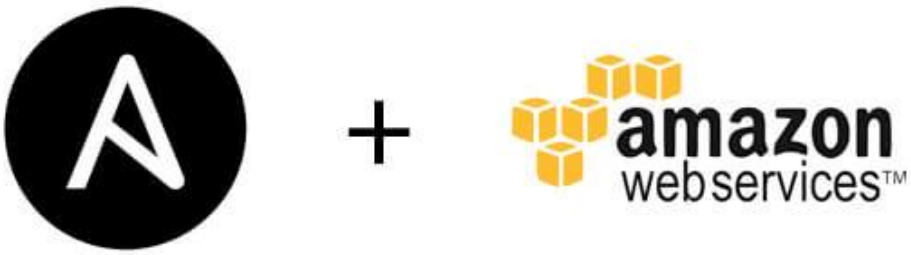


What is Ansible?

Ansible is a simple IT automation engine that automates cloud provisioning, configuration management, application deployment, intra-service orchestration, and many other IT needs.

This tool is very simple to use yet powerful enough to automate complex multi-tier IT application environments.

I assume you have Ansible installed on either your workstation or an Amazon EC2 instance — Ansible has great documentation for installation…(http://docs.ansible.com/intro_installation.html)

(http://docs.ansible.com/intro_getting_started.html)

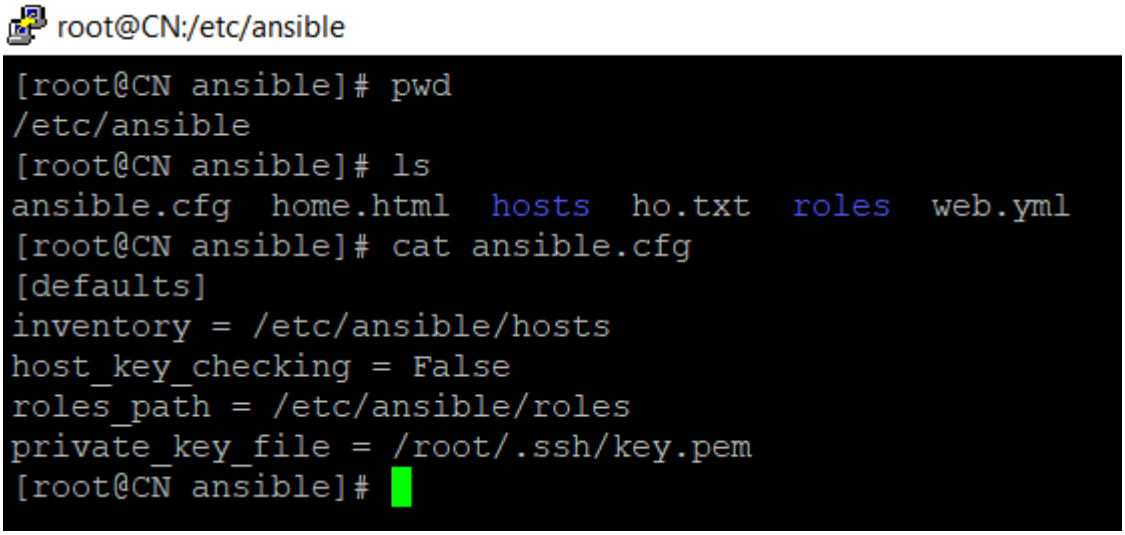
Now Let's first discuss What we are going to learn today...

Steps:

1. **Provision EC2 instance through Ansible.**
2. **Fetch the public IP using the EC2 Dynamic Inventory concept.**
3. **Configure Webserver through Ansible on EC2.**

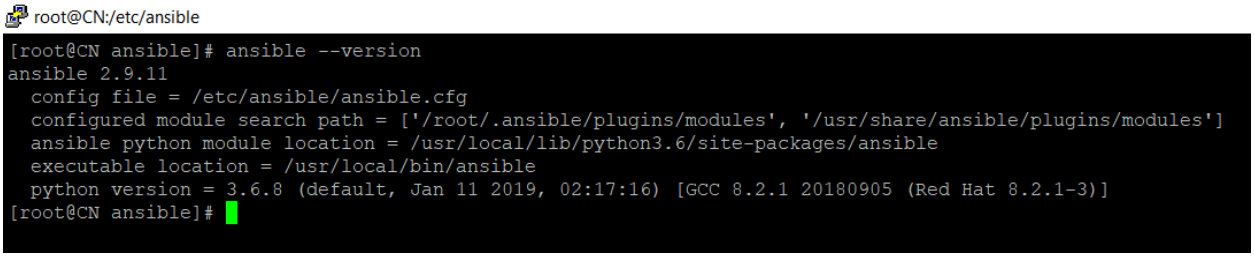
Prerequisites:

- I have chosen to use RHEL-8 for my Ansible "Master" in my local Virtual Machine.
- Having Ansible installed with Python3. I am using pip command to install Ansible. After that, you need to create directory and file manually for the ansible config file and inventory file.



You need to create **ansible.cfg** file. Here you define your inventory directory, roles directory etc. You can check from above image.

After that, for checking run command ***ansible — version.***



- Install Boto and Boto3 python library.

pip3 install boto

pip3 install boto3

- Need one IAM role in AWS because **boto** can automatically source my AWS API credentials provided by an Amazon EC2 Identity and Access Management (IAM) role to run the playbook.

We need to provide AWS credentials so that Ansible fetch the account details. For providing AWS credential, We have multiple Way…

One way, you will need to set environment variables for your Secret and Access key:

export AWS_ACCESS_KEY_ID='YOUR_AWS_API_KEY'

export AWS_SECRET_ACCESS_KEY='YOUR_AWS_API_SECRET_KEY'

And another way is for providing Credential is by using **awscli** software. You only need to download software and provide AWS_ACCESS_KEY_ID and AWS_SECRET_ACCESS_KEY using command ***aws configure.***

Setting up Dynamic Inventory:

Now to get started with **dynamic inventory** management, you'll need to grab the **EC2.py** script and the **EC2.ini** config file. The EC2.py script is written using the Boto EC2 library and will query AWS for your running Amazon EC2 instances. The EC2.ini file is the config file for EC2.py and can be used to limit the scope of Ansible's reach. You can specify the regions, instance tags, or roles that the EC2.py script will find.

Link for download EC2.py and EC2.ini file —

(https://raw.githubusercontent.com/ansible/ansible/stable-2.9/contrib/inventory/ec2.py)

(https://raw.githubusercontent.com/ansible/ansible/devel/contrib/inventory/ec2.ini)

Use wget/curl/Git to download these files in directory /etc/ansible/hosts. I had to create this directory manually because I installed Ansible using pip. After that, make this file executable using command...

chmod +x ec2.py

chmod +x ec2.ini

Now you also need to set the path for these files, so that Ansible fetch the path of inventory script.

export ANSIBLE_HOSTS=/path to/ec2.py

export EC2_INI_PATH=/path to/ec2.ini

As I'm using Linux, we need to do a small patch in the ec2.py file. By default, it gives executable location ***#!/usr/bin/env python,*** modify it to ***#!usr/bin/env python3.***

Now my dynamic Inventory is configured.

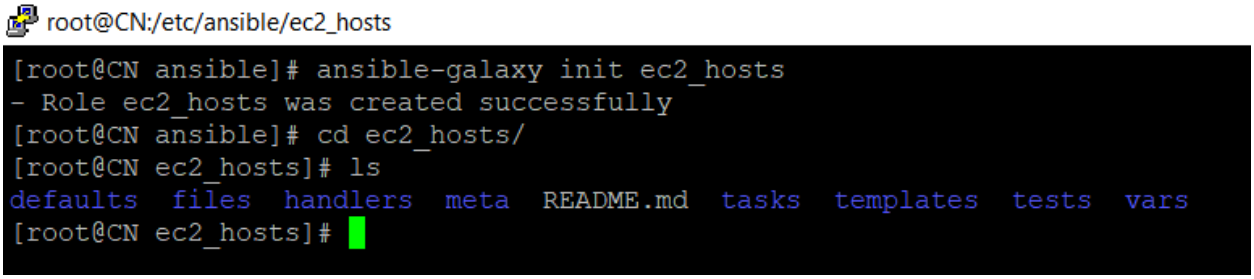
Now we ready to see Ansible shine.

Now come to task, which we are going to perform…

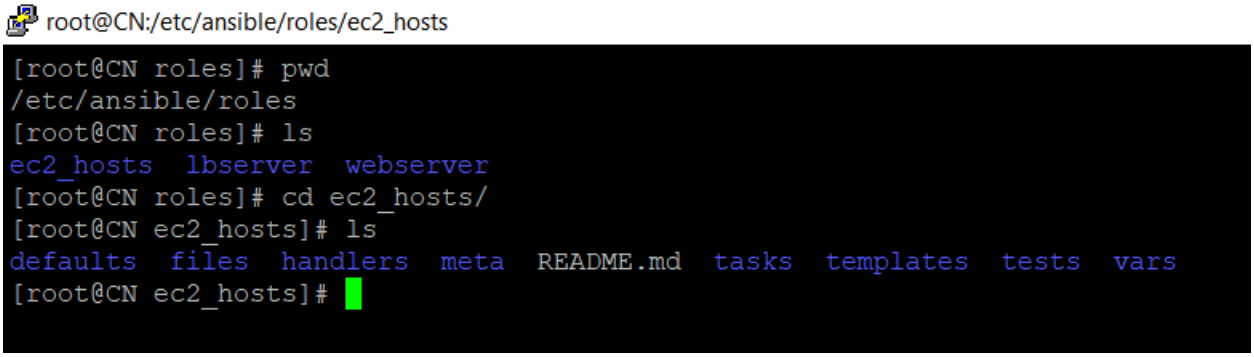
1. Provision EC2 instance through Ansible:

I am creating Ansible role for performing the task. For creating Ansible roles, run command

ansible-galaxy init role_name



This create all the above directory, you only need to put the tasks, vars and other files in their preferred location.



Now let's discuss the code…

```
# tasks file for webserver
- name: Create Key Pair
ec2_key:
name: mykey
aws_region: "{{ region }}"
register: ec2_key
- name: Copy Key to Local File
copy:
content: "{{ ec2_key.key.private_key }}"
dest: "{{ key_dest }}"
mode: '0600'
```

This code creates a **key_pair** and save it locally.

I also need to create one security_group, for this…

```
- name: Create Security Group — Allow SSh, HTTP
ec2_group:
name: sg_ansible-1
description: sg for ansible inventory
region: “{{ region }}”
rules:
— proto: tcp
from_port: 80
to_port: 80
cidr_ip: 0.0.0.0/0
— proto: tcp
from_port: 22
to_port: 22
cidr_ip: 0.0.0.0/0
rules_egress:
```

— proto: all
cidr_ip: 0.0.0.0/0

This security group we attached in our ec2_instance and, this allow ssh and http protocol.

Now let's see the code for launching the ec2_instance which we are going to use as managed_node.

- name: Launch EC2 Instance
ec2:
key_name: mykey
instance_type: t2.micro
image: "{{ image_id }}"
wait: yes
region: "{{ region }}"
count: 1
vpc_subnet_id: subnet-040fe014984c437d9
group_id: "{{ sg_ansible.group_id }}"
assign_public_ip: yes

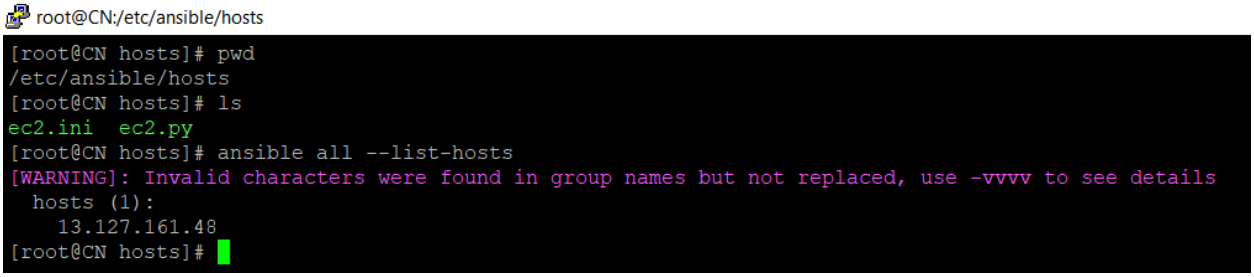
state: present
register: ec2

Now my ec2_instance is provisioned.

2. Fetch the public IP using the EC2 Dynamic Inventory concept:

We already configured the Dynamic Inventory above. Now for check how many instances are running, run command i.e.

ansible all — list-hosts



This is the IP of my ec2_instance.

3. Configure WebServer through Ansible on EC2:

I'm creating one more role for configuring the apache webserver on the top of ec2_instance.

HERE is the code for this...

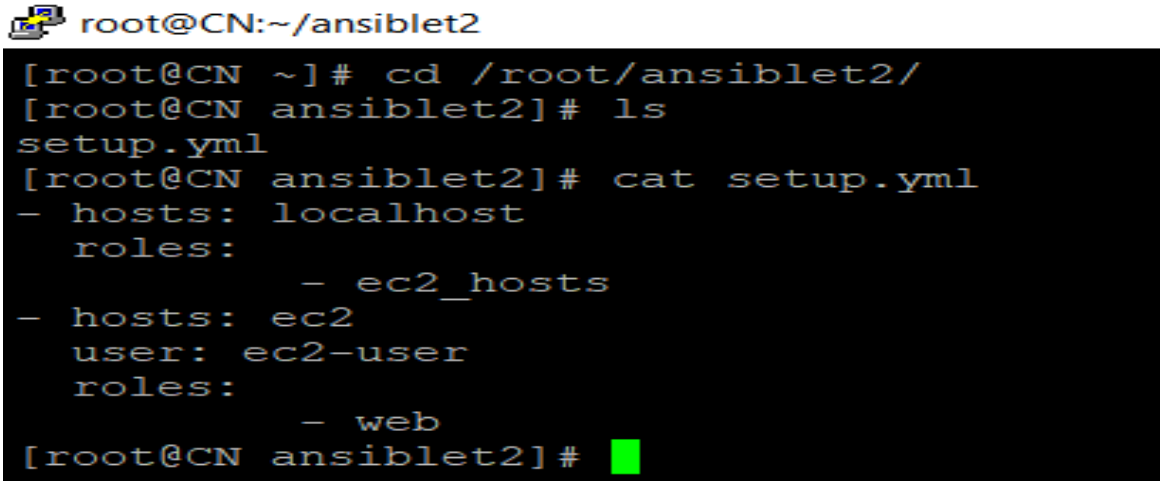
root@CN:/etc/ansible/roles/web/tasks

```
[root@CN ec2_hosts]# cd /etc/ansible/
[root@CN ansible]# cd roles/web
[root@CN web]# cd tasks/
[root@CN tasks]# vim main.yml
[root@CN tasks]# cat main.yml
---
# tasks file for web

- name: Install Required Package
  package:
    name: python3
    state: present
  become: true
- name: Install Apache Server
  package:
    name: httpd
    state: present
  become: true
- name: copy web page from url
  get_url:
    dest: "/var/www/html"
    url: "https://raw.githubusercontent.com/Anuddeeph/AWS_WS_Using_Ansible/master/index.html"
  become: true
- name: Start Apache Service
  service:
    name: httpd
    state: started
  become: true

[root@CN tasks]#
```

Now for running both the role, I created one playbook which run first role in localhost and another in ec2_instance.



But IF you try to run this playbook, it gets failed after running the first playbook. Because, by default, Ansible doesn't refresh the inventory in the mid of playbook running. For this, we need to run one more module named ***refresh_inventory.*** I put this task in my ec2_host role, Code for this...

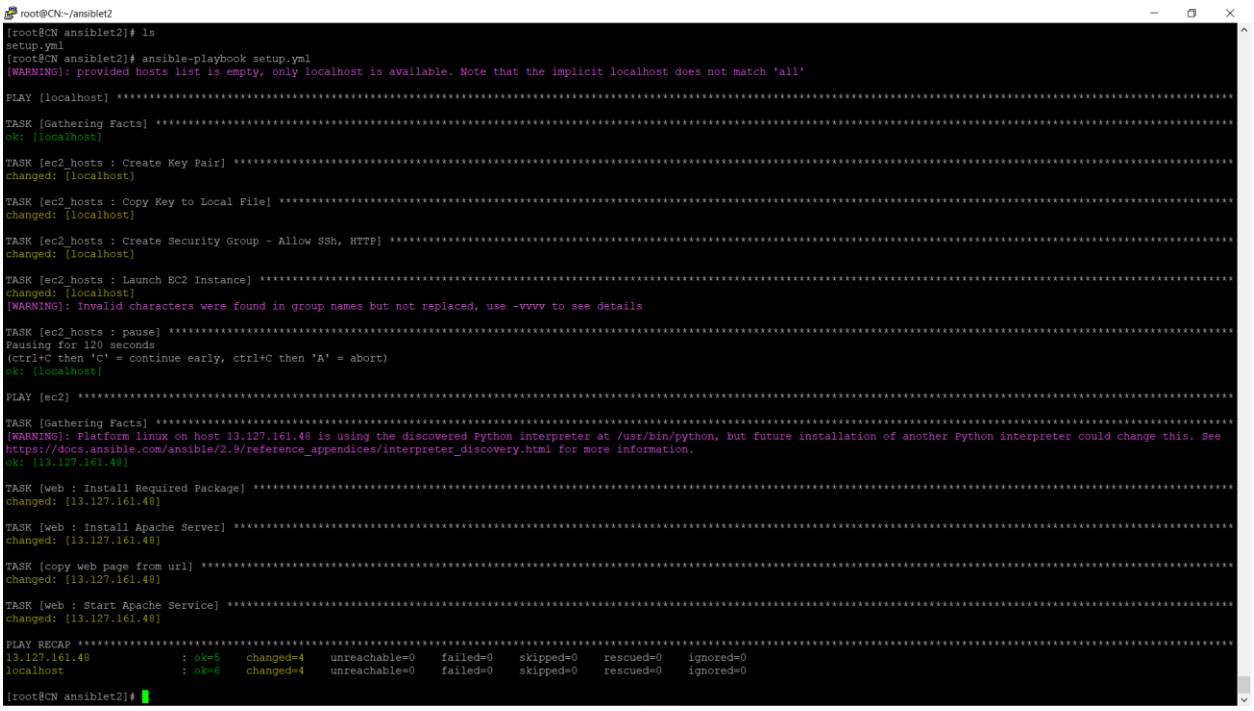
```
- name: Refresh Inventory File
meta: refresh_inventory
- pause:
minutes: 2
```

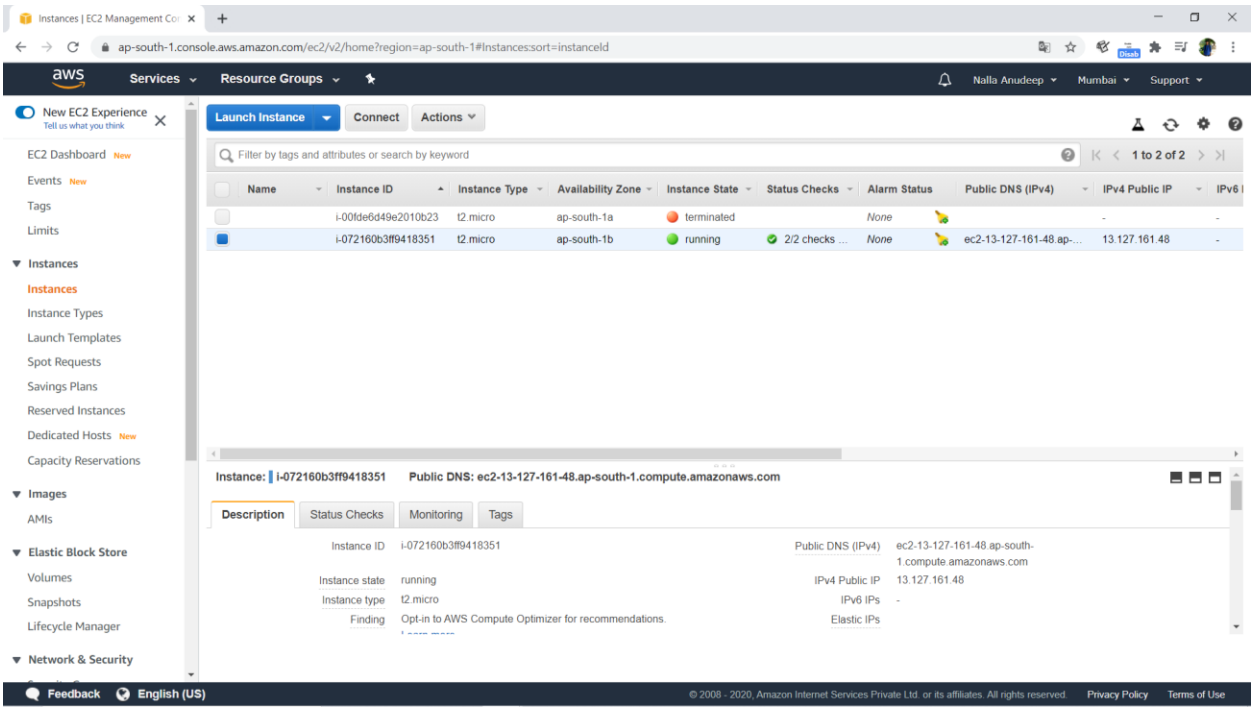
Now, when you run the playbook, it run smoothly without giving any error...

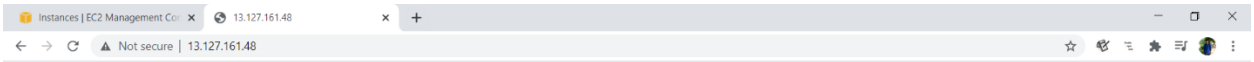
YO, Finally OUR WEBSITE IS LAUNCHED...

GitHub Link:
https://github.com/Anuddeeph/AWS_WS_Using_Ansible.git

Let us see output of this...







Finally Done! Launch Webserver in AWS using Ansible