Getting Started with Resource Modules

Gomathi Selvi Srinivasan

Senior Software Engineer

AGENDA

WHAT ?

WHEN ?

WHY ?

HOW ?

WHERE ?

What?

What are resources ?

show running configuration

```
interface FastEthernet0
 ip address dhcp
 ip nat outside
 ip virtual-reassembly in
 duplex auto
 speed auto
!
interface FastEthernet1
 ip address dhcp
 ip nat outside
 ip virtual-reassembly in
 duplex auto
 speed auto
!
.....
.....
.....
access-list 102 permit udp any any eq domain
access-list 102 permit udp any eq domain any
access-list 102 permit tcp any any eq domain
access-list 102 permit tcp any eq domain any
.....
.....
!
.....
.....
ospf 1
 filter-policy 2002 export static
 import-route static
 area 0.0.0.0
  network 192.168.1.0 0.0.0.255
!
ip route-static 172.16.16.0 255.255.255.0 NULL0
ip route-static 172.16.17.0 255.255.255.0 NULL0
!
```

os_interfaces

os_acls

os_ospfv2

os_static_routes

Resources

What?

What are Resources Modules ?

- A “Resource Module” is defined as a **specific discrete network function** mapped to a single Ansible module.

- They can **read and configure** a specific network service on a network device” (the scope comprises a single resource only) and can be combined for the configuration of complex network setups.

When ?

Introuduced in :

Ansible 2.9

Before RM :

command module : runs arbitrary commands on the device

```
- name: ensure that the desired snmp strings are present
  ios_config:
    commands:
        - snmp-server community ansible-public RO
        - snmp-server community ansible-private RW
```

config module : pushes configuration commands on to the device

(jinja template can be used for configuration commands)

```
- name: ensure that the desired snmp strings are present
  ios_config:
    config: "{{ lookup('template', 'my_template.j2') }}"
```

facts module : Fact modules return structured data about the network device

```
- name : grab interfaces info
  eos_facts:
    gather_subset: min
    gather_network_resources: interfaces
```

Why ?

In older modules

- Gathering of all facts was tedious and laborious.
- Maintenance of overly complex jinja template files was tough
- Normalization of data was missing.

In Resource Modules

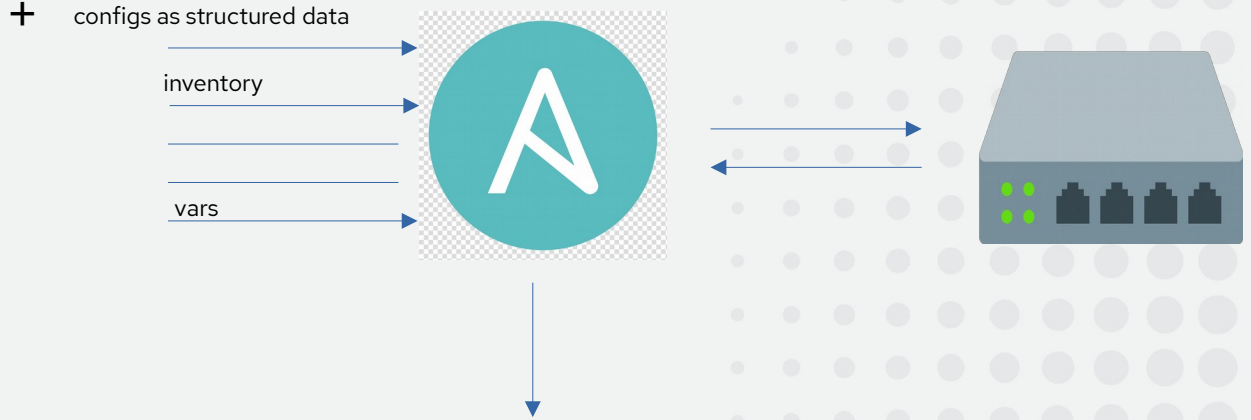
- Gathering of facts and normalization of data made easier
- Platform agnostic modules.
- Focus is shifted to specific resources
- Human-readable and well structured data is provided as source of truth.
- Data driven (Native <---> Structured data)

How ?

STATE

- **merged**
- **replaced**
- **overridden**
- **deleted**
- **gathered**
- **rendered**
- **parsed**

+



Return Payload:

- **Before** : Configuration prior to module execution
- **After** : Configuration after module execution
- **Commands**: commands sent to the device
- **Changed** : True if any configuration changes are made on the device , else False
-

Merged: Merges the given configuration with the on device configuration.

```
arista.eos.eos_acls:
  config:
    - afi: "ipv4"
      acls:
       - name: acl01
         aces:
           - sequence: 35
             grant: "deny"
             protocol: "ospf"
             source:
               subnet_address: 20.0.0.0/8
             destination:
               any: true
           - remark: "Run by ansible"
           - grant: "permit"
             protocol: "6"
             source:
              any: true
             destination:
              any: true
  state: merged
```

How ?

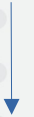
```
"after": [
    {
        "acls": [
            {
                "aces": [
                    {
                        "destination": {
                            "any": true
                        },
                        "grant": "deny",
                        "protocol": "ospf",
                        "sequence": 35,
                        "source": {
                            "subnet_address": "20.0.0.0/8"
                        }
                    },
                    {
                        "remark": "Run by ansible",
                        "sequence": 45
                    },
                    {
                        "destination": {
                            "any": true
                        },
                        "grant": "permit",
                        "protocol": "tcp",
                        "sequence": 55,
                        "source": {
                            "any": true
                        }
                    }
                ],
                "name": "acl01"
            }
        ],
        "afi": "ipv4"
    }
],
"before": [],
"changed": true,
"commands": [
    "ip access-list acl01",
    "35 deny ospf 20.0.0.0/8 any",
    "remark Run by ansible",
    "permit tcp any any"
],
```

Replaced: Replaces on-device configuration subsection with the provided configuration subsection in the task.

How ?

```
arista.eos.eos_acls:
    config:
      - afi: "ipv4"
        acls:
         - name: acl01
           aces:
             - sequence: 11
               grant: "permit"
               protocol: "tcp"
               source:
                 subnet_address: 20.0.0.0/8
               destination:
                 any: true
    state: replaced
```

```
ip access-list acl01
   35 deny ospf 20.0.0.0/8 any
   45 remark Run by ansible
   55 permit tcp any any
!
ip access-list acl02
   20 permit ospf any any
```

```
"changed": true,
"commands": [
    "ip access-list acl01",
    "no 35",
    "no 45",
    "no 55",
    "11 permit tcp 20.0.0.0/8 any"
],
```

Overridden: Overrides the on-device configuration for the resource with the provided configuration in the task.

How ?

```
arista.eos.eos_acls:
  config:
    - afi: "ipv4"
      acls:
       - name: acl01
         aces:
           - sequence: 25
             grant: "permit"
             protocol: "icmp"
             source:
               subnet_address: 20.0.0.0/8
             destination:
               any: true
  state: overridden
```

```
ip access-list acl01
   35 deny ospf 20.0.0.0/8 any
   45 remark Run by ansible
   55 permit tcp any any
!
ip access-list acl02
   20 permit ospf any any
```



```
"changed": true,
   "commands": [
       "ip access-list acl01",
       "no 35",
       "no 45",
       "no 55",
       "25 permit icmp 20.0.0.0/8 any",
       "no ip access-list acl02",
   ],
```

Deleted: Deletes on device configuration based on the given configuration.

```
arista.eos.eos_acls:
    config:
      - afi: "ipv4"
        acls:
         - name: acl01
         - name: acl02
    state: deleted
```

How ?


```
"after": [],
"before": [
    {
        "acls": [
            {
                "aces": [
                    {
                        "destination": {
                            "any": true
                        },
                        "grant": "deny",
                        "protocol": "ospf",
                        "sequence": 35,
                        "source": {
                            "subnet_address": "20.0.0.0/8"
                        }
                    },
                    {
                        "remark": "Run by ansible",
                        "sequence": 45
                    },
                    {
                        "destination": {
                            "any": true
                        },
                        "grant": "permit",
                        "protocol": "tcp",
                        "sequence": 55,
                        "source": {
                            "any": true
                        }
                    }
                ],
                "name": "acl01"
            }
        ],
        "afi": "ipv4"
    }
],
"changed": true,
"commands": [
    "no ip access-list acl01",
    "no ip access-list acl02"
],
```

Gathered: Displays the resource details gathered from the network device and accessed with the gathered key in the result.

How ?

```
arista.eos.eos_acls:
  state: gathered
```


```
"gathered": [
    {
        "acls": [
            {
                "aces": [
                    {
                        "destination": {
                            "any": true
                        },
                        "grant": "deny",
                        "protocol": "ospf",
                        "sequence": 35,
                        "source": {
                            "subnet_address": "20.0.0.0/8"
                        }
                    },
                    {
                        "remark": "Run by ansible",
                        "sequence": 45
                    },
                    {
                        "destination": {
                            "any": true
                        },
                        "grant": "permit",
                        "protocol": "tcp",
                        "sequence": 55,
                        "source": {
                            "any": true
                        }
                    }
                ],
                "name": "acl01"
            }
        ],
        "afi": "ipv4"
    }
],
```

Rendered: Renders the provided configuration in the task in the device-native format . **(No connection to device)**

How ?

```
arista.eos.eos_acls:
  config:
    - afi: "ipv4"
      acls:
       - name: acl01
         aces:
           - sequence: 35
             grant: "deny"
             protocol: "ospf"
             source:
               subnet_address: 20.0.0.0/8
             destination:
               any: true
           - remark: "Run by ansible"
           - grant: "permit"
             protocol: "6"
             source:
              any: true
             destination:
              any: true
  state: rendered
```



```
"rendered": [
    "ip access-list acl01",
    "35 deny ospf 20.0.0.0/8 any",
    "remark Run by ansible",
    "permit tcp any any"
]
```

Parsed: Parses the configuration from the running_configuration option into Ansible structured data in the parsed key in the result.
(No connection to device)

How ?

```
ip access-list acl01
   35 deny ospf 20.0.0.0/8 any
   45 remark Run by ansible
   55 permit tcp any any
!
ip access-list acl02
   20 permit ospf any any
```

```
arista.eos.eos_acls:
    running_config: "{{ lookup('file', './parsed_acls.cfg') }}"
    state: parsed
```

Round trip configuration :

```
arista.eos.eos_facts:
    gather_network_resources: acls
```

Where ?

```
arista.eos.eos_acls:
    config:

      - afi: ipv4
        acls:

          - name: test3
            aces:

              - sequence: 100
                grant: permit
                protocol: icmp
                source:
                  any: true
                destination:
                  any: true
                log: true
```

```
arista.eos.eos_acls:
    config: "{{ ansible_facts['network_resources']['acls'] }}"
    state: overridden
```

Combine RMs to seamlessly configure various services:

Where ?

```
---
- hosts: eos
  gather_facts: false
  tasks:
    - name: Configure interfaces
      eos_interfaces:
        config: "{{ interfaces }}"

    - name: Configure l2 interfaces
      eos_interfaces:
        config: "{{ l2_interfaces }}"

    - name: Configure l3 interfaces
      eos_interfaces:
        config: "{{ l3_interfaces }}"

    - name: Configure vlans
      eos_interfaces:
        config: "{{ vlans }}"
```

Resources:

Where ?

- Ansible for Network Automation
- Network Resource Modules

- IRC: #ansible-network on irc.freenode.net
- Ansible network Slack Channel
- Mailing list

Other talks/labs:

- *Ansible network Automation Workshop* by Sean Cavanaugh
- *A simplified approach to building Network Resource Modules* by Nilashish Chakraborty

Thank you

Red Hat is the world’s leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make
Red Hat a trusted adviser to the Fortune 500.


youtube.com/user/RedHatVideos


linkedin.com/company/Red-Hat


facebook.com/ansibleautomation


twitter.com/ansible