

## [AWS AI S3 Security Scanner](#)

### Project Overview

I built an AI-powered S3 Security Scanner. The system uses Python and the AWS SDK ([boto3](#)) to identify unencrypted buckets, leverages Gemini AI to explain risks in plain English, and uses Amazon EventBridge to automate the audit every 12 hours.

### Phase 1: Brain & Logic (Setup)

- **Generated Gemini API Key:** I started by visiting Google AI Studio to generate a secure API key. This serves as the "analytical brain" that allows my scanner to interpret technical data.
- **Developed Scanner Logic:** Using the Cursor IDE, I implemented a Python script ([s3\\_scanner.py](#)) that uses [boto3](#) to connect to my AWS account. I programmed it to loop through all S3 buckets and check for active server-side encryption.

```
✚ s3_scanner.py 2 •
✚ s3_scanner.py > ...
1  import boto3
2  import json
3  import os
4  import google.generativeai as genai
5
6  def lambda_handler(event, context):
7      """Scan S3 buckets for encryption and use AI to explain risks"""
8
9      # Initialize S3 client
10     s3_client = boto3.client('s3')
11
12     print("Scanning S3 buckets for encryption...")
13
14     # Get all S3 buckets
15     response = s3_client.list_buckets()
16     buckets = response['Buckets']
17
18     print(f"Found {len(buckets)} buckets to scan\n")
19
20     # Store results
21     scan_results = []
22
23     for bucket in buckets:
24         bucket_name = bucket['Name']
```

- **Integrated Gemini AI:** I drafted a specific prompt within the code that sends raw S3 configuration data to Gemini. I instructed the AI to return high-level, plain-English security insights rather than just technical error codes.

```

26     # Check if bucket has encryption enabled
27     try:
28         encryption = s3_client.get_bucket_encryption(Bucket=bucket_name)
29         encrypted = True
30         encryption_type = encryption['ServerSideEncryptionConfiguration']['Rules'][0]['ApplyServerSideEncryptionByDefault']['SSEAlgorithm']
31     except s3_client.exceptions.ServerSideEncryptionConfigurationNotFoundError:
32         encrypted = False
33         encryption_type = 'None'
34
35     # Store result
36     scan_results.append({
37         'bucket_name': bucket_name,
38         'encrypted': encrypted,
39         'encryption_type': encryption_type
40     })
41
42     status = "Encrypted" if encrypted else "Not Encrypted"
43     print(f"{status}: {bucket_name} ({encryption_type})")
44
45     # Count unencrypted buckets
46     unencrypted_count = len([r for r in scan_results if not r['encrypted']])
47     unencrypted_buckets = [r['bucket_name'] for r in scan_results if not r['encrypted']]
48
49     # Use AI to analyze security findings
50     print("\nAnalyzing security findings with Gemini AI...")
51
52     # Configure Gemini
53     api_key = os.environ.get('GOOGLE_API_KEY')
54     if not api_key:
55         ai_analysis = "AI analysis skipped: GOOGLE_API_KEY not configured"
56     else:
57         genai.configure(api_key=api_key)
58         model = genai.GenerativeModel("gemini-2.0-flash-exp")
59
60         prompt = f"""You are an AWS security expert. Analyze this S3 encryption scan and provide a brief security assessment.
61
62 Scan Results:
63
64 - Total Buckets: {len(buckets)}
65 - Encrypted: {len(buckets) - unencrypted_count}
66 - Unencrypted: {unencrypted_count}
67 - Unencrypted Bucket Names: {', '.join(unencrypted_buckets)} if unencrypted_buckets else 'None'
68
69 Provide a 2-3 sentence analysis:
70 1. What's the security risk of unencrypted buckets?
71 2. What encryption should be enabled? (AES256 or aws:kms)
72 3. What action should the user take immediately?
73
74 Be concise and actionable."""
75
76     try:
77         response = model.generate_content(prompt)
78         ai_analysis = response.text
79     except Exception as e:
80         ai_analysis = f"AI analysis failed: {str(e)}"
81
82     # Build final result
83     result = {
84         'total_buckets': len(buckets),
85         'unencrypted_buckets': unencrypted_count,
86         'encrypted_buckets': len(buckets) - unencrypted_count,
87         'scan_results': scan_results,
88         'ai_analysis': ai_analysis,
89         'alert': unencrypted_count > 0
90     }
91
92     print(f"\nScan complete: {unencrypted_count}/{len(buckets)} buckets need encryption")
93
94     return {
95         'statusCode': 200,
96         'body': json.dumps(result)
97     }
98

```

## Phase 2: Security & Permissions (IAM)

- **Created Read-Only Policy:** I logged into the IAM Console and authored a custom JSON policy. I granted the specific permissions `s3 :ListAllMyBuckets` and `s3 :GetEncryptionConfiguration` to ensure the principle of least privilege.

The screenshot shows the 'ReadS3EncryptionPolicy' details page. At the top, a green banner indicates the policy was created. Below it, the policy name is 'ReadS3EncryptionPolicy'. There are 'View policy', 'Edit', and 'Delete' buttons. The 'Policy details' section includes fields for Type (Customer managed), Creation time (January 26, 2026, 18:27 UTC-08:00), Edited time (January 26, 2026, 18:27 UTC-08:00), and ARN (arn:aws:iam::471744311739:policy/ReadS3EncryptionPolicy). Below this, tabs for 'Permissions', 'Entities attached', 'Tags', 'Policy versions (1)', and 'Last Accessed' are shown. The 'Permissions' tab is selected. The 'Permissions defined in this policy' section lists one service, S3, with the action 'Limited: List, Read' and resource 'All resources'. A search bar and a link to 'Show remaining 460 services' are also present.

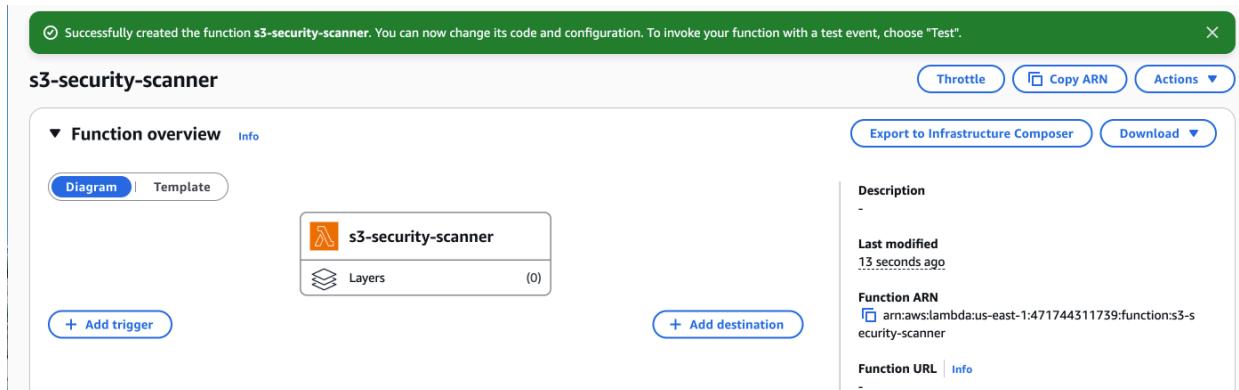
- **Established Execution Role:** I created an IAM Role named `LambdaS3ScannerRole` for the Lambda service. I attached my custom S3 policy along with the `AWSLambdaBasicExecutionRole` so the function could write logs to CloudWatch.

The screenshot shows the 'S3LambdaScannerRole' details page. At the top, a green banner indicates the role was created. Below it, the role name is 'S3LambdaScannerRole'. There are 'View role', 'Edit', and 'Delete' buttons. The 'Summary' section includes fields for Creation date (January 26, 2026, 18:29 UTC-08:00), ARN (arn:aws:iam::471744311739:role/S3LambdaScannerRole), and Maximum session duration (1 hour). Below this, tabs for 'Permissions', 'Trust relationships', 'Tags', 'Last Accessed', and 'Revoke sessions' are shown. The 'Permissions' tab is selected. The 'Permissions policies (2)' section lists two policies: 'AWSLambdaBasicExecutionRole' (AWS managed) and 'ReadS3EncryptionPolicy' (Customer managed). It includes a 'Filter by Type' dropdown set to 'All types', a search bar, and buttons for 'Simulate', 'Remove', and 'Add permissions'.

- **Configured Trust Relationship:** I verified that the trust policy correctly allows the `lambda.amazonaws.com` service to assume this role during execution.
- 

### Phase 3: Deployment (AWS Lambda)

- **Built Deployment Package:** I created a `requirements.txt` file and used `pip` to install dependencies into a local directory. To keep the package under the 50MB limit, I manually removed the pre-installed `boto3` and `botocore` libraries before zipping the files.
- **Initialized Lambda Function:** I created a new function in the AWS console using the **Python 3.12** runtime. I assigned the `LambdaS3ScannerRole` I built earlier to provide the necessary "identity" for the scan.



- **Configured Runtime & Secrets:** I updated the **Handler** setting to `s3_scanner.lambda_handler` to match my file name. I then added my Gemini API key as an **Environment Variable** (`GOOGLE_API_KEY`) to keep the credential out of the source code.
- **Adjusted Resources:** I increased the function **timeout to 30 seconds** to ensure the scan wouldn't cut off if I added more buckets in the future. As a result the scanned logs can be seen below in the screenshot.

Dhruval Rana

## Phase 4: Automation & Verification

- **Executed Manual Test:** I triggered the function using a blank test event. I then navigated to **CloudWatch Logs** to confirm the output. I observed that the AI successfully identified my unencrypted buckets and provided actionable remediation steps.

The screenshot shows the AWS Lambda Rule details page for a rule named 'daily-s3-security-scan'. The top navigation bar includes a success message 'Rule daily-s3-security-scan was updated successfully' and standard actions like Edit, Disable, Delete, and CloudFormation Template.

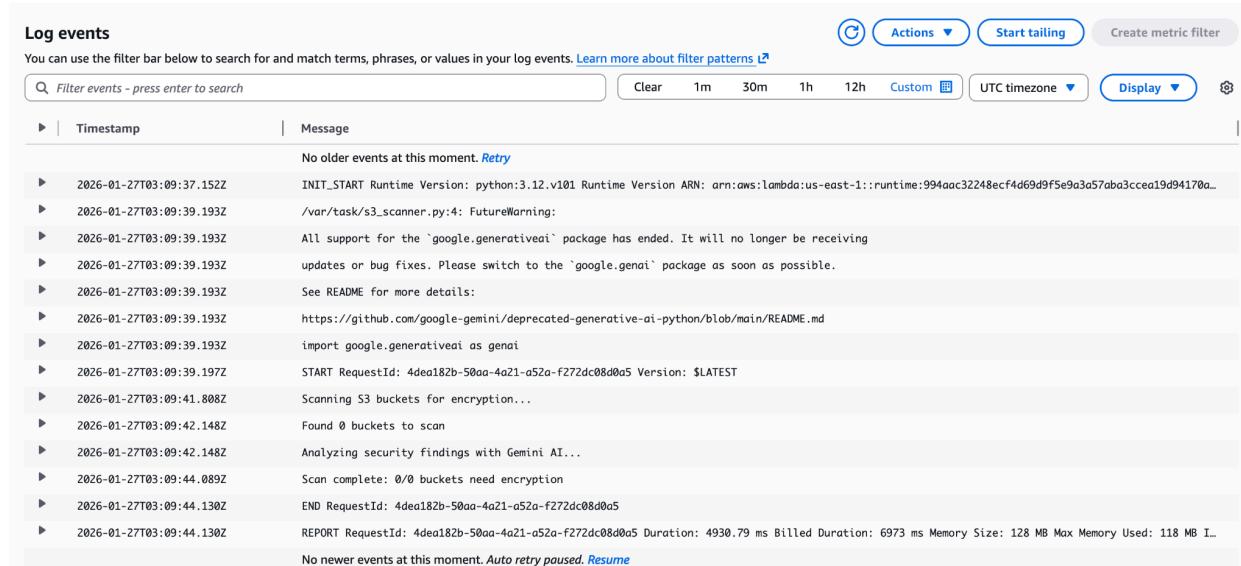
**Rule details** (Info)

<b>Rule name</b> daily-s3-security-scan	<b>Status</b> Enabled	<b>Event bus name</b> default	<b>Type</b> Scheduled Standard
<b>Description</b> Daily automated S3 encryption scan	<b>Rule ARN</b> <a href="#">arn:aws:events:us-east-1:471744311739:rule/daily-s3-security-scan</a>	<b>Event bus ARN</b> <a href="#">arn:aws:events:us-east-1:471744311739:event-bus/default</a>	

**Event schedule** (Info) Edit

Fixed rate of  
24 hour

- **Scheduled Automation:** I moved to the **Amazon EventBridge** console and created a new "Scheduled Rule." I set a **rate of 12 hours** and selected my Lambda function as the target.
- **Validated Live Results:** To verify the automation, I temporarily changed the schedule to 1 minute. I watched the logs appear in real-time, confirming that my account is now being audited automatically twice a day.



The screenshot shows the AWS CloudWatch Log Events interface. At the top, there are filter and search options, along with time range buttons (Clear, 1m, 30m, 1h, 12h, Custom, UTC timezone, Display). Below the header is a table with two columns: 'Timestamp' and 'Message'. The 'Timestamp' column shows dates from January 2026. The 'Message' column contains log entries related to a Lambda function's execution, including initialization, dependency loading, file processing, and reporting results.

Timestamp	Message
2026-01-27T03:09:37.152Z	No older events at this moment. <a href="#">Retry</a>
2026-01-27T03:09:39.193Z	INIT_START Runtime Version: python:3.12.v101 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:994aac32248ecf4d69d9f5e9a3a57aba3cce19d94170a... /var/task/s3_scanner.py:4: FutureWarning:
2026-01-27T03:09:39.193Z	All support for the `google.generativeai` package has ended. It will no longer be receiving updates or bug fixes. Please switch to the `google.genai` package as soon as possible.
2026-01-27T03:09:39.193Z	See README for more details:
2026-01-27T03:09:39.193Z	<a href="https://github.com/google-gemini/deprecated-generative-ai-python/blob/main/README.md">https://github.com/google-gemini/deprecated-generative-ai-python/blob/main/README.md</a>
2026-01-27T03:09:39.193Z	import google.generativeai as genai
2026-01-27T03:09:39.197Z	START RequestId: 4dea182b-50aa-4a21-a52a-f272dc08d0a5 Version: \$LATEST
2026-01-27T03:09:41.808Z	Scanning S3 buckets for encryption...
2026-01-27T03:09:42.148Z	Found 0 buckets to scan
2026-01-27T03:09:42.148Z	Analyzing security findings with Gemini AI...
2026-01-27T03:09:44.089Z	Scan complete: 0/0 buckets need encryption
2026-01-27T03:09:44.130Z	END RequestId: 4dea182b-50aa-4a21-a52a-f272dc08d0a5
2026-01-27T03:09:44.130Z	REPORT RequestId: 4dea182b-50aa-4a21-a52a-f272dc08d0a5 Duration: 4930.79 ms Billed Duration: 6973 ms Memory Size: 128 MB Max Memory Used: 118 MB I...
	No newer events at this moment. <a href="#">Auto retry paused.</a> <a href="#">Resume</a>

**Summary of My Results:** I successfully transitioned from a manual script to a 24/7 automated security auditor. The system now identifies vulnerabilities and translates them into a format that anyone on a security team can understand instantly.