Apache Slider简介和使用

# 1、简介

Yarn的产生是为了解决Hadoop中MapReduce框架的性能瓶颈，根据思想是将JobTrack

-er和TaskTracker两个主要的功能分离成单独的组件，分别进行资源的管理和任务的调度/监控，资源管理器负责所有应用程序计算资源的分配，每一个应用的ApplicationMaster负责相应的调度和协调。同时Yarn还提供了pluggable 结构来支持其他计算框架的引入，但是需要定向开发各计算框架的AppMaster。随着Yarn的成熟，产生了Apache Tez及Slider等类似的引擎，这列引擎可以快速的集成第三方计算和服务（processing and services）。

Tez是开源的支持DAG作业的计算框架，Tez在底层提供DAG编程接口，用户编写程序直接采用这些接口进行程序设计。相类似的框架是Oozie，但是其实比较高层次，只提供一种多类型作业（MR程序、Hive和Pig）依赖关系表达式。Tez基于Yarn的，与原有的MR共存。Apache Slider源自于Hoya（将HBase部署在Yarn上的项目），通过Slider可以在Yarn环境中运行其他的应用，这些分布式的应用可以是非Yarn-aware的，通过这种方式可以将很多的服务运行在Yarn中（不需要进行任何修改）。Slider允许这些应用使用Hadoop的数据和计算资源（同时安全，管理及Hadoop的集群的操作能力）。在Apache Slider中部署服务，和运行一个普通分布式应用程序相同，应用程序部署起来，就意味着服务部署成功。

基本术语，如下：

* Application，将应用程序打包后，由Slider部署的应用，包括一个至多个分布式组件。
* Instance，Yarn Cluster根据应用配置文件启动的应用实例，实例的状态可以使运行状态或者停止状态，当停止时配置信息及实例特定数据保存在HDFS中。Live instance可以被停止，停止时状态保存到state目录中。stopped instance可以被重新启动，启动的机器会尽量在原机器或者near 机器。Stopped instance可以被销毁。通过list命令，可以查看正在运行的实例。
* Instance directory，HDFS中创建的目录，存储了用户定义、APP默认的配置及Slider动态生成的任何数据。
* Users，创建App实例
* Components，一个实例可以有多个Components，例如HBase中HMaster,HRegion-

Server等都是Components，具体的类型由Slider app自己定义，在instance创建时App指定。用户可以扩展（flex）Application instance，动态的增加和删除components。如果instance正在运行，这个改变立即生效，如果不在运行当instance启动时会生效。

# 2、Apache Slider Apps部署

Apache Slider自带HBase、Storm和Accumulo的服务实现，如果其他应用也采用Slider在Yarn中部署，需要安装Slider规范进行编写一系列的配置和脚本，然后定义打包文件，包括下几个文件：

1）创建meteinfo.xml文件，内容包括：Apps的名称、版本、外部端口、包含的组件及实例数限制(min/max)、组件的启停顺序以及启停的python脚本、适用的OS 以及 Apps的压缩文件路径。详细规范可参考：

<http://slider.incubator.apache.org/docs/slider_specs/application_definition.html>

2）创建Apps的压缩文件。把Apps发布的 .tar.gz 下载下来，删除掉src/ 源文件目录再打包就可以了。也可以重新编译成package。

3）创建resources.json文件，内容包括：该app中各个组件的资源需求，包括<componentsName, memory, vcores, role\_priority, #instances>。还可以配置app在某段时间内失败N次的错误检测策略。详细规范可以参考：

<http://slider.incubator.apache.org/docs/slider_specs/resource_specification.html>

4）创建appConfig.json文件，以下字段是必须要定义的：application.def：Apps打包压缩文件在HDFS上的路径；java\_home：JDK路径；package\_list：Apps原始文件的路径；config\_types：配置文件名称；site.global.app\_user：运行应用的user；site.global.app\_root：Apps压缩文件解压后的路径。另外还可以定义 site.global.\* 的应用相关的配置参数。详细规范可以参考：

<http://slider.incubator.apache.org/docs/slider_specs/application_instance_configuration.html>

5）实现metainfo.xml定义的Apps相关各组件操作的脚本。详细规范可以参考：

<http://slider.incubator.apache.org/docs/slider_specs/writing_app_command_scripts>

下面介绍Slider及HBase Apps的安装和使用：

## 2.1 Slider的编译和安装

在Hadoop 2.6之前的版本都没有针对long-live service的程序进行考虑，但是在Hadoop 2.6 Yarn中增加了几点重要的feature，包括服务的注册，NM和RM的rolling update等特性。Slider中的AppMaster根据具体的Hadoop版本进行开发，因此Slider 0.30-0.50各个版本对Long-live service的管理并不好，Slider 0.60后才逐渐的改进，这里安装Slider 0.60：

下载slider源码：

$wget http://apache.fayea.com/incubator/slider/0.60.0-incubating/src/apache-slider-0.60.0-incubating-source

-release.tar.gz

解压slider

$tar -zxvf apache-slider-0.60.0-incubating-source-release.tar.gz

编译：

$cd apache-slider-0.60.0-incubating

$mvn clean package

注意：在pom.xml中hadoop的版本号是2.6.0-snapshot，改成2.6.0

编译后，在../slider-assembly/target/目录中，生成slider-0.60.0-incubating-all.zip安装文件，解压安装文件，并配置slider-client.xml及slient-env.sh文件

$unzip slider-0.60.0-incubating-all.zip

$cd slider-0.60-incubating/conf

$vim slider-client.xml

slider-client.xml的配置如下：

1. <property>
2. <name>yarn.resourcemanager.address</name>
3. <value>ambarimaster:8032</value>
4. </property>
5. <property>
6. <name>yarn.resourcemanager.scheduler.address</name>
7. <value>ambarimaster:8030</value>
8. </property>
9. <property>
10. <name>fs.defaultFS</name>
11. <value>hdfs://ambarimaster:8020</value>
12. </property>
13. <property>
14. <name>hadoop.registry.zk.quorum</name>
15. <value>ambarimaster:2181</value>
16. </property>

slider-env.sh的配置如下：

export JAVA\_HOME=${JAVA\_HOME}

export HADOOP\_CONF\_DIR=${HADOOP\_CONF\_DIR}

运行slider命令：

$bin/slider -version

结果如下：

1. 2015-01-10 20:42:26,846 [main] INFO client.SliderClient - Slider Core-0.60.0-incubating Built against commit# ${buildNumber} on Java 1.7.0\_60 by yarn
2. 2015-01-10 20:42:26,850 [main] INFO client.SliderClient - Compiled against Hadoop 2.6.0
3. 2015-01-10 20:42:26,861 [main] INFO client.SliderClient - Hadoop runtime version branch-2.6.0 with source checksum 18e43357c8f927c0695f1e9522859d6a and build date 2014-11-13T22:35Z
4. 2015-01-10 20:42:26,864 [main] INFO util.ExitUtil - Exiting with status 0

则安装成功

## 2.2 HBase的App的编译和安装

可以根据需要，选择HBase的安装版本，但是在slider-xxx的pom.xml中有hbase的版本，默认是hbase-0.98.4，对应的Apache的hbase-0.98.4-hadoop2-bin.tar.gz。

1）HBase安装文件的打包

在app-package/hbase的metainfo.xml和pom.xml中可以进行配置，不再介绍，本处使用默认的hbase的版本：

$cd ../apache-slider-0.60.0-incubating/app-package/hbase

$ mvn clean package -Phbase-app-package -Dpkg.version=0.98.4-hadoop2

-Dpkg.name=hbase-0.98.4-hadoop2.tar.gz

-Dpkg.src=/Users/user1/Downloads/2.2

2）安装应用包

在app-package/hbase/target中可以找到编译好的安装文件slider-hbase-app-package

-0.60.0-incubating.zip。使用命令将zip文件安装到hdfs中（安装后，slider可以自动找到该安装文件）

$bin/slider install-package --name HBASE --package ../slider-hbase-app-package-0.60.0-incubating.zip

默认的安装路径为/user/yarn/.slider/package/中，通过hdfs命令查看

$hdfs dfs -ls /user/yarn/.slider/package/HBASE

可以看到slider-hbase-app-package-0.60.0-incubating.zip

3）配置服务

根据具体需要，定义appConfig.json和resources.json。

appConfig.json用于配置HBase的基本参数，这部分参数在hbase-site.xml中已经进行了配置，例如hbase.rootdir,zookeeper.quorum等。必须配置的参数为application.def，其配置了HBase App安装包的所在位置

application.def": ".slider/package/HBASE/slider-hbase-app-package-0.60.0-incubating.zip

resource.json中定义了HBase需要启动的实例参数，包括HMaster,RegionServer等的组件个数，及各组件需要的memory（以后会扩展到cpu等资源）。下面是Region\_server组件的配置：

"HBASE\_REGIONSERVER": {

"yarn.role.priority": "2",

"yarn.component.instances": "3",

"yarn.memory": "1500"

}

4）创建HBase的集群实例

bin/slider create hbasetest --template ../sliderapp/appConfig.json --resources ../sliderapp/resources.json

通过该命令，会在Yarn中启动一个SliderAppMaster，根据需要从RM中获取资源，并启动Container，并在Container中启动HBase服务的各个组件。通过http://rm-host:8088，端口可以看到服务的运行状况。通过下面的命令可以看到已经启动的服务：

$slider list

5）Slider其他命令

这些服务启动后，可以进行停止，启动（重启）及销毁等操作

停止：

$slider stop hbasetest

启动：

$slider start hbasetest

销毁：

$slider destroy hbasetest

查看状态：

$slider status hbasetest

还有其他命令如，kill-container(停掉应用的某个container)，lookup（查找app）等命令，这些命令可以通过slider help进行查看

# 2. Slider操作指南

Slider命令格式如下：

$slider <ACTION> [<name>] [<OPTIONS>]

## 2.1 Common Commnad-line Options

--conf configuration.xml

指定Slider client的配置文件，从文件中获取使用的文件系统，ZK实例及其他性质，通过配置文件指定就不需要在命令行中指定。

注：配置项并不会改变Application中的配置，仅仅配置客户端。

-D name=value

定义Hadoop的配置项，覆盖Client 配置文件中指定的参数

-m --mananger url

YARN ResourceManager的URL，例如：

--fs fileSystem-uri

操作指定的文件系统URI，例如hdfs://host:port

## 2.2 Actions

### 2.2.1 build 及create

命令：

slider build [create] <name>

build命令根据指定的实例名称，创建instance，根据options进行配置，需要没有启动，使用Start命令启动。create根据指定的实例名称，创建并启动App 实例。

参数：

--package <uri-to-package>

应用安装包所在的地址，在HDFS中。如果不指定，那么从appconfig.xml读取默认路径。

--option <name> <value>，设置应用的参数，

例如设置,目标文件系统的blockSize为128M，--option site.dfs.blocksize 128m

--appconf dfspath

指定application应用模板，该参数指定配置目录，该目录并需对Yarn Cluster可见。该目录仅有一个，在create或者build时读取目录的内容

例如： --appconf hdfs://namenode/users/slider/conf/hbase-template

--appconf file://users/accumulo/conf/template

--apphome localpath

application的home目录，Slider App创建时从该目录读取二进制安装文件。

例如 -apphome /usr/hadoop.hbase

--template <filename>

app实例配置的模板文件，覆盖build-in options，但是会被--option及--compopt参数覆盖

--resources <filename>

resource 配置的文件名，覆盖build-in 选项，但是被--respot，--rescompopt,--components参数覆盖。

--image path

指定应用的安装文件，放置在hdfs中。

例如： --image hdfs://namenode/shared/binaries/hbase-0.96.tar.gz

--component <name> <count>

实例中的component的数目，例如：--component worker 16。与resource.json配置文件中的component.instances，起到的作用相同。

--compopt <component> <option> <value>

启动component启动的具体参数，例如--compopt master env.TIMEOUT 1000。配置的值保存到app\_conf.json中。但是如果配置Yarn resource分配，使用--rc参数

--rescompopt <component> <option> <value>

配置component在yarn中的需求，例如--rco worker master yarn.memory 2048

--zkhosts host1:prots,[host2:port2,host3:port3]

配置Zookeeper quorum。例如--zk1:2181,host2:2181,host3:2181。如果没有配置，则读取slider.zookeeper.quorum的值。

--zkpath <zookeeper-path>

创建App的zookeeper集群路径

--queue <queue name>

指定创建App的Yarn队列，默认为Default

--wait <milliseconds>

如果提供，Yarn Application运行后要wait的时间。即使Yarn app被启动，instance 启动也要一定延迟

--out <filename>

启动日志信息

create及build举例：

指定template及resources

$slider create hbase1 --template /usr/work/hbase/appConfig.json --resources /usr/work/hbase/resources.json

指定template，resources及queue

$slider create hbase1 --template /usr/work/hbase/appConfig.json --resources /usr/work/hbase/resources.json --queue default

### 2.2.2 destroy

销毁已经stopped的应用实例，会删除实例的数据，例如：

slider destroy instance1

### 2.2.3 exist

检测slider application实例是否产生，如果设置--live 标识，实例不运行，则返回错误code。不设置，则从文件系统中找到该application instances的定义文件，则返回succeed。例如：

slider exists instance4

返回值有以下几种：

0 : application instance is running

-1 : application instance exists but is not running

69 : application instance is unknown

设置Live flag，则服务处于NEW,NEW\_SAVING,SUBMITTED,ACCEPTED及RUNNING状态，则为Live。处于FINISHED/FAILED及KILLED状态则返回not live。举例：

slider exists instance4 --live

如果设置--state ，则检测Yarn Application 状态，Yarn state包括：

NEW: Application which was just created.

NEW\_SAVING: Application which is being saved.

SUBMITTED: Application which has been submitted.

ACCEPTED: Application has been accepted by the scheduler

RUNNING: Application which is currently running.

FINISHED: Application which finished successfully.

FAILED: Application which failed.

KILLED: Application which was terminated by a user or admin.

例如：

slider exists intance4 --state ACCEPTED

返回的Codes包括：

0 : application instance is running

-1 : application instance exists but is not in the desired state

69 : application instance is unknown

### 2.2.4 flex

Flex命令用于调整应用实例中的workers数目，如果大于原值，则需要创建新的components。如果少于，则销毁components实例。如果改变已经被提交，则返回0。变动并不是立即生效，根据Yarn的资源可用量来决定，如果返回-1，表示app没有运行。例如：

slider flex instance1 --component worker 8 --filesystem hdfs://host:port

slider flex instance1 --component master 2 --filesystem hdfs://host:prot

### 2.2.5 install-package

命令用于安装应用package，默认路径为~./slider/package。可以通过appConfig.json文件中-template参数来配置。

--name <name of package>，指定package的名称，可以与meatainfo.xml中提供的名称相同，一定要与appConfig.json文件中配置的一样。

--package <package file>：package所在的本地位置

--replacepkg，可选择，是否覆盖已安装的package。

例如：

slider install-package --name HBASE --package /usr/work/package/hbase/slider-hbase-app-package-0.98.4-hadoop2.zip

### 2.2.6 List [name][--live][--status status]

列举用户可见的Slider application，包括文件系统中的实例。没有不指定instance名称，列举所有符合条件的实例。

--live，指示live instances，可以使RUNNIG或者等待执行状态的Instance

--state <state>，RM中处于某状态的Cluster  
不指定，列举所有的Application，运行或者不运行。

指定instance name，如果不存在，则返回unkown exit code。instance存在但是和指定的状态不一样，则返回-1。使用举例：

lider list

slider list instance1

slider list --live

slider list instance1 --live

slider list --state FINISHED

slider list --state KILLED

slider list --state FAILED

slider list instance1 --state FAILED

注：列举completed状态的instances命令，可能会返回running状态的instance。例如，如果一个实例started，然后stopped后，重新创建一个新的instance，以下两个操作都会成功：

slider list instance1 --live

slider list --state FINISHED

出现这种情况的原因是List命令通过Yarn app来获取想要的参数。

### 2.2.7 registry （--list |--listconf|--gentconf <conf>） [--name <name>] [--servicetype <servicetype>] [--out <filename> ][--verbose]

列举注册的应用。这个List和Slider list命令不同，不使用Yarn application list，和Zookeeper通信，查询通过service registry注册的应用。

--name <name> ，registry entry的名称

--user <user>，部署或者服务的拥有者，默认是当前用户

--servicetype <servicetype>，选择不同的service type，默认是org-apache-slider。

--verbose，属于更详细的信息

--internal，列举部署应用的internal 参数

返回值有两类：

没有匹配的服务或者没有配置，则返回EXIT\_NOT\_FOUD

命令：

slider registry --list [--servicetype <servicetype>] [--name <name>] [--verbose] [--user <user>] [--out <filename>]

列举指定type及条件的服务，--out指定输入文件，空文件表示没找到

slider registry --listconf [--name <name>] [--internal] [--servicetype <servicetype>] [--user <user>] [--out <filename>]

列举服务的配置

slider registry --getconf <configuration> [--format (xml|json|properties)] [--servicetype <servicetype>] [--name <name>] [--dest <path>] [--internal] [--user <user>]

获取一定格式的指定名称的configuration，默认是XML。--dest <path>，保存配置文件的路径，--formant（xml|json|properties）定义输出格式。如果定义dest，则文件被保存到该文件下，例如：

slider registry --getconf hbase-site.xml --name hbase1 --dest confdir

如果confir存在，则下载hbase-site配置到confdir/hbase-site.xml。如果--dest配置的是文件，则将hbase-site.xml中的内容下载到configfile.xml中。

### 2.2.8 resolve --path <path> [--out <filename> ][--list][--destdir <dir>]

这个命令resolve注册路径下的service record或者指定路径下的service。显示在控制端或者保存到文件系统中。举例：

slider resolve --path /users/hbase/services/org-apache-hbase/instance1

resolve该目录下的服务record。

slider resolve --path /users/hbase/services/org-apache-hbase/instance1 --out hbase.json

保存到hbase.json中。

路径不再registry或者路径下没有service record，则返回EXIT\_NOT\_FOUD.

--list，用于列举目录下所有的entries和services，例如：

slider resolve --path /users/hbase/services/org-apache-hbase/ --list

列举目录下的所有services。--destdir 将结果保存到目录下。当前用户的base path用~前缀来表示。

### 2.2.9 start <name> [--wait milliseconds][--out <filename>]

启动一个stopped状态的应用实例，恢复原状态。--out 配置输出文件，保存Yarn application report。

slider start instance2

slider start instance1 --wait 60000

slider start instance1 --out appreport.json

### 2.2.10 status <name> [--out <filename>]

返回应用状态的json格式文件

slider status instance1 --manager host:port

slider status instance2 --manager host:port --out status.json

### 2.2.11 stop <name> [--force][--wait time][--message text]

--wait参数，定义application instance停止的时间

--force，直接停止应用instance

--message，request中中可选择的text message，用于Yarn RM UI诊断。

例如：

slider stop instance1 --wait 30

slider stop instance2 --force --message "maintenance session"

### 2.2.12 version

输出Slider 编译信息，包括使用的Hadoop版本

### 2.2.13 测试命令

slider kill-container <name> --id container-id

kill掉属于该应用的container，用于测试对failures的应变能力。可以从application instance status中获取containerid

slider am-suicide <name> [--exitcode code] [--message message][--wait time]

用于测试Slider app master重启，触发重建操作，而且不会关掉进程。如果应用不设置从其上限，那么Yarn会重启failed application。

slider am-suicide --exitcode 1 --wait 5000 -message "test"

# 3.Slider Router

在Yarn中可以部署Long-lived的服务实例，例如Apache Tomcat Web Servers或者HBase集群。客户端需要访问服务instance，传统的方式是通过DNS或者配置文件中注册的IP。但是Yarn中部署应用的主机名或者网络端口不能不对外暴露，因为客户端对这些应用的访问有难度。

用户如果想访问这些应用，首先要根据启动的应用进程获取端口，然后根据端口再访问服务。这种方式不符合应用托管的特性，比较理想的情况是将应用托管到YARN中，用户可以根据自己的配置访问这些服务，而不是人为的去获取，而且一些信息不容易获取，例如Web UI，REST URI等。

Yarn支持Application Master向Web URI及IPC address的基本注册。但是目前还不支持，AppMaster启动的Task的注册REST URLs,ZK path及其他Endpoint；而且当Yarn app重启启动时Instance ID在不断的变化，静态的配置Named Service不可能解决binding问题，而且静态配置会导致指向不存在的service instance的问题。

Slider Router简化用户访问托管到YARN中服务的流程，总体框架图如下所示：



为了实现slider 创建应用的托管，需要实现以上几个部分，下面对每个部分进行详细介绍。

## 3.1 Slider Router的配置和启用

1）Slider Router 配置

在slider-server.xml中增加slider -router配置项：

<property>

<name>slider.router.address</name>

<value>{host}:{port}</value>

<description>router IPC address </description>

</property>

<property>

<name>slider.router.webapp.address</name>

<value>{host}:{port} </value>

</property>

2) Slider Router启动及停止脚本

$start-router.sh

$stop-router.sh

## 3.2 Slider Rest Service

REST Server的启动端口由slider.router.webapp.address配置，REST API如下表所示：

|  |  |  |
| --- | --- | --- |
| REST API名称 | 路径 | 描述 |
| CREATE APP | /slider/v1/apps/template | 创建一个App应用，例如HBase |
| FLEX APP | /slider/v1/apps/{appName}/flex | 对创建的App进行扩容或者减少节点 |
| BUILD CLIENT | /slider/v1/apps/{appName}/buildClient | 创建hbase client端，并上传到HDFS |
| UPGRADE APP | /slider/v1/apps/{appName}/upgrade | 对已创建的服务进行升级，包括应用的版本等 |
| UPDATE APP | /slider/v1/apps/{appName}/update | 修改启动应用的配置 |
| GET CONFIG | /slider/v1/apps/{appName}/getConfig | 获取app对外提供的配置文件 |
| GET EXPORT | /slider/v1/apps/{appName}/getExport | 获取app对外提供的export |
| GET STATICS | /slider/v1/app/{user}/getStatics | 获取某个用户的app信息 |
| GET STATUS | /slider/v1/apps/{appName}/getStatus | 获取某个启动的App的Status，运行状况信息 |
| START APP | /slider/v1/apps/{appName}/start | 启动STOPPED的app |
| STOP APP | /slider/v1/apps/{appName}/stop | 停止App |
| DESTROY APP | /slider/v1/apps/{appName}/delete | 销毁不需要的服务 |

### 3.2.1 Create APP

|  |  |  |
| --- | --- | --- |
| 协议：HTTP | |  |
| 传输路径 | | |
| 请求 | 报文 | POST: /slider/v1/apps/template  HEADER:  BODY:  {  "appName":" ",  "appType":"",  "accessId":"",  "accessKey":"",  "bdocQueueName":"",  "proxyUser":"",  "updateConfig":{},  "componentOpts":{}  } |
| 说明 |  |
| 响应 | 报文 |  |

### 3.2.2 APP FLEX

|  |  |  |
| --- | --- | --- |
| 协议：HTTP | |  |
| 传输路径 | | |
| 请求 | 报文 | PUT: /slider/v1/apps/flex  HEADER:  BODY:  {  "accessId":"",  "accessKey":"",  "proxyUser":"",  "component":"",  "count":""  } |
| 说明 |  |
| 响应 | 报文 |  |

### 3.2.3 App BUILD CLIENT

|  |  |  |
| --- | --- | --- |
| 协议：HTTP | |  |
| 传输路径 | | |
| 请求 | 报文 | PUT: /slider/v1/apps/{appName}/buildClient  HEADER:  BODY:  {  "accessId":"",  "accessKey":"",  "proxyUser":"",  "hdfsdir":""  } |
| 说明 |  |
| 响应 | 报文 |  |

### 3.2.4 APP UPDATE

|  |  |  |
| --- | --- | --- |
| 协议：HTTP | |  |
| 传输路径 | | |
| 请求 | 报文 | PUT: /slider/v1/apps/{appName}/update  HEADER:  BODY:  {  "accessId":"",  "accessKey":"",  "proxyUser":"",  "updateConfig":{},  "componentOpts":{}  } |
| 说明 |  |
| 响应 | 报文 |  |

### 3.2.5 Get Config

|  |  |  |
| --- | --- | --- |
| 协议：HTTP | |  |
| 传输路径 | | |
| 请求 | 报文 | PUT: /slider/v1/apps/{appName}/getConfig  HEADER:  BODY:  {  "accessId":"",  "accessKey":"",  "proxyUser":"",  "configName":""  } |
| 说明 |  |
| 响应 | 报文 |  |

### 3.2.6 Get Export

|  |  |  |
| --- | --- | --- |
| 协议：HTTP | |  |
| 传输路径 | | |
| 请求 | 报文 | PUT: /slider/v1/apps/{appName}/getExport  HEADER:  BODY:  {  "accessId":"",  "accessKey":"",  "proxyUser":"",  "exportName":""  } |
| 说明 |  |
| 响应 | 报文 |  |

### 3.2.7 Get Statics

|  |  |  |
| --- | --- | --- |
| 协议：HTTP | |  |
| 传输路径 | | |
| 请求 | 报文 | PUT: /slider/v1/apps/{user}/getStatics  HEADER:  BODY:  {  "accessId":"",  "accessKey":"",  "proxyUser":""  } |
| 说明 |  |
| 响应 | 报文 |  |

### 3.2.8 GET Status

|  |  |  |
| --- | --- | --- |
| 协议：HTTP | |  |
| 传输路径 | | |
| 请求 | 报文 | PUT: /slider/v1/apps/{appName}/getStatus  HEADER:  BODY:  {  "accessId":"",  "accessKey":"",  "proxyUser":""  } |
| 说明 |  |
| 响应 | 报文 |  |

### 3.2.9 APP Start

|  |  |  |
| --- | --- | --- |
| 协议：HTTP | |  |
| 传输路径 | | |
| 请求 | 报文 | PUT: /slider/v1/apps/{appName}/start  HEADER:  BODY:  {  "accessId":"",  "accessKey":"",  "proxyUser":""  } |
| 说明 |  |
| 响应 | 报文 |  |

### 3.2.10 APP Stop

|  |  |  |
| --- | --- | --- |
| 协议：HTTP | |  |
| 传输路径 | | |
| 请求 | 报文 | PUT: /slider/v1/apps/{appName}/stop  HEADER:  BODY:  {  "accessId":"",  "accessKey":"",  "proxyUser":""  } |
| 说明 |  |
| 响应 | 报文 |  |

### 3.2.11 APP DELETE

|  |  |  |
| --- | --- | --- |
| 协议：HTTP | |  |
| 传输路径 | | |
| 请求 | 报文 | PUT: /slider/v1/apps/{appName}/delete  HEADER:  BODY:  {  "accessId":"",  "accessKey":"",  "proxyUser":""  } |
| 说明 |  |
| 响应 | 报文 |  |

## 3.3 Slider RPC Server

## 3.4 应用信息的注册

## 3.5 应用代理设置与使用