

# **Automated EBS Snapshots Scheduling**

Seetha Lakshmi K A P

# 1.Introduction

In cloud infrastructure management, data availability and disaster recovery are critical components. This project demonstrates the implementation of an **automated backup strategy** for Amazon EBS (Elastic Block Store) volumes using **AWS Lambda**.

Unlike manual snapshots, this solution leverages a serverless architecture to identify specific resources based on metadata tags and trigger point-in-time backups. By using **IAM Roles**, the system operates securely without the need for manual key pair management or SSH access. This integration highlights how administrative tasks can be automated to ensure consistency, reduce human error, and maintain data durability across an AWS environment.

## 2.Implementation

### Step 1: Launch EC2 Instance (Target Resource)

- Logged in to the AWS Management Console and navigated to **EC2**.
- Launched a new instance using **Amazon Linux 2023 AMI**.
- **Key Pair**: Selected "**Proceed without a key pair**" as the automation interacts via AWS APIs rather than terminal access.
- **Resource Tagging**: Assigned a custom tag to the instance to allow the Lambda function to identify it.
  - **Key**: Backup
  - **Value**: True
- **Instance ID obtained**: i-00850dc6016df70cf

The screenshot displays the AWS Management Console interface for the EC2 service. At the top, the 'Instances (1/1)' section is active, showing a table with one instance: 'Backup-Test-Server' (ID: i-00850dc6016df70cf) in a 'Running' state, using a 't3.micro' instance type. Below this, the detailed configuration for the selected instance is shown. The instance is named 'i-00850dc6016df70cf (Backup-Test-Server)'. Key details include: Instance ID (i-00850dc6016df70cf), Public IPv4 address (43.205.135.89), Private IPv4 addresses (172.31.42.187), Instance state (Running), Public DNS (ec2-43-205-135-89.ap-south-1.compute.amazonaws.com), Private IP DNS name (ip-172-31-42-187.ap-south-1.compute.internal), and Hostname type (IP name: ip-172-31-42-187.ap-south-1.compute.internal).

Name	Instance ID	Instance state	Instance type	Status check
Backup-Test-Server	i-00850dc6016df70cf	Running	t3.micro	3/3 checks passed

i-00850dc6016df70cf (Backup-Test-Server)		
<b>Instance ID</b> i-00850dc6016df70cf	<b>Public IPv4 address</b> 43.205.135.89   <a href="#">open address</a>	<b>Private IPv4 addresses</b> 172.31.42.187
<b>IPv6 address</b> -	<b>Instance state</b> Running	<b>Public DNS</b> ec2-43-205-135-89.ap-south-1.compute.amazonaws.com   <a href="#">open address</a>
<b>Hostname type</b> IP name: ip-172-31-42-187.ap-south-1.compute.internal	<b>Private IP DNS name (IPv4 only)</b> ip-172-31-42-187.ap-south-1.compute.internal	

## Step 2: Create IAM Policy and Role

- Navigated to **IAM** and used the **Visual Editor** to create a least-privilege policy.
- **Service (EC2):** Granted `DescribeInstances`, `DescribeVolumes`, `CreateSnapshot`, and `CreateTags` permissions.
- **Service (CloudWatch Logs):** Granted permission to create log groups and streams for troubleshooting.
- **Role Creation:** Created a service role named `EBS-Snapshot-Automation-Role` and attached the policy, allowing **Lambda** to assume this identity.

### Policy details

#### Policy name

Enter a meaningful name to identify this policy.

EBS-Snapshot-Automation-Policy

Maximum 128 characters. Use alphanumeric and '+=, @-\_' characters.

#### Description - optional

Add a short explanation for this policy.

Policy for Lambda to automate EBS backups based on tags.

Maximum 1,000 characters. Use alphanumeric and '+=, @-\_' characters.

### Permissions defined in this policy [Info](#)

Edit

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it

Search

Allow (2 of 462 services)

Show remaining 460 services

Service	Access level	Resource	Request condition
CloudWatch Logs	Limited: Write	All resources	None
EC2	Limited: List, Write, Tagging	All resources	None

### EBS-Snapshot-Automation-Role [Info](#)

Delete

Allows Lambda functions to call AWS services on your behalf.

#### Summary

Edit

**Creation date**  
February 20, 2026, 10:51 (UTC+05:30)

**Last activity**  
-

**ARN**  
`arn:aws:iam::816039039147:role/EBS-Snapshot-Automation-Role`

**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.



Simulate

Remove

Add permissions

Search

Filter by Type

All types

Policy name	Type	Attached entities
<input checked="" type="checkbox"/> EBS-Snapshot-Autom...	Customer managed	1

## Step 3: Lambda Function Configuration

- Navigated to the **AWS Lambda** console and created a new function named `AutomatedBackup`.
- **Runtime:** Selected **Python 3.12** for its stability and native support for the Boto3 SDK.
- **Execution Role:** Assigned the existing `EBS-Snapshot-Automation-Role`.
- **Logic:** Implemented a Python script that:
  1. Filters instances by the `Backup: True` tag.
  2. Identifies all attached EBS Volume IDs.
  3. Executes the `create_snapshot` command with a dynamic timestamp.

Python code:

```
import boto3
import datetime

# Initialize the EC2 client
ec2 = boto3.client('ec2')

def lambda_handler(event, context):
    # Your specific Instance ID
    target_instance_id = 'i-00850dc6016df70cf'

    print(f"Starting automated backup for instance: {target_instance_id}")

    try:
        # 1. Get details about the instance and its volumes
        instance_data = ec2.describe_instances(InstanceIds=[target_instance_id])

        for reservation in instance_data['Reservations']:
            for instance in reservation['Instances']:
                # 2. Loop through all volumes (EBS) attached to this instance
                for device in instance['BlockDeviceMappings']:
                    volume_id = device['Ebs']['VolumeId']
                    timestamp = datetime.datetime.now().strftime("%Y-%m-%d
%H:%M:%S")

                    # 3. Create the actual Snapshot
                    description = f"Automated Backup of {target_instance_id} -
{timestamp}"

                    snapshot = ec2.create_snapshot(
                        VolumeId=volume_id,
                        Description=description
                    )
```

```


# 4. Add a Name tag to the snapshot for your report
ec2.create_tags(
    Resources=[snapshot['SnapshotId']],
    Tags=[{'Key': 'Name', 'Value': f"Backup-
{target_instance_id}"}]
)


print(f"Success! Created Snapshot ID:
{snapshot['SnapshotId']} for Volume: {volume_id}")

except Exception as e:
    print(f"Error occurred: {str(e)}")
    raise e

```

☰ [Lambda](#) > Functions

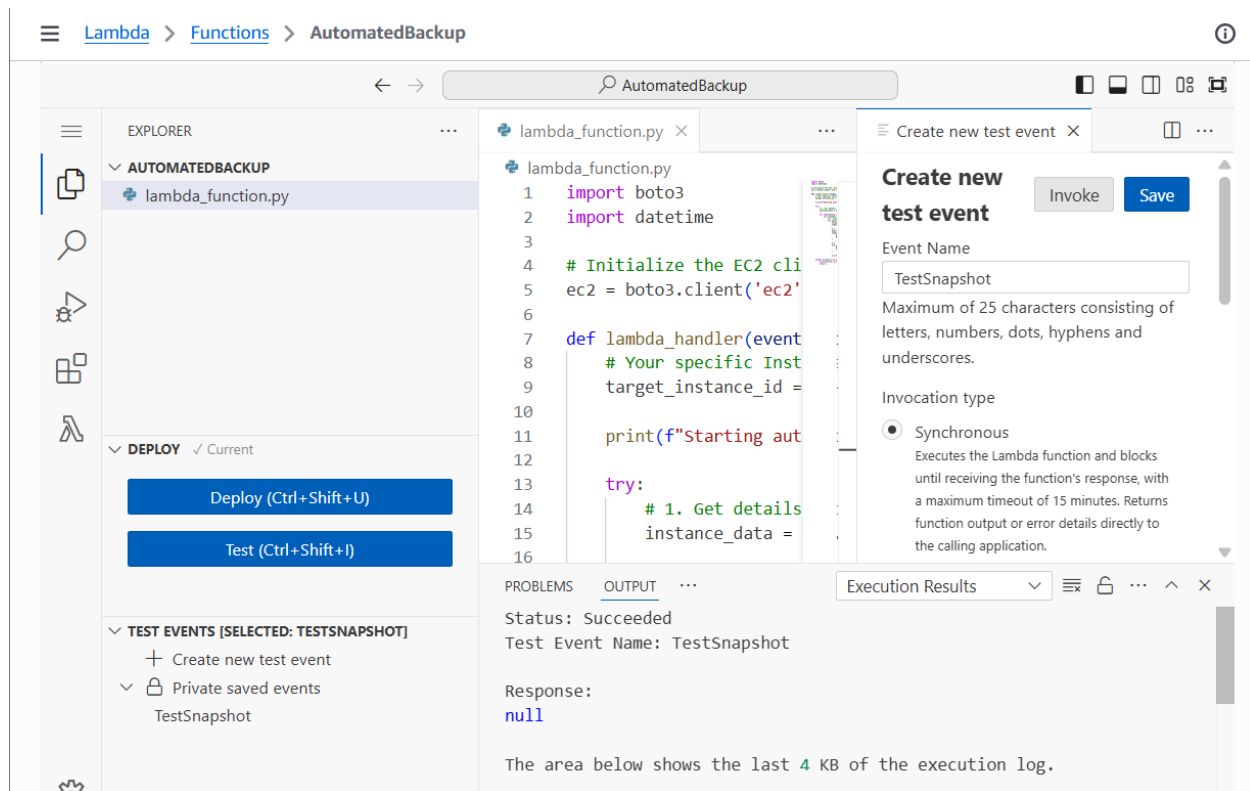
 **Build Multi-Step Workflows with Lambda Durable Functions**  
 Write sequential code in your preferred language while Lambda tracks progress, automatically retries failures, and suspends execution for up to 1 year

**Functions (1)** Last fetched 2/20/2026, 11:41:21 AM 

<input type="checkbox"/>	Function name	Description	Package type	Runtime	Type
<input type="checkbox"/>	<a href="#">AutomatedBackup</a>	-	Zip	Python 3.12	Standard

## Step 4: Deployment and Execution

- Deployed the Python code to the AWS Lambda environment.
- **Manual Trigger:** Configured a test event and invoked the function manually to verify the initial setup.
- **Automated Trigger Logic:** Discussed the use of **Amazon EventBridge** to schedule this function (e.g., daily at 10:00 AM) to fulfill the automated scheduling requirement.



## Step 5: Multi-Tier Verification (Logs & EC2 Console)

To confirm the successful execution of the automation, verification was performed at both the process level (Logs) and the resource level (Console).

- Log Verification (CloudWatch):** Navigated to **CloudWatch Logs** under the `/aws/lambda/AutomatedBackup` log group. The logs confirmed that the function successfully identified the target instance `i-00850dc6016df70cf` and executed the `create_snapshot` command.
  - Result:** Success! Success! Created Snapshot ID: `snap-02307085235add075` for Volume: `vol-0ade5485e08a9ebbb`
- Resource Verification (EC2 Snapshots):** Navigated to the **EC2 Console** under the **Snapshots** section. Verified that a new snapshot exists.
- Metadata Confirmation:** Confirmed that the snapshot contains the custom description: `"Automated Backup of i-00850dc6016df70cf"`, proving it was generated by the specific Lambda function.

Snapshots (1/2) Info

Last updated  
1 minute ago

Recycle Bin

Actions

Create snapshot

Snapshot scope  
Owned by me

Search

< 1 > ⚙

	Name	Snapshot ID	Full snapshot size	Volume size	Description
<input checked="" type="checkbox"/>	Backup-i-00850dc6016df70cf	snap-02307085235add075	1.63 GiB	8 GiB	Automated Backup
<input type="checkbox"/>		snap-0d725339c92aa6c0e	1.68 GiB	8 GiB	Created by Createlr

Snapshot ID: snap-02307085235add075 (Backup-i-00850dc6016df70cf)

⚙ ✓

Owner

816039039147

Started

Fri Feb 20 2026 11:17:26  
GMT+0530 (India Standard  
Time)

Product codes

-

Fast snapshot restore

-

Description

Automated Backup of i-  
00850dc6016df70cf - 2026-02-  
20 05:47:26

Source volume

Volume ID

vol-0ade5485e08a9ebbb

Volume size

8 GiB

CloudWatch > Log management > /aws/lambda/AutomatedBackup > 2026/02/20/[LATEST]4946f7e5b2ba466cb4de8d504c0d80bd

Log events



Actions

Start tailing

Create metric filter

You can use the filter bar below to search for and match terms, phrases, or values in your log events. [Learn more about filter patterns](#)

Filter events - press enter to search

Clear

1m

30m

1h

12h

Custom

UTC timezone

Display



Timestamp	Message
	No older events at this moment. <a href="#">Retry</a>
2026-02-20T05:47:25.157Z	INIT_START Runtime Version: python:3.12.mainlinev2.v3 Runtime Version ARN: arn:aws:lambda:ap-south-1::runtime:27990c3d0965bea97240f3380b8d623b692705303bbde4c31fb02fa7daa054...
2026-02-20T05:47:25.713Z	START RequestId: 10af8958-1669-4a3a-8d6c-dceacf5b9ebb Version: \$LATEST
2026-02-20T05:47:25.713Z	Starting automated backup for instance: i-00850dc6016df70cf
2026-02-20T05:47:26.548Z	Success! Created Snapshot ID: snap-02307085235add075 for Volume: vol-0ade5485e08a9ebbb
2026-02-20T05:47:26.551Z	END RequestId: 10af8958-1669-4a3a-8d6c-dceacf5b9ebb
2026-02-20T05:47:26.551Z	REPORT RequestId: 10af8958-1669-4a3a-8d6c-dceacf5b9ebb Duration: 836.97 ms Billed Duration: 1390 ms Memory Size: 128 MB Max Memory Used: 99 MB Init Duration: 552.30 ms
	No newer events at this moment. <a href="#">Auto retry paused. Resume</a>

### 3. Conclusion

This lab successfully demonstrated a serverless approach to infrastructure automation. By integrating **AWS Lambda** with **Amazon EBS**, we moved from manual backup operations to an automated, tag-based system. The use of **IAM Roles** instead of key pairs ensured a higher security posture by adhering to the principle of least privilege and eliminating the need for long-term credentials.

The successful verification of snapshots in the EC2 console and the corresponding logs in CloudWatch proved the reliability of the script. This project provides foundational knowledge in **event-driven administration** and **cloud-native security**, essential for managing large-scale, resilient environments where manual intervention must be minimized.