How to use this deck

Name: Webinar - Ansible Network Meta Collection

Purpose: This deck is for the webinar scheduled June 22nd, 2021 (check ansible.com/webinars) around Ansible Network Automation and the release of the new fully supported Ansible Network Meta Collection. This can be re-used by whomever wants to cover Ansible Network Meta Collection but please refer to One Stop for newest Ansible Network Automation material. Webinars are a moment in time.

Last updated: June 16, 2021

What this deck is for? Webinar Announcement of Ansible Network Meta Collection (fully supported by Red Hat with your Ansible Automation Platform subscription)

What this deck is not for? Sales conversations

Google Slides source link (Red Hat internal): https://docs.google.com/presentation/d/14UONxwd2VVyY1nzaBXXu.MenuItem08c-LO27t0H_Klb4/edit?usp=sharing

Owner: Ansible / MBU, ansible-pmm-tmm@redhat.com
Sean Cavanaugh & Sumit Jaiswal

List of all official Ansible content: Red Hat Ansible Automation Platform One Stop: https://redhat.highspot.com/items/5966647572ad8e2077bc270?frrm=srp.10
Ansible Network Meta Collection
and more!

Sean Cavanaugh
Principal Technical Marketing Manager
@IPvSean

Sumit Jaiswal
Senior Software Engineer
@JustSumit
Meet your speakers

presenting to you today are......

Sean Cavanaugh - @IPvSean

- Location: Chapel Hill, NC, USA
- Interests: electric bicycles, bourbon, running, Star Wars, gardening
- Interesting Fact: More people are familiar with my dog Lexi than me at Red Hat 🐶

Sumit Jaiswal - @justsumit/@justjais

- Location: New Delhi, Delhi, India
- Interests: Long Drive, Binge watching web-series, playing with my year old daughter.
- Interesting Fact: Having footprints all over in Ansible content
Agenda

Ansible Network Meta Collection

- What is Ansible Network Automation?
- What is a collection?
  - What is the Ansible Network Meta Collection?
- Sample use-cases
  - Infrastructure Awareness
  - Scoped Config Management
  - Operational State Validation
- Where to go next?
- Demos by Sumit
What makes a platform?

Red Hat Ansible Automation Platform

Combining the universal automation language with cloud services and certified content for automating, deploying, and operating applications, infrastructure and services securely at enterprise scale.

Ansible automation
Providing scalable, secure implementation for describing, building, and managing the deployment of enterprise IT applications across diverse enterprise architectures.

Cloud services
Cloud services that facilitate team collaboration and provide operational analytics for automating heterogeneous, hybrid environments.

Certified content
Extends native platform capabilities with certified, supported content designed to expand the automation domain and accelerate adoption for enterprise customers.
What is Ansible Network Automation?

Ansible network automation is our content domain focused on networking use cases. The goal is to provide network teams with the tools and an operational framework to implement next-generation network operations, manage network infrastructure-as-code, and better support digital transformation by connecting teams across the IT organization.

Ansible network automation is a set of Certified Content Collections designed to streamline and operationalize network operations across multiple platforms and vendors.
Ansible Network Ecosystem

- Switches
- Routers
- Enterprise Firewalls
- Load Balancers
- Controllers
- IP Address Mgmt
Ansible Collections

Simplified and consistent content schema

- A standardized way to organize and package Ansible content
- Portable and flexible delivery
- Semantic versioning

YouTube Video of Ansible Collections: https://red.ht/ansible_collection
What is in an Ansible Collection?

- Components are well defined, there is a standard for the directory structure
- Requires same standard of documentation that the Ansible Project does
- Scaffolding can be created with Ansible Galaxy command

YouTube Video of Ansible Collections: [https://red.ht/ansible_collection](https://red.ht/ansible_collection)
“but....
what the heck is a meta collection? “

Craig Brandt
Sean’s coworker in Dublin, Ireland
What is the problem we are trying to solve?

What works with what, and how do you install it all?

- ansible-galaxy collection install cisco.ios:==2.0.1
- ansible-galaxy collection install arista.eos:==2.1.1
- ansible-galaxy collection install junipernetworks.junos:==2.1.1
- ansible-galaxy collection install ansible.utils:==2.1.0
Meta Collection

Fully supported content wrapped up in one collection

network collection  network collection  network collection
Introducing... The Ansible Network Meta Collection

Fully supported network automation content packaged in one collection

- Arista: arista.eos, cisco.ios, cisco.iosxr, cisco.nxos
- Cisco: cisco.ios, cisco.iosxr, cisco.nxos
- Juniper Networks: junipernetworks.junos
- Ansible: ansible.netcommon, ansible.utils
- Red Hat: frr.frr
- Open vSwitch: openvswitch.openvswitch
- VyOS: vyos.vyos

Traditional networking platforms | Red Hat provided utilities | Open networking platforms
Network Automation Modules

How do we interact with network devices?

- **command**: run arbitrary commands
- **facts**: retrieve information
- **config**: generic catch-all configuration and templating
- **resource**: read and configure specific network resources
Network Automation Modules

How do we interact with network devices?

- **command**
  - namespace.collection.command
  - Cisco IOS -> cisco.ios.command

- **facts**
  - namespace.collection.facts
  - Arista EOS -> arista.eos.facts

- **config**
  - namespace.collection.config
  - Juniper Junos -> junipernetworks.junos.config

- **resource**
  - namespace.collection.module
  - Cisco IOS-XR -> cisco.iosxr.acls
Network resource modules
Managing device state across different devices and types

Configuration to code

- Built-in logic with commands and orchestration
- Vendor-agnostic data model
- Bidirectional with configuration to facts and facts to configuration
Sample use-cases

Quick automation victories for network engineers

**Infrastructure Awareness**
- Use Ansible facts to gain information
  - Read-only, no production config change
  - Dynamic Documentation and reporting
  - Understand your network

**Scoped Config Management**
- Focus on high yield victories
  - Automate VLANs, ACLs and SNMP config
  - Introduce source of truth concepts
  - Enforce Configuration policy

**Operational State Validation**
- Going beyond config management
  - Parsing operational state to structured values
  - Schema validation and verification
  - Enhance operational workflows
Infrastructure Awareness

Why is it important?
- Read-only, no changing of production configs
- Normalizes configs into structured data
- Dynamic Documentation and reporting

Why Ansible Automation Platform?
- Huge ecosystem of tools
- Standard data models across multiple platforms
- Constantly expanding fact capabilities
What are facts?
Structured data, the Ansible way

```plaintext
Cisco IOS XE Software, Version 16.09.02
Cisco IOS Software [Fuji], Virtual XE Software
(X86_64_LINUX_IOSD-UNIVERSALK9-M), Version 16.9.2,
RELEASE SOFTWARE (fc4)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2018 by Cisco Systems, Inc.

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

```plaintext
Cisco# show version
Cisco IOS XE Software, Version 16.09.02
Cisco IOS Software [Fuji], Virtual XE Software
(X86_64_LINUX_IOSD-UNIVERSALK9-M), Version 16.9.2,
RELEASE SOFTWARE (fc4)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2018 by Cisco Systems, Inc.

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

```
cisco# ansible -m ios_facts cisco
cisco | SUCCESS => {
  "ansible_facts": {
    "ansible_net_iostype": "IOS-XE",
    "ansible_net_version": "16.09.02",
    "ansible_net_serialnum": "9L8KQ482JFZ",
    "ansible_net_model": "CSR1000V",
  }
}
```

```
<<rest of output removed for slide brevity>>
```
Ansible Automation Platform facts

Network automation begins and ends with **facts**

Network native configuration → Convert to structured data

```yaml
ansible_facts:
  ansible_net_api: cliconf
  ansible_net_fqdn: rtr2
  ansible_net_gather_network_resources:
    - interfaces
  ansible_net_gather_subset:
    - default
  ansible_net_hostname: rtr2
  ansible_net_image: flash:EOS.swi
  ansible_net_model: vEOS
  ansible_net_python_version: 2.7.5
  ansible_net_serialnum: D00E130991A37B49F970714D8CCF7FCB
  ansible_net_system: eos
  ansible_net_version: 4.22.0F
  ansible_network_resources:
    interfaces:
      - enabled: true
        name: Ethernet1
      - enabled: true
        name: Loopback0

<<rest of output removed for slide brevity>>
```
Structured data is flexible
Create customized network reports

ansible_facts:
  ansible_net_api: cliconf
  ansible_net_fqdn: rtr2
  ansible_net_gather_network_resources:
    - interfaces
  ansible_net_gather_subset:
    - default
  ansible_net_hostname: rtr2
  ansible_net_image: flash:EOS.swi
  ansible_net_model: vEOS
  ansible_net_python_version: 2.7.5
  ansible_net_serialnum: D00E130991A37B49F970714D0CCF7FCB
  ansible_net_system: eos
  ansible_net_version: 4.22.0F
  ansible_network_resources:
    interfaces:
      - enabled: true
        name: Ethernet1
      - enabled: true
        name: Loopback0

<<rest of output removed for slide>>
Ansible Automation Platform

Dynamic Documentation

Ansible Network Automation Example Report

<table>
<thead>
<tr>
<th>Hostname</th>
<th>Transport</th>
<th>Platform</th>
<th>Code Version</th>
<th>Serial Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>switch</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>VLANs</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>vlan_id</td>
<td>Name</td>
<td>state</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>dmz</td>
<td>active</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>vlp</td>
<td>active</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>desktops</td>
<td>active</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td></td>
<td>servers</td>
<td>active</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td></td>
<td>printers</td>
<td>active</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td></td>
<td>DMZ</td>
<td>active</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Interfaces: MTU/Duplex/Speed</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>network cli</td>
<td>Arista eos</td>
<td>4.22.1FX-VEOSRouter-cloud</td>
<td>43C8BF871A2BFC183190CA1C77A81C88A</td>
</tr>
<tr>
<td></td>
<td>Ethernet1</td>
<td>none</td>
<td>default</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Duplex</td>
<td>Enabled</td>
<td>MTU</td>
</tr>
<tr>
<td></td>
<td></td>
<td>default</td>
<td>True</td>
<td>default</td>
</tr>
<tr>
<td></td>
<td>LLDP Interfaces</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>L2 Interfaces - Trunk/Access Ports</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>L3 Interfaces - IP Addresses</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>LACP</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Scoped Config Management

Why is it important?
- Automate VLANs, ACLs and SNMP config
- Introduce source of truth concepts
- Enforce Configuration policy

Why Ansible Automation Platform?
- Resource Modules provide easy automation journey
- No complex templating or coding requirements
- Simple native data models
Scoped Config Management

Source of Truth
Examples: Github, Infoblox, Device42, NetBox

Network Operator
- configure network resource

Ansible Automation Platform
- deploy automation

Arista EOS
Cisco IOS-XE
Juniper Junos
Manage specific network resources

vlans:
- name: desktops
  vlan_id: 20
- name: servers
  vlan_id: 30
- name: printers
  vlan_id: 40
- name: DMZ
  vlan_id: 50
Managing device state

Practical examples of using network resource modules

vlans:
- name: desktops
  vlan_id: 20
- name: servers
  vlan_id: 30
- name: printers
  vlan_id: 40
- name: DMZ
  vlan_id: 50

- name: add VLAN configuration
  arista.eos.vlans:
  config: "{{ vlans }}"
  state: merged

State:
- Merged
- Replaced
- Overridden
- Deleted
Understanding state parameters

**state: merged**

### Existing config

```
# sh run | s vlan
vlan 5
  name desktops
  state suspend
!
vlan 10
  name servers
!
vlan 50
  name voip
```

### YAML Source of Truth

```
vlan:
  - name: desktops
    vlan_id: 5
  - name: servers
    vlan_id: 10
  - name: dmz
    vlan_id: 20
```

### New Config

```
# sh run | s vlan
vlan 5
  name desktops
  state suspend
!
vlan 10
  name servers
!
vlan 20
  name dmz
!
vlan 50
```

### Ansible task

```
- name: add VLAN configuration
  arista.eos.vlans:
    config: "{{ vlans }}"
    state: merged
```
Ansible Tower

Operationalizing scoped configuration

- Push button deployment through WebUI allows non-playbook writers to execute automation with your configured RBAC

- Surveys allow parameterization of values into simple web forms for easy guard-rails to help users adopt automation
Scoped Config Management
Where to start with network resources for quick automation wins

ACLs  Prefix-Lists  Route-Maps  VLANs

Network Operator → deploy automation → Ansible Automation Platform → VLANs
Operational State Validation

Why is it important?

- Parsing operational state to structured values
- Schema validation and verification
- Enhance operational workflows

Why Ansible Automation Platform?

- Rollback is built-in on a module by module basis
- Workflows allow flexible scenarios to be automated
- Operational State can be treated just like Ansible Facts
Operational State Validation
Checking network state programmatically like facts
Operational state versus config
What is configured, versus what the network state is

# sh run int Ethernet1/1
interface Ethernet1/1
   no switchport
   ip address 10.101.101.2/24
   ip ospf network point-to-point
   ip ospf area 0.0.0.0

# show ip ospf neigh
Neighbor ID     Instance VRF      Pri State                  Dead Time   Address         Interface
192.168.4.4     1        default  0   FULL                   00:00:31    10.101.101.4    Ethernet1/1

OSPF Interface Configuration
OSPF Operational State
Ansible Utils

Tool chest for playbook authors

- An Ansible Collection with utilities to ease the management, manipulation, and validation of data within a playbook

- Fully supported in Automation Hub on https://cloud.redhat.com

- Included in Ansible Network Meta Collection ansible.network
Ansible Utils

Included content in `ansible.utils` collection

Filter plugins
- **manipulate data**
  - Example: `ansible.utils.from_xml`
  - Convert XML to Python Dictionary

Lookup plugins
- **retrieve data model information**
  - Example: `ansible.utils.to_paths`
  - Flatten complex object into dictionary of paths and values

Modules
- **perform action on inventory**
  - Example: `ansible.utils.cli_parse`
  - Parse CLI output or text using a variety of parsers

Test plugins
- **test data to verify type**
  - Example: `ansible.utils.private`
  - Test if an IP address is private RFC 1918
Use parser of choice

Tons of examples and pre-built content

- **native** The native parsing engine built into Ansible and requires no additional Python libraries.
- **xml** Convert XML to an Ansible native data structure.
- **textfsm** A Python module which implements a template-based state machine for parsing semi-formatted text.
- **ntc_templates** Predefined textfsm templates packages supporting a variety of platforms and commands.
- **ttp** A library for semi-structured text parsing using templates, with added capabilities to simplify the process.
- **pyats** Uses the parsers included with the Cisco Test Automation & Validation Solution.
- **jc** A Python module that converts the output of dozens of popular Linux/UNIX/macOS/Windows commands and file types to Python dictionaries or lists of dictionaries. Note: this filter plugin can be found in the [community.general] collection.
- **json** Converts JSON output at the CLI to an Ansible native data structure.
What should I parse?

Quick automation victories for operational state

- Interface State
- LLDP Neighbors
- OSPF Neighbors
- BGP Neighbors

Network Operator → parse automation → Ansible Automation Platform

Red Hat
Learning resources

Continue your automation journey with Red Hat® Ansible® Network Automation

- Technical ebook: Automate your network with Red Hat
  https://red.ht/automate_your_network
- Networking workshop
  github.com/ansible/workshops
- Deep dive into resource modules, Trishna Guha
  ansible.com/deep-dive-into-ansible-network-resource-module
- Red Hat Certification
  Ansible for Network Automation (DO457)
  redhat.com/en/services/training/do457-ansible-network-automation
Thank You

Ansible is a universal language, unraveling the mystery of how work gets done. Turn tough tasks into repeatable playbooks. Roll out enterprise-wide protocols with the push of a button.