subying

努力 自信 突破

说说gogoTester-nodejs 的实现

一直在使用goagent，所以对于查找google可用ip有了很大的兴趣，在github上面发现有一个项目是gogoTester，是用来查找google ip的，于是突发奇想自己用nodejs写了一个，为gogoTester-nodejs，用nodejs实现查询google可用ip。gogoTester-nodejs 跟 gogoTester其实没有啥联系，除了使用一样的ip range（ip 段范围），其他的都是自己折腾出来的，因为gogoTester是用c#写得，看得不懂...

git项目地址为：osc@git https://git.oschina.net/subying/gogoTester-nodejs ;github:https://github.com/subying/gogoTester-nodejs

其实实现起来并不难，毕竟菜鸟。实现的方式可以能会有多种，但是流程应该是差不多，这也是一个很傻瓜化的方式，就是拿到一堆google可能会用到的ip段，然后逐个去测试，测试通过的就是可用的ip，至少能够通过这个ip直接访问google了，可以分一下几步来实现：

1.找到可用的ip 段

我承认我也不知道，所以我直接拿的是别人的，直接从gogoTester那里拿到了，然后转换成了数组，在我git项目中的ip.js可以看到，以后扩展也是通过维护这个文件。其中的代码是这样的方式

```
var iptables = [];
iptables.push("1.179.248.0-255");
iptables.push("1.179.249.0-255");
iptables.push("1.179.250.0-255");
```

数组里面的每一项是用ip段和范围组成的，“1.179.248.0-255 ”表示1.179.248.0-1.179.248.255这样的Ip段范围，所以需要一个转换的方法。

2.把数组中的每一项转成对应的ip段

“1.179.248.0-255”表示1.179.248.0-1.179.248.255这样的Ip段范围 ,那么就用split方法把字符串分成四段，最后一段是范围值，可以这样做：

```
checkStr:function(str){//检查并转换
        var arr = str.split('.')
            ,_ipStr = arr[0]+'.'+arr[1]+'.'+arr[2]+'.'
            ,_range = arr[3].split('-')
            ,_start = _range[0] || 1
            ,_end = _range[1]
            ,i = _start
            ,_self = this
        ;
        _self._ipStr = _ipStr;
        for(;i<_end;i++){
            _self.pushTask(i);
        }
    }
```

这样通过最后的一段循环，把需要查询的ip给记录下来。

3. 测试的方法

拿到需要查询的Ip后就是进行测试了，我这里用的是简单的http请求，通过判断返回的是否为'gws'（google服务器）来判断是否为google ip。方法里面还设置了超时，毕竟大家都不想访问一个太卡的Ip,所以设置了请求响应的时间，方法如下

```
function httpGet(ip,cb){
    var req = http.get('http://'+ip)
        ,err=false
    ;
```

昵称：subying
园龄：3年2个月
粉丝：0
关注：3
+加关注

< 2016年6月 >

日	一	二	三	四	五	六
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	1	2
3	4	5	6	7	8	9

搜索

找找看
谷歌搜索

常用链接

我的随笔
我的评论
我的参与
最新评论
我的标签
更多链接

我的标签

github(2)
avalonjs(2)
js(2)
nodejs(2)
nodejs http listen 本地IP(1)
nodejs socket 端口 判断(1)
nodejs 递归 查找文件(1)
out(1)
qq 客服在线 代码 样式(1)
requirejs r.js 合并压缩(1)
更多

随笔分类

bae(1)
css(1)
git(3)
js(10)
nodejs(5)
工作随记(5)
生活哲理(2)

随笔档案

2015年12月 (2)
2015年10月 (1)
2015年7月 (1)
2015年6月 (1)

```
    function endAysnc(){
        req.abort();

        if(!err){
            err = true;
            cb();
        }
    }

    req.on('response',function(res){
        //修改了判断，直接用header信息中server的判断，加快了判断速度
        if(res.headers.server === 'gws'){
            checkIpPad.addGoodIp(ip);
        }
        res.destroy();

        endAysnc();
    })
    .on('error',function(err){
        endAysnc();
        //throw err;
    })
    .setTimeout(checkIpPad.timeout,function(){
        endAysnc();
    });

    return req;
}
```

4.执行测试的控制

nodejs的http请求都是异步的，如果你不控制请求的数量，我保证你的程序很快就挂掉了，那么就需要控制同时执行的任务数量，我这里用的是async模块，这是一个很不错的模块，这里就不介绍了，大家可以搜索来了解，方法如下：

```
var q = async.queue(function(task, callback) {
    util.log('worker is processing task: '+task.name);
    task.run(callback);
}, checkIpPad.threadNum);
/**
* 监听：如果某次push操作后，任务数将达到或超过worker数量时，将调用该函数
*/
q.saturated = function() {
    util.log('all workers to be used');
}

/**
* 监听：当最后一个任务交给worker时，将调用该函数
*/
q.empty = function() {
    util.log('no more tasks wating');
}

/**
* 监听：当所有任务都执行完以后，将调用该函数
*/
q.drain = function() {
    checkIpPad.finishTask();
}
```

5.测试顺序的方式

ip.js文件里面包含的ip段就有2000多，每个ip段里面包含了一个范围，也就是有多个ip，这样下来就会有很多ip需要测试了，这个时候需要考虑测试的方式。当然最简单的就是从头到尾测试，这是最直接的方法，但是这样可能耗时会比较多。我这里还用了一个随机测试的方法，主要是数组长度范围内的随机数，然后再去找到这个对应的Ip段，再去测试里面的Ip，需要注意的是要防止重复

2015年5月 (1)
2015年4月 (1)
2014年4月 (3)
2014年2月 (2)
2014年1月 (10)
2013年8月 (1)
2013年7月 (3)
2013年5月 (2)

阅读排行榜

1. github for window 中 git shell 设置代理方法和解决ssl证书错误的问题(1312)
2. js事件队列(513)
3. JS求一个数组元素的最小公倍数(421)
4. requireJS中如何用r.js对js进行合并和压缩css文件(350)
5. bae使用nodejs遇到的问题---'Fix depends failed. Please check requirements.txt.'(225)

```
,randomCheck:function(){//随机查询
        var _self = this
            ,_num = _self.getRandom(0,_self.len-1)
            ,_str = _self.arr[_num];
        ;
        _self.checkType='random';
        _self.index = _num;
        _self._cacheIndex = _self._cacheIndex + '_'+_num+'_';

        _self.checkStr(_str);
    }
    ,listCheck:function(){//顺序查找
        var _self = this
            ,_num = _self.isInit?_self.index+1:_self.index
            ,_str
        ;
        if(_num>=_self.len){
            return false;
        }

        _self.checkType='list';
        _self.index = _num;
        _str = _self.arr[_num];
        _self.checkStr(_str);
    }
    ,getRandom:function(t1,t2){//获取随机数
        var _self = this
            ,_num = Math.floor(Math.random()*(t2-t1)+t1)
            ,_flag = true
            ,_cache = _self._cacheIndex
        ;
        while(_flag){
            if(_cache.indexOf('_'+_num+'_')===-1){
                _flag = false;
            }else{
                _num = Math.floor(Math.random()*(t2-t1)+t1);
            }
        }
        return _num;
    }
```

这里看到了随机测试和顺序测试的方法，经过本人测试，发现随机测试平均耗时会更少一些，不过话说是靠人品的...

实现的方式就是这样，里面主要是用到了Ip段、http请求、async控制和随机测试，感兴趣的朋友可以了解一下。另外，我用到的Http请求的方式是不够严谨的，因为google用的是https协议，所以用Https会更好，但是我目前还没有实现，希望已经实现的朋友能给我一些帮助。

本文同步发表在我的个人博客：http://www.subying.com/archives/125.html。

保持一颗好奇的心

分类: nodejs

标签: nodejs, google, ip, gogotester, 可用Ip

« 上一篇：使用sshkey连接github等服务器
» 下一篇：linux下使用sublime-text写coffee遇到的编译问题

注册用户登录后才能发表评论，请 登录 或 注册，访问网站首页。