

## **TASK 4:**

### **Task Description**

- ◆ Launch an AWS instances with the help of ansible
- ◆ Retrieve the public IP which is allocated to the launched instance.
- ◆ With the help of the retrieved Public IP configure the load balancing setup using haproxy

## **CREATING A LOADBALANCER USING ANSIBLE**

### **Step 1: Prerequisites**

- To install boto3 module
- Setting up the dynamic inventory using python code
- chmod +x filename.yml
- Creating multiple roles loadbalancer and session14

```
[root@controllernode mypy]# pip3 list | grep boto
DEPRECATION: The default format will switch to columns in the future. You can use --format=(legacy|columns) (or define a format=(legacy|columns) in your pip.conf under the [list] section) to disable this warning.
boto (2.49.0)
boto3 (1.14.39)
botocore (1.17.39)
```

## Step 2: Launching an AWS Instance using ansible

```
[ansible@controllernode ~]$ ansible -m ping all
[WARNING]: Platform linux on host ansible-os is using the discovered Python interpreter at /usr/bin/python, but
installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more information.
ansible-os | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}

[WARNING]: Platform linux on host ansible-os2 is using the discovered Python interpreter at /usr/bin/python, but
installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more information.
ansible-os2 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}

[WARNING]: Platform linux on host ansible-os1 is using the discovered Python interpreter at /usr/bin/python, but
installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more information.
ansible-os1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}

[root@controllernode aws_ansible]# cat secure.yml
$ANSIBLE_VAULT;1.1;AES256
63376365393439623765656435396237386635346532383336383864323634353635303538346637
3962376333626462353934373039653734386364653063610a303066326535333837653834316164
6336353735303664623231626435316233365373664343333323364333762616664636639383763
3039373536306635630a336264643839376435663834623635396332616337393735616364646139
61396561393630326566333534323761643264393336303066373862626634326534626230663833
636363653866616262356164613923336663365643835356231336464316465613531333343332
3031636535653932626203736333635343539326437303361666336636564363138363063333662
31376630383135646533303339393861356364656434633864616636323136383832393634633263
3363
```

```
[root@controllernode aws_ansible]# cat ec2.yml
- hosts: localhost
  vars_files:
    - secure.yml
  tasks:
#   - prog -> aws_client -> ec2_user
  - name: launching os using ansible
    ec2:
      region: "ap-south-1"
      key_name: "mykey1111.pem"
      instance_type: "t2.micro"
      image: "ami-0ebc1ac48dfd14136"
      wait: yes
      count: 4
      vpc_subnet_id: "subnet-d7ead0bf"
      assign_public_ip: yes
      group_id: "sg-022d22ccade57d756"
      state: present
      aws_access_key: "{{ access_key }}"
      aws_secret_key: "{{ secret_key }}"
```

- Creating an task4.yml file
- Storing access\_key and secret\_key in a separate vault file
- Using ansible-playbook --ask-vault-pass task4.yml

```
[root@controllernode aws_ansible]# ansible-vault encrypt secure.yml
New Vault password:
Confirm New Vault password:
Encryption successful
```

## Step 3: Setting up the Dynamic inventory environment

```
root@controllernode:mypy
ntrib/inventory/ec2.py
--2020-08-20 22:15:29--  https://raw.githubusercontent.com/ansible/ansible/stable-2.9/contrib/inventory/ec2.py
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 199.232.20.133
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|199.232.20.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 73130 (71k) [text/plain]
Saving to: 'ec2.py'

ec2.py          100%[=====] 71.42K  422KB/s   in 0.2s

2020-08-20 22:15:30 (422 KB/s) - 'ec2.py' saved [73130/73130]

[root@controllernode mypy]# wget https://raw.githubusercontent.com/ansible/ansible/stable-2.9/co
ntrib/inventory/ec2.ini
--2020-08-20 22:35:52--  https://raw.githubusercontent.com/ansible/ansible/stable-2.9/contrib/inventory/ec2.ini
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 199.232.20.133
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|199.232.20.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 9529 (9.3K) [text/plain]
Saving to: 'ec2.ini'

ec2.ini          100%[=====] 9.31K  --.-KB/s   in 0.001s

2020-08-20 22:35:52 (6.31 MB/s) - 'ec2.ini' saved [9529/9529]

[root@controllernode mypy]# ls
ec2.ini  ec2.py
[root@controllernode mypy]# chmod +x ec2.ini
[root@controllernode mypy]# ls
ec2.ini  ec2.py
```

```
[root@controllernode mypy]# export AWS_SECRET_ACCESS_KEY=ch3pp3rKw1rap04tC0Tf3Jw5p0q1qEoV1v
[root@controllernode mypy]# export AWS_REGION='ap-south-1'
[root@controllernode mypy]# python3 ec2.py --list
```

```
[root@controllernode aws_ansible]# ansible all --list-hosts
[WARNING]: Invalid characters were found in group names but not replaced, use -vvvv to see details
hosts (4):
  13.234.240.55
  13.127.142.159
  13.234.78.166
  15.206.94.86
```

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with options like New EC2 Experience, EC2 Dashboard, Events, Tags, Limits, Instances (selected), Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, and Dedicated Hosts. The main area displays a table of instances:

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks
	i-01fa8f450b4dde0ae	t2.micro	ap-south-1a	terminated	No
	i-03baa95eefc3d0837	t2.micro	ap-south-1a	running	Initializing
	i-04a26acd85b0a794f	t2.micro	ap-south-1a	running	Initializing
	i-05f83b6d944495e8c	t2.micro	ap-south-1a	running	Initializing
	i-0ad3140571de476...	t2.micro	ap-south-1a	running	Initializing

At the bottom, there's a message: "Select an instance above".

## Step 4: Connect Instance with SSH

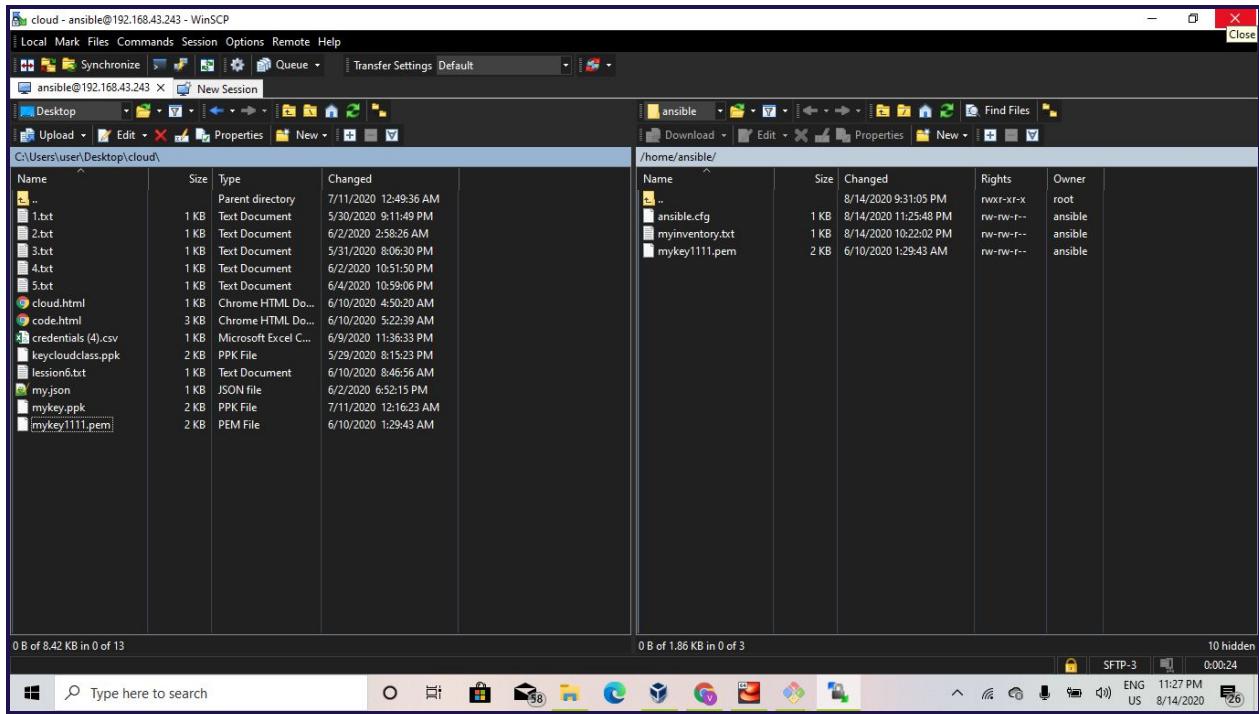
The terminal window shows the following command and its output:

```
ec2-user@ip-172-31-43-229:~$ cd cloud
C:\Users\user\Desktop>ssh -i mykey1111.pem -l ec2-user 13.234.226.61
The authenticity of host '13.234.226.61 (13.234.226.61)' can't be established.
ECDSA key fingerprint is SHA256:CkZIkSAYKdIwJ7Hk5HeImMNRR6doURu0t2RtaXVjuG4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '13.234.226.61' (ECDSA) to the list of known hosts.
Last login: Thu Aug 20 18:19:05 2020 from 223.189.163.222
Last login: Thu Aug 20 18:19:05 2020 from 223.189.163.222

[Amazon Linux 2 AMI]

https://aws.amazon.com/amazon-linux-2/
4 package(s) needed for security, out of 8 available
Run "sudo yum update" to apply all updates.
```

## Step 5: Copying key-pair using win-scp



## Step 6: Configuring host and config file

```
[ansible@controllernode ~]$ cat myinventory1.txt
[lb]
13.234.240.55 ansible_user=ec2-user ansible_ssh_private_key_file=/home/ansible/Desktop/mykey1111.pem

[webserver]
13.127.142.159 ansible_user=ec2-user ansible_ssh_private_key_file=/home/ansible/Desktop/mykey1111.pem
13.234.78.166 ansible_user=ec2-user ansible_ssh_private_key_file=/home/ansible/Desktop/mykey1111.pem
15.206.94.86 ansible_user=ec2-user ansible_ssh_private_key_file=/home/ansible/Desktop/mykey1111.pem

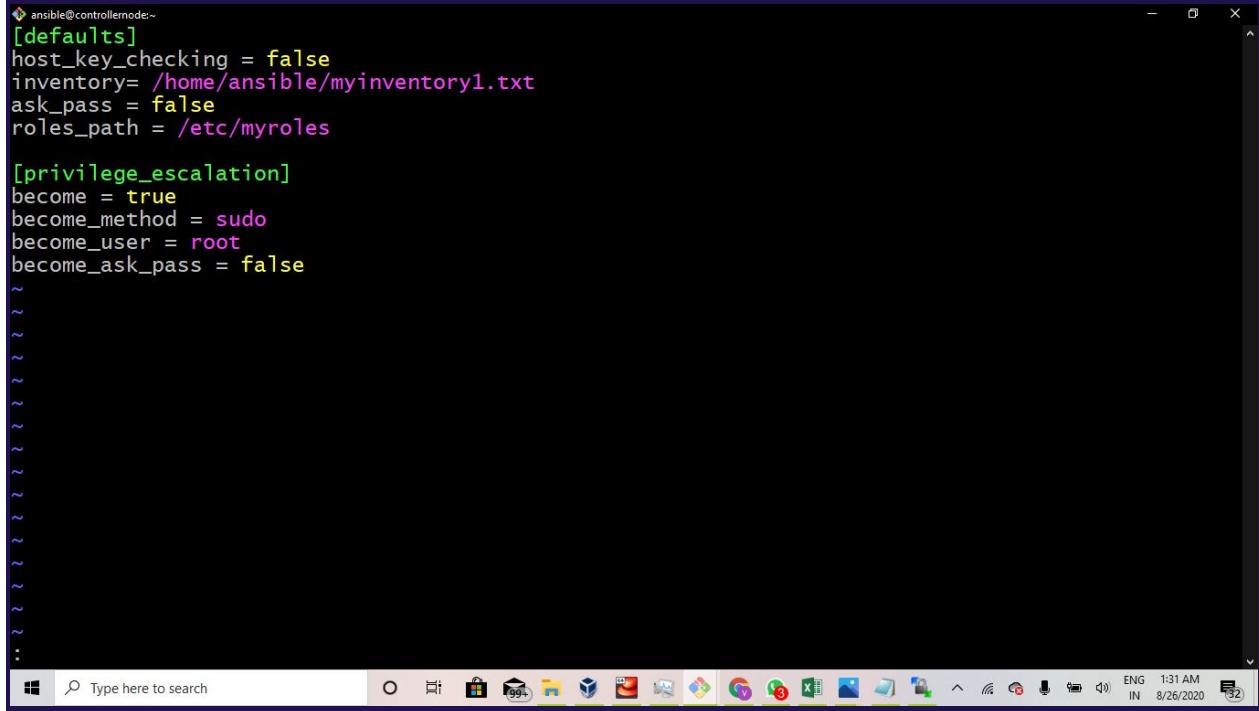
[ansible@controllernode ~]$
[ansible@controllernode ~]$ cat task4.yml
- hosts: webserver
  roles:
    - role: session14

- hosts: lb
  roles:
    - role: loadbalancersetup
```

**The information you will need is:**

- Name for the instance

- IP Address of your AWS instance(public-ip)
- The user present on your AWS instance (ec2-user)
- Location to your private key (.pem) file



A screenshot of a Windows terminal window titled "ansible@controllernode:~". The window displays Ansible configuration code in YAML format. The code includes sections for [defaults], [privilegeEscalation], and a series of approximately 25 blank lines starting with a tilde (~). The terminal window has a dark background and a light-colored text area. At the bottom, there is a taskbar with various icons and a system tray showing the date and time (8/26/2020, 1:31 AM).

```
[defaults]
host_key_checking = false
inventory= /home/ansible/myinventory1.txt
ask_pass = false
roles_path = /etc/myroles

[privilege_escalation]
become = true
become_method = sudo
become_user = root
become_ask_pass = false
~
~
```

## Step 7: Configuring the setting and making public-key to private

```

[ansible@localhost:~]
  "msg": "Failed to connect to the host via ssh: warning: Permanently added '15.20.7.19.49' (ECDSA) to the list of known hosts.\r\nnec2-user@15.207.19.49: Permission denied (publickey,gssapi-keyex,gssapi-with-mic).",
  "unreachable": true
}
[ansible@localhost ~]$ vim ansible.cfg
[ansible@localhost ~]$ vim ansible.cfg
[ansible@localhost ~]$ ansible -m ping all
ansible_ssh_private_key_file=/home/ansible/mykey1111.pem | UNREACHABLE! => {
  "changed": false,
  "msg": "Failed to connect to the host via ssh: ssh: Could not resolve hostname ansible_ssh_private_key_file=/home/ansible/mykey1111.pem: Name or service not known",
  "unreachable": true
}
ec2-instance | UNREACHABLE! => {
  "changed": false,
  "msg": "Failed to connect to the host via ssh: ec2-user@15.207.19.49: Permission denied (publickey,gssapi-keyex,gssapi-with-mic).",
  "unreachable": true
}
[ansible@localhost ~]$ ls
ansible.cfg  myinventory1.txt  myinventory.txt  mykey1111.pem
[ansible@localhost ~]$ vim myinventory1.txt
[ansible@localhost ~]$ ansible -m ping all

```

Type here to search

```

root@localhost:~
## systems).
## Syntax:
##
##       user      MACHINE=COMMANDS
##
## The COMMANDS section may have other options added to it.
##
## Allow root to run any commands anywhere
root    ALL=(ALL)          ALL

ansible ALL=(root)         NOPASSWD: ALL

## Allows members of the 'sys' group to run networking, software,
## service management apps and more.
# %sys ALL = NETWORKING, SOFTWARE, SERVICES, STORAGE, DELEGATING, PROCESSES, LOCATE, DRIVERS

## Allows people in group wheel to run all commands
%wheel  ALL=(ALL)          ALL

## Same thing without a password
# %wheel      ALL=(ALL)      NOPASSWD: ALL

## Allows members of the users group to mount and unmount the
## cdrom as root
# %users     ALL=/sbin/mount /mnt/cdrom, /sbin/umount /mnt/cdrom

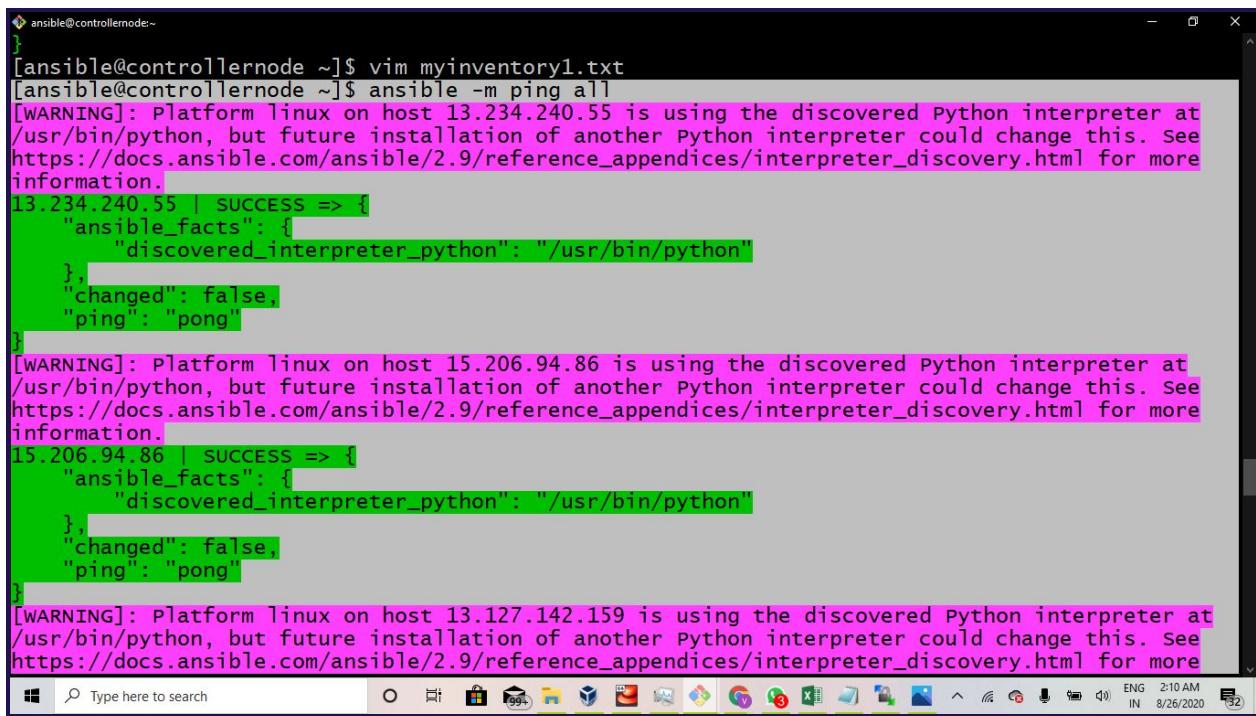
```

Type here to search

```
sudo chmod 600 /home/ansible/mykey1111.pem
```

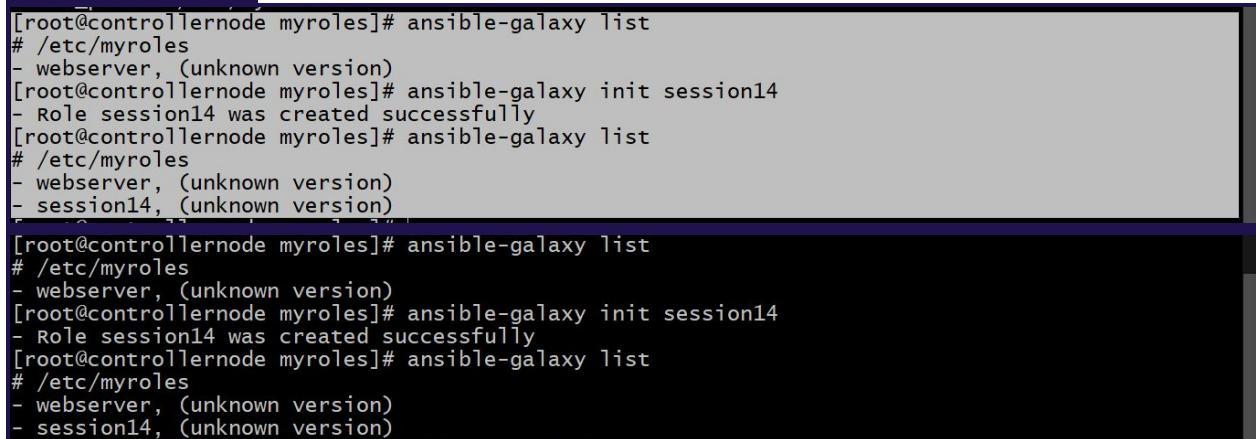
```
sudo chmod 755 ~/.ssh
```

## Step 8: Run Ansible Ping Module



```
ansible@controllernode ~]$ vim myinventory1.txt
[ansible@controllernode ~]$ ansible -m ping all
[WARNING]: Platform Linux on host 13.234.240.55 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/reference_appendices/interpreter_discovery.html for more information.
13.234.240.55 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
[WARNING]: Platform Linux on host 15.206.94.86 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/reference_appendices/interpreter_discovery.html for more information.
15.206.94.86 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
[WARNING]: Platform Linux on host 13.127.142.159 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/reference_appendices/interpreter_discovery.html for more information.
```

## Step 9: Creating roles load balancer and session14



```
[root@controllernode myroles]# ansible-galaxy list
# /etc/myroles
- webserver, (unknown version)
[root@controllernode myroles]# ansible-galaxy init session14
- Role session14 was created successfully
[root@controllernode myroles]# ansible-galaxy list
# /etc/myroles
- webserver, (unknown version)
- session14, (unknown version)

[root@controllernode myroles]# ansible-galaxy list
# /etc/myroles
- webserver, (unknown version)
[root@controllernode myroles]# ansible-galaxy init session14
- Role session14 was created successfully
[root@controllernode myroles]# ansible-galaxy list
# /etc/myroles
- webserver, (unknown version)
- session14, (unknown version)
```

## Step 10: Configuring Haproxy service inside the load balancer role

```
root@controllernode:/etc/myroles/loadbalancersetup/tasks
---
# tasks file for loadbalancersetup

- name: install haproxy software
  package:
    name: "haproxy"
    state: present

- name: copy my conf file of lb
  template:
    src: "haproxy.cfg"
    dest: /etc/haproxy/haproxy.cfg
  notify: lbrestart

- name: start haproxy service
  service:
    name: "haproxy"
    state: started

~
root@controllernode:/etc/myroles/loadbalancersetup/templates
frontend main
  bind *:8080
  acl url_static      path_beg      -i /static /images /javascript /stylesheets
  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['webserver'] %}

  server app1 {{ i }}:80 check

{% endfor %}
89,0-1 Bot
root@controllernode:/etc/myroles/loadbalancersetup/handlers
---
# handlers file for loadbalancersetup

- name: lbrestart
  service:
    name: "haproxy"
    state: restarted
```

## Step 11: Configuring httpd server inside the session14 role

```
root@controlmode:/etc/myroles/session14/tasks
---
# tasks file for session14

- name: install httpd
  package:
    name: "httpd"
    state: present

- name: copy content
  copy:
    content: "hi from {{ ansible_hostname }}"
    dest: /var/www/html/index.html

- name: start httpd service
  service:
    name: "httpd"
    state: started
~
```

## Step 12: Running ansible-playbook command

```
ansible@controllernode:~$ vim task4.yml
ansible@controllernode:~$ vim /home/ansible/ansible.cfg
ansible@controllernode:~$ ansible-playbook task4.yml

PLAY [webserver] ****
TASK [Gathering Facts] ****
[WARNING]: Platform linux on host ansible-os3 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/reference_appendices/interpreter_discovery.html for more information.
ok: [ansible-os3]
[WARNING]: Platform linux on host ansible-os1 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/reference_appendices/interpreter_discovery.html for more information.
ok: [ansible-os1]
[WARNING]: Platform linux on host ansible-os2 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/reference_appendices/interpreter_discovery.html for more information.
ok: [ansible-os2]

TASK [session14 : install httpd] ****
changed: [ansible-os3]
changed: [ansible-os2]
changed: [ansible-os1]

ok: [ansible-os2]

TASK [session14 : install httpd] ****
changed: [ansible-os3]
changed: [ansible-os2]
changed: [ansible-os1]

TASK [session14 : copy content] ****
changed: [ansible-os3]
changed: [ansible-os1]
changed: [ansible-os2]

TASK [session14 : start httpd service] ****
changed: [ansible-os3]
changed: [ansible-os1]
changed: [ansible-os2]

PLAY [lb] ****
TASK [Gathering Facts] ****
[WARNING]: Platform linux on host ansible-os 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/reference_appendices/interpreter_discovery.html for more information.
ok: [ansible-os]

TASK [loadbalancersetup : install haproxy software] ****
changed: [ansible-os]

Type here to search
```

```

ansible@controllermode:~
/usr/bin/python, but future installation of another Python interpreter could change this. See
https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more
information.
ok: [ansible-os]

TASK [loadbalancersetup : install haproxy software] ****
changed: [ansible-os]

TASK [loadbalancersetup : copy my conf file of lb] ****
changed: [ansible-os]

TASK [loadbalancersetup : start haproxy service] ****
changed: [ansible-os]

RUNNING HANDLER [loadbalancersetup : lbrestart] ****
changed: [ansible-os]

PLAY RECAP ****
ansible-os : ok=5   changed=4    unreachable=0    failed=0     skipped=0    resc
ued=0  ignored=0
ansible-os1 : ok=4   changed=3    unreachable=0    failed=0     skipped=0    resc
ued=0  ignored=0
ansible-os2 : ok=4   changed=3    unreachable=0    failed=0     skipped=0    resc
ued=0  ignored=0
ansible-os3 : ok=4   changed=3    unreachable=0    failed=0     skipped=0    resc
ued=0  ignored=0

[ansible@controllernode ~]$ ls

```

## Step 13: final Evaluation

```

Select ec2-user@ip-172-31-35-139:~
Microsoft Windows [Version 10.0.18362.959]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\user>cd Desktop

C:\Users\user\Desktop>cd cloud

C:\Users\user\Desktop\cloud>ssh -i mykey1111.pem -l ec2-user 13.234.240.55
The authenticity of host '13.234.240.55 (13.234.240.55)' can't be established.
ECDSA key fingerprint is SHA256:oe0pf4QN+WzDCycVMrRMLDH3xgzPTJcAgJhYnSbPoKk.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '13.234.240.55' (ECDSA) to the list of known hosts.
Last login: Tue Aug 25 20:16:43 2020 from ec2-13-233-177-1.ap-south-1.compute.amazonaws.com
Last login: Tue Aug 25 20:16:43 2020 from ec2-13-233-177-1.ap-south-1.compute.amazonaws.com

      _|_ _|_
      |( /   Amazon Linux 2 AMI
      __|\_\_|

https://aws.amazon.com/amazon-linux-2/
7 package(s) needed for security, out of 14 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-35-139 ~]$ rpm -q haproxy
haproxy-1.5.18-9.amzn2.x86_64
[ec2-user@ip-172-31-35-139 ~]$ 

```

```
[ec2-user@ip-172-31-35-139 ~]$ systemctl status haproxy
● haproxy.service - HAProxy Load Balancer
  Loaded: loaded (/usr/lib/systemd/system/haproxy.service; disabled; vendor preset: disabled)
  Active: active (running) since Tue 2020-08-25 20:15:55 UTC; 41min ago
    Main PID: 5737 (haproxy-systemd)
   CGroup: /system.slice/haproxy.service
           └─5737 /usr/sbin/haproxy-systemd-wrapper -f /etc/haproxy/haproxy.cfg -p /run/haproxy.pid
             ├─5738 /usr/sbin/haproxy -f /etc/haproxy/haproxy.cfg -p /run/haproxy.pid -Ds
             ├─5739 /usr/sbin/haproxy -f /etc/haproxy/haproxy.cfg -p /run/haproxy.pid -Ds

Aug 25 20:15:55 ip-172-31-35-139.ap-south-1.compute.internal systemd[1]: Started HAProxy Load Balancer.
Aug 25 20:15:55 ip-172-31-35-139.ap-south-1.compute.internal systemd[1]: Starting HAProxy Load Balan...
Aug 25 20:15:55 ip-172-31-35-139.ap-south-1.compute.internal haproxy-systemd-wrapper[5737]: haproxy-...
Aug 25 20:15:55 ip-172-31-35-139.ap-south-1.compute.internal haproxy-systemd-wrapper[5737]: [WARNING...
Aug 25 20:15:55 ip-172-31-35-139.ap-south-1.compute.internal haproxy-systemd-wrapper[5737]: [WARNING...
Hint: Some lines were ellipsized, use -l to show in full.
```

```
[ec2-user@ip-172-31-35-139 ~]$ cat /etc/haproxy/haproxy.cfg
-----
# Example configuration for a possible web application. See the
# full configuration options online.
#
#     https://www.haproxy.org/download/1.8/doc/configuration.txt
#
-----

#-----
# Global settings
#-----
global
    # to have these messages end up in /var/log/haproxy.log you will
    # need to:
    #
    # 1) configure syslog to accept network log events. This is done
    #     by adding the '-r' option to the SYSLOGD_OPTIONS in
    #     /etc/sysconfig/syslog
    #
    # 2) configure local2 events to go to the /var/log/haproxy.log
    #     file. A line like the following can be added to
    #     /etc/sysconfig/syslog
```



```
ec2-user@ip-172-31-35-139:~
```

```
frontend main
    bind *:8080
    acl url_static      path_beg     -i /static /images /javascript /stylesheets
    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

    server app1 13.127.142.159:80 check
```

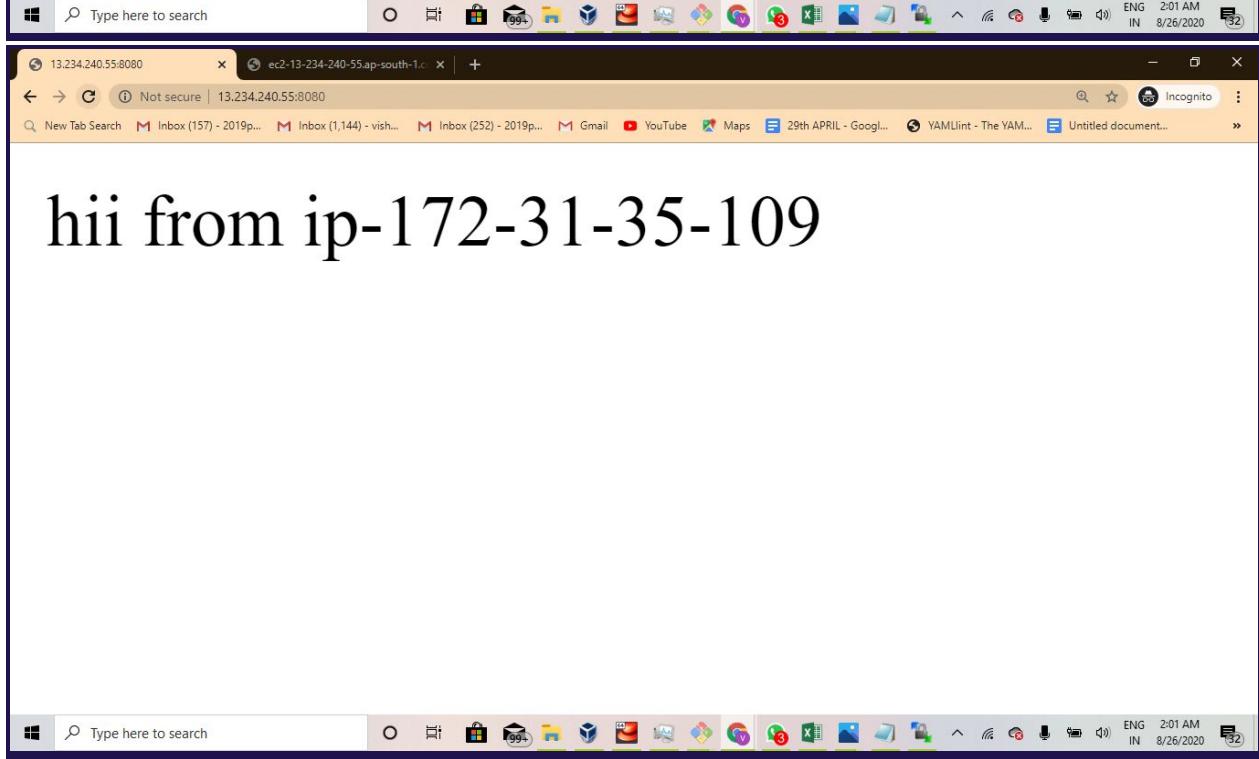
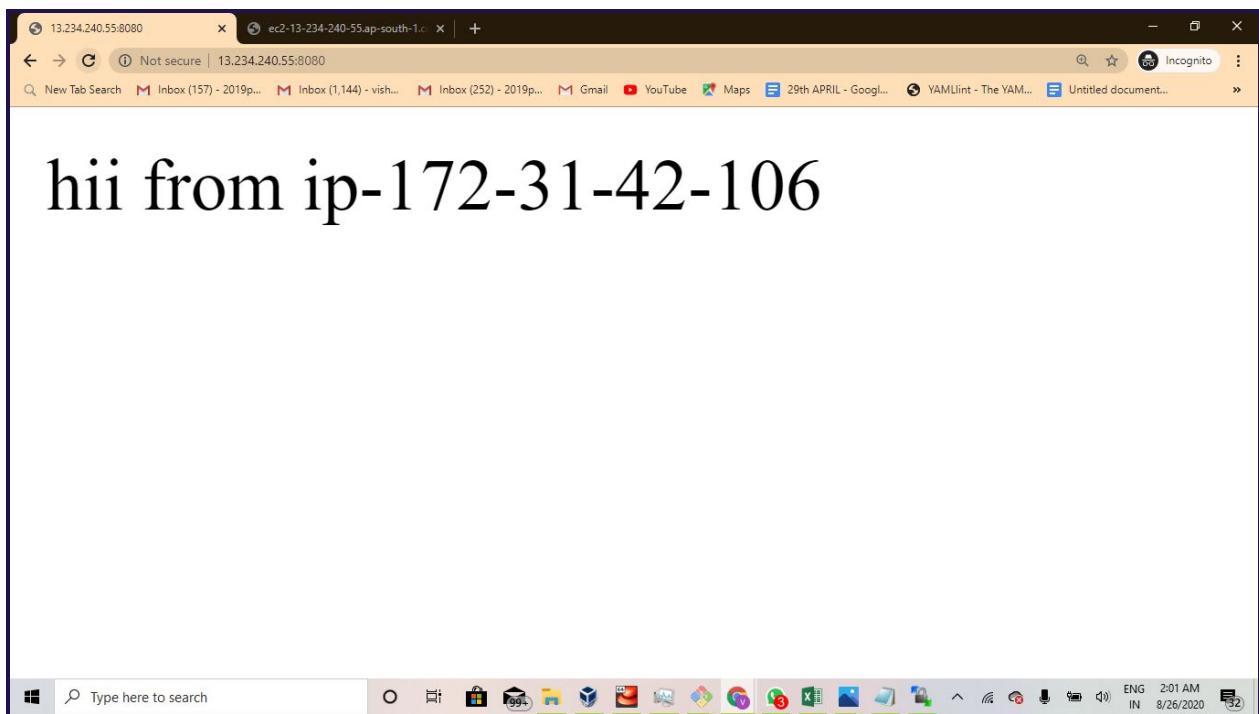


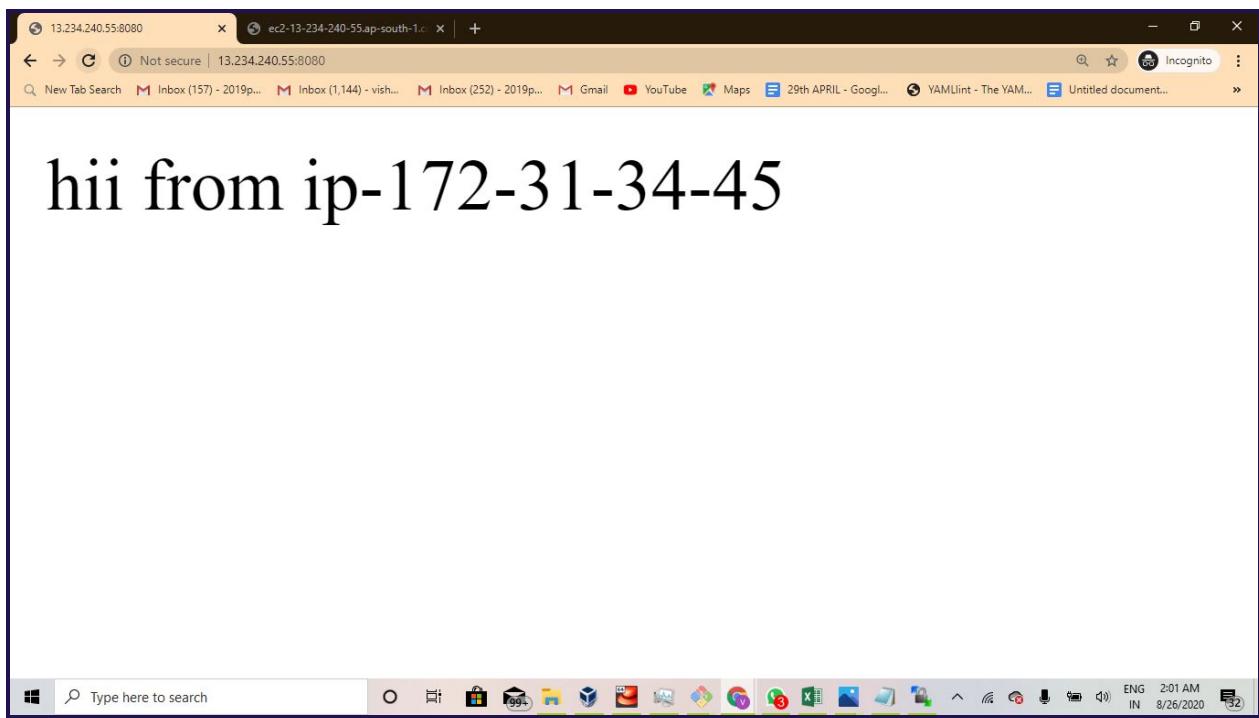
```
server app1 13.127.142.159:80 check

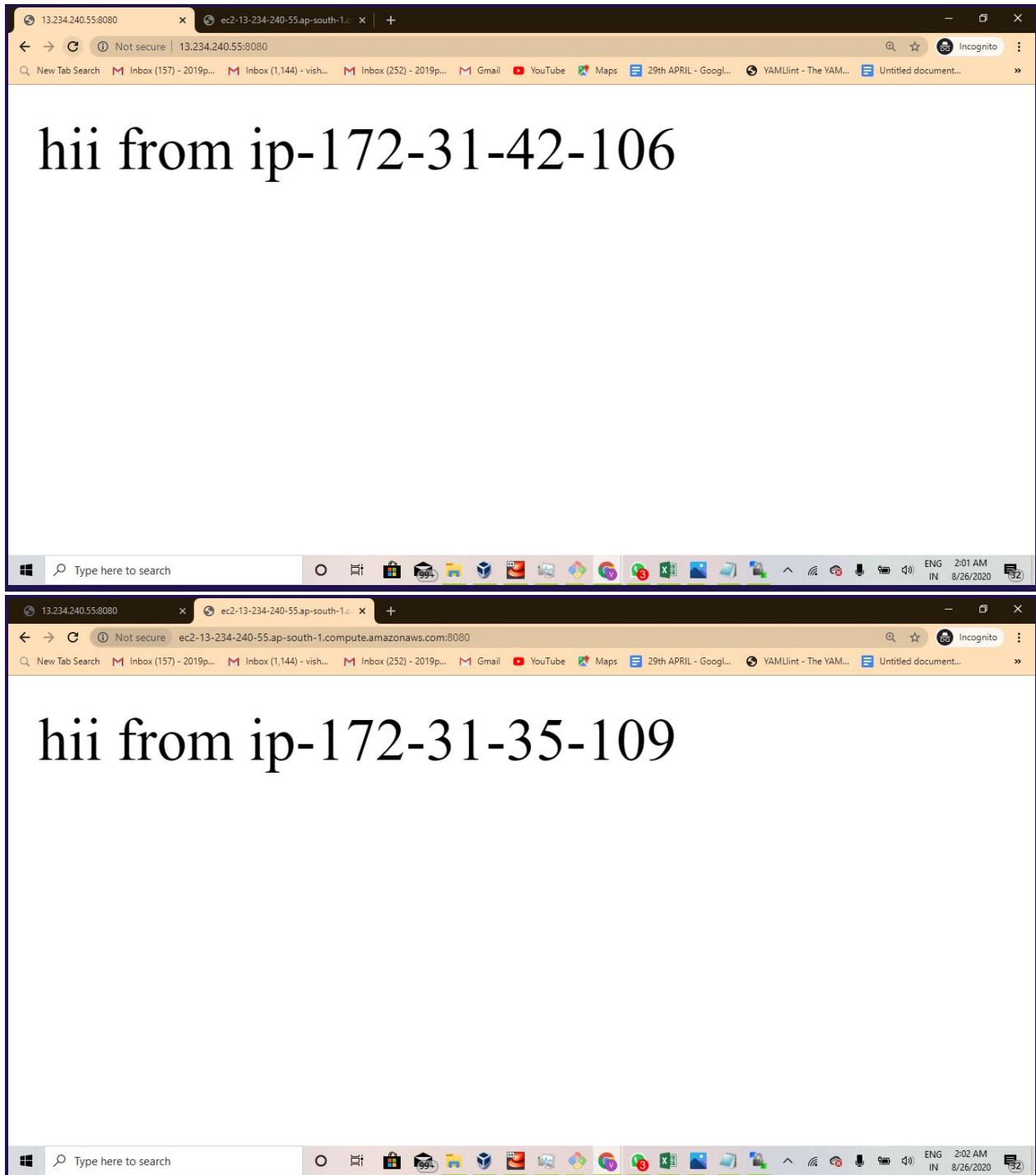
server app1 13.234.78.166:80 check

server app1 15.206.94.86:80 check
```

## Step 14: Outputs







## Note:

Allow all the traffic and launch the instances in the default vpc

Thanks