

# Email Masking Service on AWS

## Project Overview

This project creates an email masking service that forwards messages while protecting your real email address. It combines multiple AWS services to provide anonymous communication.

## What You'll Build

- **Email Masking System** - Create disposable email addresses that forward to your real inbox
- **Mail Storage**- Archive all received emails for future reference
- **Email Database** - Store your collection of masked email addresses
- **Serverless Architecture** - Fully automated email processing without servers

## Architecture Overview

This solution uses these AWS microservices:

- **Route 53** manages domain registration and DNS settings
- **SES (Simple Email Service)** handles email receiving and forwarding
- **Lambda Functions** process incoming emails and create new masked addresses
- **DynamoDB** stores the mapping between masked and real email addresses
- **S3** archives all received emails

## Prerequisites

### Software Requirements

- Web browser to access AWS Console
- Email account to receive forwarded messages
- Text editor for DNS configuration

### Cost

- Domain registration: ~\$3-15/year for .click domains
- SES: Free for receiving emails, \$0.10 per 1,000 emails sent
- Lambda: Free tier includes 1M free requests per month
- DynamoDB: Free tier includes 25GB storage
- S3: ~\$0.023/GB-month for storage

## Knowledge

- Basic understanding of AWS services
- Familiarity with email systems and DNS
- Basic understanding of JSON for testing

## Project Start

---

### Domain Registration

Use .click domains. Turn **off** auto-renew.

The screenshot shows the 'Pricing' section of the AWS IAM console. The breadcrumb navigation at the top reads 'IAM domains > Checkout'. The 'Domain pricing options' section contains three fields: 'Domain name' with the value 'k12stemtest.click', 'Duration (price)' with a dropdown menu showing '1 year (3.00 USD)', and 'Auto-renew' with an unchecked checkbox labeled 'Off'. Below these fields is a light blue information box with an icon and text stating: 'Auto-renew is turned off for this domain and the registration will expire after the selected registration period. For more information, see [Renewing Registration for a Domain](#).' At the bottom right of the pricing section, it shows 'Subtotal: 3.00 USD' and a note 'Applicable taxes will be calculated at checkout.' At the very bottom of the form, there are two buttons: 'Cancel' and 'Next'.

[IAM domains](#) > Checkout

**Pricing** [Info](#)

**Domain pricing options**

Domain name:

Duration (price):

Auto-renew: ☐ Off

**Auto-renew is turned off for this domain and the registration will expire after the selected registration period.**  
For more information, see [Renewing Registration for a Domain](#).

Subtotal: **3.00 USD**  
Applicable taxes will be calculated at checkout.

[Cancel](#) [Next](#)

Fill out all the information required for the next page, then click submit.

Now wait a few minutes.

Route 53

Dashboard

Hosted zones

Health checks

Profiles [New](#)

▼ IP-based routing

CIDR collections

▼ Traffic flow

Traffic policies

Policy records

▼ Domains

Registered domains

Requests

▼ Resolver

VPCs

Inbound endpoints

Outbound endpoints

Rules

Query logging

Outposts

DNS Firewall [🔗](#)

Application Recovery Controller [🔗](#)

Route 53 > Hosted zones > k12stemtest.click

Public k12stemtest.click [Info](#)

Delete zone

Test record

Configure query logging

► Hosted zone details

Edit hosted zone

Records (2)

DNSSEC signing

Hosted zone tags (0)

Records (2) [Info](#)

[Delete record](#)

[Import zone file](#)

Create record

Automatic mode is the current search behavior optimized for best filter results. [To change modes go to settings.](#)

Filter records by property or value

Type

Routing p...

Alias

< 1 >

⚙️

<input type="checkbox"/>	Record name	Type	Routin...	Differ...	Alias	Value/Route traffic to	TTL (s...
<input type="checkbox"/>	k12stemtest.click	NS	Simple	-	No	ns-879.awsdns-45.net. ns-1917.awsdns-47.co.uk. ns-339.awsdns-42.com. ns-1252.awsdns-28.org.	172800
<input type="checkbox"/>	k12stemtest.click	SOA	Simple	-	No	ns-879.awsdns-45.net. awsd...	900

# Amazon SES

Add your email address

To get started with Amazon SES you must provide an email address so that we can send you a verification link. This verification process shows us you're the owner of the email address.

Email address [Info](#)

Email address

A verification email will be sent to you at this address.

Enter an email address

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

Cancel


Next

On the next page, put your domain in the sending domain. Ignore the MAIL FROM domain

Skip this as well

## Deliverability enhancements - *optional*

### Virtual Deliverability Manager

Virtual Deliverability Manager is an Amazon SES feature that helps enhance email delivery and provides recommendations to improve delivery success and sender reputation. [Amazon SES pricing](#) 

#### Key features

- **Insights dashboard** - Provides time series views of key delivery and engagement metrics, sender identity, and configuration set or search for specific messages to see the exact response from recipients.
- **Advisor recommendations** - The Advisor feature analyzes your sending and configuration to provide recommendations to improve your email deliverability.
- **Optimized delivery** - Optimized shared delivery automatically chooses the optimal IP to use for your email, but does not apply to dedicated IP addresses.

#### Virtual Deliverability Manager


☐ Turn on Virtual Deliverability Manager

Now hit confirm and create your SES.

Now go press the confirmation email link in your personal email that you put in to sign up for the SES.

## Verify Domain with SES


Click on the get DNS Records on your SES page



## Verify sending domain

**k12stemtest.click**

Add DNS records to your DNS provider.

 Verification pending

[Get DNS Records](#)

Stay on this page and open up a notepad (IGNORE THE MAIL FROM records)

### Domain DNS records

Copy the following records and add them to your domain's DNS settings as instructed by your DNS provider.

#### DKIM records

DKIM-signed messages help receiving mail servers validate that a message was not forged or altered in transit. [Learn more](#)

Type	Name	Value
CNAME	<a href="#">zuzbdwrsnc5ilwk4js6mux7jh7x5pwp_domainkey.k12stemtest.click</a>	<a href="#">zuzbdwrsnc5ilwk4js6mux7jh7x5pwp.dkim.amazonses.com</a>
CNAME	<a href="#">ps63ngryqcom6nshinympgygum6rdd_domainkey.k12stemtest.click</a>	<a href="#">ps63ngryqcom6nshinympgygum6rdd.dkim.amazonses.com</a>
CNAME	<a href="#">u4ktfmlrvyqlyif5ylmseuobkbyc5ko3_domainkey.k12stemtest.click</a>	<a href="#">u4ktfmlrvyqlyif5ylmseuobkbyc5ko3.dkim.amazonses.com</a>

#### MAIL FROM records

When using a custom MAIL FROM domain, messages sent through Amazon SES will be marked as originating from your domain instead of a subdomain of amazon.com. [Learn more](#)

Type	Name	Value
MX	<a href="#">mail.k12stemtest.click</a>	<a href="#">10 feedback-smtp.us-east-1.amazonses.com</a>
TXT	<a href="#">mail.k12stemtest.click</a>	<a href="#">v=spf1 include:amazonses.com ~all"</a>

#### DMARC records

DMARC specifies how email servers should handle messages that fail the authentication checks. [Learn more](#)

Type	Name	Value
TXT	<a href="#">_dmarc.k12stemtest.click</a>	<a href="#">v=DMARC1; p=none;"</a>

[Download .csv record set](#)
[Close](#)

Now replace the values marked with [ ] in here with your own on a notepad. Do not include the brackets [ ] in the final result.

; Zone file for k12stemtest.click

\$TTL 3600 ; Default TTL of 1 hour

\$ORIGIN [domain name].

; DKIM records for Amazon SES

[cname name] IN CNAME [cname value].

[cname name] IN CNAME [cname value].

[cname name] IN CNAME [cname value].

; MAIL FROM domain settings

IN MX 10 inbound-smtp.us-east-1.amazonaws.com.

IN TXT "v=spf1 include:amazonses.com ~all"

; DMARC policy

\_dmarc IN TXT "v=DMARC1; p=none;"

Go to your Route 53

Now import zone file

↺

Import zone file

↻

Delete record

[change modes go to settings.](#)

Type ▼

Routing p...

▼

Alias ▼

Routin... ▼

Differ... ▼

Alias ▼

Value/Route traffic to

Simple

-

No

ns-879.awsdns-45.net.  
ns-1917.awsdns-47.co.uk.  
ns-339.awsdns-42.com.

Put in your zone file that you edited

#### Import zone file info

You can create records for a Route 53 hosted zone by importing a zone file.

##### Zone file

Paste the contents of your zone file below.

```
;; Zone records for amazonses.com
zubzdwrsc5ilwk4js6muxr7jh7x5pwp._domainkey IN CNAME zubzdwrsc5ilwk4js6muxr7jh7x5pwp.dkim.amazonses.com.
ps63ngryqcorn6nshinyngptygum6rdd._domainkey IN CNAME ps63ngryqcorn6nshinyngptygum6rdd.dkim.amazonses.com.
u4ktfmlrvyqiyf5ylmseuobkbvc5ko3._domainkey IN CNAME u4ktfmlrvyqiyf5ylmseuobkbvc5ko3.dkim.amazonses.com.

; MAIL FROM domain settings
mail IN MX 10 feedback-smtp.us-east-1.amazonses.com.
mail IN TXT "v=spf1 include:amazonses.com ~all"

; DMARC policy
_dmarc IN TXT "v=DMARC1; p=none;"
```

If the hosted zone already contains records that appear in the zone file, the import process fails, and no records are created. Enter multiple records on separate lines.

##### Record preview for k12stemtest.click (6)

Route 53 creates the following records when you choose Import zone file. If you edit the contents of the zone file above, the table reflects your changes.

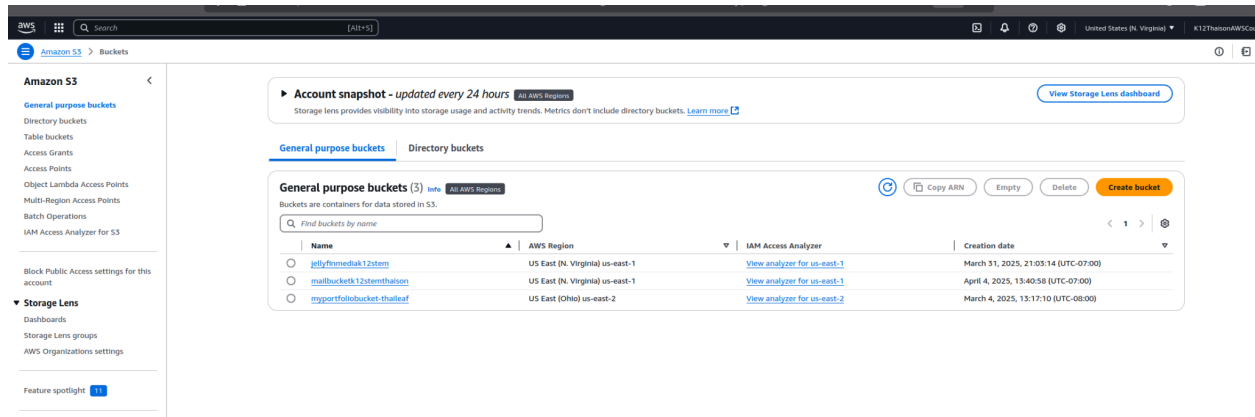
Filter records by property or value

Record name	Type	Value/Route traffic to	TTL (seconds)
_dmarc.k12stemtest.click.	TXT	"v=DMARC1; p=none;"	3600
mail.k12stemtest.click.	MX	10 feedback-smtp.us-east-1.amazonses.com.	3600
mail.k12stemtest.click.	TXT	"v=spf1 include:amazonses.com ~all"	3600
ps63ngryqcorn6nshinyngptygum6rdd._domainkey.k12stemtest.click.	CNAME	ps63ngryqcorn6nshinyngptygum6rdd.dkim.amazonses.com.	3600
u4ktfmlrvyqiyf5ylmseuobkbvc5ko3._domainkey.k12stemtest.click.	CNAME	u4ktfmlrvyqiyf5ylmseuobkbvc5ko3.dkim.amazonses.com.	3600
zubzdwrsc5ilwk4js6muxr7jh7x5pwp._domainkey.k12stemtest.click.	CNAME	zubzdwrsc5ilwk4js6muxr7jh7x5pwp.dkim.amazonses.com.	3600

Your domain is now configured to send emails to amazon SES

# Creating an S3 Bucket to store emails

Go to your main page and create an S3 bucket. Use default settings, and name it something that makes sense.



## Create bucket [Info](#)

Buckets are containers for data stored in S3.

### General configuration

#### AWS Region

US East (N. Virginia) us-east-1

#### Bucket type [Info](#)

##### ☒ General purpose

Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

##### ☐ Directory

Recommended for low-latency use cases. These buckets use only the S3 Express processing of data within a single Availability Zone.

#### Bucket name [Info](#)

sesmailbucketthaison

Bucket names must be 3 to 63 characters and unique within the global namespace. Bucket names must also begin and end with a letter or number. Valid characters are a-z, 0-9, periods (.), and hyphens (-). [Learn More](#)

#### Copy settings from existing bucket - optional

Only the bucket settings in the following configuration are copied.

[Choose bucket](#)

Format: s3://bucket/prefix

### Object Ownership [Info](#)

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

##### ☒ ACLs disabled (recommended)

All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

##### ☐ ACLs enabled

Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects is specified using ACLs.

#### Object Ownership

Bucket owner enforced

# Using DynamoDB

We need to create the table that stores the translations of masked emails to real emails.

Go to the dynamo db homepage by using the search bar.

Create a table. Name it something and keep it all the defaults. The partition key will be masked\_email, and it will be of type string.

## Create table

Table details [Info](#)

DynamoDB is a schemaless database that requires only a table name and a primary key when you create the table.

## Table name

This will be used to identify your table.

Between 3 and 255 characters, containing only letters, numbers, underscores (\_), hyphens (-), and periods (.).

## Partition key

The partition key is part of the table's primary key. It is a hash value that is used to retrieve items from your table and allocate data across hosts for scalability and availability.

String

1 to 255 characters and case sensitive.

## Sort key - optional

You can use a sort key as the second part of a table's primary key. The sort key allows you to sort or search among all items sharing the same partition key.

String

1 to 255 characters and case sensitive.

## Table settings

☒ Default settings

The fastest way to create your table. You can modify most of these settings after your table has been created. To modify these settings now, choose 'Customize settings'.

☐ Customize settings

Use these advanced features to make DynamoDB work better for your needs.

## Default table settings

These are the default settings for your new table. You can change some of these settings after creating the table.

Setting	Value	Editable after creation
Table class	DynamoDB Standard	Yes

## Set up lambda functions

### Creating the functions

#### Create The Receive email function

Create function [Info](#)

Choose one of the following options to create your function.

☒ 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

## Function name

Enter a name that describes the purpose of your function.

Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (\_).

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☐ arm64Permissions [Info](#)

By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

► Change default execution role

► Additional Configurations

Use additional configurations to set up code signing, function URL, tags, and Amazon VPC access for your function.

[Cancel](#)[Create function](#)

Now upload the provided zip file to the function



Function overview

Diagram

Template

receiveEmail

Layers (0)

+ Add trigger

+ Add destination

Export to Infrastructure Composer

Download

Description

Last modified 40 minutes ago

Function ARN `arn:aws:lambda:us-east-1:831926625893:function:receiveEmail`

Function URL

Code

Test

Monitor

Configuration

Aliases

Versions

Code source

Upload from

.zip file

Amazon S3 location

EXPLORER

index.mjs

package.json

## Configuring the receiveEmail function

Add these environment variables

`FORWARDING_EMAIL` = [no-reply@domain.click]

`DOMAIN_NAME` = [domain.click]

`MAPPING_TABLE` = [DynamoDB Mapping Table that you created]

`S3_BUCKET` = [Your S3 name bucket you created]

## Change timeout to 15s

Code

Test

Monitor

Configuration

Aliases

Versions

General configuration

Triggers

Permissions

Destinations

Function URL

Environment variables

Tags

VPC

RDS databases

Monitoring and operations tools

General configuration

Description

Timeout 0 min 15 sec

Memory 128 MB

SnapStart

Ephemeral storage 512 MB

Edit

Do the same for the creating masked email function

Updating the function `makeMaskedEmail`.

### makeMaskedEmail

Throttle Copy ARN Actions

Function overview Info

Diagram Template

makeMaskedEmail

Layers (0)

+ Add trigger + Add destination

Export to Infrastructure Composer Download

Description

Last modified 12 seconds ago

Function ARN `arn:aws:lambda:us-east-1:831926625893:function:makeMaskedEmail`

Function URL Info

Code Test Monitor Configuration Aliases Versions

### Code source Info

Upload from

EXPLORER

MAKEMASKEDEMAIL

index.mjs

```
1 // AWS Lambda function for creating new masked email addresses
2 const AWS = require('aws-sdk');
3 const crypto = require('crypto');
4 const dynamodb = new AWS.DynamoDB.DocumentClient();
5
6 exports.handler = async (event) => {
7   // ...
8 }
```

```
import { DynamoDB } from '@aws-sdk/client-dynamodb';
import { DynamoDBDocument } from '@aws-sdk/lib-dynamodb';
import crypto from 'crypto';

const dynamoClient = new DynamoDB();
const dynamodb = DynamoDBDocument.from(dynamoClient);

export const handler = async (event) => {
  try {
    // Parse the request body
    const body = event.body || '{}';
    const realEmail = body.email;
    const description = body.description || '';

    console.log("Parsed the body")

    if (!realEmail) {
      return {
        statusCode: 400,
        body: JSON.stringify({ error: 'Real email address is
required' })
      };
    }
  }
}
```

```

// Generate a random string for the masked email
const randomString = crypto.randomBytes(8).toString('hex');
const maskedEmail = `${randomString}@${process.env.DOMAIN_NAME}`;

console.log("Generated strings")

// Store the mapping in DynamoDB
await dynamodb.put({
  TableName: process.env.MAPPING_TABLE,
  Item: {
    masked_email: maskedEmail,
    real_email: realEmail,
    description: description,
    created_at: new Date().toISOString(),
    settings: {
      active: true,
      forward_attachments: true,
      forward_html: true
    },
    stats: {
      emails_received: 0
    }
  }
});

console.log("DynamoDB item created")

// Return the newly created masked email
return {
  statusCode: 200,
  body: JSON.stringify({
    masked_email: maskedEmail,
    real_email: realEmail,
    created_at: new Date().toISOString()
  })
};
} catch (error) {
  console.error('Error creating masked email:', error);

  return {
    statusCode: 500,
    body: JSON.stringify({ error: 'Failed to create masked email' })
  };
}

```

```
}  
};
```

Change the timeout to 15s.

Environment variables to add:

`DOMAIN_NAME`

`MAPPING_TABLE`

## Set up Lambda IAM Role

This allows the lambda functions to access the resources needed.

Go to the **makeEmail** lambda function and click on the permissions tab in the configuration menu.

Code

Test

Monitor

Configuration

Aliases

Versions

General configuration

Triggers

Permissions

Destinations

Function URL

Environment variables

Tags

VPC

RDS databases

Execution role

Edit

View role document

Role name

receiveEmail-role-15t5jxxv

Resource summary

To view the resources and actions that your function has permission to access, choose a service.

Amazon CloudWatch Logs

3 actions, 2 resources

By action

By resource

Resource	Actions
arn:aws:logs:us-east-1:831926625893:*	Allow: logs:CreateLogGroup

Click on the role name link.

0

min

15

sec

Execution role

Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

☒ Use an existing role

☐ Create a new role from AWS policy templates

Existing role

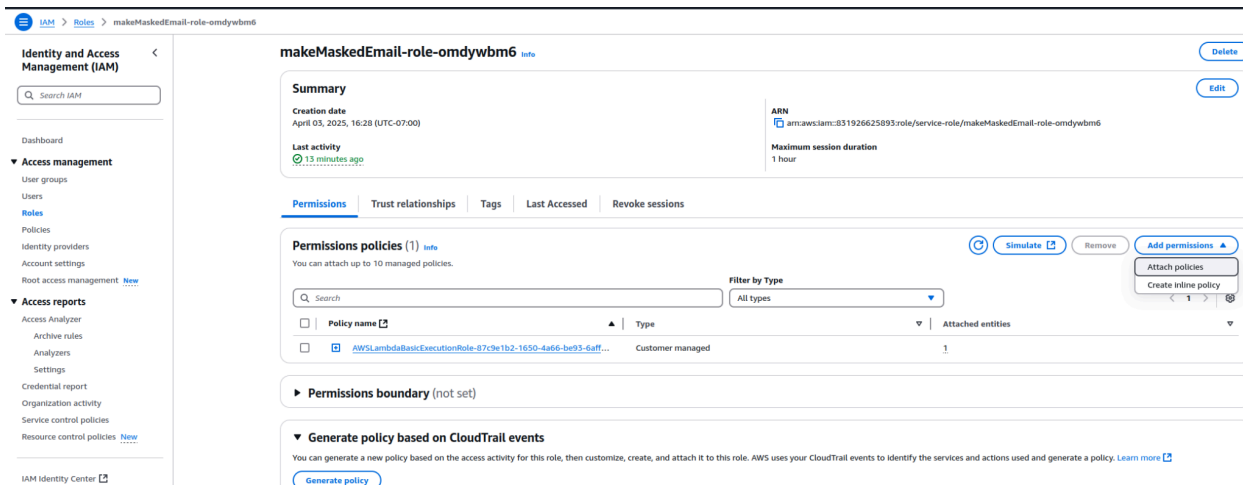
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch L

service-role/receiveEmail-role-15t5jxxv

View the receiveEmail-role-15t5jxxv role

 on the IAM console.

## Attach a policy



The screenshot shows the AWS IAM console interface for the role `makeMaskedEmail-role-omdywbm6`. The left sidebar contains navigation links for Identity and Access Management (IAM), Access management, and Access reports. The main content area displays the role's summary, including its creation date (April 03, 2025, 16:28 UTC-07:00) and last activity (13 minutes ago). The 'Permissions' tab is active, showing a list of permissions policies. The 'AmazonDynamoDBFullAccess' policy is highlighted, and the 'Add permissions' button is visible.

**makeMaskedEmail-role-omdywbm6** [Info](#) [Delete](#) [Edit](#)

**Summary**

Creation date: April 03, 2025, 16:28 (UTC-07:00)

Last activity: 13 minutes ago

ARN: `arn:aws:iam::831926625895:role/service-role/makeMaskedEmail-role-omdywbm6`

Maximum session duration: 1 hour

**Permissions** | Trust relationships | Tags | Last Accessed | Revoke sessions

**Permissions policies (1)** [Info](#)

You can attach up to 10 managed policies.

Filter by Type: All types

Policy name	Type	Attached entities
<input checked="" type="checkbox"/> <a href="#">AmazonDynamoDBFullAccess</a>	AWS managed	1

**Permissions boundary** (not set)

**Generate policy based on CloudTrail events**

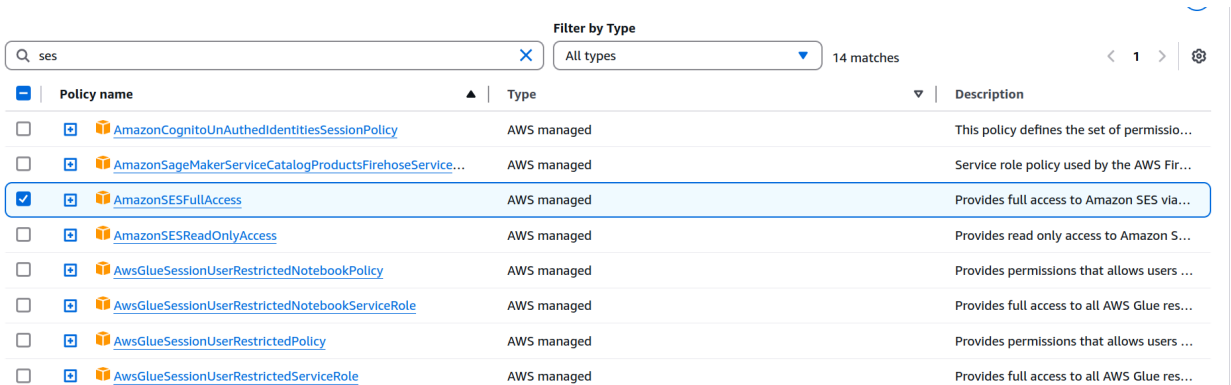
You can generate a new policy based on the access activity for this role, then customize, create, and attach it to this role. AWS uses your CloudTrail events to identify the services and actions used and generate a policy. [Learn more](#)

[Generate policy](#)

## Add the dynamoDBFullAccess

<input checked="" type="checkbox"/>	Policy name	Type	Description
<input checked="" type="checkbox"/>	<a href="#">AmazonDynamoDBFullAccess</a>	AWS managed	Provides full access to Amazon DynamoDB via the AWS Management Console.
<input type="checkbox"/>	<a href="#">AmazonDynamoDBFullAccesswithDataPipeline</a>	AWS managed	This policy is on a deprecation path. See documentation for guidance: <a href="https://docs.aws.amazon...">https://docs.aws.amazon...</a>
<input type="checkbox"/>	<a href="#">AmazonDynamoDBReadOnlyAccess</a>	AWS managed	Provides read only access to Amazon DynamoDB via the AWS Management Console.
<input type="checkbox"/>	<a href="#">AWSLambdaDynamoDBExecutionRole</a>	AWS managed	Provides list and read access to DynamoDB streams and write permissions to CloudWatch logs.
<input type="checkbox"/>	<a href="#">AWSLambdaInvocation-DynamoDB</a>	AWS managed	Provides read access to DynamoDB Streams.

Now do the same for the **receivingEmail** function, but also add the SES full access and S3 Read only access as well.



The screenshot shows the AWS IAM console search results for the term 'ses'. The search bar at the top shows 'ses' and '14 matches'. The results table lists several AWS managed policies, with 'AmazonSESEFullAccess' highlighted. The table columns are Policy name, Type, and Description.

Policy name	Type	Description
<input type="checkbox"/> <a href="#">AmazonCognitoUnAuthenticatedIdentitiesSessionPolicy</a>	AWS managed	This policy defines the set of permissio...
<input type="checkbox"/> <a href="#">AmazonSageMakerServiceCatalogProductsFirehoseService...</a>	AWS managed	Service role policy used by the AWS Fir...
<input checked="" type="checkbox"/> <a href="#">AmazonSESEFullAccess</a>	AWS managed	Provides full access to Amazon SES via...
<input type="checkbox"/> <a href="#">AmazonSESReadOnlyAccess</a>	AWS managed	Provides read only access to Amazon S...
<input type="checkbox"/> <a href="#">AwsGlueSessionUserRestrictedNotebookPolicy</a>	AWS managed	Provides permissions that allows users ...
<input type="checkbox"/> <a href="#">AwsGlueSessionUserRestrictedNotebookServiceRole</a>	AWS managed	Provides full access to all AWS Glue res...
<input type="checkbox"/> <a href="#">AwsGlueSessionUserRestrictedPolicy</a>	AWS managed	Provides permissions that allows users ...
<input type="checkbox"/> <a href="#">AwsGlueSessionUserRestrictedServiceRole</a>	AWS managed	Provides full access to all AWS Glue res...

Other permissions policies (1/1045)

Filter by Type: All types 13 matches

Policy name	Type	Description
<input type="checkbox"/> <a href="#">AmazonDMSRedshiftS3Role</a>	AWS managed	Provides access to manage S3 sett
<input type="checkbox"/> <a href="#">AmazonS3FullAccess</a>	AWS managed	Provides full access to all buckets
<input type="checkbox"/> <a href="#">AmazonS3ObjectLambdaExecutionRolePolicy</a>	AWS managed	Provides AWS Lambda functions p
<input type="checkbox"/> <a href="#">AmazonS3OutpostsFullAccess</a>	AWS managed	Provides full access to Amazon S3
<input type="checkbox"/> <a href="#">AmazonS3OutpostsReadOnlyAccess</a>	AWS managed	Provides read only access to Amaz
<input checked="" type="checkbox"/> <a href="#">AmazonS3ReadOnlyAccess</a>	AWS managed	Provides read only access to all bu
<input type="checkbox"/> <a href="#">AmazonS3TablesFullAccess</a>	AWS managed	Provides full access to all S3 table
<input type="checkbox"/> <a href="#">AmazonS3TablesReadOnlyAccess</a>	AWS managed	Provides read only access to all S3
<input type="checkbox"/> <a href="#">AWSBackupServiceRolePolicyForS3Backup</a>	AWS managed	Policy containing permissions necc
<input type="checkbox"/> <a href="#">AWSBackupServiceRolePolicyForS3Restore</a>	AWS managed	Policy containing permissions necc
<input type="checkbox"/> <a href="#">AWSQuickSetupSSMDeploymentS3BucketRolePolicy</a>	AWS managed	This policy grants permissions for
<input type="checkbox"/> <a href="#">QuickSightAccessForS3StorageManagementAnalyticsReadOnly</a>	AWS managed	Policy used by QuickSight team to
<input type="checkbox"/> <a href="#">S3UnlockBucketPolicy</a>	AWS managed	Provides access required to unlock

Cancel Add p

## Testing/Creating a new masked email

Now run a test of the code with a lambda test. This will be the makeEmail test.

Test event Info Delete CloudWatch Logs Live Tail Save Test

To invoke your function without saving an event, modify the event, then choose Test. Lambda uses the modified event to invoke your function, but does not overwrite the original event until you choose Save.

Test event action

☐ Create new event ☒ Edit saved event

Event name

test

Event JSON

Format JSON

```
1 [{"body": {"email": "lethalson2019@gmail.com", "description": "My first masked email"}}]
```

1:1 JSON Spaces: 4

Use the email you would like to mask here

```
{"body": {"email": "[email]", "description": "My first masked email"}}
```

Hit test.

You should get this type of execution

Executing function: succeeded (logs)

Details

```
{
  "statusCode": 200,
  "body": "[{"masked_email":"914db82c58d62028k12stentest.click","real_email":"lethaison2019@gmail.com","created_at":"2025-04-04T06:49:03.890Z"}]"
```

Summary

Code SHA-256

Oxstb/Xcjt9mayCwCulFRGZKfp8hIN2HexHZWita8o=

Function version

\$LATEST

Duration

873.60 ms

Resources configured

128 MB

Init duration

417.08 ms

Log output

The area below shows the last 4 KB of the execution log. [Click here](#) to view the corresponding CloudWatch log group.

```
START RequestId: c8cbcaa3-e628-427e-8381-23f1dd29f01c Version: $LATEST
2025-04-04T06:49:03.076Z c8cbcaa3-e628-427e-8381-23f1dd29f01c INFO Parsed the body
2025-04-04T06:49:03.107Z c8cbcaa3-e628-427e-8381-23f1dd29f01c INFO Generated strings
2025-04-04T06:49:03.890Z c8cbcaa3-e628-427e-8381-23f1dd29f01c INFO DynamoDB item created
END RequestId: c8cbcaa3-e628-427e-8381-23f1dd29f01c
REPORT RequestId: c8cbcaa3-e628-427e-8381-23f1dd29f01c Duration: 873.60 ms Billed Duration: 874 ms Memory Size: 128 MB Max Memory Used: 93 MB Init Duration: 417.08 ms
```

Check the DynamoDB table as well to see your masked email.

DynamoDB

Tables

DynamoDB

Dashboard

Tables

Explore items

PartiQL editor

Backups

Exports to S3

Imports from S3

Integrations

Reserved capacity

Tables (1)

Find tables

	Name	Status	Partition key	Sort key
<input type="checkbox"/>	<a href="#">maskedEmailDB</a>	Active	masked_email (S)	-

Explore table items

Tables (1)

Any tag key

Any tag value

Find tables

maskedEmailDB

maskedEmailDB

Settings

Indexes

Monitor

Global tables

Backups

Exports and streams

Permissions

Protect your DynamoDB table from accidental writes and deletes

When you turn on point-in-time recovery (PITR), DynamoDB backs up your table data automatically so that you can restore to any given second in the preceding 1 to 35 days. Additional charges apply. [Learn more](#)

Edit PITR

General information

Partition key

masked\_email (String)

Sort key

-

Capacity mode

On-demand

Table status

Active

Alarms

Point-in-time recovery (PITR)

Item count

Table size

Get live item count

It should have something like this.

Items returned (1)

<input type="checkbox"/>	masked_email (String) ▾	created_at ▾	description ▾	real_email ▾	settings ▾	stats
<input type="checkbox"/>	<a href="#">914dbd82c58d6202@k12...</a>	2025-04-04...	My first mask...	lethaison20...	{ "active": {...	{ "emails_received": { "N": "1" } }

You can also test the receive email function to see if it runs.

Below is the json AWS SES will send to the lambda. You can create a new test with this example.

```
{
  "Records": [
    {
      "eventSource": "aws:ses",
      "eventVersion": "1.0",
      "ses": {
        "mail": {
          "timestamp": "2025-04-04T21:29:50.765Z",
          "source": "lethaison2019@gmail.com",
          "messageId": "a0fv0egdj005b2gmcocvn2u9idpdgl9bu0c8ld01",
          "destination": [
            "914dbd82c58d6202@k12stemtest.click"
          ],
          "headersTruncated": false,
          "headers": [
            {
              "name": "Return-Path",
              "value": "<lethaison2019@gmail.com>"
            },
            {
              "name": "Received",
              "value": "from mail-pj1-f49.google.com
(mail-pj1-f49.google.com [209.85.216.49]) by
inbound-smtp.us-east-1.amazonaws.com with SMTP id
a0fv0egdj005b2gmcocvn2u9idpdgl9bu0c8ld01 for
914dbd82c58d6202@k12stemtest.click; Fri, 04 Apr 2025 21:29:50 +0000 (UTC)"
            },
            {
              "name": "X-SES-Spam-Verdict",
              "value": "PASS"
            }
          ]
        }
      }
    }
  ]
}
```



```

        {
            "name": "X-SES-Virus-Verdict",
            "value": "PASS"
        },
        {
            "name": "Received-SPF",
            "value": "pass (spfCheck: domain of
_spf.google.com designates 209.85.216.49 as permitted sender)
client-ip=209.85.216.49; envelope-from=lethaison2019@gmail.com;
helo=mail-pj1-f49.google.com;"
        },
        {
            "name": "Authentication-Results",
            "value": "amazonses.com; spf=pass (spfCheck:
domain of _spf.google.com designates 209.85.216.49 as permitted sender)
client-ip=209.85.216.49; envelope-from=lethaison2019@gmail.com;
helo=mail-pj1-f49.google.com; dkim=pass header.i=@gmail.com; dmarc=pass
header.from=gmail.com;"
        },
        {
            "name": "X-SES-RECEIPT",
            "value":
"AEFBQUFBQUFBQUFFTFFNVWUwZ0wwTEU4ZE54d0YxS3I4NnBBSnFndE9mNTk3UE81WWNIvXk5NV
N0SH1ld1FHTDA0WHYvQzJ6d245T2ZtVHNWb0lOU1JFRk10Y1Bva0g5TDdwMWw1UU5WczlnNVFCN
nlnG1XMmZUU1paeVVTN21xN3dZSFdzNFJDTExHYVViZ3EzTnJSNlZ0ZG1welZFBzM2eXAzR0xK
M1JxdDN0bF12c0gvY1ZyUHpldCtkU25MKzRhbEMzN09seEM1NTBra3U4U0xvRXR5WENHbURmaXF
3eHhicXQ4ajhBT1NyNDNTUlk4UXVlN0VKWU9wY0pNbUdJUUVgrRzNBYXA4eFlaSEhYSnJmUVQ2K3
BvRDRORysrR0s3bkFzWjIwMnVMM24yeldoaS81K0E9PQ=="
        },
        {
            "name": "X-SES-DKIM-SIGNATURE",
            "value": "a=rsa-sha256; q=dns/txt;
b=tg8DXavp8P+rdC/cuIUSHgUBKwOPSTnI4Sjh8Ym1sXqlj/4lvHPU6g0P56cHJFuusYYWCymbb
o3jPfUqIOs0KFhecDxjhWiM4kIENjHich/wbmX8RxWrNswKaDHKc8CIgFPejmANiB0k3gqNzMc6
Tk0jUtrSo7hNx28wu7MXD2w=; c=relaxed/simple;
s=6gbrjpgwjskckoa6a5zn6fwqkn67xbtw; d=amazonses.com; t=1743802191; v=1;
bh=ma4bidNDdb+4SkjU3n/LC1i3dRfgCI2cXwmezxt+wVU=;
h=From:To:Cc:Bcc:Subject:Date:Message-ID:MIME-Version:Content-Type:X-SES-RE
CEIPT;"
        },
        {
            "name": "Received",
            "value": "by mail-pj1-f49.google.com with

```

```
SMTP id 98e67ed59e1d1-3032aa1b764so1945901a91.1 for
<914dbd82c58d6202@k12stemtest.click>; Fri, 04 Apr 2025 14:29:50 -0700
(PDT)"

    },
    {
        "name": "DKIM-Signature",
        "value": "v=1; a=rsa-sha256;
c=relaxed/relaxed; d=gmail.com; s=20230601; t=1743802189; x=1744406989;
darn=k12stemtest.click;
h=to:subject:message-id:date:from:mime-version:from:to:cc:subject:date:mess
age-id:reply-to; bh=ma4bidNDdb+4SkjU3n/LC1i3dRfgCI2cXwmezxt+wVU=;
b=IA/vub/0N1kT6GZWZm4QMxUQdwywlukLYZ3eM19Zz9uTEFXLPBiCb/f9SvTkMmrLEmzag2GrQ
MgvVL/0Ea7dfvEuF1/XGBUbQ0c9DHQnBKkVMUTuY1J2k0w5KkrsZE9yLMSwJ0ZWvYA+GtF3774R
iW2QBvJzTmDLlmmM18+mieOHq87xzb1L3umAU/WCR9oEw7CtcJxCsDr1J/XvX819d3qcsNvh+/Q
uprluRD12rE4z2e9B317NPbr6TdVv5dSCJz5WG6w0Lb6P+Xtzj606P9xvLz1b4iCtKjhaObGstj
sH1WRdQWJHKiUt7Tea5/w3MyS8Brw4qYh+adkk1yhu+A=="
    },
    {
        "name": "X-Google-DKIM-Signature",
        "value": "v=1; a=rsa-sha256;
c=relaxed/relaxed; d=1e100.net; s=20230601; t=1743802189; x=1744406989;
h=to:subject:message-id:date:from:mime-version:x-gm-message-state
:from:to:cc:subject:date:message-id:reply-to;
bh=ma4bidNDdb+4SkjU3n/LC1i3dRfgCI2cXwmezxt+wVU=;
b=EFENVdW2UPvfGmiJ3Iqh0PqoquB5MiuU4ab9Es11zfMmJz3Nxp1WacDTgPXtJ/z800
kKoF/Bp7WGHJ7N0h7EacdhRYH+h6MiVGPL7Od43QIdfugxRZvNqWXwCmz9dT25essGS
BgZJJghZVMxUFJgiJmSQL1vvo4bFmpOGof2BqyPJzXvo2s+8BNf0Vsaz6NU8sM3Eu2fM
DbtL7CnaNKctz+rWfH9ghkuNovZML0S+HKwx4ppfZXvCXg71f6wbU0tMuoB6haT6HnN1
TMDUQPOvvvtfJ52tiQ+7qE8FiZ6YQyF00aH9822TMMCWZF7STBk7Mh0yhy6znK7A9N0x
/lig=="
    },
    {
        "name": "X-Gm-Message-State",
        "value":
"A0Ju0YxrdJNkjyGD3RL4MBjP8qvXo1UP40V6g+t0XB8U2ZjPdvouEwHt
O+9CnknSxqkWV4sSd4HqVIaUdw3t3x+u3XdzYyM4P5IW2ccvUjULsMHefMKLS4wykZ3GrpcmlmD
KtWfPDXm7dit40jFuSASqVUKjtla8qw=="
    },
    {
        "name": "X-Gm-Gg",
        "value":
"ASbGncuW05gevXqpp0Q/81VN+RG9z37fy9R10YnFzIRDN6FffLXHsTo60We2f1r0w9R
yuv6083N/OPxHO+KsPmo3G+Wf8FbTRTrZXE97Wdyg7V+kRj4mrUsHR+dxULAg/E8D4mFdY48hvN
```

```
nbckqWvGKBhWsFiZGjo03oCFKB8vw8Bd0AvlwN"
    },
    {
        "name": "X-Google-Smtp-Source",
        "value":
"AGHT+IGvc3Ai5NahZzaWgVd7QHERCpM1b60gC6B0opsgAo4hQdf45+z0HFtyVTVBLB3UQCEDXV
cKQnlMUev3A2FSXgs="
    },
    {
        "name": "X-Received",
        "value": "by
2002:a17:90a:d64c:b0:305:5f28:2d5c with SMTP id
98e67ed59e1d1-306a6173b2fmr5759506a91.15.1743802189468; Fri, 04 Apr 2025
14:29:49 -0700 (PDT)"
    },
    {
        "name": "MIME-Version",
        "value": "1.0"
    },
    {
        "name": "From",
        "value": "Thaison Le
<lethaison2019@gmail.com>"
    },
    {
        "name": "Date",
        "value": "Fri, 4 Apr 2025 14:29:38 -0700"
    },
    {
        "name": "X-Gm-Features",
        "value":
"ATxdqUHRdsBwRbKGF8FkPRo2d_ULPz50kgmimkjf6E2MzY9n4gonMsdIrppAVd4"
    },
    {
        "name": "Message-ID",
        "value":
"<CAHmH_jdV1thzR65oYP6=1_Q4jDtVA0JGCwXo3xd8zSH8hue7kQ@mail.gmail.com>"
    },
    {
        "name": "Subject",
        "value": "my love"
    },
    {
```

```
        "name": "To",
        "value": "914dbd82c58d6202@k12stemtest.click"
    },
    {
        "name": "Content-Type",
        "value": "multipart/alternative;
boundary=\"00000000000000fdd180631fa96ba\""
    }
],
"commonHeaders": {
    "returnPath": "lethaison2019@gmail.com",
    "from": [
        "Thaison Le <lethaison2019@gmail.com>"
    ],
    "date": "Fri, 4 Apr 2025 14:29:38 -0700",
    "to": [
        "914dbd82c58d6202@k12stemtest.click"
    ],
    "messageId":
"<CAHmH_jdV1thzR65oYP6=1_Q4jDtVA0JGCwXo3xd8zSH8hue7kQ@mail.gmail.com>",
    "subject": "my love"
}
},
"receipt": {
    "timestamp": "2025-04-04T21:29:50.765Z",
    "processingTimeMillis": 804,
    "recipients": [
        "914dbd82c58d6202@k12stemtest.click"
    ],
    "spamVerdict": {
        "status": "PASS"
    },
    "virusVerdict": {
        "status": "PASS"
    },
    "spfVerdict": {
        "status": "PASS"
    },
    "dkimVerdict": {
        "status": "PASS"
    },
    "dmarcVerdict": {
        "status": "PASS"
    }
}
```

```

    },
    "action": {
        "type": "Lambda",
        "functionArn":
"arn:aws:lambda:us-east-1:831926625893:function:receiveEmail",
        "invocationType": "Event"
    }
}
]
}

```

You should get a result that looks like this

✔ Executing function: succeeded ([logs](#))

▼ Details

```

{
  "statusCode": 500,
  "body": "Error processing email: Failed to retrieve email: The specified key does not exist."
}

```

**Summary**

<p><b>Code SHA-256</b></p> <p>BerG5gyb5zTvUoZxmOW4v065UazAOAkWwo/i2ohzCM=</p>	<p><b>Execution time</b></p> <p>2 seconds ago</p>
<p><b>Request ID</b></p> <p>f4c5adfa-2730-4d21-9ce0-1b849f128479</p>	<p><b>Duration</b></p> <p>1070.01 ms</p>
<p><b>Billed duration</b></p> <p>1071 ms</p>	<p><b>Resources configured</b></p> <p>128 MB</p>
<p><b>Max memory used</b></p> <p>109 MB</p>	<p><b>Init duration</b></p> <p>822.00 ms</p>

**Log output**

The area below shows the last 4 KB of the execution log. [Click here](#) to view the corresponding CloudWatch log group.

```

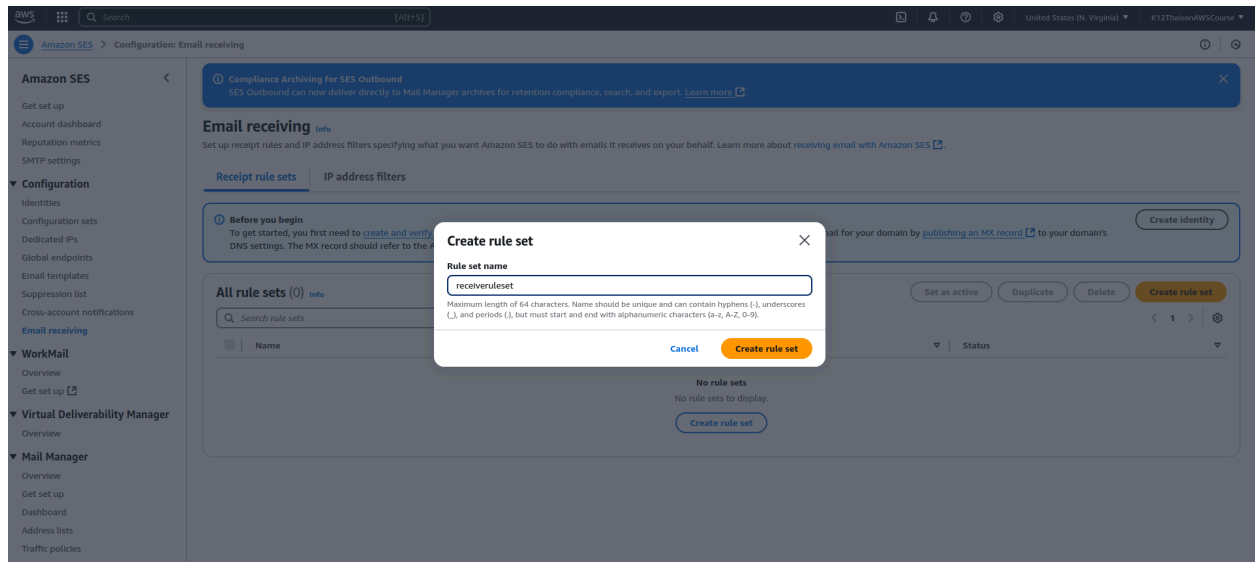
f49.google.com with SMTP id 98e67ed59e1d1-3032aa1b764aol945901a91.1 for <914dbd82c58d62028k12atentest.click>; Fri, 04 Apr 2025 14:29:50 -0700 (PDT)],{"name":"DKIM-Signature","value":"v=1; a=rsa-sha256; c=relaxed/relaxed; d=gmail.com; s=20230601; t=1743802189; x=1744406989; darn=k12atentest.click; h=to:subject:message-id:date:from:mime-version:from:to:cc:subject:date:message-id:reply-to; bh=ma4bi4NDdb+48k303n/LC1l3dRfgC12cXwexzt+wVU=; b=IA/vub/ON1kT6GZM2e4QMxQdwyw1ukLYT3eM192z9uTEFXLP81Cb/E95vTkMerrLEmzag20cQMgvVL/0Ea7dfvEuF1/XGB0Qo=90uqgSRXVMB7uTfL12k0w3krs2E9yJMSu=J0BWWYA-QcF3774k1W2Q8v0rTmDl1m8M18+e1e08q7xab113um0/WCK9v8w7Ct.cj0cAdrlJ/XvX819d3guwhh//QuprlR0D12z4z2e9B317Nfbz6TdvVv4dSCz5W06w0Lb6F+Xc=zj606F9xvLz1b41cK3haQbGst:jdl1WRdQWJXK1Uk7Tea5/w3My88r4qth+adk1lyhuA="}),{"name":"X-Google-DKIM-Signature","value":"v=1; a=rsa-sha256; c=relaxed/relaxed; d=1e100.net; s=20230601; t=1743802189; x=1744406989; h=to:subject:message-id:date:from:mime-version:x-gm-message-state :from:to:cc:subject:date:message-id:reply-to; bh=ma4bi4NDdb+48k303n/LC1l3dRfgC12cXwexzt+wVU=; b=EFENVDw2UPvfGmiJ3Iq8FQoqu85MIu04ab9KallzFMnJz3Nxp1MacDTgPxtJ/z800 kKof/Bp7WGHJ7N0h7EacdhRYH+h6MLVGPL70d43Q1dfugxR2vNgwX0cCms9dT25easGSBg2J3gh2VWx0FJ3g13eSGL1vvo4bFmpOcof2BqyF0zXwo2s+8BNF0Vear6NU8aM3EuzFM DbtL7CnaNGCt+s+WFH9ghkuNov2MLO8+HRwa4ppfEXvCkg71f6wb00cMusB6ha76Hn1 TM0UQOvOvvtFJ52t:ig+7q88F126Ygyf00ah98227M8CW2F7ST8k7MbOyhy6znKTA9Nok /liq="}],{"name":"X-Gm-Message-State","value":"AASu0Yxasd3Nk3yGD3R14MBjF8qXo1UP40V0v6vKXGB8U2ZjPdovubWlt 0+9CnksSagkMW4s4sd48qC7aBda3k3e+u3XdYtYwAF51W2cevc0J0Ls9MeEMCLs+4wykE3Gqponlnd KtWFFDxm7dit40JFuSASqV0k3jla8qe="},{"name":"X-Gm-Gg","value":"ASBdGneuW05gevXgppQ/81VN+RG9z37fy9R10YnfsIRGN6FfILkHsTo60Me2f1r0w9R yuv6083N/OPxHO+KaPmo3G+WE8FbTKTrZXES97Wdyg7V+krJ4arUsHR+dx0LLaQ/ERD4eFdy48hVn

```

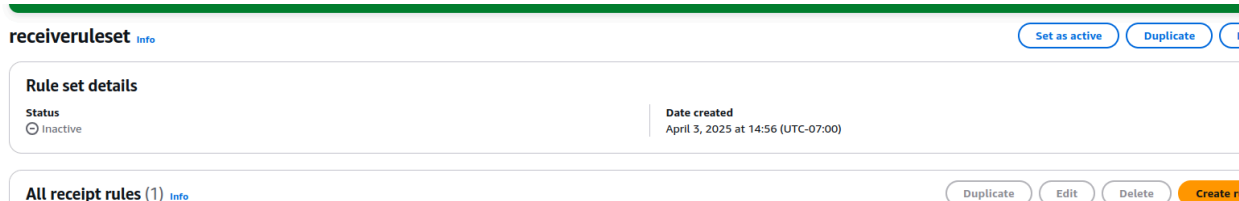
## AWS SES Rule set

We need a ruleset to make sure the emails trigger the lambda function.

Create a new ruleset. Name it something that makes sense.



Make sure you activate the rule set by clicking set as active.



Create a new rule. Also name it something that makes sense.

**Define rule settings** [Info](#)

**Receipt rule details**

**Rule name**

receiveEmail

Maximum length of 64 characters. Name should be unique and can contain hyphens (-), underscores (\_), and periods (.), but must start and end with alphanumeric characters (a-z, A-Z, 0-9).

**Status**

Amazon SES only runs enabled receipt rules within the active rule set. Uncheck this option if you don't want SES to run this rule.

☒ Enabled

**Security and protection options**

**Transport Layer Security (TLS)**

Select this option if you want Amazon SES to reject any incoming messages that aren't sent over a secure connection.

☐ Required

**Spam and virus scanning**

Select this option if you want Amazon SES to scan incoming messages for spam and viruses.

☒ Enabled

[Cancel](#) [Next](#)

Add a recipient condition. This is basically how to match the email. Put your domain there to catch all emails sent to AWS SES.

when the recipient of an incoming message matches the recipient conditions of a receipt rule, Amazon SES performs an ordered list of actions associated with that rule.

#### ▼ Guidelines

If you want to...

Then specify the following recipient condition...

Match a specific email address.

*user@example.com*

Match all addresses within a domain, but not those within its subdomains.

*example.com*

Match all addresses within a specific subdomain, but not those within the parent domain.

*subdomain.example.com*

Match all addresses within all subdomains, but not those within the parent domain.

*.example.com*

Match all addresses within a domain, and all addresses within all of its subdomains.

*example.com and .example.com*

Match all recipients in all verified domains.

None

#### Recipient conditions Info

① Amazon SES can only receive mail on your behalf for domains that you own. Any email address that you specify as a recipient condition must belong to a [verified domain identity](#).

Recipient condition

k12stemtest.click

Remove

Add new recipient condition

You can add 499 more recipient conditions.

Cancel

Previous

Next

Now we create actions to perform when it catches an email.  
For the first action, have the email delivered to the s3 bucket.

### 1. Deliver to Amazon S3 bucket [Info](#)

[Remove](#)

This action delivers the received email to an Amazon Simple Storage Service (S3) bucket.

#### S3 bucket

Specify the Amazon S3 bucket to which you want to save received emails. You must attach a policy to this bucket enabling Amazon SES to write to it.

[Create S3 bucket](#)

#### Object key prefix - optional

#### Message encryption

Optionally enable Amazon SES to encrypt received email messages before delivering them to the specified S3 bucket.

☐ Enabled

#### IAM role - optional

Specify which IAM role to use when delivering the received email to an Amazon Simple Storage Service (S3) bucket. [Learn more](#) on how to set up the role permissions.

[View role](#)[Create IAM Role](#)

#### SNS topic - optional

Specify which Amazon Simple Notification Service (SNS) topic to notify when this action is performed. If the SNS topic belongs to another account, you must give Amazon SES permission to publish to the topic.

[Create SNS topic](#)

Then have it invoke the receive email lambda function. Make sure the s3 bucket action is first though. Use the arrows to move the s3 bucket to the top if it isn't.

[Amazon SES](#) > [Configuration: Email receiving](#) > [Create rule](#)

- Step 1  
Define rule settings
- Step 2 - optional  
Add recipient conditions
- Step 3
- Step 4
- Review

### Add actions [Info](#)

A **receipt rule** consists of an ordered list of actions that Amazon SES performs whenever the recipient of an incoming message matches a recipient specified as a condition of that rule.

#### 1. Invoke AWS Lambda function [Info](#)

[Remove](#)

This action calls your code via an AWS Lambda function.

##### Lambda function

Specify which lambda function you want Amazon SES to invoke. You must attach a policy to this function enabling Amazon SES to invoke it.



##### Invocation type

Specify whether to invoke the Lambda function synchronously or asynchronously.

☒ Event Invocation

Execution of the function is invoked asynchronously. Amazon SES recommends this invocation type.

☐ RequestResponse Invocation

Execution of the function is invoked synchronously and its response is used to control mail flow.

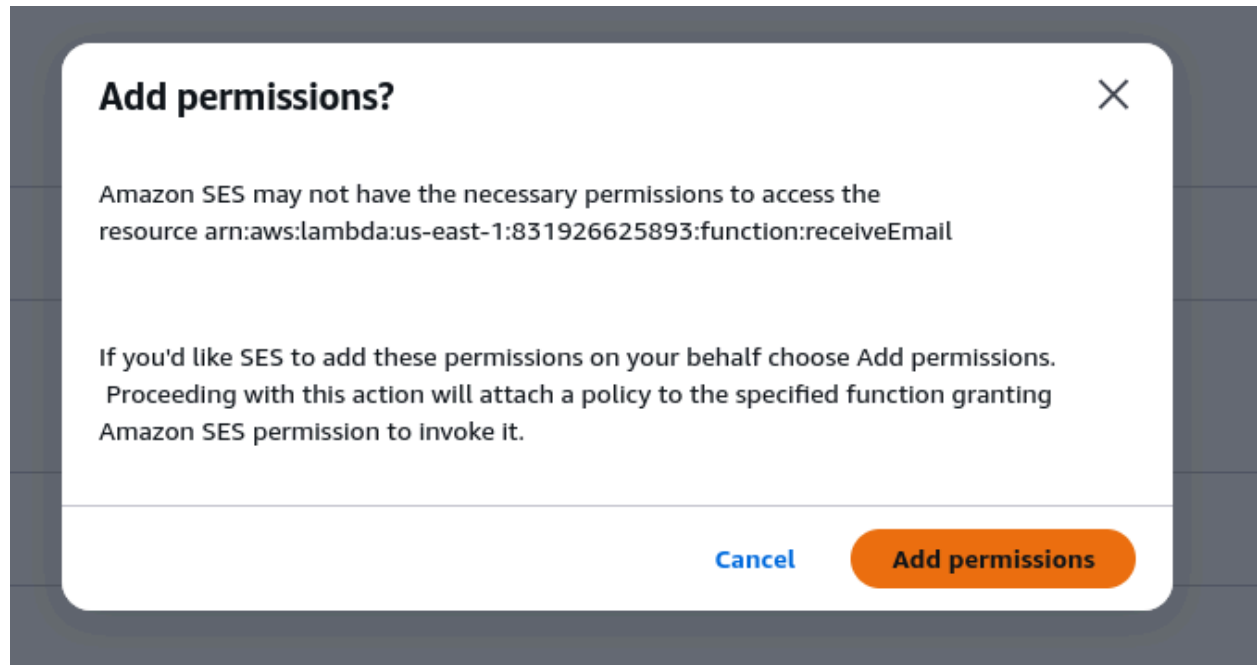
##### SNS topic - optional

Specify which Amazon Simple Notification Service (SNS) topic to notify when this action is performed. If the SNS topic belongs to another account, you must give Amazon SES permission to publish to the topic.

[Create SNS topic](#)[Add new action](#)[Cancel](#)[Previous](#)[Next](#)

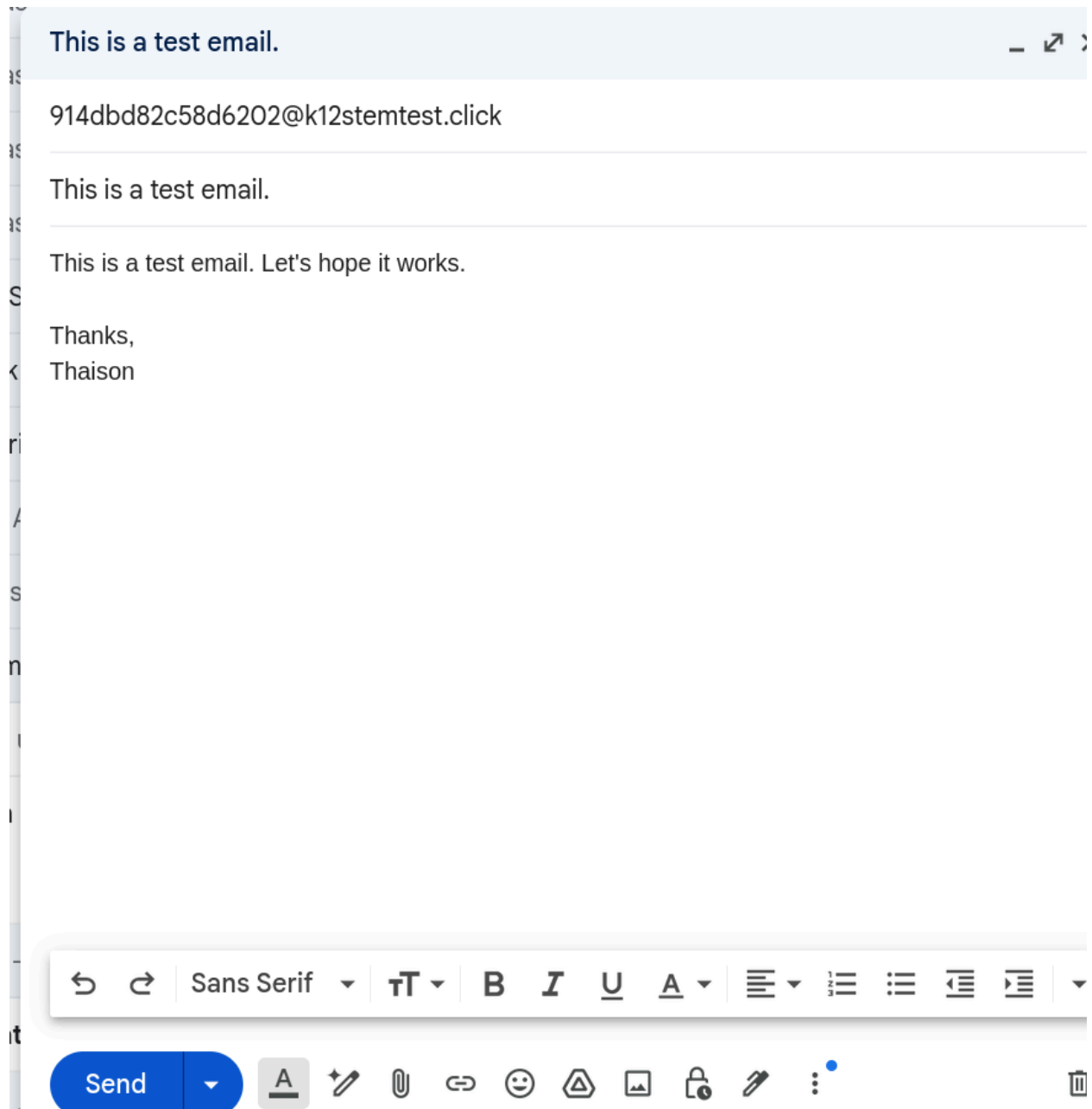
SES will ask for permissions to call the bucket. Give it the permissions.





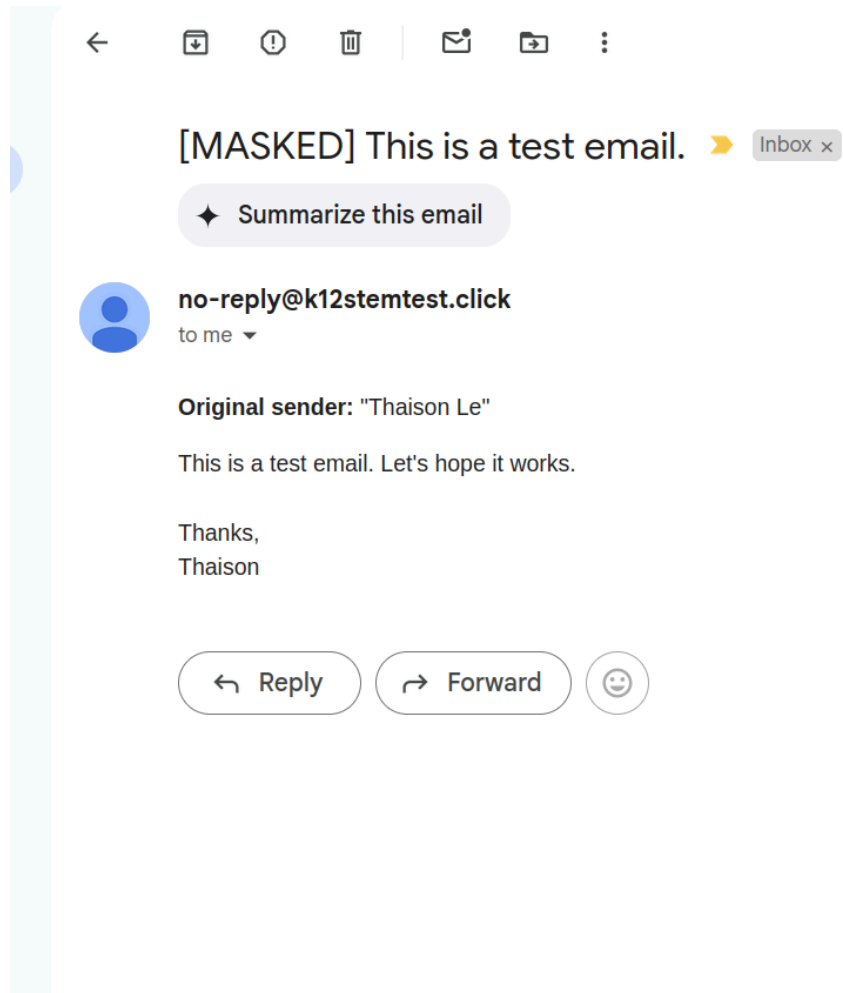
## Test the final product

Now that everything is set up, try sending an email to your masked email. If you don't remember it, you can find it in the dynamoDB table.



Wait 10-20 seconds. Now check your inbox.

You should receive an email that looks like this. Congratulations!



Potential Future improvements for you:

Two way communication? Replying to the masked email, similar to something like craigslist.

Web ui to create masked emails easily? You would also need to have the ability to authenticate (AWS Cognito)