Importing the Disposable Email Domains Data Feed to AWS S3

Posted on October 3, 2023

The intention of this document is to show you the basis of how to download the disposable email domain data feed provided by WhoisXML API to an AWS S3 bucket by leveraging a serverless Lambda function. AWS Lambda functions act as a serverless compute service that allows you to write and execute code without provisioning or managing servers. AWS S3 is an object storage service for storing and retrieving files. This document will guide you through the process of configuring both AWS Lambda and an AWS S3 bucket.

Out of scope:

- Scheduling a Lambda function
- ETL pipelining
- Importing the python requests module

Prerequisites

Please ensure you have the following setup:

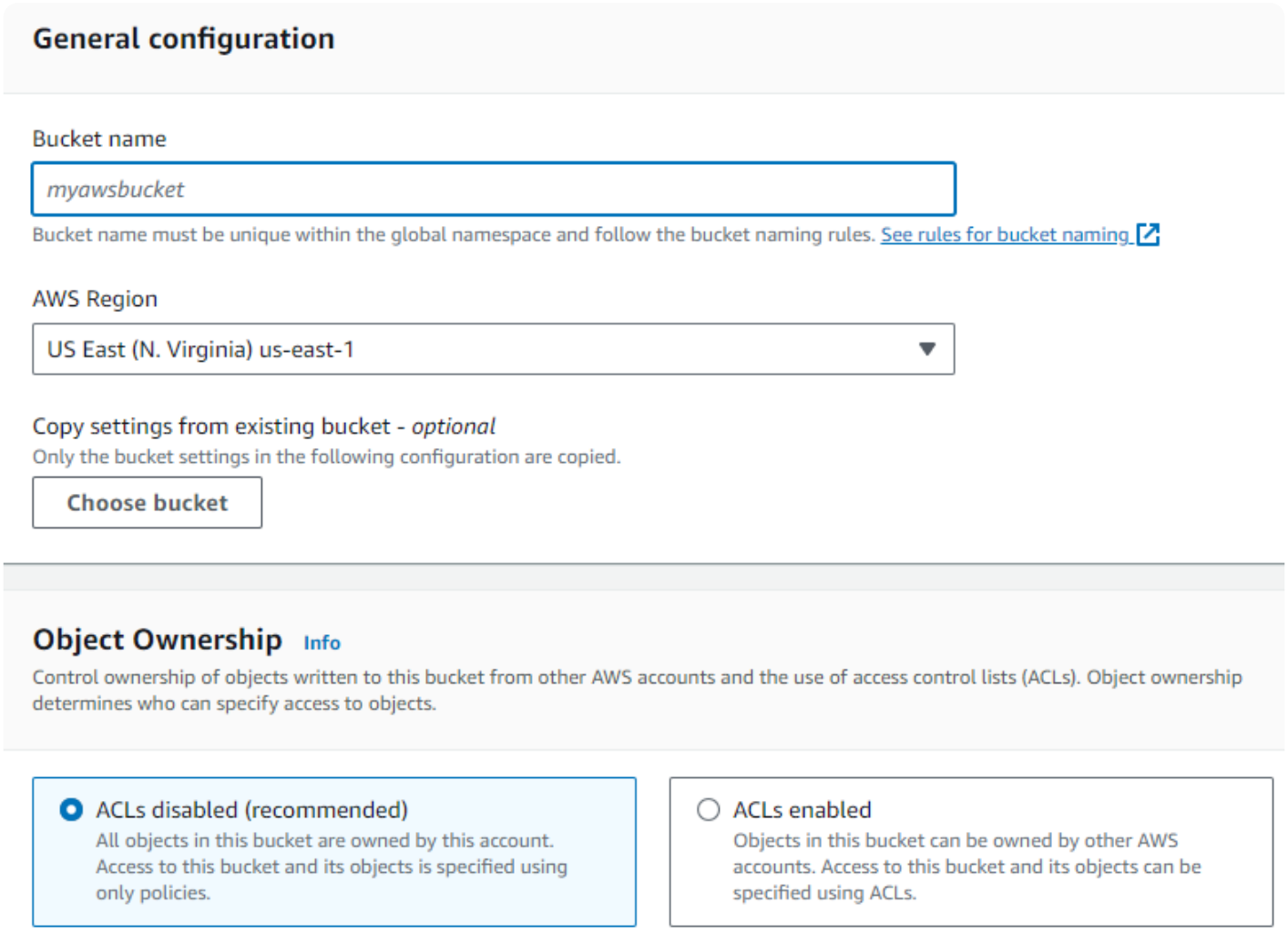
- AWS Account
- Basic to Intermediate knowledge of AWS services, specifically AWS Lambda and S3

- Some familiarity with Python which will be used in the Lambda function
- Access to the WHOIS API Disposable Email Domains data feed. You will need an API key with access to the data feed. Please contact us for more information. For more information on the specifications of the data feed, please visit here.

Step 1: Create an AWS S3 Bucket

The first step is to create an S3 bucket to write the Disposable Email Domains files to.

- In the AWS Management Console, navigate to the S3 service.
- Click on “Create Bucket”.
- Give the bucket a unique name and select the appropriate region.

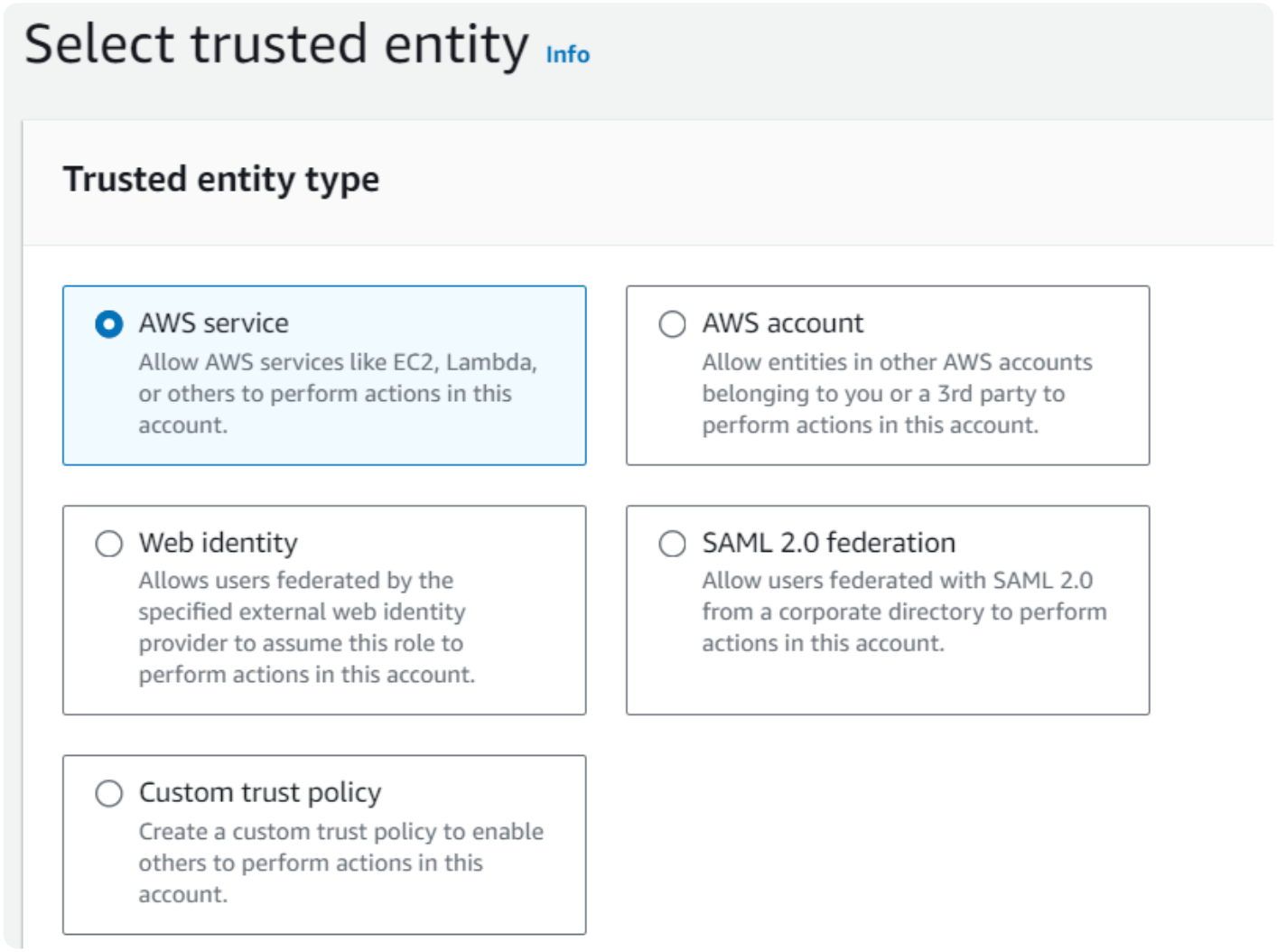


- At this time, leave the default settings and go ahead and click “Create Bucket”.

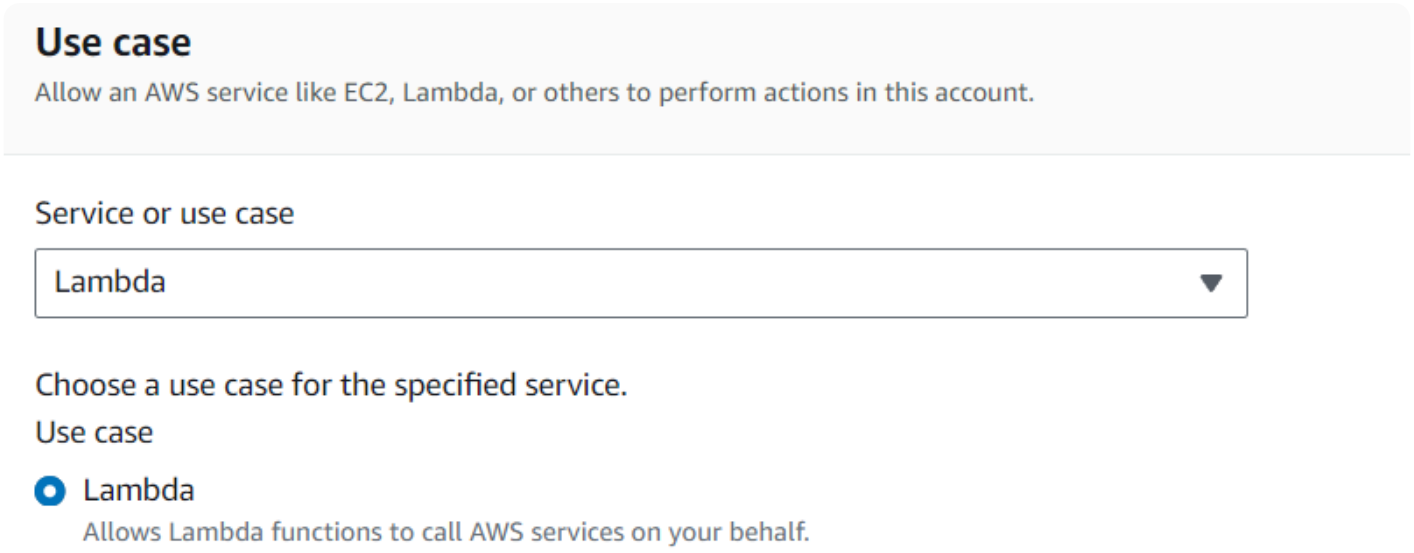
Step 2: Create an IAM Role

AWS Lambda will require an IAM role with the permissions necessary to read/write to the S3 bucket. Please follow these steps to create an IAM role:

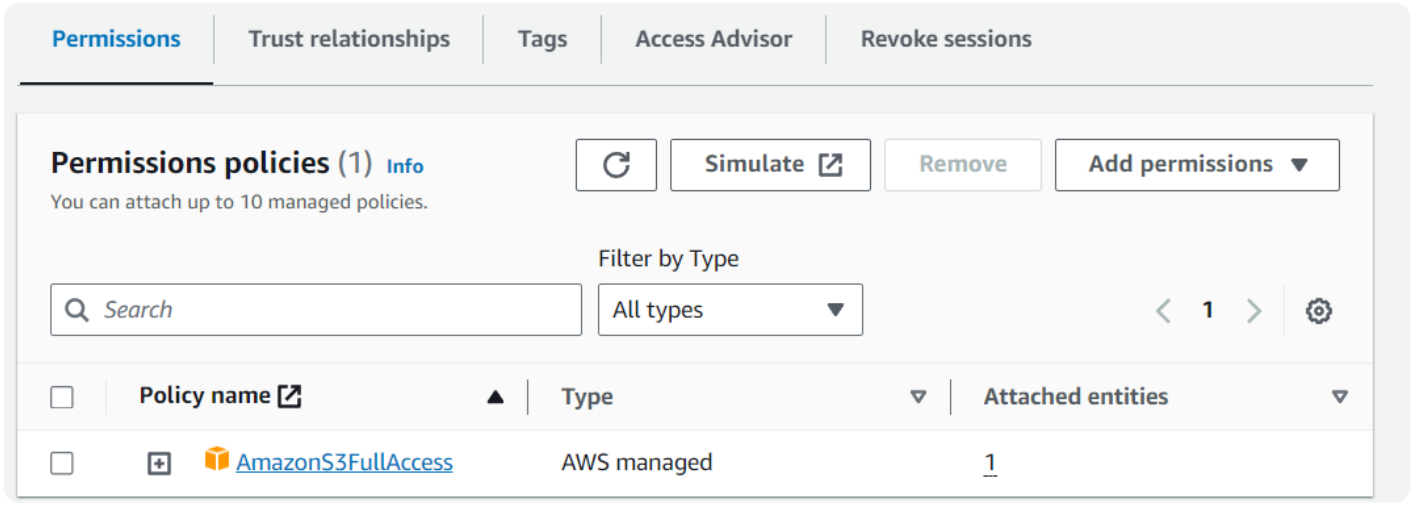
- Navigate to the IAM Service in the AWS management console.



- Click on “Roles” and then followed by “Create Role”.
- Select “Lambda” as the service for this role, and then click “Next: Permissions”.



- In the input search bar, type “S3” and then select “AWSS3FullAccess” followed by “Next: Tags”.



- Tags are optional, then click “Next: Review”.
- Give your Role and name and provide a brief description, followed by “Create Role”.

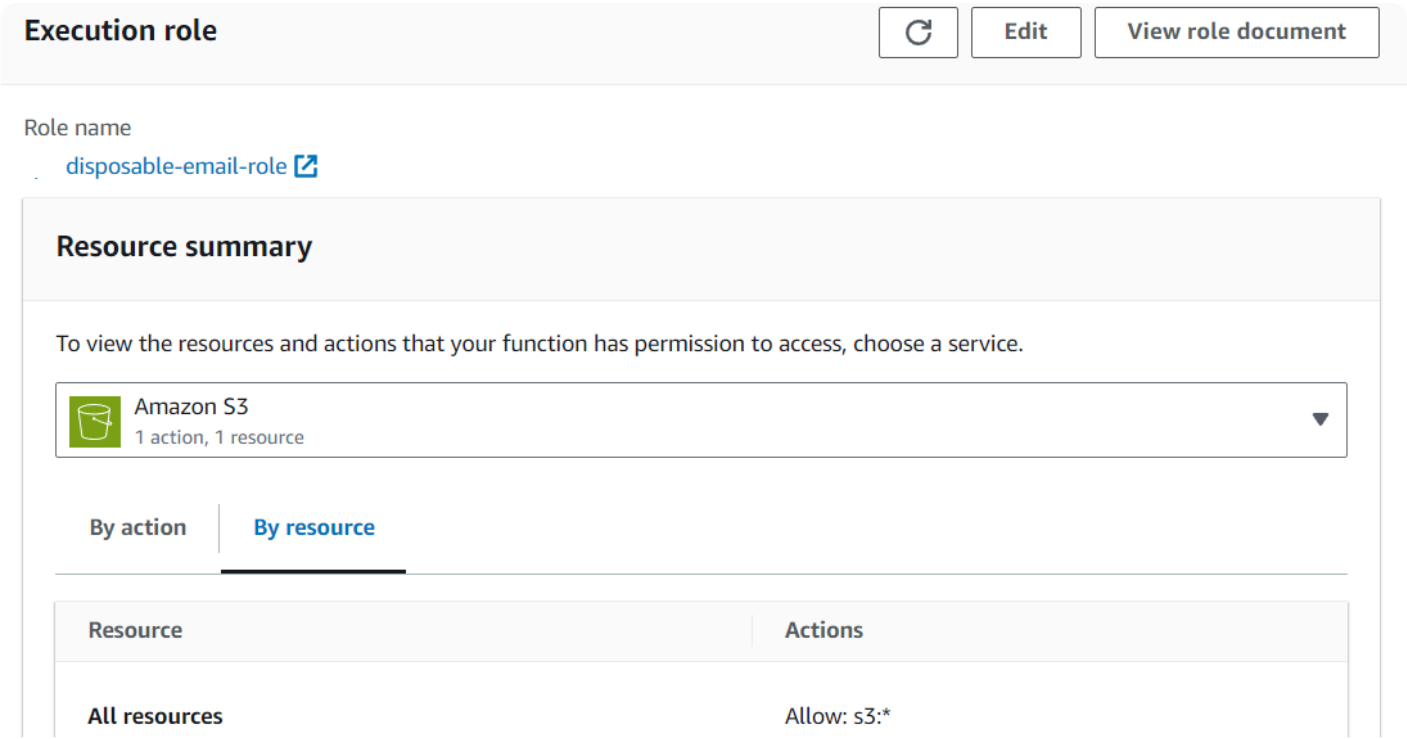
Step 3: Creating a Lambda Function

Now the magic begins. Creating Lambda functions is fun, and easy. To create a Lambda function:

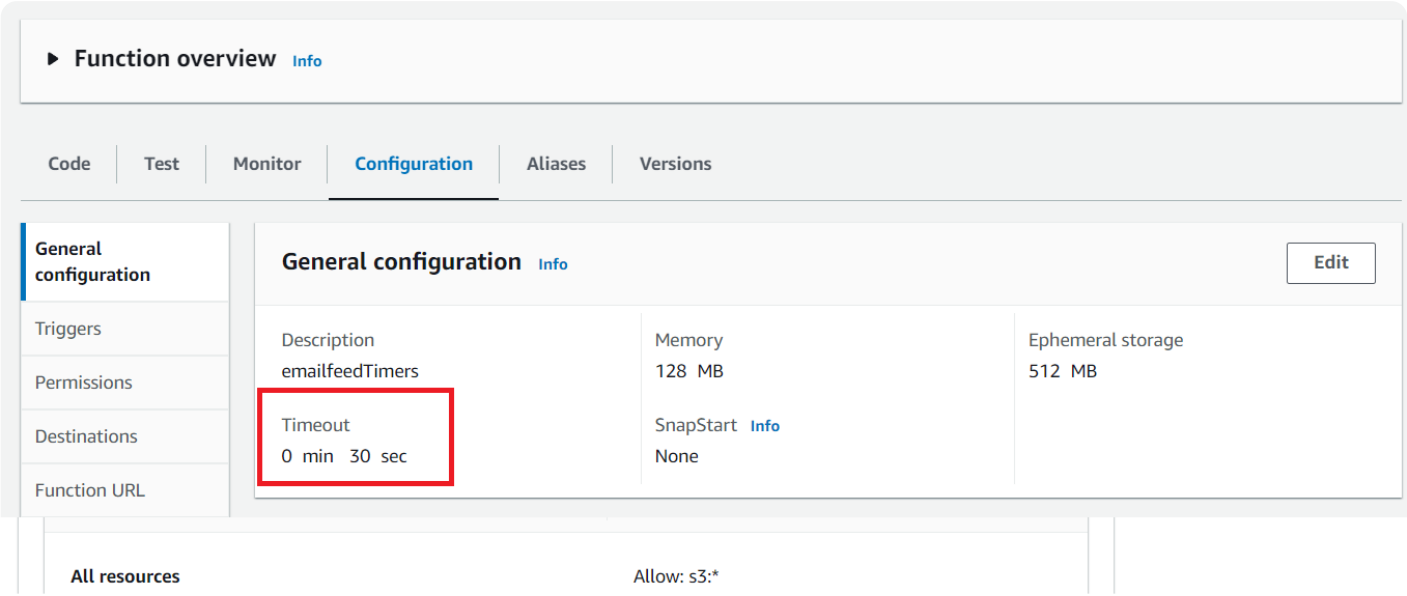
- Navigate to the Lambda service in the AWS management console.
- Click on “Create Function”.
- Provide your function with a descriptive name and select Python as the runtime. Then choose the IAM role you created in step 2 above.
- Click on “Create function”.

Notes:

Setting the execution role:



Setting the time-out value for the Lambda function. In this case, I've set it to 30 seconds.



Step 4: Write the Lambda function to import the Disposable Email Domain list to S3

The example Lambda function uses the python requests module, and you may need to import it as it is no longer part of Boto3. AWS provides vague documentation on how to do this, but various tech articles can be found on the Internet.

Example code:

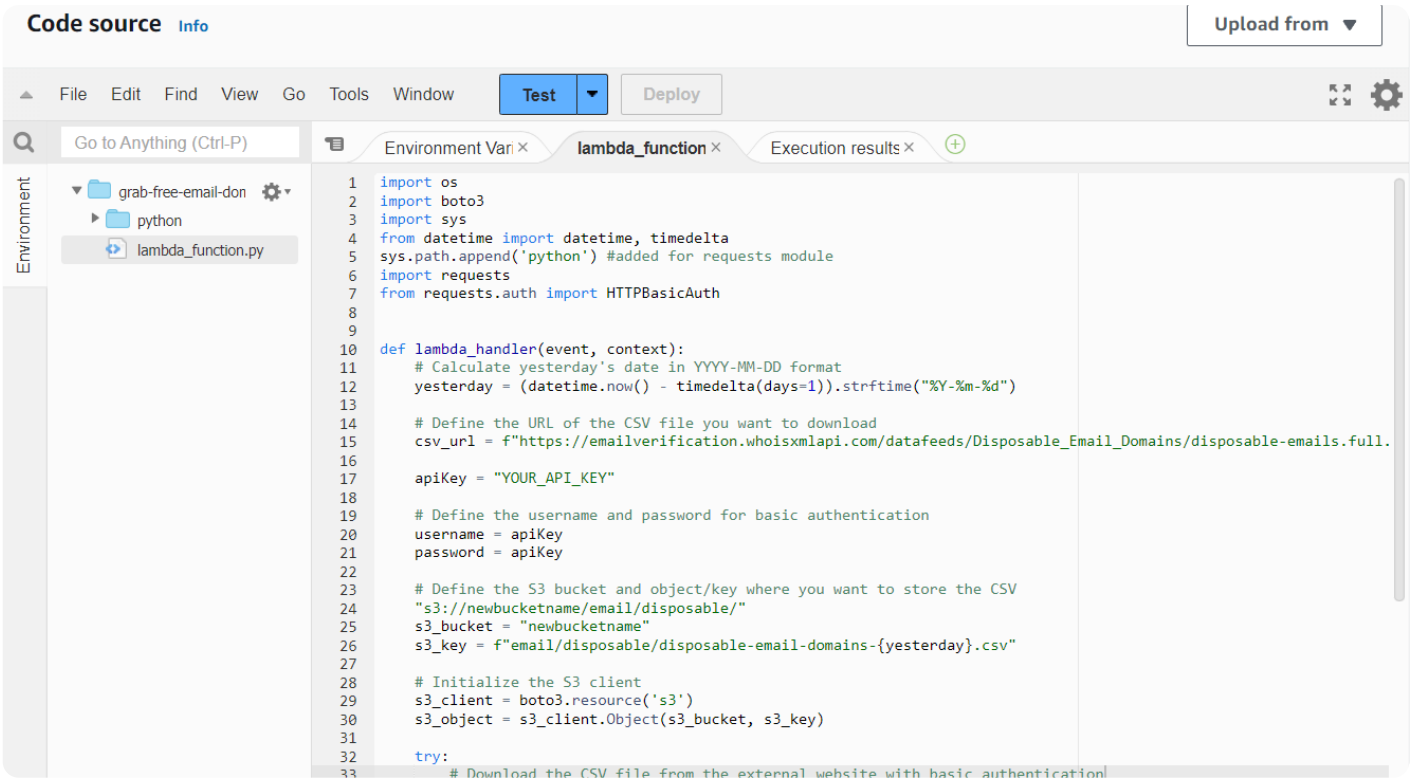
The below python code (also available on GitHub) provides the entry point for the lambda_handler function:

```
import os
import boto3
import sys
```

```
from datetime import datetime, timedelta
sys.path.append('python') #added for requests module
import requests
from requests.auth import HTTPBasicAuth


def lambda_handler(event, context):
    # Calculate yesterday's date in YYYY-MM-DD format
    yesterday = (datetime.now() - timedelta(days=1)).strftime("%Y-%m-%d")

    # Define the URL of the CSV file you want to download
    csv_url = f"https://emailverification.whoisxmlapi.com/datafeeds/Disposable_Email_Domains/disposa

    apiKey = "YOUR_API_KEY"

    # Define the username and password for basic authentication
    username = apiKey
    password = apiKey

    # Define the S3 bucket and object/key where you want to store the CSV
    "s3://newbucketname/email/disposable/"
    s3_bucket = "newbucketname"
    s3_key = f"email/disposable/disposable-email-domains-{yesterday}.csv"

    # Initialize the S3 client
    s3_client = boto3.resource('s3')
    s3_object = s3_client.Object(s3_bucket, s3_key)

    try:
        # Download the CSV file from the external website with basic authentication
        response = requests.get(csv_url, auth=HTTPBasicAuth(username, password))

        if response.status_code == 200:
            # Upload the CSV file to S3
```

```
            print(f"Uploading file to ", s3_bucket, s3_key)
            s3_object.put(Body=response.content)
            return {
                'statusCode': 200,
                'body': 'CSV file successfully downloaded and uploaded to S3'
            }
        else:
            bodyStr = f"Failed to download {csv_url}"
            return {
                'statusCode': response.status_code,
                'body': bodyStr
            }
    except Exception as e:
        return {
            'statusCode': 500,
            'body': str(e)
        }
```

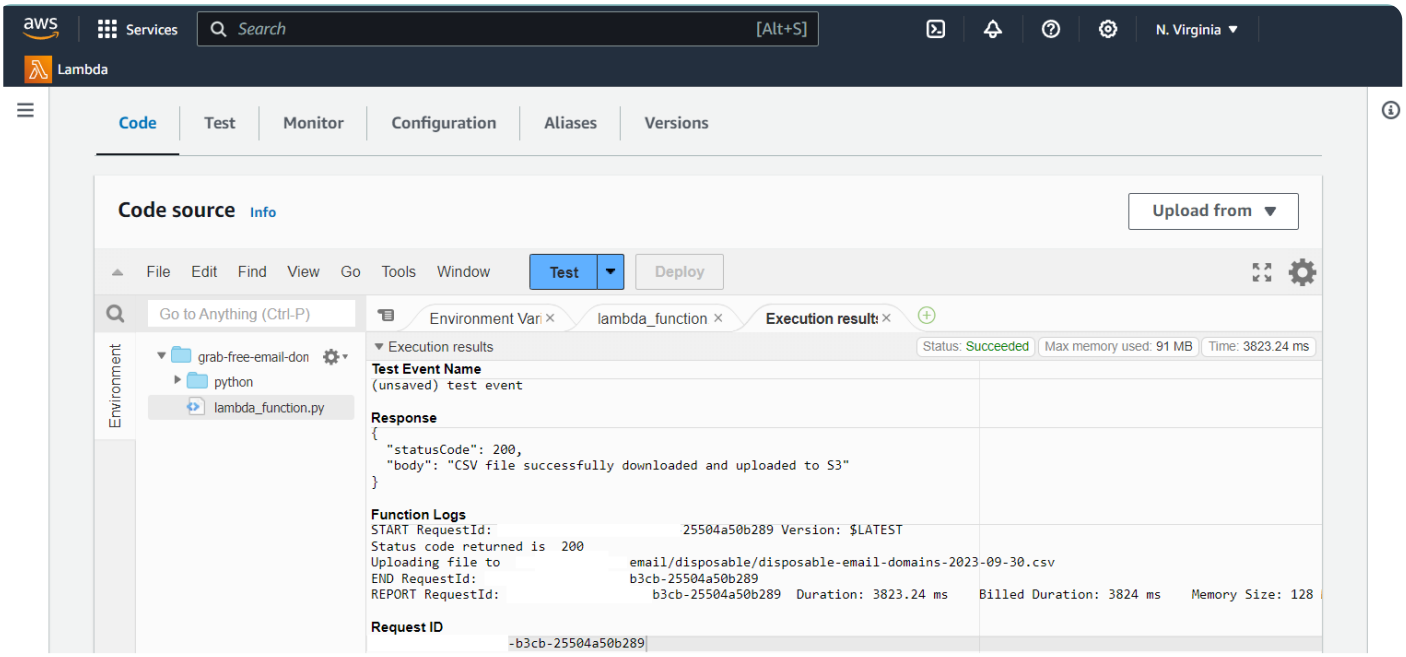
When you’re done, you should have something that resembles this:



Step 5: Testing your new Lambda function

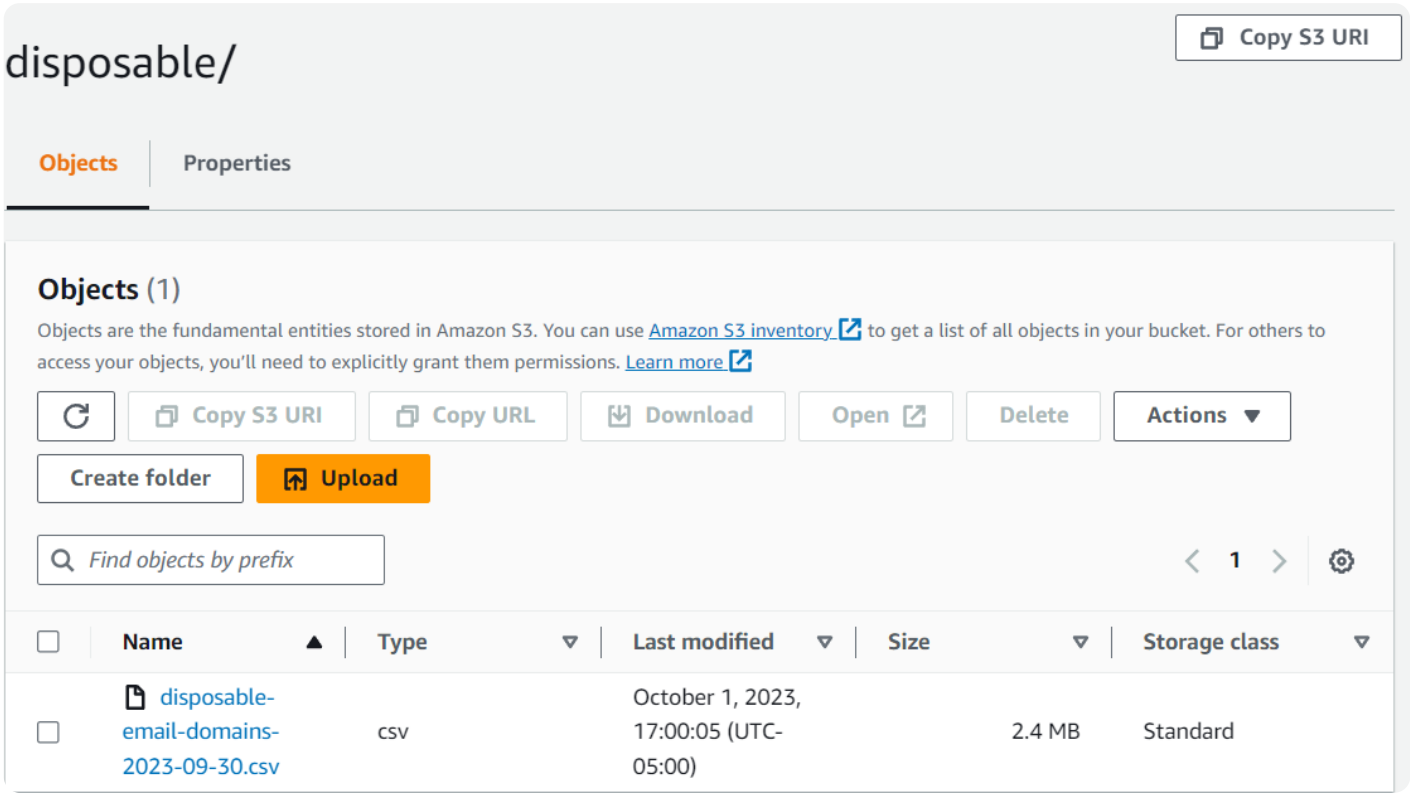
The last step is to test the Lambda function to ensure it can a) successfully retrieve the disposable email domain file, and b) write it to the S3 bucket:

- Click on “Test” at the top of the page, and you should see something similar.



- If you receive the message “requests” module not found, then you need to set up the python requests library correctly, which is outside the scope of this document.

If your Lambda function is set up correctly, the function will retrieve the file, and write it to the S3 bucket. You can navigate to the S3 bucket to verify it’s there.



Conclusion

Configuring AWS Lambda with access to an S3 bucket is a common task for cloud engineers. After walking you through the process, the next step is to determine what you want to do with this data, such as import it into Athena, Postgres or MySQL database. If you're not familiar with AWS Glue for ETL, be sure to check that out as well.