Identity and Access Management (IAM)

Amazon Web Services ([AWS](https://www.simplilearn.com/introduction-to-amazon-web-services-aws-article)) cloud provides a secure virtual platform where users can deploy their applications. Compared to an on-premises environment, AWS security provides a high level of data protection at a lower cost to its users. There are many types of security services, but Identity and Access Management (IAM) is one the most widely used. AWS IAM enables you to securely control access to [AWS services](https://www.simplilearn.com/introduction-to-amazon-web-services-aws-article) and resources for your users. Using IAM, you can create and manage AWS users and groups, and use permissions to allow and deny their access to AWS resources.

**Example:**

Suppose a person has a start-up with 3-4 members and hosted the Application over Amazon. Since it’s a small organization everybody would have access to Amazon where they can configure and perform other activities with their Amazon Account. Once the team size grows with a set of people in each department, he would not prefer to give full access to Amazon Web services, as they are all employees and the data needs to be protected. In this case, it would be advisable to create a few Amazon web service accounts called the IAM users. They are Amazon users who can perform the same activity as anyone. The advantage here is that we can control in what domain they can work, as we provide access only for S3 or Finance (Billing portal) and so on.

Now, if the company grows further, the challenge would be in giving access. For example, if we give 10 users for S3, 15 users for Finance, 20 users for EC2 and Cloud Watch, one user managing EC2 and  Cloud Watch will go to storage group and one person from finance will shift to HR. In that case we have to manage the access for each user individually. The better option is called Amazon IAM Group Concept, which is a collection of user where we can define a policy at group level. Once the policy is defined it will be applied on all the users at the group level.

The policy of each group is defined and applied to the respective group. We can move one user from a group to the other. This gives an advantage of keeping the policy the same. For example, if the organization is expanding and development can access to only certain instances, we need to change the permission and implement it as a part of policy.

Suppose the team grows to 4,000 people with various tasks and departments. It becomes difficult to manage the users. The best solution would be that Amazon supports the single sign in with the directory services. Amazon provides service supported by SAML based authentication. Whenever access is required, it would not ask for any credential when somebody from the organization logs in to the organization machine. It would then to the Amazon Portal and it would show services that the particular user is allowed to use. The biggest advantage is that there is no need to create multiple users but implement a simple sign in with the directory service. So when the user logs in to the corporate directory, he or she would be assigned certain privilege which allows the user to access Amazon. All these are possible only through Amazon IAM.

## Use Case

Suppose we create a mobile application like Instagram. Google application takes data from the users. So, whenever we upload a picture it will be stored in Amazon S3 bucket.  Now, whenever we want to upload anything in S3 bucket, we must provide our access credential. Here, Google application needs to store the access credentials which is insecure.  The solution here is that AMI is used, which provides web authorization where it supports the authentication provided from Google, Amazon and Facebook. Amazon would authorize based on these log ins. Then it would assign temporary credentials to upload items in S3. It’s important to know that it is applicable only for Amazon Resources.

## ****What Is AWS Security?****

[Cloud security](https://www.simplilearn.com/things-you-must-know-about-cyber-security-in-the-cloud-article) is the highest priority in AWS. When you host your environment in the cloud, you can be assured that it’s hosted in a data center or in a network architecture that’s built to meet the requirements of the most security-sensitive organization. Additionally, this high level of security is available on a pay-as-you-go basis, meaning there is really no upfront cost, and the cost for using the service is a lot cheaper compared to an on-premises environment.

There are many types of security services available but some of them are widely used by AWS, such as:

* IAM
* Key Management System (KMS)
* Cognito
* Web Access Firewall (WAF)

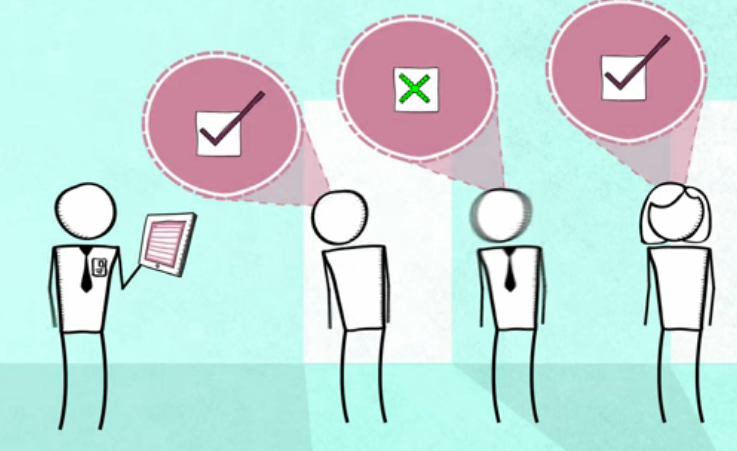
IAM enables you to manage access to AWS services and resources in a very secure manner. With IAM you can create groups and allow those users or groups to access some servers, or you can deny them access to the service.

## ****What Is IAM?****

AWS Identity and Access Management (IAM) is a web service for securely controlling access to AWS resources. It enables you to create and control services for user authentication or limit access to a certain set of people who use your AWS resources.

With the help of a single AWS IAM, the user can manage the customer and their needs. It provides Amazon [**AWS building blocks**](https://data-flair.training/blogs/aws-architecture/) which help the user to build the applications for the security purpose. AWS Identity and access management help the user to focus on the features and functionality which includes the security on the other side of the things. AWS IAM can also rotate access keys on the virtual machine instances. This makes sure that only the trusted user can access the AWS at any given time. Amazon IAM is economical and there are no additional charges for it. One can start using IAM very easily.

The biggest advantage of IAM is that it is free. But if users launch EC2 it will be charged.



## ****What is AWS KMS?****

AWS Key Management Service (AWS KMS) is a managed service that makes it easy for you to create and control the encryption keys used to encrypt your data. The customer master keys that you create in AWS KMS are protected by hardware security modules (HSMs). Our HSMs are validated by the [FIPS 140-2 Cryptographic Module Validation Program](https://csrc.nist.gov/projects/cryptographic-module-validation-program/Certificate/3139) except in the China (Beijing) and China (Ningxia) Regions.

## ****What Is Amazon Cognito?****

Amazon Cognito provides authentication, authorization, and user management for your web and mobile apps. Your users can sign in directly with a user name and password, or through a third party such as Facebook, Amazon, or Google.

## ****What Is AWS WAF?****

AWS WAF is a web application firewall that helps protect your web applications from common web exploits that could affect application availability, compromise security, or consume excessive resources. AWS WAF gives you control over which traffic to allow or block to your web applications by defining customizable web security rules. You can use AWS WAF to create custom rules that block common attack patterns, such as SQL injection or cross-site scripting, and rules that are designed for your specific application. New rules can be deployed within minutes, letting you respond quickly to changing traffic patterns. Also, AWS WAF includes a full-featured API that you can use to automate the creation, deployment, and maintenance of web security rules.

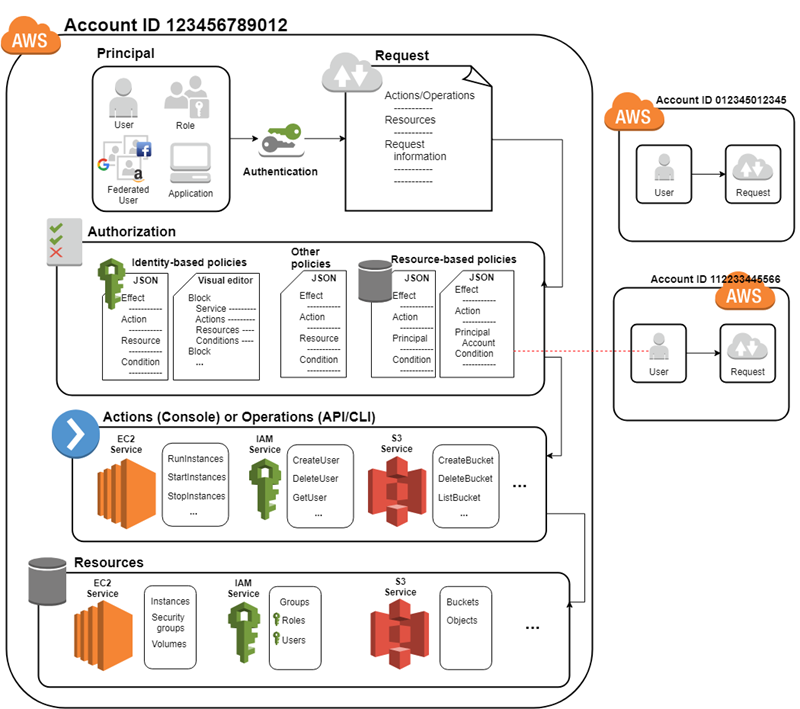
With AWS WAF you pay only for what you use. AWS WAF pricing is based on how many rules you deploy and how many web requests your web application receives. There are no upfront commitments.

You can deploy AWS WAF on either Amazon CloudFront as part of your CDN solution, the Application Load Balancer (ALB) that fronts your web servers or origin servers running on EC2, or Amazon API Gateway for your APIs.

## ****How Does IAM Work?****

The IAM workflow includes the following six elements:

1. A **principal** is an entity that can perform actions on an AWS resource. A user, a role or an application can be a principal.
2. **Authentication** is the process of confirming the identity of the principal trying to access an AWS product. The principal must provide its credentials or required keys for authentication.
3. **Request:** A principal sends a request to AWS specifying the action and which resource should perform it.
4. **Authorization:**By default, all resources are denied. IAM authorizes a request only if all parts of the request are allowed by a matching policy. After authenticating and authorizing the request, AWS approves the action.
5. **Actions:** After your request has been authenticated and authorized, AWS approves the actions or operations in your request. Operations are defined by a service, and include things that you can do to a resource, such as viewing, creating, editing, and deleting that resource
6. **Resources:** After AWS approves the operations in your request, they can be performed on the related resources within your account. A resource is an object that exists within a service.



**Resources:**

The user, group, role, policy, and identity provider objects that are stored in IAM. As with other AWS services, you can add, edit, and remove resources from IAM.

**Identities:**

The IAM resource objects that are used to identify and group. You can attach a policy to an IAM identity. These include users, groups, and roles.

**Entities:**

The IAM resource objects that AWS uses for authentication. These include users and roles. Roles can be assumed by IAM users and roles in your or another account. They can also be assumed by users federated through a web identity or SAML.

**Principals:**

A person or application that uses the AWS account root user, an IAM user, or an IAM role to sign in and make requests to AWS.

## ****Components of IAM:****

There are other basic components of IAM. First, we have the **user**; many users together form a**group**. **Policies** are the engines that allow or deny a connection based on policy. **Roles** are temporary credentials that can be assumed to an instance as needed.

### ****Users:****

An IAM user is an identity with an associated credential and permissions attached to it. This could be an actual person who is a user, or it could be an application that is a user. With IAM, you can securely manage access to AWS services by creating an IAM user name for each employee in your organization. Each IAM user is associated with only one AWS account. By default, a newly created user is not authorized to perform any action in AWS. The advantage of having one-to-one user specification is that you can individually assign permissions to each user.

### ****Groups:****

A collection of IAM users is an IAM group. You can use IAM groups to specify permissions for multiple users so that any permission applied to the group are applied to the individual users in that group as well. Managing groups is quite easy. You set permissions for the group, and those permissions are automatically applied to all the users in the group. If you add another user to the group, the new user will automatically inherit all the policies and the permissions already assigned to that group. This lessens the administrative burden.

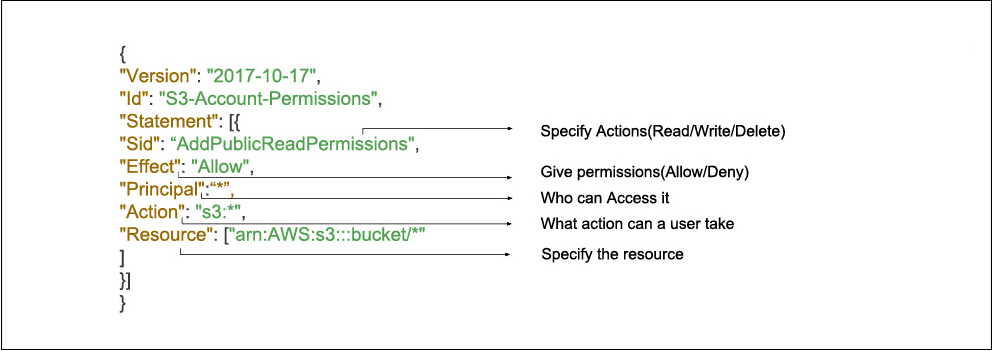
### ****Policies:****

An IAM policy sets permission and controls access to AWS resources. Policies are stored in AWS as JSON documents. Permissions specify who has access to the resources and what actions they can perform. For example, a policy could allow an IAM user to access one of the buckets in Amazon S3.

The policy would contain the following information:

1. Who can access it
2. What actions that user can take
3. Which AWS resources that user can access
4. When they can be accessed

In JSON format that would look like this:



{

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

"Id": "S3-Account-Permissions",

"Statement": [{

"Sid": "1",

"Effect": "Allow",

"Principal": {"AWS": ["arn:aws:iam::*ACCOUNT-ID-WITHOUT-HYPHENS*:root"]},

"Action": "s3:\*",

"Resource": [

"arn:aws:s3:::mybucket",

"arn:aws:s3:::mybucket/\*"

]

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

}]

}

The information in a statement is contained within a series of elements.

* **Version** – Specify the version of the policy language that you want to use. As a best practice, use the latest 2012-10-17 version.
* **Statement** – Use this main policy element as a container for the following elements. You can include more than one statement in a policy.
* **Sid** – Include an optional statement ID to differentiate between your statements.
* **Effect** – Use Allow or Deny to indicate whether the policy allows or denies access.
* **Principal** – Indicate the account, user, role, or federated user to which you would like to allow or deny access. If you are creating a policy to attach to a user or role, you cannot include this element. The principal is implied as that user or role.
* **Action** – Include a list of actions that the policy allows or denies.
* **Resource** – Specify a list of resources to which the actions apply.
* **Condition** (Optional) – Specify the circumstances under which the policy grants permission.

There are two types of policies: managed policies and inline policies.

1. A **managed policy** is a default policy that you attach to multiple entities (users, groups, and roles) in your AWS account. Managed policies, whether they are AWS-managed or customer-managed, are stand-alone identity-based policies attached to multiple users and/or groups.
2. **Inline policies** are policies that you create that are embedded directly into a single entity (user, group or role).

### ****Roles:****

An IAM role is a set of permissions that define what actions are allowed and denied by an entity in the AWS console. It is similar to a user in that it can be accessed by any type of entity (an individual or AWS service). Role permissions are temporary credentials.

For example, you might want to allow a mobile app to use AWS resources, but you do not want it to save the key, credential or password. Or you might want to give access to resources to a user who already has an identity defined outside of AWS, such as a user who already has Google or Facebook authentication. If you want to provide someone with a service or let someone access resources in your account, you can use roles for that purpose too. You also might want to grant temporary access to your account to a third party, such as a consultant or an auditor. They’re not permanent users, just users with temporary access to your environment.

## ****Features of IAM:****

IAM gives you the following features:

**Shared access to your AWS account**

You can grant other people permission to administer and use resources in your AWS account without having to share your password or access key.

**Granular permissions**

You can grant different permissions to different people for different resources. For example, you might allow some users complete access to Amazon Elastic Compute Cloud (Amazon EC2), Amazon Simple Storage Service (Amazon S3), Amazon DynamoDB, Amazon Redshift, and other AWS services. For other users, you can allow read-only access to just some S3 buckets, or permission to administer just some EC2 instances, or to access your billing information but nothing else.

**Secure access to AWS resources for applications that run on Amazon EC2**

You can use IAM features to securely provide credentials for applications that run on EC2 instances. These credentials provide permissions for your application to access other AWS resources. Examples include S3 buckets and DynamoDB tables.

**Multi-factor authentication (MFA)**

You can add two-factor authentication to your account and to individual users for extra security. With MFA you or your users must provide not only a password or access key to work with your account, but also a code from a specially configured device.

**Identity federation**

You can allow users who already have passwords elsewhere—for example, in your corporate network or with an internet identity provider—to get temporary access to your AWS account.

**Identity information for assurance**

If you use [AWS CloudTrail](https://aws.amazon.com/cloudtrail/), you receive log records that include information about those who made requests for resources in your account. That information is based on IAM identities.

**PCI DSS Compliance**

IAM supports the processing, storage, and transmission of credit card data by a merchant or service provider, and has been validated as being compliant with Payment Card Industry (PCI) Data Security Standard (DSS). For more information about PCI DSS, including how to request a copy of the AWS PCI Compliance Package, see [PCI DSS Level 1](https://aws.amazon.com/compliance/pci-dss-level-1-faqs/).

**Integrated with many AWS services**

For a list of AWS services that work with IAM, see [AWS Services That Work with IAM](https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_aws-services-that-work-with-iam.html).

**Eventually Consistent**

IAM, like many other AWS services, is [eventually consistent](https://wikipedia.org/wiki/Eventual_consistency). IAM achieves high availability by replicating data across multiple servers within Amazon's data centers around the world. If a request to change some data is successful, the change is committed and safely stored. However, the change must be replicated across IAM, which can take some time. Such changes include creating or updating users, groups, roles, or policies. We recommend that you do not include such IAM changes in the critical, high-availability code paths of your application. Instead, make IAM changes in a separate initialization or setup routine that you run less frequently. Also, be sure to verify that the changes have been propagated before production workflows depend on them. For more information, see [Changes That I Make Are Not Always Immediately Visible](https://docs.aws.amazon.com/IAM/latest/UserGuide/troubleshoot_general.html#troubleshoot_general_eventual-consistency).

**Free to use**

AWS Identity and Access Management (IAM) and AWS Security Token Service (AWS STS) are features of your AWS account offered at no additional charge. You are charged only when you access other AWS services using your IAM users or AWS STS temporary security credentials. For information about the pricing of other AWS products, see the [Amazon Web Services pricing page](https://aws.amazon.com/pricing/).

## Accessing IAM:

You can work with AWS Identity and Access Management in any of the following ways.

**AWS Management Console**

The console is a browser-based interface to manage IAM and AWS resources. For more information about accessing IAM through the console, see [The IAM Console and Sign-in Page](https://docs.aws.amazon.com/IAM/latest/UserGuide/console.html). For a tutorial that guides you through using the console, see [Creating Your First IAM Admin User and Group](https://docs.aws.amazon.com/IAM/latest/UserGuide/getting-started_create-admin-group.html).

**AWS Command Line Tools**

You can use the AWS command line tools to issue commands at your system's command line to perform IAM and AWS tasks. Using the command line can be faster and more convenient than the console. The command line tools are also useful if you want to build scripts that perform AWS tasks.

AWS provides two sets of command line tools: the [AWS Command Line Interface](https://aws.amazon.com/cli/) (AWS CLI) and the [AWS Tools for Windows PowerShell](https://aws.amazon.com/powershell/). For information about installing and using the AWS CLI, see the [AWS Command Line Interface User Guide](https://docs.aws.amazon.com/cli/latest/userguide/). For information about installing and using the Tools for Windows PowerShell, see the [AWS Tools for Windows PowerShell User Guide](https://docs.aws.amazon.com/powershell/latest/userguide/).

**AWS SDKs**

AWS provides SDKs (software development kits) that consist of libraries and sample code for various programming languages and platforms (Java, Python, Ruby, .NET, iOS, Android, etc.). The SDKs provide a convenient way to create programmatic access to IAM and AWS. For example, the SDKs take care of tasks such as cryptographically signing requests, managing errors, and retrying requests automatically. For information about the AWS SDKs, including how to download and install them, see the [Tools for Amazon Web Services](https://aws.amazon.com/tools/) page.

**IAM HTTPS API**

You can access IAM and AWS programmatically by using the IAM HTTPS API, which lets you issue HTTPS requests directly to the service. When you use the HTTPS API, you must include code to digitally sign requests using your credentials. For more information, see [Calling the API by Making HTTP Query Requests](https://docs.aws.amazon.com/IAM/latest/UserGuide/programming.html) and the [IAM API Reference](https://docs.aws.amazon.com/IAM/latest/APIReference/).