AWS ASSIGNMENT

PEER LEARNING DOCUMENT

Problem Statement -

AWS assignment: Save json files in S3 bucket from AWS Lambda

Services - IAM, EC2, S3, AWS Lambda, API Gateway, Cloudwatch

Language - python3

Packages - boto3, datetime

Objectives -

1. Create S3 bucket from AWS CLI
   a. Create an IAM role with S3 full access.
   b. Create an EC2 instance with above role
   c. Create a bucket from AWS CLI
2. Put files in S3 bucket from lambda
   a. Create custom role for AWS lambda which will have only put object access
   b. Add role to generate and access Cloudwatch logs
   c. In python script, generate json in given format and save .json file in bucket created
   d. Schedule the job to run every minute. Stop execution after 3 runs
   e. Check if cloudwatch logs are generated
3. API gateway - Lambda integration
   a. Modify lambda function to accept parameters and return file name.
   b. Create a POST API from API Gateway, pass parameters as request body to Lambda job. Return filename and status code as response.
   c. Consume API from local machine and pass unique data to lambda.
   d. Check if cloudwatch logs are generated

Instructions -

- json format to save in bucket:

```
{
  "transaction_id": 12345,
  "payment_mode": "card/netbanking/upi",
  "amount": 200.0,
  "customer_id": 101,
  "timestamp": <current timestamp>
}
```

- json format for API gateway:

```
{
  "transaction_id": 12345,
  "payment_mode": "card/netbanking/upi",
  "amount": 200.0,
  "customer_id": 101
}
```

- You can also create GET API and pass data as path/query parameters
- Calculate timestamp in the lambda function and append in the json

Rahul Kumar

Approach -

1. He had written a set of commands and ran it from CLI in order to complete the question, the commands are as follows :
    a. Providing the role with S3 full access
        i. aws iam attach-role-policy --role-name BucketMaker --policy-arn arn:aws:iam::aws:policy/AmazonS3FullAccess
    b. Create an IAM role with S3 full access
        i. aws iam create-role --role-name Bucketmaker --assume-role-policy-document file://ec2-trust-policy.json
2. To create an EC2 instance, he wrote a Python script that uses boto3 to do the job. The code uses the Boto3 library to connect to the Amazon EC2 service and create a new EC2 instance. It specifies the image (AMI) ID, instance type, key pair name, and tags for instance.
3. To create an S3 bucket, he wrote a Python script that uses boto3 to do the job. The code uses the Boto3 library to connect to the Amazon and the code creates an S3 bucket with the name "awsassignbucket1" in the "eu-west-3" region and then lists and prints the names of all existing buckets.
4. He had written the necessary import and initiated the required instances and created a handler function that does the following things:
    a. Generates a JSON object representing transaction data with various attributes.
    b. Creates a file name for the JSON file based on the current timestamp.
    c. Uploads the JSON data to the specified S3 bucket and file name.
    d. Logs the creation of the S3 object using CloudWatch Logs.
    e. Determines the number of times the Lambda function has been invoked
    f. Stops the execution of the Lambda function after the third run.
5. He had written the necessary import and initiated the required instances and created a handler function that does the following things:
    a. Parses the input data from the event parameter.
    b. Generates a timestamp using the current datetime, Adds the timestamp to the input data.
    c. Converts the modified input data to a JSON string.
    d. Uploads the JSON data to the specified S3 bucket and file name.
    e. Prints a log message indicating the successful creation of the S3 object, Returns a response object with the file name and success status.
6. He had used Postman for sending the file using a local machine.

Amit Shukla

He is currently working on his assignment.