Cloud Applications Architecture

AWS Intro

AWS is one of the many cloud providers available today. This wasn't always the case. Amazon launched it in 2006 to increase their revenue by renting their computing resources - they had a lot of spare capacity left after scaling their resources for holiday sales. It took several years for any competitor to respond, which translated to an overwhelming market share owned by AWS even today and to more mature services.

It currently offers over 200 services for various use cases (hosting, AI, ML, IoT, media encoding and streaming, etc.). Some of them are very specific to a certain use case while others are more general and most likely are the building blocks of the rest. We will focus on the latter.

AWS Regions

Amazon cloud computing resources are hosted in multiple locations worldwide. These locations are composed of AWS Regions, Availability Zones, and Local Zones. Each AWS Region is a separate geographic area. Each AWS Region has multiple, isolated locations known as Availability Zones.

* They are fundamentally different/independent clouds.

# AWS Console

is a management interface that acts as a central entry point for all AWS services

* In addition to the console, AWS cloud resources can also be managed through the AWS CLI or programmatically by using the AWS REST API/SDK.

# EC2

Amazon Elastic Compute Cloud

# AMI

An Amazon Machine Image (AMI) provides the information required to launch an instance, including the operating system and additional software.

# CIDR

Classless Inter-Domain Routing (CIDR) is a method for allocating IP addresses and for IP routing.

* under CIDR address space is allocated to Internet service providers and end-users on any address-bit boundary.

# VPC

* Amazon Virtual Private Cloud (Amazon VPC) lets you provision a logically isolated section of the AWS Cloud where you can launch AWS resources in a virtual network that you define.
* VPCs allow you to control the traffic between (sub)networks and use private IP addresses.
* AWS resources can be launched into a specified subnet. Use a **public subnet for resources that must be connected to the internet** and use a **private subnet for resources that are to remain isolated from the internet**.

# Security Groups

* A security group acts as a **virtual firewall** for your instance to **control inbound and outbound traffic**.
* **Security groups act at the instance level**, not the subnet level. Therefore, each instance in a subnet in your VPC can be assigned to a different set of security groups.

# Load Balancers

* Load balancing refers to the process of distributing a set of tasks over a set of resources (computing units), with the aim of making their overall processing more efficient.
* **Elastic Load Balancing**, the **AWS service for loading balancing**, supports the following types of load balancers:
  + **Application** - This is an actual proxy between the internet and your application. It receives a request from a client and makes another request (with the same data) to your application. It offers tons of features and it suits very well in most cases. One important tip about it is that since the ALB creates another request, but, for some reason, you need the IP address of the original client, you can look at the request header x-forwarded-for. **ALB operates at layer 7** (HTTP, WebSockets) of the OSI model,
  + **Network** - You can look at it like a (very sophisticated) network router. The **NLB handles traffic at layer 4 (TCP/UDP)** thus working with packets. You lose some features of the ALB, but gain massive performance (and scalability) and the request looks like it came directly from the original client. Also it is **cheaper than an ALB**.
  + There is a third option, **classic**, but it's deprecated.
  + (NEW) **Gateway** - **It operates at layer 3 (network**). You can use this if you need to integrate other virtual appliances such as deep packet inspection systems.

# ASG

* It contains a collection of Amazon EC2 instances that are treated as a logical grouping for the purposes of automatic scaling and management. An ASG also enables you to use Amazon EC2 Auto Scaling features such as health check replacements and scaling policies. Both maintaining the number of instances in an Auto Scaling group and automatic scaling are the core functionality of the Amazon EC2 Auto Scaling service.

# CloudFormation

* create a template that describes all the AWS resources that you want (like Amazon EC2 instances or Amazon RDS DB instances), and CloudFormation takes care of provisioning and configuring those resources for you.
* <https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/Welcome.html>

# S3

* Simple Storage Service
* It stores data (objects). It is highly durable (your data will not get corrupted), highly available (it basically never goes down) and allows you to store virtually an infinite amount of data.
* it works with HTTP(S)
* we first create a bucket (a container for our files). Each bucket has its own URL in the form of https://<bucket\_name>.s3.<region>.amazonaws.com/<file\_name>.

# Object vs Block Storage

* Block storage: This is what our computers use to store all our data (and OS and programs). Files are divided into blocks of data.
* Object storage: stores files as one big chunk of data and metadata that is identified by an ID (usually the name of the file). The main consequence is that object storage doesn’t allow us to update/edit files. If we want to update a file, we re-upload the whole file.

# Buckets

* A bucket is a container for objects. To store your data in Amazon S3, you first create a bucket and specify a bucket name and AWS Region.
* Buckets are private by default.
* if we use a bucket to host our website, we should make it public.

# CDN

* content delivery network
* used to serve files as fast as possible around the globe. It is basically a highly distributed network
* AWS’s CDN is CloudFront and the network is made of edge locations.

# S3 Configuration Methods

* ACL - Access control list: Each bucket and object has an ACL attached to it as a subresource. It defines which AWS accounts or groups are granted access and the type of access.
* Resource-based policies - are attached to a resource.
* IAM: Attach managed and inline policies to IAM identities (users, groups to which users belong, or roles). Identity-based policies grant permissions to an identity.

# Policies

Text

Description automatically generated with medium confidence

# RDS

* Relational Database Service
* tell AWS the hardware specs and database engine that we need (and some other info – mostly security and availability) and it will run and manage the database for us.
* supports several database engines such as MySQL, MariaDB (an open-source fork of MySQL), PostgreSQL

# Docker

* is a container engine.
* A container is an isolated environment where we can run programs with all its dependencies.

# ECS cluster

* uses EC2 instances as the underlying infrastructure on which to deploy container instances.

# Task definition

* The task definition tells AWS ECS how it should start your container.

# IAM Concepts

* Users: IAM users represent **entities that interact with the AWS resources under your account**.
* Groups
* Policies: policies are documents (JSON) describing what an entity can or cannot do.
* Roles: do not represent an entity, but they can be assumed by an entity. They can also **be assumed by resources** such as EC2 instances.

# Questions:

1. What database services does AWS offer for NoSQL databases?

Amazon Dynamo DB.

1. What is the S3 service?

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

1. What service can be used to build applications using a FaaS (function as a service) paradigm?

AWS Lambda.

1. Name 2-3 services based on object storage that most people are using frequently.

Anything google cloud based, amazon web services based, Microsoft azure based.

1. Graphical user interface, text, application, chat or text message

   Description automatically generated

LAYER, LOAD BALANCER

1. Graphical user interface, text, application

   Description automatically generated

AWS REGION, incorrect, a doua?? varianta este cea corecta

1. Graphical user interface, text

   Description automatically generated

S3, SIMPLE STORAGE SERVICE

1. Shape, rectangle

   Description automatically generated

RDS, database

1. Graphical user interface, text

   Description automatically generated

EC2, Configuration

1. Graphical user interface, text, application

   Description automatically generated

SOFTWARE IN CLOUDFORMATION, Incorrect

1. Shape, rectangle

   Description automatically generated

S3, HTTP

1. Shape

   Description automatically generated

EC2, AZ, Availability zone

1. Text

   Description automatically generated with medium confidence

DOCKER

1. Graphical user interface, text, application, email

   Description automatically generated

EC2, SSH

1. A picture containing text

   Description automatically generated

S3, CLOUDFRONT

1. Graphical user interface, text

   Description automatically generated

ASG

1. Text

   Description automatically generated
2. Diagram

   Description automatically generated

Diagram

Description automatically generated