DEPLOYING KUBERNETES CLUSTERS USING DEVOPS PRINCIPLES

Summary

This engagement took place a couple of years ago, when Kubernetes was just beginning to take off. The customer was engaged in a brand-new venture outside its normal realm of activity. They had partnered with a startup and were engaging internal company technical resources around the world.

Given the “start-up-y” nature of the project, the technical tools chosen were mostly open source and very bleeding edge. Moreover, they wanted to make use of DevOps practices including infrastructure as code and CI/CD. However, the team FPCO worked with came from the corporate side of this huge enterprise, and so had little internal expertise regarding either the specific technologies involved or DevOps practices and principles.

FP Complete was hired to work with the team charged with the deployment of Kubernetes cluster. The team was consistently behind schedule and struggling to implement DevOps practices. We were engaged to help get them back on track, as well as train them in best practices.

Corporation Type:	Fortune 100 Hardware and Software Technology Company
Industry/Sector:	Consumer Products
Project Type:	Consumer Product with associated E-commerce Platform
Technology Used:	Kubespray, Ansible, Terraform, Kubernetes, AWS

Project Requirements

The technical goal of the team we worked with was to create versioned, blue-green deployments of Kubernetes clusters via CI/CD pipelines. There were four parts involved in the creation of the clusters:

1. The cloud infrastructure
2. The Kubernetes cluster with basic Kubernetes functionality
3. Kubernetes add-ons such as service mesh (Istio) and monitoring (Prometheus)
4. The microservices created by the development team to be deployed in the various clusters.

There were 3 classes of cluster environments to be created: Development, Testing and Production.

One further requirement was that to the extent possible, the code was to be cloud independent, since the goal was to become multi-cloud in the not too distant future. For this reason, Kubespray was chosen as the Kubernetes creation tool, because it made the promise of providing one tool for deploying infrastructure on multiple infrastructure environments.

This was a classic opportunity for putting into place DevOps practices. The idea was to create code which created the cloud infrastructure, deploy Kubernetes, and then the add-ons. CI pipelines could be used to test the reliability of the cluster code. The actual deployment of the clusters was to be done via CD pipelines. The CD would be done in turn for each of the three environments in a staggered fashion, and so have the clusters ready for deploying the microservices (which was handled by another team).

Project Challenges

There were two key challenges the customer team faced:

1. All the major technologies involved were immature. As noted, Kubernetes was just becoming popular at this time and getting it to work the way you wanted was challenging. AWS did not yet have a serviced Kubernetes offering that would have made deployment there a bit easier. Kubespray was more promise than reality. The service mesh tool (Istio) wasn't even at version 1 and was quite buggy.
2. The customer team in charge was talented and motivated but did not have a great deal of experience in building infrastructure as code.

Because of challenge 1, the customer team spent most of its time dealing with integration issues just to get stuff to work together properly. They were involved in putting out fires. They didn't feel they had the bandwidth to get to step two which would have helped address those very integration issues.

The Solution

While FP Complete's original mandate was to train the customer team in DevOps best practices, it soon became clear that given the time constraints of the customer team and our engagement, we could be of much greater help if we actually created the first version of the infrastructure code. Our key focus was twofold:

1. to ensure that any given moment there would be an integrated tested and working version of a cluster.
2. To ensure the code was modular and re-usable so the same code base could be easily deployed with different versions of the components and in different environments.

With this focus we were able to accomplish the following:

1. Making use of our own FP Complete Terraform modules, we were able to create replicable and testable AWS infrastructure

2. We restructured the Ansible code in Kubespray and modularized it, so that it could more easily handle the constant changes in versions of Kubernetes and its various add-ons
3. We created integration tests so that the components could be tested together before deployment, to ensure there was always a baseline, working version of the cluster
4. We created an over-all modularization of the code so that it would be easy to re-use the same code base to deploy to the three different environments

Having this code in place took away the urgency of integration issues. The customer could guarantee they always had a working version of the cluster that could be deployed and tested incrementally to the three environments. Integration issues of newer versions could be handled "off-line" and not become a barrier to meeting schedules.

New Challenges / Lessons Learned for FP Complete

This project exposed FP Complete to the complexities of working with Kubernetes and how poor the existing tooling is in getting Kubernetes up and running in a reliable, replicable fashion. In particular it helped us understand that Kubernetes has multiple layers of challenges:

1. Getting the proper infrastructure in place in different environments (cloud and on-premise)
2. Deploying a working version of Kubernetes itself
3. Integrating all the necessary add-ons to make Kubernetes reuseable
4. Creating an on-ramp for easily deploying applications onto the Kubernetes cluster

Over time, as more of our customers began adopting Kubernetes a pattern emerged: our customers are eager to find a packaged solution that address these challenges which all of them face.

Conclusion

As is always the case, FP Complete expertise helped this customer get back on track. With the infrastructure code in place, the customer team used it as the basis of continued development. It eased the pressure of dealing with Kubernetes integration problems. They could more easily meet project deadlines and provide other teams with the necessary Kubernetes platform.

Recently, FP Complete has developed our own product, Kube360 which addresses all four of the challenges noted above. Kube360 allows our customers to easily and quickly get up and running a secure, replicable, well-integrated and full-featured application deployment environment, in multiple clouds and on premise. Kube360 significantly helps the many customers similar to this one who are adopting Kubernetes. It allows them to focus on building their applications, and not have to worry about the enormous complexities around deployment infrastructure.