**Ansible-03**

**1.Write a single ansible playbook which will install apache and nginx. Note: Playbook should not be hardcoded and pass the variables from different file.**

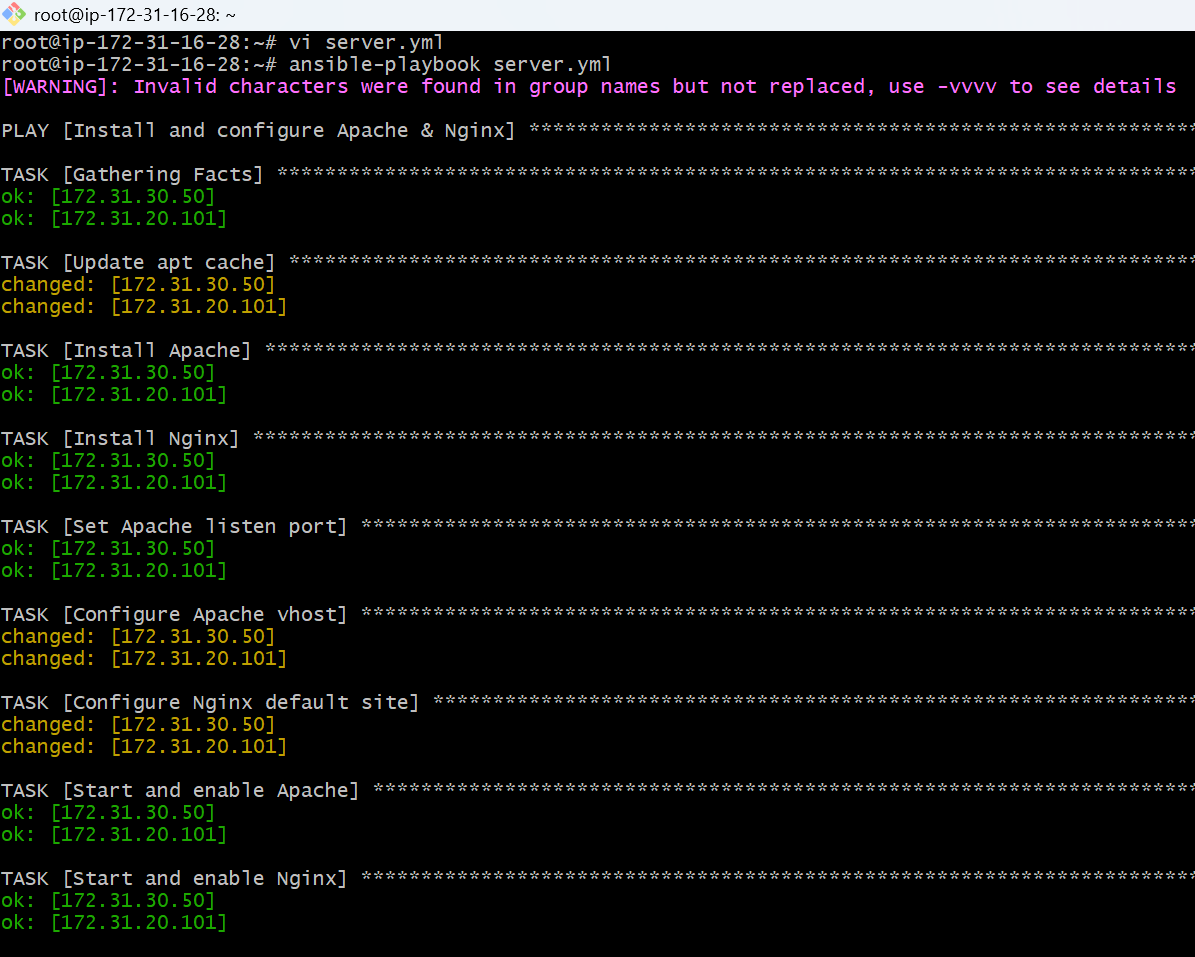
**Create a playbook file and vars.yml file for installing apache and nginx**

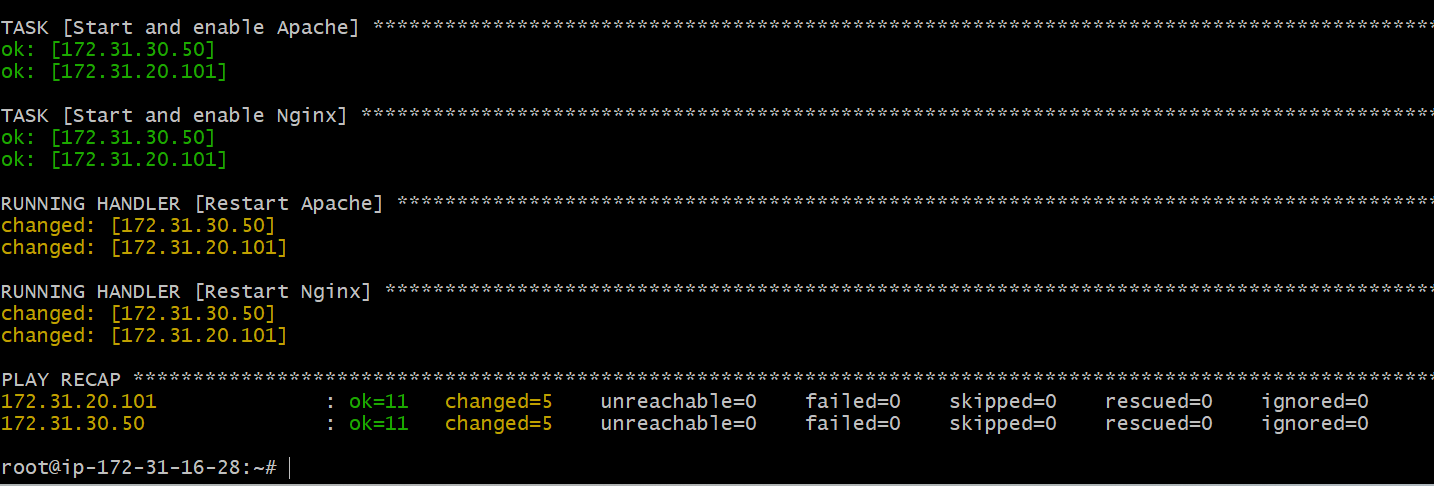
---  
- name: Install and configure Apache & Nginx  
  hosts: all  
  become: yes  
  vars\_files:  
    - vars.yml  tasks:  
    # ------------------------  
    # Update & Install packages  
    # ------------------------  
    - name: Update apt cache  
      apt:  
        update\_cache: yes    - name: Install Apache  
      apt:  
        name: apache2  
        state: present    - name: Install Nginx  
      apt:  
        name: nginx  
        state: present    # ------------------------  
    # Configure Apache  
    # ------------------------  
    - name: Set Apache listen port  
      lineinfile:  
        path: /etc/apache2/ports.conf  
        regexp: '^Listen'  
        line: "Listen {{ apache\_port }}"  
      notify: Restart Apache    - name: Configure Apache vhost  
      copy:  
        dest: /etc/apache2/sites-enabled/000-default.conf  
        content: |  
          <VirtualHost \*:{{ apache\_port }}>  
              DocumentRoot /var/www/html  
              <Directory /var/www/html>  
                  AllowOverride All  
                  Require all granted  
              </Directory>  
          </VirtualHost>  
      notify: Restart Apache    # ------------------------  
    # Configure Nginx  
    # ------------------------  
    - name: Configure Nginx default site  
      copy:  
        dest: /etc/nginx/conf.d/default.conf  
        content: |  
          server {  
              listen {{ nginx\_port }};  
              server\_name \_;  
              root /usr/share/nginx/html;  
              index index.html;  
          }  
      notify: Restart Nginx    # ------------------------  
    # Start and enable services  
    # ------------------------  
    - name: Start and enable Apache  
      systemd:  
        name: apache2  
        state: started  
        enabled: yes    - name: Start and enable Nginx  
      systemd:  
        name: nginx  
        state: started  
        enabled: yes  handlers:  
    - name: Restart Apache  
      service:  
        name: apache2  
        state: restarted    - name: Restart Nginx  
      service:  
        name: nginx  
        state: restarted

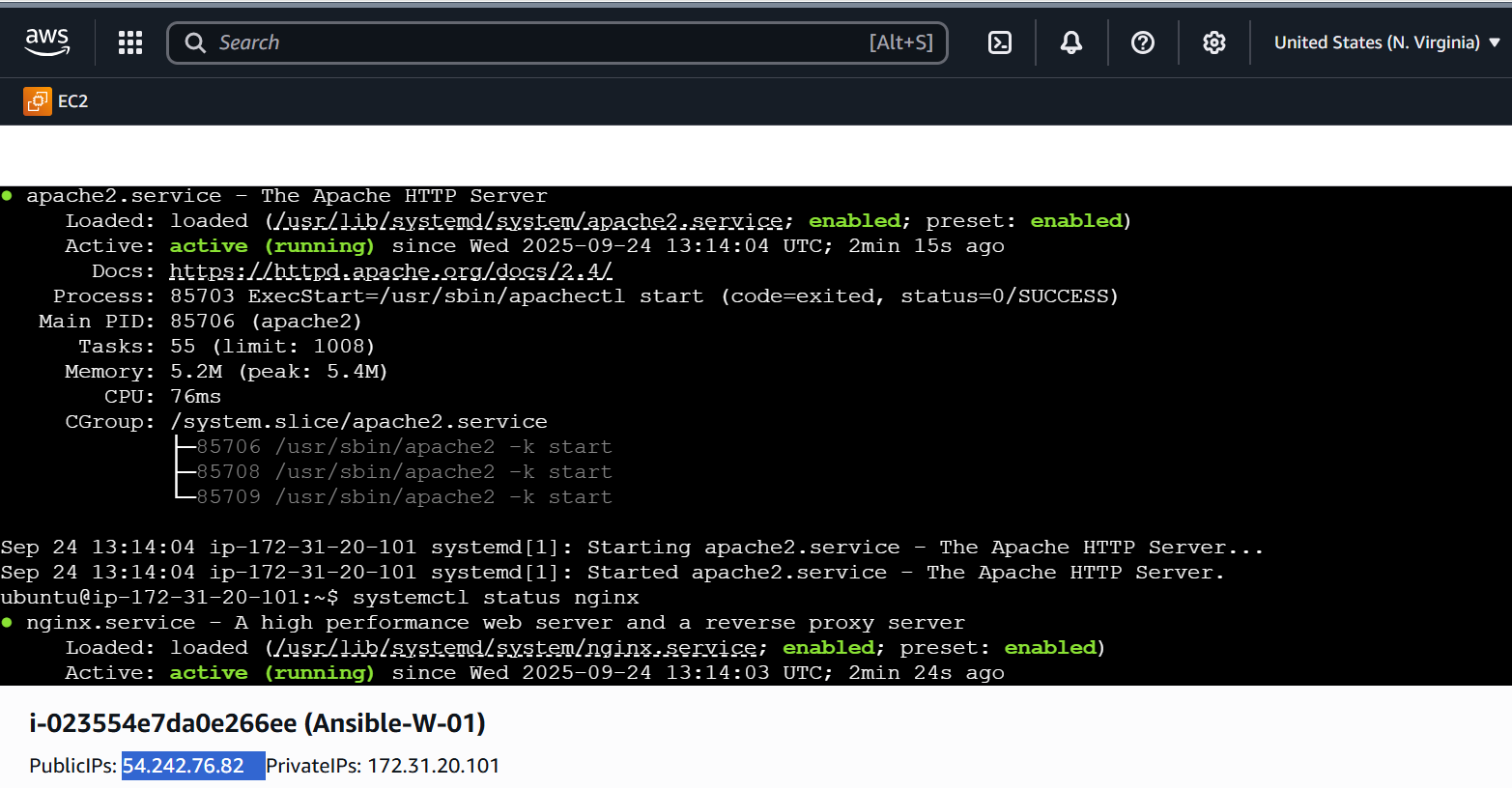
var.yml

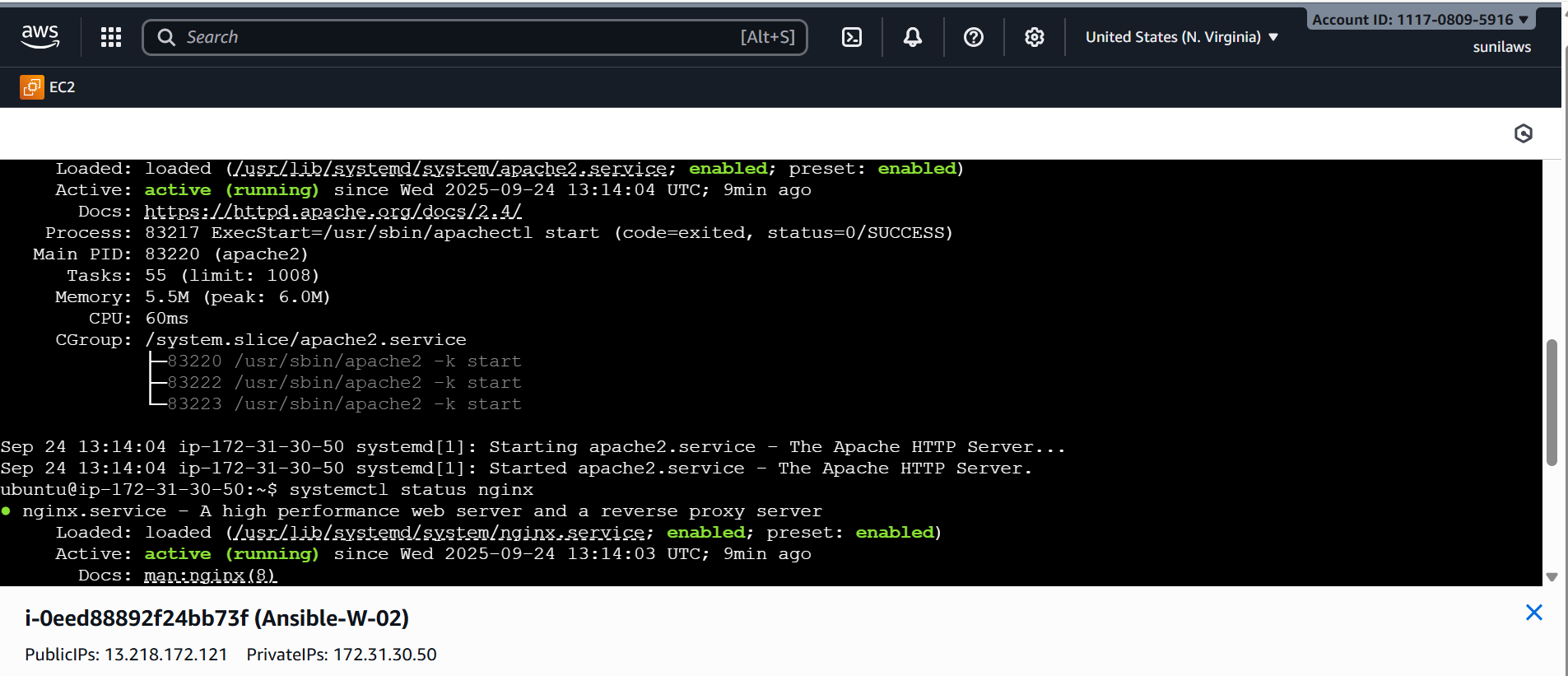
[6:22](https://techiehorizon.slack.com/archives/D0993FPE7B5/p1758718377663639)

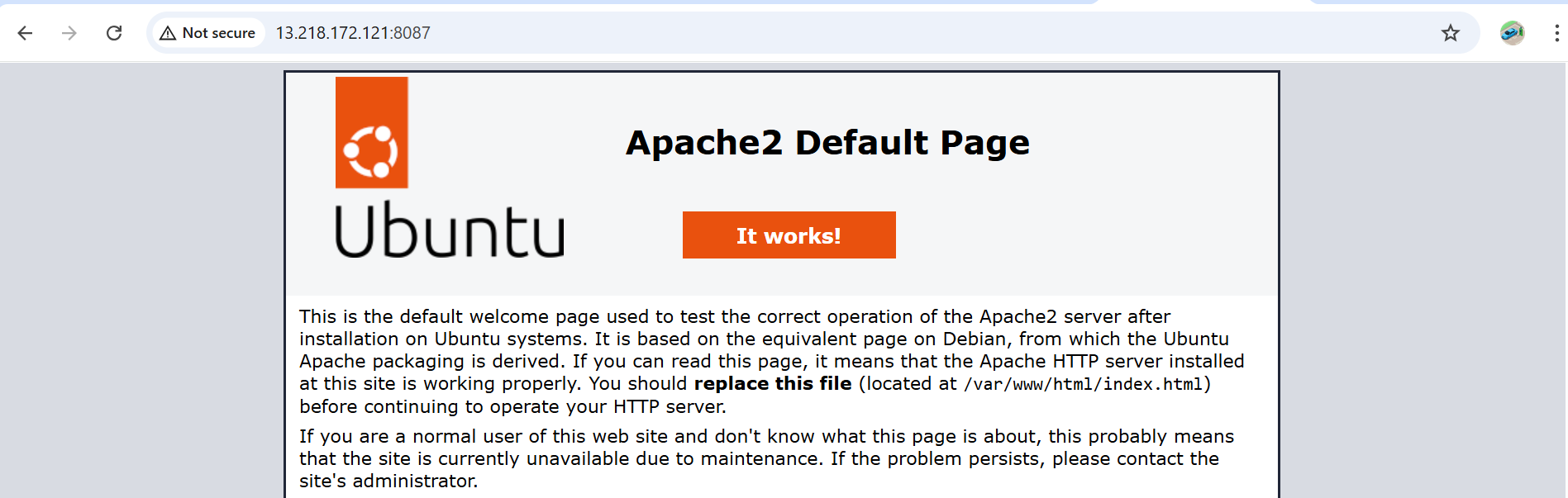
apache\_port: 8087  
nginx\_port: 80

****

****

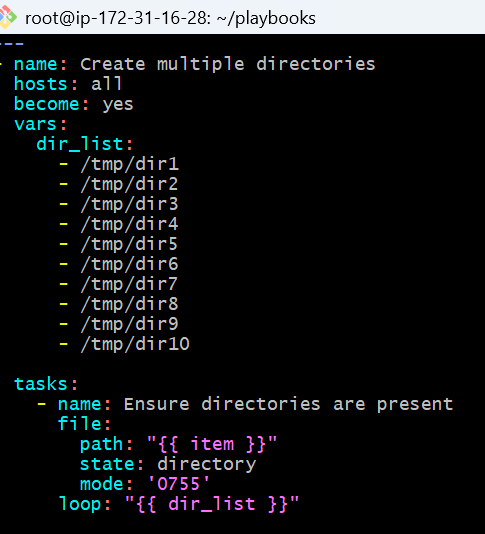
****

****

****

**2.Ansible playbook to create 10 different directories with minimal code and directory names should be passed as variables.**

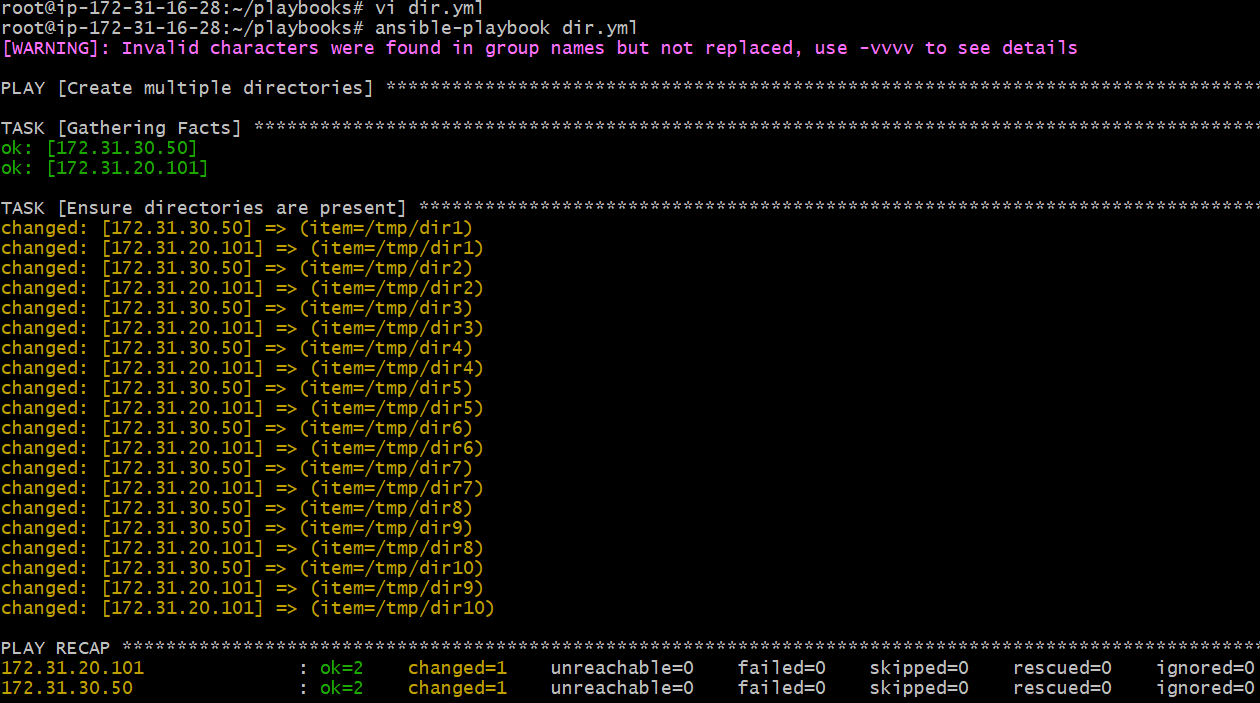
1. Create a Playbook file

****

**Run the Playbook**

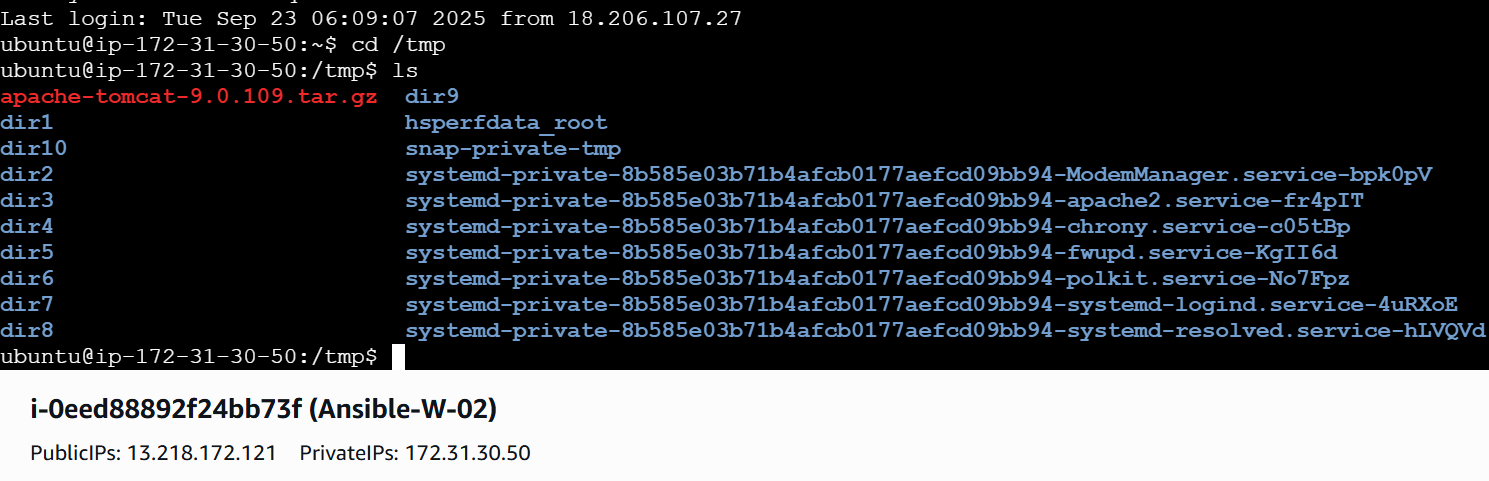
Execute the playbook:

ansible-playbook dir.yml

****

### ****Verify Directories****

Check if the directories are created:

****

**3.Ansible playbook to copy ssh-keygen from master to worker nodes. Note: a)Provision new 3 ec2 machines, one master and two worker nodes.**

## Step 1: Install AWS Collections

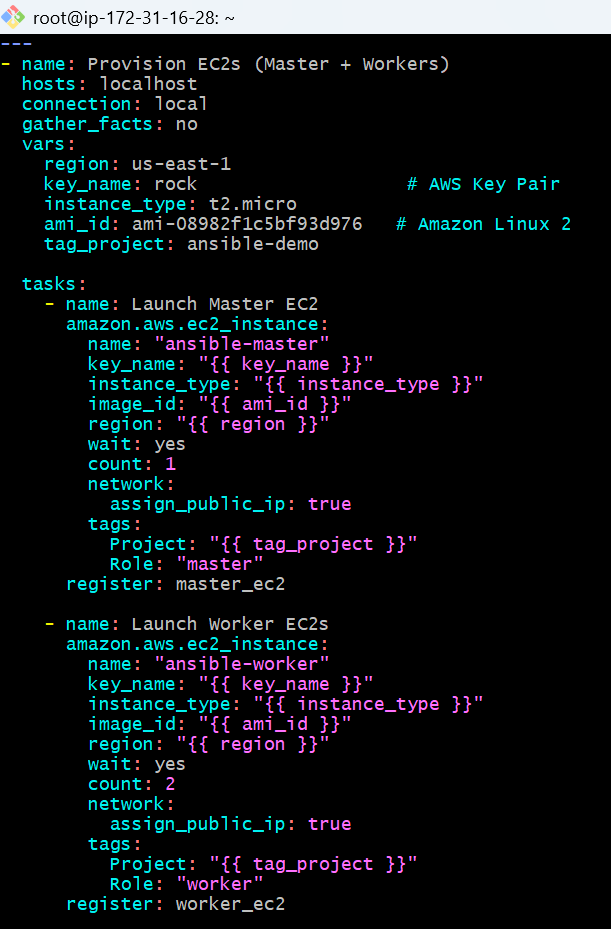
On your Ansible controller:

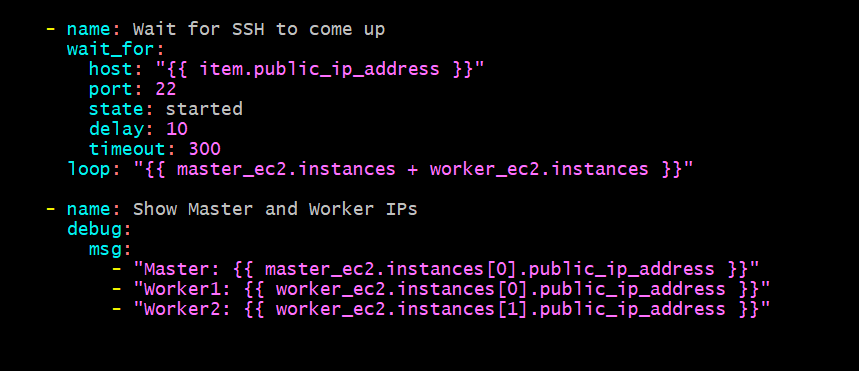
ansible-galaxy collection install amazon.aws

pip install boto3 botocore

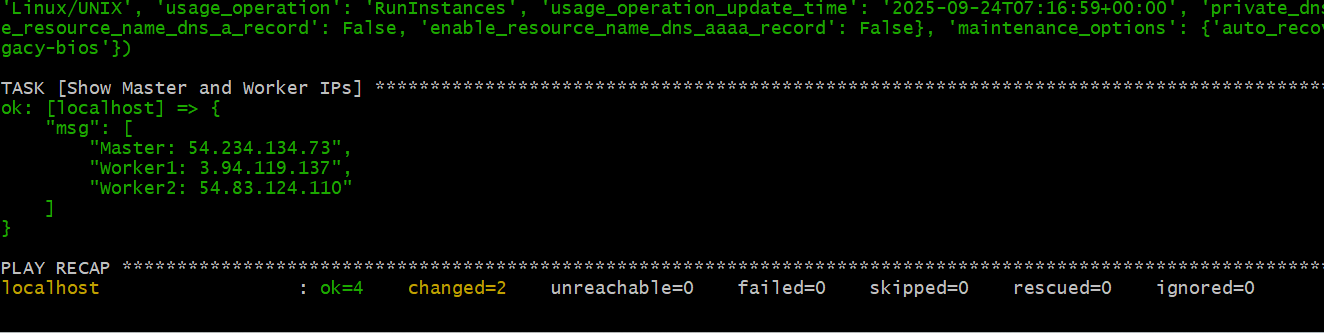
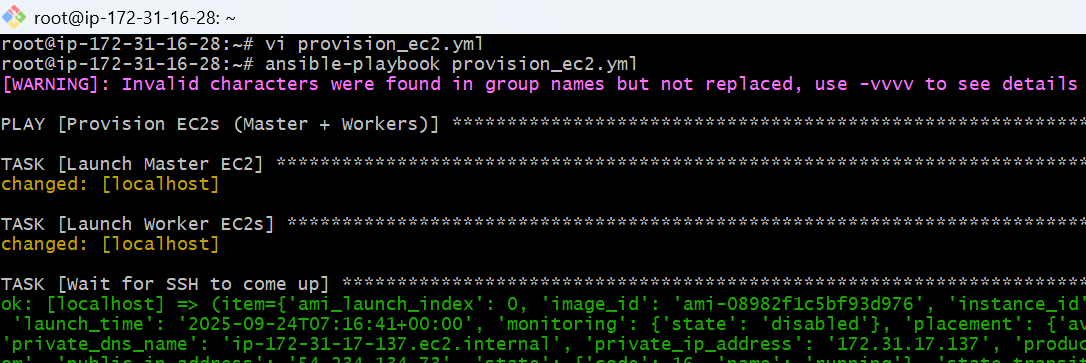
Step 2: Create Ansible Playbook to Provision 3 EC2

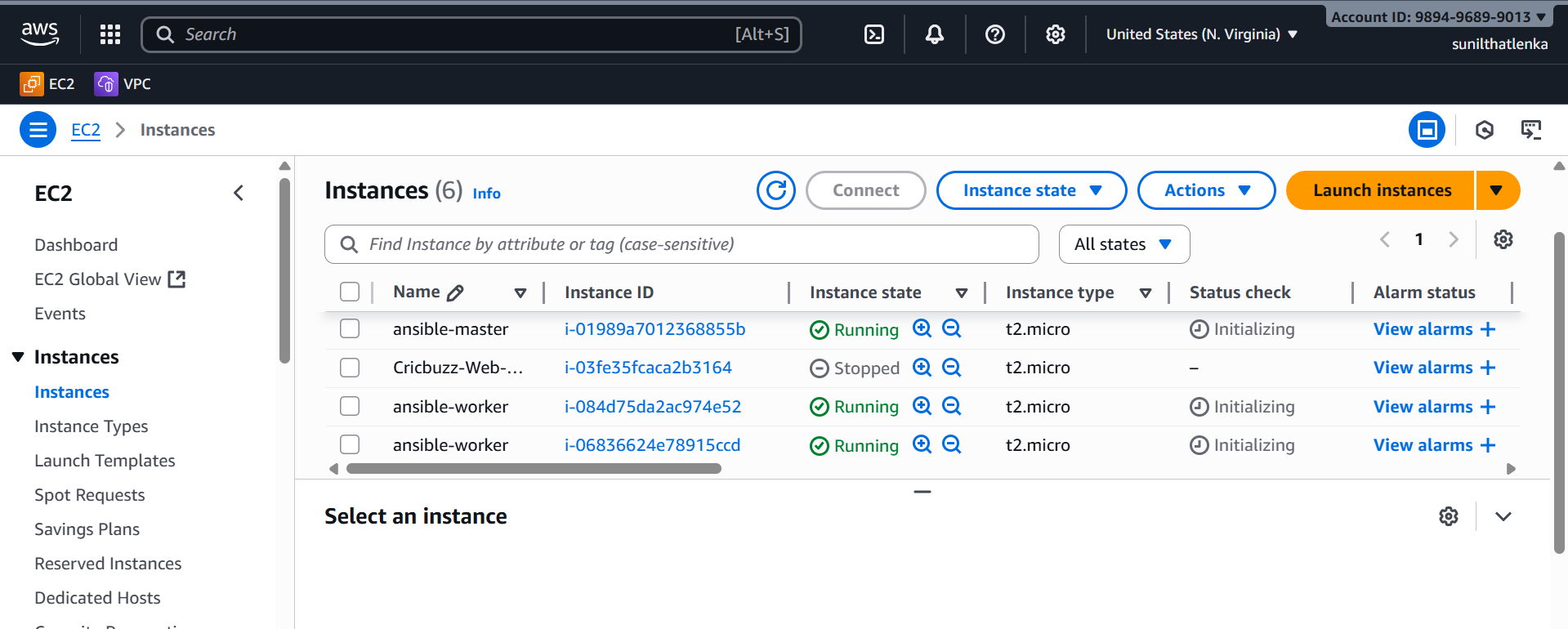
Provision **3 EC2s** → (1 Master + 2 Workers).

****

****

**Run ansible-playbook provision\_ec2.yml**

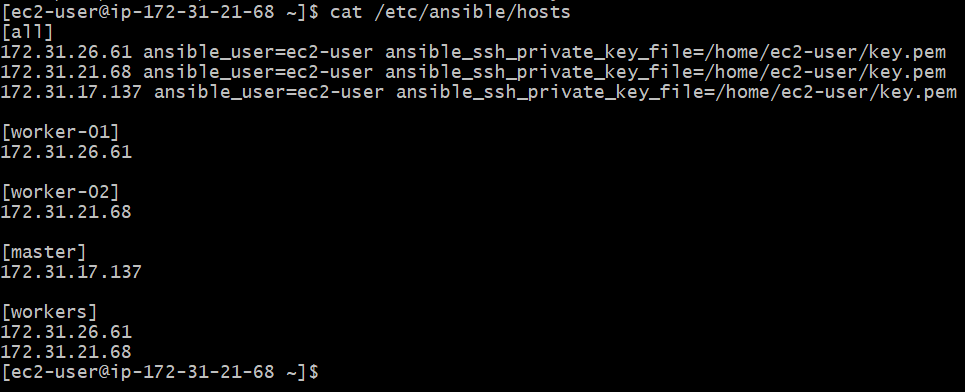
****

****

**b)Create common user called ansadm and provide sudo priviliges on 3 ec2 instances.**

**copy your pem key to the controller node and give chmod400**

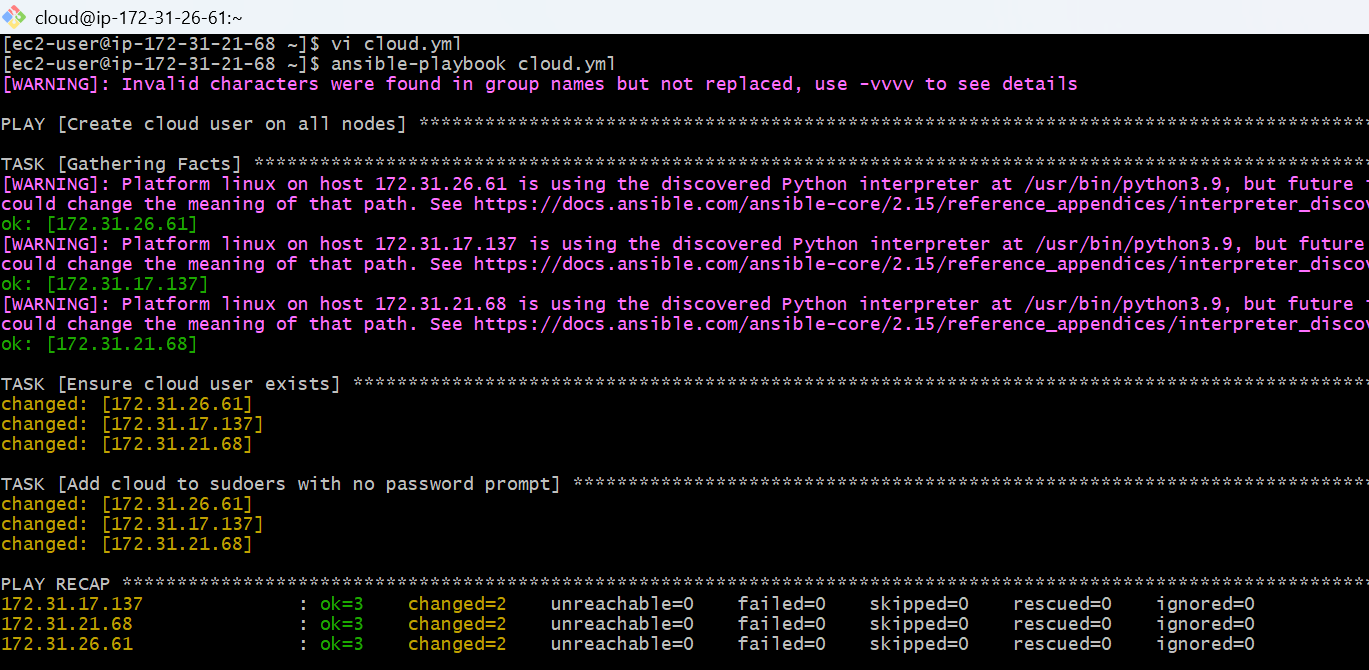
**after that give the path of the key in your host file**

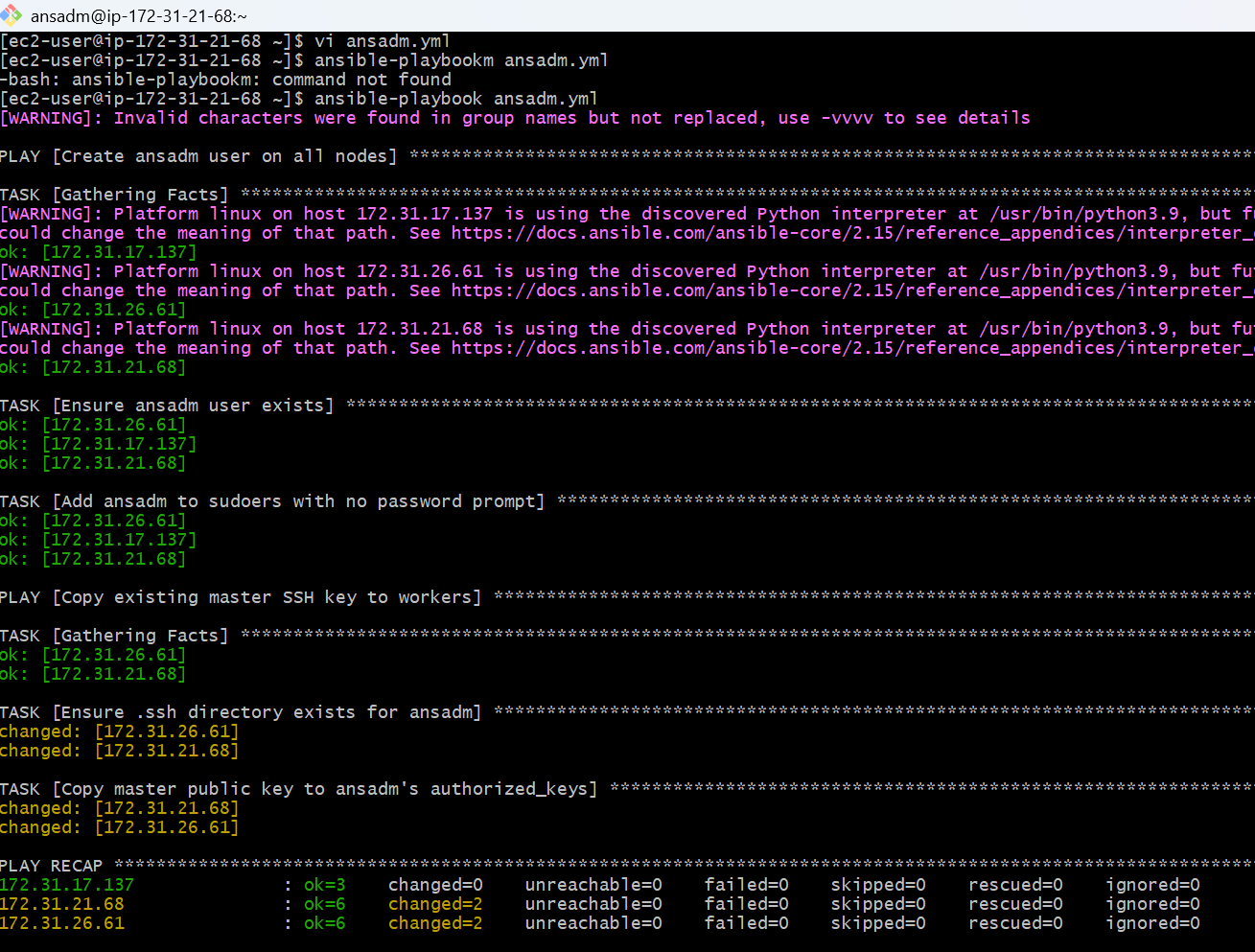
****

Ansible needs a way to SSH into each node.

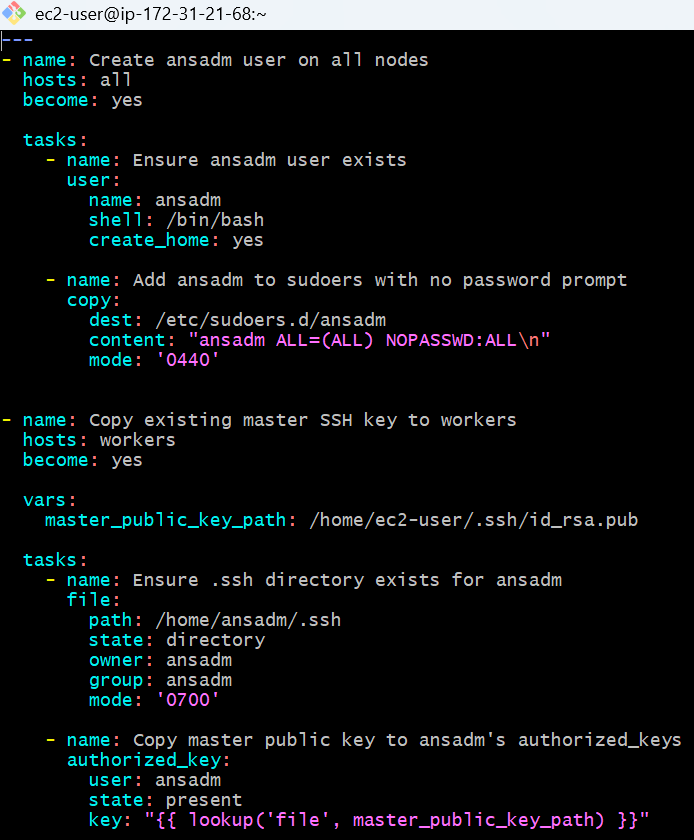
Since AWS EC2 instances usually only allow key-based login, you provide the **private key path** on the machine where you run Ansible (usually the master).

**Create a ansadm.yml playbook and run I have created 2 users ansadm and cloud(optional)**

****

****

Step 1: Create a common user (ansadm) on all nodes

****

**Process explanation:**

1. **Target all nodes**: hosts: all means the playbook runs on **master + worker nodes**.
2. **Use sudo privileges**: become: yes allows creating users and modifying system files.
3. **Create the user**:.

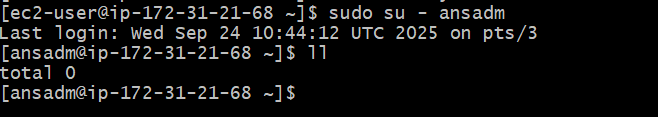
At the end of this step, all nodes have the same ansadm user.

## ****Give**** ansadm ****sudo privileges****

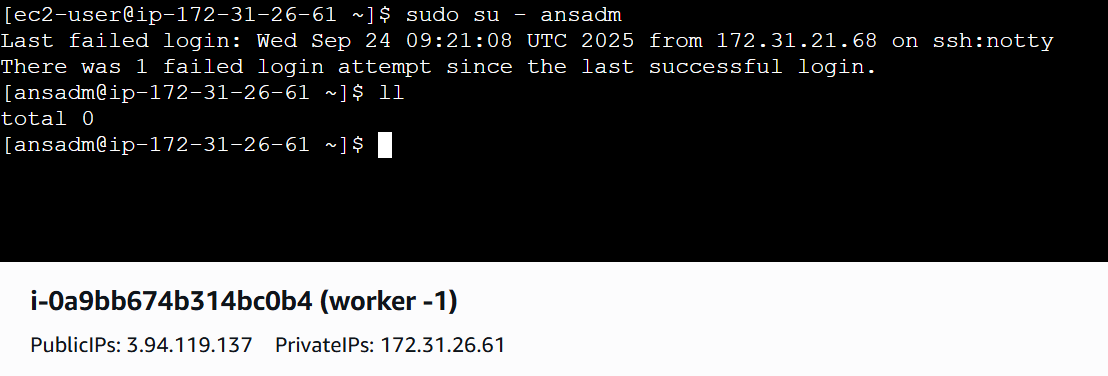
1. Creates a file /etc/sudoers.d/ansadm on each node.
2. Adds the line:
3. ansadm ALL=(ALL) NOPASSWD:ALL
   * This allows ansadm to run **any command with sudo** **without entering a password**.
4. mode: '0440' sets proper file permissions for security (only root can read).

At the end of this step, ansadm can run administrative commands on any node.

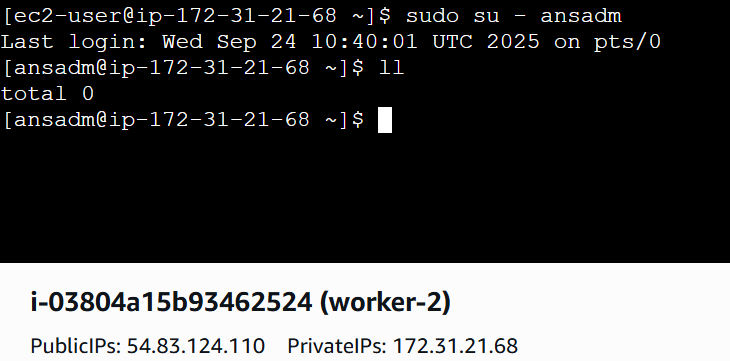
Sudo su – ansadm on master node

****

**Ansadm on worker-01**

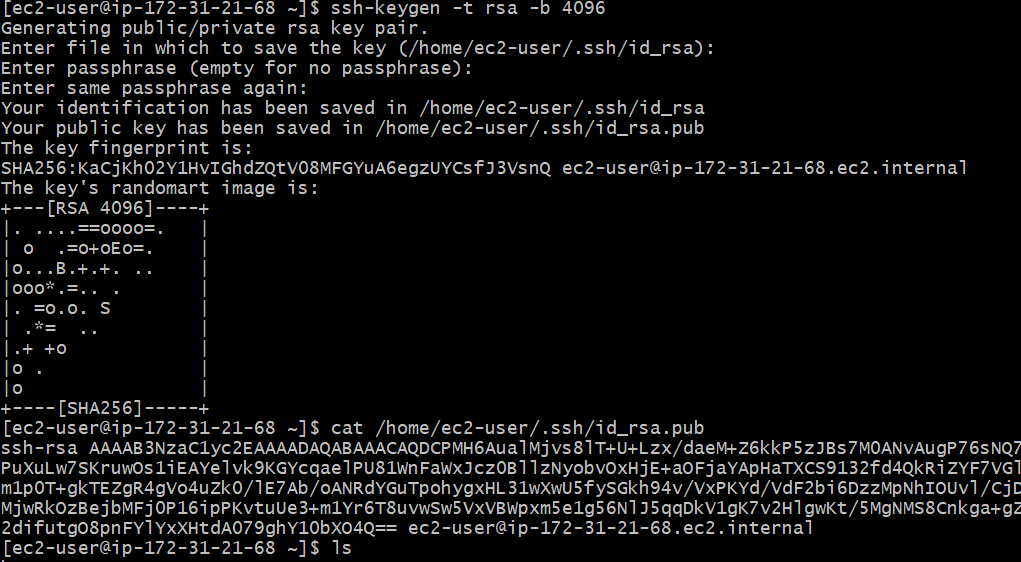
****

**Ansadm on worker-02**

****

**c)Create ssh-keygen in master and your playbook should copy the keygen making it password less authentication.**

In controller create public key  
ssh-keygen -t rsa -b 4096

****

Create a playbook and run the playbook

If the SSH key pair **already exists** for ec2-user on the master, the playbook **will not generate a new one**.

 **Master node:**

* Generate SSH key pair (id\_rsa + id\_rsa.pub) for ec2-user.

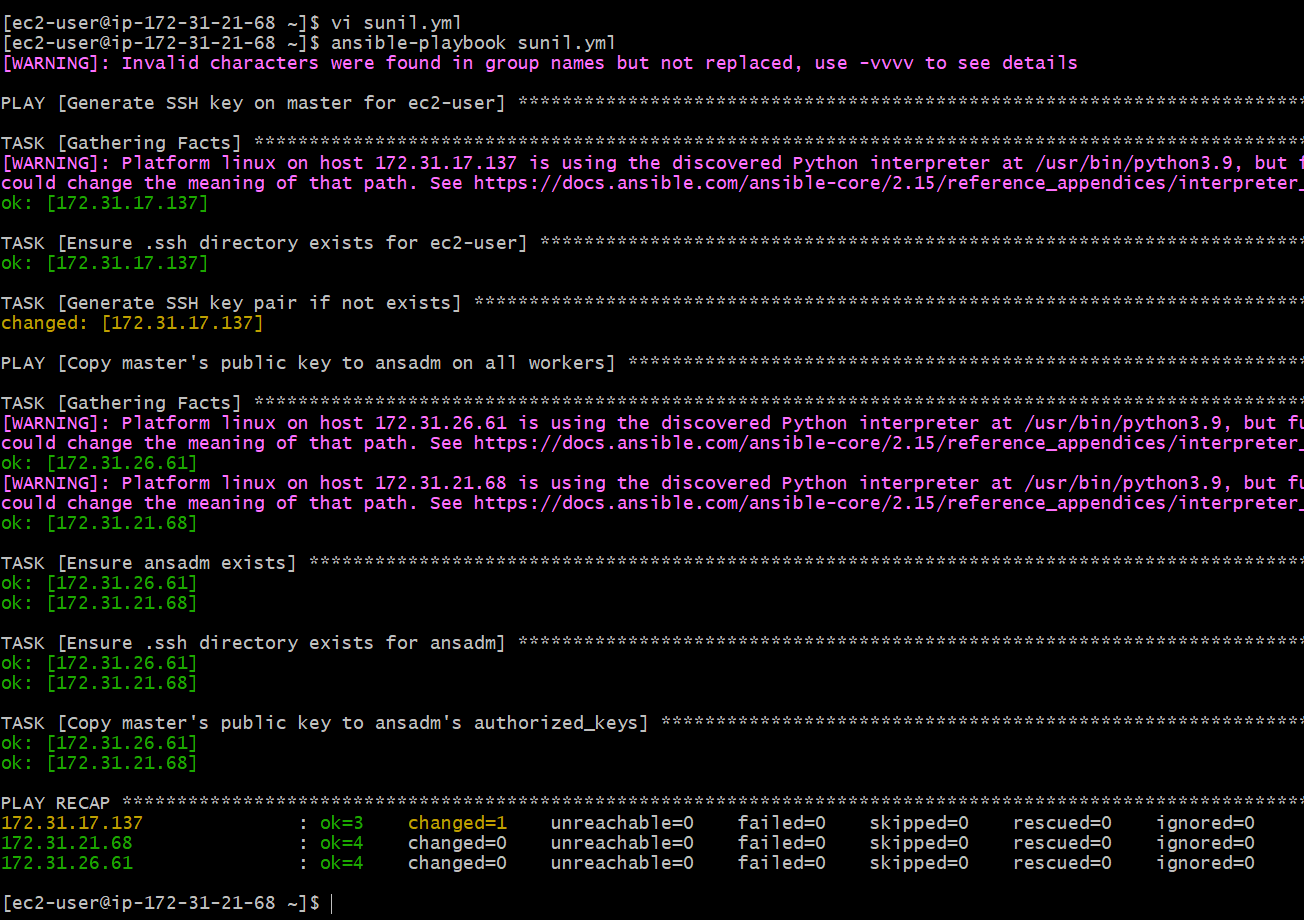
 **Worker nodes:**

* Create user ansadm if it doesn’t exist.
* Create /home/ansadm/.ssh folder with correct permissions.
* Copy the **master’s public key** to /home/ansadm/.ssh/authorized\_keys.

 **Result:**

* Master’s ansadm (or ec2-user) can SSH into worker nodes as ansadm **without a password**.
* All permissions and directories are correctly set.

---  
- name: Generate SSH key on master for ec2-user  
  hosts: master  
  become: yes  
  vars:  
    ssh\_key\_path: /home/ec2-user/.ssh/id\_rsa  
  tasks:  
    - name: Ensure .ssh directory exists for ec2-user  
      file:  
        path: /home/ec2-user/.ssh  
        state: directory  
        owner: ec2-user  
        group: ec2-user  
        mode: '0700'    - name: Generate SSH key pair if not exists  
      openssh\_keypair:  
        path: "{{ ssh\_key\_path }}"  
        type: rsa  
        size: 4096  
        state: present  
        owner: ec2-user  
        group: ec2-user  
        mode: '0600'- name: Copy master's public key to ansadm on all workers  
  hosts: workers  
  become: yes  
  vars:  
    master\_public\_key\_path: /home/ec2-user/.ssh/id\_rsa.pub  
  tasks:  
    - name: Ensure ansadm exists  
      user:  
        name: ansadm  
        shell: /bin/bash  
        create\_home: yes    - name: Ensure .ssh directory exists for ansadm  
      file:  
        path: /home/ansadm/.ssh  
        state: directory  
        owner: ansadm  
        group: ansadm  
        mode: '0700'    - name: Copy master's public key to ansadm's authorized\_keys  
      authorized\_key:  
        user: ansadm  
        state: present  
        key: "{{ lookup('file', master\_public\_key\_path) }}"

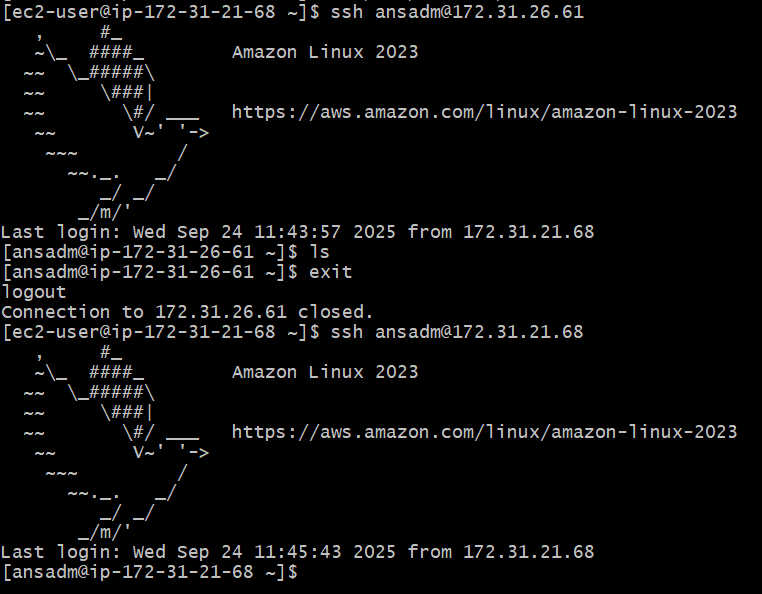
****

After running the playbook:

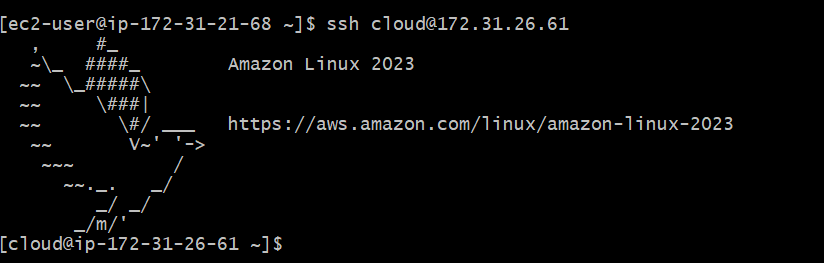
* Master can do:

ssh ansadm@172.31.26.61 # worker-01

ssh ansadm@172.31.21.68 # worker-02

****

**Usrer cloud**

****

**4.Ansible playbook to inject ansible vault variables.**

### ****Step 1: Create Ansible Vault File****

Create a file to store sensitive variables, e.g., vault.yml:

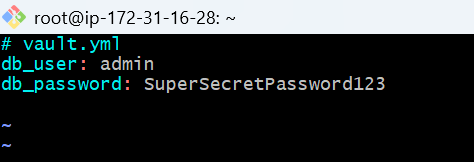
ansible-vault create vault.yml

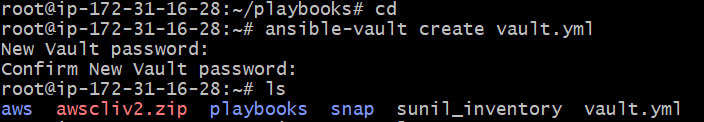
Add content like:

# vault.yml

db\_user: admin

db\_password: SuperSecretPassword123

****



You can also edit later using:

ansible-vault edit vault.yml

### ****Step 2: Create Playbook Using Vault Variables****

Create a playbook secret.yml:

---

- name: Example Playbook Using Vault Variables

hosts: localhost

vars\_files:

- vault.yml # inject vault variables here

tasks:

- name: Show DB username

debug:

msg: "The DB username is {{ db\_user }}"

- name: Show DB password

debug:

msg: "The DB password is {{ db\_password }}"

### ****Step 3: Run the Playbook****

Run it with the vault password:

ansible-playbook secret.yml --ask-vault-pass

enter the vault password

