## **Building an Execution Environment**

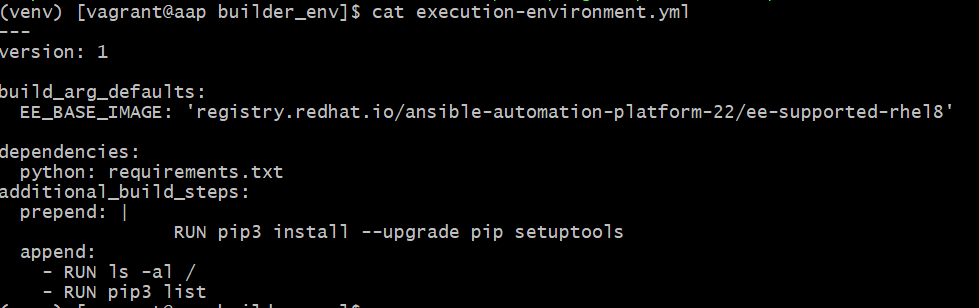
Using Ansible content that depends on non-default dependencies (custom virtual environments) can be tricky. Packages must be installed on each node, play nicely with other software installed on the host system, and be kept in sync. Previously, jobs ran inside of a virtual environment at /var/lib/awx/venv/ansible by default, which was pre-loaded with dependencies for ansible-runner and certain types of Ansible content used by the Ansible control machine.

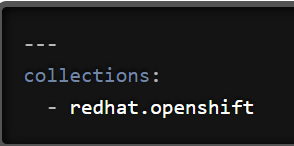
To help simplify this process, container images can be built that serve as Ansible [control nodes](https://docs.ansible.com/ansible/latest/network/getting_started/basic_concepts.html#control-node). These container images are referred to as automation execution environments, which you can create with ansible-builder and then ansible-runner can make use of those images.

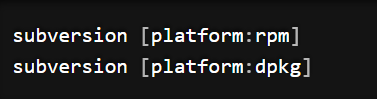
Note: Must Be Done as Root – cause UID/GID issues running as

**Creating a new Execution Environment**

3 Files can be used with a execution-environment.yml build file

Example File

* requirements.yml
  + points to a valid requirements file for the ansible-galaxy collection install –r
* requirements.txt
  + points to a Python requirements file for pip install –r
* Text

  Description automatically generatedbindep.txt
  + requirements file specifies cross-platform requirements, if there are any. These get processed by bindep and then passed to dnf

Additional commands may be specified in the additional\_build\_steps section, to be executed before the main build steps (prepend) or after (append). The syntax needs to be either a multi-line string (as shown in the prepend section of the example definition file) or a list (as shown via the example’s append section).

### Customizable Options

Before we run the build command, let’s discuss the customizable options you can use alongside it.

'-f', '--file'

This flag points to the specific definition file of the Execution Environment; it will default to execution-environment.yml if a different file name is not specified.

'-b', '--base-image'

The parent image for the Execution Environment; when not mentioned, it defaults to quay.io/ansible/ansible-runner:devel.

'-c', '--context'

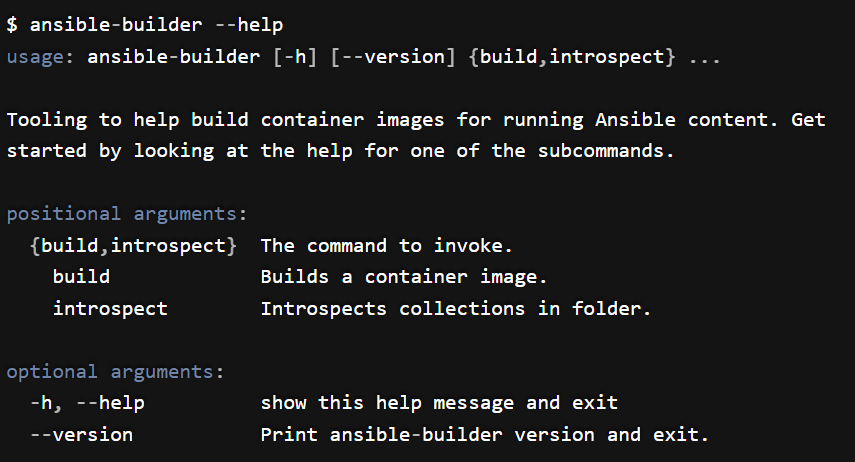
The directory to use for the build context, if it should be generated in a specific place. The default location is $PWD/context.

'--container-runtime'

Specifies which container runtime to use; the choices are podman (default option) or docker.

'--tag'

The name for the container image being built; when nothing is specified with this flag, the image will be named ansible-execution-env.



**Setting Up and Building**

With the virtual environment activated

(venv)$ pip install ansible-builder

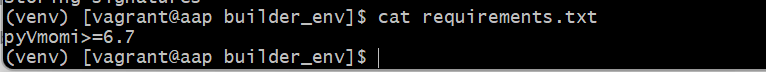
Make a folder for working Execution-Environments

(venv)$ mkdir ee\_build

(venv)$ vi requirements.txt

Add   
pyVmomi>=6.7  
jmespath

requests  
pyVim



Create execution-environment.yml

(venv)$ vi execution-environment.yml

---  
version: 1  
dependencies:  
 python: requirements.txt

---

version: 1

build\_arg\_defaults:

EE\_BASE\_IMAGE: 'registry.redhat.io/ansible-automation-platform-22/ee-supported-rhel8'

dependencies:

python: requirements.txt

additional\_build\_steps:

prepend: |

RUN pip3 install --upgrade pip setuptools

append:

- RUN ls -al /

- RUN pip3 list

Log in to registry.redhat.io

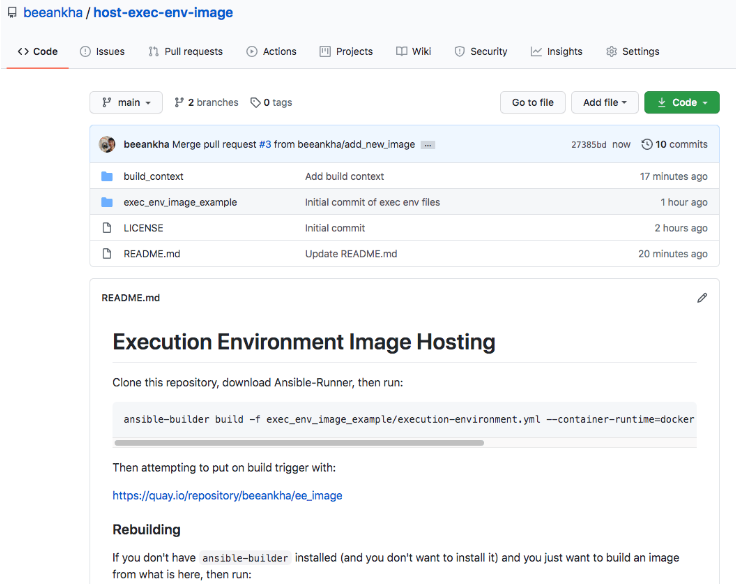
Podman login registry.redhat.io

Log in to automationhub ip

The ansible-builder build command takes an execution environment definition as an input. It outputs the build context necessary for building an execution environment image, and proceeds with building that image. The image can be re-built with the build context elsewhere, and produces the same result. By default, it looks for a file named execution-environment.yml in the current directory.

(venv)$ ansible-builder build –t <automationhub ip or fqdn>/vmware\_ee -v 3

After an Execution Environment image has been built using Ansible Builder, all of the build context files can be pushed to GitHub (or any other version control system) for distribution. See below for an example of a repository that hosts everything necessary for re-building a specific image:



Push Image up to Automation Hub

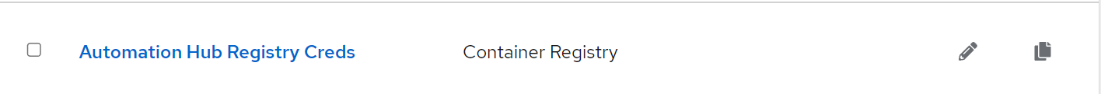
podman login https://<<automation-hub ip/url>> --tls-verify=false

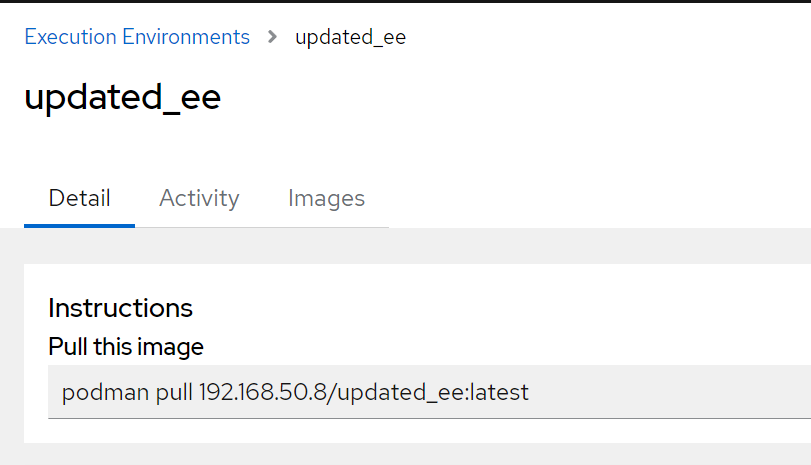
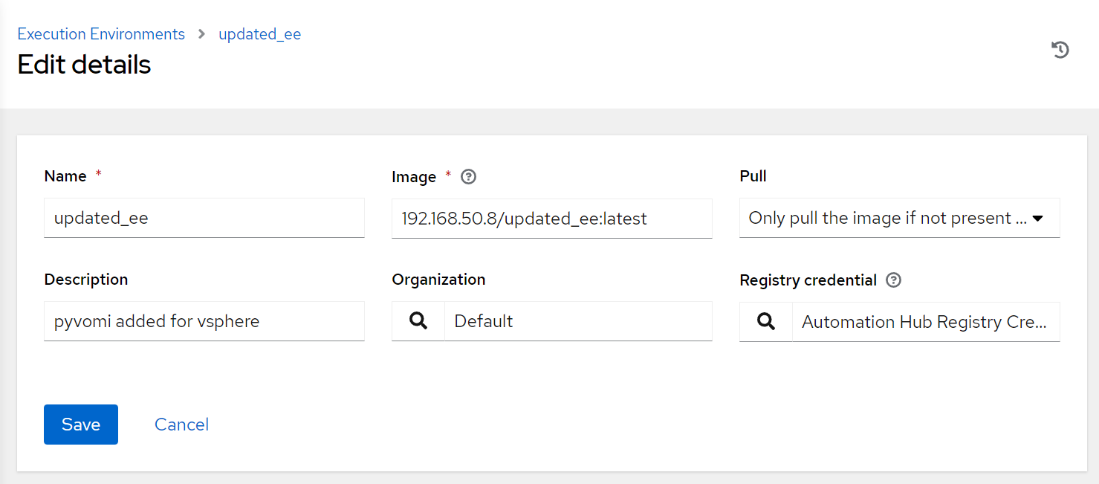
podman images

podman push 0438081a9a2e <<automationhub ip/dns name>>/vmware\_ee --tls-verify=false --remove-signatures

^ image Id

The --remove-signatures flag is required when signed images from registry.redhat.io are pushed to the automation hub container registry. The push operation re-compresses image layers during the upload, which is not guaranteed to be reproducible and is client implementation dependent. This may lead to image-layer digest changes and a failed push operation, resulting in Error: Copying this image requires changing layer representation, which is not possible (image is signed or the destination specifies a digest).

Next Steps:  
  
Create an Automation Hub Container Registry Credential in Ansible Controller  


Verify new Execution-Environment is in Automation Hub  
  
  
Add Execution Environment in Ansible Controller  


Validate localhost in inventory  
  
requires matching ansible\_python\_interpretor

**Addition Notes**  
quay.io images are currently ansible 2.12.5 - python 3.8 "ansible\_python\_interpreter": "/usr/bin/python3.8"

Redhat-registry has current ansible core 2.13.5 - python 3.9 "ansible\_python\_interpreter": "/usr/bin/python3.9"

'registry.redhat.io/ansible-automation-platform-22/ee-supported-rhel8'  
  
  
**# locally install collections – check for python requirements**  
  
ansible-galaxy collection install community.dns community.vmware community.windows community.network  
  
ansible**-**builder introspect **--**sanitize **~/.**ansible**/**collections**/**

**Notes: On Changes to Localhost**  
  
<https://cloudautomation.pharriso.co.uk/post/implicit_localhost/>