What's the difference between Apache's Mesos and Google's Kubernetes

What exactly is the difference between Apache's Mesos and Google's Kubernetes? I understand both are server cluster management software. Can anyone elaborate where the main differences are - when would which framework be preferred?

Why would you want to use Kubernetes on top of Mesosphere?

cloud cluster-computing mesosphere kubernetes

edited Nov 7 '14 at 19:10

asked Nov 2 '14 at 22:28

binaryanomaly

1,220 3 8 19

5 Answers

Kubernetes is an open source project that brings 'Google style' cluster management capabilities to the world of virtual machines, or 'on the metal' scenarios. It works very well with modern operating system environments (like CoreOS or Red Hat Atomic) that offer up lightweight computing 'nodes' that are managed for you. It is written in Golang and is lightweight, modular, portable and extensible. We (the Kubernetes team) are working with a number of different technology companies (including Mesosphere who curate the Mesos open source project) to establish Kubernetes as the standard way to interact with computing clusters. The idea is to reproduce the patterns that we see people needing to build cluster applications based on our experience at Google. Some of these concepts include:

- *pods* — a way to group containers together
- *replication controllers* — a way to handle the lifecycle of containers
- *labels* — a way to find and query containers, and
- *services* — a set of containers performing a common function.

So with Kubernetes alone you will have something that is simple, easy to get up-and-running, portable and extensible that adds 'cluster' as a noun to the things that you manage in the lightest weight manner possible. Run an application on a cluster, and stop worrying about an individual machine. In this case, cluster is a flexible resource just like a VM. It is a logical computing unit. Turn it up, use it, resize it, turn it down quickly and easily.

With Mesos, there is a fair amount of overlap in terms of the basic vision, but the products are at quite different points in their lifecycle and have different sweet spots. Mesos is a distributed systems kernel that stitches together a lot of different machines into a logical computer. It was born for a world where you own a lot of physical resources to create a big static computing cluster. The great thing about it is that lots of modern scalable data processing application run well on Mesos (Hadoop, Kafka, Spark) and it is nice because you can run them all on the same basic resource pool, along with your new age container packaged apps. It is somewhat more heavy weight than the Kubernetes project, but is getting easier and easier to manage thanks to the work of folks like Mesosphere.

Now what gets really interesting is that Mesos is currently being adapted to add a lot of the Kubernetes concepts and to support the Kubernetes API. So it will be a gateway to getting more capabilities for your Kubernetes app (high availability master, more advanced scheduling semantics, ability to scale to a very large number of nodes) if you need them, and is well suited to run production workloads (Kubernetes is still in an alpha state).

When asked, I tend to say:

1. Kubernetes is a great place to start if you are new to the clustering world; it is the quickest, easiest and lightest way to kick the tires and start experimenting with cluster oriented development. It offers a very high level of portability since it is being supported by a lot of different providers (Microsoft, IBM, Red Hat, CoreOs, MesoSphere, VMWare, etc).
2. If you have existing workloads (Hadoop, Spark, Kafka, etc), Mesos gives you a framework that let's you interleave those workloads with each other, and mix in a some of the new stuff including Kubernetes apps.
3. Mesos gives you an escape valve if you need capabilities that are not yet implemented by the community in the Kubernetes framework.

edited May 1 '15 at 22:09

Alex Robinson
4,276 9 23

answered Nov 6 '14 at 21:02

Craig Mcluckie
3,084 1 5 3

1 Great overview. Two brief thoughts: 1) I believe Kubernetes is now beta rather than alpha? 2) Add information about Marathon? – knite Dec 1 '14 at 20:09

19 To sum up (for the quick read - I hope I get it right): kubernetes is a cluster manager for containers (only?) while mesos is a distributed system kernel that will make your cluster look like one giant computer system to all supported frameworks and apps that are build to be run on mesos. Yet kubernetes is one (amongst others) framework that can be run on mesos. Thus, combineing both you end up with a cluster that is no cluster and a cluster manager that has no cluster to manage. Great new world :-) (J/K there are alot of benefits you get from this since kub. is more then clst. phys. res.) – masi Jan 1 '15 at 14:36

4 Here's Mesosphere talking about exactly this at the Kubernetes 1.0 Launch event: youtube.com/… - disclaimer: this is me. – Air Jul 23 '15 at 21:53

Both projects aim to make it easier to deploy & manage applications inside containers in your datacenter or cloud.

In order to deploy applications on top of Mesos, one can use Marathon or Kubernetes for Mesos.

Marathon is a cluster-wide init and control system for running Linux services in cgroups and Docker containers. Marathon has a number of different canary deploy features and is a very mature project.

Marathon runs on top of Mesos, which is a highly scalable, battle tested and flexible resource manager. Marathon is proven to scale and runs many in many production environments.

The Mesos and Mesosphere technology stack provides a cloud-like environment for running existing Linux workloads, but it also provides a native environment for building new distributed systems.

Mesos is a distributed systems kernel, with a full API for programming directly against the datacenter. It abstracts underlying hardware (e.g. bare metal or VMs) away and just exposes the resources. It contains primitives for writing distributed applications (e.g. Spark was originally a Mesos App, Chronos, etc.) such as Message Passing, Task Execution, etc. Thus, entirely new applications are made possible. Apache Spark is one example for a new (in Mesos jargon called) framework that was built originally for Mesos. This enabled really fast development - the developers of Spark didn't have to worry about networking to distribute tasks amongst nodes as this is a core primitive in Mesos.

To my knowledge, Kubernetes is not used inside Google in production deployments today. For production, Google uses Omega/Borg, which is much more similar to the Mesos/Marathon model. However the great thing about using Mesos as the foundation is that both Kubernetes and Marathon can run on top of it.

More resources about Marathon:

https://mesosphere.github.io/marathon/

Video: https://www.youtube.com/watch?v=hZNGST2vIds

edited Feb 25 '15 at 17:44

answered Feb 25 '15 at 17:39

mesospherian
421 4 4

> Kubernetes and Mesos are a match made in heaven. Kubernetes enables the Pod (group of co-located containers) abstraction, along with Pod labels for service discovery, load-balancing, and replication control. Mesos provides the fine-grained resource allocations for pods across nodes in a cluster, and can make Kubernetes play nicely with other frameworks running on the same cluster resources.

from readme of kubernetes-mesos

answered Jan 21 '15 at 8:01

herodot
989 7 9

Mesos and Kubernetes can both be used to manage a cluster of machines and abstract away the hardware.

Mesos, by design, doesn't provide you with a scheduler (to decide where and when to run processes and what to do if the process fails), you can use something like Marathon or Chronos, or write your own.

Kubernetes will do scheduling for you out of the box, and can be used as a scheduler for Mesos (please correct me if I'm wrong here!) which is where you can use them together. Mesos can have multiple schedulers sharing the same cluster, so in theory you could run kubernetes and chronos together on the same hardware.

Super simplistically: if you want control over how your containers are scheduled, go for Mesos, otherwise Kubernetes rocks.

edited Mar 2 '15 at 10:49

answered Feb 25 '15 at 12:56

user2851943
475 6 15

Mesos *is* a scheduler, so this answer is misleading. – Air Feb 25 '15 at 18:18

This answer is inaccurate and confusing. There's no easy way to run Mesos on Kubernetes - and in fact, that would be an inversion of architecture. Since Kubernetes is less general in focus than Mesos, it makes more sense to run it on top of Mesos. – ssk2 Feb 25 '15 at 19:33

2 @air I'm interested to know how you'd define scheduler here? Mesos itself doesn't appear to provide any of the scheduling logic? This is all handled in Chronos/Marathon/etc ? (perhaps I've missed something! :)) – user2851943 Feb 26 '15 at 11:23

4 I think I see what you're getting at - Mesos is a framework that allows schedulers to be plugged in. I was confused by the wording suggesting that Mesos omitted something important ("Mesos doesn't provide you with"), when that's by design. I removed my downvote. – Air Feb 27 '15 at 22:38

3 This answer is accurate. Mesos focuses on resource management and it decouples scheduling by allowing pluggable frameworks. A good example is what Netflix did by writing a scheduling framework: Fenzo techblog.netflix.com/2015/08/… – Camilo Crespo Feb 3 '16 at 5:37

I like this short video here mesos learning material

with bare metal clusters, you would need to spawn stacks like HDFS, SPARK, MR etc... so if you launch tasks related to these using only bare metal cluster management, there will be a lot cold starting time.

with mesos, you can install these services on top of the bare metals and you can avoid the bring up time of those base services. This is something mesos does well. and can be utilised by kubernetes building on top of it.

answered Feb 28 '16 at 10:50

zinking
2,437 3 26 55
