**Problem Statement**

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

**Solution Approach**

**Step 1: Create and Setup ec2 instances**

* We will create 3 ec2 instance: Master, Slave 1 and Slave 2
* We will set up the connections such that we can ssh from the Master into both Slave 1 and 2

**Step 2: Setting up Master-Slave architecture**

Used the following set of codes to set up master slave architecture between the Master and 2 Slaves

1. which python3 # check if python is installed

2.

3. # Install Ansible only in the master node

4. sudo yum install -y ansible

5.

6. # Create a new user with the name "ansible" in the MASTER node

7. sudo useradd ansible

8. sudo passwd ansible # Generate a password for the user

9. su – ansible # log in to the user

10.

11. # Create a new user with the name "ansible" in the both slave nodes

12. sudo useradd ansible

13. sudo passwd ansible # Generate a password for the user

14. su - ansible

15.

16. # Provide sudo access to the ansible user in the both the slave nodes

17. cd /etc/ # go to folder

18. sudo vi sudoers # open sudoers in a text editor

19. # add this to wheel: ansible ALL=(ALL) NOPASSWD: ALL

20.

21. # Generate a public and private key in the MASTER node

22. su - ansible # log in to the user

23. ssh-keygen # command to generate keypairs

24.

25. # In order to establish the ssh connection, we need the public IP of the Slave nodes

26. curl ifconfig.me

27.

28. # Run this in SLAVE node to enable ssh port

29. cd /etc/ssh # go to ssh folder

30. sudo vi sshd\_config # the file sshd\_config has details that needs to be changed

31. # Search for the setting "PasswordAuthentication" and change the setting from "no" to "yes"

32. sudo systemctl restart sshd # restart sshd service

33.

34. # Now try gaining remote access to both servers from Master

35. ssh ansible@3.111.149.212 # Slave 1

36. ssh ansible@ 65.0.122.233 # Slave 2

37.

38. # Copy public key from master to remote server

39. ssh-copy-id ansible@{slave public ip}

**Step 3: Creating ansible configuration and inventory file**

1. Creating the configuration file as follows:

1. [defaults]

2. inventory = /etc/ansible/hosts.ini

3. become = True

4. become\_method = sudo

5. become\_user = root

6. fork = 5

7. timeout = 30

1. Creating the inventory file as follows:

1. ---

2. [slave1]

3. slave1 ansible\_host=3.109.123.221 ansible\_user=ansible

4.

5. [slave2]

6. slave2 ansible\_host=65.0.122.233 ansible\_user=ansible

**Step 4: Creating the Ansible Playbook**

The ansible playbook with the name *task1.yml* has been designed to do the required tasks:

1. Install Java in slave 1
2. Install mysql-server in slave 2

1. ---

2. - name: Install Java on slave1

3. hosts: slave1

4. remote\_user: ansible

5. become: true

6. tasks:

7. - name: Install Java

8. yum:

9. name: java

10. state: present

11.

12. - name: Install MySQL on slave2

13. hosts: slave2

14. remote\_user: ansible

15. become: true

16. tasks:

17. - name: Import MySQL Repo

18. shell: sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql-2022

19. - name: Task 2

20. shell: wget http://dev.mysql.com/get/mysql57-community-release-el7-8.noarch.rpm

21. - name: Task 3

22. shell: sudo yum localinstall -y mysql57-community-release-el7-8.noarch.rpm

23. - name:

24. yum:

25. name: mysql-community-server

26. state: present

27.

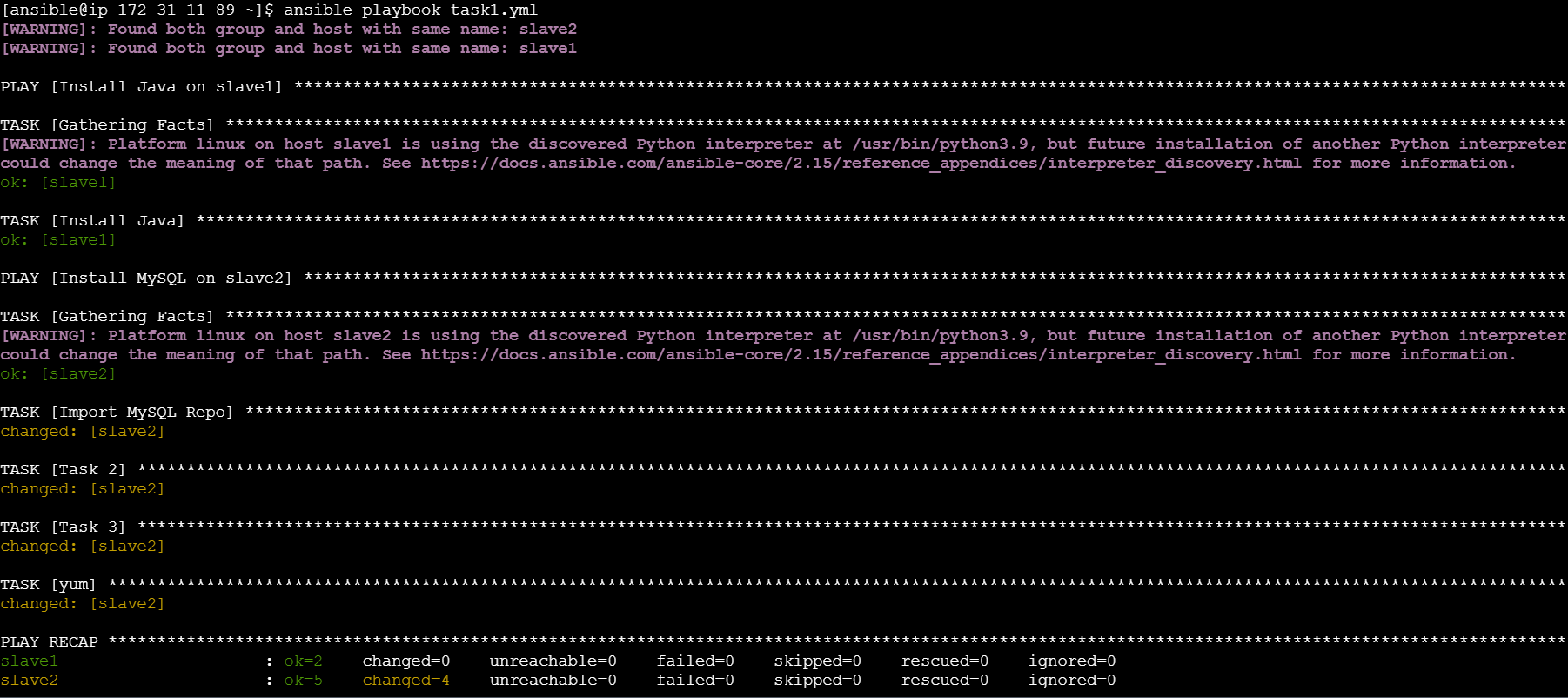
**Step 5: Executing the Ansible Playbook**

The ansible playbook was executed using the below commands:

1. ansible-playbook task1.yml --syntax-check   # code to check for syntax error

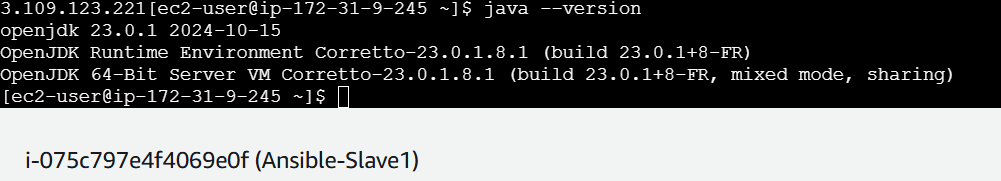
2. ansible-playbook task1.yml --check          # dry run on terminal

3. ansible-playbook task1.yml                  # final run on the server



**Step 6: Checking if Installations have happened in slaves**

Java on Slave 1:



MySQL Server on Slave 2:

