Name: Chaudhari Ganesh Dadabhau

Deploy a Load Balancer and multiple Web servers on AWS instances using Ansible.

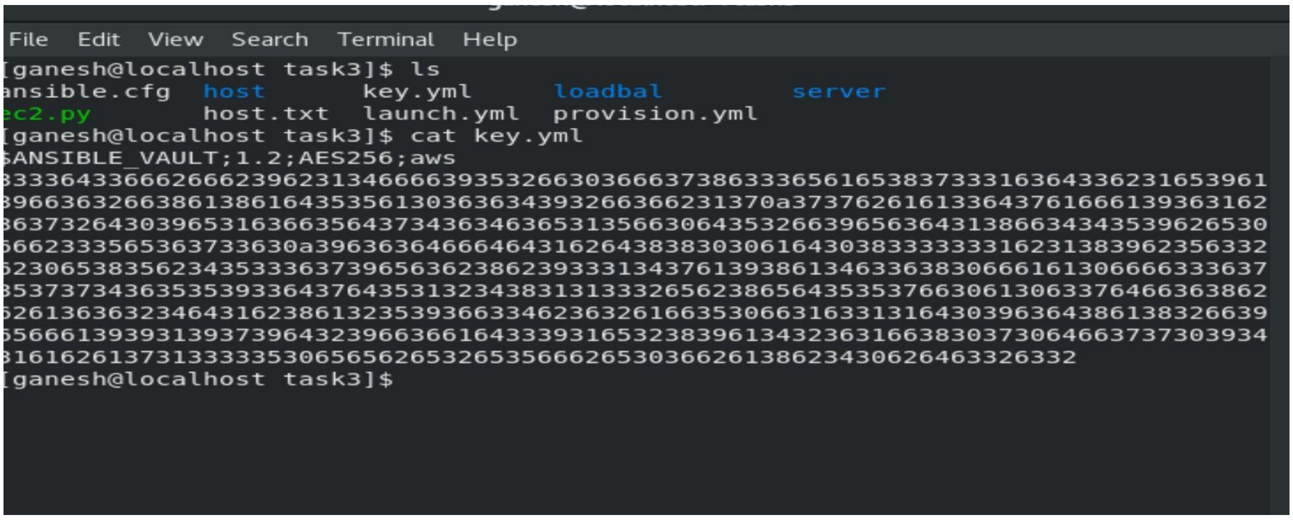
Ansible Task-3:

1. **Provision of EC2 instances.**
2. **Retrieve the IP Address of instances Dynamically.**
3. **Configure web servers**
4. **Configure Load Balancer with web servers IP addresses.**

In this task we are gonna configure Load Balancer HAPROXY on AWS using Ansible.

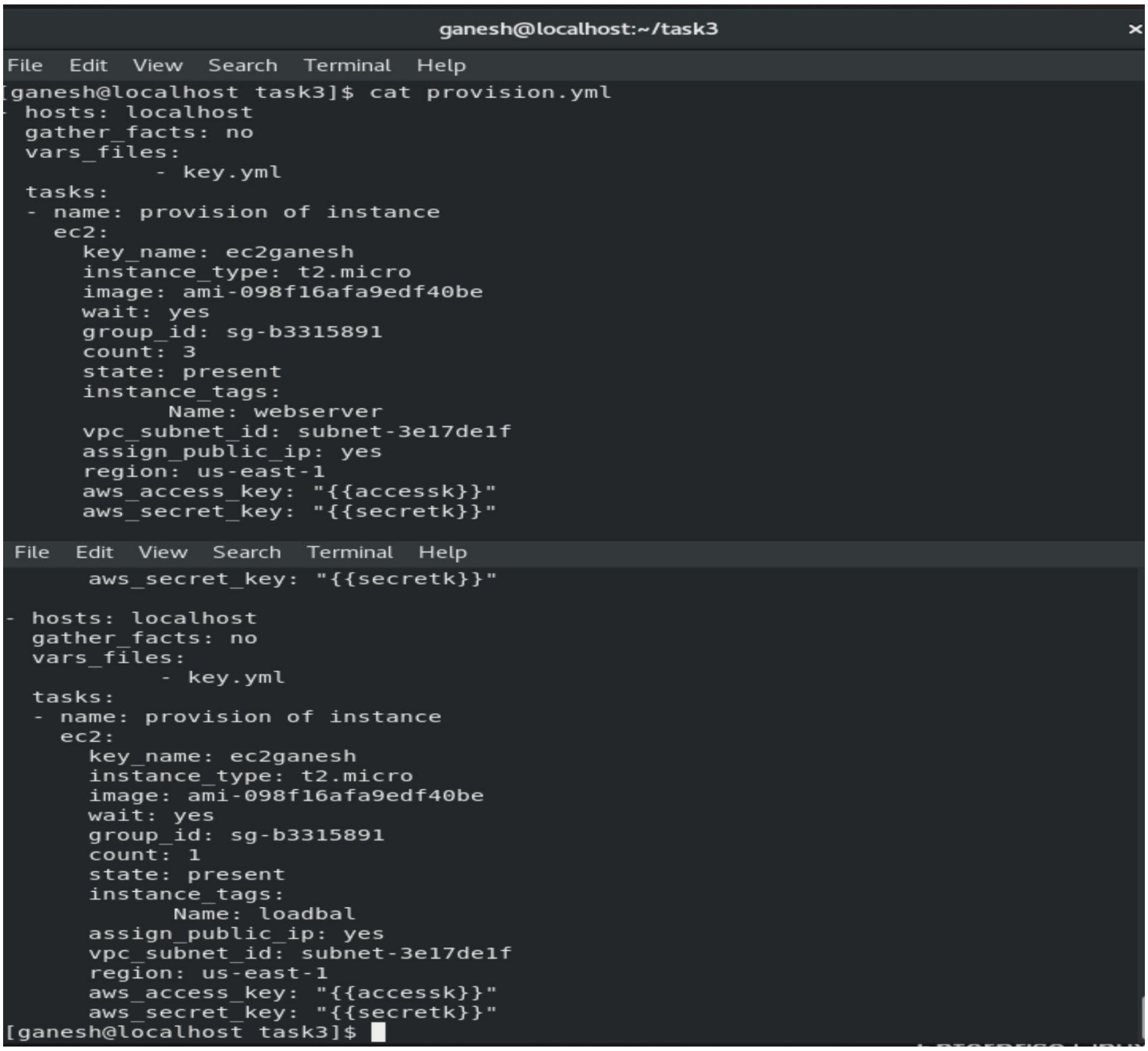
1.Provision of EC2 instances:

In this we will launch ec2 instances i.e. three for server configuration and one for load balancer configuration. lets write playbook for provision of EC2 instances before that we should write access key ID and secret access key of AWS account in one file. we should encrypt that file using **ansible-vault encrypt --vault-id aws@prompt filename.** after that



```
[ganesh@localhost task3]$ ls
ansible.cfg  host       key.yml     loadbal         server
ec2.py       host.txt   launch.yml  provision.yml
[ganesh@localhost task3]$ cat key.yml
$ANSIBLE_VAULT;1.2;AES256;aws
33336433666266623962313466663935326630366637386333656165383733316364336231653961
39663632663861386164353561303636343932663662313 70a37376261613364376166613936316 2
36373264303965316366356437343634636531356630643532663965636431386634343539626530
6662333565363733630a396363646664643162643838303061643038333333316231383962356332
62306538356234353336373965636238623933313437613938613463363830666161306666333637
35373734363535393364376435313234383131333265623865643535376630613063376466363862
62613636323464316238613235393663346236326166353066316331316430396364386138326639
65666139393139373964323966366164333931653238396134323631663830373064663737303934
31616261373133333530656562653265356662653036626138623430626463326332
[ganesh@localhost task3]$
```

write playbook like

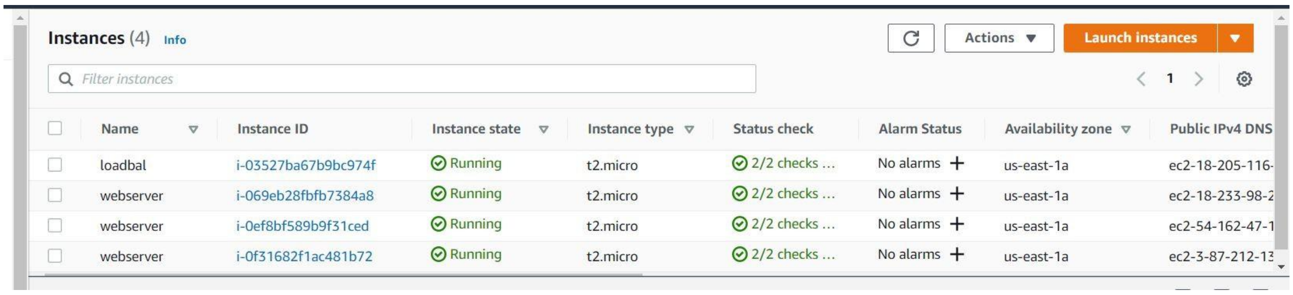


ganesh@localhost:~/task3

```
[ganesh@localhost task3]$ cat provision.yml
- hosts: localhost
  gather_facts: no
  vars_files:
           - key.yml
  tasks:
  - name: provision of instance
    ec2:
      key_name: ec2ganesh
      instance_type: t2.micro
      image: ami-098f16afa9edf40be
      wait: yes
      group_id: sg-b3315891
      count: 3
      state: present
      instance_tags:
            Name: webserver
      vpc_subnet_id: subnet-3e17de1f
      assign_public_ip: yes
      region: us-east-1
      aws_access_key: "{{accessk}}"
      aws_secret_key: "{{secretk}}"
```

```
      aws_secret_key: "{{secretk}}"

- hosts: localhost
  gather_facts: no
  vars_files:
           - key.yml
  tasks:
  - name: provision of instance
    ec2:
      key_name: ec2ganesh
      instance_type: t2.micro
      image: ami-098f16afa9edf40be
      wait: yes
      group_id: sg-b3315891
      count: 1
      state: present
      instance_tags:
            Name: loadbal
      assign_public_ip: yes
      vpc_subnet_id: subnet-3e17de1f
      region: us-east-1
      aws_access_key: "{{accessk}}"
      aws_secret_key: "{{secretk}}"
[ganesh@localhost task3]$
```

Run this playbook **ansible-playbook --vault-id aws@prompt provision.yml**



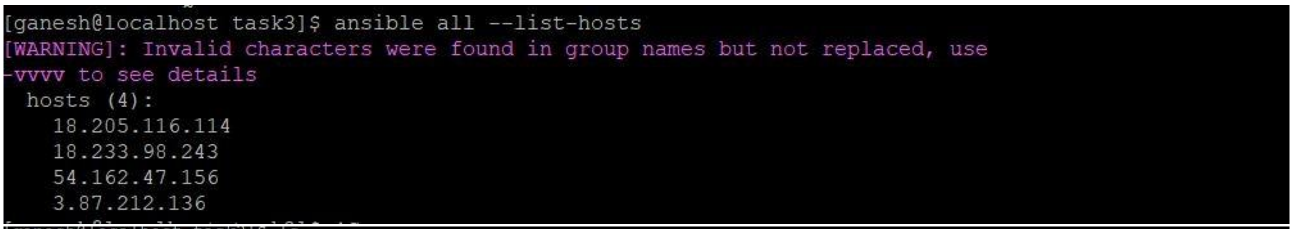
Instances (4)

	Name	Instance ID	Instance state	Instance type	Status check	Alarm Status	Availability zone	Public IPv4 DNS
☐	loadbal	i-03527ba67b9bc974f	Running	t2.micro	2/2 checks ...	No alarms +	us-east-1a	ec2-18-205-116-
☐	webserver	i-069eb28fbfb7384a8	Running	t2.micro	2/2 checks ...	No alarms +	us-east-1a	ec2-18-233-98-2
☐	webserver	i-0ef8bf589b9f31ced	Running	t2.micro	2/2 checks ...	No alarms +	us-east-1a	ec2-54-162-47-1
☐	webserver	i-0f31682f1ac481b72	Running	t2.micro	2/2 checks ...	No alarms +	us-east-1a	ec2-3-87-212-13

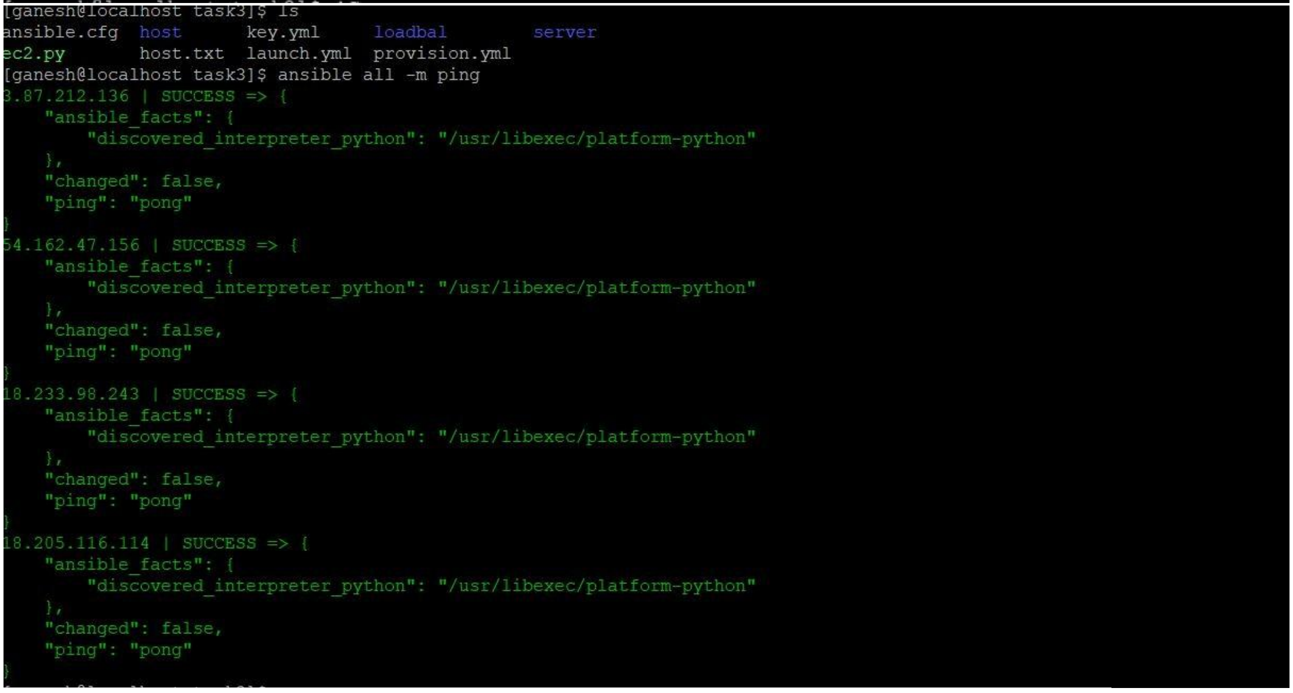
2.Retrieve the IP Address of instances Dynamically.

To retrieve IP of running instances of AWS using ec2.py. make ec2.py executable. export   export AWS_ACCESS_KEY_ID=' ' export

AWS_SECRET_ACCESS_KEY=' ' after that run **ansible all --list-hosts or ansible all - m ping**

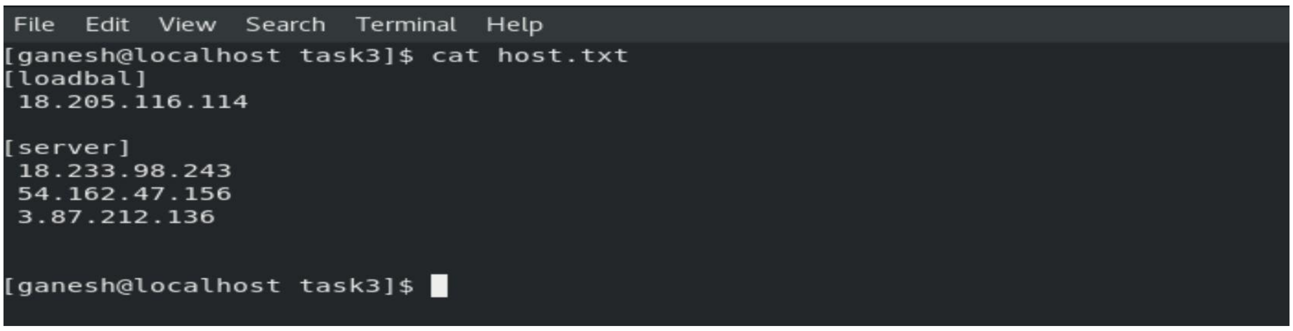


```
[ganesh@localhost task3]$ ansible all --list-hosts
[WARNING]: Invalid characters were found in group names but not replaced, use
-vvvv to see details
  hosts (4):
    18.205.116.114
    18.233.98.243
    54.162.47.156
    3.87.212.136
```



```
[ganesh@localhost task3]$ ls
ansible.cfg  host      key.yml     loadbal        server
ec2.py       host.txt  launch.yml  provision.yml
[ganesh@localhost task3]$ ansible all -m ping
3.87.212.136 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
54.162.47.156 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
18.233.98.243 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
18.205.116.114 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/libexec/platform-python"
    },
    "changed": false,
    "ping": "pong"
}
```

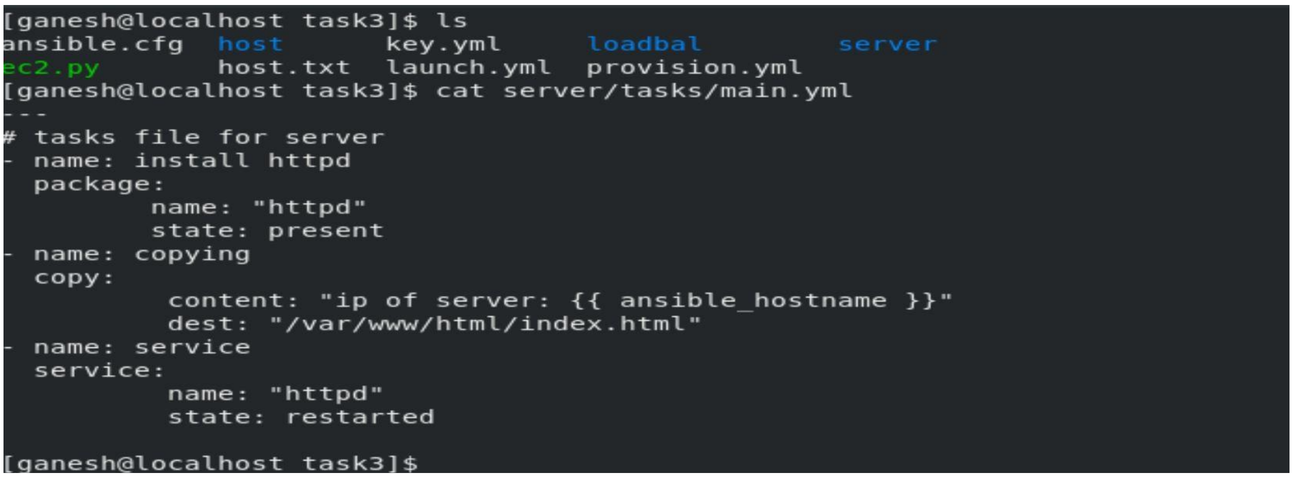
after that update inventory file like below:



```
[ganesh@localhost task3]$ cat host.txt
[loadbal]
 18.205.116.114

[server]
 18.233.98.243
 54.162.47.156
 3.87.212.136


[ganesh@localhost task3]$
```

3.Configure web servers

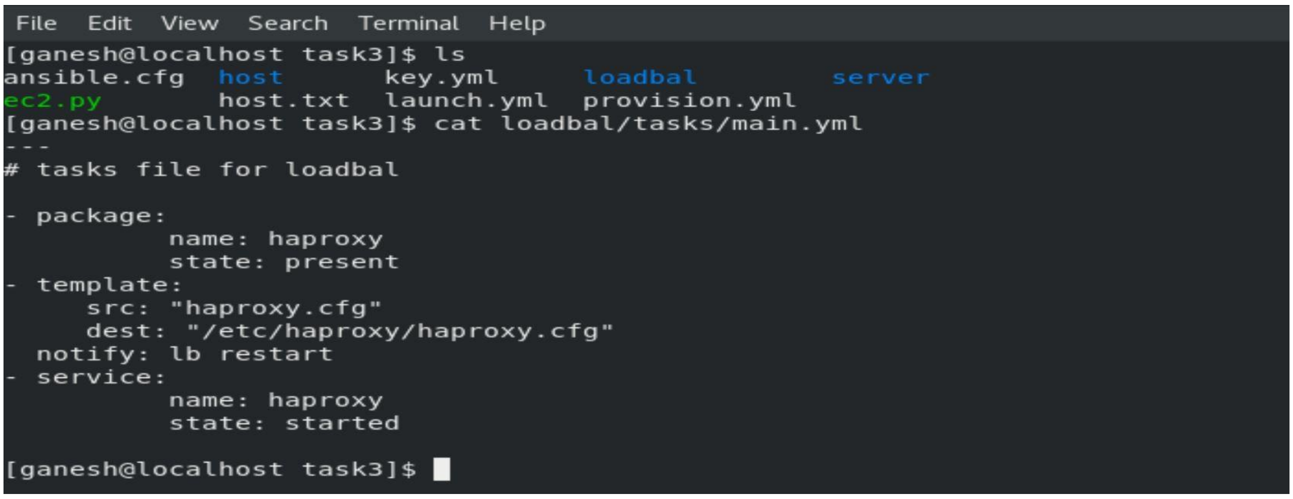
To configure web servers first we need to write role using **ansible-galaxy init server** then write a task like



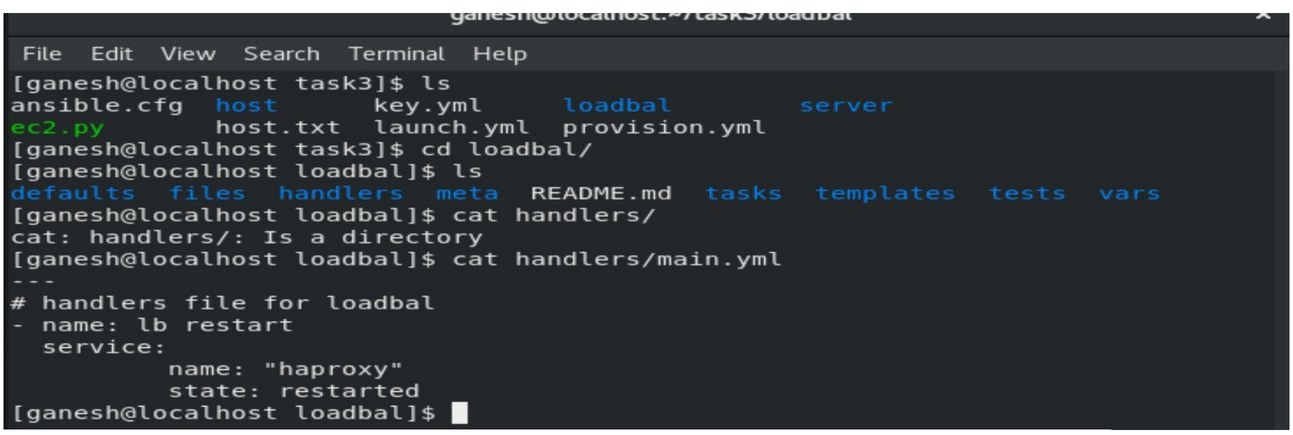
```
[ganesh@localhost task3]$ ls
ansible.cfg  host       key.yml      loadbal         server
ec2.py       host.txt   launch.yml   provision.yml
[ganesh@localhost task3]$ cat server/tasks/main.yml
---
# tasks file for server
- name: install httpd
  package:
         name: "httpd"
         state: present
- name: copying
  copy:
          content: "ip of server: {{ ansible_hostname }}"
          dest: "/var/www/html/index.html"
- name: service
  service:
          name: "httpd"
          state: restarted

[ganesh@localhost task3]$
```

4.Configure Load Balancer with web servers IP addresses.

To configure HAPROXY on AWS write one more role **ansible-galaxy init loadbal** after that write playbook like

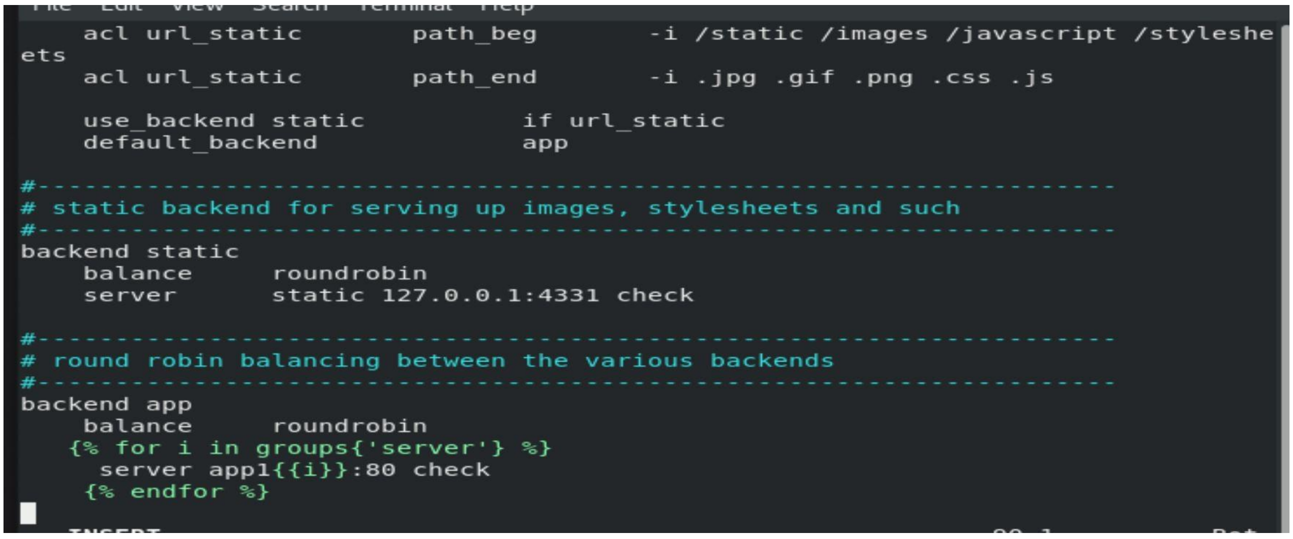


```
[ganesh@localhost task3]$ ls
ansible.cfg  host       key.yml      loadbal         server
ec2.py       host.txt   launch.yml   provision.yml
[ganesh@localhost task3]$ cat loadbal/tasks/main.yml
---
# tasks file for loadbal

- package:
          name: haproxy
          state: present
- template:
      src: "haproxy.cfg"
      dest: "/etc/haproxy/haproxy.cfg"
  notify: lb restart
- service:
          name: haproxy
          state: started

[ganesh@localhost task3]$
```


ganesh@localhost:~/task3/loadbal
File Edit View Search Terminal Help

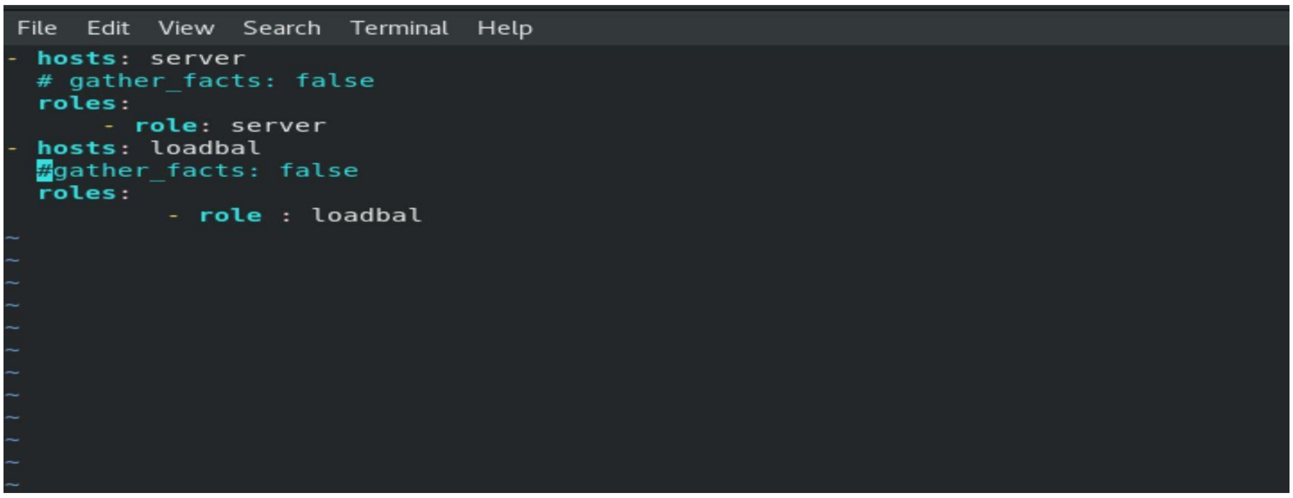
```
[ganesh@localhost task3]$ ls
ansible.cfg  host       key.yml      loadbal         server
ec2.py       host.txt   launch.yml   provision.yml
[ganesh@localhost task3]$ cd loadbal/
[ganesh@localhost loadbal]$ ls
defaults  files  handlers  meta  README.md  tasks  templates  tests  vars
[ganesh@localhost loadbal]$ cat handlers/
cat: handlers/: Is a directory
[ganesh@localhost loadbal]$ cat handlers/main.yml
---
# handlers file for loadbal
- name: lb restart
  service:
          name: "haproxy"
          state: restarted
[ganesh@localhost loadbal]$
```

Here we need to update IP of web servers in haproxy.cfg To do that we need to do write code to update IP of server and port number. like



```
    acl url_static       path_beg       -i /static /images /javascript /styleshe
ets
    acl url_static       path_end       -i .jpg .gif .png .css .js

    use_backend static          if url_static
    default_backend             app

#---------------------------------------------------------------------
# static backend for serving up images, stylesheets and such
#---------------------------------------------------------------------
backend static
    balance     roundrobin
    server      static 127.0.0.1:4331 check

#---------------------------------------------------------------------
# round robin balancing between the various backends
#---------------------------------------------------------------------
backend app
    balance     roundrobin
   {% for i in groups{'server'} %}
     server app1{{i}}:80 check
    {% endfor %}
```

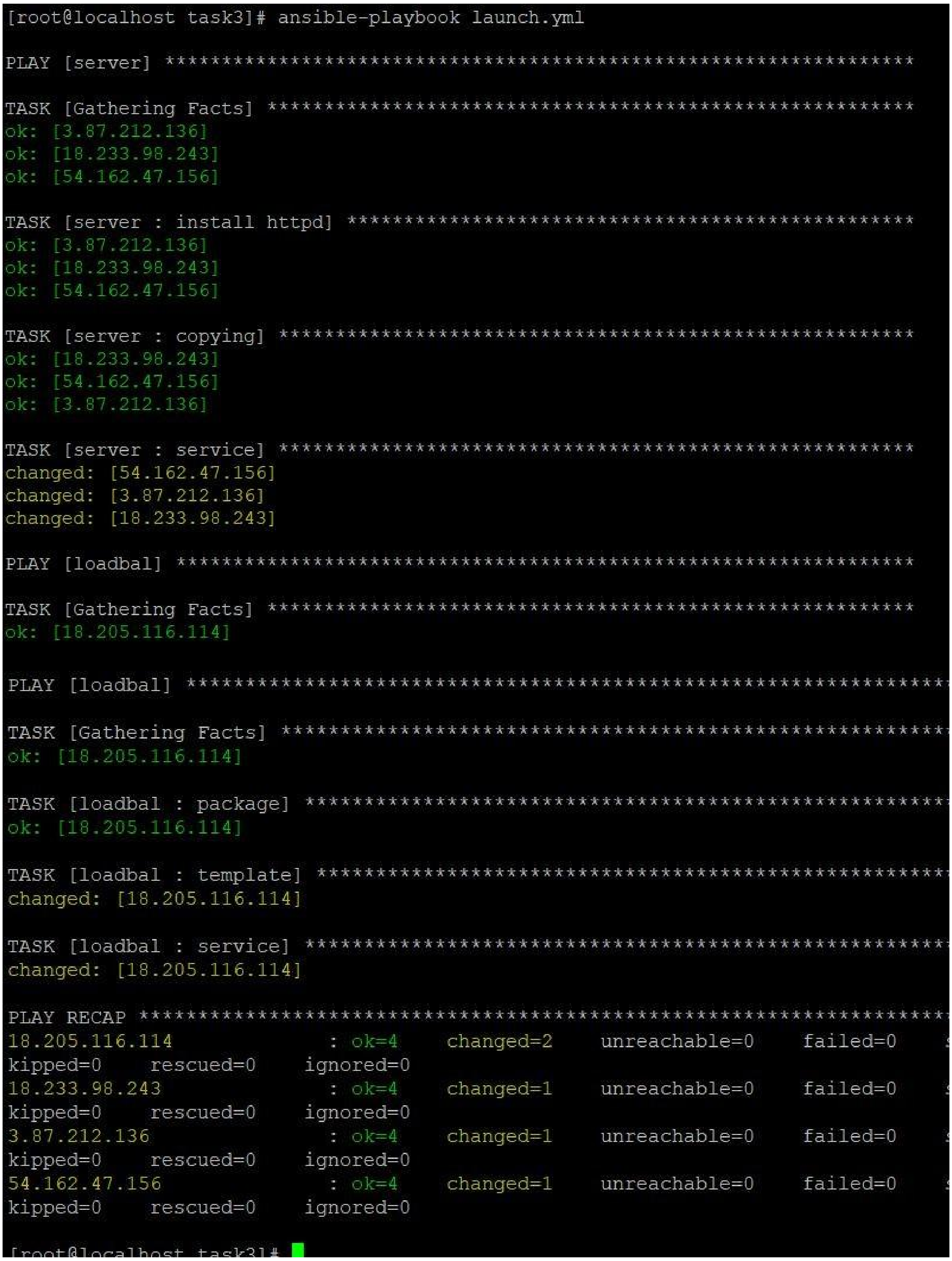
After that completing above steps write launch.yml file to run roles like



```
- hosts: server
  # gather_facts: false
  roles:
      - role: server
- hosts: loadbal
  #gather_facts: false
  roles:
          - role : loadbal
~
~
~
```

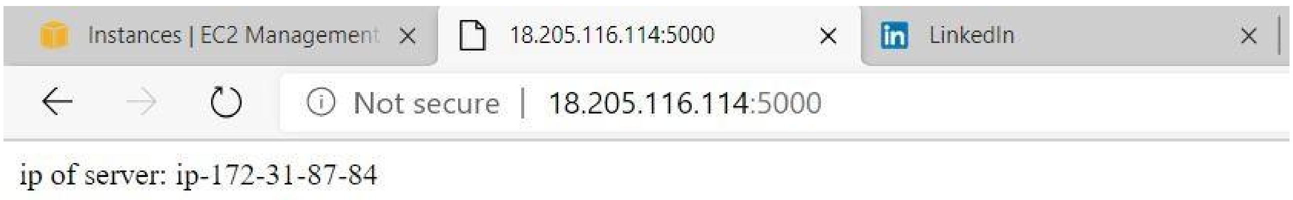
Run this command

# ansible-playbook launch.yml

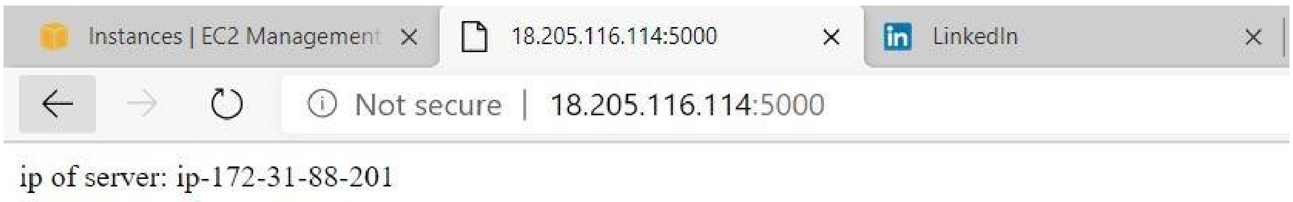


```
[root@localhost task3]# ansible-playbook launch.yml

PLAY [server] *********************************************************************

TASK [Gathering Facts] ************************************************************
ok: [3.87.212.136]
ok: [18.233.98.243]
ok: [54.162.47.156]

TASK [server : install httpd] *****************************************************
ok: [3.87.212.136]
ok: [18.233.98.243]
ok: [54.162.47.156]

TASK [server : copying] ***********************************************************
ok: [18.233.98.243]
ok: [54.162.47.156]
ok: [3.87.212.136]

TASK [server : service] ***********************************************************
changed: [54.162.47.156]
changed: [3.87.212.136]
changed: [18.233.98.243]

PLAY [loadbal] ********************************************************************

TASK [Gathering Facts] ************************************************************
ok: [18.205.116.114]


 PLAY [loadbal] *********************************************************************

 TASK [Gathering Facts] *************************************************************
 ok: [18.205.116.114]

 TASK [loadbal : package] ***********************************************************
 ok: [18.205.116.114]

 TASK [loadbal : template] **********************************************************
 changed: [18.205.116.114]

 TASK [loadbal : service] ***********************************************************
 changed: [18.205.116.114]

 PLAY RECAP *************************************************************************
 18.205.116.114             : ok=4    changed=2    unreachable=0    failed=0    s
 kipped=0    rescued=0    ignored=0
 18.233.98.243              : ok=4    changed=1    unreachable=0    failed=0    s
 kipped=0    rescued=0    ignored=0
 3.87.212.136               : ok=4    changed=1    unreachable=0    failed=0    s
 kipped=0    rescued=0    ignored=0
 54.162.47.156              : ok=4    changed=1    unreachable=0    failed=0    s
 kipped=0    rescued=0    ignored=0

 [root@localhost task3]#
```

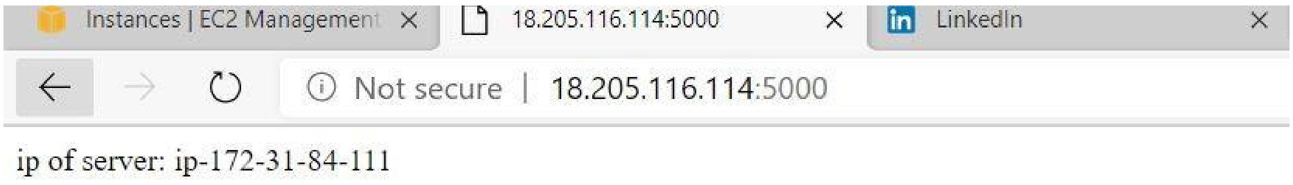
After that write load balancer socket address means(IP+ port) then result will be



ip of server: ip-172-31-87-84



ip of server: ip-172-31-88-201



ip of server: ip-172-31-84-111

Task 3 successfully completed