OpenConfig with NAPALM

David Barroso <dbarrosop@dravetech.com>

1. What is YANG?
2. What is OpenConfig?
3. What is napalm-yang?

1. **What is YANG?**
2. What is OpenConfig?
3. What is napalm-yang?

What is YANG?

1. A Data Modeling Language for the Network Configuration Protocol (NETCONF)
2. RFC6020

What is not YANG?

1. An API
2. Developed exclusively for NETCONF/gRPC
3. JSON or XML

⚠ Not a YANG tutorial so forgive the handwaving

YANG vs JSON vs XML

person.yang

```
module person {
  prefix "person";
  namespace "http://test.local/person";

  container Person {
    leaf name {
      description "Person's name";
      type string;
    }
    leaf age {
      description "Person's age";
      type uint16;
    }
  }
}
```

person.json

```
{
  "Person": {
    "name": "John",
    "age": 42
  }
}
```

person.xml

```
<person>
    <name>John</name>
    <age>42</age>
</person>
```

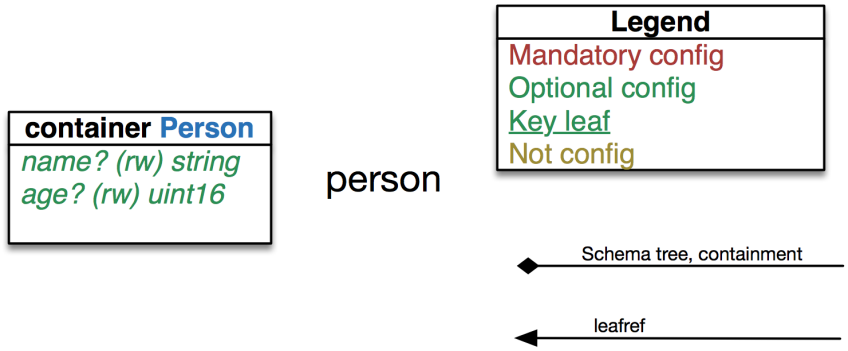
Why do I care?

generate documentation

```
$ pyang -f tree person.yang
module: person
    +--rw Person
       +--rw name?   string
       +--rw age?    uint16
```

generate diagrams

```
$ pyang -f omni person.yang > person.scpt
$ osascript person.scpt
```



Why do I care?

generate code

```
$ export PYBINDPLUGIN=`/usr/bin/env python -c \
 'import pyangbind; import os; print "%s/plugin" % os.path.dirname(pyangbind.__file__)'`
$ pyang --plugindir $PYBINDPLUGIN -f pybind person.yang > person.py
```

person.py (excerpt)

```
class yc_Person_person__Person(PybindBase):
  _yang_name = 'Person'

  def __init__(self, *args, **kwargs):
    self.__age = YANGDynClass(base=RestrictedClassType(base_type=int, restriction_dict={
    self.__name = YANGDynClass(base=unicode, is_leaf=True, yang_name="name", parent=self

  def _get_name(self):
  def _set_name(self, v, load=False):
```

Why do I care?

```
>>> import person
>>> import pyangbind.lib.pybindJSON as pybindJSON
>>>
>>> data = {'Person': {'age': 40, 'name': 'Jane'}}
>>> p1 = pybindJSON.loads(data, person, 'person')
>>> p1.Person.name, p2.Person.age
(u'Jane', 40)
>>>
>>> p2 = person.person()
>>> p2.Person.name = "John"
>>> p2.Person.age = "40"  # <---- it's letter O, not number 0
Traceback (most recent call last):
  File "person.py", line 134, in _set_age
    'generated-type': """YANGDynClass(base=RestrictedClassType(base_type=int, restri
ValueError: {'error-string': 'age must be of a type compatible with uint16', 'genera
>>> p2.Person.age = "40"
>>> p2.get()
{'Person': {'age': 40, 'name': u'John'}}
```

Why do I care?

```
>>> import person
>>> import pyangbind.lib.pybindJSON as pybindJSON
>>>
>>> data = {'Person': {'age': 40, 'name': 'Jane'}}
>>> p1 = pybindJSON.loads(data, person, 'person')
>>> p1.Person.name, p2.Person.age
(u'Jane', 40)
>>>
>>> p2 = person.person()
>>> p2.Person.name = "John"
>>> p2.Person.age = "40"  # <---- it's letter O, not number 0
Traceback (most recent call last):
  File "person.py", line 134, in _set_age
    'generated-type': """YANGDynClass(base=RestrictedClassType(base_type=int, restri
ValueError: {'error-string': 'age must be of a type compatible with uint16', 'genera
>>> p2.Person.age = "40"
>>> p2.get()
{'Person': {'age': 40, 'name': u'John'}}
```

Why do I care?

```
>>> import person
>>> import pyangbind.lib.pybindJSON as pybindJSON
>>>
>>> data = {'Person': {'age': 40, 'name': 'Jane'}}
>>> p1 = pybindJSON.loads(data, person, 'person')
>>> p1.Person.name, p2.Person.age
(u'Jane', 40)
>>>
>>> p2 = person.person()
>>> p2.Person.name = "John"
>>> p2.Person.age = "40"  # <---- it's letter O, not number 0
Traceback (most recent call last):
  File "person.py", line 134, in _set_age
    'generated-type': """YANGDynClass(base=RestrictedClassType(base_type=int, restri
ValueError: {'error-string': 'age must be of a type compatible with uint16', 'genera
>>> p2.Person.age = "40"
>>> p2.get()
{'Person': {'age': 40, 'name': u'John'}}
```

Why do I care?

```
>>> import person
>>> import pyangbind.lib.pybindJSON as pybindJSON
>>>
>>> data = {'Person': {'age': 40, 'name': 'Jane'}}
>>> p1 = pybindJSON.loads(data, person, 'person')
>>> p1.Person.name, p2.Person.age
(u'Jane', 40)
>>>
>>> p2 = person.person()
>>> p2.Person.name = "John"
>>> p2.Person.age = "40"  # <---- it's letter O, not number 0
Traceback (most recent call last):
  File "person.py", line 134, in _set_age
    'generated-type': """YANGDynClass(base=RestrictedClassType(base_type=int, restri
ValueError: {'error-string': 'age must be of a type compatible with uint16', 'genera
>>> p2.Person.age = "40"
>>> p2.get()
{'Person': {'age': 40, 'name': u'John'}}
```

Extensibility (augmenting the model)

Augmentations are part of the “standard” model

person-extension.yang

```
module person-extension {
  import person { prefix person; }

  identity GENDER {
    description "Gender p identifies with";
  }
  identity MAN {
    base GENDER;
  }
  identity WOMAN {
    base GENDER;
  }
  identity OTHER {
    base GENDER;
  }
```

person-extension.yang (cont’d)

```
  grouping person-extended {
    leaf gender {
      type identityref {
        base GENDER;
      }
    }
  }
  augment "/person:Person" {
    uses person-extended;
  }
}
```

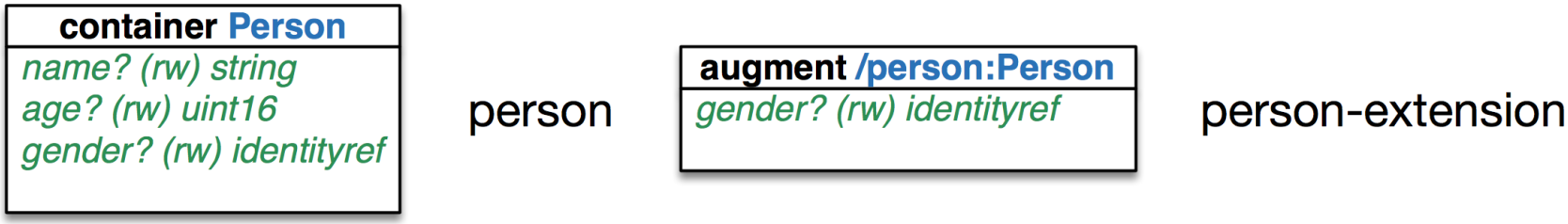
Extensibility (augmenting the model)

generate documentation

```
$ pyang -f tree person-extension.yang person.yang
module: person
    +--rw Person
       +--rw name?                          string
       +--rw age?                           uint16
       +--rw person-extension:gender?       enumeration
```

generate diagrams

```
$ pyang -f omni person-extended.yang person.yang > person-extended.scpt
$ osascript person-extended.scpt
```



Extensibility (deviating the model)

Deviations are failures in model implementation

person-deviation.yang

```
module person-deviation {
  deviation "/person:Person/person:age" {
    deviate not-supported;
  }
}
```

generate documentation

```
$ pyang -f tree person-deviation.yang person.yang
module: person
    +--rw Person
       +--rw name?                          string
                                                  # no age
```

I'm Outta Here

No, Seriously

Most people won’t really care but having YANG models means you have one single source of truth to generate documentation, code and validate data. You might not consume YANG models yourself to do those things but you might indirectly via your NMS or tools.

1. What is YANG?
2. **What is OpenConfig?**
3. What is napalm-yang?

What is OpenConfig?

1. OpenConfig is an industry effort policed by Google to create vendor agnostic models
2. **A bunch of models:**
   repo: https://github.com/openconfig/public
   docs: http://ops.openconfig.net/branches/master/

What is not OpenConfig?

1. Very open despite of what the name might imply :P

Why OpenConfig

OpenConfig provides a set of models you can use to configure and to validate the state of any device regardless of the vendor or OS.

The sad truth

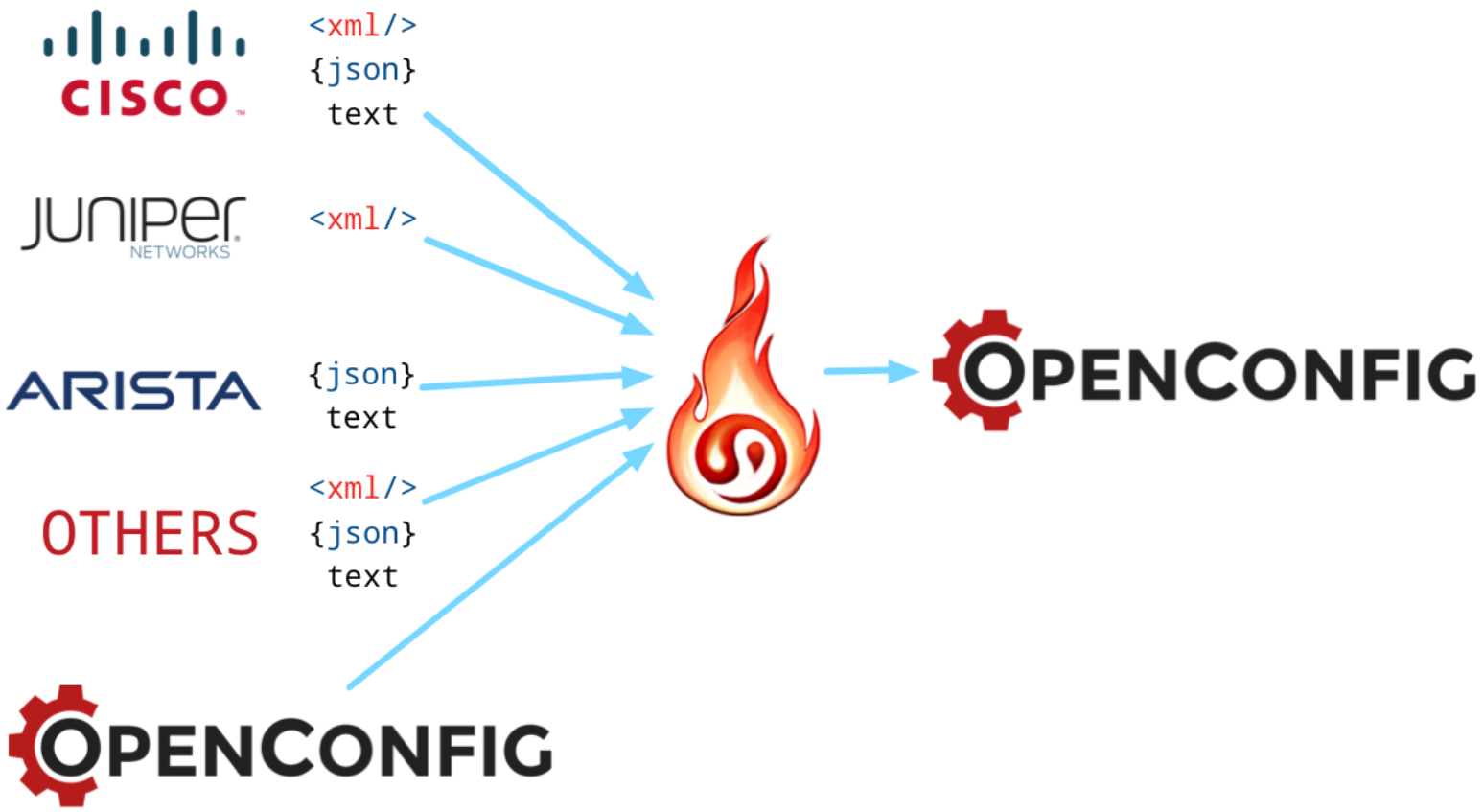
1. Very few vendors care (or care very little)
2. Those who care implement a few models only in their very latest releases
3. Vendors implement almost as many deviations as models
4. Current implementations don’t seem very stable

1. What is YANG?
2. What is OpenConfig?
3. **What is napalm-yang?**
    1. **Basics**
    2. Advanced features
    3. Integration with `ansible`

What is napalm-yang?

A library that provides a framework to map native configuration/state to OpenConfig and vice versa

Parsing native data (1)



Parsing native data (2)

parse_example_1.py

```
1  def main():
2      ios_driver = get_network_driver("ios")
3      with ios_driver(**configuration.ios) as device:
4          r = device.yang.get_openconfig_interfaces()
5          pretty_print(r)
6
7      junos_driver = get_network_driver("junos")
8      with junos_driver(**configuration.junos) as device:
9          r = device.yang.get_openconfig_interfaces()
10         pretty_print(r)
```

Parsing native data (3)

parse_example_1.py

```
1  def main():
2      ios_driver = get_network_driver("ios")
3      with ios_driver(**configuration.ios) as device:
4          r = device.yang.get_openconfig_interfaces()
5          pretty_print(r)
6
7      junos_driver = get_network_driver("junos")
8      with junos_driver(**configuration.junos) as device:
9          r = device.yang.get_openconfig_interfaces()
10         pretty_print(r)
```

ios-native

```
1  interface GigabitEthernet1
2   ip address dhcp
3   negotiation auto
4   no mop enabled
5  !
6  interface GigabitEthernet2
7   no ip address
8   shutdown
9   negotiation auto
10 !
```

ios-openconfig

```
1  {
2      "interfaces": {
3          "interface": {
4              "GigabitEthernet1": {
5                  "name": "GigabitEthernet1",
6                  "routed-vlan": {
7                      "ipv4": {
8                          "config": {
9                              "enabled": true
10                         }
11                     }
12                 },
13                 "config": {
14                     "type": "ethernetCsmacd",
15                     "enabled": true,
16                     "mtu": 1500
17                 }
18             },
19             "GigabitEthernet2": {
20                 "name": "GigabitEthernet2",
21                 "routed-vlan": {
22                     "ipv4": {
23                         "config": {
24                             "enabled": true
25                         }
26                     }
27                 },
28                 "config": {
```

Parsing native data (4)

parse_example_1.py

```python
def main():
    ios_driver = get_network_driver("ios")
    with ios_driver(**configuration.ios) as device:
        r = device.yang.get_openconfig_interfaces()
        pretty_print(r)

    junos_driver = get_network_driver("junos")
    with junos_driver(**configuration.junos) as device:
        r = device.yang.get_openconfig_interfaces()
        pretty_print(r)
```

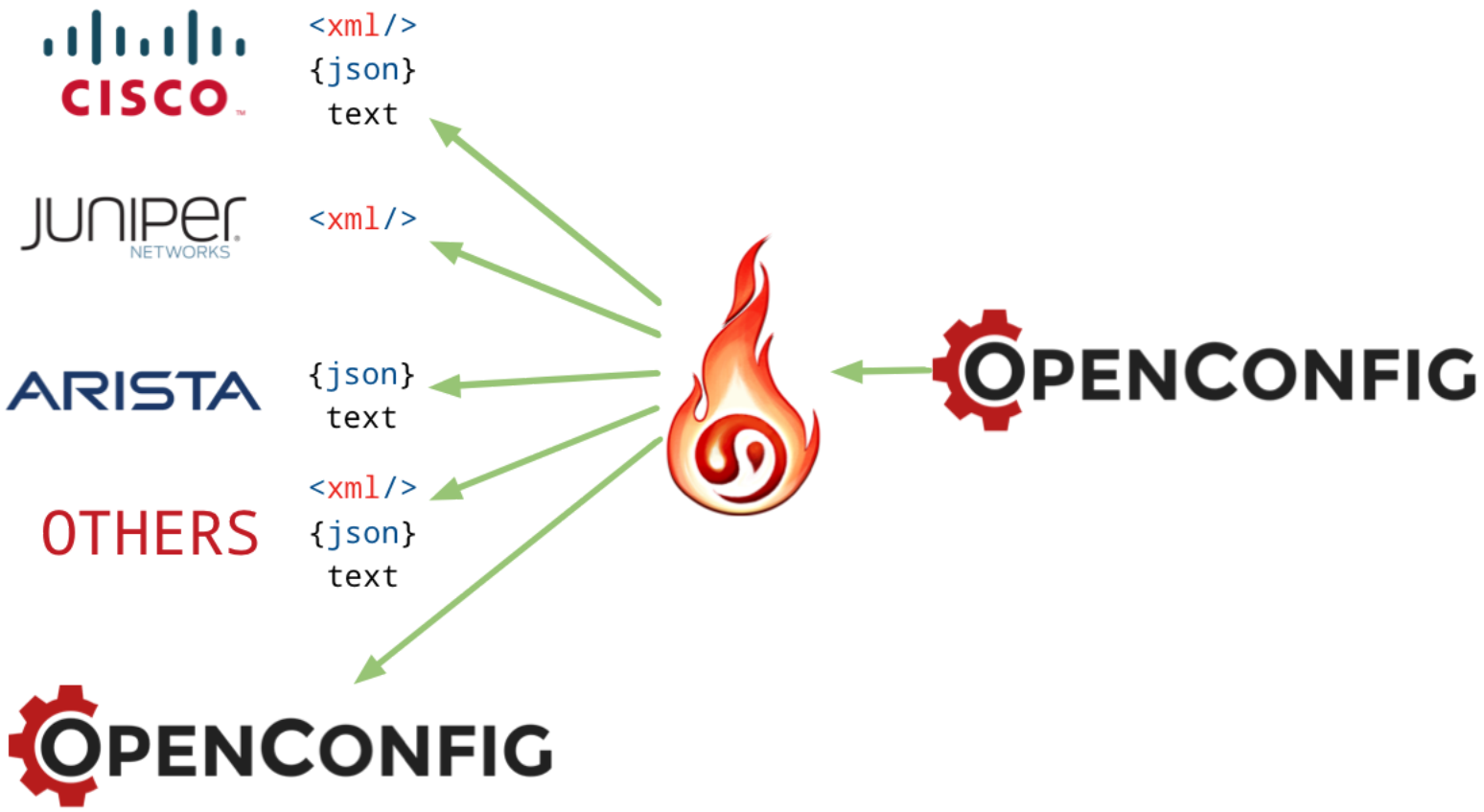
junos-native

```
ge-0/0/0 {
    unit 0 {
        family inet {
            dhcp;
        }
    }
}
```

junos-openconfig

```
{
    "interfaces": {
        "interface": {
            "ge-0/0/0": {
                "name": "ge-0/0/0",
                "subinterfaces": {
                    "subinterface": {
                        "0": {
                            "index": "0",
                            "ipv4": {
                                "config": {
                                    "enabled": true
                                }
                            },
                            "config": {
                                "enabled": true,
                                "name": "0"
                            }
                        }
                    }
                },
                "routed-vlan": {
                    "ipv4": {
                        "config": {
                            "enabled": false
                        }
                    }
                },
```

Translating to native data



Translating to native data

translate_example_1.py

```
def main():
    with open("translate_example_1.data.json", 'r') as f:
        data = json.loads(f.read())

    ios_driver = get_network_driver("ios")
    with ios_driver(**configuration.ios) as device:
        device.yang.candidate.load_dict(data)
        r = device.yang.translate()
        print(r)

    junos_driver = get_network_driver("junos")
    with junos_driver(**configuration.junos) as device:
        device.yang.candidate.load_dict(data)
        r = device.yang.translate()
        print(r)
```

Translating to native data

translate_example_1.py

```python
def main():
    with open("translate_example_1.data.json", 'r') as f:
        data = json.loads(f.read())

    ios_driver = get_network_driver("ios")
    with ios_driver(**configuration.ios) as device:
        device.yang.candidate.load_dict(data)
        r = device.yang.translate()
        print(r)

    junos_driver = get_network_driver("junos")
    with junos_driver(**configuration.junos) as device:
        device.yang.candidate.load_dict(data)
        r = device.yang.translate()
        print(r)
```

translate_example_1.data.json

```json
"eth1": {
    "name": "eth1",
    "routed-vlan": {
        "ipv4": {
            "config": {
                "enabled": true
            },
            "addresses": {
                "address": {
                    "192.168.1.1/24": {
                        "ip": "192.168.1.1/24",
                        "config": {
                            "ip": "192.168.1.1",
                            "prefix-length": 24
                        }
                    }
                }
            }
        }
    },
    "config": {
        "type": "ethernetCsmacd",
        "enabled": false,
        "description": "my description",
        "mtu": 9000
    }
}
```

Translating to native data

translate_example_1.py

```
1  def main():
2      with open("translate_example_1.data.json", 'r') as f:
3          data = json.loads(f.read())
4
5      ios_driver = get_network_driver("ios")
6      with ios_driver(**configuration.ios) as device:
7          device.yang.candidate.load_dict(data)
8          r = device.yang.translate()
9          print(r)
10
11     junos_driver = get_network_driver("junos")
12     with junos_driver(**configuration.junos) as device:
13         device.yang.candidate.load_dict(data)
14         r = device.yang.translate()
15         print(r)
```

ios

```
1  interface eth1
2      no switchport
3      ip address 192.168.1.1 255.255.255.0
4      shutdown
5      description my description
6      mtu 9000
7      exit
```

Translating to native data

translate_example_1.py

```
def main():
    with open("translate_example_1.data.json", 'r') as f:
        data = json.loads(f.read())

    ios_driver = get_network_driver("ios")
    with ios_driver(**configuration.ios) as device:
        device.yang.candidate.load_dict(data)
        r = device.yang.translate()
        print(r)

    junos_driver = get_network_driver("junos")
    with junos_driver(**configuration.junos) as device:
        device.yang.candidate.load_dict(data)
        r = device.yang.translate()
        print(r)
```

junos

```
<configuration>
  <interfaces>
    <interface>
      <name>eth1</name>
      <family>
        <inet>
          <address>
            <name>192.168.1.1/24</name>
          </address>
        </inet>
      </family>
      <disable/>
      <description>my description</description>
      <mtu>9000</mtu>
    </interface>
  </interfaces>
</configuration>
```

Profiles/Mappings

A mapping is a set of rules to map native data to a YANG model and vice versa

- A mappings maps to a specific YANG model and follows its structure
- There is a mapping for each YANG model and supported NOS pair

A profile is a set of mappings for a specific NOS

- A NOS might have different associated profiles.
- For example, `['nos15', 'nos']`

Example (parser)

mappings/ios/parsers/config/openconfig-interfaces/interfaces.yaml

```
---
metadata:
    processor: TextTree
    execute:
        - method: cli
          kwargs:
              commands: ["show running-config all"]
interfaces:
    _process: unnecessary
    interface:
        _process:
            - path: interface
              regexp: "^(?P<value>(\\w|-)*\\d+(\\/\\d+)*)$"
              from: root_interfaces.0
        config:
            _process: unnecessary
            enabled:
                _process:
                    - path: "shutdown"
                      present: no
            description:
                _process:
                    - path: description
            mtu:
                _process:
                    - path: ip.mtu
```

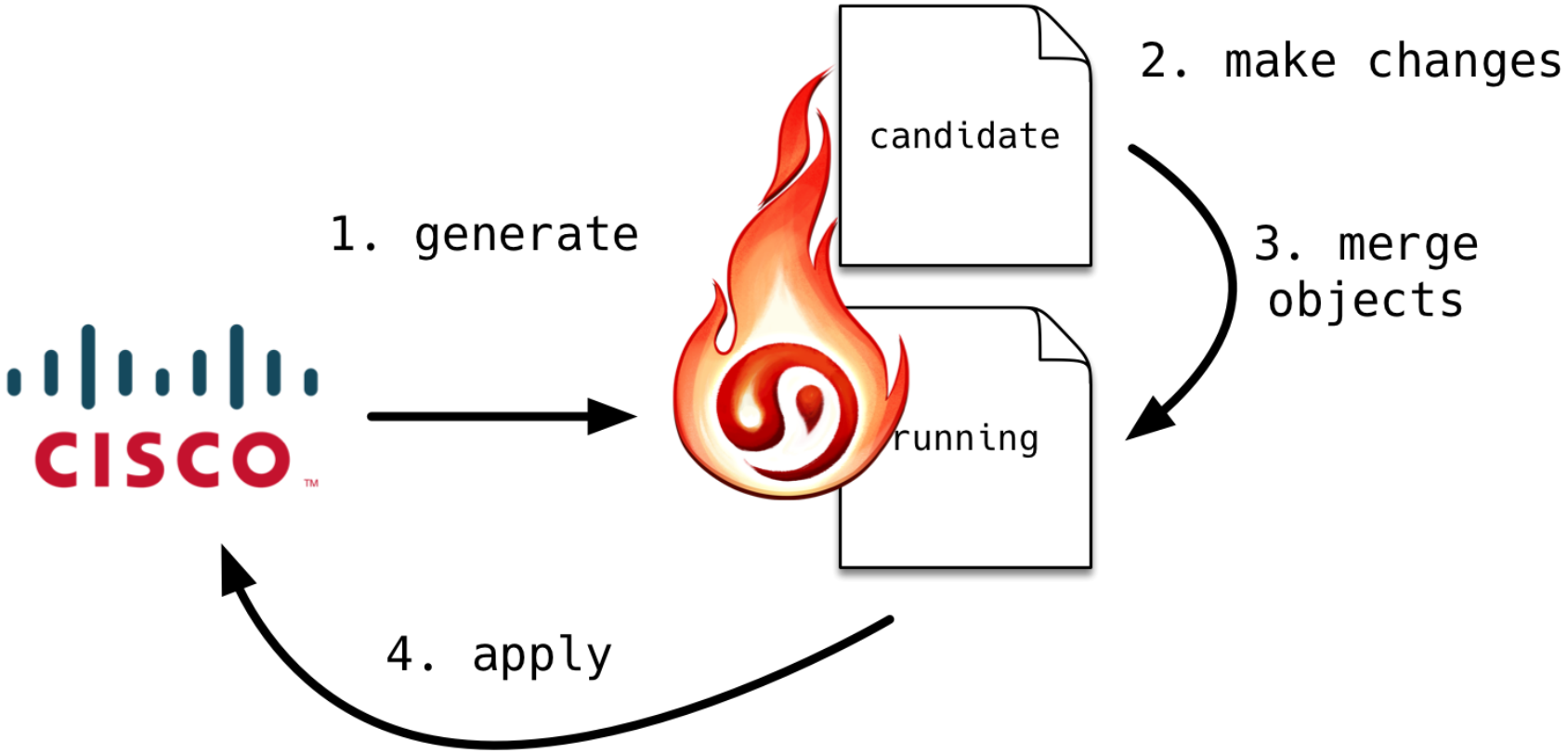
Example (translator)

napalm_yang/mappings/ios/translators/openconfig-interfaces/interfaces.yaml

```
---
metadata:
    processor: TextTranslator
    root: true
interfaces:
    _process: unnecessary
    interface:
        _process:
            - key_value: "interface {{ interface_key }}\n"
              negate: "{{ 'no' if interface_key[0:4] in ['Port', 'Loop'] else 'default' }} interface {{ interface_key }}\n"
              end: "    exit\n"
        config:
            _process: unnecessary
            enabled:
                _process:
                    - value: "    shutdown\n"
                      when: "{{ not model }}"
            description:
                _process:
                    - value: "    description {{ model }}\n"
                      negate: "    default description\n"
            mtu:
                _process:
                    - value: "    mtu {{ model }}\n"
                      negate: "    default mtu\n"
```

1. What is YANG?
2. What is OpenConfig?
3. **What is napalm-yang?**
    1. Basics
    2. **Advanced features**
    3. Integration with `ansible`

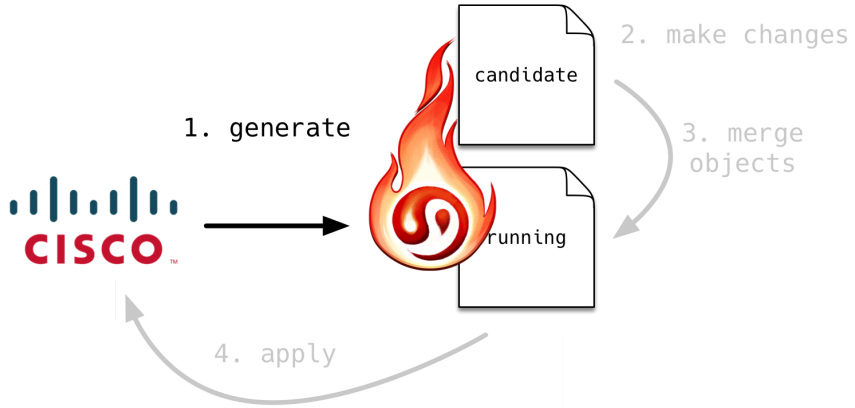
Merging Configurations



Merging Configurations (1)

merge_config.py

```
1  def main():
2      ios_driver = get_network_driver("ios")
3      with ios_driver(**configuration.ios) as device:
4          device.yang.get_openconfig_interfaces(candidate=True)
5
6          iface = device.yang.candidate.interfaces.\
7            interface['GigabitEthernet2']
8
9          addr = iface.routed_vlan.ipv4.addresses.address.\
10                 add('192.168.2.1')
11         addr.config.ip = '192.168.2.1'
12         addr.config.prefix_length = 24
13
14         iface.routed_vlan.ipv4.addresses.address.\
15           delete('192.168.1.1')
16
17         pretty_print(device.yang.diff())
18
19         merge_config = device.yang.translate(merge=True)
20         print(merge_config)
21
22         device.load_merge_candidate(config=merge_config)
23         device.commit_config()
24
25         device.yang.get_openconfig_interfaces()
26         pretty_print(device.yang.diff())
27
28
```



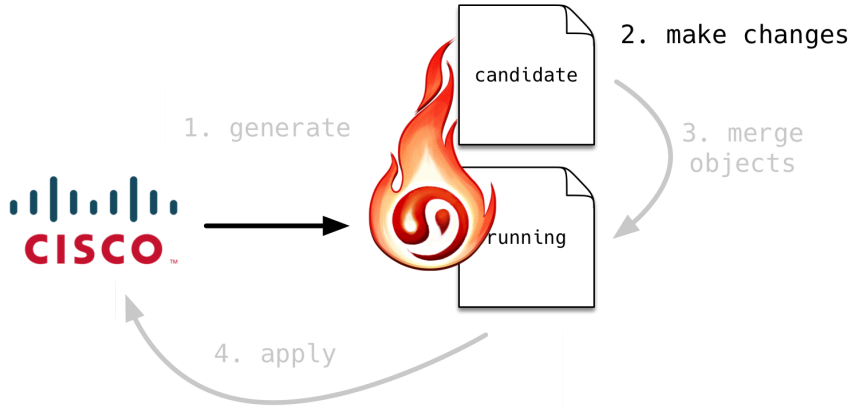
running config

```
1  vagrant_box#show run int Gi2
2  Building configuration...
3
4  Current configuration : 100 bytes
5  !
6  interface GigabitEthernet2
7   ip address 192.168.1.1 255.255.255.0
8   shutdown
9   negotiation auto
10 end
```

Merging Configurations (2)

merge_config.py

```
1  def main():
2      ios_driver = get_network_driver("ios")
3      with ios_driver(**configuration.ios) as device:
4          device.yang.get_openconfig_interfaces(candidate=True)
5
6          iface = device.yang.candidate.interfaces.\
7            interface['GigabitEthernet2']
8
9          addr = iface.routed_vlan.ipv4.addresses.address.\
10                 add('192.168.2.1')
11         addr.config.ip = '192.168.2.1'
12         addr.config.prefix_length = 24
13
14         iface.routed_vlan.ipv4.addresses.address.\
15           delete('192.168.1.1')
16
17         pretty_print(device.yang.diff())
18
19         merge_config = device.yang.translate(merge=True)
20         print(merge_config)
21
22         device.load_merge_candidate(config=merge_config)
23         device.commit_config()
24
25         device.yang.get_openconfig_interfaces()
26         pretty_print(device.yang.diff())
27
28
```



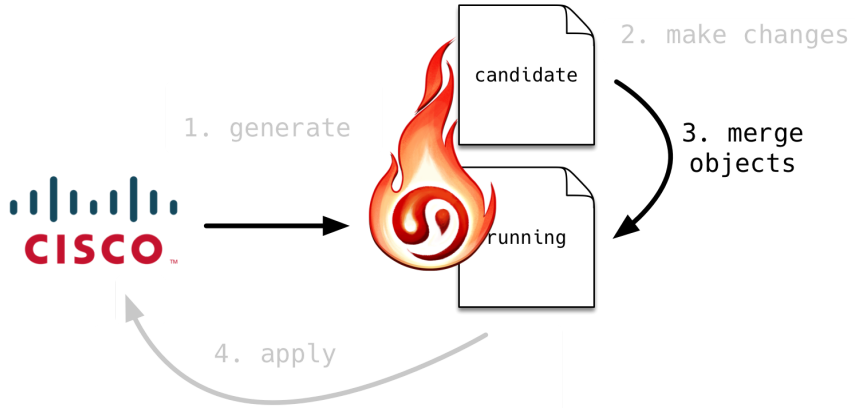
running config

```
1  vagrant_box#show run int Gi2
2  Building configuration...
3
4  Current configuration : 100 bytes
5  !
6  interface GigabitEthernet2
7   ip address 192.168.1.1 255.255.255.0
8   shutdown
9   negotiation auto
10 end
```

Merging Configurations (3)

merge_config.py

```
1  def main():
2      ios_driver = get_network_driver("ios")
3      with ios_driver(**configuration.ios) as device:
4          device.yang.get_openconfig_interfaces(candidate=True)
5
6          iface = device.yang.candidate.interfaces.\
7            interface['GigabitEthernet2']
8
9          addr = iface.routed_vlan.ipv4.addresses.address.\
10                 add('192.168.2.1')
11         addr.config.ip = '192.168.2.1'
12         addr.config.prefix_length = 24
13
14         iface.routed_vlan.ipv4.addresses.address.\
15           delete('192.168.1.1')
16
17         pretty_print(device.yang.diff())
18
19         merge_config = device.yang.translate(merge=True)
20         print(merge_config)
21
22         device.load_merge_candidate(config=merge_config)
23         device.commit_config()
24
25         device.yang.get_openconfig_interfaces()
26         pretty_print(device.yang.diff())
27
28
```



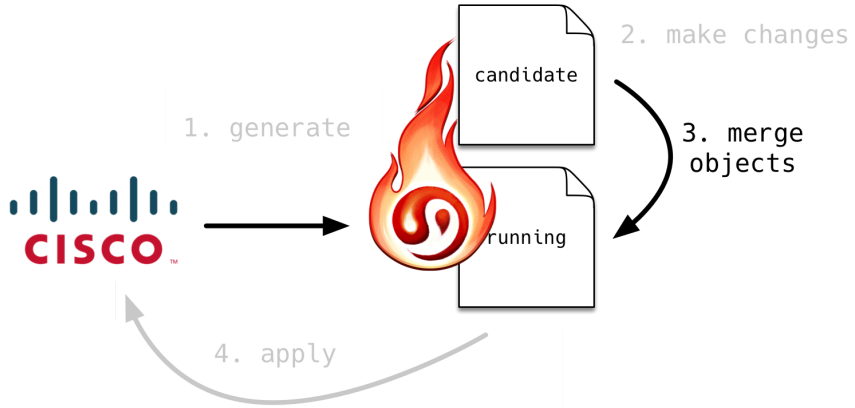
diff

```
1  {"interfaces": { "interface": {
2          "both": {
3              "GigabitEthernet2": {
4                  "routed_vlan": {
5                      "ipv4": {
6                          "addresses": {
7                              "address": {
8                                  "second_only": [
9                                      "192.168.1.1"
10                                 ],
11                                 "first_only": [
12                                     "192.168.2.1"
13                                 ]}}}}}}}}}
```

Merging Configurations (4)

merge_config.py

```
1  def main():
2      ios_driver = get_network_driver("ios")
3      with ios_driver(**configuration.ios) as device:
4          device.yang.get_openconfig_interfaces(candidate=True)
5
6          iface = device.yang.candidate.interfaces.\
7            interface['GigabitEthernet2']
8
9          addr = iface.routed_vlan.ipv4.addresses.address.\
10                 add('192.168.2.1')
11         addr.config.ip = '192.168.2.1'
12         addr.config.prefix_length = 24
13
14         iface.routed_vlan.ipv4.addresses.address.\
15           delete('192.168.1.1')
16
17         pretty_print(device.yang.diff())
18
19         merge_config = device.yang.translate(merge=True)
20         print(merge_config)
21
22         device.load_merge_candidate(config=merge_config)
23         device.commit_config()
24
25         device.yang.get_openconfig_interfaces()
26         pretty_print(device.yang.diff())
27
28
```



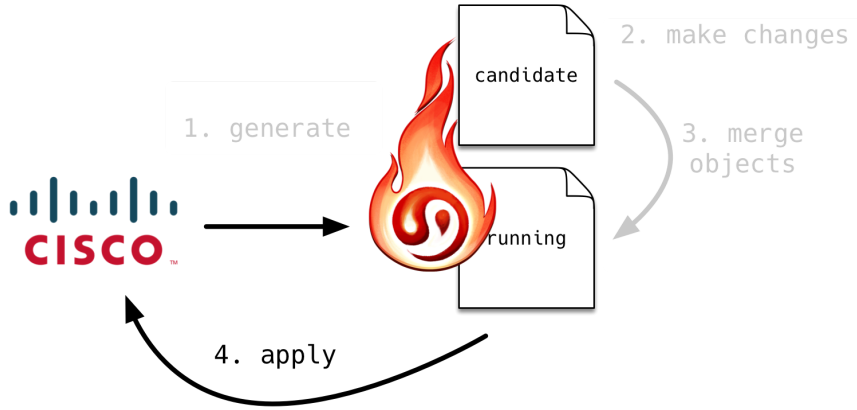
merge commands

```
1  interface GigabitEthernet2
2      ip address 192.168.2.1 255.255.255.0
3      no ip address 192.168.1.1 255.255.255.0
4      exit
```

Merging Configurations (5)

merge_config.py

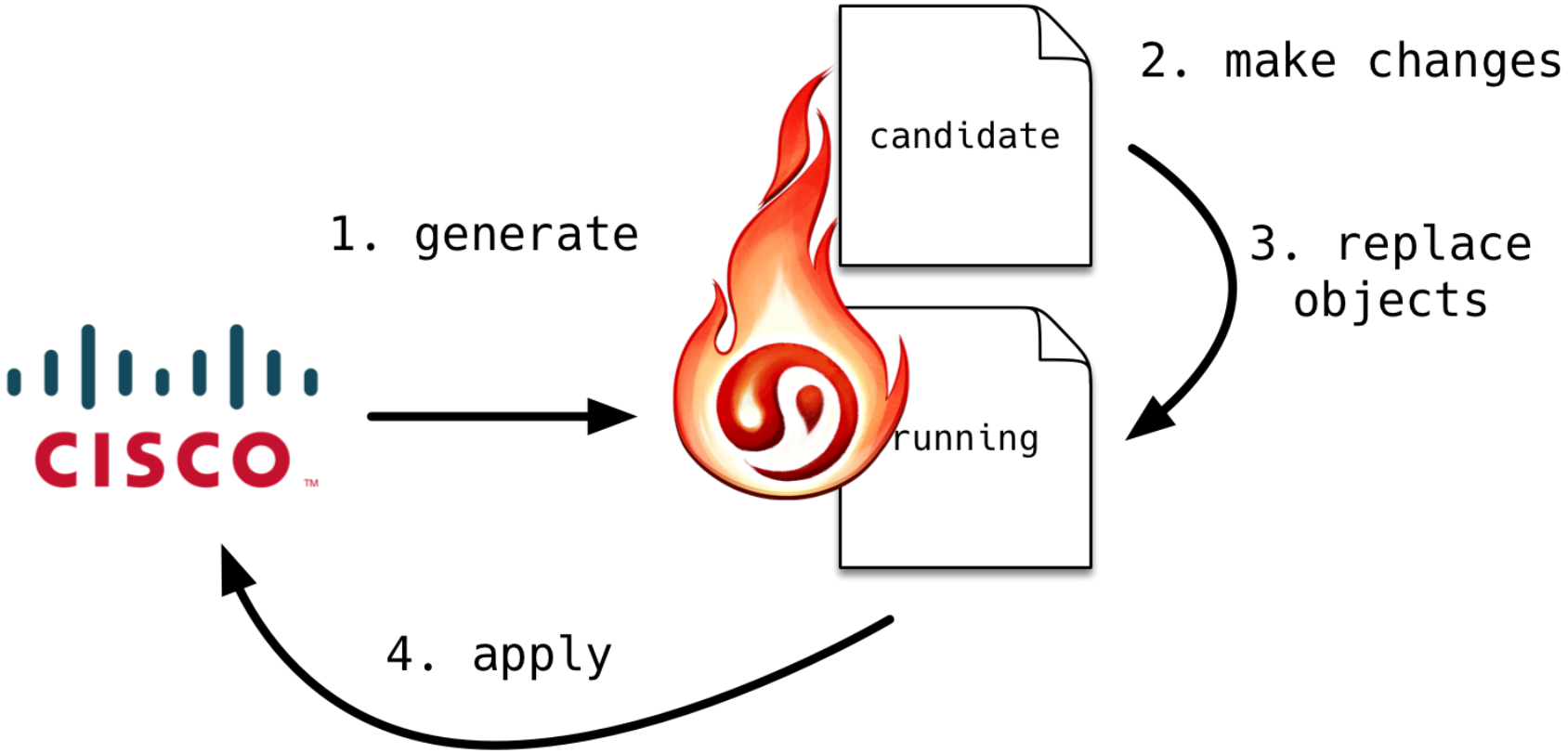
```
1  def main():
2      ios_driver = get_network_driver("ios")
3      with ios_driver(**configuration.ios) as device:
4          device.yang.get_openconfig_interfaces(candidate=True)
5
6          iface = device.yang.candidate.interfaces.\
7            interface['GigabitEthernet2']
8
9          addr = iface.routed_vlan.ipv4.addresses.address.\
10                 add('192.168.2.1')
11         addr.config.ip = '192.168.2.1'
12         addr.config.prefix_length = 24
13
14         iface.routed_vlan.ipv4.addresses.address.\
15           delete('192.168.1.1')
16
17         pretty_print(device.yang.diff())
18
19         merge_config = device.yang.translate(merge=True)
20         print(merge_config)
21
22         device.load_merge_candidate(config=merge_config)
23         device.commit_config()
24
25         device.yang.get_openconfig_interfaces()
26         pretty_print(device.yang.diff())
27
28
```



running config

```
1  vagrant_box#show run int Gi2
2  Building configuration...
3
4  Current configuration : 100 bytes
5  !
6  interface GigabitEthernet2
7   ip address 192.168.2.1 255.255.255.0
8   shutdown
9   negotiation auto
10 end
```

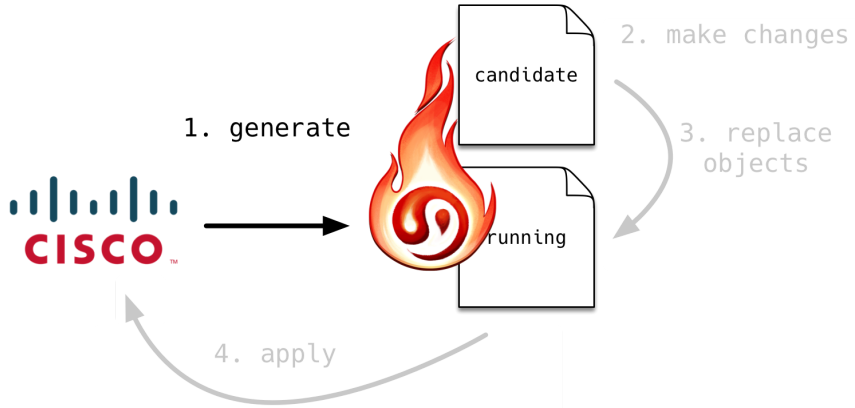
Replacing Configuration



Replacing Configurations (1)

replace_config.py

```
1  def main():
2      junos_driver = get_network_driver("junos")
3      with junos_driver(**configuration.junos) as device:
4          device.yang.get_openconfig_interfaces(candidate=True)
5
6          iface = device.yang.candidate.interfaces.\
7              interface['ge-0/0/1'].\
8              subinterfaces.subinterface['0']
9
10         addr = iface.ipv4.addresses.address.\
11             add('192.168.2.1')
12         addr.config.ip = '192.168.2.1'
13         addr.config.prefix_length = 24
14
15         iface.ipv4.addresses.address.\
16             delete('192.168.1.1')
17
18         pretty_print(device.yang.diff())
19
20         merge_config = device.yang.translate(replace=True)
21         print(merge_config)
22
23         device.load_merge_candidate(config=merge_config)
24         device.commit_config()
25
26         device.yang.get_openconfig_interfaces()
27         pretty_print(device.yang.diff())
28
```



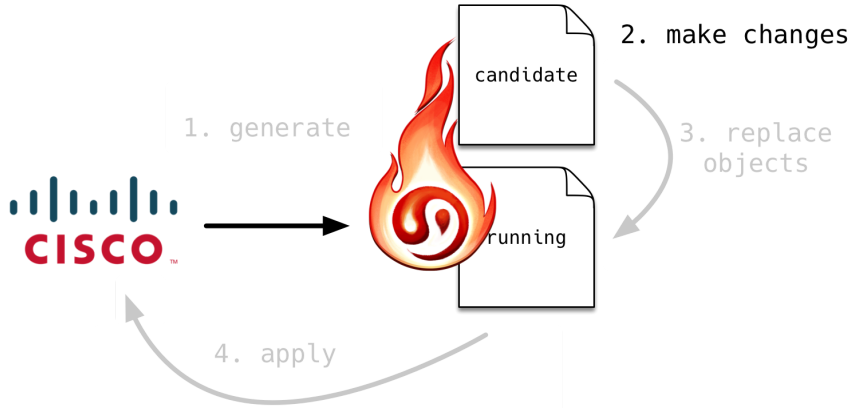
running config

```
1  root@vsrx> show configuration interfaces
2  ge-0/0/0 {
3      unit 0 {
4          family inet {
5              address 10.0.2.15/24;
6          }
7      }
8  }
9  ge-0/0/1 {
10     promiscuous-mode;
11     unit 0 {
12         family inet {
13             address 192.168.1.1/24;
14         }
15     }
```

Replacing Configurations (2)

replace_config.py

```
1  def main():
2      junos_driver = get_network_driver("junos")
3      with junos_driver(**configuration.junos) as device:
4          device.yang.get_openconfig_interfaces(candidate=True)
5
6          iface = device.yang.candidate.interfaces.\
7              interface['ge-0/0/1'].\
8              subinterfaces.subinterface['0']
9
10         addr = iface.ipv4.addresses.address.\
11             add('192.168.2.1')
12         addr.config.ip = '192.168.2.1'
13         addr.config.prefix_length = 24
14
15         iface.ipv4.addresses.address.\
16             delete('192.168.1.1')
17
18         pretty_print(device.yang.diff())
19
20         merge_config = device.yang.translate(replace=True)
21         print(merge_config)
22
23         device.load_merge_candidate(config=merge_config)
24         device.commit_config()
25
26         device.yang.get_openconfig_interfaces()
27         pretty_print(device.yang.diff())
28
```



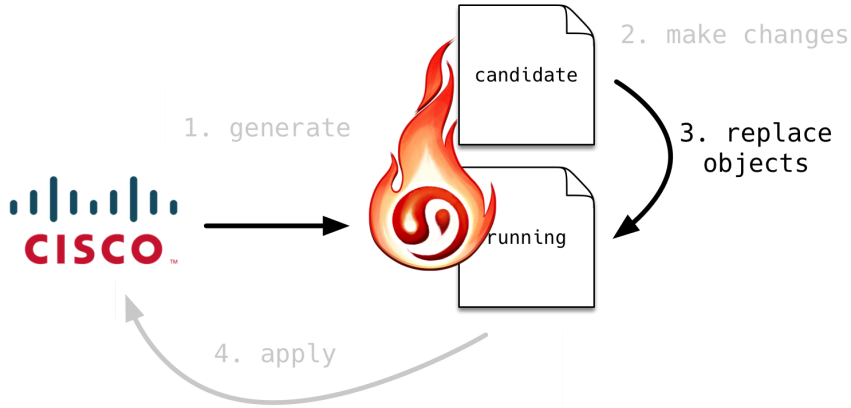
running config

```
1  root@vsrx> show configuration interfaces
2  ge-0/0/0 {
3      unit 0 {
4          family inet {
5              address 10.0.2.15/24;
6          }
7      }
8  }
9  ge-0/0/1 {
10     promiscuous-mode;
11     unit 0 {
12         family inet {
13             address 192.168.1.1/24;
14         }
15     }
```

Replacing Configurations (3)

replace_config.py

```
1  def main():
2      junos_driver = get_network_driver("junos")
3      with junos_driver(**configuration.junos) as device:
4          device.yang.get_openconfig_interfaces(candidate=True)
5
6          iface = device.yang.candidate.interfaces.\
7              interface['ge-0/0/1'].\
8              subinterfaces.subinterface['0']
9
10         addr = iface.ipv4.addresses.address.\
11             add('192.168.2.1')
12         addr.config.ip = '192.168.2.1'
13         addr.config.prefix_length = 24
14
15         iface.ipv4.addresses.address.\
16             delete('192.168.1.1')
17
18         pretty_print(device.yang.diff())
19
20         merge_config = device.yang.translate(replace=True)
21         print(merge_config)
22
23         device.load_merge_candidate(config=merge_config)
24         device.commit_config()
25
26         device.yang.get_openconfig_interfaces()
27         pretty_print(device.yang.diff())
28
```



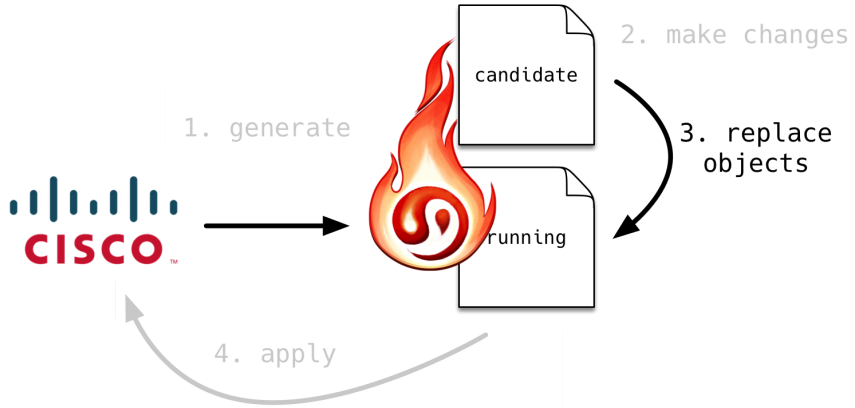
diff

```
1  { "interfaces": { "interface": { "both": {
2      "ge-0/0/1": {
3          "subinterfaces": { "subinterface": {
4                  "both": {
5                      "0": {
6                          "ipv4": {
7                              "addresses": {
8                                  "address": {
9                                      "second_only": [
10                                         "192.168.1.1"
11                                     ],
12                                     "first_only": [
13                                         "192.168.2.1"
14                                     ]
15 }}}}}}}}}}}}
```

Replacing Configurations (4)

replace_config.py

```
1  def main():
2      junos_driver = get_network_driver("junos")
3      with junos_driver(**configuration.junos) as device:
4          device.yang.get_openconfig_interfaces(candidate=True)
5
6          iface = device.yang.candidate.interfaces.\
7              interface['ge-0/0/1'].\
8              subinterfaces.subinterface['0']
9
10         addr = iface.ipv4.addresses.address.\
11             add('192.168.2.1')
12         addr.config.ip = '192.168.2.1'
13         addr.config.prefix_length = 24
14
15         iface.ipv4.addresses.address.\
16             delete('192.168.1.1')
17
18         pretty_print(device.yang.diff())
19
20         merge_config = device.yang.translate(replace=True)
21         print(merge_config)
22
23         device.load_merge_candidate(config=merge_config)
24         device.commit_config()
25
26         device.yang.get_openconfig_interfaces()
27         pretty_print(device.yang.diff())
28
```



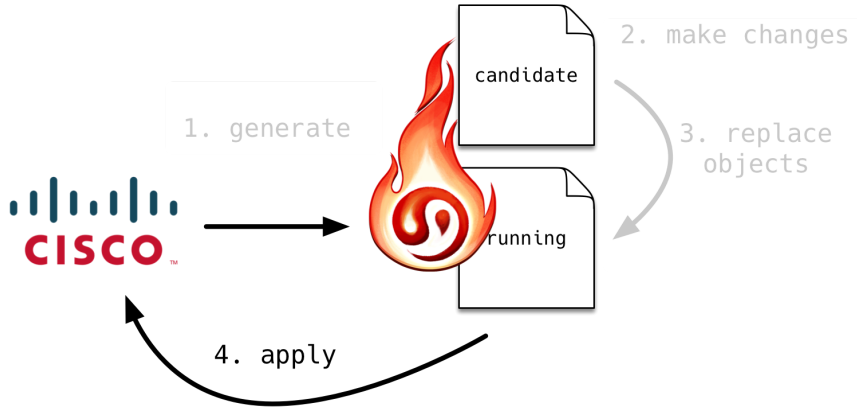
replace commands

```
1   <interfaces replace="replace">
2     <interface>
3       <name>ge-0/0/1</name>
4       <unit>
5         <name>0</name>
6         <family>
7           <inet>
8             <address>
9               <name>192.168.2.1/24</name>
10            </address>
11          </inet>
12        </family>
13      </unit>
14    </interface>
15  </interfaces>
```

Replacing Configurations (5)

replace_config.py

```
1  def main():
2      junos_driver = get_network_driver("junos")
3      with junos_driver(**configuration.junos) as device:
4          device.yang.get_openconfig_interfaces(candidate=True)
5
6          iface = device.yang.candidate.interfaces.\
7              interface['ge-0/0/1'].\
8              subinterfaces.subinterface['0']
9
10         addr = iface.ipv4.addresses.address.\
11             add('192.168.2.1')
12         addr.config.ip = '192.168.2.1'
13         addr.config.prefix_length = 24
14
15         iface.ipv4.addresses.address.\
16             delete('192.168.1.1')
17
18         pretty_print(device.yang.diff())
19
20         merge_config = device.yang.translate(replace=True)
21         print(merge_config)
22
23         device.load_merge_candidate(config=merge_config)
24         device.commit_config()
25
26         device.yang.get_openconfig_interfaces()
27         pretty_print(device.yang.diff())
28
```



running config

```
1  root@vsrx> show configuration interfaces
2  ge-0/0/0 {
3      unit 0 {
4          family inet {
5              address 10.0.2.15/24;
6          }
7      }
8  }
9  ge-0/0/1 {
10     unit 0 {
11         family inet {
12             address 192.168.2.1/24;
13         }
14     }
```

Merging vs Replacing

Merge

Works fine when you don’t control the entirety of the configuration

Replace

Best option for greenfield, fully automated environments

1. What is YANG?
2. What is OpenConfig?
3. **What is napalm-yang?**
    1. Basics
    2. Advanced features
    3. **Integration with ansible**

Setup

```
[dev]
junos dev_os=junos hostname=127.0.0.1 username=vagrant password="" port=12203
ios dev_os=ios hostname=127.0.0.1 username=vagrant password=vagrant port=12204
```

Parsing Configuration (playbook)

playbook_parse.yaml

```
---
- name: Playbook to gather OpenConfig models
  hosts: all
  gather_facts: false
  connection: local
  tags: print_action

  tasks:
    - name: Let's gather config of interfaces from device
      napalm_parse_yang:
        dev_os: "{{ dev_os }}"
        hostname: "{{ hostname }}"
        username: "{{ username }}"
        password: "{{ password }}"
        mode: "config"
        optional_args:
            port: "{{ port }}"
        models:
            - openconfig_interfaces
      register: running
    - debug:
       msg: "{{ running|to_nice_json }}"
```

Parsing Configuration (junos)

ansible-playbook –limit junos playbook_parse.yaml

```
# Let's gather config of interfaces from device ***************************************************************
  * junos                      - changed=False --  ------------------------------------------
# debug ********************************************************************************************************
  * junos                      - changed=False ------------------------------------------------
    {
        "changed": false,
        "yang_model": {
            "interfaces": {
                "interface": {
                    "ae0": {
                        "config": {
                            "enabled": true,
                            "name": "ae0",
                            "type": "ieee8023adLag"
                        },
                        "name": "ae0",
                        "routed_vlan": {
                            "ipv4": {
                                "config": {
                                    "enabled": false
                                }
                            }
                        },
                        "subinterfaces": {
                            "subinterface": {
                                "0": {
                                    "config": {
                                        "description": "A new description",
                                        "enabled": true,
                                        "name": "0"
```

Parsing Configuration (ios)

ansible-playbook –limit ios playbook_parse.yaml

```
# Let's gather config of interfaces from device ********************************************************************
  * ios                        - changed=False --  ------------------------------------------
# debug ************************************************************************************************************
  * ios                        - changed=False ------------------------------------------------
    {
        "changed": false,
        "yang_model": {
            "interfaces": {
                "interface": {
                    "GigabitEthernet1": {
                        "config": {
                            "description": "This is a description",
                            "enabled": true,
                            "mtu": 1514,
                            "type": "ethernetCsmacd"
                        },
                        "name": "GigabitEthernet1",
                        "routed_vlan": {
                            "ipv4": {
                                "config": {
                                    "enabled": true
                                }
                            }
                        }
                    },
                    "GigabitEthernet2": {
                        "config": {
                            "description": "so much oc",
                            "enabled": false,
                            "mtu": 1514,
```

Configuring Devices (data, junos)

host_vars/junos.yaml

```
yang_data:
    interfaces:
        interface:
            ae0:
                config:
                    enabled: true
                    name: ae0
                    type: ieee8023adLag
                name: ae0
                subinterfaces:
                    subinterface:
                        0:
                            config:
                                description: A new description
                                enabled: true
                                name: 0
                            index: 0
                            ipv4:
                                addresses:
                                    address:
                                        172.20.100.1:
                                            config:
                                                ip: 172.20.100.1
                                                prefix_length: 24
                                            ip: 172.20.100.1
                                        192.168.100.1:
                                            config:
                                                ip: 192.168.100.1
                                                prefix_length: 24
                                            ip: 192.168.100.1
```

Configuring Devices (data, ios)

host_vars/ios.yaml

```
yang_data:
    interfaces:
        interface:
            GigabitEthernet1:
                config:
                    description: This is a description
                    mtu: 1514
                    enabled: true
                    type: ethernetCsmacd
                name: GigabitEthernet1
                routed_vlan:
                    ipv4:
                        config:
                            enabled: true
            GigabitEthernet2:
                config:
                    description: so much oc
                    enabled: false
                    mtu: 1514
                    type: ethernetCsmacd
                name: GigabitEthernet2
                routed_vlan:
                    ipv4:
                        addresses:
                            address:
                                192.168.0.1:
                                    config:
                                        ip: 192.168.0.1
                                        prefix_length: 24
                                        secondary: false
```

Configuring Devices (playbook)

playbook_configure.yaml

```
---
- name: Playbook to configure devices using YANG models
  hosts: all
  gather_facts: false
  connection: local
  tags: print_action

  tasks:
    - name: Install Config and save diff
      napalm_yang_install_config:
        dev_os: "{{ dev_os }}"
        hostname: "{{ hostname }}"
        username: "{{ username }}"
        password: "{{ password }}"
        optional_args:
            port: "{{ port }}"
        config: "{{ yang_data }}"
        profiles: "{{ profiles }}"
      register: result
    - name: merge configuration
      debug:
        msg: "{{ result.config }}"
    - name: napalm-yang diff
      debug:
        msg: "{{ result.yang_diff|to_nice_json }}"
    - name: native diff
      debug:
        msg: "{{ result.native_diff }}"
```

Configuring Devices (junos, execution)

playbook_configure.yaml

```yaml
---
- name: Playbook to configure devices using YANG models
  hosts: all
  gather_facts: false
  connection: local
  tags: print_action

  tasks:
    - name: Install Config and save diff
      napalm_yang_install_config:
        dev_os: "{{ dev_os }}"
        hostname: "{{ hostname }}"
        username: "{{ username }}"
        password: "{{ password }}"
        optional_args:
            port: "{{ port }}"
        config: "{{ yang_data }}"
        profiles: "{{ profiles }}"
      register: result
    - name: merge configuration
      debug:
        msg: "{{ result.config }}"
    - name: napalm-yang diff
      debug:
        msg: "{{ result.yang_diff|to_nice_json }}"
    - name: native diff
      debug:
        msg: "{{ result.native_diff }}"
```

ansible-playbook –limit junos …

```
# Install Config and save diff ********************************
  * junos                      - changed=True -- --------------
```

Configuring Devices (junos, config)

playbook_configure.yaml

```yaml
---
- name: Playbook to configure devices using YANG models
  hosts: all
  gather_facts: false
  connection: local
  tags: print_action

  tasks:
    - name: Install Config and save diff
      napalm_yang_install_config:
        dev_os: "{{ dev_os }}"
        hostname: "{{ hostname }}"
        username: "{{ username }}"
        password: "{{ password }}"
        optional_args:
            port: "{{ port }}"
        config: "{{ yang_data }}"
        profiles: "{{ profiles }}"
      register: result
    - name: merge configuration
      debug:
        msg: "{{ result.config }}"
    - name: napalm-yang diff
      debug:
        msg: "{{ result.yang_diff|to_nice_json }}"
    - name: native diff
      debug:
        msg: "{{ result.native_diff }}"
```

ansible-playbook –limit junos …

```
# Install Config and save diff ******************************
  * junos                        - changed=True -- -------------
# merge configuration ***************************************
  * junos                        - changed=False ---------------
    <configuration>
      <interfaces>
        <interface>
          <name>ae0</name>
          <unit>
            <name>1</name>
            <vlan-id>1</vlan-id>
            <family>
              <inet>
                <address>
                  <name>192.168.101.1/24</name>
                </address>
              </inet>
            </family>
            <disable/>
            <description>ae0.1</description>
          </unit>
          <vlan-tagging/>
          <unit>
            <name>0</name>
            <vlan-id>100</vlan-id>
            <family>
              <inet>
                <address>
```

Configuring Devices (junos, yang diff)

playbook_configure.yaml

```yaml
---
- name: Playbook to configure devices using YANG models
  hosts: all
  gather_facts: false
  connection: local
  tags: print_action

  tasks:
    - name: Install Config and save diff
      napalm_yang_install_config:
        dev_os: "{{ dev_os }}"
        hostname: "{{ hostname }}"
        username: "{{ username }}"
        password: "{{ password }}"
        optional_args:
            port: "{{ port }}"
        config: "{{ yang_data }}"
        profiles: "{{ profiles }}"
      register: result
    - name: merge configuration
      debug:
        msg: "{{ result.config }}"
    - name: napalm-yang diff
      debug:
        msg: "{{ result.yang_diff|to_nice_json }}"
    - name: native diff
      debug:
        msg: "{{ result.native_diff }}"
```

ansible-playbook –limit junos …

```
# napalm-yang diff ****************************************
  * junos                        - changed=False ---------------
    {
        "interfaces": {
            "interface": {
                "both": {
                    "ge-0/0/0": {
                        "config": {
                            "description": {
                                "first": "management interface
                                "second": ""
                            },
                            "mtu": {
                                "first": "1400",
                                "second": "0"
                            }
                        },
                        "subinterfaces": {
                            "subinterface": {
                                "both": {
                                    "0": {
                                        "config": {
                                            "description": {
                                                "first": "ge-0
                                                "second": ""
                                            }
                                        }
                                    }
```

Configuring Devices (junos, native diff)

playbook_configure.yaml

```
---
- name: Playbook to configure devices using YANG models
  hosts: all
  gather_facts: false
  connection: local
  tags: print_action

  tasks:
    - name: Install Config and save diff
      napalm_yang_install_config:
        dev_os: "{{ dev_os }}"
        hostname: "{{ hostname }}"
        username: "{{ username }}"
        password: "{{ password }}"
        optional_args:
            port: "{{ port }}"
        config: "{{ yang_data }}"
        profiles: "{{ profiles }}"
      register: result
    - name: merge configuration
      debug:
        msg: "{{ result.config }}"
    - name: napalm-yang diff
      debug:
        msg: "{{ result.yang_diff|to_nice_json }}"
    - name: native diff
      debug:
        msg: "{{ result.native_diff }}"
```

ansible-playbook –limit junos …

```
# native diff ************************************************
  * junos                        - changed=False ---------------
    [edit interfaces ge-0/0/0]
    +   description "management interface";
    +   mtu 1400;
    [edit interfaces ge-0/0/0 unit 0]
    +    description ge-0/0/0.0;
    [edit interfaces]
    +   ge-0/0/1 {
    +       description ge-0/0/1;
    +       disable;
    +   }
    +   ae0 {
    +       vlan-tagging;
    +       unit 0 {
    +           description "A new description";
    +           vlan-id 100;
    +           family inet {
    +               address 192.168.100.1/24;
    +               address 172.20.100.1/24;
    +           }
    +       }
    +       unit 1 {
    +           disable;
    +           description ae0.1;
    +           vlan-id 1;
    +           family inet {
    +               address 192.168.101.1/24;
```

Configuring Devices (junos, again)

playbook_configure.yaml

```
---
- name: Playbook to configure devices using YANG models
  hosts: all
  gather_facts: false
  connection: local
  tags: print_action

  tasks:
    - name: Install Config and save diff
      napalm_yang_install_config:
        dev_os: "{{ dev_os }}"
        hostname: "{{ hostname }}"
        username: "{{ username }}"
        password: "{{ password }}"
        optional_args:
            port: "{{ port }}"
        config: "{{ yang_data }}"
        profiles: "{{ profiles }}"
      register: result
    - name: merge configuration
      debug:
        msg: "{{ result.config }}"
    - name: napalm-yang diff
      debug:
        msg: "{{ result.yang_diff|to_nice_json }}"
    - name: native diff
      debug:
        msg: "{{ result.native_diff }}"
```

ansible-playbook –limit junos …

```
# Install Config and save diff ******************************
  * junos                      - changed=False --  -----------
# merge configuration ***************************************
  * junos                      - changed=False ---------------
# napalm-yang diff ******************************************
  * junos                      - changed=False -- {} ---------
# native diff ***********************************************
  * junos                      - changed=False --  -----------

# STATS *****************************************************
junos     : ok=4    changed=0         failed=0 unreachable=0
```

Configuring Devices (ios, execution)

playbook_configure.yaml

```yaml
---
- name: Playbook to configure devices using YANG models
  hosts: all
  gather_facts: false
  connection: local
  tags: print_action

  tasks:
    - name: Install Config and save diff
      napalm_yang_install_config:
        dev_os: "{{ dev_os }}"
        hostname: "{{ hostname }}"
        username: "{{ username }}"
        password: "{{ password }}"
        optional_args:
            port: "{{ port }}"
        config: "{{ yang_data }}"
        profiles: "{{ profiles }}"
      register: result
    - name: merge configuration
      debug:
        msg: "{{ result.config }}"
    - name: napalm-yang diff
      debug:
        msg: "{{ result.yang_diff|to_nice_json }}"
    - name: native diff
      debug:
        msg: "{{ result.native_diff }}"
```

ansible-playbook –limit ios …

```
# Install Config and save diff ********************************
  * ios                        - changed=True --  -------------
```

Configuring Devices (ios, config)

playbook_configure.yaml

```yaml
---
- name: Playbook to configure devices using YANG models
  hosts: all
  gather_facts: false
  connection: local
  tags: print_action

  tasks:
    - name: Install Config and save diff
      napalm_yang_install_config:
        dev_os: "{{ dev_os }}"
        hostname: "{{ hostname }}"
        username: "{{ username }}"
        password: "{{ password }}"
        optional_args:
            port: "{{ port }}"
        config: "{{ yang_data }}"
        profiles: "{{ profiles }}"
      register: result
    - name: merge configuration
      debug:
        msg: "{{ result.config }}"
    - name: napalm-yang diff
      debug:
        msg: "{{ result.yang_diff|to_nice_json }}"
    - name: native diff
      debug:
        msg: "{{ result.native_diff }}"
```

ansible-playbook –limit ios …

```
# Install Config and save diff ******************************
  * ios                          - changed=True --  ------------
# merge configuration ***************************************
  * ios                          - changed=False ---------------
    interface Port-channel1
        description blah
        mtu 9000
        exit
    interface Port-channel1.1
        exit
    interface GigabitEthernet1
        description This is a description
        mtu 1514
        exit
    interface Loopback1
        description a loopback
        exit
    interface GigabitEthernet2
        ip address 192.168.0.1 255.255.255.0
        description so much oc
        mtu 1514
        exit
    interface GigabitEthernet2.1
        encapsulation dot1q 10
        ip address 172.20.1.1 255.255.255.0 secondary
        ip address 192.168.1.1 255.255.255.0
        description another subiface
        exit
```

Configuring Devices (ios, yang diff)

playbook_configure.yaml

```yaml
---
- name: Playbook to configure devices using YANG models
  hosts: all
  gather_facts: false
  connection: local
  tags: print_action

  tasks:
    - name: Install Config and save diff
      napalm_yang_install_config:
        dev_os: "{{ dev_os }}"
        hostname: "{{ hostname }}"
        username: "{{ username }}"
        password: "{{ password }}"
        optional_args:
            port: "{{ port }}"
        config: "{{ yang_data }}"
        profiles: "{{ profiles }}"
      register: result
    - name: merge configuration
      debug:
        msg: "{{ result.config }}"
    - name: napalm-yang diff
      debug:
        msg: "{{ result.yang_diff|to_nice_json }}"
    - name: native diff
      debug:
        msg: "{{ result.native_diff }}"
```

ansible-playbook –limit ios …

```
# napalm-yang diff ****************************************
  * ios                          - changed=False ---------------
    {
        "interfaces": {
            "interface": {
                "both": {
                    "GigabitEthernet1": {
                        "config": {
                            "description": {
                                "first": "This is a descriptio
                                "second": ""
                            },
                            "mtu": {
                                "first": "1514",
                                "second": "1500"
                            }
                        }
                    },
                    "GigabitEthernet2": {
                        "config": {
                            "description": {
                                "first": "so much oc",
                                "second": ""
                            },
                            "mtu": {
                                "first": "1514",
                                "second": "1500"
                            }
```

Configuring Devices (ios, native diff)

playbook_configure.yaml

```yaml
---
- name: Playbook to configure devices using YANG models
  hosts: all
  gather_facts: false
  connection: local
  tags: print_action

  tasks:
    - name: Install Config and save diff
      napalm_yang_install_config:
        dev_os: "{{ dev_os }}"
        hostname: "{{ hostname }}"
        username: "{{ username }}"
        password: "{{ password }}"
        optional_args:
            port: "{{ port }}"
        config: "{{ yang_data }}"
        profiles: "{{ profiles }}"
      register: result
    - name: merge configuration
      debug:
        msg: "{{ result.config }}"
    - name: napalm-yang diff
      debug:
        msg: "{{ result.yang_diff|to_nice_json }}"
    - name: native diff
      debug:
        msg: "{{ result.native_diff }}"
```

ansible-playbook –limit ios …

```
# native diff ************************************************
  * ios                        - changed=False ---------------
    +interface Port-channel1
    +    description blah
    +interface Port-channel1.1
    +    exit
    +interface GigabitEthernet1
    +    description This is a description
    +    mtu 1514
    +    exit
    +interface Loopback1
    +    description a loopback
    +interface GigabitEthernet2
    +    ip address 192.168.0.1 255.255.255.0
    +    description so much oc
    +    mtu 1514
    +    exit
    +interface GigabitEthernet2.1
    +    encapsulation dot1q 10
    +interface GigabitEthernet2.2
    +    encapsulation dot1q 20
    +Error: received error code 5 when looking for missing lin
    +!Error: cfgdiff returned error 5

# STATS ******************************************************
ios    : ok=4      changed=1          failed=0 unreachable=0
```

Configuring Devices (ios, again)

playbook_configure.yaml

```yaml
---
- name: Playbook to configure devices using YANG models
  hosts: all
  gather_facts: false
  connection: local
  tags: print_action

  tasks:
    - name: Install Config and save diff
      napalm_yang_install_config:
        dev_os: "{{ dev_os }}"
        hostname: "{{ hostname }}"
        username: "{{ username }}"
        password: "{{ password }}"
        optional_args:
            port: "{{ port }}"
        config: "{{ yang_data }}"
        profiles: "{{ profiles }}"
      register: result
    - name: merge configuration
      debug:
        msg: "{{ result.config }}"
    - name: napalm-yang diff
      debug:
        msg: "{{ result.yang_diff|to_nice_json }}"
    - name: native diff
      debug:
        msg: "{{ result.native_diff }}"
```

ansible-playbook –limit ios …

```
# Install Config and save diff ******************************
  * ios                        - changed=False --  -----------
# merge configuration ***************************************
  * ios                        - changed=False --  -----------
# napalm-yang diff ******************************************
  * ios                        - changed=False -- {} ---------
# native diff ***********************************************
  * ios                        - changed=False --  -----------

# STATS *****************************************************
ios     : ok=4      changed=0          failed=0 unreachable=0
```

Data ain’t easy

Abstractions/Intentions

From simple service definitions to complex/descriptive data structures

service.yaml

```yaml
fabric_link:
    - ip: 10.0.0.0/31
      connects_to: spine00
```

expanded.yaml

```yaml
interfaces:
    interface:
        ethernet1:
            config:
                description: uplink to spine00
                enabled: false
                mtu: 9000
                type: ethernetCsmacd
            name: GigabitEthernet2
            routed_vlan:
                ipv4:
                    addresses:
                        address:
                            10.0.0.0:
                                config:
                                    ip: 10.0.0.0
                                    prefix_length: 31
                                    secondary: false
                                ip: 10.0.0.0
                    config:
                        enabled: true
bgp:
    neigbhors:
        neighbor:
            10.0.0.1:
```

Summary

1. YANG is a very extensible data modeling language
2. YANG is not an API
3. Openconfig is an industry effort to define data models
4. `napalm-yang` aims to help transitioning to YANG/Openconfig models
5. `napalm-yang` integrates with the usual suspects; `ansible`, `salt`, etc.

References

https://github.com/dravetech/openconfig-with-napalm

https://github.com/napalm-automation/napalm

https://github.com/napalm-automation/napalm-yang

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

@napalm_auto @dbarrosop