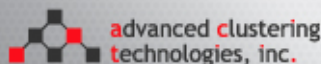


[ABOUT](#)[CONTRIBUTORS](#)[CONTACT](#)[THE REGISTER](#)[THE CHANNEL](#)**TURN-KEY HPC SOLUTIONS****REQUEST A QUOTE**[HOME](#)[COMPUTE](#)[STORE](#)[CONNECT](#)[CONTROL](#)[CODE](#)[ANALYZE](#)[HPC](#)[ENTERPRISE](#)[HYPERSCALE](#)[CLOUD](#)

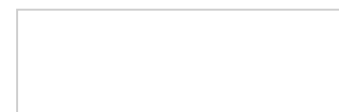
A DECADE OF CONTAINER CONTROL AT GOOGLE

March 22, 2016 Timothy Prickett Morgan



Search engine giant Google did not invent software containers for operating systems, but it has helped perfect them for the Linux operating system and brought the orchestration of billions of containers across perhaps millions of servers to something of an art form. It has much to teach, and is still humble enough to know that it has much to learn.

That, perhaps more than anything else, was one of the reasons why Google decided to open source a container management system called Kubernetes nearly two years ago. In the past, Google has been content to deliver papers on how some of its more advanced technology works, showing the way for the rest of the industry when it comes to tackling large-scale analytics, storage, or compute problems. But this method backfires sometimes, as it has with the MapReduce method of computation that inspired rival Yahoo to create Hadoop and spawned an entire industry. Now, it is Google that is incompatible with the rest of the world rather than the world following a standard created by Google and bequeathed to the industry.

**ISC HIGH
PERFORMANCE****JUNE 19 -23, 2016
FRANKFURT
GERMANY****Join Us!****THE NEXT PLATFORM WEEKLY**

So Kubernetes, we think, is an attempt by Google to share techniques and to foster the development of a container control system that could, in theory, maybe someday far off in the future, replace the internal Borg system that has been at the heart of its clusters for many years and that, as its name suggests, has been pulling in different technologies (like the Omega scheduler created several years ago). This time around, though, Google is seeking the help of the industry and is getting fellow hyperscalers as well as HPC experts to help flesh out Kubernetes, which has to span more than just the kind of relatively homogeneous workloads that run – albeit at large scale on clusters with 5,000 to as many as 50,000 nodes – inside of Google.

The key developers who created and extended Borg inside of Google are the same people who have created Kubernetes and who continue to be very influential in this open source project. This includes Brendan Burns, a co-founder of the Kubernetes effort; John Wilkes, principle software engineer at Google who has been working on cluster management systems there for the past eight years; Brian Grant, the technical lead of the Borg container management system, the founder of the Omega add-on to Borg to make it more flexible, and the design lead for Kubernetes; David Oppenheimer, technical lead for Kubernetes; and Eric Brewer, a vice president of infrastructure at Google and a professor at the University of California at Berkeley.

Google has published two technical papers describing its container management system, and it did it backwards. **The first one described Omega in October 2013**, and **Wilkes gave a fascinating presentation** at the Large Installation System Administration conference in November 2013 after the publication of the Omega paper. This shed some light on Google's Borg system, and this disclosure predates the launch of Kubernetes in the summer of 2014. In the wake of Kubernetes, and a certain amount of confusion about how Omega related to Borg, a second paper describing its **view of large scale cluster management was released in April 2015**, and **we talked to Wilkes after that paper was released** to get some more insight into Google's thinking about containers and cluster scheduling.

Now, the top Google techies who work on Borg, Omega, and Kubernetes have published **a less technical paper in ACM Queue** that describes some of the lessons learned from a decade of container management, providing a little more insight into its thinking and perhaps some indications of where the company would like to see the Kubernetes community push ahead.

Borg, as it turns out, was not the first cluster and application management tool that the search engine giant created. Google, like all companies, has a mix of batch workloads and long-running service workloads, and it partitioned its clusters physically to support these very different kinds of jobs. A program called Babysitter ran the long-running service jobs (like Web and other infrastructure servers) while another called the Global Work Queue ran the batch jobs like the MapReduce big data analytics framework that inspired Hadoop and that is still used today inside Google. But running these applications on different clusters meant that compute capacity was often stranded, and that was unacceptable when Google was growing explosively a decade ago.



Tap the stack to painlessly subscribe for a weekly email edition of The Next Platform, featuring highlights, analysis, and stories from the week directly from us to your inbox with nothing in between.

SIMILAR VEIN



What Makes Containers At Scale So Difficult



Kubernetes Has A Ways To Go To Scale Like Google, Mesos



Enter software containers, which are by no stretch of the imagination a new concept and were not even a decade ago. One could argue that the PR/SM logical partitions created by clone mainframe maker Amdahl in 1989 were containers, and certainly IBM's own VM operating system created virtual machines that were, in essence, software containers and still, to this day, underpin the Linux instances on System z mainframes. FreeBSD had jail partitions and Solaris eventually got its own containers, but Google was a Linux shop and it therefore had to do a lot of the grunt work in adding container features to the Linux kernel. The LXC containers that are now part of every Linux distribution were founded on Google's work, and Docker is an offshoot of this effort.

In a way, Kubernetes is a mix of the approaches taken with the monolithic Borg controller and the more flexible Omega controller (which allowed for multiple workload schedulers to fight for resources rather than just wait in line), but in the end, it is really Google's third container controller. And in the end, it might be Kubernetes that ends up being Borg if its level of sophistication and scale continues to rise. This, we think, is one of Google's goals. The other, as we have elucidated before, is for Kubernetes to become a de facto container management standard and therefore making it more likely that enterprises building private clouds will be enticed to use the Google Cloud Platform public cloud and its container service, which is based on Kubernetes.

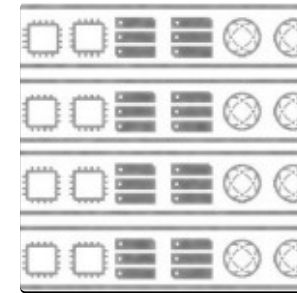
The big thing that can be gleaned from the latest paper out of Google on its container controllers is that the shift from bare metal to containers is a profound one – something that may not be obvious to everyone seeking containers as a better way – and we think cheaper way – of doing server virtualization and driving up server utilization higher. Everything becomes application-centric rather than machine-centric, which is the nirvana that IT shops have been searching for. The workload schedulers, cluster managers, and container controllers work together to get the right capacity to the application when it needs it, whether it is a latency-sensitive job or a batch job that has some slack in it, and all that the site recovery engineers and developers care about is how the application is performing and they can easily see that because all of the APIs and metrics coming out of them collect data at the application level, not on a per-machine basis.

To do this means adopting containers, period. There is no bare metal at Google, and let that be a lesson to HPC shops or other hyperscalers or cloud builders that think they need to run in bare metal mode. We have heard chatter from people who would know that many of the analytics and data storage services that Amazon Web Services sells as platform services are on bare metal machines, not on its EC2 compute instances with the custom Xen hypervisor underneath, but after reading that, AWS has come back to us and said this is not true. We do not know if AWS is using containers underneath any of its infrastructure, but it clearly has a container service atop it.

The point is, bare metal is not a foregone conclusion, and the tradeoff of a slight performance hit – a whole lot less than full-on server virtualization using a KVM or Xen hypervisor – versus the flexibility of management and deployment that comes through containers is an absolute fair trade. (If Intel wanted to settle the issue, it could add a container management core to each chip and bet



Containers For The
Masses Now That
Kubernetes Is Set Free



HP Follows Hyperscale
Lead With Composable
Infrastructure



Scale And Database
Pioneer Talks Hadoop
Evolution



Mesos Brings The Google
Way To The Global 2000

done with it, and it could have added two or three specialized cores to Xeon server chips to run hypervisors, too.)

“The isolation and dependency minimization provided by containers have proved quite effective at Google, and the container has become the sole runnable entity supported by the Google infrastructure,” the authors write. “One consequence is that Google has only a small number of OS versions deployed across its entire fleet of machines at any one time, and it needs only a small staff of people to maintain them and push out new versions.”

With Kubernetes, a container is a single runtime with an image of application software to execute on it, and multiple containers that represent the microservices that comprise what we would think of as an application are aggregated into a higher-level construct called a pod. Kubernetes is, first and foremost, a pod management system, keeping the container collectives aligned and running. Borg has a similar architecture, with a Linux container being the lowest level of granularity and an allocation, or *alloc* for short, being the higher level wrapper for a collection of containers. Borg allows some containers to run outside of allocs, and the authors say “this has been a source of much inconvenience,” probably a very dry understatement indeed. Kubernetes does not let you do this. You can, however, run containers inside of other containers or inside of full-on virtual machines if you want more security and isolation between containers, say in a multi-tenant environment like a public cloud. In fact, this is precisely what Google does with Cloud Platform. A server has a giant container laid down on it, then a KVM hypervisor, and then individual containers are created that expose specific Compute Engine instance types to those buying capacity on that public cloud.

Just like Omega imposed some order on the gathering and use of state information from around Google’s clusters to do a better job of allocating resources to containerized applications running in batch and online mode, Kubernetes is imposing a strict consistency in the APIs that are used to interface with it – unlike Borg, which had a hodge-podge of API types and styles added over the years.

“The design of Kubernetes as a combination of microservices and small control loops is an example of control through choreography – achieving a desired emergent behavior by combining the effects of separate, autonomous entities that collaborate,” Google says in the latest paper. “This is a conscious design choice in contrast to a centralized orchestration system, which may be easier to construct at first but tends to become brittle and rigid over time, especially in the presence of unanticipated errors or state changes.”

The impression we get is that Kubernetes is simply a better, if somewhat less mature, tool than Borg, and that alone should give the entire IT industry pause.

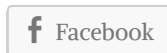
Google is giving the benefit of more than a decade of its experience to the open source community,

Sponsored links:

[ISC High Performance early-bird registration ends May 11](#)

[Download the whitepaper: Best practises for getting started with software-defined storage](#)

even after it did nudge the AMPLab at Berkeley early on to foster the development of the Mesos system now being commercialized by Mesosphere. This is probably something that it now regrets doing, but we presume that it was not an easy task for Google to create and release Kubernetes. The only other option was to let Mesos become another *de facto* standard like Hadoop, and we have seen how that movie plays out. Not in Google's favor. Google is the only hyperscaler that cannot easily shift from homegrown MapReduce to Hadoop, and over time, that could mean its costs for infrastructure software development go up over time.

SHARE THIS:**SIMILAR VEIN**

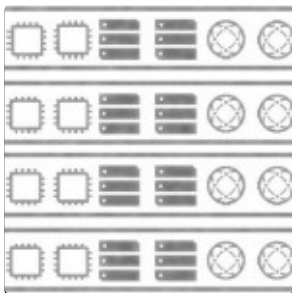
What Makes Containers
At Scale So Difficult



Kubernetes Has A Ways
To Go To Scale Like
Google, Mesos



Containers For The
Masses Now That
Kubernetes Is Set Free



HP Follows Hyperscale
Lead With Composable
Infrastructure



Scale And Database
Pioneer Talks Hadoop
Evolution



Mesos Brings The Google
Way To The Global 2000

Categories: [Control](#), [Hyperscale](#)

Tags: [Borg](#), [Google](#), [Kubernetes](#), [Omega](#)

From Systems to Services: IBM
Floats Cloud Concept for 2016

The Road To 200G Networks
Starts With The Transceiver

6 THOUGHTS ON “A Decade Of Container Control At Google”



Stefanka says:

March 22, 2016 at 1:26 pm

Some of the links to the papers are broken.

[Reply](#)



Timothy Prickett Morgan says:

March 22, 2016 at 1:59 pm

All fixed. Not sure why the static links didn't work.

[Reply](#)



Geek says:

March 23, 2016 at 9:25 am

Interesting thoughts. I'm quite ignorant about the topic, but there should be a forum with all the big players, where a “container standard” is defined and we ensure interoperability over time. Does that exist?

[Reply](#)



Craig Box says:

March 24, 2016 at 10:08 pm

Like the Cloud Native Computing Foundation? <http://cncf.io/>

[Reply](#)

**Greg Gregson says:**

March 27, 2016 at 6:05 pm

“Google is the only hyperscaler that cannot easily shift from homegrown MapReduce to Hadoop, and over time, that could mean its costs for infrastructure software development go up over time.”

Having used both MapReduce/BigTable internally at Google, and Hadoop/HBase externally, I can assure you that there are literally no circumstances under which Google would wish to switch to Hadoop. For one thing, the performance is atrocious in comparison.

[Reply](#)**Thomas Hoberg says:**

April 15, 2016 at 12:34 pm

I tend to think that the best products do not need marketing, but the history of OpenVZ perhaps shows that this is not quite the case.

Virtuozzo has been sold as a product since 2002, it open sourced as OpenVZ in 2005 and has been running Linux containers for a long time all over the world, certainly far more than LXC ever did.

Been using it to run PCI-DSS compliant workloads since 2008 and very much like how well it handles nesting Docker containers (better deployment) inside OpenVZ ones (better operational control).

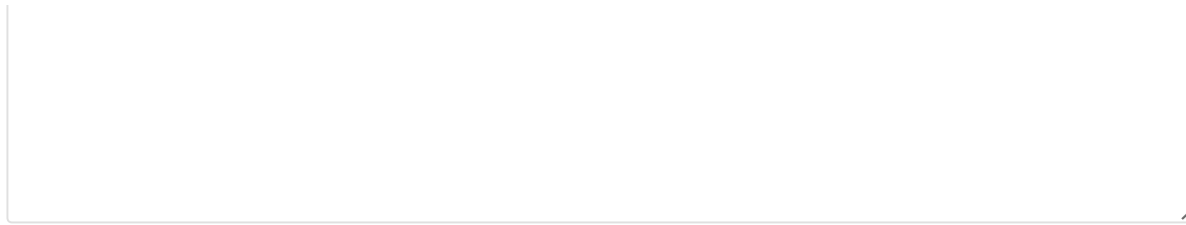
I understand that this article is more about Google propaganda, but since Docker and LXC are mentioned an honorable mention to this fantastic pioneer in Linux containers would have been in order.

[Reply](#)

LEAVE A REPLY

Your email address will not be published. Required fields are marked *

Comment



Name *

Email *

Website

PAGES

[About](#)
[Contact](#)
[Contributors](#)

RECENT POSTS

[EMC Shoots For Explosive
Performance With Isilon Nitro](#)
[Cloud Foundry Is Crossing The](#)



Newsletter

Chasm

The Long Future Ahead For Intel

Xeon Processors

Mashing Up OpenStack With

Hyperconverged Storage

Storage Performance Models

Highlight Burst Buffers at Scale

Copyright © 2015 The Next Platform