MODULE 4 · STUDY NOTES

AWS Cloud Security

The shared responsibility model, IAM, account hardening, data protection, and compliance

Security is AWS's stated highest priority, but "secure" in the cloud is a partnership: AWS protects the machinery of the cloud, and you protect what you build inside it. This module draws that boundary precisely, then tours the toolbox on your side of the line — IAM for identities and permissions, the hardening checklist every new account needs, the account-level guardrails (Organizations, KMS, Cognito, Shield), encryption for data at rest and in transit, and the compliance machinery (Config, Artifact, and AWS's certification programs) that lets regulated workloads run in the cloud.

LEARNING OBJECTIVES

- Recognize the shared responsibility model
- Identify the responsibility of the customer and AWS
- Recognize IAM users, groups, and roles
- Describe different types of security credentials in IAM
- Identify the steps to securing a new AWS account
- Explore IAM users and groups
- Recognize how to secure AWS data
- Recognize AWS compliance programs

4.1 The shared responsibility model

AWS and the customer split security duties along one clean line. **AWS is responsible for security OF the cloud**: the physical data centers (with strictly controlled, need-based access), the hardware and networking, and the virtualization layer that everything runs on. **The customer is responsible for security IN the cloud**: everything they choose to put on that infrastructure — the guest operating system and its patches, application software, security-group and firewall configuration, IAM identities, and the data itself.

AWS – OF THE CLOUD	CUSTOMER – IN THE CLOUD
the infrastructure that runs everything	*everything you configure and upload*
▪ Physical data centers, hardware, facilities	▪ Guest OS on EC2 — updates and security patches
▪ Hypervisor / virtualization layer	▪ Security groups and firewall configuration
▪ Global network infrastructure	▪ IAM users, credentials, and permissions
▪ Software underlying managed services such as RDS, Lambda, and DynamoDB	▪ Application code, customer data, and encryption choices

Where exactly the line sits depends on how managed the service is. The rule of thumb: **the more managed the service, the more security AWS carries.**

TABLE 4.1 Service model vs. security responsibility

MODEL	CHARACTER	CUSTOMER'S SHARE	EXAMPLES
IaaS	Basic building blocks; maximum flexibility and control	Most: OS patching, security groups, apps, access control	Amazon EC2
PaaS	AWS runs the underlying infrastructure, OS, and platform	Data, asset classification, permissions — AWS handles OS/DB patching, firewall, DR	AWS Lambda, Amazon RDS
SaaS	Centrally hosted software via browser, app, or API; subscription or pay-as-you-go	Only their data and access decisions	Trusted Advisor, AWS Shield, Amazon Chime

> **COMMON PITFALL**
>
> Exam questions love the boundary cases. Patching the **guest OS on EC2** is the customer's job; patching the OS and database engine **under RDS** is AWS's. Configuring the security group is always the customer's.

4.2 Identity and Access Management (IAM)

IAM answers two questions for every request: **who are you** (authentication) and **what are you allowed to do** (authorization). It is a feature of every AWS account, offered at no charge, and it is **global** — users, groups, roles, and policies exist account-wide, not per Region.

KEY TERMS

IAM user
A person or application with long-term credentials. Console access uses a password; programmatic access uses an access key ID + secret access key.

IAM group
A collection of users — the right way to grant the same permissions to many people. Groups cannot be nested.

IAM role
An identity with permissions but no long-term credentials. Users, applications, or services **assume** the role and receive temporary security credentials.

IAM policy
A JSON document that explicitly allows or denies actions on resources. Attachable to users, groups, and roles (identity-based) or to resources like S3 buckets (resource-based).

MFA
Multi-factor authentication — a one-time code on top of the password. Strongly recommended for every user, mandatory in spirit for root.

How authorization is evaluated. By default every request is **implicitly denied**. IAM then looks at all applicable policies: an *allow* statement grants the action, but a single **explicit deny anywhere overrides every allow**. Best practice is to follow the **principle of least privilege** — grant only the permissions a task requires, and let everything else stay denied.

Identity-based vs. resource-based policies. Identity-based policies travel with the user, group, or role. Resource-based policies are attached to the resource itself — an S3 bucket policy is the classic example — and specify who may act on that resource, which is useful for cross-account access.

ROLES IN PRACTICE

- An application on an EC2 instance needs to read an S3 bucket → attach a role to the instance; the app receives temporary credentials automatically.
- Never store access keys on an instance or in code — roles eliminate the need.
- Roles also power federated sign-in and cross-account access.

COMMON PITFALL

"IAM is a Regional service" is a favorite false statement — IAM is **global**. Also watch the wording *authentication* (proving identity) vs. *authorization* (permissions): IAM does both, but exam answers often hinge on which word is used.

4.3 Securing a new AWS account

Every new account starts with a single all-powerful identity: the **account root user**, reachable only with the email address and password used to create the account. Root can do literally anything — including closing the account — so step one of account hygiene is to stop using it. Root should keep no access keys, carry MFA, and be reserved for the handful of tasks that genuinely require it (like changing the account's support plan or closing the account).

THE HARDENING CHECKLIST, IN ORDER

- **Enable MFA on root** — a compromised root password alone then gets an attacker nowhere.
- **Delete (or never create) root access keys** — programmatic access should belong to IAM identities.
- **Create individual IAM users** and put them in groups with least-privilege policies; sign in as those users from then on.
- **Enable AWS CloudTrail** — records every API call; the console shows 90 days of event history free, and a trail extends retention by delivering logs to S3.
- **Enable billing reports** (e.g. the AWS Cost and Usage Report) — unexpected spend is often the first sign of compromise.
- **Set an IAM password policy** — minimum length, character classes, expiration, and reuse rules for all IAM user passwords.

The IAM console's **security status checks** track exactly this list — MFA on root, individual IAM users created, groups used for permissions, and a password policy applied. The module's lab walks through creating users and groups and seeing the effect of policies firsthand.

COMMON PITFALL

Root user vs. IAM administrator: an IAM user with `AdministratorAccess` can manage the account day-to-day, but only **root** can do things like change the root password, change the support plan, or close the account. The exam expects you to know that daily work never belongs on root.

4.4 Securing accounts at the organization level

AWS Organizations consolidates many AWS accounts under central management. Accounts group into **organizational units (OUs)**, and policies attach per OU — e.g. one OU for workloads bound by a regulation, with a policy blocking non-compliant services. Its sharpest security tool is the **service control policy (SCP)**: a JSON policy, written in IAM-like syntax, that sets the **maximum permissions** for member accounts. An SCP never grants anything — a user's effective permissions are the **intersection** of what the SCPs allow and what IAM grants inside the account, and an SCP restriction overrides even a member account's administrator.

TABLE 4.2 Account-level security services

SERVICE	JOB	DETAILS WORTH KEEPING
AWS KMS	Create and manage encryption keys; control encryption across services and apps	Keys protected by **FIPS 140-2 validated HSMs**; every key use logged to CloudTrail
Amazon Cognito	Sign-up, sign-in, and access control for web and mobile apps	Scales to millions of users; federates social providers (Google, Facebook, Amazon) and enterprise directories via **SAML 2.0**; HIPAA-eligible and PCI DSS-capable
AWS Shield	Managed DDoS protection with always-on detection and inline mitigation	**Standard**: automatic, free, for everyone. **Advanced**: optional paid tier for larger attacks (EC2, ELB, CloudFront, Global Accelerator, Route 53); DDoS Response Team requires Business/ Enterprise Support

COMMON PITFALL

SCP questions almost always test the same fact: SCPs **limit**, they never **grant**. If an SCP blocks an action, a member-account admin granting it in IAM changes nothing.

4.5 Securing data on AWS

Data at rest — data physically stored on disk or tape — is protected by encrypting it with **AES-256**. With AWS KMS managing the keys, encryption and decryption happen transparently: no application changes needed. Any KMS-integrated service can store encrypted data, including S3, EBS, EFS, and RDS. If policy or regulation requires encryption at rest, the guidance is simple: turn it on everywhere data lives.

Data in transit — data moving across a network — is protected with **TLS** (formerly SSL) using AES-256. **AWS Certificate Manager** provisions, deploys, and auto-renews TLS certificates for resources like load balancers and CloudFront distributions. Plain HTTP is readable in flight; **HTTPS** wraps traffic in a TLS tunnel, defeating eavesdropping and man-in-the-middle attacks. Typical encrypted paths: an EC2 instance mounting EFS, or Storage Gateway syncing on-premises data to S3.

SECURING AMAZON S3

- New buckets and objects are **private and protected by default** — nothing is public until you make it public.
- **S3 Block Public Access** is the simple, blunt safeguard against accidental exposure.
- Finer control: IAM policies, **bucket policies**, and access control lists (ACLs).
- When sharing is required: least-privilege permissions plus S3 encryption.

COMMON PITFALL

Whose job is encryption? **Yours.** AWS provides the machinery (KMS, ACM, service integrations), but deciding to encrypt — and enabling it — sits on the customer side of the shared responsibility line.

4.6 Working to ensure compliance

Customers inherit AWS's certifications for the infrastructure layer, but must still demonstrate compliance for what they build. AWS engages third-party certifying bodies and independent auditors, and packages the evidence for customers. Compliance programs fall into three buckets:

TABLE 4.3 AWS compliance program categories

CATEGORY	MEANING	EXAMPLES
Certifications & attestations	Assessed by an independent third-party auditor	ISO 27001, 27017, 27018, ISO/IEC 9001
Laws, regulations, privacy	AWS provides features and legal agreements supporting your compliance	GDPR, HIPAA
Alignments & frameworks	Industry- or function-specific requirements	CIS, EU-US Privacy Shield

AWS Config continuously records resource configurations and evaluates them against rules you define, giving you configuration history, change tracking, and a compliance dashboard that flags noncompliant resources. Note that Config is a **Regional** service — enable it in every Region you use (an aggregator can merge views across Regions and accounts). **AWS Artifact** is the document counter: on-demand downloads of AWS ISO certifications, PCI and SOC reports, plus online agreements such as the **Business Associate Agreement (BAA)** that HIPAA-regulated customers need. Artifact covers AWS's side only — evidence for your own applications is still yours to produce.

4.7 Additional security services

Beyond the core, four services round out the security toolbox — expect one-line "which service does X?" questions about each:

TABLE 4.4 Know each service's one-line job

SERVICE	ONE-LINE JOB
AWS Service Catalog	Catalogs of organization-approved IT products (AMIs, servers, databases, whole architectures) with constraints like allowed Regions or IP ranges — governance and consistent deployments
Amazon Macie	Machine-learning discovery, classification, and protection of sensitive data (PII) stored in S3, with alerts when it's accessed or moves
Amazon Inspector	Automated security assessments of deployed applications — findings listed by severity against best-practice baselines
Amazon GuardDuty	Intelligent threat detection that continuously analyzes CloudTrail events, VPC Flow Logs, and DNS logs with ML and threat intelligence

Module summary

- AWS secures the cloud (facilities, hardware, network, virtualization); the customer secures what's in it (OS, apps, data, IAM, firewall rules).
- The more managed the service (IaaS → PaaS → SaaS), the more of the security burden AWS carries.
- IAM = users, groups, roles, policies; global and free. Requests start implicitly denied; explicit deny beats every allow; grant least privilege.
- Roles issue temporary credentials — the right way to give applications (and EC2 instances) permissions.
- New account: MFA on root, no root access keys, individual IAM users, CloudTrail on, billing reports on, password policy set.
- Organizations + SCPs cap what member accounts can do; effective permissions = SCP ∩ IAM. SCPs never grant.
- Encrypt at rest with AES-256 via KMS; encrypt in transit with TLS via ACM; S3 is private by default — keep it that way.
- Compliance: AWS Config records and evaluates configurations (Regional!); AWS Artifact serves compliance reports and agreements; Macie / Inspector / GuardDuty watch data, apps, and threats.

Review questions

1. A company runs a web app on EC2 behind a security group, storing data in RDS. List two things AWS secures and two things the company secures.

Answer: AWS: physical hosts/data centers, virtualization layer, RDS's underlying OS and database patching. Company: the EC2 guest OS patches, security-group rules, IAM permissions, the data itself (and its encryption).

2. Why do AWS best practices favor roles over access keys for an application on EC2?

Answer: A role attached to the instance supplies temporary, automatically rotated credentials — no long-lived secret sits on the instance or in code where it can leak.

3. What exactly happens when an IAM policy allows s3:GetObject but an SCP on the account blocks all S3 actions?

Answer: The request is denied. Effective permissions are the intersection of SCP and IAM — the SCP caps what IAM can grant, even for admins.

4. Walk through the order of policy evaluation for any request.

Answer: Start at implicit deny → any applicable allow grants the action → any explicit deny overrides everything.

5. Name the six steps for securing a new AWS account.

Answer: Root MFA on; delete root access keys; create individual IAM users/groups; enable CloudTrail; enable billing reports; set an IAM password policy.

6. Which services would you use to (a) prove AWS's PCI compliance to an auditor, (b) detect a resource whose configuration drifted from your rules, (c) find PII sitting in S3?

Answer: (a) AWS Artifact, (b) AWS Config, (c) Amazon Macie.

7. **What's the difference between Shield Standard and Shield Advanced?**

   Answer: Standard is automatic, free DDoS protection for all customers; Advanced is a paid tier with protections for larger/sophisticated attacks and access to the DDoS Response Team (with Business or Enterprise Support).

8. **Data must be encrypted at rest and in transit for a regulated workload. Which AWS services deliver each half?**

   Answer: At rest: AWS KMS-managed keys (AES-256) in S3/EBS/EFS/RDS. In transit: TLS certificates issued and renewed by AWS Certificate Manager (HTTPS).