SHIFTING TO CONTINUAL UPDATES: REALIZING CI/CD IN OPENSTACK
INTRODUCTION

Application design has changed rapidly over the last few years. With the advent of "no ops," containers bringing infrastructure into code and CI/CD and DevOps both reaching widespread adoption, applications are experiencing a new set of problems. In particular, the line between infrastructure and application is now so thin it’s practically invisible. This is particularly true for containerized applications.

Containerized applications necessarily carry a great deal of knowledge about their operating system, their location on particular host machines, who their neighbor containers are and where they can reach data. It wasn’t long ago that such information was explicitly abstracted away from the application, and now it is considered part of how the application delivers services.

It’s also true of bare metal applications. We routinely see examples of highly mobile application delivery platforms, like Docker, sitting next to giant stoic Hadoop and Cassandra stacks, all equally sharing in the burden of user experience.

At first glance, the marriage of Hadoop stacks and containers may seem like a bit of a contradiction from a cloud computing perspective, but they both share a high awareness of and reliance on the underlying hardware. For example, Docker is intrinsically reliant on the host operating system’s kernel and host networking, and Hadoop arranges data to spread over racks of cabinets. And neither is very friendly with the abstraction of virtualization.

Because the separation of infrastructure and application is thinning, software delivery teams are seeing new problems arise where outages occur because of a change to an underlying host system that did not occur in testing. It is also becoming more apparent that the ability to re-provision hardware into a reliable state is a serious contender for slowing development processes. In truth, you will find any environment that combines platforms of varying agilities to be in need of API-driven hardware deployment.

Recently, we had a customer engagement with one of the world’s largest banks. Long-standing customers of another public cloud, their ongoing costs and flexibility limitations were breaking their ability to innovate. Looking for an opportunity to partner with one of the leading OpenStack® experts, they came to us to drive a complete shift toward DevOps and CI/CD by leveraging OpenStack’s capability to handle both bare-metal and compute nodes seamlessly. This whitepaper details how we helped them reach that goal.

IRONIC, EPHYRA AND CI/CD

In OpenStack, Ironic provides bare metal provisioning. At its heart, Ironic PXE-boots a small in-memory image that then writes an operating system to disk. The real benefits of Ironic are apparent when you run it alongside a general compute cloud as it allows you to use the same images for both bare metal and compute instances. Thus, you are able to ensure that an image is the same regardless of what it is deployed onto. Additionally, up-and-coming versions of Ironic will allow for Neutron overlay networks to be extended to bare metal nodes, meaning you can share tenant networks between cloud instances and bare metal servers. That allows for some really creative designs, where applications can run on the platform that best serves their needs.

But simply having the same images is often not sufficient. After all, running on a blade server is vastly different than running on KVM. To do true CI/CD in bare metal environments, you need a way to carry the artifacts after testing into production. And that means you need to stamp the APIs themselves as artifacts. To accomplish this, we implemented A/B Control Planes using the following steps:

1. Set up the initial OpenStack private cloud with two control planes.
2. Build a replacement, upgraded control plane.
3. Sphon off compute nodes and Ironic nodes one at a time into the new control plane.
4. Continue testing as we move nodes to the upgraded control plane.
5. Move the rest over, gradually, until all workloads are on the new control plane.
6. Retire the old control plane.

There is a bit of lock stepping involved, as applications are getting upgraded at the same time. In other words, when an application is triggered for redeployment with a new version, it is also deployed on a new version of the hardware platform itself is part of the software delivery model, once it is built it is considered immutable. Rather than patch or change something, you would write code to your deployment framework and run it through the CI/CD system. That way there is versionability to your deployments. If new problems are introduced that can be tracked to a new kernel version, you can see explicitly when it happened and what version to rollback to.

And in fact, the way you think about operations changes. Operations is not a “team,” it’s a set of actions/interventions that must be taken to ensure product outcomes. Therefore, we codify any operations activities outside of emergencies. The real art form to that sort of a rollout is to understand when a server can be decommissioned and rebuilt on the other side. We do this with a new OpenStack project called Ephyra that is currently in incubation. Ephyra syncs Ironic environments between two control planes. After syncing images and SSH keys, Ephyra moves Ironic nodes from the old control plane to the new one and registers them there. The end result is a slow trickle of nodes from one side to the other as they are taken down and re-provisioned. This allows OpenStack to effectively be upgraded without any downtime. You just need to keep enough spare capacity to seed the new control plane. As for integration into your application CI/CD, an event triggered at the end of a build from Jenkins can signal which nodes are ready to be reclaimed as they are now empty.

WHY THIS MATTERS

In following with a CI/CD model, the way you interact with infrastructure also changes. Since the hardware platform itself is part of the software delivery model, once it is built it is considered immutable. Rather than patch or change something, you would write code to your deployment framework and run it through the CI/CD system. That way there is versionability to your deployments. If new problems are introduced that can be tracked to a new kernel version, you can see explicitly when it happened and what version to rollback to.

And in fact, the way you think about operations changes. Operations is not a “team,” it’s a set of actions/interventions that must be taken to ensure product outcomes. Therefore, we codify any operations activities outside of emergencies. The way you would change a software dependency—write code to do what you want, write tests for that code, send it through the CI/CD system to be adopted by the platform.

CONCLUSION

Rackspace exists to deliver the best OpenStack experience in the world, with industry-leading reliability, unmatched scalability, innovation-driven agility and a superior approach. As part of that commitment, helping customers deliver business agility through CI/CD is a top priority. This customer worked with us to build an on-premise cloud with over 2300 nodes. Our OpenStack insight
and ability helped them leverage Ironic and Ephyra to migrate from their old public cloud to their new OpenStack private cloud, almost immediately running production workloads on OpenStack to great success.

As a result of the migration and the shift to CI/CD, they’ve been able to move from upgrading applications twice a year to upgrading them every two weeks. Their agility has skyrocketed, and along with that, a significant reduction in costs. But perhaps most surprisingly, we’ve been able to demonstrate an effective way to upgrade OpenStack on the fly as improvements emerge and new opportunities demand enhancements.

If you’re trying to drive the same shift, come talk to us. Sign up for our free OpenStack strategy session at http://go.rackspace.com/OpenStackExperts.
ABOUT RACKSPACE

Rackspace (NYSE: RAX), the #1 managed cloud company, helps businesses tap the power of cloud computing without the complexity and cost of managing it on their own. Rackspace engineers deliver specialized expertise, easy-to-use tools, and Fanatical Support® for leading technologies developed by AWS, Google, Microsoft, OpenStack, VMware and others. The company serves customers in 120 countries, including more than half of the FORTUNE 100.

Rackspace was named a leader in the 2015 Gartner Magic Quadrant for Cloud-Enabled Managed Hosting, and has been honored by Fortune, Forbes, and others as one of the best companies to work for.

Learn more at www.rackspace.com or call us at 1-800-961-2888.

© 2016 Rackspace US, Inc.