Demo Project: Data Backup and Restore

This project demonstrates how to automate data backup, cleanup, and restore for EC2 volumes using Python and AWS Boto3 library.

Step 1. Setup EC2 Instances

1. Create two EC2 instances:
    - **Name:** dev
    - **Name:** prod
2. Assign the following tag to each instance:
    - Key: `Name`
    - Value: `dev` or `prod`

Step 2: Install Required Libraries

- `pip install boto3 schedule`

Step 3: Automating EC2 Volume Snapshots

Create File: `volume-backups.py`

```
import boto3
import schedule

ec2_client = boto3.client('ec2', region_name="us-east-1")


def create_volume_snapshots():
    volumes = ec2_client.describe_volumes(
        Filters=[
            {
                'Name': 'tag:Name',
                'Values': ['prod']
            }
        ]
    )
    for volume in volumes['Volumes']:
        new_snapshot = ec2_client.create_snapshot(
            VolumeId=volume['VolumeId']
        )
        print(new_snapshot)


schedule.every().day.do(create_volume_snapshots)

while True:
    schedule.run_pending()
```

Step 4: Cleaning Up Old Snapshots

Create File: `cleanup-snapshots.py`

```
import boto3
from operator import itemgetter
```

```
ec2_client = boto3.client('ec2', region_name="us-east-1")

volumes = ec2_client.describe_volumes(
    Filters=[
        {
            'Name': 'tag:Name',
            'Values': ['prod']
        }
    ]
)

for volume in volumes['Volumes']:
    snapshots = ec2_client.describe_snapshots(
        OwnerIds=['self'],
        Filters=[
            {
                'Name': 'volume-id',
                'Values': [volume['VolumeId']]
            }
        ]
    )

    sorted_by_date = sorted(snapshots['Snapshots'], key=itemgetter('StartTim
e'), reverse=True)

    for snap in sorted_by_date[2:]:
        response = ec2_client.delete_snapshot(
            SnapshotId=snap['SnapshotId']
        )
        print(response)
```

Step 5: Restoring EC2 Volumes

Create File: `restore-volume.py`

```
import boto3
from operator import itemgetter

ec2_client = boto3.client('ec2', region_name="us-east-1")
ec2_resource = boto3.resource('ec2', region_name="us-east-1")

instance_id = "i-0671d0fe02906a969"

volumes = ec2_client.describe_volumes(
   Filters=[
      {
         'Name': 'attachment.instance-id',
         'Values': [instance_id]
      }
   ]
)

instance_volume = volumes['Volumes'][0]

snapshots = ec2_client.describe_snapshots(
   OwnerIds=['self'],
   Filters=[
      {
         'Name': 'volume-id',
         'Values': [instance_volume['VolumeId']]
      }
   ]
)

latest_snapshot = sorted(snapshots['Snapshots'], key=itemgetter('StartTime'),
reverse=True)[0]
print(latest_snapshot['StartTime'])

new_volume = ec2_client.create_volume(
   SnapshotId=latest_snapshot['SnapshotId'],
```

```
    AvailabilityZone="us-east-1a",
    TagSpecifications=[
        {
            'ResourceType': 'volume',
            'Tags': [
                {
                    'Key': 'Name',
                    'Value': 'prod'
                }
            ]
        }
    ]
)

while True:
    vol = ec2_resource.Volume(new_volume['VolumeId'])
    print(vol.state)
    if vol.state == 'available':
        ec2_resource.Instance(instance_id).attach_volume(
            VolumeId=new_volume['VolumeId'],
            Device='/dev/xvdb'
        )
        break
```

Step 6: Testing the Project

1. Run `volume-backups.py` to create snapshots.
2. Run `cleanup-snapshots.py` to delete old snapshots.
3. Run `restore-volume.py` to restore volumes from the latest snapshot.