Unit4
Identity Access Management (IAM)

AWS IAM:

- **AWS** Identity and Access Management (**IAM**) is a web service that helps you securely control access to **AWS** resources.
- AWS Identity and Access Management (IAM) enable you to manage access to AWS services and resources securely. Using IAM, you can create and manage AWS users and groups, and use permissions to allow and deny their access to AWS resources.
- IAM is a feature of your AWS account offered at no additional charge. You will be charged only for use of other AWS services by your users.

Key features of IAM:

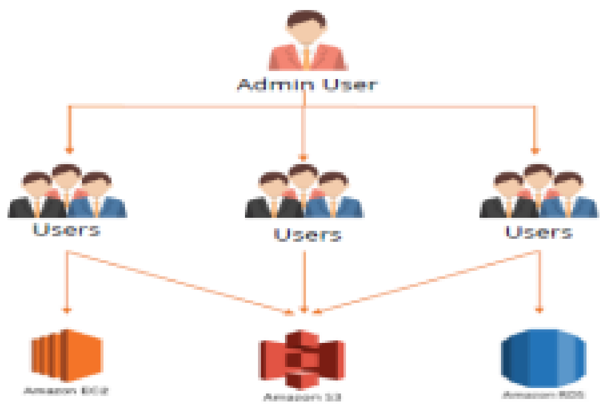
1. Shared Access to your Account
2. Granular Permissions
3. Secure Access to AWS Resources
4. Identity Federation
5. Identity Information for Assurance
6. Payment Card Industry (PCI) Data Security Standard (DSS) Compliance
   · Password Policy
7. Multi Factor Authentication (MFA)

IAM features

IAM gives you the following features:

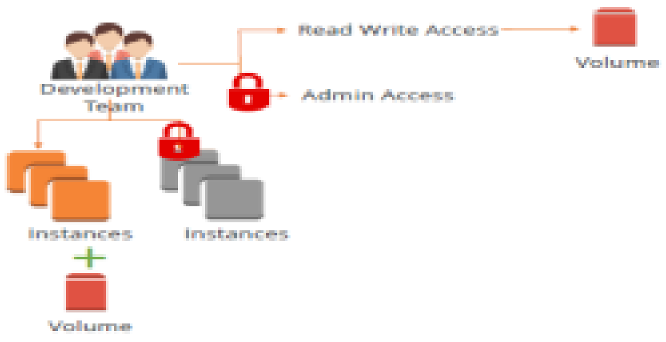
1. **Shared access to your AWS account**

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

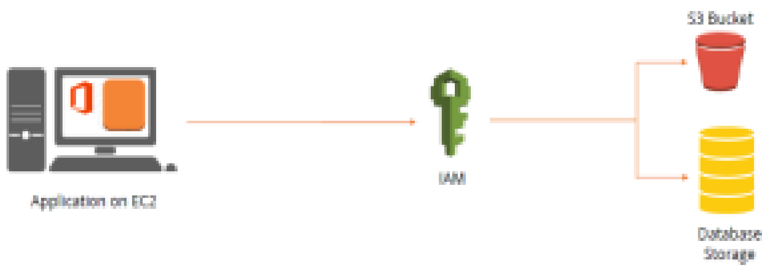


2. **Granular permissions**

   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.

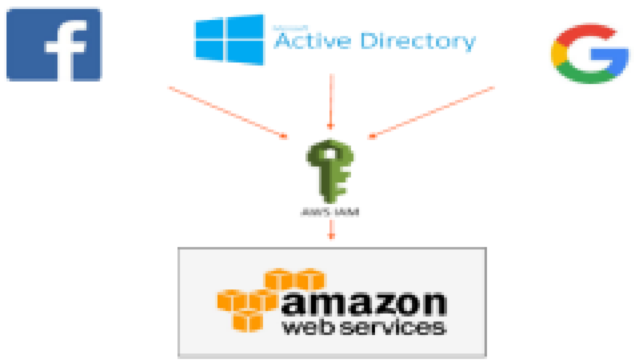


3. **Secure access to AWS resources for applications that run on Amazon EC2** You can use IAM features to securely provide credentials for applications that run on EC2 instances. These credentials provide permissions for your application to access other AWS resources. Examples include S3 buckets and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



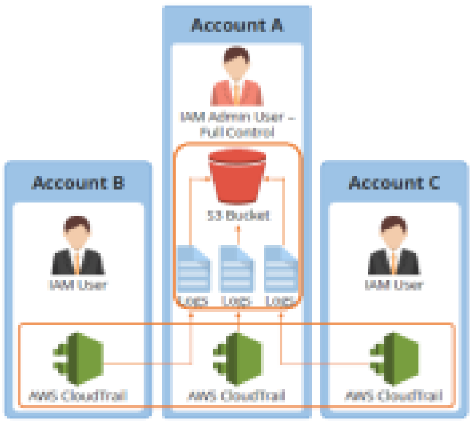
4. **Identity federation**

   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.



5. **Identity information for assurance**

   If you use AWS Cloud Trail, you receive log records that include information about those who made requests for resources in your account. That information is based on IAM identities.

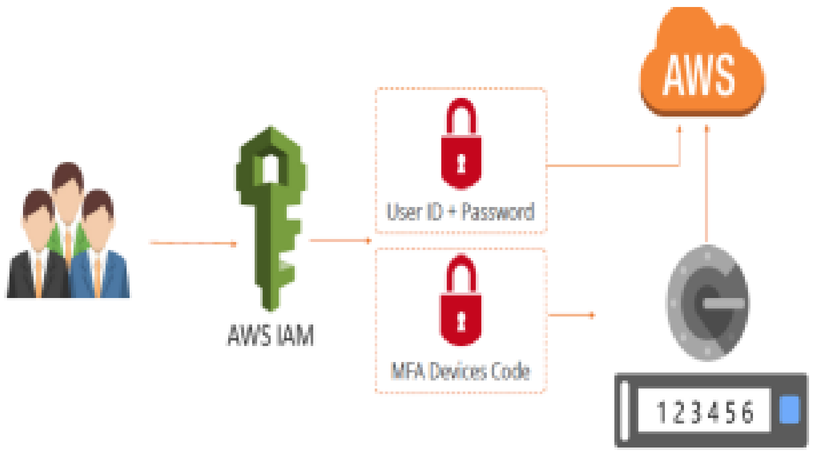


6. **PCI DSS Compliance**

   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).

7. **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. If you already use a FIDO security key with other services, and it has an AWS supported configuration, you can use WebAuthn for MFA security.



8. **Eventually Consistent**

   IAM, like many other AWS services, is eventually consistent. IAM achieves high availability by replicating data across multiple servers within Amazon's data canters 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.

9. **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.

<u>IAM Policy and its types:</u>

- A **policy** is an entity that, when attached to an identity or resource, defines their permissions.
- **Policies** are stored in **AWS** as JSON documents and are attached to principals as identity-based **policies** in **IAM**.
- You can attach an identity-based **policy** to a principal (or identity), such as an **IAM** group, user, or **role**.
- There are around 200 predefined policies available for you to choose from.

AWS Policies are of two kinds:

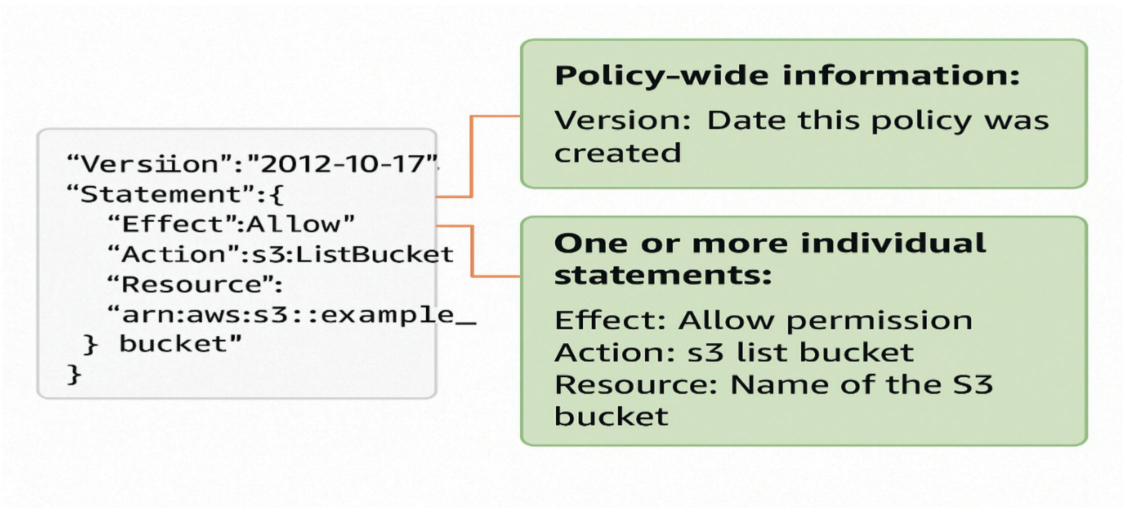
1. **Identity-based policies:** The identity-based policy is the one that can be attached directly with AWS identities like user, group or a role. IAM policy is an example of that. These policies can be AWS managed or a customer managed.
2. **Resource-based policies:** Resource-based policies are the ones which can be directly attached to the AWS resource like S3(called Amazon S3 bucket policy). Resource-based policies are available only for certain services.

Ex:

- AdministratorAccesspolicy provides full access to AWS services and resources.
- AmazonEC2FullAccess policy provides AWS Directory Service user or group's full access to the Amazon EC2 services and resources.
- AmazonS3ReadOnlyAccess policy provides read-only access to all buckets using the AWS Management Console.

JSON:

• AWS policies are written using JavaScript Object Notation (JSON).



```
"Version":"2012-10-17"
"Statement":{
   "Effect":Allow"
   "Action":s3:ListBucket
   "Resource":
   "arn:aws:s3::example_
 }  bucket"
}
```

Policy-wide information:
Version: Date this policy was created
One or more individual statements:
Effect: Allow permission
Action: s3 list bucket
Resource: Name of the S3 bucket

Password Policy:

IAM allows you to define password strength and rotation policies.

• You can use a password policy to do these things:

• Set a minimum password length.

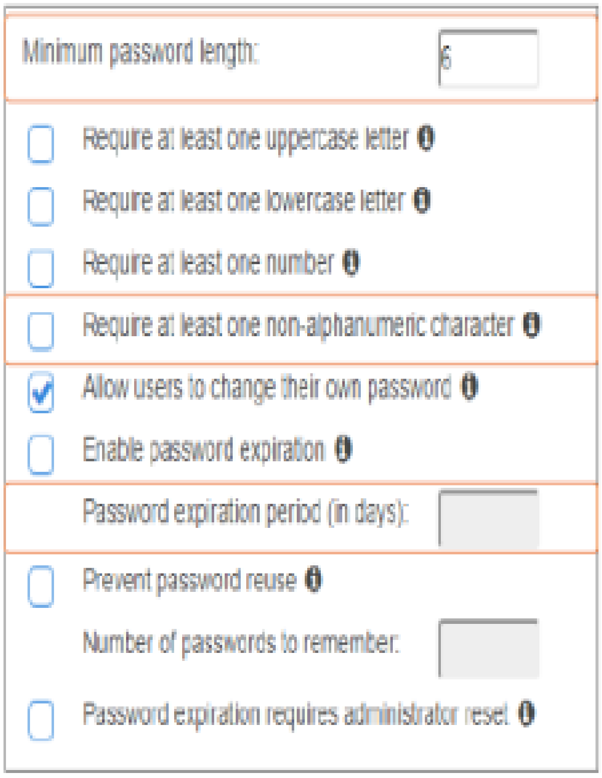
• Require specific character types, including uppercase letters, lowercase letters, numbers, and non-alphanumeric characters. Be sure to remind your users that passwords are case sensitive.

• Allow all IAM users to change their own passwords.

• Require IAM users to change their password after a specified period of time (enable password expiration).

• Prevent IAM users from reusing previous passwords.

• Force IAM users to contact an account administrator when the user has allowed his or her password to expire.



Minimum password length: 6

- [ ] Require at least one uppercase letter
- [ ] Require at least one lowercase letter
- [ ] Require at least one number
- [ ] Require at least one non-alphanumeric character
- [✓] Allow users to change their own password
- [ ] Enable password expiration

Password expiration period (in days):

- [ ] Prevent password reuse

Number of passwords to remember:

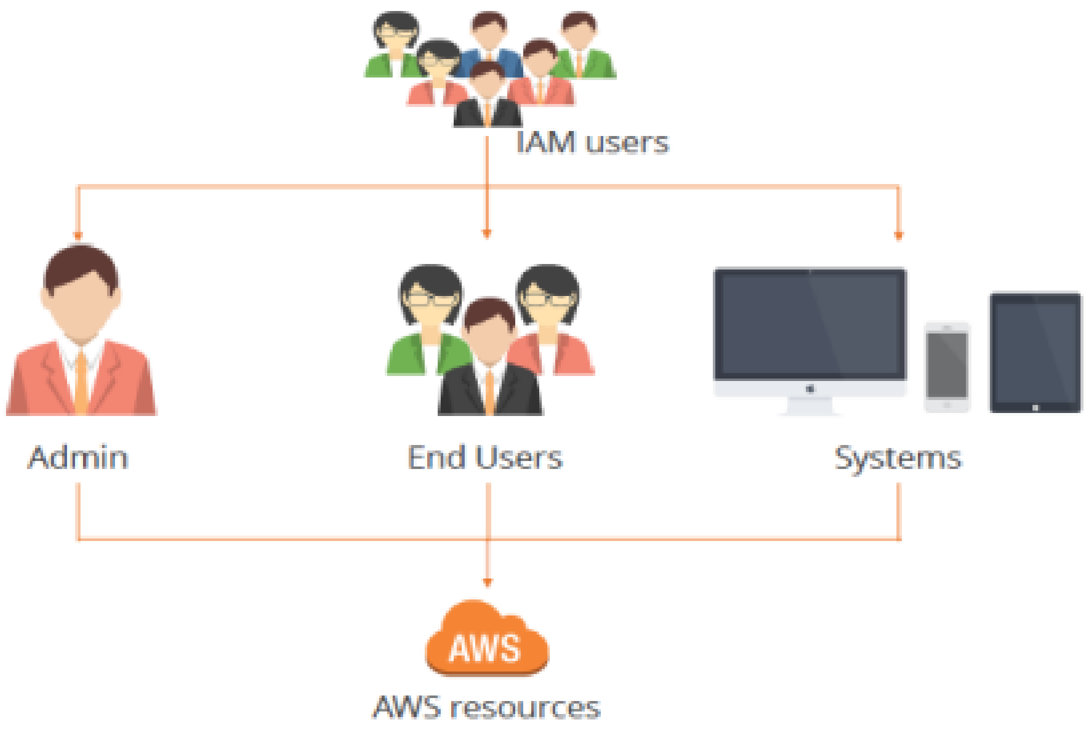
- [ ] Password expiration requires administrator reset

IAM Users:

- An **IAM User** is an entity created in AWS that provides a way to interact with AWS resources.
- The main purpose of **IAM Users** is that they can sign in to the AWS Management Console and can make requests to the AWS services.
- The newly created **IAM users** have no password and no access key. If a user wants to use the AWS resources using the AWS Management Console, you need to create the user password.
- The security of the user's credentials can be enhanced by using the feature, i.e., Multi-Factor Authentication.
- The newly created IAM Users do not have permissions, i.e., they are not authorized to access the AWS resources.
- An advantage of using individual IAM Users is that you can assign the

permissions individually. You can even assign the administrative permissions, who can administer your AWS resources and also administer other IAM Users.

- Users are defined as the people or systems that use your AWS resources.



<u>Security Credentials of IAM Users:</u>

You can access AWS in different ways depending on the user credentials:

Email address and password

• They are created when you sign up to use AWS

• They are used to sign in to AWS web pages

IAM user name and password

• They allow multiple individuals or applications access to your AWS account

• Individuals use their user names and passwords to sign in

Multi-Factor Authentication (MFA)

With AWS MFA enabled, users are prompted for a user name and password and for an authentication code from an MFA device

Access keys

• They consist of an access key and a secret access key

• They use access keys to sign programmatic requests.

Key pairs

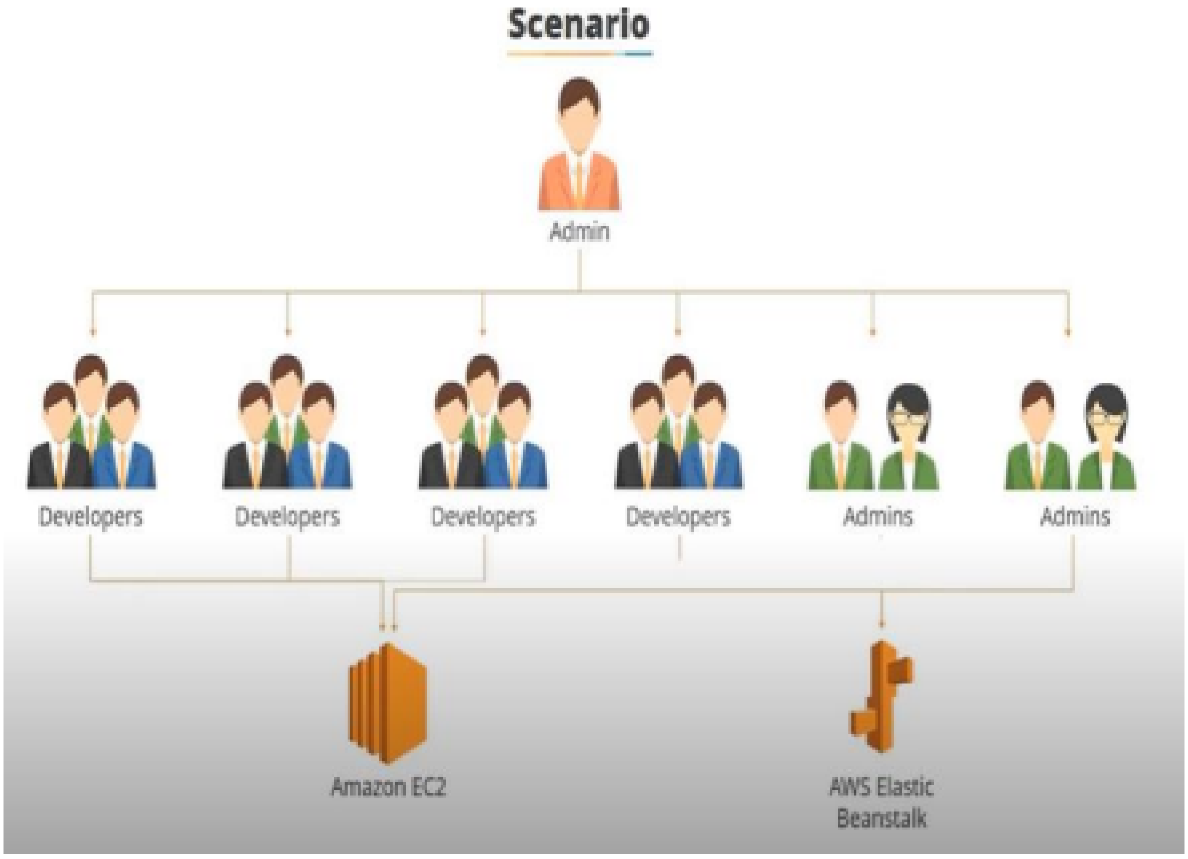
- They consist of a public and private key
- A private key is used to create a digital signature
    - AWS uses the corresponding public key to validate the signature

IAM Groups:

· An IAM Group is a collection of users.

· Group specifies the permission for a collection of users, and it also makes it possible to manage the permissions easily for those users.

· You created a group known as Admin and assigned the permissions to the group that administrators typically need. Any user joins the admin group; then the user will have all the permissions that are assigned to the group. If a new user joins the organization, then he should have administrator privileges, and you can assign the appropriate permissions by adding him to the group. If a person changes his job profile, instead of editing his permissions, you can remove him from a group and add him to the group.



Characteristics of IAM Group

• A group is a collection of users, and a user can also belong to multiple groups. •
Groups cannot be nested, i.e., a group cannot contain another group.
• No default group that automatically includes all the users in AWS account. If you want a group like this, create a group and then add the users in a group.
• There is a limit to the number of groups that you can have and also have a limit to the number of groups that a user can belong to.
• User **policy size** cannot exceed 2,048 characters. **Role policy size** cannot exceed 10,240 characters.

IAM Roles:

- A role is a set of permissions that grant access to actions and resources in AWS. These permissions are attached to the role, not to an IAM User or a group.
- An IAM User can use a role in the same AWS account or a different account.
- An IAM User is similar to an IAM User; role is also an AWS identity with permission policies that determine what the identity can and cannot do in AWS.
- A role is not uniquely associated with a single person; it can be used by anyone who needs it.
- A role does not have long term security credential, i.e., password or security key. Instead, if the user uses a role, temporarily security credentials are created and provided to the user.

Overview of Databases:

Relational Databases

The most common form of databases are relational databases or SQL databases. They are used to store structured data.

A relational database is a collection of data items organized as a set of formally described tables. It is also known as the relational model.

The most popular relational database management systems are:
Microsoft SQL Server, Oracle, MySQL, PostgreSQL, AuroraDB, MariaDB

Non-Relational Databases

Non-relational databases are referred to as NoSQL databases or Not Only SQL databases. Features of non-relational databases:

· Do not use relational models

· Open source

· Designed to run on large clusters

· Do not use schema

· Allow adding fields to records

Online Analytical Processing(OLAP):

Online Analytical Processing consists of a type of software tools that are used for data analysis for business decisions. OLAP provides an environment to get insights from the database retrieved from multiple database systems at one time.

Examples – Any type of Data warehouse system is an OLAP system. Uses of OLAP are as follows:

Netflix movie recommendation system.

Online transaction processing(OLTP)–

Online transaction processing provides transaction-oriented applications in a 3-tier architecture. OLTP administers day to day transaction of an organization.

Examples – Uses of OLTP are as follows:

ATM center is an OLTP application.

Big Data

NoSQL databases are commonly used where huge amounts of unstructured data need to be delivered.

The most popular NoSQL databases are Cassandra, MongoDB, and Couch DB, Amazon Dynamo DB.

Data Warehousing:

A data warehouse stores data from multiple sources for analytical purposes. The most popular data warehouses are Cognos, SQL Server Reporting Services, and Oracle Hyperion, Redshift.

In-Memory Databases

In-memory databases are cache-based databases that store results in memory to reduce the load on your database infrastructure and to improve user response time.

Amazon Relational Database Service (RDS)

- AWS RDS is a **managed service** that makes it easy to set up, operate, and scale a relational database in the cloud. It automates tasks such as provisioning, patching, backup, recovery, and scaling.

  Amazon Relational Database Service (RDS) is a managed service that allows you to run databases in the AWS cloud on EC2 instances without having to worry about database administration management tasks.

  Amazon RDS is available in Amazon Aurora, Oracle, Microsoft SQL Server, PostgreSQL, MySQL, and MariaDB.

Advantages of AWS RDS:

1. **Managed Service:** AWS handles administrative tasks like provisioning, patching, and backups.
2. **Automated Backups & Recovery: Automatic** backups, snapshots, and point-in-time recovery reduce data loss risks.
3. **High Availability & Failover Support:** Multi-AZ deployments provide built-in disaster recovery.
4. **Scalability:** Easy to scale up or down in terms of compute and storage without major downtime.
5. **Security:** Built-in encryption, VPC isolation, IAM integration, and access control.
6. **Monitoring & Metrics:** Integration with **Amazon CloudWatch** for performance and health monitoring.
7. **Multi-Engine Support:** Supports popular relational databases: MySQL,

PostgreSQL, SQL Server, Oracle, and Aurora.

8. **Cost-Effective Options:** Offers pay-as-you-go pricing, reserved instances, and storage autoscaling.

Disadvantages of AWS RDS:

1. **Limited Customization:** Full control of the underlying OS or DB engine settings is restricted.
2. **Cost:** Managed services can be more expensive than self-managed databases on EC2, especially for long-term workloads.
3. **Latency:** Network latency can be higher than on-premises or EC2-hosted databases in certain cases.
4. **Vendor Lock-In: Heavily** relies on AWS infrastructure, making migration to another provider more complex.
5. **Feature Gaps:** Some advanced features from database vendors may not be supported in RDS versions.
6. **Maintenance Limitations:** Maintenance windows are scheduled and can cause brief downtime.

Amazon Route 53:

Amazon Route 53 is a highly available and scalable Domain Name System (DNS) web service. You can use Route 53 to perform three main functions in any combination: domain registration, DNS routing, and health checking.

If you choose to use Route 53 for all three functions, be sure to follow the

order below:

1. Register domain names

Your website needs a name, such as example.com. Route 53 lets you register a name for your website or web application, known as a *domain name*.

2. Route internet traffic to the resources for your domain

When a user opens a web browser and enters your domain name (example.com) or subdomain name (acme.example.com) in the address bar, Route 53 helps connect the browser with your website or web application.

3. Check the health of your resources

Route 53 sends automated requests over the internet to a resource, such as a web server, to verify that it's reachable, available, and functional. You also can choose to receive notifications when a resource becomes unavailable and choose to route internet traffic away from unhealthy resources.

A record type:

You use an A record to route traffic to a resource, such as a web server, using an IPv4 address in dotted decimal notation.

Example for the Amazon Route 53 console

192.0.2.1

Example for the Route 53 API

<Value>192.0.2.1</Value>

AAAA record type

You use an AAAA record to route traffic to a resource, such as a web server, using an IPv6 address in colon-separated hexadecimal format.

Example for the Amazon Route 53 console

2001:0db8:85a3:0:0:8a2e:0370:7334

Example for the Route 53 API

<Value>2001:0db8:85a3:0:0:8a2e:0370:7334</Value>

CAA record type

A CAA record specifies which certificate authorities (CAs) are allowed to issue certificates for a domain or subdomain. Creating a CAA record helps to prevent the wrong CAs from issuing certificates for your domains. A CAA record isn't a substitute for the security requirements that are specified by your certificate authority, such as the requirement to validate that you're the owner of a domain.

You can use CAA records to specify the following:

- Which certificate authorities (CAs) can issue SSL/TLS certificates, if any
- The email address or URL to contact when a CA issues a certificate for the domain or subdomain
- When you add a CAA record to your hosted zone, you specify three settings separated by spaces:

  flags tag "value"

Note the following about the format for CAA records:

- The value of tag can contain only the characters A-Z, a-z, and 0-9.
- Always enclose value in quotation marks ("").
- Some CAs allow or require additional values for value. Specify additional values as name-value pairs, and separate them with semicolons (;)
  for example: 0 issue "ca.example.net; account=123456"
- If a CA receives a request for a certificate for a subdomain (such as www.example.com) and if no CAA record for the subdomain exists, the CA submits a DNS query for a CAA record for the parent domain (such as example.com). If a record for the parent domain exists and if the certificate request is valid, the CA issues the certificate for the subdomain.
- We recommend that you consult with your CA to determine what values to specify for a CAA record.
- You can't create a CAA record and a CNAME record that have the same name because DNS doesn't allow using the same name for both a CNAME record and any other type of record

Route53 Routing Policies Available On AWS Application Services:

When you create a record, you choose a routing policy, which determines how Amazon Route 53 responds to queries.

- **Simple routing policy** – Use for a single resource that performs a given function for your domain,
  for example, a web server that serves content for the example.com website.
  You can use simple routing to create records in a private hosted zone.
- **Failover routing policy** – Use when you want to configure active-passive failover.
  You can use failover routing to create records in a private hosted zone.
- **Geolocation routing policy** – Use when you want to route traffic based on the location of your users.
  You can use geolocation routing to create records in a private hosted zone.
- **Geoproximity routing policy** – Use when you want to route traffic based on the location of your resources and, optionally, shift traffic from resources in one location to resources in another location.
- **Latency routing policy** – Use when you have resources in multiple AWS Regions and you want to route traffic to the Region that provides the best latency.
  You can use latency routing to create records in a private hosted zone.
- **IP-based routing policy** – Use when you want to route traffic based on the location of your users, and have the IP addresses that the traffic originates from.
- **Multivalue answer routing policy** – Use when you want Route 53 to respond to DNS queries with up to eight healthy records selected at random.
  You can use multivalue answer routing to create records in a private hosted zone.
- **Weighted routing policy** – Use to route traffic to multiple resources in proportions that you specify.
  You can use weighted routing to create records in a private hosted zone

DNS Basics:

All computers on the Internet, from your smart phone or laptop to the servers that serve content for massive retail websites, find and communicate with one another by using numbers. These numbers are known as IP addresses. When you open a web browser and go to a website, you don't have to remember and enter a long number. Instead, you can enter a domain name like example.com and still end up in the right place.

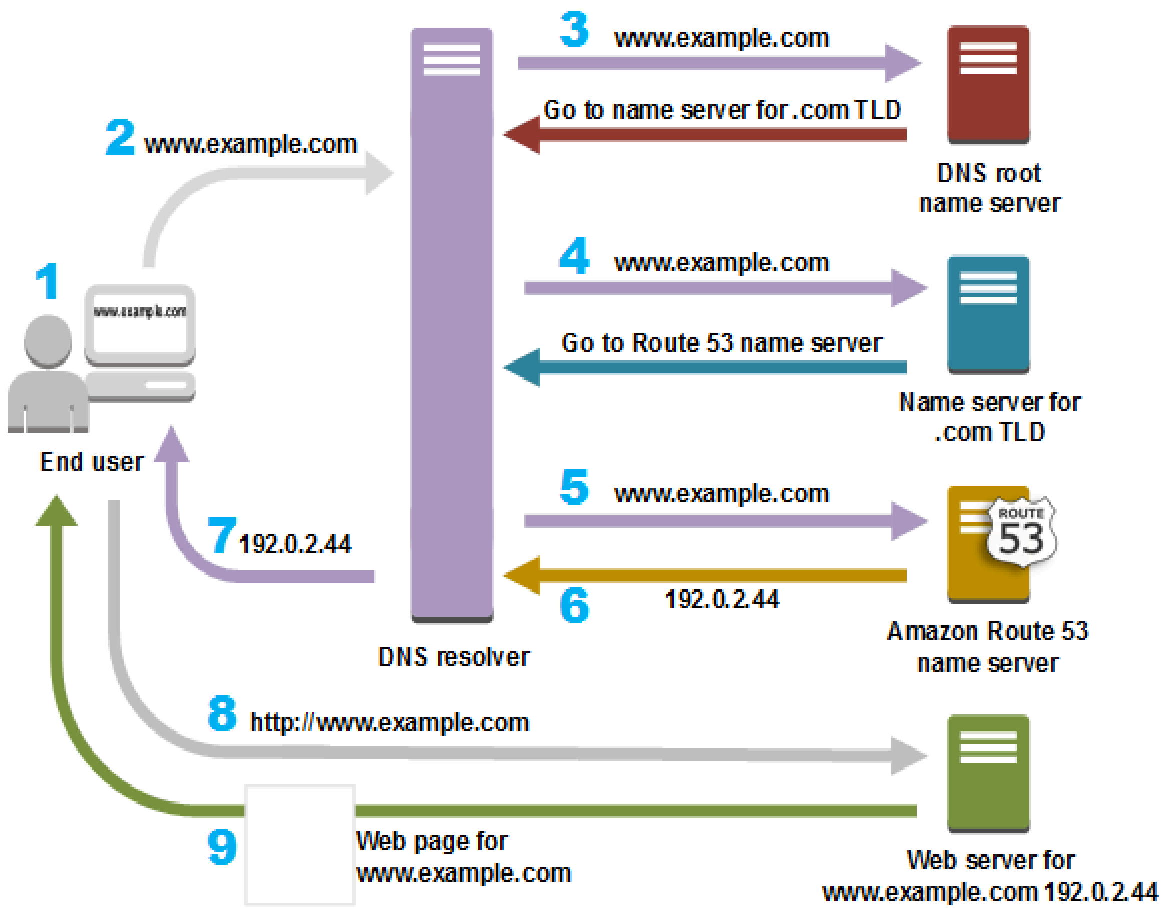
A DNS service such as Amazon Route 53 is a globally distributed service that translates human readable names like www.example.com into the numeric IP addresses like 192.0.2.1 that computers use to connect to each other. The Internet’s DNS system works much like a phone book by managing the mapping between names and numbers. DNS servers translate requests for names into IP addresses, controlling which server an end user will reach when they type a domain name into their web browser. These requests are called queries.

Types of DNS Service:

Authoritative DNS: An authoritative DNS service provides an update mechanism that developers use to manage their public DNS names. It then answers DNS queries, translating domain names into IP address so computers can communicate with each other. Authoritative DNS has the final authority over a domain and is responsible for providing answers to recursive DNS servers with the IP address information**. Amazon Route 53 is an authoritative DNS system.**

Recursive DNS: Clients typically do not make queries directly to authoritative DNS services. Instead, they generally connect to another type of DNS service known a **resolver**, or a recursive DNS service. A recursive DNS service acts like a hotel concierge: while it doesn't own any DNS records, it acts as an intermediary who can get the DNS information on your behalf. If a recursive DNS has the DNS reference **cached**, or stored for a period of time, then it answers the DNS query by providing the source or IP information. If not, it passes the query to one or more authoritative DNS servers to find the information.

How Does DNS Route Traffic To Your Web Application?



1. A user opens a web browser, enters www.example.com in the address bar, and presses Enter.
2. The request for www.example.com is routed to a DNS resolver, which is typically managed by the user's internet service provider (ISP), such as a cable internet provider, a DSL broadband provider, or a corporate network.
3. The DNS resolver for the ISP forwards the request for www.example.com to a DNS root name server.
4. The DNS resolver forwards the request for www.example.com again, this time to one of the TLD name servers for .com domains. The name server for .com domains responds to the request with the names of the four

Route 53 name servers that are associated with the example.com domain. The DNS resolver caches (stores) the four Route 53 name servers. The next time someone browses to example.com, the resolver skips steps 3 and 4 because it already has the name servers for example.com. The name servers are typically cached for two days.

5. The DNS resolver chooses a Route 53 name server and forwards the request for www.example.com to that name server.
6. The Route 53 name server looks in the example.com hosted zone for the www.example.com record, gets the associated value, such as the IP address for a web server, 192.0.2.44, and returns the IP address to the DNS resolver.
7. The DNS resolver finally has the IP address that the user needs. The resolver returns that value to the web browser.
8. The web browser sends a request for www.example.com to the IP address that it got from the DNS resolver. This is where your content is, for example, a web server running on an Amazon EC2 instance or an Amazon S3 bucket that's configured as a website endpoint.
9. The web server or other resource at 192.0.2.44 returns the web page for www.example.com to the web browser, and the web browser displays the page.

Application Services

AWS Application Services are a collection of AWS services designed to help developers build, deploy, and manage applications on the cloud. These services provide a wide range of capabilities, including application integration, hosting, migration, and data analytics.

Amazon Simple Queue Service (SQS)

Amazon Simple Queue Service (SQS) is a fast, reliable, scalable, fully managed message queuing service."

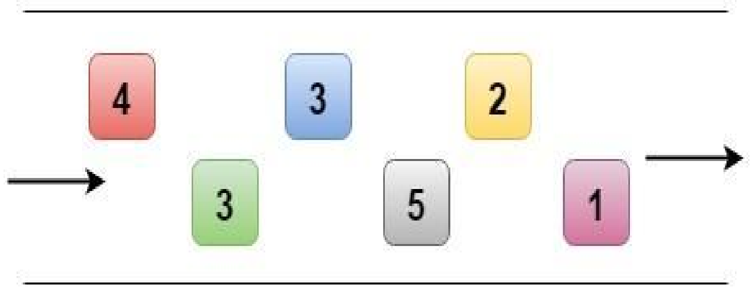
- SQS stands for Simple Queue Service.
- SQS was the first service available in AWS.
- Amazon SQS is a web service that gives you access to a message queue that can be used to store messages while waiting for a computer to process them.
- Amazon SQS is a distributed queue system that enables web service applications to quickly and reliably queue messages that one component in the application generates to be consumed by another component where a queue is a temporary repository for messages that are awaiting processing.
- With the help of SQS, you can send, store and receive messages between software components at any volume without losing messages.
- Using Amazon sqs, you can separate the components of an application so that they can run independently, easing message management between components.
- Any component of a distributed application can store the messages in the queue.
- Messages can contain up to 256 KB of text in any format such as json, xml, etc.
- Any component of an application can later retrieve the messages programmatically using the Amazon SQS API.
- The queue acts as a buffer between the component producing and saving data, and the component receives the data for processing. This means that the queue resolves issues that arise if the producer is producing work faster than the consumer can process it, or if the producer or consumer is only intermittently connected to the network.
- If you got two EC2 instances which are pulling the SQS Queue. You can configure the autoscaling group if a number of messages go over a certain

limit. Suppose the number of messages exceeds 10, then you can add additional EC2 instance to process the job faster. In this way, SQS provides elasticity.

There are two types of Queue:

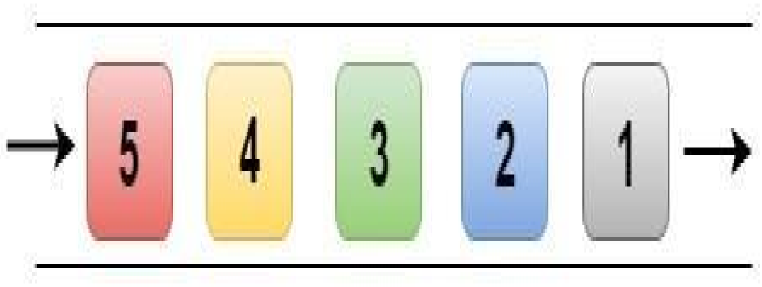
1. Standard Queues (default)
2. FIFO Queues (First-In-First-Out)

1. Standard Queues



- SQS offers a standard queue as the default queue type.
- It allows you to have an unlimited number of transactions per second.
- It guarantees that a message is delivered at least once. However, sometime, more than one copy of a message might be delivered out of order.
- It provides best-effort ordering which ensures that messages are generally delivered in the same order as they are sent but it does not provide a guarantee

2. FIFO Queue



- The FIFO Queue complements the standard Queue.
- It guarantees ordering, i.e., the order in which they are sent is also received in the same order.

- The most important features of a queue are FIFO Queue and exactly-once processing,

  i.e., a message is delivered once and remains available until consumer processes and deletes it.
- FIFO Queue does not allow duplicates to be introduced into the Queue.
- It also supports message groups that allow multiple ordered message groups within a single Queue.
- FIFO Queues are limited to 300 transactions per second but have all the capabilities of standard queues.

<u>Amazon Simple Notification Service (SNS):</u>

Amazon Simple Notification Service (SNS) is a fast, reliable, scalable, fully managed message queuing service.”

- SNS stands for Simple Notification Service.
- It is a web service which makes it easy to set up, operate, and send a notification from the cloud.
- It provides developers with the highly scalable, cost-effective, and flexible capability to publish messages from an application and sends them to other applications.
- It is a way of sending messages. When you are using Auto-scaling, it triggers an SNS service which will email you that "your EC2 instance is growing".
- SNS can also send the messages to devices by sending push notifications to Apple, Google, Fire OS, and Windows devices, as well as Android devices in China with Baidu Cloud Push.
- Besides sending the push notifications to the mobile devices, Amazon SNS sends the notifications through SMS or email to an Amazon Simple Queue Service (SQS), or to an HTTP endpoint.
- SNS notifications can also trigger the Lambda function. When a message is published to an SNS topic that has a Lambda function associated with it, Lambda function is invoked with the payload of the message. Therefore, we can say that the Lambda function is invoked with a message payload as an input parameter and manipulate the information in the message and then sends the message to other SNS topics or other AWS services.
- Amazon SNS allows you to group multiple recipients using topics where the topic is a logical access point that sends the identical copies of the same

message to the subscribe recipients.

- Amazon SNS supports multiple endpoint types. For example, you can group together IOS, Android and SMS recipients. Once you publish the message to the topic, SNS delivers the formatted copies of your message to the subscribers.
- To prevent the loss of data, all messages published to SNS are stored redundantly across multiple availability zones.
- An SNS topic is a communication channel that allows you to send messages and subscribe to notifications.

Amazon SNS is a web service that manages sending messages to the subscribing endpoint.

There are two clients of SNS:

• Subscribers

• Publishers

• Publishers

Publishers are also known as producers that produce and send the message to the SNS which is a logical access point.

• Subscribers

Subscribers such as web servers, email addresses, Amazon SQS queues, AWS Lambda functions receive the message or notification from the SNS over one of the supported protocols (Amazon SQS, email, Lambda, HTTP, SMS).

SNS Features:

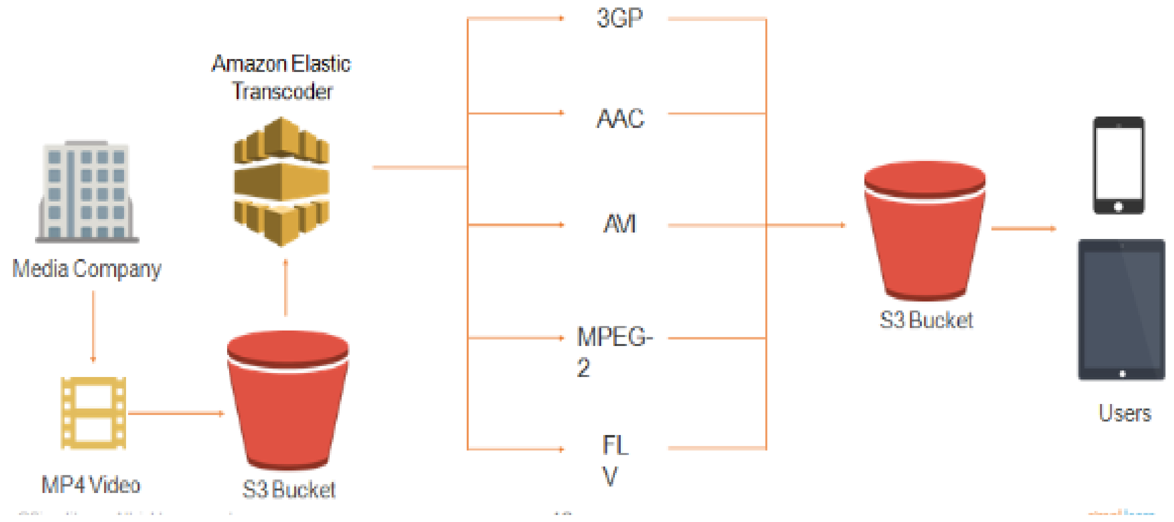
- Instantaneous push based delivery
- Multiple transfer protocol
- A pay-as-you-go model
- A simple web-based interface
- Messages stored redundantly across multiple availability zones

<u>Amazon Elastic Transcoder:</u>

- Elastic Transcoder is an aws service used to convert the media files stored in an S3 bucket into the media files in different formats supported by different devices.
- Elastic Transcoder is a media transcoder in the cloud.
- It is used to convert media files from their original source format into different formats that will play on smartphones, tablets, PC's, etc.
- It provides transcoding presets for popular output formats means that you don't need to guess about which settings work best on particular devices.
- If you use Elastic Transcoder, then you need to pay based on the minutes that you transcode and the resolution at which you transcode.

Elastic Transcoder Example:

A media company creates a new MP4 video for distribution online. You can upload the video to an S3 bucket and configure Elastic Transcoder to pick it up and convert it into a range of other specified formats. The new formats are saved back into an Amazon S3 bucket.



Components of Elastic Transcoder:

Elastic Transcoder consists of four components:

- Jobs
- Pipelines
- Presets
- Notifications

Jobs
The main task of the job is to complete the work of transcoding. Each job can convert a file up to 30 formats. For example, if you want to convert a media file into eight different formats, then a single job creates files in eight formats. When you create a job, you need to specify the name of the file that you want to transcode.

Pipelines
Pipelines are the queues that consist of your transcoding jobs. When you create a job, then you need to specify which pipeline you want to add your job. If you want a job to create more than one format, Elastic Transcoder creates the files for each format in the order you specify the formats in a job. You can create either of the two pipelines, i.e., standard-priority jobs and high-priority jobs. Mainly jobs go into the standard-priority jobs. Sometimes you want to transcode the file immediately; the high-priority pipeline is used.

Presets
Presets are the templates that contain the settings for transcoding the media file from one format to another format. Elastic transcoder consists of some default presets for common formats. You can also create your own presets that are not included in the default presets. You need to specify a preset that you want to use when you create a job.

Notifications
Notification is an optional field which you can configure with the Elastic Transcoder. Notification Service is a service that keeps you updated with the status of your job: when Elastic Transcoder starts processing your job, when Elastic Transcoder finishes its job, whether the Elastic Transcoder encounters an error condition or not. You can configure Notifications when you create a pipeline.