Lab 4 – Lambda Integration with IAM, EC2, and CloudWatch

Create an EC2 Instance

1. Log into the AWS console and verify that the N. Virginia region is selected
2. In the services search text box, type in **EC2**
3. Click **EC2**
4. Click **Instances**
5. Click **Launch Instances**
6. Name: `Snaptest`
7. This EC2 instance **must** be created in the default VPC for the N. Virginia region.
8. Expand **Advanced details**
    a. Paste the following text into the User data area:
    ```
    #!/bin/bash
    yum update -y
    yum install httpd -y
    systemctl enable httpd.service
    systemctl start httpd.service
    echo "<html><h1>Lambda Test Success</h1></html>" >
    /var/www/html/index.html
    ```
9. Scroll to the bottom and click **Launch Instance**
10. Either use an existing key pair that you have access to, or create a new one.
11. Click **Launch Instance**
12. Click **View All Instances**

Configure IAM Role

13. In the services search text box, type in **IAM**
14. Click **IAM**
15. Click **Roles**
16. Click **Create Role**
    a. Trusted entity type: **AWS service**
    b. Use case: **Lambda**
    c. Click **Next**
    d. Do not add any permissions
    e. Click **Next**

   f. Role name: LambdaSnap
   g. Review **Step 1: Select trusted entities**
      i. Notice that the action is going to allow the principal to assume this IAM role.
      ii. The principal is the Lambda service.
   h. Click **Create role**
17. Find the role you just created (LambdaSnap) and click the name
18. Under **Permissions**, click **Add permissions**
   a. Choose **Create inline policy**
   b. Click **JSON**
      i. Erase the existing text
      ii. Paste the following text:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:*"
            ],
            "Resource": "arn:aws:logs:*:*:*"
        },
         {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateSnapshot",
                "ec2:DeleteSnapshot",
                "ec2:Describe*",
                "ec2:CreateTags",
                "ec2:ModifySnapshotAttribute",
                "ec2:ResetSnapshotAttribute"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

The role we created earlier can be assumed by Lambda to use the permissions in this inline policy.

The policy allows Lambda to describe EC2 instances, take snapshots, delete snapshots and more.

iii. Click **Next**
iv. Policy name: `TrainingLambda`
v. Click **Create policy**

Create a Lambda Function to Take Daily Snapshots

19. In the services search text box, type in **Lambda**
20. Click **Lambda**
21. Click **Functions**
22. Click **Create Function**
    a. Select Author from scratch
    b. Function name: `EC2Snapshots`
    c. Runtime: Python 3.12
    d. Click **Change default execution role**
        i. Use an existing role
        ii. Existing role: LambdaSnap
        This will allow this Lambda function to take snapshots of EC2 instances
    e. Click **Create function**
23. Once the Lambda function is created, scroll down and view the auto-populated code.
24. Replace the code with the following text:
    NOTE: Make sure that when you paste in the code, it is properly indenting the lines as shown below

```
# Backup all in-use volumes in all regions

import boto3

def lambda_handler(event, context):
    ec2 = boto3.client('ec2')

    # Get list of regions
    regions = ec2.describe_regions().get('Regions',[] )

    # Iterate over regions
    for region in regions:
        reg=region['RegionName']

        # Connect to region
        ec2 = boto3.client('ec2', region_name=reg)

        # Get all in-use volumes in all regions
        result = ec2.describe_volumes( Filters=[{'Name': 'status', 'Values': ['in-use']}])

        for volume in result['Volumes']:

            # Create snapshot
            result = ec2.create_snapshot(VolumeId=volume['VolumeId'],Description='Created by
Lambda backup function ebs-snapshots')

            # Get snapshot resource
            ec2resource = boto3.resource('ec2', region_name=reg)
            snapshot = ec2resource.Snapshot(result['SnapshotId'])

            volumename = 'N/A'

            # Find name tag for volume if it exists
            if 'Tags' in volume:
                for tags in volume['Tags']:
                    if tags["Key"] == 'Name':
                        volumename = tags["Value"]

            # Add volume name to snapshot for easier identification
            snapshot.create_tags(Tags=[{'Key': 'Name','Value': volumename}])
```

25. Click **Deploy**

    The image for this lambda function is going to be updated to include this code. Any time I make changes to the code in my lambda function, deploy pushes those changes live. Deploy saves the updated code to the lambda function.

26. Click the **Configuration** tab
    a. On the left, click **General configuration**
    b. Click **Edit**
    c. Change the timeout to **1 minute**
    d. Click **Save**

       It is going to take more than a minute to take snapshots of EC2 instances. This will allow the function to complete before it times out.

    e. On the left, click **Triggers**
    f. Click **Add trigger**
       i. Select a source: **EventBridge (CloudWatch Events)**
       ii. Rule: **Create a new rule**
       iii. Rule name: `DailyEC2Snap`
       iv. Rule type: **Schedule expression**
       v. Schedule esxpression: `rate(1 day)`
       vi. Click **Add**
27. Click the **Test** tab (To the right of **Code**)
    a. Create new event
    b. Event name: `test2`
    c. Private should be selected by default.
    d. Template: Choose **Cloudwatch**

       This test will invoke the lambda function using a specific service. There are many services that could potentially invoke this lambda function. This test is going to invoke this lambda function as if it was getting invoked by cloud watch.

       i. Click **Save**
    e. Here you can see your saved test event that you can run repeatedly.
       i. Click **Test**

       When you execute a test you are actually invoking the lambda function.

28. It should take about 30 seconds for the function to complete.
29. You should see a green check above the test event.

Verify Snapshots were Taken

30. In the services search text box, type in **EC2**
31. Click **EC2**

32. Click **Snapshots**
    a. There should be 1 snapshot with the following description:
    Created by Lambda backup function ebs-snapshots

Clean Up Scheduled Function

33. In the services search text box, type in **Lambda**
34. Click **Lambda**
35. Click **Functions**
36. Select the check box for EC2Snapshots
    a. Actions > Delete