Validation workflow guide

Red Hat Ansible Automation Platform

Version 2.2
March 2026

Contents

Introduction

Validated Content is intended to provide customers with content focused on use cases, examples, and best practices from Red Hat and technology partners that can be adapted to customer-specific environments.

Validated Content is available on Ansible automation hub for Ansible Automation Platform versions 2.4 and later. Unlike Red Hat Ansible Certified Content, Validated Content is unsupported by Red Hat.

This guide describes the Ansible Validated Content workflow. It is intended to be used by partners who are interested in building Validated Content and publishing it on Ansible automation hub.

STEP 1: Create your Validated Content

Validated Content is packaged inside of a collection, which is the standard format for Ansible content. For more information about developing Ansible Content Collections, see Developing collections and the Ansible collections checklist.

Validated Content is intended to allow partner ISVs to provide value to Ansible Automation Platform customers. Therefore, the only business requirement for Validated Content is for it to be published on Ansible automation hub. This allows Red Hat and partners to collect targeted feedback that can be used to quickly evolve, re-assess, or retire content. Partners are not required to publish their Validated Content on Galaxy, which is the open source collection repository.

Naming your Ansible Validated Content

The suggested naming convention for Ansible Validated Content is *domain.platform*.

The *domain* name must be one of the following:

- Your partner namespace on the automation hub.
- One of the domains listed in the following table (for Red Hat content).

To ensure that Validated Content is differentiated at a collection level, Ansible Validated Content created by Red Hat must use one of the domain-specific namespaces in the following table:

cloud	For public cloud, private cloud, and container native use cases
edge	For edge use cases
infra	For infrastructure use cases (for example, OS and enterprise applications automation)
network	For network automation use cases
security	For security automation use cases

The *platform* name must be the product with which the content is intended to be used. Where complexity or scope requires it, you can use *domain.platform_usecase*. For example, use *infra.aap_configuration* to indicate that the content is for configuring Ansible Automation Platform.

Validated Content can also be created for platform-agnostic content. In this case, use *domain.use_case* as the naming convention, for example, *network.backup*. All namespaces and collection names must follow the Ansible collection and namespace requirements.

Requesting an automation hub namespace

Once you have created the Validated Content namespace, you can request that namespace on automation hub by emailing ansiblepartners@redhat.com. Include the following details in your email:

- Requested namespace
- Account number

  NOTE: You can retrieve your account number by clicking your username in automation hub.

STEP 2: Review business requirements for Ansible Validated Content

Ansible Validated Content must adhere to the following general and content requirements:

General requirements for Ansible Validated Content

- Content must support a viable business use case.
- Content must not be a demo.
- Content must be production-ready and able to run on Red Hat Ansible Automation Platform.
- Content must provide a solution to an end-to-end automation use case that brings value to our customers.

Content requirements for Ansible Validated Content

- Content in Validated Content collections must be YAML only, except for necessary plugin(s) for specific use cases that are not generic enough for a Certified or community collection.
- Content must not duplicate content found in a Red Hat Ansible Certified Content Collection.
- Any external plugin or module used in Validated Content must be added as a dependency in the `galaxy.yml` file.

- We recommend that your collections depend exclusively on Validated Content collections, Red Hat Ansible Certified Content Collections, or collections that have an equivalent support level provided by a Red Hat partner. Collections that fulfill this requirement enable more efficient validation.
- Collections that depend on community content will be evaluated and are required to have these dependencies clearly documented as "community".
- External Python dependencies must be listed in a `requirements.txt` file and must be within an open range (>=) in order to prevent issues during the creation of execution environments, particularly when bundling collections that have conflicting versions of the same dependencies.
- Collections must have a `requires_ansible` value, which includes a supported ansible-core version. For more information, see the `requires_ansible` settings section of this guide and Red Hat Ansible Automation Platform life cycle.
- All plugins (including modules) in Validated Content collections must be written exclusively in Python, or PowerShell for Windows-specific use cases. Binary plugins or plugins written in other scripting languages are not permitted.

Node counting for Ansible Automation Platform

Your collection affects how Ansible Automation Platform counts managed nodes for customers inventories. Node counting enables customers to track individual automated systems.

Understanding node counting is essential when your collection:

- Interacts with multiple systems which utilize a controller, for example, Azure, Cisco Catalyst Center, or VMWare, etc.
- Orchestrates workflows across platforms
- Manages infrastructure at scale

For detailed information about node counting methodologies, automation execution patterns, and how different types of automation tasks affect node counts, see the Indirect Node Counting Taxonomy Guide.

Repository and maintainership requirements

- Validated Content created by partners must remain within the partner's GitHub Organization. Validated Content created by internal Red Hat teams can optionally be migrated into the Ansible Community of Practice GitHub organization. Contact ansiblepartners@redhat.com for instructions on migrating your Content.

- The source code of the collection must be placed under an approved open source license and respects its prerequisites. For more information, see Licenses.
- The `README` must align with the recommended collection `README` template.
- Validated Content must have two or more active maintainers listed in a `CODEOWNERS` file.

Testing and validation requirements

- Validated Content must pass `ansible-lint` tests for the production profile with no errors or warnings.
- The `ansible-lint.yml` file (if present) must only contain justified ignores or `skip_list` entries with a comment explaining the need for each ignore or `skip_list` entry.
- If the Content includes Python, the Content must pass `ansible-test` sanity.

Other policy requirements

- The content must not address a use case that is already covered by another Validated Content collection.
- The collection must follow semantic versioning requirements and be at or above version 1.0.0. For more information, see Semantic Versioning 2.0.0.
- There must be no policy or other legal reason preventing the content from being published.

`requires_ansible` settings

- Validated Content must specify a value for the `requires_ansible` key in the `meta/runtime.yml` file. The value must correspond to one of the `ansible-core` versions that Red Hat Ansible Automation Platform supports. Supported `ansible-core` versions include `ansible-core` 2.15 through `ansible-core` 2.18, for example:
  - `requires_ansible` >= 2.16
  - `requires_ansible` >= 2.16, < 2.19

NOTE: `ansible-core` 2.19 and 2.20 is in technical preview only and cannot be utilized in Validated Content.

STEP 3: Review content technical requirements

Content candidates are required to pass all `ansible-lint` production profile checks. For more information, see Profiles. If the Validated Content includes Python content, it must pass `ansible-test` sanity checks. For more information, see Sanity Tests.

Installing and running ansible-lint

Run `ansible-lint` against your Validated Content to ensure it passes the production profile with no errors or warnings. Use one of the following methods to install and run `ansible-lint`:

- Install and run `ansible-lint` as a GitHub Action
- Install and run `ansible-lint` locally

Installing and running `ansible-lint` as a GitHub Action

To run `ansible-lint` as a GitHub Action, use the pre-built GHA for ansible-lint.

Installing and running `ansible-lint` locally

You can install `ansible-lint` from `dnf` or `pip3` using the following commands:

```
$ pip3 install ansible-lint
```

```
$ dnf install ansible-lint
```

For more information, see Installing `ansible-lint`.

To run `ansible-lint` with all tests enforced, specify the production profile using the following command:

```
$ ansible-lint --profile production
```

NOTE: Specifying the production profile is required for Validated Content.

For more information on available installation options, run `ansible-lint --help` and refer to the `ansible-lint` documentation.

Running `ansible-test` sanity

Sanity tests are made up of scripts and tools that are used to perform static code analysis. The primary purpose of these tests is to enforce Ansible coding standards and requirements. For common ways to run `ansible-test`, see the following examples:

```
$ cd {...}/ansible_collections/{namespace}/{collection_name}/  # navigate to collection

$ ansible-test sanity   # run all sanity tests
```

For more information on sanity tests, see Sanity Tests.

If there are issues with a piece of content, Red Hat will notify the author of the content. Delays in addressing and resolving an issue may result in Red Hat removing the content. For more information, see the Validated Content removal policy section of this guide.

The new collection must be placed under validation and testing of a CI/CD pipeline and must guarantee a high test coverage of the collection. At a minimum, each role, module, or plugin must be involved at least once in the testing cycle.

STEP 5: Content approval and distribution

When Validated Content is uploaded to automation hub, Partner Engineering performs a review of the content. Once the content is approved, it will be signed and published on automation hub for consumption.

If the content is rejected, or if there are concerns that need to be addressed before approval, Partner Engineering will reach out to the author and provide an explanation of the issue preventing approval.

Validated Content removal policy

Red Hat reserves the right to remove a piece of Validated Content from Ansible automation hub at any time. Red Hat may remove content in the following example scenarios:

- There is a major security vulnerability within the content.
- There are functionality issues with the content, and the creator does not resolve these issues in a timely manner.
- Bug reports and bugfix PRs have accumulated without being reviewed.
- The content is outdated or otherwise no longer relevant to a platform or product.
- There has been no maintainer activity in the collection repository for several months.
- Continuous Integration (CI) has stopped passing or has not been running for several months.

Ansible partner announcements

Subscribe to the Ansible partner announcements list for periodic updates from the Ansible partner team. Email your subscription request to ansiblepartners@redhat.com.

Exceptions and revisions

Exceptions to this policy can be requested by emailing ansiblepartners@redhat.com. Exceptions are granted at Red Hat's discretion. Red Hat may change the requirements for validating content at any time. Red Hat will give partners advance notification of policy changes.

Feedback

Email ansiblepartners@redhat.com with feedback or questions regarding this document.