**IAM Roles**

## **IAM role** is an AWS identity with permission policies that determine what the identity can and cannot do in AWS.

Also, a role does not have standard long-term credentials such as a password or access keys associated with it. Instead, it provides you with temporary security credentials for your role session.

You can use roles to delegate access to users, applications, or services that don't normally have access to your AWS resources. For example, you might want to grant users in your AWS account access to resources they don't usually have, or grant users in one AWS account access to resources in another account. Or, you might want to grant access to your account to third parties so that they can perform an audit on your resources.

Usage:

* **AWS service role**   
  When you set up some AWS service environments, you must define a role for the service to assume. This service role must include all the permissions required for the service to access the AWS resources that it needs. Service roles provide access only within your account and cannot be used to grant access to services in other accounts.
* **AWS service role for an EC2 instance**   
  A special type of service role that an application running on an Amazon EC2 instance can assume to perform actions in your account.   
     
  This role is assigned to the EC2 instance when it is launched. Applications running on that instance can retrieve temporary security credentials and perform actions that the role allows.  
     
   An EC2 instance can only have **one** role attached at a time.
* **AWS service-linked role**   
   A unique type of service role that is linked directly to an AWS service. Service-linked roles are predefined by the service and include all the permissions that the service requires to call other AWS services on your behalf.
* **Role chaining**   
  Role chaining occurs when you use a role to assume a second role through the AWS CLI or API. For example, assume that User1 has permission to assume RoleA and RoleB. Additionally, RoleA has permission to assume RoleB. You can assume RoleA by using User1's long-term user credentials in the AssumeRole API operation. This operation returns RoleA short-term credentials. To engage in role chaining, you can use RoleA's short-term credentials to assume RoleB.
* **Delegation**   
  The granting of permissions to someone to allow access to resources that you control. Delegation involves setting up a trust between two accounts.
* **Federation** The creation of a trust relationship between an external identity provider and AWS. Users can sign in to a web identity provider, such as Login with Amazon, Facebook, Google, etc.
* **Federated user**   
  Instead of creating an IAM user, you can use existing identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as **federated users**. AWS assigns a role to a federated user when access is requested through an [identity provider](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_providers.html).
* **Trust policy** A JSON policy document in which you define the principals that you *trust* to assume the role.
* **Permissions policy**A permissions document in JSON format in which you define what actions and resources the role can use.
* **Permissions boundary**  
  An advanced feature in which you use policies to limit the maximum permissions that an identity-based policy can grant to a role.
* **Role for cross-account access**   
   A role that grants access to resources in one account to a trusted principal in a different account.

<https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles.html>

<https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_terms-and-concepts.html>