

i春秋 第二届春秋欢乐赛 Misc 题目名称: CryMisc

原创

loading... 于 2019-01-18 21:24:58 发布 1970 收藏 5

分类专栏: [CTF](#) 文章标签: [ctf Misc](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: https://blog.csdn.net/qq_41137110/article/details/86546501

版权



[CTF 专栏收录该内容](#)

11 篇文章 0 订阅

订阅专栏

i春秋 第二届春秋欢乐赛

分值: 300分 类型: Misc 题目名称: CryMisc

题目内容: [Download](#)

下载链接:

http://static2.ichunqiu.com/icq/resources/fileupload/CTF/icqhappycyf/CryMisc_E1C844B98C4CAC14060994BD

审题发现题目只给了一个下载链接,

1, 点击Download下载了一个压缩包, 解压后有两个文件, crypto.zip和jiami.py

crypto.zip	2017/5/15 20:23	WinRAR ZIP 压缩...	227 KB
jiami.py	2017/5/15 20:23	Python File	1 KB

打开jiami.py文件内容如下:

```
# -*- coding:utf8 -*-

import pyminizip
from hashlib import md5
import os

def create(files, zfile):
    password = os.urandom(15)#随机产生一个15字节的字符串
    password = md5(password).hexdigest()#获取这个字符串的md5值
    pyminizip.compress_multiple(files, zfile, password, 0)#这是一个压缩文件的方法把files参数中文件压缩成zfile
    #指定的文件，压缩密码是password,0可以理解为复杂程度值为1~9，默认是0
    pass

if __name__ == '__main__':
    files = ['jiami.py','gogogo.zip']
    zfile = 'crypto.zip'#就是把jiami.py和gogogo.zip两个文件压缩成crypto.zip密码是上面随机字符串的md5值
    create(files, zfile)
```

接下来打开crypto.zip提示有密码，也就是上述md5值，这个密码我们是不知道的。

因为crypto.zip压缩包中有jiami.py文件，而jiami.py这个文件是已知的，因此我们可以用zip明文攻击。

明文攻击需要利用两个压缩包（已知文件的压缩包，和加密的需要破解的压缩包），这两个压缩包压缩方式要相同，直接对jiami.py文件进行压缩是不行的，也要用python的pyminizip模块进行压缩，脚本如下：

```
# -*- coding: cp936 -*-
import pyminizip
pyminizip.compress(r"jiami.py","",r"jiami.zip","",0)#没有密码
```

然后进行明文攻击，利用一个工具apzr



爆破成功保存为压缩包UnEncrypted.zip解压后得到gogogo.zip,
解压得到AES.encrypt、AESencrypt.py和RSA.encrypt三个文件

File Name	Date	Type	Size
gogogo.zip	2017/5/15 17:25	WinRAR ZIP 压缩...	227 KB
AESencrypt.py	2017/5/15 17:24	Python File	1 KB
AES.encrypt	2017/5/15 17:24	ENCRYPT 文件	226 KB
RSA.encrypt	2017/5/15 17:24	ENCRYPT 文件	1 KB

2, 打开AESencrypt.py文件如下:

```
# -*- coding:utf8 -*-
```

```
from Crypto.Cipher import AES
```

```
s=open('next.zip','rb').read()
```

```
BS=16
```

```
pad_len=BS-len(s)%BS
```

```
padding=chr(pad_len)*pad_len
```

```
s+=padding#把最后不满16个字节的用所缺字节个数值ASCII码对应的字符补足16个字节，如缺5个字节就补5个ascii码为5的字符，因为AES明文是128bit的倍数
```

```
key='我后来忘了#AES密钥（未知）
```

```
n=0x48D6B5DAB6617F21B39AB2F7B14969A7337247CABB417B900AE1D986DB47D971#两个大素数p与q乘积
```

```
e=0x10001#RSA公钥65537
```

```
m=long(key.encode('hex'),16)#密钥key转16进制转整型，作为RSA的明文
```

```
c=pow(m,e,n)#c是RSA加密后的密文
```

```
c='0{:x}'.format(c).decode('hex')
```

```
with open('RSA.encrypt','wb') as f:
```

```
    f.write(c)
```

#RSA.encrypt文件的16进制就是RSA的密文，即就是AES密钥key加密后的密文，已知

```
obj=AES.new(key,AES.MODE_ECB)
```

```
with open('AES.encrypt','wb') as f:
```

```
    f.write(obj.encrypt(s))#对next.zip进行AES加密，密钥为key,AES.encrypt文件中的内容即为AES加密后的密文已知。
```

整合已有信息就是，已知AES加密后的密文要解出AES的明文；但是AES密钥不知道，要先求出AES密钥key

key就是RSA的明文，已知RSA的密文c,两个大素数乘积n也已知可以求出p,q,然后求出RSA私钥，然后根据私钥，密文，n可以求出RSA明文(就是AES的密钥key)，然后根据AES密文和密钥key就能解除AES的明文。

首先n转成10进制为

32945885482421841602167475970472000545315534895409154025267147105384142461297

登录网站<http://factordb.com/>,对n进行因数分解，解出p和q

The screenshot shows the factordb.com interface. At the top, there are navigation tabs: Sequences, Report results, Factor tables, Status, and Downloads. Below these is a search bar containing the number 32945885482421841602167475970472000545315534895409154025267147105384142461297. To the right of the search bar is a 'Factorize!' button and a help icon. Below the search bar, the result is displayed under the heading 'Result:'. The result shows the number 3294588548...97 factored into two large prime numbers: 177334994338425644535647498913444186659 and 185783328357334813222812664416930395483.

脚本如下：

```
# -*- coding: cp936 -*-
```

```
from Crypto.PublicKey import RSA
```

```
from Crypto.Cipher import AES
```

```
def egcd(a,b):
```

```
    if a==0:
```

```
        return (b,0,1)
```

```
    else:
```

```
        g,y,x=egcd(b%a,a)
```

```
        return (g,x-(b//a)*y,y)
```

```
def modinv(a,m):
```

```
    g,x,y=egcd(a,m)
```

```
    if g!=1:
```

```
        raise Exception('modular inverse does not exist')
```





```
    else:
```

```
        return x%m
```

```
p=177334994338425644535647498913444186659
q=185783328357334813222812664416930395483
n=32945885482421841602167475970472000545315534895409154025267147105384142461297
e=65537#公钥
```

```
d=modinv(e,(p-1)*(q-1))#RSA私钥
c=open("RSA.encrypt","rb").read();
c=long(c.encode('hex'),16)
m=pow(c,d,n)#m是明文,转成字符串就是key="copy__white__key"
key="copy__white__key"
obj=AES.new(key,AES.MODE_ECB)
s=open("AES.encrypt","rb").read()
str=obj.decrypt(s)
with open(r'next.zip','wb') as f:
    f.write(str)#解密后得到next.zip文件
```

3, 解压next.zip文件得到encrypt.py、first、second三个文件。

 next.zip	2019/1/18 20:18	WinRAR ZIP 压缩...	226 KB
 first	2017/5/15 16:47	文件	194 KB
 second	2017/5/15 16:47	文件	194 KB
 encrypt.py	2017/5/15 16:47	Python File	1 KB

打开encrypt.py内容如下:

```
# -*- coding:utf8 -*-
from base64 import *

s=open('flag.jpg','rb').read()
s='-'.join(map(b16encode,list(s)))

#list(s)每个字节（一个字符）作为列表的一项

#map(b16encode,list(s)), 列表的每一项都执行b16encode（这个函数其实是得到字符对应的十六进制值），并将结果作为新列表中的项

#'-'.join(), 列表中的每项都用'-'分隔。

#最终的执行结果是flag.jpg文件的用'-'分隔字节的十六进制数据

s=map("".join,zip(*(s.split('-'))))

#zip(*(s.split('-')))得到两个元组，一个是每个字节的第一个16进制值组成的，一个是每个字节的第二个16进制值组成的

#然后作为一个新列表

with open('first','wb') as f:
    f.write(b16decode(s[0]))#把第一个列表元素转成字符串写入first文件
with open('second','wb') as f:
    f.write(b16decode(s[1]))#把第二个列表元素转成字符串写入second文件
```

代码分析完也就是把flag.jpg文件的16进制值分成了两部分，每个字节的前4位组合成first文件，后4位组合成second文件。写脚本如下：

```
# -*- coding: cp936 -*-
from base64 import *
s1=open(r'first','rb').read()
s2=open(r'second','rb').read()
s1=".join(map(b16encode,list(s1)))#获取16进制数据
s2=".join(map(b16encode,list(s2)))
str=""
for i in range(0,len(s1)):
    str+=s1[i]+s2[i];#得到flag.jpg16进制数据
str=str.decode('hex')
with open(r'flag.jpg','wb') as f:
    f.write(str)
```

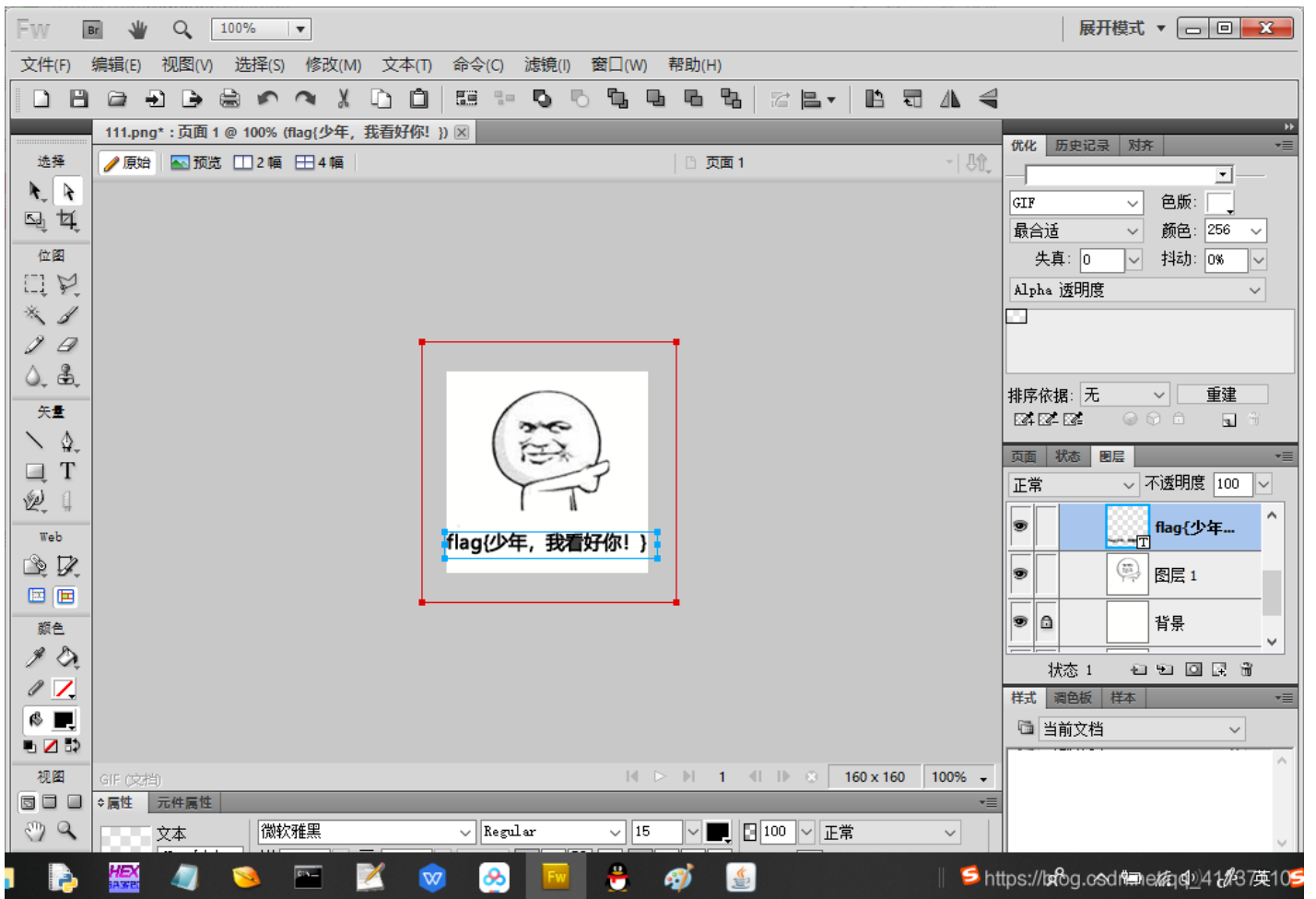
得到flag.jpg图片



4. 打开图片没有flag信息，用winhex（或010 Editor）打开查看16进制数据，在图片结尾（FFD9）后面还有内容，

00139024	00 AC C4 61 F8 63 4F D5 74 EB 4B 98 F5 BD 5C EA	.?殿鳩0?t?K?堇\?
00139040	B3 C9 73 24 91 CB F6 75 87 CB 89 8F CB 1E 17 AE	成s\$劉錫嘖? ? .?
00139056	39 E4 FA FA 0A 1B B8 1B B4 00 50 01 40 05 00 14	9? ..? ? P.@...
00139072	00 50 01 40 05 00 7F FF D9 38 42 50 53 00 01 00	.P.@... Ü8BPS...
00139088	00 00 00 00 00 00 03 00 00 01 90 00 00 01 90 00
00139104	08 00 03 00 00 00 00 00 00 71 44 38 42 49 4D 04 qD8BIM.
00139120	04 00 00 00 00 00 0F 1C 01 5A 00 03 1B 25 47 1C Z...%G.
00139136	02 00 00 02 00 00 00 38 42 49 4D 04 25 00 00 00 8BIM.%...
00139152	00 00 10 CD CF FA 7D A8 C7 BE 09 05 70 76 AE AF	...?销)?蓄..pv梨
00139168	05 C3 4E 38 42 49 4D 04 24 00 00 00 00 3C B1 3C	?N8BIM.\$....<?
00139184	3F 78 70 61 63 6B 65 74 20 62 65 67 69 6E 3D 22	?xpacket begin="

16进制头为38 42 50 53，根据文件头可以知道这是psd（Photoshop Document）文件，利用formost分离文件失败，直接把第一个FFD9后面的所有16进制数据复制成一个新文件，打开010Editor，复制保存为flag.psd文件。用photoshop打开flag.psd文件（用fireworks也行），打开后图片上有一行字flag{少年，我看好你！}



我以为这是flag,提交好几遍都错误。再观察图层发现有一个空白的背景

其实最顶层的文字是假的,这里关键在于锁定的“背景”层,看似是新建图片时所留下的默认背景图,而本题就是把flag隐藏在里面,把上面2层隐藏掉,然后对背景色另存为png格式(这样才能完好的保留颜色),使用stegsolve打开,并按下向左的按钮得到一个二维码,扫描得到flag

