**Snova翻墙代理更新，支持Heroku/CloundFoundry/OpenShift/Jelastic及Google AppEngine平台**

2012年1月21日 | 分类: [翻墙相关](http://igfw.net/archives/category/fanqiangxiangguan) | 标签: [CloundFoundry](http://igfw.net/archives/tag/cloundfoundry), [Google Appengine](http://igfw.net/archives/tag/google-appengine), [Heroku](http://igfw.net/archives/tag/heroku), [Jelastic](http://igfw.net/archives/tag/jelastic), [OpenShift](http://igfw.net/archives/tag/openshift), [snova](http://igfw.net/archives/tag/snova)

Snova是[hyk-proxy](http://igfw.net/archives/3900)的延续，以其强大的功能完美的图形化界面稳定的性能著称，网友**tianye**提醒说其昨天又升级了一次，支持了Heroku/CloundFoundry/OpenShift/Jelastic这几个PaaS平台，**其中HTTPS是原生支持不需要伪造证书**。不过看了下guide，这几个平台部署起来大部分都够折腾的，喜欢折腾的又有得玩了，一起看看相关教程吧（不折腾的简单用法参考 <http://igfw.net/archives/7040>）。

**关于**

snova是一个插件化的web proxy框架，目前自动集成基于Google AppEngine平台的GAE插件，支持Heroku/CloundFoundry/OpenShift/Jelastic平台的C4插件，以及增强的PAC插件SPAC.

**安装部署**

* 安装运行依赖

JRE 1.6+

* **GAE**

参考[GAEInstallation](http://code.google.com/p/snova/wiki/GAEInstallation)安装配置部署GAE Plugin

参考<http://igfw.net/archives/6848>

* **CloundFoundry**(可选)

参考[CloudFoundryInstallation](http://code.google.com/p/snova/wiki/CloudFoundryInstallation)安装配置部署C4 Plugin到CloundFoundry上

**Step 1: 注册CloundFoundry环境**

到官方链接[CloundFoundry](https://my.cloudfoundry.com/signup)注册帐号， 注意，注册不是马上成功，一般第二天才会收到注册成功的邮件，其中包含用户名密码

**Step 2: 安装CloundFoundry部署工具**

* 参考官方说明安装命令行工具vmc，注意安装依赖ruby以及gem的安装， [vmc安装](http://start.cloudfoundry.com/tools/vmc/installing-vmc.html)
* 注意，gem安装vmc时可能会被防火墙中断，可能需要设置代理，代理可以用snova设置，如

       gem install --http-proxy http://127.0.0.1:48100 vmc

**Step 3: 部署服务到CloundFoundry**

* 将snova-c4-server-[version].war放到任意的空目录下，然后在命令行下进入该目录，逐个执行下面的三行命令

       vmc target api.cloudfoundry.com

       vmc login

       vmc push <appname>  —— 此处appname为任意名称，为域名一部分，此命令执行后有类似下面的交互内容，参照下面的输入Y/N

        Would you like to deploy from the current directory? [Yn]: Y

        Application Deployed URL [<appname>.cloudfoundry.com]: <回车>

        Detected a Java Web Application, is this correct? [Yn]: Y

        Memory Reservation (64M, 128M, 256M, 512M, 1G, 2G) [512M]:   <回车>

        Creating Application: OK

        Would you like to bind any services to '<appname>'? [yN]: n

        Uploading Application:

        Checking for available resources: OK

        Processing resources: OK

        Packing application: OK

        Uploading (843K): OK

        Push Status: OK

        Staging Application: OK

        Starting Application: OK

* 浏览器中输入<appname>.cloundfoundry.com， 查看是否部署成功，否则检查上述步骤是否执行成功

**Step 4: 配置Snova C4 Plugin的客户端**

* 从snova-0.12.0120开始自动集成c4 plugin，无需单独安装
* 修改<snova>/plugins/c4/conf/c4-client.xml

       <!--修改domain为Step2创建的域名, 可加多个域名-->

       <WorkerNode domain="xyz.herokuapp.com" />

* 修改<snova>/conf/snova.xml，将heroku改为默认proxy实现(可选)

       <!--修改Proxy实现为C4， 默认为GAE-->

       <ProxyEventHandler>C4</ProxyEventHandler>

**Step 4: 与SPAC集成（可选）**

* 在SPAC的解释语言中，c4被定义为$C4=”C4″
* 参考SPAC的说明[SpecialProxyAutoConfig](http://code.google.com/p/snova/wiki/SpecialProxyAutoConfig)修改SelectProxy方法打造自己的proxy环境
* **Heroku**(可选)

参考[HerokuInstallation](http://code.google.com/p/snova/wiki/HerokuInstallation)安装配置部署C4 Plugin到Heroku上

**Step 1: 准备Heroku SDK环境**

按照Heroku官方的[QuickStart](http://devcenter.heroku.com/articles/quickstart)注册帐号，安装SDK（注册较简单，仅需要email）

**Step 2: 部署Snova C4 Plugin的服务端到Heroku**

* 下载snova-c4-heroku-serverr-[version].zip， 在任意目录下解压
* 在命令行下进入解压的目录，依次顺序执行以下的命令,每一行单独执行

       heroku login

       git init

       git add .

       git commit -m "init"

       heroku create --stack cedar

       git push heroku master

* 留意执行“heroku create –stack cedar”时的输出，一般会显示创建的域名，为 “xx.herokuapp.com”, 记下该域名，为配置Client准备（该域名也可登录Heroku的Web找到）

**Step 3: 配置Snova C4 Plugin的客户端**

* 从snova-0.12.0120开始自动集成c4 plugin，无需单独安装
* 修改<snova>/plugins/c4/conf/c4-client.xml

       <!--修改domain为Step2创建的域名, 可加多个域名-->

       <WorkerNode domain="xyz.herokuapp.com" />

* 修改<snova>/conf/snova.xml，将heroku改为默认proxy实现(可选)

       <!--修改Proxy实现为C4， 默认为GAE-->

       <ProxyEventHandler>C4</ProxyEventHandler>

**Step 4: 与SPAC集成（可选）**

* 在SPAC的解释语言中，heroku被定义为$HEROKU=”Heroku”
* 参考SPAC的说明[SpecialProxyAutoConfig](http://code.google.com/p/snova/wiki/SpecialProxyAutoConfig)修改SelectProxy方法打造自己的proxy环境
* **OpenShift**(可选)

参考[OpenShiftInstallation](http://code.google.com/p/snova/wiki/OpenShiftInstallation)安装配置部署C4 Plugin到OpenShift上

**Step 1: 注册OpenShift环境**

到官方链接[OpenShift注册](https://openshift.redhat.com/)注册帐号

**Step 2: 安装OpenShift部署工具**

* 参考官方说明安装命令行工具rhc，注意安装依赖ruby以及gem的安装， [rhc安装](https://openshift.redhat.com/app/express#quickstart)
* 注意，gem安装rhc时可能会被防火墙中断，可能需要设置代理，代理可以用snova设置，如

       gem install --http-proxy http://127.0.0.1:48100 rhc

**Step 3: 部署服务到OpenShift**

* 将snova-c4-server-[version].war放到任意的空目录下，然后在命令行下进入该目录，逐个执行下面的命令

      rhc-create-domain -n <domainName> -l <yourId> -p <yourPassword>  创建主域名， 部署新应用是这一步可不执行

      rhc-create-app -a <appName> -t jbossas-7.0 -p <yourPassword>  创建app

      cd <appName>

      mv ../snova-c4-server-[version].war <appName>/deployments/ROOT.war

      git rm -r src pom.xml

      git commit –m “delete”

      git push                         以上三步重新部署同一个app时可不执行

      git init

      git add .

      git commit –m “deploy”

      git push

* 浏览器中输入<yourdomain>.rhcloud.com， 查看是否部署成功，否则检查上述步骤是否执行成功

**Step 4: 配置Snova C4 Plugin的客户端**

* 从snova-0.12.0120开始自动集成c4 plugin，无需单独安装
* 修改<snova>/plugins/c4/conf/c4-client.xml

       <!--修改domain为Step2创建的域名, 可加多个域名-->

       <WorkerNode domain="xyz.herokuapp.com" />

* 修改<snova>/conf/snova.xml，将heroku改为默认proxy实现(可选)

       <!--修改Proxy实现为C4， 默认为GAE-->

       <ProxyEventHandler>C4</ProxyEventHandler>

**Step 4: 与SPAC集成（可选）**

* 在SPAC的解释语言中，c4被定义为$C4=”C4″
* 参考SPAC的说明[SpecialProxyAutoConfig](http://code.google.com/p/snova/wiki/SpecialProxyAutoConfig)修改SelectProxy方法打造自己的proxy环境
* **Jelastic**(可选)

参考[JelasticInstallation](http://code.google.com/p/snova/wiki/JelasticInstallation)安装配置部署C4 Plugin到Jelastic上

**Step 1: 注册Jelastic环境**

到官方链接[jelastic](http://jelastic.com/)注册帐号

**Step 2: 部署**

* 完全图形化的操作，无需安装工具，按照说明将snova-c4-server-[version].war上传并deploy到ROOT下即可 [jelastic部署指南](http://jelastic.com/docs/upload-deploy-application)

**Step 4: 配置Snova C4 Plugin的客户端**

* 从snova-0.12.0120开始自动集成c4 plugin，无需单独安装
* 修改<snova>/plugins/c4/conf/c4-client.xml

       <!--修改domain为Step2创建的域名, 可加多个域名-->

       <WorkerNode domain="xyz.herokuapp.com" />

* 修改<snova>/conf/snova.xml，将heroku改为默认proxy实现(可选)

       <!--修改Proxy实现为C4， 默认为GAE-->

       <ProxyEventHandler>C4</ProxyEventHandler>

**Step 4: 与SPAC集成（可选）**

* 在SPAC的解释语言中，c4被定义为$C4=”C4″
* 参考SPAC的说明[SpecialProxyAutoConfig](http://code.google.com/p/snova/wiki/SpecialProxyAutoConfig)修改SelectProxy方法打造自己的proxy环境

* **SPAC**(可选)

参考[SpecialProxyAutoConfig](http://code.google.com/p/snova/wiki/SpecialProxyAutoConfig)配置SPAC

**简介**

* 由于基于GAE/Heroku的proxy有一些固有的限制（如1m限制， DELETE with body等）或者其他一些proxy实现也有一些具体限制,在这些限制情况下无法完成proxy功能；因此需要一个自动选择proxy机制来简化用户频繁 切换proxy的动作
* 大部分情况下，用户可以用PAC脚本来配置选择合适的proxy；但PAC有自身的一些限制：
* 只能匹配URL选择，致使无法完美匹配某些受限proxy的受限条件；
* 一般情况下，有些受限proxy实现限制请求类型（如限制DELETE/TRACE请求），或者限制上传大小等等，这些都是PAC无法表达的

 目前基于GAE的proxy的限制

* 上传/下载不能超过1m(下载在大部分情况下可以绕过)
* 只支持GET/POST/PUT/DELETE/HEAD 请求
* 只支持POST/PUT携带body

 鉴于PAC的限制，snova提供了一个SPAC插件来完成更为强大的类PAC功能

**SPAC插件**

* snova从V0.12.0101开始集成SPAC插件.
* 用户可以定义任意的第三方proxy client，不仅仅局限于Snova GAE/Heroku本身;换句话说， Snova SPAC可以仅仅作为一个类PAC实现运行
* SPAC插件中包含一个TykeDog脚本语言解释器，用户需要少量修改tykedog脚本完成自定义PAC功能

**TykeDog脚本语言**

* 此脚本语言较简单，而执行效率比JS等其他脚本为高10倍以上，语法类似C/shell
* 修改编写TykeDog脚本注意几点即可：
* if/elif/else/while后的代码必须用 “{ }” (与C不同)
* 变量只有两个作用域：全局作用域以及函数作用域（第三方proxy都是全局变量表示）
* 语法上支持几乎所有运算符，但语义上仅实现了字符串/整数的比较（”>，<，！＝，==等”），此由SPAC插件实现决定
* “＃ //”都可以作为注释符
* 脚本的目前入口有三处：OnInit/OnRoutine/SelectProxy, 入口的详细解释请参考后面”配置/脚本”章节
* 目前支持的内置函数：

              1. 打印函数 Print($content)：打印到标准输出

                  eg: Print("hello,world!");

              2. 获取头域值 GetHeader($req, $name) : 获取HTTP请求/响应中具体头域

                  eg： $host = GetHeader($request, "Host");

              3. int转化函数 Int($str): 将字符串转化为int

                  eg:  $contentLength = Int(GetHeader($errorRes, "Content-Length"));

              4. 调用外部程序 System($cmd): 执行外部程序

                  eg：$output=System("./my.sh");

              5. 日志函数 Log($content): 记录内容到日志文件中

                  eg: Log("hello, world");

**安装使用**

* snova从V0.12.0101开始集成SPAC插件,无需单独安装
* 按照下一节修改配置/脚本后，启动snova即生效

**配置&脚本**

* 首先启用SPAC需要修改配置文件<snova>/conf/snova.xml

      <!--修改Proxy实现为SPAC， 默认为GAE-->

       <ProxyEventHandler>SPAC</ProxyEventHandler>

* 其次修改脚本文件spac.td, spac.td位于spac插件目录下
* 第一步定义需要的第三方proxy，需要在spac.td的开头定义

          ##基本只需要定义IP+port，用变量表示

          ##Third proxy client

          $TOR="127.0.0.1:9050";

          $PUFF="127.0.0.1:1984";

* 默认的proxy都是HTTP proxy，若是socks类型需要特殊定义：

          $SSHD="socks5:127.0.0.1:7070";

* spac.td中定义了Routine接口方法OnRoutine, 用户控制routine频率，若有定期执行某项动作的需要，可在此定义(如修改选择的proxy地址等):

              def OnRoutine()

              {

               #Do your routine business here

               #eg: $output=System("./mysh.sh");

               #    Log($output);

               #Tell the script engine to invoke this method 10s later

                return 10;

               }

* 第二步修改选择proxy方法，spac.td定义了proxy选择点方法SelectProxy：

##此示例方法在GAE的限制条件下选择Heroku/Google作为proxy， 用户可以参考修改

def SelectProxy(protocol, method, url, headers)

{

     #Set your https proxy here

     $host = GetHeader($headers, "Host");

     if $host~=".google" || $host~="ytimg."|| $host~="gstatic."

            || $host~="developer.android.com" || $host~=".appspot.com"

            || $host~="www.youtube.com" || $host~="golang.org"

     {

        return $GOOGLE;

     }

     $contentLength = Int(GetHeader($headers, "Content-Length"));

     if $contentLength > 10240000

     {

         return $HEROKU;

     }

     ##Only "GET/POST/PUT/DELETE/HEAD" supported in GAE

     if $method != "GET" && $method != "POST"

        && $method != "PUT" && $method != "DELETE"

        && $method != "HEAD"

     {

        return $HEROKU;

     }

     ##Request with body not allowed in GAE if it's not POST/PUT

     if $contentLength > 0

     {

        if $method != "POST" && $method != "PUT"

        {

           return $HEROKU;

        }

     }

     return $GAE;

}

**项目地址**：<http://code.google.com/p/snova/>  
**下载地址**：<http://code.google.com/p/snova/downloads/list>

**本文原始地址**：<http://igfw.net/archives/7237>