Five great use cases with Ansible Network Collections

Ansiblefest 2020

Sean Cavanaugh

Technical Marketing Engineer

AnsibleFest

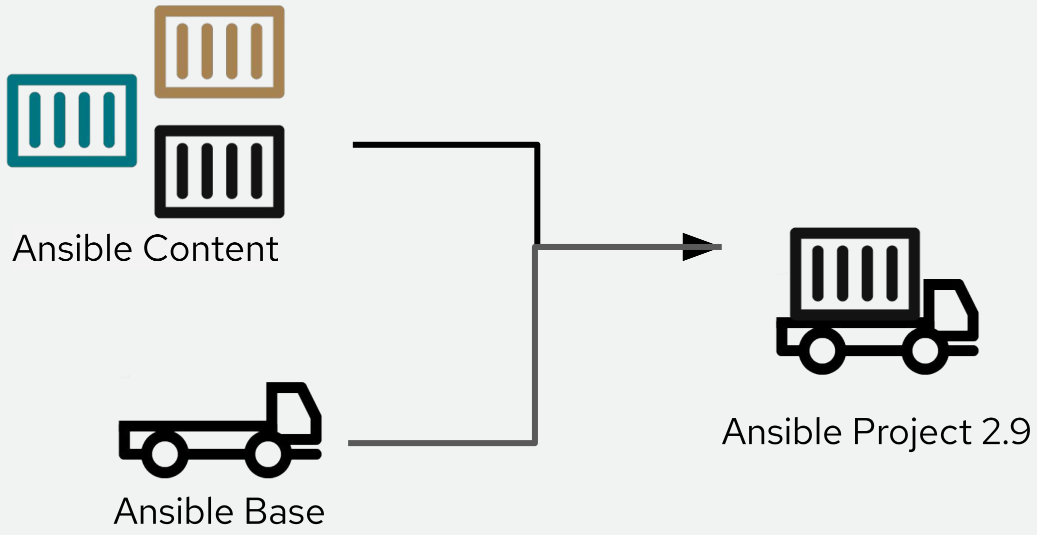
Agenda

What are we even talking about?

- What is an Ansible Collection? Why do I care?
- How do I install and use it?
- Five great use-cases using
  the Ansible Network Collection
- Where do I go next?

What is an Ansible Collection?

Content is now modular



Where do I get it?

Ansible Galaxy

galaxy.ansible.com

- Community supported
- Extended to leverage Collections framework
- “Latest and greatest”

Ansible Automation Hub

cloud.redhat.com

- Certified, jointly supported by Red Hat and Partner
- Access to advanced analytics
- “Slow and steady”

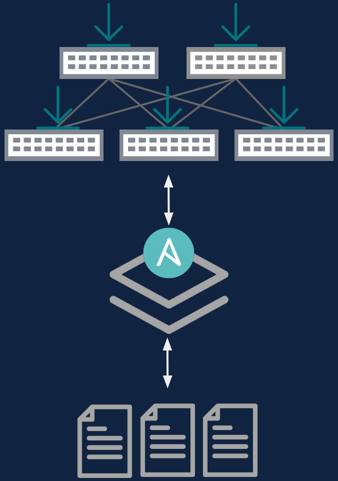
How do I install it?

Install an Ansible Collection:

```
ansible-galaxy collection install cisco.ios
```

This installs (by default) into:

```
~/.ansible/collections/ansible_collections
```



Backup and Restore

Why is it important?

- Read-only, no changing of production configs
- Ubiquitous use case
- Easy scheduling in Ansible Automation Platform

Fully platform agnostic backups

```
- name: backup config
  cisco.ios.ios_config:
    backup: true
```

```
- name: backup config
  arista.eos.eos_config:
    backup: true
```

```
- name: backup config
  junipernetworks.junos.junos_config:
    backup: true
```

Fully platform agnostic ~~backups~~ restore

Cisco IOS-XE

```
- name: restore config
  cisco.ios.ios_config:
    src: "{{inventory_hostname}}"
```

Arista EOS

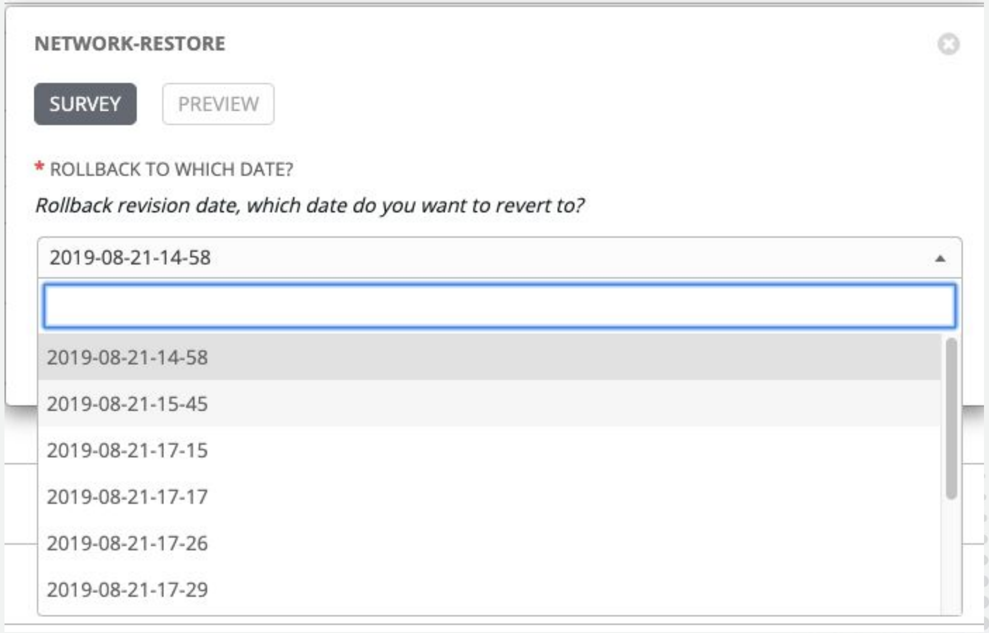
```
- name: restore config
  arista.eos.eos_config:
    replace: config
    src: "{{inventory_hostname}}"
```

Juniper Junos

```
- name: restore config
  junipernetworks.junos.junos_config:
    update: replace
    src: "{{inventory_hostname}}"
```

Elevate Tasks via Surveys

- No network platform specific knowledge required!
- Automate the routine and boring tasks away
- Empower novices to take on more activities with guardrails



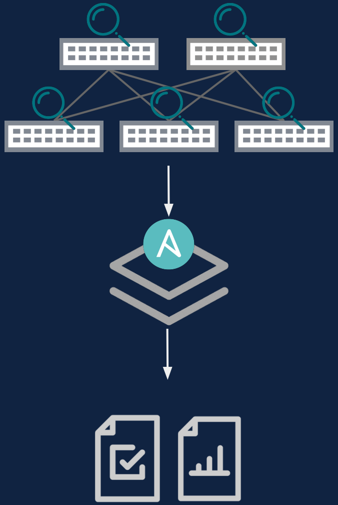
* ROLLBACK TO WHICH DATE?

Rollback revision date, which date do you want to revert to?

2019-08-21-14-58

2019-08-21-14-58
2019-08-21-15-45
2019-08-21-17-15
2019-08-21-17-17
2019-08-21-17-26
2019-08-21-17-29

AnsibleFest



Fact Collection

Why is it important?

- Read-only, no changing of production configs
- Normalizes configs into structured data
- Builds reports for easy consumption

Resource modules - state parameters

- **parsed**:
  Reads the configuration from running_config option and transforms it into JSON
- **gathered:**
  retrieve facts for single resource
- **rendered**:
  transforms the configuration in config option to platform specific CLI commands

Resource Modules - parsed

```
interface Loopback0
 ip address 192.168.1.101 255.255.255.0
interface Loopback1
 ip address 10.1.1.101 255.255.255.0
interface Loopback2
 ip address 10.15.1.1 255.255.255.255
interface Tunnel0
 ip address 10.100.100.1 255.255.255.0

<<output removed for slide brevity>>
```

Resource Module

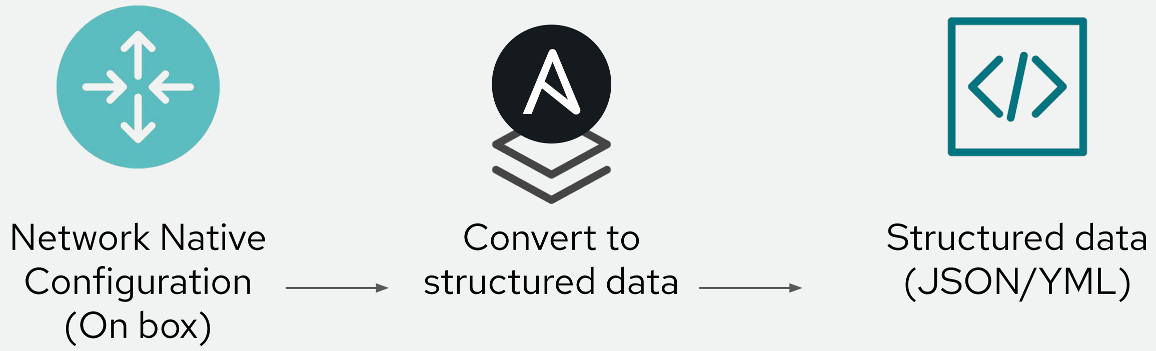
```
- name: loopback0
  ipv4:
  - address: 192.168.1.101 255.255.255.0
- name: loopback1
  ipv4:
  - address: 10.1.1.101 255.255.255.0
- name: loopback2
  ipv4:
  - address: 10.15.1.1 255.255.255.255
- name: GigabitEthernet1
  ipv4:
  - address: dhcp
```

Backup Configuration (no active device connection)

Example resource module **ios_l3_interfaces**

JSON parsed configuration

Resource Modules - gathered


Network Native Configuration (On box)
Convert to structured data
Structured data (JSON/YML)

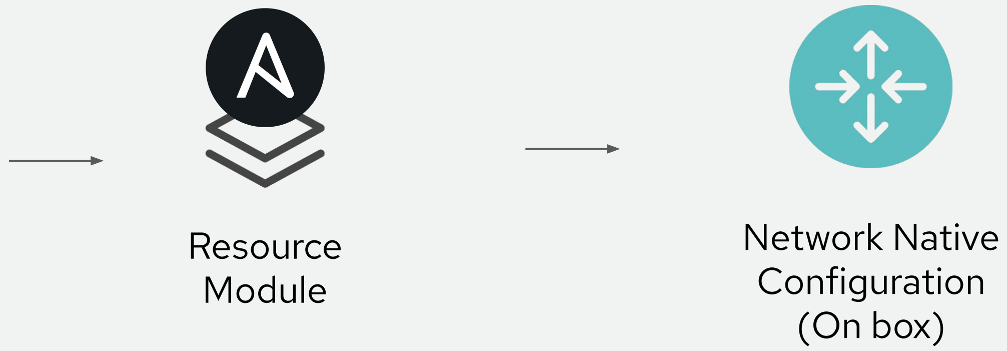
Retrieve single resource

```yaml
---
- hosts: cisco
  gather_facts: false
  tasks:

  - name: grab info
    cisco.ios.ios_l3_interfaces:
      state: gathered
    register: l3_info

  - name: push structured data to hostvars
    debug:
      msg: "{{l3_info.gathered}}"
```

```yaml
- name: loopback0
  ipv4:
  - address: 192.168.1.101 255.255.255.0
- name: loopback1
  ipv4:
  - address: 10.1.1.101 255.255.255.0
- name: loopback2
  ipv4:
  - address: 10.15.1.1 255.255.255.255
- name: GigabitEthernet1
  ipv4:
  - address: dhcp
```

Resource Modules - rendered

```yaml
interfaces:
- enabled: true
  name: Ethernet1
  mtu: '1476'
- enabled: true
  name: Loopback0
- enabled: true
  name: Loopback1
- enabled: true
  mtu: '1476'
  name: Tunnel0
- enabled: true
  name: Ethernet1
```



Example structured data
interfaces

Example resource module
eos_interfaces

Example Rendered configuration

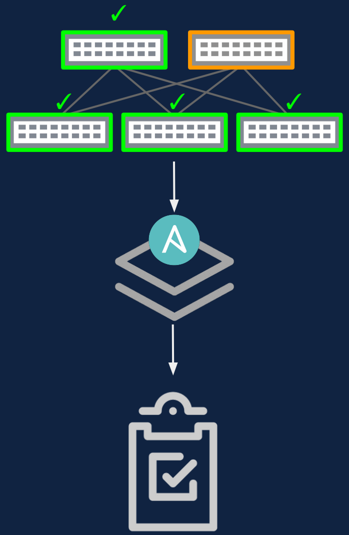
```
interface Tunnel0
   mtu 1476
!
```

AnsibleFest

render network device commands

```yaml
---
- hosts: cisco
  gather_facts: false
  tasks:

  - name: render commands
    cisco.ios.ios_l3_interfaces:
      config: "{{ l3_interfaces }}"
      state: rendered
    register: l3_info

  - name: display commands
    debug:
      msg: "{{l3_info.rendered}}"
```

```
msg:
 - interface loopback0
 - ip address 192.168.1.101 255.255.255.0
 - interface loopback1
 - ip address 10.1.1.101 255.255.255.0
 - interface loopback2
 - ip address 10.15.1.1 255.255.255.255
 - interface GigabitEthernet1
 - ip address dhcp
```



Config Management

Why is it important?

- Enforces configuration policy
- Corrects configuration drift
- Forces multiplier for config changes
- Locks down configs to known good “golden masters”

Resource module config management

YAML variables

```
ip_address_info:
  - name: Loopback100
    ipv4:
    - address: 10.10.10.1/24
    ipv6:
      - address: fc00::100/64
      - address: fc00::101/64
  - name: Loopback200
    ipv4:
    - address: 10.10.20.1/24
```

Ansible Playbook Task

```
- name: ensure that the IP address information is accurate
  cisco.ios.ios_l3_interfaces:
    config: "{{ ip_address_info }}"
    state: merged
```

Ansible Playbook output

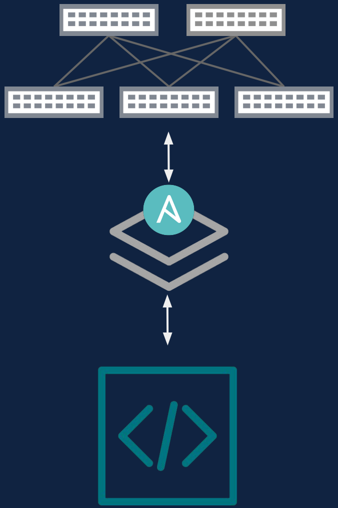
```
[student1@ansible ~]$ ansible-playbook ip_address.yml

PLAY [rtr2]
*************************************************************

TASK [ensure that the IP address information is accurate]
***********************
changed: [rtr2]

PLAY RECAP
********************
rtr2   ok=1     changed=1     unreachable=0    failed=1
skipped=0     rescued=0     ignored=0
```

Resource modules - state parameters for config mgmt

- **merged:** configuration merged with the provided configuration (**default**)
- **replaced**: configuration of provided resources will be replaced with the provided configuration
- **overridden**: The configuration of the provided resources will be replaced with the provided configuration, extraneous resource instances will be removed
- **deleted**: The configuration of the provided resources will be deleted/defaulted

Resource modules - return values

- **before**
  The configuration prior to module execution is always returned.
- **commands**
  delta command set for the device
- **after**
  the configuration post module execution



Creating a Source of Truth

Why is it important?

- Incremental steps to infrastructure as code
- Structured variables for network config
- Simple and agnostic data model

Create a structured SOT (source of truth)

Network Native Configuration (on box) → Convert to structured data → Store in host_vars (off box)

Convert facts into flat-file variables

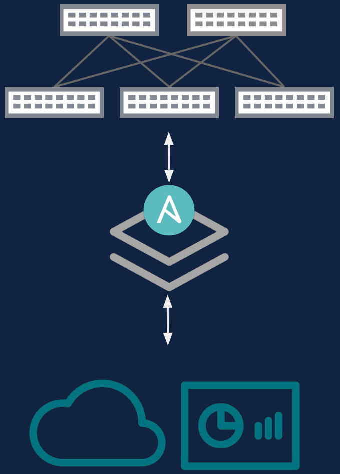
```
---
- hosts: cisco
  gather_facts: false
  tasks:

    - name: grab info
      cisco.ios.ios_facts:
        gather_subset: min
        gather_network_resources: all

    - name: push structured data to hostvars
      copy:
        content: "{{ansible_network_resources | to_nice_yaml}}"
        dest: "{{playbook_dir}}/host_vars/{{inventory_hostname}}"
```

Convert facts into flat-file variables

```
[user@ansible ~]$ cat host_vars/rtr2
```

```
interfaces:
- enabled: true
  name: Ethernet1
  mtu: '1476'
- enabled: true
  name: Loopback0
- enabled: true
  name: Loopback1
- enabled: true
  mtu: '1476'
  name: Tunnel0
vlans:
- name: None
  vlan_id: 2
- name: None
  vlan_id: 100
- name: None
  state: suspend
  vlan_id: 5
<... rest of output removed for brevity...>
```

- Each resource is a list of dicts
- The key is the resource (e.g. interfaces, vlans)
- Resources that are not used will show empty (e.g. lacp: {})

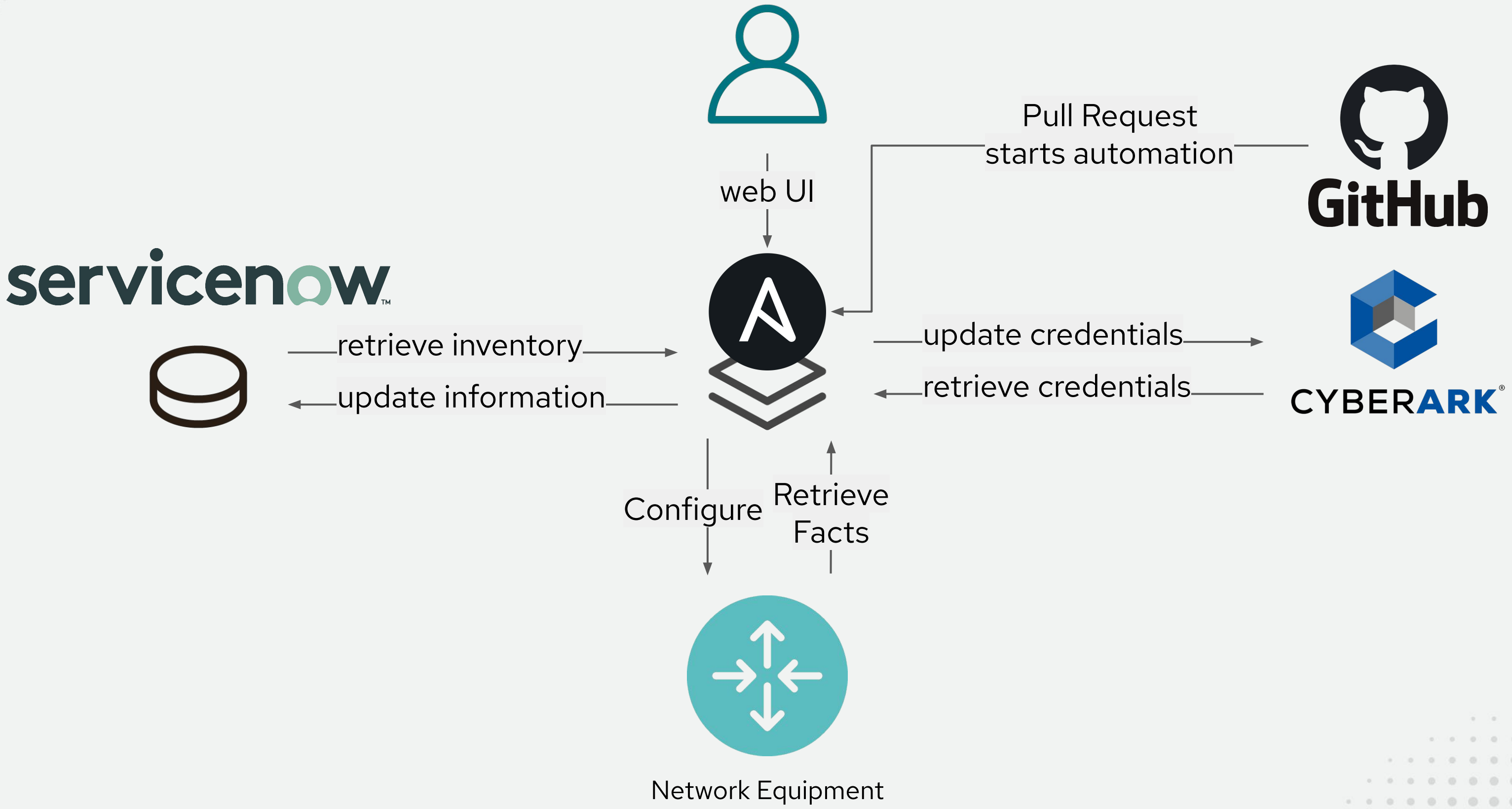


Third-party Integrations

Why is it important?

- Existing networks have existing tools
- Ansible Automation Platform API
- Ansible Automation Platform has “inbox” integrations

AnsibleFest


web UI
Pull Request
starts automation
GitHub
servicenow
retrieve inventory
update information
update credentials
retrieve credentials
CYBERARK
Configure
Retrieve
Facts
Network Equipment

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/AnsibleAutomation

linkedin.com/company/Red-Hat

facebook.com/ansibleautomation

twitter.com/ansible

AnsibleFest