Container Orchestration: Which Conductor?

ContainerCon Europe, Berlin, Oct 2016

Mike Bright, @mjbright
Haikel Guemar, @hguemar
Mario Loriedo, @mariolet
First ...

A little bit of history
So let's first look at recent container history ...

Unix containers, Unikernels
Linux containers (LXC), Mesos
LXC (CloudFoundry, DotCloud PaaS)
Docker
Micro-OSes, Swarm, Rkt, LXD
Container Orchestration Options
PaaS adoptions of Docker
Docker buys Unikernel Systems
Docker "Swarm mode", OCID...

... 2009 2013 2014 2015 2016 ...

@hguemar @mjbright @mariolet
μ-OSES

Many vendors are developing μ-OSES, small OS (mainly Linux-based) to be the basis for container engine hosts whether they be bare-metal or virtual host machines.

They're small, with fast startup, use few resources and have a small attack surface and often "atomic" software updates.

<table>
<thead>
<tr>
<th>OS</th>
<th>Vendor</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoreOS</td>
<td>(CoreOS)</td>
</tr>
<tr>
<td>Project Atomic</td>
<td>(RedHat)</td>
</tr>
<tr>
<td>RancherOS</td>
<td>(Rancher Labs)</td>
</tr>
<tr>
<td>Photon</td>
<td>(VMWare)</td>
</tr>
<tr>
<td>Nano Server OS</td>
<td>(Microsoft)</td>
</tr>
<tr>
<td>Ubuntu Snappy Core</td>
<td>(Canonical)</td>
</tr>
</tbody>
</table>

...Unikernels
μ-Services
μ-services

From monoliths to μ-services

Remember when *high availability* meant this ...?

Servers running *monolithic applications* in *Active-Standby* modes, as 1+1, N+1, or N+M or split across 3 tiers.

Scaling meant to "*scale up*" by adding CPU, RAM, disk. But there's a limit to this ... then you have to "*scale out*"
μ-services

From monoliths to μ-services

Then came μ-services ..

As the industry moved to virtualized micro-services this allowed to obtain greater efficiencies (higher utilisation of resources) and the redesign of applications allows to scale out and achieve high availability.

Containers facilitate this move, allowing faster scaling and even greater efficiencies with less redundancy (no OS to reproduce).

@hguemar @mjbright @mariolet
How containers help?

Container solutions such as Docker go beyond the isolation capabilities of LXC by providing simple to use tools to enable packaging of apps with their dependencies allowing portable applications between systems.

Containers are lightweight

Versioned images containing all dependancies can be shared

Containers allow to use the same application binaries on development, test and production systems whether that be on a laptop, server or in the cloud.

It's a no brainer for developers, who can build and share their own images

@hguemar @mjbright @mariolet
μ-services From monoliths to μ-services

But 1000's of nodes are unmanageable ... aren't they?

We can't take care of our

so we have to treat them like

that's cloud native!

@hguemar @mjbright @mariolet
So we need container orchestration
What was Container Orchestration again?

- Architecture - Composition & Stitching
- Workflows & Policies to
  - Scale in/out (maybe automatically)
  - Place workloads for
    - load balancing, fault tolerance, resource optimization
  - Adapt to faults

Tasks:
Orchestration

Getting to "Desired State"

To manage 100's, 1000's, 10,000's of nodes we need to express "desired state" rather than "do this".

<table>
<thead>
<tr>
<th>Imperative</th>
<th>Declarative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tell system</td>
<td>Do this</td>
</tr>
<tr>
<td>&quot;start a new node&quot;</td>
<td>&quot;3 mysql nodes&quot;</td>
</tr>
</tbody>
</table>

- **Intelligence**
  - Operator
  - Orchestration Engine

- **Flexibility**
  - Best
  - Least

It is not longer feasible for an operator to

- know the resources available (e.g. SSD/HDD, GPU, ...)
- react to failure, know when to scale ...

@hguemar @mjbright @mariolet
Choice is great - when you know what you want ...
Orchestration

The Big 3 - Main Orchestration Choices

- Docker Swarm ("Swarm Mode")
- Apache Mesos
- Kubernetes
Orchestration

The Big 3 - Main Orchestration Choices

- Docker Swarm ("Swarm Mode")
- Apache Mesos
- Kubernetes

... more Choices ...

- Rancher  (Rancher Labs)
- Fleet     (CoreOS)
- Nomad     (HashiCorp)
- Kontena
- OpenStack Magnum
Orchestration

The Big 3 - What does Google Trends say?

Clearly Kubernetes has a lead in Google "search trends"

But we can expect "Docker Swarm" to make quick progress thanks to the new "swarm mode"
Docker Swarm
Docker Swarm

Dec 2014 ... **Docker Swarm** is announced
Orchestration using Docker Compose

Jun 2016 ... **Swarm Toolkit** released
OpenSource Orchestration Toolkit

Jun 2016 ... **Swarm Mode** announced
Orchestration integrated into Docker Engine

*Docker 1.12* is the first release to integrate "**Swarm Mode**" The original *Docker Swarm* is maintained for legacy use.

"**Swarm Mode**" is a revolution bringing:

- Orchestration directly in the Docker Engine
- Advanced networking features
  - mesh network, vxlan
- Load balancing
- Service Discovery
- *Docker* traditional ease of use

[docker.com](http://docker.com)  @hguemar  @mjbright  @mariolet
Docker Swarm

Architecture

Tasks:

- Docker Engine
- Gossip Network
- gRPC

Apps:

- App1
- ... AppN

Nodes:

- Master
- Worker

Network:

- Distrib Cluster Config (quorum)
Using Docker "Swarm Mode"

Create a new swarm by creating the master node:

```bash
$ docker swarm init --advertise-addr 192.168.2.100
Swarm initialized: current node (dxn1zf6l1qsb1josjja83) is now a manager.
```

Join a new Worker node to the swarm:

```bash
$ docker swarm join --token TOKEN 192.168.2.100:2377
```

Join a new Master node to the swarm:

```bash
$ docker swarm join-token manager
```

docker.com  @hguemar @mjbright @mariolet
Docker Swarm Demo
Docker Swarm Demo

- Creation of a 3 node cluster
- Run a service on the cluster and scale it to 3 replicas
- Make a rolling update of the service
- Drain a node of the cluster
Docker Swarm

Getting started

**Bruno Cornecs' Docker 101 Lab**
New to Docker? Come to [http://sched.co/7oHf](http://sched.co/7oHf)

- Docker 101 Lab, 9am - Friday 7th October
  
  [https://github.com/bcornec/Labs/tree/master/Docker](https://github.com/bcornec/Labs/tree/master/Docker)

Followed by ..

**Jerome Petazzoni's Orchestration Workshop**

or more advanced? Come to [http://sched.co/7oHx](http://sched.co/7oHx)

- Orchestrating Containers in Production at Scale with Docker Swarm, Friday 7th October
  
  [https://github.com/jpetazzo/orchestration-workshop](https://github.com/jpetazzo/orchestration-workshop)

---

docker.com  @hguemar  @mjbright  @mariolet
Kubernetes

From the Greek: "Steersman, helmsman, sailing master"
Kubernetes

Google created based on extensive experience running containers internally ~ billions of containers a year

Started Oct 2014, reached v1.0 in July 2015, now at v1.4

Managed by the Cloud Native Computing Foundation
https://cncf.io/

Commercial offerings from CoreOS (Tectonic) and Canonical

Integrated in:

- GKE (Google Container Engine)
- OpenStack above Kubernetes
  - Stackanetes (CoreOS, uses Tectonic)
    - Self healing OpenStack demo
  - Mirantis (OpenStack CI/CD based on Kolla)
- Various PaaS:
  - RedHat OpenShift CP
  - HPE Stackato v.40
  - Deis

kubernetes.io  @hguemar @mjbright @mariolet
Kubernetes Architecture

Kubernetes

Master

Worker

Pods: 192.168.100.10

App1  ...  AppN

etcd

Services
Labels, Selectors

kubernetes.io  @hguemar @mjbright @mariolet
Apache Mesos

The most proven orchestrator today, exists since 2009.

Can scale to ~ 10,000 nodes.

Used in production by:

- Uber
- Twitter
- Paypal
- Hubspot
- Airbnb
- eBay
- Groupon
- Netflix

Supports Containerizers to isolate tasks

mesos.apache.org
Apache Mesos

Architecture

Master

Worker (agent)

Tasks:

Spark Executor

Resources Available

Register

Offer

Accept

Spark

Scheduler

AppN

Master

Worker (agent)

Tasks:

Spark Executor

Master

Worker (agent)

Executor

Distrib

Cluster

Config

(quorum)

mesos.apache.org
Apache Mesos

Mesos is used in conjunction with Frameworks such as

- For long running tasks:
  - Marathon (Mesosphere), Aurora or Singularity

- For job orchestration:
  - Chronos "cron", Jenkins

- For Big Data Processing:
  - Hadoop, Spark, Storm
  - Cassandra, ElasticSearch, ...

mesos.apache.org
@hguemar @mjbright @mariolet
So isn't it time we told you what to choose?
... let's just compare them ...
What's common

Docker Swarm and Kubernetes are creating rich Orchestration stacks with integrated runtimes.

They're moving incredibly quickly ...

They are adding features such as networking capabilities, load balancing, services, labels.

They have a more 'declarative' approach

They support or are looking to support different runtime engines (*)

@hguemar @mjbright @mariolet
Docker "Swarm Mode"

Simple to use (despite underlying complexity)
All-in-one container engine plus orchestration
Uses Docker API and familiar docker commands
Advanced networking
  - mesh networking
  - Load Balancing and Service Discovery

Replication

@hguemar @mjbright @mariolet
What advantages?

Kubernetes

Rich conceptual model
Pods as groupings of containers
Labels and Selectors (for all components)
Large ecosystem
Networking
- Load Balancing and Service Discovery

Replication

@hguemar @mjbright @mariolet
What advantages?

Apache Mesos

- Most mature
- Battle tested by many service providers
- Scales to 10,000 nodes
- DataCenter OS - appears as 1 resource
- Not just containers
- Many frameworks available
- Difficult ramp-up

@hguemar @mjbright @mariolet
Hands on ...
Hands-on

Come along

This afternoon's tutorial session led by Mario:
Tuesday, October 4 - 15:30 - 16:20

5 Containers for 5 Languages: Patterns for Software Development Using Containers - Mario Loriedo, Red Hat

Tomorrow's lab session led by Haikel:
Wednesday, October 5 - 11:00 - 12:50

Container Orchestration Lab: Swarm, Mesos, Kubernetes - Haïkel Guémar, Fedora Project

Lab setup instructions here

- Docker Swarm
- Kubernetes
- Apache Mesos

@hguemar @mjbright @mariolet
Questions?
Thank you

@hguemar @mjbright @mariolet
Resources

@hguemar @mjbright @mariolet
<table>
<thead>
<tr>
<th>Publisher</th>
<th>Title</th>
<th>Author</th>
</tr>
</thead>
<tbody>
<tr>
<td>OReilly</td>
<td>Docker Cookbook</td>
<td>Sébastien Goasguen</td>
</tr>
<tr>
<td>OReilly</td>
<td>Docker Up &amp; Running</td>
<td>Karl Matthias, Sean P. Kane</td>
</tr>
<tr>
<td>OReilly</td>
<td>Using Docker</td>
<td>Adrian Mouat</td>
</tr>
<tr>
<td></td>
<td>[Early Access]</td>
<td></td>
</tr>
<tr>
<td>OReilly</td>
<td>Kubernetes Up &amp; Running</td>
<td>Kelsey Hightower</td>
</tr>
<tr>
<td>Manning</td>
<td>[MEAP] CoreOS in Action</td>
<td>Matt Bailey</td>
</tr>
<tr>
<td>Manning</td>
<td>[MEAP] Kubernetes in Action</td>
<td>Marko Lukša</td>
</tr>
</tbody>
</table>
Resources

Articles/Organisms

Cloud Native Computing Foundation - Kubernetes, Prometheus https://cncf.io/


@hguemar @mjbright @mariolet
Resources

Videos

- June 2016 - Container Orchestration Wars, Karl Isenberg, Mesosphere
- Mar 2016 - Container Orchestration with Kubernetes, Docker Swarm & Mesos-Marathon - Adrian Mouat, Container Solutions
- Jan 2016 - Docker, Kubernetes, and Mesos: Compared, Adrian Otto, Southern California Linux Expo

Repos

@hguemar @mjbright @mariolet
Documentation

- **Getting started guides**
  - Creating a Kubernetes Cluster
  - port Kubernetes to a new environment
    - in Getting Started from Scratch
- **User documentation**
  - to run programs on an existing Kubernetes cluster
    - Kubernetes User Guide: Managing Applications
  - the **Kubectl Command Line Interface** is a detailed reference on the kubectl CLI
  - User FAQ

kubernetes.io  @hguemar @mjbright @mariolet
• **Cluster administrator documentation**
  ◦ for people who want to create a Kubernetes cluster and administer it
  ◦ in the [Kubernetes Cluster Admin Guide](#)

• **Developer and API documentation**
  ◦ to write programs using the Kubernetes API, write plugins or extensions, or modify core code
  ◦ [Kubernetes Developer Guide](#)
  ◦ [notes on the API](#)
  ◦ [API object documentation](#), a detailed description of all fields found in the core API objects

• **Walkthroughs and examples**
  ◦ hands-on introduction and example config files
  ◦ in the [user guide](#)
  ◦ in the [docs/examples directory](#)

• **Contributions from the Kubernetes community**
  ◦ in the [docs/contrib directory](#)

---

[kubernetes.io](https://kubernetes.io)  @hguemar @mjbright @mariolet
- Design documentation and design proposals
  - to understand the design of Kubernetes, and
    feature proposals
  - Kubernetes Design Overview and the
docs/design directory
  - docs/proposals directory

- Wiki/FAQ
  - the wiki
  - troubleshooting guide

Community, discussion, contribution, and support

Consider joining the Cloud Native Computing
Foundation. For details about who's involved and how
Kubernetes plays a role, read their announcement.