

SCENARIO:-

Lab Prerequisites:

To successfully complete this lab, you should be familiar with basic Amazon EC2 usage and with basic Linux server administration. You should feel comfortable using the Linux command-line tools.

Background:

This experiment focuses on Amazon Elastic Block Store (Amazon EBS), a key underlying storage mechanism for Amazon EC2 instances. In this lab, you will learn how to create an Amazon EBS volume, attach it to an instance, apply a file system to the volume, and then take a snapshot backup.

Problem Scenario:

Steven is a Cloud Administrator of GotArray Techno Pvt Ltd, Their majority of clients are e-commerce and OTP service providers. Initially, they wanted to set up one virtual windows/Linux server using Amazon EC2 which can be resizable and provide compute capacity along with a web-scale cloud computing solution. After 6-month Steven and his team realized that they needed more cloud storage with an elastic, high performance block storage service. Design for EC2 machine due you to increase web traffic on their e-commerce client. Set the storage of 60 GB and attach them to the EC2 server and maintain all operations using logical volume management.

AIM:-

Create Terraform Configuration that can create EC2 instances, EBS volume (Customized Size) and attach it together.

Steps:-

1. Configuring terraform script.

```
- CODE:- (main.tf)
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 3.0"
    }
  }
  required_version = ">= 0.13.5"
}
# Configure the AWS Provider
provider "aws" {
  region      = "us-east-1"
  access_key  = "YOUR_ACCESS_KEY"
  secret_key  = "YOUR_SECRET_KEY"
}

# Create a Instance
resource "aws_instance" "vm" {
  ami          = "ami-0cff7528ff583bf9a"
  instance_type = "t2.micro"
  key_name     = "MyKey"
```

```
tags = {
  Name = "AMI Linux 2 ~ EBS storage"
}

resource "aws_ebs_volume" "ebs_volume" {
  availability_zone = aws_instance.vm.availability_zone
  size              = 10
  type              = "gp2"
  tags = {
    Name = "ebs-volume-e4"
  }
}

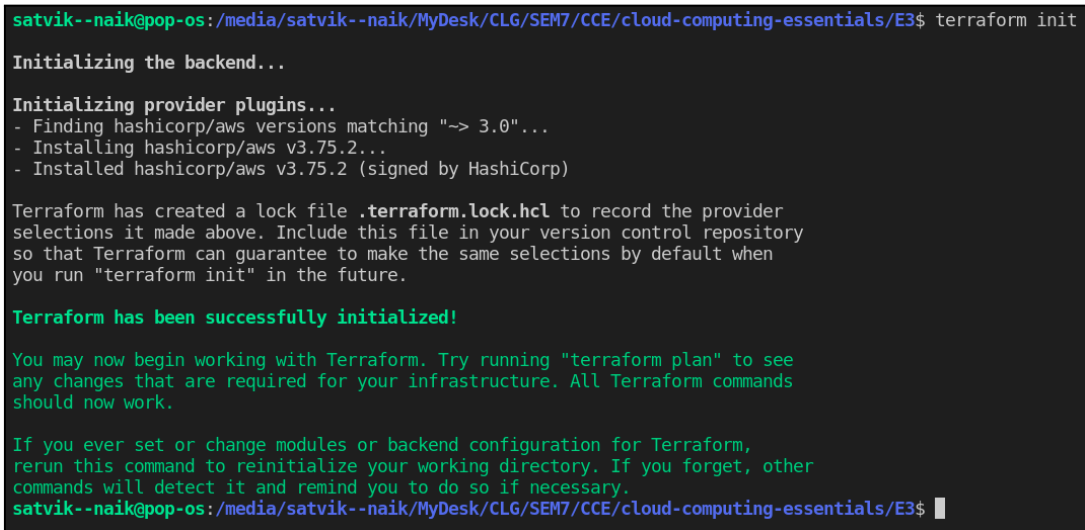
resource "aws_volume_attachment" "ebs_volume_attachment" {
  device_name = "/dev/xvdf"
  volume_id   = aws_ebs_volume.ebs_volume.id
  instance_id = aws_instance.vm.id
}

output "EbsVolumeId" {
  value = aws_ebs_volume.ebs_volume.id
}

output "InstanceId" {
  value = aws_instance.vm.id
}
```

2. Launch terraform script.

- terraform init



```
satvik--naik@pop-os: /media/satvik--naik/MyDesk/CLG/SEM7/CCE/cloud-computing-essentials/E3$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 3.0"...
- Installing hashicorp/aws v3.75.2...
- Installed hashicorp/aws v3.75.2 (signed by HashiCorp)

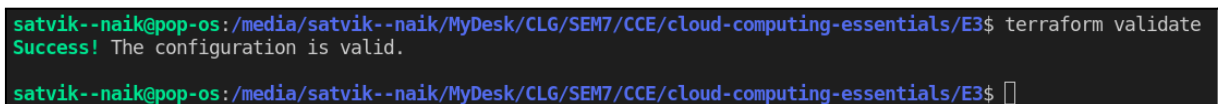
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
satvik--naik@pop-os: /media/satvik--naik/MyDesk/CLG/SEM7/CCE/cloud-computing-essentials/E3$
```

- terraform validate



```
satvik--naik@pop-os: /media/satvik--naik/MyDesk/CLG/SEM7/CCE/cloud-computing-essentials/E3$ terraform validate
Success! The configuration is valid.

satvik--naik@pop-os: /media/satvik--naik/MyDesk/CLG/SEM7/CCE/cloud-computing-essentials/E3$
```

- terraform plan

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER GITLENS
+ root_block_device {
+   delete_on_termination = (known after apply)
+   device_name            = (known after apply)
+   encrypted              = (known after apply)
+   iops                   = (known after apply)
+   kms_key_id             = (known after apply)
+   tags                   = (known after apply)
+   throughput             = (known after apply)
+   volume_id              = (known after apply)
+   volume_size            = (known after apply)
+   volume_type            = (known after apply)
+ }
}

# aws_volume_attachment.ebs_volume_attachment will be created
+ resource "aws_volume_attachment" "ebs_volume_attachment" {
+   device_name = "/dev/xvdf"
+   id          = (known after apply)
+   instance_id = (known after apply)
+   volume_id   = (known after apply)
+ }

Plan: 3 to add, 0 to change, 0 to destroy.

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run
"terraform apply" now.
satvik--naik@pop-os: /media/satvik--naik/MyDesk/CLG/SEM7/CCE/cloud-computing-essentials/E3$
```

- terraform apply

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER GITLENS

Plan: 3 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ EbsVolumeId = (known after apply)
+ InstanceId  = (known after apply)
aws_instance.vm: Creating...
aws_instance.vm: Still creating... [10s elapsed]
aws_instance.vm: Still creating... [20s elapsed]
aws_instance.vm: Still creating... [30s elapsed]
aws_instance.vm: Still creating... [40s elapsed]
aws_instance.vm: Still creating... [50s elapsed]
aws_instance.vm: Creation complete after 56s [id=i-073f637e8bb3dafbc]
aws_ebs_volume.ebs_volume: Creating...
aws_ebs_volume.ebs_volume: Still creating... [10s elapsed]
aws_ebs_volume.ebs_volume: Creation complete after 13s [id=vol-0235016188cc92a94]
aws_volume_attachment.ebs_volume_attachment: Creating...
aws_volume_attachment.ebs_volume_attachment: Still creating... [10s elapsed]
aws_volume_attachment.ebs_volume_attachment: Still creating... [20s elapsed]
aws_volume_attachment.ebs_volume_attachment: Still creating... [30s elapsed]
aws_volume_attachment.ebs_volume_attachment: Creation complete after 37s [id=vai-2663629394]

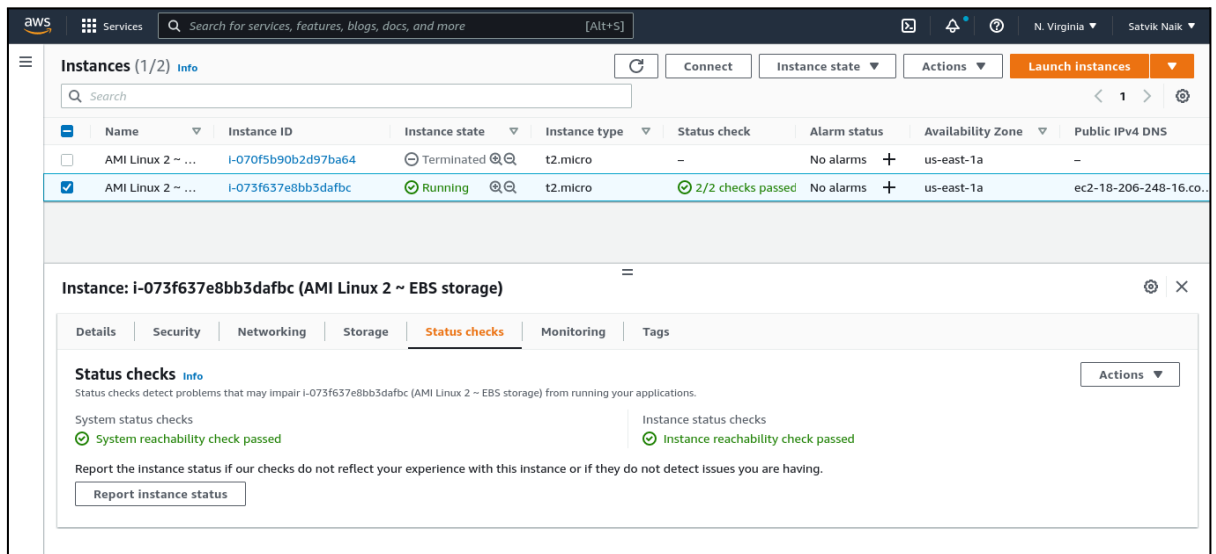
Apply complete! Resources: 3 added, 0 changed, 0 destroyed.

Outputs:
EbsVolumeId = "vol-0235016188cc92a94"
InstanceId  = "i-073f637e8bb3dafbc"
satvik--naik@pop-os: /media/satvik--naik/MyDesk/CLG/SEM7/CCE/cloud-computing-essentials/E3$
```

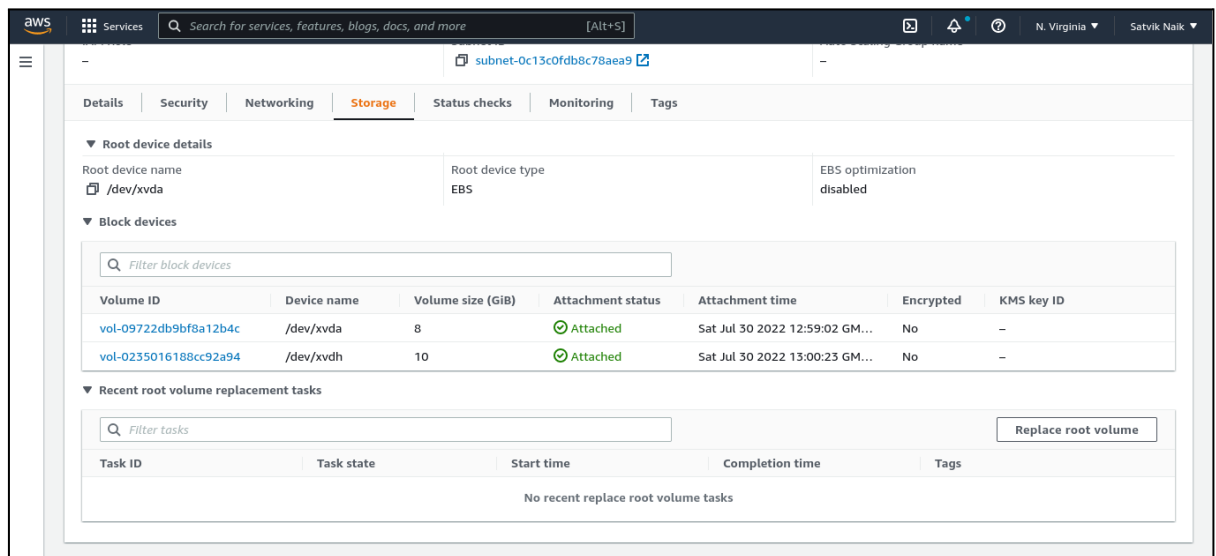
ICT GANPAT UNIVERSITY
COMPUTER SCIENCE DEPARTMENT
Cloud Computing Essentials (2CSE710)

3. Verifying resources created on AWS Cloud.

- Instance created successfully.



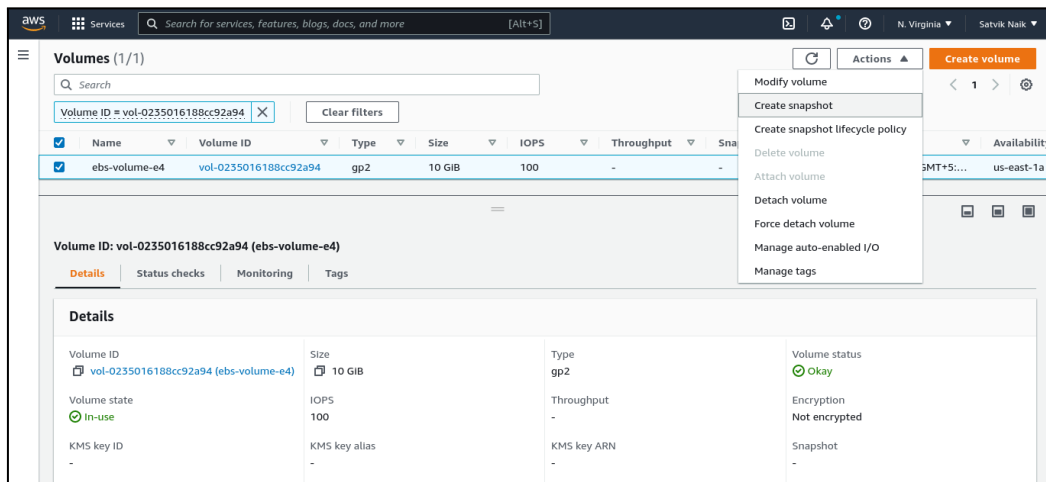
- Additional **EBS volume** attached of size **10GB**.



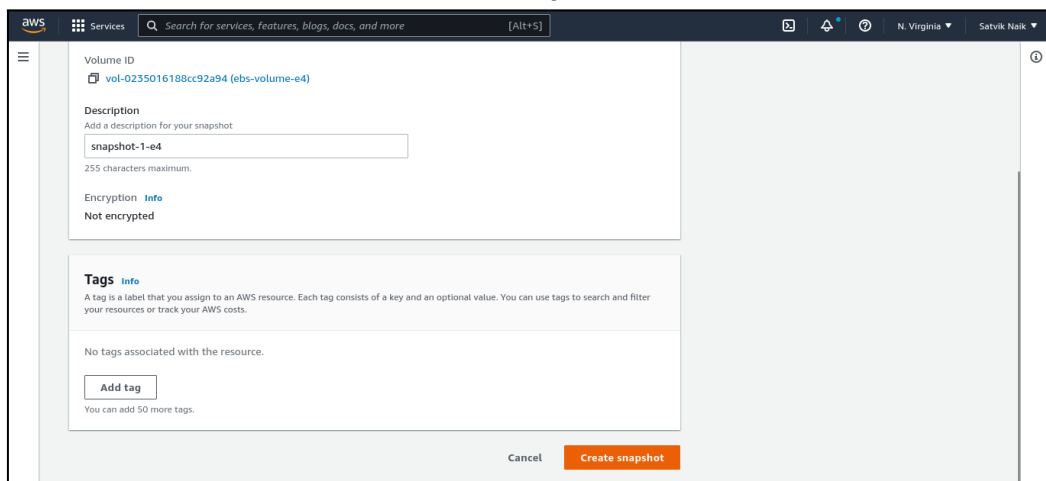
ICT GANPAT UNIVERSITY
COMPUTER SCIENCE DEPARTMENT
Cloud Computing Essentials (2CSE710)

4. Create a snapshot of your volume.

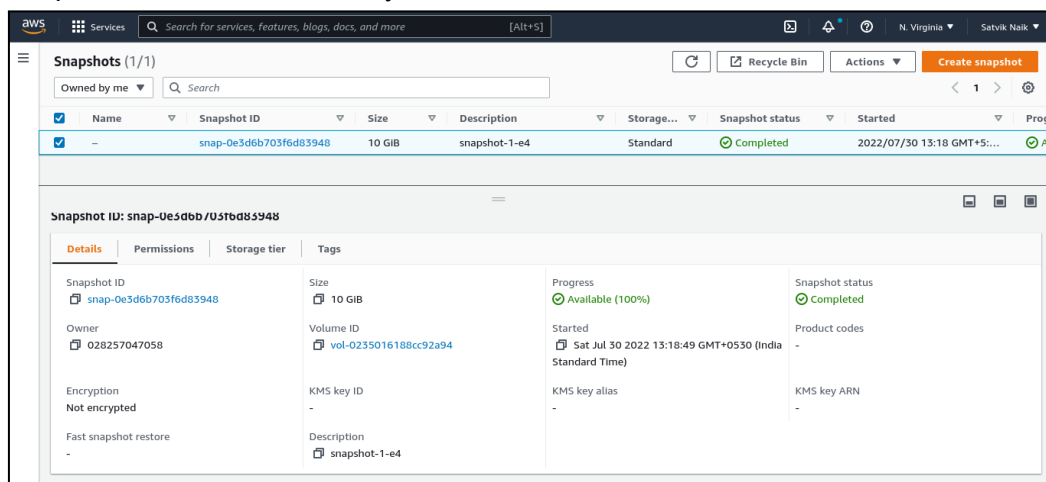
- Select the volume and go to **Actions > Create snapshot**.



- Provide description & click on **'Create snapshot'**.



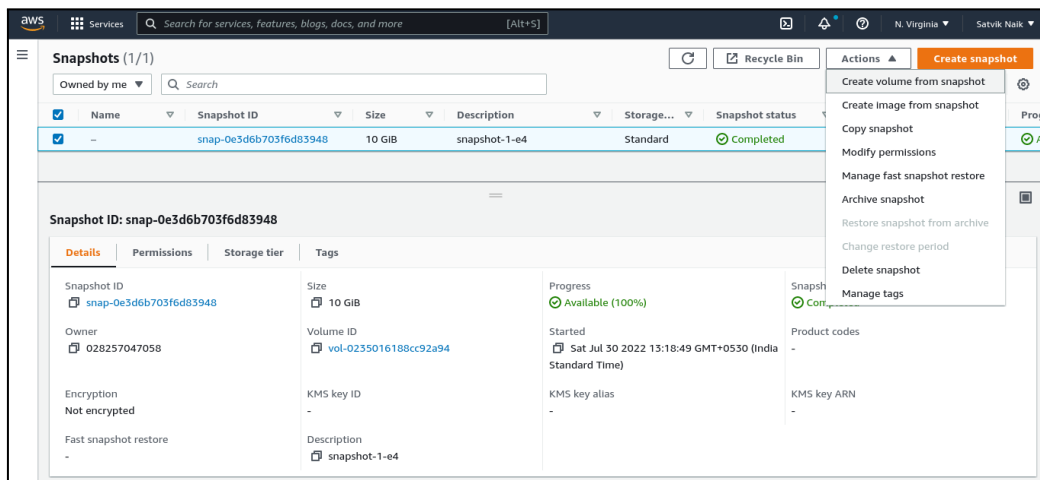
- Snapshot created successfully.



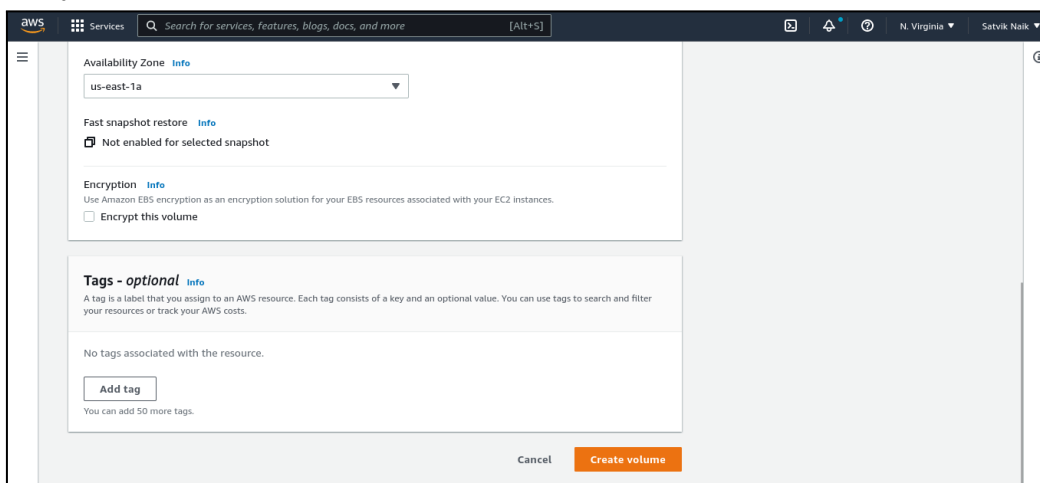
ICT GANPAT UNIVERSITY
COMPUTER SCIENCE DEPARTMENT
Cloud Computing Essentials (2CSE710)

5. Create a new volume from your snapshot.

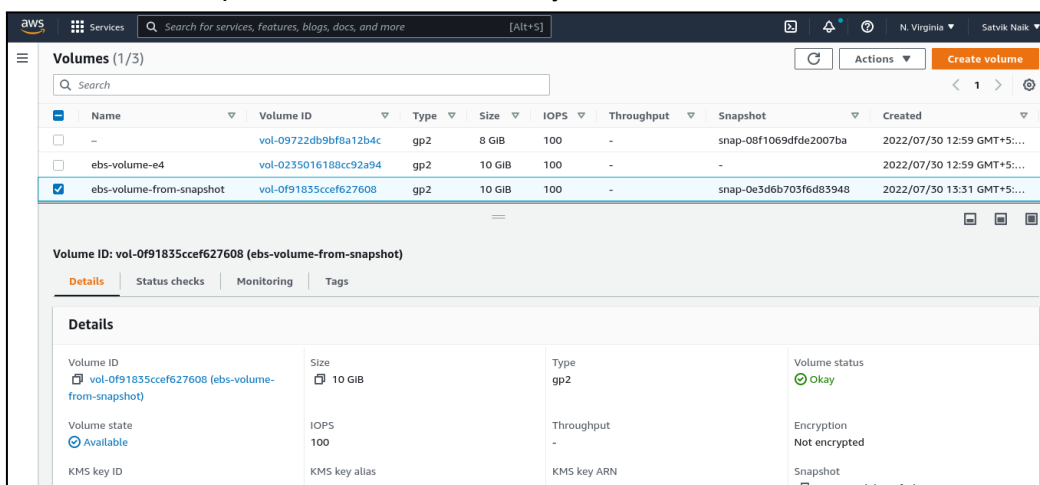
- Select the snapshot and go to **Actions > Create volume from snapshot**.



- Verify details & click on **'Create volume'**.



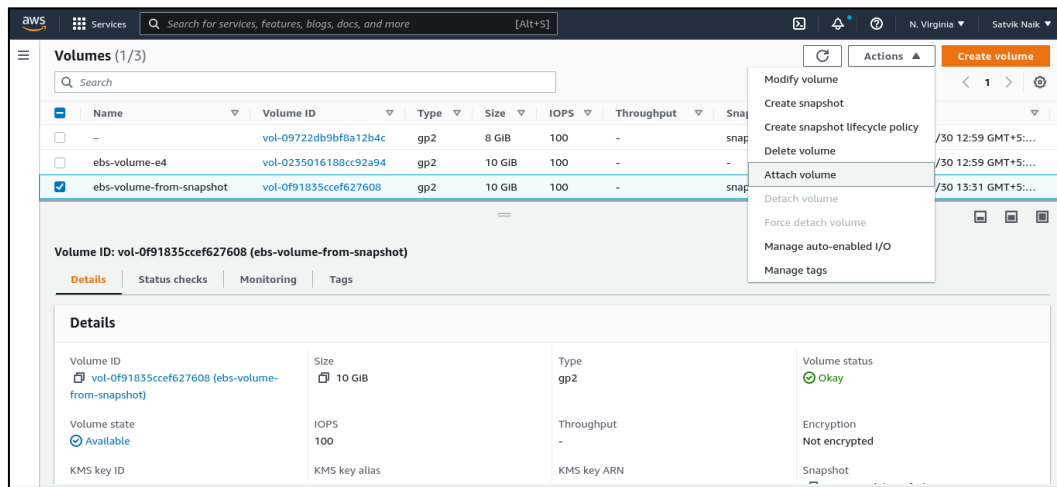
- Volume from snapshot created successfully.



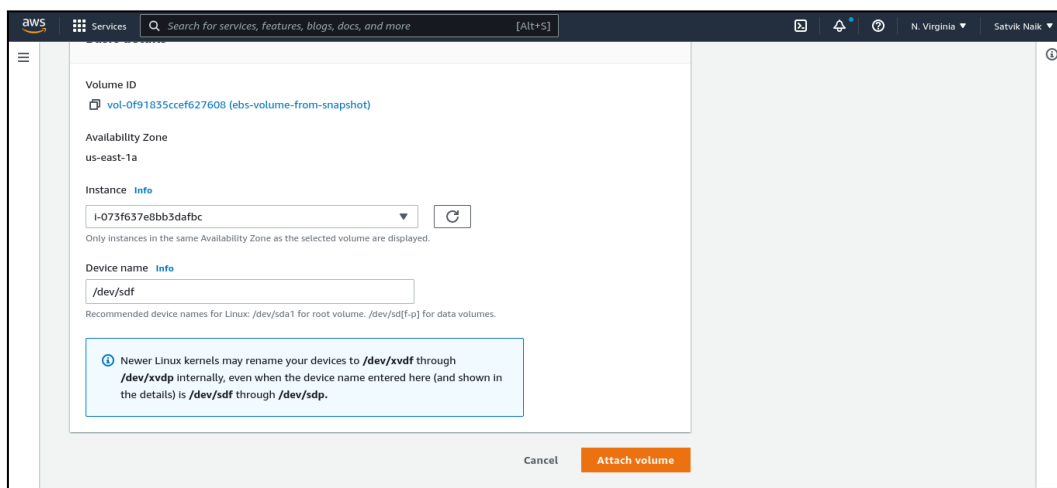
ICT GANPAT UNIVERSITY
COMPUTER SCIENCE DEPARTMENT
Cloud Computing Essentials (2CSE710)

6. Attach and mount your new volume to your EC2 instance.

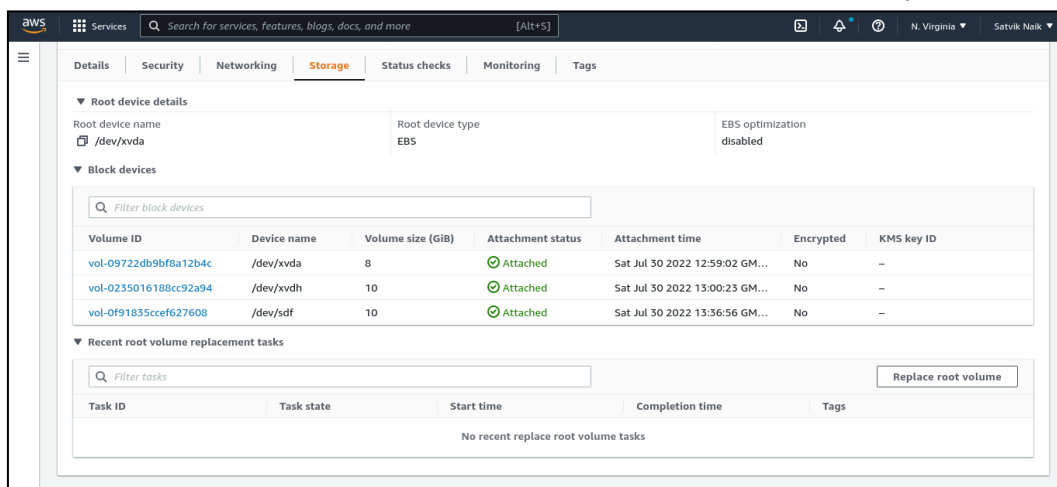
- Select the volume & go to **Actions > Attach volume**.



- Select the instance & click on **'Attach Volume'**.



- New volume which was created from a snapshot attached successfully.



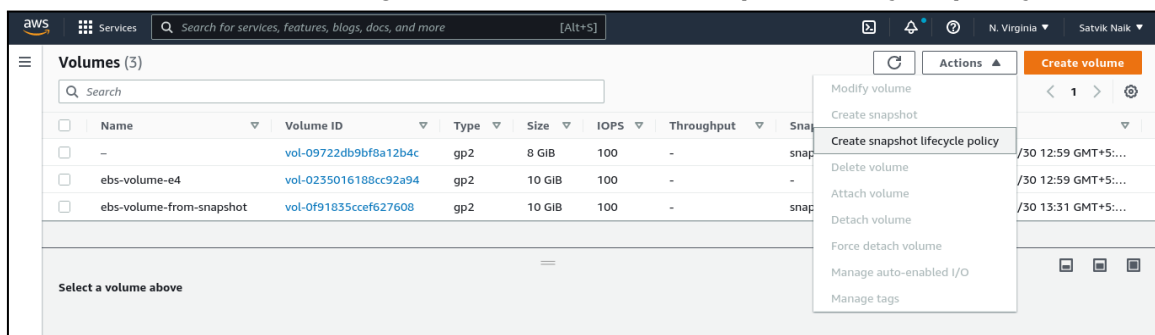
ICT GANPAT UNIVERSITY
COMPUTER SCIENCE DEPARTMENT
Cloud Computing Essentials (2CSE710)

```
aws
Services
Search for services, features, blogs, docs, and more

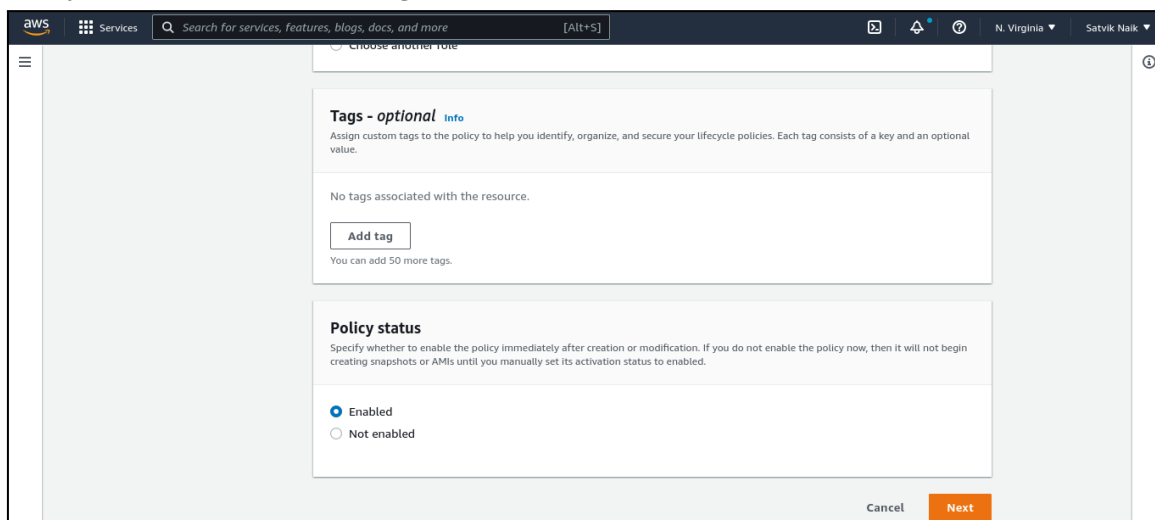
[ec2-user@ip-172-31-80-73 ~]$ lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
xvda        202:0    0   8G  0 disk
└─xvda1     202:1    0   8G  0 part /
xvdf        202:80   0  10G  0 disk
xvdh        202:112  0  10G  0 disk
[ec2-user@ip-172-31-80-73 ~]$
```

7. Schedule the snapshot.

- Under the **Volumes** section, go to **Actions > Create snapshot lifecycle policy**.



- Target resource types: **Volumes**
- Target resource tags: **ebs-volume-e4**
- Policy description:- **Scheduling of ebs-volume-e4**



ICT GANPAT UNIVERSITY
COMPUTER SCIENCE DEPARTMENT
Cloud Computing Essentials (2CSE710)

- Schedule details.

The screenshot shows the 'Schedule details' page in the AWS console. The 'Schedule name' is 'Schedule 1'. The 'Frequency' is set to 'Weekly'. Under 'on', 'Sun' is selected. 'Starting at' is '09:00' in 'UTC'. 'Retention type' is 'Count' with a value of '1'. A note at the bottom states: 'All schedules must have the same retention type. You can specify the retention type for Schedule 1 only. Schedules 2, 3, and 4 inherit the retention type from Schedule 1. Each schedule can have its own retention count or period.'

Click on **‘Review Policy’**.

- Review & Create Policy.

The screenshot shows the 'Review & Create Policy' page. It displays 'Target resource types' as 'Volume' and 'Target resource tags' as 'Name: ebs-volume-e4'. The 'Description' is 'Scheduling of ebs-volume-e4'. The 'Policy status' is 'Enabled'. Below this is 'Step 2: Schedule 1 configuration' with a 'Modify' button. The 'Schedule details' section shows 'Schedule name' as 'Schedule 1', 'Frequency' as 'Every Sunday starting at 09:00 UTC', and 'Retention' as '1 most recent snapshot(s)'. At the bottom are 'Cancel', 'Previous', and 'Create policy' buttons.

- Snapshot successfully scheduled.

The screenshot shows the 'Data Lifecycle Manager' page with a table of policies. The policy 'policy-084fc734631936ad8' is listed with a description of 'Scheduling of ebs-vol...' and a state of 'Enabled'. Below the table, the details for this policy are shown, including its ID, description, state, resource type (Volume), IAM role, and creation date.

Name	Policy ID	Description	Policy type	State
-	policy-084fc734631936ad8	Scheduling of ebs-vol...	EBS snapshot policy	Enabled

Policy: policy-084fc734631936ad8			
Details			
Policy ID policy-084fc734631936ad8	Description Scheduling of ebs-volume-e4	Resource type Volume	Resource location cloud
Policy type EBS snapshot policy	State Enabled	IAM role arn:aws:iam::028257047058:role/service-role/AWSDataLifecycleManagerDefaultRole	Date created Sat Jul 30 2022 13:53:10 GMT+0530 (India Standard Time)
Date modified	Target volumes with these tags		