

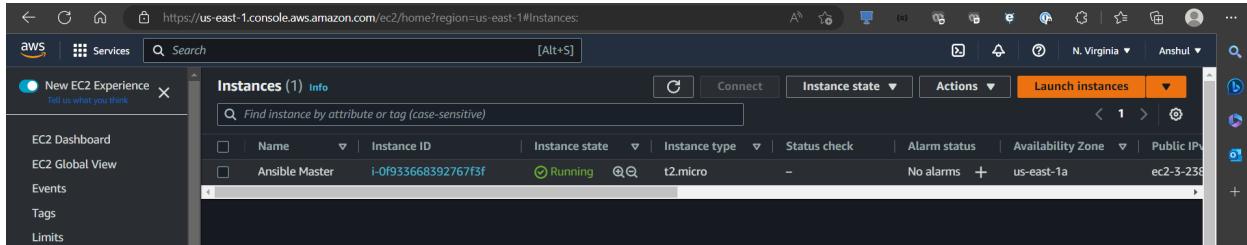
Ansible - 1

You have been asked to:

- Setup Ansible cluster with 3 nodes
- On slave1 install java
- On slave 2 install mysql-server

Do the above tasks using Ansible playbooks.

Let us create a Ansible Master server on EC2 first.

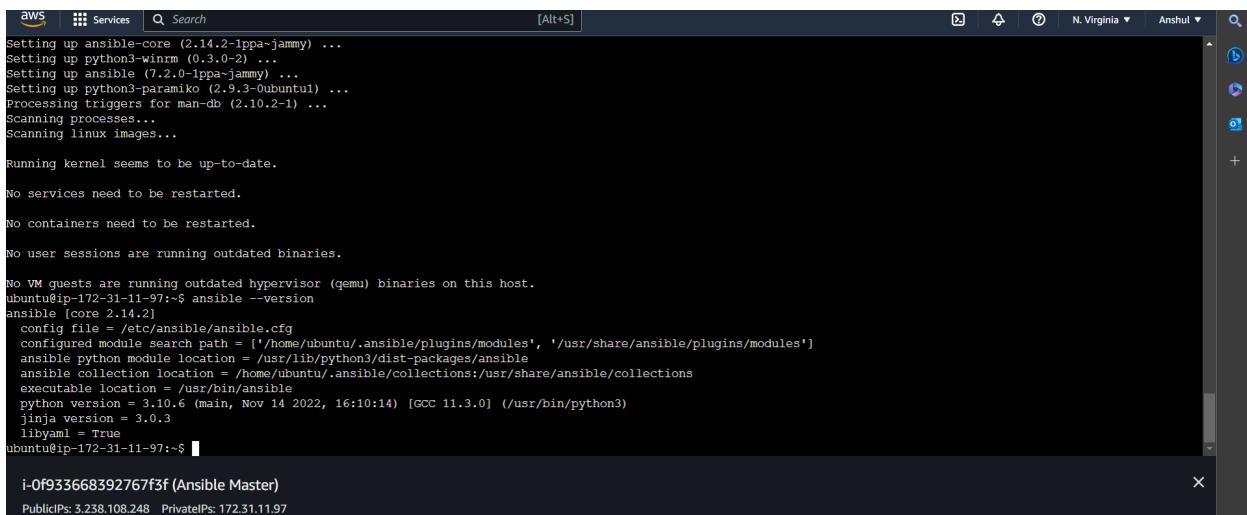


Then we SSH into it to install Ansible to create a cluster.

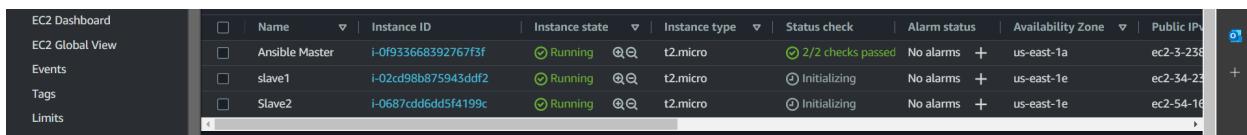
To install ansible on ubuntu we use these commands:

```
$ sudo apt update  
$ sudo apt install software-properties-common  
$ sudo add-apt-repository --yes --update ppa:ansible/ansible  
$ sudo apt install ansible
```

Ansible is installed after using above commands:ansible --version



As per task, let us create two slaves i.e. two more EC2 instances to complete a 3 node cluster. And then SSH to them.



Now to add them in hosts we need to give them keypair details for ssh.

We use system-keygen on master server.

```
ubuntu@ip-172-31-11-97:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ubuntu/.ssh/id_rsa
Your public key has been saved in /home/ubuntu/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:+Bduk2KwGN3RpWYH0Ypf0whXoCFyoriLJY77KTZY ubuntu@ip-172-31-11-97
The key's randomart image is:
+---[RSA 3072]---+
|...
|+ . .
|o . *
|o . B o
|o . O S .
|o . =.+ + o
|o. E ==o+ = .
|*++..+= =
|B+o. .o o .
+---[SHA256]---+
ubuntu@ip-172-31-11-97:~$
```

Now we will do cat command on that file path to get key. Copy this key.

```
the key's randomart image is:
+---[RSA 3072]---+
|...
|+ . .
|o . *
|o . B o
|o . O S .
|o . =.+ + o
|o. E ==o+ = .
|*++..+= =
|B+o. .o o .
+---[SHA256]---+
ubuntu@ip-172-31-11-97:~$ cat /home/ubuntu/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQCBYJQ9/aenBcZ9ZerF9RPSSvS/0k+HKzk2WxaEF54mxo0Chaa2MuWS1BvzoAjmbN19WnV0qumx1snmQdubaBmMpIgB5xzoi1hQ36liEvH32ZoqWIDWTzCvptNi0g5qxw
fjUomyx1mXXubCoritRgj+yK12MzvovOfcB1nsJ3xd1lx6UZwdzaymTl+DsgxchcPP3KnMsUfv8aGaoJ5vg96nRp7MSmina3h1mvvaUCDzQyNWtzQy3nrsvWlglU1lsrn54ypK01uvv+2zmY63vs7XH7mqjjaZiaEB
Rb4gtnt4ePhftvbIU00PK7NAN1aPv5mr3v/oY6KBeilPv7swccszNohwWpQMfHICekptSdbvzR4MRZz5uGcs4lw1k8H68yoRM9DuvfzLkNAG/feBFSkj+TKxX9eiwssZ7xj8ar09b0+GhwxYvJM2J4VIfsnxt2lo3j0vk
B6jvW2UJGC7edbq4uhvhbTVIp+SyaBcNzR/Eog7xRBfdwWjw= ubuntu@ip-172-31-11-97:~$
```

Now that we got the keys, we will paste them into slave servers so that later we can use them from master server.

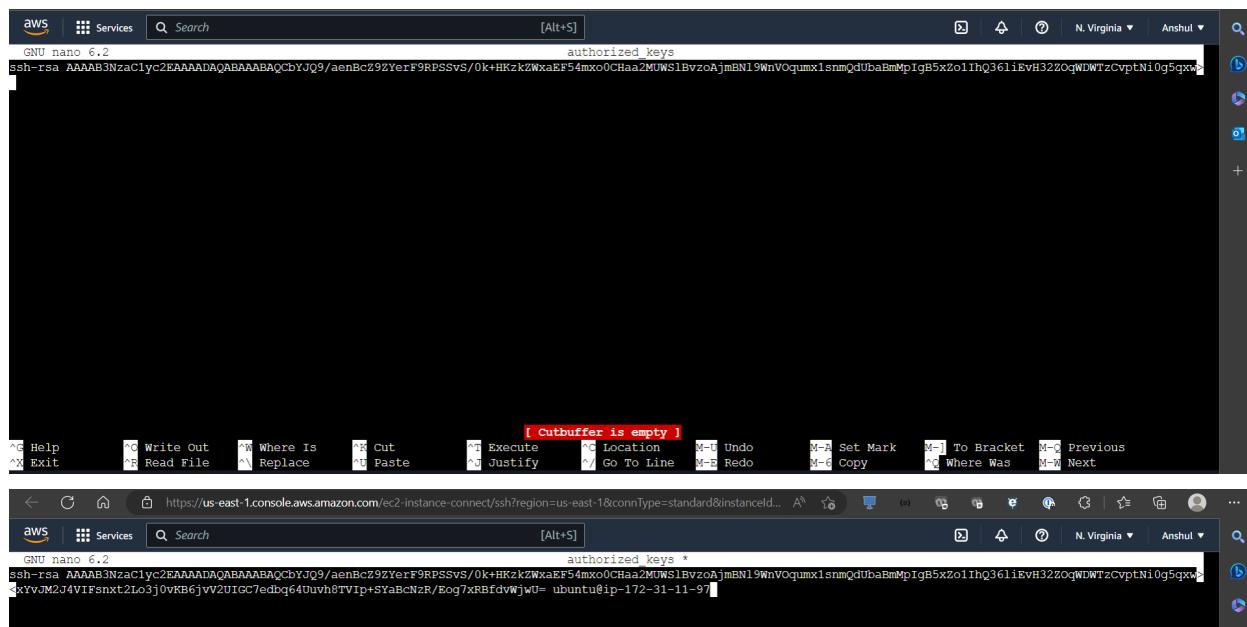
GO to one of the slave server, then use these commands:

cd .ssh

Nano authorized_keys

```
Reading state information... Done
17 packages can be upgraded. Run 'apt list --upgradable' to see them.
ubuntu@ip-172-31-60-154:~$ cd .ssh
c: command not found
ubuntu@ip-172-31-60-154:~$ ssh5 nano authorized_keys
ubuntu@ip-172-31-60-154:~$
```

Now here we have a key already their. Paste the key(that you got from master on next line, and save it. Do this on both slave servers.



```
authorised keys
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQCBYJQ9/aenBcZ9ZerF9RPSSvS/0k+HKzk2WxaEF54mxo0Chaa2MuWS1BvzoAjmbN19WnV0qumx1snmQdubaBmMpIgB5xzoi1hQ36liEvH32ZoqWIDWTzCvptNi0g5qxw
fjUomyx1mXXubCoritRgj+yK12MzvovOfcB1nsJ3xd1lx6UZwdzaymTl+DsgxchcPP3KnMsUfv8aGaoJ5vg96nRp7MSmina3h1mvvaUCDzQyNWtzQy3nrsvWlglU1lsrn54ypK01uvv+2zmY63vs7XH7mqjjaZiaEB
Rb4gtnt4ePhftvbIU00PK7NAN1aPv5mr3v/oY6KBeilPv7swccszNohwWpQMfHICekptSdbvzR4MRZz5uGcs4lw1k8H68yoRM9DuvfzLkNAG/feBFSkj+TKxX9eiwssZ7xj8ar09b0+GhwxYvJM2J4VIfsnxt2lo3j0vk
B6jvW2UJGC7edbq4uhvhbTVIp+SyaBcNzR/Eog7xRBfdwWjw=
```

Next, we need to add both the slaves in the hosts list.

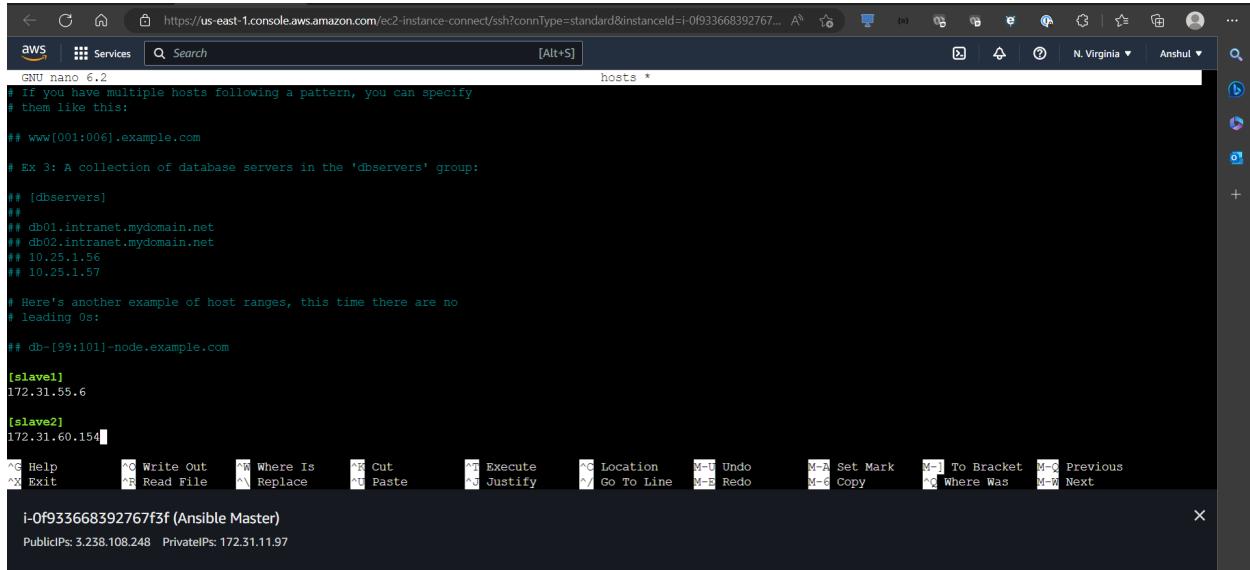
For this, copy the private ip's of both the slave instances.

Goto master server. Use these commands:

cd /etc/ansible

Nano hosts

Here paste the private ip of slave servers as shown below.



```
GNU nano 6.2
# If you have multiple hosts following a pattern, you can specify
# them like this:
## www[001:006].example.com
# Ex 3: A collection of database servers in the 'dbservers' group:
## [dbservers]
## db01.intranet.mydomain.net
## db02.intranet.mydomain.net
## 10.25.1.56
## 10.25.1.57

# Here's another example of host ranges, this time there are no
# leading 0s:
## db-[99:101]-node.example.com

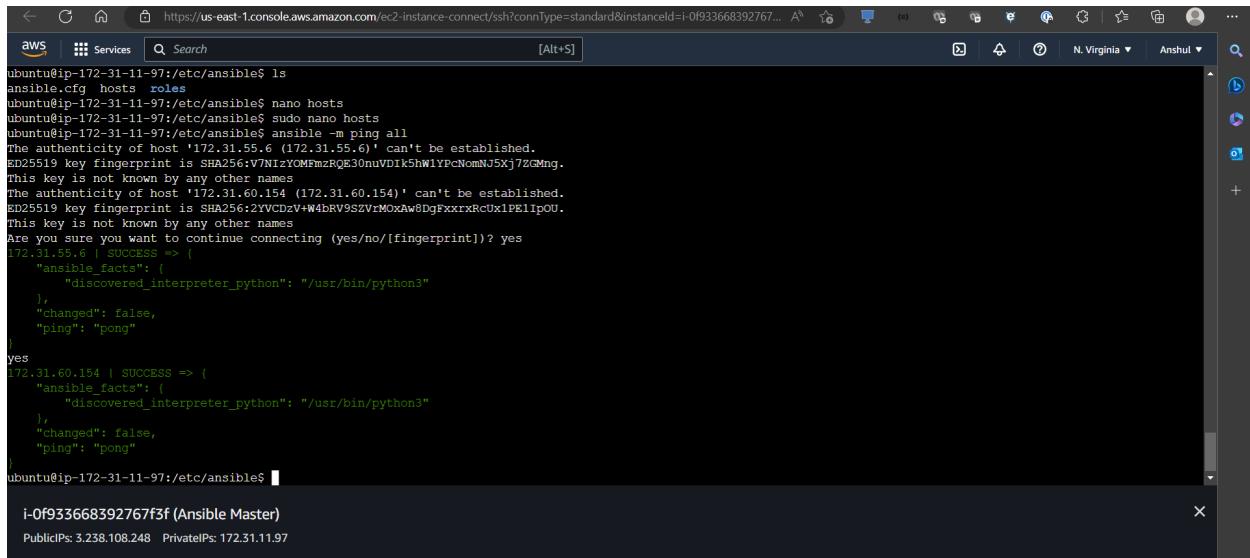
[slave1]
172.31.55.6

[slave2]
172.31.60.154

i-Of933668392767f3f (Ansible Master)
PublicIPs: 3.238.108.248 PrivateIPs: 172.31.11.97
```

Now that both the slave nodes are added. Let us ping them to check the connectivity by using the command:

Ansible -m ping all



```
ubuntu@ip-172-31-11-97:/etc/ansible$ ls
ansible.cfg  hosts  roles
ubuntu@ip-172-31-11-97:/etc/ansible$ nano hosts
ubuntu@ip-172-31-11-97:/etc/ansible$ sudo nano hosts
ubuntu@ip-172-31-11-97:/etc/ansible$ ansible -m ping all
The authenticity of host '172.31.55.6 (172.31.55.6)' can't be established.
ED25519 key fingerprint is SHA256:V7N1zXOMFmzRQE30nuVDIk5hW1YPcNomNJ5Xj7ZGmng.
This key is not known by any other names
The authenticity of host '172.31.60.154 (172.31.60.154)' can't be established.
ED25519 key fingerprint is SHA256:2tVCDEv+W4bRV9S2VRMxawBdgFxxrxRcUx1PElpoU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
172.31.55.6 | SUCCESS => (
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
)
yes
172.31.60.154 | SUCCESS => (
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
)
ubuntu@ip-172-31-11-97:/etc/ansible$ i-Of933668392767f3f (Ansible Master)
PublicIPs: 3.238.108.248 PrivateIPs: 172.31.11.97
```

Now that nodes are ready.

We will create a YAML file playbook for our task and then deploy them.

We are supposed to install java on slave1 and mysql on slave2.

For that YAML file playbook is as follows:

```
---
- name: tasks for assign1 on w1
  hosts: slave1
  become: true
  tasks:
    - name: installing java
      apt: name=openjdk-11-jdk update-cache=yes state=latest
- name: tasks for assign2 on w2
  hosts: slave2
  become: true
  tasks:
    - name: installing mysql
      apt: name=mysql-server update-cache=yes state=latest
```

The first section, named "tasks for assign1 on w1," is executed on the "slave1" host. It uses the apt module to install the OpenJDK 11 JDK package, which is a Java development kit. The update-cache=yes option tells Ansible to update the package cache before attempting to install the package, and the state=latest option tells Ansible to install the latest available version of the package.

The second section, named "tasks for assign2 on w2," is executed on the "slave2" host. It uses the apt module to install the MySQL Server package. Like before, the update-cache=yes option tells Ansible to update the package cache before attempting to install the package, and the state=latest option tells Ansible to install the latest available version of the package.

Overall, this playbook is designed to automate the installation of Java and MySQL Server on two different hosts, "slave1" and "slave2," respectively. The become: true option allows Ansible to execute the tasks with elevated privileges (i.e. as root), which is required for package installation.

We will create a directory and then create this YAML file playbook init on master server.

```
mkdir Playbook
Cd Playbook
nano assignment1.yaml
```

Then paste the yaml file written above here.

The screenshot shows a terminal window titled "assignmen1.yaml" in a browser-based interface. The file contains the following Ansible YAML code:

```
---
- name: tasks for assign1 on w1
  hosts: slave1
  become: true
  tasks:
    - name: installing java
      apt: name=openjdk-11-jdk update-cache=yes state=latest
- name: tasks for assign2 on w2
  hosts: slave2
  become: true
  tasks:
    - name: installing mysql
      apt: name=mysql-server update-cache=yes state=latest
```

The terminal also displays various keyboard shortcuts at the bottom, such as Help, Write Out, Where Is, Cut, Execute, Location, Undo, Redo, Set Mark, To Bracket, Previous, Exit, Read File, Replace, Justify, Go To Line, Copy, Where Was, and Next. The status bar at the bottom shows the session ID i-0f933668392767f3f (Ansible Master) and IP addresses 3.238.108.248 and 172.31.11.97.

**Let us run a status check and dry run test to test if the yaml file will run properly or not.
For syntax check use command: ansible-playbook <file.yaml>--syntax-check**

The screenshot shows a terminal window executing the command "ansible-playbook assignmen1.yaml --syntax-check". The output indicates no syntax errors were found:

```
playbook: assignmen1.yaml
ubuntu@ip-172-31-11-97:~/Playbook$ ansible-playbook assignmen1.yaml --syntax-check
playbook: assignmen1.yaml
ubuntu@ip-172-31-11-97:~/Playbook$
```

The status bar at the bottom shows the session ID i-0f933668392767f3f (Ansible Master) and IP addresses 3.238.108.248 and 172.31.11.97.

For dry run use command:ansible-playbook <file.yaml> --check

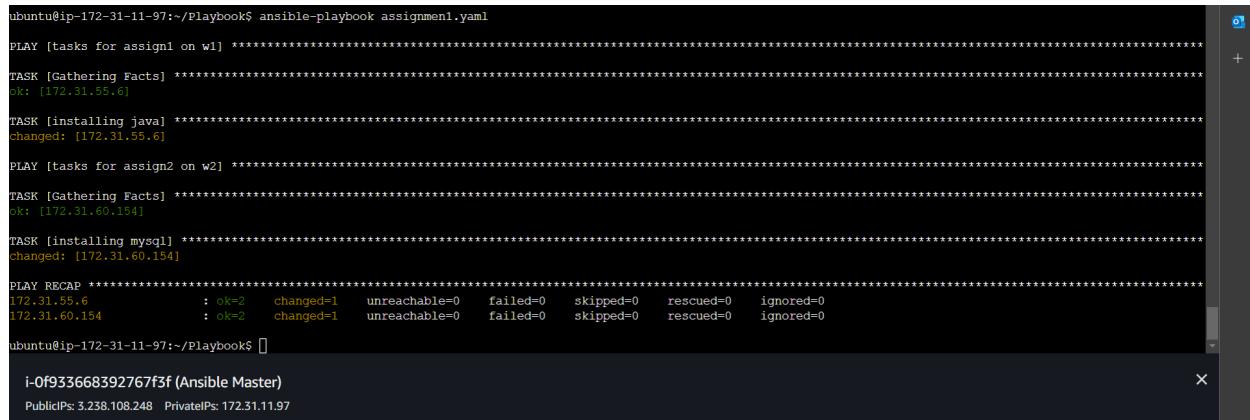
The screenshot shows a terminal window executing the command "ansible-playbook assignmen1.yaml --check". The output shows the execution of the playbook across two hosts (w1 and w2), detailing tasks like Java and MySQL installations, and reporting success (ok) and changes (changed) for each host:

```
playbook: assignmen1.yaml
ubuntu@ip-172-31-11-97:~/Playbook$ ansible-playbook assignmen1.yaml --check
PLAY [tasks for assign1 on w1] *****
TASK [Gathering Facts] *****
ok: [172.31.55.6]
TASK [installing java] *****
Changed: [172.31.55.6]
PLAY [tasks for assign2 on w2] *****
TASK [Gathering Facts] *****
ok: [172.31.60.154]
TASK [installing mysql] *****
Changed: [172.31.60.154]
PLAY RECAP *****
172.31.55.6 : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.31.60.154: ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

ubuntu@ip-172-31-11-97:~/Playbook$
```

The status bar at the bottom shows the session ID i-0f933668392767f3f (Ansible Master) and IP addresses 3.238.108.248 and 172.31.11.97.

**Now that both the tests are good, we can actually run those files. We use command:
ansible-playbook <filename.yaml>**

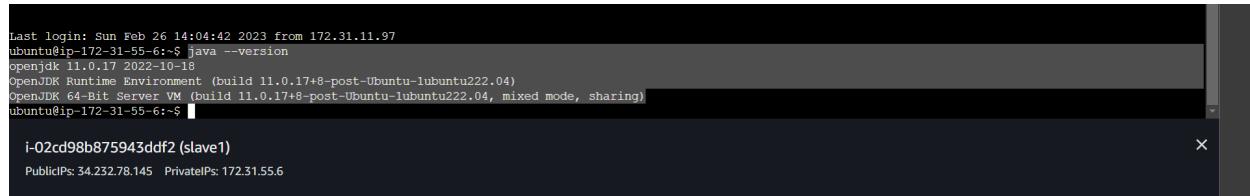


```
ubuntu@ip-172-31-11-97:~/Playbook$ ansible-playbook assignmen1.yaml
PLAY [tasks for assign1 on w1] ****
TASK [Gathering Facts] ****
ok: [172.31.55.6]
TASK [installing java] ****
changed: [172.31.55.6]
PLAY [tasks for assign2 on w2] ****
TASK [Gathering Facts] ****
ok: [172.31.60.154]
TASK [installing mysql] ****
changed: [172.31.60.154]
PLAY RECAP ****
172.31.55.6      : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
172.31.60.154    : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu@ip-172-31-11-97:~/Playbook$
```

Now that task is completed from master side.

Lets check if java and mysql are actually installed on worker side.

On slave1 Java is found.



```
Last login: Sun Feb 26 14:04:42 2023 from 172.31.11.97
ubuntu@ip-172-31-55-6:~$ java --version
openjdk 11.0.17 2022-10-18
OpenJDK Runtime Environment (build 11.0.17+8-post-Ubuntu-ubuntu22.04)
OpenJDK 64-Bit Server VM (build 11.0.17+8-post-Ubuntu-ubuntu22.04, mixed mode, sharing)
ubuntu@ip-172-31-55-6:~$
```

i-0f933668392767f3f (Ansible Master)

PublicIPs: 3.238.108.248 PrivateIPs: 172.31.11.97

And on slave2 mysql is installed.



```
Last login: Sun Feb 26 14:05:21 2023 from 172.31.11.97
ubuntu@ip-172-31-60-154:~$ mysql --version
mysql Ver 8.0.32-0ubuntu0.22.04.2 for Linux on x86_64 ((Ubuntu))
ubuntu@ip-172-31-60-154:~$
```

i-02cd98b875943ddf2 (slave1)

PublicIPs: 34.232.78.145 PrivateIPs: 172.31.55.6

Conclusion: We can configure slave/woker/servers from master and install files on them from master side successfully.