BCNET 2019

Network Automation

Ansible, python, Juniper PyEz

Brett Runnings, BCIT

Challenges

- Large number of devices to keep track of
- Devices being added/removed from the network
- Configuration drift
- Efficient use of time
- How to collect operational data and be proactive

Solutions

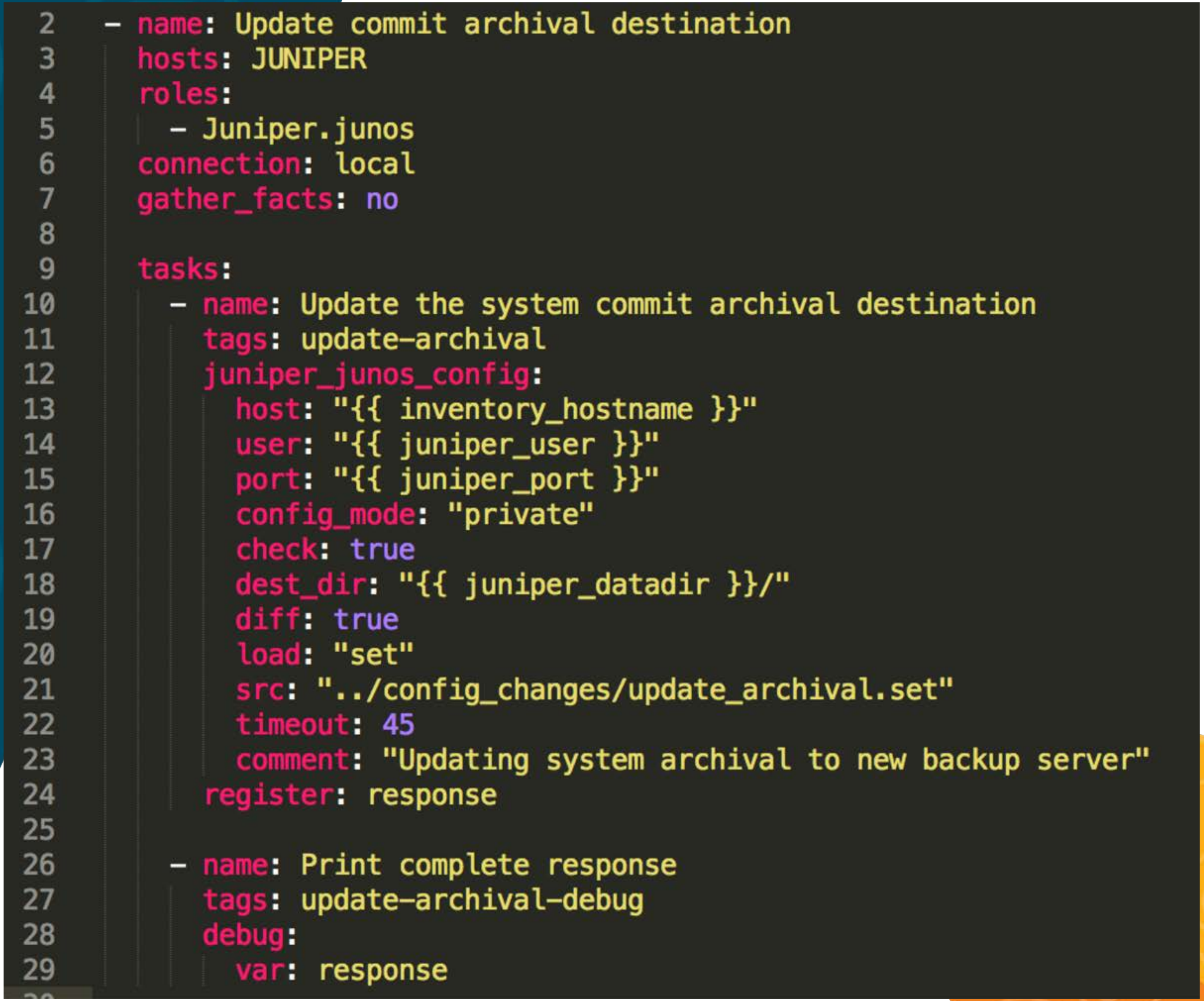
- Python
- Ansible
- Junos PyEZ
- Jinja2 (templating)
- Netconf
- NAPALM (multi-vendor)

Ansible

- Tool to manage network devices
- Automatically build inventories of your devices
- Collect operational information
- Preform configuration changes
- Deploy templated configurations

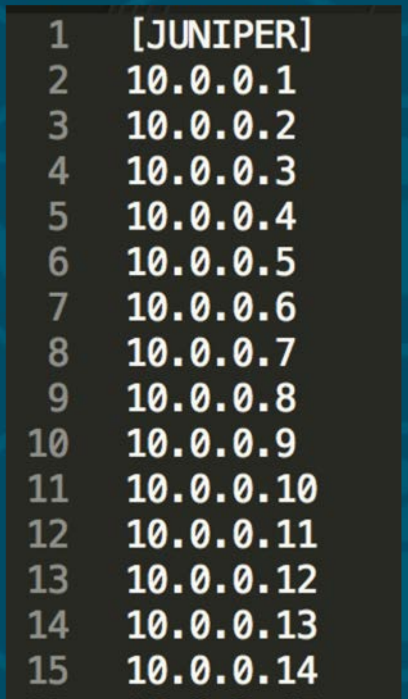
Ansible playbook

- Written in YAML
- Variables in braces “ {{   }}”
- Combine with an inventory, and some configuration commands
- Can be run in “test” mode



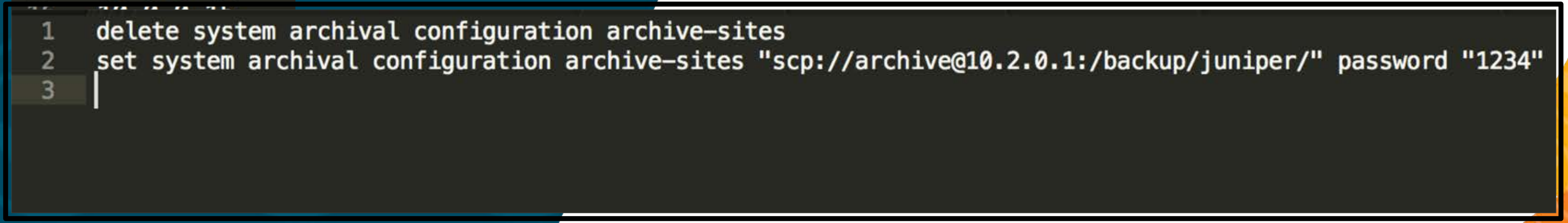
```
- name: Update commit archival destination
  hosts: JUNIPER
  roles:
    - Juniper.junos
  connection: local
  gather_facts: no

  tasks:
    - name: Update the system commit archival destination
      tags: update-archival
      juniper_junos_config:
        host: "{{ inventory_hostname }}"
        user: "{{ juniper_user }}"
        port: "{{ juniper_port }}"
        config_mode: "private"
        check: true
        dest_dir: "{{ juniper_datadir }}/"
        diff: true
        load: "set"
        src: "../config_changes/update_archival.set"
        timeout: 45
        comment: "Updating system archival to new backup server"
      register: response

    - name: Print complete response
      tags: update-archival-debug
      debug:
        var: response
```

Inventory File



```
[JUNIPER]
10.0.0.1
10.0.0.2
10.0.0.3
10.0.0.4
10.0.0.5
10.0.0.6
10.0.0.7
10.0.0.8
10.0.0.9
10.0.0.10
10.0.0.11
10.0.0.12
10.0.0.13
10.0.0.14
```

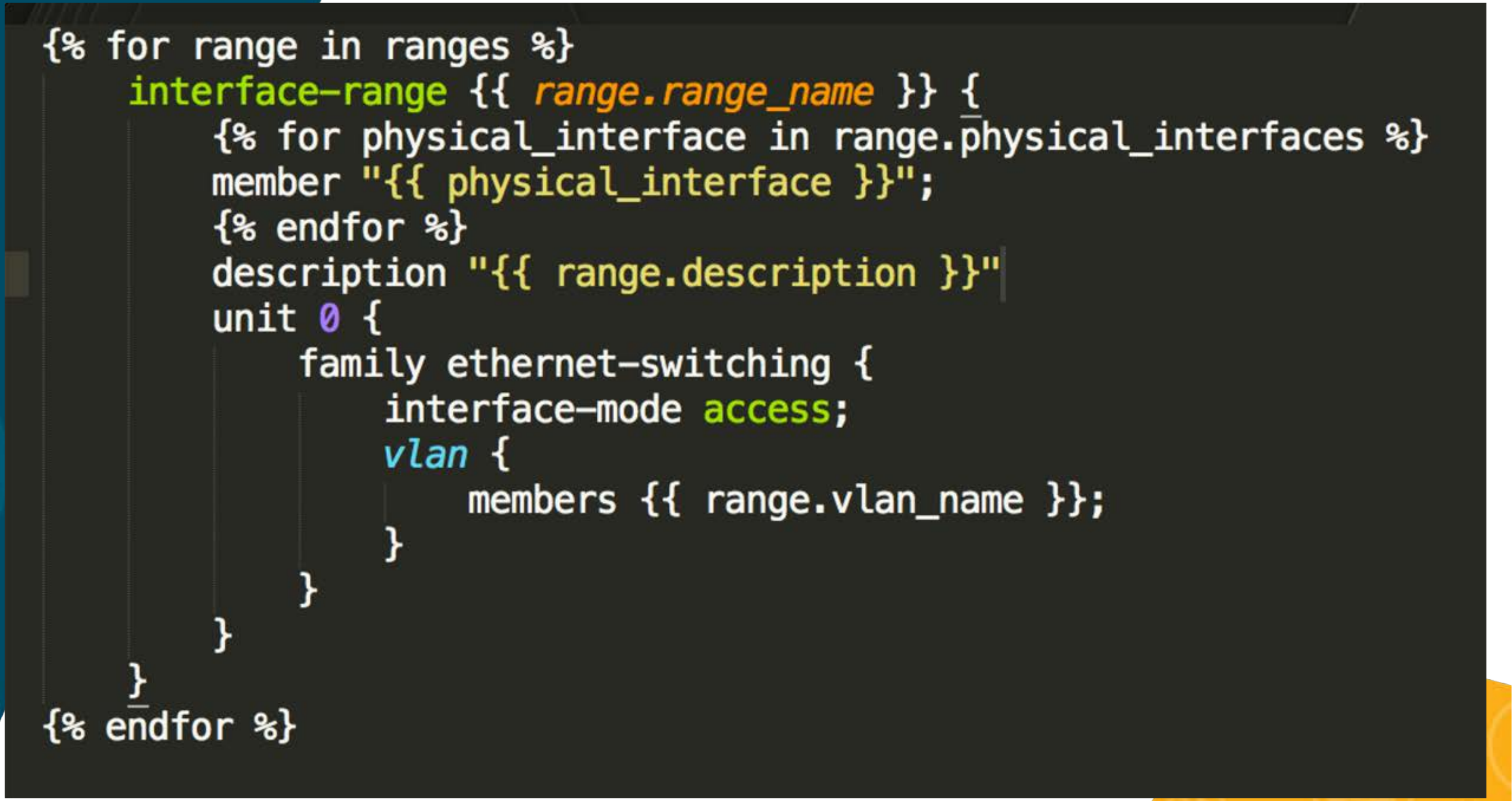
- Can be much more complex
- Configuration module supports many options
    - Merge, override, set
    - replace
    - Commit confirmed
    - Rollbacks
- Can leverage forking to run many connections in parallel



```
delete system archival configuration archive-sites
set system archival configuration archive-sites "scp://archive@10.2.0.1:/backup/juniper/" password "1234"

```

Jinja2 templates

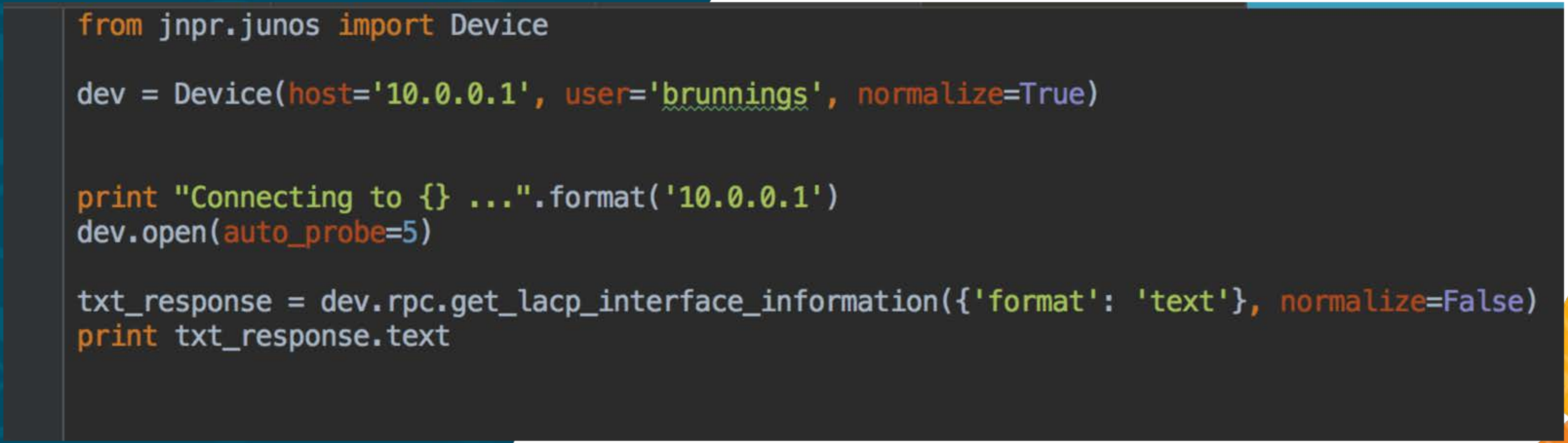
- Create your templates
- In your ansible playbook, supply the required variables.
- Result is consistent configurations
- Ansible & Jinja2 both support filters
  - Text manipulation
  - Regular Expressions
  - XML/JSON searching
  - Math



```
{% for range in ranges %}
    interface-range {{ range.range_name }} {
        {% for physical_interface in range.physical_interfaces %}
        member "{{ physical_interface }}";
        {% endfor %}
        description "{{ range.description }}"
        unit 0 {
            family ethernet-switching {
                interface-mode access;
                vlan {
                    members {{ range.vlan_name }};
                }
            }
        }
    }
{% endfor %}
```

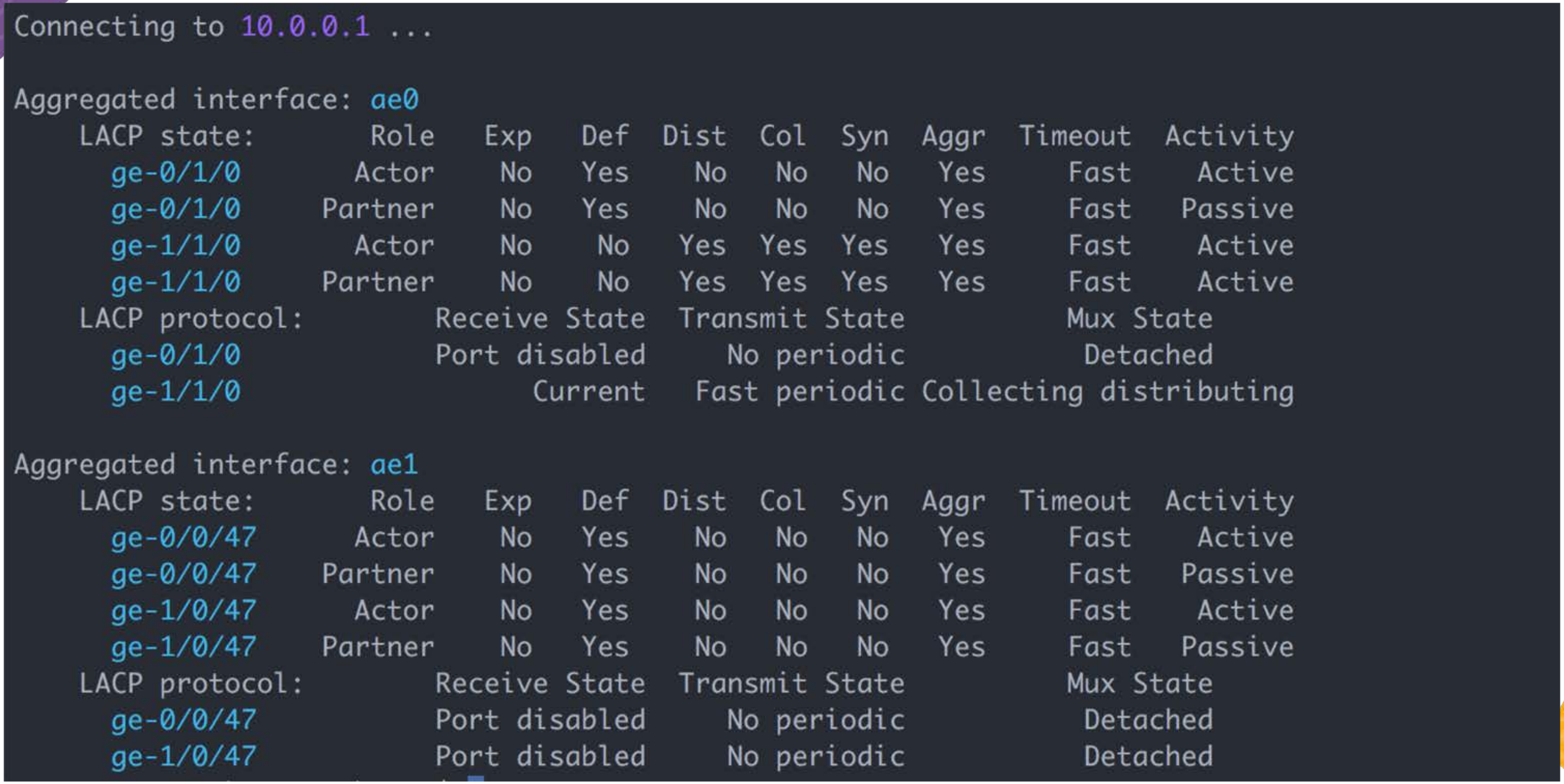
PyEZ

- Allows more granularity than some of the ansible modules
- Response available in different formats, XML, TXT, JSON
- Supports operational and configuration commands
- Many good learning resources available



```
from jnpr.junos import Device

dev = Device(host='10.0.0.1', user='brunnings', normalize=True)


print "Connecting to {} ...".format('10.0.0.1')
dev.open(auto_probe=5)

txt_response = dev.rpc.get_lacp_interface_information({'format': 'text'}, normalize=False)
print txt_response.text
```

Results of the previous example…



```
Connecting to 10.0.0.1 ...

Aggregated interface: ae0
    LACP state:       Role   Exp   Def  Dist  Col  Syn  Aggr  Timeout  Activity
      ge-0/1/0       Actor    No   Yes    No   No   No   Yes     Fast    Active
      ge-0/1/0     Partner    No   Yes    No   No   No   Yes     Fast   Passive
      ge-1/1/0       Actor    No    No   Yes  Yes  Yes   Yes     Fast    Active
      ge-1/1/0     Partner    No    No   Yes  Yes  Yes   Yes     Fast    Active
    LACP protocol:        Receive State  Transmit State          Mux State
      ge-0/1/0            Port disabled     No periodic           Detached
      ge-1/1/0                  Current   Fast periodic Collecting distributing

Aggregated interface: ae1
    LACP state:       Role   Exp   Def  Dist  Col  Syn  Aggr  Timeout  Activity
      ge-0/0/47      Actor    No   Yes    No   No   No   Yes     Fast    Active
      ge-0/0/47    Partner    No   Yes    No   No   No   Yes     Fast   Passive
      ge-1/0/47      Actor    No   Yes    No   No   No   Yes     Fast    Active
      ge-1/0/47    Partner    No   Yes    No   No   No   Yes     Fast   Passive
    LACP protocol:        Receive State  Transmit State          Mux State
      ge-0/0/47           Port disabled     No periodic           Detached
      ge-1/0/47           Port disabled     No periodic           Detached
```