****

**Internship Report:**

Detailed Setup of IAM Policies, Secure Storage, and Data Encryption

**Submitted by: Ritwiz**

**Intern ID:CT04DH918**

**Table of Contents**

**1.Introduction**

**2.IAM Policies Setup**2.1 User and Group Creation  
2.2 Custom Policy Definition  
2.3 Role-Based Access Control (RBAC)  
2.4 IAM Best Practices Implemented

**3.Secure Storage Setup (Amazon S3)**3.1 Bucket Configuration  
3.2 Access Control Strategies  
3.3 Bucket Policy Example to Enforce Encryption  
3.4 Logging and Monitoring  
3.5 Data Lifecycle Management

**4.Data Encryption Setup**4.1 Server-Side Encryption (SSE)  
4.2 AWS KMS Setup  
4.3 Client-Side Encryption  
4.4 Enforcing Encryption via Bucket Policy  
4.5 Encryption in Transit

**5.Challenges and Resolutions**

**6.Conclusion**

**Introduction**

With the rapid adoption of cloud computing, securing digital assets, controlling access, and protecting data have become crucial. Organizations must ensure that sensitive information is adequately protected from unauthorized access, data leaks, and compliance breaches. Cloud providers like Amazon Web Services (AWS) offer robust tools and services for managing identity, access, and data security. This report details the practical implementation and configuration of IAM (Identity and Access Management) policies, secure storage configurations using Amazon S3, and data encryption techniques using AWS Key Management Service (KMS).

This internship project was aimed at gaining hands-on experience in building secure cloud infrastructure components by:

* Designing and applying granular IAM policies to enforce least privilege access control.
* Implementing secure data storage mechanisms using AWS S3.
* Applying data encryption at rest and in transit using server-side and client-side techniques.

**IAM Policies Setup**

IAM policies are JSON-based documents used to define permissions for users, groups, and roles in AWS. The core aim was to manage and restrict access effectively.

**2.1 User and Group Creation**

* Created users for different roles: developers, auditors, testers.
* Grouped users based on function to simplify permission management (e.g., DevGroup, AdminGroup).

**2.2 Custom Policy Definition**

* Example: Created a policy to allow users to upload files to a specific S3 bucket only.

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Action": "s3:PutObject",

"Resource": "arn:aws:s3:::secure-data-bucket/\*"

}

]

}

**2.3 Role-Based Access Control (RBAC)**

* Used IAM roles to assign temporary permissions to EC2 instances, Lambda functions, and cross-account users.
* Ensured least privilege by avoiding wildcards and overly broad actions.

**2.4 IAM Best Practices Implemented**

* Enabled MFA for all privileged accounts.
* Regularly reviewed IAM Access Analyzer findings to detect unintended access.
* Avoided use of root account for any routine operations.

**Secure Storage Setup (Amazon S3)**

Secure storage was ensured through proper configuration of Amazon S3, including access control, logging, and encryption.

**3.1 Bucket Configuration**

* Created buckets with descriptive naming conventions (e.g., "prod-backup-logs").
* Enabled versioning to retain old versions of files.

**3.2 Access Control Strategies**

* Blocked public access settings enforced by default.
* Used a combination of IAM policies and bucket policies for fine-grained control.
* Disabled ACLs (Access Control Lists) to avoid conflicts.

**3.3 Bucket Policy Example to Enforce Encryption**

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "DenyUnencryptedUploads",

"Effect": "Deny",

"Principal": "\*",

"Action": "s3:PutObject",

"Resource": "arn:aws:s3:::secure-data-bucket/\*",

"Condition": {

"StringNotEquals": {

"s3:x-amz-server-side-encryption": "AES256"

}

}

}

]

}

**3.4 Logging and Monitoring**

* Enabled S3 server access logs.
* Integrated S3 with AWS CloudTrail and AWS CloudWatch for monitoring and alerting.

**3.5 Data Lifecycle Management**

* Configured lifecycle rules to archive data to Glacier after 90 days.
* Deleted data automatically after 365 days to reduce storage costs.

**Data Encryption Setup**

Data encryption protects sensitive data by encoding it. AWS provides various encryption options that were explored and implemented during the project.

**4.1 Server-Side Encryption (SSE)**

* Enabled SSE-S3 (Amazon S3 managed keys) by default on buckets.
* Used SSE-KMS for sensitive data, providing enhanced key management features.

**4.2 AWS KMS Setup**

* Created Customer Managed Keys (CMKs) using AWS KMS.
* Defined key policies to restrict access to CMKs.
* Enabled automatic key rotation for compliance.

**4.3 Client-Side Encryption**

* Used AWS SDK (boto3 in Python) to encrypt data before uploading to S3.
* Managed local keys securely and stored encrypted objects in designated buckets.

**4.4 Enforcing Encryption via Bucket Policy**

* Configured bucket policies to deny unencrypted uploads.
* Ensured that only encrypted objects using SSE-KMS were accepted.

**4.5 Encryption in Transit**

* Enforced HTTPS for all API interactions.
* Verified TLS connections during client-server communication.

**Challenges and Resolutions**

**Challenge 1:** Misconfigured IAM policies led to denied access for legitimate users.

**Resolution:** Used IAM Policy Simulator to test and debug policy behavior.

**Challenge 2:** KMS key access errors during S3 uploads.

**Resolution:** Reviewed and updated KMS key policy to grant required permissions.

**Challenge 3:** Objects being uploaded without encryption.

**Resolution:** Implemented strict bucket policy to deny non-encrypted uploads.

**Conclusion**

This project provided critical hands-on experience in implementing key cloud security features. From controlling access via IAM, to securing data storage with Amazon S3, and protecting sensitive data with AWS KMS encryption, each component contributed to building a secure cloud environment. This work is foundational for roles in cloud security and DevSecOps, and offers practical knowledge for ensuring security and compliance in any cloud-first organization.