Assignment 3 – S3

- [task 1] Screen shot of the file name after downloading it from the S3 bucket in EC2
- [task 2] SNS topic
- [task 2] Event notification in S3 properties.
- [task 2] The email from S3 Event notification.
- [task 3] The signed URL.
- [task 3] Your Lambda code that generates a signed URL.

Always follow the least privilege principle for the IAM policies and security groups.

Task 1. Download/upload a file from S3 in EC2.

a. Create an S3 bucket and put a file in it.
b. Create an instance with an IAM role (AKA instance profile).

Add the following IAM policy to the role. JSON policy can be generated with the policy creation wizard.

```
{
        "Version": "2012-10-17",
        "Statement": [
                {
                        "Effect": "Allow"
                        "Action": ["s3:GetObject", "s3:PutObject"],
                        "Resource": "arn:aws:s3:::<your_bucket>/*",
                }
        ]
}
```

The policy above is an identity-based authorization since you are attaching it to an IAM role. Alternatively, you can write a resource-based policy on S3 that allows access from the EC2 instance. So, even though the instance role (profile) does not have S3 access, it can still access S3 as you allowed that access on the S3 side.

---

c. Download and update the file with the following CLI command.

```
aws s3 cp s3://<bucket_name>/<file_name_in_s3> <new_file_name_in_EC2>
```

Task 2. S3 Event Notification

Send an email to yourself when the object is created in the bucket.

- You need to create an SNS topic.
- Subscribe to the topic with your email.
- You have to write a resource-based policy in the SNS topic after creation. Remember, if you are connecting 2 different AWS services, you must write an IAM policy. That allows S3 to publish messages on the topic. [**Include it in the PDF**]

```
{
  "Version": "2008-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "s3.amazonaws.com"
      },
      "Action": "SNS:Publish",
      "Resource": "<Your SNS ARN>",
      "Condition": {
        "StringEquals": {
          "AWS:SourceAccount": "<Your AWS Account Number>"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:s3:*:*:<Your Bucket Name>"
        }
      }
    }
  ]
}
```

Task3. S3 Signed URL

Write a lambda that returns a Signed URL of the object in S3. Make sure the LabRole has an inline policy that allows getting objects from the bucket. Create a “Test Event” to trigger the lambda.

For more : https://docs.aws.amazon.com/AWSJavaScriptSDK/v3/latest/index.html

```
import { S3Client } from '@aws-sdk/client-s3';
import { GetObjectCommand } from '@aws-sdk/client-s3';
import { getSignedUrl } from '@aws-sdk/s3-request-presigner'; const s3 = new
S3Client({ region: 'us-east-1' }); export const handler= async (event) => {
const params = { Bucket: '<bucket_name>', Key: '<object_key>' };
   const command = new GetObjectCommand(params);
  const url = await getSignedUrl(s3, command, { expiresIn: 3600 });
  return url;
};
```

Extra:

Read a file in S3 in EC2 using **S3 Gateway Endpoint**. After a successful connection, write S3 resource-based policy that allows reading access only from the VPC endpoint in the bucket policy. Refer: Amazon S3 and VPC Endpoints. The difference between IAM role and VPC endpoint is.