OpenDataSource设计文档

OpenDataSource数据采集状态服务

1. OpenDataSource数据采集状态更新
   1. 数据定义
      1. 在数据采集模块系统的tmp文件夹中内置一个文件，crawlstate.json，存放各site采集状态的数据
      2. 在crawlstate.json采用数组定义
         1. [
         2. {
         3. sitename : “”,
         4. siteurl: “”,
         5. sitedesc : “”,
         6. crawlstate : [
         7. “<crawlTime>”,
         8. “<crawlTime>”
         9. ]
         10. }，{}]
      3. 系统中配置一个默认的URL，作为上传服务器数据接收端口
   2. 采集状态更新
      1. 在每次采集抓取数据的时候，会有一个动作feedDog
      2. 在抓取完成之后，如果抓取处理成功，添加一个动作crawlStateUpdate
         1. 查找是否在内置路径下，存在有该文件，如果没有则使用内置的模板生成；
         2. 如果有，读入该文件，查找有没有link对应的站点信息，如果没有则将生成站点信息，再将当前时间放入crawlstate数组中；
         3. 完成文件写入。
   3. 采集状态上传
      1. 从配置中取得服务器数据接收端口，上传tmp文件夹中的crawlstate.json文件
      2. 上传成功确认后，删除crawlstate.json文件
      3. 采集模块会在非抓取时间段内上传采集状态文件；
2. OpenDataSource状态查询
   1. queryCrawlState： 内置方法：的思想是针对某一个数据源，某时间段进行统计，各查询的方式都是围绕这个组织，汇总
      1. 参数：
         1. 数据源对象
         2. 起点时间
         3. 终止时间
      2. 在执行查询之后，返回json字符串，其中五个参数：
         1. {
         2. sitename : “”,
         3. siteUrl : “”,
         4. siteDesc :””,
         5. count: “”，
         6. lastCrawlDate : “”
         7. }
   2. 服务函数：queryCrawlState，用以查询所有的数据源采集情况，
      1. 参数：都是Timestamp，字符串必须是yyyy-MM-dd HH:mm:ss
         1. 起点时间，字符串，起点时间可以没有，如果没有，则从1990开始
         2. 终止时间，字符串，终止时间可以没有，如果没有，则到当前时间为止
      2. 返回上述queryCrawlState对于每个site的查询结果组成的JSONArray。
3. OpenDataSource总控模块中对上述流程的调用流程
   1. 先查看对应目录下有没有crawlstate.json文件，若无，则直接返回
   2. 若有，则执行上传文件，等待返回结果
   3. 若成功，则删除crawlstate.json文件，否则直接返回

OpenDataSource数据采集模块预装的系统软件

1. 在SD卡中烧写系统完成后: 现在用的是2016-03-18-raspbian-jessie.img

2. 输入raspi-config命令来扩展磁盘空间到整个SD卡

3. 安装mysql: 在装之前一定要执行apt-get update

1. 执行 sudo apt-get install mysql-server
2. 在安装过程中输入root用户密码

4. 安装samba, 执行 apt-get install samba

1. 安装完成之后的配置文件
   1. 进入到/etc/samba目录下, mv smb.conf smb.conf.bak
   2. nano smb.conf
      1. [global]
      2. log file = /var/log/samba/log.%m
      3. [tmp]
      4. comment = Temporary file space
      5. path = /tmp
      6. read only = no
      7. public = yes
   3. 重启samba : sudo /etc/init.d/samba restart
   4. 添加pi 用户 :　sudo smbpasswd -a pi
2. 安装openvpn，执行 apt-get install openvpn
   1. 安装完成之后，测试: openvpn –config /usr/local/vpnsettings/<\*.ovpn>

OpenDataSource数据采集模块的系统流程

1. 基本过程

1. 每隔五分钟执行一次后台任务，其中系统的时间要安排好，需要在crontab文件中定义时区，其次是在crontab文件中要定义好路径
2. 周期性任务开始
   1. 先判断是否为采集数据周期
   2. 若是，执行采集状态恢复命令，再执行数据采集命令：
      1. java –jar <crawl.jar> --normal
      2. java –jar -Dfile.encoding=UTF-8 <crawl.jar>
   3. 若否，则先执行采集状态上传命令, 执行完成之后，再执行数据上传命令
      1. java –jar <crawl.jar> --report
      2. java –jar <crawl.jar> --upload

OpenDataSource数据采集模块的系统配置

1. 配置静态的IP地址，所有的模块地址，从192.168.1.108开始
   1. sudo nano /etc/dhcpcd.conf

interface eth0

static ip\_address=192.168.1.108/24

static routers=192.168.1.253

static domain\_name\_servers=8.8.8.8 218.2.135.1

* 1. sudo reboot

1. 创建VPN配置文件目录： /usr/local/vpnsettings
   1. 拷贝配置文件到这个目录下
   2. 测试vpn ： openvpn –config /usr/local/vpnsettings/<\*.ovpn>
2. 创建相关目录：
   1. 数据采集目录： /usr/local/opensourcedata
   2. 日志目录:　/usr/local/opensourcedata/logs
3. 拷贝采集数据控制文件： crawl.sh, 以及<crawl.jar> 到该采集目录下
   1. 数据采集的crawl.jar文件统一命名为: crawler.jar
4. 拷贝phantomjs和crawl.js到 /usr/local目录下，
   1. 测试phantomjs ： phantomjs crawl.js <http://www.google.com> /tmp/a.html
   2. 注： 一定要记得url加上http
5. 数据采集模块所有的文件目录结构
   * 1. 数据采集文件目录: /usr/local/opensourcedata,包含
        1. 数据采集jar包: crawler.jar
        2. 数据采集脚本: workflow.sh
        3. 一个目录: logs, 保存输出日志,一个抓取日志,另一个是VPN日志
     2. 采集数据存储目录: /opensource\_data, 包含
        1. tmp: 采集到的数据原始文件夹, 其中有一个feed.txt, 是用于看门狗使用的文件,一个crawlstate.json文件,是用于上传抓取的状态文本;
        2. images : 采集到的图片文件夹
        3. 其它文件夹暂时未用
     3. 在usr/local下面有二个下载数据必须要的配置文件:
        1. phantomjs : 这用于抓取需要动态处理的页面
        2. crawl.js : 配合phantomjs的js文件

OpenDataSource数据采集模块脚本的流程

本脚本在后台每十分钟调用一次

1. 参数定义:
   1. VPNTRYTIMES=6 // VPN尝试连接的次数
   2. VPNINDEX=1 //分享VPN的序号
   3. VPNSHAREMEMS=4 //一起分享VPN的模块数目
   4. VPNLOG="/usr/local/opensourcedata/logs/openvpn.log" //输出的VPN日志
   5. VPNTIMESLICE=90 //每次工作时间
   6. VPNSETTINGFILES[0]="/usr/local/vpnsettings/canada.ovpn" //VPN配置文件
   7. VPNSETTINGFILES[1]="/usr/local/vpnsettings/iceland.ovpn" //VPN配置文件
   8. VPNSETTINGFILES[2]="/usr/local/vpnsettings/germany.ovpn" //VPN配置文件
   9. VPNSETTINGFILES[3]="/usr/local/vpnsettings/usa.ovpn" //VPN配置文件
   10. WORKFLAG=0; //当前应执行的工作标志
   11. CRAWLDIR="/usr/local/opensourcedata" //采集文件目录
   12. CRAWLLOG="/usr/local/opensourcedata/logs/crawl.log" //采集输出的日志
2. 先判断当前模块的工作标志
   1. WORKFLAG = 0 //则为正常的维护阶段, 此时应做的工作是
      1. 杀死数据采集进程,中断和让出VPN,
      2. 报告采集数据的状态
      3. 上传采集到的数据到云平台中
   2. WORKFLAG = 1 或 2 //则为本模块正常工作的阶段, 此时应做的工作是
      1. 先判断是否是正好进入到工作阶段,如果是,则先跳过当前10分钟,以便上个处于工作状态的模块退出工作状态
      2. 如果是进入工作阶段的第二个10分钟,则此时进入数据采集状态即WORKFLAG=1
         1. 杀死数据采集进程,中断和让出VPN;
         2. 启动VPN,并监控VPN状态,直到VPN链接成功
         3. 恢复上一次数据抓取状态
         4. 开始数据采集
      3. 如果是进入工作阶段的其它时间段,则进入数据采集状态监控阶段
         1. 看门狗查看feed.txt,判断上次抓数据的时间,是否已经超时,如果超时按ii. 步处理

OpenDataSource数据采集模块crontab配置

1. TZ=Asia/Shanghai
2. SHELL=/bin/sh
3. PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin
4. \*/5 \* \* \* \* /usr/local/opensourcedata/workflow.sh