## 第2章 Slider设计理念与基本架构

在第1章，介绍了Slider学习环境的搭建方法，这是学习Slider需要进行的是最基本的准备工作。在这一章中，我们做将从设计理念和基本架构方面对Slider进行介绍，这也是准备工作的一部分。通过本章的介绍将会为下面几章深入剖析Slider内部实现奠定基础。将已有的分布式应用运行在资源管理系统YARN，需要专业人员开发客户端和ApplicationMaster，可能需要更改应用程序，难度较大。Slider提供了将其他应用运行在YARN中的能力。在本章，将从背景、设计思想和基本架构方面对Slider框架进行介绍。

### 2.1 Slider产生背景

为了解决MR1扩展性差、可靠性差、资源利用率低和无法支持多种计算框架等局限性，Apache社区将其升级了计算框架MRv2。Hadoop将资源管理功能抽象成了一个独立的通用系统YARN，如下图所示：

图2-1 以MapReduce为核心和以YARN为核心的软件栈对比

在以MapReduce为核心的软件栈中，资源管理系统YARN是可插拔的，比例选择Mesos替代YARN，一旦MapReduce接口改变，所有的资源管理系统的实现均需要跟着改变。但是以YARN为核心的软件栈则不同，所有框架都需要实现YARN定义的对外接口以允许在YARN之上，从而形成一个以YARN核心的生态系统。

随着YARN的功能完善，其已形成了一个弹性计算平台，支持多种框架，如下图所示：

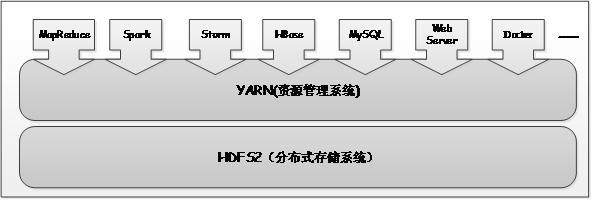


图2-2 以YARN为核心的生态系统

在YARN中可以运行MapReduce、Spark这样的短作业，也可以部署向Tomcat、MySQL这种长服务。YARN根据各种计算框架、应用的负载或者需求去调整各自占用的资源，实现集群资源共享计弹性收缩。

开发者为了使YARN支持自己的计算框架或者服务，需要编写组件Client（客户端）和ApplicationMaster，向YARN申请资源、任务调度与容错、网络容错，并且要监控各个任务状态等细节，对开发者要求过高，而且每增加一个框架或者服务都要重新编写相对应的组件，重复工作；如果想将已有服务运行在YARN中，需要修改服务的代码，以与YARN框架兼容，代价较大。

为了解决这个问题，Apache社区推出了Slider，其源于Hoya，一个尝试将HBase运行在YARN中的项目。Apache Slider目前是孵化项目，可以使用户在不对已存在服务进行任务修改的情况下部署到YARN集群中，并提供部署、管理及扩展等操作。

同 时，Slider可以在集群中部署多版本、异构的应用，每个应用可以根据需求配置不同的参数，在运行过程中动态的扩展或者减少资源的应用，并且对失败的进程透明的回复。在YARN中，每个应用都以Application的形式运行，每个进程都运，行在Container中，Container目录中包含了应用的配置、脚本和数据等；在YARN中运行应用，可以使应用充分整合Hadoop生态系统，例如Hadoop数据，计算/ 存储资源及安全、管理和操作能力。

### 2.2 Slider基础知识

为了便于下面分析Apache Slider， 本小节对Slider涉及的术语进行比较全面的介绍。

1. YARN

YARN是Hadoop 2.0中的资源管理系统，它是一个通用的资源管理模块，可为各类应用程序进行资源管理和调度。YARN不仅局限于MR一种框架使用，也可以提供其他框架，比如TEZ、Spark及Storm等。

1. Application

可以运行在YARN中的程序都可以成为Application，包括计算框架MapReduce、内存计算、流计算等；常驻服务，如Apache HBase、Accumulo及Storm等。其中Slider是运行在YARN中用于Application管理的框架，本质也是Application。

1. Instances

Application程序在YARN中的一次执行，YARN Cluster根据应用配置文件启动的应用实例，实例的状态可以是运行状态或者停止状态，当停止时配置信息或者Instance数据保根据存在HDFS中。同一个Application可以在YARN中同时执行多次（共享相同的服务器）。每个Application Instance由一个或者多个Componet的集合组成，Application State是Instance的模型，记录每个Instances的正确状态、所需要的资源、运行历史信息等。

1. Components

Component是Application Instances的组成构件，代表应用的程序的一个运行模块，例如HBase中的HMaster、HRegionServer、Rest Server都可以称为Component，其以YARN Container的形式运行在YARN中。不同的Component可以执行不同的程序或者命令，也可以使用不同的配置选项和参数。

1. Role

ApplicationMaster为了实现对Component（运行在YARN Container中）进行管理，需要对Component及其Container的实时运行信息进行记录及更新。Component在ApplicationMaster的映射形式为Role（Server-Sider login），其实例化为RoleInstances。SliderAppMaster根据Role的信息进行任务的调度和服务的管理。根据具体的Component不同分为SliderAppMaster Role和Component Role。

1. Service Registry

客户端需要访问运行在YARN中的服务，但是客户端可能运行在集群外，而且运行在YARN中的服务是随机分配到各个主机上，不可预测而且端口也是随机的，为了解决这些问题需要引入服务注册（Service Registry）。服务的注册的方式很多，包括ZK、DNS、Floating IP Address、LDAP等。

当前Slider使用的方式是YARN Service Registry，将YARN Application的binding信息发布到ZK中，包括IPC端口,Service-aware URL及配置等，客户端从ZK中获取所需要的访问信息。对于Service Registry，除了ZK，有很多的替代方式，如Apache Curator、Helix Service Registry ,Twill Registry等，以后会进行详述。

### 2.3 Slider基本架构

Slider将分布式应用以YARN Application的形式运行在YARN中，其基本设计思想是将分布式应用的服务进程以Container的形式运行在YARN资源管理系统中，客户端通过与YARN的交互来对应用进行管理。

#### 2.3.1 Slider基本组成结构

Slider（YARN Application）的总体结构仍然是Master/Slave结构，运行在资源管理系统YARN中，SliderAppMaster为Master，Slider Agent为Slave。SliderAppMaster向RM申请资源，并要求NM启动Sllider Agent进程，Agent根据服务的定义启动相对应的Component。

图2-3描述了Slider的基本组成结构，Slider主要由SliderAppMaster、Slider Agent和App Component（由Linux Service启动，可以是Docker或者JVM程序等不同程序），Agent启动在Container中，但是Component进程由Linux系统启动，可能会不属于Cotainer。



图2-3 Apache Slider的基本架构

1. SliderAppMaster

根据app的配置解析并生成Application的启动计划，SliderAppMaster根据计划向YARN申请Container(对应一个Component)。在Application Instance启动后，监控其运行状态，并当服务的Component因异常停止后，自动恢复。

1. Slider Agent

SliderAppMaster申请到Container后，在Container中启动Slider Agent进程，Agent向Slider AppMaster注册并发送心跳信息，Agent根据心跳响应获取SliderAppMaster发送的命令并执行，包括服务的启动、停止及其他操作。

1. App Component

每个application instance有一个或者多个Component的集合，每个Component有不同的程序或者命令，使用的配置和参数也会不同，例如HBase HMaster component会通过Slider Agent启动HMaster JVM。当前Slider Agent也支持Docker App的启动。

1. Slider Client

与SliderAppMaster及YARN通过REST或者RPC通信，对外提供CLI和底层API来操作Application

#### 2.3.2 Slider 通信协议

通信协议是Slider各个组件的大动脉，了解不同组件之间的通信协议有助于更深入的学习Slider框架，其中RPC协议是连接各个组件的“大动脉”，对于RPC，通信双方有一端是Client，另一端是Server，且Client总是主动连接Server，因此YARN中的组件采用的是pull-based通信模型。另外SliderAppMaster对外提供REST通信方式，主要用于与Slider Agent的心跳与注册过程，采用的Pull-based通信模型。如图2-4所示，实线箭头指向的组件是RPC Server，而箭头尾部的组件是RCP Client，虚线箭头指向REST Server，虚线箭尾指向REST Client端，由以下部分组成：



图 2-4 Apache Slider的交互协议

1. Slider Client与RM之间的协议——ApplicationClientProtocol;Slider Client通过该RPC协议提交应用程序，查询应用程序状态等
2. Admin与RM之间的协议——ResourceManagerAdministrationProtocol；Admin通过该RPC协议更新系统配置文件，比如节点黑白名单，用户队列等。
3. AM与RM之间的协议——ApplicationMasterProtocol；AM通过该RPC协议向RM注册和撤销自己，并为各个任务申请资源
4. AM与NM之间的协议——ContainerManagementProtocol；AM通过该RPC协议要求NM启动或者停止Container，获取各个Container的使用状态等信息
5. NM与RM之间的协议——ResourceTracker；NM通过向该RPC协议向RM注册，并定时发送心跳信息汇报当前节点的资源使用情况和Container运行情况
6. Slider Client与AM之间的协议——SliderClusterProtocol；Slider Client通过该RPC协议请求AM进行App的信息的查询
7. Slider Op User与AM之间REST协议——ManagementResource、PublisherResource、RegistryResource及ApplicationResource，用户通过该REST协议进行应用信息的查询，服务的操作等
8. Slider Agent与AM之间的REST协议——AgentResource，Agent向AM注册，并实现Agent与AM之间的心跳信息的交互

### 2.4 Slider工作流程

通过Slider将常驻服务运行在YARN中，与常规的提交YARN App(eg,MR)大致相同，分为两个阶段运行该应用程序；第一个阶段是启动SliderAppMaster；第二个阶段是由SliderAppMaster创建应用程序，为它申请资源，并监控它的整个运行过程，直到运行完成。如图2-5所示，Slider的工作流程分为以下几个步骤：

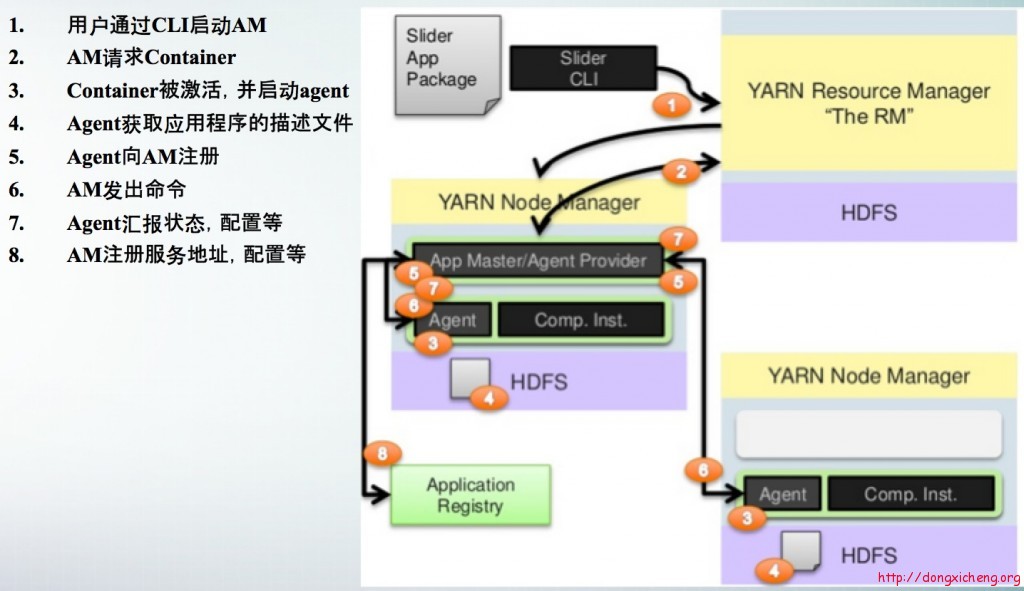


图2-5 Slider运行流程

步骤1：用户通过CLI启动SliderAppMaster，SliderClient通过调用create及start向YARN提交应用请求，分配资源并启动SliderAppMaster，启动过程中从App package中读取所需要的配置信息，初始化服务

步骤2：SliderAppMaster启动后，向YARN申请资源，并启动container

步骤3：Container被激活并启动Agent，SliderAppMaster分配Container后，启动Container，并启动Slider Agent

步骤4：Slider Agent的启动后，从应用程序的描述文件(metainfo.xml)中读取应用的启动脚本及目录

步骤5：Slider Agent向SliderAppMaster注册，通过SliderAppMaster向SliderAMWebApp服务的REST(/agent/register)完成注册

步骤6：SliderAppMaster向SliderAgent发送命令，Slider Agent根据应用程序的描述文件启动对应的脚本，启动对应的服务（Component）

步骤7：SliderAgent与SliderAppMaster进行心跳交互，SliderAgent向SliderAppMaster汇报服务状态及配置等

步骤8：SliderAppMaster通过YARN Registry向zk中注册服务地址，客户端获取服务地址，并对服务进行信息的查询和操作

### 2.5 小结

本章介绍了Slider的设计理念和基本架构，涉及到的内容较多，包括Slider产生背景、Slider术语解释、Slider架构和通信协议等。从YARN的角度上讲，Slider与MRAppMaster等Application的执行过程相同。在后面几章中，将深入探讨Slider内部实现原理，以便进一步深层次理解Slider。