[**Apache Mesos总体架构**](http://dongxicheng.org/apache-mesos/meso-architecture/)

## 1. 前言

同其他大部分分布式系统一样，Apache Mesos为了简化设计，也是采用了master/slave结构，为了解决master单点故障，将master做得尽可能地轻量级，其上面所有的元数据可以通过各个slave重新注册而进行重构，故很容易通过zookeeper解决该单点故障问题。

## 2. Apache mesos中的基本术语解释

(1)    **Mesos-master**：Mesos master，主要负责管理各个framework和slave，并将slave上的资源分配给各个framework

(2)    **Mesos-slave**：Mesos slave，负责管理本节点上的各个mesos-task，比如：为各个executor分配资源

(3)    **Framework**：计算框架，如：Hadoop，Spark等，通过MesosSchedulerDiver接入Mesos

(4)    **Executor**：执行器，安装到mesos-slave上，用于启动计算框架中的task。

当用户试图添加一种新的计算框架到Mesos中时，需要实现一个Framework scheduler和executor以接入Mesos。

## 3. 总体架构

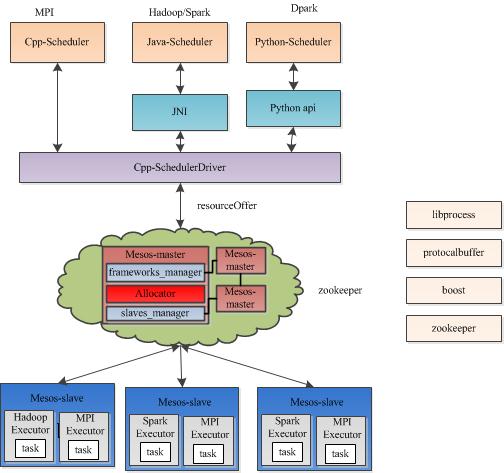
Apache Mesos由四个组件组成，分别是Mesos-master，mesos-slave，framework和executor。

Mesos-master是整个系统的核心，负责管理接入mesos的各个framework（由frameworks\_manager管理）和slave（由slaves\_manager管理），并将slave上的资源按照某种策略分配给framework（由独立插拔模块Allocator管理）。

Mesos-slave负责接收并执行来自mesos-master的命令、管理节点上的mesos-task，并为各个task分配资源。mesos-slave将自己的资源量发送给mesos-master，由mesos-master中的Allocator模块决定将资源分配给哪个framework，当前考虑的资源有CPU和内存两种，也就是说，mesos-slave会将CPU个数和内存量发送给mesos-master，而用户提交作业时，需要指定每个任务需要的CPU个数和内存量，这样，当任务运行时，mesos-slave会将任务放到包含固定资源的linux container中运行，以达到资源隔离的效果。很明显，master存在单点故障问题，为此，mesos采用了zookeeper解决该问题。

Framework是指外部的计算框架，如Hadoop，Spark等，这些计算框架可通过注册的方式接入mesos，以便mesos进行统一管理和资源分配。Mesos要求可接入的框架必须有一个调度器模块，该调度器负责框架内部的任务调度。当一个framework想要接入mesos时，需要修改自己的调度器，以便向mesos注册，并获取mesos分配给自己的资源， 这样再由自己的调度器将这些资源分配给框架中的任务，也就是说，整个mesos系统采用了双层调度框架：第一层，由mesos将资源分配给框架；第二层，框架自己的调度器将资源分配给自己内部的任务。当前Mesos支持三种语言编写的调度器，分别是C++，java和python，为了向各种调度器提供统一的接入方式，Mesos内部采用C++实现了一个MesosSchedulerDriver（调度器驱动器），framework的调度器可调用该driver中的接口与Mesos-master交互，完成一系列功能（如注册，资源分配等）。

Executor主要用于启动框架内部的task。由于不同的框架，启动task的接口或者方式不同，当一个新的框架要接入mesos时，需要编写一个executor，告诉mesos如何启动该框架中的task。为了向各种框架提供统一的执行器编写方式，Mesos内部采用C++实现了一个MesosExecutorDiver（执行器驱动器），framework可通过该驱动器的相关接口告诉mesos启动task的方法。



转载自[董的博客](http://dongxicheng.org/) posted:2012-04-15