orphan:

Search paths in Ansible

You can control the paths Ansible searches to find resources on your control node (including configuration, modules, roles, ssh keys, and more) as well as resources on the remote nodes you are managing. Use absolute paths to tell Ansible where to find resources whenever you can. However, absolute paths are not always practical. This page covers how Ansible interprets relative search paths, along with ways to troubleshoot when Ansible cannot find the resource you need.

Config paths

By default these should be relative to the config file, some are specifically relative to the current working directory or the playbook and should have this noted in their description. Things like ssh keys are left to use the current working directory because it mirrors how the underlying tools would use it.

Task paths

Relative paths used in a task typically refer to remote files and directories on the managed nodes. However, paths passed to lookup plugins and some paths used in action plugins such as the "src" path for the :ref:`template <ansible_collections.ansible.builtin.template_module>` and :ref:`copy <ansible_collections.ansible.builtin.copy_module>` modules refer to local files and directories on the control node.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\(ansible-devel)(docs)(docsite)(rst)(user_guide)playbook_pathing.rst`, line 21); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\ansible-devel\docs\docsite\rst\user_guide\(ansible-devel)(docs)(docsite)(rst)(user_guide)playbook_pathing.rst`, line 21); *backlink*

Unknown interpreted text role "ref".

Resolving local relative paths

When you specify a relative path for a local file, Ansible will try to find that file first in the current task's role, then in other roles that included or depend on the current role, then relative to the file in which the task is defined, and finally relative to the current play. It will take the first matching file that it finds. This way, if multiple files with the same filename exist, Ansible will find the file that is closest to the current task and that is most likely to be file you wanted.

Specifically, Ansible tries to find the file

1. In the current role.
   1. In its appropriate subdirectoryâ€"files", "vars", "templates" or "tasks", depending on the kind of file Ansible is searching for.
   2. Directly in its directory.
2. Like 1, in the parent role that called into this current role with *include_role*, *import_role*, or with a role dependency. If the parent role has its own parent role, Ansible will repeat this step with that role.
3. Like 1, in the current task file's directory.
4. Like 1, in the current play file's directory.

Ansible does not search the current working directory. (The directory you're in when you execute Ansible.) Also, Ansible will only search within a role if you actually included it with an *include_role* or *import_role* task or a dependency. If you instead use *include*, *include_task* or *import_task* to include just the tasks from a specific file but not the full role, Ansible will not search that role in steps 1 and 2.

When you execute Ansible, the variable *ansible_search_path* will contain the paths searched, in the order they were searched in but without listing their subdirectories. If you run Ansible in verbosity level 5 by passing the *-vvvvv* argument, Ansible will report each directory as it searches, except when it searches for a tasks file.

Note

The current working directory might vary depending on the connection plugin and if the action is local or remote. For the remote it is normally the directory on which the login shell puts the user. For local it is either the directory you executed ansible from or in some cases the playbook directory.