Container Pods with Docker Compose in Apache Mesos

Meghdoot Bhattacharya , Cloud Engineering @ PayPal

Apachecon 2017

©2015 PayPal Inc. Confidential and proprietary.

Summary

Goals:

1. *Treating Apache Mesos and docker as first class citizens, the platform needs to seamlessly run and scale docker container pods in Mesos with a standardized pod spec file.*

2. *Developers can develop and run the pod locally using a spec file and then use the same spec file to launch it in a QA/Production cluster.*

Solution:

Docker Compose Mesos Executor (https://github.com/PayPal/dce-go)

What are Pods?

- *Pods represent a collection of containers treated as a single unit for scheduling and deployment.*
- *Pods are treated as single scaling unit.*
- *Containers in Pods will generally share one or more namespaces: network, pid, ipc etc*
- *Containers in Pods should have a common cgroup to be kept under check as a unit to not steal resources from other pods in the host.*
- *Colocation using constraints != Pod*

Why are Pods needed?

- *Migrating legacy workloads running in a single node.*
  - *Lift and shift.*
  - *Gives time to extract common services duplicated in each pod into a system service when relevant.*

- *Pods helps to create a modular application by composing different services.*
  - *Side-car, Adapter, Ambassador are common patterns*

- *Pods helps eliminate pre and post deployment steps.*
  - *Helps model transient short tasks (short lived containers)*

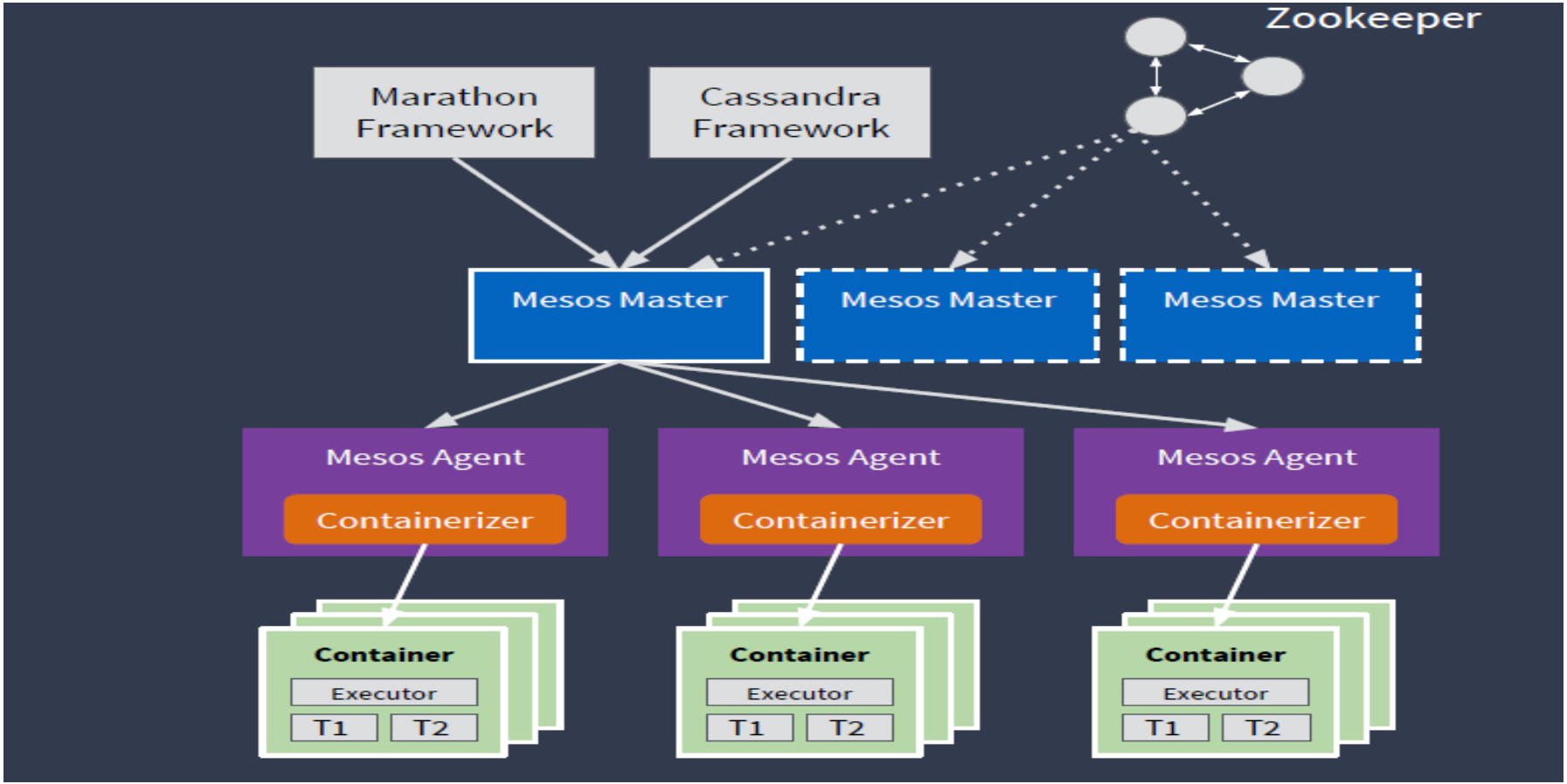
Docker Compose

- *Compose is a elegant tool for defining and running multiple docker containers.*
- *Cherished tool in the community over the years for local development.*
- *Version 2.X preserves strictly all the local features. In this version, it interacts with a single docker engine, mostly running locally.*
- *Version 3.X introduces compose for docker swarm and deprecates certain features of 2.X. For now, they remain as 2 separate version tracks.*

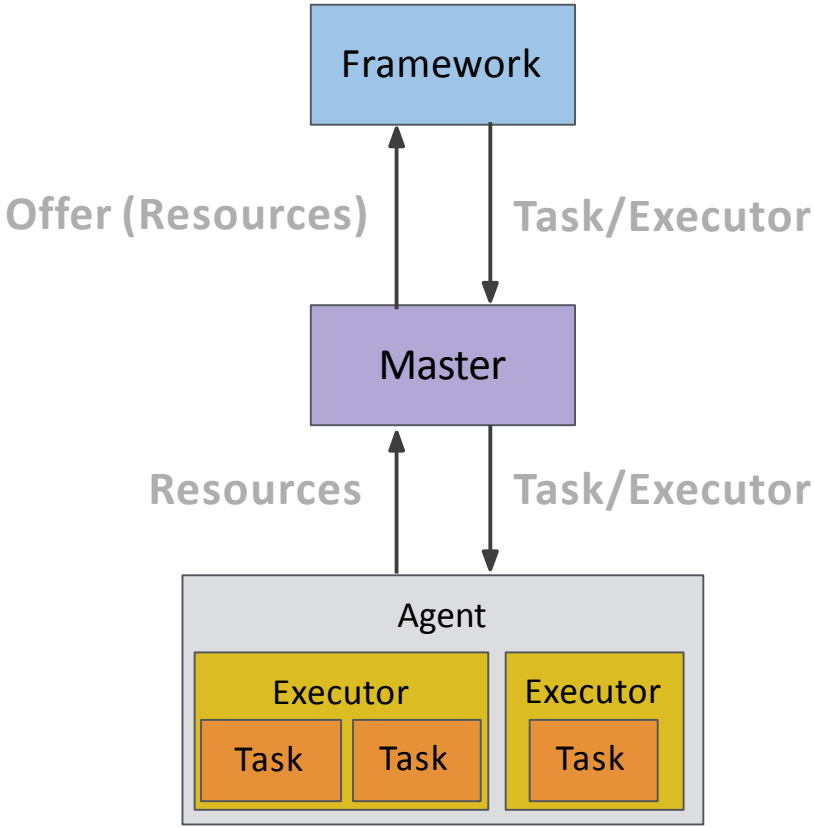
Pods Modelled in Docker Compose

- *Pods are containers bundled together locally. So, relies on version 2.x compose version.*
- *Pods represented by compose can preserve all the first class docker volume and network plugins.*
- *Pods can have flexibility on collapsing namespaces in any combination between the containers.*
- *Containers in Pods can have strict ordering guarantees by using conditional constructs of depends_on.*
- *Pods can refer to externally created volumes and networks.*
- *Multiple files can be merged to construct the right pod definition for an environment. Ex: base, qa, prod compose files.*
- *Easy to spawn multiple pods of same application with different versions in same local environment without conflicts.*

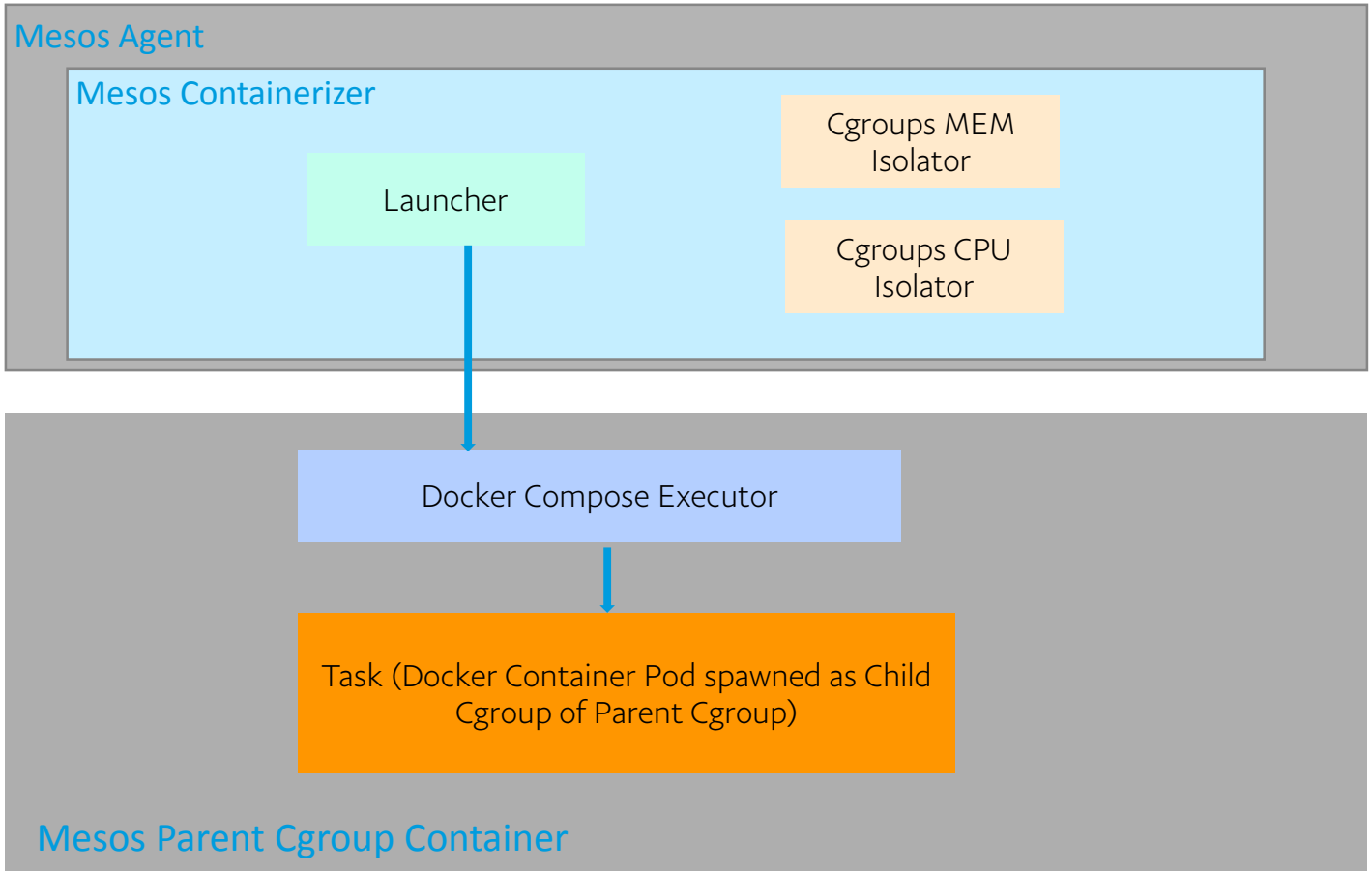
Mesos Architecture



Mesos Key Abstractions



Docker Compose Executor in Mesos



Cgroup Hierarchy

- *Each container in pod has a child cgroup under the parent mesos task cgroup. Meets Pod criteria of containers in pod sharing a cgroup.*
- *Cgroups CFS hard limits (bandwidth controls) and memory limits assigned to the parent cgroups will cover all containers in pods.*
- *Individual containers will not be limited unless specified but cannot go over parent.*
- *Make sure to enable memory hierarchy with use_hierarchy flag.*

DCE-GO features

- *Implements mesos executor callbacks to maintain the lifecycle of a pod.*
- *Massages compose file to add cgroup parent, mesos labels and edit certain sections to resolve any naming conflict etc*
- *Collapses network namespace by default.*
- *Provides pod monitor to not only kill entire pod on unexpected container exit but also when a container becomes unhealthy as per docker healthchecks.*
- *Supports running multiple compose files.*
- *Mesos Module provided to prevent pod leaks in rare case of executor crashes.*
- *Provides plugins.*
- *Last but not the least any existing Mesos Frameworks like Aurora, Marathon etc can use DCE directly without making ANY framework changes.*

PLUGINS

©2015 PayPal Inc. Confidential and proprietary.

What are Plugins?

- *Plugins provides a way to easily extend inner workings of DCE.*
- *Plugins can be used to customize DCE without having to understand exactly how DCE is implemented internally. Plugins make it easy to experiment with new features.*
- *Plugin mechanism helps you easily enable and disable features.*
- *Plugins essentially provide hooks before and after launch/kill task mesos callbacks to implement custom behavior.*
- *Plugins can be chained with ordering.*

Default Plugin

DCE-GO comes with default General Plugin. This Plugin updates compose files so that multiple pods are able to launch on a host. It largely covers following:

- *Decorate various compose sections to resolve all the conflicts.*
- *Label each container with specific taskId and executorId. This information is used to clean up pod.*
- *Adding pod to parent mesos task cgroup.*
- *Creating infrastructure container in pod for allowing to collapse network namespace for containers in a pod.*

Mesos Hook Module for Compose Pods

- *Mesos Modules help extend inner functionality using shared libs. Can run in Master and/or Agent.*
- *Mesos Modules should be built against the mesos version running in cluster.*
- *Different classification of Modules: Allocator, Isolator, Hooks etc*
- *Hooks Modules tie into events and their context. DCE-GO leverages executor removal event hook in an agent. Implements ComposePodCleanupHook Module.*
- *That hook ensures pods are cleaned up on any unexpected executor crash.*

Current Ecosystem around Pods

1. *Docker swarm (as of 1.2.6) does not support local pods.*

- *Docker compose deploy takes a compose definition and schedules the containers across swarm cluster and connects them via overlay network.*
- *Using constraints not the same.*
- *Most likely to be supported in future.*

2. *K8 has excellent support for pods but does not treat docker as first class.*

- *Different volume and network specs*
- *CRI mostly going to hook to containerd directly. Skip docker engine.*
- *Pod spec different than compose spec and docker commands do not work (equivalent command provided).*
- *Image is the only common thing.*

3. *Mesos added in 1.1 pod support via experimental Task Groups and Nested Container.*

- *Not docker specific and pod can represent any collection of tasks.*
- *Frameworks needs to make changes to support this.*
- *Task Groups spec obviously separate from compose spec.*
- *Universal Containerizer and set of isolators defining a container runtime separate from docker.*
- *However, Mesos continues to remain extremely flexible!*

DCE-GO DEMO

PayPal ©2015 PayPal Inc. Confidential and proprietary.

Links and References

- *DCE-GO project (https://github.com/PayPal/dce-go)*
- *Deprecates (https://github.com/mesos/docker-compose-executor)*
- *Mesos Architecture and Key Abstractions diagrams(https://www.slideshare.net/InfoQ/mesos-a-stateoftheart-container-orchestrator)*