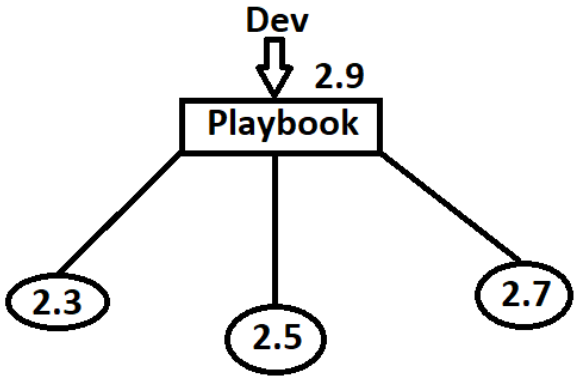
Ansible

Session 8 - 15th Jan 2023 Summary

- ➢ In controller node, we need **ansible package, python, modules and plugins** for running any ansible playbook.

- ➢ A Developer creates a playbook using one of the ansible versions and the developer can run the playbook in their environment. But when a developer shares that ansible playbook to another person and there they might use different ansible versions and the ansible playbook might not work. Therefore before sharing we have to test it in different-different ansible versions.

- ➢ Then for testing we need different-different OS for installing different-different ansible versions. And for installing OS we can use **container technology** because it will launch the OS within seconds.



- ➢ In container technology, the **OS is known as a container**.

- ➢ In container we have to install perticular version of ansible package and supported python and this is known as **environment or execution environment.**

➢ For setting up the above environment we need a **container image** which provides the above software when we launch the container.

➢ For launching the execution environment in the container automatically then Ansible provides one tool known as **Ansible Navigator**.

➢ Ansible Navigator provides a **builder tool** which will help to create our own custom environment.

➢ We launch a container for the execution environment and that container is also known as **ansible runner**.

➢ Till now whatever we do using ansible, ansible-playbook, ansible-doc, ansible-config commands that will also be done by ansible-navigator command. If we use ansible-navigator command then we have mentioned that where we want to run playbook means in the base system or in a container.

➢ When we run playbook then that playbook needs a **ansible-configuration file and inventory**.

➢ As we already know ansible is built using python language. Therefore for python, ansible tools is one of the libraries. Therefore for installing ansible-navigator we can use the pip3 command.

1) For installing pip3 command :
   # *yum install python3-pip*

2) For installing ansible navigator command :
   # *pip3 install ansible-navigator*

➢ For launching containers there are multiple technologies such as **docker, podman, crio,** etc. But by default ansible navigator used podman.

For installing podman :

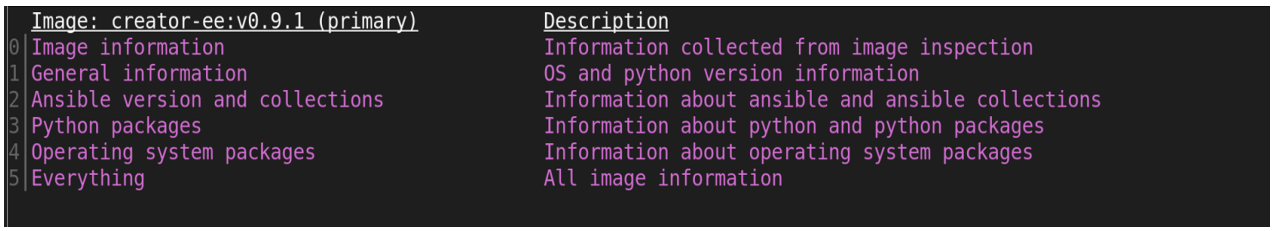
 # yum install podman

➢ We use a pre-created image which is provided by **quay**. When we download an image then by default "***quay.io/ansible/creator-ee:v0.9.1***" downloaded. For downloading image,

 # ansible-navigator images

➢ After downloading image it will look like this,

When we press 0 then it will look like this and as we can see multiple options there. We can see what we want using number press.



Using the "esc" button we can go back.

➢ If we want all information about image as standard output then,

 # ansible-navigator images -m stdout

➢ When we run an ansible playbook then we have to provide a playbook, inventory and configuration file. So if we run playbook using ansible-navigator command then they provide multiple options such as,
--container-engine : for providing container engine,
--execution-environment : for providing environment,
--execution-environment-image : for providing image,

--inventory : for providing an inventory file, etc.

➢ When we want to display inventory :

 # ansible-navigator inventory -i new_inventory

➢ Run ansible playbook using ansible-navigator command.

```
[root@localhost ansible-navigator-ws]# ansible-navigator run web.yml --inventory new_inventory --container-engin
e podman --execution-environment  true --execution-environment-image   quay.io/ansible/creator-ee:v0.9.1
```

➢ Rather than write above arguments again & again we can write all options in one file and when we run ansible-navigator command it will automatically find that file and it contents, that file is **ansible-navigator.yml**

```
ansible-navigator:
  execution-environment:
    image: quay.io/ansible/creator-ee:v0.9.1
    pull:
      policy: missing

  ansible:
    config:
      path: ansible.cfg
    inventory:
      entries:
      - new_inventory
~
~
```

➢ After creating the above file then we only need to provide playbook name.

 # ansible-navigator run web.yml

or

 # ansible-navigator run web.yml -m stdout

➢ Using an ansible role we can organize code in a better way. But if whatever modules and plugins are needed to run a role that is not available in our environment then that role is not able to run. So the solution is collection.

➢ Install pre-created roles from ansible galaxy.

 # ansible-galaxy install geerlingguy.apache

where, **geerlingguy is account name**
apache is the role name.

➢ If we want to search module from command line using role name or platform name or tag name then,

 # ansible-galaxy role search mysql

 # ansible-galaxy role search --galaxy-tags apache

 # ansible-galaxy role search --platforms EL

➢ If we want to see information about perticular role then,

 # ansible-galaxy role info geerlingguy.apache

➢ **Collection** : Collections are a distribution format for Ansible content that can include playbooks, roles, modules, and plugins.

➢ Install collection from ansible galaxy.

 # ansible-galaxy collection install amazon.aws

➢ In ansible collection, if we see the name of collection then we have to write in some kind of way for example,
accountName.collectionName.ModuleName

This type of format is known as **FQMN/CN (Fully Qualified Module Name/Collection Name)**.

➢ If we want to launch an ec2 instance of aws cloud using ansible then we need some python library that can do help to connect from our side to aws and that library is **boto3 and botocore**.

 # pip3 install botocore boto3

➢ Launch ec2-instance using ansible :

```
- hosts: localhost
  tasks:
  - amazon.aws.ec2_instance:
      name: "myinstance"
      image_id: "ami-1234567890"
      vpc_subnet_id: "subnet-5ca1ab1e"
      instance_type: "t2.micro"
      key_name: "mykey"
      count: 1
      region: "ap-south-1"
      state: present
      access_key: "ACCESS_KEY"
      secret_key: "SECRET_KEY"
```