17/04/2023

Twilio and AWS Lambda Integration

This document provides a step-by-step guide on how to use Twilio and AWS Lambda integration to notify team members of system issues or errors via phone calls. By following the procedures outlined in this guide, you can set up an EventBridge rule that triggers a Lambda function, which in turn integrates with Twilio to make phone calls under specific conditions.

Alev Ayaz
GANTEK TECHNOLOGY

Twilio and AWS Lambda Integration

Prerequisites

- AWS Free Tier account
- Twilio trial account
- Twilio Phone Number
- Node.js installed on your computer.

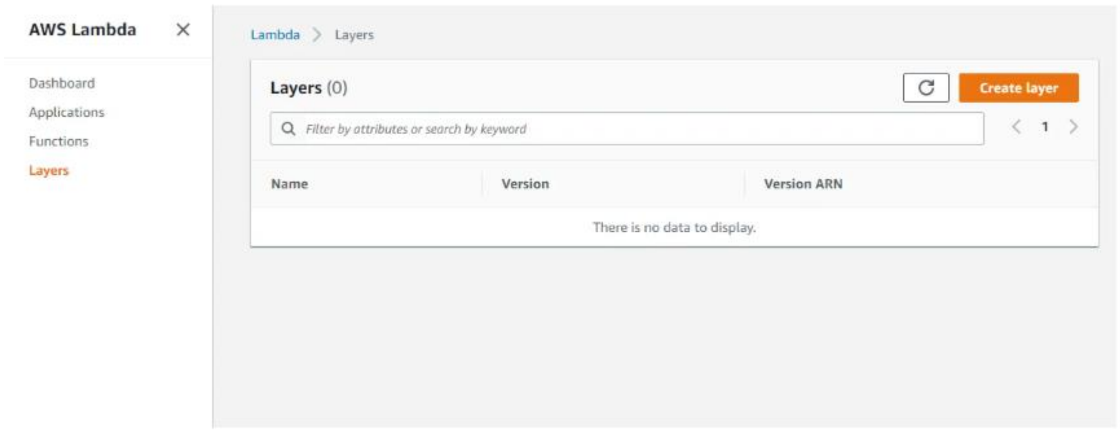
A. Prepare the ZIP archive containing Twilio's library and Create and AWS Lambda Layer.

1. On your Desktop create a new folder: nodejs. This folder will contain all the dependencies required for your code to run.
2. In your terminal, navigate to your project directory and run the following commands to initialize the project and install the required Twilio dependency:

```
npm init -y
```

```
npm install twilio --save
```

3. If your operating system is Windows, find the nodejs folder you just created right-click on it, then click Send to > Compressed (zipped) folder.
4. Open the AWS Lambda console at: https://console.aws.amazon.com/lambda/
5. On the left navigation panel, click Layers to open the Layers page.



6. Click the **Create layer** button. You should see the Create layer page.

7. In the Layer configuration section, enter “**twilioNodeLibrary**” for the Name.

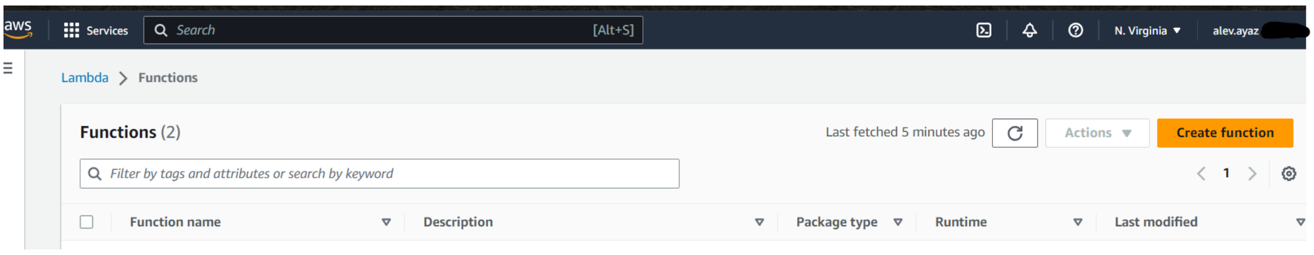
8. Choose **Upload** a .zip file.

9. Click Upload and, when prompted, add the nodejs.zip archive from your Desktop.

10. From the Compatible runtimes drop-down, select all three Node.js versions (Node.js 10.x, Node.js 12.x, Node.js 8.10), since Twilio’s Node.js library is compatible with all of them. Click the **Create** button.

B. Create an AWS Lambda Function

1. Open the AWS Lambda console https://console.aws.amazon.com/lambda/
2. Click **Create function**.



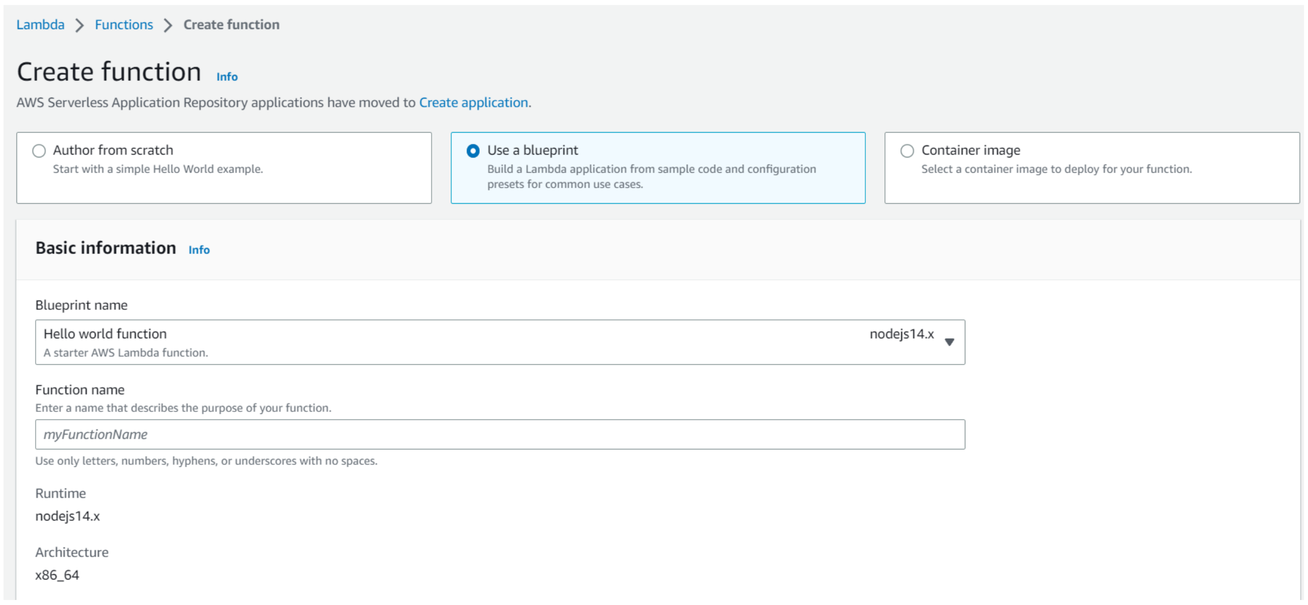
Functions (2)

Last fetched 5 minutes ago

Filter by tags and attributes or search by keyword

Function name	Description	Package type	Runtime	Last modified

3. On the Create function page, select Use a **blueprint**.
4. In the Blueprints panel, type "**hello world function**" for the filter, press Enter, and choose the hello-world (nodejs) blueprint.
5. For the execution role chose a "**create a new role with basic lambda permissions**"



Create function

AWS Serverless Application Repository applications have moved to Create application.

Author from scratch
Start with a simple Hello World example.

[✓ Use a blueprint]
Build a Lambda application from sample code and configuration presets for common use cases.

Container image
Select a container image to deploy for your function.

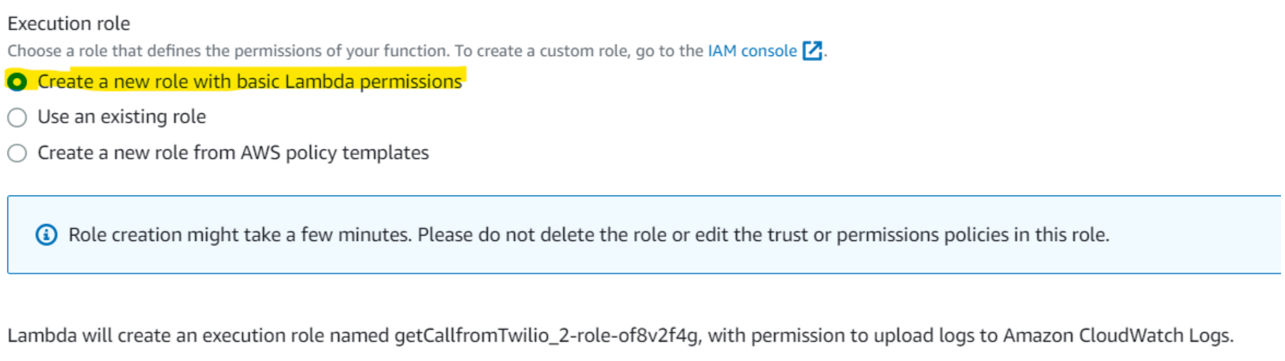
Basic information

Blueprint name: Hello world function — A starter AWS Lambda function. — nodejs14.x

Function name
Enter a name that describes the purpose of your function.
myFunctionName
Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime: nodejs14.x

Architecture: x86_64



Execution role

Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.

[✓ Create a new role with basic Lambda permissions]

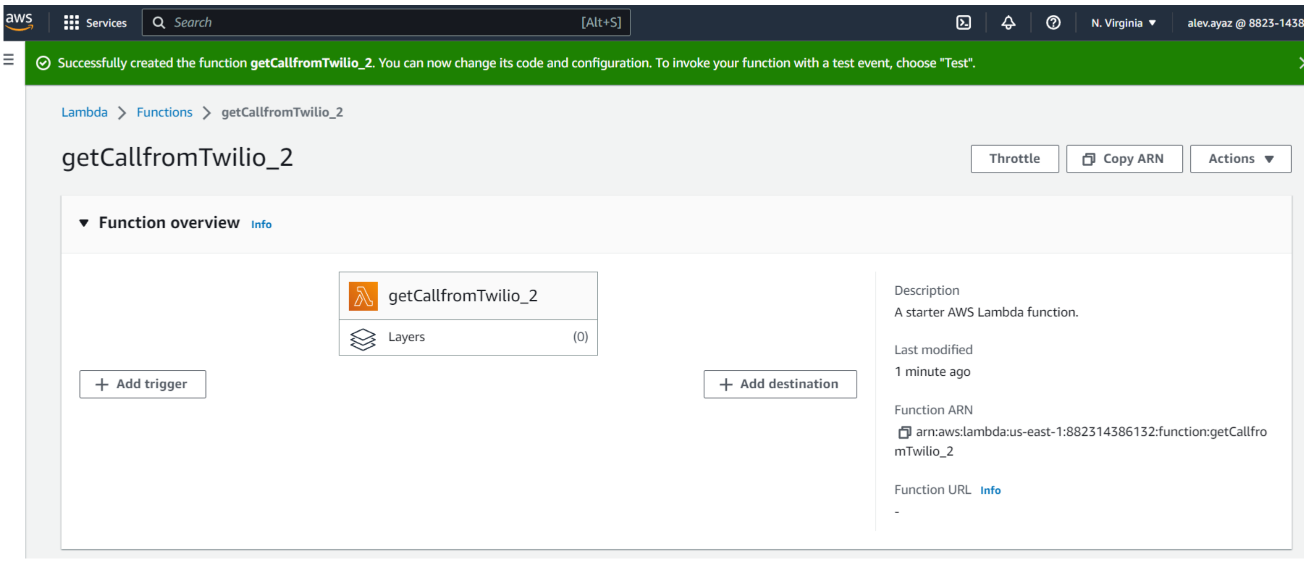
Use an existing role

Create a new role from AWS policy templates

Role creation might take a few minutes. Please do not delete the role or edit the trust or permissions policies in this role.

Lambda will create an execution role named getCallfromTwilio_2-role-of8v2f4g, with permission to upload logs to Amazon CloudWatch Logs.

6. Leave the function code as it is and click **create function**. Then you will the screen below.



Successfully created the function **getCallfromTwilio_2**. You can now change its code and configuration. To invoke your function with a test event, choose "Test".

Lambda > Functions > getCallfromTwilio_2

getCallfromTwilio_2

Function overview

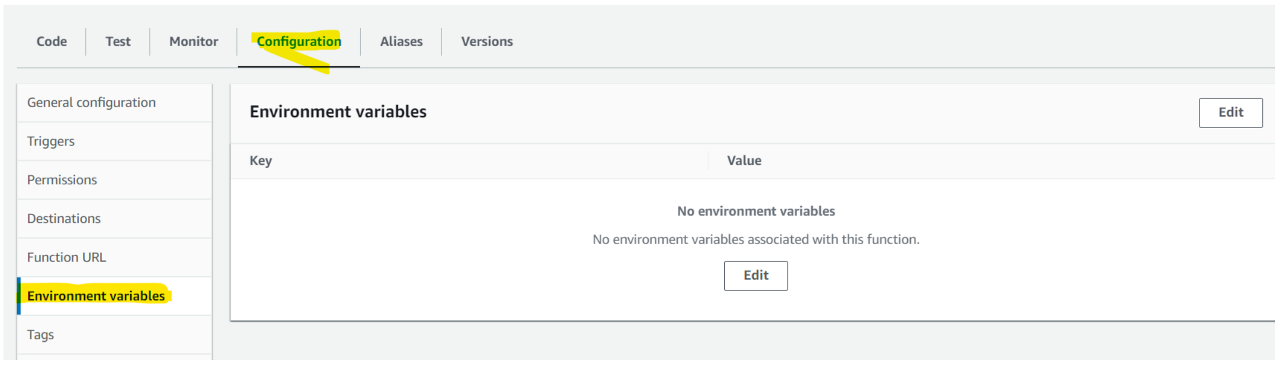
Description
A starter AWS Lambda function.

Last modified
1 minute ago

Function ARN
arn:aws:lambda:us-east-1:882314386132:function:getCallfromTwilio_2

Function URL
-

7. Go to the Configuration tab and move to Environmental Variables then press **edit** and configure as below.
8. You should add *TWILO_ACCOUNT_SID* and *TWILIO_ACCOUNT_TOKEN* which you can find on Twilio Console.

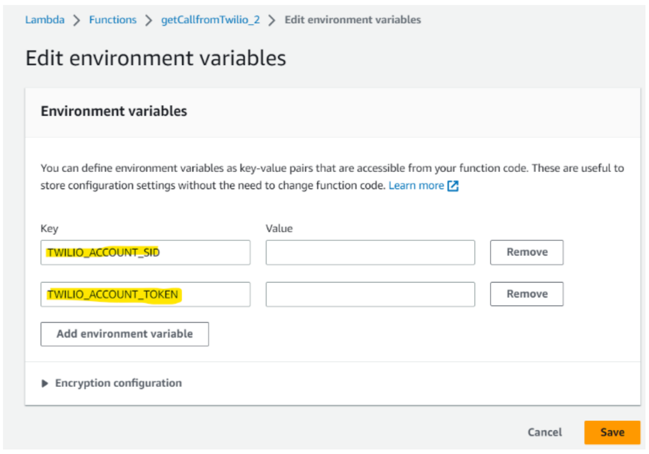


Environment variables

Key	Value

No environment variables

No environment variables associated with this function.



Lambda > Functions > getCallfromTwilio_2 > Edit environment variables

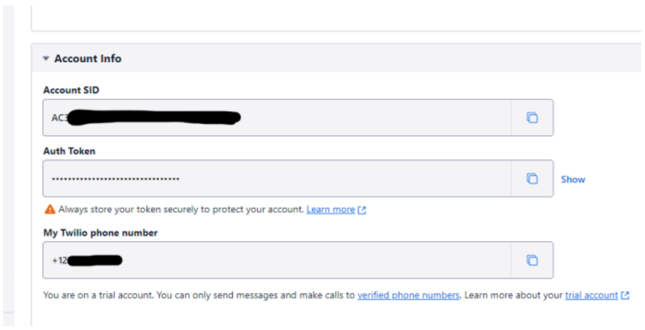
Edit environment variables

Environment variables

You can define environment variables as key-value pairs that are accessible from your function code. These are useful to store configuration settings without the need to change function code. Learn more

Key	Value
TWILIO_ACCOUNT_SID	
TWILIO_ACCOUNT_TOKEN	

* From Twilio Console:



Account Info

Account SID
AC[redacted]

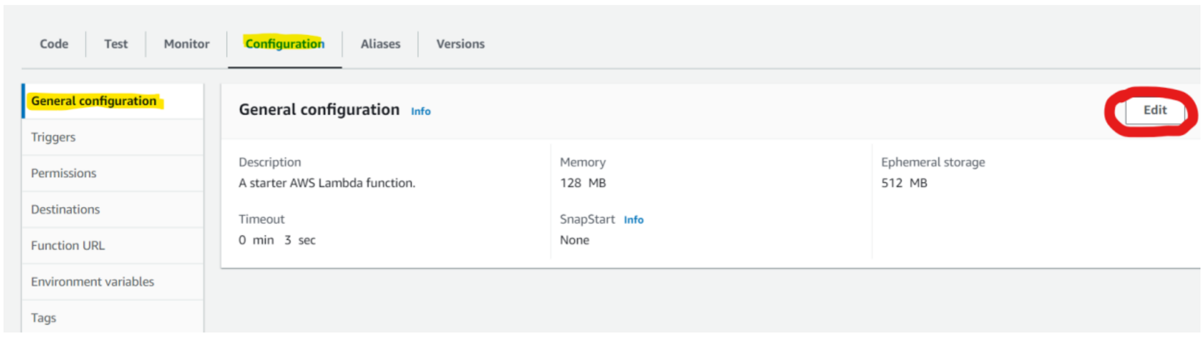
Auth Token
....

Always store your token securely to protect your account. Learn more

My Twilio phone number
+12[redacted]

You are on a trial account. You can only send messages and make calls to verified phone numbers. Learn more about your trial account

9. Go to the Configuration tab again and chose General Configuration

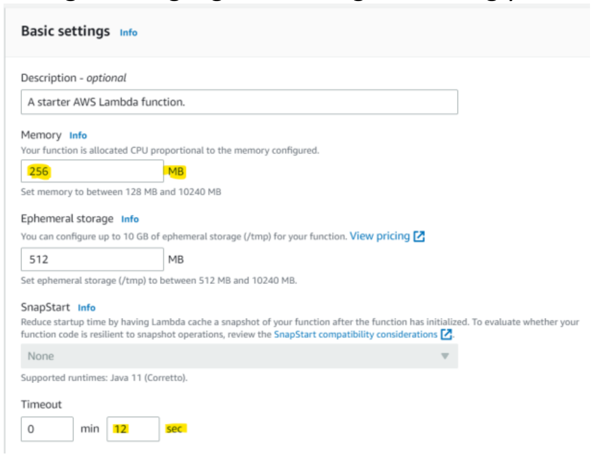

Code | Test | Monitor | Configuration | Aliases | Versions

General configuration
Triggers
Permissions
Destinations
Function URL
Environment variables
Tags

General configuration Info — Edit

Description: A starter AWS Lambda function.
Memory: 128 MB
Ephemeral storage: 512 MB
Timeout: 0 min 3 sec
SnapStart Info: None

Change the highlighted settings accordingly. Then save it.


Basic settings Info

Description - *optional*: A starter AWS Lambda function.

Memory Info
Your function is allocated CPU proportional to the memory configured.
256 MB
Set memory to between 128 MB and 10240 MB

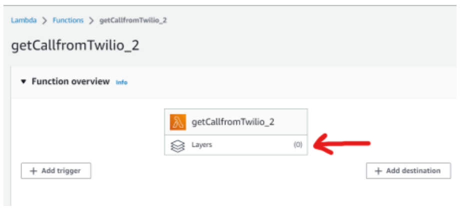
Ephemeral storage Info
You can configure up to 10 GB of ephemeral storage (/tmp) for your function. View pricing
512 MB
Set ephemeral storage (/tmp) to between 512 MB and 10240 MB.

SnapStart Info
Reduce startup time by having Lambda cache a snapshot of your function after the function has initialized. To evaluate whether your function code is resilient to snapshot operations, review the SnapStart compatibility considerations.
None
Supported runtimes: Java 11 (Corretto).

Timeout
0 min 12 sec

C. Add a layer to your Lambda function.

1. Click Layers and Add a Layer.

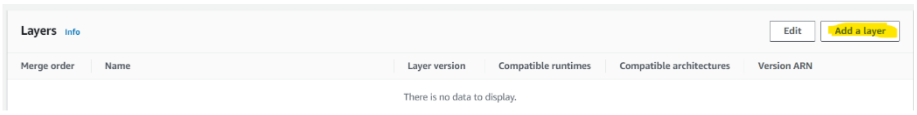

Lambda > Functions > getCallfromTwilio_2

getCallfromTwilio_2

Function overview Info

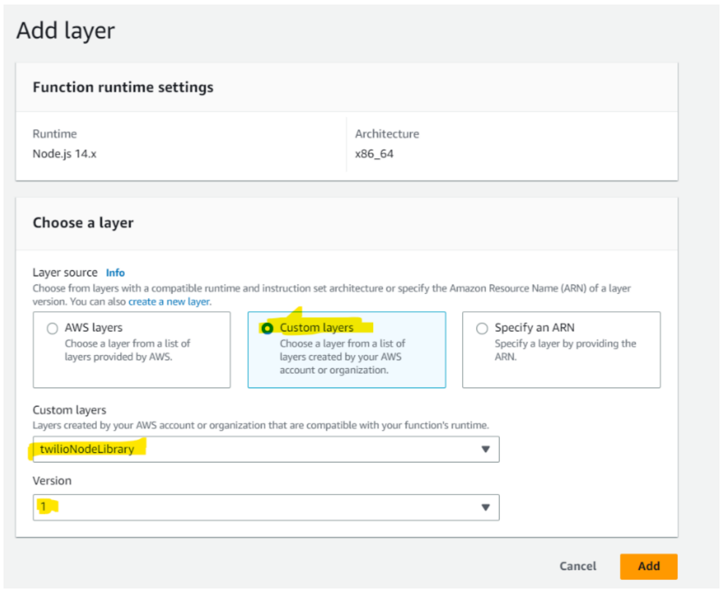
getCallfromTwilio_2
Layers (0)

+ Add trigger
+ Add destination


Layers Info — Edit — Add a layer

Merge order	Name	Layer version	Compatible runtimes	Compatible architectures	Version ARN

There is no data to display.


Add layer

Function runtime settings

Runtime: Node.js 14.x
Architecture: x86_64

Choose a layer

Layer source Info
Choose from layers with a compatible runtime and instruction set architecture or specify the Amazon Resource Name (ARN) of a layer version. You can also create a new layer.

AWS layers — Choose a layer from a list of layers provided by AWS.
[✓ Custom layers — Choose a layer from a list of layers created by your AWS account or organization.]
Specify an ARN — Specify a layer by providing the ARN.

Custom layers
Layers created by your AWS account or organization that are compatible with your function's runtime.
twilioNodeLibrary

Version
1

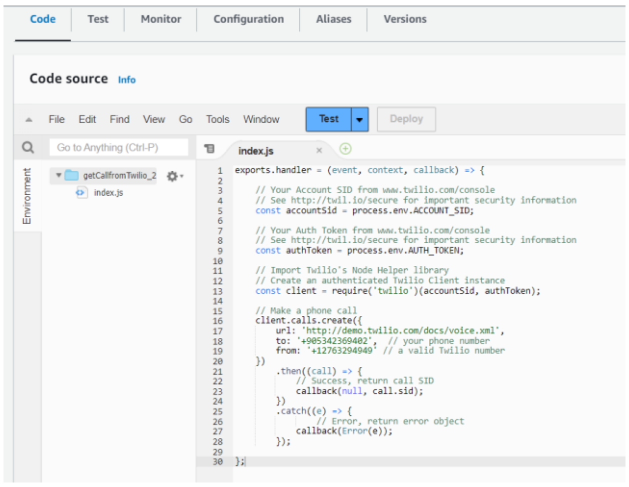
Cancel — Add

2. Select the "Custom Layers" option and choose the "twilioNodeLibrary" layer with Version 1, which we added in Step A.
3. Click **Add**.

4. Add the Lambda Code

Delete the existing code and copy paste the code below. Then press **Deploy**.

```
exports.handler = (event, context, callback) => {


   // Your Account SID from www.twilio.com/console


   const accountSid = process.env.ACCOUNT_SID;


   // Your Auth Token from www.twilio.com/console


   const authToken = process.env.AUTH_TOKEN;


   // Import Twilio's Node Helper library

   // Create an authenticated Twilio Client instance

   const client = require('twilio')(accountSid, authToken);


   // Make a phone call

   client.calls.create({

      url: 'http://demo.twilio.com/docs/voice.xml',

      to: '+111111111111',  // your phone number

      from: '+2222222222' // a valid Twilio number

   })

      .then((call) => {

         // Success, return call SID

         callback(null, call.sid);

      })

      .catch((e) => {

            // Error, return error object

         callback(Error(e));

      });


};
```

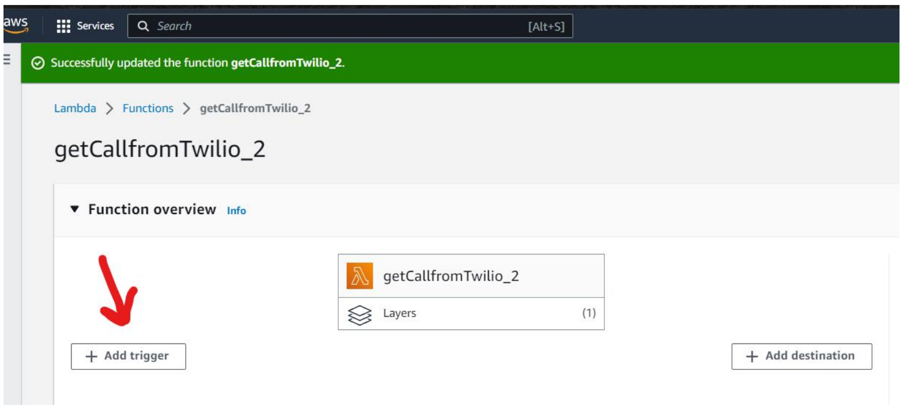


```
exports.handler = (event, context, callback) => {

    // Your Account SID from www.twilio.com/console
    // See http://twil.io/secure for important security information
    const accountSid = process.env.ACCOUNT_SID;

    // Your Auth Token from www.twilio.com/console
    // See http://twil.io/secure for important security information
    const authToken = process.env.AUTH_TOKEN;

    // Import Twilio's Node Helper library
    // Create an authenticated Twilio Client instance
    const client = require('twilio')(accountSid, authToken);

    // Make a phone call
    client.calls.create({
        url: 'http://demo.twilio.com/docs/voice.xml',
        to: '+905342369402',  // your phone number
        from: '+12763294949' // a valid Twilio number
    })
        .then((call) => {
            // Success, return call SID
            callback(null, call.sid);
        })
        .catch((e) => {
                // Error, return error object
            callback(Error(e));
        });

};
```

PS: You can **test** your code to check if it works without an error

5. Add Trigger

1. Navigate to the Function tab and click on the 'Add Trigger' button



Successfully updated the function **getCallfromTwilio_2.**

Lambda > Functions > getCallfromTwilio_2

getCallfromTwilio_2

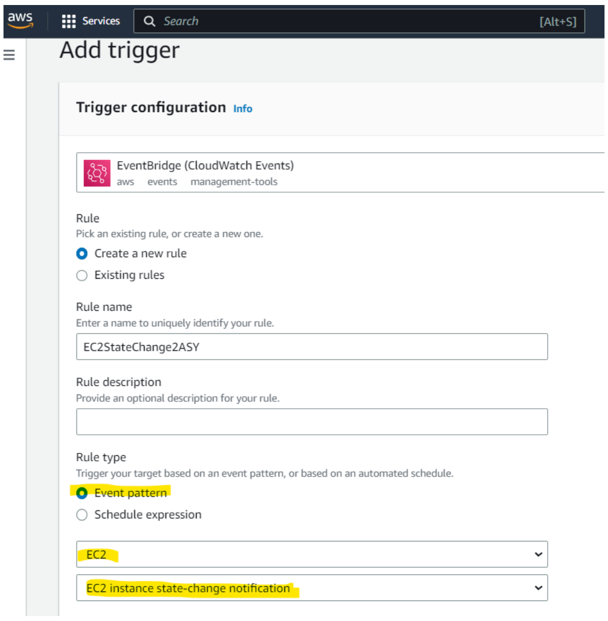
Function overview Info

getCallfromTwilio_2

Layers (1)

2. Then chose EventBridge and ‘Create a New Rule’ with ‘Event Pattern’

Based on the state that was selected, the Lambda function will be triggered whenever an EC2 instance is in the process of **stopping**, has **stopped**, or has been **terminated**.



Add trigger

Trigger configuration Info

EventBridge (CloudWatch Events)
aws events management-tools

Rule
Pick an existing rule, or create a new one.
[✓ Create a new rule]
Existing rules

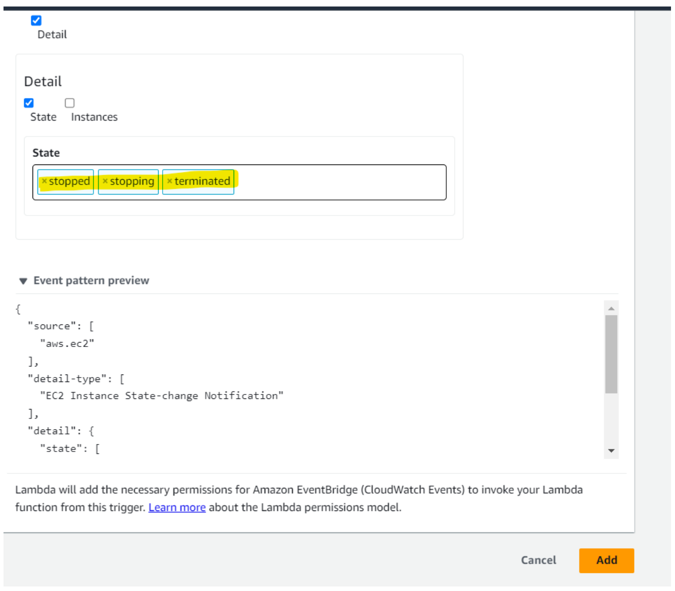
Rule name
Enter a name to uniquely identify your rule.
EC2StateChange2ASY

Rule description
Provide an optional description for your rule.

Rule type
Trigger your target based on an event pattern, or based on an automated schedule.
[✓ Event pattern]
Schedule expression

EC2

EC2 instance state-change notification



[✓ Detail]

Detail
[✓ State] Instances

State
stopped stopping terminated

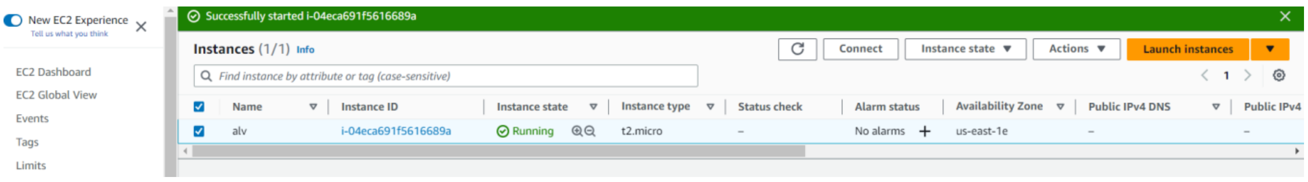
Event pattern preview

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

Lambda will add the necessary permissions for Amazon EventBridge (CloudWatch Events) to invoke your Lambda function from this trigger. Learn more about the Lambda permissions model.

6. Test The Configuration

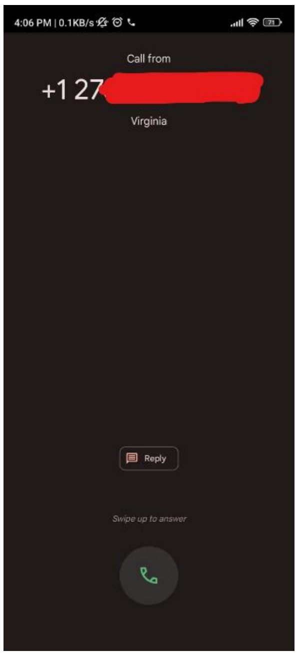
If you haven't already done so, create an EC2 instance and start it up



Successfully started i-04eca691f5616689a

Instances (1/1) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4
alv	i-04eca691f5616689a	Running	t2.micro	–	No alarms	us-east-1e	–	–



After successfully running the function, navigate to the instance state and stop the instance. As soon as the instance is in the process of stopping, a Lambda function integrated with Twilio and triggered by an EventBridge rule defined for this specific condition will be invoked, resulting in a phone call being received.

Summary

We integrated Twilio, a cloud communications platform, with Lambda to trigger a phone call whenever a certain state of the EC2 instance was detected. To do this, we used the Twilio Node.js library, which is added to the Lambda function as a custom layer.

Finally, we used Amazon EventBridge, a serverless event bus, to monitor and detect the state changes of the EC2 instance. When EventBridge detects a specific state change, it triggers the Lambda function which in turn triggers a phone call using Twilio.

So, in summary, Lambda acts as the glue between EventBridge and Twilio, allowing you to automate and streamline your communication processes based on specific events happening in your infrastructure.