TASK 2 ANSIBLE

INTODUCTION

ANSIBLE

Ansible is an open-source software provisioning, configuration management, and application-deployment tool enabling infrastructure as code. It runs on many Unix-like systems, and can configure both Unix-like systems as well as Microsoft Windows. It includes its own declarative language to describe system configuration.

AWS

Amazon web service is a platform that offers flexible, reliable, scalable, easy-to-use and cost-effective cloud computing solutions. AWS is a comprehensive, easy to use computing platform offered Amazon.

WEB- SERVER

The Apache HTTP Server, colloquially called Apache , is a free and open-source cross-platform web server software, released under the terms of Apache License 2.0. Apache is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation.
The vast majority of Apache HTTP Server

instances run on a Linux distribution, but current versions also run on Microsoft Windows and a wide variety of Unix-like systems.

DESCRIPTION OF TASK

Statement : Deploy Web Server on AWS through ANSIBLE!

- Provision EC2 instance through ansible
- Retrieve the IP Address of instance using dynamic inventory concept.
- Configure the web server through ansible!
- Create role for webserver to customize the Instance and deploy the webpage to root directory.

PRE-REQUISITES

- Boto and Boto3 Libraries Installed
- RHEL 8 with ansible installed for controller node
- AWS Account

Create dynamic inventory

What is ansible inventory?

An Inventory is a collection of hosts against which jobs may be launched, the same as an Ansible inventory file. Inventories are divided into groups and these groups contain the actual hosts.

Static Host Inventory File

In **Ansible**, a static inventory file is a plain text file that contains a list of managed hosts declared under a host group using either hostnames or IP addresses.
A host group name is enclosed in square brackets i.e `[group name]`. The managed host entries are later listed below the group name, each on its own line. As discussed earlier, the hosts are listed using either hostnames or IP addresses.

Dynamic Host Inventory File

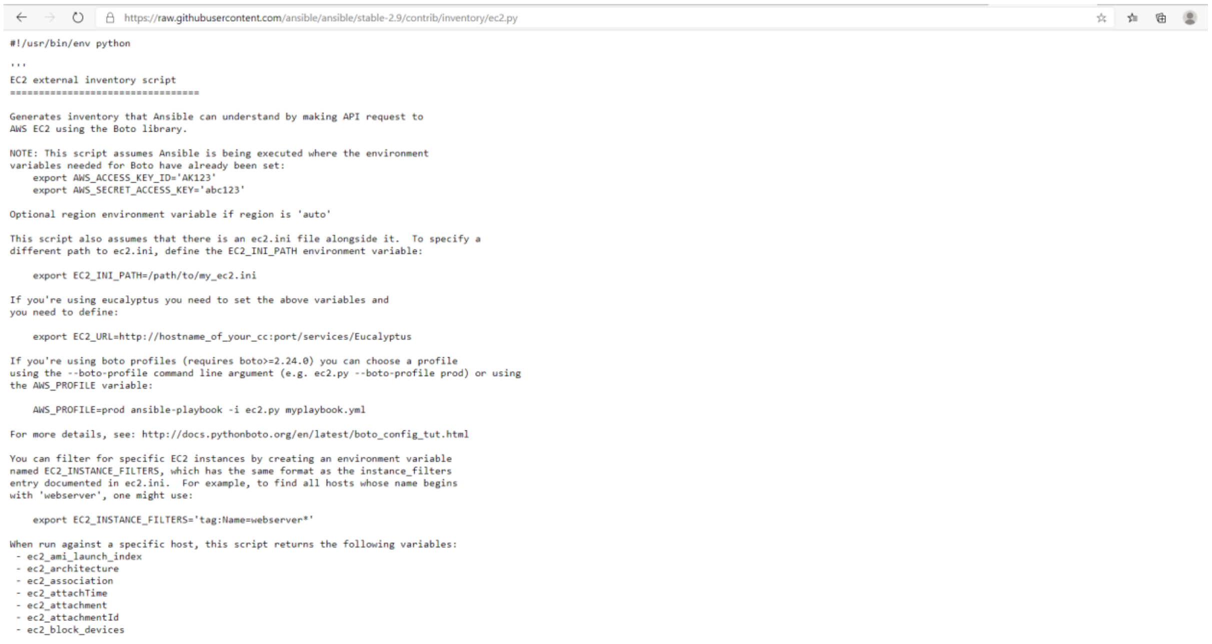
In a configuration – especially a cloud setup such as **AWS** where the inventory file keeps constantly changing as you add or decommission servers, keeping tabs on the hosts defined in the inventory file becomes a real challenge. It becomes inconvenient going back to the host file and updating the list of hosts with their IP addresses.
And this is where a **dynamic inventory** comes to play. So what is a dynamic inventory? A dynamic inventory is a script written in Python, PHP or any other programming language. It comes in handy in cloud environments such as AWS where IP addresses change once a virtual server is stopped and started again.
Ansible already has developed inventory scripts for public cloud platforms such as Google

Compute Engine, Amazon EC2 instance, OpenStack, RackSpace, cobbler, among others.

We can get the code for dynamic inventory from this link:

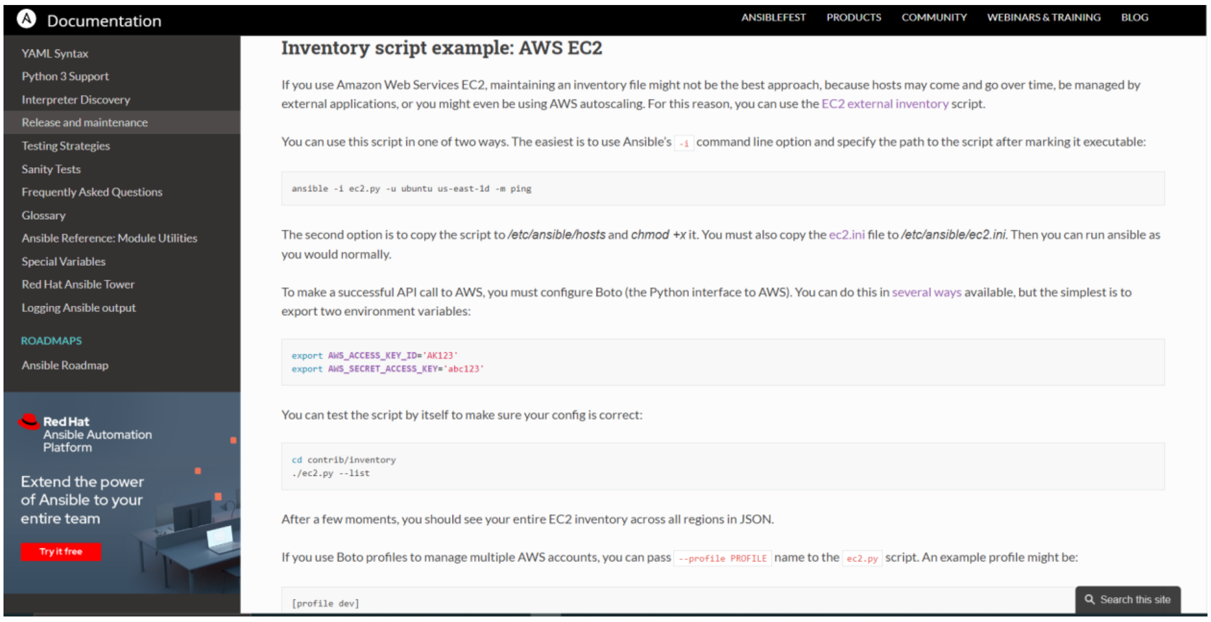
Download the code in the inventory folder

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



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

```
#!/usr/bin/env python

'''
EC2 external inventory script
=================================

Generates inventory that Ansible can understand by making API request to
AWS EC2 using the Boto library.

NOTE: This script assumes Ansible is being executed where the environment
variables needed for Boto have already been set:
    export AWS_ACCESS_KEY_ID='AK123'
    export AWS_SECRET_ACCESS_KEY='abc123'

Optional region environment variable if region is 'auto'

This script also assumes that there is an ec2.ini file alongside it.  To specify a
different path to ec2.ini, define the EC2_INI_PATH environment variable:

    export EC2_INI_PATH=/path/to/my_ec2.ini

If you're using eucalyptus you need to set the above variables and
you need to define:

    export EC2_URL=http://hostname_of_your_cc:port/services/Eucalyptus

If you're using boto profiles (requires boto>=2.24.0) you can choose a profile
using the --boto-profile command line argument (e.g. ec2.py --boto-profile prod) or using
the AWS_PROFILE variable:

    AWS_PROFILE=prod ansible-playbook -i ec2.py myplaybook.yml

For more details, see: http://docs.pythonboto.org/en/latest/boto_config_tut.html

You can filter for specific EC2 instances by creating an environment variable
named EC2_INSTANCE_FILTERS, which has the same format as the instance_filters
entry documented in ec2.ini.  For example, to find all hosts whose name begins
with 'webserver', one might use:

    export EC2_INSTANCE_FILTERS='tag:Name=webserver*'

When run against a specific host, this script returns the following variables:
 - ec2_ami_launch_index
 - ec2_architecture
 - ec2_association
 - ec2_attachTime
 - ec2_attachment
 - ec2_attachmentId
 - ec2_block_devices
```



Inventory script example: AWS EC2

If you use Amazon Web Services EC2, maintaining an inventory file might not be the best approach, because hosts may come and go over time, be managed by external applications, or you might even be using AWS autoscaling. For this reason, you can use the EC2 external inventory script.

You can use this script in one of two ways. The easiest is to use Ansible's `-i` command line option and specify the path to the script after marking it executable:

```
ansible -i ec2.py -u ubuntu us-east-1d -m ping
```

The second option is to copy the script to */etc/ansible/hosts* and *chmod +x* it. You must also copy the ec2.ini file to */etc/ansible/ec2.ini*. Then you can run ansible as you would normally.

To make a successful API call to AWS, you must configure Boto (the Python interface to AWS). You can do this in several ways available, but the simplest is to export two environment variables:

```
export AWS_ACCESS_KEY_ID='AK123'
export AWS_SECRET_ACCESS_KEY='abc123'
```

You can test the script by itself to make sure your config is correct:

```
cd contrib/inventory
./ec2.py --list
```

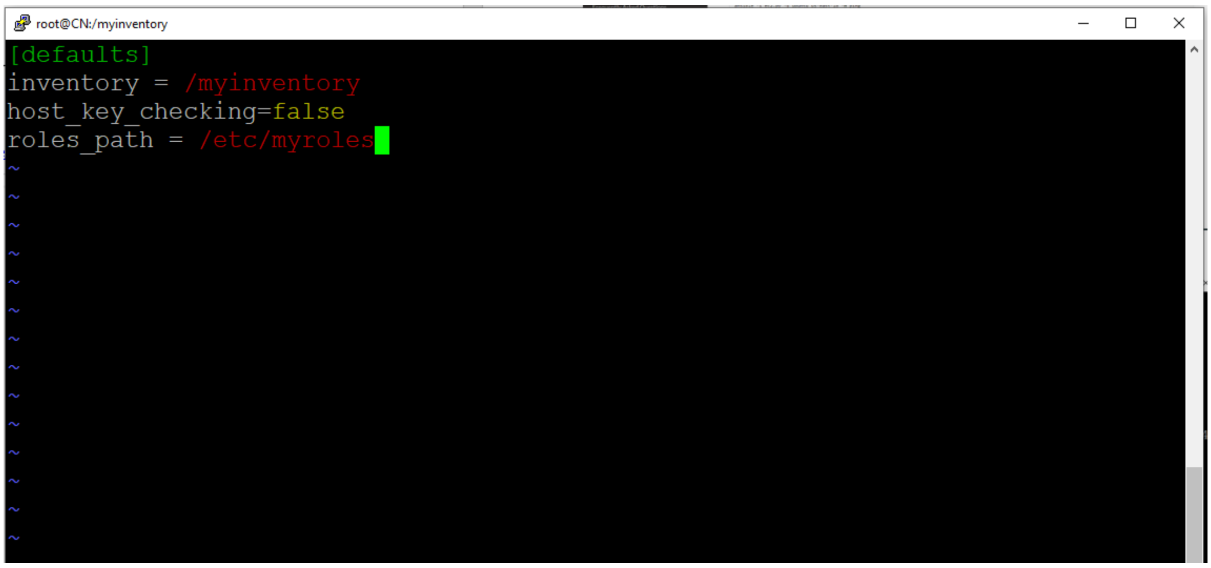
After a few moments, you should see your entire EC2 inventory across all regions in JSON.

If you use Boto profiles to manage multiple AWS accounts, you can pass `--profile PROFILE` name to the `ec2.py` script. An example profile might be:

```
[profile dev]
```

Update config file

Change the path of the inventory with the dynamic inventory folder.

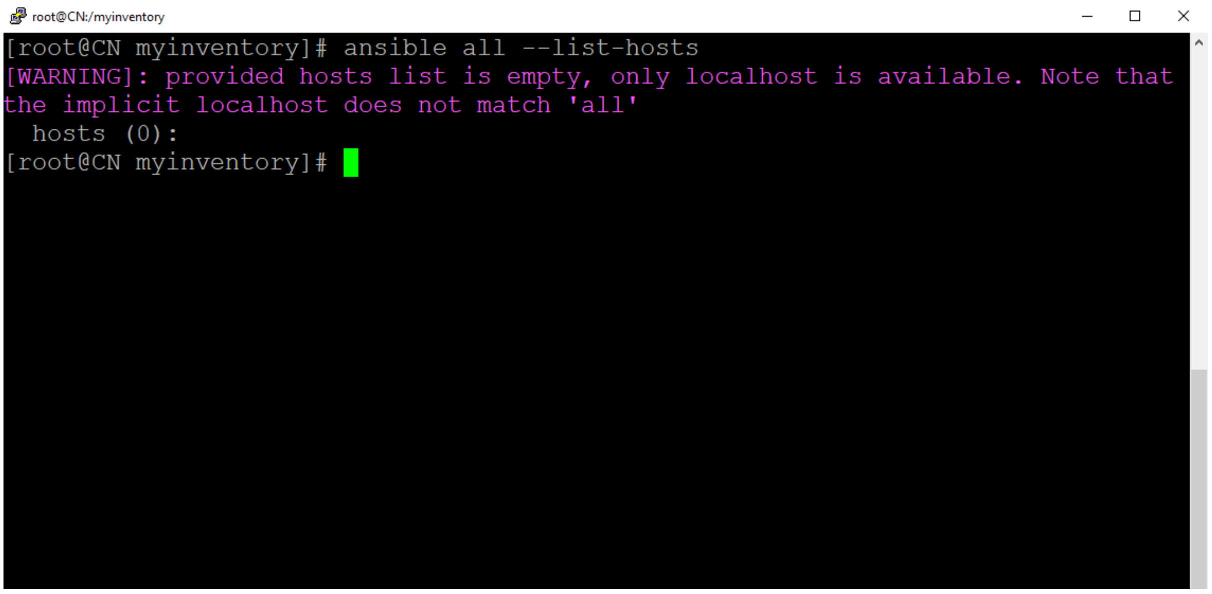


```
[defaults]
inventory = /myinventory
host_key_checking=false
roles_path = /etc/myroles
~
~
~
~
~
~
~
~
~
~
~
~
~
~
```

Checking for hosts

Now to check that the file is working or not we can run the command

ansible all –list-hosts



```
[root@CN myinventory]# ansible all --list-hosts
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'
  hosts (0):
[root@CN myinventory]#
```

Dynamic inventory working fine.

ROLE in ANSIBLE

Roles provide a framework for fully independent, or interdependent collections of variables, tasks, files, templates, and modules.

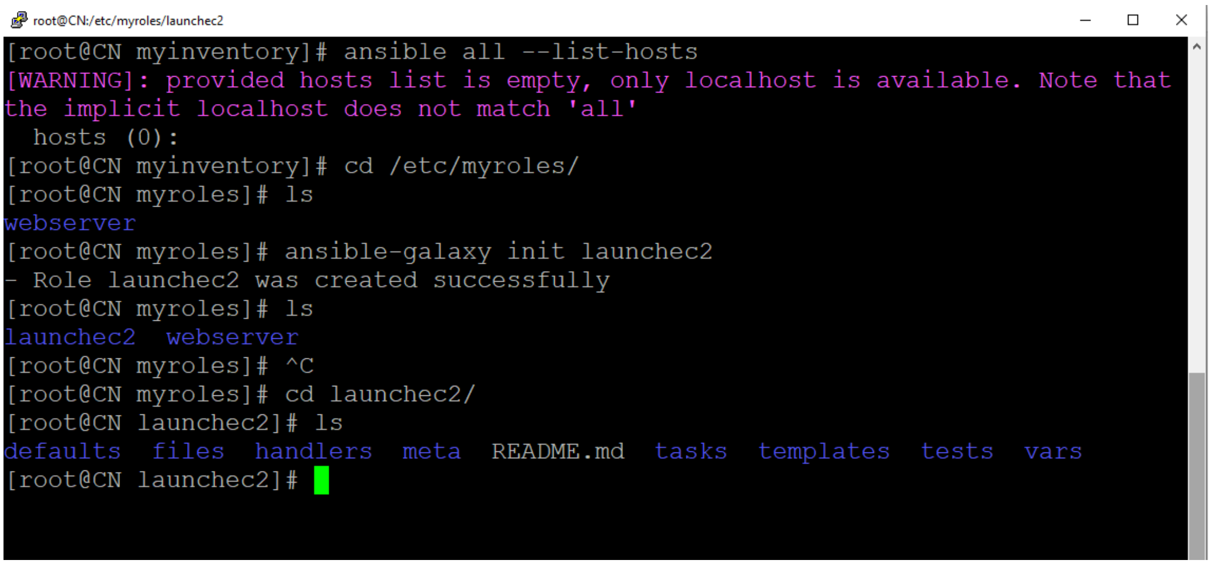
In Ansible, the role is the primary mechanism for breaking a playbook into multiple files. This simplifies writing **complex playbooks**, and it makes them easier to reuse. The breaking of playbook allows you to logically break the playbook into reusable components.

Each role is basically limited to a particular functionality or desired output, with all the necessary steps to provide that result either within that role itself or in other roles listed as dependencies.

Roles are not playbooks. Roles are small functionality which can be independently used but have to be used within playbooks. There is no way to directly execute a role. Roles have no explicit setting for which host the role will apply to.

Top-level playbooks are the bridge holding the hosts from your inventory file to roles that should be applied to those hosts.

Create role for ec2 instance



```
[root@CN myinventory]# ansible all --list-hosts
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'
  hosts (0):
[root@CN myinventory]# cd /etc/myroles/
[root@CN myroles]# ls
webserver
[root@CN myroles]# ansible-galaxy init launchec2
- Role launchec2 was created successfully
[root@CN myroles]# ls
launchec2  webserver
[root@CN myroles]# ^C
[root@CN myroles]# cd launchec2/
[root@CN launchec2]# ls
defaults  files  handlers  meta  README.md  tasks  templates  tests  vars
[root@CN launchec2]# 
```

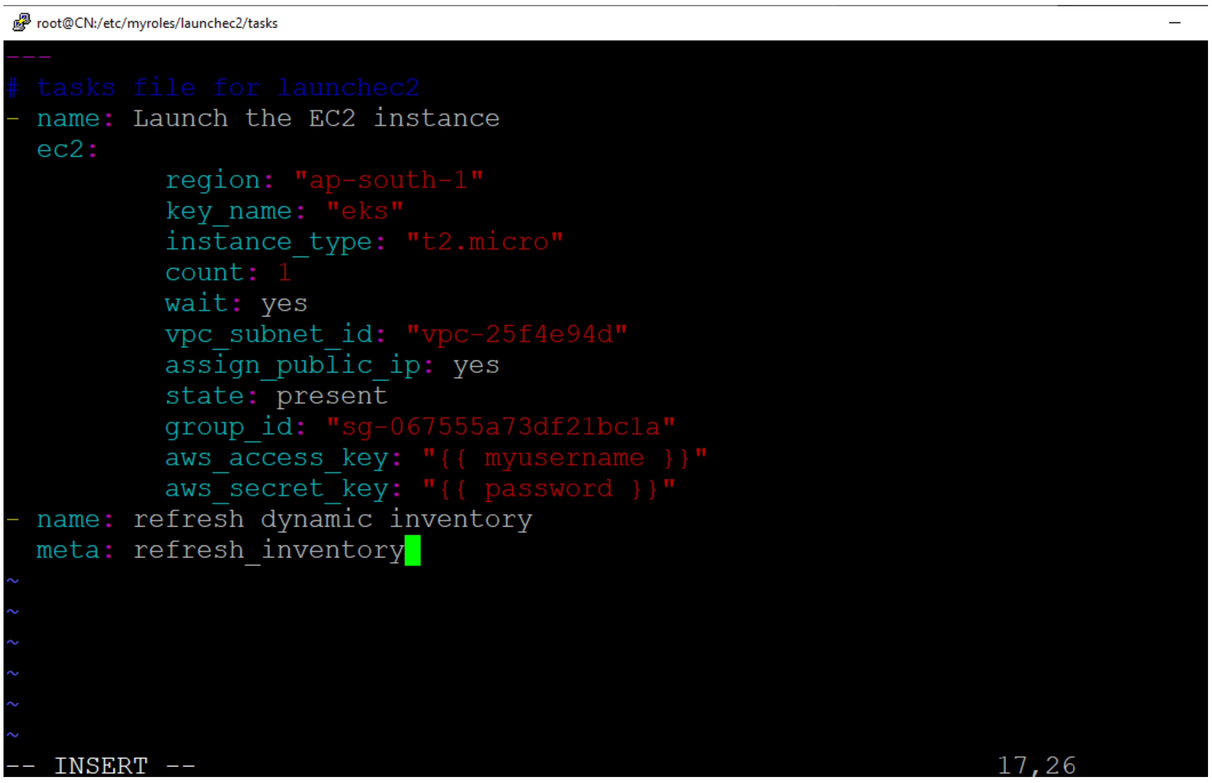
We can create the role using the command:

ansible-galaxy init launchec2

Configure role for ec2

Tasks.yml – it contains all the tasks related to launching of the ec2 instance.

And vault secure.yml – it is used for storing the AWS Credentials.


root@CN:/etc/myroles/launchec2/tasks

```
---
# tasks file for launchec2
- name: Launch the EC2 instance
  ec2:
          region: "ap-south-1"
          key_name: "eks"
          instance_type: "t2.micro"
          count: 1
          wait: yes
          vpc_subnet_id: "vpc-25f4e94d"
          assign_public_ip: yes
          state: present
          group_id: "sg-067555a73df21bc1a"
          aws_access_key: "{{ myusername }}"
          aws_secret_key: "{{ password }}"
- name: refresh dynamic inventory
  meta: refresh_inventory
~
~
~
~
~
~
-- INSERT --                                                  17,26
```

Role for webserver

We can create the role for webserver using the command

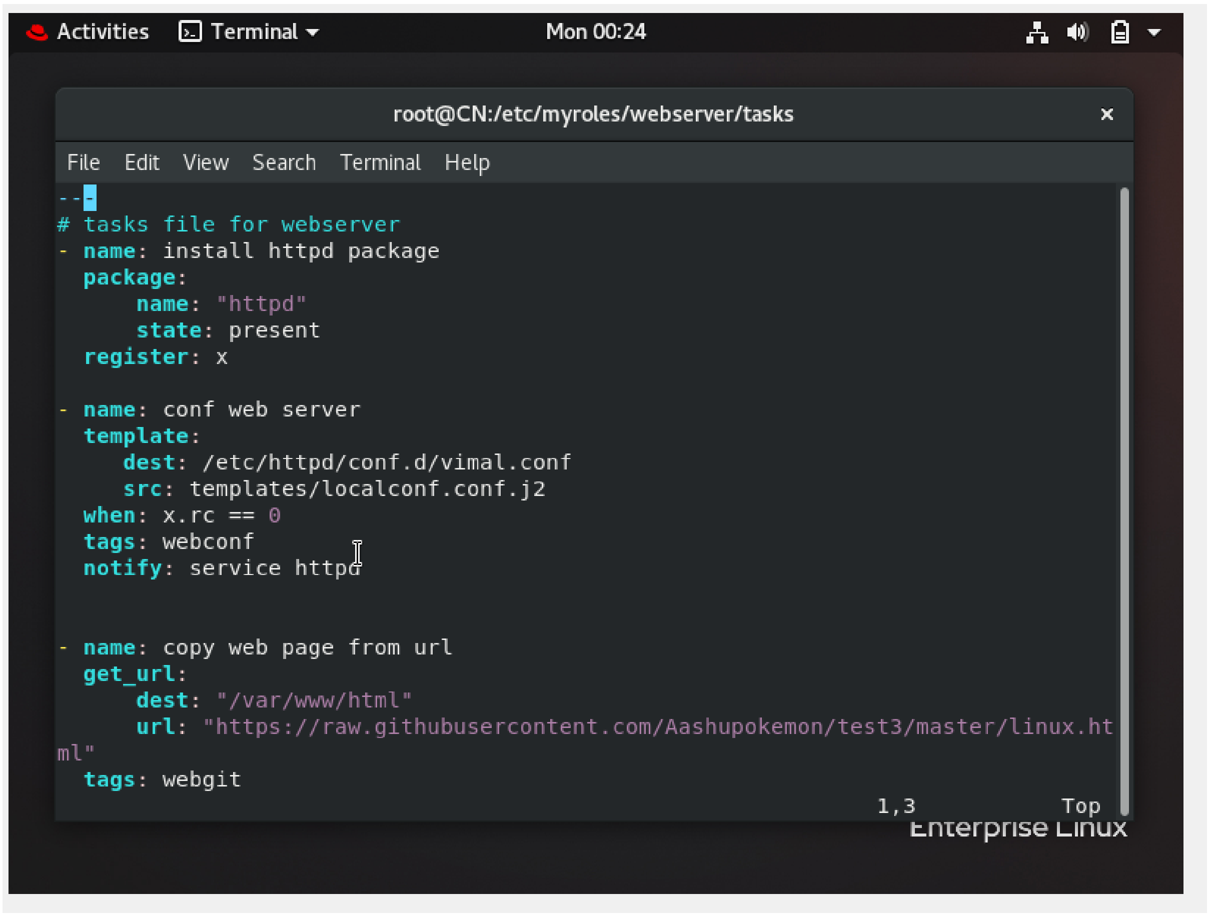
ansible-galaxy init webserver

Write all the tasks in the task file and variables in the vars file also handlers in handler file.

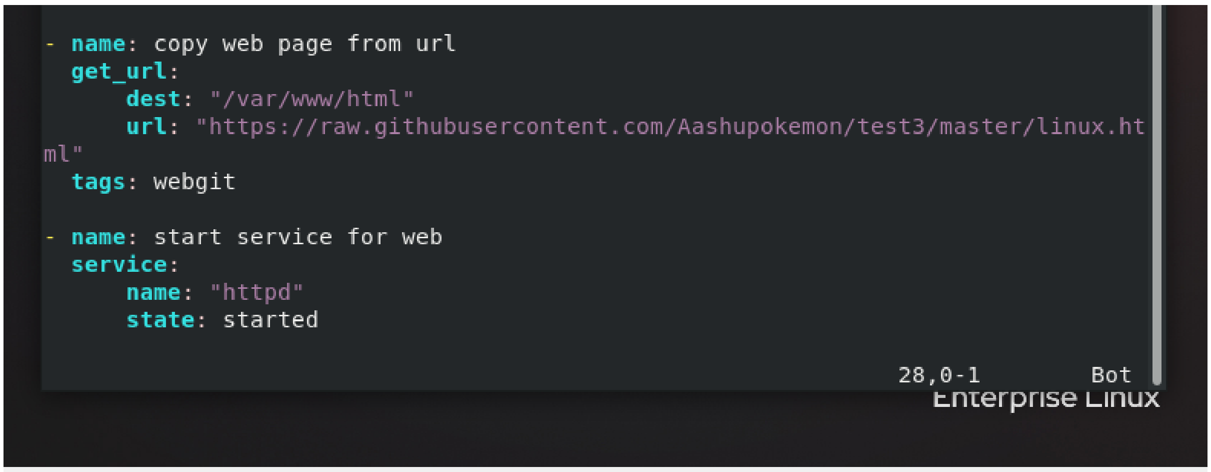
Ansible will automatically run all the files accordingly.

We have copied the code for webpage from the Github

Into the document root.

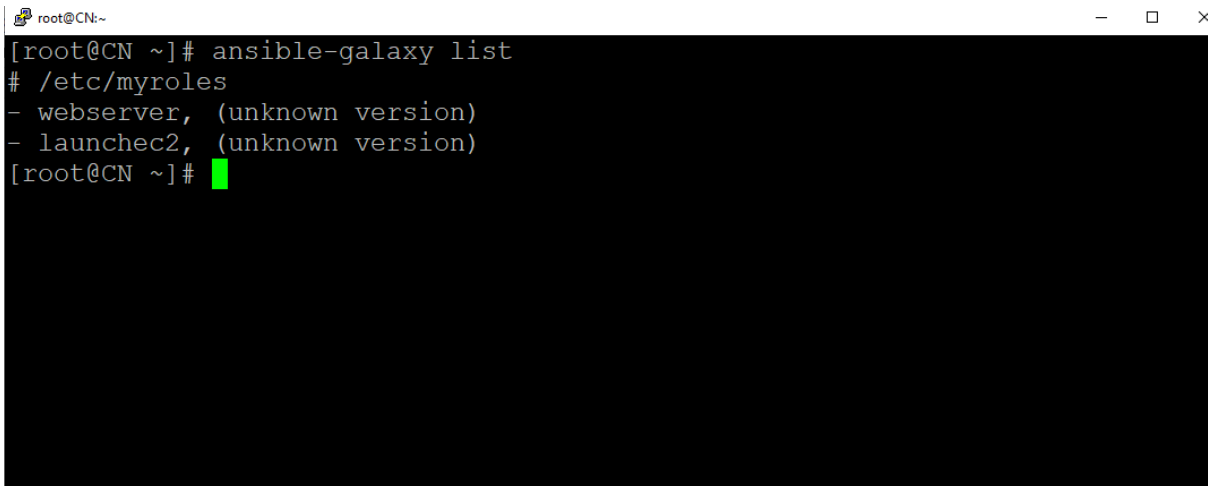

Activities
Terminal
Mon 00:24
root@CN:/etc/myroles/webserver/tasks
File Edit View Search Terminal Help
Enterprise Linux

```
---
# tasks file for webserver
- name: install httpd package
  package:
      name: "httpd"
      state: present
  register: x

- name: conf web server
  template:
     dest: /etc/httpd/conf.d/vimal.conf
     src: templates/localconf.conf.j2
  when: x.rc == 0
  tags: webconf
  notify: service httpd


- name: copy web page from url
  get_url:
      dest: "/var/www/html"
      url: "https://raw.githubusercontent.com/Aashupokemon/test3/master/linux.ht
ml"
  tags: webgit
                                                                1,3           Top
```



```
- name: copy web page from url
  get_url:
      dest: "/var/www/html"
      url: "https://raw.githubusercontent.com/Aashupokemon/test3/master/linux.ht
ml"
  tags: webgit

- name: start service for web
  service:
      name: "httpd"
      state: started

                                                                28,0-1        Bot
```

To check all the roles we use the following command

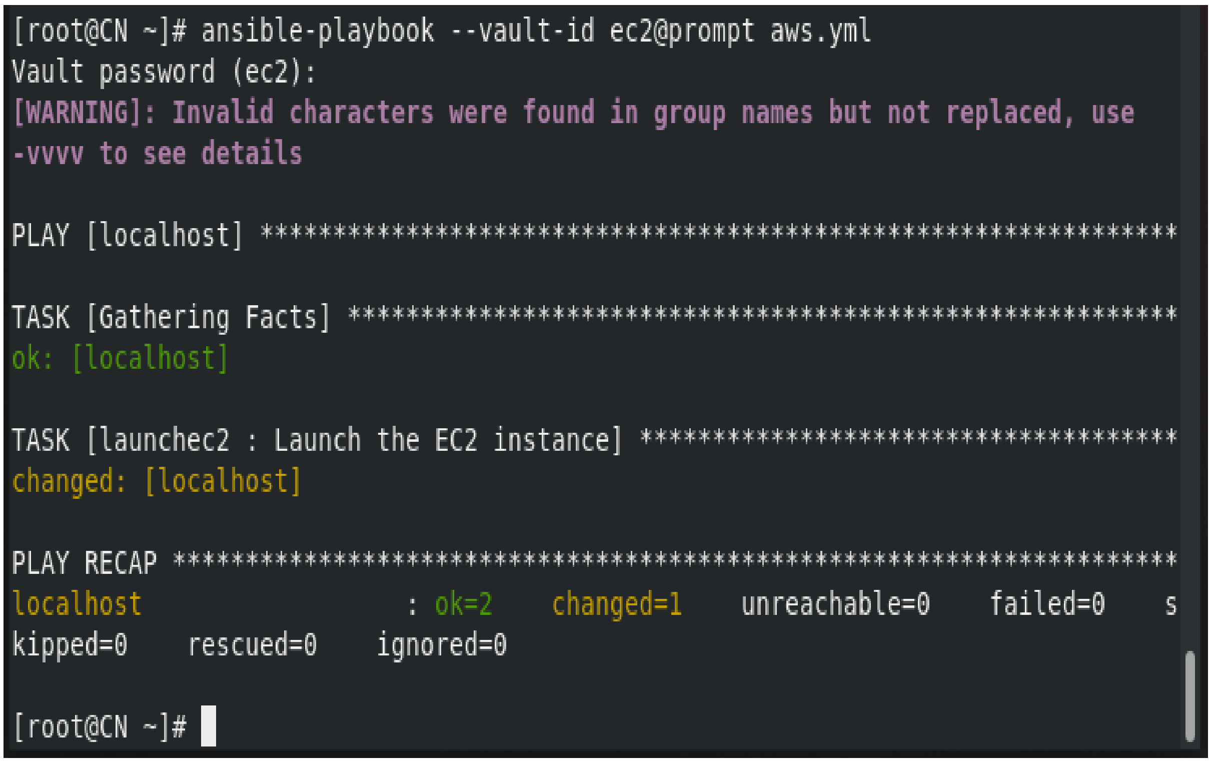


```
[root@CN ~]# ansible-galaxy list
# /etc/myroles
- webserver, (unknown version)
- launchec2, (unknown version)
[root@CN ~]#
```

Now we can see two roles are present.

CREATING PLAYBOOK

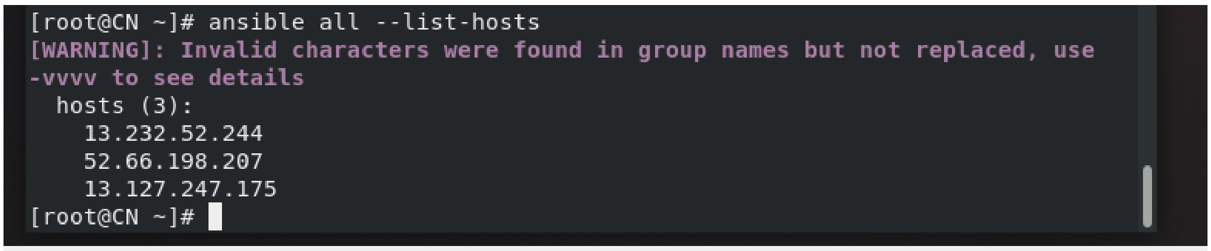
Now we will write a playbook to run the roles and will confgure the whole setup in just one click.



```
[root@CN ~]# ansible-playbook --vault-id ec2@prompt aws.yml
Vault password (ec2):
[WARNING]: Invalid characters were found in group names but not replaced, use
-vvvv to see details

PLAY [localhost] **************************************************************

TASK [Gathering Facts] ********************************************************
ok: [localhost]

TASK [launchec2 : Launch the EC2 instance] ************************************
changed: [localhost]

PLAY RECAP ********************************************************************
localhost                  : ok=2    changed=1    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0

[root@CN ~]#
```

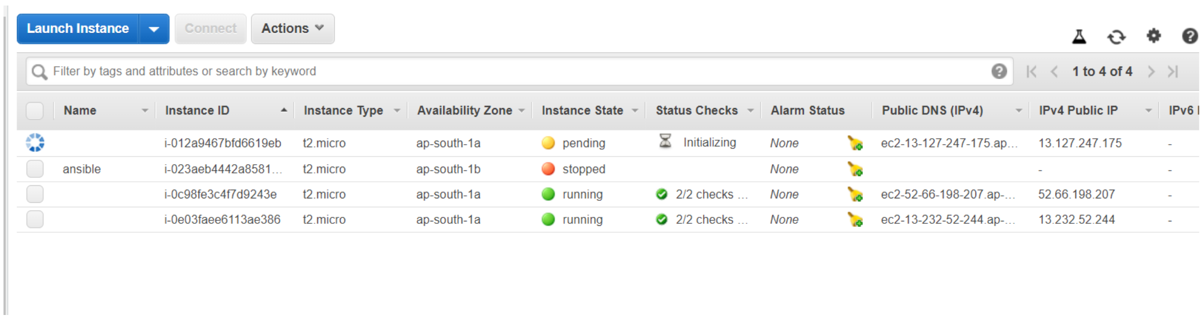
As we launch the instance the inventory will be updated.

Inventory dynamically updated we can see below.

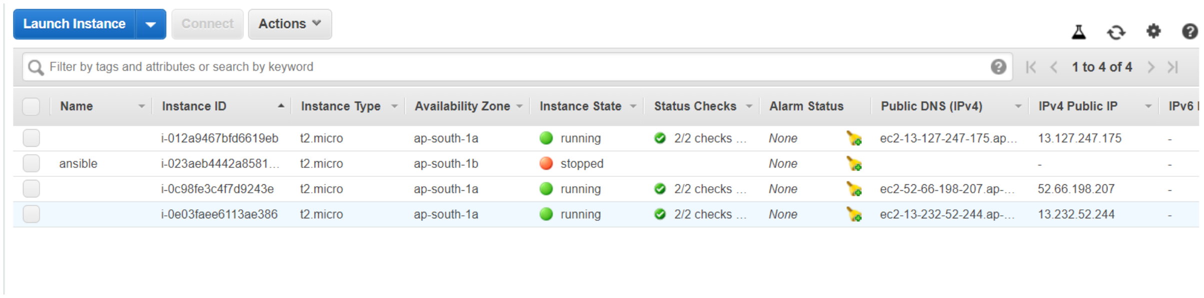


```
[root@CN ~]# ansible all --list-hosts
[WARNING]: Invalid characters were found in group names but not replaced, use
-vvvv to see details
  hosts (3):
    13.232.52.244
    52.66.198.207
    13.127.247.175
[root@CN ~]#
```

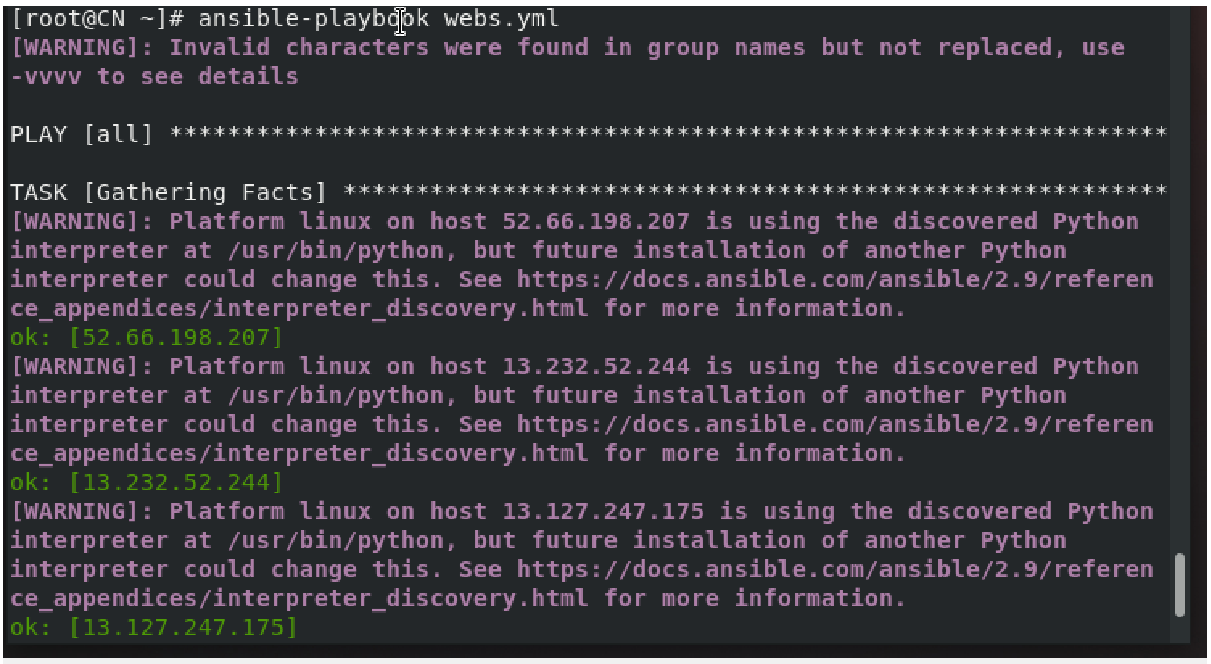
The instances being launched by the ansible in AWS cloud.



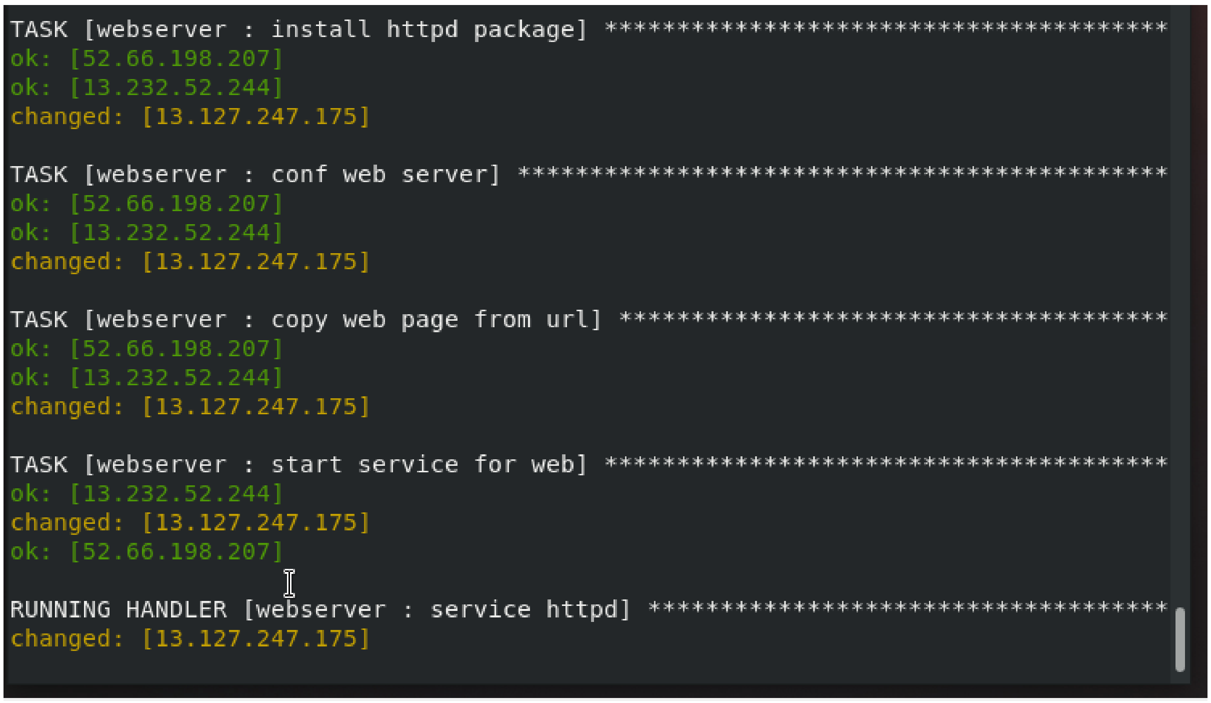
	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP	IPv6 I
		i-012a9467bfd6619eb	t2.micro	ap-south-1a	🟡 pending	Initializing	None	ec2-13-127-247-175.ap...	13.127.247.175	-
	ansible	i-023aeb4442a8581...	t2.micro	ap-south-1b	🔴 stopped		None		-	-
		i-0c98fe3c4f7d9243e	t2.micro	ap-south-1a	🟢 running	✓ 2/2 checks ...	None	ec2-52-66-198-207.ap-...	52.66.198.207	-
		i-0e03faee6113ae386	t2.micro	ap-south-1a	🟢 running	✓ 2/2 checks ...	None	ec2-13-232-52-244.ap-...	13.232.52.244	-



	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP	IPv6 I
		i-012a9467bfd6619eb	t2.micro	ap-south-1a	🟢 running	✓ 2/2 checks ...	None	ec2-13-127-247-175.ap...	13.127.247.175	-
	ansible	i-023aeb4442a8581...	t2.micro	ap-south-1b	🔴 stopped		None		-	-
		i-0c98fe3c4f7d9243e	t2.micro	ap-south-1a	🟢 running	✓ 2/2 checks ...	None	ec2-52-66-198-207.ap-...	52.66.198.207	-
		i-0e03faee6113ae386	t2.micro	ap-south-1a	🟢 running	✓ 2/2 checks ...	None	ec2-13-232-52-244.ap-...	13.232.52.244	-

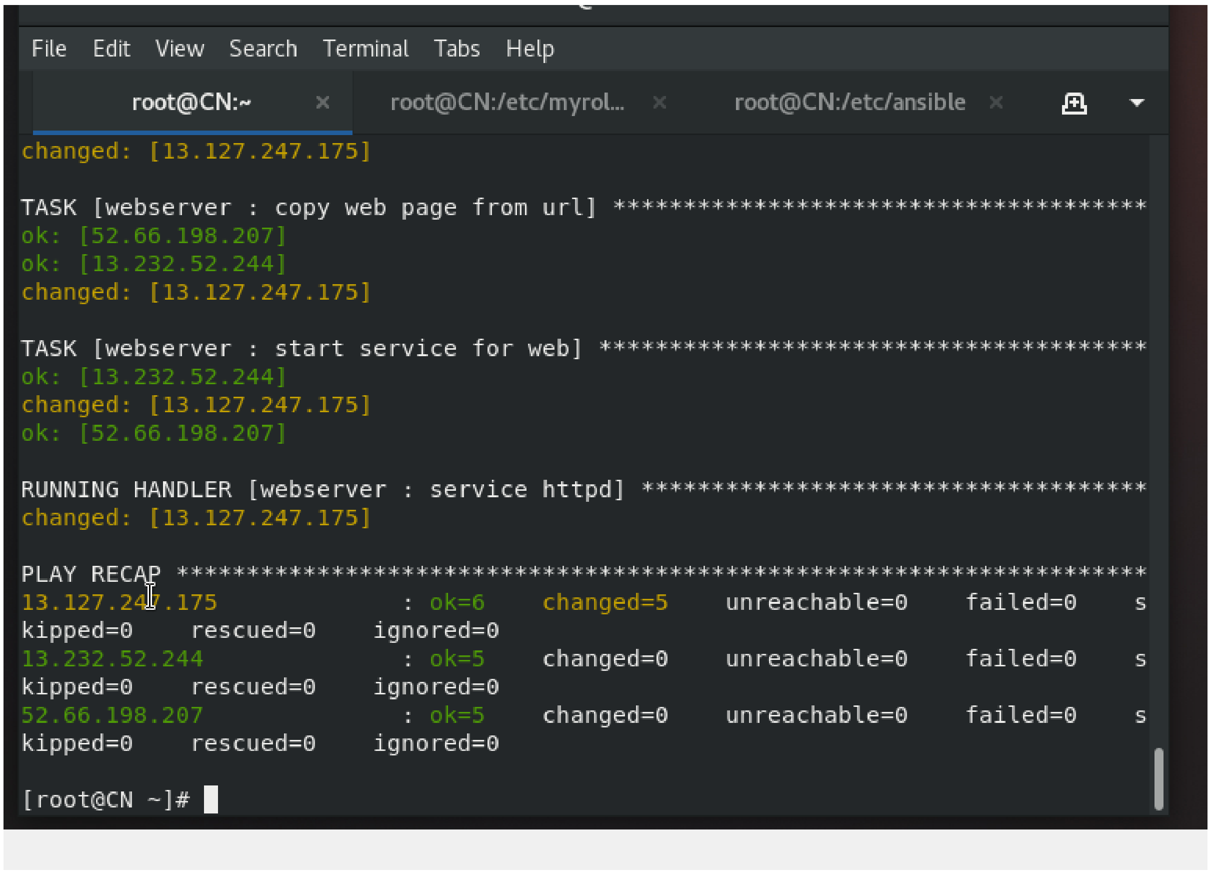


```
[root@CN ~]# ansible-playbook webs.yml
[WARNING]: Invalid characters were found in group names but not replaced, use
-vvvv to see details

PLAY [all] *******************************************************************

TASK [Gathering Facts] *******************************************************
[WARNING]: Platform linux on host 52.66.198.207 is using the discovered Python
interpreter at /usr/bin/python, but future installation of another Python
interpreter could change this. See https://docs.ansible.com/ansible/2.9/referen
ce_appendices/interpreter_discovery.html for more information.
ok: [52.66.198.207]
[WARNING]: Platform linux on host 13.232.52.244 is using the discovered Python
interpreter at /usr/bin/python, but future installation of another Python
interpreter could change this. See https://docs.ansible.com/ansible/2.9/referen
ce_appendices/interpreter_discovery.html for more information.
ok: [13.232.52.244]
[WARNING]: Platform linux on host 13.127.247.175 is using the discovered Python
interpreter at /usr/bin/python, but future installation of another Python
interpreter could change this. See https://docs.ansible.com/ansible/2.9/referen
ce_appendices/interpreter_discovery.html for more information.
ok: [13.127.247.175]
```



```
TASK [webserver : install httpd package] ***************************************
ok: [52.66.198.207]
ok: [13.232.52.244]
changed: [13.127.247.175]

TASK [webserver : conf web server] *********************************************
ok: [52.66.198.207]
ok: [13.232.52.244]
changed: [13.127.247.175]

TASK [webserver : copy web page from url] **************************************
ok: [52.66.198.207]
ok: [13.232.52.244]
changed: [13.127.247.175]

TASK [webserver : start service for web] ***************************************
ok: [13.232.52.244]
changed: [13.127.247.175]
ok: [52.66.198.207]

RUNNING HANDLER [webserver : service httpd] ************************************
changed: [13.127.247.175]
```

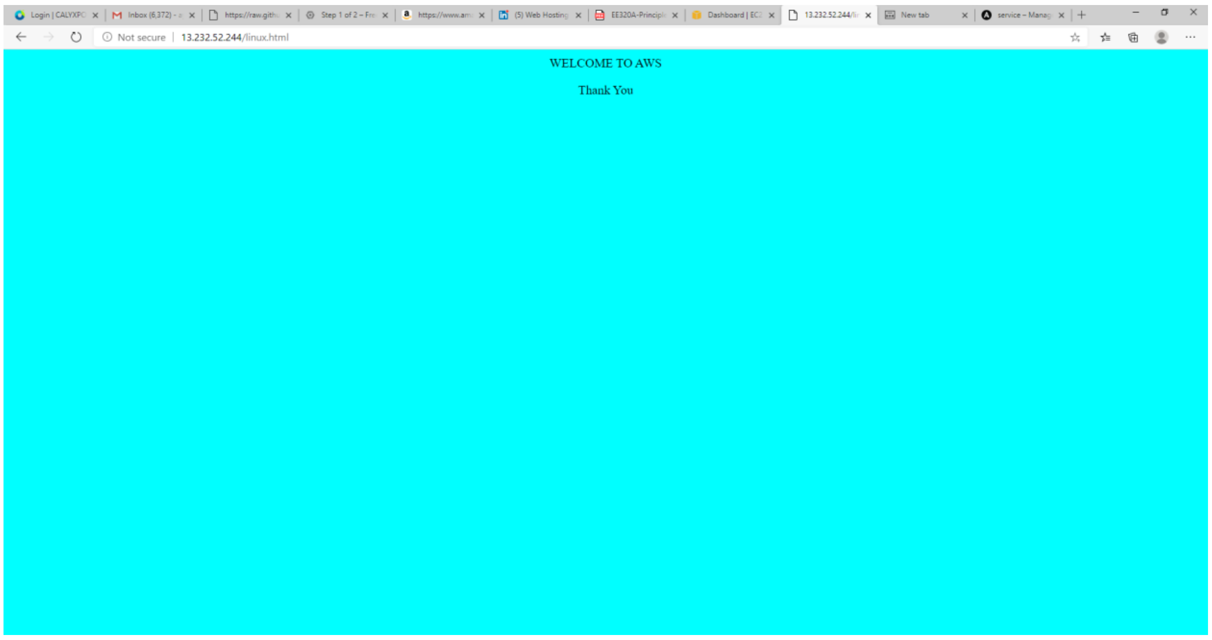
Everything being configured automatically.



```
changed: [13.127.247.175]

TASK [webserver : copy web page from url] **************************************
ok: [52.66.198.207]
ok: [13.232.52.244]
changed: [13.127.247.175]

TASK [webserver : start service for web] ***************************************
ok: [13.232.52.244]
changed: [13.127.247.175]
ok: [52.66.198.207]

RUNNING HANDLER [webserver : service httpd] ************************************
changed: [13.127.247.175]

PLAY RECAP *********************************************************************
13.127.247.175             : ok=6    changed=5    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
13.232.52.244              : ok=5    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
52.66.198.207              : ok=5    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0

[root@CN ~]#
```

All three instances running with same page

IP http://13.232.52.244/linux.html

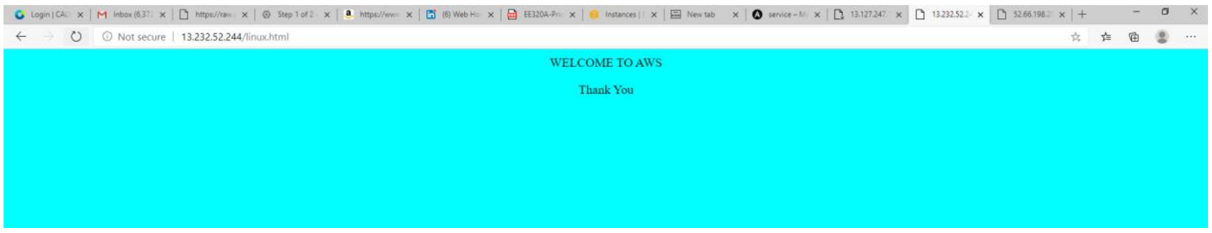


Not secure | 13.232.52.244/linux.html

WELCOME TO AWS

Thank You

http://52.66.198.207/linux.html

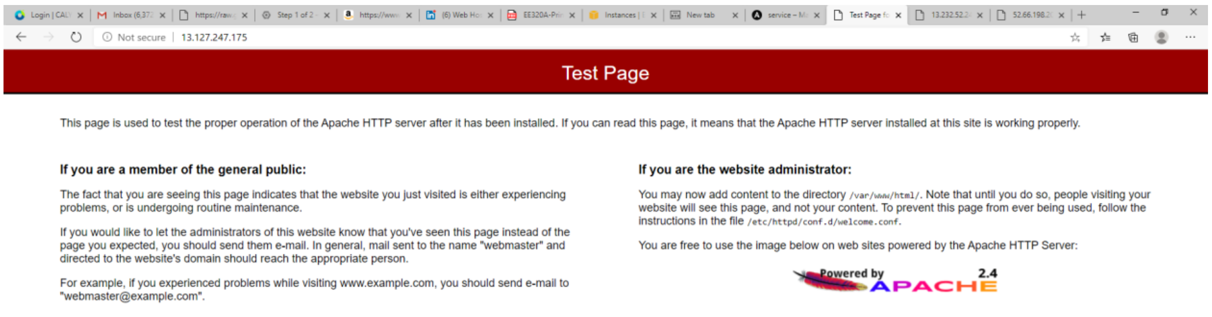


Not secure | 13.232.52.244/linux.html

WELCOME TO AWS

Thank You

http://13.127.247.175/



Not secure | 13.127.247.175

Test Page

This page is used to test the proper operation of the Apache HTTP server after it has been installed. If you can read this page, it means that the Apache HTTP server installed at this site is working properly.

If you are a member of the general public:

The fact that you are seeing this page indicates that the website you just visited is either experiencing problems, or is undergoing routine maintenance.

If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.

For example, if you experienced problems while visiting www.example.com, you should send e-mail to "webmaster@example.com".

If you are the website administrator:

You may now add content to the directory `/var/www/html/`. Note that until you do so, people visiting your website will see this page, and not your content. To prevent this page from ever being used, follow the instructions in the file `/etc/httpd/conf.d/welcome.conf`.

You are free to use the image below on web sites powered by the Apache HTTP Server:

THANK YOU!!