**AWS (Amazon web services**) is a cloud computing platform with an ever-expanding set of services.

AWS was launched in **2006**.

Started with three major services **Compute, Storage and Messaging**.

AWS is a **CAAS (Compute as a service).**

**AWS uses multi-tenancy with hypervisors.**

Multi-tenancy: Sharing underlying hardware between virtual machines.

Everything in AWS is built on REST APIs

**Ways to interact with AWS Services**

1. **AWS Management Console**: The AWS Management Console includes wizards and workflows that you can use to complete tasks in AWS services.
2. **AWS SDKs (software development kit)** (AWS SDK for Python (Boto3)): Software development kits (SDKs) enable you to develop AWS applications in supported programming languages.
3. **AWS Command Line Interface (CLI)**: The AWS Command Line Interface (AWS CLI) is used to automate actions for AWS services and applications through scripts.
4. AWS Elastic BeanStalk and AWS CloudFormation

AWS Core Technologies

Compute | Storage | Database | Security | Management | Networking

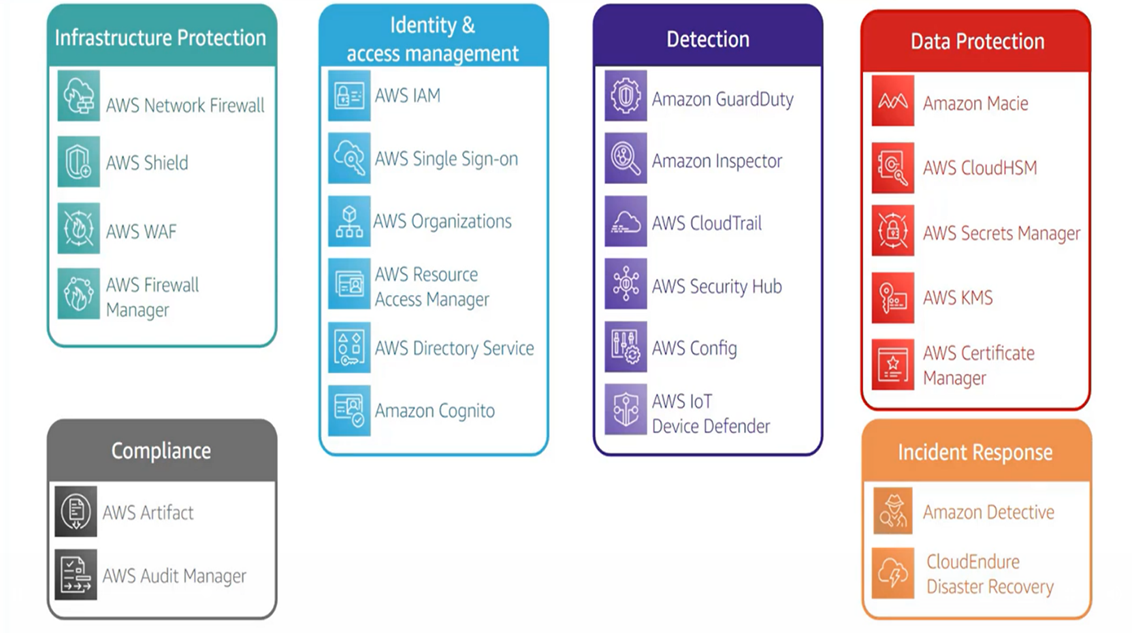
Vertical Scaling VS Horizontal Scaling

**Horizontal scaling (scaling out)** refers to adding additional [nodes](https://medium.com/coinmonks/what-are-nodes-nodes-in-a-nutshell-f5d567bc9662) or machines to your [infrastructure](https://www.cloudzero.com/blog/cloud-infrastructure) to cope with new demands.

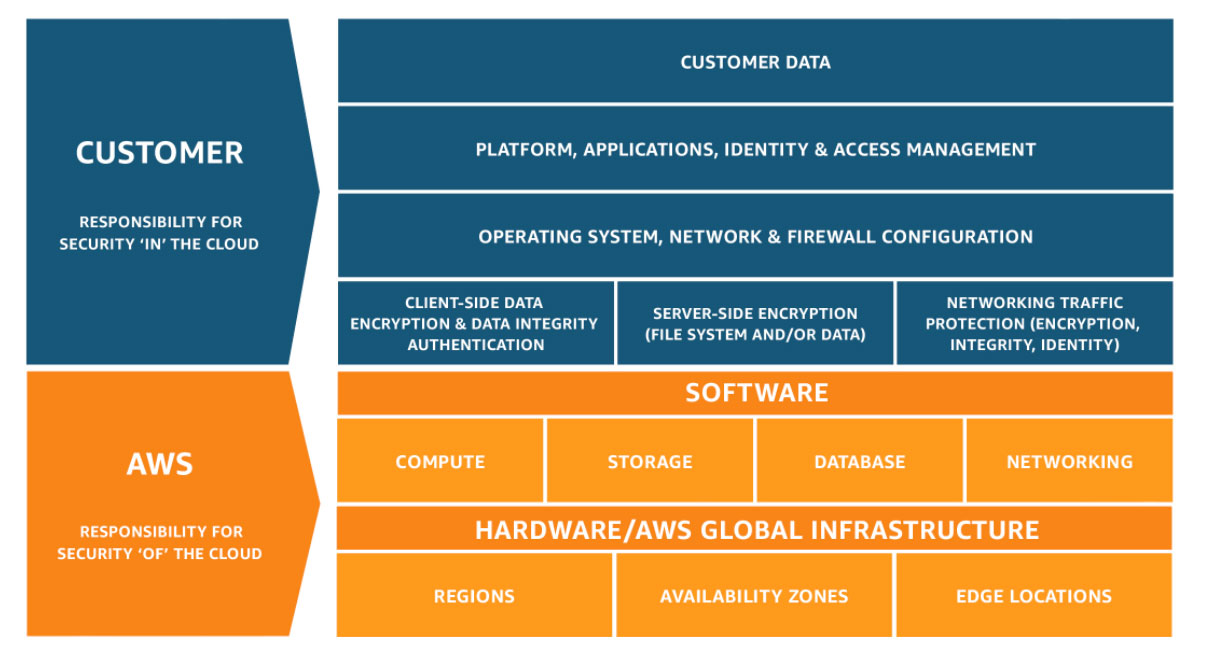
**Vertical scaling (scaling in)** describes adding more power to your current machines.

For instance, if your server requires more processing power, vertical scaling will mean upgrading the CPUs. You can also vertically scale the memory, storage, or network speed.

**Security**

****

*AWS shared responsibility model*



# IAM (Identity and access management)

## **User groups**

A user group is a collection of IAM users. Use groups to specify permissions for a collection of users.

## **Users**

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

## **Roles**

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short duration. Roles can be assumed by entities that you trust.

## **Policies**

A policy is a JSON object in AWS that defines permissions.

The Seven Rs for Migrating approach

A picture containing diagram

Description automatically generated

Cloud Architecture Best Practices

* ***Design for failure and nothings fails***
  + Avoid single points of failure
  + Multiple Instances
  + Multiple Availability zones
  + Separate single server into multiple tiered application
  + For Amazon RDS, use multi-AZ feature
* ***Build security in every layer***
  + Encrypt Data in rest and in transit
  + Enforce principle of least privilege in IAM
  + Implement both security groups and Network Access control lists (NACL)
  + Consider advanced security features and services
* ***Leverage different storage option***
  + Move static web assets to Amazon S3
  + Use Amazon CloudFront to serve globally
  + Store session state in DynamoDB
  + Use ElastiCache between hosts and databases
* ***Implement elasticity***
  + Implement Auto Scaling polices
  + Architect resiliency to reboot and relaunch
  + Leverage managed service like Amazon S3 and Amazon DynamoDB
* ***Think parallel***
  + Scale Horizontally, not vertically
  + Decouple compute from session/ state
  + Use Elastic Load Balancing
  + Right size your infrastructure
* ***Loose coupling sets you free***
  + Instead of single, ordered workflow, use multiple queues
  + Use Amazon Simple Queue Service (SQS) and Simple Notification Service (SNS)
* ***Don’t fear constraints***
  + Rethink traditional constraints

**AWS Organizations**

In AWS Organizations, you can apply **service control policies (SCPs)** to the organization root, an individual member account, or an OU. An SCP affects all IAM users, groups, and roles within an account, including the AWS account root user.

You can apply IAM policies to IAM users, groups, or roles. You cannot apply an IAM policy to the AWS account root user.

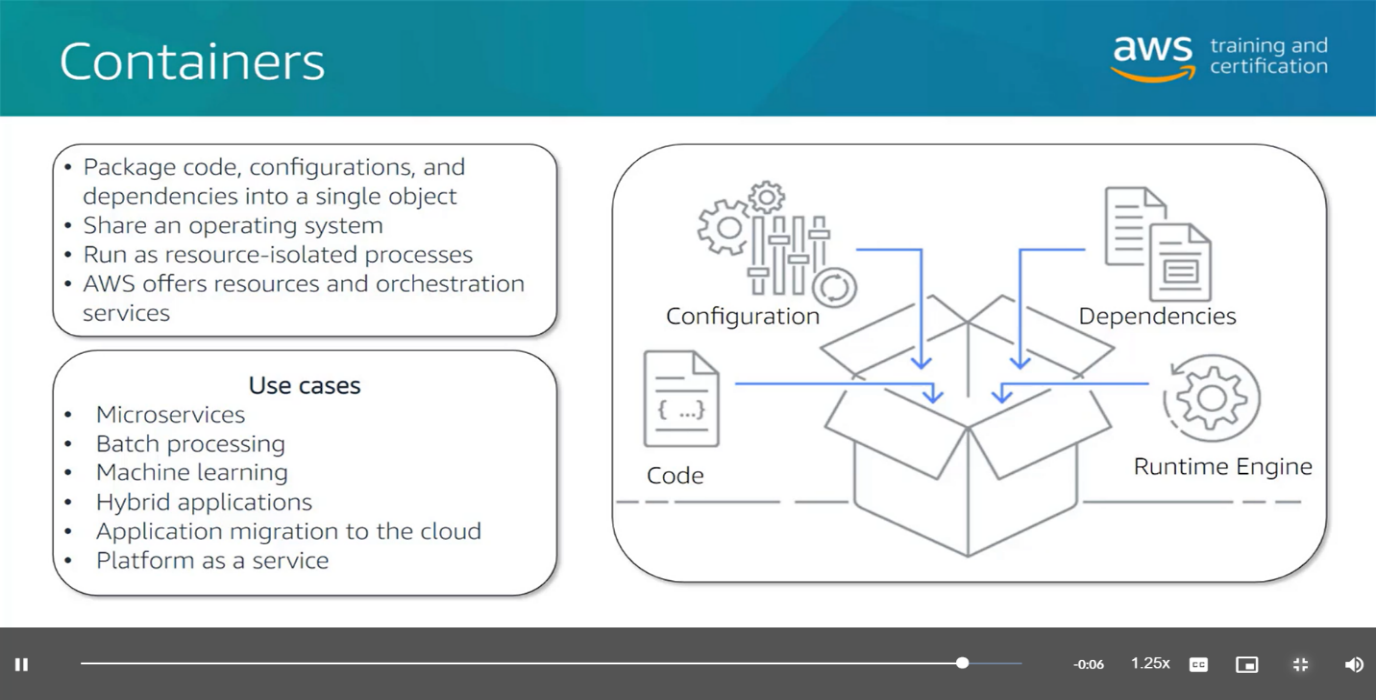
Migration Process

Graphical user interface, text

Description automatically generated

Modernization

**Containers**



**Serverless**

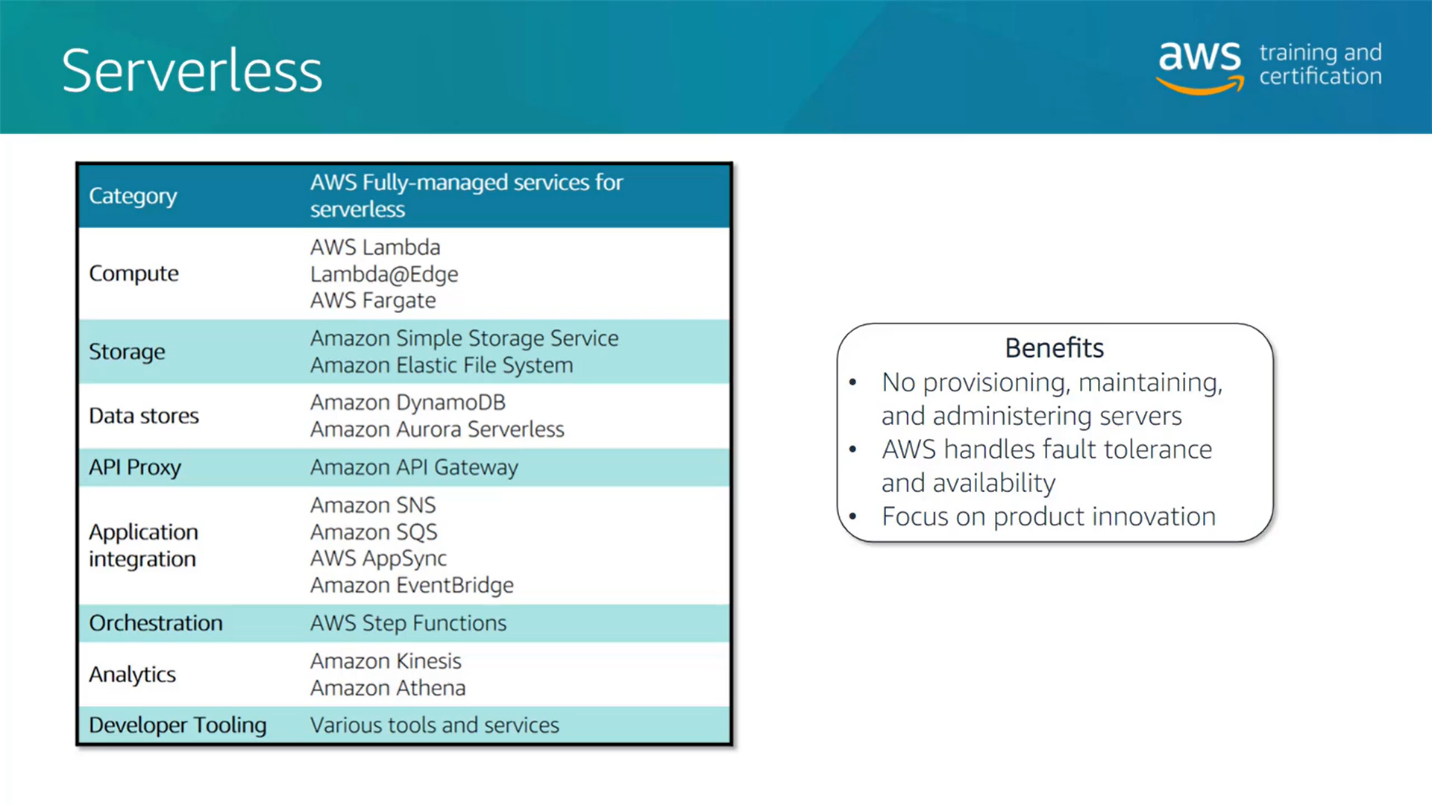
The term “serverless” means that your code runs on servers, but you do not need to provision or manage these servers. With serverless computing, you can focus more on innovating new products and features instead of maintaining servers.

Another benefit of serverless computing is the flexibility to scale serverless applications automatically. Serverless computing can adjust the applications' capacity by modifying the units of consumptions, such as throughput and memory

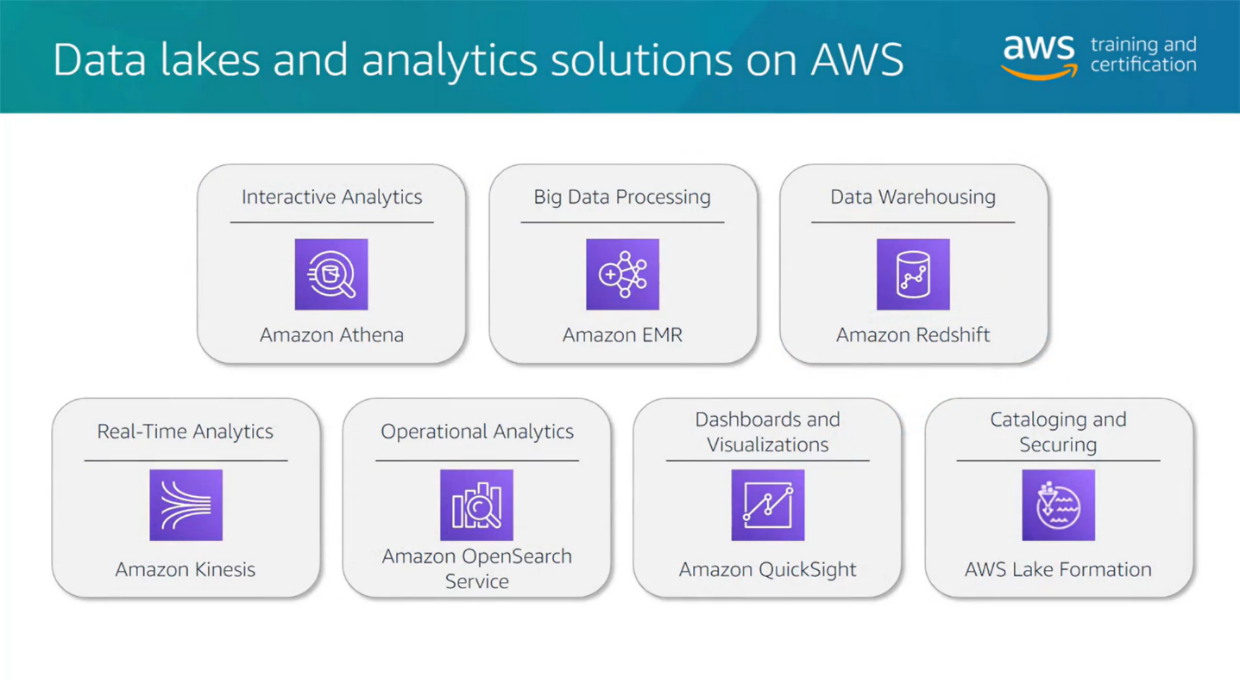
***Serverless = Function AS A Service (FAAS) model.***

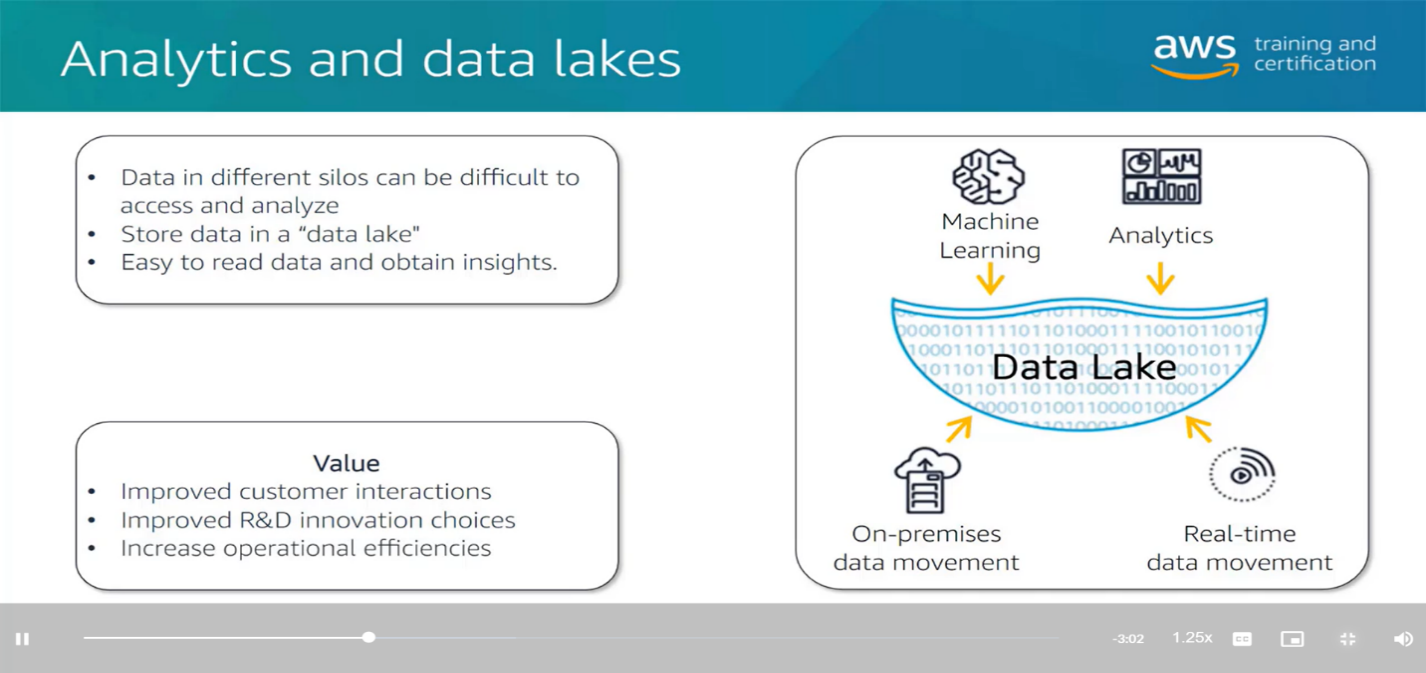
**A screenshot of a computer

Description automatically generated with medium confidence**

****

**Analytics and data lakes**

****

****