Migrating Ansible content to a different collection

When you move content from one collection to another, for example to extract a set of related modules out of `community.general` to create a more focused collection, you must make sure the transition is easy for users to follow.

Migrating content

Before you start migrating content from one collection to another, look at Ansible Collection Checklist.

To migrate content from one collection to another, if the collections are parts of Ansible distribution:

1. Copy content from the source (old) collection to the target (new) collection.
2. Deprecate the module/plugin with `removal_version` scheduled for the next major version in `meta/runtime.yml` of the old collection. The deprecation must be released after the copied content has been included in a release of the new collection.
3. When the next major release of the old collection is prepared:

   - remove the module/plugin from the old collection
   - remove the symlink stored in `plugin/modules` directory if appropriate (mainly when removing from `community.general` and `community.network`)
   - remove related unit and integration tests
   - remove specific module utils
   - remove specific documentation fragments if there are any in the old collection
   - add a changelog fragment containing entries for `removed_features` and `breaking_changes`; you can see an example of a changelog fragment in this pull request
   - change `meta/runtime.yml` in the old collection:
     - add `redirect` to the corresponding module/plugin's entry
     - in particular, add `redirect` for the removed module utils and documentation fragments if applicable
     - remove `removal_version` from there
   - remove related entries from `tests/sanity/ignore.txt` files if exist
   - remove changelog fragments for removed content that are not yet part of the changelog (in other words, do not modify *changelogs/changelog.yaml* and do not delete files mentioned in it)
   - remove requirements that are no longer required in `tests/unit/requirements.txt`, `tests/requirements.yml` and `galaxy.yml`

To implement these changes, you need to create at least three PRs:

1. Create a PR against the new collection to copy the content.
2. Deprecate the module/plugin in the old collection.
3. Later create a PR against the old collection to remove the content according to the schedule.

Adding the content to the new collection

Create a PR in the new collection to:

1. Copy ALL the related files from the old collection.
2. If it is an action plugin, include the corresponding module with documentation.
3. If it is a module, check if it has a corresponding action plugin that should move with it.
4. Check `meta/` for relevant updates to `runtime.yml` if it exists.
5. Carefully check the moved `tests/integration` and `tests/units` and update for FQCN.
6. Review `tests/sanity/ignore-*.txt` entries in the old collection.
7. Update `meta/runtime.yml` in the old collection.

Removing the content from the old collection

Create a PR against the source collection repository to remove the modules, module_utils, plugins, and docs_fragments related to this migration:

1. If you are removing an action plugin, remove the corresponding module that contains the documentation.
2. If you are removing a module, remove any corresponding action plugin that should stay with it.
3. Remove any entries about removed plugins from `meta/runtime.yml`. Ensure they are added into the new repo.
4. Remove sanity ignore lines from `tests/sanity/ignore\*.txt`
5. Remove associated integration tests from `tests/integrations/targets/` and unit tests from

`tests/units/plugins/`.

6. if you are removing from content from `community.general` or `community.network`, remove entries from `.github/BOTMETA.yml`.
7. Carefully review `meta/runtime.yml` for any entries you may need to remove or update, in particular deprecated entries.
8. Update `meta/runtime.yml` to contain redirects for EVERY PLUGIN, pointing to the new collection name.

> **Warning**
>
> Maintainers for the old collection have to make sure that the PR is merged in a way that it does not break user experience and semantic versioning:
>
> 1. A new version containing the merged PR must not be released before the collection the content has been moved to has been released again, with that content contained in it. Otherwise the redirects cannot work and users relying on that content will experience breakage.
> 2. Once 1.0.0 of the collection from which the content has been removed has been released, such PRs can only be merged for a new **major** version (in other words, 2.0.0, 3.0.0, and so on).

Updating BOTMETA.yml

The `BOTMETA.yml`, for example in community.general collection repository, is the source of truth for:

- ansibullbot

If the old and/or new collection has `ansibullbot`, its `BOTMETA.yml` must be updated correspondingly.

Ansibulbot will know how to redirect existing issues and PRs to the new repo. The build process for docs.ansible.com will know where to find the module docs.

```
$modules/monitoring/grafana/grafana_plugin.py:
    migrated_to: community.grafana
$modules/monitoring/grafana/grafana_dashboard.py:
    migrated_to: community.grafana
$modules/monitoring/grafana/grafana_datasource.py:
    migrated_to: community.grafana
$plugins/callback/grafana_annotations.py:
    maintainers: $team_grafana
    labels: monitoring grafana
    migrated_to: community.grafana
$plugins/doc_fragments/grafana.py:
    maintainers: $team_grafana
    labels: monitoring grafana
    migrated_to: community.grafana
```

Example PR

- The `migrated_to:` key must be added explicitly for every *file*. You cannot add `migrated_to` at the directory level. This is to allow module and plugin webdocs to be redirected to the new collection docs.
- `migrated_to:` MUST be added for every:
  - module
  - plugin
  - module_utils
  - contrib/inventory script
- You do NOT need to add `migrated_to` for:
  - Unit tests
  - Integration tests
  - ReStructured Text docs (anything under `docs/docsite/rst/`)
  - Files that never existed in `ansible/ansible:devel`

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\dev_guide\(ansible-devel)(docs)(docsite)(rst)(dev_guide)developing_collections_migrating.rst`, line 127)

Unknown directive type "seealso".

```
.. seealso::

   :ref:`collections`
       Learn how to install and use collections.
   :ref:`contributing_maintained_collections`
       Guidelines for contributing to selected collections
   `Mailing List <https://groups.google.com/group/ansible-devel>`_
       The development mailing list
   :ref:`communication_irc`
       How to join Ansible chat channels
```