Migrating to Ansible Collections

December 8, 2020

Andrius Benokraitis
Product Management
@AndriusB

Sean Cavanaugh
Technical Marketing
@IPvSean

Anshul Behl
Partner Engineering
@Anshul_Behl

Red Hat
Ansible Automation Platform
What we’ll discuss today

- Ansible basics, teeing it up
- Getting ready for Collections
- Examples of using Collections in 2.9
- Debugging and Troubleshooting
- Resources and Next Steps
Who uses Ansible?

Spoiler: Lots of different (and siloed) personas

IT Operations
Enterprise-wide automation requires attention to managing multi-site inventory, RBAC, and security at scale.

Platform Developers
These are the “plumbers” that ensure endpoints are automatable, and maintain the Ansible modules, plugins, roles (content) to be used.

Automation Implementers
Ansible playbook writers stitching together each play by play, task by task using tested and validated released developer content.
Who uses Ansible?

Spoiler: Lots of different (and siloed) personas

IT Operations
Enterprise-wide automation requires attention to managing multi-site inventory, RBAC, and security at scale.

Platform Developers
These are the “plumbers” that ensure endpoints are automatable, and maintain the Ansible modules, plugins, roles (content) to be used.

Automation Implementers
Ansible playbook writers stitching together each play by play, task by task using tested and validated released developer content.
What happened?

Selected Ansible content has been moved out

Ansible 2.9

Ansible Content (plugins, modules, etc.)

Ansible 2.10 Base (soon to be renamed Core)
Can I stay on Ansible 2.9 with Collections?

You sure can!

Ansible 2.9

Ansible Content
(plugins, modules, etc.)

Ansible 2.10 Base
(soon to be renamed Core)
Introducing the Ansible Collection

Simplified and consistent content schema

- A standardized way to organize and package Ansible content (roles, modules, module utilities, plugins, documentation)
- Semantic versioning
- Portable and flexible delivery
Content = Collections

Automation Hub

Galaxy

Content = Apps

Apple App Store

Apple

iPhone

Google Play

Android

Ansible Automation Platform
Collection Directory Structure

- **galaxy.yml**: source data for the MANIFEST.json that will be part of the collection package
- **README.md**: “Front page” for documentation
- **docs/**: local documentation for the collection
- **meta/**: metadata files including `runtime.yml` (for redirection rules, compatibility, deprecation)
- **playbooks/**: playbook snippets
  - **tasks/**: holds ‘task list files’ for include_tasks/import_tasks usage
- **plugins/**: all Ansible plugins, each in its own subdir
  - **modules/**: module plugins (aka “modules”)
  - **lookups/**: lookup plugins
  - **filters/**: Jinja2 filter plugins
  - **connection/**: connection plugins required if not using default
- **roles/**: Ansible roles
- **tests/**: sanity, unit, integration tests
**Goal:** Create a downstream for command line Ansible
(but remember, command line Ansible is only one piece of the **Platform**!)

*Ansible Base* in 2.10 to be renamed to Ansible Core in 2.11

Source:
https://access.redhat.com/articles/5392421#:~:text=Ansible%202.10%20is%20not%20supported,considered%20the%20Upstream%20project%20distribution.
Where did a module or plugin go?

Check the `ansible_builtin_runtime.yml` file

```yaml
podman_container_info:
  redirect: containers.podman.podman_container_info

podman_image_info:
  redirect: containers.podman.podman_image_info

podman_image:
  redirect: containers.podman.podman_image

podman_volume_info:
  redirect: containers.podman.podman_volume_info
```

**NOTES:**
- This file is extremely important. When using Ansible 2.9 and migrated Collections the file above documents the redirection/mapping of content that was once in 2.9 and migrated out. It also is a source of truth for all future Collection migrations or moves. FQCN is **not** required.
- Net new Collections, or new content in an existing Collection FQCN is **required**.
Part 1 - Installing a collection

Getting Ansible 2.9 control nodes ready for Collections

How do I get Ansible Content?
Common “paths” for implementing Ansible Collections

1. Download and install the Collection into your runtime or virtual Ansible environment, and update all playbooks

2. Provide the downloaded Collection as part of your source control management (SCM) tree

3. Update the requirements.yml file
Common “paths” for implementing Ansible Collections

Focus on Method 1 and Method 3 for this presentation

1. Download and install the Collection into your runtime or virtual Ansible environment, and update all playbooks

2. Provide the downloaded Collection as part of your source control management (SCM) tree

3. Update the requirements.yml file
Installing a Collection via CLI

An Example upstream (Galaxy) and downstream (Automation Hub)

**Galaxy**

galaxy.ansible.com
Community / Upstream

- https://galaxy.ansible.com/theforeman/foreman

- ansible-galaxy collection install theforeman.foreman

**Automation Hub**

cloud.redhat.com
Supported / Downstream

- https://cloud.redhat.com/ansible/automation-hub/redhat/satellite

- ansible-galaxy collection install redhat.satellite

---

NOTE: In many cases the upstream and downstream Collections could be the same, cisco.ios
Configuring remote servers and lookup precedence

```
[galaxy]
server_list = automation_hub, my_private_hub, release_galaxy

[galaxy_server.automation_hub]
url=https://cloud.redhat.com/api/automation-hub/
token=my_ah_token

[galaxy_server.my_private_hub]
url=https://automation.my_org/
username=my_user
password=my_pass

[galaxy_server.release_galaxy]
url=https://galaxy.ansible.com/
```

Each Collection repository is checked in order.

Galaxy does not require a token except to publish content.
Configuring local paths and lookup precedence

$ cat ansible.cfg
# Paths to search for collections, colon separated
# collections_paths = ~/.ansible/collections:/usr/share/ansible/collections

You can add a local directory path in your ansible.cfg to increase control (point to local Collection)
Collection locations and lookup precedence

Order of precedence (this is default, similar to ansible.cfg lookup):

1. Home folder (where Ansible Collections are installed)
   ~/.ansible/ansible-collections/

2. Ansible top level collections directory (on RHEL, for example)
   /usr/share/ansible/collections
Part 1 - Installing a collection – Strategy 1

→ **ansible-galaxy collection install arista.eos**
  Process install dependency map
  Starting collection install process
  Skipping 'ansible.netcommon' as it is already installed
  Installing 'arista.eos:1.2.0' to
  '/Users/sean/.ansible/collections/ansible_collections/arista/eos'

→ **ansible-galaxy collection install amazon.aws:1.2.0**
  Process install dependency map
  Starting collection install process
  Skipping 'ansible.netcommon' as it is already installed
  Installing 'amazon.aws:1.2.0' to
  '/Users/sean/.ansible/collections/ansible_collections/amazon/aws'

Optionally specify an exact version and pin the collection
Process install dependency map
Initial connection to galaxy_server: https://galaxy.ansible.com/
Opened /Users/sean/.ansible/galaxy_token
Calling Galaxy at https://galaxy.ansible.com/api/
Processing requirement collection 'ansible.tower'
Collection requirement 'ansible.tower' is the name of a collection
Found API version 'v1, v2' with Galaxy server galaxy (https://galaxy.ansible.com/api/)
Calling Galaxy at https://galaxy.ansible.com/api/v2/collections/ansible/tower/versions/
Initial connection to galaxy_server: https://cloud.redhat.com/api/automation-hub/
Collection 'ansible.tower' is not available from server galaxy https://galaxy.ansible.com/api/
Calling Galaxy at https://cloud.redhat.com/api/automation-hub/
Found API version 'v3' with Galaxy server automation_hub (https://cloud.redhat.com/api/automation-hub/)
Calling Galaxy at https://cloud.redhat.com/api/automation-hub/v3/collections/ansible/tower/versions/3.8.0/
Collection 'ansible.tower' obtained from server automation_hub https://cloud.redhat.com/api/automation-hub/
Starting collection install process
Installing 'ansible.tower:3.8.0' to '/Users/sean/.ansible/collections/ansible_collections/ansible/tower'
Validating downloaded file hash 3403adb7ce471d8a29034dfb8aa3a19a12416d5343380fb08d61926003836ad3 with expected hash 3403adb7ce471d8a29034dfb8aa3a19a12416d5343380fb08d61926003836ad3

Verbose output
Provide the collection as part of your SCM tree

Dynamically loaded content can be put in .gitignore

Content can also be hardcoded into your SCM tree
Part 1 - Installing a collection - Strategy 3

Updating the requirements.yml file

Dynamically loads collections for Ansible Automation Platform

```
[andrius@rhel8 hello_collection]$ tree
.
├── collections
│   └── requirements.yml
├── playbooks
│   ├── ansible.cfg
│   ├── terminate_instance.yml
│   ├── no_tags.yml
│   ├── time_test.yml
│   └── turn_off_time.yml
└── README.md
```

Directory needs to be named collections at top level within the repository

YAML file lists out collections you need and want to use
Updating the requirements.yml file

Dynamically loads collections for Ansible Automation Platform

collections:
- name: community.aws
  source: https://galaxy.ansible.com
- name: amazon.aws
  source: https://galaxy.ansible.com
- name: ansible.tower
  version: 15.0.0
  source: https://cloud.redhat.com/api/automation-hub/
- name: benokraitis.sink
  version: 1.0.1
  source: https://private_ah.andrius.us

Be very specific on where you load the content from, and which version is desired to be used.

Private Automation Hub can be used as well.
Part 1 - Installing a collection - Strategy 3

Installing via Ansible command line

$ ansible-galaxy install -r requirements.yml

Process install dependency map
Starting collection install process
Installing 'community.aws:1.2.1' to
'/home/student1/.ansible/collections/ansible_collections/community/aws'
Installing 'amazon.aws:1.2.1' to
'/home/student1/.ansible/collections/ansible_collections/amazon/aws'
Installing 'ansible.netcommon:1.4.1' to
'/home/student1/.ansible/collections/ansible_collections/ansible/netcommon'
If you choose not to use a source, you can have Ansible Automation Platform choose in order similarly to ansible.cfg
Example 1: Creating a user in Cyberark Vault
Method 1: Using `collections` keyword (easier but more risk)

Example 1: Using a Collection in a Playbook

**Without Collections (2.9 and earlier)**

```yaml
---
- name: create user with cyberark vault
  hosts: localhost
  tasks:
    - name: Logon - use_shared_logon_authentication
cyberark_authentication:
      api_base_url: "https://www.my-server.com"
      use_shared_logon_authentication: True
    - name: Create user & add it to a group
cyberark_user:
      username: username
      initial_password: password
group_name: GroupOfUser
cyberark_session: '{{ cyberark_session }}'
---
```

**With Collections (2.9 and later)**

```yaml
---
- name: create user with cyberark vault
  hosts: localhost
  collections:
    - cyberark.pas
  tasks:
    - name: Logon - use_shared_logon_authentication
cyberark_authentication:
      api_base_url: "https://www.my-server.com"
      use_shared_logon_authentication: True
    - name: Create user & add it to a group
cyberark_user:
      username: username
      initial_password: password
group_name: GroupOfUser
cyberark_session: '{{ cyberark_session }}'
---
```
Example 1: Using a Collection in a Playbook

Method 2: Using FQCN per task (recommended)

<table>
<thead>
<tr>
<th>Without Collections (2.9 and earlier)</th>
<th>With Collections (2.9 and later)</th>
</tr>
</thead>
<tbody>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td><strong>name</strong>: create user with cyberark vault</td>
<td><strong>name</strong>: create user with cyberark vault</td>
</tr>
<tr>
<td><strong>hosts</strong>: localhost</td>
<td><strong>hosts</strong>: localhost</td>
</tr>
<tr>
<td><strong>tasks</strong>:</td>
<td><strong>tasks</strong>:</td>
</tr>
<tr>
<td>- <strong>name</strong>: Logon - use_shared_logon_authentication</td>
<td>- <strong>name</strong>: Logon - use_shared_logon_authentication</td>
</tr>
<tr>
<td>cyberark_authentication:</td>
<td>cyberark.pas.cyberark_authentication:</td>
</tr>
<tr>
<td>use_shared_logon_authentication: True</td>
<td>use_shared_logon_authentication: True</td>
</tr>
<tr>
<td>- <strong>name</strong>: Create user &amp; add it to a group</td>
<td>- <strong>name</strong>: Create user &amp; add it to a group</td>
</tr>
<tr>
<td>cyberark_user:</td>
<td>cyberark.pas.cyberark_user:</td>
</tr>
<tr>
<td>username: username</td>
<td>username: username</td>
</tr>
<tr>
<td>initial_password: password</td>
<td>initial_password: password</td>
</tr>
<tr>
<td>group_name: GroupOfUser</td>
<td>group_name: GroupOfUser</td>
</tr>
<tr>
<td>cyberark_session: '{% cyberark_session %}'</td>
<td>cyberark_session: '{% cyberark_session %}'</td>
</tr>
</tbody>
</table>

**NOTE**: Please use FQCN per task due to potential duplicate naming/ordering conflicts with playbooks referencing multiple collections. The collections directive has other limitations, therefore FQCN is recommended.
Example 2: Configure an Arista network interface, modify inventory and connection settings
Example 2: Using a Collection in a Playbook

Method 1: Using `collections` keyword (easier but more risk)

### Without Collections (2.9 and earlier)

```yaml
---
- name: configure interface on Arista
  hosts: switch1

  tasks:
  - name: Merge provided configuration
    eos_l3_interfaces:
      config:
        - name: Ethernet1
          ipv4:
            - address: 198.51.100.14/24
---
```

### With Collections (2.9 and later)

```yaml
---
- name: configure interface on Arista
  hosts: switch1
  collections:
    - arista.eos

  tasks:
    eos_l3_interfaces:
      config:
        - name: Ethernet1
          ipv4:
            - address: 198.51.100.14/24
```
Example 2: Using a Collection in a Playbook

Method 2: Using FQCN per task

Without Collections (2.9 and earlier)

```yaml
---
- name: configure interface on Arista
  hosts: switch1

  tasks:
  - name: Merge provided configuration
    eos_l3_interfaces:
      config:
        - name: Ethernet1
          ipv4:
            address: 198.51.100.14/24
---
```

With Collections (2.9 and later)

```yaml
---
- name: configure interface on Arista
  hosts: switch1

  tasks:
    arista.eos.eos_l3_interfaces:
    config:
      - name: Ethernet1
        ipv4:
          address: 198.51.100.14/24
---
```

**NOTE:** Please use FQCN per task due to potential duplicate naming/ordering conflicts with playbooks referencing multiple collections. The collections directive has other limitations, therefore FQCN is recommended.
Example 2: Using a Collection in a Playbook

Futureproofing inventory and connection plugin specifications

```
[arista]
switch1 ansible_host=172.16.1.34
switch2 ansible_host=172.16.1.35

[arista:vars]
anible_user=admin
ansible_network_os=arista.eos.eos
ansible_connection=ansible.netcommon.network_cli
ansible_become=true
ansible_become_method=enable
ansible_ssh_private_key_file=/home/andrius/.ssh/private.pem
```

These two parameters use the connection plugin (network_cli/arista) from the Netcommon Collection, which is automatically installed.
Debugging and troubleshooting Ansible Collections
Tools that help you troubleshoot running Collections

[sean@rhel8]$ ansible-playbook arista.yml -vvv

Three v’s provides enough debug verbosity to see exactly which module is being used, and where it is being loaded from.

TASK [Merge provided configuration] ****************************
<!--- output removed for brevity ---!>
Using module file
/home/student1/.ansible/collections/ansible_collections/arista/eos/plugins/modules/eos_l
3_interfaces.py
<!--- output removed for brevity ---!>

Documentation on additional verification: https://docs.ansible.com/ansible/latest/user_guide/collections_using.html#verifying-collections
Debugging Playbooks

Tools that help you troubleshoot running Collections

```
[sean@rhel8]$ ansible-playbook arista.yml -vvvv
```

Four v’s provides enough debug verbosity to see exactly which connection plugin is being used.
Resources and tools

How do I use Ansible Collections?
Using a Collection

A net new Collection

```
ansible-galaxy collection init namespace.collection_name
```

```
[user@rhel ~]$ ansible-galaxy collection init test.hello_world
- Collection test.hello_world was created successfully
```

```
[user@rhel ~]$ tree test
test
 └── hello_world
       └── docs
       └── galaxy.yml
       └── plugins
           └── README.md
       └── README.md
       └── roles
```
Ansible Roles

subtitle
Migrating Roles into an Ansible Collection

How can I organize my existing Ansible Roles?

github.com/norbert_miller/collection

- LICENSE
- playbooks
  - example_playbook.yml
  - README.md
- roles
  - rhel_role
    - meta
      - main.yml
      - README.md
      - tasks
        - main.yml
      - templates
        - readme.j2
  - network_role
    - meta
      - main.yml
      - README.md
      - tasks
        - main.yml
      - templates
        - readme.j2

github.com/norbert_miller/rhel_role

github.com/norbert_miller/network_role
Documentation

Collection Docs

Collection Index

These are the collections with docs hosted on docs.ansible.com.

- amazon.aws
- ansible.builtin
- ansible.netcommon
- ansible.posix
- ansible.windows
- arista.eos
- awx.awx
- azure.azcollection
- check_point.mgmt
- chocolatey.chocolatey
- cisco.aci
- cisco.asa
- cisco.intersight
- cisco.ios
- cisco.iosxr

https://docs.ansible.com/ansible/latest/collections/
Collections are a distribution format for Ansible content that can be distributed as a Git repository into collections, the module documentation will move to the collection format. You can install and use collections through Ansible Galaxy.

- For details on how to develop collections see Developing collections
- For the current development status of Collections and FAQ see https://docs.ansible.com/ansible/latest/user_guide/

- Installing collections
  - Installing collections with ansible-galaxy
  - Installing an older version of a collection
  - Installing a collection from a git repository
  - Default repository search locations
  - Specifying the location to search for collections
  - Install multiple collections with a requirements file
  - Downloading a collection for offline use
  - Configuring the ansible-galaxy client
- Downloading collections
Listing all installed Collections

Ansible 2.10 and newer

$ ansible-galaxy collection list

# /home/astark/.ansible/collections/ansible_collections
Collection     Version
-------------- -------
cisco.aci      0.0.5
Cisco.mso      0.0.4
Splunk.es      0.0.5

# /usr/share/ansible/collections/ansible_collections
Collection     Version
-------------- -------
fortinet.fortios 1.0.6
Pureport.pureport 0.0.8
Sensu.sensu_go   1.3.0
List specific Collection information

Ansible 2.10 and newer

```
$ ansible-galaxy collection list fortinet.fortios

# /home/astark/.ansible/collections/ansible_collections
Collection       Version
---------------- -------
fortinet.fortios 1.0.1

# /usr/share/ansible/collections/ansible_collections
Collection       Version
---------------- -------
fortinet.fortios 1.0.6
```
Other Resources

Potpourri

- Developer Guide
- Ansible Collections Blog
- Developer Community
- Subscribe to The Bullhorn
- Past Ansible Webinars
- Self-paced and instructor-led Workshops
Thank you

Please subscribe to our YouTube Channel!

We have lots of great content coming out!

linkedin.com/company/red-hat
youtube.com/AnsibleAutomation
twitter.com/Ansible