**1. What is AWS IAM and why is it important?**

**Answer:** AWS IAM (Identity and Access Management) is a service that enables you to securely manage access to AWS resources.

* **Importance:**
  + Controls who can access what (authentication & authorization)
  + Enforces least privilege
  + Supports MFA for security
  + Enables centralized management of users, roles, and policies

**2. Explain the difference between an IAM user, group, and role.**

**Answer:**

* **IAM User:** Individual identity with long-term credentials (password, access keys).
* **IAM Group:** Collection of users to manage permissions collectively.
* **IAM Role:** Temporary credentials assumed by users, AWS services, or external accounts. Roles do not have long-term credentials.

**3. What are IAM policies and how are they structured?**

**Answer:** IAM policies are JSON documents that define permissions. **Structure:**

{

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

"Statement": [

{

"Effect": "Allow",

"Action": "s3:ListBucket",

"Resource": "arn:aws:s3:::my-bucket"

}

]

}

* **Effect** – Allow/Deny
* **Action** – AWS service actions
* **Resource** – ARN of the resource
* **Condition** – Optional conditions

**4. What is the difference between identity-based and resource-based policies?**

**Answer:**

* **Identity-based policies**: Attached to users, groups, or roles (e.g., allow S3 access).
* **Resource-based policies**: Attached to resources like S3 buckets, Lambda, etc., to specify who can access them.

**5. What is the IAM root user and why should its usage be minimized?**

**Answer:** The root user is the initial account owner with full permissions.

* **Why minimize?**
  + High-security risk if compromised
  + Best practices: Enable MFA, create IAM admin users, and never use root for daily operations.

**6. Scenario: Multiple developers work on different projects. How to restrict each to only their project resources?**

**Answer:**

* Create separate **IAM groups** for each project.
* Attach least privilege policies (project-specific).
* Assign developers to the right group.

**7. How can you enforce MFA for IAM users in an AWS account?**

**Answer:**

* Enable MFA per user in the console.
* Use a **policy condition** to require MFA:

"Condition": { "Bool": { "aws:MultiFactorAuthPresent": "true" } }

**8. What happens if two conflicting IAM policies are applied to a user?**

**Answer:**

* **Explicit DENY** always overrides any ALLOW.
* If there’s no explicit allow, access is denied by default.

**9. Scenario: Allow an EC2 instance to access an S3 bucket without storing credentials.**

**Answer:**

* Create an **IAM Role** with S3 permissions.
* Attach the role to the EC2 instance.
* The EC2 instance can now access S3 using temporary credentials from the instance profile.

**10. Scenario: Grant temporary access to an external vendor without creating a permanent IAM user.**

**Answer:**

* Use **IAM Roles with AWS STS (AssumeRole)** to generate temporary credentials.
* Set a trust policy for the vendor's AWS account.

**11. Scenario: Restrict a user to only list objects in an S3 bucket.**

**Answer:** Attach this policy:

{

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

"Statement": [

{

"Effect": "Allow",

"Action": "s3:ListBucket",

"Resource": "arn:aws:s3:::my-bucket"

}

]

}

**12. Scenario: Lambda should read from DynamoDB but not write.**

**Answer:**

* Create an IAM Role with a policy allowing only dynamodb:GetItem and dynamodb:Scan.
* Attach the role to the Lambda function.

**13. How to audit IAM users who have not rotated their access keys in 90 days?**

**Answer:**

* Use **IAM Credential Report** (CSV).
* Or use **AWS Config rule: access-keys-rotated**.

**14. Scenario: Allow IAM users in Account A to access resources in Account B securely.**

**Answer:**

* Create a **Cross-Account IAM Role** in Account B.
* Add a trust policy allowing Account A to assume it.
* Users in Account A use sts:AssumeRole.

**15. Scenario: Prevent an IAM user from deleting CloudTrail logs.**

**Answer:**

* Use **S3 bucket policy** with an explicit deny for delete actions.
* Optionally use **Service Control Policies (SCP)** in AWS Organizations.

**16. Practical: IAM Policy to allow Start/Stop EC2 but not Terminate.**

{

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

"Statement": [

{

"Effect": "Allow",

"Action": [

"ec2:StartInstances",

"ec2:StopInstances"

],

"Resource": "\*"

},

{

"Effect": "Deny",

"Action": "ec2:TerminateInstances",

"Resource": "\*"

}

]

}

**17. How to verify what permissions an IAM user actually has?**

**Answer:**

* Use **IAM Policy Simulator** in AWS Console.
* Or use AWS CLI:
* aws iam simulate-principal-policy --policy-source-arn arn:aws:iam::123456789012:user/Alice --action-names s3:ListBucket

**18. Practical: Create IAM user, attach policy, and generate access keys via CLI.**

aws iam create-user --user-name dev-user

aws iam attach-user-policy --user-name dev-user --policy-arn arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess

aws iam create-access-key --user-name dev-user

**19. How to implement least privilege access in AWS?**

**Answer:**

* Give only necessary permissions.
* Use roles instead of users where possible.
* Regularly audit policies.
* Avoid \* in actions/resources.

**20. What AWS service helps manage permissions across multiple accounts?**

**Answer:**

* **AWS Organizations** with **Service Control Policies (SCPs)** is used to centrally control permissions.

**21. What are inline policies and how do they differ from managed policies?**

**Answer:**

* **Inline Policy** – Policy embedded directly into a user, group, or role. Tied to that entity.
* **Managed Policy** – Standalone policy that can be reused across multiple entities.
* **Best Practice:** Prefer managed policies for reusability and easier management.

**22. What is an IAM permission boundary and when would you use it?**

**Answer:** A **permission boundary** is an advanced feature that sets the **maximum permissions** an IAM entity can have, even if other policies grant more.

* **Use Case:** Limit what IAM users/roles created by other admins can do.

**23. How does IAM integrate with AWS Organizations Service Control Policies (SCPs)?**

**Answer:**

* SCPs define the **maximum permissions** for accounts under an AWS Organization.
* IAM policies cannot grant permissions beyond what SCP allows.

**24. How do you allow a third-party SaaS application to access your AWS resources securely?**

**Answer:**

* Create an IAM Role with external ID and trust policy.
* Share role ARN with the third party.
* They assume the role using STS without needing long-term credentials.

**25. Scenario: A user reports "Access Denied" even though the policy allows access. What could be the reasons?**

**Answer:**

* SCP explicitly denies access.
* Resource policy denies access.
* Missing required conditions (MFA, IP restrictions).
* Permissions boundary or session policy restricting access.

**26. What are IAM session policies and when are they used?**

**Answer:** Session policies are **policies passed during the AssumeRole or federation process**.

* **Use Case:** Grant **temporary, limited permissions** without modifying the role.

**27. What is AWS STS and how is it used with IAM?**

**Answer:** AWS **Security Token Service (STS)** issues temporary credentials for users, services, or federated identities.

* Used in cross-account roles, MFA enforcement, and external user access.

**28. Scenario: How do you restrict an IAM user to only access AWS services from a specific IP address?**

**Answer:** Attach a policy with an IP condition:

"Condition": { "IpAddress": { "aws:SourceIp": "203.0.113.0/24" } }

**29. How can you allow federated users (from Active Directory) to access AWS resources?**

**Answer:**

* Use **AWS SSO** or **SAML 2.0 federation**.
* Map AD users to IAM roles via trust policies.

**30. Scenario: You need to ensure users can only create EC2 instances of a specific type (e.g., t2.micro).**

**Answer:** Use a condition in the IAM policy:

"Condition": {

"StringEquals": {

"ec2:InstanceType": "t2.micro"

}

}

**31. What are the differences between AWS IAM and AWS Cognito?**

**Answer:**

* **IAM:** Manages AWS resources access (for internal users, services).
* **Cognito:** Manages authentication for external app users (e.g., customers).

**32. Scenario: Prevent an IAM user from accidentally deleting an S3 bucket.**

**Answer:**

* Add a policy with Effect: Deny for s3:DeleteBucket.
* Use S3 bucket policy with explicit deny.

**33. What is the difference between IAM Role and AWS Resource Policy-based access (like S3 bucket policy)?**

**Answer:**

* IAM Role – Grants permissions to the entity assuming it.
* Resource Policy – Controls access directly on the resource, allowing cross-account access without a role.

**34. How do you monitor and detect unauthorized IAM activities?**

**Answer:**

* Enable **CloudTrail** to log API calls.
* Set **CloudWatch Alarms** for suspicious activities.
* Use AWS **GuardDuty** for threat detection.

**35. Scenario: You want to enforce that all IAM users must use TLS (HTTPS) when accessing S3.**

**Answer:** Use a policy with condition:

"Condition": { "Bool": { "aws:SecureTransport": "true" } }

**36. What is IAM Access Analyzer and how is it useful?**

**Answer:** IAM Access Analyzer helps identify **resources shared publicly or with external accounts**.

* Helps detect unintended access.

**37. How do you enforce tagging of resources at the IAM level?**

**Answer:** Use IAM policy conditions like:

"Condition": {

"StringEqualsIfExists": { "aws:RequestTag/Environment": "Prod" }

}

**38. Scenario: How can you ensure an IAM user cannot use the AWS Management Console but only CLI/SDK?**

**Answer:**

* Do not set a console password.
* Attach policy denying aws:ViaAWSService conditions when console access is detected.

**39. What are best practices for managing IAM access keys?**

**Answer:**

* Rotate regularly (every 90 days or less).
* Avoid embedding in code.
* Use roles wherever possible.
* Disable unused keys.

**40. Scenario: Limit a developer to access only one specific DynamoDB table, not others.**

**Answer:** Attach a policy with resource-level restriction:

"Resource": "arn:aws:dynamodb:us-east-1:123456789012:table/Orders"