Getting Started with Communication

In this lesson, we will find out why we need to establish communication between Pods.

WE'LL COVER THE FOLLOWING

The Problem

Pods are the smallest unit in Kubernetes and have a relatively short life-span. They are born, and they are destroyed. They are never healed. The system heals itself by creating new Pods (cells) and by terminating those that are unhealthy or those that are surplus. The system is long-living, Pods are not.

Controllers, together with other components like the *scheduler*, are making sure that the Pods are doing the right thing. They control the scheduler. We used only one of them so far.

ReplicaSet is in charge of making sure that the desired number of Pods is always running. If there's too few of them, new ones will be created. If there's too many of them, some will be destroyed. Pods that become unhealthy are terminated as well. All that, and a bit more, is controlled by ReplicaSet.

The problem with our current setup is that there are no communication paths. Our Pods cannot speak with each other. So far, only containers inside a Pod can talk with each other through `localhost`. That led us to the design where both the API and the database needed to be inside the same Pod. That was a lousy solution for quite a few reasons.

The main problem is that we cannot scale one without the other. We could not design the setup in a way that there are, for example, three replicas of the API and one replica of the database. The primary obstacle was communication.

Truth be told, each Pod does get its own address. We could have split the API and the database into different Pods and configure the API Pods to communicate with the database through the address of the Pod it lives in.

However, since Pods are unreliable, short-lived, and volatile, we cannot assume that the database would always be accessible through the IP of a Pod. When that Pod gets destroyed (or fails), the ReplicaSet would create a new one and assign it a new address.

We need a stable, never-to-be-changed address that will forward requests to whichever Pod is currently running.

The Solution

Kubernetes Services provide addresses through which associated Pods can be accessed.

Let’s see Services in action.

> ℹ All the commands from this chapter are available in the 05-svc.sh Gist.

Creating A Cluster

You know the drill. Every chapter starts by pulling the latest code from the vfarcic/k8s-specs repository, and with the creation of a new Minikube cluster.

```
cd k8s-specs
git pull
minikube start --vm-driver=virtualbox
kubectl config current-context
```

Now we have the latest code pulled and the Minikube cluster is running (again).

In the next lesson, we will proceed to the first example of Services.