Slider技术内幕

——深入解析Slider架构设计与实现原理

# 第一部分 准备篇

YARN是一个资源管理系统，负责集群资源的管理和分配，如果想要将一个新的应用程序运行在YARN之上，需要编写两个组件：客户端和ApplicationMaster，考虑RPC调用、任务容错等细节，往往由专业的开发人员编写这两个组件，并提供给上层的应用程序用户使用。Apache Slider源自于Hoya(将HBase部署在YARN之上的项目)，通过Slider可以在YARN的环境中运行其他应用，使开发者不需要针对所有应用开发ApplicationMaster组件。Slider使得分布式应用（也可以是非YARN-Aware应用）运行在YARN中，使用Hadoop的数据和计算资源。本文档第一部分将介绍学习Slider前的准备工作，并给出Slider的基本概念和架构。

## 第1章 环境准备

一般而言，在深入研究一个系统的技术之前，先要进行一些基本的准备工作，比如准备代码阅读环境，搭建运行环境并尝试使用该系统等。本章的目的是帮助开发者构建一个高效的Slider源码学习环境，包括阅读环境、使用环境和Slider源代码编译测试环境等，主要涉及如下内容：

* 在Linux环境下搭建Slider源码阅读环境
* 在Linux环境下搭建Slider使用环境
* Slider的基本使用方法，主要介绍Slider Shell
* Slider源代码编译和调试方法，打印Slider的调试日志

### 1.1 准备学习环境

对于大部分开发者而言，实验和生产环境中的服务器集群部署的是Linux操作系统，考虑到Linux服务器在服务器市场中具有的统计地位，本文档仅介绍基于Linux操作系统的简介和使用（支持Windows系统）。Slider采用了具有跨平台特性的Java和Python作为主要的编程语言，Server端代码基本使用Java语言，Agent端使用Python语言，下面会进行介绍。

#### 1.1.1 基础软件下载

前面提到Slider采用的开发语言是Java和Python，而且搭建Slider环境所需要最基础软件应该包括Java基础开发包JDK、项目管理工具Maven和Python基本开发库，考虑到源代码阅读和调试的便利性，本书采用功能强大的Intellij IDEA，搭建Slider阅读环境需要的各种软件以及下载方式如表1-1所示：

表1-1 搭建Slider阅读环境所需的软件

|  |  |  |
| --- | --- | --- |
| 软件 | 下载网址 | 推荐版本 |
| JDK | http://www.oracle.com/technetwork/java/javase/overview/index.html | 1.7以上 |
| Python | https://www.python.org/downloads/ | 2.6以上 |
| MAVEN | https://maven.apache.org/download.cgi | 3.2.0以上 |
| Intellij Idea | http://www.jetbrains.com/idea/download/ | 13.0以上 |

#### 1.1.2 如何准备Linux环境

本节介绍如何准备Linux下的Slider学习环境，搭建Linux学习环境需要安装JDK和Intellij IDEA等软件，为了方便介绍Slider源代码编译方法，顺便安装Slider项目管理工具Javen，本文以64 bit Centos为例，介绍安装这些软件的方法，最终安装完成的目录结构为:

ROOT

|—home

| |—feng

| |—tools

| |—idea-IC-135

|—usr

|—lib

|—apache-maven-3.2.3

|—jvm

|—jdk1.7.0\_45

|—python 2.6

1.JDK的安装和配置

一般而言，Centos系统不会自带JDK，可按一下步骤进行安装

步骤1：安装JDK

将下载的tar.gz文件复制到Linux的某个目录下，比如/usr/lib/jvm，然后在Shell中执行以下命令解压JDK安装文件：

$tar -zxvf jdk-7u45-linux-x64.tar.gz

文件解压到/usr/lib/jvm目录，下面进行配置：

步骤2: 配置JDK

修改/etc/profile文件，在里面添加以下内容：

export JAVA\_HOME=/usr/lib/jvm/jdk1.7.0\_45

export PATH=$PATH:$JAVA\_HOME/bin

export CLASSPATH=$CLASSPATH:$JAVA\_HOME/lib:$JAVA\_HOME/jre/lib；

步骤3: 验证JDK是否安装成功

重启Shell终端，执行java -version命令，若输出以下内容，则说明安装成功：

java version "1.7.0\_45"

Java(TM) SE Runtime Environment (build 1.7.0\_45-b18)

Java HotSpot(TM) 64-Bit Server VM (build 24.45-b08, mixed mode)

2.安装、配置Maven及Intellij IDEA

下面介绍Maven和Intellij IDEA的安装和配置方法

1）安装与配置Maven

首先解压下载包，比如解压到文件/usr/lib/apache-maven-3.2.3目录下，然后修改/etc/profile文件，在里面添加以下内容

export MAVEN\_HOME=/usr/lib/apache-maven-3.2.3

export PATH=$PATH:$MAVEN\_HOME/bin

输入以下命令使配置生效：

source /etc/profile

同Windows下的验证方式一样，重启中断，执行mvn -version命令，若输入以下内容，则说安装成功：

Apache Maven 3.2.3 (33f8c3e10656a31e8fdf4; 2014-08-12T04:58:10+08:00)

2）安装Intellij IDEA

同Windows环境下安装方式一样，直接解压即可使用

### 1.2 获取Slider源代码

Apache Slider 主要由Hontonworks研发，目前处于孵化阶段，在该版本基础上，我们进行了一些封装和优化，推出了自己的版本，但是，本文主要以Apache Slider作为分析对象，可以从Slider SVN地址：http://www.apache.org/dyn/closer.cgi/incubator/slider

/查看和下载目前的所有版本，另外可以从Git hub上查看所有版本及最新版本代码：

https://github.com/apache/incubator-slider。

### 1.3 搭建Slider源代码阅读环境

本文从github上直接获取最新代码，也可以直接从Apache Slider SVN上获取源码压缩包。构建Maven工程，步骤如下：

1）从github上获取最新的代码

git clone -b develop https://github.com/apache/incubator-slider.git

将代码下载到工作目录下

2）导入Maven工程

在Intellij IDEA中，依次选择："File" —> "Import Project" —> 打开项目目录pom.xml，单击Next按钮，直到选择JDK，然后单击Finish按钮完成Maven项目的导入。之后，Intellij Idea会自动通过网络从Maven库中下载依赖的第三方库。注意，如果主机不能联网，需要将依赖的库添加到Maven本地库。

将Slider源代码导入Maven项目后，会生成5个工程，这些都是通过Maven构建处理的，每个工程都是一个代码模块，且彼此相对独立，可以单独编译。可以在某个工程下的src目录下查看对应的源代码。

3）代码阅读技巧

下面介绍Intellij源码阅读的一些技巧，例如如何查看一个基类有哪些派生类，一个方法被其他哪些方法调用等。Intellij IDEA有多种代码使用模式，本处选择Eclipe模式（File->Setting->keyMap->选择eclipse）

1.查看一个基类或接口的派生类或实现类

在Intellij中，选中某个基类或者接口名称，右击，在弹出的快捷菜单中选择"GO TO" —>

"Implementations"，可在新窗口中看到对应的所有派生类或实现类

2.查看方法的调用关系

选中某个方法名称，右击，在弹出的快捷菜单中选择"Find Usages"，会在窗口Usage中看到所有调用该方法的方法及类

3.快速查找类对应的相关信息

选中该类，然后快捷键ctrl+O，会跳转到类定义(快捷键ctrl+H，查找字符串出现的地方)

### 1.4 Slider源代码组织结构

在Slider的tar.gz安装包解压后的目录slider-{VERSION}中包含了Slider全部的管理脚本和JAR包，下面简单对这些文件和目录进行介绍：

* bin: Slider最基本的使用脚本所在目录，当前包含Slider对各类服务的管理，包括创建、启动、停止及更改服务信息等
* conf: Slider配置文件所在的目录，包括slider-client.xml、slider-server.xml、slider-env.sh及log4j.properties等
* lib: 该目录中包含了Slider对外提供的编程动态库、静态库及程序运行的依赖库
* agent/conf: 该目录中包含了Slider-Agent启动时的初始配置，包括agent进程的注册及心跳端口，安装、日志及启动目录等配置项

在Slider的源代码解压后的目录slider-{VERSION}-src中可看到如1-1 图所示的目录结构，其中重要目录有app-package、slider-core和slider-agent等，下面分别介绍这几个目录的作用。



图1-1 Slider源代码目录结构

* slider-core: 该部分给出了Slider Server的核心实现，本质上是YARN ApplicationMaster的实现，包括Slider底层库的实现，其中有状态机、事件库、RPC通信库、Web界面等；SliderAppMaster的实现，应用的创建、维护及管理等；Slider Client ,封装与SliderAppMaster RPC协议交互相关的库，方便用户通过命令行管理应用程序。
* slider-agent: 该部分给出了Slider Agent进程的核心实现，该进程用于启动服务的管理，向SliderAppMaster注册并发送心跳，接收SliderAppMaster发送过来的命令并将执行结果传回SliderAppMaster
* app-packages：该部分包含了Slider所支持服务的安装、管理的脚本及文件，目前支持的服务包括hbase、memcached、solr、storm及kafka等。

### 1.5 Slider初体验

一般而言，要想深入学习一个新的系统时，首先要尝试使用该系统，了解系统对外提供的功能，然后再通过某个功能逐步深入其实现细节。本节将介绍如何在使用Slider，包括搭建Slider环境，Slider向YARN提交应用并介绍应用的基本操作，但是本节仅代表性的介绍Slider的一些基本使用方法，使应用会Slider有一个初步认识，并引导用户逐步进行更全面的学习。

#### 1.5.1搭建Slider环境

本小结介绍分布式环境下的搭建方法，其依赖Hadoop的安装步骤和配置方法不再介绍，需要注意的是，由于不同用户拥有的Linux环境不尽相同（比如已经安装的软件不同、统一软件的版本不同等），Slider部署及使用过程中遇到的问题可能不同，此时需要根据具体的日志提示解决问题。

1. 社区slider的部署步骤:

步骤1:解压Slider安装包，将slider的安装包，解压到安装目录中：

$tar –zxvf slider-{VERSION}-all.tar.gz

步骤2:修改Slider配置文件

1. 设置环境变量，在${SLIDER\_HOME}/conf/slider-env.sh中，添加JAVA安装目录和Hadoop的配置目录，命令如下：

export JAVA\_HOME=/usr/jdk64/jdk1.7.0\_67

export HADOOP\_CONF\_DIR=/etc/hadoop/conf

修改conf下面的slider-client.xml和slider-server.xml两个文件，在<configuration>与</configuration>之间添加的内容见下面的介绍

1. 在${SLIDER\_HOME}/conf下，将slider-client.xml中添加以下内容：

<property>

<name>slider.base.path</name>

<value>/slider/.slider/</value>

</property>

<property>

<name>slider.registry.path</name>

<value>/registry</value>

</property>

<property>

<name>slider.zookeeper.quorum</name>

<value>{ZK\_HOSTS}:{ZK\_PORT}</value>

</property>

步骤3:验证Slider的安装

$slider version

若输出内容如下：

INFO client.SliderClient-Slider Core-0.80.0-bc1.3.0 Built against commiton Java 1.7.0\_67 by root

INFO client.SliderClient - Compiled against Hadoop 2.6.0-bc1.3.0

INFO client.SliderClient - Hadoop runtime version Unknown with source checksum ate

INFO util.ExitUtil - Exiting with status 0

则说明安装成功

#### 1.5.2 Slider Shell介绍

在1.4节中提到，bin目录下是最基础的集群管理集群，用户可以通过该脚本完成各种功能，如应用安装包的上传、应用的创建、应用的管理等，更加详细的脚本使用说明，可参看附录B。

作为入门，本节介绍的是bin目录下的Slider脚本的使用方法，该脚本的使用的方法为：

slider COMMAND [options]

其中COMMAND是具体的某个命令，常用的命令有应用管理命令、应用查询命令及应用的更新等，

1. 应用管理命令，用法基本一样，均为：

bin/slider command [genericOptions] [commandOptions]

其中command可以是create、start、stop及destroy等。genericOptions是一些通用选项，commandOptions是命令附加的命令选项，看下面的两个例子：

1）在YARN中创建一个HBase应用，命令格式如下：

bin/slider create demo-hbase --template appConfig.json  --resources resources.json

创建应用demo-hbase，其中—template和—resources为命令的genericOptions，demo-hbase为commandOptions。下面是options的介绍：

|  |  |
| --- | --- |
| <application> | 应用的命名 |
| --manager <manager> | 用于绑定YARN RM（一般是Host:port），选填，slider-client.xml中已经配置 |
| --provider <provider> | 指定集群应用的provider |
| --debug | 采用Debug模式 |
| --rm <rmAddress> | 与--manager有点类似 |
| --zkhosts <zkhosts> | Zookeeper hosts，以逗号隔开（选填）,slider-client.xml |
| --resopt <resOtionTuples> | 资源选项，--resopt <name><value>，一般在resource.json中指定 |
| --component <componetTuples> | 指定component的数目，一般在resoure.json中指定，对其覆盖 |
| --package <packageUri> | Slider package的URI，选填 |
| --fileSystem <filesystemBiding> | 绑定文件系统，选填 |
| --S <sysprops> | Jvm启动后，设置的系统性质 |
| --template <template> | 应用配置的模板应用 |
| --out <output File> | 应用的report的输出目录 |
| --apphome <appHomeDir> | 预安装应用的目录 |
| --queue <queue> | 提交应用的队列 |
| --wait <waittime> | Action运行结束的时间 |
| --zkpath <appZKPath> | 应用的zookeeper路径 |
| --image <image> | 包括application的tar或者.tar.gz目录 |
| --resources <resources> | 实例资源的定义环境 |

2）启动一个Stopped应用，命令格式如下：

bin/slider start demo-hbase --wait 10

demo-hbase位commandOptions，--wait为genericOptions。

|  |  |
| --- | --- |
| --manager <manager> | 用于绑定YARN RM（一般是Host:port），选填，slider-client.xml中已经配置 |
| --basepath <basePath> | 应用的路径 |
| --wait <waittime> | 等待应用处于RUNNING状态的时间 |
| --rm <rmAddress> | 与--manager有点类似 |

2．应用查询命令，查询slider的创建的应用信息，其中命令包括list、status、exists、lookup及registry/resolve等，下面介绍两个例子：

1）查询已经创建的应用，命令如下：

bin/slider list [genericOptions]

genericOptions详述如下所示：

|  |  |
| --- | --- |
| --live | 仅列举正在运行的应用 |
| --state [STATE] | 列举处于某特定YARN State状态的应用 |
| --containers | 列举应用的同时列举出containers |
| --components [CMPLIST] | 根据Component的名字查询containers |

2）查看创建应用的registry信息，包括对外提供的端口、container日志、应用安装目录等，命令如下：

slider registry [genericOptions]

其中genericOptions详述如下表所示：

|  |  |
| --- | --- |
| --manager <manager> | 用于绑定YARN RM（一般是host:port），选填，slider-client.xml中已经配置 |
| --list | 列举service |
| --listfiles <listFiles> | 列举指定files |
| --getexp <getExport> | 获取export，对外暴露的Endpoint |
| --getconf <getConf> | 获取服务的配置 |
| --user <user> | 获取指定用户的服务 |
| --listconf | 列举可提供的配置 |
| --listexp | 列举可提供的export |
| --name <name> | 查询指定的app name的信息 |

3.应用的更新，命令格式如下：

slider update <application> <genericOptions>

其中genericOptions详述如下表所示:

|  |  |
| --- | --- |
| --manager <manager> | 用于绑定YARN RM（一般是Host:port），选填，slider-client.xml中已经配置 |
| --compopt <compOptTriples> | 修改组件，以三元组的形式<component> <name> <option> |
| --resopt <resOptionTuples> | 修改resource的选项，<name> <value> |
| --option <optionTuples> | 修改参数值,<name> <value> |
| --rescompopt <resCompOptTriples> | 资源选项，--resopt <name><value>，一般在resource.json中指定 |
| --component <componetTuples> | 修改指定component的数目 |
| --appconf <confdir> | 指定hdfs上的集群的配置目录 |
| --template | 修改应用的模板 |
| --queue | 修改应用启动的队列 |
| --template <template> | 应用配置的模板应用 |
| --apphome <appHomeDir> | 预安装应用的目录 |
| --queue <queue> | 提交应用的队列 |
| --appdef | 指定应用的安装文件 |
| --metainfo | 修改app的metainfo |
| --resources | 修改resources文件中指定的组件 |

其他更多命令，可自行查阅Slider官方设计文档

### 1.6 编译及调试Slider源代码

读者在阅读源代码过程中，可能需要修改部分源代码或者使用调试工具以便进行跟踪某些变量值变化过程，此时要用到Slider源代码编译和调试方法。本节介绍直接在Linux平台下编译和调试Slider代码。

#### 1.6.1 编译Slider源代码

在Linux环境下编译源代码之前，需要进行以下准备工作：

* 确保安装的Maven版本在3.0.2之上
* Protocol Buffers安装版本在2.5.0
* 第一次编译代码，需确认可以连接互联网，Maven要从代码库中下载依赖包

Maven编译命令如下表所示：

|  |  |
| --- | --- |
| mvn clean | 清理编译结果 |
| mvn compile | 编译源代码 |
| mvn test | 运行测试程序 |
| mvn package | 创建JAR包 |
| mvn install | 将JAR包放到M2缓存中 |
| mvn deploy | 将JAR包放到Maven仓库中 |

每个模块编译后生成的JAR包放到了与源代码目录平级的target目录中，如果修改了某个模块的代码，可编译后，将对应的JAR包覆盖到${SLIDER\_HOME}/lib目录中对应的JAR包中。如果仅编译Slider的一个子模块，需要将该模块依赖的JAR包作为它的第三方引入，简单的实现方式是 在Slider的源代码目录中输入以下命令编译所有的源代码：

mvn install –DskipTests

然后进入子模块目录，编译生成对应的JAR包。

在app-packages子项目中，包含了hbase、accumulo及storm等多个应用的安装脚本、配置等相关文件，在使用前需要使用maven进行打包，以hbase为例，命令格式如下：

mvn clean package - Phbase-app-package -Dhbase.version=<hbase version>

-Dpkg.version=<app package version>

-Dpkg.name=<file name of app tarball>

-Dpkg.src=<folder location where the pkg is available>

例子如下：

mvn clean package -Phbase-app-package -Dhbase.version=0.98.5-hadoop2

-Dpkg.version=0.98.5-hadoop2

-Dpkg.name=hbase-0.98.5-hadoop2.tar.gz

-Dpkg.src=/Users/user1/Downloads/0.98.5-hadoop2

其余应用的应用可以在该应用项目的目录下的README.md文件中找到相对应的编译命令

#### 1.6.2打印Slider调试日志

Slider使用Apache log4j作为基本日志库，该日志库将日志分为5个级别，分别为DEBUG、INFO、WARN、ERROR和FATAL。这5个级别是有顺序的，即DEBUG < INFO <WARN < ERROR < FATAL，分别用来指定日志信息的重要程度。日志输出规则为之输出不低于设定级别的日志信息，比如级别设定为INFO，则INFO、WARN、ERROR和FATAL级别的日志信息都会输出，但是级别比INFO低的DEBUG则不会输出。

在Slider源代码中，大部分Java和Python文件中存在调试日志（Debug级别），但默认情况下，日志级别是INFO，为了查看更详细的运行日志，可采用修改log4j.properties文件来打开Debug日志，在文件中增加以下配置选项：

# for debugging Slider

log4j.logger.org.apache.slider=DEBUG

log4j.logger.org.apache.slider=DEBUG

# uncomment to debug service lifecycle issues

log4j.logger.org.apache.hadoop.yarn.service.launcher=DEBUG

log4j.logger.org.apache.hadoop.yarn.service=DEBUG

# uncomment for YARN operations

log4j.logger.org.apache.hadoop.yarn.client=DEBUG

# uncomment this to debug security problems

log4j.logger.org.apache.hadoop.security=DEBUG

小技巧是为了专门调试某个Java文件，需要把该文件的相关日志输入到一个单独文件中，可在log4j.properties中添加

#设置文件路径

log4j.appender.stdout.File=${slider.log.dir}/xxx.log

这个配置选项会把相对应的DEBUG日志写到日志目录下的xxx.log文件中。在阅读源代码的过程中，为了跟踪某个变量值的变化，开发者可能需要自己添加DEBUG日志。在Slider源代码中，大部分类会定义一个日志打印对象，通过该对象可打印各个级别的日志。比如在SliderClient中用以下代码定义对象LOG:

publi static final Log LOG = LogFactory.getLog(SliderClient.class);

用户可使用LOG对象打印调试日后，比如，可在SliderClient的exec方法中添加以下:

LOG.debug("Start to execute Command ....");

然后重新编译Slider源代码，并将org.apache.slider.client.SliderClient的调试级别修改为DEBUG，重新启动Hadoop后便可以看到该调试信息。

### 1.7 小结

搭建一个高效的源代码学习环境是深入学习Slider的良好开端，本章主要内容正是帮助用户及开发者搭建一个这样的环境。一个高效的Slider学习环境至少应该包括源代码阅读环境，Slider使用环境和源代码编译调试环境，而本章正是围绕着三个环境的搭建方法组织的。

本章介绍了Linux环境下搭建Slider源代码阅读环境的方法，在此基础上，进一步介绍了Slider的基本使用方法，主要涉及Slider Shell的使用。最后介绍Slider源代码的编译和调试方法，其中调试方法仅介绍了打印调试日志，其他方法有待进一步完善。