

Enterprise Configuration Monitoring and Compliance Enforcement on AWS

Author: Adedayo

Specialization: Cloud Security & Compliance

Platform: Amazon Web Services (AWS)

## 1. Introduction

This document describes the design, implementation, and validation of a configuration monitoring and compliance enforcement framework in AWS.

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
- Prevent 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.

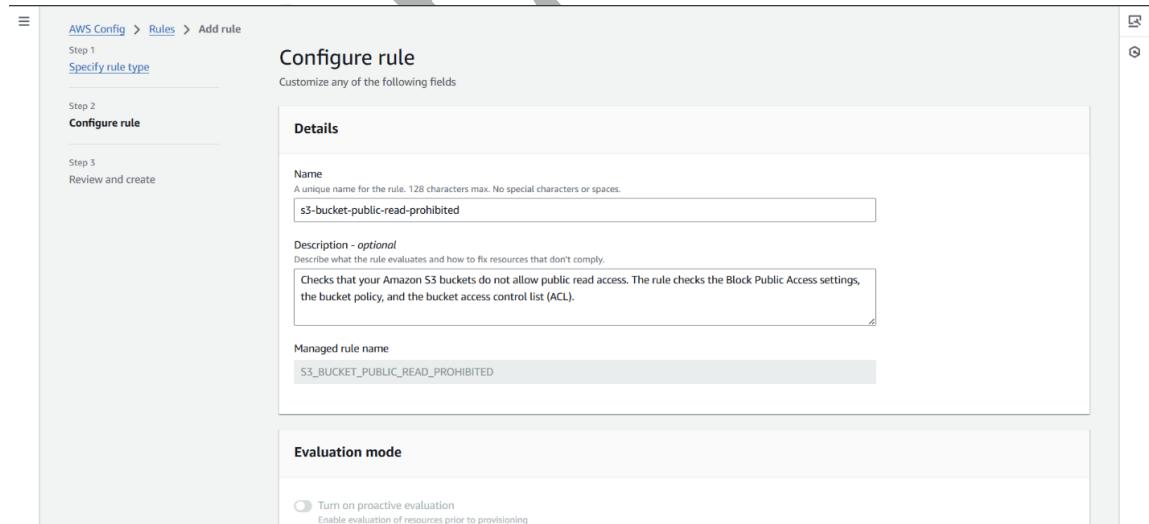
### 5.2 Compliance Requirements

- 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 AWS Config 'Configure rule' interface. The left sidebar has steps: Step 1 (Specify rule type), Step 2 (Configure rule - selected), and Step 3 (Review and create). The main area is titled 'Configure rule' with the sub-section 'Details'. It shows a 'Name' field containing 's3-bucket-public-read-prohibited', a 'Description - optional' field with the note '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).', and a 'Managed rule name' field with 'S3\_BUCKET\_PUBLIC\_READ\_PROHIBITED'. At the bottom, there's an 'Evaluation mode' section with a radio button for 'Turn on proactive evaluation'.

- s3-bucket-public-write-prohibited

The screenshot shows the AWS Config 'Configure rule' step. The left sidebar has three steps: Step 1 'Specify rule type', Step 2 'Configure rule' (selected), and Step 3 'Review and create'. The main area is titled 'Configure rule' with the sub-section 'Details'. It shows a 'Name' field containing 's3-bucket-public-write-prohibited' and a 'Description - optional' field with the text: '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).'. Below this is a 'Managed rule name' field with 'S3\_BUCKET\_PUBLIC\_WRITE\_PROHIBITED'. At the bottom is an 'Evaluation mode' section with a radio button for 'Turn on proactive evaluation'.

## 5.4 Compliance Testing

Public access was introduced intentionally and detected successfully.

The screenshot shows the AWS Config 'Compliance Dashboard'. The left sidebar includes 'Dashboard', 'Conformance packs', 'Rules', 'Resources', 'Aggregators' (selected), 'Compliance Dashboard', 'Conformance packs', 'Rules', 'Inventory Dashboard', 'Resources', 'Authorizations', 'Advanced queries', 'Settings', and 'What's new'. The right panel displays a table for an S3 Bucket. The table columns are 'Frameworks' (listing 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), 'Scope of changes' (Resource), 'Last evaluation status' (Successful), 'Trigger type' (Periodic: 24 hours, Configuration changes), 'Resources in scope' (Noncompliant), and 'AWS Config resource types' (S3 Bucket). A detailed view of the 'Resources in scope' table shows one entry: ID: 'secureops-testbucket', Type: 'S3 Bucket', Status: '-', Annotation: 'The S3 bucket policy allows public read access.', and Compliance: 'Noncompliant'.

## 5.5 Automated Remediation

A Lambda-based remediation workflow was implemented using EventBridge.

The function:

The screenshot shows the AWS Lambda Function Editor interface. The left sidebar displays the project structure under 'REMEDIEATE-S3-PUBLIC-ACCESS' with 'lambda\_function.py' selected. The main editor area shows the Python code for the function:

```

lambda_function.py
lambda_function.py

def _bucket_from_event(event: dict) -> str:
    """
    Supports:
    1) Manual test event:
        {
            "bucketName": "my-bucket"
        }

    2) AWS Config / EventBridge compliance events (common shapes):
        event.detail.resourceId == bucket name or arn:aws:s3:::bucket
        event.detail.resources[0] == bucket name or arn:aws:s3:::bucket
    """

    if isinstance(event, dict) and event.get("bucketName"):
        return event["bucketName"]

    detail = (event or {}).get("detail", {})

    rid = detail.get("resourceId")
    if rid:
        if rid.startswith("arn:aws:s3:::"):
            return rid.replace("arn:aws:s3:::", "")

    return rid

resources = detail.get("resources", [])
if resources and isinstance(resources, list) and isinstance(resources[0], str):
    r0 = resources[0]

```

Below the code, there are buttons for 'Deploy (Ctrl+Shift+U)' and 'Test (Ctrl+Shift+I)'. A sidebar on the left also shows 'TEST EVENTS (NONE SELECTED)' and 'Create new test event'. The bottom right corner features a large, stylized 'Lambda' logo.

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

The screenshot shows the CloudWatch Log Management interface. The left sidebar lists various monitoring categories like Alarms, AI Operations, Application Signals, Infrastructure Monitoring, and Logs. The main area shows log events for the function execution:

Timestamp	Message
2026-01-21T23:33:44.978Z	START RequestId: 48710541-4189-403f-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: 48710541-4189-403f-923a-4dddec00e43ae
2026-01-21T23:33:45.577Z	REPORT RequestId: 48710541-4189-403f-923a-4dddec00e43ae Duration: 598.13 ms Billed Duration: 1104 ms Memory Size: 512 MB
2026-01-21T23:36:46.318Z	START RequestId: 34f4ecb5-7521-4c38-bc31-8f0b2271af17 Version: \$LATEST
2026-01-21T23:36:46.707Z	[INFO] Starting S3 public access remediation for bucket: secureops-testbucket
2026-01-21T23:36:46.757Z	[OK] Block Public Access enabled for bucket: secureops-testbucket
2026-01-21T23:36:46.777Z	[INFO] No bucket policy found for bucket: secureops-testbucket
2026-01-21T23:36:46.777Z	END RequestId: 34f4ecb5-7521-4c38-bc31-8f0b2271af17
	REPORT RequestId: 34f4ecb5-7521-4c38-bc31-8f0b2271af17 Duration: 459.83 ms Billed Duration: 460 ms Memory Size: 512 MB

No newer events at this moment. Auto retry paused. [Resume](#)

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

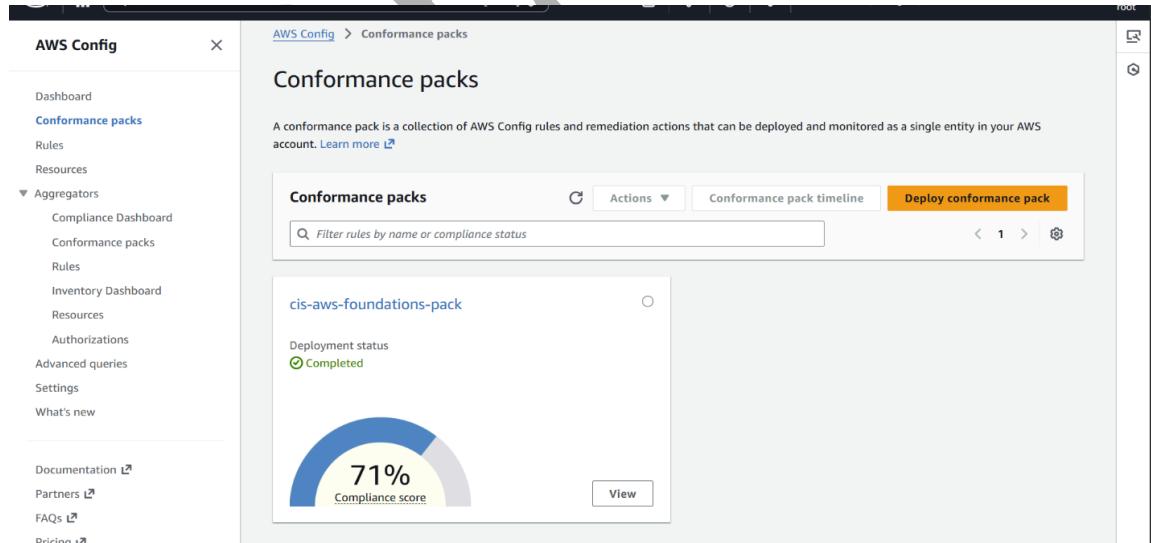
Preventive, detective, and corrective controls were implemented.

## 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.