General IAM Concepts

AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources.

You use IAM to control who is authenticated (signed in) and authorized (has permissions) to use resources.

IAM makes it easy to provide multiple users secure access to AWS resources.

When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account.

This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account.

IAM can be used to manage:

- Users.
- Groups.
- Access policies.
- Roles.
- User credentials.
- User password policies.
- Multi-factor authentication (MFA).
- API keys for programmatic access (CLI).

IAM provides the following features:

- Shared access to your AWS account.
- Granular permissions.
- Secure access to AWS resources for application that run on Amazon EC2.
- Multi-Factor authentication.
- Identity federation.
- Identity information for assurance.
- PCI DSS compliance.
- Integrated with may AWS services.
- Eventually consistent.
- Free to use.

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

- AWS Management Console.
- AWS Command Line Tools.
- AWS SDKs.
- IAM HTTPS API.

By default new users are created with NO access to any AWS services – they can only login to the AWS console.

Permission must be explicitly granted to allow a user to access an AWS service.

IAM users are individuals who have been granted access to an AWS account.

Each IAM user has three main components:

- A user-name.
- A password.
- Permissions to access various resources.

You can apply granular permissions with IAM.

You can assign users individual security credentials such as access keys, passwords, and multi-factor authentication devices.

IAM is not used for application-level authentication.

Identity Federation (including AD, Facebook etc.) can be configured allowing secure access to resources in an AWS account without creating an IAM user account.

Multi-factor authentication (MFA) can be enabled/enforced for the AWS account and for individual users under the account.

MFA uses an authentication device that continually generates random, six-digit, single-use authentication codes.

You can authenticate using an MFA device in the following two ways:

- Through the **AWS Management Console** – the user is prompted for a user name, password and authentication code.
- Using the **AWS API** – restrictions are added to IAM policies and developers can request temporary security credentials and pass MFA parameters in their AWS STS API requests.
- Using the **AWS CLI** by obtaining temporary security credentials from STS (aws sts get-session-token).

It is a best practice to always setup multi-factor authentication on the root account.

IAM is universal (global) and does not apply to regions.

IAM replicates data across multiple data centres around the world.

The “root account” is the account created when you setup the AWS account. It has complete Admin access and is the only account that has this access by default.

It is a best practice to avoid using the root account for anything other than billing.

Power user access allows all permissions except the management of groups and users in IAM.

Temporary security credentials consist of the AWS access key ID, secret access key, and security token.

IAM can assign temporary security credentials to provide users with temporary access to services/resources.

To sign-in you must provide your account ID or account alias in addition to a user name and password.

The sign-in URL includes the account ID or account alias, e.g:

https://***My_AWS_Account_ID***.signin.aws.amazon.com/console/.

Alternatively, you can sign-in at the following URL and enter your account ID or alias manually:

https://console.aws.amazon.com/

IAM integrates with many different AWS services.

Authentication Methods

Console password:

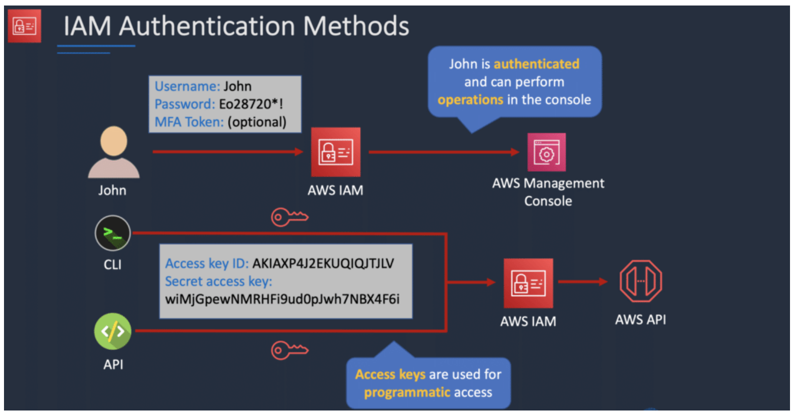
- A password that the user can enter to sign in to interactive sessions such as the AWS Management Console.
- You can allow users to change their own passwords.
- You can allow selected IAM users to change their passwords by disabling the option for all users and using an IAM policy to grant permissions for the selected users.

Access Keys:

- A combination of an **access key ID** and a **secret access key.**
- You can assign two active access keys to a user at a time.
- These can be used to make programmatic calls to AWS when using the **API** in program code or at a command prompt when using the **AWS CLI** or the **AWS PowerShell** tools.
- You can create, modify, view or rotate access keys.
- When created IAM returns the access key ID and secret access key.
- The secret access is returned only at creation time and if lost a new key must be created.
- Ensure access keys and secret access keys are stored securely.
- Users can be given access to change their own keys through IAM policy (not from the console).
- You can disable a user’s access key which prevents it from being used for API calls.

Server certificates:

- SSL/TLS certificates that you can use to authenticate with some AWS services.
- AWS recommends that you use the AWS Certificate Manager (ACM) to provision, manage and deploy your server certificates.
- Use IAM only when you must support HTTPS connections in a region that is not supported by ACM.


IAM Authentication Methods
Username: John
Password: Eo28720*!
MFA Token: (optional)
John is authenticated and can perform operations in the console
John
AWS IAM
AWS Management Console
CLI
Access key ID: AKIAXP4J2EKUQIQJTJLV
Secret access key: wiMjGpewNMRHFi9ud0pJwh7NBX4F6i
AWS IAM
AWS API
API
Access keys are used for programmatic access

IAM Users

An IAM user is an entity that represents a person or service.

Can be assigned:

- An access key ID and secret access key for programmatic access to the AWS API, CLI, SDK, and other development tools.
- A password for access to the management console.

By default, users cannot access anything in your account.

The account root user credentials are the email address used to create the account and a password.

The root account has full administrative permissions and these cannot be restricted.

Best practice for root accounts:

- Don't use the root user credentials.
- Don't share the root user credentials.
- Create an IAM user and assign administrative permissions as required.
- Enable MFA.

IAM users can be created to represent applications and these are known as "service accounts".

You can have up to 5000 users per AWS account.

Each user account has a friendly name and an ARN which uniquely identifies the user across AWS.

A unique ID is also created which is returned only when you create the user using the API, Tools for Windows PowerShell or the AWS CLI.

You should create individual IAM accounts for users (best practice not to share accounts).

The Access Key ID and Secret Access Key are not the same as a password and cannot be used to login to the AWS console.

The Access Key ID and Secret Access Key can only be used once and must be regenerated if lost.

A password policy can be defined for enforcing password length, complexity etc. (applies to all users).

You can allow or disallow the ability to change passwords using an IAM policy.

Access keys and passwords should be changed regularly.

Groups

Groups are collections of users and have policies attached to them.

A group is not an identity and cannot be identified as a principal in an IAM policy.

Use groups to assign permissions to users.

Use the principle of least privilege when assigning permissions.

You cannot nest groups (groups within groups).

Roles

Roles are created and then “assumed” by trusted entities and define a set of permissions for making AWS service requests.

With IAM Roles you can delegate permissions to resources for users and services without using permanent credentials (e.g. user name and password).

IAM users or AWS services can assume a role to obtain temporary security credentials that can be used to make AWS API calls.

You can delegate using roles.

There are no credentials associated with a role (password or access keys).

IAM users can temporarily assume a role to take on permissions for a specific task.

A role can be assigned to a federated user who signs in using an external identity provider.

Temporary credentials are primarily used with IAM roles and automatically expire.

Roles can be assumed temporarily through the console or programmatically with the **AWS CLI**, **Tools for Windows PowerShell** or **API.**

IAM roles with EC2 instances:

- IAM roles can be used for granting applications running on EC2 instances permissions to AWS API requests using instance profiles.
- Only one role can be assigned to an EC2 instance at a time.
- A role can be assigned at the **EC2 instance creation time or at any time afterwards.**
- When using the AWS CLI or API instance profiles must be created manually (it's automatic and transparent through the console).
- Applications retrieve temporary security credentials from the instance metadata.

Role Delegation:

- Create an IAM role with two policies:
  - Permissions policy – grants the user of the role the required permissions on a resource.
  - Trust policy – specifies the trusted accounts that are allowed to assume the role.
- Wildcards (*) cannot be specified as a principal.
- A permissions policy must also be attached to the user in the trusted account.

Policies

Policies are documents that define permissions and can be applied to users, groups and roles.

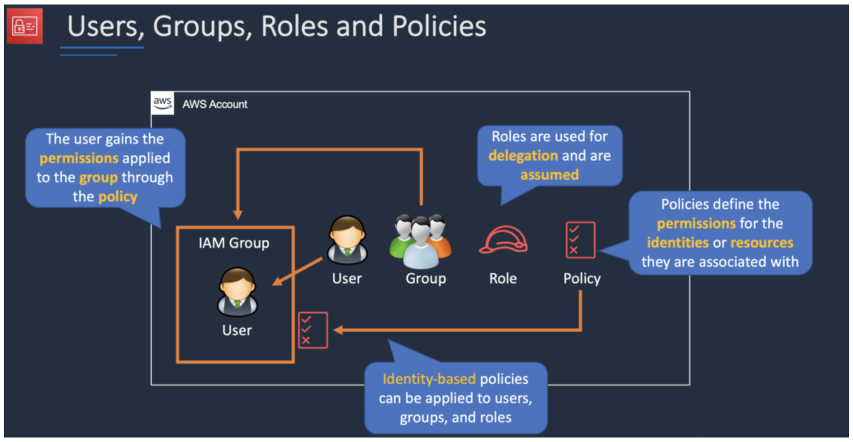
Policy documents are written in JSON (key value pair that consists of an attribute and a value).

All permissions are implicitly denied by default.

The most restrictive policy is applied.

The IAM policy simulator is a tool to help you understand, test, and validate the effects of access control policies.

The Condition element can be used to apply further conditional logic.


Users, Groups, Roles and Policies
AWS Account
The user gains the permissions applied to the group through the policy
Roles are used for delegation and are assumed
Policies define the permissions for the identities or resources they are associated with
IAM Group
User
User
Group
Role
Policy
Identity-based policies can be applied to users, groups, and roles

STS

The AWS Security Token Service (STS) is a web service that enables you to request temporary, limited-privilege credentials for IAM users or for users that you authenticate (federated users).

Temporary security credentials work almost identically to long-term access key credentials that IAM users can use, with the following differences:

- Temporary security credentials are short-term.
- They can be configured to last anywhere from a few minutes to several hours.
- After the credentials expire, AWS no longer recognizes them or allows any kind of access to API requests made with them.
- Temporary security credentials are not stored with the user but are generated dynamically and provided to the user when requested.
- When (or even before) the temporary security credentials expire, the user can request new credentials, as long as the user requesting them still has permission to do so.

Advantages of STS are:

- You do not have to distribute or embed long-term AWS security credentials with an application.
- You can provide access to your AWS resources to users without having to define an AWS identity for them (temporary security credentials are the basis for IAM Roles and ID Federation).
- The temporary security credentials have a limited lifetime, so you do not have to rotate them or explicitly revoke them when they're no longer needed.
- After temporary security credentials expire, they cannot be reused (you can specify how long the credentials are valid for, up to a maximum limit)

Users can come from three sources.

Federation (typically AD):

- Uses SAML 2.0.
- Grants temporary access based on the users AD credentials.
- Does not need to be a user in IAM.
- Single sign-on allows users to login to the AWS console without assigning IAM credentials.

Federation with Mobile Apps:

- Use Facebook/Amazon/Google or other OpenID providers to login.

Cross Account Access:

- Lets users from one AWS account access resources in another.
- To make a request in a different account the resource in that account must have an attached resource-based policy with the permissions you need.
- Or you must assume a role (identity-based policy) within that account with the permissions you need.

IAM Best Practices

Lock away the AWS root user access keys.

Create individual IAM users.

Use AWS defined policies to assign permissions whenever possible.

Use groups to assign permissions to IAM users.

Grant least privilege.

Use access levels to review IAM permissions.

Configure a strong password policy for users.

Enable MFA.

Use roles for applications that run on AWS EC2 instances.

Delegate by using roles instead of sharing credentials.

Rotate credentials regularly.

Remove unnecessary credentials.

Use policy conditions for extra security.

Monitor activity in your AWS account.

AWS Security

As an AWS customer you inherit all the best practices of AWS policies, architecture, and operational processes.

The AWS Cloud enables a shared responsibility model.

AWS manages security OF the cloud, you are responsible for security IN the cloud.

You retain control of the security you choose to implement to protect your own content, platform, applications, systems, and networks no differently than you would in an on-site data center.

Benefits of AWS Security

- **Keep Your Data Safe** – the AWS infrastructure puts strong safeguards in place to help.
- **Protect your privacy** – All data is stored in highly secure AWS data centers.
- **Meet Compliance Requirements** – AWS manages dozens of compliance programs in its infrastructure. This means that segments of your compliance have already been completed.
- **Save Money** – cut costs by using AWS data centers. Maintain the highest standard of s security without having to manage your own facility.
- **Scale Quickly** – security scales with your AWS Cloud usage. No matter the size of your business, the AWS infrastructure is designed to keep your data safe.

Compliance

AWS Cloud Compliance enables you to understand the robust controls in place at AWS to maintain security and data protection in the cloud.

As systems are built on top of AWS Cloud infrastructure, compliance responsibilities will be shared.

Compliance programs include:

-
    -
    - Certifications / attestations.
    - Laws, regulations, and privacy.
    - Alignments / frameworks.

AWS Artifact

AWS Artifact is your go-to, central resource for compliance-related information that matters to you.

It provides on-demand access to AWS' security and compliance reports and select online agreements.

Reports available in AWS Artifact include our Service Organization Control (SOC) reports, Payment Card Industry (PCI) reports, and certifications from accreditation bodies across geographies and compliance verticals that validate the implementation and operating effectiveness of AWS security controls.

Agreements available in AWS Artifact include the Business Associate Addendum (BAA) and the Nondisclosure Agreement (NDA).

Amazon GuardDuty

Amazon GuardDuty offers threat detection and continuous security monitoring for malicious or unauthorized behavior to help you protect your AWS accounts and workloads.

Intelligent threat detection service.

Detects account compromise, instance compromise, malicious reconnaissance, and bucket compromise.

Continuous monitoring for events across:

- AWS CloudTrail Management Events.
- AWS CloudTrail S3 Data Events.
- Amazon VPC Flow Logs.
- DNS Logs.

AWS WAF & AWS Shield

WAF:

-
  -
    - AWS WAF is a web application firewall.
    - Protects against common exploits that could compromise application availability, compromise security or consume excessive resources.
    - WAF lets you create rules to filter web traffic based on conditions that include IP addresses, HTTP headers and body, or custom URIs.
    - WAF makes it easy to create rules that block common web exploits like SQL injection and cross site scripting.
    - The rules are known as Web ACLs.

Shield:

-
  -
    - AWS Shield is a managed Distributed Denial of Service (DDoS) protection service.
    - Safeguards web application running on AWS with always-on detection and automatic inline mitigations.
    - Helps to minimize application downtime and latency.
    - Two tiers – Standard and Advanced.

AWS Key Management Service

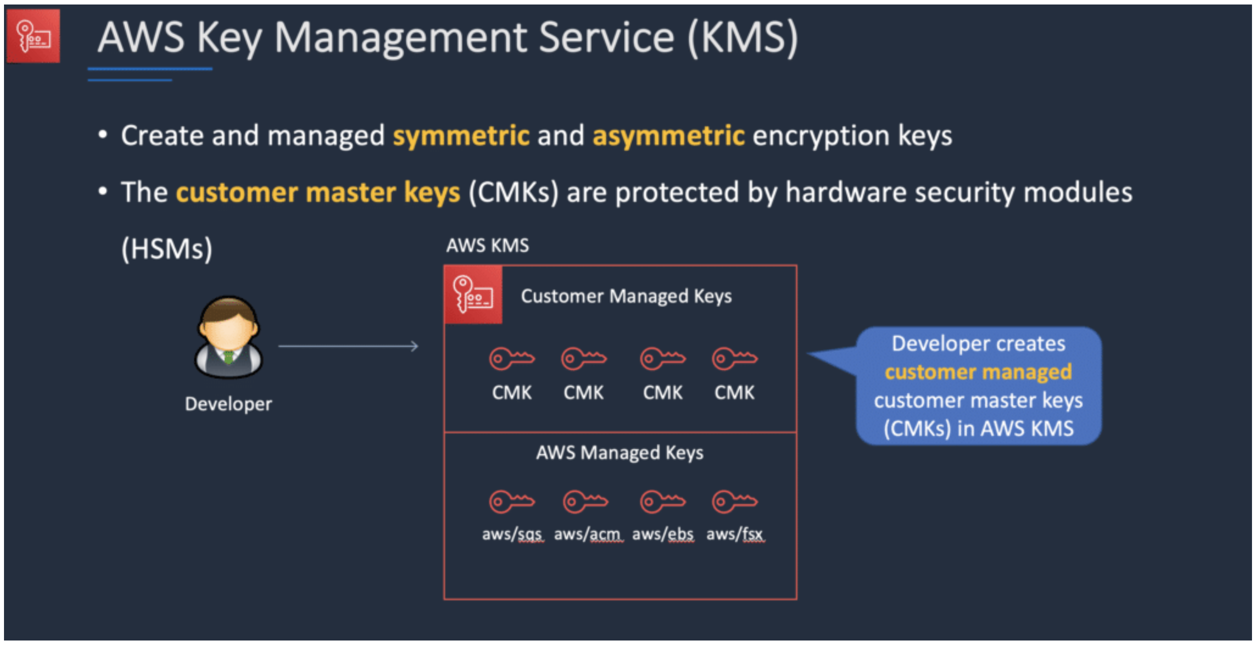
AWS Key Management Service gives you centralized control over the encryption keys used to protect your data.

You can create, import, rotate, disable, delete, define usage policies for, and audit the use of encryption keys used to encrypt your data.

AWS Key Management Service is integrated with most other AWS services making it easy to encrypt the data you store in these services with encryption keys you control.

AWS KMS is integrated with AWS CloudTrail which provides you the ability to audit who used which keys, on which resources, and when.

AWS KMS enables developers to easily encrypt data, whether through 1-click encryption in the AWS Management Console, or using the AWS SDK to easily add encryption in their application code.



AWS Key Management Service (KMS)

- Create and managed **symmetric** and **asymmetric** encryption keys
- The **customer master keys** (CMKs) are protected by hardware security modules (HSMs)

AWS CloudHSM

AWS CloudHSM is a cloud-based hardware security module (HSM) that enables you to easily generate and use your own encryption keys on the AWS Cloud.

With CloudHSM, you can manage your own encryption keys using FIPS 140-2 Level 3 validated HSMs.

CloudHSM offers you the flexibility to integrate with your applications using industry-standard APIs, such as PKCS#11, Java Cryptography Extensions (JCE), and Microsoft CryptoNG (CNG) libraries.

AWS Inspector and AWS Trusted Advisor

AWS Inspector:

-
  - ○
  - Inspector is an automated security assessment service that helps improve the security and compliance of applications deployed on AWS.
  - Inspector automatically assesses applications for vulnerabilities or deviations from best practices.
  - Uses an agent installed on EC2 instances.
  - Instances must be tagged.

AWS Trusted Advisor:

-
  - ○
  - Trusted Advisor is an online resource that helps to reduce cost, increase performance and improve security by optimizing your AWS environment.
  - Trusted Advisor provides real time guidance to help you provision your resources following best practices.
  - Advisor will advise you on Cost Optimization, Performance, Security, and Fault Tolerance.

Trusted Advisor scans your AWS infrastructure and compares is to AWS best practices in five categories:

-
  - ○
  - Cost Optimization.
  - Performance.
  - Security.
  - Fault Tolerance.
  - Service Limits.

Trusted Advisor comes in two versions.

Core Checks and Recommendations (free):

-
  - ○
  - Access to the 7 core checks to help increase security and performance.
  - Checks include S3 bucket permissions, Security Groups, IAM use, MFA on root account, EBS public snapshots, RDS public snapshots.

Full Trusted Advisor Benefits (business and enterprise support plans):

- Full set of checks to help optimize your entire AWS infrastructure.
- Advises on security, performance, cost, fault tolerance and service limits.
- Additional benefits include weekly update notifications, alerts, automated actions with CloudWatch and programmatic access using the AWS Support API.

Penetration Testing

Penetration testing is the practice of testing one's own application's security for vulnerabilities by simulating an attack.

AWS allows penetration testing. There is a limited set of resources on which penetration testing can be performed.

You do not need permission to perform penetration testing against the following services:

- Amazon EC2 instances, NAT Gateways, and Elastic Load Balancers.
- Amazon RDS.
- Amazon CloudFront.
- Amazon Aurora.
- Amazon API Gateways.
- AWS Lambda and Lambda Edge functions.
- Amazon Lightsail resources.
- Amazon Elastic Beanstalk environments.

You can read the full vulnerability and penetration testing support policy here.

In case an account is or may be compromised, AWS recommend that the following steps are taken:

1. Change your AWS root account password.
2. Change all IAM user's passwords.
3. Delete or rotate all programmatic (API) access keys.
4. Delete any resources in your account that you did not create.
5. Respond to any notifications you received from AWS through the AWS Support Center and/or contact AWS Support to open a support case.

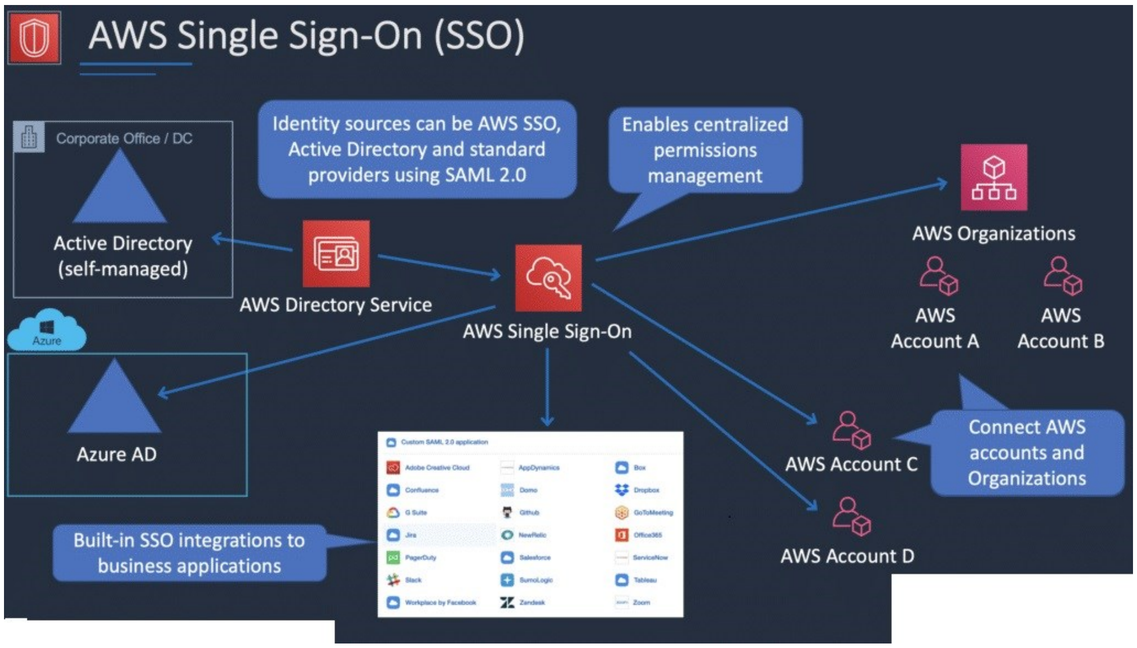
AWS Single Sign-On (AWS SSO)

AWS Single Sign-On is a cloud-based single sign-on (SSO) service that makes it easy to centrally manage SSO access to all of your AWS accounts and cloud applications.

It helps you manage SSO access and user permissions across all your AWS accounts in AWS Organizations.

AWS SSO also helps you manage access and permissions to commonly used third-party software as a service (SaaS) applications, AWS SSO-integrated applications as well as custom applications that support Security Assertion Markup Language (SAML) 2.0.

AWS SSO includes a user portal where your end-users can find and access all their assigned AWS accounts, cloud applications, and custom applications in one place.



Amazon Cognito

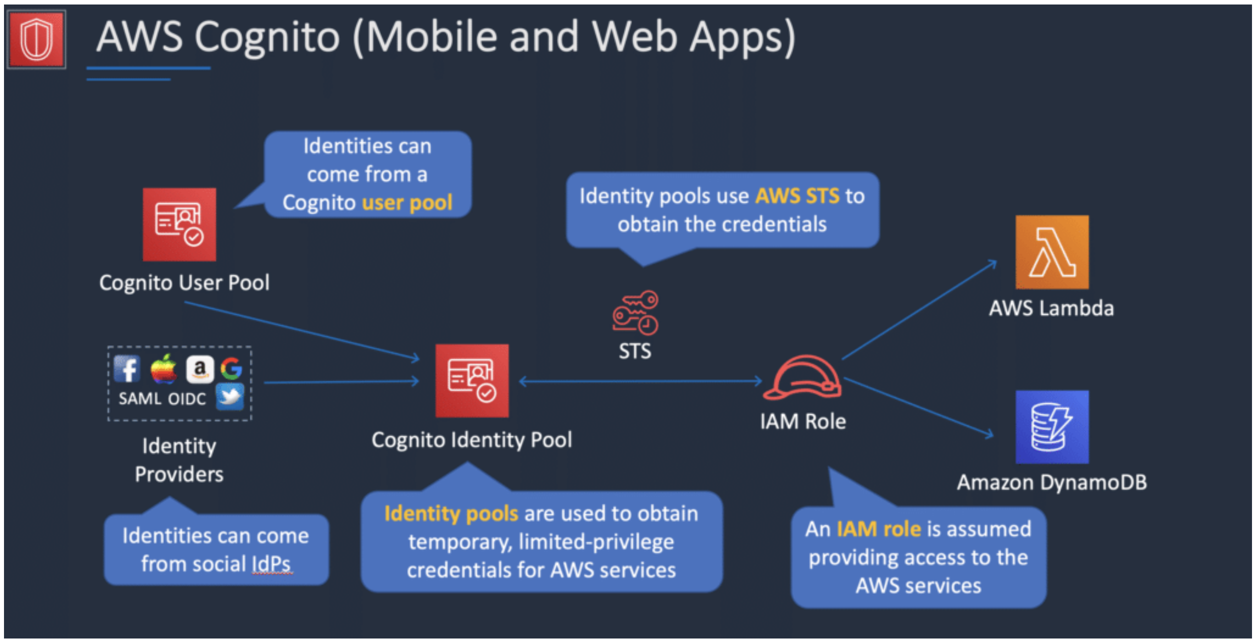
Amazon Cognito lets you add user sign-up, sign-in, and access control to your web and mobile apps quickly and easily.

Amazon Cognito scales to millions of users and supports sign-in with social identity providers, such as Apple, Facebook, Google, and Amazon, and enterprise identity providers via SAML 2.0 and OpenID Connect.

The two main components of AWS Cognito are user pools and identity pools:

- User pools are user directories that provide sign-up and sign-in options for your app users.
- Identity pools enable you to grant your users access to other AWS services.

You can use identity pools and user pools separately or together.


AWS Cognito (Mobile and Web Apps)
Identities can come from a Cognito user pool
Cognito User Pool
Identity pools use AWS STS to obtain the credentials
STS
AWS Lambda
SAML OIDC
Identity Providers
Cognito Identity Pool
IAM Role
Amazon DynamoDB
Identities can come from social IdPs
Identity pools are used to obtain temporary, limited-privilege credentials for AWS services
An IAM role is assumed providing access to the AWS services

AWS Directory Services

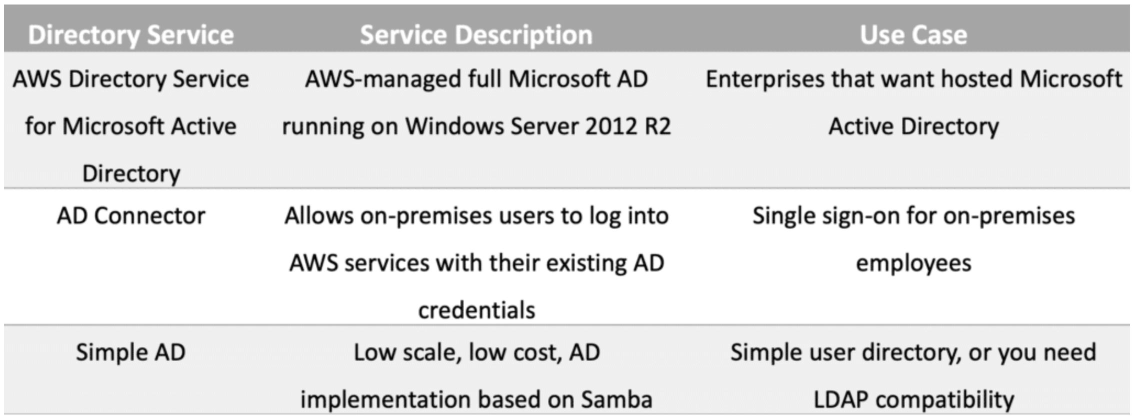
AWS provide a number of directory types.

The following three types currently feature on the exam and will be covered on this page:

- Active Directory Service for Microsoft Active Directory.
- Simple AD.
- AD Connector.

As an alternative to the AWS Directory service you can build your own Microsoft AD DCs in the AWS cloud (on EC2).

The table below summarises the directory services covered on this page as well as a couple of others, and provides some typical use cases:



Directory Service	Service Description	Use Case
AWS Directory Service for Microsoft Active Directory	AWS-managed full Microsoft AD running on Windows Server 2012 R2	Enterprises that want hosted Microsoft Active Directory
AD Connector	Allows on-premises users to log into AWS services with their existing AD credentials	Single sign-on for on-premises employees
Simple AD	Low scale, low cost, AD implementation based on Samba	Simple user directory, or you need LDAP compatibility

AWS Systems Manager Parameter Store

Provides secure, hierarchical storage for configuration data management and secrets management.

It is highly scalable, available, and durable.

You can store data such as passwords, database strings, and license codes as parameter values.

You can store values as plaintext (unencrypted data) or ciphertext (encrypted data).

You can then reference values by using the unique name that you specified when you created the parameter.

AWS Secrets Manager

Similar to Parameter Store.

Allows native and automatic rotation of keys.

Fine-grained permissions.

Central auditing for secret rotation.