How Kubernetes Saved OpenStack

When virtualization was big and cloud was born, the idea of open source private cloud seemed powerful.

VMware had shown that organizations wanted more efficient ways to utilize hardware. Both VMware and AWS enabled self-service and rapid provisioning of virtual machines. But both could be uncomfortably expensive at scale, so the thought was to create something that could be used on-prem to provide a self-service environment for users who wanted to create their own VMs.

A properly running OpenStack cloud could provide resources solidly enough for entire telecommunications systems to run on it; in fact, an entire branch of networking, Network Functions Virtualization (NFV), evolved to replace single-purpose hardware such as firewalls and load balancers with virtual machines, managed by OpenStack more often than not.

But while the idea of open source private cloud remains powerful, there came a point where containers pointed the way to various other kinds of efficiency, and Kubernetes emerged and drew focus away from virtualization.

Containers had a lot of advantages over virtual machines, after all, such as better portability and smaller sizes. Somewhere in the back of many developers' heads there grew a notion that everything should be containerized, whether or not it made sense for the particular application. People began to talk about containerizing legacy applications, and even considered turning Virtual Network Functions (VNFs) (the workloads of NFV) into Containerized Network Functions (CNFs). Developers were twisting themselves (and their code) into knots just to containerize their workloads

Part of the reason containers and Kubernetes gained such traction is that OpenStack, as powerful as it is, was in a sense a weaker vessel, in that it was hard to deploy, operate, and maintain. Standing up an OpenStack cloud be a multi-week (or month) job for multiple staff members. And upgrades could be so fraught with danger (or at least perceived danger) that production installations often didn't upgrade at all, staying 2, 6, even 10 versions behind rather than take the chance to get the newer functions available in OpenStack.

But even seven years after its initial release, nothing in Kubernetes-space has stepped in to replace OpenStack. While various efforts to run virtual machine-like things on Kubernetes have been fielded, there's no satisfactory generally-accepted standards for how this should work.

Still, the need for Infrastucture as a Service (IaaS) remains and grows. Kubernetes' efficiencies, alone, aren't enough to solve for the problem of high public cloud costs, need for utilization efficiency, and so on. For that matter, Kubernetes itself needs somewhere to run, and VMs are most often the answer,.

And yet there was still this thought that containers were better than VMs, and there was even an effort to containerize OpenStack, with the thought that this might solve the installation problem.

But then someone realized that Kubernetes — designed for running big, complex applications — is itself a good host for OpenStack.

OpenStack, after all, consists of a large number of independent services. With Kubernetes, it's possible to architect OpenStack as a microservices application that can be easily deployed, operated, and managed. If a service goes down, Kubernetes instantly restarts it. Kubernetes can manage multiple service instances, leading to a robust, highly available architecture.

But perhaps the most important thing Kubernetes has done to save OpenStack is to solve that one seemingly intractable problem: upgrades. Running OpenStack as a collection of services on Kubernetes enables you to easily upgrade between versions, because Kubernetes itself manages the process. After all, it's specifically designed to replace one container (or Pod, in this case) with another in case of failure, so Kubernetes can use that capability to upgrade to a new version of that service without breaking the entire architecture — and while leaving workloads largely intact.

So instead of replacing OpenStack, Kubernetes has made it possible for organizations to more confidently continue using it, saving them from having to re-architect legacy applications or force new applications into models for which they are not appropriate, while improving uptime easing operations, and making frequent upgrades not only possible but probable.

Long live OpenStack — on Kubernetes!

Learn more at Mirantis.com