

Enterprise Configuration Monitoring and Compliance Enforcement on AWS

Author: Adedayo

Specialization: Cloud Security & Compliance

Platform: Amazon Web Services (AWS)

## 1. Problem Statement

A regulated enterprise required continuous monitoring of AWS configurations to maintain compliance standards. Manual configuration reviews were inefficient and prone to oversight. A centralized compliance monitoring and automated remediation framework was needed.

I implemented this framework to ensure cloud resources comply with security and regulatory requirements aligned with CIS, NIST, and ISO 27001 standards.

The solution uses AWS-native services to provide continuous monitoring, automated remediation, and audit-ready reporting.

## 2. Objectives

The primary objectives were to:

- Enforce security and configuration standards
- Preventing insecure resource deployment
- Detect configuration drift
- Automate remediation
- Support audit and compliance reporting
- Improve governance and accountability

## 3. Compliance Monitoring Architecture

The compliance framework was built using:

- AWS Config
- Amazon EventBridge
- AWS Lambda
- AWS Security Hub
- Amazon CloudWatch

All supported resources are continuously recorded and evaluated against defined rules.

## 4. EBS Encryption Compliance Control

### 4.1 Control Overview

All Amazon EBS volumes must be encrypted at rest to protect sensitive data.

### 4.2 Compliance Requirements

- CIS AWS Foundations Benchmark
- NIST SP 800-53 (SC-12, SC-28)
- ISO/IEC 27001 (A.10.1)

### 4.3 AWS Config Implementation

The AWS-managed rule encrypted-volumes was enabled to monitor EBS encryption.

EBS encryption by default was enabled at the account level.

### 4.4 Compliance Testing

Attempts to create unencrypted volumes were blocked, confirming preventive enforcement.

### 4.5 Remediation Strategy

Preventive controls were prioritized.

For legacy volumes, snapshots were encrypted and migrated.

Status: Compliant

## 5. S3 Public Access Compliance Control

## 5.1 Control Overview

All S3 buckets must be protected from public access.

## 5.2 Compliance Requirements

- CIS AWS Foundations Benchmark
- NIST SP 800-53 (AC-3, AC-6)
- ISO/IEC 27001 (A.9)

## 5.3 AWS Config Implementation

Managed rules enabled:

- s3-bucket-public-read-prohibited

The screenshot shows the 'Configure rule' wizard in the AWS Config console. The current step is 'Step 1: Specify rule type', which has been completed. The next step, 'Step 2: Configure rule', is the active one. The final step, 'Step 3: Review and create', is shown at the bottom. The 'Configure rule' section is titled 'Configure rule' and includes a note: 'Customize any of the following fields'. The 'Details' section contains the following fields:

- Name:** A unique name for the rule, 128 characters max. No special characters or spaces.  
Value: s3-bucket-public-read-prohibited
- Description - optional:** Describe what the rule evaluates and how to fix resources that don't comply.  
Value: Checks that your Amazon S3 buckets do not allow public read access. The rule checks the Block Public Access settings, the bucket policy, and the bucket access control list (ACL).
- Managed rule name:**  
Value: S3\_BUCKET\_PUBLIC\_READ\_PROHIBITED

The 'Evaluation mode' section is partially visible below, showing an option to turn on proactive evaluation.

- s3-bucket-public-write-prohibited

AWS Config > Rules > Add rule

Step 1 Specify rule type

Step 2 Configure rule

Step 3 Review and create

### Configure rule

Customize any of the following fields

#### Details

**Name:** A unique name for the rule. 128 characters max. No special characters or spaces.  
s3-bucket-public-write-prohibited

**Description - optional:** Describe what the rule evaluates and how to fix resources that don't comply.  
Checks that your Amazon S3 buckets do not allow public write access. The rule checks the Block Public Access settings, the bucket policy, and the bucket access control list (ACL).

**Managed rule name:** S3\_BUCKET\_PUBLIC\_WRITE\_PROHIBITED

#### Evaluation mode

Turn on proactive evaluation  
Enable evaluation of resources prior to provisioning.

## 5.4 Compliance Testing

Public access was introduced intentionally and detected successfully.

**AWS Config**

- Dashboard
- Conformance packs
- Rules**
- Resources
- Aggregators
  - Compliance Dashboard
  - Conformance packs
  - Rules
  - Inventory Dashboard
  - Resources
  - Authorizations
  - Advanced queries
  - Settings
  - What's new
- Documentation
- Partners
- FAQs
- Pricing

**Frameworks**

- AWS-WAF-v1.0
- PCI-DSS-v4.0
- ISO-IEC-27001:2013-Annex-A
- NIST-SP-800-53-r5
- CCCS-Medium-Cloud-Control-May-2019
- CIS-AWS-Benchmark-v1.2
- PCI-DSS-v3.2.1
- CIS-AWS-Benchmark-v1.3
- CIS-AWS-Benchmark-v1.4

<b>Scope of changes</b>	<b>Last evaluation status</b>	<b>Trigger type</b>
Resources	Successful	<ul style="list-style-type: none"> <li>Periodic: 24 hours</li> <li>Configuration changes</li> </ul>
<b>Remediation action</b>	<b>Last successful evaluation time</b>	<b>AWS Config resource types</b>
Not set	January 21, 2026 8:21 PM	S3 Bucket
	<b>Detective compliance</b>	
	Noncompliant resource(s)	

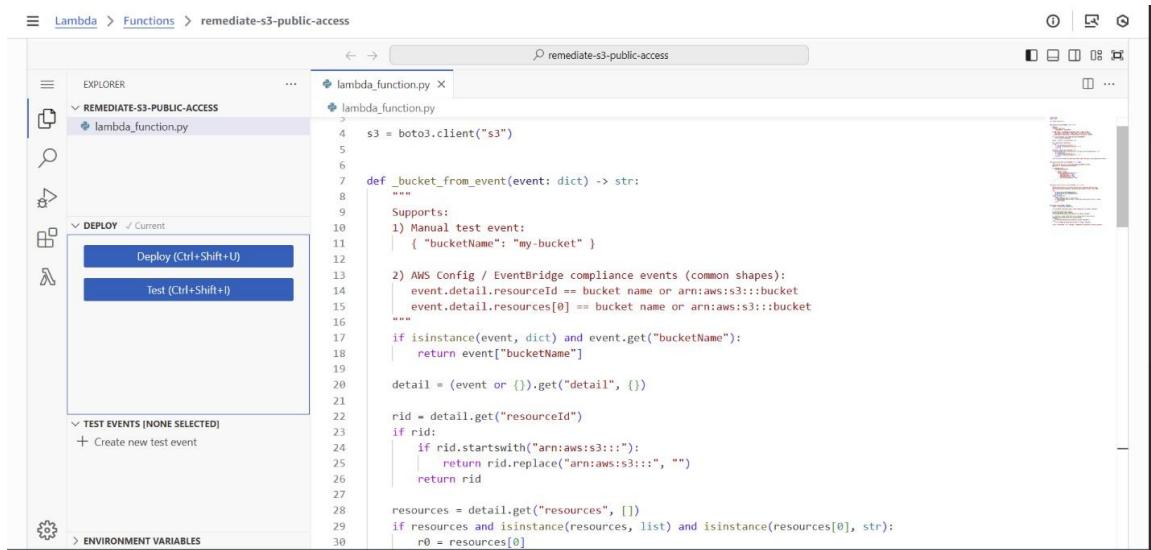
**Resources in scope**

ID	Type	Status	Annotation	Compliance
secureops-testbucket	S3 Bucket	-	The S3 bucket policy allows public read access.	⚠ Noncompliant

## 5.5 Automated Remediation

A Lambda-based remediation workflow was implemented using EventBridge.

The function:

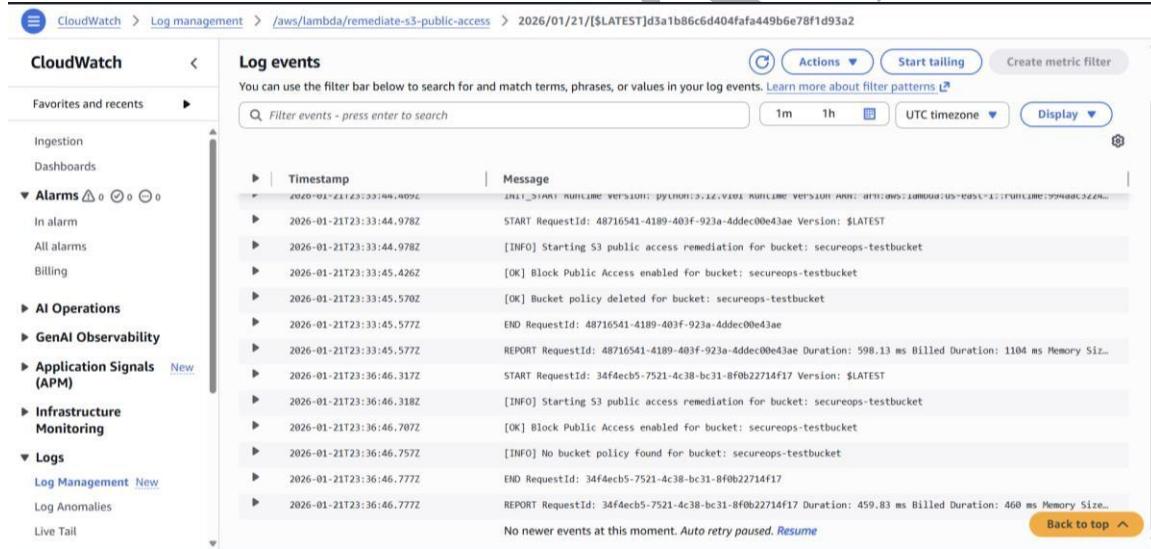


```

 4     s3 = boto3.client("s3")
 5
 6     def _bucket_from_event(event: dict) -> str:
 7         """
 8             Supports:
 9                 1) Manual test event:
10                     {
11                         "bucketName": "my-bucket"
12                     }
13
14             2) AWS Config / EventBridge compliance events (common shapes):
15                 event.detail.resourceId == bucket name or arn:aws:s3:::bucket
16                 event.detail.resources[0] == bucket name or arn:aws:s3:::bucket
17
18             if isinstance(event, dict) and event.get("bucketName"):
19                 return event["bucketName"]
20
21             detail = (event or {}).get("detail", {})
22
23             rid = detail.get("resourceId")
24             if rid:
25                 if rid.startswith("arn:aws:s3:::"):
26                     return rid.replace("arn:aws:s3:::", "")
27                 return rid
28
29             resources = detail.get("resources", [])
30             if resources and isinstance(resources, list) and isinstance(resources[0], str):
31                 r0 = resources[0]

```

- Enables Block Public Access
- Removes insecure policies
- Logs actions to CloudWatch



**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](#)

Timestamp	Message
2026-01-21T23:33:44.978Z	START RequestId: 48716541-4189-403f-923a-4dddec00e43ae Version: \$LATEST
2026-01-21T23:33:44.978Z	[INFO] Starting S3 public access remediation for bucket: secureops-testbucket
2026-01-21T23:33:45.426Z	[OK] Block Public Access enabled for bucket: secureops-testbucket
2026-01-21T23:33:45.570Z	[OK] Bucket policy deleted for bucket: secureops-testbucket
2026-01-21T23:33:45.577Z	END RequestId: 48716541-4189-403f-923a-4dddec00e43ae
2026-01-21T23:33:45.577Z	REPORT RequestId: 48716541-4189-403f-923a-4dddec00e43ae Duration: 598.13 ms Billed Duration: 1104 ms Memory Size: 32 MB
2026-01-21T23:36:46.317Z	START RequestId: 34f4ecb5-7521-4c38-bc31-8f0b2271af17 Version: \$LATEST
2026-01-21T23:36:46.318Z	[INFO] Starting S3 public access remediation for bucket: secureops-testbucket
2026-01-21T23:36:46.707Z	[OK] Block Public Access enabled for bucket: secureops-testbucket
2026-01-21T23:36:46.757Z	[INFO] No bucket policy found for bucket: secureops-testbucket
2026-01-21T23:36:46.777Z	END RequestId: 34f4ecb5-7521-4c38-bc31-8f0b2271af17
2026-01-21T23:36:46.777Z	REPORT RequestId: 34f4ecb5-7521-4c38-bc31-8f0b2271af17 Duration: 459.83 ms Billed Duration: 460 ms Memory Size: 32 MB

Status: Compliant (Automated Remediation)

## 6. Account-Level Preventive Controls

Account-level S3 Block Public Access and EBS encryption by default were reviewed and enforced.

These controls prevent misconfigurations before deployment.

## 7. Compliance Mapping and Alignment

Controls were mapped to major frameworks:

- CIS AWS Foundations
- NIST SP 800-53
- ISO/IEC 27001

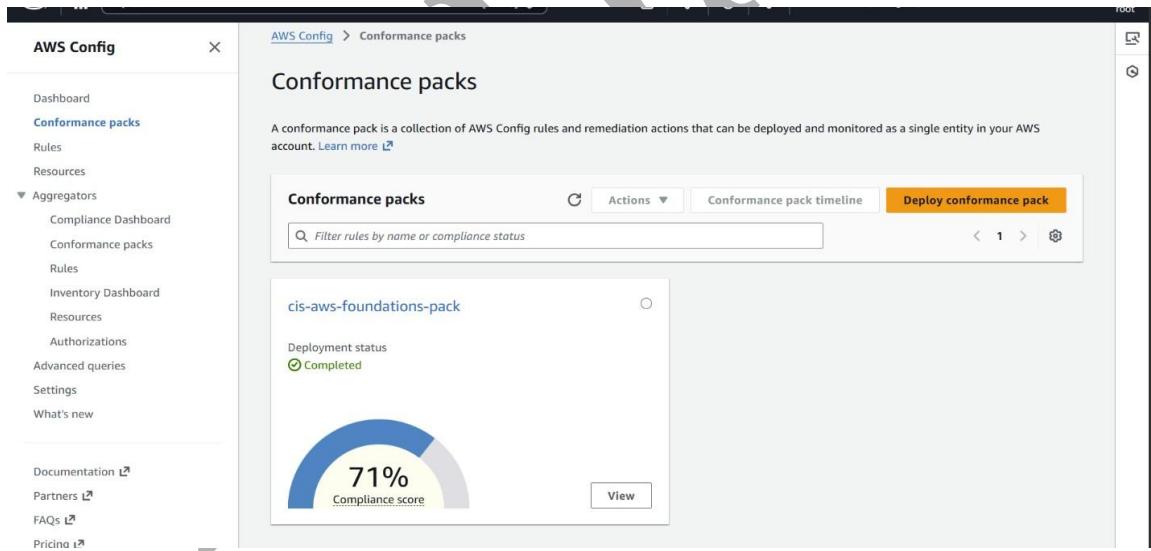
Preventive, detective, and corrective controls were implemented.

## 8. Conformance Pack Deployment

An AWS Config CIS conformance pack was deployed.

The pack automatically evaluated multiple security controls.

Dashboards provided visibility into compliance posture.



## 9. Automated Remediation Framework

Compliance change events trigger EventBridge rules.

Lambda functions perform corrective actions.

Manual procedures were defined as fallback.

## 10. Governance and Documentation

All compliance rules, workflows, and remediation processes were documented.

Documentation supports:

- Regulatory audits
- Internal reviews
- Incident investigations
- Knowledge transfer

## 11. Reporting and Visibility

AWS Config dashboards and Security Hub were used for reporting.

Reports include:

- Compliance status
- Violations
- Remediation progress

Reports are reviewed regularly.

## 12. Testing and Validation

Controls were validated using intentional misconfigurations:

- Public S3 buckets
- Disabled encryption
- Policy changes

Violations were detected and remediated automatically.

## 13. Outcomes and Impact

This implementation delivered:

- Continuous compliance monitoring
- Reduced configuration drift
- Automated remediation
- Improved governance
- Audit-ready reporting
- Reduced regulatory risk

## 14. Conclusion

I designed and implemented an enterprise-grade compliance monitoring and enforcement framework on AWS.

Through continuous evaluation, automated remediation, and governance controls, this solution ensures secure, compliant, and well-governed cloud operations.