EayunCloud公有云业务

定时任务系统分析

变更说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **日期** | **版本** | **变更位置** | **说明** | **作者** |
| 2016.6.12 | 1.0 |  | 编写文档 | 曾波 |
| 2016.6.13 | 1.1 | 4.定时任务升级需求列表 | 增加第4节 | 祝军 |
| 2016.6.17 | 1.2 | 数据库设计  主要业务逻辑 | 增加第5、6节 | 曾波 |
| 2016.6.22 | 1.3 | 接口说明 | 增加第7节 | 曾波 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目录

[1 业务背景和需求 4](#_Toc453840659)

[2 系统现状 4](#_Toc453840660)

[2.1 架构 4](#_Toc453840661)

[2.2 主要功能 5](#_Toc453840662)

[2.3 任务明细 5](#_Toc453840663)

[3 问题与优化 5](#_Toc453840664)

[3.1 问题 5](#_Toc453840665)

[3.2 优化建议 6](#_Toc453840666)

[4 定时任务升级需求列表 6](#_Toc453840667)

[5 数据库设计 7](#_Toc453840668)

# 业务背景和需求

**背景：**因公有云业务需要，目前开发了35个计划任务，用于定时统计分析、数据采集、邮件短信发送等，使用的框架为Quartz，通过配置文件配置触发器和JOB，且定时任务项目独立于eayun-ecsc和eayun-ecmc单独部署，由startup.sh脚本启动。由于没有可视化页面来监控定时任务的执行情况以及对定时任务进行简单操作，增加了维护成本。

**需求：**可视化界面查看定时任务列表，以及任务的执行情况，如：上次执行时间、下次执行之间、执行频率、状态等信息；也可进行手工触发任务执行。

# 系统现状

## 架构

定时任务系统独立与ECSC和ECSC而单独部署，构建eayun-schedule.jar时，将eayun-schedule以及相关的依赖打包，再通过Shell脚本startup.sh启动。定时任务直接与openstack、数据库等进行交互。另外，mailSendTrigger（邮件触发器）和smsSendTrigger（短信触发器） 没有集成在eayun-schedule中，由于特殊性暂不考虑这两个触发器。





## 主要功能

目前定时任务涉及的主要模块有：

* 告警（短信和邮件的发送）
* 工单（扫描无人受理的工单）
* 日志（删除过期日志）
* 流量（流量数据统计）
* 消息（统计消息阅读量和收藏量）
* 对象存储（分周期统计下载流量、请求次数等）
* 监控（云主机性能指标）
* 同步（云资源状态同步）

## 任务明细



# 问题与优化

## 问题

**Q1、定时任务以及相关服务不集中**

Eayun-schedule应该为一个单独的项目，任务类（\*Job.java）既存在于项目eayun-schedule，也存在于其他项目。并且eayun-schedule中存在如/eayun-schedule/src/main/java/com/eayun/schedule/service/CloudFloatIpService.java

的接口，而eayun-virtualization-intf项目中也存在/eayun-virtualization-intf/src/main/java/com/eayun/virtualization/service/CloudFloatIpService.java

**Q2、缺少接口**

由于ECMC和定时任务是相互独立的，如果需要通过ECMC来操作定时任务（如暂停、启动、手工执行等），则需要eayun-schedule提供操作接口，目前eayun-schedule使用java命令启动，无法提供接口服务。

## 优化建议

A1、将Job任务分发至各个项目，由eayun-schedule项目依赖于其他项目，实现分而治之。

A2、在eayun-schedule项目中提供RESTful接口，供ECMC调用。

# 定时任务升级需求列表

经过已上分析，整理定时任务升级需求列表如下：

|  |  |  |
| --- | --- | --- |
| 编号 | 需求项 | 说明 |
|  | 任务列表 | 在Ui中分页显示系统中所有任务 |
|  | 任务手动触发 | 立即执行任务 |
|  | 任务调度停止 | 停止高度后，当下一个调度时间点到达时，任务不会被执行 |
|  | 任务调度恢复 | 从停止状态恢复。 |
|  | 创建任务 | 在ui中新增自动任务 |
|  | 修改任务 | 对以有的任务信息，如cron表达式，进行编辑 |
|  | 删除任务 | 删除系统中不需要的任务 |
|  | 任务调度HA(已有) | 高可用性 |
|  | 任务调度日志 | 任务调度系统级日志，如，调度开始时间，结束时间，结果，调度的任务信息 |
|  | 出错告警(暂不实现) | 任务调度执行出错时，mail通知相关人员 |
|  | 调度数据统计分析 | 如每天调用次数，失败次数，折线图。 |

# 数据库设计

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **数据表名称** | | **schedule\_info** | | | | |
| **说明** | | **定时任务扩展表** | | | | |
| 序列 | 字段 | 类型 | 注释 | 非空 | 外键 | 主键 |
| 1 | trigger\_name | VC(200) | 任务唯一ID | Y | Y | y |
| 2 | create\_time | DT | 创建时间 | Y |  |  |
| 3 | create\_by | VC(32) | 创建人 | Y | Y |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **数据表名称** | | **schedule\_log** | | | | |
| **说明** | | **定时任务运行日志表** | | | | |
| 序列 | 字段 | 类型 | 注释 | 非空 | 外键 | 主键 |
| 1 | id | VC(32) | 主键ID | Y |  | Y |
| 2 | bean\_name | VC(200) | Bean名称 | Y |  |  |
| 3 | method\_name | VC(200) | 方法名 | Y |  |  |
| 4 | description | VC(250) | 任务描述 |  |  |  |
| 5 | trigger\_name | VC(200) | 任务唯一ID | Y | Y |  |
| 6 | exc\_time | DT | 执行时间 | Y |  |  |
| 7 | is\_success | CHAR(1) | 是否成功 | Y |  |  |
| 8 | error\_msg | VC(2000) | 异常\错误信息 |  |  |  |
| 9 | take\_time | INT | 任务耗时（ms） | Y |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **数据表名称** | | **schedule\_statistics** | | | | |
| **说明** | | **定时任务统计表** | | | | |
| 序列 | 字段 | 类型 | 注释 | 非空 | 外键 | 主键 |
| 1 | id | VC(32) | 主键ID | Y |  | Y |
| 2 | trigger\_name | VC(200) | 任务唯一ID | Y | Y |  |
| 3 | statistics\_date | DATE | 统计日期 | Y |  |  |
| 4 | total\_count | INT | 执行总次数 | Y |  |  |
| 5 | suc\_count | INT | 执行成功次数 | Y |  |  |
| 6 | fal\_count | INT | 执行失败次数 | Y |  |  |

建表语句

CREATE TABLE `schedule\_info` (

`trigger\_name` varchar(200) NOT NULL COMMENT '任务ID（关联QRTZ\_TRIGGERS表的TRIGGER\_NAME字段）',

`create\_time` datetime NOT NULL COMMENT '任务创建时间',

`create\_by` varchar(32) NOT NULL DEFAULT '' COMMENT '任务创建人',

PRIMARY KEY (`trigger\_name`)

) ENGINE=MyISAM DEFAULT CHARSET=utf8;

CREATE TABLE `schedule\_log` (

`id` varchar(32) NOT NULL COMMENT 'ID',

`trigger\_name` varchar(200) NOT NULL COMMENT '任务ID',

`bean\_name` varchar(200) NOT NULL COMMENT 'bean名称',

`method\_name` varchar(200) NOT NULL COMMENT '方法名称',

`description` varchar(250) default NULL COMMENT '描述',

`exc\_time` datetime NOT NULL COMMENT '执行时间',

`is\_success` char(1) NOT NULL COMMENT '是否成功',

`error\_msg` varchar(2000) default NULL COMMENT '异常\\错误信息',

`take\_time` int(20) NOT NULL COMMENT '本次执行耗时（ms）',

PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

# 主要业务逻辑

## 添加任务



## 查看、删除任务



## 修改任务



## 暂停、恢复任务



## 手动触发任务



# 接口说明

## EcmcScheduleInfoController(/ecmc/system/schedule)

### addTask(/addtask)

|  |  |
| --- | --- |
| 说明 | 添加任务 |
| URL | /ecmc/system/schedule/addtask.do |
| 方式 | post |
| Post数据 | {  "beanName": "testBean",  "methodName": "testMethod",  "cronExpression": "0/10 \* \* \* \* ? \*",  "taskDesc": "testDesc",  "params": [  {  "key": "key1",  "value": "value1"  },  {  "key": "key2",  "value": "value1"  }  ]  } |
| 返回Json | {  "respCode": "000000"  } |

### getTaskList(/gettasklist)

|  |  |
| --- | --- |
| 说明 | 任务列表 |
| URL | /ecmc/system/schedule/gettasklist.do |
| 方式 | post |
| Post数据 | {  "pageNumber": 1,  "pageSize": 5,  "params": {  "queryStr": ""  }  } |
| 返回Json | {  "pageSize": 5,  "totalCount": 1,  "result": [  {  "taskId": "b8313c6d62f44e5a9741b73862e38dc0",  "beanName": "testBean",  "methodName": "testMethod",  "taskState": "正常",  "cronExpression": "0/5 \* \* \* \* ? \*",  "preExcTime": 1466578300000,  "nextExcTime": 1466578305000,  "taskDesc": "testDesc"  }  ],  "totalPageCount": 1,  "currentPageNo": 1  } |

### getTask(/gettask)

|  |  |
| --- | --- |
| 说明 | 任务详细信息 |
| URL | /ecmc/system/schedule/gettask.do |
| 方式 | post |
| Post数据 | {  "taskId": "8d6be22a4b064451a23be995e83fec4e",  "beanName": "testBean"  } |
| 返回Json | {  "respCode": "000000",  "data": {  "triggerName": "8231fbc60fed460d927e937a488427e0",  "createTime": 1466668252000,  "createBy": "40288ee4545b5b5001545b901fac0007",  "taskId": "8231fbc60fed460d927e937a488427e0",  "beanName": "beeeeeee",  "methodName": "meeeeeee",  "taskState": "正常",  "cronExpression": "0/10 \* \* \* \* ? \*",  "preExcTime": 1466673420000,  "nextExcTime": 1466673430000,  "taskDesc": "88888",  "dataMap": {  "222": "333"  }  }  } |

### pauseTask(/pausetask)

|  |  |
| --- | --- |
| 说明 | 暂停任务 |
| URL | /ecmc/system/schedule/pausetask.do |
| 方式 | post |
| Post数据 | {  "taskId": "8d6be22a4b064451a23be995e83fec4e"  } |
| 返回Json | {  "respCode": "000000"  } |

### resumeTask(/resumetask)

|  |  |
| --- | --- |
| 说明 | 恢复任务 |
| URL | /ecmc/system/schedule/resumetask.do |
| 方式 | post |
| Post数据 | {  "taskId": "8d6be22a4b064451a23be995e83fec4e"  } |
| 返回Json | {  "respCode": "000000"  } |

### deleteTask(/deletetask)

|  |  |
| --- | --- |
| 说明 | 删除任务 |
| URL | /ecmc/system/schedule/deletetask.do |
| 方式 | post |
| Post数据 | {  "beanName": "testBean",  "taskId":"8d6be22a4b064451a23be995e83fec4e"  } |
| 返回Json | {  "respCode": "000000"  } |

### modifyTask(/modifytask)

|  |  |
| --- | --- |
| 说明 | 修改任务 |
| URL | /ecmc/system/schedule/modifytask.do |
| 方式 | post |
| Post数据 | {  "beanName": "test1Bean",  "oldBeanName": "testBean",  "taskId":"7d1bc8d498f94bc6856f490c389b861e",  "methodName": "test1Method",  "cronExpression": "0/10 \* \* \* \* ? \*",  "taskDesc": "test1Desc",  "params": [  {  "key": "key11",  "value": "value1"  },  {  "key": "key21",  "value": "value1"  }  ]  } |
| 返回Json | {  "respCode": "000000"  } |

### triggerTask(/triggertask)

|  |  |
| --- | --- |
| 说明 | 触发任务 |
| URL | /ecmc/system/schedule/triggertask.do |
| 方式 | post |
| Post数据 | {  "beanName": "test1Bean",  "taskId":"8d6be22a4b064451a23be995e83fec4e"  } |
| 返回Json | {  "respCode": "000000"  } |

### checkCronExpression(/checkcronexpression)

|  |  |
| --- | --- |
| 说明 | 验证触发时间表达式 |
| URL | /ecmc/system/schedule/checkcronexpression.do |
| 方式 | post |
| Post数据 | {  "cronExpression": "0 4 0 \* \* ? \*"  } |
| 返回Json | {  "respCode": "000000",  "data": true  } |

### checkBeanName(/checkbeanname)

|  |  |
| --- | --- |
| 说明 | 验证Bean是否存在 |
| URL | /ecmc/system/schedule/checkbeanname.do |
| 方式 | post |
| Post数据 | {  "beanName": "test1Bean"  } |
| 返回Json | {  "respCode": "000000",  "data": true  } |

### checkMethodName(/checkMethodName)

|  |  |
| --- | --- |
| 说明 | 验证Bean中的方法名是否存在 |
| URL | /ecmc/system/schedule/checkmethodname.do |
| 方式 | post |
| Post数据 | {  "beanName": "testBean",  "methodName":"testMethod"  } |
| 返回Json | {  "respCode": "000000",  "data": true  } |

### getAllTaskId(getalltaskid)

|  |  |
| --- | --- |
| 说明 | 获取所有任务ID |
| URL | /ecmc/system/schedule/getalltaskid.do |
| 方式 | post |
| Post数据 | {  } |
| 返回Json | {  "respCode": "000000",  "data": [  "4904b47e20a2425bb3ddbd62336eef82",  "ebb7bf2c75eb473e8e57111a8ae856d4"  ]  } |

## EcmcScheduleLogController(/ecmc/system/schedule/log)

### getLogList(/getloglist)

|  |  |
| --- | --- |
| 说明 | 历史记录列表 |
| URL | /ecmc/system/schedule/log/getloglist.do |
| 方式 | post |
| Post数据 | {  "pageNumber": 1,  "pageSize": 5,  "params": {  "taskId":"7d1bc8d498f94bc6856f490c389b861e",  "startTime":"1466662770000",  "endTime":"1466662800000",  "queryStr": "1"  }  } |
| 返回Json | {  "pageSize": 5,  "totalCount": 2,  "result": [  {  "id": "40288eee557be6a601557be9fc7b0011",  "beanName": "test1Bean",  "methodName": "test1Method",  "triggerName": "7d1bc8d498f94bc6856f490c389b861e",  "excTime": 1466662780000,  "errorMsg": "No bean named 'test1Bean' is defined",  "takeTime": 0,  "success": false  },  {  "id": "40288eee557be6a601557bea23910012",  "beanName": "test1Bean",  "methodName": "test1Method",  "triggerName": "7d1bc8d498f94bc6856f490c389b861e",  "excTime": 1466662790000,  "errorMsg": "No bean named 'test1Bean' is defined",  "takeTime": 0,  "success": false  }  ],  "totalPageCount": 1,  "currentPageNo": 1  } |

## EcmcScheduleStatisticsController(/ecmc/system/schedule/statistics)

### getStatisticsList(/getstatisticslist)

|  |  |
| --- | --- |
| 说明 | 数据统计列表 |
| URL | /ecmc/system/schedule/statistics/getstatisticslist.do |
| 方式 | post |
| Post数据 | {  "pageNumber": 1,  "pageSize": 5,  "params": {  "taskId":"ebb7bf2c75eb473e8e57111a8ae856d4",  "startTime":"1466662770000",  "endTime":"1466662800000"  }  } |
| 返回Json | {  "pageSize": 5,  "totalCount": 1,  "result": [  {  "id": "40288ef0558075af015580783734000c",  "triggerName": "ebb7bf2c75eb473e8e57111a8ae856d4",  "statisticsDate": 1466697600000,  "totalCount": 18,  "sucCount": 18,  "falCount": 0  }  ],  "totalPageCount": 1,  "currentPageNo": 1  } |

### getChartData(/getchartdata)

|  |  |
| --- | --- |
| 说明 | 数据统计图表 |
| URL | /ecmc/system/schedule/statistics/getchartdata.do |
| 方式 | post |
| Post数据 | {  "taskId":"4904b47e20a2425bb3ddbd62336eef82",  "startTime":"1466697500000",  "endTime":"1466697700000"  } |
| 返回Json | {  "ySucData": [  629  ],  "xData": [  "2016-06-24"  ],  "yFalData": [  0  ]  } |

# 集群节点关闭

## 需求

当关闭集群中的某一个节点时，将该节点下的非正在执行任务，划到其他节点下运行，而当前节点下正在执行的任务，则需所有任务执行完成后，再划到其他节点下执行

## 原理

Quartz定时任务框架自带安全关闭接口org.quartz.impl.StdScheduler.shutdown(boolean)，满足该需求。

## 测试步骤

1. 新建一个工程项目，加入Quartz框架，创建25个任务（MyJob1……MyJob25），并在Quartz配置文件中配置，每个任务每隔5秒执行一次，任务的内容为：让当前线程沉睡6秒。这样，就能让任务一直处于运行中。
2. 自定义一个插件，用于Ctrl-C的时候，触发该插件，并配置到quartz.properties

org.quartz.plugin.shutdownhook.class = com.xxx.schedule.plugin.QuartzPlugin

org.quartz.plugin.shutdownhook.cleanShutdown = true

插件内容为：

getLog().info("Shutting down Quartz...");

try {

getLog().info("当前{}个任务正在执行", scheduler.getCurrentlyExecutingJobs().size());

scheduler.shutdown(isCleanShutdown());

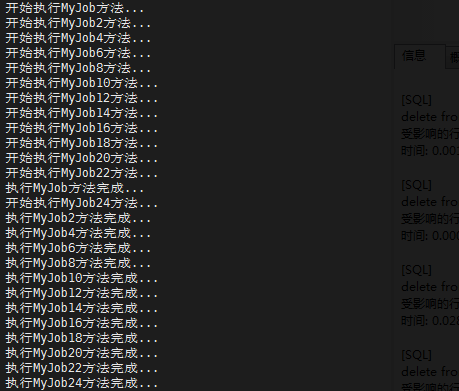
} catch (SchedulerException e) {

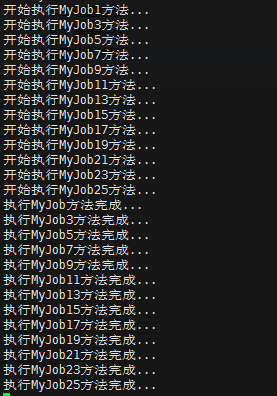
getLog().info(

"Error shutting down Quartz: " + e.getMessage(), e);

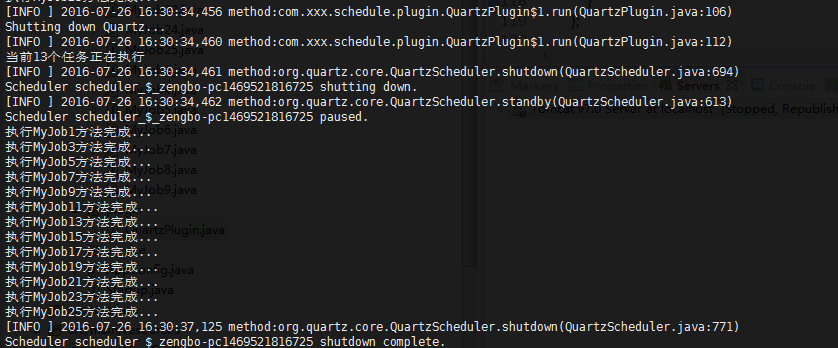
}

1. 生成jar，在命令行下分别启动两个节点，node1、node2
2. Quartz会将25个任务，轮训分配到两个节点下





1. 当使用Ctrl-C停止1节点时，打印日志如下：



1. 由上图可看出，节点停止是在所有任务执行完成之后。

## 结论

关闭节点，可使用org.quartz.impl.StdScheduler.shutdown(boolean)进行安全关闭。

## 实现方式

1. 注册Shutdownhook接收ctrl+c 或kill -15 pid 信号，调用schedule.shutdown(true)

2. 实现ui页面,通过按钮，触发 调用schedule.shutdown(true)