Enterprise Data Protection and Encryption Management on AWS
Author: Adedayo
Specialization: Cloud Security & Data Protection
Platform: Amazon Web Services (AWS)

1. Problem Statement

A financial services organization handling sensitive financial records required consistent encryption governance across storage, databases, and logs. Inconsistent encryption configurations created regulatory risk.

I implemented this framework to ensure that sensitive data stored in S3, EBS, and RDS is protected using strong encryption, centralized key management, and continuous compliance monitoring.

2. Objectives

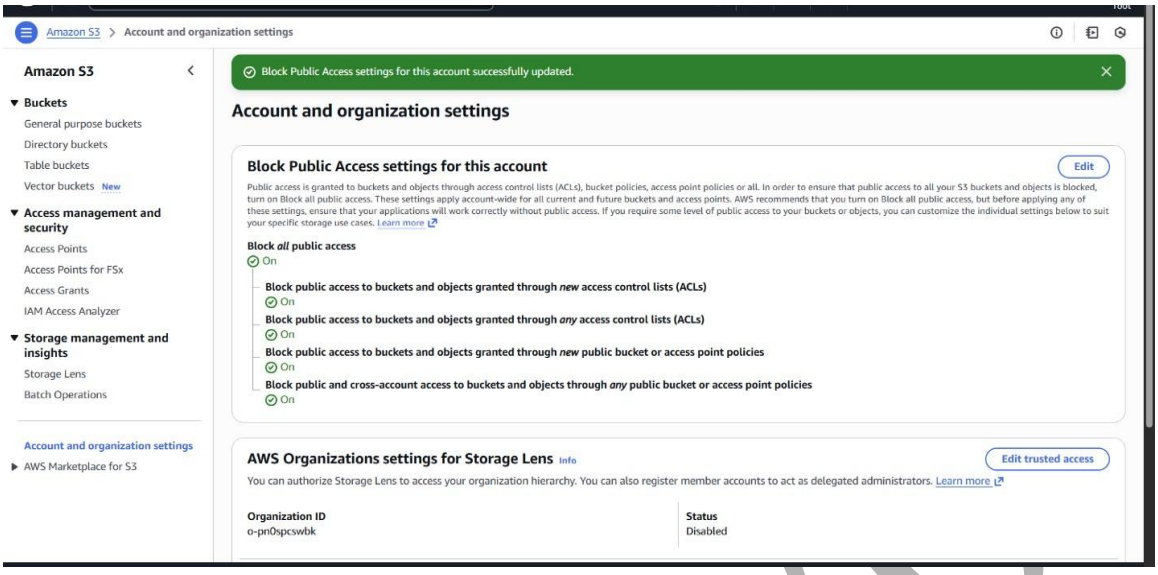
The primary objectives of this implementation were to:

- Prevent public exposure of sensitive data
- Enforce encryption at rest across all storage services
- Centralize key management using AWS KMS
- Enable automatic and manual key rotation
- Maintain continuous compliance visibility
- Reduce the risk of data leakage

3. S3 Data Protection

3.1 Account-Level Public Access Controls

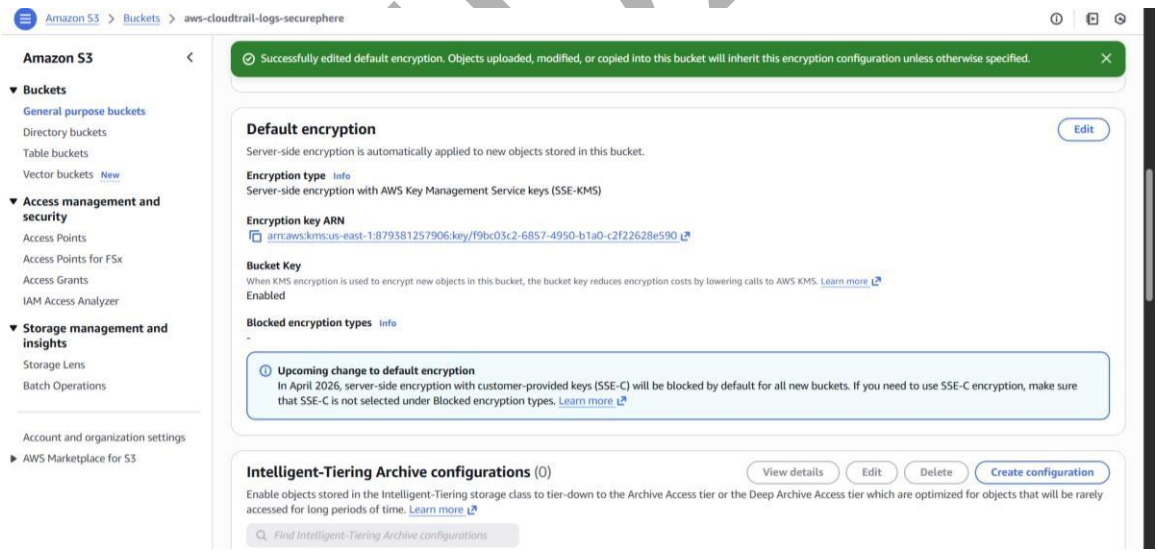
I enabled S3 Block Public Access at the account level to prevent accidental public exposure of any bucket.



This ensured that all existing and newly created buckets were protected by default.

3.2 S3 Encryption

All S3 buckets were encrypted using AWS SSE-KMS with customer-managed keys.

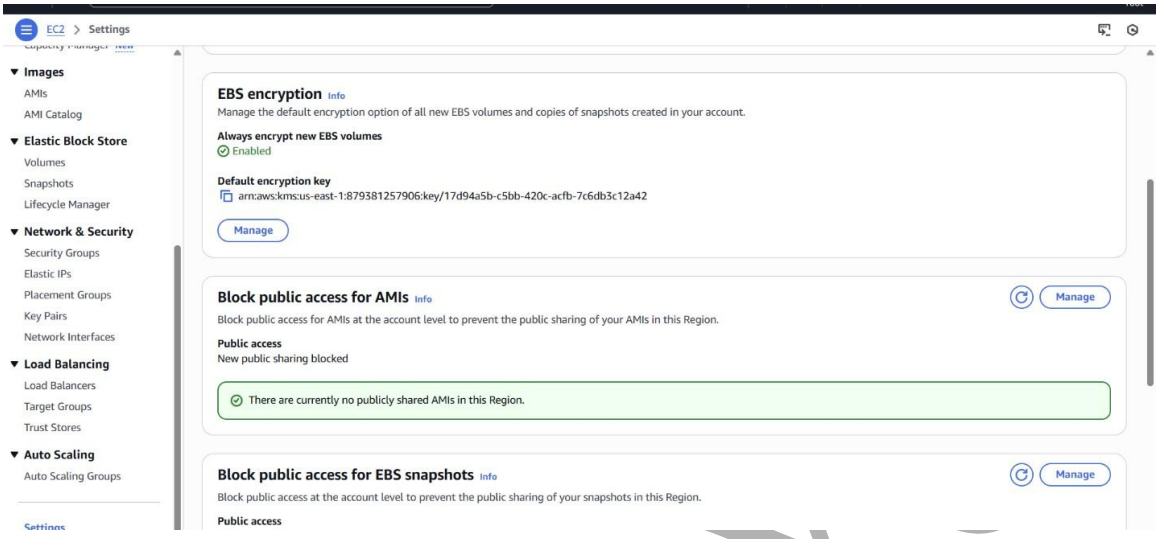


This provided full control over encryption policies, auditing, and key lifecycle management.

4. EBS Encryption Management

4.1 Default Encryption

EBS encryption by default was enabled to ensure that all new volumes and snapshots are automatically encrypted.



4.2 Migrating Unencrypted Volumes

An unencrypted EBS volume was identified on an existing EC2 instance.

To remediate this:

- A snapshot was created from the unencrypted volume
- The snapshot was copied and encrypted
- A new encrypted volume was created
- The instance was stopped
- The encrypted volume was attached

This process ensured encryption without data loss.

5. RDS Encryption Enforcement

All RDS databases were reviewed to confirm encryption at rest.

Encryption was enforced at creation time, as RDS encryption cannot be enabled after deployment.

Operational guidance was provided to ensure future databases follow this standard.

6. Key Management with AWS KMS

6.1 Customer-Managed Keys

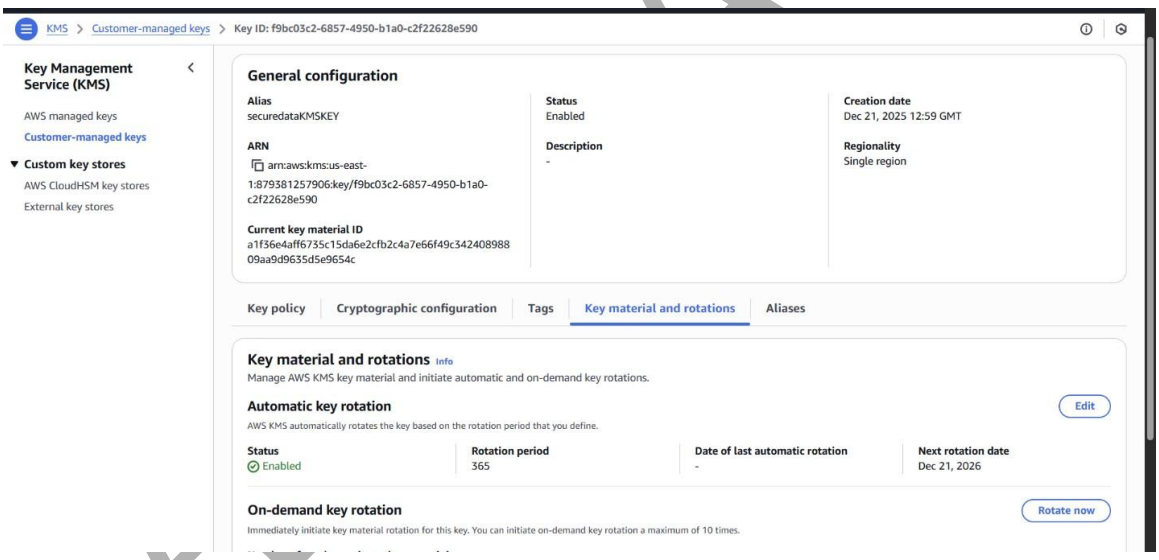
A customer-managed symmetric KMS key was created for workloads.

This key was used to encrypt:

- S3 buckets
- EBS volumes
- RDS databases

6.2 Automatic Key Rotation

Automatic key rotation was enabled with a 365-day cycle.



This reduced the risk associated with long-lived cryptographic keys.

6.3 Manual Key Rotation Process

For keys that do not support automatic rotation, a documented manual rotation process was implemented:

Step 1: Identify the active key

Step 2: Create a new customer-managed key

Step 3: Update services to use the new key

Step 4: Monitor system behavior

Step 5: Disable the old key

Step 6: Retain the old key for recovery and compliance

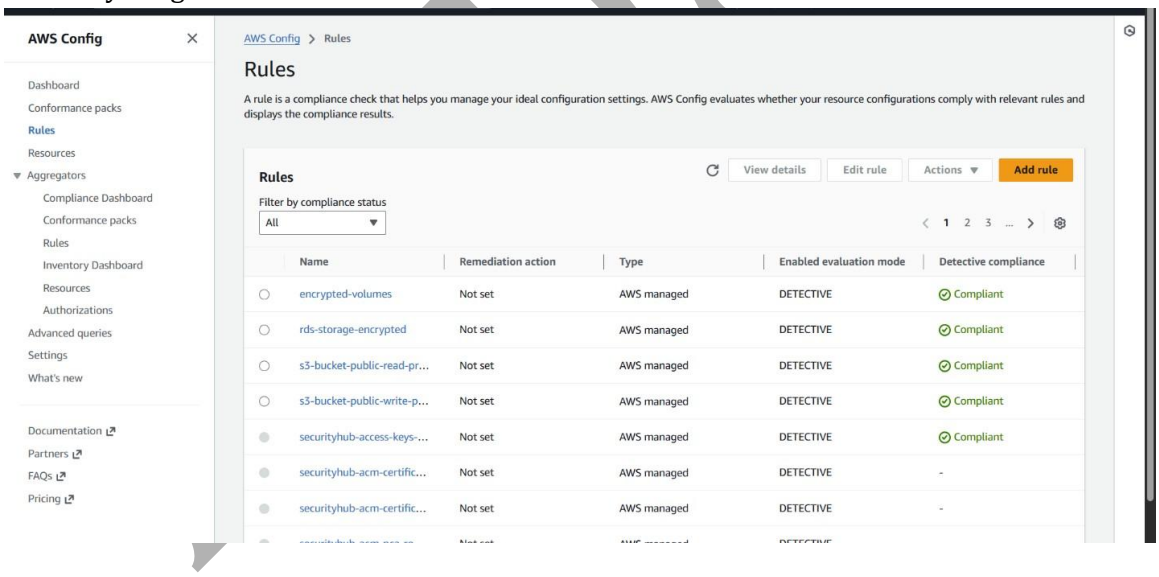
This process ensured secure rotation without service disruption.

7. Compliance Monitoring and Alerting

7.1 AWS Config Rules

AWS Config rules were created to monitor:

- S3 public access settings
- Storage encryption status
- KMS key usage



These rules provided real-time compliance visibility.

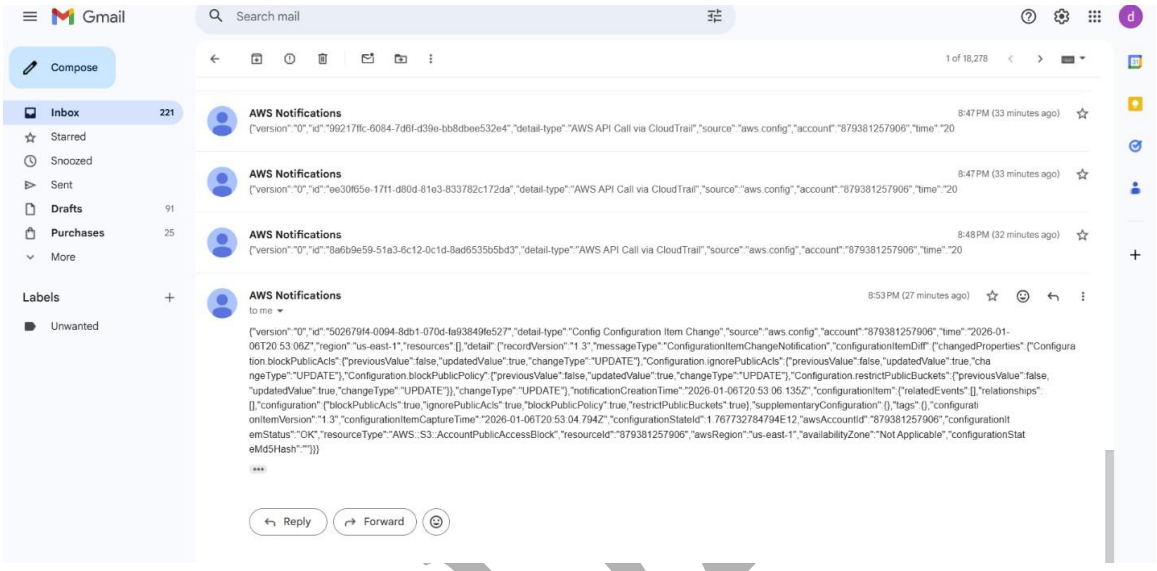
7.2 Alerting with SNS

Amazon SNS topics were configured to deliver alerts when non-compliant resources were detected.

7.3 Validation Testing

Public access settings were temporarily modified to validate alert delivery.

Successful notifications confirmed correct configuration.



8. Governance and Documentation

All encryption policies, key management procedures, and compliance controls were documented.

This supported:

- Security audits
- Regulatory compliance
- Incident investigations
- Operational continuity

9. Outcomes and Impact

This implementation delivered the following results:

- Eliminated public S3 exposure risk

- Enforced encryption across storage services
- Centralized cryptographic key management
- Reduced operational risk
- Improved audit readiness
- Enabled real-time compliance monitoring

10. Conclusion

I designed and implemented a secure, compliant, and centrally managed data protection framework on AWS.

Through strong encryption, effective key management, and continuous monitoring, this solution protects sensitive data and supports regulatory requirements.