What is Kubernetes?

This lesson will get us introduced to the world of Kubernetes.

WE'LL COVER THE FOLLOWING

Don’t Run Containers Directly

To understand Kubernetes, it is important to realize that running containers directly is a bad option for most use cases. Containers are low-level entities that require a framework on top. They need something that will provide all the additional features we expect from services deployed to clusters. In other words, containers are handy but are not supposed to be run directly.

The reason is simple. Containers, by themselves, do not provide fault tolerance. They cannot be deployed easily to the optimum spot in a cluster, and, to cut a long story short, are not operator friendly. That does not mean that containers by themselves are not useful. They are, but they require much more if we are to harness their real power. If we need to operate containers at scale, be fault tolerant and self-healing, and have the other features we expect from modern clusters, we need more. We need at least a scheduler, probably more.

The Kubernetes

Kubernetes was first developed by a team at Google. It is based on their experience from running containers at scale for years. Later on, it was

donated to Cloud Native Computing Foundation (CNCF). It is a true open source project with probably the highest velocity in history.

Why Kubernetes?

Let's discuss how Kubernetes is not only a container scheduler but a lot more.

- We can use it to deploy our services, to roll out new releases without downtime, and to scale (or de-scale) those services.
- It is portable.
- It can run on a public or private cloud.
- It can run on-premise or in a hybrid environment.
- We can move a Kubernetes cluster from one hosting vendor to another without changing (almost) any of the deployment and management processes.
- Kubernetes can be easily extended to serve nearly any needs. We can choose which modules we'll use, and we can develop additional features ourselves and plug them in.
- Kubernetes will decide where to run something and how to maintain the state we specify.
- Kubernetes can place replicas of a service on the most appropriate server, restart them when needed, replicate them, and scale them.
- Self-healing is a feature included in its design from the start. On the other hand, self-adaptation is coming soon as well.
- Zero-downtime deployments, fault tolerance, high availability, scaling, scheduling, and self-healing add significant value in Kubernetes.
- We can use it to mount volumes for stateful applications.
- It allows us to store confidential information as secrets.

- We can use it to validate the health of our services.
- It can load balance requests and monitor resources.
- It provides service discovery and easy access to logs.

And so on and so forth.

Concluding Remarks

The list of what Kubernetes does is long and rapidly increasing. Together with Docker, it is becoming a platform that envelops the whole software development and deployment lifecycle.

The Kubernetes project has just started. It is in its infancy, and we can expect vast improvements and new features coming soon. Still, do not be fooled with “infancy”. Even though the project is young, it has one of the biggest communities behind it and is used in some of the biggest clusters in the world.

Do not wait. Adopt it now!

Getting anxious to go hands-on with Kubernetes? In the next chapter, we will start running a Kubernetes cluster on our machine.