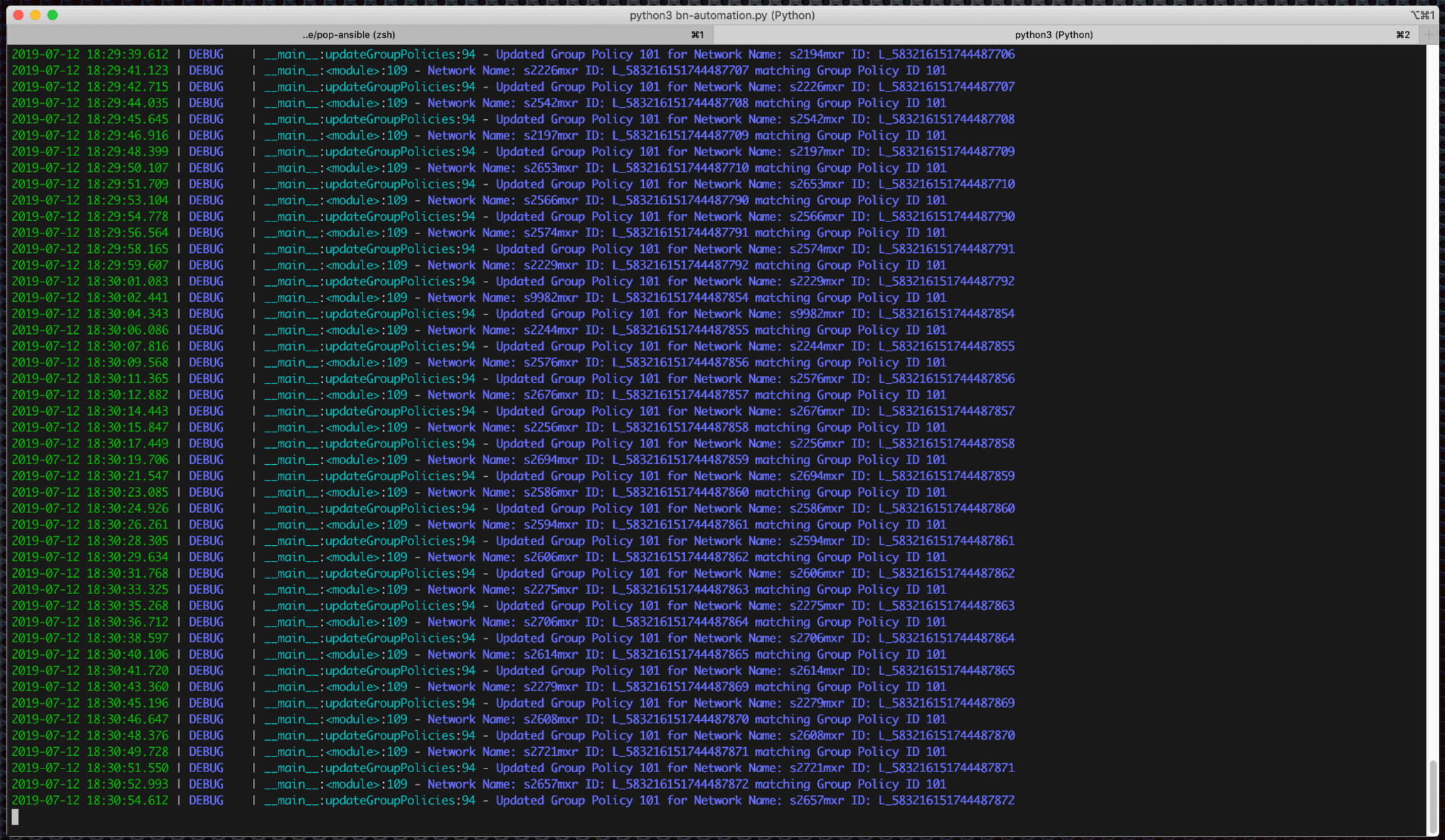
Use Case: Bulk Security Policy Change

craigb@Craigs-Mac-mini: ~/Projects/meraki-tools (zsh)

..ons/repo/code (zsh) ⌘1 | ..ndations/repo (zsh) ⌘2 | ../meraki-tools (zsh) ⌘3 | python3 (Python) ⌘4

```
➜ meraki-tools git:(master) jq . removeCatGroupPolicy.json
{
  "firewallAndTrafficShaping": {
    "settings": "custom",
    "trafficShapingRules": [],
    "l3FirewallRules": [],
    "l7FirewallRules": [
      {
        "policy": "deny",
        "type": "applicationCategory",
        "value": {
          "id": "meraki:layer7/category/8",
          "name": "Peer-to-peer (P2P)"
        }
      },
      {
        "policy": "deny",
        "type": "applicationCategory",
        "value": {
          "id": "meraki:layer7/category/6",
          "name": "Gaming"
        }
      },
      {
        "policy": "deny",
        "type": "applicationCategory",
        "value": {
          "id": "meraki:layer7/category/9",
          "name": "Social web & photo sharing"
        }
      },
      {
        "policy": "deny",
        "type": "applicationCategory",
        "value": {
          "id": "meraki:layer7/category/12",
          "name": "Software & anti-virus updates"
        }
      },
      {
        "policy": "deny",
        "type": "applicationCategory",
```

Use Case: Bulk Security Policy Change



Use Case: New Datacenter Build

- Full stack, compute/storage/FC switching/routing/switching/firewall/management etc.
- Remote site in Halifax, Nova Scotia
- Netbox DCIMs as a source of truth
  - Abstract the config specifics out of your automation code, and into a more universal DCIM tool
  - Interact with Netbox using its API & SDK

Rack C2-2-2-1 (HFX1-H35)

Created March 7, 2019 · Updated 1 week, 6 days ago

Rack	
Site	HEUC Pods > HLFXNS02W02
Group	None
Facility ID	HFX1-H35
Tenant	None
Status	Active
Role	None
Serial Number	—
Asset Tag	—
Devices	16

Dimensions	
Type	4-post cabinet
Width	19 inches
Height	42U (ascending)
Outer Width	—
Outer Depth	—

Tags

No tags assigned

Comments

None

Power Feeds

Panel	Feed	Status	Type	Utilization
HLFXNS02W02 Test Power Panel	C2-2-2-1 A Side	Active	Primary	0%
HLFXNS02W02 Test Power Panel	C2-2-2-1 B Side	Active	Redundant	0%

Images

None

Non-Racked Devices

Name	Role	Type	Parent
C2-2-2-1-PDU A	PDU	Generic Generic HEUC PDU	—
C2-2-2-1-PDU B	PDU	Generic Generic HEUC PDU	—

HEUC-HAL-3850-01

+ Add Components | Edit this device | Delete this device

Created March 26, 2019 · Updated 3 weeks, 6 days ago

Device | Inventory 0 | Status | LLDP Neighbors | Configuration | Config Context | Changelog

Device

Site	HEUC Pods › HLFXNS02W02
Rack	C2-2-2-1 (HFX1-H35)
Position	U28 / Front
Tenant	Customers-E › EHCS-HEUC
Device Type	Cisco WS-C3850-48T (1U)
Serial Number	—
Asset Tag	HEUC-HAL-3850-01

Management

Role	Management
Platform	Cisco IOS
Status	Active
Primary IPv4	172.30.80.1
Primary IPv6	—

Custom Fields

CRM Account Number	—

Tags

No tags assigned

Comments

OOB Mangement switch for HEUC environment

Console / Power

Console	HAL-959	HEUC-HAL-TS-01	Port 1	

+ Add console port | + Add power port

Secrets

None found

+ Add secret

Services

None

+ Assign service

Images

None

+ Attach an image

Related Devices

None found

Interfaces

Show IPs

Name	LAG	Description	MTU	Mode	Cable	Connection	
Gi0/0		OOB Management Port	—	Access	HAL-1177	Meraki-OOB-HAL	LAN 11
Gi1/0/1		—	—	—	HAL-960	HEUC-HAL-TS-01	LAN-1
Gi1/0/2		—	—	—	#1170	HEUC-HAL-ASR-01	Gi0
Gi1/0/3		—	—	—	#1171	HEUC-HAL-ASR-02	Gi0
Gi1/0/4		—	—	—	HAL-1070	HEUC-HAL-FI-A	Mgmt0
Gi1/0/5		—	—	—	HAL-1071	HEUC-HAL-FI-B	Mgmt0

Driving Ansible with Netbox Data

```yaml
- name: Find ID of switch
  uri:
    url: "http://{{netbox_server}}/api/dcim/devices/?site={{ site_name }}&name={{ inventory_hostname }}"
    method: GET
    headers:
      Authorization: "{{ netbox_auth_header }}"
    body_format: json
    return_content: yes
    follow_redirects: yes
  register: json_response
  tags:
    - site_devices

- name: Find switch interfaces
  uri:
    url: "http://{{netbox_server}}/api/dcim/interfaces/?device_id={{ json_response.json.results[0].id }}"
    method: GET
    headers:
      Authorization: "{{ netbox_auth_header }}"
    body_format: json
    return_content: yes
    follow_redirects: yes
  register: json
  tags:
    - site_devices

- name: Label all used interfaces
  ios_interface:
    name: "{{ item.name }}"
    description: "[<- {{ item.connected_endpoint.device.name }} - {{ item.connected_endpoint.name }} ->]"
  loop: "{{ json.json.results }}"
  when: (
          (item.connected_endpoint)
          and
          (item.connected_endpoint.device.name)
          and
          (item.connected_endpoint.name)
        )
  tags: label-interfaces
```

Driving Ansible with Netbox Data

craigb@Craigs-Mac-mini: ~/Projects/pop-ansible (zsh) ⌥⌘1

..ons/repo/code (zsh) ⌘1 | ..n Foundations (zsh) ⌘2 | ..s/pop-ansible (zsh) ⌘3

```
➜ pop-ansible git:(master) ✗ ansible-playbook playbooks/management-switches.yml -t site_devices,label-interfaces

PLAY [3850 Management Switch Configuration] ************************************************************************************

TASK [Find ID of switch] *******************************************************************************************************
 [WARNING]: The value True (type bool) in a string field was converted to 'True' (type string). If this does not look like what you
expect, quote the entire value to ensure it does not change.

ok: [HEUC-HAL-3850-01]

TASK [Find switch interfaces] **************************************************************************************************
ok: [HEUC-HAL-3850-01]

TASK [Label all used interfaces] ***********************************************************************************************
changed: [HEUC-HAL-3850-01] => (item={'id': 11301, 'device': {'id': 491, 'url': 'https://netbox.cbadnms.com/api/dcim/devices/491/', 'name':
 'HEUC-HAL-3850-01', 'display_name': 'HEUC-HAL-3850-01'}, 'name': 'Gi0/0', 'type': {'value': 1000, 'label': '1000BASE-T (1GE)'}, 'form_fact
or': {'value': 1000, 'label': '1000BASE-T (1GE)'}, 'enabled': True, 'lag': None, 'mtu': None, 'mac_address': None, 'mgmt_only': True, 'desc
ription': 'OOB Management Port', 'connected_endpoint_type': 'dcim.interface', 'connected_endpoint': {'id': 11297, 'url': 'https://netbox.cb
adnms.com/api/dcim/interfaces/11297/', 'device': {'id': 483, 'url': 'https://netbox.cbadnms.com/api/dcim/devices/483/', 'name': 'Meraki-OOB
-HAL', 'display_name': 'Meraki-OOB-HAL'}, 'name': 'LAN 11', 'cable': 1177, 'connection_status': {'value': True, 'label': 'Connected'}}, 'co
nnection_status': {'value': True, 'label': 'Connected'}, 'cable': {'id': 1177, 'url': 'https://netbox.cbadnms.com/api/dcim/cables/1177/', '
label': 'HAL-1177'}, 'mode': {'value': 100, 'label': 'Access'}, 'untagged_vlan': None, 'tagged_vlans': [], 'tags': [], 'count_ipaddresses':
 0})
changed: [HEUC-HAL-3850-01] => (item={'id': 9920, 'device': {'id': 491, 'url': 'https://netbox.cbadnms.com/api/dcim/devices/491/', 'name':
'HEUC-HAL-3850-01', 'display_name': 'HEUC-HAL-3850-01'}, 'name': 'Gi1/0/1', 'type': {'value': 1000, 'label': '1000BASE-T (1GE)'}, 'form_fac
tor': {'value': 1000, 'label': '1000BASE-T (1GE)'}, 'enabled': True, 'lag': None, 'mtu': None, 'mac_address': None, 'mgmt_only': False, 'de
scription': '', 'connected_endpoint_type': 'dcim.interface', 'connected_endpoint': {'id': 9918, 'url': 'https://netbox.cbadnms.com/api/dcim
/interfaces/9918/', 'device': {'id': 492, 'url': 'https://netbox.cbadnms.com/api/dcim/devices/492/', 'name': 'HEUC-HAL-TS-01', 'display_nam
e': 'HEUC-HAL-TS-01'}, 'name': 'LAN-1', 'cable': 960, 'connection_status': {'value': True, 'label': 'Connected'}}, 'connection_status': {'v
alue': True, 'label': 'Connected'}, 'cable': {'id': 960, 'url': 'https://netbox.cbadnms.com/api/dcim/cables/960/', 'label': 'HAL-960'}, 'mo
de': None, 'untagged_vlan': None, 'tagged_vlans': [], 'tags': [], 'count_ipaddresses': 0})
changed: [HEUC-HAL-3850-01] => (item={'id': 9921, 'device': {'id': 491, 'url': 'https://netbox.cbadnms.com/api/dcim/devices/491/', 'name':
'HEUC-HAL-3850-01', 'display_name': 'HEUC-HAL-3850-01'}, 'name': 'Gi1/0/2', 'type': {'value': 1000, 'label': '1000BASE-T (1GE)'}, 'form_fac
tor': {'value': 1000, 'label': '1000BASE-T (1GE)'}, 'enabled': True, 'lag': None, 'mtu': None, 'mac_address': None, 'mgmt_only': False, 'de
scription': '', 'connected_endpoint_type': 'dcim.interface', 'connected_endpoint': {'id': 9803, 'url': 'https://netbox.cbadnms.com/api/dcim
/interfaces/9803/', 'device': {'id': 488, 'url': 'https://netbox.cbadnms.com/api/dcim/devices/488/', 'name': 'HEUC-HAL-ASR-01', 'display_na
me': 'HEUC-HAL-ASR-01'}, 'name': 'Gi0', 'cable': 1170, 'connection_status': {'value': True, 'label': 'Connected'}}, 'connection_status': {'
value': True, 'label': 'Connected'}, 'cable': {'id': 1170, 'url': 'https://netbox.cbadnms.com/api/dcim/cables/1170/', 'label': ''}, 'mode':
 None, 'untagged_vlan': None, 'tagged_vlans': [], 'tags': [], 'count_ipaddresses': 0})
changed: [HEUC-HAL-3850-01] => (item={'id': 9922, 'device': {'id': 491, 'url': 'https://netbox.cbadnms.com/api/dcim/devices/491/', 'name':
'HEUC-HAL-3850-01', 'display_name': 'HEUC-HAL-3850-01'}, 'name': 'Gi1/0/3', 'type': {'value': 1000, 'label': '1000BASE-T (1GE)'}, 'form_fac
tor': {'value': 1000, 'label': '1000BASE-T (1GE)'}, 'enabled': True, 'lag': None, 'mtu': None, 'mac_address': None, 'mgmt_only': False, 'de
scription': '', 'connected_endpoint_type': 'dcim.interface', 'connected_endpoint': {'id': 9786, 'url': 'https://netbox.cbadnms.com/api/dcim
```

Resulting Switch Config

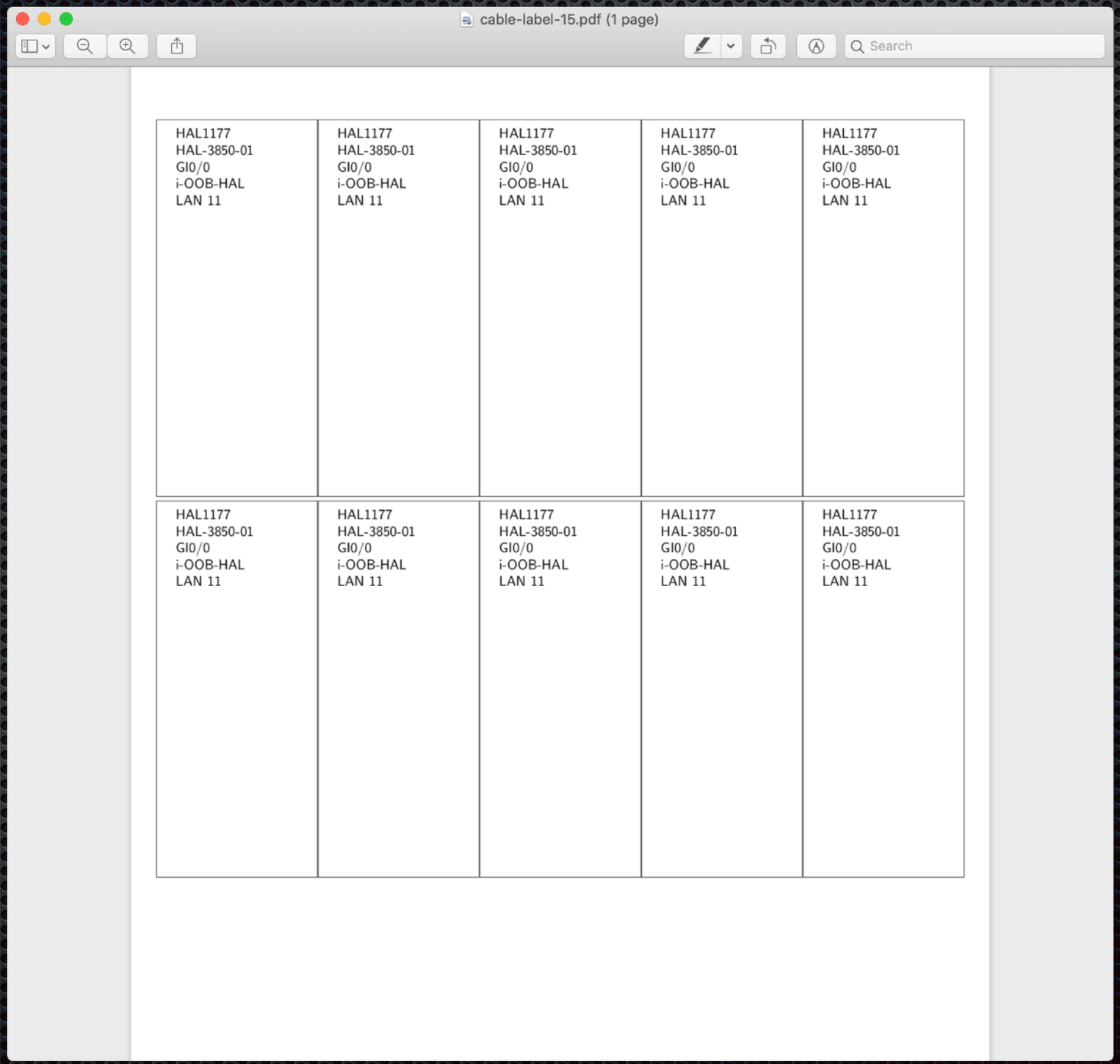
HAL-3850

```
Interface                      Status         Protocol Description
Vl1                            admin down     down
Vl11                           up             up
Gi0/0                          up             up       [<- Meraki-OOB-HAL - LAN 11 ->]
Gi1/0/1                        up             up       [<- HEUC-HAL-TS-01 - LAN-1 ->]
Gi1/0/2                        up             up       [<- HEUC-HAL-ASR-01 - Gi0 ->]
Gi1/0/3                        up             up       [<- HEUC-HAL-ASR-02 - Gi0 ->]
Gi1/0/4                        up             up       [<- HEUC-HAL-FI-A - Mgmt0 ->]
Gi1/0/5                        up             up       [<- HEUC-HAL-FI-B - Mgmt0 ->]
Gi1/0/6                        up             up       [<- HEUC-HAL-MDS-A - Mgmt0 ->]
Gi1/0/7                        up             up       [<- HEUC-HAL-MDS-B - Mgmt0 ->]
Gi1/0/8                        up             up       [<- HEUC-HAL-ASA4110-01 - Mgmt0 ->]
Gi1/0/9                        up             up       [<- HEUC-HAL-ASA4110-02 - Mgmt0 ->]
Gi1/0/10                       up             up       [<- HEUC-HAL-N9K-01 - Mgmt0 ->]
Gi1/0/11                       up             up       [<- HEUC-HAL-N9K-02 - Mgmt0 ->]
Gi1/0/12                       down           down
Gi1/0/13                       up             up       [<- HEUC-HAL-ASA4110-01 - Eth1/8 ->]
Gi1/0/14                       up             up       [<- HEUC-HAL-ASA4110-02 - Eth1/8 ->]
Gi1/0/15                       down           down
```

Automatic Label Generation with Netbox Data

```python
#!/usr/local/bin/python3

import requests
import json
import csv

device_url = "https://netbox.cbadnms.com/api/dcim/interfaces/?device_id=491"

payload = ""
headers = {
    'Authorization': "Token ",
    'Content-Type': "application/json",
    'Accept': "application/json",
    'cache-control': "no-cache",
    'Postman-Token': "e46ffd75-f6f8-451c-a77f-d1d558968dd2"
    }

response = requests.request("GET", device_url, data=payload, headers=headers)
device_js = json.loads(str(response.text))

def getCables(device_js):
    with open('cable-data.txt', mode="w") as f:
        for interface in device_js['results']:
            if interface['cable']:
                print("Found cable " + str(interface['cable']['id']))
                cable_url = (interface['cable']['url'])
                response = requests.request("GET", cable_url, data=payload, headers=headers)
                cable = json.loads(str(response.text))
                f.write("HAL-" + str(cable['id']) + '\n')
                f.write(cable['termination_a']['device']['name'][5:] + '\n')
                f.write(cable['termination_a']['name'].upper() + '\n')
                f.write(cable['termination_b']['device']['name'][5:] + '\n')
                f.write(cable['termination_b']['name'].upper() + '\n')
                f.write('\n')

def makeCSV(cables):
    with open('cable-matrix.csv', mode='w') as f:
        fieldnames = ['Cable ID', 'Device A', 'Interface A', 'Device Z', 'Interface Z']
        writer = csv.writer(f)
        writer.writerows(cables)

getCables(device_js)
#makeCSV(device_js)
```

Resulting Patch Cable Labels



Advanced Example: Velocloud SD-WAN on AWS Provisioning using API and Terraform

- YouTube

Please, No More Tools!

- This automation journey has an rapidly evolving, seemingly infinite toolchain
- Postman tool to interact with Web Service APIs
- Continuous Integration / Continuous Deployment (CI/CD) Tools
  - Travis, Jenkins, etc.
- Other Infrastructure as Code (IAC) Tools
  - Terraform, Cobbler, Chef, Puppet
- Python Libraries
  - Napalm, Paramiko, Netmiko
- Loads of SDKs by OEMs and community
- Terminals, shells, and add-ons

Do You Even Tool, Bro?

Thoughts on DevOps

- “DevOps is a cultural and professional movement, focused on how we build and operate high velocity organizations, born from the experiences of its practitioners” - Adam Jacob, CTO Chef
- There is no “The Business” - We’re all The Business
- How can these tools and strategies change my job and affect my business processes?

Where to Start?

- Evaluate your environment, and which components have mature tooling (APIs, SDKs, Ansible Modules, etc)
- Preferably, pick something you have domain knowledge of
- “To err is human; to propagate errors massively at scale is automation.”
  - Read operations —> Config Generation —> Direct Config Manipulation
  - Education / Community / Consulting & Professional Services
  - Some low fruit
    - Auditing / Reporting
    - Labeling
    - Greenfield deployments, perhaps L2 switching

Resources

- Courses
  - Charles Severance Python for Everybody
  - realpython.com
  - A Cloud Guru - Git, Ansible, Chef, AWS, Azure…
    - https://www.tutorialspoint.com/ansible
- Books
  - Network Programmability and Automation: Skills for the Next-Generation Network Engineer
- Community
  - IRC Freenode #ansible, #ansible-network
  - Packet Pushers - Datanauts 166

Feedback

- Understandable?
- Useful?
- Labs?
- Less of…
- More of…

Future Topics

- Ansible 201
- Python 201
- Git 201
- Terraform
- Specific SDKs
- AWS networking
- Container networking
- Serverless
- CI/CD

Thanks for coming, I'll be here all week, be sure to tip the waitress