## Your goals

* Be able to describe identities in IAM
* Understand standard API request flow
* Know what Amazon Resource Name (ARN) is
* Learn about different types of IAM policies and what they consist of
* Be able to tell what types of AWS credentials exist
* Know about IAM Access Analyzer and IAM Policy Simulator

It's very important to understand that every interaction with AWS services is a request to API endpoint - create instance, run Lambda function or observe the billing. All of this is available through API. Each request is passed via HTTPS[1], validated on API endpoint[2], authenticated and authorized on IAM[3] and recorded in CloudTrail[4] even the request was unsuccessful.

This means that each small interaction in the AWS is authenticated via IAM.

## Use cases / Considerations

* **Fine-grained access control to AWS resources** - IAM enables [your users](https://aws.amazon.com/iam/features/manage-users/) to [control access](https://aws.amazon.com/iam/features/manage-permissions/) to AWS service APIs and to specific resources. IAM also enables you to add [specific conditions](https://aws.amazon.com/iam/features/manage-permissions/) such as time of day to control how a user can use AWS, their originating IP address, whether they are using SSL, or whether they have authenticated with a [multi-factor authentication device](https://aws.amazon.com/iam/features/mfa/). IAM is secure by default; users have no access to AWS resources until permissions are explicitly granted.
* **Temporary credentials** - In addition to defining access permissions directly to users and groups, IAM lets you create roles. Roles allow you to define a set of permissions and then let authenticated users or EC2 instances assume them, increasing your security posture by granting temporary access to the resources you define.
* **Multi-factor authentication for highly privileged users** - protect your AWS environment by using [AWS MFA](https://aws.amazon.com/iam/features/mfa/), a security feature available at no extra cost that augments user name and password credentials. MFA requires users to prove physical possession of a hardware MFA token or MFA-enabled mobile device by providing a valid MFA code.
* **Analyze access** - IAM helps you [analyze access](https://aws.amazon.com/iam/features/analyze-access/) across your AWS environment. Your security teams and administrators can quickly validate that your policies only provide the intended public and cross-account access to your resources. You can also easily identify and refine your policies to allow access to only the services being used. This helps you to better adhere to the principle of least privilege.
* **Integrate with your corporate directory** - IAM can be used to grant your employees and applications [federated access](https://aws.amazon.com/identity/federation/) to the AWS Management Console and AWS service APIs, using your existing identity systems such as Microsoft Active Directory. You can use any [identity management](http://docs.aws.amazon.com/IAM/latest/UserGuide/IdP-solution-providers.html) solution that supports [SAML 2.0](https://aws.amazon.com/identity/saml/), or feel free to use one of our federation samples ([AWS Console SSO](http://aws.amazon.com/code/4001165270590826) or [API federation](http://aws.amazon.com/code/1288653099190193)).
* **Seamlessly integrated into AWS services** - IAM is integrated into most AWS services. This provides the ability to define access controls from one place in the AWS Management Console that will take effect throughout your AWS environment.

### Access keys

Access keys are long-term credentials for an IAM user or the AWS account root user. You can use access keys to sign programmatic requests to the AWS CLI or AWS API (directly or using the AWS SDK).

1. What is the difference between IAM identity and IAM principle?
2. Which type of request to AWS is recorded by CloudTrail?
3. What's the difference between customer managed IAM policy and AWS managed IAM policy?
4. How does inline policy differ from managed policy?
5. What are the limits for inline IAM policy?
6. What are the best practices when working with permissions?
7. The AWS CLI credentials and configuration settings take precedence in which order?(name first 3)
8. What is the allow/deny priority order when policies are configured on different levels (group, user, etc.)?

### IAM policy concepts

You use IAM policies to define permissions for your IAM entities ([groups](http://docs.aws.amazon.com/IAM/latest/UserGuide/id_groups.html), [users](http://docs.aws.amazon.com/IAM/latest/UserGuide/id_users.html), and [roles](http://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_terms-and-concepts.html)). [Policies](http://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html) are composed of one or more [statements](http://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_elements_statement.html) that include the following elements:

* [Effect](http://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_elements_effect.html): Determines if a policy statement allows or explicitly denies access.
* [Action](http://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_elements_action.html): Defines AWS service actions in a policy (these typically map to individual AWS APIs.)
* [Resource](http://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_elements_resource.html): Defines the AWS resources to which actions can apply. The defined resources must be supported by the actions defined in the Action element for permissions to be granted.
* [Condition](http://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_elements_condition.html): Defines when a permission is allowed or denied. The conditions defined in a policy must be supported by the actions defined in the Action element for the permission to be granted.

Cloudtrail make logging. Creating users, delete processes. You can store cloudtrail logs in s3. We can

Connect cloudwatch to alert if some situation happens.

Key management service, KMS

aws kms list-keys

AWS KMS solution uses an envelope encryption strategy with AWS KMS keys. Envelope encryption is the practice of encrypting plaintext data with a data key, and then encrypting the data key under another key. Use KMS keys to generate, encrypt, and decrypt the data keys that you use outside of AWS KMS to encrypt your data. KMS keys are created in AWS KMS and never leave AWS KMS unencrypted.

AWS KMS supports three types of KMS keys: Customer-managed keys, AWS managed keys, and AWS owned keys (for more information see here - [AWS KMS keys](https://docs.aws.amazon.com/kms/latest/developerguide/concepts.html#master_keys)in the AWS KMS Developer Guide). For many FSI customers, Customer-managed KMS keys will be the preferred option because it allows for control of the permissions to use keys from any of their applications or AWS services. Customer-managed KMS keys also provide added flexibility for key generation and storage. In addition, every use of a KMS key or modification to its policy is logged to AWS CloudTrail for auditing purposes.

KMS does not store data keys itself, so we are responsible for storing the CMK-encrypted data key. When we want to use it, we make a single call to KMS - to decrypt the data key - which we then use locally to do our processing. This solves the problems identified above - we can encrypt more data, quicker, because we are doing the majority of the work locally.

KMS does not store the data key itself; it is your responsibility to store the encrypted version of the key somewhere, so that you can access it and decrypt it via KMS when you need to use it in future. In our example here we've stored it in an environment variable; in a real-world use case, we'd likely store it alongside the encrypted data, for instance, in a database.

AWS KMS supports symmetric and asymmetric CMKs. A symmetric CMK represents a 256-bit key that is used for encryption and decryption. An asymmetric CMK represents an RSA key pair that is used for encryption and decryption or signing and verification (but not both), or an elliptic curve (ECC) key pair that is used for signing and verification.

When you connect to ec2 instance it wants to configure aws user. If user does not have permission to see s3 bucket, then can instance see it?

First I create testPolicy that sees s3bucket ex testbucket;

Then I try to connect ec2 testinsance. There I try to call s3 bucket. If I am admin I can call all the time.

If not I cannot call even testInstance has testPolicy

You can configure the service control policies (SCPs) in your organization to work as either of the following:

* A [deny list](https://docs.aws.amazon.com/organizations/latest/userguide/orgs_manage_policies_scps_strategies.html#orgs_policies_denylist) – actions are allowed by default, and you specify what services and actions are prohibited
* An [allow list](https://docs.aws.amazon.com/organizations/latest/userguide/orgs_manage_policies_scps_strategies.html#orgs_policies_allowlist) – actions are prohibited by default, and you specify what services and actions are allowed

You can create the following types of budgets:

* **Cost budgets** – Plan how much you want to spend on a service.
* **Usage budgets** – Plan how much you want to use one or more services.
* **RI utilization budgets** – Define a utilization threshold and receive alerts when your RI usage falls below that threshold. This lets you see if your RIs are unused or under-utilized.
* **RI coverage budgets** – Define a coverage threshold and receive alerts when the number of your instance hours that are covered by RIs fall below that threshold. This lets you see how much of your instance usage is covered by a reservation.
* **Savings Plans utilization budgets** – Define a utilization threshold and receive alerts when the usage of your Savings Plans falls below that threshold. This lets you see if your Savings Plans are unused or under-utilized.
* **Savings Plans coverage budgets** – Define a coverage threshold and receive alerts when your Savings Plans eligible usage that is covered by Savings Plans fall below that threshold. This lets you see how much of your instance usage is covered by Savings Plans.

## Policy types

The following policy types, listed in order from most frequently used to less frequently used, are available for use in AWS. For more details, see the sections below for each policy type.

* [**Identity-based policies**](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html#policies_id-based) – Attach [managed](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html#managedpolicy) and [inline](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html#inline) policies to IAM identities (users, groups to which users belong, or roles). Identity-based policies grant permissions to an identity.
* [**Resource-based policies**](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html#policies_resource-based) – Attach inline policies to resources. The most common examples of resource-based policies are Amazon S3 bucket policies and IAM role trust policies. Resource-based policies grant permissions to the principal that is specified in the policy. Principals can be in the same account as the resource or in other accounts.
* [**Permissions boundaries**](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html#policies_bound) – Use a managed policy as the permissions boundary for an IAM entity (user or role). That policy defines the maximum permissions that the identity-based policies can grant to an entity, but does not grant permissions. Permissions boundaries do not define the maximum permissions that a resource-based policy can grant to an entity.
* [**Organizations SCPs**](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html#policies_scp) – Use an AWS Organizations service control policy (SCP) to define the maximum permissions for account members of an organization or organizational unit (OU). SCPs limit permissions that identity-based policies or resource-based policies grant to entities (users or roles) within the account, but do not grant permissions.
* [**Access control lists (ACLs)**](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html#policies_acl) – Use ACLs to control which principals in other accounts can access the resource to which the ACL is attached. ACLs are similar to resource-based policies, although they are the only policy type that does not use the JSON policy document structure. ACLs are cross-account permissions policies that grant permissions to the specified principal. ACLs cannot grant permissions to entities within the same account.
* [**Session policies**](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html#policies_session) – Pass advanced session policies when you use the AWS CLI or AWS API to assume a role or a federated user. Session policies limit the permissions that the role or user's identity-based policies grant to the session. Session policies limit permissions for a created session, but do not grant permissions. For more information, see [Session Policies](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html#policies_session).

IAM Policy Simulator; cloudtrail; assign roles to ec2 instances; [Envelope Encryption](https://catalog.us-east-1.prod.workshops.aws/workshops/aad9ff1e-b607-45bc-893f-121ea5224f24/en-US/keymanagement-kms/envelope-encryption" \l "envelope-encryption)