Launching Docker Containers in YARN

Runtime system有时候简称为run-time system，程序执行时一定的策略，包括在所在计算机的系统及硬件资源的集合（不仅仅是程序执行所依赖的语言执行库）。从OS view去看，RTS是用户程序的一部分，但是RTS并不由编程者去编写，可以理解RTS是程序的execution model，不属于操作系统，不由开发者显式提供，包括程序执行所需要的所有条件。

从YARN上角度讲，容器技术（Container）利用操作系统虚拟技术隔离运行的进程，有多种Container Runtime。当前社区最活跃的是Docker，基于Linux Container技术整合了一堆易用接口构建的轻量的高级容器引擎。YARN的执行框架如下图所示：



YARN执行框架

在YARN中，ApplicationMasters将Tasks分配到已启动的Containers执行。Container只执行分配给自己的任务。但是Containers之间可能会相互抢占内存，cpu及文件系统资源等，例如Container在运行过程中安装的软件可能会干扰其他Container的执行。例如两个Ruby streaming jobs需要两个版本的Ruby Gem，这可能需要所有的NM都有这两个版本。

Hadoop和YARN在应用场景中越来越多，在Hadoop Clusters中会执行越来越复杂的Jobs。用户可能会在相同的集群中运行Matlab、R及Ruby streaming jobs，这就需要在所有的NM上安装所有Matlab,R和Ruby streaming lib库。这给Cluster Admin用户带来复杂的配置管理和维护成本。

但是如果Tasks运行在Docker containers中，每个Task拥有自己的文件系统，因此用户可以在container中安装任意版本的软件，这样Tasks之间不会相互影响。而且Docker Containers的cpu和memory的资源隔离更充分，同时可以以non-privileged users的方式执行：

1. DefaultContainerExcecutor，将user yarn作为privileged user，通过该用户运行tasks，without any guards around them
2. LinuxContainerExecutor，使用Non-privileged user来提交Job，有cpu和memory限制
3. DockerContainerExecutor，和LCE一样以non-privileged user来运行的Task，cpu和memory限制。而且Tasks之间独立的文件系统，不会产生依赖冲突。

注:hadoop 3.0中已经没有了DCE，Docker的启动及管理由LCE来管理，在LCE中增加了ContainterRuntime相关实现，将LCE升级了成类似于CompositeContainerExecutor的角色。

使用LinuxContainerExecutor 中的DockerRuntime（YARN-3611），将YARN NM服务可以将其container进程运行在Docker容器中。用户可以自定义运行YARN Container的Docker镜像。这些Container提供了自定义的，与外部NM隔离的运行环境，还可以指定安装与NM不同版本的软件包，设置可以运行与NM不同的Linux发型版本。

在YARN中运行Docker的系统架构图如下所示：



Running Docker in YARN

注: 在Hadoop 2.9之前，使用DockerContainerExecutor(YARN-1964)，YARN中启动Docker Container，Hadoop 2.9中将该模块从YARN中去除（YARN-5388）。在Hadoop 3.0中，使用LCE的DockerRuntime实现相同的功能。