Lab manual – Ansible Env Setup details with Ansible playbook.

<u>Objective of the LAB.</u>

Setup an Ansible Controller.

Setup 2 server, Web and DB

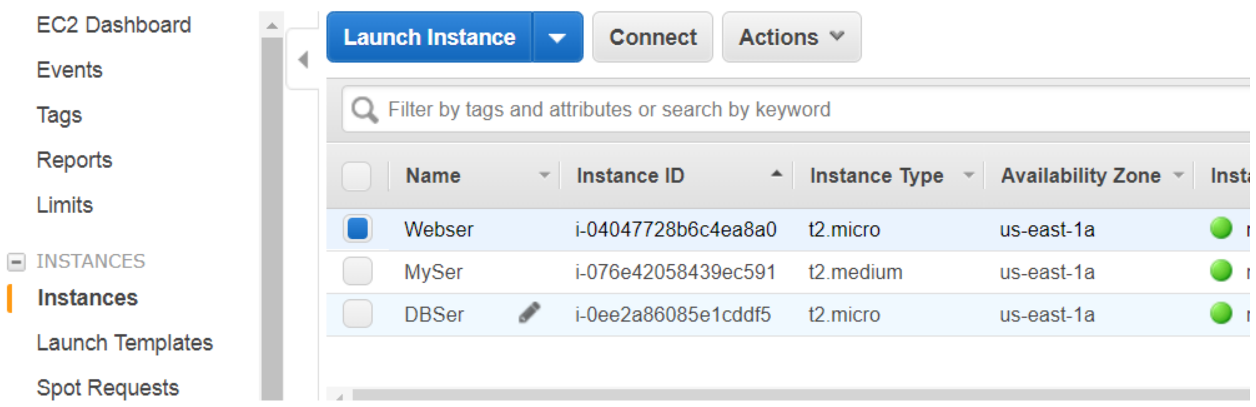
Configure the SSH on the ansible to authenticate on servers "web & DB".

Create an yaml file to install, web service on WEB host and MYSQL on DB server and start the service.

Also copy an index.html file to the "/var/www/html/" folder of WEB server.

You should be able to get the web page.

Step1: Create 3 EC2 instance in the AWS Account as below.



Name	Instance ID	Instance Type	Availability Zone	Insta
Webser	i-04047728b6c4ea8a0	t2.micro	us-east-1a	●
MySer	i-076e42058439ec591	t2.medium	us-east-1a	●
DBSer	i-0ee2a86085e1cddf5	t2.micro	us-east-1a	●

Let all the 3 EC2 instance have an **centos** image.

Note: -- so that the configuration of ssh authentication becomes easier.

Step2:

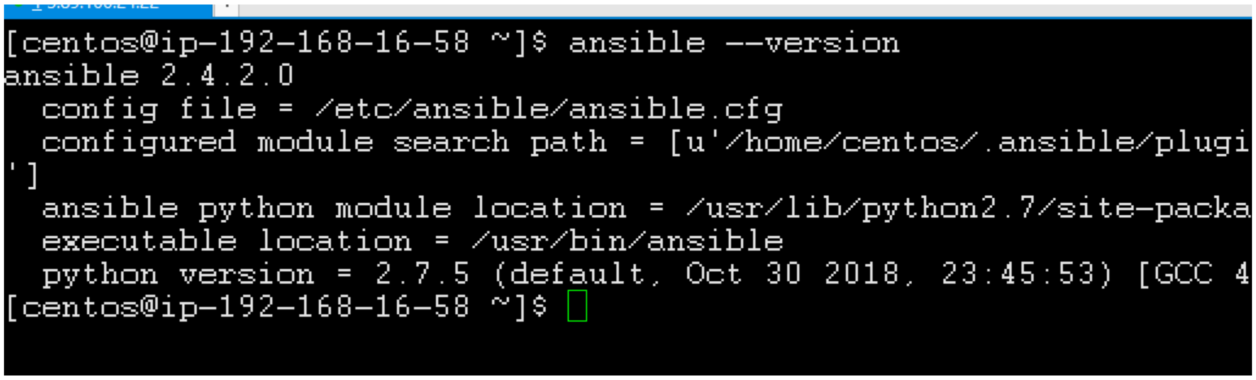
Configure the ssh login on the ansible server to auto login by ansible server itself

1. Login to the ansible server.

$

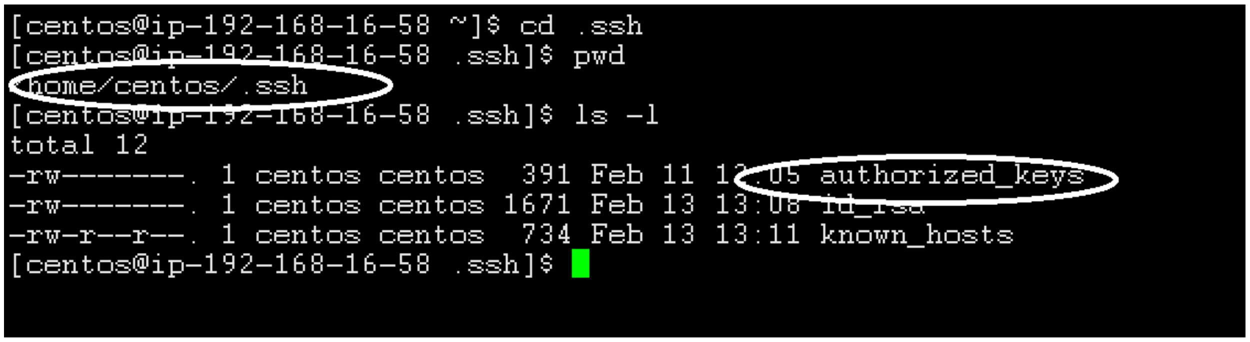
2. Install the Ansible software

$ yum install ansible



```
[centos@ip-192-168-16-58 ~]$ ansible --version
ansible 2.4.2.0
  config file = /etc/ansible/ansible.cfg
  configured module search path = [u'/home/centos/.ansible/plugi
']
  ansible python module location = /usr/lib/python2.7/site-packa
  executable location = /usr/bin/ansible
  python version = 2.7.5 (default, Oct 30 2018, 23:45:53) [GCC 4
[centos@ip-192-168-16-58 ~]$
```

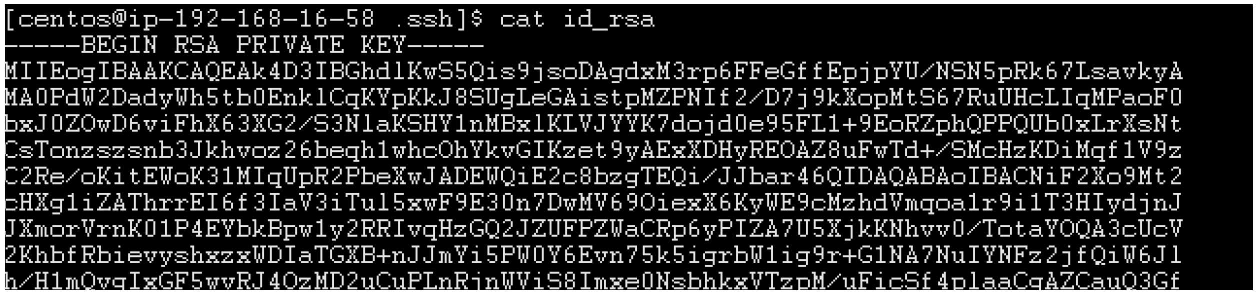
3. Make sure the ".ssh" folder exists in ansible server as below.



```
[centos@ip-192-168-16-58 ~]$ cd .ssh
[centos@ip-192-168-16-58 .ssh]$ pwd
/home/centos/.ssh
[centos@ip-192-168-16-58 .ssh]$ ls -l
total 12
-rw-------. 1 centos centos  391 Feb 11 12:05 authorized_keys
-rw-------. 1 centos centos 1671 Feb 13 13:08 id_rsa
-rw-r--r--. 1 centos centos  734 Feb 13 13:11 known_hosts
[centos@ip-192-168-16-58 .ssh]$
```

Also verify that **"authorized_keys"** file is present there.

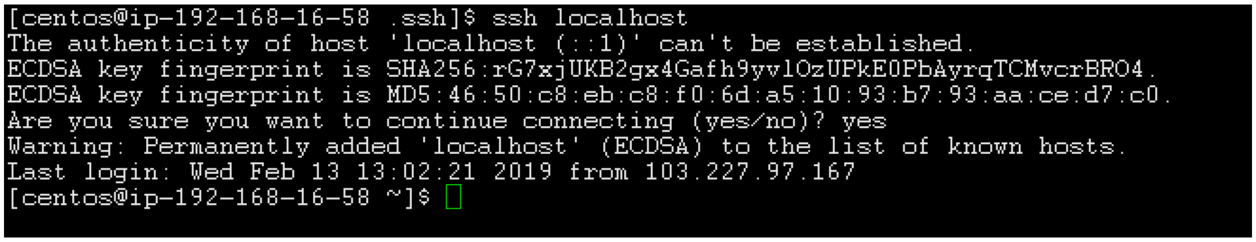
4. Now that the ansible server need to authenticate, it would need the **“private key”** as well.
   So copy the content of the “**.pem”** file of the ec2 instance in the below path **“/home/centos/.ssh”**

   Create an file “id_rsa”.
   Remember the file name has to be exactly as above.
   

   ```
   [centos@ip-192-168-16-58 .ssh]$ cat id_rsa
   -----BEGIN RSA PRIVATE KEY-----
   MIIEogIBAAKCAQEAk4D3IBGhdlKwS5Qis9jsoDAgdxM3rp6FFeGffEpjpYU/NSN5pRk67LsavkyA
   MA0PdW2DadyWh5tb0EnklCqKYpKkJ8SUgLeGAistpMZPNIf2/D7j9kXopMtS67RuUHcLIqMPaoF0
   bxJ0ZOwD6viFhX63XG2/S3NlaKSHY1nMBxlKLVJYYK7dojd0e95FL1+9EoRZphQPPQUb0xLrXsNt
   CsTonzszsnb3Jkhvoz26beqh1whcOhYkvGIKzet9yAExXDHyREOAZ8uFwTd+/SMcHzKDiMqf1V9z
   C2Re/oKitEWoK31MIqUpR2PbeXwJADEWQiE2c8bzgTEQi/JJbar46QIDAQABAoIBACNiF2Xo9Mt2
   cHXg1iZAThrrEI6f3IaV3iTul5xwF9E30n7DwMV69OiexX6KyWE9cMzhdVmqoa1r9i1T3HIydjnJ
   JXmorVrnK01P4EYbkBpw1y2RRIvqHzGQ2JZUFPZWaCRp6yPIZA7U5XjkKNhvv0/TotaYOQA3cUcV
   2KhbfRbievyshxzxWDIaTGXB+nJJmYi5PW0Y6Evn75k5igrbW1ig9r+G1NA7NuIYNFz2jfQiW6J1
   h/H1mQvqIxGF5wvRJ4OzMD2uCuPLnRjnWViS8Imxe0NsbhkxVTzpM/uFicSf4plaaCqAZCauQ3Gf
   ```

   The content of this file has to be same as your **“.pem”** that you would hve downloaded while creating the EC2 instance.

5. Test whether ansible controller is able to itself without prompting for password



```
[centos@ip-192-168-16-58 .ssh]$ ssh localhost
The authenticity of host 'localhost (::1)' can't be established.
ECDSA key fingerprint is SHA256:rG7xjUKB2gx4Gafh9yvlOzUPkE0PbAyrqTCMvcrBRO4.
ECDSA key fingerprint is MD5:46:50:c8:eb:c8:f0:6d:a5:10:93:b7:93:aa:ce:d7:c0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
Last login: Wed Feb 13 13:02:21 2019 from 103.227.97.167
[centos@ip-192-168-16-58 ~]$
```

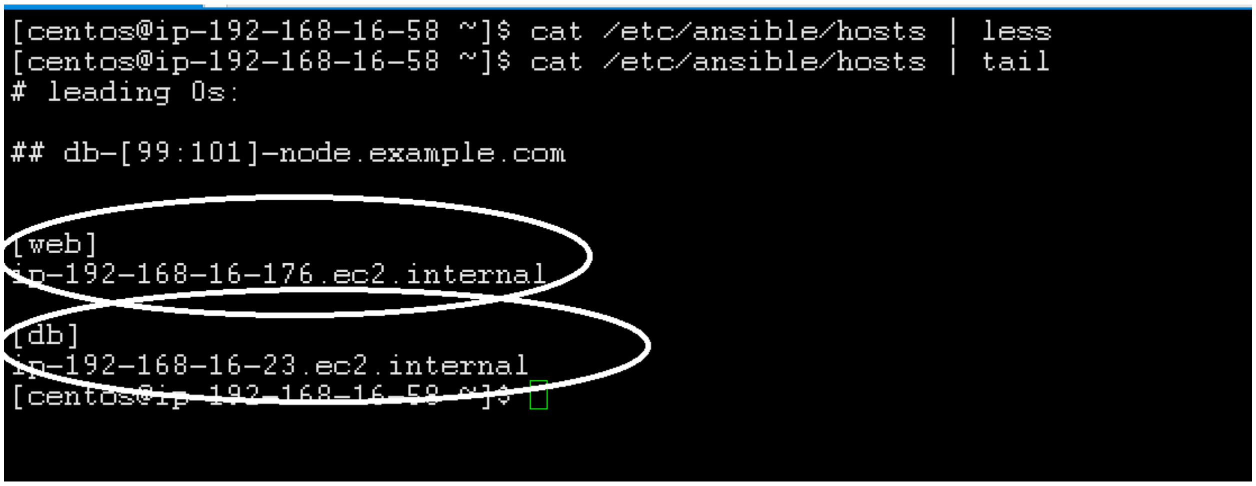
If this fails.

$ chmod 600 ~/.ssh/id_rsa

This would set the permission on the key file to private, only to the owner.

Step3:

1. Now let's configure the "**Ansible hosts"** on the Ansible controller as below.

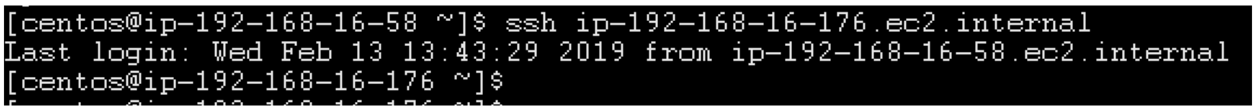


```
[centos@ip-192-168-16-58 ~]$ cat /etc/ansible/hosts | less
[centos@ip-192-168-16-58 ~]$ cat /etc/ansible/hosts | tail
# leading 0s:

## db-[99:101]-node.example.com


[web]
ip-192-168-16-176.ec2.internal

[db]
ip-192-168-16-23.ec2.internal
[centos@ip-192-168-16-58 ~]$
```

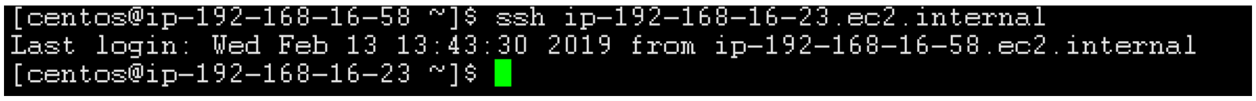
The one circled are the private host name's of the EC2 instances, ie. **"web"** and **"db".**

In this example we have all the EC2 instance in the same VPC. (if required keep it in the same subnet).

2. Now test whether you are able to login to the **"web"** and "**db"** from the ansible controller.

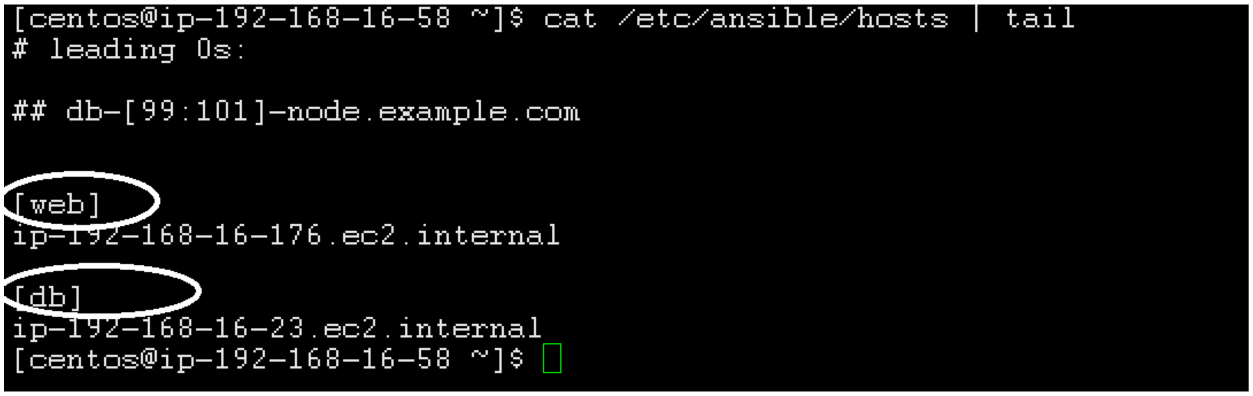


```
[centos@ip-192-168-16-58 ~]$ ssh ip-192-168-16-176.ec2.internal
Last login: Wed Feb 13 13:43:29 2019 from ip-192-168-16-58.ec2.internal
[centos@ip-192-168-16-176 ~]$
```



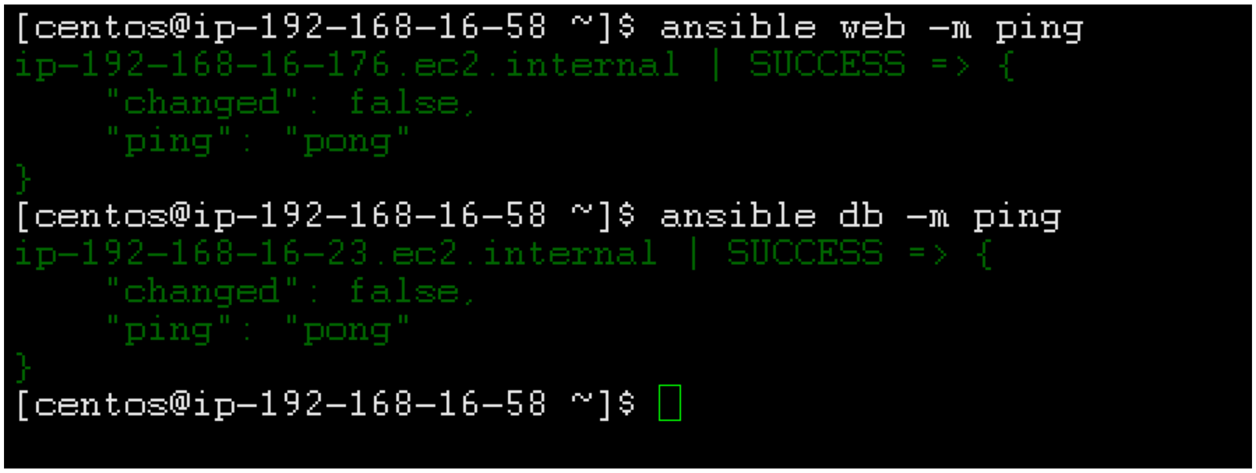
```
[centos@ip-192-168-16-58 ~]$ ssh ip-192-168-16-23.ec2.internal
Last login: Wed Feb 13 13:43:30 2019 from ip-192-168-16-58.ec2.internal
[centos@ip-192-168-16-23 ~]$
```

3. Now let's test a simple ansible command.
   In the ansible hosts file we need to define the group of server as marked in circle.



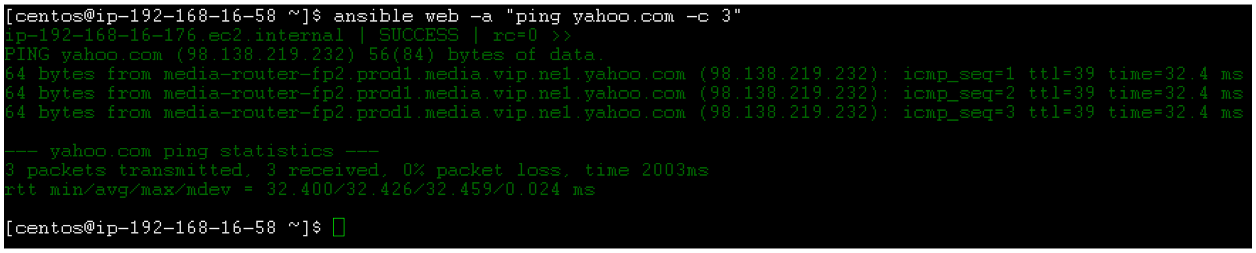
```
[centos@ip-192-168-16-58 ~]$ cat /etc/ansible/hosts | tail
# leading 0s:

## db-[99:101]-node.example.com


[web]
ip-192-168-16-176.ec2.internal

[db]
ip-192-168-16-23.ec2.internal
[centos@ip-192-168-16-58 ~]$
```

The below command will just "ping" test from the destination servers **"web"** and **"db".**
This uses the **"ansible"** module called **"ping" to** ping from the destination server to the ansible server.



```
[centos@ip-192-168-16-58 ~]$ ansible web -m ping
ip-192-168-16-176.ec2.internal | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
[centos@ip-192-168-16-58 ~]$ ansible db -m ping
ip-192-168-16-23.ec2.internal | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
[centos@ip-192-168-16-58 ~]$
```

4. Now lets try to execute an command on the **“web”** server.



```
[centos@ip-192-168-16-58 ~]$ ansible web -a "ping yahoo.com -c 3"
ip-192-168-16-176.ec2.internal | SUCCESS | rc=0 >>
PING yahoo.com (98.138.219.232) 56(84) bytes of data.
64 bytes from media-router-fp2.prod1.media.vip.ne1.yahoo.com (98.138.219.232): icmp_seq=1 ttl=39 time=32.4 ms
64 bytes from media-router-fp2.prod1.media.vip.ne1.yahoo.com (98.138.219.232): icmp_seq=2 ttl=39 time=32.4 ms
64 bytes from media-router-fp2.prod1.media.vip.ne1.yahoo.com (98.138.219.232): icmp_seq=3 ttl=39 time=32.4 ms

--- yahoo.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 32.400/32.426/32.459/0.024 ms

[centos@ip-192-168-16-58 ~]$
```

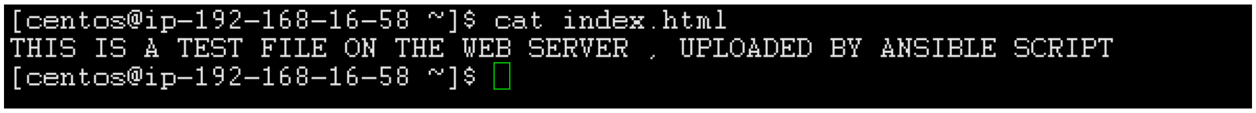
This would execute “ping yahoo.com -c 3” on the destination “web” server.

Step4:

Now lets install & start the web service (httpd) on the **“web”** server and install & start the DB (mysql) on the **“db”** server

Also we would be coping “index.html” from the working directory of ansible controller to the **“web”** server.

Content of the **“index.html”**



```
[centos@ip-192-168-16-58 ~]$ cat index.html
THIS IS A TEST FILE ON THE WEB SERVER , UPLOADED BY ANSIBLE SCRIPT
[centos@ip-192-168-16-58 ~]$
```

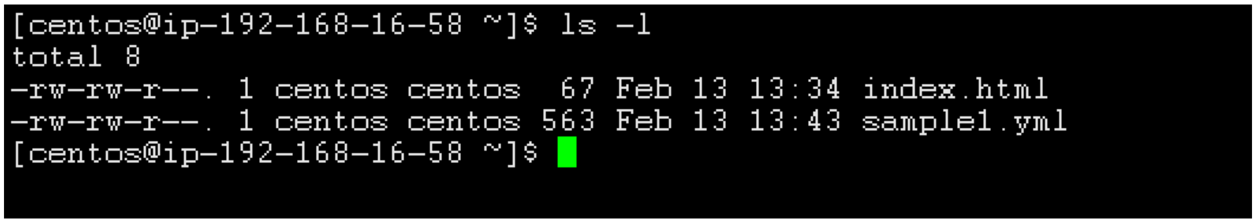
We would be doing this with an sample.yml file created on the ansible controller.

The content of the file as below.

```
---
- name: Install httpd
  hosts: web
  become: true
  tasks:
  - name: Install httpd on web server
    yum:
          name: httpd
          state: present
  - name: Insert index page
    template:
        src: index.html
        dest: /var/www/html/index.html
  - name: start the httpd service
    service:
        name: httpd
        state: started

- name: Install DB
  hosts: db
  become: true

  tasks:
  - name: Install DB on DB Server
    yum:
        name: mysql-server
        state: present
- name: Install DB on DB Server
    service:
        name: mysqld
        state: started
```

====================================

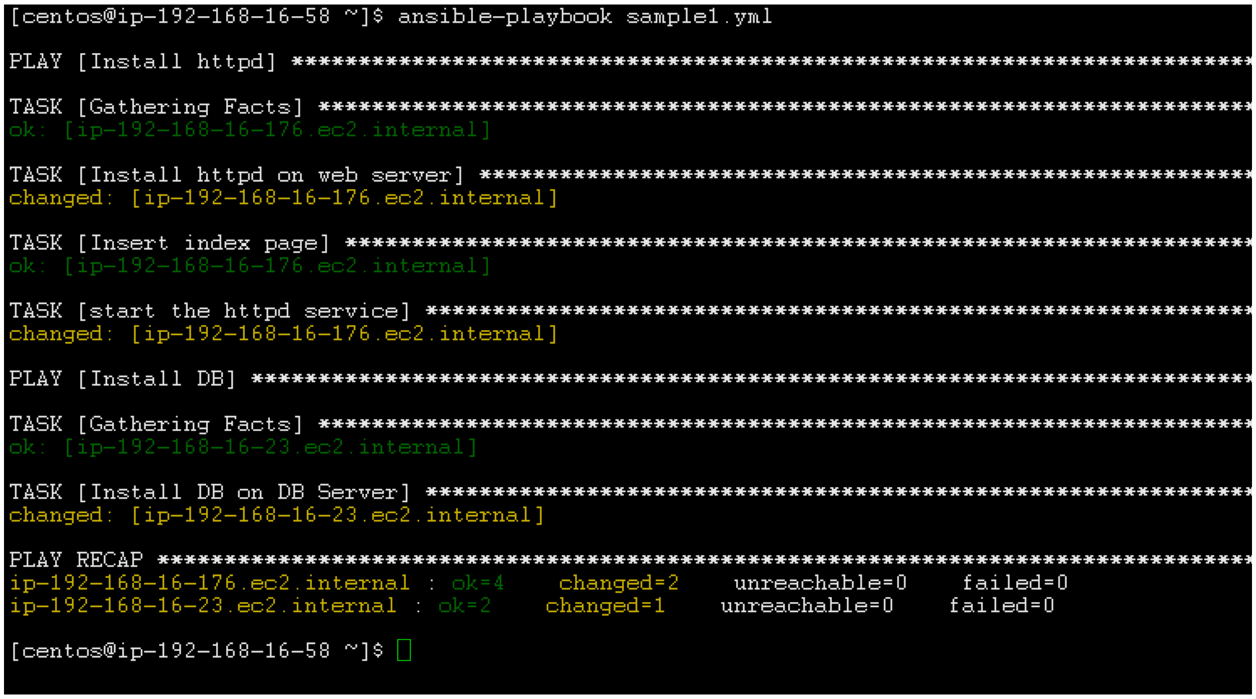
Also create an index.html file in the same folder of the “sample.yml” file on the ansible controller.



```
[centos@ip-192-168-16-58 ~]$ ls -l
total 8
-rw-rw-r--. 1 centos centos  67 Feb 13 13:34 index.html
-rw-rw-r--. 1 centos centos 563 Feb 13 13:43 sample1.yml
[centos@ip-192-168-16-58 ~]$
```

Now that all the env and the file is ready, let’s execute the yaml file for the result.

$ ansible-playbook sample1.yml



```
[centos@ip-192-168-16-58 ~]$ ansible-playbook sample1.yml

PLAY [Install httpd] *********************************************************************

TASK [Gathering Facts] *******************************************************************
ok: [ip-192-168-16-176.ec2.internal]

TASK [Install httpd on web server] *******************************************************
changed: [ip-192-168-16-176.ec2.internal]

TASK [Insert index page] *****************************************************************
ok: [ip-192-168-16-176.ec2.internal]

TASK [start the httpd service] ***********************************************************
changed: [ip-192-168-16-176.ec2.internal]

PLAY [Install DB] ************************************************************************

TASK [Gathering Facts] *******************************************************************
ok: [ip-192-168-16-23.ec2.internal]

TASK [Install DB on DB Server] ***********************************************************
changed: [ip-192-168-16-23.ec2.internal]

PLAY RECAP *******************************************************************************
ip-192-168-16-176.ec2.internal : ok=4    changed=2    unreachable=0    failed=0
ip-192-168-16-23.ec2.internal  : ok=2    changed=1    unreachable=0    failed=0

[centos@ip-192-168-16-58 ~]$
```

The above shows the step by step execution of the ansible lines.

4 parts in the output.

PLAY

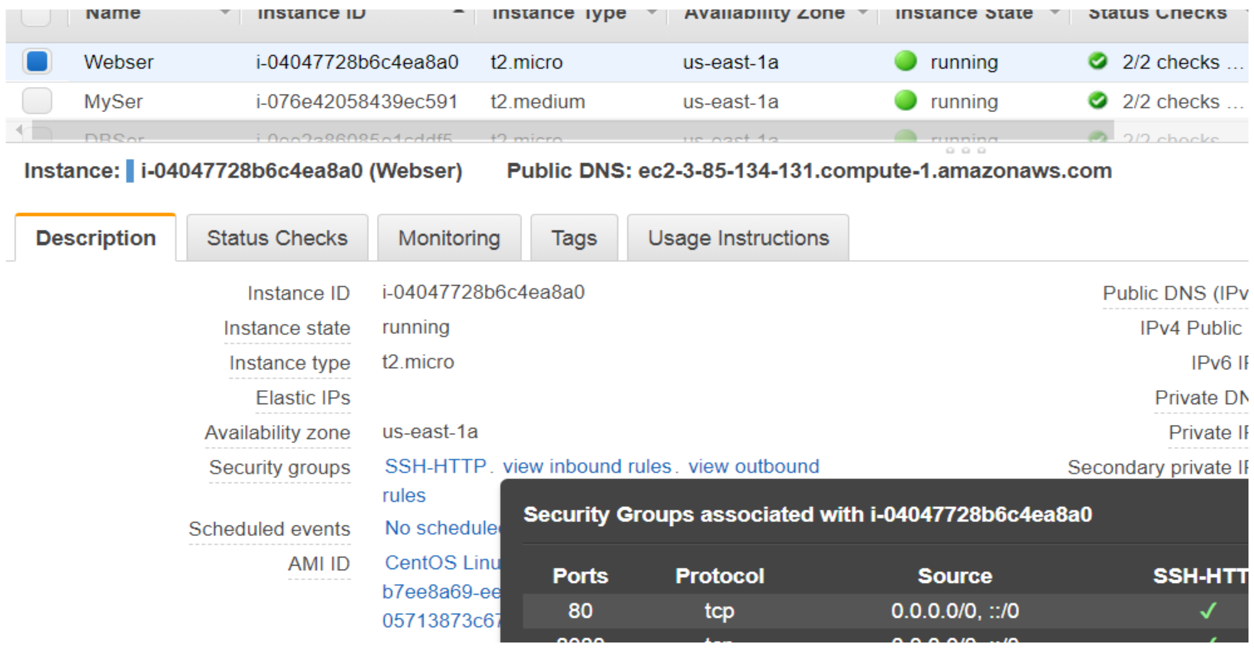
TASK (Gathering Facts)

TASK (Installing DB……)

PLAY RECAP

Step5:

Let’s check the output now.



	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks
■	Webser	i-04047728b6c4ea8a0	t2.micro	us-east-1a	● running	✓ 2/2 checks ...
	MySer	i-076e42058439ec591	t2.medium	us-east-1a	● running	✓ 2/2 checks ...
	DBSer	i-0ec2a86085e1cddf5	t2.micro	us-east-1a	● running	✓ 2/2 checks

Instance: i-04047728b6c4ea8a0 (Webser)   Public DNS: ec2-3-85-134-131.compute-1.amazonaws.com

Description | Status Checks | Monitoring | Tags | Usage Instructions

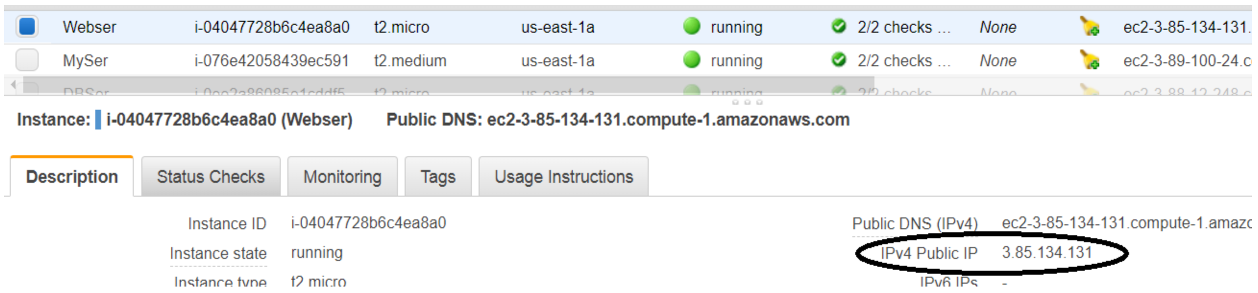
Instance ID	i-04047728b6c4ea8a0
Instance state	running
Instance type	t2.micro
Elastic IPs	
Availability zone	us-east-1a
Security groups	SSH-HTTP. view inbound rules. view outbound rules
Scheduled events	No schedule
AMI ID	CentOS Linu b7ee8a69-ee 05713873c67

Public DNS (IPv
IPv4 Public
IPv6 IP
Private DN
Private IP
Secondary private IP

Security Groups associated with i-04047728b6c4ea8a0

Ports	Protocol	Source	SSH-HTT
80	tcp	0.0.0.0/0, ::/0	✓
8080	tcp	0.0.0.0/0, ::/0	✓

Make sure the port **“80”** is enabled on the webserver.



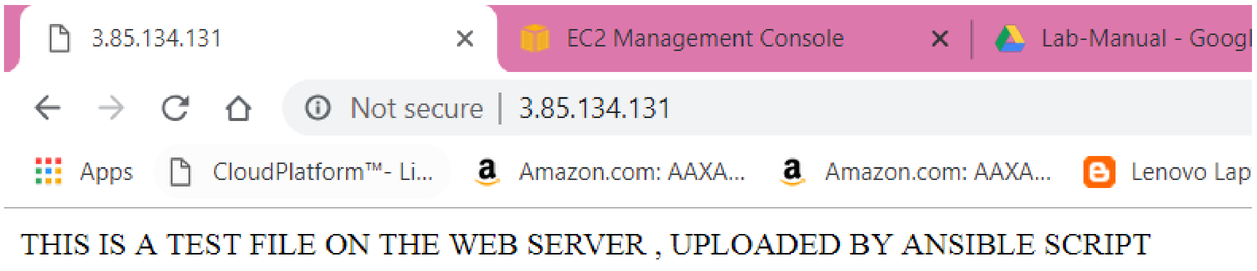
	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks			
■	Webser	i-04047728b6c4ea8a0	t2.micro	us-east-1a	● running	✓ 2/2 checks ...	None		ec2-3-85-134-131.
	MySer	i-076e42058439ec591	t2.medium	us-east-1a	● running	✓ 2/2 checks ...	None		ec2-3-89-100-24.c
	DBSer	i-0ec2a86085e1cddf5	t2.micro	us-east-1a	● running	✓ 2/2 checks	None		ec2-3-88-12-248.c

Instance: i-04047728b6c4ea8a0 (Webser)   Public DNS: ec2-3-85-134-131.compute-1.amazonaws.com

Description | Status Checks | Monitoring | Tags | Usage Instructions

Instance ID	i-04047728b6c4ea8a0	Public DNS (IPv4)	ec2-3-85-134-131.compute-1.amazo
Instance state	running	IPv4 Public IP	3.85.134.131
Instance type	t2.micro	IPv6 IPs	-

Let’s check the output with public ip of the webserver.



THIS IS A TEST FILE ON THE WEB SERVER , UPLOADED BY ANSIBLE SCRIPT