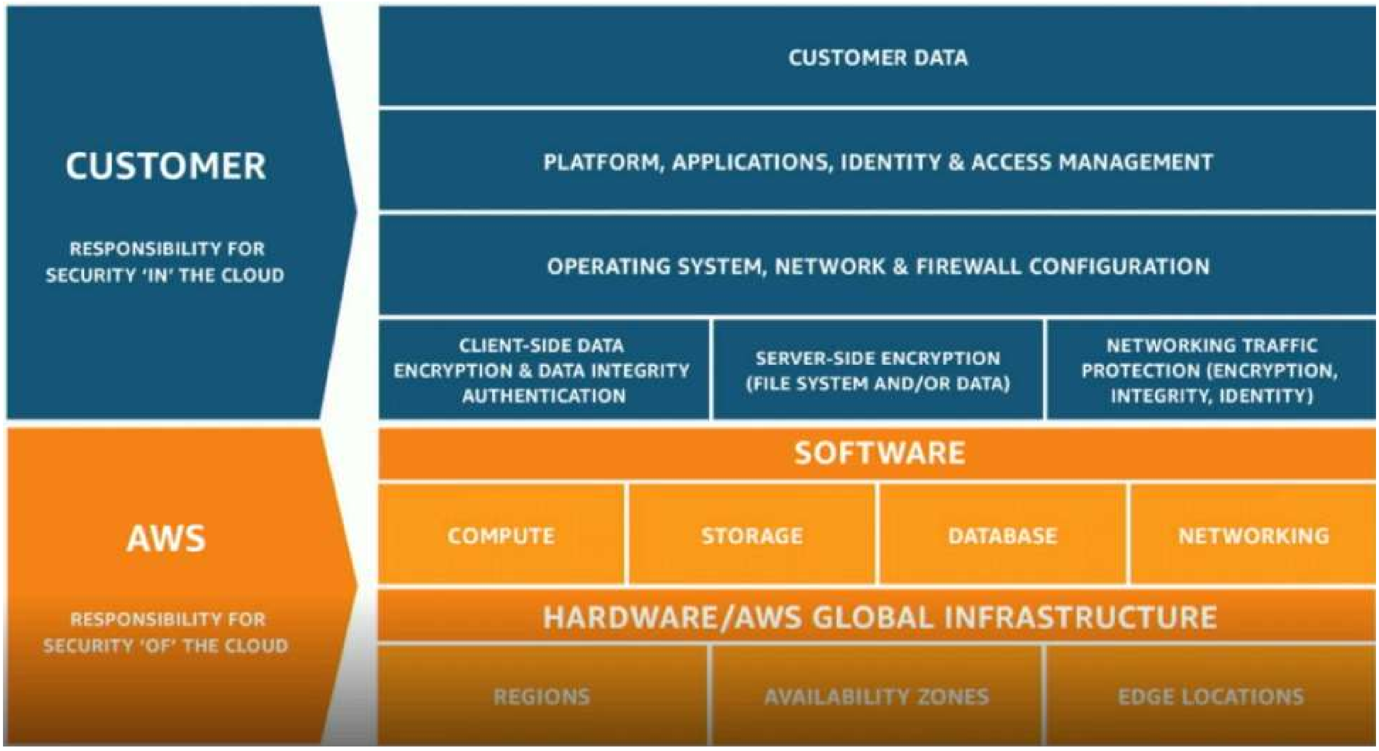
AWS Module 4 - AWS Cloud Security

By lemasyma

Posted Feb 9, 2021 • Updated Oct 3, 2021 • 12 min read • 1 views

Lien de la note Hackmd

Section 1: AWS shared responsibility model


CUSTOMER
RESPONSIBILITY FOR SECURITY 'IN' THE CLOUD
CUSTOMER DATA
PLATFORM, APPLICATIONS, IDENTITY & ACCESS MANAGEMENT
OPERATING SYSTEM, NETWORK & FIREWALL CONFIGURATION
CLIENT-SIDE DATA ENCRYPTION & DATA INTEGRITY AUTHENTICATION
SERVER-SIDE ENCRYPTION (FILE SYSTEM AND/OR DATA)
NETWORKING TRAFFIC PROTECTION (ENCRYPTION, INTEGRITY, IDENTITY)
AWS
RESPONSIBILITY FOR SECURITY 'OF' THE CLOUD
SOFTWARE
COMPUTE
STORAGE
DATABASE
NETWORKING
HARDWARE/AWS GLOBAL INFRASTRUCTURE
REGIONS
AVAILABILITY ZONES
EDGE LOCATIONS

AWS:

- Security of the cloud
- Physical facilities and system
    - Hardware, software for running AWS services

Customers:

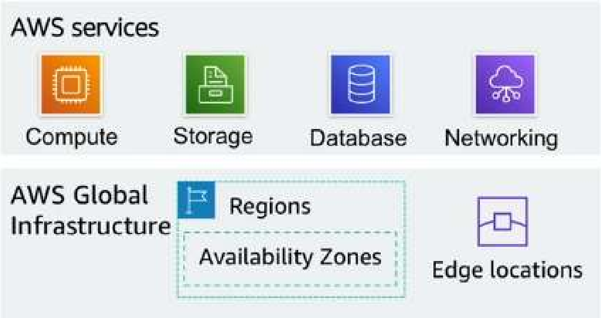
- Securing app and datasets in the cloud
    - Data encryption in transit from one systeme to another
- Use Amazon Tools
- Network configured for security
- Firewall configuration and security of OS

AWS responsability: Security *of* the cloud

AWS responsibilites:

- Physical security of data centers

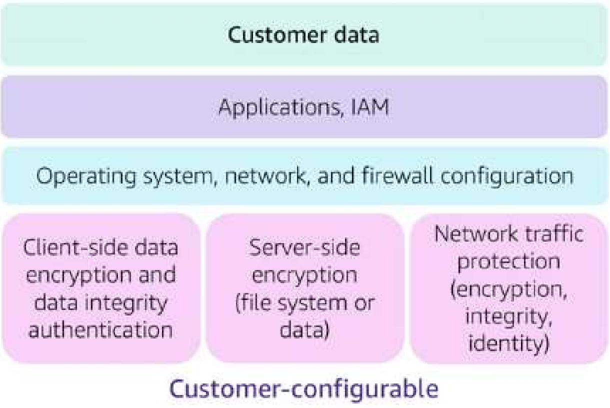
- Hardware and software infrastructure
  - Storage decommissioning, host operating, system (OS) access logging, and auditing
- Network infrastructure
  - Intrusion detection
- Virtualization infrastructure
  - Instance isolation
  - Between customers workloads



Customer responsibility: Security *in* the cloud

Customer responsabilities:

- Amazon Elastic Compute Cloud (Amazon EC2) instance *operating system*
  - Including patching, maintenance
- *Applications*
  - Passwords, role-based access, etc.
- *Security group* configuration
- OS or host-based *firewalls*
  - Including intrusion detection or prevention systems
- *Network* configurations
- Account management
  - Login and permission settings for each user

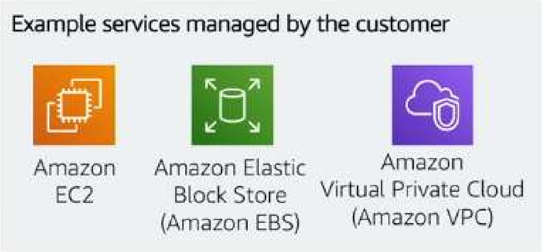

Customer data
Applications, IAM
Operating system, network, and firewall configuration
Client-side data encryption and data integrity authentication
Server-side encryption (file system or data)
Network traffic protection (encryption, integrity, identity)
Customer-configurable

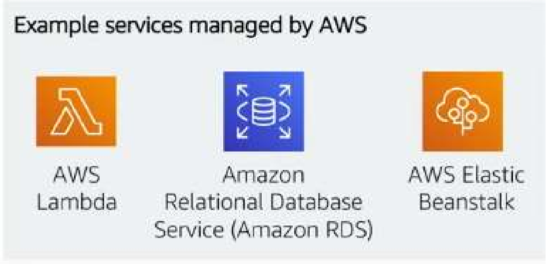
Infrastructure as a service (IaaS)

- Customer has more flexibility over configuring networking and storage settings
- Customer is responsible for managing more aspects of the security
- Customer configures the access controls

Platform as a service (PaaS)

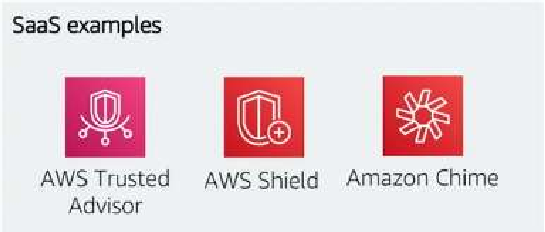
- Customer does not need to manage the underlying infrastructure
- AWS handles the operating system, database patching, firewall configuration, and disaster recovery
- Customer can focus on managing code or data


Example services managed by the customer
Amazon EC2
Amazon Elastic Block Store (Amazon EBS)
Amazon Virtual Private Cloud (Amazon VPC)


Example services managed by AWS
AWS Lambda
Amazon Relational Database Service (Amazon RDS)
AWS Elastic Beanstalk

Software as a service (SaaS)

- Sofware is centrally hosted
- Licensed on a subscription model or pay-as-you-go basis
- Services are typically accessed via web browser, mobile app, or application programming interface (API)
- Customers do not need to manage the infrastructure that supports the service



Section 2: AWS Identity and Access Management (IAM)

- Use *IAM* to manage access to *AWS resources*
  - A resource is an entity in an AWS account that you can work with
  - Example resources; An Amazon EC2 instance or an Amazon S3 bucket
- *Example*: control who can terminate Amazon EC2 instances

- *Who* can access the resours
- Which resources can be accessed and what can the user do to the resource
- *How* resources can be accessed

- IAM is a no-cost account feature

IAM: Essential components

- IAM user
  - A *person* or *application* that can authenticate with a AWS account
- IAM group
  - A *collection of IAM users* that are granted identical authorization
- IAM policy
  - The document that defines *which resources can be accessed* and the *level of access* to each resource
  - Created independently than users and groups
- IAM role
  - Usefule mechanism to grant a set of permissions for making AWS service requests
  - Grant temporary access to a service
  - Similar to sudo in Linux

Authenticate as an IAM user to gain access

When you define an **IAM user**, you select what *types of access* the user is permitted to use.

Can use either programmatic access, *AWS Management Console access*, or both.

Programmatic access

- Authenticate using:
  - Acces key ID
  - Secret access key
- Provides AWS CLI and AWS SDK access

AWS Management Console access

- Autheticate using:
  - 12-digit Account ID or *alias*
  - IAM user name
  - IAM password
- If enabled **multi-factor authentificatuin (MFA)** prompts for an authentification code

IAM MFA

- MFA provides increased security

Authorization: What actions are permitted

After the user or application is connected to the AWS account, what are they allowed to do ?

IAM: Authorization

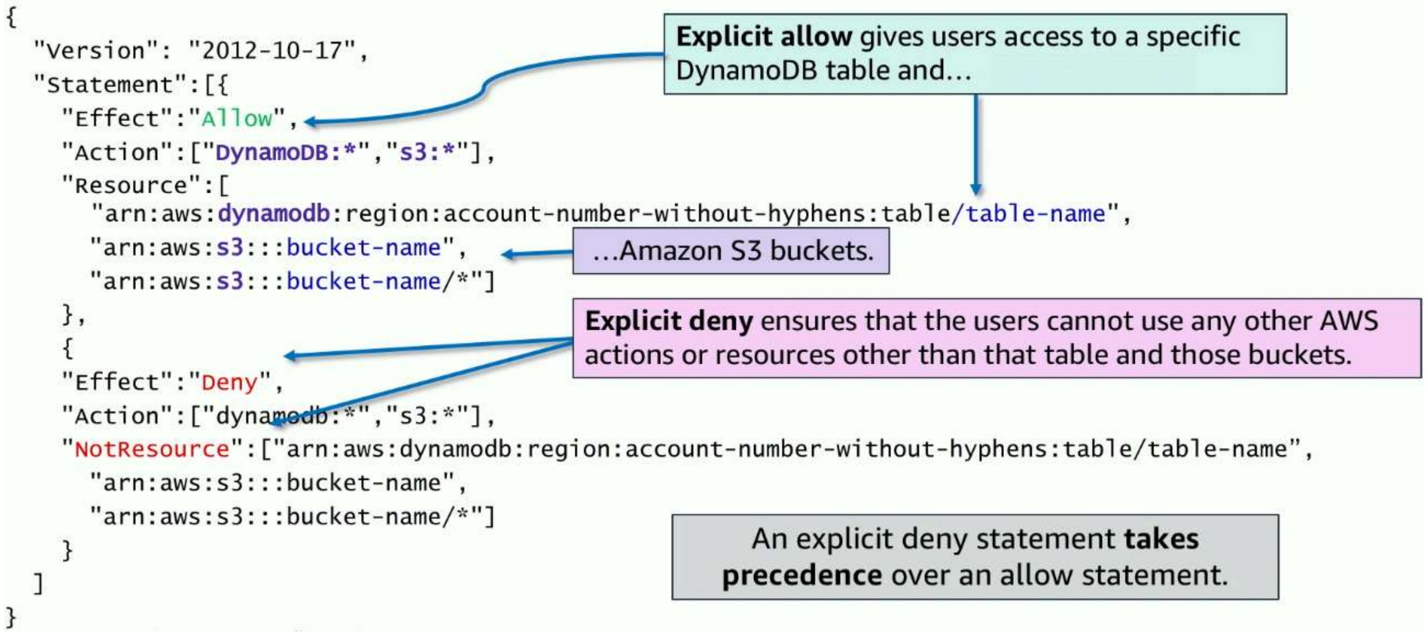
- Assign permissions by creating an IAM policy
- Permissions determine **which resources and operations** are allowed:
    - All permissions are implicitly denied by default
    - Is something is explicitly denied, it is never allowed

> **Best practice:** Follow the principle of *least privilege*.

Note: the scope of IAM service configurations is *global*. Settings apply accross all AWS Regions

IAM Policies

- An IAM policy is a document in JSON that defines permissions
    - Enables fine-grained access control
- 2 types of policies
    1. *identity-base*
    2. *resource-based*
        1. **Identity-based** policies * Attach a policy to any IAM entity
            - An IAM user, an IAM group or an IAM role * Policies specify;
            - Actions that *may* be performed by the entity
            - Actions that *may not* be performed by the entity * A single *policy* can be attached to multiple *entities* * A single *entity* can have multiple *policies* attached to it
        2. **Resource-based** policies * Attached to a resource (such as an S3 bucket)



```
{
  "Version": "2012-10-17",
  "Statement":[{
    "Effect":"Allow",
    "Action":["DynamoDB:*","s3:*"],
    "Resource":[
      "arn:aws:dynamodb:region:account-number-without-hyphens:table/table-name",
      "arn:aws:s3:::bucket-name",
      "arn:aws:s3:::bucket-name/*"]
    },
    {
    "Effect":"Deny",
    "Action":["dynamodb:*","s3:*"],
    "NotResource":["arn:aws:dynamodb:region:account-number-without-hyphens:table/table-name",
      "arn:aws:s3:::bucket-name",
      "arn:aws:s3:::bucket-name/*"]
    }
  ]
}
```

Explicit allow gives users access to a specific DynamoDB table and…

…Amazon S3 buckets.

Explicit deny ensures that the users cannot use any other AWS actions or resources other than that table and those buckets.

An explicit deny statement **takes precedence** over an allow statement.

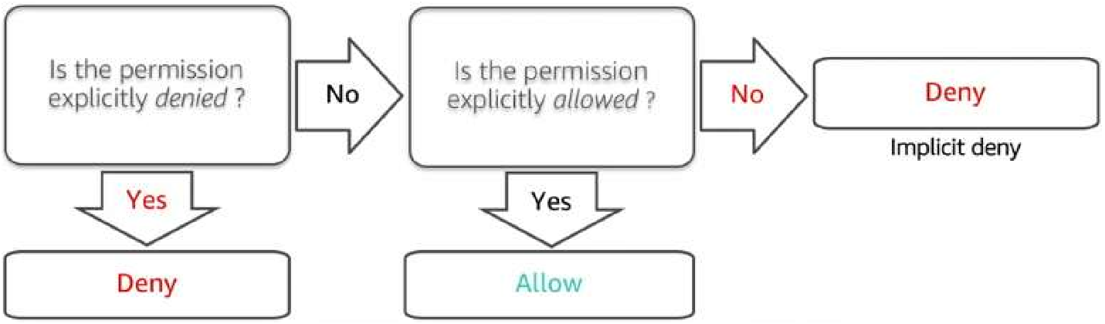
- Any actions not explicitly allowed are denied → out-of-the-box access are always deny (implicit deny)
- Any actions explicitly denied are always denied
- If there is a competition betwee an allowed statement and a deny statement, the deny statement always wins

Resource-based policies

- *Identity-based policies* are attached to a user, group or role
- **Ressource-based policies** are attached to a resource (*not* to a user, group or role)
- Characteristics of resource-based policies
  - Specifies who has access to the resource and what actions they can perform on it
  - The policies are *inline* only, not managed
- Resource-based policies are supported only by some AWS services

IAM permissions

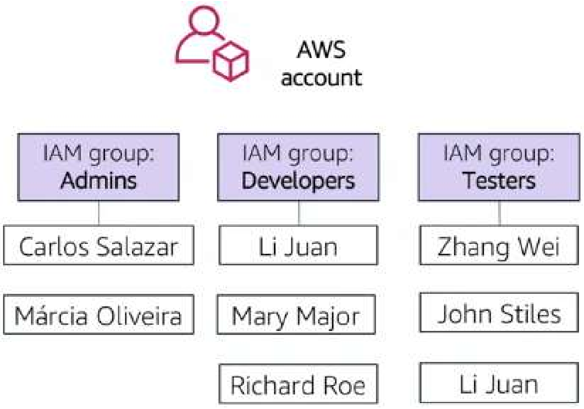
How IAM deterines permissions:


Is the permission explicitly *denied* ?
No
Is the permission explicitly *allowed* ?
No
Deny
Implicit deny
Yes
Deny
Yes
Allow

IAM groups

- An *IAM group* is a collection of IAM users

- A user can belong to multiple groups
- There is no default group
- Groups cannot be nested



IAM role

- An *IAM role* is an IAM identity with specific permissions
- Similar to an IAM user
  - attach permissions policies to it
- Different from IAM user
  - Not uniquely associated with one person
  - Intended to be *assumable* by a person, application or service
- Role provides *temporary* security credentials
- Examples of how IAM roles are used to **delegate** access
  - Used by an IAM user in the same AWS account as the role
  - Used by an AWS service (such as Amazon EC2) in the same account as the role
  - Used by an IAM user in a different AWS account than the role

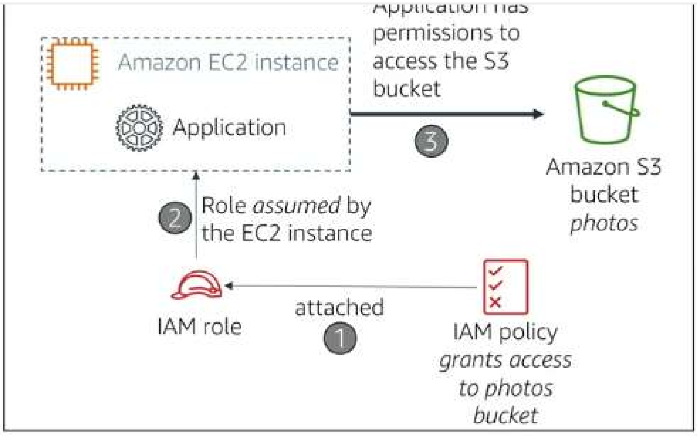
Example use of an IAM role

Scenario:

- An app that runs on an EC2 instance needs access to a S3 bucket

Solution:

- Define an IAM policy that grants read-only access to the S3 bucket
- Attach the policy to a role
- Allow the EC2 instance to assume the role


Application has permissions to access the S3 bucket
Amazon EC2 instance
Application
3
Amazon S3 bucket *photos*
2
Role *assumed* by the EC2 instance
IAM role
attached
1
IAM policy *grants access to photos bucket*

Section 3: Securing a new AWS account

AWS account root user access versus IAM access

- Best practice: **Do not use the AWS account root user except when necessary**
    - Access to the *account root user* requires logging in the the *email address* (and password) that you used to create the accout
- Example actions that can only be done with the account root user:
    - Update the account root user password
    - Changed the AWS Support plan
    - Restore an IAM user's permissions
    - Change account settings (for example, contact info, allowed Regions)

Securing a new AWS account: Account root user

Step 1: Stop using the account root user as soon as possible

The account root user has unrestricted access to all resources

To stop using the account root user:

1. While you are logged in as the account root user, *create an IAM user* for yourself. Save the access keys if needed
2. Create an IAM group, give it full administrator permissions, and add the IAM user to the group
3. Disable and *remove your account root user access keys*, if they exist
4. *Enable a password policy* for users
5. Sign in with your new IAM user credentials
6. Store your account root user credentials in a secure place

Step 2: Enable multi-factor authentication (MFA)

- Require MFA for your *account root user* and for *all IAM users*
- You can also use MFA to control access to AWS service APIs
- Options for retrieving the MFA token
    - Virtual MFA-compliant applications
        - Google Authenticator

  - U2F security key devices
    - YubiKey
  - Hardware MFA options
    - Key fob or dispLy card offered by Gemalto

Step 3: Use AWS CloudTrail

- CloudTrail tracks user activity on your account
  - Logs all API requests to resources in all supported services your account
- **Basic AWS Cloud** event history is enabled by default and is free
  - It contains all management event data on latest 90 days of account activity
- To accces CloudTrail
  1. Log in to the **AWS Management Console** and choose the **CloudTrail** service
  2. Click *Event History* to view, filter and search the last 90 days of events
- To enable logs beyond 90 days and enable specified event alerting, create a trail
  1. From the CloudTrail Console trails page, click *Create trail*
  2. Give it a name, apply it to all Regions, and create a new Amazon S3 bucket for log storage
  3. Configure access restrictions on the S3 bucket (for example, only admin users should have access)

Step 4: Enable a *billing report*, such as the AWS Cost and Usage Report

- Billing reports provide info about your use of AWS resources and estimated costs for that use
- AWS delivers the reports to an Amazon S3 bucket that you specify
  - report is updated at least one per day
- The *AWS Cost and Usage Report* tracks your AWS usage and provides estimated charges associated with you AWS account, either by the hour or by the day

Section 4: Securing accounts

AWS Oganizations

- **AWS Organizations** enables you to consolidate multiple AWS accounts so that you centrally manage them
- Security features of AWS Organizations:
  - *Group AWS accounts into organizational units* (OUs) and attach different access policies to each OU
  - *Integration and support for IAM*: permissions to a user are the intersection of what is allowed by AWS Organizations and what is granted by IAM in that account
  - *Use service control policies* to establish control over the AWS services and API actions that each AWS account can access

Service control policies

> Offer centralized control over accounts: limit permissions that are available in an account that is part of an organization

- In JSON
- Ensure that accounts compuly with access control guidelines
- SCPs are *similar* to IAM permissions policies
  - They use similar syntax
  - However, an SCP never grants permissions
  - Instead, SCPs *specify the maximum permissions* for an organization

AWS Key Management Service (AWS KMS)

- Enables you to **create and manage encryption keys**
- Enables you to control the use of encryption across AWS services and in your applications
- Integrated with AWS CloudTrail to log all key usage
- Uses hardware security modules (HSMs) that are validated by Federal Information Processing Standards (FIPS) 140-2 to protect keys

Amazon Cognito

- **Adds user sign-up, sign-in and access control to your web and mobile app**
- Scales to millions of users
- Support Sign-in with social identity providers, such as Facebook, Google and Amazon, and enterprise identity providers, such as Microsoft Active Directory via Security Assertion Markup Language (SAML) 2.0
- Help meet security requirementes

AWS Shield

- is a managed distributed denial of service (DDoS) protection service
- Safeguards applications running on AWS
- Provides always-on detextion and automatic inline mitigations
- AWS Shield Standard enabled for at no additional cost. AWS Shield Advanced is an optional paid service
  - Available to all customers
- Use it to **minimize application downtime and latency**

Section 5: securing data on AWS

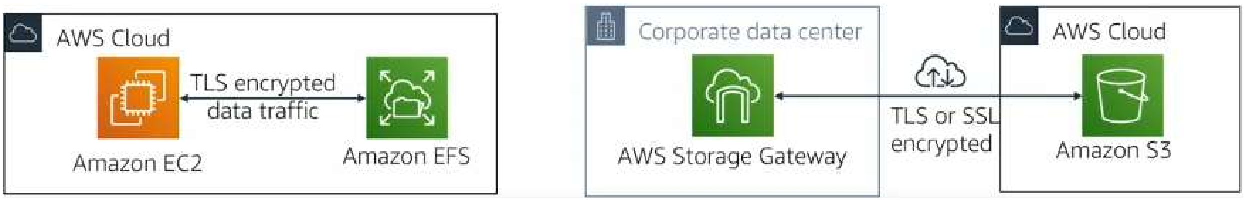
Encrytpion of data *at rest*

- **Encryption** encodes data with a *secret key*, wich makes it unreadable
  - Only those who have the secret key can decode the data
  - *AWS KMS* can manage you secret keys
- AWS supports encryption of *data at rest*
  - Data at rest = Data stored physically
    - Can encrypt any data supported by AWS key management service

    - Amazon S3
    - Amazon EBS
    - Amazon Elastic File System (Amazon EFS)
    - Amazon RDS managed databases

Encryption of data *in transit*

- Encryption of **data in transit** (data moving across a network)
    - Transport Layer Security (TLS) (formerly SSL) is an open standard protocol
    - **AWS Certificate Manager** provides a way to manage, deploy and renew TLS or SSL certificates
- Secure HTTP (HTTPS) creates a secure tunnel
    - uses TLS or SSL for the bidirectional exchange of data
- **AWS services support data in transit ecryption**



Securing Amazon S3 buckets and objects

- Newly created S3 buckets and objects are *private* and *protected* by default
- When use cases require sharing data objects on Amazon S3
    - It is essential to manage and control the data access
    - Follow the *permissions that follow the principle of least privilege* and consider using Amazon S3 encryption
- Tools and options for controlling access to S3 data include
    - Amazon S3 Block Public Access feature
    - IAM policies
    - Bucket policies: when can't log with IAM
    - Access control lists (ACLs): a legacy access control mechanism
    - AWS Trusted Advisor bucket permission check: a free feature

Section 6: Working to ensure compliance

AWS compliance programs

Customers are subject to many different security and compliance regulations and requirements

> AWS engages with certifyin bodies and independent auditors to provide customers with detailed infromation about the policies, processes, and controls that are established and operated by AWS

Compliance programs can be brodaly categorized

  - Assessed by a third-party, independent auditor
  - Examples: ISO 27001, 27017, 27018 and ISO/IEC 9001

- Laws, regulations, and privacy
  - AWS provides security features and legal agreements to support compliance
  - Examples: EU *General Data Protection regulation (GDPR)*, HIPAA

- Alignments and framework
  - Industry- or function-specific security or compliance requirements
  - Examples: Center for Internet Security (CIS), EU-US Privacy Shield certified

AWS Config

- **Assess, audit and evaluate the configurations of AWS resources**
- Use for continuous monitoring of configurations
- Automatically evaluate *recorded* configurations versus *desired* configurations
- Review configuration changes
- View detailed configuration histories
- **Simplify complicance auditing and security analysis**

AWS Artifact

- **Is a resource for compliance-related information**
- Provide access to security and compliance reports, and select online agreements
- Can access example downloads:
  - AWS ISO certifications
  - Payment Card Industry (PCI) and Service Organization Control (SOC) reports

- Access AWS Artifact directly from the AWS Management Console
  - Under *Security, Identity |& Compliance*

- Accept agreements with AWS on multiple accounts

Wrap-up

Sample exam question

Which of the following is AWS's reponisbility under the AWS shared responsibility model ?

1. Configuring a third-party app
2. Maintaining physical hardware
3. Securing app access and data
4. Managing custom Amazon Machine Image (AMIs)

▶ Answer

tronc commun AWS S8

Share:

Further Reading

Feb 8, 2021

AWS Module 1 - Cloud Concepts Overview

Lien de la note Hackmd Introduction Intro to cloud computing Advantages of cloud…

Feb 8, 2021

AWS Module 2 - Cloud Economics and Billing

Lien de la note Hackmd Section 1: Fundamentals of pricing AWS pricing mode…

Feb 9, 2021

AWS Module 3 - AWS Global Infrastructure Overview

Lien de la note Hackmd Section 1: AWS Global Infrastructure The AWS Global…

OLDER

AWS Module 3 - AWS Global Infrastructure Overview

NEWER

StartUp Lab