Slider App Management Design

——fys@cmss.com

Last updated:2016.7.4

# Problem Statement

Marathon及相关软件如下所示:



Marathon的实现与Slider不同，Slider仅作为YARN app的提交客户端，启动SliderAppMaster后由SliderAppMaster进行程序的管理。

Marathon与SliderAppMaster在mesos与YARN的位置相同。但是SliderAppMaster仅启动了任务的调度与程序的启动（也包括yarn container异常重启），其与Marathon相比功能太单薄，相比而言缺少以下模块：

1. HealthCheck模块，在Slider使用过程中，发现YARN App启动后，app相关进程并没有实时启动完成，会发生yarn状态处于RUNNING，但是App并未启动的状态（app不可用）。
2. GroupManager，启动的应用无群组关系
3. 缺少ArtifactStore，作为存储app依赖数据，例如tomcat的war包等
4. Constraints，指定app启动的主机，当app启动在某一主机时，运行过程中会产生数据写到本地磁盘，例如Mysql存储数据，再次启动时需要读取该数据。但是Slider并不能保证一定会启动在该台主机上（当前在该主机空闲时，才会启动在该主机上，并不保证）。Marathon支持多种条件的主机分配策略
5. Security，SliderAppMaster启动程序后，并没有ACL的关系，仅应用的所属者才能操作该应用
6. Event bus，事件订阅机制，需要进一步使用及调研
7. 服务发现及负载均衡模块，SliderAppMaster将启动的app信息（export）写入zk中，没有提供服务发现及负载均衡机制。Marathon在该模块相关的两个组件为Marathon-lb和Mesos-DNS。Marathon-lb使用HAProxy作为服务发现服务器，比较类似于我们当前使用的nginx方案，启动Marathon-lb使用event bus作为load balancer的数据源机制还需要进一步调研。Mesos-DNS实现Hostname与Ip的解析，启动SRV记录实现的hostname与ip:port的对应，其直接和Mesos-Master通信，和Marathon无必然对应关系。
8. Auto-Scale，自动扩展功能，marathon-autoscale
9. Docker
10. 架构分析



1. yarn docker container和docker daemon
2. YARN ASSEMBLE

# Architecture



Slider框架图

1. Health Check

http |tcp |command

1. Artifact Store

#hdfs

1. Constraints调度

strict placement | anti-affinity-desired | affinity

Slider-82 ,Slider-868,YARN-1042

1. Docker support
2. Nginx重构
3. 引入DNS
4. Auto-scale
5. Metrics
6. REST Service及Web UI

# 下一步工作计划：

Hcontrol部分

Health Check

Web UI

BDOC部分

参考slider hbase的实现，但是是否需要优化，与Hcontrol的结合？？