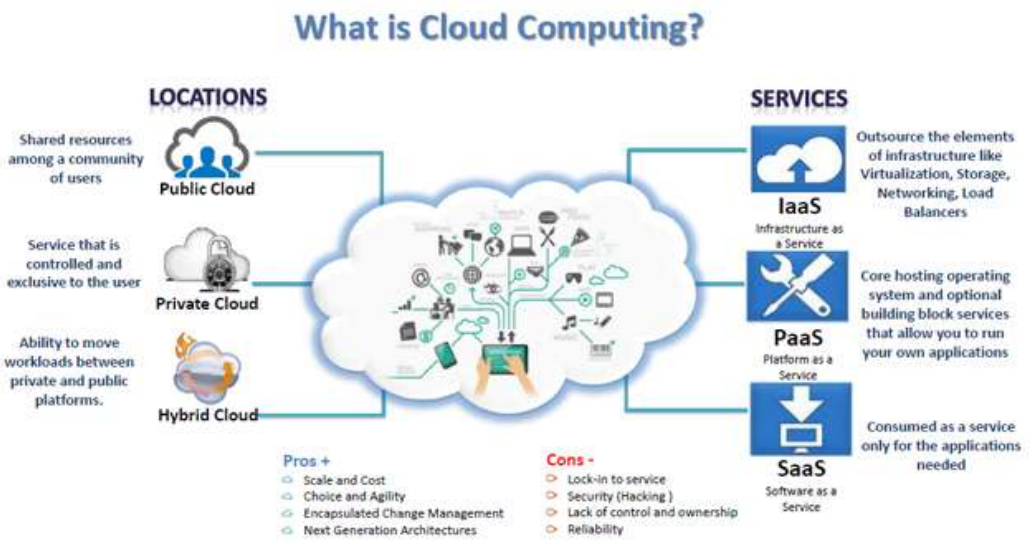
AMAZON WEB SERVICES

AWS for Data Engineering & Data Analytics

Basic Concepts & Terminology

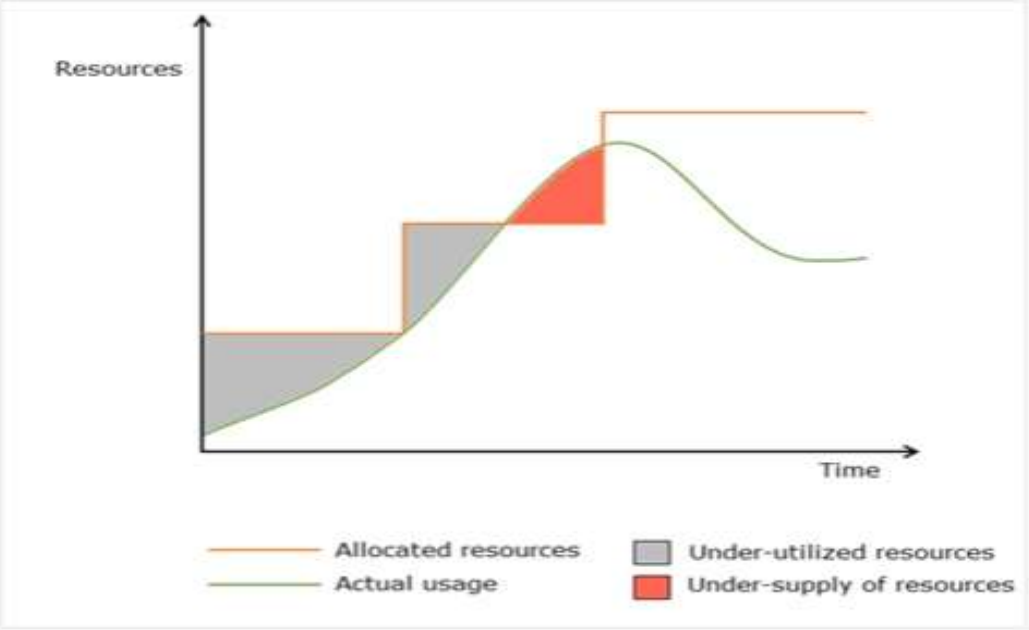
Understanding Cloud Computing

Cloud Computing is a term that indicates delivery of computing resources such as servers, storage, network etc. over the internet by a service provider and is accessible from anywhere and at any time.

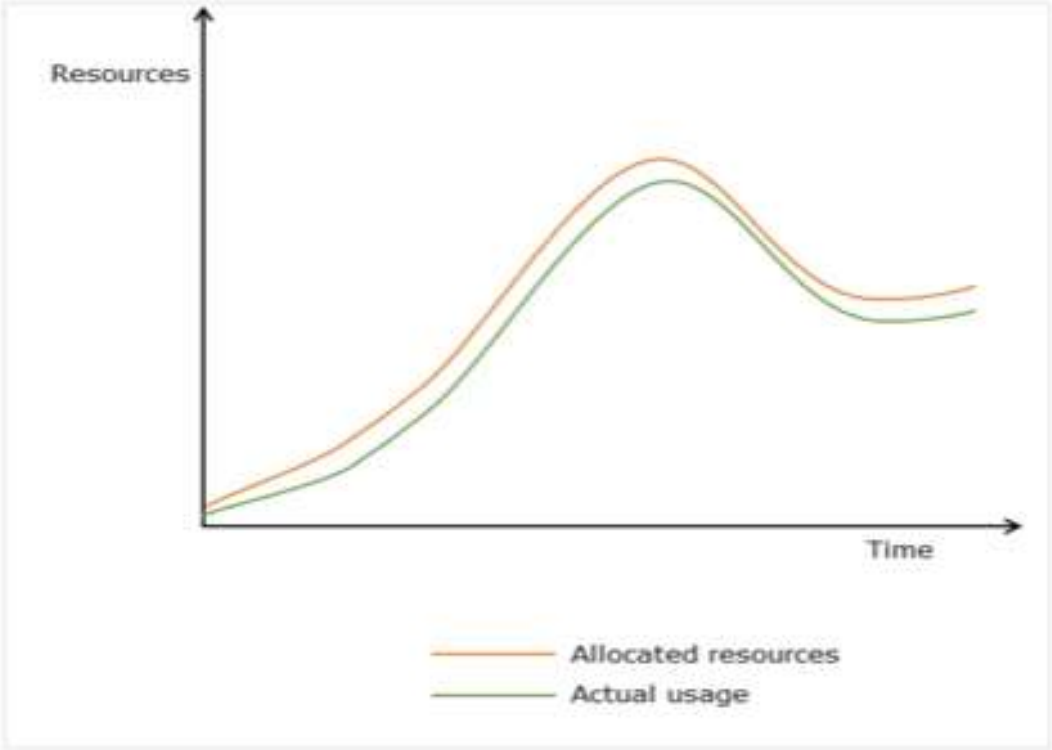


Problems with traditional computing model

- Under-utilization or Over-supply
- Not cost effective



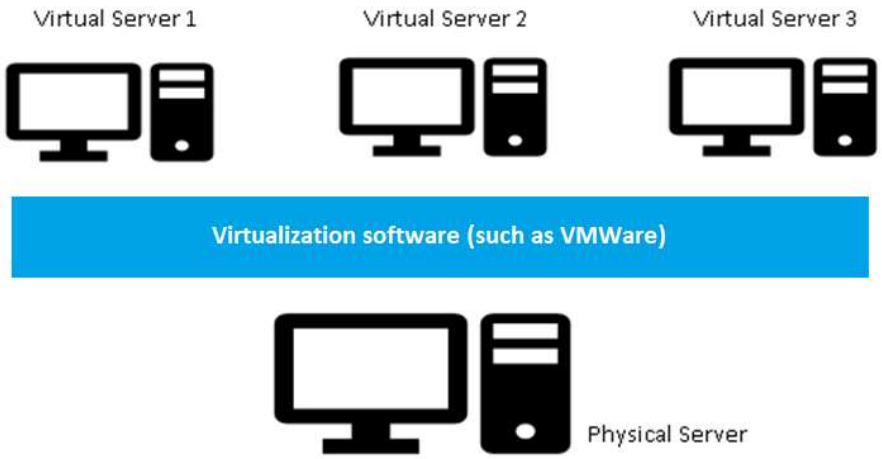
Runtime provisioning with cloud



Benefits of cloud computing

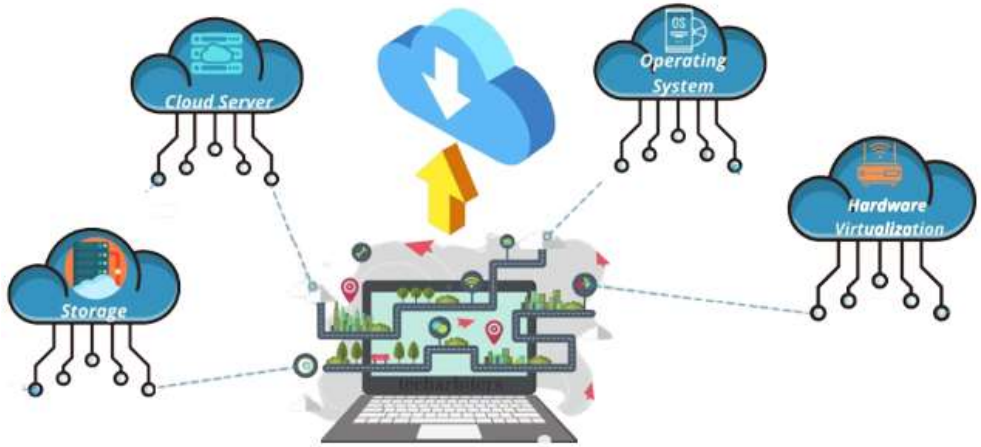
What is Virtualization?

Virtualization is a process that allows a computer to share its hardware resources with multiple digitally separated environments. Each virtualized environment runs within its allocated resources, such as memory, processing power, and storage. With virtualization, organizations can switch between different operating systems on the same server without rebooting.



Why is virtualization important?

- By using virtualization, you can interact with any hardware resource with greater flexibility.
- Physical servers consume electricity, take up storage space, and need maintenance. You are often limited by physical proximity and network design if you want to access them. Virtualization removes all these limitations by abstracting physical hardware functionality into software. You can manage, maintain, and use your hardware infrastructure like an application on the web.

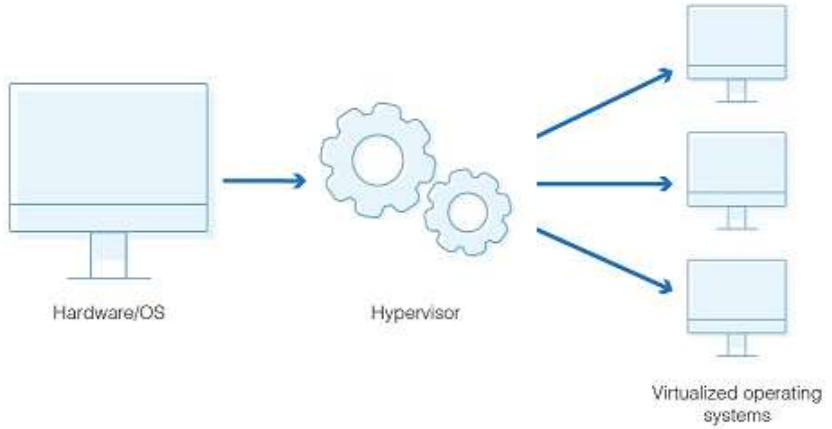


Virtual machine

- A virtual machine is a software-defined computer that runs on a physical computer with a separate operating system and computing resources.
- The physical computer is called the host machine and virtual machines are guest machines.
- Multiple virtual machines can run on a single physical machine. Virtual machines are abstracted from the computer hardware by a hypervisor.

Hypervisor

- The hypervisor is the virtualization software that you install on your physical machine.
- It is a software layer that acts as an intermediary between the virtual machines and the underlying hardware or host operating system.
- The hypervisor coordinates access to the physical environment so that several virtual machines have access to their own share of physical resources.

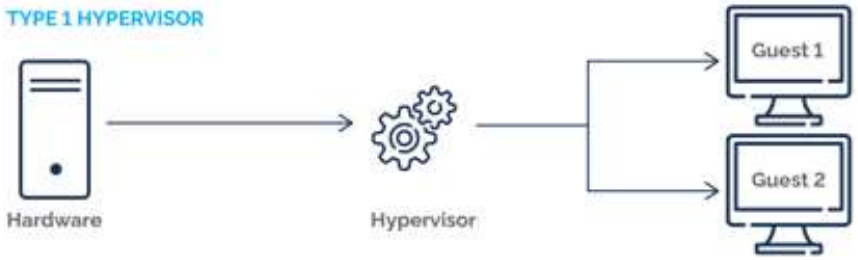


For example, if the virtual machine requires computing resources, such as computer processing power, the request first goes to the hypervisor. The hypervisor then passes the request to the underlying hardware, which performs the task.

Types of hypervisors

Type 1 hypervisors

A type 1 hypervisor—also called a bare-metal hypervisor—runs directly on the computer hardware. It has some operating system capabilities and is highly efficient because it interacts directly with the physical resources.



Type 2 hypervisors

A type 2 hypervisor runs as an application on computer hardware with an existing operating system. Use this type of hypervisor when running multiple operating systems on a single machine.

Cloud computing service models

Software as a Service

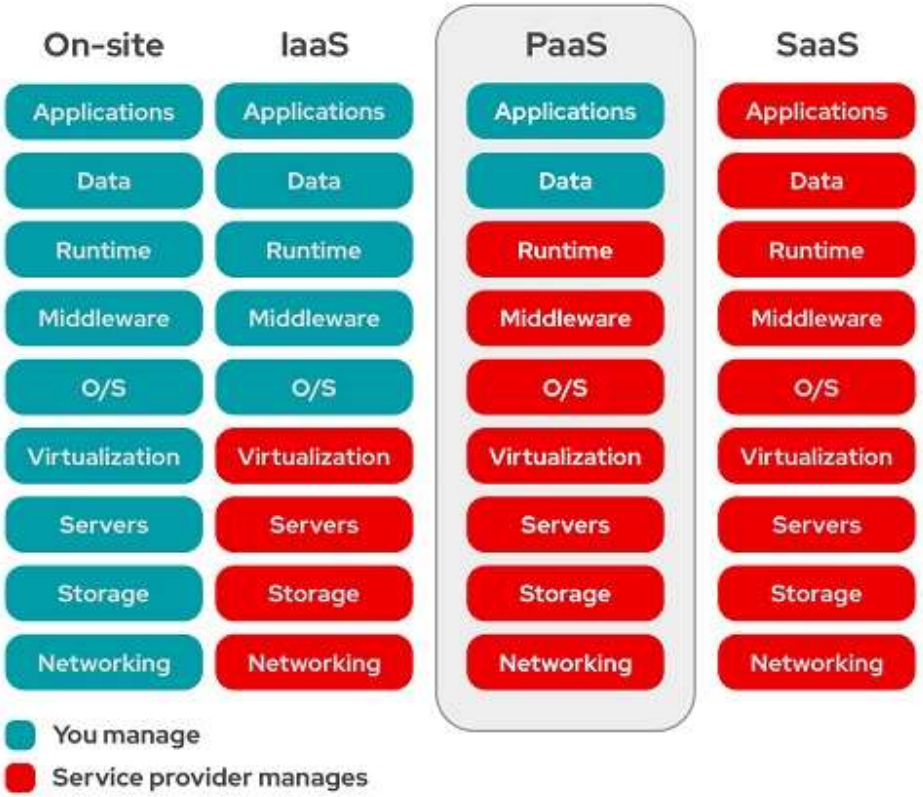
Includes supporting Platform and Infrastructure

Platform as a Service

Includes underlying Infrastructure

Infrastructure as a Service

Cloud computing service models



Cloud computing service models

A cloud delivery model refers to any cloud-based solution one can access through a web browser from any device with internet capabilities.

- **Software as a Service (SaaS)** - This is the software distribution model whereby developers put their applications into a cloud-based delivery system. Customers access these applications via the Internet, usually through a browser. Ex: Google Workspace, Dropbox, Salesforce, Cisco WebEx etc.

- **Platform as a Service (PaaS)** - This type of model is often run by an organization where the users can not only create and run applications on the cloud but also effectively maintain them themselves. Ex: Google App Engine, Apache Stratos, Heroku, Force.com etc.

- **Infrastructure as a Service (IaaS)** - This model provides the infrastructure necessary for companies to run their operations. It includes virtual and non-virtual servers, storage, and data center space all in one place.

AWS Basics

What is AWS?

Amazon Web Services (AWS) is a comprehensive cloud computing platform that includes:

- Infrastructure as a service (IaaS)
- Platform as a service (PaaS)

AWS services offer scalable solutions for :

- Compute
- Storage
- Databases
- Analytics

and many more…

Why AWS ?

- **Easy to use**

  AWS is designed to allow application providers to quickly and securely host your applications – whether an existing application or a new SaaS-based application. You can use the AWS Management Console or well-documented web services APIs to access AWS's application hosting platform.

- **Flexible**

  AWS enables you to select the operating system, programming language, web application platform, database, and other services you need. With AWS, you receive a virtual environment that lets you load the software and services your application requires. This eases the migration process for existing applications while preserving options for building new solutions.

- **Cost-Effective**

  You pay only for the compute power, storage, and other resources you use, with no long-term contracts or up-front commitments. For more information on comparing the costs of other hosting alternatives with AWS, see the AWS Economics Center.

Why AWS ?

- **Reliable**

  With AWS, you take advantage of a scalable, reliable, and secure global computing infrastructure, the virtual backbone of Amazon.com's multi-billion dollar online business that has been honed for over a decade.

- **Scalable and high-performance**

  Using AWS tools, Auto Scaling, and Elastic Load Balancing, your application can scale up or down based on demand. Backed by Amazon's massive infrastructure, you have access to compute and storage resources when you need them.

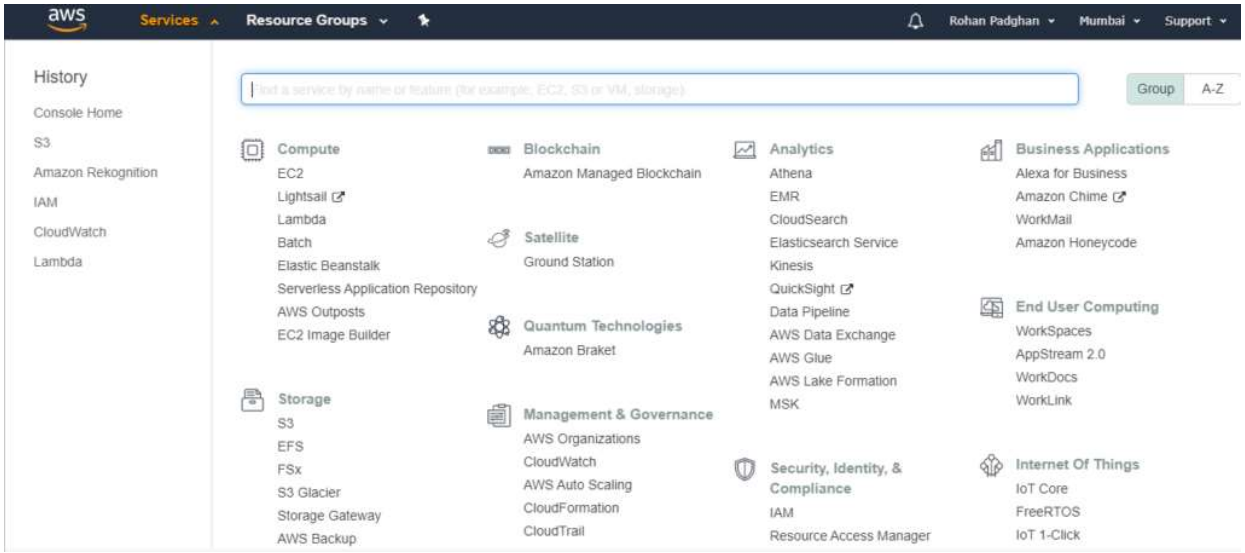
- **Secure.**

  AWS utilizes an end-to-end approach to secure and harden our infrastructure, including physical, operational, and software measures. For more information, see the AWS Security Center.

Understanding console and various services

Demo.

The AWS Management Console is a web application that comprises and refers to a broad collection of service consoles for managing AWS resources. When you first sign in, you see the console home page.



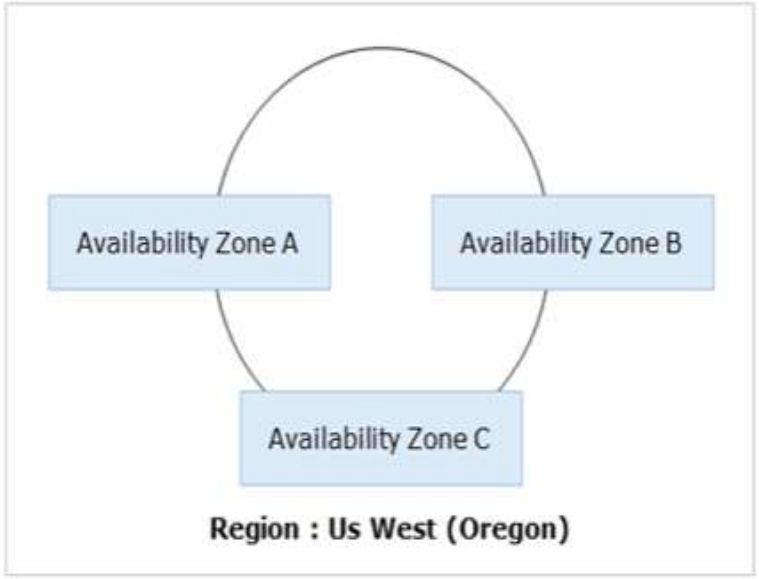
AWS Global Cloud Infrastructure

Global cloud infrastructure of AWS

The AWS Cloud spans 99 Availability Zones within 31 geographic regions around the world, with announced plans for 15 more Availability Zones and 5 more AWS Regions in Canada, Israel, Malaysia, New Zealand, and Thailand. (as of March 2023)

Regions & Availability zones

- AWS **Region** is a physical location around the world where AWS clusters data centers called **Availability Zones**. Each group of logical data centers is called an Availability Zone (AZ).
- Each AWS Region consists of multiple, isolated, and physically separate Availability Zones within a geographic area.
- Each AZ has independent power, cooling, and physical security and is connected via redundant, ultra-low-latency networks.



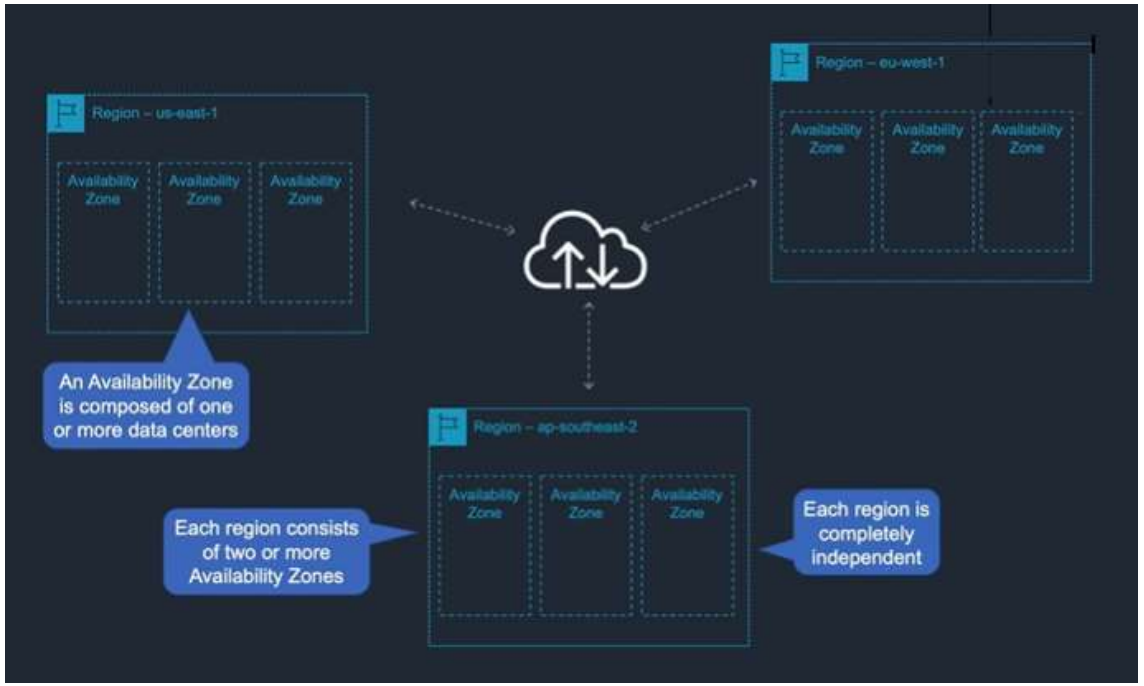
Regions & Availability zones

Region – Two or more AZs

Availability Zone – One or more data centers

Each region is completely independent

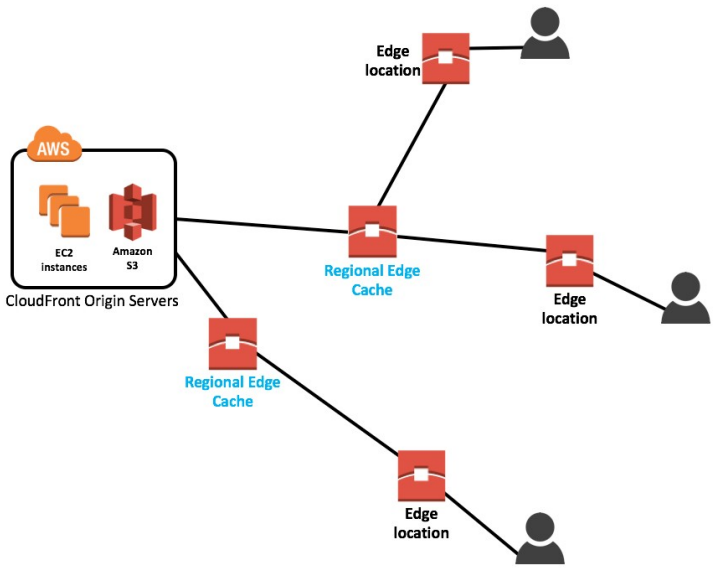
All regions are connected via high bandwidth, fully redundant network.



AWS Edge Networks

The other component of the AWS Global Cloud Infrastructure is the edge networks of Point-of-Presence or PoP. It consists of **Edge Locations** and **Regional Edge Caches,** which enables you to distribute your content with low-latency to your global users. Basically, a PoP serves as an access point that allows two different networks to communicate with each other.

By using these global edge networks, a user request doesn't need to travel far back to your origin just to fetch data. The cached contents can quickly be retrieved from regional edge caches that are closer to your end-users.



AWS provides a service called **Amazon CloudFront** to leverage edge network for low-latest content delivery.

AWS Local Zones

AWS Local Zones are a type of infrastructure deployment that places compute, storage, database, and other select AWS services close to large population and industry centers.

Local zone gives users access to single-digit milliseconds latency with AWS Direct Connect and the ability to meet data redundancy requirements.

Local zones are connected to their parent region via AWS' redundant and high bandwidth private network.

AWS Wavelength Zones

Wavelength Zones are AWS infrastructure deployments that embed AWS compute and storage services within telecommunications providers' data centers at the edge of the 5G network, so application traffic can reach application servers running in Wavelength Zones without leaving the mobile providers' network.

Wavelength Zones by region:

https://docs.aws.amazon.com/wavelength/latest/developerguide/available-wavelength-zones.html

AWS Outposts

AWS Outposts is a family of fully managed solutions delivering AWS infrastructure and services to virtually any on-premises or edge location for a truly consistent hybrid experience.

Outposts solutions allow you to extend and run native AWS services on premises, and is available in a variety of form factors, from 1U and 2U Outposts servers to 42U Outposts racks, and multiple rack deployments.

With AWS Outposts, you can run some AWS services locally and connect to a broad range of services available in the local AWS Region.

The 'U' in any server description is short for "RU" which stands for Rack Unit -- this is the standardized designation for the form factor of the server: 1U = 1.75" in height or thickness. 2U is 1.75" x2 = 3.5 inches.

AWS Identity and Access Management (IAM)

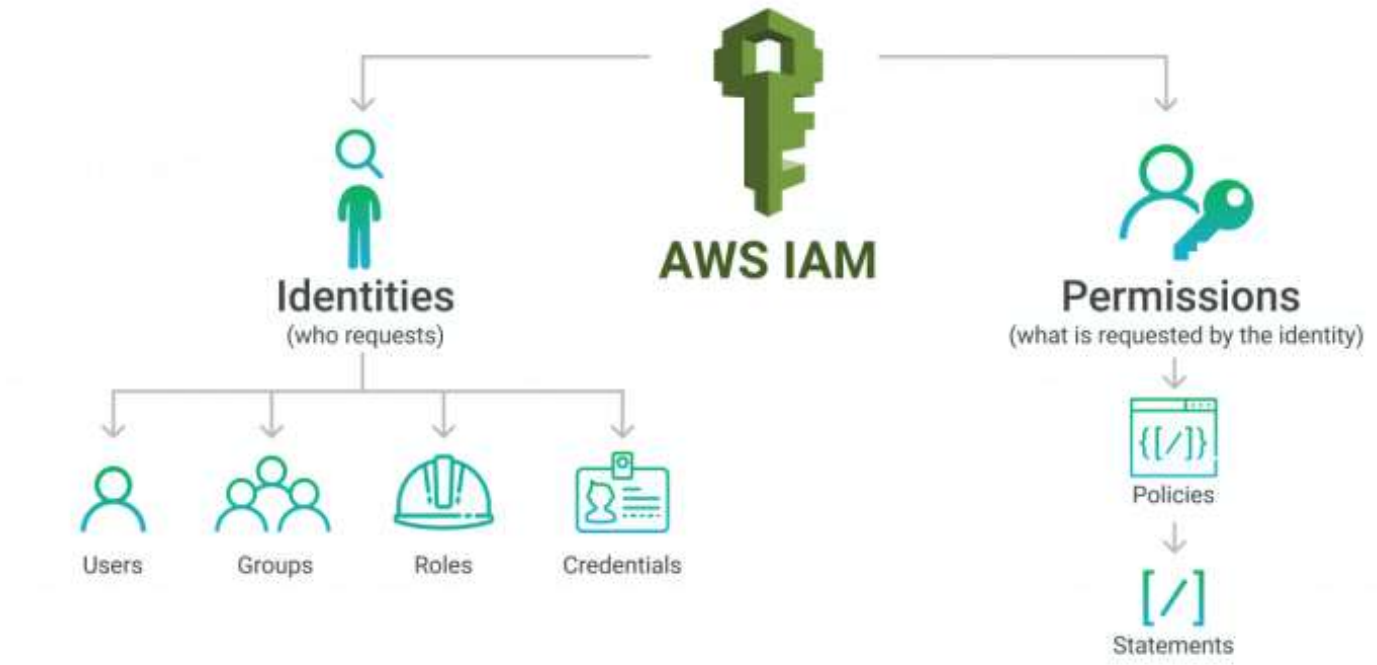
AWS IAM

- AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources for your users.
- IAM is used to control
  - **Identity** – who can use your AWS resources (authentication)
  - **Access** – what resources they can use and in what ways (authorization)

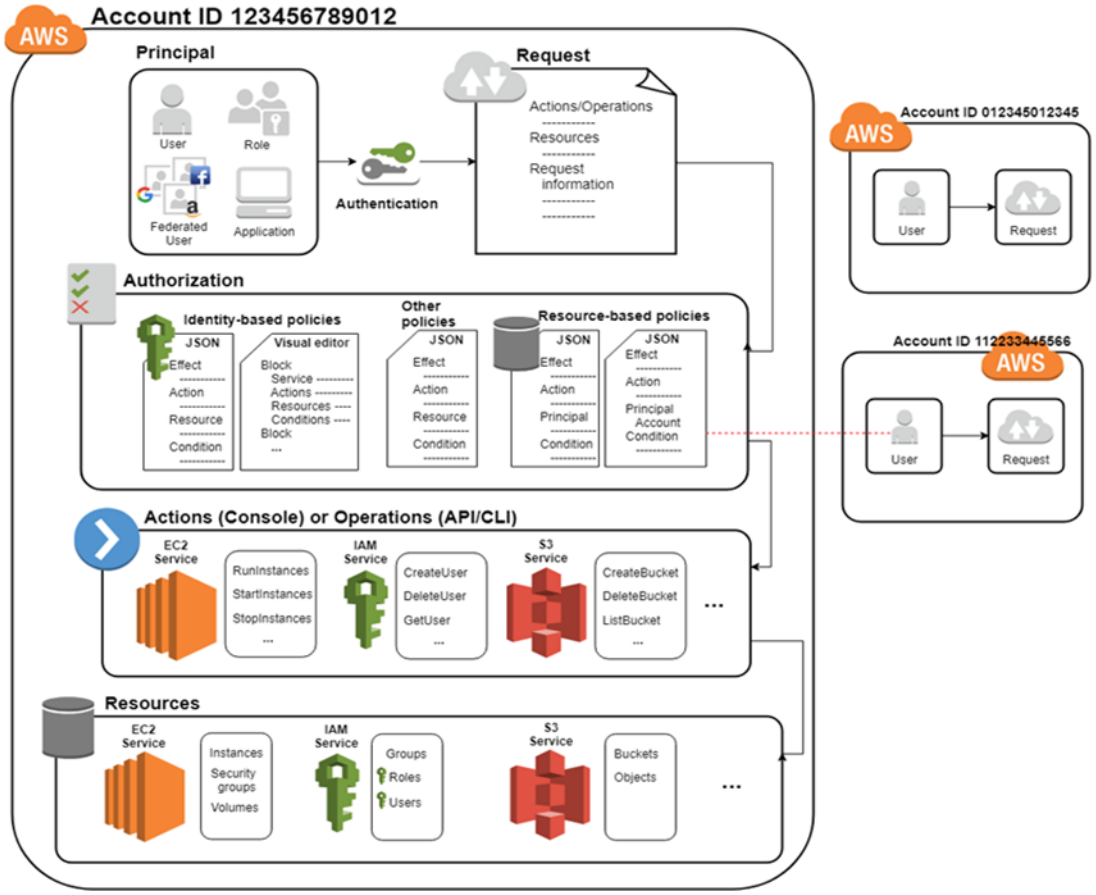
AWS IAM

With AWS Identity and Access Management (IAM), you can specify who or what can access services and resources in AWS, centrally manage fine-grained permissions, and analyze access to refine permissions across AWS.

AWS IAM


AWS IAM
Identities
(who requests)
Users
Groups
Roles
Credentials
Permissions
(what is requested by the identity)
Policies
[/]
Statements

How IAM works ?



Amazon Simple Storage Service (S3)

Amazon S3

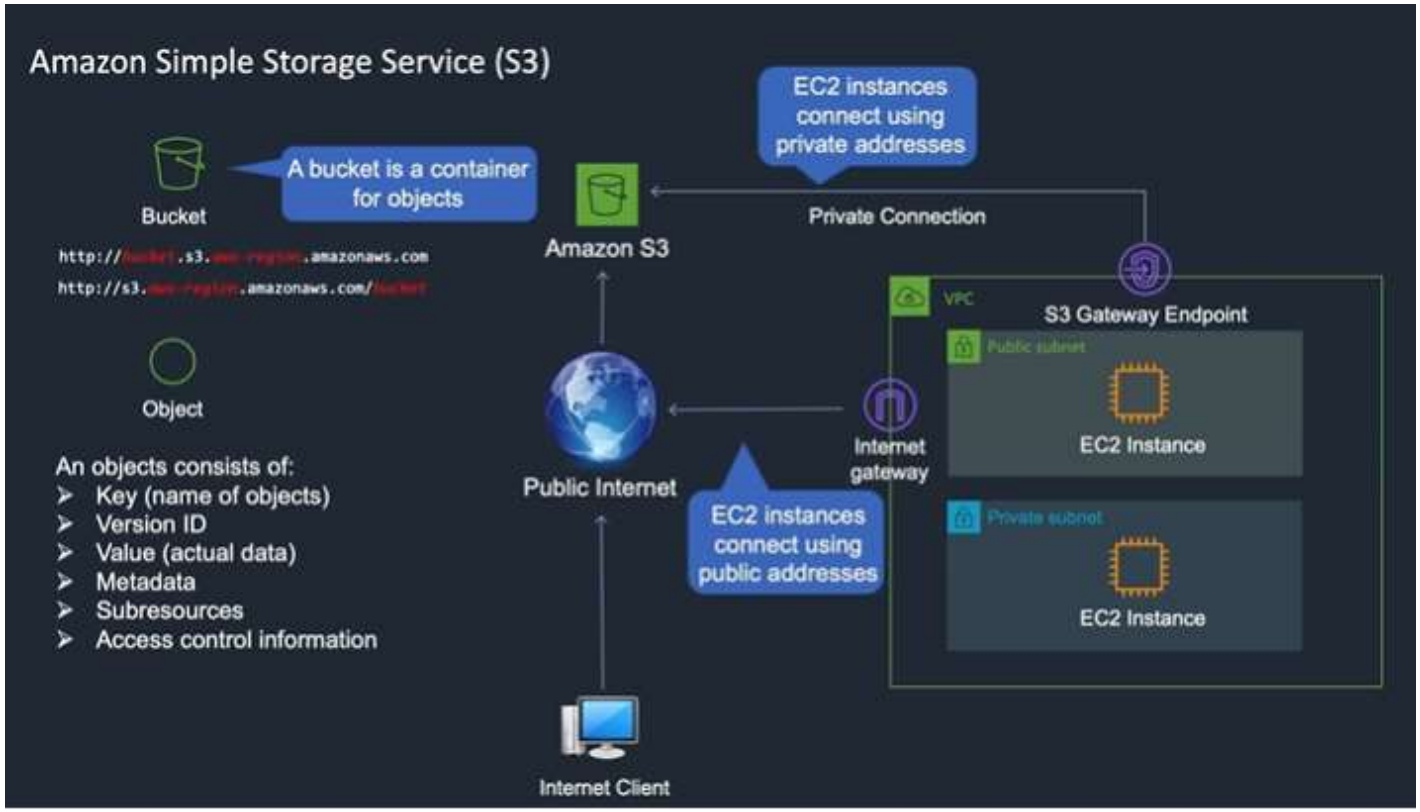
- Amazon **Simple Storage Service** (Amazon **S3**) is an **object storage service** that offers industry-leading scalability, data availability, security, and performance.

- S3 is a web-based service designed for online backup and archiving of data and application programs. You can use Amazon S3 to store and retrieve any amount of data at any time, from anywhere via an internet URL.

- S3 allows to upload, store, and download any type of files up to **5 TB** in size.

- In S3 we store **objects** in **buckets**
  - Bucket is a container of objects. Each bucket can have any number of objects.
  - Bucket name must be globally unique (as it is accessed via a URL).
  - Bucket is created in a region.

Amazon S3


Amazon Simple Storage Service (S3)
Bucket
A bucket is a container for objects
http://bucket.s3.aws-region.amazonaws.com
http://s3.aws-region.amazonaws.com/bucket
Object
An objects consists of:
Key (name of objects)
Version ID
Value (actual data)
Metadata
Subresources
Access control information
Amazon S3
EC2 instances connect using private addresses
Private Connection
VPC
S3 Gateway Endpoint
Public subnet
EC2 Instance
Internet gateway
Public Internet
EC2 instances connect using public addresses
Private subnet
EC2 Instance
Internet Client

Amazon S3

- You can store any type of file in S3
- Files can be anywhere from 0 bytes to 5 TB
- There is unlimited storage available
- S3 is a universal namespace so bucket names must be unique globally. However, you create your buckets within a region
- It is a best practice to create buckets in regions that are physically closest to your users to reduce latency
- There is no hierarchy for objects within a bucket
- Delivers strong read-after-write consistency

Amazon S3 Buckets

- Files are stored in buckets
- A bucket can be viewed as a container for objects
- A bucket is a flat container of objects
- It does not provide a hierarchy of objects
- You can use an object key name (prefix) to mimic folders
- 100 buckets per account by default
- You can store unlimited objects in your buckets
- You cannot create nested buckets

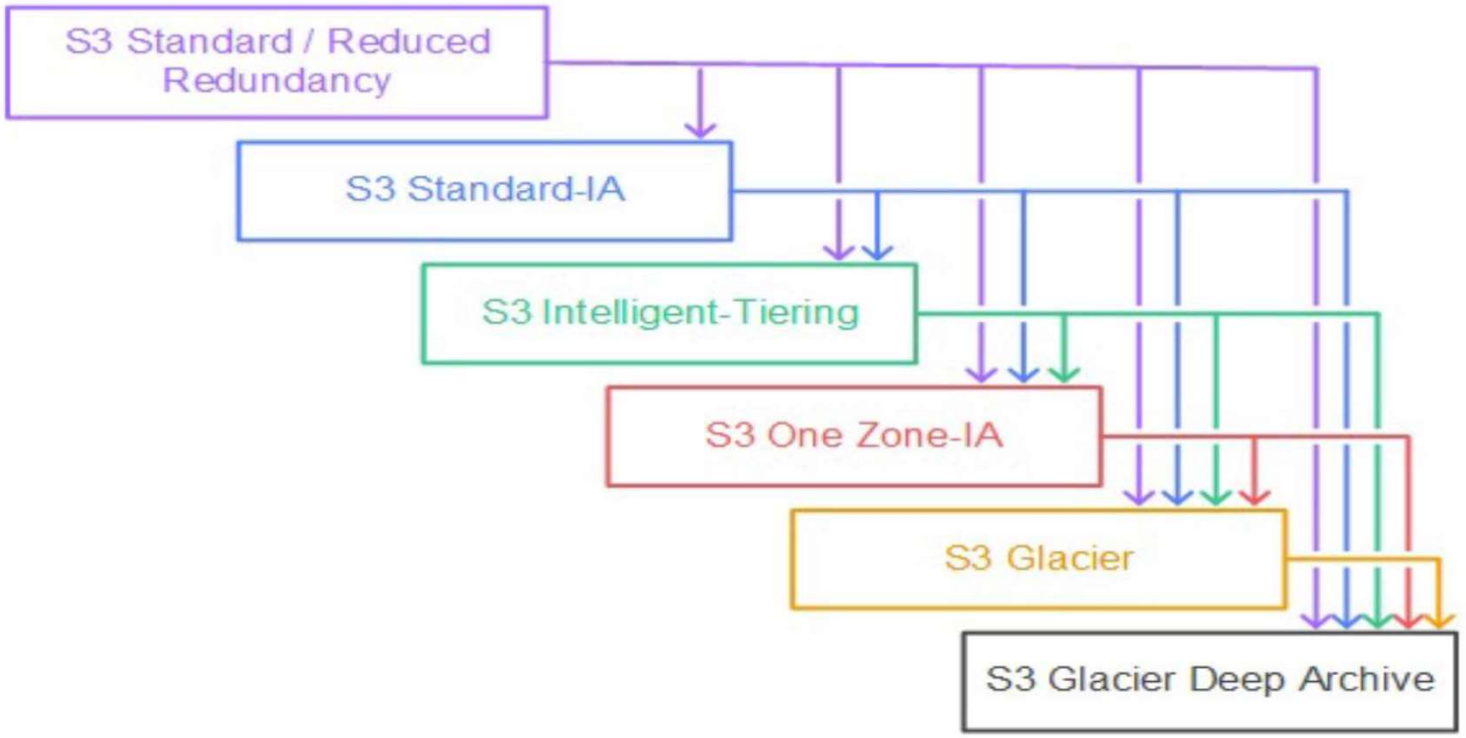
Amazon S3 Objects

- An object is a file uploaded to S3
- S3 supports any file type
- Each object is stored and retrieved by a unique key
- Objects remain in the region they are stored in. You can setup replication.
- Permissions can be defined on objects at any time
- Storage class is set at the object level

Amazon S3 storage classes

	S3 Standard	S3 Intelligent Tiering	S3 Standard-IA	S3 One Zone-IA	S3 Glacier	S3 Glacier Deep Archive
Designed for durability	99.999999999%	99.999999999%	99.999999999%	99.999999999%	99.999999999%	99.999999999%
Designed for availability	99.99%	99.9%	99.9%	99.5%	99.99%	99.99%
Availability SLA	99.9%	99%	99%	99%	99.9%	99.9%
Availability Zones	≥3	≥3	≥3	1	≥3	≥3
Minimum capacity charge per object	N/A	N/A	128KB	128KB	40KB	40KB
Minimum storage duration charge	N/A	30 days	30 days	30 days	90 days	180 days
Retrieval fee	N/A	N/A	Per GB retrieved	Per GB retrieved	Per GB retrieved	Per GB retrieved
First byte latency	milliseconds	milliseconds	milliseconds	milliseconds	select minutes or hours	select hours
Storage type	Object	Object	Object	Object	Object	Object
Lifecycle transitions	Yes	Yes	Yes	Yes	Yes	Yes

Amazon S3 storage class transitions



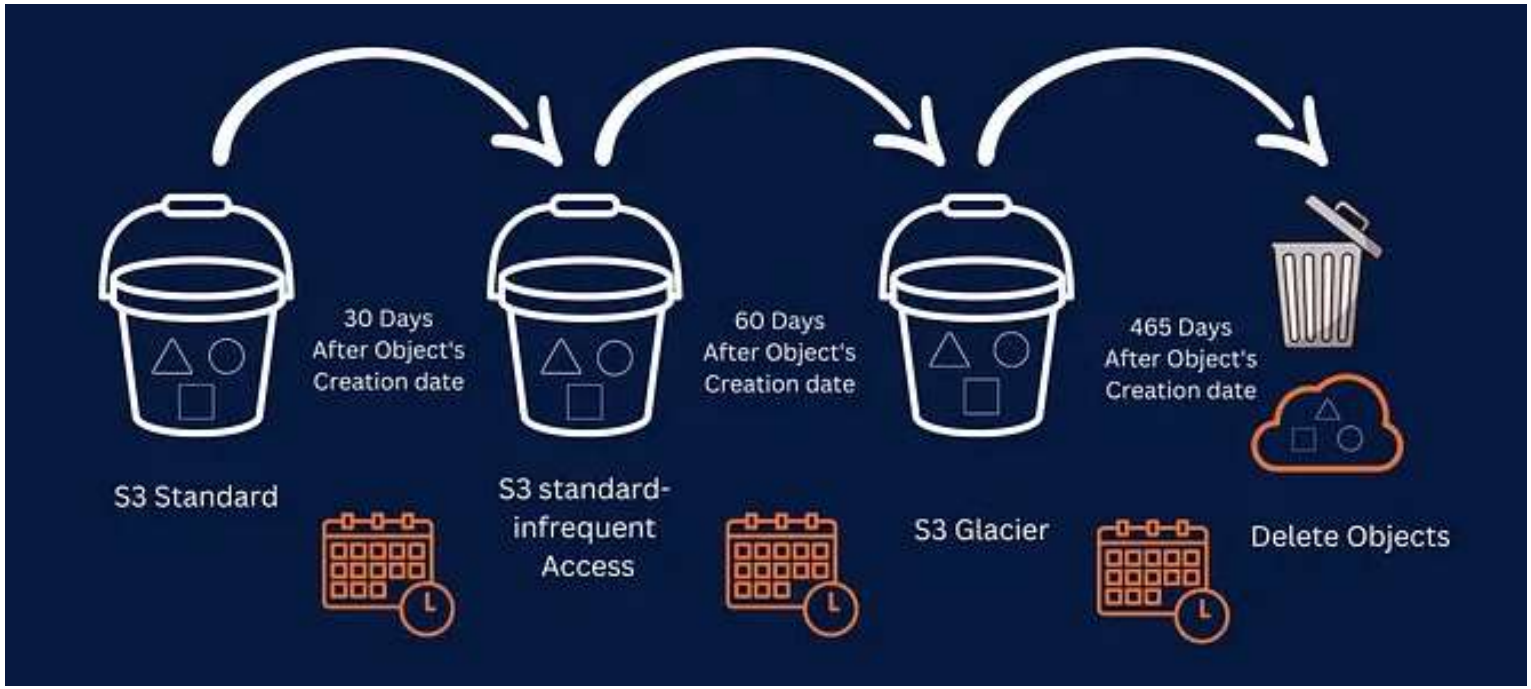
Amazon S3 versioning

- In S3, when you create a bucket, versioning is disabled by default.
- To enable versioning:
  - Go to properties tab of the bucket
  - Click on **Edit** button in the **Bucket versioning**
  - Enable the versioning
- If versioning is enabled, when you upload the same object multiple times to the same bucket, different versions are create.
- When you delete an object, a delete marker is created on the object. You can delete some of the delete marked files to go back to older versions.
- To delete an object from a bucket, you should not only delete the object itself, but also delete all the versions as well.

Amazon S3 life cycle management

- Lifecycle Management is used to store objects cost-effectively throughout their lifecycle. A lifecycle configuration is a set of rules that define the actions applied by S3 to a group of objects.
- The lifecycle defines two types of actions:
  - **Transition actions**: define when objects transition to another storage class.
    - For example, you choose to transit the objects to the Standard IA storage class 30 days after you have created them or archive the objects to the Glacier storage class 60 days after you have created them.
  - **Expiration actions**: define when objects expire. Amazon S3 deletes the expired object on your behalf.

Amazon S3 life cycle management



Amazon S3 server access logging (SAL)

- Disabled by default
- If enabled, SAL provides detailed records for the requests that are made to a bucket
  - Details include the requester, bucket name, request time, request action, response status, and error code (if applicable)
- Only pay for the storage space used
- Must configure a separate bucket as the destination (can specify a prefix)
- Must grant write permissions to the Amazon S3 Log Delivery group on destination bucket

Amazon S3 bucket policy

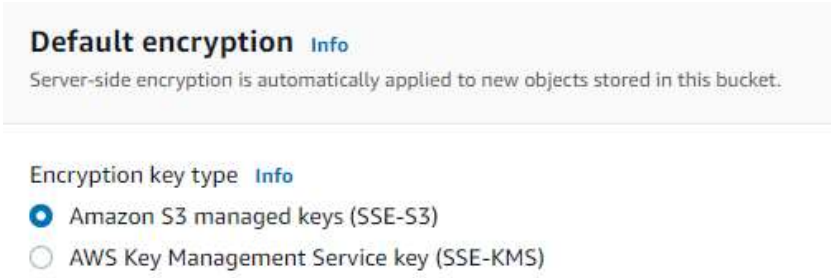
- Bucket policies are an IAM mechanism for controlling access to resources. They are a critical element in securing your S3 buckets against unauthorized access and attacks.
- A bucket policy is a JSON object that is attached to a bucket and allows you to manage access to specific S3 objects in a bucket.
- You can specify permissions for each resource to allow or deny actions requested by a principal (a user or role).
- When you create a new Amazon S3 bucket, you should set a policy granting the relevant permissions to the data forwarder's principal roles.

Amazon S3 encryption

- S3 encryption types
  - Server side encryption
    - Amazon S3 managed keys (SSE-S3)
    - AWS Key Management Service key (SSE-KMS)
  - Client side encryption

Server side encryption with S3 managed keys (SSE-S3)

- All Amazon S3 buckets have encryption configured by default, and objects are automatically encrypted by using server-side encryption with Amazon S3 managed keys (SSE-S3).
- This encryption setting applies to all objects in your Amazon S3 buckets.
- When you use server-side encryption, Amazon S3 encrypts an object before saving it to disk and decrypts it when you download the object.



Server side encryption with KMS managed keys (SSE-KMS)

- If you need more control over your keys, such as managing key rotation and access policy grants, you can choose to use server-side encryption with AWS Key Management Service (AWS KMS) keys (SSE-KMS).

- When you configure your bucket to use default encryption with SSE-KMS, you can also enable S3 Bucket Keys to decrease request traffic from Amazon S3 to AWS KMS and reduce the cost of encryption.

S3 client side encryption

- S3 client side encryption puts all the responsibility for the encryption onto the user. Rather than allowing AWS to encrypt your data, you perform the encryption within your own data center and upload the encrypted data directly to AWS.

- S3 Client-Side Encryption also comes in two options:
  - Using server-side master key storage
  - Using client-side master key storage.

Amazon S3 Labs

Lab 1: S3 bucket with versioning & storage class

Instructions Document: L01-S3-Versioning-Storageclass.txt

In this lab we demo:

1. Creating and using S3 buckets
2. Applying versioning to S3 buckets
3. Applying & changing storage classes for an S3 bucket
4. Defining life cycle rules for an S3 bucket

Lab 2: S3 user policies and bucket policies

Instructions Document: L02-S3-PermissionPolicies.txt

In this lab we demo:

1. User policies: Apply inline policies to a specific user to control what kind of permission he may have one or more S3 buckets

2. Bucket policies: Apply bucket policy to a bucket to control which IAM user/role can access this bucket.

AWS Lambda

AWS Lambda

- **AWS Lambda** is a serverless, event-driven compute service that lets you run code for virtually any type of application or backend service without provisioning or managing servers. You can trigger Lambda from over 200 AWS services and SaaS applications.
- AWS Lambda executes your code only when needed and scales automatically, from a few requests per day to thousands per second.
- You pay only for the compute time you consume - there is no charge when your code is not running

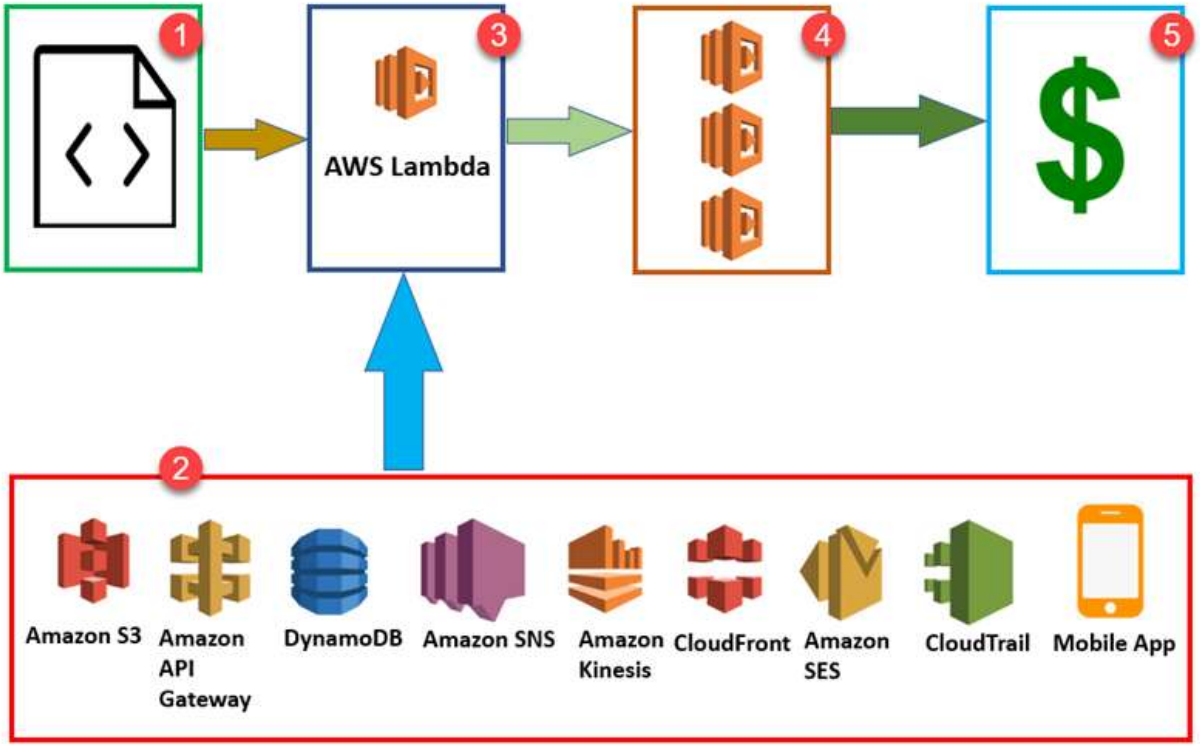
AWS Lambda

- To get working with **AWS Lambda**, we just have to push the code in AWS Lambda service. All other tasks and resources such as infrastructure, operating system, maintenance of server, code monitoring, logs and security is taken care by AWS.

- The code is executed based on response to events in AWS services such as adding/removing files in S3 bucket, updating Amazon DynamoDB tables, sending SNS notifications etc.

- AWS Lambda supports languages such as Java, Node.js, Python, C#, Go, Ruby

How Lambda works ?



When to use AWS Lambda ?

You can use AWS Lambda to create new backend application services triggered on demand using the Lambda application programming interface (API) or custom API endpoints built using Amazon API Gateway.

Some popular use-cases for using Lambda are:

1. Serverless Website
2. Document Conversion at Faster Pace
3. Log Analysis
4. Real-Time Data Processing
5. Predictive Page Rendering
6. Mass Emailing
7. Efficient Monitoring
8. Building Serverless Chatbots

AWS Lambda triggers

- Entry into an S3 object
- Inserting, updating and deleting data in Dynamo DB table
- Push notifications from SNS
- GET/POST calls to API Gateway
- Log entries in AWS Kinesis data stream
- Log history in CloudTrail

and many more..

AWS Lambda function configurations

- **General Configuration**
  - Memory: default 128 MB (CPU is proportional to memory)
  - Ephemeral storage: default 512 MB (/tmp space allocated for the function)
  - Timeout: default 3 sec

- **Triggers** – you can add triggers here. Triggers are events that fire the lambda function.

- **Permissions** – you can manage permissions such as execution role and policies attached to the function.

- **Environment Variables** – you can use these to pass run-time values to the function.
  - Python: `region = os.environ['AWS_REGION']`

- **Destinations** – to add destinations to your function to retain records of your function invocations or to retain discarded events. You can have SNS, SQS, EventBridge and Lambda as destinations.

- **VPC** – if you want to have your function access resources in a custom VPC

AWS Lambda Labs

Lab 1 : Lambda basics demo

- Instructions Document: L01-Lambda-Basics.txt
- Script File: cts-lambda-basics.py

Lab 2 : Lambda to log SNS messages to CloudWatch

- Instructions Document: L02-Lambda-SNS-Cloudwatch.txt
- Script File: cts-lambda-sns-cloudwatch.py

Lab 3 : Lambda to move files from one S3 bucket to another

- Instructions Document: L03-Lambda-S3-S3-Move.txt
- Script File: cts-lambda-s3-s3-move.py

Lab 4 : Lambda to delete files from S3 bucket

- Instructions Document: L04-Lambda-S3-Delete.txt
- Script File: cts-lambda-s3-delete.py

Amazon Athena

Amazon Athena

- Amazon Athena is a serverless interactive query service or interactive data analysis tool which is used for processing complex queries and in a lesser amount of time. Due to its serverless feature, it needs no infrastructure to manage or to setup.

- It can quickly analyze the data with the help of Amazon S3 using standard SQL. It even does not need to load the data in Athena.

- All we require to do is to point to the data in Amazon S3, define the particular schema and start querying using the standard SQL. With the help of Amazon Athena, we can process any of data, whether it is structured, semi-structured or unstructured data, i.e., it can handle the data in CSV, Avro, Parquet and ORC.

- Athena uses S3 as its storage layer.

Amazon Athena

When to use Amazon Athena ?

- You should use Amazon Athena if you want to run interactive ad hoc SQL queries against data on Amazon S3, without having to manage any infrastructure or clusters.
    - Athena is useful if you want to run a quick query on web logs to troubleshoot a performance issue on your site. You just define a table for your data and start querying using standard SQL. You can analyze data stored in CSV, JSON, AVRO, Parquet and ORC data formats.
- Athena integrates with QuickSight for easy data visualization. You can use Athena to generate reports or to explore data with BI tools or SQL clients connected with a JDBC or ODBC driver.
- Athena integrates with the AWS Glue Data Catalog, which offers a persistent metadata store for your data in Amazon S3. This allows you to create tables and query data in Athena based on a central metadata store available throughout your AWS account and integrated with the ETL and data discovery features of AWS Glue.

Amazon Athena Labs

Lab 1 – Athena basic operations

Instructions Document: L01-Athena-Basics.txt

In this lab, we perform the following operations:

- Setup Output S3 bucket for Amazon Athena
- Create a Catalog table and database using Athena query editor using S3 as data source
- Run queries on the table using ANSI SQL
- Managing query and its results
- Download the results of the query in CSV format

Lab 2 – Create table using Glue catalog and Glue crawler

Instructions Document: L02-Athena-Glue-Catalog.txt

In this lab, we perform the following operations:

- Create a metadata table using Glue data catalog from S3 bucket
- Query the table from Athena
- Create a metadata table using Glue crawler by crawling an S3 bucket
- Query the table from Athena using ANSI SQL

Lab 3 – Create table using Hive DDL command

Instructions Document: L03-Athena-Hive-DDL.txt

In this lab, we perform the following operations:

- Create a metadata table in Athena query editor using Hive DDL command
- Query the data using Athena

AWS Glue

AWS Glue

- AWS Glue is a **serverless data integration and managed ETL service**.
- AWS Glue makes it easy to discover, prepare, and combine data for analytics, machine learning, and application development.
- AWS Glue provides all the capabilities needed for data integration so that you can start analyzing your data and putting it to use in minutes instead of months.

Benefits of AWS Glue

- AWS Glue can scan through all the available data with a crawler and discovers and stores your schemas.
- Final processed data can be stored in many different places (Amazon RDS, Amazon Redshift, Amazon S3 etc.)
- It's a cloud service, hence no upfront investments required for on-premises infrastructure.
- It's a serverless ETL service, hence no provisioning of hardware required.
- It gives you the Python/Scala ETL code right off the bat.

When to use Glue?

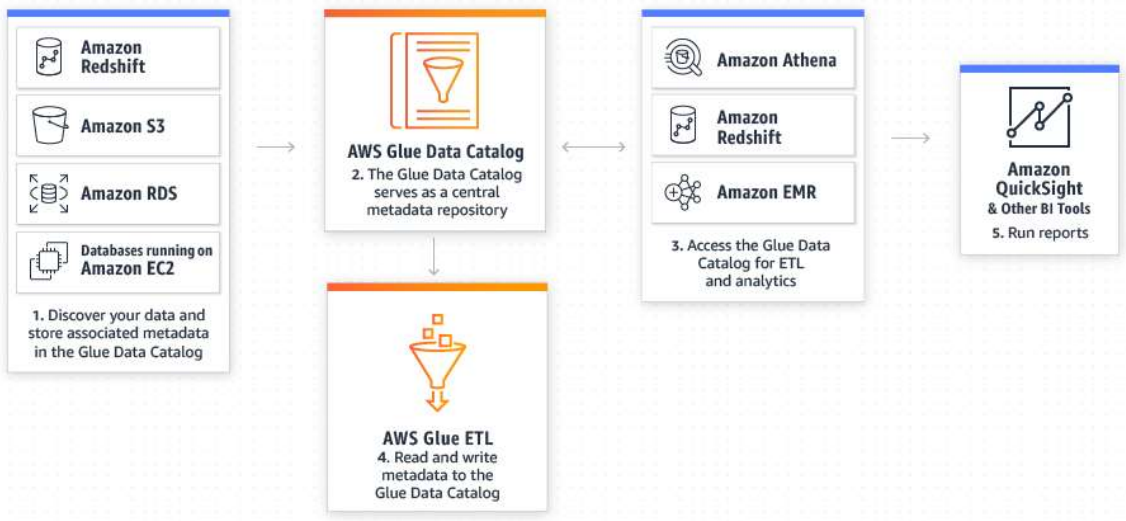
- Build event-driven ETL pipelines

  AWS Glue can run your ETL jobs as new data arrives. For example, you can use an AWS Lambda function to trigger your ETL jobs to run as soon as new data becomes available in Amazon S3. You can also register this new dataset in the AWS Glue Data Catalog as part of your ETL jobs.

When to use Glue?

- Create a unified catalog to find data across multiple data stores

  You can use the AWS Glue Data Catalog to quickly discover and search across multiple AWS data sets without moving the data. Once the data is cataloged, it is immediately available for search and query using Amazon Athena, Amazon EMR, and Amazon Redshift Spectrum.

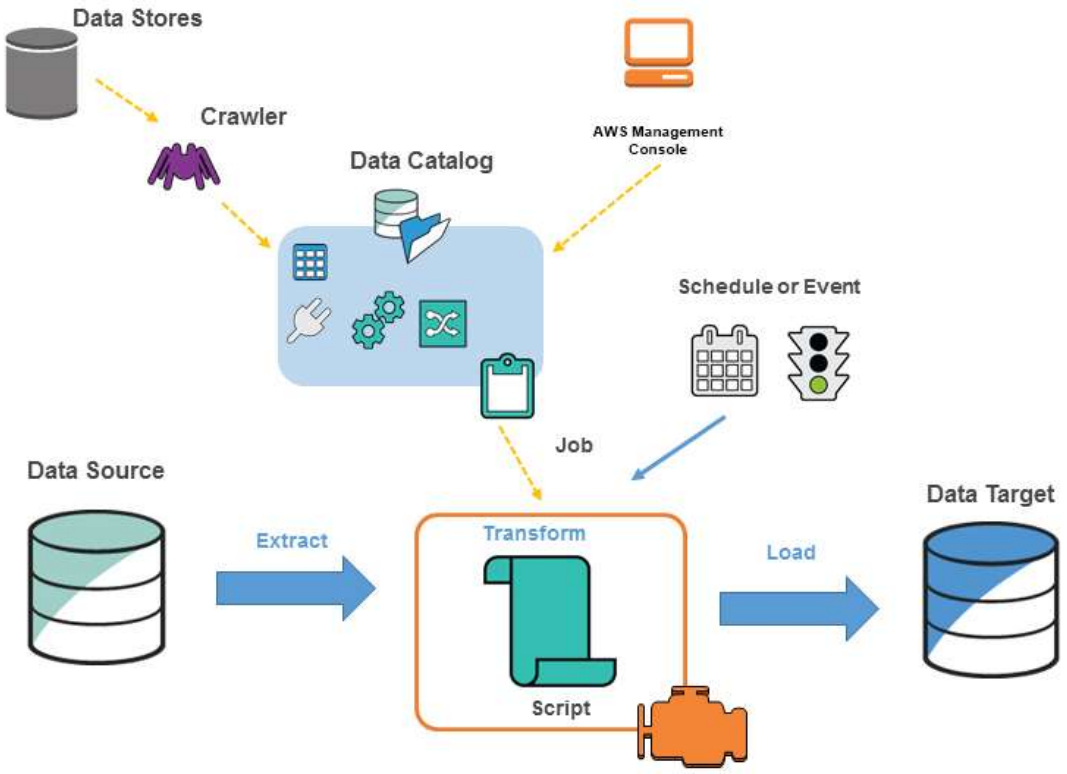


When to use Glue?

- Create, run, and monitor ETL jobs without coding

  AWS Glue Studio makes it easy to visually create, run, and monitor AWS Glue ETL jobs. You can compose ETL jobs that move and transform data using a drag-and-drop editor, and AWS Glue automatically generates the code. You can then use the AWS Glue Studio job run dashboard to monitor ETL execution.

AWS Glue Architecture



AWS Glue components

- **Data catalog**
  The data catalog holds the metadata and the structure of the data.
- **Database**
  It is used to create or access the database for the sources and targets.
- **Table**
  Create one or more tables in the database that can be used by the source and target.
- **Crawler**
  A crawler is used to retrieve data from the source using built-in or custom classifiers. It creates/uses metadata tables that are pre-defined in the data catalog.

AWS Glue components

- **Job**
  A job is business logic that carries out an ETL task. Internally, Apache Spark with Python or Scala language writes this business logic.

- **Trigger**
  A trigger starts the ETL job execution **on-demand** or on a **schedule** or when a **job or crawler event** is generated.

- **Development endpoint**
  Creates a development environment where the ETL job script can be tested, developed, and debugged.

- **Workflow**
  A workflow is an orchestration used to visualize and manage the relationship and execution of multiple triggers, jobs and crawlers.

AWS Glue Labs

Lab 1 – Basics of a Glue Job (CSV to Parquet)

Instructions Document: L01-Glue-Basics-CSV-to-Parquet.txt

In this lab, we perform the following operations:

- Create Crawler and Catalog table for flights data
- Validate using Athena
- Create a Policy and a Role to access S3 from Glue
- Create a Glue Job to convert file format from CSV to Parquet
- Run and Monitor the job
- Create Catalog table on top of new location with parquet file format
- Validate new table using Athena

Lab 2 – Create and run a Glue trigger

Instructions Document: L02-Glue-Triggers.txt

In this lab, we perform the following operations:

- Create a Glue trigger to invoke a job
- Run the trigger to execute the job
- Validate the results using Athena

Lab 3 – Setup and run a Glue workflow

Instructions Document: L03-Glue-Workflow.txt

In this lab, we perform the following operations:

- Build the workflow for the pipeline
  - Crawl the source folder for flights data to refresh the table
  - Run Glue Job to convert the file format
  - Crawl the target folder to refresh the table
- Run and monitor the workflow
- Validate both source and target tables using Athena

Lab 4 – Glue ETL pipeline using PySpark - S3 to S3

- Instructions Document: L04-Glue-S3-to-S3-ETL.txt
- PySpark Script File: glue-s3-to-s3.py

In this lab, we perform the following operations:

- Load the required data to S3
- Create an IAM Role with required policies
- Create and run a Glue crawler
- Create the Glue job and provide the ETL script (PySpark code)
- Run the job and validate the results

Lab 5 – Glue PySpark job with Joins

- Instructions Document: L05-Glue-ETL-PySpark-Joins.txt
- PySpark Script File: moviesratings-joins.py

In this lab, we perform the following operations:

- Load the required data to S3
- Create and run the Glue crawlers
- Create the Glue job and provide the PySpark script
- Run the job and validate the results

AWS PROJECTS

Creating end-to-end data pipelines using AWS services

Project 1 – Change data capture from RDS(MySQL) to S3

Instructions Document: Change-Data-Capture-RDS-S3.txt

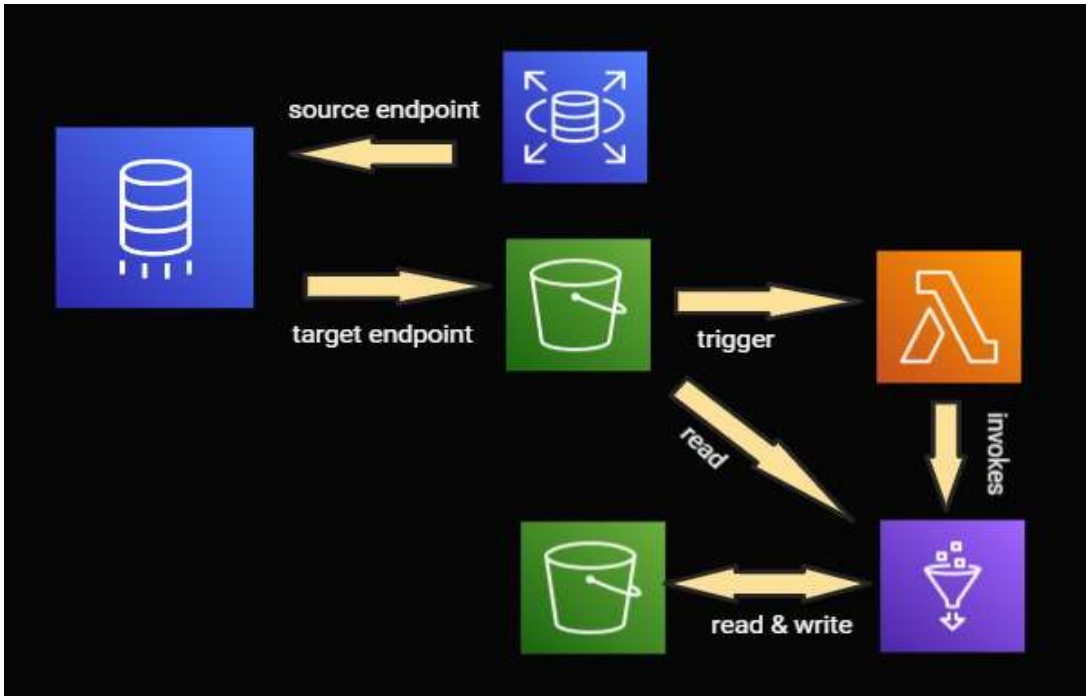
In this project, we capture the changes that happen to an RDS database (MySQL) and replicate those changes to an S3 bucket. This process involves initial load and incremental loads whenever data is inserted, updates or deleted in the source database and updates the data in S3 bucket so that the S3 bucket always holds the current data of the database

AWS services used in this lab:

- Amazon RDS (MySQL instance)
- AWS Database migration service (DMS)
- AWS Glue
- AWS Lambda
- S3
- PySpark

Project 1 – Change data capture from RDS(MySQL) to S3

Instructions Document: Change-Data-Control-RDS-S3.txt



Project 2 – Real-time clickstream anomaly detection

Instructions Document: Real-Time-Clickstream-Anomaly-Detection.txt

In this project, we use Kinesis services such as Firehose delivery streams and Kinesis data analytics application to detect anomalies from the real time data ingested into the streams and alert the user by sending an email whenever an anomaly is detected in the streaming data in real time.

AWS services used in this lab:

- AWS CloudFormation
- Amazon Kinesis Data Firehose
- Amazon Kinesis Data Analytics
- Amazon Kinesis Data Generator
- AWS Lambda
- Amazon S3
- Amazon SNS

Project 2 – Real-time clickstream anomaly detection

Instructions Document: Real-Time-Clickstream-Anomaly-Detection.txt

THANK YOU