Ansible 101 : Hands-On

By Akshay Ithape - DevOps Engineer

$whoami

Akshay Ithape, CKA/AD,AWS(2x),RedHat(2x),Terraform,Azure

DevOps Engineer @ Eastern Enterprise , Pune

empowering your software

Passionate About & DEVOPS

Writer @ OpenSource For U .com

The complete portal on open source

I truly believes in Open Source so I like to share my knowledge with community in as many ways possible and helping people.

akshayithape

akshayithape-devops

akshayithape-devops

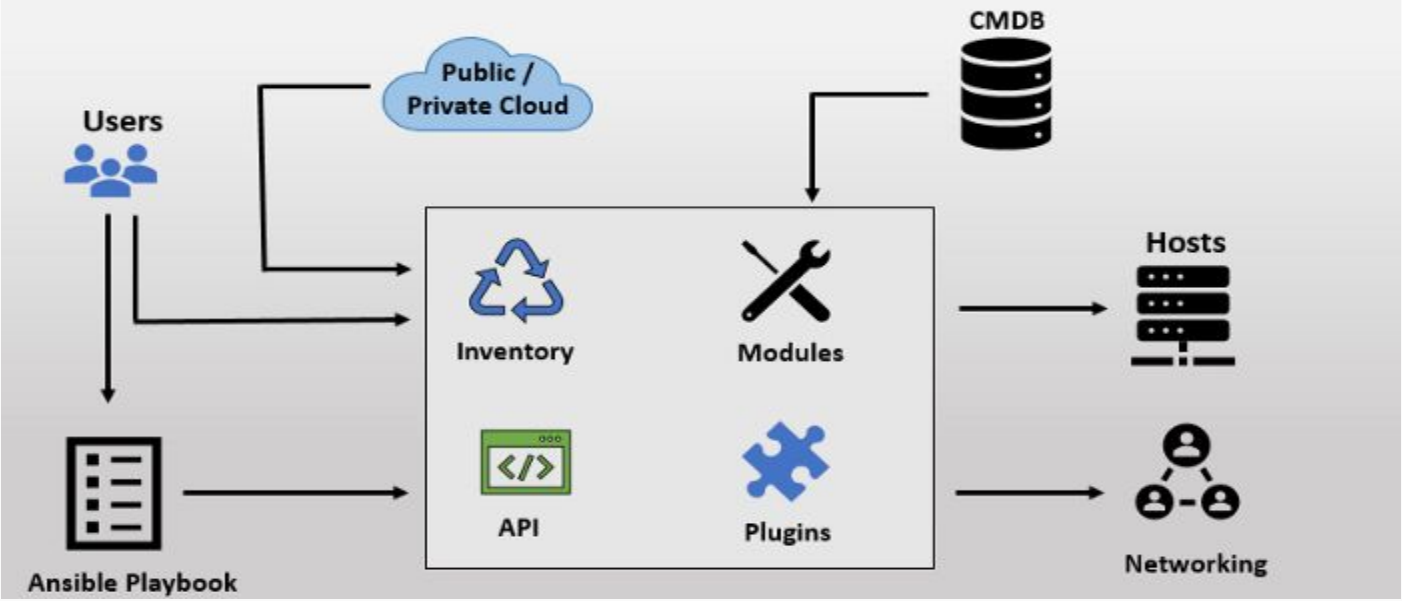
What is Ansible ?

- Ansible is an open source IT automation tool that automates provisioning, configuration management, application deployment, orchestration, and many other manual IT processes.
- Ansible’s main goals are simplicity and ease-of-use.

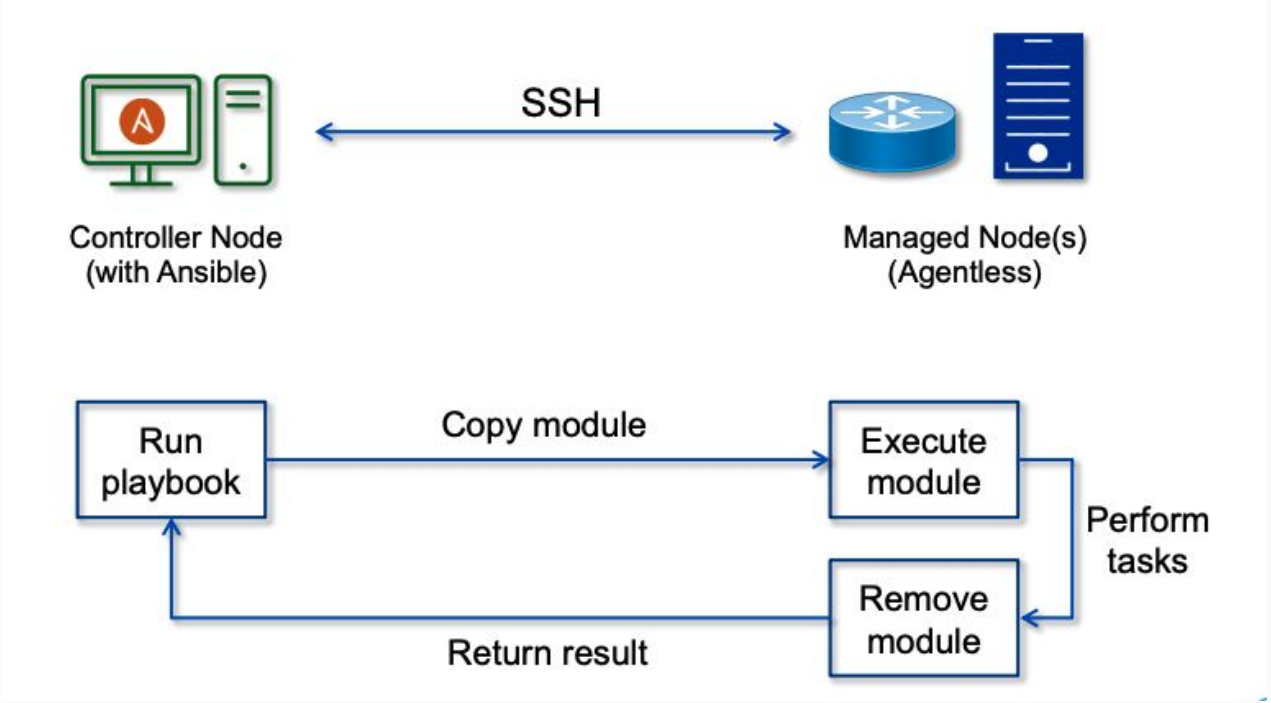
Why Ansible ?

- Free
- Very simple to set up and use
- Agentless
- Easy learn
- Push Model
- Worked on SSH/Powershell Remote Protocol.

Ansible Architecture



How Ansible Works ?



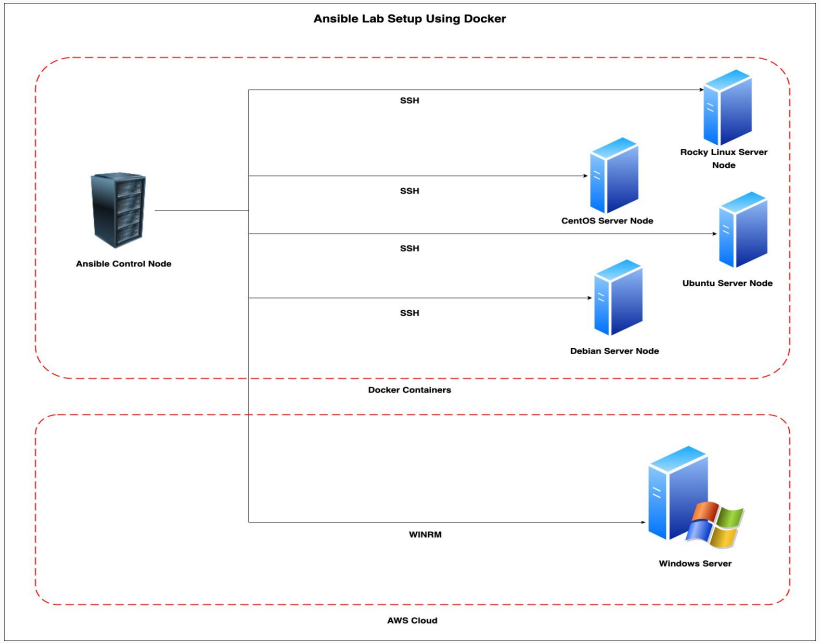
Ansible ad-hoc commands

- Example of an ad-hoc ansible task
  - Module: Yum
  - Arguments: name=bash state=installed

```
$ ansible localhost -m yum -a "pkg=bash state=installed"
localhost | SUCCESS => {
    "changed": false,
    "msg": "Nothing to do"
}
```

- What if I wanted to do more than one thing? Playbooks!

Lab Environment using Docker



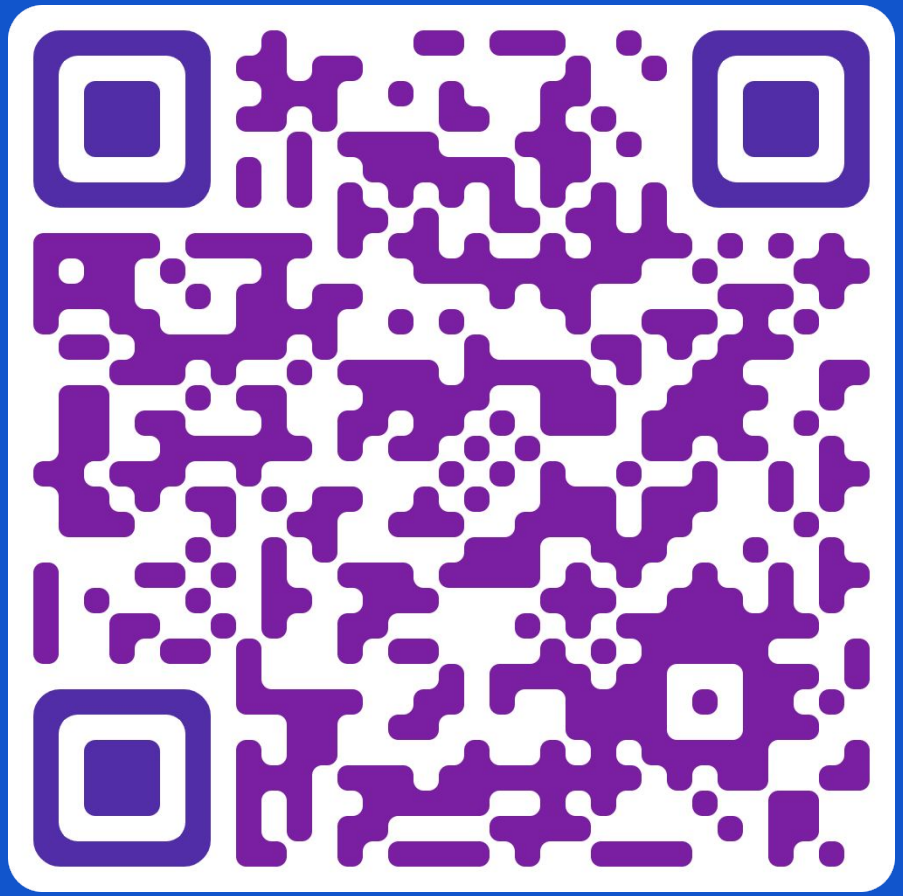

```
docker compose up -d

docker exec -it ansible-node /bin/bash
```

Lab Server Details

Server Name	Server IP	OS	Username
Ansible Node	10.10.0.2	Rocky Linux	root
Rocky Linux Node	10.10.0.11	Rocky Linux	root
CentOs Node	10.10.0.12	CentOS 7	root
Ubuntu Node	10.10.0.13	Ubuntu 22.04	root
Debian	10.10.0.14	Debian 11	root

Github Repository



https://github.com/akshayithape-devops/Mastering-Ansible

Lab #1 : Run Your First Ansible Command

- Understand how ansible works.
- Understand ad-hoc command working.
- Understand ping module.

Ansible Inventory

Inventory to defines hosts and groups of hosts

- Special "all" group that is implicitly defined as the sum of all hosts in your inventory.
- Also, “localhost” is a built-in and does not need to be defined
- We used SSH for linux machine & WINRM for windows machine
- Using ansible_user, we can define remote user.

```
[appservers]
app1.example.com
app1.example.com

[webservers]
webserver1.example.com
webserver2.example.com
```

```
[targets]

localhost              ansible_connection=local
other1.example.com     ansible_connection=ssh        ansible_user=myuser
other2.example.com     ansible_connection=ssh        ansible_user=myotheruser
```

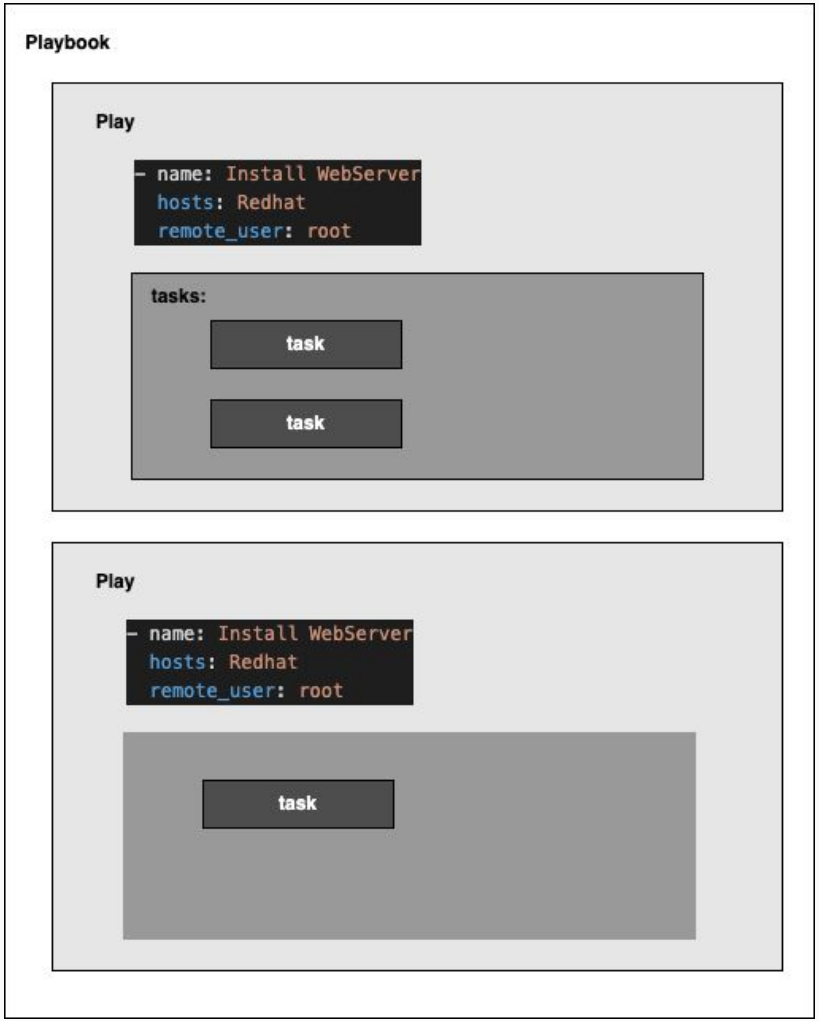
Ansible Playbooks

Playbooks are a way to combine many tasks, written in YAML, to be carried out against one or many hosts.

```
---
- name: common things to run on all hosts
  hosts: all
  tasks:
    - name: make sure bash is installed
      yum:
        pkg: bash
        state: installed
```

```
- name: webserver-only tasks
  hosts: webservers
  tasks:
    - name: start and enable httpd service
      service:
        name: httpd
        state: started
        enabled: yes
```

Ansible Playbook Structure



Lab #2 : Write First Ansible Playbook

- Write ansible inventory file.
- Write ansible ansible playbook.
- Run ansible playbook.

Lab #3 : Update Ansible Playbook to Support Debian Nodes

- Update playbook.yml to support debian nodes

Ansible Playbook Structure

```yaml
---

- name: Install WebServer
  hosts: Redhat
  remote_user: root

  tasks:
    - name: Ensure Apache is at the latest version
      yum:
        name: httpd
        state: latest
    - name: Ensure Apache is enable & running
      service:
        name: httpd
        state: started
        enabled: yes
```

Ansible Playbook Structure

```yaml
---

- name: Install WebServer on Redhat Distro
  hosts: Redhat
  remote_user: root

  tasks:
    - name: Ensure Apache is at the latest version
      yum:
        name: httpd
        state: latest
    - name: Ensure Apache is enable & running
      service:
        name: httpd
        state: started
        enabled: yes

- name: Install WebServer on Debian Distro
  hosts: Debian
  remote_user: root

  tasks:
    - name: Ensure Apache is at the latest version
      apt:
        name: apache2
        state: latest
    - name: Ensure Apache is enable & running
      service:
        name: apache2
        state: started
        enabled: yes
```

Roles

- Roles are reusable logical groupings of tasks that (normally) define a service
  - Role-level subdirs for namespaced variable defaults, files, templates, and handlers
  - Can pass variables to roles to modify behavior per-use
  - Searched for and/or shared via Ansible Galaxy
    - https://galaxy.ansible.com/

```
$ ansible localhost -m yum -a "pkg=bash state=installed"
localhost | SUCCESS => {
    "changed": false,
    "msg": "Nothing to do"
}
```

Typical Role Layout

```
myrole/
├── defaults
│   └── main.yml
├── files
├── handlers
│   └── main.yml
├── tasks
│   └── main.yml
├── templates
└── vars
    └── main.yml
```

Lab #4 : Install Nginx using Ansible Role

- Understand how ansible role is work.
- Install Nginx using ansible role.

QUIZ

https://konf.me/q706794

Thank You EveryOne

Be In Touch

Linkedin : https://www.linkedin.com/in/akshayithape/

Gmail : ithapeakshay.02@gmail.com

GitHub : https://github.com/akshayithape-devops

Medium : https://akshayithape.medium.com/