# **Automated Scheduled EC2 Start/Stop using AWS Lambda and EventBridge**

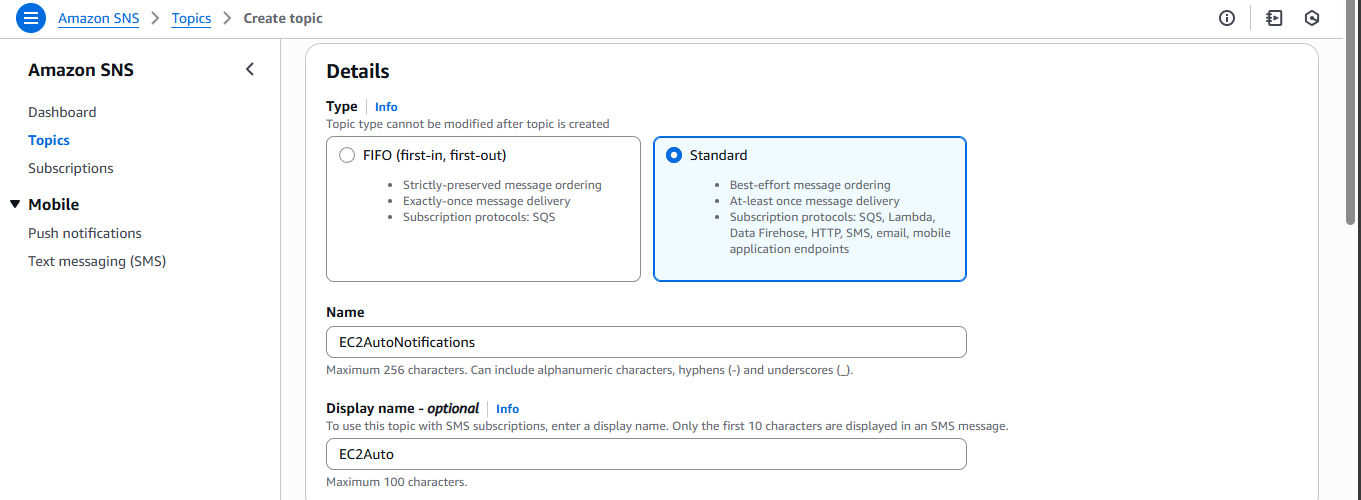
## **Overview – Part B**

Extending it further, we’ll now generate email notifications every time the start/stop lambda function is triggered. For generating email notifications, we’ll use Amazon SNS.

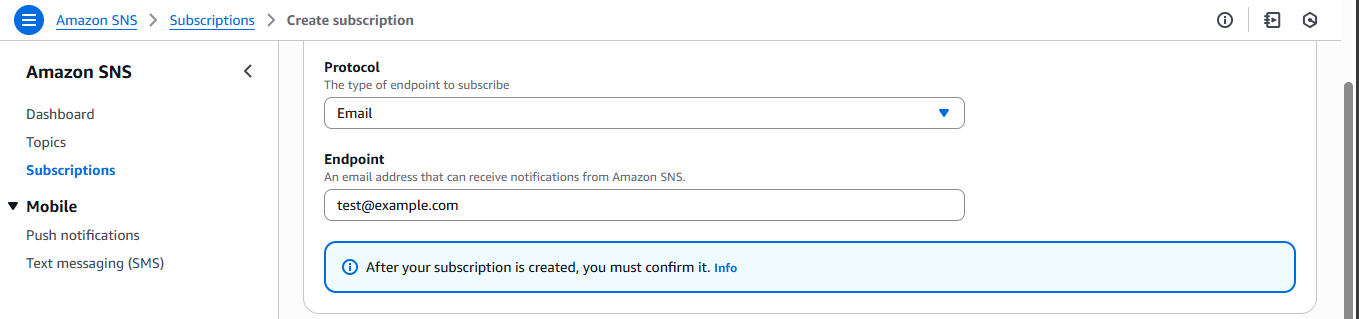
## **Step-by-Step Implementation**

### Create a Topic

* Here, as we can see, we need to choose the type either FIFO or standard. The basic difference between these types is that **FIFO topics** preserve the order of messages and ensure each message is delivered exactly once while **Standard topics** don’t guarantee message order and may deliver the same message more than once. For  
  this task, I have chosen standard type.

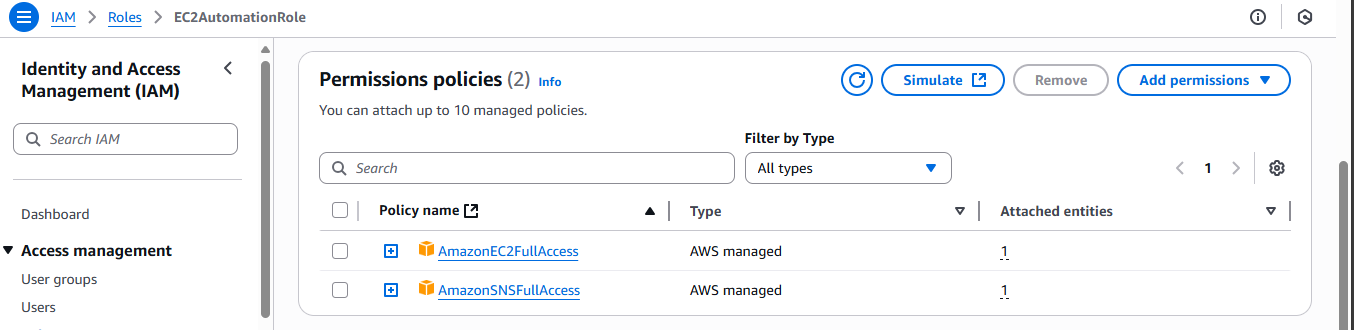


### Create a Subscription

* We need to choose a protocol (Email for this task) and endpoint – the desired email ID. 

### Update the IAM role

* It is necessary to update the ‘EC2AutomationRole’ and provide the Amazon SNS Access Policy.



* I had provided broad access, for example, ‘AmazonSNSFullAccess’ and ‘AmazonEC2FullAccess’. This is not considered a best practice. We should always follow the principle of least privilege and only grant the exact permissions required to perform the task.
* To achieve this, we will create custom IAM policies and attach them to the EC2AutomationRole.

#### EC2StartStop Policy:

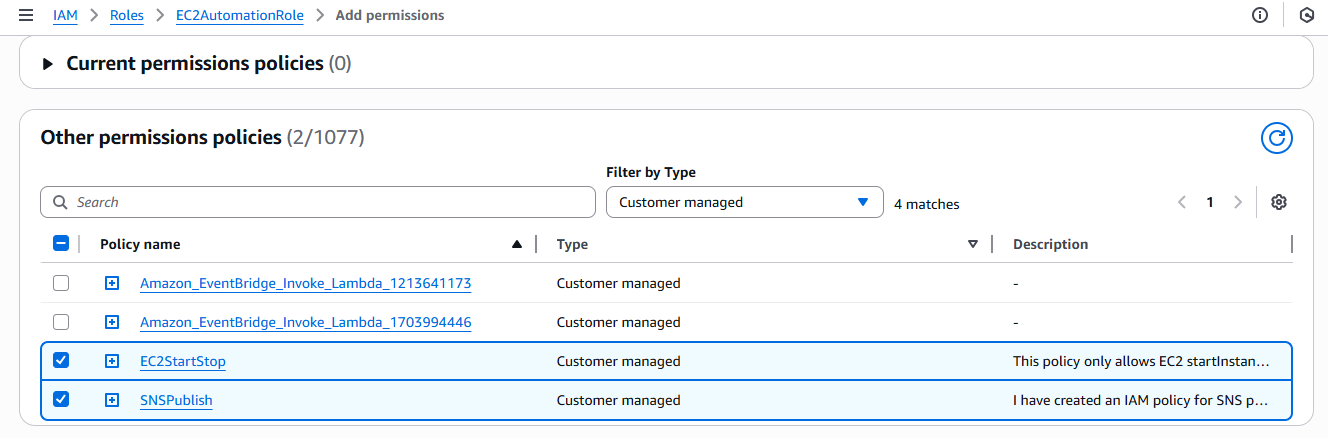
This policy provides permission only to start and stop a specific EC2 instance. Use the following JSON format

|  |
| --- |
| {  "Version": "2012-10-17",  "Statement": [  {  "Sid": "Statement1",  "Effect": "Allow",  "Action": [  "ec2:StartInstances",  "ec2:StopInstances"  ],  "Resource":"arn:aws:ec2:ap-south-1:598728965456:instance/i-0f2b64a491cd4ccd8"  }  ]  } |

#### SNSPublish Policy –

|  |
| --- |
| {  "Version": "2012-10-17",  "Statement": [  {  "Sid": "Statement1",  "Effect": "Allow",  "Action": "sns:Publish",  "Resource":"arn:aws:sns:ap-south1: 598728965456:EC2AutoNotifications"  }  ]  } |

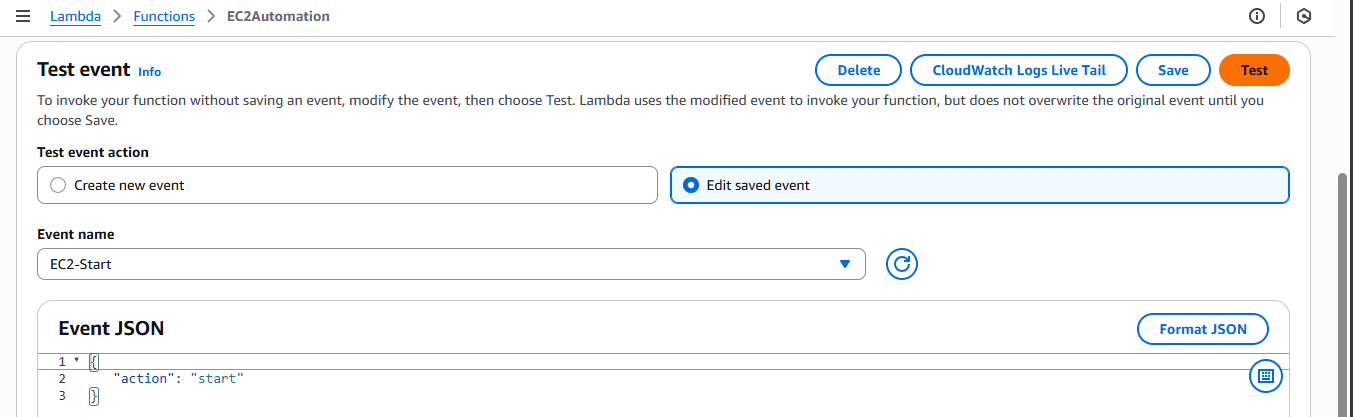
* Once created, attach them to the EC2AutomationRole.



### Update the lambda function

|  |
| --- |
| import json  import boto3  from datetime import datetime  def lambda\_handler(event, context):  ec2 = boto3.client('ec2')  sns = boto3.client('sns')  instance\_id = 'i-0f2b64a491cd4ccd8' # Your instance ID  action = event.get('action')  topic\_arn = 'arn:aws:sns:ap-south-1:522035495679:EC2AutoNotifications' # Replace with your SNS Topic ARN  response\_message = ''  try:  if action == 'start':  ec2.start\_instances(InstanceIds=[instance\_id])  response\_message = f'Started instance {instance\_id} at {datetime.utcnow()} UTC'  elif action == 'stop':  ec2.stop\_instances(InstanceIds=[instance\_id])  response\_message = f'Stopped instance {instance\_id} at {datetime.utcnow()} UTC'  else:  response\_message = 'No valid action provided'  # Publish notification to SNS  sns.publish(  TopicArn=topic\_arn,  Subject=f'EC2 Automation: {action.capitalize()} action executed',  Message=response\_message  )  print(response\_message)  except Exception as e:  error\_message = f'Error during EC2 {action} action: {str(e)}'  sns.publish(  TopicArn=topic\_arn,  Subject=f'EC2 Automation: Error during {action} action',  Message=error\_message  )  print(error\_message)  raise e |

After updating our lambda function, just test it once to check the expected output.



## Conclusion

By integrating AWS SNS with Lambda and EventBridge, we now receive timely email alerts every time an EC2 instance is started or stopped, ensuring transparency and immediate awareness of infrastructure state changes. This serverless and scalable architecture requires no manual intervention and improves operational efficiency, helping us stay informed and maintain better control over their AWS resources.

Below are screenshots of the email notifications received upon execution of the scheduled Lambda functions for starting and stopping the EC2 instance:

## 