M7024E Laboratory 2: Programming Cloud Services - Storage Services

submitted by

Muiz Olalekan Raheem, Md Asif Mahmod Tusher Siddique

November 25, 2022

submitted to

Dr. Karan Mitra

1 Objectives

The objective of this lab is to:

- Setup a programming environment for building (using programming tools and languages) Cloud services for a major Cloud provider, for example, Amazon Web services (AWS);

- Develop Cloud services for file storage, listing and retrieval using the APIs provided by the AWS.

2 Exercise a

In this exercise, you will learn how to setup the programming environment to create Cloud services using the APIs provided by AWS.

- Go through the lecture on "Programming Cloud Services" and setup your credentials on your computer. You should have the credentials when an AWS account was created for you in lab 1.

- Download the AWS SDK for Java Developer Guide: Rel. 2.0 2.

- Download the latest version of the Java SDK.

- Download the latest version of Java IDE, for example Netbeans, IntelliJ or Eclipse.

- Setup the AWS SDK for Java as described in the lecture. You might want to use Apache Maven that is usually a part of an Java IDE, for example, NetBeans.

- Setup your Java project and the pom.xml file. See pg. 8 of the Java Developer Guide.

Answer: All the steps are followed and all the dependencies are installed.

3 Exercise b

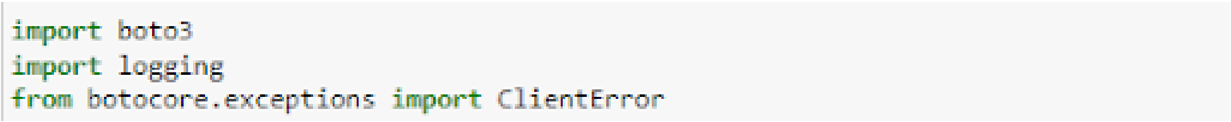
In this exercise, you will learn how to use the storage service provided by AWS.

1. Identify ways of creating the Amazon S3 service clients. Look at the lecture slides and the Java Developer Guide.

(a) Explain in detail how the Amazon S3 service clients are created by providing details of the packages and classes involved. Create a diagram of the dependencies involved.

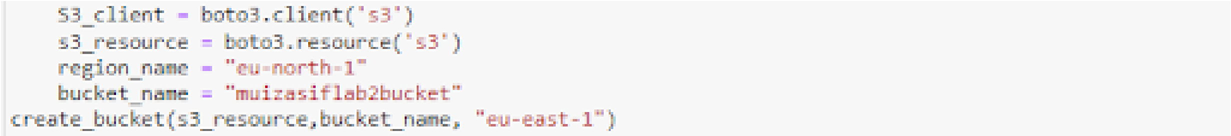
Answer:

For this lab, we utilized boto3 which is the Amazon Web Services (AWS) Software Development Kit (SDK) for Python. It allows Python developers to write software that makes use of services like Amazon S3 and Amazon EC2.



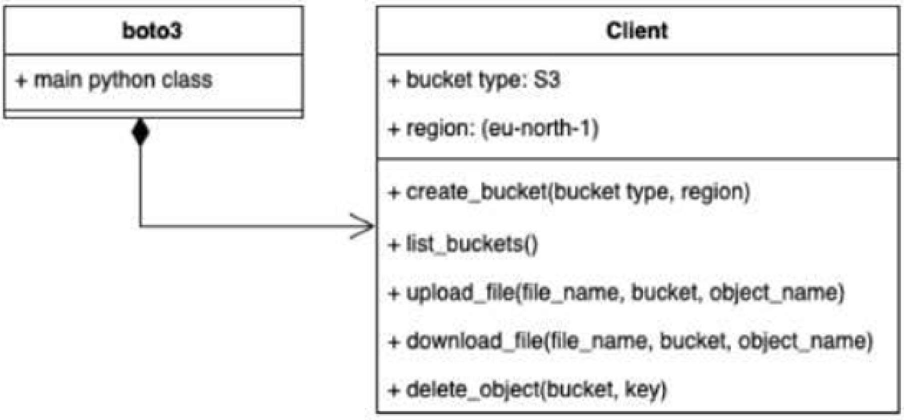
```
import boto3
import logging
from botocore.exceptions import ClientError
```

An Amazon S3 bucket is a storage location to hold files. S3 files are referred to as objects.



```
    s3_client = boto3.client('s3')
    s3_resource = boto3.resource('s3')
    region_name = "eu-north-1"
    bucket_name = "muizasiflab2bucket"
create_bucket(s3_resource,bucket_name, "eu-east-1")
```

Class diagram of the used classes and methods for s3 management *AWS, howpublished = http://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3.html, note = Accessed: 2022-11-20:*



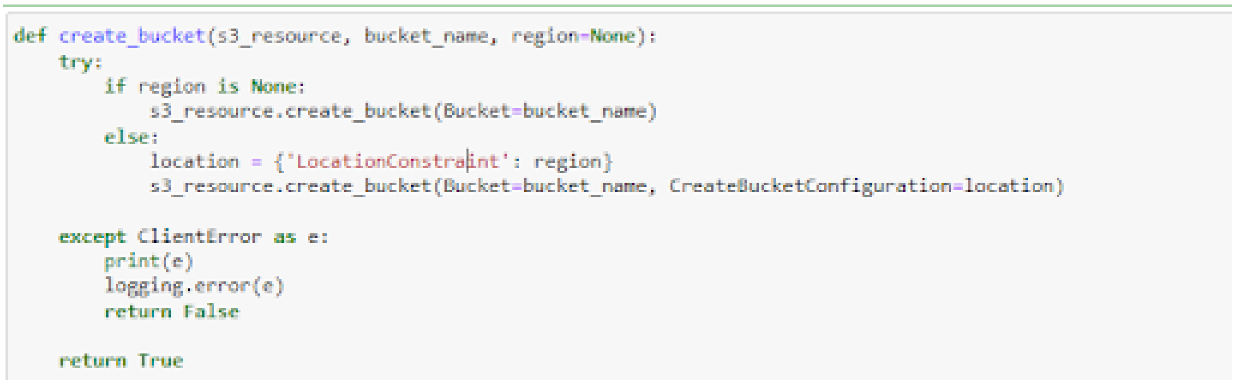
boto3
+ main python class
Client
+ bucket type: S3
+ region: (eu-north-1)
+ create_bucket(bucket type, region)
+ list_buckets()
+ upload_file(file_name, bucket, object_name)
+ download_file(file_name, bucket, object_name)
+ delete_object(bucket, key)

2. Create a Java program to create a bucket in three regions of your choice.

(a) Explain in detail the steps involved, and explain the output.

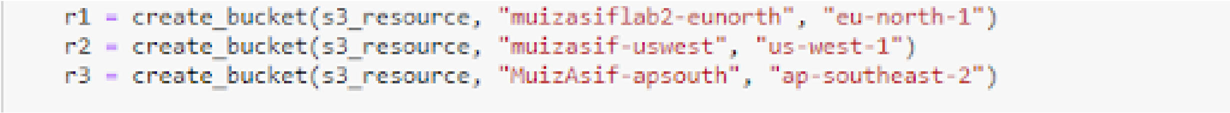
Answer:

To create a bucket in one or more regions, we first defined a function that create an S3 bucket in a specified region. The code to do that is as displayed below

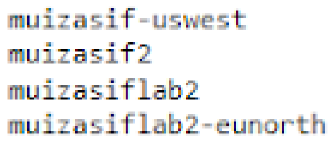


```
def create_bucket(s3_resource, bucket_name, region=None):
    try:
        if region is None:
            s3_resource.create_bucket(Bucket=bucket_name)
        else:
            location = {'LocationConstraint': region}
            s3_resource.create_bucket(Bucket=bucket_name, CreateBucketConfiguration=location)

    except ClientError as e:
        print(e)
        logging.error(e)
        return False

    return True
```

The create_bucket function takes in 3 parameters, namely, s3_resource, bucket_name, and the specified region. If no region is specified, the S3 default region, us-east-1 is used. To call the function, the image below shows how it was done.



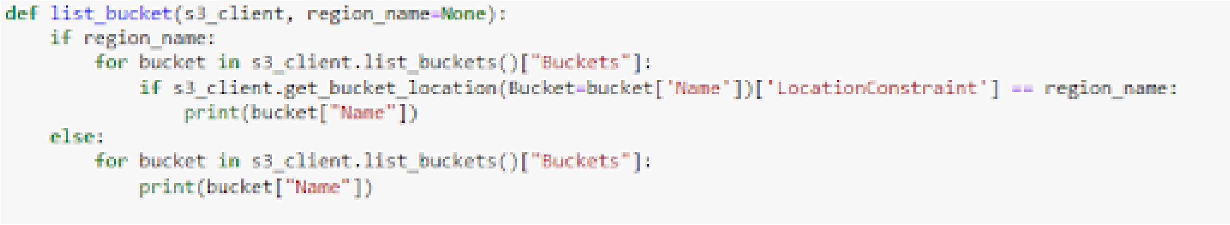
```
r1 = create_bucket(s3_resource, "muizasiflab2-eunorth", "eu-north-1")
r2 = create_bucket(s3_resource, "muizasif-uswest", "us-west-1")
r3 = create_bucket(s3_resource, "MuizAsif-apsouth", "ap-southeast-2")
```



```
muizasif-uswest
muizasif2
muizasiflab2
muizasiflab2-eunorth
```

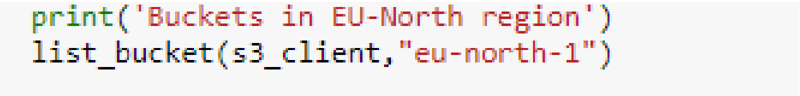
3. Create a Java program that lists your buckets in the region of your choice.

Answer: The function



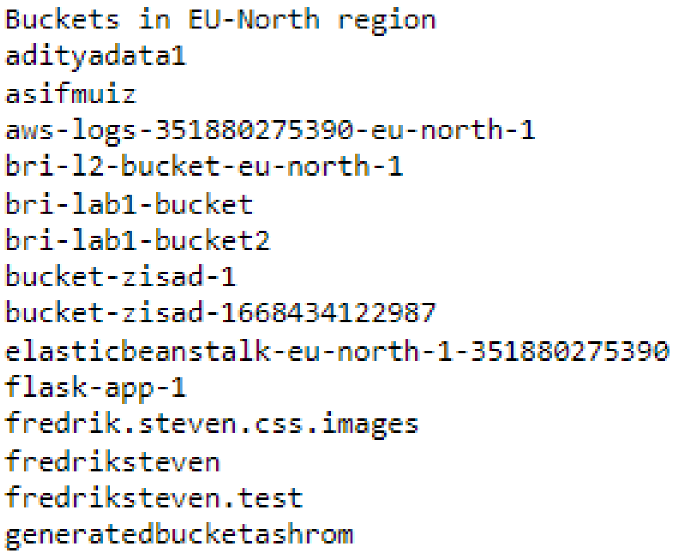
```
def list_bucket(s3_client, region_name=None):
    if region_name:
        for bucket in s3_client.list_buckets()["Buckets"]:
            if s3_client.get_bucket_location(Bucket=bucket['Name'])['LocationConstraint'] == region_name:
                print(bucket["Name"])
    else:
        for bucket in s3_client.list_buckets()["Buckets"]:
            print(bucket["Name"])
```

The input



```
print('Buckets in EU-North region')
list_bucket(s3_client,"eu-north-1")
```

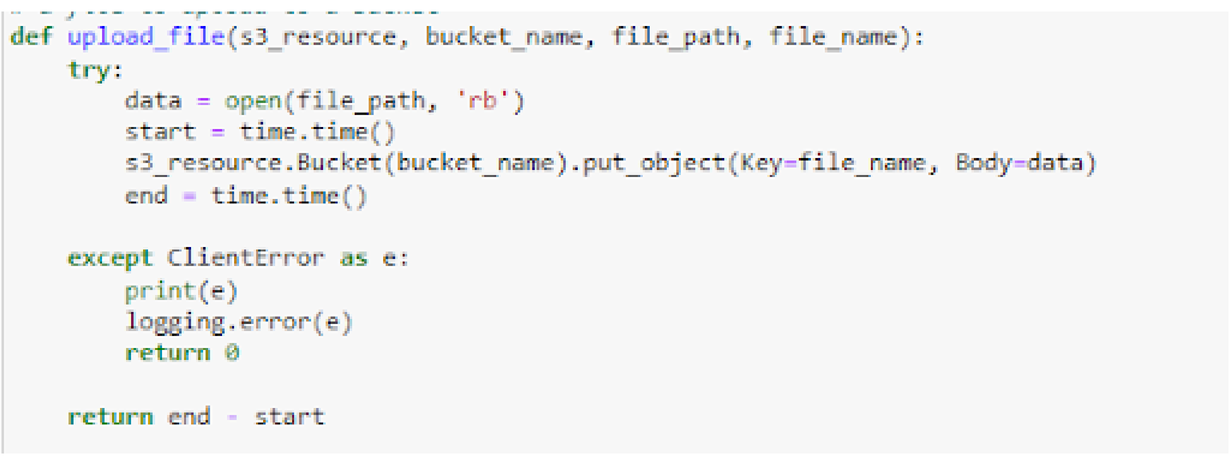
The output



```
Buckets in EU-North region
adityadata1
asifmuiz
aws-logs-351880275390-eu-north-1
bri-l2-bucket-eu-north-1
bri-lab1-bucket
bri-lab1-bucket2
bucket-zisad-1
bucket-zisad-1668434122987
elasticbeanstalk-eu-north-1-351880275390
flask-app-1
fredrik.steven.css.images
fredriksteven
fredriksteven.test
generatedbucketashrom
```

4. Create a Java program to upload objects in your newly created bucket.

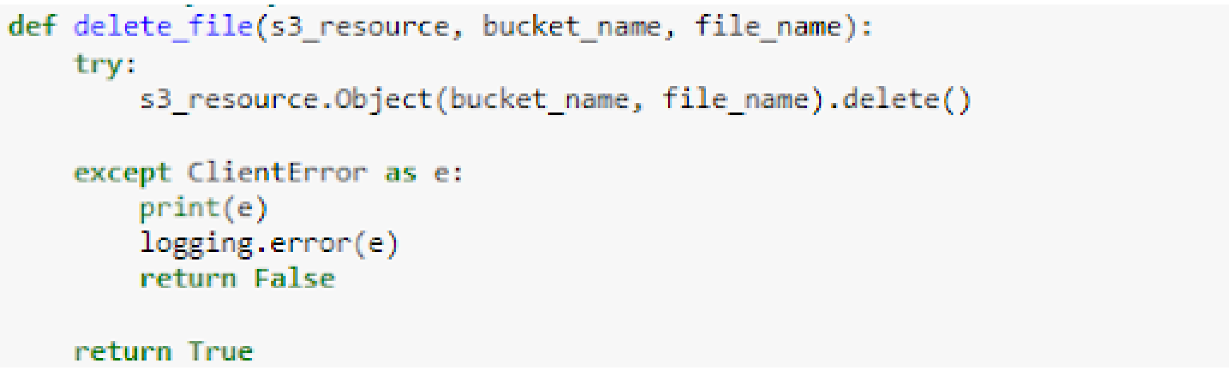
Answer:



```
def upload_file(s3_resource, bucket_name, file_path, file_name):
    try:
        data = open(file_path, 'rb')
        start = time.time()
        s3_resource.Bucket(bucket_name).put_object(Key=file_name, Body=data)
        end = time.time()

    except ClientError as e:
        print(e)
        logging.error(e)
        return 0

    return end - start
```

5. Create a Java program to delete a particular objects from your newly created bucket.

Answer:



```
def delete_file(s3_resource, bucket_name, file_name):
    try:
        s3_resource.Object(bucket_name, file_name).delete()

    except ClientError as e:
        print(e)
        logging.error(e)
        return False

    return True
```

4 Exercise c

Create a Java program to upload and download objects (sizes 1MB, 10 MB, 100 MB and 500MB) from the three regions used in the above exercise. Measure the object upload and download latency from these regions. Plot and explain the results in your report. Think about statistical analysis and any assumptions made.

Answer:

For this exersize we first created some dummy files of the required file sizes (1 Megabyte, 10 Megabytes and 100 Megabytes) using a command as follows:

feutil file createnew pathfilename

We then wrote function that calculated the time taken to upload and download the files to and from AWS server in different regions. Following figures shows the mean time taken to Upload and Download the files to servers. The Experiment was performed 30 times for each file size and each end point.

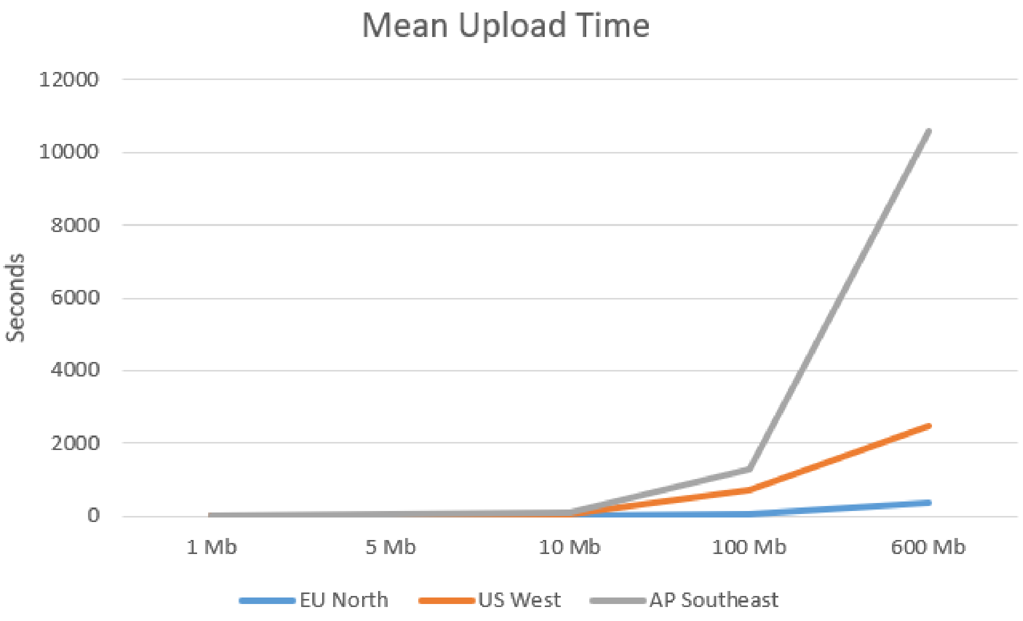


Figure 1: Mean Upload time

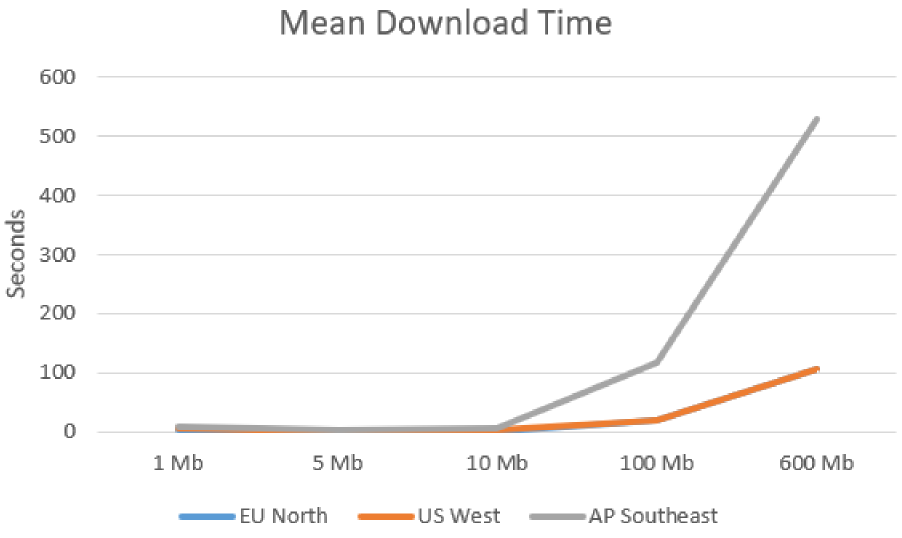
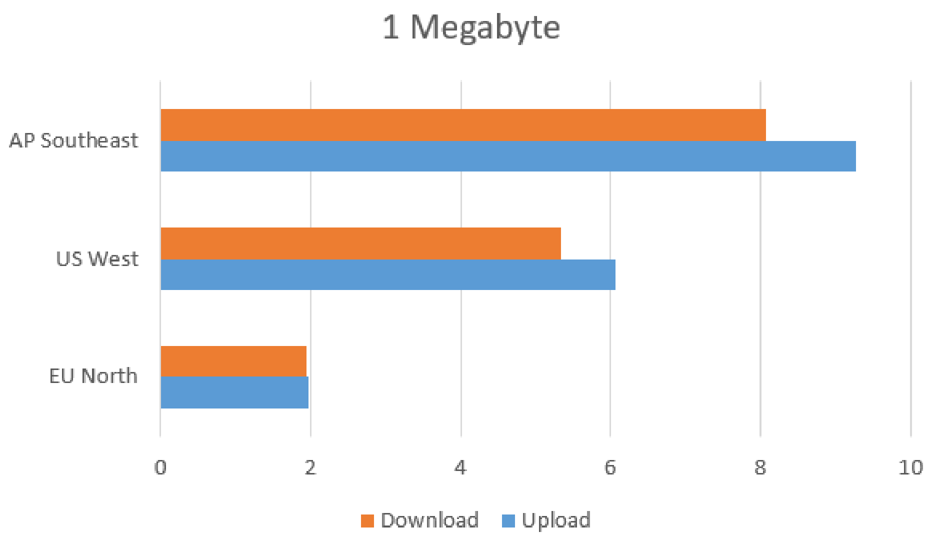
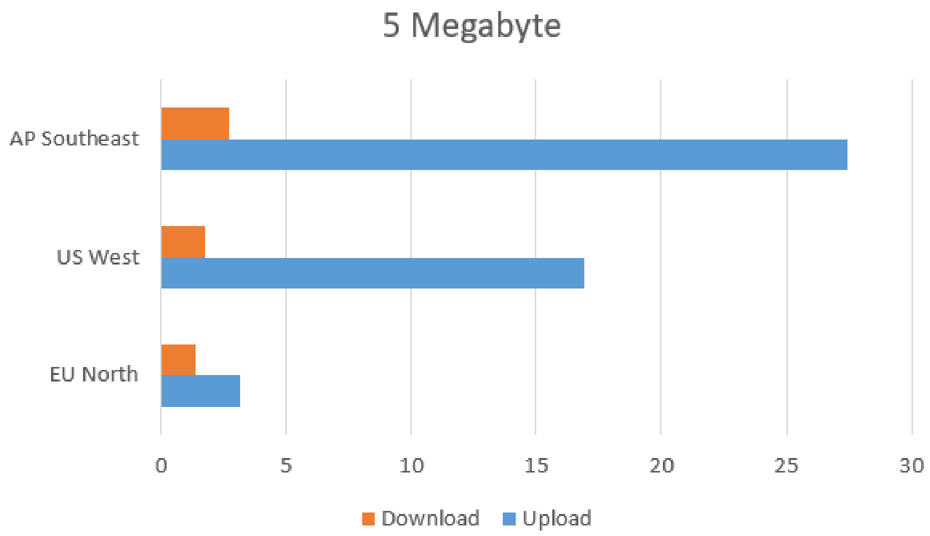
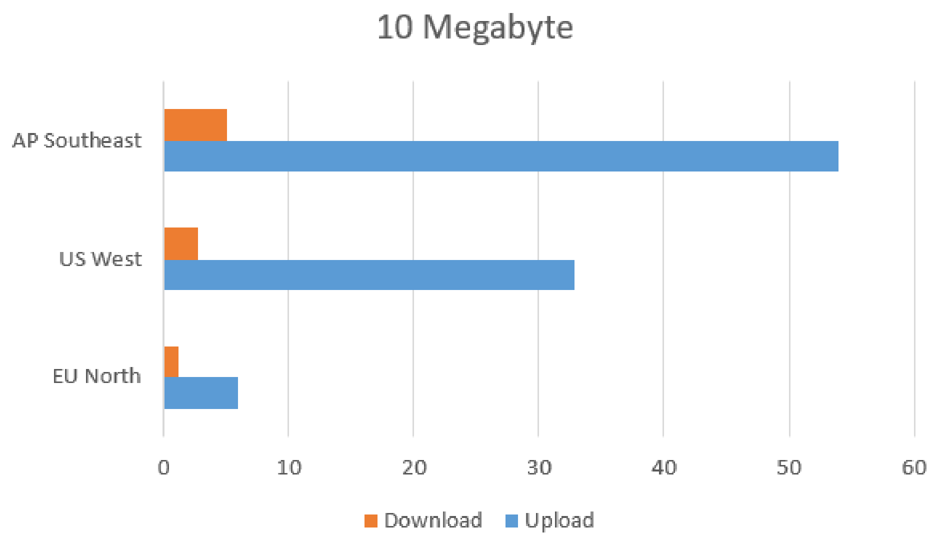


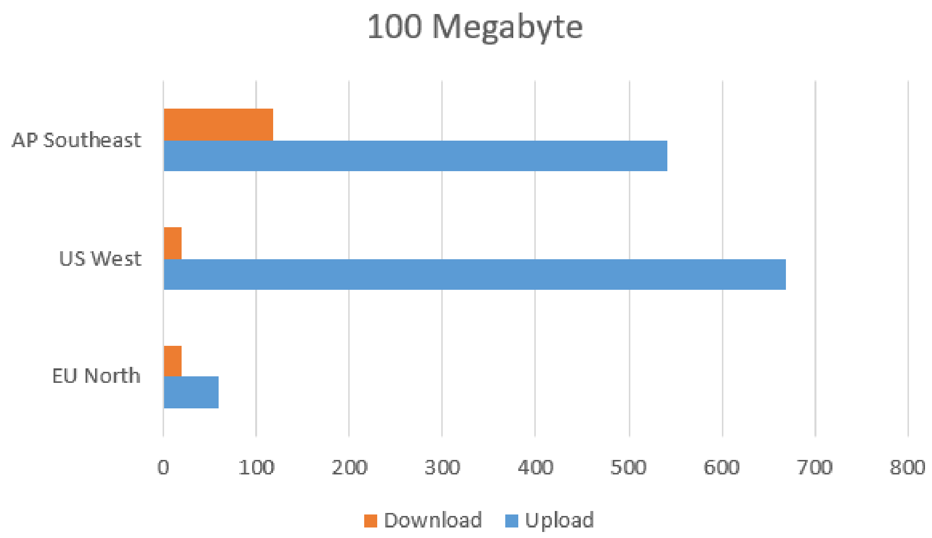
Figure 2: Mean Download Time

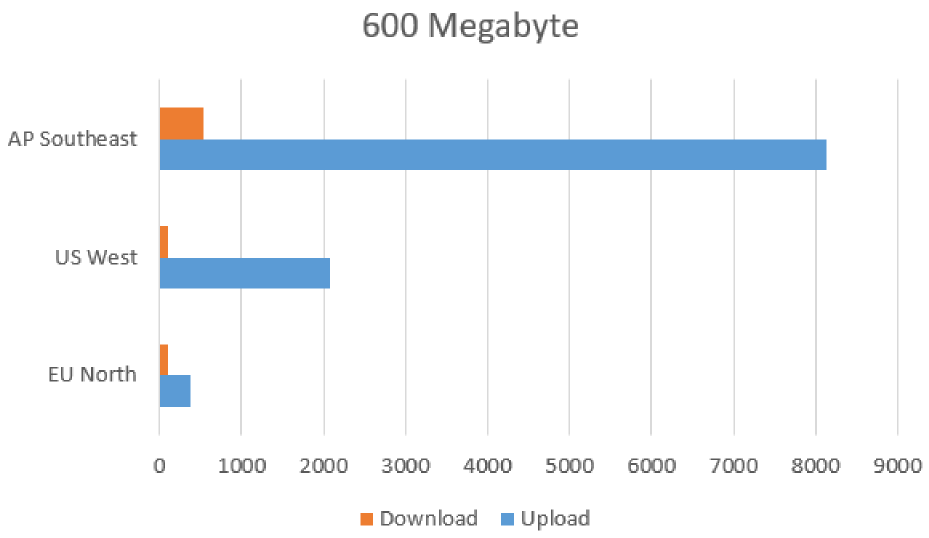
Following figures show the difference between upload and download time for different filesizes:


1 Megabyte
AP Southeast
US West
EU North
0
2
4
6
8
10
Download
Upload


5 Megabyte
AP Southeast
US West
EU North
0
5
10
15
20
25
30
Download
Upload


10 Megabyte
AP Southeast
US West
EU North
0
10
20
30
40
50
60
Download
Upload


100 Megabyte
AP Southeast
US West
EU North
0
100
200
300
400
500
600
700
800
Download
Upload



From this we get the following insights:

- AP Southeast and US-west have high upload and download latency due to location/region and network connectivity

- AP Southeast region has the highest latency followed by us-west region

- overall download speeds are better than the upload speeds

References

AWS, howpublished = http://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3.html, note = Accessed: 2022-11-20.