

1. Validate your email address in SES.

To validate the email address, we have to create an identity with type Email address.
From Amazon SES>Configuration: Verified Identities>Create Identity

Amazon SES > Configuration: Verified identities > Create identity

Create identity

A *verified identity* is a domain, subdomain, or email address you use to send email through Amazon SES. Identity verification at the domain level extends to all email addresses under one verified domain identity.

Identity details [Info](#)

Identity type

☐ **Domain**
To verify ownership of a domain, you must have access to its DNS settings to add the necessary records.

☒ **Email address**
To verify ownership of an email address, you must have access to its inbox to open the verification email.

Email address

Email address can contain up to 320 characters, including plus signs (+), equals signs (=) and underscores (_).

Now, we can provide our email address. And when we click on the Create Identity button, we get an email with a link that activates the email subscription. After clicking the link we can get our email address verified:

Congratulations!

You have successfully verified an email address. You can now start sending email from this address.

2. Create an SNS topic.

Add subscription as Protocol Email and endpoint your Email Address
Create Event Bridge rule to monitor EC2 state change events. Set
above created SNS topic as target.

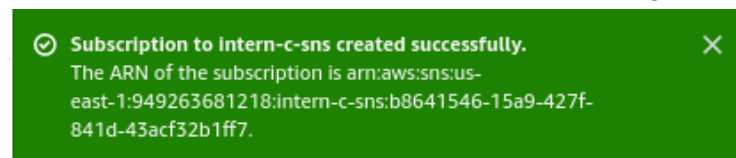
First, we can create SNS topic from Amazon SNS>Topics>Create Topic

The screenshot shows the 'Create topic' page in the Amazon SNS console. The breadcrumb trail is 'Amazon SNS > Topics > Create topic'. The page title is 'Create topic'. Under the 'Details' section, there are two tabs: 'Type' and 'Info'. Below the tabs, a note states 'Topic type cannot be modified after topic is created'. There are two radio button options for the topic type: 'FIFO (first-in, first-out)' and 'Standard'. The 'Standard' option is selected. The 'FIFO' option lists: 'Strictly-preserved message ordering', 'Exactly-once message delivery', 'High throughput, up to 300 publishes/second', and 'Subscription protocols: SQS'. The 'Standard' option lists: 'Best-effort message ordering', 'At-least once message delivery', 'Highest throughput in publishes/second', and 'Subscription protocols: SQS, Lambda, HTTP, SMS, email, mobile application endpoints'. Below the topic type selection, there is a 'Name' field with the value 'intern-c-sns'. A note below the field states: 'Maximum 256 characters. Can include alphanumeric characters, hyphens (-) and underscores (_)'.

After creating an SNS topic, and by selecting it and clicking on the Create Subscription button, we can create an subscription with our individual email as an endpoint by selecting Protocol 'Email'.

The screenshot shows the 'Create Subscription' page in the Amazon SNS console. The 'Topic ARN' field contains 'arn:aws:sns:us-east-1:949263681218:intern-c-sns'. The 'Protocol' dropdown menu is set to 'Email'. The 'Endpoint' field contains 'bijaykandel37@gmail.com'. A note below the endpoint field states: 'An email address that can receive notifications from Amazon SNS.'

After 'Create Subscription' we can see this message:



This also sends us an email to verify the subscription. After clicking on Confirm Subscription, we can get a confirmed message from SNS as:

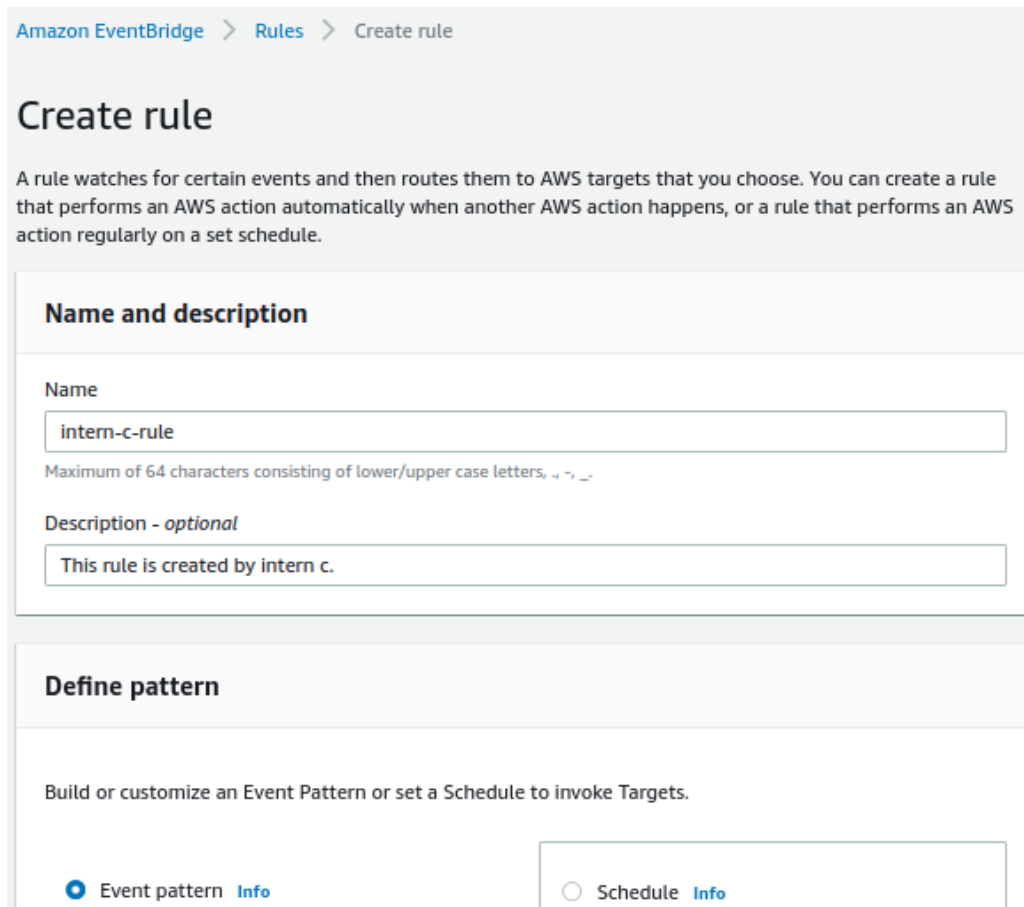


The image shows an email from Amazon Simple Notification Service (SNS). The header includes the Amazon Web Services logo and the text "Simple Notification Service". The main body of the email is a light blue box with the heading "Subscription confirmed!". Below this, it states "You have successfully subscribed." and provides the subscription ID: "arn:aws:sns:us-east-1:949263681218:intern-c-sns:b8641546-15a9-427f-841d-43acf32b1ff7". It also includes a link to unsubscribe. At the bottom, there is a table with subscription details.

<input type="radio"/>	b8641546-15a9-427f-841d-43acf32b1ff7	bijaykandel37@gmail.com	<input checked="" type="checkbox"/> Confirmed	EMAIL	intern-c-sns
-----------------------	--------------------------------------	-------------------------	---	-------	--------------

Now our email got verified and the subscription got confirmed.

Then we can create a Event Bridge rule from Amazon EventBridge>Rules>Create rule



The image shows the "Create rule" page in the Amazon EventBridge console. The breadcrumb navigation is "Amazon EventBridge > Rules > Create rule". The page title is "Create rule". Below the title is a description: "A rule watches for certain events and then routes them to AWS targets that you choose. You can create a rule that performs an AWS action automatically when another AWS action happens, or a rule that performs an AWS action regularly on a set schedule." The form is divided into two main sections: "Name and description" and "Define pattern".

Name and description

Name:
Maximum of 64 characters consisting of lower/upper case letters, -, ~, and _.

Description - optional:

Define pattern

Build or customize an Event Pattern or set a Schedule to invoke Targets.

☒ Event pattern [Info](#) ☐ Schedule [Info](#)

☒ Pre-defined pattern by service

☐ Custom pattern

Service provider

AWS services or custom/partner services

AWS

▼

Service name

The name of partner service selected as the event source

EC2

▼

Event type

The type of events as the source of the matching pattern

EC2 Instance State-change Notification

▼

☒ Any state

☐ Specific state(s)

▼

☒ Any instance

☐ Specific instance Id(s)

2"source": ["aws.ec2"],

3"detail-type": ["EC2 Instance State-ch

4 }

▶ Sample event(s)

Above created SNS topic is selected as target:

Select target(s) to invoke when an event matches your event pattern or when schedule is triggered (limit of 5 targets per rule).

Target

Select target(s) to invoke when an event matches your event pattern or when schedule is triggered (limit of 5 targets per rule).

Remove

SNS topic

Topic

intern-c-sns

3. Create standard SQS.

Add this SQS as target in above created Event Bridge rule (in addition to existing SNS)

Add lambda trigger in SQL to sendEmail lambda function.

Standard SQS is created from Amazon SQS > Queues> Create Queue

Amazon SQS > Queues > Create queue

Create queue

Details

Type
Choose the queue type for your application or cloud infrastructure.

Standard Info
At-least-once delivery, message ordering isn't preserved

- At-least once delivery
- Best-effort ordering

FIFO Info
First-in-first-out delivery, message ordering is preserved

- First-in-first-out delivery
- Exactly-once processing

Name

Intern-c-sqs

A queue name is case-sensitive and can have up to 80 characters. You can use alphanumeric characters, hyphens (-), and underscores (_).

Queue Intern-c-sqs created successfully
You can now send and receive messages.

And this SQS as target is added in above created Event Bridge rule by Selecting our rule which we created and Edit button. From there, we scroll down to the Target and Select 'Add target' and we can select SQS queue and mention our queue name as shown:

Target

Select target(s) to invoke when an event matches your event pattern or when schedule is triggered (limit of 5 targets per rule).

Remove

SQS queue

Queue*

Intern-c-sqs

Now, we can see both targets:

Target(s) (2)		View details
	Name	Type
<input type="radio"/>	intern-c-sqs	SQS queue
<input type="radio"/>	intern-c-sns	SNS topic

To add a lambda trigger in SQL to the sendEmail lambda function:

Basic information

Function name
Enter a name that describes the purpose of your function.

Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime [Info](#)
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Architecture [Info](#)
Choose the instruction set architecture you want for your function code.

☒ x86_64

☐ arm64