Automated EBS Backup & Recovery System

Project Overview

Services used: EC2, EBS, Lambda, CloudWatch Events (EventBridge), IAM, CloudWatch Logs, (optional: SNS)

Goal: Automate EBS snapshot creation on a schedule, tag snapshots for identification, and implement a retention policy that deletes snapshots older than a configured retention window. Ensure reliable disaster recovery for compute workloads with minimal operational effort.

Skills demonstrated: Cloud automation, backup strategies, AWS Lambda scripting (Python / boto3), event-driven architecture, IAM least-privilege, monitoring.

Architecture (high level)

1. **EventBridge (CloudWatch Events)** triggers an AWS Lambda function on a schedule (e.g., daily at 02:00 UTC).
2. The **Lambda function** discovers EBS volumes to back up (via tags or by scanning instances), creates snapshots, and tags the snapshots with metadata (Name, BackupType, CreatedBy, RetentionDays, OriginVolume, InstanceID).
3. The Lambda function also **applies retention rules**: it lists snapshots that match the tag and deletes those older than the configured retention period.
4. **CloudWatch Logs** capture Lambda logs. Optionally, **SNS** notifies administrators of success/failure.

Design decisions & assumptions

- Volumes to be backed up are identified by a tag (recommended: Backup=true or Backup: daily) — this keeps scope explicit and safe.
- Snapshots are tagged with the creation timestamp and retention window to make deletion deterministic.
- Retention is enforced by a single Lambda (create + cleanup). Alternatively, separate Lambdas could be used for separation of concerns.
- For consistency of application-level data, you may wish to coordinate with the instance OS (freeze writes, flush fs) — for Linux, use aws ec2 create-image or application-specific hooks. This project focuses on EBS-level snapshots.

Prerequisites

1. AWS account with permission to create Lambda, EventBridge Rule, IAM Role, and to create/delete EBS snapshots.
2. Python 3.11 runtime for Lambda (or Node.js if you prefer — sample uses Python).
3. AWS CLI / Console access to deploy and test.

Security & IAM

IAM Role for Lambda should follow least privilege. Example permissions (scope down with Resource ARNs):

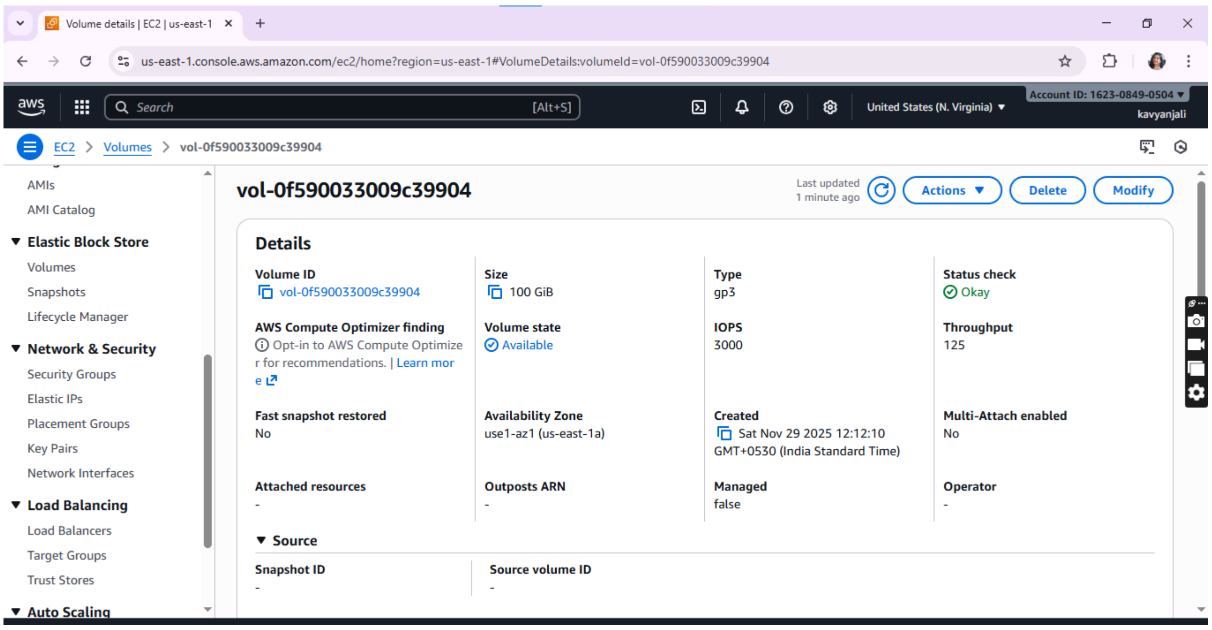
- ec2:DescribeVolumes
- ec2:DescribeInstances
- ec2:CreateSnapshot

- ec2:CreateTags
- ec2:DescribeSnapshots
- ec2:DeleteSnapshot
- logs:CreateLogGroup, logs:CreateLogStream, logs:PutLogEvents
- sns:Publish

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "EC2SnapshotAndDescribePermissions",
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeVolumes",
        "ec2:DescribeInstances",
        "ec2:CreateSnapshot",
        "ec2:CreateTags",
        "ec2:DescribeSnapshots",
        "ec2:DeleteSnapshot"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CloudWatchLogsWrite",
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": "*"
    }
  ]
}
```

STEP 1 — Tag the EBS Volumes for Backup

1. Open the AWS Console → Go to **EC2 Dashboard**.
2. In the left menu, click **Elastic Block Store → Volumes**.
3. Select the volume(s) you want to back up automatically.
4. Click **Tags → Add Tag**.
5. Enter:
   - **Key:** Backup
   - **Value:** daily
6. Click **Save.**



Volume details | EC2 | us-east-1

us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#VolumeDetails:volumeId=vol-0f590033009c39904

United States (N. Virginia) Account ID: 1623-0849-0504 kavyanjali

EC2 > Volumes > vol-0f590033009c39904

AMIs, AMI Catalog, Elastic Block Store: Volumes, Snapshots, Lifecycle Manager; Network & Security: Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces; Load Balancing: Load Balancers, Target Groups, Trust Stores; Auto Scaling

vol-0f590033009c39904

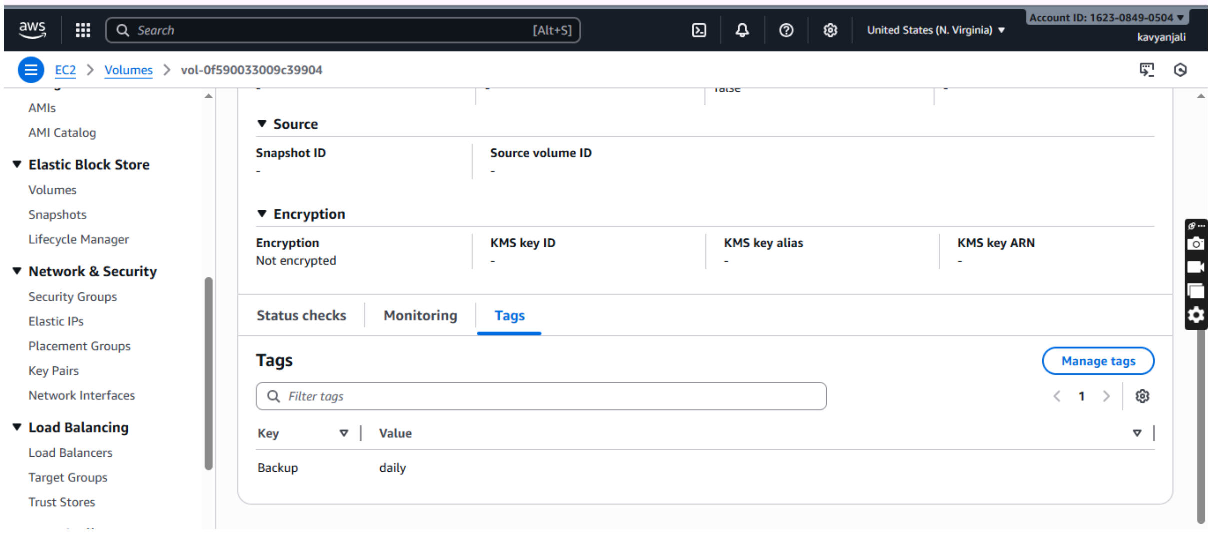
Last updated 1 minute ago | Actions | Delete | Modify

Details

Volume ID vol-0f590033009c39904	Size 100 GiB	Type gp3	Status check Okay
AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations. | Learn more	Volume state Available	IOPS 3000	Throughput 125
Fast snapshot restored No	Availability Zone use1-az1 (us-east-1a)	Created Sat Nov 29 2025 12:12:10 GMT+0530 (India Standard Time)	Multi-Attach enabled No
Attached resources -	Outposts ARN -	Managed false	Operator -

Source

Snapshot ID -	Source volume ID -



EC2 > Volumes > vol-0f590033009c39904

Source

Snapshot ID -	Source volume ID -

Encryption

Encryption Not encrypted	KMS key ID -	KMS key alias -	KMS key ARN -

Status checks | Monitoring | Tags

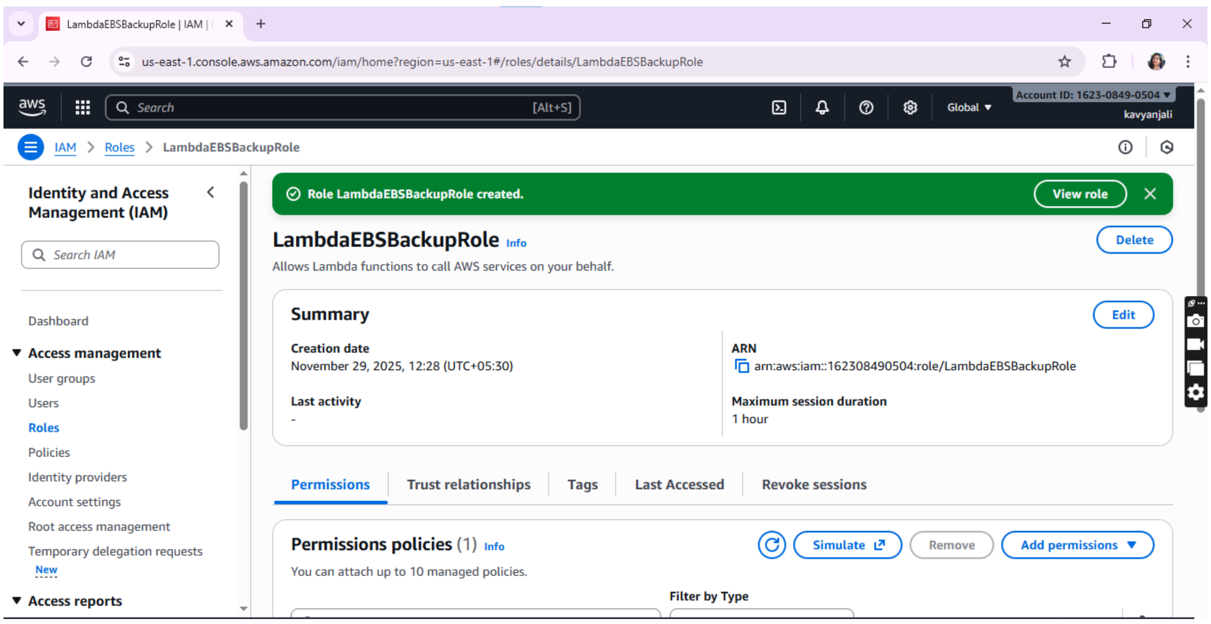
Tags — Manage tags

Filter tags

Key	Value
Backup	daily

STEP 2 — Create the IAM Role for the Lambda Function

1. Go to **IAM Console**.
2. Click **Roles** → **Create role**.
3. Select **AWS Service** → Choose **Lambda** → Click **Next**.
4. Click **Create Policy** (opens in new tab):
   - Choose **JSON** tab.
   - Paste the minimum required permissions (EC2 Describe/Create/Delete, Logs permissions).
   - Click **Next** → Name it: EBSBackupPolicy → Create.
5. Return to role creation window:
   - Search and select the **EBSBackupPolicy**.
   - Add **AWSLambdaBasicExecutionRole**.
6. Name the role: LambdaEBSBackupRole.
7. Click **Create Role**.



Role LambdaEBSBackupRole created.

LambdaEBSBackupRole Info

Allows Lambda functions to call AWS services on your behalf.

Summary

Creation date: November 29, 2025, 12:28 (UTC+05:30)

ARN: arn:aws:iam::162308490504:role/LambdaEBSBackupRole

Last activity: -

Maximum session duration: 1 hour

Permissions policies (1) Info

You can attach up to 10 managed policies.

STEP 3 — Create the Lambda Function

1. Open **AWS Lambda Console**.
2. Click **Create function**.
3. Choose:
   - **Author from scratch**
   - Function name: EBS-Auto-Backup
   - Runtime: **Python** (choose the latest available)
   - Execution role: **Use existing role** → select LambdaEBSBackupRole
4. Click **Create Function**.

Upload Code

1. **Code Source** section.

```
import os
import boto3
import logging
from datetime import datetime, timezone, timedelta
```

```
logger = logging.getLogger()
logger.setLevel(logging.INFO)

ec2 = boto3.client('ec2')
sns = boto3.client('sns') if os.getenv('SNS_TOPIC_ARN') else None

BACKUP_TAG_KEY = os.getenv('BACKUP_TAG_KEY', 'Backup')
BACKUP_TAG_VALUE = os.getenv('BACKUP_TAG_VALUE', 'daily')
RETENTION_DAYS = int(os.getenv('RETENTION_DAYS', '7'))
CREATED_BY_TAG = 'ebs-automated-backup'
SNAPSHOT_DESC_PREFIX = os.getenv('SNAPSHOT_DESCRIPTION_PREFIX',
'AutoBackup')


def find_volumes_to_backup():
# Find volumes tagged with BACKUP_TAG_KEY=BACKUP_TAG_VALUE
filters = [
{'Name': f"tag:{BACKUP_TAG_KEY}", 'Values': [BACKUP_TAG_VALUE]}
]
resp = ec2.describe_volumes(Filters=filters)
volumes = resp.get('Volumes', [])
logger.info(f"Found {len(volumes)} volumes to back up")
return volumes


def create_snapshot(volume):
vol_id = volume['VolumeId']
timestamp = datetime.now(timezone.utc).isoformat()
desc = f"{SNAPSHOT_DESC_PREFIX} {vol_id} {timestamp}"
resp = ec2.create_snapshot(VolumeId=vol_id, Description=desc)
snap_id = resp['SnapshotId']
tags = [
{'Key': 'CreatedBy', 'Value': CREATED_BY_TAG},
{'Key': 'RetentionDays', 'Value'
```

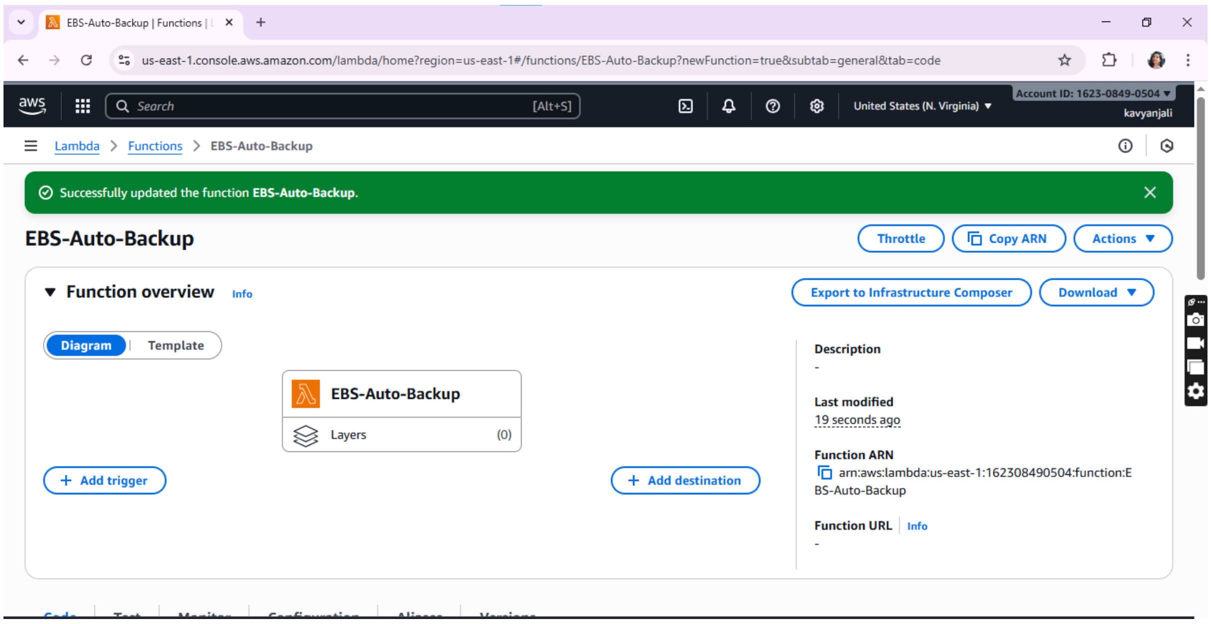
2. Replace the default code with the provided Lambda script.
3. Click **Deploy**.

Add Environment Variables

1. In the Lambda page → Click **Configuration**.
2. Choose **Environment variables → Edit → Add environment variable**.
   - BACKUP_TAG_KEY → Backup
   - BACKUP_TAG_VALUE → daily
   - RETENTION_DAYS → 7
   - (Optional) SNS_TOPIC_ARN
3. Click **Save**.

Set Timeout

1. Go to **Configuration** → **General configuration** → **Edit**.
2. Change Timeout to **1–3 minutes**.
3. Click **Save**.



us-east-1.console.aws.amazon.com/lambda/home?region=us-east-1#/functions/EBS-Auto-Backup?newFunction=true&subtab=general&tab=code

Lambda > Functions > EBS-Auto-Backup

Successfully updated the function **EBS-Auto-Backup**.

EBS-Auto-Backup

Function overview

Description
-

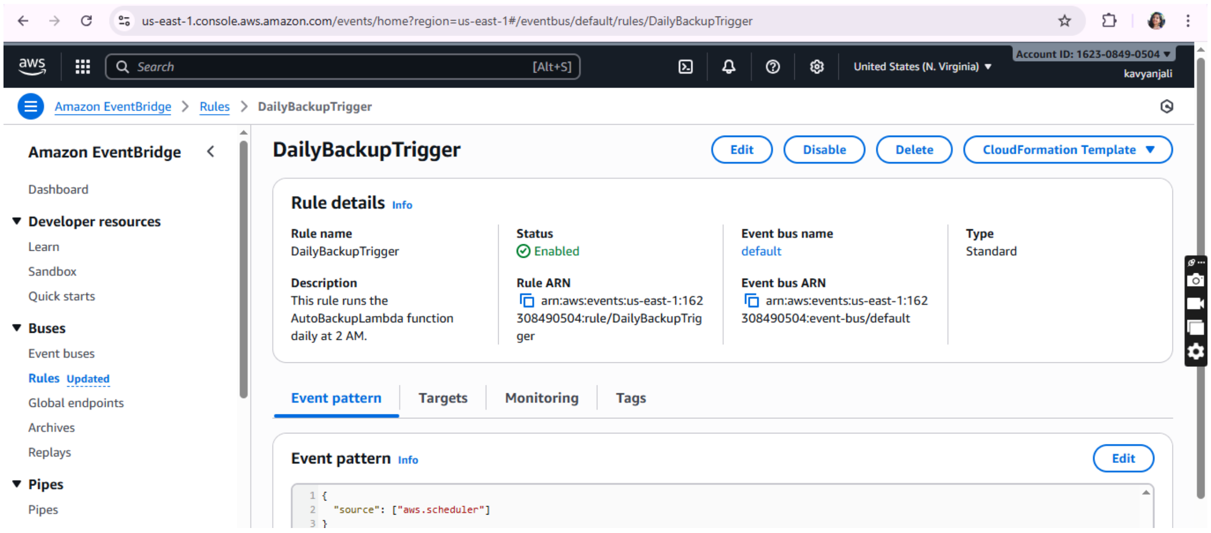
Last modified
19 seconds ago

Function ARN
arn:aws:lambda:us-east-1:162308490504:function:EBS-Auto-Backup

Function URL
-

STEP 4 — Create a Scheduled Event (EventBridge Rule)

1. Go to **Amazon EventBridge Console**.
2. Click **Rules** → **Create rule**.
3. Rule name: DailyEBSBackupSchedule.
4. Choose **Schedule**.
5. Enter CRON expression for daily 2 AM:



us-east-1.console.aws.amazon.com/events/home?region=us-east-1#/eventbus/default/rules/DailyBackupTrigger

Amazon EventBridge > Rules > DailyBackupTrigger

DailyBackupTrigger

Rule details

Rule name	Status	Event bus name	Type
DailyBackupTrigger	Enabled	default	Standard

Description	Rule ARN	Event bus ARN
This rule runs the AutoBackupLambda function daily at 2 AM.	arn:aws:events:us-east-1:162308490504:rule/DailyBackupTrigger	arn:aws:events:us-east-1:162308490504:event-bus/default

Event pattern

```
{
  "source": ["aws.scheduler"]
}
```

Test the Lambda function manually

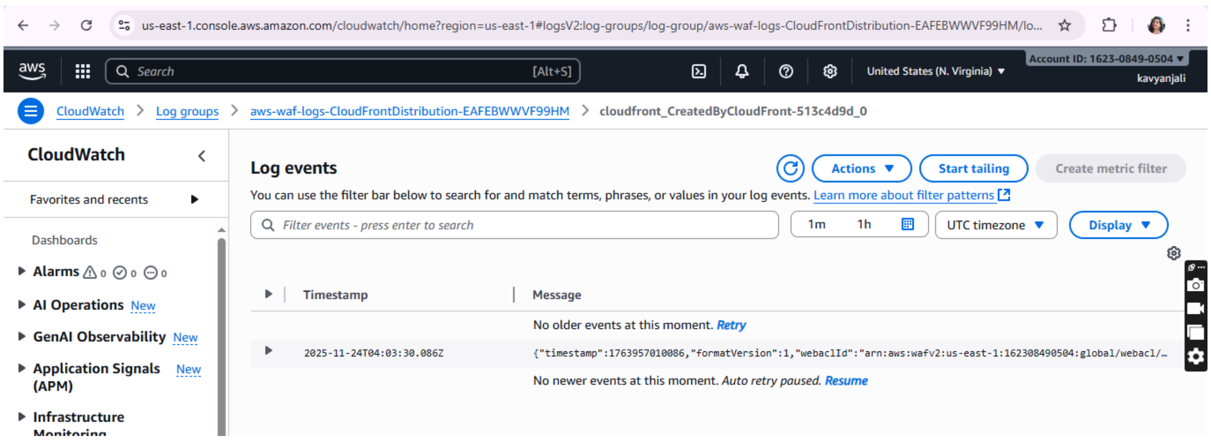
1. Lambda console → select EBS-Auto-Backup → **Test**.
2. **Configure test event** → name it e.g., manual-test → leave default template (or empty JSON {}) → **Create**.

3. Click **Test**.
4. Immediately check **Execution result** (top-right) and **Logs** in the console.
5. Console → **Services** → **EC2** → **Snapshots** → confirm new snapshot(s) appear with tags:
   - CreatedBy: ebs-automated-backup
   - RetentionDays and BackupTimestamp

Monitoring & Alerts

CloudWatch Logs

1. Console → **CloudWatch** → **Log groups** → select /aws/lambda/EBS-Auto-Backup.
2. Inspect recent log streams for successful runs and errors.



us-east-1.console.aws.amazon.com/cloudwatch/home?region=us-east-1#logsV2:log-groups/log-group/aws-waf-logs-CloudFrontDistribution-EAFEBWWVF99HM/lo...
Search [Alt+S]
United States (N. Virginia)
Account ID: 1623-0849-0504
kavyanjali
CloudWatch > Log groups > aws-waf-logs-CloudFrontDistribution-EAFEBWWVF99HM > cloudfront_CreatedByCloudFront-513c4d9d_0
CloudWatch
Favorites and recents
Dashboards
Alarms 0 0 0
AI Operations New
GenAI Observability New
Application Signals (APM) New
Infrastructure Monitoring
Actions
Start tailing
Create metric filter
1m
1h
UTC timezone
Display

Log events

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

Timestamp	Message
	No older events at this moment. *Retry*
2025-11-24T04:03:30.086Z	{"timestamp":1763957010086,"formatVersion":1,"webaclId":"arn:aws:wafv2:us-east-1:162308490504:global/webacl/…
	No newer events at this moment. *Auto retry paused.* *Resume*

Check snapshots:

Go to EC2 → Snapshots.

Verify that new snapshots are created daily.

Check tags: CreatedBy, DeleteOn.

Verify old snapshots disappear after retention period.

Restore from Snapshot (Recovery)

If your volume fails or you need to roll back:

1. Go to EC2 → Snapshots.

2. Choose your snapshot → Actions → Create Volume from Snapshot.

3. Choose the same Availability Zone as your instance.

4. Once created, Attach Volume to your EC2 instance.

5. Mount it inside the instance to access data.

Final Summary – Automated EBS Backup & Retention System (AWS)

This project implements a fully automated backup and retention mechanism for Amazon EBS volumes using **Lambda**, **CloudWatch Events**, **EC2**, and **EBS Snapshots**. It removes the need for manual backups and ensures reliable disaster recovery for EC2 workloads.