Lambda-Based Automated Cloud Resource Reporter

Suhas Pashte

1. INTRODUCTION

"My project is a fully serverless AWS automation system that automatically generates a daily cloud report. The idea came from a real DevOps need—companies must check their AWS environment every day, like which EC2 instances are running, how many S3 buckets they have, or which IAM users have MFA enabled. Doing all this manually from the AWS console every morning takes time and can lead to mistakes.

To solve this, I created an automation workflow using AWS Lambda, Boto3, CloudWatch Scheduler, IAM Roles, and Mailjet SMTP. The Lambda function runs automatically every day through a CloudWatch event, collects details of EC2, S3, and IAM services using Boto3 APIs, prepares a well-formatted report, and sends it directly to the admin through email. Since this is completely serverless, there is no need to maintain any EC2 instance or set up cron jobs on a machine. AWS handles the scheduling, scaling, and execution on its own. I also used least-privilege IAM roles and secure environment variables for SMTP credentials to ensure strong security. Overall, this project shows practical skills in automation, cloud monitoring, serverless architecture, and DevOps best practices."

2. PROBLEM STATEMENT

Most companies need visibility into their cloud environments. Typical questions include:

- How many EC2 instances are running?
- Which instances are stopped or terminated?
- How many S3 buckets exist and what are their details?
- Which IAM users have MFA enabled or disabled?
- Are there any misconfigurations in user security posture?

Doing this checking manually every day from AWS Console:

- Wastes time
- Increases operational workload
- Can miss critical security issues
- Is not suitable for large-scale environments

Hence a fully automated, cloud-native reporting system is required.

3. OBJECTIVES

The main objectives were:

1. Create a serverless automation system

Solution must run without requiring any VM, laptop, or local scheduler.

2. Fetch AWS Resource Data Automatically

Data required daily:

- EC2 instance ID, State, Launch Time, Type
- S3 bucket names and creation dates
- IAM user MFA enable/disable status

3. Email Reports Automatically

- Send report to administrator daily
- Use SMTP automation

4. Build a scalable and secure system

- Least-privilege IAM role
- CloudWatch-based triggering
- Error-free scheduling

5. Zero maintenance

No OS updates
No servers
No downtime

4. SYSTEM ARCHITECTURE

4.1 CloudWatch Scheduler

- A daily cron job is configured.
- It automatically triggers the Lambda function at a specific time every day.

4.2 AWS Lambda Function

- This is the main component of the system.
- It runs Python code along with the Boto3 SDK.

- Lambda is serverless, so no server or machine is required to run or maintain it.

4.3 Boto3 AWS SDK

- Lambda uses Boto3 to interact with AWS services.
- Fetches EC2 data: instance ID, type, state
- Fetches S3 data: bucket list
- Fetches IAM data: users and their MFA status

4.4 Report Generation

- Lambda converts all the fetched data into a clean, readable format.
- A proper text-based report is generated.

• Mailjet SMTP Integration

- Lambda sends the generated report through Mailjet SMTP.
- The report is delivered automatically to the admin's email.

• IAM Role (Security)

- Lambda has only read-only permissions.
- This ensures Lambda cannot modify or delete any AWS resource.

4.5 CloudWatch Logs

- Every execution is stored in CloudWatch Logs.
- Logs contain details like errors, success messages, and execution time.

5. Working

1. Every day CloudWatch runs a scheduled cron job at a specific time.
2. This cron job automatically triggers the AWS Lambda function.
3. When Lambda starts, it uses Boto3 to connect with AWS services.
4. Lambda fetches EC2 details, like instance IDs, types, and states.
5. It also fetches S3 bucket names and IAM users with their MFA status.
6. After collecting all this information, Lambda creates a clean, formatted report.

7. Lambda then connects to Mailjet SMTP and sends the report to the admin’s email.
8. Finally, CloudWatch Logs store all the results, including success or errors.
9. The whole process finishes in a few seconds, and no manual work is needed.

6. TOOLS & TECHNOLOGIES USED

Technology	Purpose
AWS Lambda	Core compute engine (serverless)
Python + Boto3	AWS resource fetching
CloudWatch Scheduler	Daily automated triggering
Mailjet SMTP	Email delivery
IAM Role	Secure AWS access
CloudWatch Logs	Logging & debugging
Environment Variables	Secure credential storage

7. IAM ROLE DESIGN (Least Privilege)

Lambda requires only read-only permissions:

Required Policies:

- AmazonEC2ReadOnlyAccess
- AmazonS3ReadOnlyAccess
- IAMReadOnlyAcces

8. CHALLENGES FACED

1. SMTP email errors
2. Handling AWS rate limits
3. Formatting multi-line report
4. IAM permission issues
5. CloudWatch cron syntax

9. FUTURE SCOPE

- Convert reports to PDF
- Store reports in S3
- Generate historical dashboards
- Integrate with AWS SES
- Multi-account AWS monitoring
- Slack/Teams notifications
- Add Billing & Cost Explorer data
- Generate alerts for non-MFA users

10. CONCLUSION

This project successfully automates daily monitoring of AWS cloud infrastructure using a complete serverless workflow built on top of AWS Lambda, CloudWatch Scheduler, Boto3 API, IAM security, and SMTP automation.

It delivers:

- Zero-maintenance
- Highly scalable
- Cost-efficient
- Secure
- Real-world DevOps automation experience