2016 年京津冀大学生网络安全知识技能挑战赛 writeup

队名：C-cUp

成员：杨昌鑫、李南均、王程

0x01 签到题

扫一扫，反正我是没扫出来，队友说是用百度扫的

0x02 Hello

查看页面的代码发现如下表达式，要求 sha1(var)==Ciphertext，显然这里的 Ciphertext 应该是个变量

```
</head>
<body>
<!--ROUND ONE-->
<!--找出明文 -->
<!--sha1(三个数字+fF8JpdJFFgEXSXS1TXuhperA7B0iMR)==Ciphertext -->
<!--请于10s内提交答案 -->
  <div class="panel">
  <div class="wrap">
    <form method="POST" action="#">
```

发现请求页面时响应头部有 Ciphertext 的值。

那思路就很清晰了：访问页面获得 sha1()括号中的字符串以及头部的 Ciphertext 的值，通过暴力穷举的方式找到满足条件的三个数字

脚本如下

```
full_url='http://106.75.67.214:2250/?pass=0debc27caa70ce48#'
ycx = requests.session()
data = ycx.post(full_url)
Data = data.content
head = data.headers["Ciphertext"]

string_list = Data.split()
result = string_list[19]
result = result[22:52]


final = list(itertools.permutations([0,1,2,3,4,5,6,7,8,9],3))
for i in range(0,720):
        my_best = ''.join('%s' % id for id in final[i])
        print my_best+result
        secret = hashlib.sha1(my_best+result).hexdigest()
        #print secret
        if(hashlib.sha1(my_best+result).hexdigest()==head):
```

跑了之后竟然得到这个，还需要算个表达式；

```
974kwGN7KexeA5kco4U7FAEv9LgqVQDPg
974
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Hello</title>
  <link rel="stylesheet" href="css/style.css" media="screen" type="text
</head>

<body>
<!--ROUND TWO-->
<!--找出明文 -->
<!--算出值提交：499*533*3+1-1*4*297-4*516-14+4+282+5-427-91-48+55 -->
<!--请于10s内提交答案 -->
  <div class="panel">
  <div class="wrap">
    <form method="POST" action="#">
      <input type="text" name="pass" placeholder=" here"/>
      <button onclick="form.submit();">Shhh!</button>
    </form>
  </div>
</div>
<div style="text-align:center;clear:both">
```

大同小异，继续提取页面中的表达式 eval()计算提交即可：

```
499*533*3+1-1*4*297-4*516-14+4+282+5-427-91-48+55
794416
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Hello</title>
  <link rel="stylesheet" href="css/style.css" media="screen" type="text/c
</head>

<body>
<!--找出明文 -->
<!--flag{f325c62b-9505-4c13-ad4b-010bddb23c68} -->
<!--请于10s内提交答案 -->
  <div class="panel">
  <div class="wrap">
    <form method="POST" action="#">
      <input type="text" name="pass" placeholder=" here"/>
      <button onclick="form.submit();">Shhh!</button>
    </form>
  </div>
</div>
<div style="text-align:center;clear:both">
</div>
```

0x03 crack MD5

访问页面得到如下返回，大意是第二行的是 md5 密文，其对应的明文就是第三行的那些字母，不过顺序被打乱了。

看明白题思路就清晰了：首先访问页面，提取密文及明文字符串，对明文字符串进行排列组合并加密，将结果与已给的密文比较即可

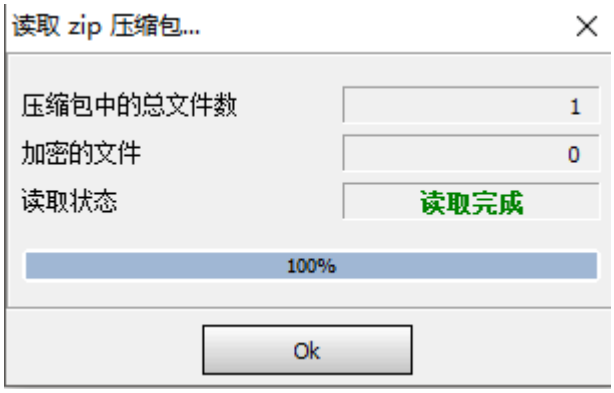
```
27
28 final = list(itertools.permutations([result[0],result[1],result[2],result
   ,result[4],result[5],result[6],result[7],result[8]],9))
29
30 #print len(final)
31
32 for i in range(0,362880):
33         my_best = ''.join(final[i])
34         if(md5(my_best)==secret):
35                 print my_best
36                 success_url = "http://106.75.67.214:2050/?code="+my_best
37                 flag = ycx.get(success_url)
38                 print flag.content
39                 break
40         else:
```

但是这样做很耗时，大部分情况都是 too late，不过还好侥幸跑出来一次。

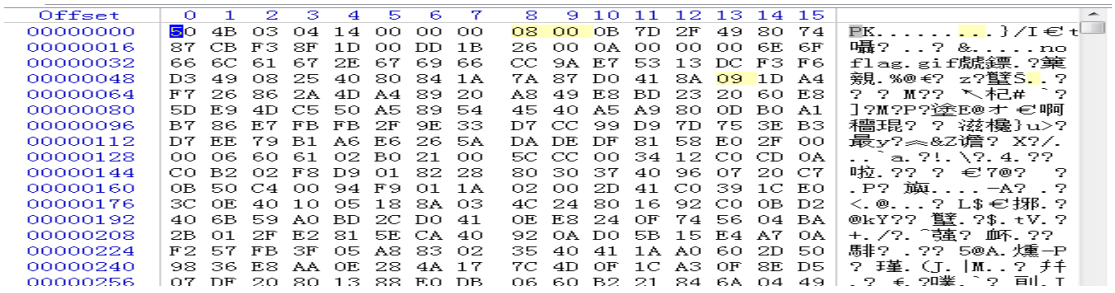
```
try_again
Z6[zPd3P(
<!DOCTYPE html>
<html>
<head>
        <title>Brute force crack MD5</title>
</head>
<body>
<h1>Brute force crack MD5</h1>
flag{25e54f71-6bfe-4962-811c-454aa9a07e62}</body>
</html>
```

0x04 破解

文件是个压缩包，解压还要密码，丢到某破解软件中，提示加密的文件为 0。



那应该是伪加密了，用十六进制编辑器打开，将文件相应的标志位 0900 修改为 0800 即可。



解压文件得到 noflag.gif。

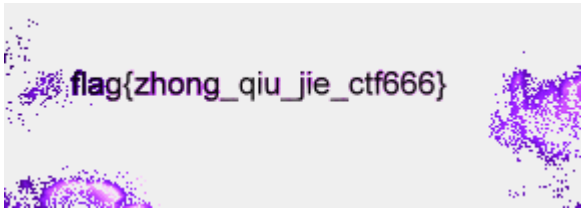
并没有什么，想用记事本打开图片，竟然把我电脑卡死了。继续用 winhex 打开。

```
:3 45 78  61 52 7A 42 33 56 44 4A | WVlExcExaRzB3VDJ
:3 48 5A  54 52 56 55 30 56 56 52 | 3NFFWcHZTRVU0VVR
:A 7A 42  57 56 7A 6C 52 56 46 56 | oblRuZzBWVzlRVFV-
:5 6A 5A  55 61 30 35 46 55 31 68 | aRWRGUjZUa05FU1h
:7 6B 39  6B 57 47 52 58 54 6A 4A | GMU0xWk9kWGRXTjJ
:2 58 5A  57 56 54 6C 4A 59 55 56 | kRGR6RXZWVTlJYUV
:A 45 35  5A 65 6B 35 32 55 56 64 | RMlZYZE5Zek52UVd
:A 48 46  6A 56 55 70 4E 56 46 56 | aTlMyZHFjVUpNVFVV
:F 54 42  61 53 47 68 5A 54 6C 5A | WV2NXOTBaSGhZTlZ
:1 6D 39  53 52 45 4A 31 57 6C 52 | kTlRuQm9SREJ1WlR
:4 58 4A  6A 4D 6C 5A 61 5A 46 4D | oSmJqTXJjMlZaZFM
```

发现有一大长串似乎是 base64 码的字母，提取出来解码，发现解码后还是 base64，有情况啊！继续解码，还是 base64，三次之后解出如下字符串，很明显是一副 gif。

用 base64 直接解码成图片

还是动的，用 gif 编辑器打开得到 flag

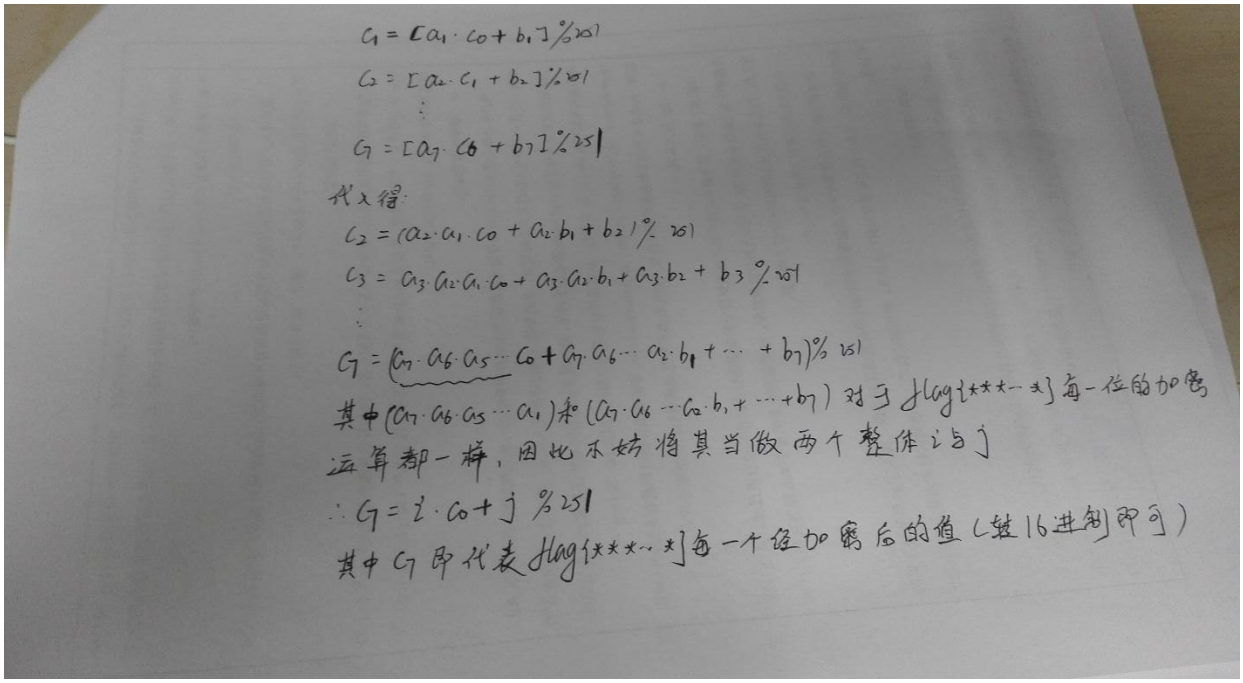


0x05 tryhard

加密部分算法，flag{******}每个字符会生成一个两位的 16 进制形式的字符，而密文 40 位，所以 flag{*****}共 20 位，其中 key=****** 长 14 位，对于从 flag{*****}开始的每一个字符，加密过程由 key 中的[0,7],[1,8]……位依次参与迭代运算：

```
for c in flag:
    print c
    c = ord(c)
    print c
    for a,b in zip(key[0:n], key[n:2*n]):
        print a,b
        c = (ord(a) * c + ord(b)) % 251
    print c
    encrypted += '%02x' % c
    print encrypted
print encrypted
print len(encrypted)
```

加密算法迭代化简一下就更清晰了：



由于 flag{********}中 flag{}及其对应的密文是已知的，在 mod251 之后 i,j 可认为是[0,251]之间的数，带进去暴力破解即可

```
import sys

rev_key = 'de3d93ed23a293b4dec3f0b4b4deb1d2a5e1c341'
test = []
result = []
for i in range(0,5):
        test.append(rev_key[2*i:i*2+2])
        result.append(int(test[i],16))
print result

flag= 'flag{'

for i in range(0,251):
        for j in range(0,251):
                secret = []
                for k in range(0,5):
                        secret.append((i*ord(flag[k])+j)%251)
                if(secret==result):
                        print i,j
```

求得 i,j 分别等于 15 和 198。

```
root@kali:~/Desktop/tryhard# python reverse.py
[222, 61, 147, 237, 35]
15 198
```

现已知 i,j 那么密文中 flag{*****}对应*******的那一串根据化简出来的表达式，反带即可

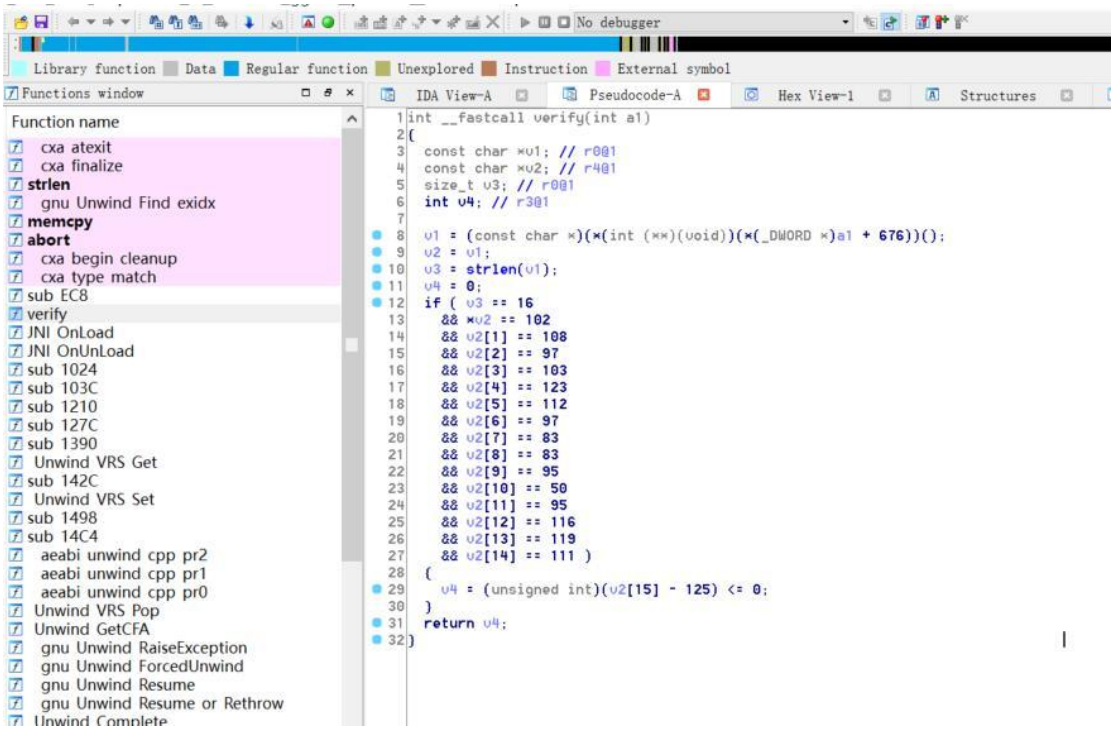
```
rev_key = 'de3d93ed23a293b4dec3f0b4b4deb1d2a5e1c341'
test = []
result = []
for i in range(0,20):
        test.append(rev_key[2*i:i*2+2])
        result.append(int(test[i],16))
print result
print len(result)

key = []
for i in range(0,14):
        for j in range(0,251):
                if((15*j+198)%251==result[i+5]):
                        key.append(chr(j))
print key
```

```
[222, 61, 147, 237, 35, 162, 147, 180, 222, 195, 240, 180, 180, 222, 177
65, 225, 195, 65]
20
['b', 'a', '1', 'f', '2', '5', '1', '1', 'f', 'c', '3', '0', '4', '2']
```

0x06 re50

这是我队友做的，打完比赛他就出去玩了，我帮他写一下，不周之处还请包涵。IDA 载入看到 verify 函数，里面的 v2[]数组即是 flag



0x07 cry50

首先 base64 解码密文 Ot7lAO72opsedkxTngbD3FhwP50x8sosA f9oL

OkIpr8PN7J0Omq7nWxvgvaiRn+Tp95zcTDj，

```
for n in range(salt_length):
    salt += chr(random.randrange(256))
data = salt + crypt(data, sha1(key + salt).digest())
if encode:
    data = encode(data)
return data
```

解码后的密文前 16 位即是 salt，后面是 crypt()加密的结果，key 已知，salt 已知，则 sha1(key+salt).digest()可知，然后将结果及解码后密文 crypt()加密的部分丢进去，解密即可得到 flag

```
a2YsP9zCpXUdO1+CaemC5oCVQNQpPzXCbSWKZDYZnKZfXXn7/r+GEBuzseKa0w==
flag{197a9b67f0ebc04647c1eeefcec99808}
root@kali:~/Desktop/cry100#
```