How do containers scale on OpenStack?

TON NGO / @TANGO245 / NOVEMBER 16, 2016 / 0 COMMENTS

Co-authored with Winnie Tsang (IBM), Ricardo Rocha (CERN), Spyros Strigazis (CERN)

The OpenStack community has been developing support for containers for over two years since the Paris Summit. Containers are now alive and well on OpenStack: not only can users easily deploy containers but OpenStack services themselves have been well containerized, as evident in projects such as Kolla and OpenStack Ansible. The next natural question is, how do containers scale? In other words, what happens when ten of thousands containers are deployed in the cloud? And how do they perform?

We set out to answer these questions in a collaboration between two teams from IBM and CERN, with assistance from a team from Rackspace who also participated in the CERN Openlab collaboration. Of course to study scalability, we need access to large OpenStack environments. Fortunately, the CERN lab was able to run some tests on part of their large OpenStack cloud, and the CNCF lab provided a 100 nodes cluster for the study. The CERN cloud ran the Mitaka release while the CNCF cloud ran the Newton release.

We have presented the results we have found so far at the last OpenStack Summit in Barcelona; you can view the video from this link:

https://www.openstack.org/videos/video/toward-10000-containers-on-openstack

In this blog, we describe the key findings from that presentation.

We consider 3 perspectives:

First, the infrastructure in OpenStack to host the containers must be created. This is the scope of Magnum, the OpenStack service that manages container infrastructure by creating all the necessary resources such as VM/baremetal nodes, storage, networking, etc. The type of

container infrastructure includes clusters for Kubernetes, Swarm, or Mesos. The question here is, how well does OpenStack deploy these container infrastructures at scale?

Second, once the infrastructure is up and running, the containers are deployed. This is the scope of the particular container platform, i.e. Kubernetes, Swarm, or Mesos. The question at this level is, how well do these platforms deploy containers on OpenStack on a large scale?

Third, once the containers are running, the apps running in the containers will handle the workload that they are designed for. The scope at this level is specific to the app. The question is, how does the app running in containers on OpenStack perform at scale?

For our benchmarks, we use Rally for the first two scenarios, using a plugin designed for driving Magnum, Kubernetes, Swarm, and Mesos. For the third scenario where an app is measured, we use a benchmark from Google: many nginx containers serve web requests generated by a load driver. Google has published performance characteristics from the benchmark, allowing us to compare with result from OpenStack.

In the first scenario, we attempted to deploy various combination of many small clusters or a few large clusters. Magnum handles the requests by launching Heat templates which in turn create and configure all the OpenStack resources. We found that Magnum does not generate a heavy load and so it is itself not a bottleneck. However, Heat does have to handle a large number of nested templates and does exhibit signs of problems at the point of around 200 medium sized clusters. This limitation was a focus topic at a design session with the Heat team at the Barcelona Summit. We also found RabbitMQ to be a bottleneck, and this is consistent with observations from others. In fact, there were multiple presentations at the Summit on experiences with Rabbit as well as techniques to alleviate the bottleneck. Overall, it became apparent that scalability for container infrastructure is really a general OpenStack scalability issue. If your OpenStack cloud has been tuned to scale well, then there should be no problem with scaling the infrastructure for containers.

In the second scenario, we measured how the 3 container platforms deploy containers. This work is still ongoing and we don’t have the complete data yet, but some interesting observation did emerge. The time to deploy containers is similar on Kubernetes and Swarm, but is significantly higher on Mesos. The reason is that Mesos only manages the resources, therefore a framework such as Marathon is needed to manage the actual containers. In the current Mesos implementation, a container request is sent to Marathon, then Marathon needs to go through a negotiation protocol with Mesos to obtain a resource offer and to start the Docker container. This overhead suggests that Mesos is appropriate as a container platform when resources need to be shared among multiple workloads. Running Marathon alone on Mesos to deploy Docker containers would not be efficient. In a previous talk, we have shown how to run both Kubernetes and Swarm containers on the same Mesos cluster:

https://www.openstack.org/videos/video/mesos-and-openstack-the-perfect-tag-team-for-containers

In the third scenario, we created a Kubernetes cluster with 800 vCPU, as indicated by the Google benchmark. As the replication controller scaled up the pods running nginx, we found that the cluster easily handled 1 million requests per second from the load driver. With larger clusters, we have observed up to 3 million requests per second. This result matches the data published by Google and suggests that once the infrastructure and the containers are in place, the container app performs as well on OpenStack as on other platforms.

The result so far is encouraging, suggesting that there is no fundamental issue with scaling containers on OpenStack. But our scalability study is continuing. We are testing container app benchmark for Swarm and Mesos, similar to the Google benchmark. We are experimenting with techniques to tune Rabbit and Heat find the best practices for scalability. In future tests, we want to measure performing key lifecycle operations on the cluster while the cluster is handling a significant load. This includes operations such as scaling the cluster, upgrading the cluster, and upgrading a running service by updating its image.

We will share more result in future presentation and blogs.

TAGGED: CERN / CNCF / CONTAINERS / KUBERNETES / MAGNUM / MESOS / OPENSTACK / SCALABILITY / SWARM

Leave a comment

Name * Email *

Comment

Submit Comment

- Notify me of follow-up comments by email.
- Notify me of new posts by email.