Kubernetes is the right place for serverless

Abhishek Tiwari

Citation: *A. Tiwari*, "Kubernetes is the right place for serverless", Abhishek Tiwari, 2017. doi:10.59350/kmqs7-bjw36

Published on: January 20, 2017

Function as a Service (FaaS) remains at the core of the serverless movement. That said, most of practical serverless implementations I have seen use a combination of serverless functions and more conventional microservices running in containers. Instead of a one-size-fits-all approach, i.e. trying to use serverless function for everything, a balanced approach is highly recommended. To a large extent, Kubernetes is best placed to bring serverless functions and conventional microservices together seamlessly.

At the beginning, there was only one FaaS implementation available for Kubernetes - Funktion. Funktion introduced event driven lambda style programming model on top of Kubernetes but not without limitations. However, recently several serverless frameworks introduced support for running functions on Kubernetes including but not limited to IronFunctions and Fission.

Fission is a serverless framework for Kubernetes with focus on developer productivity and high performance. Fission maintains a fleet of warm containers with dynamic loader waiting to load functions which makes Fission fast. With Fission, typical cold-start latencies are in 100 millisecond range. As Fission runs on Kubernetes, existing logging and monitoring for Kubernetes pods can be used which makes it ops friendly.

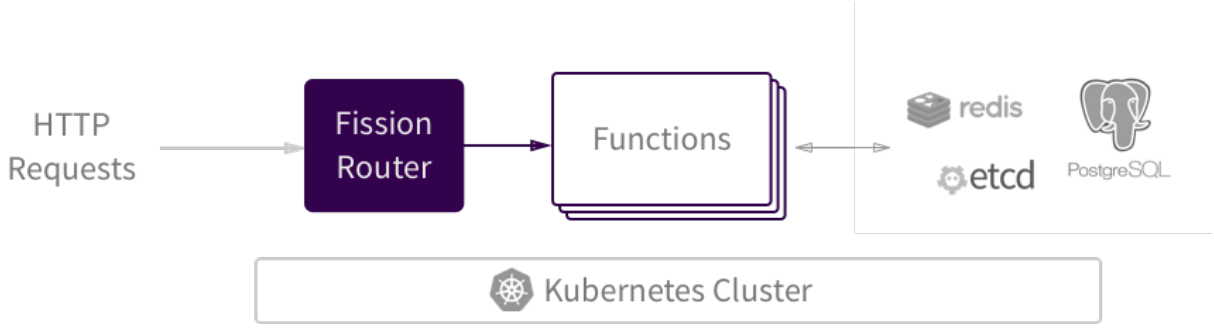

Figure 1: Just write functions, and map them to Fission routes. Image Credits Fission.io

Moreover, Fission abstracts away containers by default - no containers to build, and no container registries to manage. Developer just need to write short lived functions in the language of their choice, and map them to Fission router. You can map a HTTP request, a webhook or an event trigger to your route in order to invokes your function. Fission takes care of the rest: deployment, routing, scalability, availability.

```
$ cat hello.js
module.exports = function(context, callback) {
    callback(200, "Hello, world!\n");
}

$ fission function create --name hello --env nodejs --code hello.js

$ fission route add --function hello --url /hello
```

```
$ curl http://router.fission/hello
Hello, world!
```

At this stage both, IronFunction and Fission, are yet to be production ready. That said, I am more excited about seamless possibilities where both conventional microservices and serverless functions will run on same container infrastructure. This will offer better interoperability, seamless integration and lower latencies. And all of that without dealing with a vendor specific function flavour.