Securing Access to Cloud Resources

IAM · USERS, GROUPS & ROLES · MFA & STS · IDENTITY VS RESOURCE POLICIES · POLICY EVALUATION · FEDERATION · ORGANIZATIONS

STUDY & REVIEW

01 MUST KNOW

if you remember five things, remember these

01 **Two jobs, one service:** IAM controls both who is *authenticated* (signed in) and who is *authorized* (has permissions). **Authentication** establishes the identity of the requester — a **principal**, meaning a person or application — through credentials; **authorization** checks the policies relevant to the request to decide **allow or deny**.

02 **Four building blocks:** a **user** represents a person or application and has *permanent* credentials; a **group** is a collection of users granted identical permissions; a **role** grants *temporary* permissions that an identity assumes; a **policy** is the JSON document that defines the access. Best practice → attach policies to **groups**, then add users to groups.

03 **Roles are temporary:** a role is **not** tied to one person — a user, application, or AWS service **assumes** it and receives temporary security credentials issued by **AWS STS**. Assuming a role temporarily sets aside the assumer's own permissions. Roles delegate access: EC2 apps, cross-account access, mobile apps.

04 **Deny wins:** by default **every request is denied**. An explicit **Allow** overrides the default deny; an explicit **Deny** overrides any Allow. All applicable policies are evaluated and **order never matters** — the most restrictive result stands.

05 **Only two types grant: identity-based** and **resource-based** policies grant permissions. **Permissions boundaries**, **SCPs**, and **session policies** only set a *maximum* — they can never grant. An action succeeds only if *every* applicable policy allows it.

02 KEY TERMS

TERM	DEFINITION
IAM	Web service that helps you securely control who is authenticated and authorized to use AWS resources; integrated into most AWS services
Principal	A person or application that uses the root user, an IAM user, or an IAM role to sign in and make requests (includes assumed roles and federated users)
IAM entity	The IAM objects AWS uses for authentication: IAM users and IAM roles
IAM identity	The IAM objects used to identify and group, to which a policy can be attached: users, groups, and roles
IAM role	An identity that grants a temporary set of permissions; it is assumed as needed and has no long-term credentials
IAM policy	A JSON document that defines which resources can be accessed and the level of access to each
AWS STS	AWS Security Token Service — issues the temporary security credentials used for a role session
Identity-based policy	Attached to an IAM user, group, or role; states what that identity can do
Resource-based policy	Attached to a resource; states which principal may act on it (enables cross-account access)
Permissions boundary	Advanced managed-policy feature that sets the maximum permissions an identity-based policy can grant to an entity; grants nothing itself
SCP	Service control policy — an AWS Organizations policy that sets the maximum permissions for member accounts or OUs; cannot grant permissions
Identity federation	A system of trust between an identity provider (authenticates users) and a service provider (controls resource access)

03 IAM USER VS. IAM ROLE

permanent identity vs. temporary session

IAM USER	IAM ROLE
who you are — long-term	*what you assume — temporary*
— Represents one person or application	— Not uniquely associated with one person
— Has a *permanent* set of credentials (password and/or access keys) until rotated	— A user, application, or AWS service **assumes** it as needed
— Used for everyday, long-term access	— **AWS STS** issues temporary credentials for the session
— Root user is the first IAM user on a new account	— Delegates access: EC2 apps, cross-account, mobile apps

04 THE SIX POLICY TYPES

only identity- and resource-based grant

POLICY TYPE	ATTACHED TO	GRANTS PERMISSIONS?
Identity-based	IAM user, group, or role	Yes — what the identity can do
Resource-based	A resource (e.g., an S3 bucket)	Yes — to the principal it names
Permissions boundary	An IAM entity (advanced)	No — caps identity-based grants
SCP	An account or OU in Organizations	No — caps member-account permissions
ACL	A resource; cross-account only	Yes — the only non-JSON policy type
Session policy	A role or federated-user session	No — limits the session

05 HOW IAM EVALUATES A REQUEST

explicit deny always wins

01 Default: **every request is denied** → 02 Evaluate **all** applicable policies (order is irrelevant) → 03 Any explicit **Deny**? → **DENY** (final) → 04 Else any explicit **Allow**? → **ALLOW** → 05 Else → **DENY** (implicit deny)

06 ADDITIONAL IDENTITY & ACCESS SERVICES

know the one-line job of each

SERVICE	WHAT IT DOES
AWS SSO	Create/connect identities once and manage access centrally across accounts; user portal — best for a single centralized directory
IAM (federation)	Federate when you have multiple directories or want attribute-based permissions
AWS Directory Service	AWS Managed Microsoft AD — extend on-premises Active Directory to AWS using existing credentials
Amazon Cognito	Sign-up, sign-in, and access control for web/mobile apps; user pools + identity pools; social IdPs
AWS Organizations	Consolidate and centrally manage multiple accounts; OUs; SCPs; consolidated billing

07 EXAM TRAPS

where the knowledge check gets you

- × “An IAM role works like a user, with a password and access keys.” — A role has **no long-term credentials**; AWS STS issues temporary credentials only when the role is assumed.
- × “Assuming a role adds the role's permissions on top of your own.” — When you assume a role, your prior permissions are **temporarily set aside**; you get only the role's permissions for the session.
- × “Permissions boundaries and SCPs grant permissions.” — Neither grants anything; both only set the **maximum**. You still need an identity- or resource-based policy to allow the action.
- × “The order policies are evaluated in decides the outcome (first match wins).” — Order **never** matters; AWS evaluates all applicable policies, and an explicit **deny** always overrides any allow.
- × “A brand-new IAM user can use the account's resources right away.” — By default an authenticated user, group, or role **can't access anything** until you grant permissions (implicit deny).
- × “A resource-based policy is attached to the identity it applies to.” — It is attached to the **resource** and names the principal; **identity-based** policies are the ones attached to a user, group, or role.
- × “All IAM policies are JSON documents.” — All except **ACLs** — the ACL is the only policy type that doesn't use the JSON format.
- × “Use AWS SSO for multiple directories and IAM for one.” — Backwards: use **AWS SSO** for a single centralized directory; use **IAM** when you have multiple directories or want attribute-based permissions.

08 SELF-QUIZ

answers are upside-down below

Q1 In IAM, what name is given to a person or application that signs in and makes requests to AWS?

Q2 Which IAM identity provides only temporary permissions instead of long-term credentials?

Q3 Which AWS service issues the temporary security credentials used when a role is assumed?

Q4 What is the AWS best practice for giving many users the same set of permissions?

Q5 Before any policy is attached, what access does an authenticated IAM user have to account resources?

Q6 A policy attached to a user, group, or role that states what that identity may do is which kind of policy?

Q7 In IAM policy evaluation, which result always overrides an explicit allow?

Q8 Aside from SCPs and session policies, which policy type sets the maximum permissions an identity-based policy can grant to an entity?

Q9 Which AWS service centrally manages multiple accounts and enforces service control policies across organizational units?

Q10 How would an administrator add an extra layer of login security beyond a user name and password for the console?

ANSWER KEY (rotate the page)

Q1 a principal **Q2** an IAM role **Q3** AWS Security Token Service (AWS STS) **Q4** attach the policy to a group, then add the users to the group **Q5** none — everything is implicitly denied until granted **Q6** an identity-based policy **Q7** an explicit deny **Q8** a permissions boundary **Q9** AWS Organizations **Q10** activate multi-factor authentication (MFA)