EC2, ECS and Lambda

Compute services in the cloud are used for the provision of processing power/compute capacity required to run a workload. So, the virtual machines that are going to be running your applications on the cloud for example.

AWS has three main compute services, namely: **EC2**, **ECS** and **Lambda**. These are discussed further below:

EC2 (Elastic Compute Cloud): For the purposes of this book, the definition of a Virtual Machine (VM) given in the background, i.e. any software-based emulation of a physical computer that is being run on hardware infrastructure owned by the cloud provider (in our case, AWS), should suffice. VMs allow us to create and manage an isolated and portable computer with its own OS (Operating System) on the servers owned by someone other than ourselves, a “Virtual” machine if you will.

EC2 is the AWS service that allows us to rent virtual machines in the cloud, known as instances, offering a wide range of flexibility. These instances can be optimized for various use cases, including memory, compute, or storage. Said EC2 instances are highly customizable, allowing users to configure CPU, memory, storage, and networking capacity according to their needs. They also come with security features like security groups and IAM roles for access control.

Mentioned features as well as the different types of EC2 instances will be discussed in future sections.

ECS (Elastic Container Service): Containers are the “hot new thing” that’s gaining popularity throughout the IT world due to their cost-effective computations. As monolithic applications continue to be replaced by microservice applications, it’s only a matter of time before container usage becomes more prevalent than instance usage.

Docker, the most widely used containerization technology, defines containers as “a standard unit of software that packages up code and all its dependencies such that the application can run quickly and reliably, even after being transferred from one computing environment to another.” Put simply, containers are standardized, lightweight, and secure pre-packaged solutions that run efficiently no matter what situation they’re thrown into. Also worth mentioning is that these containers are usually, though not exclusively, created using Docker and its related packages.

ECS is a managed container orchestration service designed for running containerized applications. While it supports and is mostly used to run Docker containers, though it has made an active effort not to be limited to just docker containers, with the service accommodating all containers built to the Open Container Initiative (OCI) image format. Users can manage ECS clusters themselves on EC2 instances or opt for AWS Fargate, a serverless option that abstracts away infrastructure management. With Fargate, users focus solely on deploying and managing containers, without dealing with server provisioning.

A more detailed exploration of Elastic Container Service, and its related services and configurations will be discussed in future sections.

Lambda: Functions are one of the most basic yet powerful concepts in programming, and can be understood as self-contained pieces of code designed to accomplish a certain task. In order to calculate the sum of two numbers for example, a programmer could create a self-contained piece of code called *sumCalc* that would take two numbers, let us call them *num1* and *num2* as arguments, usually represented as: `sumCalc(num1, num2)`

Lambda is a serverless compute service that allows users to run function code such as the one mentioned above without managing servers. Users upload their code, and Lambda handles scaling and infrastructure management automatically. When a Lambda function is invoked, users receive an ARN (Amazon Resource Name) that uniquely identifies the function. Lambda is commonly used with API Gateway to create serverless REST APIs, enabling infinitely scalable endpoints. Lambda prioritizes simplicity, offering a streamlined development experience without the burden of managing infrastructure.

A more detailed exploration of Lambda and its associated configurations will be discussed in future sections.