Notes on Mesos and Docker - CSE 408222 – Cloud Computing

Caution: These are high level notes that I use to organize my lecture. You may find them useful for reviewing main points, but they aren't a substitute for participating in class

References:

- Mesos: A Platform for Fine-Grained Resource Sharing in the Data Center, Benjamin Hindman et al, USENIX NSDI 2011.
  http://static.usenix.org/events/nsdi11/tech/full_papers/Hindman_new.pdf

- Apache Mesos Project
  http://mesos.apache.org/

- Docker Project
  https://docs.docker.com/

Introduction

Every programming model has strengths and weaknesses, none are universally applicable: Hadoop, Spark, MPI, Makeflow, etc…

However, most systems assume that they own the whole cluster.

Non-solutions: static partitioning, assignment of VMs, global scheduling.

Key Idea: Mesos allocates resources to frameworks, which do their own scheduling.

Exactly this problem occurs at Facebook and Yahoo (and presumably elsewhere.)

Architecture

Master node, standby masters, slave nodes, and schedulers.

(Zookeeper tracks the master state and enables fail-over.)

Worker nodes report resource availability to the master.

Master offers resources to the schedulers.

Schedulers may accept offers and dispatch tasks, or reject offers.

Master dispatches tasks to the workers.

Optimization: schedulers may filter on node names.

Allocation Policy

Policy is pluggable: fair share and strict priority are common.

Resources are generally reallocated when tasks end.

However, if a task takes too long, it can be killed to reallocate.

Storage Management

Problem: storage goes away with an allocation is removed.

Two solutions:

- Run HDFS continuously as a cluster allocation within Mesos
- A user-level file server is provided to access local storage.
- Hadoop HDFS runs continuously to provide cluster storage.

Isolation and Compatibility

Two traditional approaches to compatibility:

- Run every task on the base OS – Lightweight but not robust.
- Run every task in a VM – Heavyweight but robust.

A middle approach: OS-level containers.

- Private namespace for filesystem, network, processes, etc…
- Private resource allocation enforced by the OS.
- One kernel shared among all containers.
- Relatively lightweight with most of the benefits of VMs.

Idea was presented as “Resource Containers” by G. Banga et all in OSDI 1999.

Implemented as “Zones” in Solaris and “Containers” in Linux.

Docker – A system for managing application containers.

Brings together a variety of Linux technologies to implement isolation.

Architecture: Docker command, dockerd, local registry, public registry.

Operations: run, stop, ps, commit, build, pull, push

Each process in a docker container gets its own personal filesystem.

Growing approach: build minimal OS image to run one application.

Summary

Mesos is like a cluster operating system – it provides the equivalent of malloc() to obtain resources on multiple nodes, then lets each framework implement the desired abstraction.

Docker provides isolation between applications on the same node, with less overhead than a full virtual machine.