# Identity & Access Management

* Configure who uses AWS and their level of access to the AWS Console.
* Centralized control over AWS Account
* Share access for AWS Account
* Granular permissions for users / services
* Identity Federation – Facebook, LinkedIn and Active Directory- You can login to AWS with your corporate credentials.
* Multi-factor authentication – helps secure the account. Especially for root account
* Temporary access to users and services
* Setup password rotation policy
* Integration with other AWS services.
* Supports PCI-DSS compliance

### Critical Terms

* **Users**: End users
* **Groups**: Collection of users under one set of permissions
* **Roles**: Roles are created and assigned to AWS resources
* **Policies**: JSON Document that defines one or more permissions. These are attached to users, groups or roles

### IAM Features

* **IAM is a global service**. It is not region specific
* Root account
  + The email address you use to sign up for AWS
  + AWS recommends very limited usage of root account
  + Setup MFA on root account
  + root has complete admin access by default
* You can attach permissions to individual users and groups.
* New users are assigned Access Key ID and Secret Access keys when first created. (Used for CLI/API access, not console)
* Secret access key can be retrieved only once during user creation. In case you lose it then you can re-generate it.
* IAM Password policy can be set to access the admin console.
* New users have no permissions when first created. Everything has to be explicitly added.
* Power User Access allows Access to all AWS services except the management of groups and users within IAM.

### Manage AWS resources via

* Management console – Using username and password
* Rest APIs – Using Access Key ID and Secret Access Key
* AWS CLI - Using Access Key ID and Secret Access Key
* AWS SDK – various programming languages supported.

Using Access Key ID and Secret Access Key – can be used only via accessing programmatically. Akin to username and password used while accessing the console

### Security Token Services (STS)

* STS allows you to create temporary security credentials that grant trusted users access to your AWS resources.
* These can be active for a few minutes to a few hours
* Once expired, they can no longer be used to access resources
* When a STS API call is made, creds are returned with three components
  + Session Token
  + Access Key ID
  + Secret Access Key
* Since creds are temporary, you don’t have to rotate or revoke them
* Allows you to grant access to resources without having to create IAM identity for them
* Used for:
  + Identity federation
    - Enterprise ID federation (SAML for use with MS AD)
    - Web ID Federation: 3rd party providers such as Facebook, Google etc.
  + Roles for cross account Access
    - For organizations with multiple AWS accounts
  + Roles for EC2
    - Grant access to apps running on EC2 to access other AWS services without embedded credentials
* STS API Calls
  + AssumeRole\_ Cross Account delegation
  + AssumeRoleWithWebIdentity
  + AssumeRoleWithSAML
  + GetFederationToken: Federation thru custom ID Broker
  + GetSessionToken: Temp creds for users in untrusted environments