Running our own Heroku… It shouldn't be that hard, right?

We have a small set of servers we use to run our internal applications. Nothing too complex, just monitoring, our ELK stack, Jenkins, and a few internal services.

Given our rather modest requirements it may seem obvious that our first attempt at deployment automation, Chef, was a bit overkill for our needs. Not only that, we also wanted our engineers to be able to easily deploy applications to our servers without having to set up a Chef recipe — like the role Heroku plays in many of our client projects. We could have decided to run our internal applications on Heroku as well, but their pricing model wasn't compatible with our relatively small-scale requirements.

We researched a number of solutions and decided to go with Kubernetes. Having read positive reviews it initially seemed like a great solution to many of the problems we had encountered with Chef:

Everything ran in containers

The system automatically managed multiple instances of individual applications

The configuration was much less complex than Chef

However, a primary goal of our solution was to automate the process of going from code in a repository to a deployed application in our cluster — much like Heroku. Kubernetes did not appear have any mechanism to support this type of workflow, nor a handful of other features we had hoped to find.

Not all who wander are lost, but we were...

Clearly we were missing something, so we went to StackOverflow to search for answers.

As you can see from this incredibly helpful answer from Bill Prin on StackOverflow, what we actually wanted was a PaaS (Platform-as-a-Service), and Kubernetes only provided a piece of that functionality.

That realization opened a whole new ecosystem of solutions to investigate. After researching several potential solutions, we tried a number of different options, but nothing seemed to fit our scale or use cases as well as we had hoped.

We first attempted Openshift, which recommended a minimum configuration of far more and larger servers than were cost effective for the type of solution we wanted to create. Additionally our team wasn't looking to navigate the many different versions of Openshift available: Openshift Enterprise, Openshift Origin, Openshift Online, and Openshift 2.

We then looked at Dokku, but realized that only a single host was supported, and we definitely wanted the option of a cluster with multiple, redundant servers.

Each solution had a drawback that prevented it from meeting all of our requirements, until we happened upon Flynn.

It seemed like exactly what we were looking for: an open-source Heroku that we can run on our own infrastructure. Flynn even uses Heroku's buildpacks, which allow us to easily hand off our clients' projects onto infrastructure they can manage themselves.

Flynn was a bit young (pre-1.0 when we began), but it seemed to fit our needs perfectly. We installed it on a small server cluster and began deploying a few of our applications to the new Flynn PaaS. We had a few hiccups along the way, but with some amazing help from the team at Flynn, everything was up and running smoothly. With the release of 1.0 during our integration, its stability improved even further.

Despite Flynn's comprehensive featureset however, Chef was still required to automate certain tasks. Flynn does not support server monitoring or alerting, and there is no way to redirect Flynn's application logs to a service such as Elasticsearch.

However, we are looking forward to a number of features on Flynn's development roadmap, including support for persistent volumes and user accounts. Persistent volumes would allow us to run Elasticsearch within Flynn instead of managing it separately via Chef. User accounts will also be a huge step forward toward making Flynn a more mature and complete solution.

While it may not meet each and every one of our requirements, and certainly hasn't reached feature-parity with Heroku (yet), Flynn has performed very well for us thus far and will be an integral part of our production infrastructure for the foreseeable future.
We encourage you to learn more at https://flynn.io/ or contact us to learn how it could improve your architecutre.