Introduction to Swarm
a Docker-native clustering system

@aluzzardi - @vieux - @abronan
Alexandre Beslic
@abronan
Swarm in a nutshell
Docker Engine

Docker CLI

Docker CLI

Docker CLI
With Docker Swarm
Swarm 0.3.0

• **Docker REST API** (99%)

• **Resource management** (CPU, Mem, Networking)

• **Advanced scheduling** with constraints and affinities

• **Multiple Discovery Backends** (hub, etcd, consul, zookeeper)

• **TLS**: Encryption & Authentication

• **Multi Tenancy** / Leader Election
Setup using the hosted discovery service

- Create a cluster:
  
  ```bash
  $ swarm create
  ```

- Add nodes to a cluster:
  
  ```bash
  $ swarm join --add=<node_ip> token://<token>
  ```

- Start Swarm
  
  ```bash
  $ swarm manage --addr=<swarm_ip> token://<token>
  ```

Or you can use your own etcd, zookeeper or consul

Contributions are welcome ☺
Swarm Scheduler

2 steps:

• 1- Apply filters to exclude nodes
  - ports
  - constraints
  - affinity
  - health
  - dependency

• 2- Use a strategy to pick the best node
  - binpack
  - spread
  - random

Contributions are welcome ☺
Resource Management

• Memory
  $ docker run -m 1g ...

• CPU
  $ docker run -c 1 ...

• Ports
  $ docker run -p 80:80 ...

• More to come, ex: network interfaces
Constraints

- Standard constraints induced from docker info
  docker run -e "constraint:operatingsystem==*fedora*" ...
  docker run -e "constraint:storagedriver==*aufs*" ...

- Custom constraints with host labels
  docker -d --label "region==us-east"
  docker run -e "constraint:region==us-east" ...

- Pin a container to a specific host
  docker run -e "constraint:node==ubuntu-2" ...
Affinities

• Containers affinities
  `docker run --name web nginx`
  `docker run -e “affinity:container==web” logger`

• Containers Anti-affinities
  `docker run --name redis-master redis`
  `docker run --name redis-slave -e “affinity:container!=redis*” …`

• Images affinities
  `docker run -e “affinity:image==redis” redis`
Soft Affinities

- Containers soft affinities
  
  `docker run -d --name redis5 -e affinity:container!~=redis* redis`
Swarm Beta: Integrations

• Fully integrated with **Machine**
  
  $\textit{machine create }-d\textit{ azure }--\textit{swarm }--\textit{swarm-discovery token://<token>} \ldots$

• Partially integrated with **Compose**

  $\textit{DOCKER\_HOST=}<\textit{swarm\_addr}> \textit{compose up}$

• **Mesos & DCOS** integration has started in collaboration with **Mesosphere**.

  $\textit{swarm manage }-c\textit{ mesos zk://<zookeeper\_addr>/swarm}$
Swarm Beta: Upcoming features

• Attempt at Re-scheduling on Node failure

• Overlay networking (libnetwork integration into docker)
Demo
Scheduling containers on Swarm
Demo

us-east

hdd

us-west

ssd

Docker CLI

RAM: 4GB
CPU: 1core

Swarm
Swarm, Kubernetes and Mesos
Overview
Swarm Agent Internal

Host
Swarm

• Serves the standard Docker API
• Easy to get started and create a cluster
• Completely stateless
• Reduces the operational complexity
• Can fall short in some aspects (replication, services, network)
• More reliable on a cross Cloud provider scenario
Kubernetes

API  Kubernetes API / kubectl

Masters

Minions  Pod

.Metadata Storage

.Flat shared network namespace

.yml Description
Kubernetes

• Centered on Service Management
• Very Powerful API, notion of Replication and Service
• Stateful / Stateless model (rely on the store heavily for watch events)
• Hard to get started and troubleshoot
• A few opinionated decision that might bother some (Pod, Networking)
• Can be hard to deploy on anything else than GCE
Mesos

Frameworks

Schedulers/Plugins/Frameworks

Masters

Slaves

Host/Bridge networking

Storage

zookeeper
Mesos Slave Internal

Mesos Slave

Master

Hadoop Executor

Container

VM

Hypervisor

task

task

or
Mesos

- Centered on Resource Management
- Widely adopted
- Flexible model with the use of Frameworks
- Well fitted / mature for Big Data workloads (Hadoop, Spark, etc..)
- Can be hard to troubleshoot and deploy
- Badly coded Frameworks can hurt the Cluster resource utilization (unfair share)
Machine + Swarm + Compose
Swarm + Machine + Compose

Compose

.yml Description

Docker CLI

Machine

create / destroy

Compose

up / scale / stop / kill / rm
Mesos Integration
Mesos cluster
Mesos cluster + Docker Swarm

- Mesos CLI
- Docker CLI
- Marathon
Mesos cluster + Docker Swarm

Mesos CLI + Docker Compose

Docker CLI

Docker Swarm

Marathon

Ubuntu + Docker

Fedora + Docker

YML
Thank You. Questions?

http://github.com/docker/swarm

#docker-swarm on freenode

@aluzzardi - @vieux - @abronan