

EBS Volume Backup Automation using Lambda, EventBridge and SNS (simple notification service)

Introduction

This project automates the creation of EBS volume backups (snapshots) using AWS Lambda. The function is triggered automatically by Amazon EventBridge, and once the snapshot is created, a notification is sent to your email using Amazon SNS.

This ensures regular, reliable, hands-free backups without manual work.

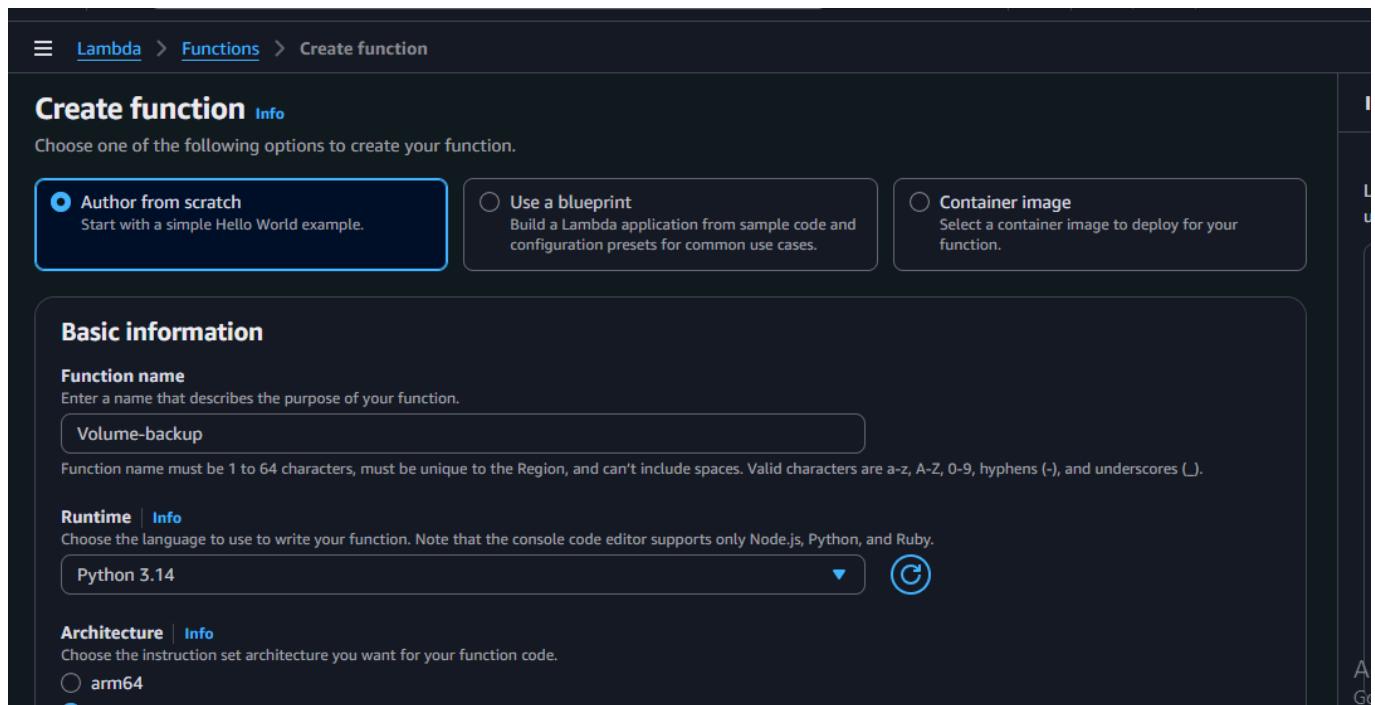
Project Objective

- Automatically create EBS volume backups.
- Schedule backup using EventBridge (cron).
- Send email notifications via SNS.
- Reduce manual effort and prevent human error.

Architecture

Step 1: Create a Lambda Function

- Open AWS Lambda console -> Click create function -> Author from scratch



- Runtime → Python 3.14
- Architecture → x86_64
- Default execution role
- Click Create Function

Architecture | [Info](#)
Choose the instruction set architecture you want for your function code.
 arm64
 x86_64

Permissions | [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 networking, security, and governance for your function. These settings help secure and customize your Lambda function deployment.

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

Here, Successfully created Lambda Function

Function overview | [Info](#)

Throttle | [Copy ARN](#) | Actions ▾

Export to Infrastructure Composer | Download ▾

Description

Last modified
12 seconds ago

Function ARN
[arn:aws:lambda:us-east-1:730356633190:function:Volume-backup](#)

Function URL | [Info](#)

[Code](#) | [Test](#) | [Monitor](#) | [Configuration](#) | [Aliases](#) | [Versions](#)

Step 2: Add Code for Creating EBS Snapshot

```
import boto3

def lambda_handler(event, context):
    regions = []
    # Get a list of all regions
    ec2_client = boto3.client('ec2')
    regions_response = ec2_client.describe_regions()

    for region in regions_response['Regions']:
        regions.append(region['RegionName'])
```

```
snapshots_created = []

# Iterate through each region
for region in regions:
    print(f"Processing region: {region}")
    ec2 = boto3.client('ec2', region_name=region)

    # Get all volumes in 'in-use' state
    volumes = ec2.describe_volumes(
        Filters=[{'Name': 'status', 'Values': ['in-use']}]
    )['Volumes']

    for volume in volumes:
        volume_id = volume['VolumeId']
        print(f"Creating snapshot for Volume: {volume_id}")

        # Create a snapshot
        try:
            snapshot = ec2.create_snapshot(
                VolumeId=volume_id,
                Description=f"Snapshot of {volume_id} from region {region}"
            )

            snapshots_created.append({
                "Region": region,
                "VolumeId": volume_id,
                "SnapshotId": snapshot['SnapshotId']
            })

            print(f"Snapshot created: {snapshot['SnapshotId']}")

        except Exception as e:
            print(f"Error creating snapshot for volume {volume_id} in region {region}: {str(e)}")

    return {
        "statusCode": 200,
        "body": f"Snapshots created: {snapshots_created}" }
```

- After the Lambda function is created, scroll down the code section and remove default code and paste above code.
- Click Deploy

The screenshot shows the VS Code interface with the following details:

- EXPLORER:** Shows a folder named "VOLUME-BACKUP" containing "lambda_function.py". Below it, under "DEPLOY", there are two buttons: "Deploy (Ctrl+Shift+U)" and "Test (Ctrl+Shift+I)".
- CODE EDITOR:** The file "lambda_function.py" contains Python code for creating EC2 volume snapshots. The code includes imports for `lambda_handler` and `ec2`, and logic to describe volumes, loop through them, and create snapshots.
- OUTPUT:** The "Execution Results" tab shows the response from the Lambda function execution. It includes a JSON object with "statusCode": 200 and "body": "Snapshots created: [...]".
- TEST EVENTS:** A section titled "TEST EVENTS [SELECTED: TEST-1]" shows options to "Create new test event" or "Private saved events" (with one entry "Test-1").

Step 3: Create SNS Topic for Email Notification

The screenshot shows the Amazon SNS "Topics" page with the following details:

- Left sidebar:** Shows navigation links for "Amazon SNS", "Dashboard", "Topics", "Subscriptions", and "Mobile" (with "Push notifications" listed).
- Top bar:** Includes "Topics (1)", "Edit", "Delete", "Publish message", and "Create topic" buttons.
- Table:** Displays a single topic named "Global" with Type "Standard" and ARN "arn:aws:sns:us-east-1:730356633190:Global".

- Scroll down -> Create subscription in protocol enter your email address
- Go to your inbox and click confirm subscription

The screenshot shows the Amazon SNS "Subscriptions" page with the following details:

- Left sidebar:** Shows navigation links for "Amazon SNS", "Dashboard", "Topics", "Subscriptions", and "Mobile" (with "Push notifications" and "Text messaging (SMS)" listed).
- Top bar:** Includes "Subscriptions (1)", "Edit", "Delete", "Request confirmation", "Confirm subscription", and "Create subscription" buttons.
- Table:** Displays a single subscription with ID "3e82409f-b448-4d90-...", Endpoint "sakshikarle10@gmail.com", Status "Confirmed", Protocol "EMAIL", and Topic "Global".

Step 4: Attach Permissions to Lambda Role

- Open IAM Console → Roles.
- Search for the role created for your Lambda → example: demo-function-role.
- Click Attach policies.
- Add

AmazonEC2FullAccess

Step 1 Select trusted entity

Step 2 Add permissions

Step 3 Name, review, and create

Add permissions Info

Permissions policies (1/1092) Info

Choose one or more policies to attach to your new role.

Filter by Type

Q ec2fu All types 1 match

Policy name	Type	Description
AmazonEC2FullAccess	AWS managed	Provides full access to Amazon EC2 via t...

▶ Set permissions boundary - *optional*

Cancel Previous Next

- Enter the role name

Step 1 Select trusted entity

Step 2 Add permissions

Step 3 Name, review, and create

Name, review, and create

Role details

Role name
Enter a meaningful name to identify this role.

Maximum 64 characters. Use alphanumeric and '+-_@-' characters.

Description
Add a short explanation for this role.

Maximum 1000 characters. Use letters (A-Z and a-z), numbers (0-9), tabs, new lines, or any of the following characters: _+=,. @-/[\[]/\[\]]#\$%^*();`"

- Attach Role to your Lambda Function Open Lambda Console → Functions

- Select your function: Volume-backup
- Go to Configuration → Permissions
- Click Edit under Execution Role
- Select:
- Use an existing role

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](#). For Python and .NET runtimes, [view pricing](#).

None

Supported runtimes: .NET 8 (C#/F#/PowerShell), Java 11, Java 17, Java 21, Python 3.12, Python 3.13, Python 3.14.

Timeout
1 min 30 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 Logs.

Lambda-EC2FullAccess

View the Lambda-EC2FullAccess role [on the IAM console](#).

Cancel **Save**

Step 5: Verify Snapshot in EC2 Console

- Go to AWS EC2 Console.
- On the left panel → Click Snapshots under Elastic Block Store (EBS).

You will see the automatically created snapshot from your Lambda function.

Snapshots (1) Info

Last updated less than a minute ago

Name	Snapshot ID	Full snapshot size	Volume size	Description	Storage tier
	snap-005a855a8c60f45bf	1.65 GiB	8 GiB	Snapshot of vol-0a171891...	Standard

To create an EBS volume, you must have at least one EC2 instance running, because EBS volumes attach to EC2 instances.

Step 6: Create EventBridge Rule (Trigger Lambda Automatically)

- Click Create Rule.
- Rule name → rule 1

Amazon EventBridge > Rules > Create rule

Step 1 Define rule detail

Rule detail

Name
rule-1
Maximum of 64 characters consisting of numbers, lower/upper case letters, -, _.

Description - optional
Enter description

Event bus | Info
Select the event bus this rule applies to, either the default event bus or a custom or partner event bus.
default

Enable the rule on the selected event bus

Rule type | Info

- Rule with an event pattern
A rule that runs when an event matches the defined event pattern. EventBridge sends the event to the specified target.
- Schedule
A rule that runs on a schedule

Activate Windows

- Choose Schedule → rate

Example (10 minutes):

Amazon EventBridge > Rules > Create rule

Step 1 Define rule detail

Step 2 Define schedule

Schedule pattern

Schedule pattern
Choose the schedule type that best meets your needs.

- A fine-grained schedule that runs at a specific time, such as 8:00 a.m. PST on the first Monday of every month.
- A schedule that runs at a regular rate, such as every 10 minutes.

Rate expression | Info
Enter a value and the unit of time to run the schedule.
 rate(10 Minutes)

Value: 10
Unit, e.g. mins, hours...: Minutes

Cancel Previous Next

- Choose Target → sns topic

Select target(s)

Target 1

Target types
Select an EventBridge event bus, EventBridge API destination (SaaS partner), or another AWS service as a target.

EventBridge event bus
 EventBridge API destination
 AWS service

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

SNS topic

- Click Create Rule. Here, Rule is Created

Select event bus

Event bus
Select or enter event bus name

default

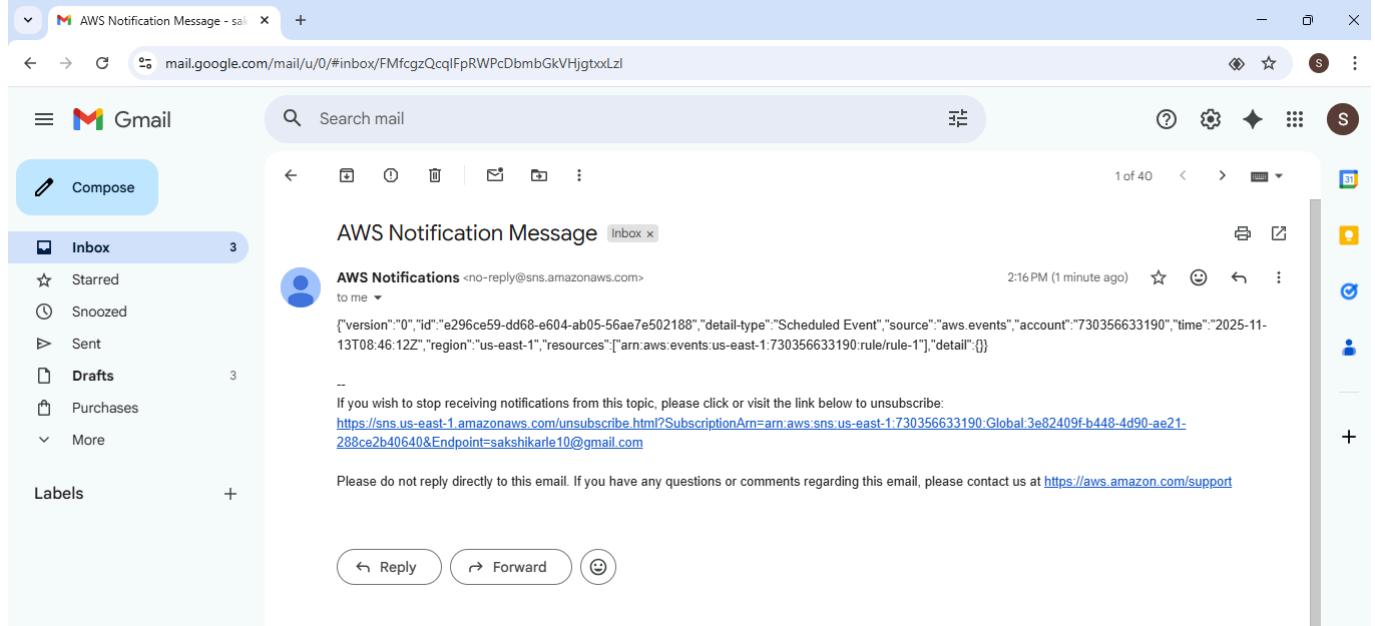
Rules on default event bus (1)

Name	Status	Type	Event bus	ARN	Description
rule-1	Enabled	Scheduled Standard	default	arn:aws:event:sus-east-1:73035:6633190:rule/rule-1	-1

Step 7: Check SNS Email Notification

1. Open your email inbox.
2. You should receive a message from SNS showing: Snapshot ID, Date & Time

Confirmation that backup was created successfully



Conclusion:

In this project, we automated the process of creating EBS snapshots across all AWS regions using AWS Lambda and EventBridge. The Lambda function identifies all in-use EBS volumes and automatically creates snapshots for backup and recovery. EventBridge triggers the Lambda function on a scheduled basis, ensuring regular and consistent backups without manual effort. This automation improves data protection, reduces operational overhead, and ensures reliable disaster recovery for critical EC2 instances.