

Practical 5

Reg. no.: 2147116

Name: Ishaan Bose

1. Describe STaaS.

Storage as a Service (STaaS) is the practice of using public cloud storage resources to store your data. Using STaaS is more cost efficient than building private storage infrastructure, especially when you can match data types to cloud storage offerings. The key benefit to STaaS is that you are offloading the cost and effort to manage data storage infrastructure and technology to a third-party CSP. This makes it much more effective to scale up storage resources without investing in new hardware or taking on configuration costs. You can also respond to changing market conditions faster. With just a few clicks you can rent terabytes or more of storage, and you don't have to spin up new storage appliances on your own.

2. Mention the different storage types offered by the cloud providers and list the example services for each type in GCP/AWS/Azure.

Storage types:

- **Block storage** breaks data into segmented pieces and distributes them to the storage environment wherever it is most efficient for the platform to do so. This simulates the same functionality as writing data to a standard hard disk drive or solid-state drive. Data remains available for quick access, but it is also costly to maintain and works best for warm or hot data storage.
- **File storage** lists data in a navigable hierarchy, usually a file directory. This is most like the file storage system that you would find on a PC or in cloud storage apps like Microsoft OneDrive. Because it is designed for humans to navigate, file storage is ideal anytime you need to collaborate on a project with other people or businesses. Whether the data is hot or cold doesn't matter as much. However, file storage does not scale well. The more files you add, the more complex the system becomes and the more difficult it is to navigate.
- **Object-based storage** organizes data by adding meta information to it, making it easy to recognize and retrieve at any time. This type of cloud storage scales up in the most cost-efficient manner, because you can keep adding to it. It is typically the least expensive type of STaaS and best suited for massive amounts of cold media or data files.

AWS:

- Amazon Simple Storage Service (S3)
- Amazon Elastic File System (EFS)
- Amazon FSx
- Amazon Elastic Block Storage (EBS)

GCP:

- Cloud storage

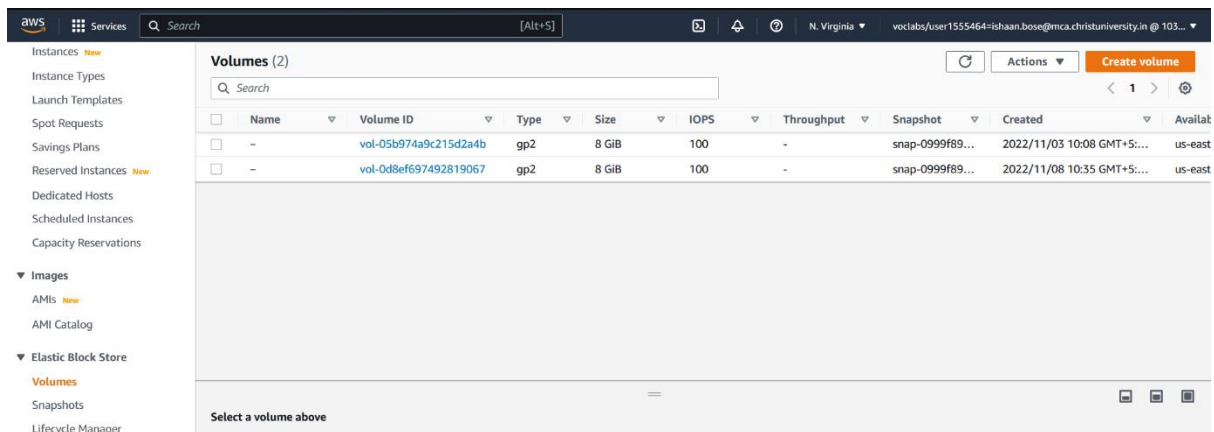
- Filestore
- Persistent Disk
- Local SSD

Azure:

- Azure Blobs
- Azure Files
- Azure Queues
- Azure Tables
- Azure Disks
- Azure NetApp Files

3. Demonstrate the following: Create a New Block Store and attach it to an VM instance.

1. To create a new block store, scroll down the navigation pane to Elastic Block Store and click on Volumes.
2. On the new page, click on Create volume.



3. Fill in the desired specifications for the new EBS. NOTE: Make sure that availability zone matches the availability zone of the ec2 instance which will be using this EBS.

Volume settings

Volume type [Info](#)

General Purpose SSD (gp2)

Size (GiB) [Info](#)

1

Min: 1 GiB, Max: 16384 GiB. The value must be an integer.

IOPS [Info](#)

100 / 3000

Baseline of 3 IOPS per GiB with a minimum of 100 IOPS, burstable to 3000 IOPS.

Throughput (MiB/s) [Info](#)

Not applicable

Availability Zone [Info](#)

us-east-1a

Snapshot ID - optional [Info](#)

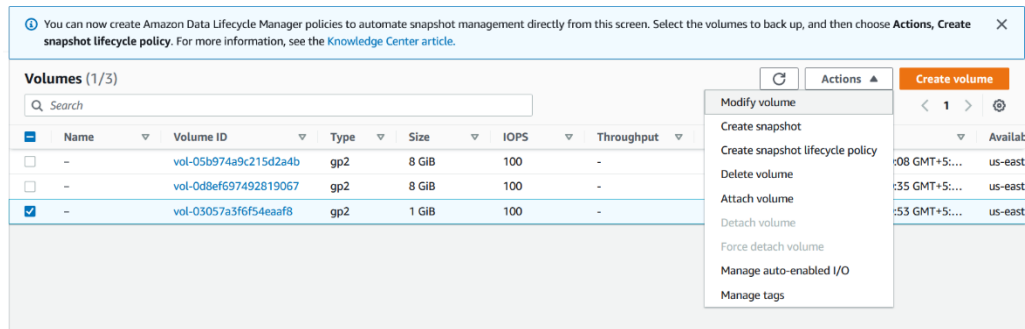
Don't create volume from a snapshot

Encryption [Info](#)

Use Amazon EBS encryption as an encryption solution for your EBS resources associated with your EC2 instances.

☐ Encrypt this volume

- Once the new EBS is created, wait for Volume state to be "Available".
- Once Available, click on the volume and then on Actions -> Attach volume



- Select the desired instance, and provide an appropriate device name.

Attach a volume to an instance to use it as you would a regular physical hard disk drive.

Basic details

Volume ID
vol-03057a3f6f54eaaf8

Availability Zone
us-east-1a

Instance [Info](#)
i-02e1cb5722c4780ce

Device name [Info](#)
/dev/sda2

Recommended device names for Linux: /dev/sda1 for root volume. /dev/sd[f-p] for data volumes.

Info Newer Linux kernels may rename your devices to /dev/xvdf through /dev/xvdp internally, even when the device name entered here (and shown in the details) is /dev/sdf through /dev/sdp.

Cancel **Attach volume**

- Connect to your instance.
- Run command lsblk to view the instance which has been attached but not mounted.

```
ubuntu@ip-172-31-30-37:~$ lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
loop0       7:0      0  25.1M 1 loop /snap/amazon-ssm-agent/5656
loop1       7:1      0  55.6M 1 loop /snap/core18/2560
loop2       7:2      0  63.2M 1 loop /snap/core20/1623
loop3       7:3      0   103M 1 loop /snap/lxd/23541
loop4       7:4      0    47M 1 loop /snap/snapd/16292
loop5       7:5      0    48M 1 loop /snap/snapd/17336
loop6       7:6      0  55.6M 1 loop /snap/core18/2620
loop7       7:7      0  63.2M 1 loop /snap/core20/1695
loop8       7:8      0   24.4M 1 loop /snap/amazon-ssm-agent/6312
xvda        202:0     0     8G  0 disk
└─xvda1     202:1     0    7.9G  0 part /
└─xvda14    202:14    0     4M  0 part
└─xvda15    202:15    0   106M  0 part /boot/efi
xvdb        202:16    0     1G  0 disk
```

- Run sudo mkfs -t xfs /dev/xvdb to make file system for the volume

```

ubuntu@ip-172-31-30-37:~$ sudo mkfs -t xfs /dev/xvdb
meta-data=/dev/xvdb             isize=512    agcount=4, agsize=65536 blks
=                               sectsz=512   attr=2, projid32bit=1
=                               crc=1        finobt=1, sparse=1, rmapbt=0
=                               reflink=1    bigtime=0 inobtcount=0
data      =                       bsize=4096   blocks=262144, imaxpct=25
=                               sunit=0        swidth=0 blks
naming    =version 2           bsize=4096   ascii-ci=0, ftype=1
log        =internal log      bsize=4096   blocks=2560, version=2
=                               sectsz=512    sunit=0 blks, lazy-count=1
realtime  =none                extsz=4096   blocks=0, rtextents=0
ubuntu@ip-172-31-30-37:~$ ll

```

10. Create a directory to mount the volume.
11. Run command: `sudo mount /dev/xvdb ./data` to mount volume to data dir
12. Run `lsblk` again to confirm mounting

```

ubuntu@ip-172-31-30-37:~$ lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
loop0       7:0      0  25.1M  1 loop /snap/amazon-ssm-agent/5656
loop1       7:1      0  55.6M  1 loop /snap/core18/2560
loop2       7:2      0  63.2M  1 loop /snap/core20/1623
loop3       7:3      0  103M   1 loop /snap/lxd/23541
loop4       7:4      0   47M   1 loop /snap/snapd/16292
loop5       7:5      0   48M   1 loop /snap/snapd/17336
loop6       7:6      0  55.6M  1 loop /snap/core18/2620
loop7       7:7      0  63.2M  1 loop /snap/core20/1695
loop8       7:8      0  24.4M  1 loop /snap/amazon-ssm-agent/6312
xvda        202:0     0    8G   0 disk
├─xvda1     202:1     0   7.9G  0 part /
├─xvda14    202:14    0    4M   0 part
├─xvda15    202:15    0  106M  0 part /boot/efi
└─xvdb      202:16    0    1G   0 disk /home/ubuntu/data
ubuntu@ip-172-31-30-37:~$

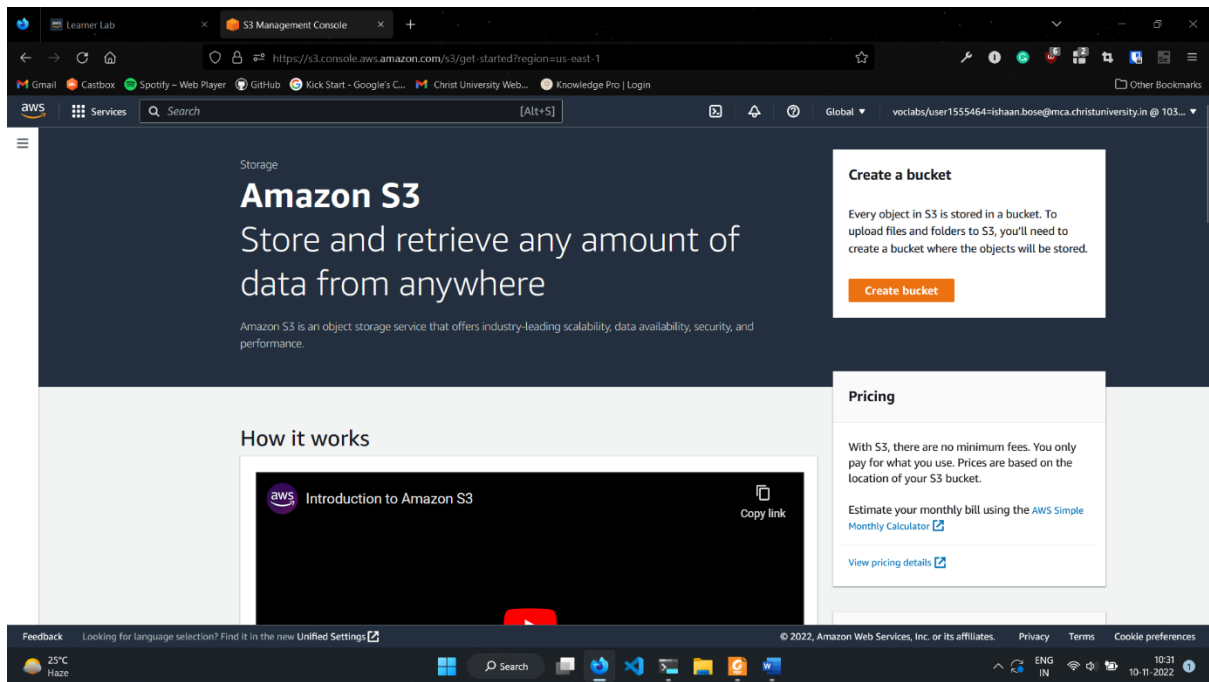
```

4. Assume that, an online news agency wants to support their agents and editors for fast publication of the news using cloud technologies. They are expecting a small software application for the news agents and editors for the following requirements.

1. Provision to upload the images of the events from venue to the folder named “oimage” present in the cloud storage.
2. After the uploading of the image a thumbnail image should be created for the same and it will be stored separately in a folder called “timages” for the selection of the right image for the news feed.

Identify a suitable Cloud Storage for the requirement and demonstrate the scenario using python.

1. Navigate to Amazon S3 console and create a bucket.



2. Choose an appropriate bucket name and leave all other settings as is.

Create bucket [Info](#)
Buckets are containers for data stored in S3. [Learn more](#)

General configuration

Bucket name
news-agency-imgs
Bucket name must be globally unique and must not contain spaces or uppercase letters. [See rules for bucket naming](#)

AWS Region
US East (N. Virginia) us-east-1

Copy settings from existing bucket - *optional*
Only the bucket settings in the following configuration are copied.
[Choose bucket](#)

3. Click on the newly created bucket and click on create folder. Give name oimage to it and click on Create folder.

Folder

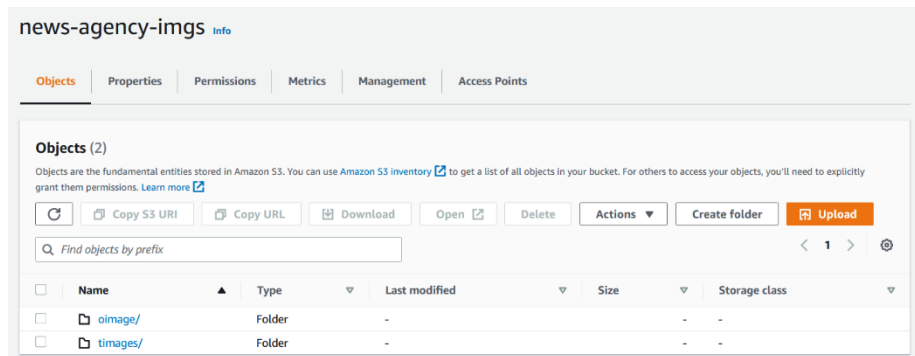
Folder name
oimg /
Folder names can't contain "/". [See rules for naming](#)

Server-side encryption

The following settings apply only to the new folder object and not to the objects contained within it.

Server-side encryption
☒ Disable
☐ Enable

[Cancel](#) [Create folder](#)



4. Once the folders are created, we can start uploading files to them.
5. For this exercise, we use a python script to allow users to choose which image they want to upload.

Python script:

```
import boto3
from tkinter import Tk
from tkinter.filedialog import askopenfilename
import cv2
import os

BUCKET_NAME = "news-agency-imgs"

s3_client = boto3.client('s3')
s3_resource = boto3.resource('s3')

Tk().withdraw() # we don't want a full GUI, so keep the root window from
appearing
filepath = askopenfilename() # show an "Open" dialog box and
return the path to the selected filename
print(filepath)

filename = filepath.split("/")[-1] # file name
print(filename)

s3_resource.meta.client.upload_filename(
    filepath, BUCKET_NAME, "oimage/" + filename
) # uploading image to oimage folder

print("Uploaded to oimage!")

s3_resource.Object(BUCKET_NAME, "oimage/" +
filename).download_filename(f"./.temp/{ filename }") # downloading image
from oimage folder into .temp directory
print("Downloaded image!")

image = cv2.imread(filepath, 1)
```

```
thumbnail = cv2.resize(image, (300, 300), interpolation=cv2.INTER_CUBIC)
# downsizing image to size 300x300

cv2.imwrite(f"./.temp/t_{ filename }", thumbnail) # writing to .temp file

s3_resource.meta.client.upload_filename(
    f"./.temp/t_{ filename }", BUCKET_NAME, f"timages/t_{filename}"
) # uploading to timages folder

print("Uploaded to timages")

dir = "./.temp"

for f in os.listdir(dir): # deleting all files in .temp folder
    os.remove(os.path.join(dir, f))

print("Deleted contents of .temp directory.")
```