Automated Cost Optimization for EBS Volumes and Snapshots in AWS Using the Console

By Philip Essel,
Certified Cloud Solutions Architect:
philipessel2006@gmail.com

Reason for Doing Project

The primary goal of this project is to build an AWS cost optimization architecture that automatically identifies and deletes stale EBS volumes and snapshots once they are no longer attached to EC2 instances. By automating this process, you can avoid unnecessary storage costs that accumulate from unused resources, improve efficiency, and ensure that AWS accounts remain optimized in terms of both performance and cost. This project will help reduce operational overhead by leveraging AWS tools to monitor, alert, and act on cost-saving opportunities.

Additionally, this project provides hands-on experience with AWS services such as Lambda, EventBridge, SNS Topic, Cost Explorer, CloudWatch logs, AWS Budgets and IAM.

Project Overview

This project involves the creation of a cost optimization architecture using several AWS services. It automates the identification of terminated instances and deletion of stale Elastic Block Store (EBS) volumes and their associated snapshots. The solution relies on monitoring instance states (termination) using EventBridge, monitoring volumes and Snapshots using EventBridge and Lambda, triggering automation through Lambda and notifying administrators using Amazon SNS. AWS Cost Explorer and AWS Budgets will be used to monitor and track cost savings over time.

The architecture will operate as follows:

When an instance is terminated, EventBridge will detect the instance state and trigger Lambda function to send SNS notification to admin. A second EventBridge will be scheduled to trigger another Lambda function to query EBS volumes and snapshots on weekly basis. Anytime Lambda finds an EBS volume and Snapshot that have not been attached to instance for more than 7 days (stale resources), it will delete them. This same function will then send notification to administrator about the deletion of these stale resources and then create logs of them in CloudWatch logs. AWS Cost Explorer and AWS Budgets will at the same time be used to monitor and track cost savings over time.

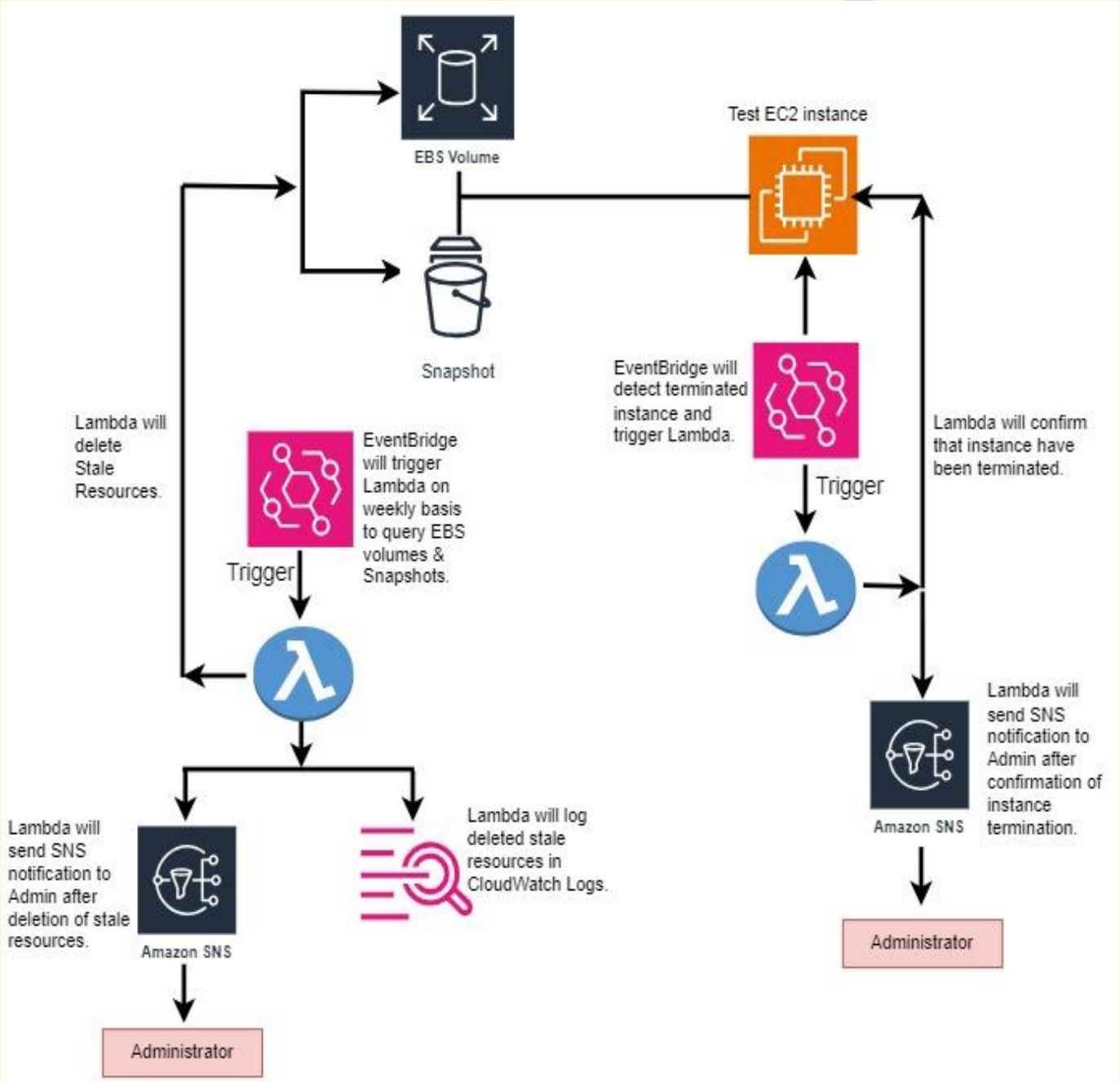
Technologies Used

Technologies and services used for this project include:

1. **EC2**: The project will detect instances that are terminated.
2. **AWS Lambda**: Automates the process of detecting the states of EC2 Instances (termination), checking stale volumes/Snapshots and deleting them.
3. **Amazon EventBridge**: Will trigger Lambda function for automation.
4. **Amazon SNS**: Sends notifications to administrators when Instances are terminated, EBS volumes and Snapshots are deleted and budget thresholds are reached or exceeded.
5. **EBS volumes**: The project deals with managing and deleting EBS volumes to save costs.

6. **EBS Snapshots**: The project also deals with managing and deleting EBS snapshots to save costs.
7. **AWS Cost Explorer/Budgets**: Helps in tracking cost savings and setting up budgets for volume usage.
8. **AWS IAM**: Manages permissions for the Lambda functions.

Architectural Diagram



Project Steps

The following are the detailed steps for this project using the AWS Console.

Step 1: Create SNS Topics for Notifications

***Note**: We will create 3 SNS topics. Two of these SNS topic will later be configured separately in our Lambda functions. The remaining SNS topic will be configured in our AWS Budget.*

Create first SNS topic. This SNS topic will be responsible for notifications related to EC2 instance terminations. Once EC2 instances are terminated, EventBridge will trigger Lambda to send instant notification to admin.

1. Create an SNS Topic
    - Navigate to **Amazon SNS**.
    - In the **Amazon SNS Dashboard**, select **Topics** on the left panel, then click **Create topic**.
    - Under **Create topic**, choose **Standard** as the topic type.
    - **Enter a name** for your topic (Eg. EC2-Instance-Termination-Notification).
    - Click **Create topic**.
    - Copy the ARN of this SNS topic. We will be needing it whiles configuring our Lambda function. Our Lambda function can only access the SNS topic through this ARN.
2. Subscribe to the Topic by Email
    - After creating the topic, you'll be redirected to the topic's **Details** page.
    - Under **Subscriptions**, click **Create subscription**.
    - In the **Create subscription** section:
        - For **Protocol**, select **Email**.
        - For **Endpoint**, enter your email address where notifications will be sent (Eg. philipessel2006@gmail.com).
    - Click **Create subscription**. An email will be sent to the provided address with a confirmation link.
3. Confirm Email Subscription
    - Open your email inbox and look for an email from **AWS Notifications**.
    - Click the **Confirm subscription** link in the email. This will verify and activate your subscription to the SNS topic.
    - After confirming, you'll see a message that your subscription has been confirmed. You should now be able to receive notifications sent to this topic.

Create Second SNS topic. This SNS topic will be responsible for notifications related to the periodic cleanup of stale EBS volumes and Snapshots. Amazon EventBridge will be scheduled to trigger a second Lambda function which will query EBS volumes and snapshots on weekly basis to determine which of them has not been attached to an instance for more than 7 days. Anytime it finds one, it will delete them. The function will then send notification to administrator about the deletion of these stale resources.

1. Create an SNS Topic
    - Navigate to **Amazon SNS**.
    - In the **Amazon SNS Dashboard**, select **Topics** on the left panel, then click **Create topic**.
    - Under **Create topic**, choose **Standard** as the topic type.
    - **Enter a name** for your topic (Eg. Stale-Volume-Snapshot-Cleanup-Notifications).
    - Click **Create topic**.
    - Copy the ARN of this SNS topic. We will be needing it whiles configuring our second Lambda function. Our Lambda function can only access this SNS topic through this ARN.
2. Subscribe to the Topic by Email
    - After creating the topic, you'll be redirected to the topic's **Details** page.
    - Under **Subscriptions**, click **Create subscription**.
    - In the **Create subscription** section:
        - For **Protocol**, select **Email**.
        - For **Endpoint**, enter your email address where notifications will be sent (Eg. philipessel2006@gmail.com).
    - Click **Create subscription**. An email will be sent to the provided address with a confirmation link.
3. Confirm Email Subscription
    - Open your email inbox and look for an email from **AWS Notifications**.
    - Click the **Confirm subscription** link in the email. This will verify and activate your subscription to the SNS topic.
    - After confirming, you'll see a message that your subscription has been confirmed. You should now be able to receive notifications sent to this topic.

Create Third SNS topic. This SNS topic will be responsible for notifications coming from AWS Budgets. Lambda will not be involved in triggering this SNS notification. Basically, when spending for the selected services approaches or exceeds our specified thresholds or our budget, AWS Budget will send notification to this SNS topic.

1. Create an SNS Topic

- Navigate to **Amazon SNS**.
- In the **Amazon SNS Dashboard**, select **Topics** on the left panel, then click **Create topic**.
- Under **Create topic**, choose **Standard** as the topic type.
- **Enter a name** for your topic (Eg. Budget-Resources-Notifications).
- Click **Create topic**.
- In this case we will not need to copy the ARN of this SNS topic.

2. Subscribe to the Topic by Email
   - After creating the topic, you'll be redirected to the topic's **Details** page.
   - Under **Subscriptions**, click **Create subscription**.
   - In the **Create subscription** section:
     - For **Protocol**, select **Email**.
     - For **Endpoint**, enter your email address where notifications will be sent (Eg. philipessel2006@gmail.com).
   - Click **Create subscription**. An email will be sent to the provided address with a confirmation link.
3. Confirm Email Subscription
   - Open your email inbox and look for an email from **AWS Notifications**.
   - Click the **Confirm subscription** link in the email. This will verify and activate your subscription to the SNS topic.
   - After confirming, you'll see a message that your subscription has been confirmed. You should now be able to receive notifications sent to this topic.

Step 2: Create IAM ROLE (Lambda Role)

***Note**: We will be creating two IAM roles for two separate Lambda functions. The first IAM role will have permissions to enable our first Lambda function to send SNS notifications to Amin when an EC2 instance is terminated.*

The second IAM role will have permissions to enable our second Lambda function to delete stale EBS volumes and snapshots (resources that have not been attached to instances for more than 7 days) and then send SNS notification informing Admin about the deletion. It will also enable our function to create logs of our deleted resources in CloudWatch logs.

1. First IAM Role.
   - Create an IAM Role for First Lambda
     - Go to **IAM** by searching for it in the AWS Console search bar.

- In the left sidebar, select **Roles**.
- Click **Create role**.
- Under **Select trusted entity**, choose **AWS service**.
- Choose **Lambda** as the use case.
- Click **Next** to proceed to permissions.

- Attach Required Permissions to the Role

Note: The role requires policies to manage EC2 instances and SNS notifications. We will add 2 AWS managed policies. Managed policies are already-made policies provided by AWS. We will just search for them by name and attach them to our role. They are **AmazonEC2ReadOnlyAccess** (Will allow Lambda to view our EC2 instance states) and **AmazonSNSFullAccess** (Will grant full access to publish notifications through SNS).

- In the **Add permissions** page, search for each policy by name and select the checkbox next to it.
- After selecting these policies, proceed to **Next**.

- Review and Create the Role

- After attaching our managed policies, click **Next**.
- Enter a **Role name:** (Eg. Instance-Termination-Lambda-Trigger-Role).
- Add an optional **description** (Eg. "Allows Lambda function to confirm EC2 instance termination and publishes notifications to SNS about the instance).
- Review the policies attached.
- Click **Create role**.
- Our role is now ready for use. We will attach this role to our first Lambda function when configuring it.

2. Second IAM Role.

- Create an IAM Role for Second Lambda

- Navigate to **IAM**.
- In the left sidebar, select **Roles**.
- Click **Create role**.
- Under **Select trusted entity**, choose **AWS service**.
- Choose **Lambda** as the use case.

- Click **Next** to proceed to permissions.

- Attach Required Permissions to the Role

Note: The role requires policies to manage our EC2 instances resources (EBS volumes and snapshots), SNS notifications, and CloudWatch Logs. For this role we will add 3 AWS managed policies. Let's just search for them by name and attach them to our role. They are **AmazonEC2FullAccess** (Will allow Lambda to delete EBS volumes and snapshots), **CloudWatchLogsFullAccess** (Will grant full access to create and write to CloudWatch Logs) and **AmazonSNSFullAccess** (Will grant full access to publish notifications through SNS).

- In the **Add permissions** page, search for each policy by name and select the checkbox next to it.
- After selecting these policies, proceed to **Next**.

- Review and Create the Role

- After attaching policies, click **Next**.
- Enter a **Role name:** (Eg. StaleResourceCleanUp-Lambda-Trigger-Role).
- Add an optional **description** (Eg. "Allows Lambda function to delete stale EBS volumes and snapshots, publish notifications to SNS and log in CloudWatch logs).
- Review the policies attached to ensure they match the required permissions.
- Click **Create role**.
- Our role is now ready for use. We will attach this role to our second Lambda function when configuring it.

Step 3: Write our First Lambda Function for the Application

***Note**: We will call our first Lambda function, Instance-Termination-Function. Anytime an EC2 instance is terminated, EventBridge will detect it and trigger this function to send SNS notification to Admin.*

1. First, Create Function.
   - Click on Create Function.
   - Choose **Author from Scratch**.
   - **Name**: Give your function a name (Eg. Instance-Termination-Function).
   - **Runtime**: Choose **Python**.
   - **Architecture**: Choose **x86_64** (Cheapest option).
   - **Permissions**: Choose **change default execution role**.

- Select **Use an existing role**.
- Attach the IAM role we created (Instance-Termination-Lambda-Trigger-Role).
- Select **Create function**.

2. Second, Configure Environment Variables in the Lambda function:

   Note: Our Lambda function will be able to send notifications to our SNS topic through the topic's ARN. But we want to avoid hardcoding this value directly into our function code. So we will use Environment Variables to achieve this. We will simply use this variable to store the ARN of our SNS topic. This will make our function more flexible and easier to manage, as you can update environment variables without editing the code.

   - Navigate to the Lambda function you created (Instance-Termination-Function).
   - In the Lambda function dashboard, click on the **Configuration** tab.
   - In the Configuration panel, select **Environment variables**.
   - Click on **Edit** and then **Add environment variable**.
   - Add our key-value pairs for the environment variables:

     Key-Value Pair

     - **Key**: SNS_TOPIC_ARN
     - **Value**: Paste the ARN of the **EC2-Instance-Termination-Notification** SNS topic.
   - After adding the environment variables, click **Save** to apply them to the Lambda function.

3. Third, Paste Lambda Function Code (In Python).
   - In the Lambda function dashboard, click on the **Code** tab.
   - Replace the default code with the attached python code (**Instance Termination Lambda Function Code**).
   - Save the Lambda function.

4. Forth, Test the Function
   - Click the **Test** tab (Adjacent to Code tab).
   - Choose **Create new event**.
   - Name the event (Eg, TestEvent).
   - Choose the **Hello World** template.

     ```
     {
       "key1": "value1",
     ```

```
    "key2": "value2",
    "key3": "value3"
}
```

- Click **Save**.
- Click the **Code** tab.
- At **Test** tab (Adjacent to Deploy tab), select our test event (TestEvent)
- Click the **Deploy** tab.
- Click the **Test** (Adjacent to Deploy tab) to invoke our test Lambda function.
- After running the test:
  - Check your mail to confirm that an email notification has been sent to you.
  - Check CloudWatch Logs: Look at the logs generated by the Lambda function to verify actions taken.

Step 4: Create & Configure First EventBridge to Trigger First Lambda Function When Instance is terminated.

***Note**: We will create a rule the will make EventBridge detect terminated EC2 instances and trigger our Lambda function (Instance-Termination-Function) to send SNS notification to Admin about the termination.*

1. Go to the EventBridge Console
   - Navigate to **Amazon EventBridge**.
   - In the left-hand menu, click **Rules**.
   - Click on **Create rule** to start setting up a new rule.
2. Define Rule Details
   - Name and Description:
     - Enter a **name**: (Eg. Manage-EC2-Terminations).
     - Add a **description**: (Eg. "Triggers Lambda for EC2 instance terminations.").
   - Define Rule Type:
     - Select **Event pattern**.
3. Set the Event Pattern for EC2 Termination Events
   - Under **Event source**, choose **AWS services**.

- For **Service name**, select **EC2**.
- For **Event type**, choose **EC2 Instance State-change Notification**.
- Next, select **Specific state(s)**.
- Next, choose **terminated** to capture only termination events for EC2 instances.
- Next, select **Any instance**.
- Your event pattern preview should look like this:

```
{
  "source": ["aws.ec2"],
  "detail-type": ["EC2 Instance State-change Notification"],
  "detail": {
    "state": ["terminated"]
  }
}
```

4. Add the Lambda Function as a Target
   - Scroll down and click **Add target**.
   - In the **Target type** dropdown, choose **AWS service** and then select **Lambda function**.
   - In the **Function** dropdown, select the Lambda function you created for this EventBridge (Instance-Termination-Function).
5. Review and Create the Rule
   - Review the Configuration:
     - Verify the rule name, event patterns, target Lambda function, and permissions.
   - Click Create Rule:
     - Click **Create** to enable the rule.

Our EventBridge rule is now configured to trigger our first Lambda function. What this means is that, anytime an EC2 instance is terminated, EventBridge will detect it and trigger Lambda to confirm the termination and send notification through SNS topic to Admin about the instance.

6. Verify the Configuration
   - **EC2 Termination Test**:
     - Create a test EC2 instance and terminate it to verify that the Lambda function triggers.
   - **SNS Notifications Check**:
     - Check your SNS topic (EC2-Instance-Termination-Notification) for notifications based on the event source. You can equally check your mail for notification from SNS topic.

Note: When you create EventBridge and you attach Lambda to it as target, EventBridge will automatically have access to invoke the Lambda function without requiring an explicit IAM role attached to the EventBridge. This is because EventBridge and Lambda are tightly integrated services. When you create an EventBridge rule with Lambda target, EventBridge uses the Lambda API to invoke the function.

You can use these steps to verify permissions that EventBridge have to invoke Lambda:

- Open the **Lambda Console** in a new tab and navigate to the Lambda function that was targeted by our EventBridge (Instance-Termination-Function).
- Under **Configuration > Permissions**, click on the **Resource-based policy** to view the permissions EventBridge added. You should see an InvokeFunction permission for EventBridge.
- In the unlikely situation that permissions aren't automatically added, manually add the following policy to the Lambda function's **Resource-based policy**:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "events.amazonaws.com"
      },
      "Action": "lambda:InvokeFunction",
      "Resource": "arn:aws:lambda:<region>:<account-id>:function:<function-name>"
    }
  ]
}
```

Replace <region>, <account-id>, and <function-name> with your AWS region, account ID, and Lambda function name respectively.

Step 5: Write Second Lambda Functions for the Application

***Note**: We will call this Lambda function, Stale-Resource-CleanUp-Function. It will be configured with a second EventBridge. EventBridge will be scheduled to trigger a second Lambda function that will query EBS volumes and snapshots on weekly basis. Anytime the function finds EBS volumes and Snapshots that have not been attached to instances for more than 7 days (stale resources), this Lambda function will delete them. It will then send SNS notification to Admin about the deletion of the stale resources and then create logs of them in CloudWatch logs.*

1. First, Create Function.
    - Click on Create Function.
    - Choose **Author from Scratch**.
    - **Name**: Give your function a name (Eg. Stale-Resource-CleanUp-Function).
    - **Runtime**: Choose **Python**.
    - **Architecture**: Choose **x86_64**.
    - **Permissions**: Choose **change default execution role**.
    - Select **Use an existing role**.
    - Attach the IAM role we created (StaleResourceCleanUp-Lambda-Trigger-Role).
    - Select **Create function**.
2. Second, Configure Environment Variables in the Lambda function (Stale-Resource-CleanUp-Function):
    - In the Lambda function dashboard, click on the **Configuration** tab.
    - In the Configuration panel, select **Environment variables**.
    - Click on **Edit** and then **Add environment variable**.
    - Add key-value pairs for the environment variables:

        Key-Value Pair

        - **Key**: SNS_TOPIC_ARN
        - **Value**: Paste the ARN of the **Stale-Volume-Snapshot-Cleanup-Notifications** SNS topic.
    - After adding the environment variables, click **Save** to apply them to the Lambda function.
3. Third, Paste Lambda Function Code (In Python).
    - In the Lambda function dashboard, click on the **Code** tab.
    - Replace the default code with the attached python (**Stale Resource CleanUp Lambda Function Code**).
    - Save Lambda function.

5. Forth, Test the Function
    - Click the **Test** tab (Adjacent to Code tab).
    - Choose **Create new event**.
    - Name the event (Eg, TestEvent).
    - Choose the **Hello World** template.

```
{
  "key1": "value1",
  "key2": "value2",
  "key3": "value3"
}
```

- Click **Save**.
- Click the **Code** tab.
- At **Test** tab (Adjacent to Deploy tab) select our test event (TestEvent)
- Click the **Deploy** tab.
- Click the **Test** (Adjacent to Deploy tab) to invoke our test Lambda function.
- After running the test:
  - Check your mail to confirm that an email notification has been sent to you.
  - Check CloudWatch Logs: Look at the logs generated by the Lambda function to verify actions taken.

Step 6: Create & Schedule Second EventBridge to Trigger Second Lambda Function

Note: *At this point, we will create and configure another EventBridge to trigger the second Lambda function (Stale-Resource-CleanUp-Function). This second EventBridge will be scheduled to trigger Lambda weekly to enable Lambda detect EBS volumes and Snapshots that have not been attached to instances for more than 7 days. When Lambda finds any such resource, it will delete them. Our Lambda function will again send an SNS notification to Admin about the deletions.*

1. Go to the EventBridge Console
   - Navigate to **Amazon EventBridge**.
   - In the left-hand menu, click **Rules**.
   - Click on **Create rule** to start setting up a new rule.
2. Define Rule Details
   - Name and Description:
     - Enter a **name**: (Eg. Weekly-Stale-Resource-CleanUp-Rule).
     - Add a **description**: (Eg. "Triggers Lambda for Cleanup of stale EBS volumes and snapshots, logging of deletions and sending notifications to Admin.").

- Define Rule Type:
    - Choose **Schedule** for this rule.
    - Select **Next**.
    - Select **Cron expression** to set the schedule.
    - Enter a cron expression for a weekly trigger. The cron expression below will trigger Lambda every Sunday at midnight UTC:

      cron(0 0 ? * SUN *)

      Breakdown of this cron expression
      0 – The minute when the event should run (0 means it will run on the hour).

      0 – The hour when the event should run (0 means midnight).

      ? – The day of the month; using ? means it's not specified (you either specify a day of the month or a day of the week, not both).

      * – The month; * means every month.

      SUN – The day of the week; SUN means it will only run on Sundays.

      * – The year; * means every year.

    - Click **Next**.

3. Add the Lambda Function as a Target
    - In the **Target type** dropdown, choose **AWS service**.
    - Then select **Lambda function**.
    - In the **Function** dropdown, select the Lambda function you created for this EventBridge (Stale-Resource-CleanUp-Function).
4. Review and Create the Rule
    - Click **Next**.
    - Click **Next**.
    - Verify the rule name, event patterns, target Lambda function, and permissions.
    - Click **Create** to enable the rule.

Our second EventBridge rule is now configured to trigger our second Lambda function. What this means is that, EventBridge will trigger our Lambda function weekly (on Sunday at midnight). The function will check for EBS

volumes and snapshots that have not been attached to instances for more than 7 days (stale resources), delete any found stale resources, send notifications to admins and log the actions in CloudWatch Logs.

5. Verify the Configuration
    - Follow steps below to temporarily change cron expression to cron(* * * * ? *). This cron expression will trigger Lambda every minute. Remember to revert back to the initial cron expression (cron(0 0 ? * SUN *)) using same steps below since triggering an event every minute can lead to high costs or excessive logging. We are doing this for just test purpose.
        - Navigate Amazon **EventBridge** Console and select the **region** where your rule is located.
        - In the left sidebar, click on **Rules** to view all your existing EventBridge rules.
        - Find and click on the rule you want to modify (Weekly-Stale-Resource-CleanUp-Rule).
        - On the rule details page, click the **Edit** tab at the top right.
        - In the Define pattern section, choose Event Source as **Schedule**.
        - Select **Cron expression** in the schedule options
        - Enter our test cron expression (cron(* * * * ? *)).
        - Click **Update** to save your changes.
        - Your rule will now run according to the new cron expression you specified.
    - **Check SNS Notifications** to confirm a message is sent after stale resources are deleted.
    - **Verify CloudWatch Logs** for the Lambda function to see a log of deleted EBS volumes and snapshots.

      Note: Remember to revert back to the initial cron expression (cron(0 0 ? * SUN *)) using same steps above. Per minute invocation can be costly.

Step 7: Track Cost Savings with AWS Cost Explorer and Budgets

***Note**: We will use AWS Cost Explorer and Budgets to help us track the effectiveness of the project in reducing costs. Our AWS Cost Explore will generate reports and visualize cost trends over time. This will help identify services consuming the most budget. Our AWS Budget will set up budget alarms to receive notifications when spending approaches or exceeds predefined thresholds, ensuring proactive cost management. The services we will be tracking in this project are EC2, Lambda, EBS, CloudWatch and IAM.*

1. Setting Up AWS Cost Explorer to Monitor Multiple Services
    - Navigate to **Billing and Cost Management**.
    - Select **Cost Explorer** from the left sidebar.

- If not already enabled, click **Enable Cost Explorer** to activate it.
- Once in Cost Explorer, select **Cost Explorer** again to open the report creation tool.
- Click on **Create report** to start a new custom report.
- Filter by Service:
    - Under **Filters**, click on **Service** and select the services we want to track (EC2, Lambda, EBS, CloudWatch, EventBridge and IAM).
- Choose Time Range and Granularity:
    - Set a **Time range** (Eg. last 6 months) to analyze historical data.
    - Select a **Granularity** (Daily, Monthly, or Hourly) based on how closely you want to monitor cost changes.
- View By:
    - Set **View By** to **Usage Type** to get a breakdown of costs by individual service components (Eg. EC2-Compute, EBS-Storage).
- Once you have the report set up, click **Save As** and name the report (Eg. Multi-Service Cost Tracking).
- Regularly check your report from **Saved Reports** to view trends for EC2, Lambda, EBS, CloudWatch and IAM.

2. Setting Up AWS Budgets to Monitor Multiple Services.
    - In the **AWS Management Console**, go to **Billing and Cost Management**.
    - Select **Budgets** from the left sidebar.
    - Click **Create a budget**.
    - Choose **Cost budget** as the budget type, which will track your spending.
    - **Set a Budget Name**: Enter a name (Eg. My-AWS-Resources-Cost-Budget).
    - **Period**: Select a budget period (Eg. Monthly) to align with your cost review cycle.
    - **Amount**: Define the total budget limit (Eg. $500 per month). This is the threshold that will trigger notifications.
    - **Filter by Service**:
        - In **Advanced Options**, use the **Service** filter to select EC2, Lambda, EBS, CloudWatch, EventBridge and IAM only. This will limit the budget to these services.
    - Define Alert Thresholds:
        - Set up alert thresholds (Eg. 80% of the budgeted amount) to receive warnings as you approach the budget limit.

- Choose Notification Settings:
    - Under **Notification settings**, select **Send alert to an Amazon SNS topic**.
    - Choose an existing SNS topic (Budget-Resources-Notifications).
- Review all settings and click **Create budget** to finalize.
- You will receive email notifications from AWS Budgets when spending for the selected services (EC2, Lambda, EBS, CloudWatch, EventBridge and IAM) approaches or exceeds the specified thresholds. You can also adjust your budget or thresholds as needed based on usage trends observed in Cost Explorer.

END
