How to Convert Cloud Foundry to Code Engine in 3 Easy Steps

#codeengine #cloudfoundry #ibmcloud

Cloud foundry is a great platforms and I have used it a lot in recent years but I was interested in simplifying the deployment of my multitiered apps. Naturally I was thinking Kubernetes but I settled on the IBM Cloud® Code Engine which is a fully managed, serverless platform built on Kubernetes.

I recently posted about a Savings Bond Wizard conversion app that is deployed using a combination of Cloud Foundry and OpenWhisk. Don't worry if you don't know what a Saving Bond Wizard is. It not important here but its a nice little app to show the three steps I used to move from Cloud Foundry to Code Engine.

Step 1- Create a project
The cloud foundry apps that made up the tiers of my web app were deployed separately. I had them in different resource groups and tagged with different keywords but there was nothing in the deployment that intrinsically marked them as part of a single system. Compared to my cloud foundry deployment, code engine deployments are cleaner. With code engine my tiers are logically grouped as part of a **project** which is basically a Kubernetes namespace. Once I have a project, I can deploy the app tiers as containerized apps.

Step 2 – Create Container app images
In my original code I used the nginx buildpack to deploy the Vue.js front-end. That buildpack is really just creating an image for me and I can still use that strategy with Code Engine and Cloud Native Buildpacks but I wanted to take more control over this process so I created a `Dockerfile` to build and run my front-end. The advantage for me is that both parts of my app now use `Dockerfiles` so there is an internal consistency. There is a not a build pack that covers the one-off setup I need for the C++ and node setup I use for the backend.

The Red Hat registry has great set of curated and secured base images so for the build phase of the front-end I chose the Node.js 14 image which runs "`yarn build`" to generate the deployment files. To run the front-end I chose the Nginx 1.20 image which matches my original cloud foundry choice. I also refactored the reverse proxy config I used to with cloud foundry so that I could use it with this Nginx image.

For my services tier, I chose the CentOS base image from docker.io for the build phase which made it easy to add the compiler and build tools I need to create the "`sbw2csv`" tool that the services tier uses. To run the services tier I chose the Node.js 14 Minimal image from Red Hat and set the entry point to "`yarn start`".

After creating both of these images I pushed them to the IBM registry.

Step 3 – Deploy applications
If you are using the IBM container registry, make sure your project has access. In the project click on the "Registry access" tab and make sure the server name and credentials are set for accessing the registry. The username for access is always "iamapikey" and the password is your API key. Have a look at Accessing IBM Cloud Container Registry for details. Now you are ready to deploy an application.

How to Convert Cloud Foundry to Code Engine in 3 Easy Steps

In the code engine web site I selected my project, clicked “Applications” and “Create”. From there I chose “services” as the name and set the image reference to the services image I pushed in step 2. I took the defaults for all the other settings. My services layer also needs access to a Cloudant db and a Cloud Object Storage bucket. I set those up on the command line using “`ibmcloud ce app bind`”. See the README in git for those details.

Next I went back to “Applications” and clicked “Create” again. I chose a name for my application and set the image reference to the UI image I pushed in step 2. I also expanded the “Environment variables (optional)” section and set a value for “COS_DOWNLOAD” URL which is the same value I used in the cloud foundry version of the app.

Done
That was it. Much easier than I anticipated. The cloud foundry site has a comparison between itself and Kubernetes where it says “By way of analogy, one can imagine that Cloud Foundry is a doll’s house, and Kubernetes is a box of building blocks from which you can create a doll’s house.” Using that same analogy, Code Engine is a doll’s house built on Kubernetes.

All the code is on GitHub here.
You can see the running app here.