Ansible Shutdown playbook :

Shutdown.yaml

```
- hosts: all
  become: true

  tasks:
    - name: Shutdown ubuntu instances only
      ansible.builtin.command: /sbin/shutdown -t now
      when:
       ansible_facts['os_family'] == "Debian"
```

```
$ ansible-playbook -i inventory.ini shutdown.yaml
```

Working with AWS from Ansible

Working with AWS from Ansible involves using Ansible's Amazon AWS collection, which includes numerous modules designed for automating management and provisioning of AWS resources. Ansible can automate infrastructure tasks such as creating EC2 instances, managing VPCs, subnets, security groups, IAM roles, and more through idempotent, declarative playbooks written in YAML.

Ansible ec2 creation

Prerequisites

- Install the Ansible Amazon AWS collection:
  ansible-galaxy collection install amazon.aws
- Ensure Python >= 3.6, boto3 >= 1.28.0, and botocore >= 1.31.0 are installed on the host executing the playbook.
- Set AWS authentication credentials using environment variables, AWS CLI profiles, or Ansible Vault to keep secrets secure.

Check Python Version

Run this command to check your Python version:

```
python3 --version
```

Ensure the version is 3.6 or newer.

Install or Upgrade Python

- On Ubuntu/Debian:

```
sudo apt update
sudo apt install python3 python3-pip
```

Check Python Version

Run this command to check your Python version:

```
python3 --version
```

Ensure the version is 3.6 or newer.

Install or upgrade boto3 and botocore

Use pip to install or upgrade boto3 and botocore:

```
sudo apt install python3-boto3
```

Verify Installed Versions

To check the versions of boto3 and botocore:

```
python3 -c "import boto3; print(boto3.__version__)"
python3 -c "import botocore; print(botocore.__version__)"
```

Make sure boto3 version is at least 1.28.0 and botocore version at least 1.31.0.

Example Playbook to Create EC2 Instance

ec2_playbook.yaml

```yaml
- name: Create EC2 and AWS resources
  hosts: localhost
  gather_facts: no
  vars:
    region: us-east-1
    instance_type: t2.micro
    ami_id: ami-0f918f7e67a3323f0  # Example AMI ID
    key_name: yogeshpem         # Your AWS EC2 key pair name
```

```
  vpc_cidr_block: 10.0.0.0/16
  subnet_cidr_block: 10.0.1.0/24
tasks:
  - name: Create VPC
    amazon.aws.ec2_vpc_net:
      name: MyVPC
      cidr_block: "{{ vpc_cidr_block }}"
      region: "{{ region }}"
    register: vpc

  - name: Create Subnet
    amazon.aws.ec2_vpc_subnet:
      vpc_id: "{{ vpc.vpc.id }}"
      cidr: "{{ subnet_cidr_block }}"
      region: "{{ region }}"
    register: subnet

  - name: Launch EC2 Instance
    amazon.aws.ec2_instance:
      name: "MyInstance"
      instance_type: "{{ instance_type }}"
      region: "{{ region }}"
      image_id: "{{ ami_id }}"
      key_name: "{{ key_name }}"
      subnet_id: "{{ subnet.subnet.id }}"
      wait: yes
      network :
        assign_public_ip: yes
      tags:
        Environment: Testing
    register: ec2

  - name: Output instance ID
    debug:
```

```
    msg: "Instance ID is {{ ec2.instances[0].instance_id }}"
```

Run ansible-playbook command..

```
$ ansible-playbook ec2_playbook.yaml
```

Check on AWS Ec2 a new ubuntu instance is created.

Variables in Ansible

Adding variables in Ansible

1. Create role for ec2

```
$ ansible-galaxy role init ec2
```

2. Go into ansible ec2/defaults

```
$ cd ec2/defaults/
$ vi main.yaml
```

```
region: ap-south-1
instance_type: t2.micro
ami_id: ami-0861f4e788f5069dd
key_name: yogeshpem
vpc_cidr_block: 10.0.0.0/16
subnet_cidr_block: 10.0.1.0/24
```

3. Now, update ec2/tasks/main.yml file to include variables

```
---
- name: Create VPC
  amazon.aws.ec2_vpc_net:
    name: MyVPC
    cidr_block: "{{ vpc_cidr_block }}"
    region: "{{ region }}"
  register: vpc

- name: Create Subnet
  amazon.aws.ec2_vpc_subnet:
    vpc_id: "{{ vpc.vpc.id }}"
    cidr: "{{ subnet_cidr_block }}"
    region: "{{ region }}"
```

```
  register: subnet

- name: Launch EC2 Instance
  amazon.aws.ec2_instance:
    name: "MyInstance"
    instance_type: "{{ instance_type }}"
    region: "{{ region }}"
    image_id: "{{ ami_id }}"
    key_name: "{{ key_name }}"
    subnet_id: "{{ subnet.subnet.id }}"
    wait: yes
    network:
      assign_public_ip: yes
    tags:
      Environment: Testing
  register: ec2

- name: Output instance ID
  debug:
    msg: "Instance ID is {{ ec2.instances[0].instance_id }}"
```

4. How to use this role in a playbook

```
$ vi ec2_role.yaml
```

```
- name: Create EC2 and AWS resources using role
  hosts: localhost
  gather_facts: no
  roles:
    - ec2
```

5. Run playbook in a playbook

```
$ ansible-playbook ec2_role.yaml
```