

Project - 2: Publishing Amazon SNS Messages Privately

Problem Statement:

How to secure patient records online and send it privately to the intended party

Topics:

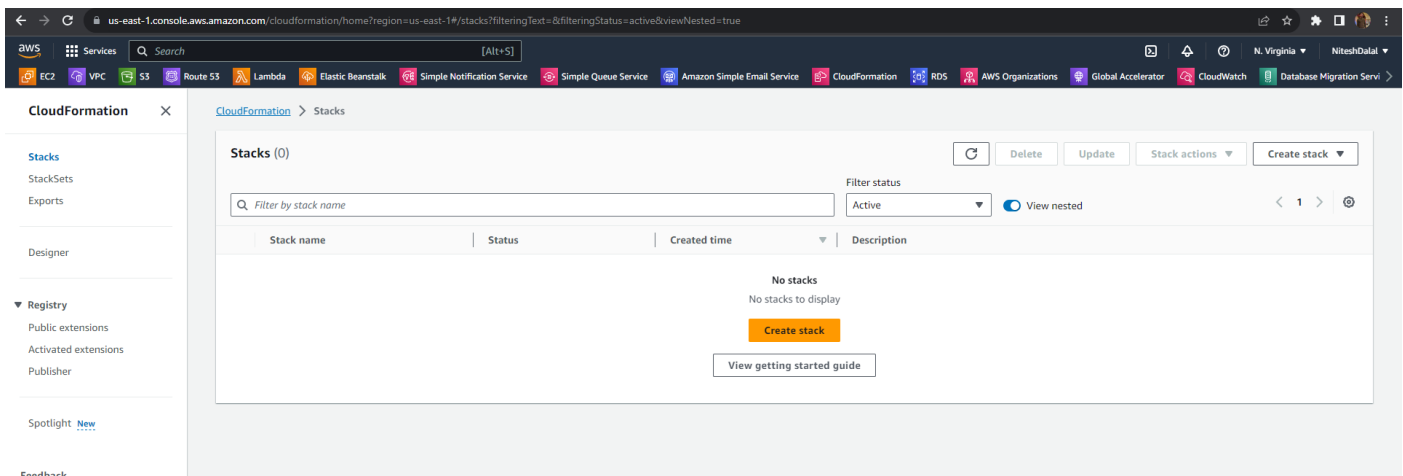
In this project, you will be working on a hospital project to send reports online and develop a platform so the patients can access the reports via mobile and push notifications. You will publish the report to an Amazon SNS keeping it secure and private. Your message will be hosted on an EC2 instance within your Amazon VPC. By publishing the messages privately, you can improve the message delivery and receipt through Amazon SNS.

Highlights:

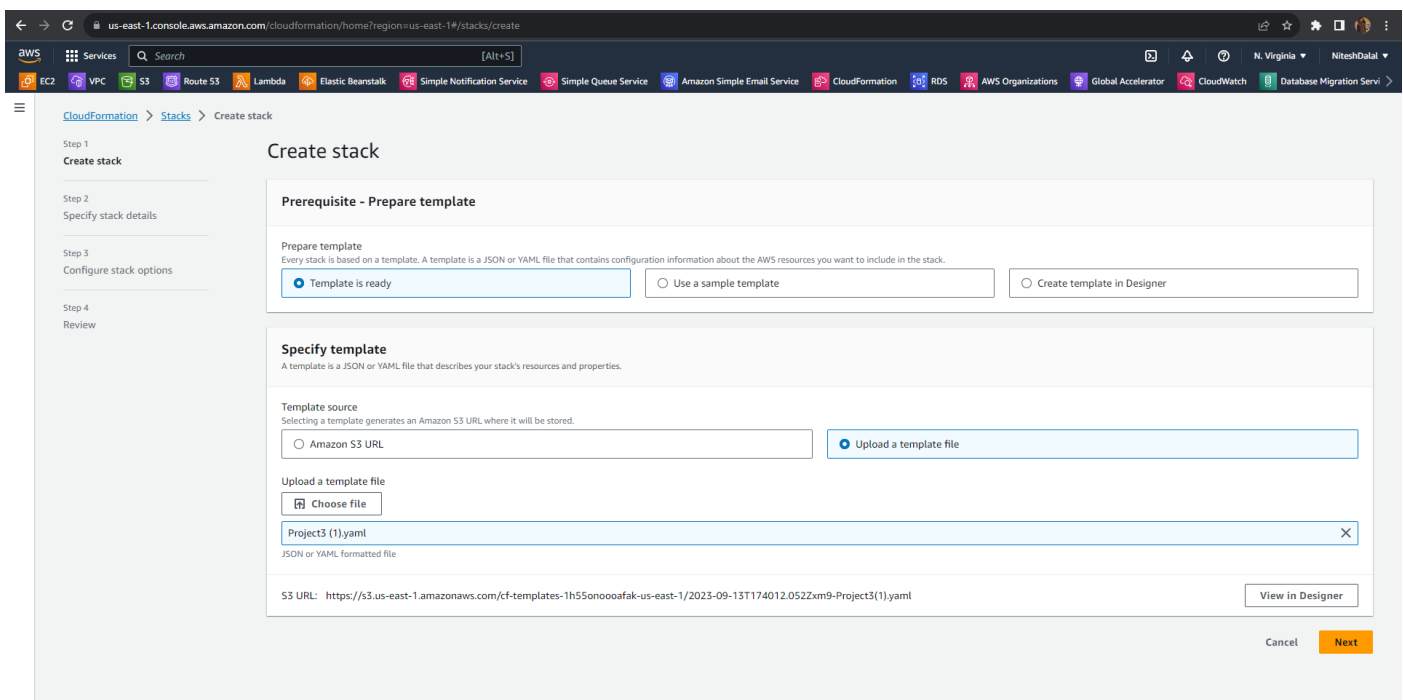
1. AWS CloudFormation to create a VPC
2. Connect VPC with AWS SNS
3. Publish message privately with SNS.

Solution:

1. Login into AWS and navigate to CloudFormation management console.



2. Click create stack and upload the template you have for creating the stack. Either from S3 or from your local PC.



3. I will upload the template code here in this step.

AWSTemplateFormatVersion: 2010-09-09

Description: CloudFormation Template for SNS VPC Endpoints Tutorial

Parameters:

KeyName:

Description: Name of an existing EC2 KeyPair to enable SSH access to the instance

Type: 'AWS::EC2::KeyPair::KeyName'

ConstraintDescription: must be the name of an existing EC2 KeyPair.

SSHLocation:

Description: The IP address range that can be used to SSH to the EC2 instance

Type: String

MinLength: '9'

MaxLength: '18'

Default: 0.0.0.0/0

AllowedPattern: '(\d{1,3})\.\d{1,3})\.\d{1,3})\.\d{1,3})/(\d{1,2})'

ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.

Mappings:

RegionMap:

us-east-1:

AMI: ami-0c293f3f676ec4f90

us-east-2:

AMI: ami-08b6f2a5c291246a0

us-west-1:

AMI: ami-051317f1184dd6e92

us-west-2:

AMI: ami-0b9f27b05e1de14e9

Resources:

VPC:

Type: 'AWS::EC2::VPC'

Properties:

CidrBlock: 10.0.0.0/16

EnableDnsSupport: 'true'

EnableDnsHostnames: 'true'

Tags:

- Key: Name

Value: VPCE-Tutorial-VPC

Subnet:

Type: 'AWS::EC2::Subnet'

Properties:

VpcId: !Ref VPC

CidrBlock: 10.0.1.0/24

Tags:

- Key: Name

Value: VPCE-Tutorial-Subnet

InternetGateway:

Type: 'AWS::EC2::InternetGateway'

Properties:

Tags:

- Key: Name

Value: VPCE-Tutorial-InternetGateway

VPCGatewayAttachment:

Type: 'AWS::EC2::VPCGatewayAttachment'

Properties:

VpcId: !Ref VPC

InternetGatewayId: !Ref InternetGateway

RouteTable:

Type: 'AWS::EC2::RouteTable'

Properties:

VpcId: !Ref VPC

Tags:

- Key: Name

Value: VPCE-Tutorial-RouteTable

SubnetRouteTableAssociation:

Type: 'AWS::EC2::SubnetRouteTableAssociation'

Properties:

RouteTableId: !Ref RouteTable

SubnetId: !Ref Subnet

InternetGatewayRoute:

Type: 'AWS::EC2::Route'

Properties:

RouteTableId: !Ref RouteTable

GatewayId: !Ref InternetGateway

DestinationCidrBlock: 0.0.0.0/0

SecurityGroup:

Type: 'AWS::EC2::SecurityGroup'

Properties:

GroupName: Tutorial Security Group

GroupDescription: Security group for SNS VPC endpoing tutorial

VpcId: !Ref VPC

SecurityGroupIngress:

- IpProtocol: '-1'

CidrIp: 10.0.0.0/16

- IpProtocol: tcp

FromPort: '22'

ToPort: '22'

CidrIp: !Ref SSHLocation

SecurityGroupEgress:

- IpProtocol: '-1'

CidrIp: 10.0.0.0/16

Tags:

- Key: Name

Value: VPCE-Tutorial-SecurityGroup

EC2Instance:

Type: 'AWS::EC2::Instance'

Properties:

KeyName: !Ref KeyName

InstanceType: t2.micro

ImageId: !FindInMap

- RegionMap

- !Ref 'AWS::Region'

- AMI

NetworkInterfaces:

- AssociatePublicIpAddress: 'true'

DeviceIndex: '0'

GroupSet:

- !Ref SecurityGroup

SubnetId: !Ref Subnet

IamInstanceProfile: !Ref EC2InstanceProfile

Tags:

- Key: Name

Value: VPCE-Tutorial-EC2Instance

EC2InstanceProfile:

Type: 'AWS::IAM::InstanceProfile'

Properties:

Roles:

- !Ref EC2InstanceRole

InstanceProfileName: myEC2InstanceProfile-Sudip

EC2InstanceRole:

Type: 'AWS::IAM::Role'

Properties:

RoleName: VPCE-Tutorial-EC2InstanceRole

AssumeRolePolicyDocument:

Version: 2012-10-17

Statement:

- Effect: Allow

Principal:

Service: ec2.amazonaws.com

Action: 'sts:AssumeRole'

ManagedPolicyArns:

- 'arn:aws:iam::aws:policy/AmazonSNSFullAccess'

LambdaExecutionRole:

Type: 'AWS::IAM::Role'

Properties:

AssumeRolePolicyDocument:

Version: 2012-10-17

Statement:

- Effect: Allow

Principal:

Service: lambda.amazonaws.com

Action: 'sts:AssumeRole'

ManagedPolicyArns:

- 'arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole'

LambdaFunction1:

Type: 'AWS::Lambda::Function'

Properties:

Code:

ZipFile: |

```
from __future__ import print_function
```

```
print('Loading function')
```

```
def lambda_handler(event, context):
```

```
    message = event['Records'][0]['Sns']['Message']
```

```
    print("From SNS: " + message)
```

```
    return message
```

Description: SNS VPC endpoint tutorial lambda function 1

FunctionName: VPCE-Tutorial-Lambda-1

Handler: index.lambda_handler

Role: !GetAtt

- LambdaExecutionRole

- Arn

Runtime: python3.9

Timeout: '3'

LambdaPermission1:

Type: 'AWS::Lambda::Permission'

Properties:

Action: 'lambda:InvokeFunction'

FunctionName: !Ref LambdaFunction1

Principal: sns.amazonaws.com

SourceArn: !Ref SNSTopic

LambdaLogGroup1:

Type: 'AWS::Logs::LogGroup'

Properties:

LogGroupName: !Sub "/aws/lambda/\${LambdaFunction1}"

RetentionInDays: '7'

LambdaFunction2:

Type: 'AWS::Lambda::Function'

Properties:

Code:

ZipFile: |

```
from __future__ import print_function
```

```
print('Loading function')
```

```
def lambda_handler(event, context):
```

```
    message = event['Records'][0]['Sns']['Message']
```

```
    print("From SNS: " + message)
```

```
    return message
```

Description: SNS VPC endpoint tutorial lambda function 2

FunctionName: VPCE-Tutorial-Lambda-2

Handler: index.lambda_handler

Role: !GetAtt

- LambdaExecutionRole

- Arn

Runtime: python3.9

Timeout: '3'

LambdaPermission2:

Type: 'AWS::Lambda::Permission'

Properties:

Action: 'lambda:InvokeFunction'

FunctionName: !Ref LambdaFunction2

Principal: sns.amazonaws.com

SourceArn: !Ref SNSTopic

LambdaLogGroup2:

Type: 'AWS::Logs::LogGroup'

Properties:

LogGroupName: !Sub "/aws/lambda/\${LambdaFunction2}"

RetentionInDays: '7'

SNSTopic:

Type: 'AWS::SNS::Topic'

Properties:

DisplayName: VPCE-Tutorial-Topic

TopicName: VPCE-Tutorial-Topic

Subscription:

- Endpoint: !GetAtt
- LambdaFunction1
- Arn

Protocol: lambda

- Endpoint: !GetAtt
- LambdaFunction2
- Arn

Protocol: lambda

4. Now you need to pass the parameters value that is required. *Note a key is required to be present in your system to make SSH connection to your EC2*. I have chosen one that I already have.

The screenshot shows the AWS CloudFormation console in the 'us-east-1' region. The 'Create stack' wizard is at Step 2, 'Specify stack details'. The 'Stack name' is 'Project3'. Under 'Parameters', 'KeyName' is 'N_Virgina_6june' and 'SSHLocation' is '0.0.0.0/0'. The 'Next' button is highlighted in orange.

5. Rest all things are optional, just review & acknowledge it at last step. And click submit.

Notification options

SNS topic ARN

No notification options
There are no notification options defined

Stack creation options

Timeout
-

Termination protection
Deactivated

Quick-create link

Capabilities

The following resource(s) require capabilities: [AWS::IAM::Role]
This template contains Identity and Access Management (IAM) resources. Check that you want to create each of these resources and that they have the minimum required permissions. In addition, they have custom names. Check that the custom names are unique within your AWS account. [Learn more](#)

☒ I acknowledge that AWS CloudFormation might create IAM resources with custom names.

Create change set

Cancel Previous Submit

6. It will take up to 5-6 minutes. But once completed it will look like this.

us-east-1.console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks/events?stackId=arn%3Aaws%3Acloudformation%3Aus-east-1%3A165381191696%3Astack%2FProject3%2F4c5d8ea0-525d-11ee-8fa3-0aeb07baa7e7&filteringText=&f...

CloudFormation > Stacks > Project3

Stacks (1)

Filter status: Active View nested

Stacks

Project3
2023-09-13 23:15:16 UTC+0530
CREATE_COMPLETE

Project3

Delete Update Stack actions Create stack

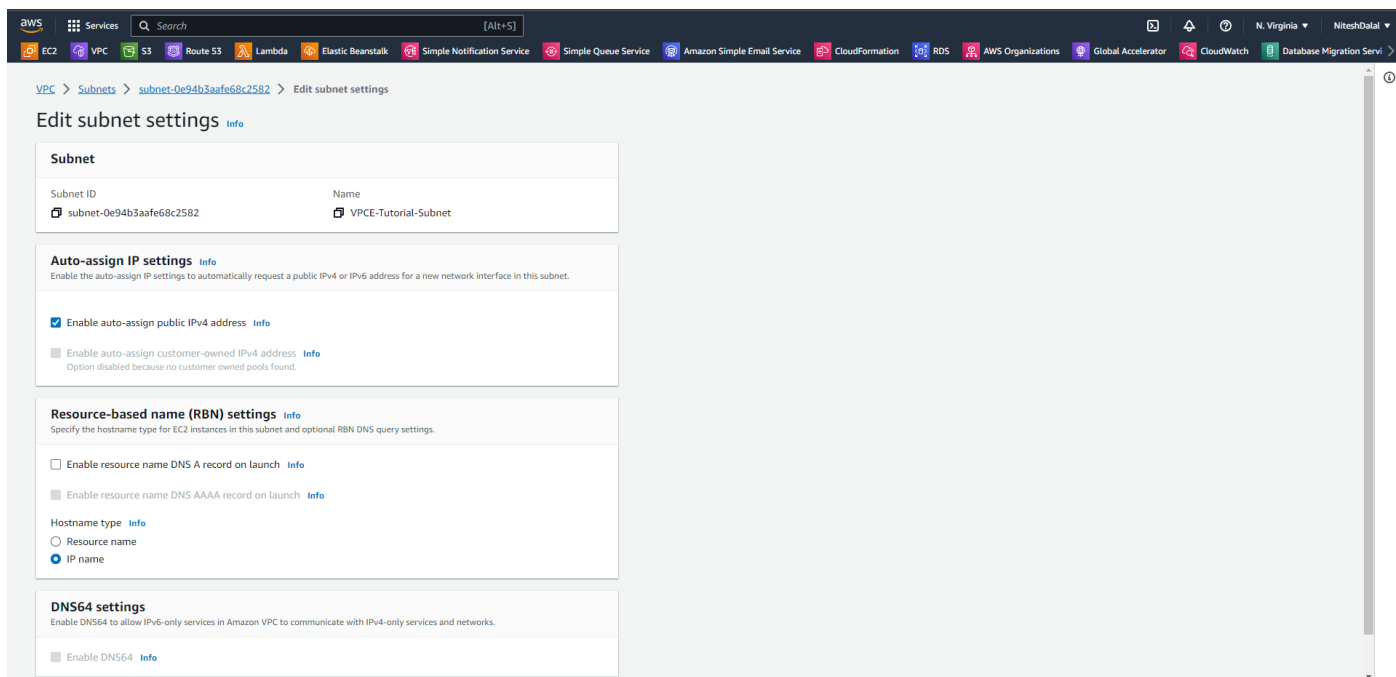
Stack info Events Resources Outputs Parameters Template Change sets

Events (59)

Search events

Timestamp	Logical ID	Status	Status reason
2023-09-13 23:18:23 UTC+0530	Project3	CREATE_COMPLETE	-
2023-09-13 23:18:22 UTC+0530	EC2Instance	CREATE_COMPLETE	-
2023-09-13 23:17:50 UTC+0530	EC2Instance	CREATE_IN_PROGRESS	Resource creation Initiated
2023-09-13 23:17:49 UTC+0530	EC2Instance	CREATE_IN_PROGRESS	-
2023-09-13 23:17:48 UTC+0530	EC2InstanceProfile	CREATE_COMPLETE	-
2023-09-13 23:16:12 UTC+0530	SubnetRouteTableAssociation	CREATE_COMPLETE	-
2023-09-13 23:15:56 UTC+0530	LambdaPermission2	CREATE_COMPLETE	-
2023-09-13 23:15:55 UTC+0530	LambdaPermission1	CREATE_COMPLETE	-
2023-09-13 23:15:55 UTC+0530	LambdaPermission2	CREATE_IN_PROGRESS	Resource creation Initiated
2023-09-13 23:15:55 UTC+0530	LambdaPermission1	CREATE_IN_PROGRESS	Resource creation Initiated
2023-09-13 23:15:54 UTC+0530	LambdaPermission2	CREATE_IN_PROGRESS	-
2023-09-13 23:15:54 UTC+0530	LambdaPermission1	CREATE_IN_PROGRESS	-
2023-09-13 23:15:54 UTC+0530	SNSTopic	CREATE_COMPLETE	-

7. Now first thing to do is to Go to your VPC that is created by this cloud formation template and edit the subnet settings. Your subnet of VPC should allow auto assign public IPV4 address. Else you will fail to connect internet and public message to SNS.



8. Now connect your EC2 instance through SSH or on web browser.

```

ec2-user@ip-10-0-1-189:~$ ssh -i "M_Virgina_6June.pem" ec2-user@ec2-3-235-181-81.compute-1.amazonaws.com
Microsoft Windows [Version 10.0.22621.2215]
(c) Microsoft Corporation. All rights reserved.

C:\Users\WITZ>cd downloads
C:\Users\WITZ\Downloads>ssh -i "M_Virgina_6June.pem" ec2-user@ec2-3-235-181-81.compute-1.amazonaws.com
The authenticity of host 'ec2-3-235-181-81.compute-1.amazonaws.com (3.235.181.81)' can't be established.
ED25519 key fingerprint is SHA256:g+x362ZL3uLZJv4kH2U94ZmyFlu2kago6vLezFYFU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-235-181-81.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

 _ _ | _ _ | _ _ |
 _ _ | _ _ | _ _ | Amazon Linux 2 AMI
 _ _ | _ _ | _ _ |

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-10-0-1-189 ~]$

```

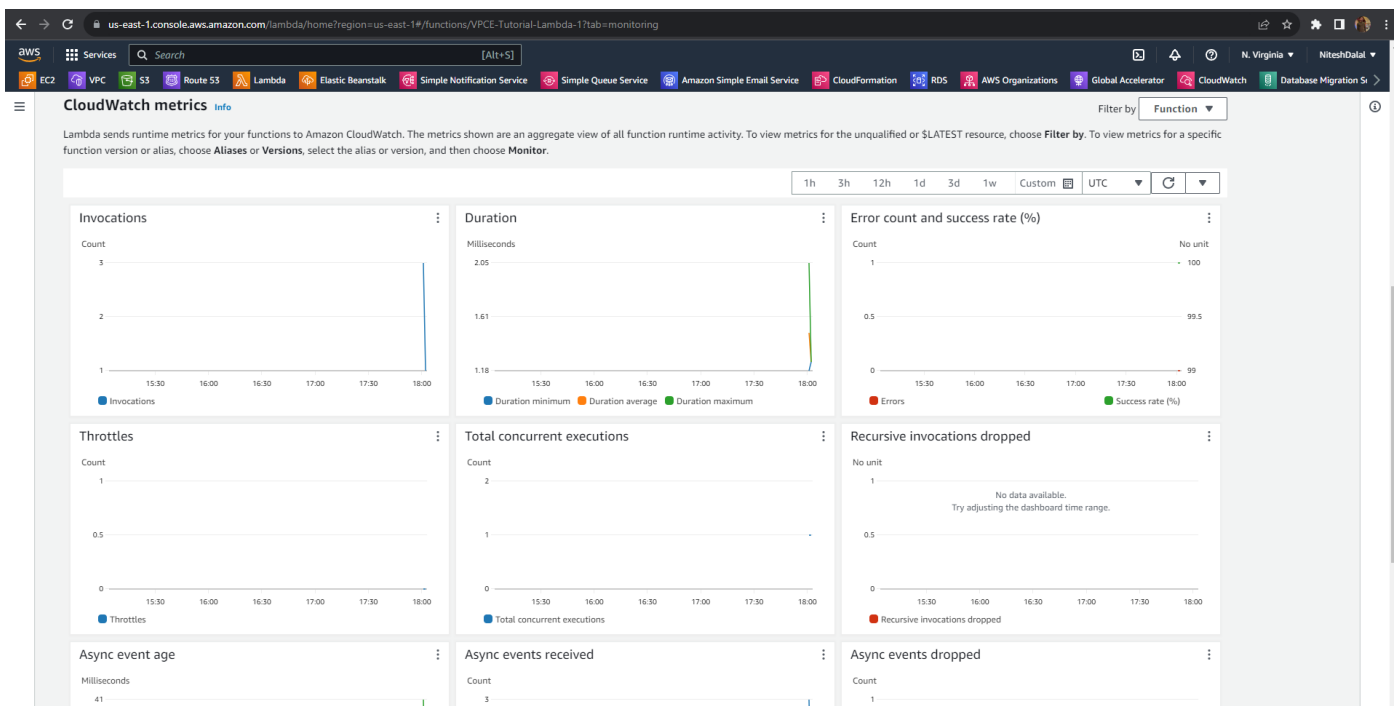
9. Try hitting the command to publish message i.e.
aws sns publish --region us-east-1 --topic-arn arn:aws:sns:us-east-1:165381191696:VPCE-Tutorial-Topic --message "Hello"

```

ec2-user@ip-10-0-1-189:~$ aws sns publish --region us-east-1 --topic-arn arn:aws:sns:us-east-1:165381191696:VPCE-Tutorial-Topic --message "Hello"
{
  "MessageId": "c66887f3-58d3-5243-8e8f-da88a39768c6"
}
[ec2-user@ip-10-0-1-189 ~]$ aws sns publish --region us-east-1 --topic-arn arn:aws:sns:us-east-1:165381191696:VPCE-Tutorial-Topic --message "My name is Nitesh"
{
  "MessageId": "759cc516-1fb1-5cd0-8f44-986de4e14cd6"
}
[ec2-user@ip-10-0-1-189 ~]$ aws sns publish --region us-east-1 --topic-arn arn:aws:sns:us-east-1:165381191696:VPCE-Tutorial-Topic --message "I am learning AWS"
{
  "MessageId": "5b81ed75-3d10-56d0-ab1a-b08ec674cd54"
}
[ec2-user@ip-10-0-1-189 ~]$ A

```

10. Now to monitor you can open lambda function and see from there. Also you can open cloudwatch logs groups to see the messages invocations.



11. Also from Logs.

The screenshot shows the AWS CloudWatch Log events dashboard. The dashboard is titled "Log events" and includes a filter by "Function". The log events are displayed as a table with columns for "Timestamp" and "Message". The events are filtered by the function name "VPCE-Tutorial-Lambda-1". The log events show the function's execution, including initialization, loading, and processing of requests.

Timestamp	Message
2023-09-13T23:31:18.382+05:30	INIT_START Runtime Version: python:3.9.v30 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:86d4ce888432216337acec891c716c38002d8ed911f5e9574e36852e7527d6ab
2023-09-13T23:31:18.530+05:30	Loading function
2023-09-13T23:31:18.531+05:30	START RequestId: f2030a81-b129-4bb2-96da-8d7684acea9b Version: SLATEST
2023-09-13T23:31:18.532+05:30	From SNS: Hello
2023-09-13T23:31:18.532+05:30	END RequestId: f2030a81-b129-4bb2-96da-8d7684acea9b
2023-09-13T23:31:18.532+05:30	REPORT RequestId: f2030a81-b129-4bb2-96da-8d7684acea9b Duration: 1.23 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 36 MB Init Duration: 148.45 ms
2023-09-13T23:31:40.806+05:30	START RequestId: 32381b0c-4797-4156-9012-e392eb40fa90 Version: SLATEST
2023-09-13T23:31:40.806+05:30	From SNS: Hello
2023-09-13T23:31:40.807+05:30	END RequestId: 32381b0c-4797-4156-9012-e392eb40fa90
2023-09-13T23:31:40.807+05:30	REPORT RequestId: 32381b0c-4797-4156-9012-e392eb40fa90 Duration: 2.05 ms Billed Duration: 3 ms Memory Size: 128 MB Max Memory Used: 36 MB
2023-09-13T23:31:52.545+05:30	START RequestId: f9ed4843-6ab6-43aa-bee6-c43098d7c93e Version: SLATEST
2023-09-13T23:31:52.545+05:30	From SNS: My name is Nitesh
2023-09-13T23:31:52.546+05:30	END RequestId: f9ed4843-6ab6-43aa-bee6-c43098d7c93e
2023-09-13T23:31:52.546+05:30	REPORT RequestId: f9ed4843-6ab6-43aa-bee6-c43098d7c93e Duration: 1.18 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 37 MB
2023-09-13T23:32:13.030+05:30	START RequestId: a241cd5e-3957-4588-aa49-e30210aab2ac Version: SLATEST
2023-09-13T23:32:13.030+05:30	From SNS: I am learning AWS
2023-09-13T23:32:13.031+05:30	END RequestId: a241cd5e-3957-4588-aa49-e30210aab2ac
2023-09-13T23:32:13.031+05:30	REPORT RequestId: a241cd5e-3957-4588-aa49-e30210aab2ac Duration: 1.25 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 37 MB