**Domain 1 - Design Secure Architectures (30%)**

## **1. Secure Access to AWS Resources**

### ***IAM (Identity and Access Management)***

- Định danh và phân quyền, quản lý ai và cái gì có thể access như thế nào đến các resource trên AWS.

- Thiết lập quy tắc bảo vệ và phòng ngừa cho toàn tổ chức.

#### **1.1 Users**

* **Khái niệm:**
  + Là đại diện cho cá nhân hoặc ứng dụng cần truy cập tài nguyên AWS.
  + User mặc định khi tạo sẽ không có quyền gì. Cần cấp quyền cho User thông qua Policy hoặc Group
* **Đặc điểm:**
  + User có thể cấp phát access-key/secret-key để sử dụng CLI hoặc test SDK trong quá trình test code.
  + User có credentials:
    - Password: Sử dụng để đăng nhập AWS Management Console.
    - Access Keys: Dùng cho AWS CLI, SDK hoặc API.
  + User được gán permissions thông qua policies.
* **Sử dụng:**
  + Dùng để cấp quyền truy cập cho các ứng dụng hoặc người dùng cần truy cập tài nguyên AWS.

#### **1.2 Groups**

* **Khái niệm:**
  + Nhóm tập hợp nhiều IAM Users có chung mục đích.
  + Nhóm IAM là một cách để nhóm các người dùng lại với nhau và áp dụng các quyền truy cập giống nhau cho toàn bộ nhóm.
  + Một group không thể chứa group khác(lồng nhau)
  + Một user có thể không thuộc group nào hoặc thuộc nhiều group
  + Một group có thể không có user nào hoặc có nhiều users
* **Ứng dụng:**
  + Gán permissions tập trung bằng cách thêm policies vào Group thay vì từng User.
* **Sử dụng:**
  + Quản lý quyền truy cập của một nhóm người dùng thay vì quản lý từng người dùng riêng lẻ.
* **Ví dụ:**
  + Nhóm "Developers" với quyền truy cập EC2 và S3.

#### **1.3 Roles**

* **Khái niệm:**
  + Đại diện cho 1 quyền trên AWS(KHÔNG giống khái niệm role của 1 user)
  + Sử dụng khi muốn cấp quyền cho một thực thể tương tác với các resource khác trên AWS. Thường dùng gắn vào EC2, Lambda, Container
  + Gán quyền tạm thời cho dịch vụ AWS hoặc người dùng AWS khác.
  + Vai trò IAM là một bộ quyền truy cập có thể được gán cho một dịch vụ AWS hoặc một người dùng. Một vai trò có thể gán cho EC2 instances hoặc Lambda functions, giúp các dịch vụ này truy cập tài nguyên AWS mà không cần cung cấp các khóa API.
* **Ứng dụng:**
  + EC2 truy cập S3: Gán IAM Role thay vì Access Keys.
  + Lambda truy cập DynamoDB: Gán IAM Role với quyền truy cập.
* **Sử dụng:**
  + Gán cho các dịch vụ AWS hoặc các ứng dụng bên ngoài để cho phép chúng truy cập vào các tài nguyên AWS mà không cần dùng IAM users.

#### **1.4 Policies**

* **Khái niệm:**
  + Quy định ai-cái gì có thể hoặc không thể làm gì.
  + Tập hợp các quy tắc định nghĩa quyền truy cập tài nguyên AWS.
  + Một policy thường gồm nhiều statement quy định Allow/Deny hành động trên resource dựa trên condition.
* **Loại Policies:**
  + Managed Policies: Được tạo riêng và có thể gắn vào nhiều User/Group/Role.
    - AWS managed
    - User managed
  + Inline Policies: Được gán trực tiếp vào một User, Group, hoặc Role và không thể tái sử dụng ở Role/User/Group khác.
* **Cấu trúc:**
  + JSON chứa các thành phần chính:
    - Effect: "Allow" hoặc "Deny".
    - Action: Các hành động được phép (vd: s3:GetObject).
    - Resource: Tài nguyên áp dụng (vd: ARN của bucket S3).
    - Condition(có thể có hoặc không)
* **Sử dụng:**
  + Là các chính sách mặc định của AWS hoặc chính sách tùy chỉnh do người dùng tạo ra.
* **IAM Policy và Resource Policy:**
  + Một số resource như S3, SQS, KMS hỗ trợ policy ở cấp độ resource
  + Về cơ bản cấu trúc của resource policy tương tự IAM policy nhưng được gán cho một resource cụ thể.
* **Ví dụ: Policy này quy định đối tượng được gán policy này được phép thực hiện 2 hành động là StartInstance và StopInstance trên toàn bộ EC2 instance với điều kiện instance đó có 1 thẻ tag tên Enviroment và giá trị = Dev**

{

“Version”: “2012-10-17”,

“Statement”:[

{

“Effect”: “Allow”,

“Action”: [

“ec2:StartInstance”

“ec2:StopInstance”

],

“Resource”: “arn:aws:ec2:\*:\*:instance/\*”,

“Condition”: {

“StringEquals”:{

“aws:ResourceTag/Enviroment”: “Dev”

}

}

}

}

### ***Multi-Account Management và AWS Control Tower***

#### **2.1 Multi-Account Management**

* **Khái niệm**: Quản lý nhiều tài khoản
* **AWS Organizations:**
  + Quản lý nhiều tài khoản AWS từ một điểm trung tâm.
  + Cho phép phân quyền thông qua **Service Control Policies (SCPs)**.
* **Organizational Units (OU):**
  + Nhóm tài khoản theo mục đích.
  + VD: "Production", "Development".
* **Service Control Policies (SCPs):**
  + Gán vào OU để hạn chế quyền truy cập tài nguyên ở mức toàn bộ tài khoản trong OU.
  + VD: Chỉ cho phép tài khoản trong OU "Production" khởi chạy EC2 Instances.
* **Ví dụ:**
  + Giả sử có ba tài khoản AWS:
* Tài khoản 1: Phát triển (dev)
* Tài khoản 2: Kiểm tra (test)
* Tài khoản 3: Sản xuất (prod)
  + Có thể sử dụng AWS Organizations để tạo ra ba Organizational Units (OUs) tương ứng và áp dụng các chính sách bảo mật và chi phí cho từng OU.
* Dev OU: Áp dụng chính sách giúp kiểm soát chi phí, như việc giới hạn sử dụng các dịch vụ nhất định.
* Test OU: Cho phép kiểm tra các dịch vụ mới mà không ảnh hưởng đến môi trường sản xuất.
* Prod OU: Đảm bảo rằng môi trường sản xuất có các chính sách bảo mật nghiêm ngặt hơn.

#### **2.2 AWS Control Tower**

* **Khái niệm:**
  + Là dịch vụ được AWS cung cấp để giúp thiết lập và quản lý các tài khoản AWS trong môi trường đa tài khoản
  + Công cụ tự động thiết lập và quản lý môi trường multi-account dựa trên best practices.
* **Tính năng chính:**
  + **Guardrails:** Quy tắc bảo mật và compliance được thiết lập tự động.
  + **Account Factory:** Cung cấp quy trình tự động để tạo tài khoản AWS mới.
  + **Best practices**: AWS Control Tower áp dụng các best practices (thực hành tốt nhất) để quản lý tài khoản và môi trường AWS.
  + **Tự động hóa các quy trình**: Control Tower tự động hóa việc tạo các tài khoản AWS mới, áp dụng các chính sách bảo mật và tuân thủ, cũng như cấu hình các dịch vụ như AWS Organizations.
  + **Set-up landing zone**: AWS Control Tower giúp bạn thiết lập một "landing zone" (môi trường cơ sở), đây là một cấu hình tập trung để quản lý nhiều tài khoản AWS.

### ***Best Practices***

#### **3.1 Principle of Least Privilege**

* **Khái niệm**: Quyền giới hạn tối thiểu là một nguyên tắc bảo mật trong đó người dùng, nhóm người dùng, và hệ thống chỉ được cấp quyền truy cập vào tài nguyên mà họ cần thiết để thực hiện công việc của mình
* Chỉ cấp quyền cần thiết cho người dùng/dịch vụ để thực hiện công việc.
* Giảm thiểu rủi ro khi tài khoản bị lạm dụng.
* **Ví dụ:**
* Giả sử một nhân viên trong bộ phận tài chính chỉ cần quyền truy cập vào dịch vụ Amazon S3 để đọc dữ liệu báo cáo tài chính. Thay vì cấp quyền rộng rãi cho tài nguyên S3, chỉ cấp quyền **"s3"** cho thư mục báo cáo tài chính cụ thể, giúp bảo vệ các tài nguyên khác mà không liên quan đến công việc của họ.

#### **3.2 Multi-Factor Authentication (MFA)**

* MFA là phương thức xác thực yêu cầu người dùng phải cung cấp hai yếu tố hoặc nhiều yếu tố khi đăng nhập: một là cái người dùng biết (mật khẩu) và một là cái người dùng có (mã xác thực qua ứng dụng MFA).
* Bật MFA cho tài khoản root và IAM Users quan trọng.
* Hỗ trợ thiết bị MFA vật lý (YubiKey) hoặc ứng dụng MFA (Google Authenticator).

## **2. Secure Workloads and Applications**

### ***VPC Security***

VPC là một service cho phép người dùng tạo một mạng ảo và control toàn bộ network in/out của mạng đó.

#### **1.1 Security Groups (SG)**

* **Khái niệm:**
  + Thường dùng để gom nhóm các resource có chung network setting(in, out, protocol, port)
  + Security Groups là các tường lửa cấp độ instance (máy ảo EC2) trong AWS.
  + Chúng kiểm soát lưu lượng mạng vào và ra từ các instances.
  + Tường lửa ảo kiểm soát inbound/outbound traffic ở mức instance.
  + Security Groups là stateful
* **Đặc điểm:**
  + Mặc định:
    - Chặn tất cả inbound traffic.
    - Cho phép tất cả outbound traffic.
  + **Stateful:** Tự động cho phép traffic ngược chiều với kết nối đã được cho phép.

#### **1.2 Network Access Control Lists (NACLs)**

* **Khái niệm:**
  + Là công cụ bảo mật dùng để kiểm soát lưu lượng vào và ra khỏi các subnet trong một VPC trên AWS.
  + Kiểm soát inbound/outbound traffic ở mức subnet.
* **Đặc điểm:**
  + **Stateless(không trạng thái):** Yêu cầu quy tắc cho cả inbound và outbound. Mỗi yêu cầu mạng đến và đi đều phải được kiểm tra theo quy tắc riêng biệt mà không phụ thuộc vào trạng thái của yêu cầu trước đó
  + Hỗ trợ cả "allow" và "deny".
  + NACLs là stateless.
* **Cấu trúc:**
  + **Inbound Rules (Quy tắc vào)**: Các quy tắc xác định lưu lượng mạng được phép hoặc bị từ chối khi đi vào subnet.
  + **Outbound Rules (Quy tắc ra)**: Các quy tắc xác định lưu lượng mạng được phép hoặc bị từ chối khi đi ra từ subnet.
* **Ví dụ:**
* **Subnet 1**: Public subnet (có thể truy cập từ internet).
* **Subnet 2**: Private subnet (không thể truy cập từ internet trực tiếp).

#### **1.3 NAT Gateways**

* **Khái niệm:** là dịch vụ AWS cho phép các tài nguyên trong một **Private Subnet** trong VPC có thể truy cập Internet mà không cần phải có địa chỉ IP công cộng. NAT Gateways là một dịch vụ quản lý hoàn toàn của AWS giúp xử lý các yêu cầu từ các tài nguyên private đến Internet.
* Cho phép instances trong private subnet truy cập internet.
* Không hiển thị địa chỉ IP public của instance ra ngoài.
* Cho phép EC2 instances trong subnet riêng truy cập internet.

### ***AWS Security Services***

- AWS Identity Center: công cụ thay thế SSO của AWS, quản lý tập trung quyền access tới nhiều account và application trên AWS.

#### **2.1 AWS WAF (Web Application Firewall)**

* Bảo vệ ứng dụng web khỏi SQL Injection, Cross-Site Scripting (XSS).
* Để bảo vệ ứng dụng web khỏi các lỗ hổng phổ biến

#### **2.2 AWS Shield**

* **Standard:** Tự động bảo vệ chống DDoS cho tất cả tài khoản AWS.
* **Advanced:** Bảo vệ mở rộng với khả năng phân tích nâng cao.

#### **2.3 Amazon Cognito**

* Cung cấp authentication và authorization cho ứng dụng di động và web.
* Hỗ trợ Social Sign-In (Google, Facebook).

#### **2.4 AWS GuardDuty**

* Dịch vụ phát hiện mối đe dọa dựa trên hành vi bất thường từ CloudTrail, VPC Flow Logs.
* Phát hiện quyền truy cập trái phép và các hoạt động độc hại trong môi trường AWS của bạn

### ***Data Encryption***

#### **3.1 AWS KMS (Key Management Service):**

* Tạo và quản lý encryption keys(khóa mã hóa).
* Dùng để mã hóa dữ liệu trên S3, RDS, EBS.
* Quản lý và xoay vòng các khóa mã hóa một cách an toàn
* Mã hóa tại chỗ

#### **3.2 AWS Certificate Manager (ACM):**

* Quản lý SSL/TLS certificates cho các dịch vụ AWS.
* Tự động gia hạn chứng chỉ.

## **3. Data Security Controls**

### ***Encryption***

#### **1.1 Encryption at Rest**

* **Khái niệm:**
* Là việc mã hóa bảo vệ dữ liệu khi nó được lưu trữ trên các thiết bị, đĩa, hoặc hệ thống lưu trữ (ví dụ: Amazon S3, Amazon EBS, Amazon RDS).
* Dữ liệu lưu trữ được mã hóa bằng KMS.
* KMS để quản lý khóa mã hóa
* **Mục đích:**
  + Bảo vệ dữ liệu khỏi việc bị đánh cắp trong trường hợp thiết bị lưu trữ bị xâm phạm hoặc bị mất.
* **Các phương pháp mã hóa khi lưu trữ trên AWS:**
  + **Amazon S3**: Bạn có thể sử dụng mã hóa S3 (SSE) để tự động mã hóa các đối tượng khi được tải lên S3
  + **Amazon EBS (Elastic Block Store)**: Mã hóa EBS giúp mã hóa các ổ đĩa gắn liền với EC2 instance, giúp bảo vệ dữ liệu lưu trữ trên đĩa trong suốt vòng đời sử dụng của ổ đĩa.
  + **Amazon RDS (Relational Database Service)**: Mã hóa RDS giúp mã hóa dữ liệu tại rest cho cơ sở dữ liệu MySQL, MariaDB, Oracle, PostgreSQL và SQL Server

#### **1.2 Encryption in Transit**

* Là quá trình mã hóa dữ liệu khi nó di chuyển giữa các dịch vụ hoặc giữa người dùng và dịch vụ.
* Sử dụng TLS/SSL để mã hóa dữ liệu truyền qua mạng.
* AWS hỗ trợ mã hóa SSL/TLS qua các dịch vụ như CloudFront, API Gateway, và Elastic Load Balancer để bảo vệ dữ liệu khi truyền tải qua mạng.
* ACM hỗ trợ cài đặt TLS trên CloudFront, ALB.
* Bảo vệ dữ liệu khi lưu trữ trong các dịch vụ AWS

### ***Backup & Disaster Recovery***

#### **2.1 Backup**

* Sử dụng AWS Backup để quản lý backup định kỳ.
* Hỗ trợ nhiều dịch vụ: EBS, RDS, DynamoDB.
* RPO: khi xảy ra sự cố thì có bao nhiêu dât mất và không thể khôi phục
* RTO: mất bao lâu để khôi phục lại hệ thống
* Các cách backup:
* Backup EC2:
* Snapshot Data volumn
* Create AMI of whole instance
* Enable Life cycle policy
* Backup RDS:
* Dump data to external volumn
* Enable default backup
* Cretae custom backup using AWS backup
* Enable Backtracking feature
* Backup EFS
* Backup S3 bucket

#### **2.2 Disaster Recovery (DR)**

* **Khái niệm**: là chiến lược và quy trình để khôi phục các hệ thống và dữ liệu quan trọng sau khi xảy ra thảm họa hoặc sự cố lớn, chẳng hạn như mất điện, lỗi phần cứng, hoặc thảm họa thiên nhiên.
* AWS cung cấp nhiều công cụ DR như AWS Elastic Disaster Recovery, Route 53, và khả năng triển khai ứng dụng ở nhiều vùng khác nhau để giảm thiểu gián đoạn khi có thảm họa.
* **Chiến lược DR phổ biến:**
  + Đảm bảo ứng dụng luôn sẵn sàng và chịu lỗi
  + Backup and Restore: Lưu trữ và khôi phục khi cần.
  + Pilot Light: Duy trì các thành phần cốt lõi trong trạng thái standby.
  + Warm Standby: Có sẵn phiên bản với công suất giảm.
  + Multi-Site Active-Active: Triển khai đầy đủ ở nhiều vùng.

### ***Key Rotation và Secrets Manager***

#### **3.1 Key Rotation**

* Tự động xoay vòng (rotate) encryption keys với KMS.
* Nó đảm bảo rằng các khóa mã hóa cũ không còn được sử dụng để bảo vệ dữ liệu.
* AWS KMS

#### **3.2 AWS Secrets Manager**

* Lưu trữ và quản lý thông tin nhạy cảm (password, API keys).
* Hỗ trợ tự động xoay vòng secrets để đảm bảo bảo mật.

# **Key 1: Secure access to AWS resources**

## **1. IAM (Users, Groups, Roles, Policies, Best Practice)**

**IAM Users**

* **IAM Users** đại diện cho các cá nhân có quyền truy cập vào tài nguyên AWS.
* Mỗi user có thể có thông tin như username, mật khẩu và access keys.
* **Access keys** giúp IAM user truy cập vào AWS API và các dịch vụ AWS thông qua CLI, SDK hoặc AWS Console.
* Quyền của user được xác định thông qua các IAM policies gắn với user.
* **Best Practice**: Không sử dụng root account cho các tác vụ hàng ngày, tạo IAM users thay thế.

**IAM Groups**

* **IAM Groups** là tập hợp của các IAM Users giúp dễ dàng quản lý quyền.
* Các nhóm có thể được gán các **IAM Policies** để kiểm soát quyền truy cập của tất cả các thành viên.
* Giúp dễ dàng áp dụng quyền hạn cho nhiều user cùng lúc mà không phải chỉnh sửa từng user.
* Có thể chia nhóm theo chức năng hoặc vai trò (ví dụ: Dev, Admin).
* **Best Practice**: Không gán quyền trực tiếp cho users mà thay vào đó gán quyền thông qua groups.

**IAM Roles**

* **IAM Roles** là các quyền được gán tạm thời cho người dùng hoặc dịch vụ AWS (ví dụ: EC2, Lambda).
* Không liên kết với user cụ thể, mà thay vào đó là một tập hợp quyền có thể được sử dụng bởi các entity khác.
* **IAM Roles** giúp giảm thiểu việc sử dụng access keys dài hạn.
* Roles có thể được sử dụng trong các tài khoản AWS khác thông qua **Cross-Account Access**.
* **Best Practice**: Sử dụng IAM Roles thay vì gán quyền lâu dài trực tiếp cho IAM User.

**IAM Policies**

* **IAM Policies** là các tài liệu JSON xác định quyền truy cập vào các tài nguyên AWS.
* Chính sách có thể có các quyền như ReadOnly, Write, hoặc FullAccess.
* IAM policies có thể được áp dụng cho users, groups hoặc roles.
* Các **Managed Policies** (do AWS cung cấp) và **Customer Managed Policies** (do người dùng tạo) có thể được sử dụng.
* **Best Practice**: Áp dụng nguyên tắc "least privilege", chỉ cấp quyền cần thiết nhất.

**IAM Best Practices**

* **Least Privilege**: Cấp quyền tối thiểu cho user hoặc role để hoàn thành công việc.
* **Enable MFA**: Sử dụng Multi-Factor Authentication (MFA) để bảo vệ tài khoản.
* **Rotate Keys Regularly**: Thường xuyên xoay vòng access keys.
* **Monitor with CloudTrail**: Theo dõi và ghi lại mọi hành động của user trong tài khoản AWS bằng AWS CloudTrail.
* **Use Groups for Permissions**: Gán quyền cho group thay vì gán trực tiếp cho user.

## **2. Multi-account management, AWS Control Tower**

**AWS Organizations**

* **AWS Organizations** giúp quản lý và kiểm soát các tài khoản AWS từ một tài khoản quản lý trung tâm.
* **Service Control Policies (SCP)** được sử dụng để kiểm soát quyền hạn trên các tài khoản con.
* **Consolidated Billing** giúp giảm chi phí và quản lý chi phí giữa các tài khoản.
* **AWS Organizations** giúp dễ dàng mở rộng quy mô hạ tầng khi thêm tài khoản mới mà không cần cấu hình lại.
* **Best Practice**: Tổ chức tài khoản theo mô hình riêng biệt (ví dụ: một tài khoản cho production, một cho development).

**Service Control Policies (SCP)**

* **SCPs** giúp hạn chế quyền truy cập và quyền hoạt động của các tài khoản trong AWS Organization.
* SCPs được áp dụng trên các OUs (Organizational Units) và các tài khoản.
* Giúp bảo vệ các tài khoản AWS khỏi việc thực hiện các hành động không mong muốn.
* **SCPs** không cấp quyền, mà chỉ giới hạn quyền đã được cấp qua IAM.
* **Best Practice**: Áp dụng SCPs để tăng cường bảo mật cho tài khoản và dịch vụ AWS.

**AWS Control Tower**

* **AWS Control Tower** cung cấp dịch vụ quản lý đa tài khoản, thiết lập các best practice tự động cho môi trường AWS.
* Tạo và quản lý một landing zone, bao gồm các tài khoản và thiết lập bảo mật.
* **Guardrails** tự động áp dụng để đảm bảo các thực hành bảo mật và quản lý tuân thủ.
* **AWS Control Tower** tích hợp với AWS Organizations để quản lý tài khoản.
* **Best Practice**: Sử dụng AWS Control Tower để đơn giản hóa việc quản lý tài khoản và tuân thủ chính sách bảo mật.

**Cross-Account Access**

* **Cross-Account Access** cho phép một tài khoản AWS truy cập tài nguyên của một tài khoản AWS khác.
* Được thực hiện thông qua **IAM Roles** và chính sách xác thực.
* **Cross-Account Access** giúp giảm thiểu việc phải duy trì các access keys.
* Có thể sử dụng cho các ứng dụng và dịch vụ AWS cần truy cập tài nguyên giữa các tài khoản khác nhau.
* **Best Practice**: Tạo IAM Roles với chính sách quyền phù hợp và sử dụng khi cần thiết.

**Consolidated Billing**

* **Consolidated Billing** giúp tổng hợp chi phí từ nhiều tài khoản AWS dưới một tài khoản chính.
* Giúp tiết kiệm chi phí thông qua giảm giá volume (mức giá giảm cho khối lượng sử dụng lớn).
* **Linked Accounts** cho phép theo dõi chi phí và sử dụng riêng biệt cho từng tài khoản con.
* Tối ưu hóa chi phí bằng cách phân loại chi phí cho từng nhóm dự án hoặc đội ngũ.
* **Best Practice**: Sử dụng **Consolidated Billing** để dễ dàng quản lý chi phí và tối ưu hóa.

## **3. Best practices: Least privilege, MFA**

**Least Privilege Principle**

* **Least Privilege**: Cấp quyền tối thiểu cho user hoặc dịch vụ cần thiết để thực hiện công việc của mình.
* Giảm thiểu khả năng tấn công nếu user bị xâm nhập.
* Giảm thiểu rủi ro bằng cách giảm quyền cho những người không cần thiết.
* **Least Privilege** là một trong những nguyên tắc quan trọng nhất trong bảo mật AWS.
* **Best Practice**: Sử dụng chính sách phân quyền chi tiết và cụ thể thay vì gán quyền quá rộng.

**Enable MFA (Multi-Factor Authentication)**

* **MFA** cung cấp một lớp bảo mật bổ sung ngoài mật khẩu, yêu cầu người dùng cung cấp mã xác thực từ một thiết bị thứ hai.
* Cần bật MFA cho tài khoản root và các IAM users có quyền cao.
* AWS hỗ trợ **Virtual MFA** (trên điện thoại) và **Hardware MFA** (thiết bị phần cứng).
* **MFA** làm tăng bảo mật đối với tài khoản AWS của bạn, đặc biệt là khi sử dụng các quyền nhạy cảm.
* **Best Practice**: Bật MFA cho mọi tài khoản quan trọng và yêu cầu cho các IAM users có quyền truy cập cao.

**Password Policies**

* **Password Policies** giúp bảo vệ tài khoản khỏi việc sử dụng mật khẩu yếu hoặc dễ đoán.
* Có thể yêu cầu mật khẩu phải chứa chữ hoa, chữ thường, số và ký tự đặc biệt.
* Cài đặt yêu cầu thay đổi mật khẩu định kỳ và không cho phép sử dụng mật khẩu cũ.
* **IAM Password Policies** cho phép bạn quy định độ dài mật khẩu tối thiểu và các yêu cầu bảo mật khác.
* **Best Practice**: Sử dụng mật khẩu phức tạp và thay đổi thường xuyên.

**Access Key Rotation**

* **Access Key Rotation** giúp lộ bảo vệ tài khoản khỏi việc khóa truy cập dài hạn.
* Thực hiện xoay vòng khóa định kỳ, đặc biệt là với các IAM users.
* **Access Key Rotation** có thể được tự động hóa bằng các script hoặc thông qua các dịch vụ như AWS Secrets Manager.
* Đảm bảo rằng không có khóa truy cập lâu dài hoặc không sử dụng.
* **Best Practice**: Thực hiện **access key rotation** định kỳ và quản lý khóa qua AWS Secrets Manager.

**Monitor Access Logs**

* **CloudTrail** theo dõi và ghi lại tất cả các hành động trong tài khoản AWS.
* Giúp phát hiện các hành động không bình thường và điều tra các sự cố bảo mật.
* **CloudWatch Logs** có thể được sử dụng để cảnh báo và gửi thông báo khi phát hiện hành động lạ.
* **Best Practice**: Thiết lập CloudTrail cho tất cả các tài khoản AWS và sử dụng CloudWatch để giám sát.

# **Key 2: Secure workloads and applications**

## **1. VPC Security (Security Groups, NACLs, NAT Gateways)**

**Security Groups**

* **Security Groups** là tường lửa ảo cho các instance EC2, giúp kiểm soát lưu lượng vào và ra của instance.
* Giới hạn truy cập đến EC2 và từ EC2 ra bên ngoài.
* Các **Security Groups** hoạt động ở mức **stateful**, tức là nếu lưu lượng đi vào được phép, thì lưu lượng phản hồi sẽ tự động được phép mà không cần cấu hình riêng biệt.
* Có thể gán nhiều **Security Groups** cho một instance, và mỗi group có các quy tắc riêng.
* Quy tắc của **Security Group** có thể chỉ định lưu lượng được phép đi vào hoặc đi ra, dựa trên địa chỉ IP, cổng và giao thức.
* **Best Practice**: Cấu hình các **Security Group** theo mô hình **Least Privilege**, chỉ mở các cổng cần thiết cho từng ứng dụng.

**Network Access Control Lists (NACLs)**

* **NACLs** là tường lửa cấp subnet, giúp kiểm soát lưu lượng vào và ra của subnet trong VPC.
* **NACLs** là **stateless**, tức là mỗi lưu lượng phải được kiểm tra hai lần: một khi vào và một khi ra.
* Các **NACLs** có thể có các quy tắc "ALLOW" hoặc "DENY" và có thể áp dụng cho cả lưu lượng inbound và outbound.
* **NACLs** hỗ trợ phân quyền theo các dải IP, trong khi **Security Groups** chủ yếu sử dụng IP hoặc nhóm bảo mật.
* **Best Practice**: Sử dụng **NACLs** để kiểm soát và lọc lưu lượng ở cấp subnet, và dùng **Security Groups** để quản lý lưu lượng cho các instance EC2.

**NAT Gateways**

* **NAT Gateways** giúp các instance trong một subnet private truy cập Internet để tải xuống các bản cập nhật hoặc truy xuất dữ liệu, nhưng vẫn giữ chúng cách ly khỏi Internet.
* **NAT Gateways** cho phép outbound traffic từ các instance private subnet ra Internet mà không cần expose trực tiếp các instance đó.
* **NAT Gateways** hoạt động trong một VPC và có thể chịu tải cao, giúp duy trì sự khả dụng và hiệu suất cho các ứng dụng.
* Một **NAT Gateway** có thể có IP công cộng để các instance trong subnet private có thể truy cập tài nguyên bên ngoài VPC.
* **Best Practice**: Sử dụng **NAT Gateways** cho các instance trong private subnet và theo dõi lưu lượng qua **CloudWatch**.

**VPC Peering**

* **VPC Peering** cho phép kết nối giữa hai VPC khác nhau, giúp các tài nguyên trong mỗi VPC có thể giao tiếp với nhau.
* **VPC Peering** có thể xảy ra trong cùng một tài khoản AWS hoặc giữa các tài khoản AWS khác nhau.
* **VPC Peering** hỗ trợ **Private IP Communication**, giúp tiết kiệm băng thông và tối ưu hóa chi phí.
* **VPC Peering** không hỗ trợ transitive peering (tức là không thể truyền tải dữ liệu qua một VPC khác).
* **Best Practice**: Giới hạn **VPC Peering** với các VPC đáng tin cậy và cấu hình **route tables** cho phù hợp.

**VPC Flow Logs**

* **VPC Flow Logs** giúp theo dõi và ghi lại tất cả lưu lượng mạng trong VPC, bao gồm lưu lượng đi vào và đi ra từ các interface.
* **VPC Flow Logs** có thể được ghi vào **CloudWatch Logs** hoặc **S3** để dễ dàng phân tích.
* Giúp phát hiện các hành động bất thường và phục vụ cho việc điều tra sự cố bảo mật.
* **Best Practice**: Bật **VPC Flow Logs** cho tất cả các VPC và cấu hình **CloudWatch** để theo dõi các sự kiện và hành động đáng ngờ.

## **2. AWS Security services: AWS WAF, Shield, Cognito, GuardDuty**

**AWS WAF (Web Application Firewall)**

* **AWS WAF** là dịch vụ bảo vệ ứng dụng web khỏi các tấn công phổ biến như SQL injection, cross-site scripting (XSS), và các cuộc tấn công DDoS.
* **AWS WAF** cho phép tạo các quy tắc tùy chỉnh để bảo vệ ứng dụng và API từ các lưu lượng không mong muốn.
* Các quy tắc trong **AWS WAF** có thể kiểm tra các đặc điểm của yêu cầu HTTP như địa chỉ IP, chuỗi User-Agent, và URI.
* **AWS WAF** có thể được triển khai trên các dịch vụ như **CloudFront**, **ALB** và **API Gateway**.
* **Best Practice**: Sử dụng **AWS WAF** để bảo vệ ứng dụng khỏi các tấn công thường gặp và cấu hình các quy tắc phù hợp.

**AWS Shield**

* **AWS Shield** bảo vệ các tài nguyên AWS khỏi các cuộc tấn công DDoS (Distributed Denial of Service).
* **AWS Shield** cung cấp hai cấp độ bảo vệ: **Shield Standard** (bảo vệ tự động cho tất cả các dịch vụ AWS như EC2, ELB, CloudFront) và **Shield Advanced** (bảo vệ cấp cao hơn cho các ứng dụng có yêu cầu cao về bảo mật).
* **Shield Advanced** cung cấp các tính năng bảo vệ bổ sung như **DDoS Cost Protection** và **24x7 access to the AWS DDoS Response Team (DRT)**.
* **Best Practice**: Sử dụng **Shield Advanced** cho các tài nguyên có tính quan trọng cao và dễ bị tấn công DDoS.

**Amazon Cognito**

* **Amazon Cognito** giúp xác thực và quản lý người dùng trong các ứng dụng web và mobile.
* **Cognito** hỗ trợ **MFA** và các phương thức xác thực linh hoạt như **social login** (Facebook, Google) và **SAML-based identity providers**.
* **Cognito** cung cấp khả năng đồng bộ hóa dữ liệu người dùng giữa các thiết bị và ứng dụng khác nhau.
* **Best Practice**: Sử dụng **Amazon Cognito** để quản lý xác thực người dùng và tăng cường bảo mật với MFA.

**AWS GuardDuty**

* **AWS GuardDuty** là dịch vụ phát hiện mối đe dọa bảo mật, giúp phát hiện hành động đáng ngờ trong tài khoản AWS và môi trường VPC.
* **GuardDuty** sử dụng các nguồn dữ liệu như **VPC Flow Logs**, **CloudTrail**, và **DNS logs** để phân tích và phát hiện các hành động bất thường.
* **Best Practice**: Kích hoạt **GuardDuty** cho mọi tài khoản và môi trường VPC để theo dõi các hoạt động bất thường và giảm thiểu rủi ro.

**AWS Macie**

* **AWS Macie** là dịch vụ bảo mật dữ liệu sử dụng machine learning để phát hiện và bảo vệ dữ liệu nhạy cảm, đặc biệt là dữ liệu cá nhân và thông tin thanh toán.
* **Macie** giúp tự động phát hiện các dữ liệu nhạy cảm và cung cấp báo cáo chi tiết để hỗ trợ tuân thủ.
* **Best Practice**: Sử dụng **AWS Macie** để bảo vệ dữ liệu nhạy cảm và giúp đảm bảo tuân thủ các quy định bảo mật.

## **3. Data Encryption (KMS, ACM)**

**AWS Key Management Service (KMS)**

* **AWS KMS** giúp tạo và quản lý các keys để mã hóa dữ liệu trong các dịch vụ AWS.
* **KMS** cung cấp **Customer Managed Keys (CMKs)** để bạn có thể kiểm soát quyền truy cập vào các keys.
* **KMS** hỗ trợ mã hóa dữ liệu khi lưu trữ (at rest) và khi truyền tải (in transit).
* **Best Practice**: Tạo các **Customer Managed Keys (CMKs)** và bật logging để theo dõi các hoạt động liên quan đến keys.

**AWS Certificate Manager (ACM)**

* **ACM** giúp quản lý SSL/TLS certificates cho các dịch vụ AWS như Elastic Load Balancer (ELB) và CloudFront.
* **ACM** hỗ trợ tự động gia hạn chứng chỉ, giúp giảm thiểu các rủi ro liên quan đến việc hết hạn chứng chỉ.
* **ACM** cũng cung cấp chứng chỉ miễn phí cho các dịch vụ web.
* **Best Practice**: Sử dụng **ACM** để quản lý chứng chỉ SSL/TLS và bảo mật giao tiếp qua HTTPs.

# **Key 3: Data security controls**

## **1. Encryption (at rest và in transit)**

**Encryption at rest**

* **Encryption at rest** bảo vệ dữ liệu khi nó được lưu trữ trên các thiết bị (ví dụ: đĩa cứng, ổ lưu trữ, cơ sở dữ liệu) để đảm bảo rằng dữ liệu không thể đọc được nếu bị truy cập trái phép.
* **AWS KMS (Key Management Service)** là dịch vụ chính để quản lý các keys dùng cho mã hóa dữ liệu tại rest.
* **EBS**, **S3**, **RDS**, và **Redshift** đều hỗ trợ mã hóa dữ liệu tại rest sử dụng các khóa được quản lý qua **KMS**.
* **S3** hỗ trợ mã hóa dữ liệu tại rest bằng cách sử dụng **SSE-S3**, **SSE-KMS**, hoặc **SSE-C**.
* **Best Practice**: Bật mã hóa tại rest cho tất cả các dịch vụ lưu trữ, sử dụng **Customer Managed Keys (CMK)** trong **KMS** để quản lý khóa mã hóa.

**Encryption in transit**

* **Encryption in transit** bảo vệ dữ liệu khi nó đang di chuyển qua mạng, giúp tránh rủi ro bị rò rỉ thông tin khi truyền tải.
* **TLS/SSL** là phương thức chính để mã hóa dữ liệu trong transit giữa các ứng dụng và người dùng, bảo vệ dữ liệu khỏi các cuộc tấn công man-in-the-middle (MITM).
* Các dịch vụ AWS như **S3**, **EC2**, và **CloudFront** hỗ trợ mã hóa dữ liệu trong transit bằng **TLS/SSL**.
* **AWS Certificate Manager (ACM)** cung cấp chứng chỉ SSL/TLS để mã hóa giao tiếp qua Internet và các giao thức mạng nội bộ.
* **Best Practice**: Sử dụng **TLS/SSL** để mã hóa dữ liệu trong transit và quản lý chứng chỉ qua **ACM**.

**AWS Encryption SDK**

* **AWS Encryption SDK** là thư viện mã hóa mạnh mẽ cho các ứng dụng, giúp tích hợp mã hóa vào ứng dụng của bạn.
* **Encryption SDK** hỗ trợ các thuật toán mã hóa mạnh mẽ như **AES-GCM** và giúp xử lý việc mã hóa dữ liệu dễ dàng.
* SDK này cũng hỗ trợ tính năng quản lý khóa với **KMS** để mã hóa và giải mã dữ liệu một cách tự động.
* **Best Practice**: Sử dụng **AWS Encryption SDK** để mã hóa dữ liệu trong ứng dụng và đảm bảo tính bảo mật.

**Managed keys vs Customer-managed keys**

* **Managed keys** (CMK được quản lý bởi AWS) có thể dễ dàng sử dụng nhưng bạn không thể kiểm soát các cài đặt về quyền truy cập hay vòng đời của chúng.
* **Customer-managed keys** cho phép bạn kiểm soát các quyền truy cập vào khóa mã hóa và có thể cấu hình vòng đời và các chính sách liên quan đến khóa.
* **Best Practice**: Sử dụng **Customer-managed keys** khi bạn cần sự linh hoạt và kiểm soát hoàn toàn về cách thức mã hóa và quản lý dữ liệu của mình.

**Data Encryption Best Practices**

* **Best Practice**: Đảm bảo mã hóa tất cả dữ liệu quan trọng tại rest và trong transit, đặc biệt là đối với dữ liệu nhạy cảm như thông tin cá nhân và tài chính.
* **Best Practice**: Bảo vệ khóa mã hóa với **IAM** và **KMS** để đảm bảo chỉ những người có quyền mới có thể truy cập và quản lý khóa.

## **2. Backup & Disaster Recovery**

**Backup**

* **Backup** là quá trình sao lưu dữ liệu quan trọng để tránh mất mát dữ liệu trong trường hợp xảy ra sự cố hoặc mất dữ liệu.
* **AWS Backup** cung cấp giải pháp quản lý sao lưu tập trung, giúp bạn sao lưu các tài nguyên AWS như **EC2**, **RDS**, **EFS**, và **S3**.
* **Amazon S3** có tính năng **Versioning**, cho phép bạn giữ nhiều phiên bản của một đối tượng và khôi phục lại phiên bản trước đó khi cần.
* **Best Practice**: Sao lưu dữ liệu định kỳ và sử dụng **AWS Backup** để tự động hóa quy trình sao lưu.

**Disaster Recovery (DR)**

* **Disaster Recovery** là chiến lược phục hồi hệ thống và dữ liệu sau khi xảy ra sự cố, giúp bảo vệ tổ chức khỏi sự gián đoạn dịch vụ kéo dài.
* Các chiến lược DR phổ biến bao gồm **Backup and Restore**, **Pilot Light**, **Warm Standby**, và **Multi-site**.
* **AWS CloudEndure Disaster Recovery** cung cấp giải pháp DR nâng cao cho các ứng dụng và cơ sở hạ tầng.
* **Best Practice**: Lập kế hoạch DR rõ ràng và sử dụng các dịch vụ AWS để đảm bảo khả năng phục hồi nhanh chóng và bảo vệ khỏi các sự cố.

**RTO và RPO**

* **Recovery Time Objective (RTO)**: Thời gian tối đa mà hệ thống có thể chịu đựng gián đoạn sau sự cố mà không ảnh hưởng đến doanh nghiệp.
* **Recovery Point Objective (RPO)**: Lượng dữ liệu tối đa có thể mất đi mà không gây thiệt hại nghiêm trọng.
* **Best Practice**: Định rõ **RTO** và **RPO** cho các dịch vụ và ứng dụng quan trọng, đảm bảo các giải pháp DR phù hợp với các mục tiêu này.

**Cross-region backups**

* **Cross-region backups** cho phép sao lưu dữ liệu từ một region AWS sang region khác để đảm bảo tính sẵn sàng và phục hồi khi xảy ra sự cố lớn.
* **Amazon S3 Cross-Region Replication (CRR)** giúp sao lưu và sao chép đối tượng S3 tự động giữa các vùng khác nhau.
* **Best Practice**: Sử dụng **Cross-region backups** để bảo vệ dữ liệu khỏi các sự cố xảy ra ở mức vùng (region).

**Disaster Recovery Testing**

* **Best Practice**: Thực hiện kiểm tra DR định kỳ để đảm bảo rằng kế hoạch DR có hiệu quả và các quy trình khôi phục dữ liệu hoạt động đúng cách.

## **3. Key rotation, Secrets Manager**

**Key Rotation**

* **Key Rotation** là quá trình thay đổi định kỳ các khóa mã hóa để giảm thiểu rủi ro nếu khóa bị lộ hoặc tấn công.
* **AWS KMS** hỗ trợ tự động xoay vòng khóa (key rotation) cho các **Customer Managed Keys (CMKs)** mỗi năm.
* **Best Practice**: Thiết lập chính sách tự động xoay vòng khóa trong **KMS** và **IAM** để duy trì bảo mật liên tục cho các tài nguyên và dữ liệu.

**Secrets Manager**

* **AWS Secrets Manager** giúp quản lý và bảo vệ các bí mật như mật khẩu, API keys, và thông tin đăng nhập cơ sở dữ liệu.
* **Secrets Manager** cung cấp khả năng tự động xoay vòng, kiểm soát truy cập và ghi lại tất cả các hành động liên quan đến bí mật.
* **Best Practice**: Sử dụng **Secrets Manager** để lưu trữ và quản lý các bí mật thay vì hard-code chúng vào mã nguồn ứng dụng.

**IAM Role with Secrets Manager**

* **Secrets Manager** có thể tích hợp với **IAM roles**, giúp các ứng dụng và dịch vụ AWS truy xuất các bí mật một cách an toàn và có kiểm soát.
* **Best Practice**: Gán quyền truy cập vào **Secrets Manager** qua **IAM roles** thay vì chia sẻ thông tin bí mật trực tiếp trong mã nguồn.

**Encryption for Secrets**

* Các bí mật lưu trữ trong **Secrets Manager** và **S3** có thể được mã hóa bằng **KMS** để bảo vệ khỏi truy cập trái phép.
* **Best Practice**: Bảo vệ tất cả các bí mật trong **Secrets Manager** bằng mã hóa và chỉ cấp quyền truy cập cho các dịch vụ cần thiết.

**Audit and Monitoring**

* **Best Practice**: Sử dụng **CloudTrail** và **CloudWatch Logs** để theo dõi và ghi lại các hành động liên quan đến các bí mật và khóa mã hóa.
* **Best Practice**: Đảm bảo rằng tất cả các hành động truy cập bí mật đều được ghi lại và có thể được phân tích trong trường hợp sự cố.