

bugku--杂项WP

原创

[Hydra](#) 于 2018-11-18 22:36:46 发布 2140 收藏 9

分类专栏: [CTF—WriteUp](#)

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_40657585/article/details/83958514

版权



[CTF—WriteUp](#) 专栏收录该内容

10 篇文章 2 订阅

订阅专栏

目录

2.这是一张单纯的图片

3.隐写

4.telnet

5.眼见非实(ISCCCTF)

6.啊哒

7.又一张图片, 还单纯吗

8.猜

9.宽带信息泄露

10.隐写2

13.come_game

14.linux

15.隐写3

16. 做个游戏(08067CTF)

17.想蹭网先解开密码

18.Linux2

19.账号被盗了

21.爆照(08067CTF)

22.猫片(安恒)

23.多彩

24.旋转跳跃

25.普通的二维码

26.乌云邀请码

27. 神秘的文件

28.图穷匕现

29.convert

白哥的鸽子

论剑

1.签到题

签到题

50

关注微信公众号: Bugku
即可获得flag

下面也有二维码

qrcode_for_gh_...

https://blog.csdn.net/qq_40657585



微信扫码即可

2.这是一张单纯的图片

这是一张单纯的图片

50

<http://123.206.87.240:8002/misc/1.jpg>

FLAG在哪里？

Flag

Submit

https://blog.csdn.net/qq_40657585

用winhex打开，发现一串Unicode

WinHex - [1.jpg]

File Edit Search Navigation View Tools Specialist Options Window Help

Case Data

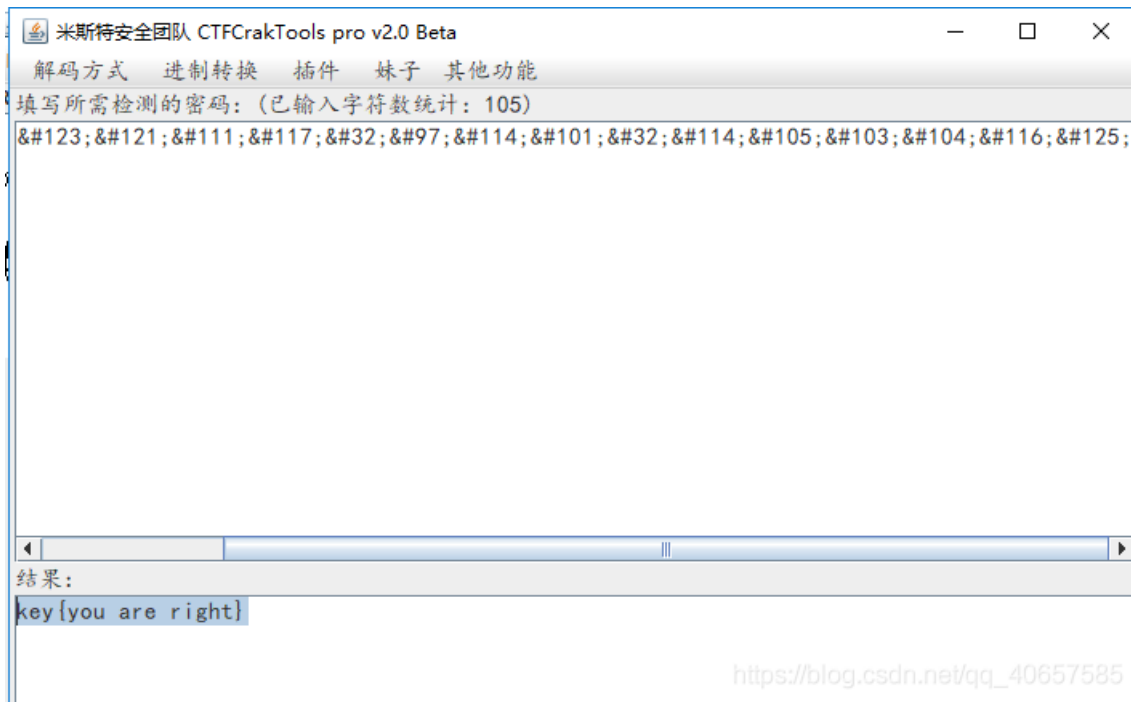
File Edit

1.jpg

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	ANSI ASCII
00001520	1D	A6	FB	B7	81	48	24	E5	66	66	05	B6	87	20	06	C7	!û·Hçáff ¶# Ç
00001530	CB	D4	1E	6B	E9	1F	08	FC	0B	F8	6D	6B	A4	43	3D	B6	ËÔ ké ü œmk=C=¶
00001540	94	BA	B2	5D	41	95	BC	BB	99	A4	32	A3	8C	86	00	61	"°:]A·4»»M2fG† a
00001550	14	E0	8C	32	A8	23	D7	35	D0	7C	6C	F0	7B	78	D7	E1	àQ2"×5D lô{x×á
00001560	F5	F6	99	02	EE	BB	46	5B	98	00	00	92	C8	4E	55	79	ôö" i»F[" 'ËNUy
00001570	1C	B2	96	51	92	06	58	64	E2	B9	1F	D9	5B	C4	D2	6A	"-Q' Xdá² Ú[ÃÖj
00001580	BE	01	97	41	D4	0B	8D	4F	C3	F3	9B	49	12	4C	87	11	% -AÔ OÃó>I L#
00001590	12	4A	64	1E	98	21	D0	0E	38	41	40	1E	B7	A3	69	56	Jd ~!D 8A@ ·fiV
000015A0	1A	26	9F	15	86	91	67	05	9D	9C	43	09	0C	28	15	47	éÿ t'g œC (G
000015B0	A9	E3	BE	79	27	A9	35	7E	8A	28	00	A2	8A	28	00	A2	€ãxy'€5~Š(ºŠ(º
000015C0	8A	28	00	A2	8A	28	00	A2	8A	28	00	AC	3F	1B	D9	41	Š(ºŠ(ºŠ(-? ÛA
000015D0	A8	F8	37	5D	B3	BB	B8	6B	5B	79	EC	67	8D	E7	54	DC	"ø7]"»·k[yàg çTÛ
000015E0	62	06	36	1B	C0	EE	57	A8	FA	0A	28	A0	0F	9F	BC	2D	b 6 ãiW"ú (Ÿ4-
000015F0	F0	EB	5A	F1	0F	8A	B4	5F	15	A5	90	B0	8B	4B	1A	72	øèZñ Š' _ ¥ °<K r
00001600	05	68	FC	A9	2E	A4	5B	85	7B	92	E1	B0	7E	40	D2	26	hü@.M[...'á°~@Ô&
00001610	71	F3	18	D4	2D	7D	3B	45	14	00	51	45	14	00	57	9E	qó Ô-);E QE WŽ
00001620	DA	78	3A	2D	0F	E2	C3	EB	FA	54	0D	0D	BE	AF	03	A5	Úx:- áãéúT *~ ¥
00001630	E7	95	1E	E5	33	0F	98	97	FE	EE	ED	AA	43	72	01	57	ç· á3 ~-pii°Cr W
00001640	1D	64	06	8A	28	03	D0	A8	A2	8A	00	28	A2	8A	00	28	d Š(D"ºŠ(ºŠ(
00001650	A2	8A	00	FF	26	23	31	30	37	3B	26	23	31	30	31	3B	ºŠ ýke
00001660	26	23	31	32	31	3B	26	23	31	32	33	3B	26	23	31	32	y{
00001670	31	3B	26	23	31	31	31	3B	26	23	31	31	37	3B	26	23	1;ou&#
00001680	33	32	3B	26	23	39	37	3B	26	23	31	31	34	3B	26	23	32;ar&#
00001690	31	30	31	3B	26	23	33	32	3B	26	23	31	31	34	3B	26	101; r&
000016A0	23	31	30	35	3B	26	23	31	30	33	3B	26	23	31	30	34	#105;gh
000016B0	3B	26	23	31	31	36	3B	26	23	31	32	35	3B	D9	D9		;t}ÛÜ

https://blog.csdn.net/qq_40657585

转ascii



3.隐写



下载下来之后是这样一张图



BU

https://blog.csdn.net/qq_40657585

bugku...是不是感觉隐藏的东西了，改一下高度，查案像素值

属性	值
来源	
拍摄日期	
图像	
分辨率	500 x 420
宽度	500 像素
高度	420 像素
位深度	32

用程序员计算器算一下 16进制



```

Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000 89 50 4E 47 0D 0A 1A 0A 00 00 00 0D 49 48 44 52 %PNG.....IHDR
00000010 00 00 01 F4 00 00 01 F4 08 06 00 00 00 CB D6 DF ..ö...ö.....ËÖß
00000020 8A 00 00 00 09 70 48 59 73 00 00 12 74 00 00 12 Š....pHYs...t...
00000030 74 01 DE 66 1F 78 00 00 0A 4D 69 43 43 50 50 68 t.řf.x...MiCCPPh
00000040 6F 74 6F 73 68 6F 70 20 49 43 43 20 70 72 6F 66 otoshop ICC prof
00000050 69 6C 65 00 00 78 DA 9D 53 77 58 93 F7 16 3E DF ile..xŮ.SwX"+.>ß
00000060 F7 65 0F 56 42 D8 F0 B1 97 6C 81 00 22 23 AC 08 ÷e.VBØð±-l.."#-..
00000070 C8 10 59 A2 10 92 00 61 84 10 12 40 C5 85 88 0A È.Yç.'a,,...@Ă...^
00000080 56 14 15 11 9C 48 55 C4 82 D5 0A 48 9D 88 E2 A0 V...œHUĂ,Ŏ.H.^á
00000090 28 B8 67 41 8A 88 5A 8B 55 5C 38 EE 1F DC A7 B5 (,gĂŠ^Z<U\8i.Ůŝu
000000A0 7D 7A EF ED ED FB D7 FB BC E7 9C E7 FC CE 79 CF }ziiü*Ů+çœçüÿÿ
000000B0 0F 80 11 12 26 91 E6 A2 6A 00 39 52 85 3C 3A D8 .€..&'æçj.9R...:Ø
000000C0 1F 8F 4F 48 C4 C9 BD 80 02 15 48 E0 04 20 10 E6 ..OHĂĚ>€..Hà. .æ
000000D0 CB C2 67 05 C5 00 00 F0 03 79 78 7E 74 B0 3F FC ĚĂg.Ă..đ.yx~t°?ü
000000E0 01 AF 6F 00 02 00 70 D5 2E 24 12 C7 E1 FF 83 BA .Ů...pŎ.ŝ.Çáÿf°
000000F0 50 26 57 00 20 91 00 E0 22 12 E7 0B 01 90 52 00 P&W. `à".ç...R.
00000100 C8 2E 54 C8 14 00 C8 18 00 B0 53 B3 64 0A 00 94 È.TÈ..È..°S'd.."
00000110 00 00 6C 79 7C 42 22 00 AA 0D 00 EC F4 49 3E 05 ..ly|B".*...iđI>.
00000120 00 D8 A9 93 DC 17 00 D8 A2 1C A9 08 00 8D 01 00 .Ø@"Ů..Øç.©.....
00000130 99 28 47 24 02 40 BB 00 60 55 81 52 2C 02 C0 C2 " (Gŝ.©». `U.R.,ĂĂ
00000140 00 A0 AC 40 22 2E 04 C0 AE 01 80 59 B6 32 47 02 dn.n~@`u.Ă@.€Yſ2G5
00000150 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

```

这里我把高度改成了和宽度一样，保存，再次打开



4.telnet


```
#!/usr/bin/python
# -*- coding:utf8 -
import binascii

text = "d87 x65 x6c x63 o157 d109 o145 b100000 d116 b1101111 o40 x6b b1100101 b1101100 o141 d105 x62 d101 b
solution = ''

text2 = text.split(' ')
for x in text2:
    print(x)
    if x[0] == 'b': #binary
        solution += chr(int(x[1:],2))
    elif x[0] == 'x': # hexadecimal
        solution += chr(int(x[1:],16)) #python3中没有decode方法
    # elif x[0] == 'x':
    #     solution += x[1:].decode("hex")
    elif x[0] == 'd':
        solution += chr(int(x[1:]))
    elif x[0] == 'o':
        solution += chr(int(x[1:],8))
print(solution)
```

捕捉到一个flag

```
d111
x64
d32
o164
b1101001
x6d
o145
x7e
Welcome to kelaibei. Give you a flag as a gift. flag{1e4bf81a6394de5abc005ac6e39a387b} . Have a good time~
https://blog.csdn.net/qq\_40657585
```

6.啊哒

啊哒!



一张图片，看属性



十六进制转换

```
p = "73646E6973635F32303138"  
s = []  
for i in range(0, len(p), 2):  
    b = p[i:i+2]  
    s +=chr(int(b, 16))  
print(''.join(s))
```

```
sdnisc_2018
```

应该是密码什么的

binwalk一下果然发现其他东西

foremost分离出一个压缩包

输入sdnisc_2018解压

00000427.zip	2018/11/18 11:23	好压 ZIP 压缩文件	1 KB
flag.txt	2018/10/18 15:44	文本文档	1 KB

得到 flag

7.又一张图片，还单纯吗



另存为，查看属性，啥都没有，binwalk发现东西，foremost分离一下

```
选中了"00000310.jpg" (27.7 KB)
158822 0x26C66 TIFF image data, big-endian, offset of first image
directory: 8
159124 0x26D94 JPEG image data, JFIF standard 1.02
162196 0x27994 JPEG image data, JFIF standard 1.02
164186 0x2815A Unix path: /www.w3.org/1999/02/22-rdf-syntax-ns#">
<rdf:Description rdf:about="" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns
:xap="htt
168370 0x291B2 Copyright string: "Copyright (c) 1998 Hewlett-Pack
ard Company"
root@kali:~/Desktop/MISC# foremost 2.jpg
ERROR: /root/Desktop/MISC/output is not empty
Please specify another directory or run with -T.
root@kali:~/Desktop/MISC# foremost 2.jpg -T
Processing: 2.jpg
|*|
root@kali:~/Desktop/MISC#
```

发现flag

8.猜



这个还用猜，刘亦菲

百度了一下别人的，百度识图



图中可能是 义彩明星周边影视刘亦菲神雕侠侣小

把图片拖拽到此区域 本地上传 粘贴网址

相关商品

[礼品](#) [图书](#)



45.00

义彩 明星周边影视刘亦菲神雕侠侣小龙女方形抱枕
柔软PP棉同学朋友生日礼物来图定制 刘亦菲主题抱
京东商城

https://blog.csdn.net/qq_40657585

9.宽带信息泄露

宽带信息泄露

60

flag格式:

flag{宽带用户名}

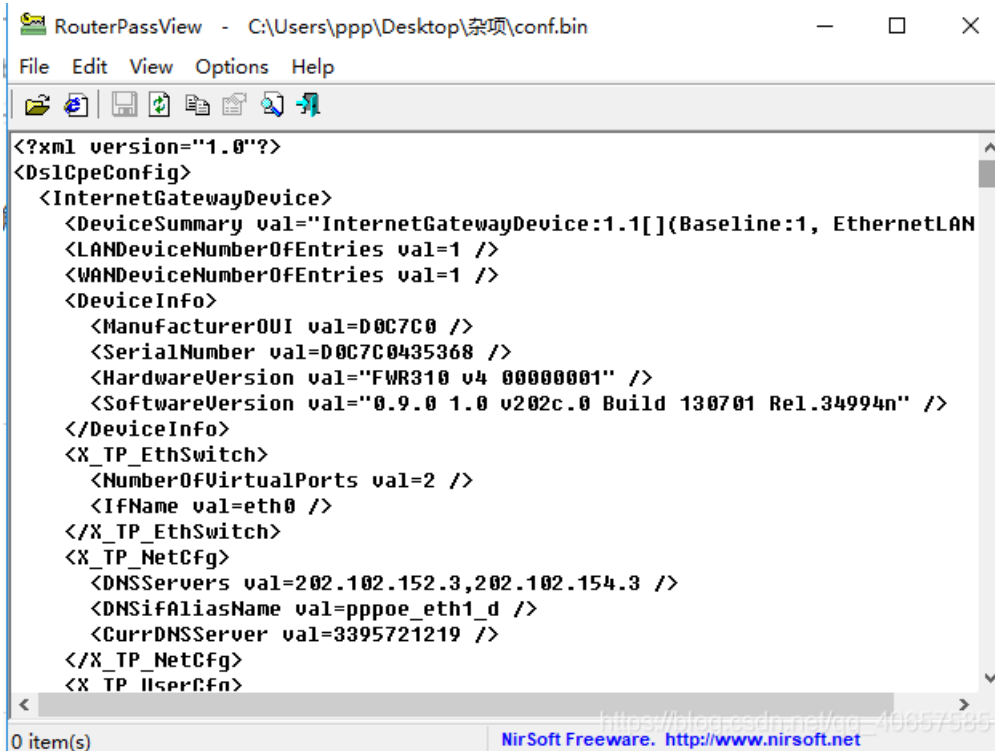
conf.bin

Flag

Submit

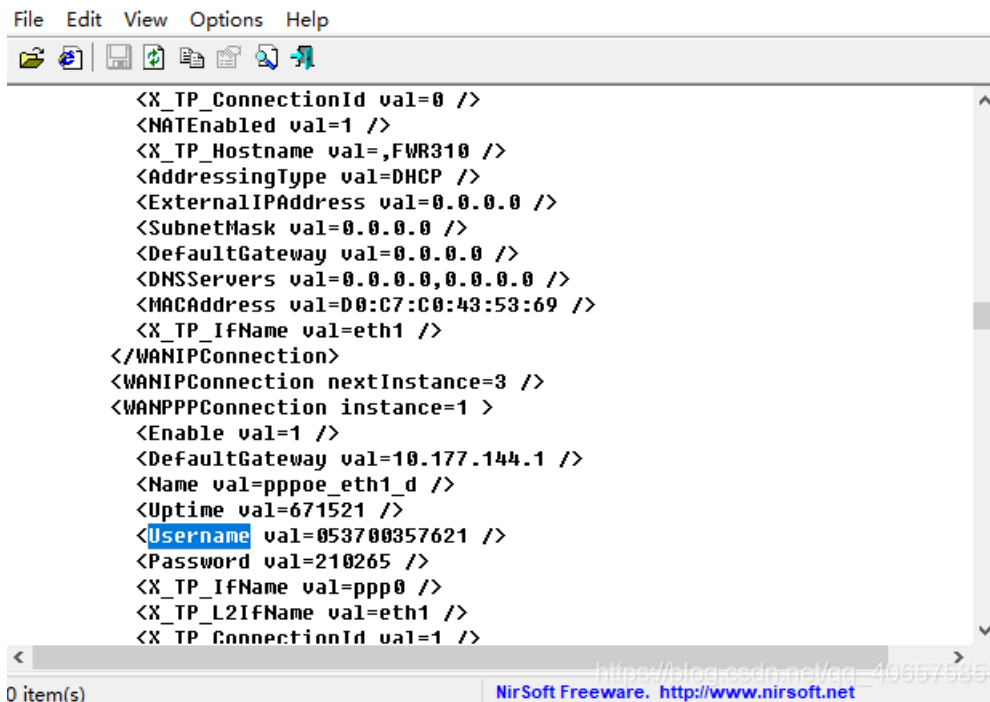
https://blog.csdn.net/qq_40657585

打开之后是乱码，被加密了，到网上寻找该配置程序的解密工具，找到一款名为RouterPassView的工具，使用该工具打开.bin文件，如下所示：



```
<?xml version="1.0"?>
<Ds1CpeConfig>
  <InternetGatewayDevice>
    <DeviceSummary val="InternetGatewayDevice:1.1[(Baseline:1, EthernetLAN
    <LANDeviceNumberOfEntries val=1 />
    <WANDeviceNumberOfEntries val=1 />
    <DeviceInfo>
      <ManufacturerOUI val=D0C7C0 />
      <SerialNumber val=D0C7C0435368 />
      <HardwareVersion val="FWR310 v4 00000001" />
      <SoftwareVersion val="0.9.0 1.0 v202c.0 Build 130701 Re1.34994n" />
    </DeviceInfo>
    <X_TP_EthSwitch>
      <NumberOfVirtualPorts val=2 />
      <IfName val=eth0 />
    </X_TP_EthSwitch>
    <X_TP_NetCfg>
      <DNSServers val=202.102.152.3,202.102.154.3 />
      <DNSifAliasName val=pppoe_eth1_d />
      <CurrDNSServer val=3395721219 />
    </X_TP_NetCfg>
    <X_TP_IUserCfn>
```

xml文件，题目说的是用户名，搜一下username



```
<X_TP_ConnectionId val=0 />
<NATEnabled val=1 />
<X_TP_Hostname val=,FWR310 />
<AddressingType val=DHCP />
<ExternalIPAddress val=0.0.0.0 />
<SubnetMask val=0.0.0.0 />
<DefaultGateway val=0.0.0.0 />
<DNSServers val=0.0.0.0,0.0.0.0 />
<MACAddress val=D0:C7:C0:43:53:69 />
<X_TP_IfName val=eth1 />
</WANIPConnection>
<WANIPConnection nextInstance=3 />
<WANPPPConnection instance=1 >
  <Enable val=1 />
  <DefaultGateway val=10.177.144.1 />
  <Name val=pppoe_eth1_d />
  <Uptime val=671521 />
  <Username val=053700357621 />
  <Password val=210265 />
  <X_TP_IfName val=ppp0 />
  <X_TP_L2IfName val=eth1 />
  <X_TP_ConnectionId val=1 />
```

10.隐写2



想拿到flag？心中ないいくつかB数かの？

右键另存为，binwalk，foremost，发现压缩包，有提示

告诉你们一个秘密，密码是3个数哦。

查理曼：

查理曼，法兰克王国国王，征服了西欧与中欧大部分土地，具有了至高无上的权威，下令全国人民信仰基督教，查理重振了西罗马帝国。

雅典娜：

女神帕拉斯·雅典娜，是希腊神话中的女战神也是智慧女神，雅典是以她命名的。

兰斯洛特，

英格兰传说中的人物，是亚瑟王圆桌骑士团中的一员。看上去就是一个清秀年轻的帅小伙儿，由于传说中他是一名出色的箭手，因此梅花J手持箭支。兰斯洛特与王后的恋爱导致了他与亚瑟王之间的战争。

Hint:

其实斗地主挺好玩的。

https://blog.csdn.net/qq_40657585

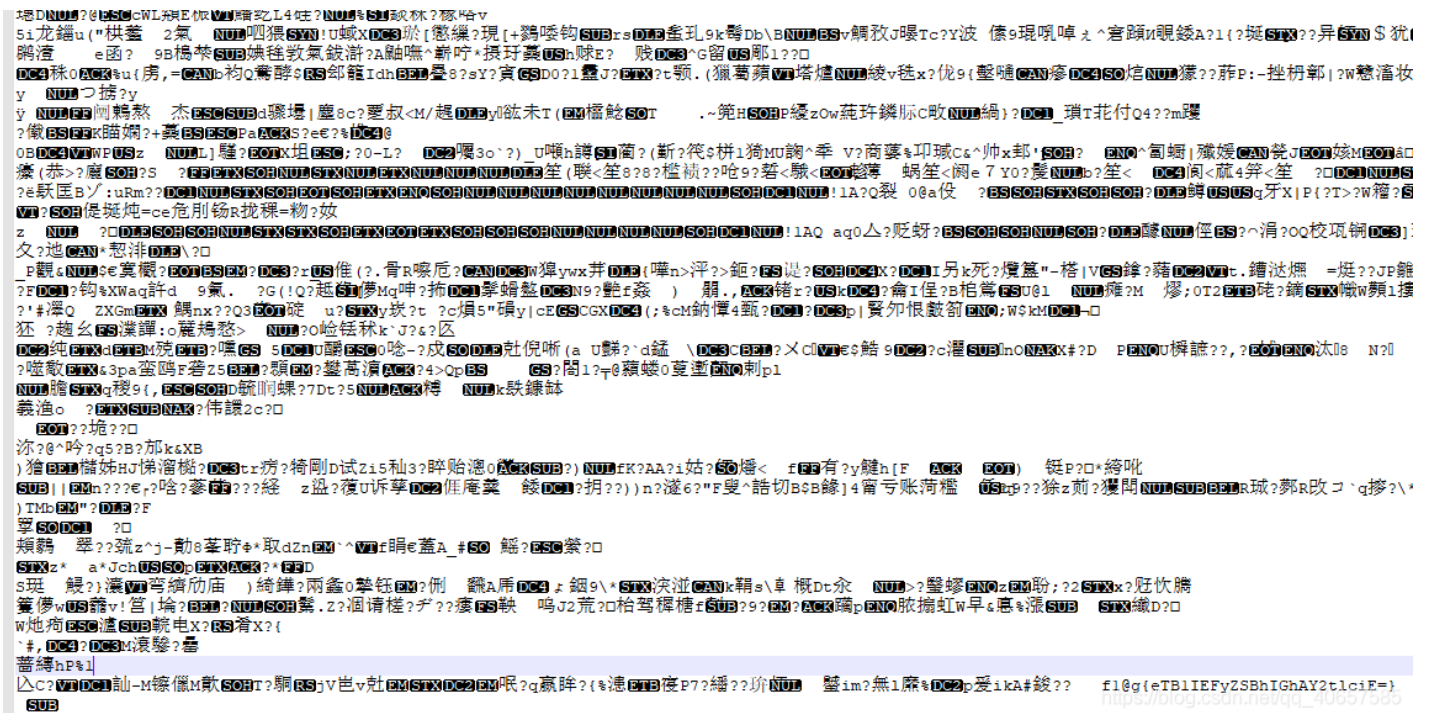
爆破白



解压



右键 notepad++ 打开



得到 f1@g{eTB1IEFyZSBhIGhAY2tlciE=}, base64解码, @变为a

13.come_game

Challenge

203 Solves

×

come_game

60

听说游戏通关就有flag

题目来源：第七季极客大挑战

game_1.zip

Flag

Submit

https://blog.csdn.net/qq_40657585



靠玩，能得到flag的请私聊我一下，泪

MACOSX	2016/10/9 23:47	文件夹	
DeathTime	2018/11/18 18:31	文件	1 KB
joker's.exe	2016/10/9 23:45	应用程序	8,106 KB
save1	2018/11/18 17:02	文件	1 KB
save1.bak	2018/11/18 16:55	BAK 文件	1 KB
save3	2018/11/18 18:29	文件	1 KB
temp	2018/11/18 18:31	文件	1 KB

发现有存档，

```

文件(F) 编辑(E) 搜索(S) 查看(V) 分析(A) 附加(X) 窗口(W) 关于(A)
16 ANSI 十六进制
save3
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000 00 01 32 00 00 41 00 05 43 00 00 00 00 00 00 00 ..2..A..C....
00000010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...|.

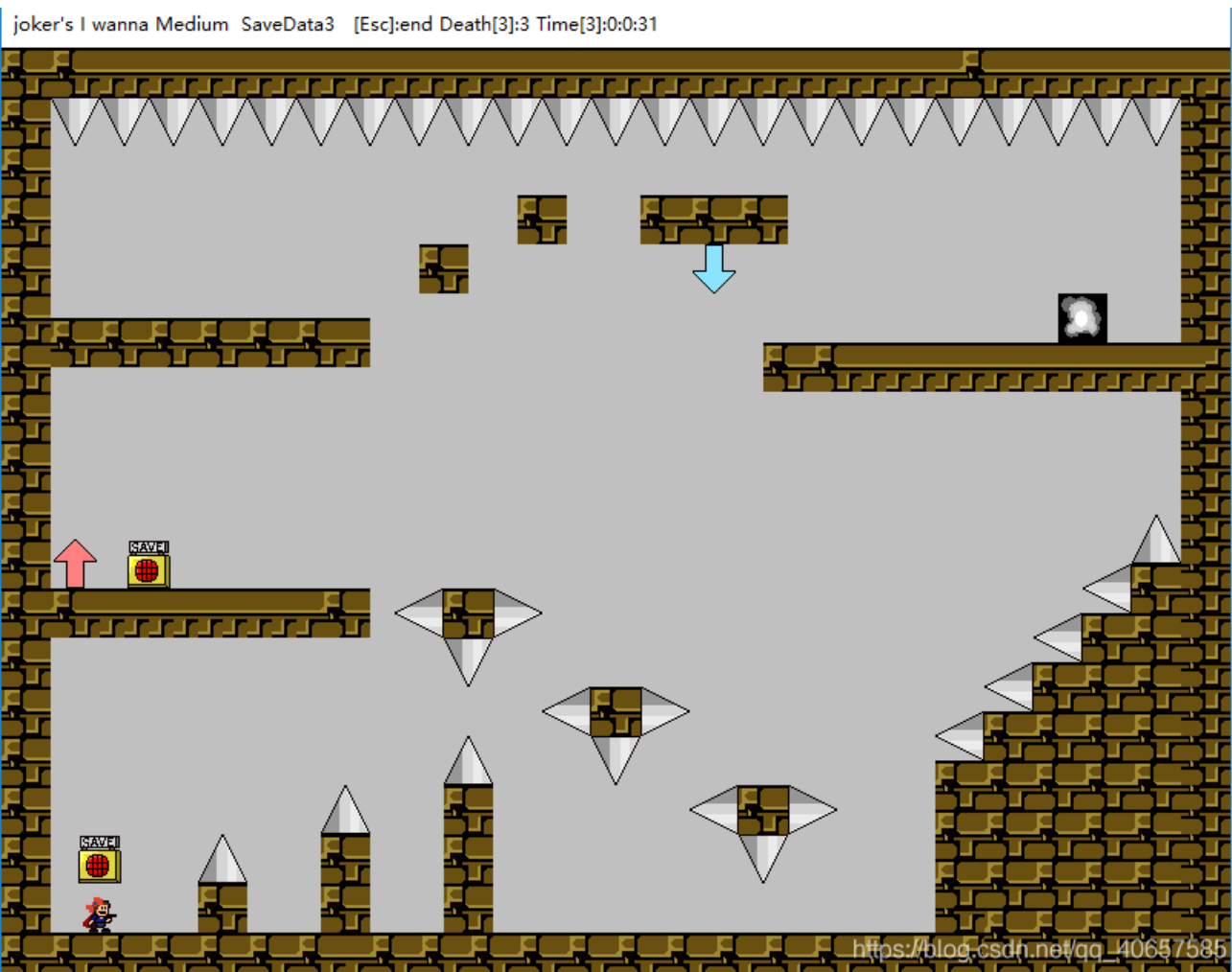
```

发现一个2,修改为3,

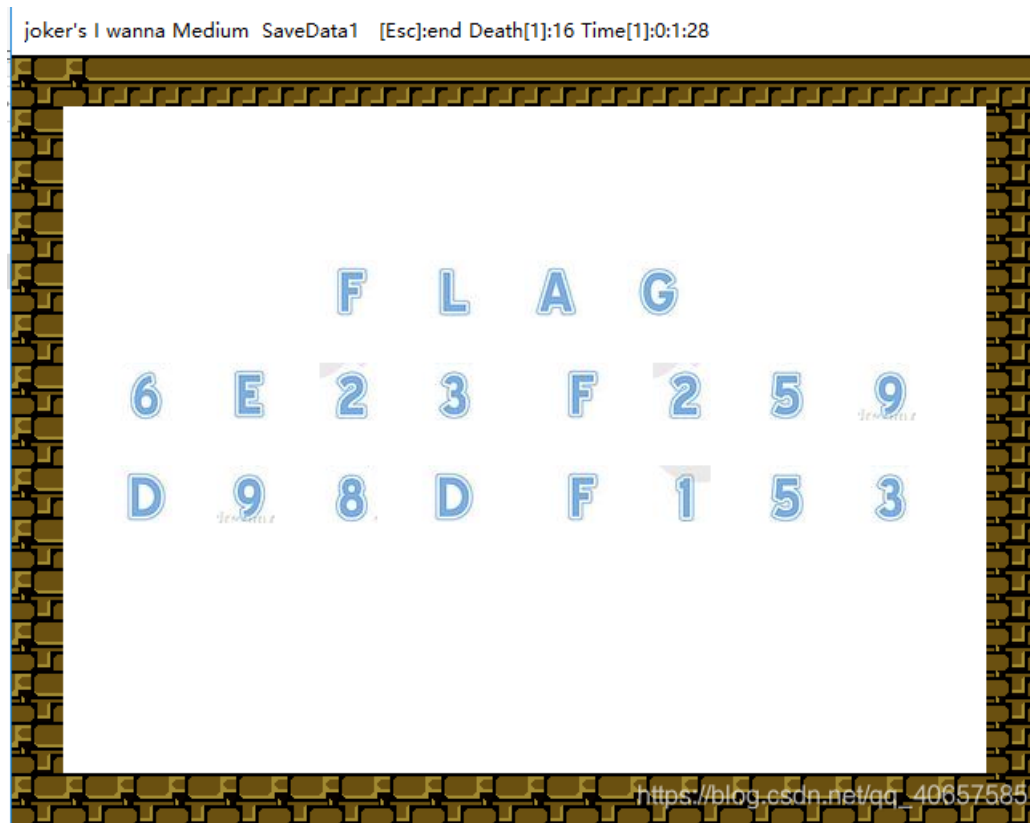
```

save3
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000 00 01 33 00 00 41 00 05 43 00 00 00 00 00 00 00 ..3..A..C.....
00000010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....

```



发现新关卡，依次修改为4,5，得到flag



14.linux

解压得到flag

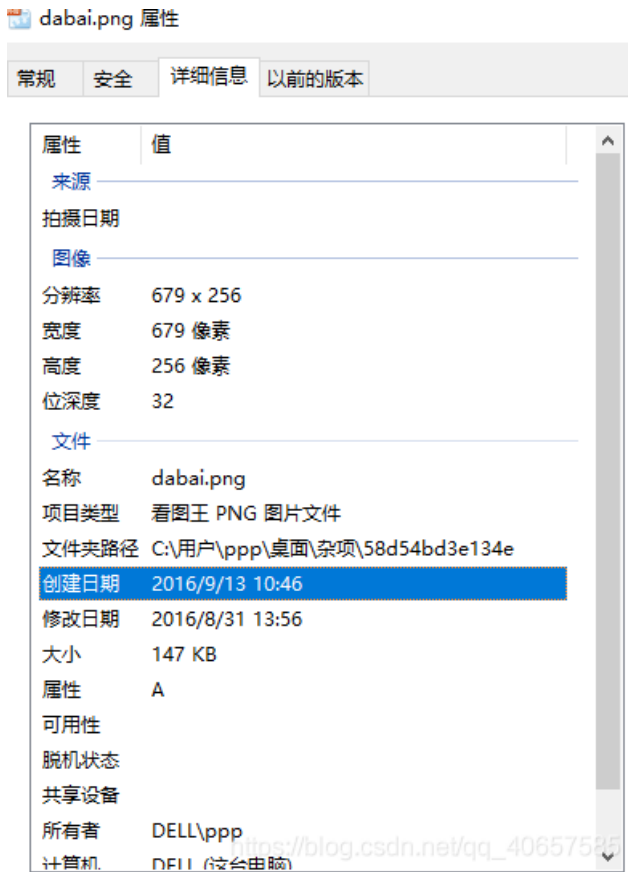
strings flag

```
root@kali: ~/Desktop/MISC/output
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
flag.txt
.goutputstream-LSCRJY
lost+found
game
.Trash-0
flag.txt
.goutputstream-LSCRJY
[WI/
xy#^
/media/test
[WI/
root@kali: ~/Desktop/MISC/output
root@kali:~/Desktop/MISC/output# foremost flag
info sing: flag
files
[Trash Info]
Path=game
DeletionDate=2016-06-27T12:27:37
key{}
key{}
key{feb81d3834e2423c9903f4755464060b}
game.trashinfo
game.trashinfo.0L1PJY
game
root@kali:~/Desktop/MISC/output#
```

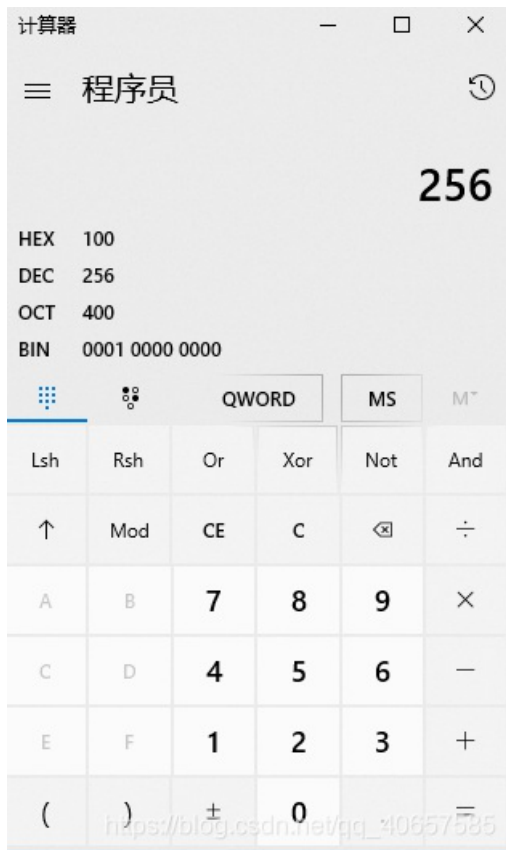
15.隐写3



感觉没显示完整，查看一下属性



计算一下16进制



将01修改为02

```

dabai.png
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000 89 50 4E 47 0D 0A 1A 0A 00 00 00 0D 49 48 44 52 %PNG.....IHDR
00000001 00 00 02 A7 00 00 02 00 08 06 00 00 00 6D 7C 71 ...S...m|q
00000002 35 00 00 00 01 73 52 47 42 00 AE CE 1C E9 00 00 5....sRGB.@I.e..
00000003 00 04 67 41 4D 41 00 00 B1 8F 0B FC 61 05 00 00 ..gAMA..t..ua...
00000004 00 09 70 48 59 73 00 00 0E C4 00 00 0E C4 01 95 ..pHYs...A...A.*
00000005 2B 0E 1B 00 00 FF A5 49 44 41 54 78 5E EC BD 07 +....ÿIDATx`i%.
00000006 A0 A5 57 59 EE FF EE BE 4F 9B DE 93 4C 7A 0F 84 %WYiyi%O>P"Lz..
00000007 24 24 60 0C 04 A5 2B 20 45 10 10 BB 88 8A A8 57 $$`.ÿ+ E..»^Š`W
00000008 BD FC EF BD 7A F5 5A AE 7A BD 5E CB BD 2A 62 05 %üi%szöZ@z%`E%*b.
00000009 04 69 52 04 E9 01 42 48 48 42 7A EF 7D 52 A6 CF .iR.é.BHHBzi)R;I
0000000A 9C 7E 76 FD 3F BF F7 DB EF 39 6B 76 F6 4C 26 C9 æ~vý?ç=Ü19kvöL&É
0000000B 4C 32 E5 7B CE 59 7B F5 DE 9E 6F 7D 6B AD AF D0 L2â{ÏY{öPžo}k.¯Ð
0000000C 15 2C 47 8E 1C 39 72 1C 90 60 88 2E 14 0A 3D DD .,GŽ.9r..`^...=ÿ
0000000D DE 63 6F FD A5 53 C0 93 8D A7 D3 E9 F4 54 66 C5 %coy%SA`.şÓéôTfÅ
0000000E 62 D1 E5 34 BC 34 0D AD 56 CB 1A 8D 86 35 9B 4D bÑâ4+4..VĚ..+5>M
0000000F 17 B3 B3 B3 36 37 37 E7 72 98 21 70 87 DC 6E B7 .''677qr"!p+Ûn·
00000100 3D FC 90 01 E1 11 0F 61 22 CA E5 B2 8B 4A A5 62 =ü..á..a"Êâ<J%b
00000101 A5 52 C9 D5 B5 5A CD AA D5 AA CB 88 7A BD EE 22 %RÉÖµZÍ*Ö*È`z%i"
00000102 DC 45 3A FB 11 E9 24 3E 80 1E B7 91 87 34 2F 81 ÜE:ù.é$>E..`+4/.
00000103 30 8B F4 A1 DE 9D DB 7E F4 BB D9 1B 3F 39 72 1C 0<ô;P.Û~ô»Û.¿9r.
00000104 0E D8 3E D9 B4 9C 9C E6 C8 91 23 C7 41 8C 18 C2 .Ø>Û`æææE`#ÇAE.Å
00000105 FB 89 4D 6A 9E 12 1F D4 88 7E 32 99 CA B8 1D 14 û%Mjž..Ô~2mĚ...
00000106 5E 6A D6 6F 0F A1 1C 1F B7 6D DB B6 D9 8E 1D ^jÖo.;...mÛqÛž.
00000107 3B 6C E7 CE 9D 2E 63 36 3D 3D 6D 93 93 93 2E 26 ;lçİ..c6==m"".&
00000108 26 26 9C 90 4E 4D 4D CD 93 53 08 21 E1 47 1C 91 &&æ.NMMÍ"S.!áG.`
00000109 16 D4 29 31 45 40 42 43 1F 72 10 D6 C8 13 6A 08 .Ô)lE@BC.r.ÖĚ.j.
0000010A E9 C8 C8 88 13 D4 E1 E1 61 5B BC 78 B1 9B 2D 5A éĚĚ^..Ôááa[+x+>-Z
0000010B B4 C8 C6 C6 C6 5C 3F 3A 3A EA F2 F2 E5 CB 6D D9 `ĚĚĚ[?::êòòâĚmÛ
0000010C B2 65 EE 1E 12 3B 08 FD F1 01 D2 16 E9 45 44 1A %ei...;ÿñ.Ô.éED.
0000010D 30 47 0E 3F 20 DC F5 BB 0F 3B 10 E1 85 3E 47 8E 0G.? Üö».;.á...GŽ
0000010E C3 11 39 39 CD 91 23 47 8E 83 10 29 B1 49 D5 20 Å.99Í`#GŽf.)±IÖ
0000010F F4 31 B4 A7 EE 52 A4 7A DC 04 A1 42 86 00 62 8F ôl`šîRwzÛ.;B+.b.
00000200 9A 95 4C 48 24 82 95 CF 99 99 19 DB B4 69 93 6D š.LHš,.İ"m.Û`i`m
00000201 D9 B2 C5 65 04 24 14 02 8A E8 5F 11 8D F0 22 1D Û`Åe.$..šë...8".
00000202 A1 0E 01 06 C9 08 D2 D0 9F 8F 90 83 C8 61 1F 72 ;...É.Öðÿ..fĚa.r
00000203 00 38 44 88 71 0F 84 1A BB 58 61 85 B4 0F 0D 0D .D`..V%`

```

得到flag



16. 做个游戏(08067CTF)



使用java反编译工具反编译一下

heiheihei.jar

MANIFEST.MF

cn.bjstx

plane

- Bullet.class
- Bullet
- Explode.class
- Explode
- GameObject.class
- GameObject
- Plane.class
- PlaneGameFrame.class

util

- Constant.class
- GameUtil.class
- MyFrame.class

images

- explode
- ball.png
- bg.jpg
- plane.png

```

54     printInfo(g, "你的持久度才" + period + "秒", 50, 150, 250);
56     switch (period / 10)
57     {
58     case 0:
59         printInfo(g, "真.头顶一片青青草原", 50, 150, 300);
60         break;
61     case 1:
62         printInfo(g, "这东西你也要抢着带?", 50, 150, 300);
63         break;
64     case 2:
65         printInfo(g, "如果梦想有颜色, 那一定是原谅色", 40, 30, 300);
66         break;
67     case 3:
68         printInfo(g, "哟, 炊事班长呀兄弟", 50, 150, 300);
69         break;
70     case 4:
71         printInfo(g, "加油你就是下一个老王", 50, 150, 300);
72         break;
73     case 5:
74         printInfo(g, "如果撑过一分钟我岂不是很没面子", 40, 30, 300);
75         break;
76     case 6:
77         printInfo(g, "flag{RGFqaURhbG1fSmlud2FuQ2hpamk=}", 50, 150, 300);
78         break;
79     }
80     }
81     }
82     }
83     }
84     }
85     }
86     }
87     }
88     }
89     }
90     }
91     }
92     }
93     }
94     }
95     }
96     }
97     }
98     }
99     }
100    }
101    }
102    }
103    }
104    }
105    }
106    }
107    }
108    }
109    }
110    }
111    }
112    }
113    }
114    }
115    }
116    }
117    }
118    }
119    }
120    }
121    }
122    }
123    }
124    }
125    }
126    }
127    }
128    }
129    }
130    }
131    }
132    }
133    }
134    }
135    }
136    }
137    }
138    }
139    }
140    }
141    }
142    }
143    }
144    }
145    }
146    }
147    }
148    }
149    }
150    }
151    }
152    }
153    }
154    }
155    }
156    }
157    }
158    }
159    }
160    }
161    }
162    }
163    }
164    }
165    }
166    }
167    }
168    }
169    }
170    }
171    }
172    }
173    }
174    }
175    }
176    }
177    }
178    }
179    }
180    }
181    }
182    }
183    }
184    }
185    }
186    }
187    }
188    }
189    }
190    }
191    }
192    }
193    }
194    }
195    }
196    }
197    }
198    }
199    }
200    }
201    }
202    }
203    }
204    }
205    }
206    }
207    }
208    }
209    }
210    }
211    }
212    }
213    }
214    }
215    }
216    }
217    }
218    }
219    }
220    }
221    }
222    }
223    }
224    }
225    }
226    }
227    }
228    }
229    }
230    }
231    }
232    }
233    }
234    }
235    }
236    }
237    }
238    }
239    }
240    }
241    }
242    }
243    }
244    }
245    }
246    }
247    }
248    }
249    }
250    }
251    }
252    }
253    }
254    }
255    }
256    }
257    }
258    }
259    }
260    }
261    }
262    }
263    }
264    }
265    }
266    }
267    }
268    }
269    }
270    }
271    }
272    }
273    }
274    }
275    }
276    }
277    }
278    }
279    }
280    }
281    }
282    }
283    }
284    }
285    }
286    }
287    }
288    }
289    }
290    }
291    }
292    }
293    }
294    }
295    }
296    }
297    }
298    }
299    }
300    }
301    }
302    }
303    }
304    }
305    }
306    }
307    }
308    }
309    }
310    }
311    }
312    }
313    }
314    }
315    }
316    }
317    }
318    }
319    }
320    }
321    }
322    }
323    }
324    }
325    }
326    }
327    }
328    }
329    }
330    }
331    }
332    }
333    }
334    }
335    }
336    }
337    }
338    }
339    }
340    }
341    }
342    }
343    }
344    }
345    }
346    }
347    }
348    }
349    }
350    }
351    }
352    }
353    }
354    }
355    }
356    }
357    }
358    }
359    }
360    }
361    }
362    }
363    }
364    }
365    }
366    }
367    }
368    }
369    }
370    }
371    }
372    }
373    }
374    }
375    }
376    }
377    }
378    }
379    }
380    }
381    }
382    }
383    }
384    }
385    }
386    }
387    }
388    }
389    }
390    }
391    }
392    }
393    }
394    }
395    }
396    }
397    }
398    }
399    }
400    }
401    }
402    }
403    }
404    }
405    }
406    }
407    }
408    }
409    }
410    }
411    }
412    }
413    }
414    }
415    }
416    }
417    }
418    }
419    }
420    }
421    }
422    }
423    }
424    }
425    }
426    }
427    }
428    }
429    }
430    }
431    }
432    }
433    }
434    }
435    }
436    }
437    }
438    }
439    }
440    }
441    }
442    }
443    }
444    }
445    }
446    }
447    }
448    }
449    }
450    }
451    }
452    }
453    }
454    }
455    }
456    }
457    }
458    }
459    }
460    }
461    }
462    }
463    }
464    }
465    }
466    }
467    }
468    }
469    }
470    }
471    }
472    }
473    }
474    }
475    }
476    }
477    }
478    }
479    }
480    }
481    }
482    }
483    }
484    }
485    }
486    }
487    }
488    }
489    }
490    }
491    }
492    }
493    }
494    }
495    }
496    }
497    }
498    }
499    }
500    }
501    }
502    }
503    }
504    }
505    }
506    }
507    }
508    }
509    }
510    }
511    }
512    }
513    }
514    }
515    }
516    }
517    }
518    }
519    }
520    }
521    }
522    }
523    }
524    }
525    }
526    }
527    }
528    }
529    }
530    }
531    }
532    }
533    }
534    }
535    }
536    }
537    }
538    }
539    }
540    }
541    }
542    }
543    }
544    }
545    }
546    }
547    }
548    }
549    }
550    }
551    }
552    }
553    }
554    }
555    }
556    }
557    }
558    }
559    }
560    }
561    }
562    }
563    }
564    }
565    }
566    }
567    }
568    }
569    }
570    }
571    }
572    }
573    }
574    }
575    }
576    }
577    }
578    }
579    }
580    }
581    }
582    }
583    }
584    }
585    }
586    }
587    }
588    }
589    }
590    }
591    }
592    }
593    }
594    }
595    }
596    }
597    }
598    }
599    }
600    }
601    }
602    }
603    }
604    }
605    }
606    }
607    }
608    }
609    }
610    }
611    }
612    }
613    }
614    }
615    }
616    }
617    }
618    }
619    }
620    }
621    }
622    }
623    }
624    }
625    }
626    }
627    }
628    }
629    }
630    }
631    }
632    }
633    }
634    }
635    }
636    }
637    }
638    }
639    }
640    }
641    }
642    }
643    }
644    }
645    }
646    }
647    }
648    }
649    }
650    }
651    }
652    }
653    }
654    }
655    }
656    }
657    }
658    }
659    }
660    }
661    }
662    }
663    }
664    }
665    }
666    }
667    }
668    }
669    }
670    }
671    }
672    }
673    }
674    }
675    }
676    }
677    }
678    }
679    }
680    }
681    }
682    }
683    }
684    }
685    }
686    }
687    }
688    }
689    }
690    }
691    }
692    }
693    }
694    }
695    }
696    }
697    }
698    }
699    }
700    }
701    }
702    }
703    }
704    }
705    }
706    }
707    }
708    }
709    }
710    }
711    }
712    }
713    }
714    }
715    }
716    }
717    }
718    }
719    }
720    }
721    }
722    }
723    }
724    }
725    }
726    }
727    }
728    }
729    }
730    }
731    }
732    }
733    }
734    }
735    }
736    }
737    }
738    }
739    }
740    }
741    }
742    }
743    }
744    }
745    }
746    }
747    }
748    }
749    }
750    }
751    }
752    }
753    }
754    }
755    }
756    }
757    }
758    }
759    }
760    }
761    }
762    }
763    }
764    }
765    }
766    }
767    }
768    }
769    }
770    }
771    }
772    }
773    }
774    }
775    }
776    }
777    }
778    }
779    }
780    }
781    }
782    }
783    }
784    }
785    }
786    }
787    }
788    }
789    }
790    }
791    }
792    }
793    }
794    }
795    }
796    }
797    }
798    }
799    }
800    }
801    }
802    }
803    }
804    }
805    }
806    }
807    }
808    }
809    }
810    }
811    }
812    }
813    }
814    }
815    }
816    }
817    }
818    }
819    }
820    }
821    }
822    }
823    }
824    }
825    }
826    }
827    }
828    }
829    }
830    }
831    }
832    }
833    }
834    }
835    }
836    }
837    }
838    }
839    }
840    }
841    }
842    }
843    }
844    }
845    }
846    }
847    }
848    }
849    }
850    }
851    }
852    }
853    }
854    }
855    }
856    }
857    }
858    }
859    }
860    }
861    }
862    }
863    }
864    }
865    }
866    }
867    }
868    }
869    }
870    }
871    }
872    }
873    }
874    }
875    }
876    }
877    }
878    }
879    }
880    }
881    }
882    }
883    }
884    }
885    }
886    }
887    }
888    }
889    }
890    }
891    }
892    }
893    }
894    }
895    }
896    }
897    }
898    }
899    }
900    }
901    }
902    }
903    }
904    }
905    }
906    }
907    }
908    }
909    }
910    }
911    }
912    }
913    }
914    }
915    }
916    }
917    }
918    }
919    }
920    }
921    }
922    }
923    }
924    }
925    }
926    }
927    }
928    }
929    }
930    }
931    }
932    }
933    }
934    }
935    }
936    }
937    }
938    }
939    }
940    }
941    }
942    }
943    }
944    }
945    }
946    }
947    }
948    }
949    }
950    }
951    }
952    }
953    }
954    }
955    }
956    }
957    }
958    }
959    }
960    }
961    }
962    }
963    }
964    }
965    }
966    }
967    }
968    }
969    }
970    }
971    }
972    }
973    }
974    }
975    }
976    }
977    }
978    }
979    }
980    }
981    }
982    }
983    }
984    }
985    }
986    }
987    }
988    }
989    }
990    }
991    }
992    }
993    }
994    }
995    }
996    }
997    }
998    }
999    }
1000   }
1001   }
1002   }
1003   }
1004   }
1005   }
1006   }
1007   }
1008   }
1009   }
1010   }
1011   }
1012   }
1013   }
1014   }
1015   }
1016   }
1017   }
1018   }
1019   }
1020   }
1021   }
1022   }
1023   }
1024   }
1025   }
1026   }
1027   }
1028   }
1029   }
1030   }
1031   }
1032   }
1033   }
1034   }
1035   }
1036   }
1037   }
1038   }
1039   }
1040   }
1041   }
1042   }
1043   }
1044   }
1045   }
1046   }
1047   }
1048   }
1049   }
1050   }
1051   }
1052   }
1053   }
1054   }
1055   }
1056   }
1057   }
1058   }
1059   }
1060   }
1061   }
1062   }
1063   }
1064   }
1065   }
1066   }
1067   }
1068   }
1069   }
1070   }
1071   }
1072   }
1073   }
1074   }
1075   }
1076   }
1077   }
1078   }
1079   }
1080   }
1081   }
1082   }
1083   }
1084   }
1085   }
1086   }
1087   }
1088   }
1089   }
1090   }
1091   }
1092   }
1093   }
1094   }
1095   }
1096   }
1097   }
1098   }
1099   }
1100   }
1101   }
1102   }
1103   }
1104   }
1105   }
1106   }
1107   }
1108   }
1109   }
1110   }
1111   }
1112   }
1113   }
1114   }
1115   }
1116   }
1117   }
1118   }
1119   }
1120   }
1121   }
1122   }
1123   }
1124   }
1125   }
1126   }
1127   }
1128   }
1129   }
1130   }
1131   }
1132   }
1133   }
1134   }
1135   }
1136   }
1137   }
1138   }
1139   }
1140   }
1141   }
1142   }
1143   }
1144   }
1145   }
1146   }
1147   }
1148   }
1149   }
1150   }
1151   }
1152   }
1153   }
1154   }
1155   }
1156   }
1157   }
1158   }
1159   }
1160   }
1161   }
1162   }
1163   }
1164   }
1165   }
1166   }
1167   }
1168   }
1169   }
1170   }
1171   }
1172   }
1173   }
1174   }
1175   }
1176   }
1177   }
1178   }
1179   }
1180   }
1181   }
1182   }
1183   }
1184   }
1185   }
1186   }
1187   }
1188   }
1189   }
1190   }
1191   }
1192   }
1193   }
1194   }
1195   }
1196   }
1197   }
1198   }
1199   }
1200   }
1201   }
1202   }
1203   }
1204   }
1205   }
1206   }
1207   }
1208   }
1209   }
1210   }
1211   }
1212   }
1213   }
1214   }
1215   }
1216   }
1217   }
1218   }
1219   }
1220   }
1221   }
1222   }
1223   }
1224   }
1225   }
1226   }
1227   }
1228   }
1229   }
1230   }
1231   }
1232   }
1233   }
1234   }
1235   }
1236   }
1237   }
1238   }
1239   }
1240   }
1241   }
1242   }
1243   }
1244   }
1245   }
1246   }
1247   }
1248   }
1249   }
1250   }
1251   }
1252   }
1253   }
1254   }
1255   }
1256   }
1257   }
1258   }
1259   }
1260   }
1261   }
1262   }
1263   }
1264   }
1265   }
1266   }
1267   }
1268   }
1269   }
1270   }
1271   }
1272   }
1273   }
1274   }
1275   }
1276   }
1277   }
1278   }
1279   }
1280   }
1281   }
1282   }
1283   }
1284   }
1285   }
1286   }
1287   }
1288   }
1289   }
1290   }
1291   }
1292   }
1293   }
1294   }
1295   }
1296   }
1297   }
1298   }
1299   }
1300   }
1301   }
1302   }
1303   }
1304   }
1305   }
1306   }
1307   }
1308   }
1309   }
1310   }
1311   }
1312   }
1313   }
1314   }
1315   }
1316   }
1317   }
1318   }
1319   }
1320   }
1321   }
1322   }
1323   }
1324   }
1325   }
1326   }
1327   }
1328   }
1329   }
1330   }
1331   }
1332   }
1333   }
1334   }
1335   }
1336   }
1337   }
1338   }
1339   }
1340   }
1341   }
1342   }
1343   }
1344   }
1345   }
1346   }
1347   }
1348   }
1349   }
1350   }
1351   }
1352   }
1353   }
1354   }
1355   }
1356   }
1357   }
1358   }
1359   }
1360   }
1361   }
1362   }
1363   }
1364   }
1365   }
1366   }
1367   }
1368   }
1369   }
1370   }
1371   }
1372   }
1373   }
1374   }
1375   }
1376   }
1377   }
1378   }
1379   }
1380   }
1381   }
1382   }
1383   }
1384   }
1385   }
1386   }
1387   }
1388   }
1389   }
1390   }
1391   }
1392   }
1393   }
1394   }
1395   }
1396   }
1397   }
1398   }
1399   }
1400   }
1401   }
1402   }
1403   }
1404   }
1405   }
1406   }
1407   }
1408   }
1409   }
1410   }
1411   }
1412   }
1413   }
1414   }
1415   }
1416   }
1417   }
1418   }
1419   }
1420   }
1421   }
1422   }
1423   }
1424   }
1425   }
1426   }
1427   }
1428   }
1429   }
1430   }
1431   }
1432   }
1433   }
1434   }
1435   }
1436   }
1437   }
1438   }
1439   }
1440   }
1441   }
1442   }
1443   }
1444   }
1445   }
1446   }
1447   }
1448   }
1449   }
1450   }
1451   }
1452   }
1453   }
1454   }
1455   }
1456   }
1457   }
1458   }
1459   }
1460   }
1461   }
1462   }
1463   }
1464   }
1465   }
1466   }
1467   }
1468   }
1469   }
1470   }
1471   }
1472   }
1473   }
1474   }
1475   }
1476   }
1477   }
1478   }
1479   }
1480   }
1481   }
1482   }
1483   }
1484   }
1485   }
1486   }
1487   }
1488   }
1489   }
1490   }
1491   }
1492   }
1493   }
1494   }
1495   }
1496   }
1497   }
1498   }
1499   }
1500   }
1501   }
1502   }
1503   }
1504   }
1505   }
1506   }
1507   }
1508   }
1509   }
1510   }
1511   }
1512   }
1513   }
1514   }
1515   }
1516   }
1517   }
1518   }
1519   }
1520   }
1521   }
1522   }
1523   }
1524   }
1525   }
1526   }
1527   }
1528   }
1529   }
1530   }
1531   }
1532   }
1533   }
1534   }
1535   }
1536   }
1537   }
1538   }
1539   }
1540   }
1541   }
1542   }
1543   }
1544   }
1545   }
1546   }
1547   }
1548   }
1549   }
1550   }
1551   }
1552   }
1553   }
1554   }
1555   }
1556   }
1557   }
1558   }
1559   }
1560   }
1561   }
1562   }
1563   }
1564   }
1565   }
1566   }
1567   }
1568   }
1569   }
1570   }
1571   }
1572   }
1573   }
1574   }
1575   }
1576   }
1577   }
1578   }
1579   }
1580   }
1581   }
1582   }
1583   }
1584   }
1585   }
1586   }
1587   }
1588   }
1589   }
1590   }
1591   }
1592   }
1593   }
1594   }
1595   }
1596   }
1597   }
1598   }
1599   }
1600   }
1601   }
1602   }
1603   }
1604   }
1605   }
1606   }
1607   }
1608   }
1609   }
1610   }
1611   }
1612   }
1613   }
1614   }
1615   }
1616   }
1617   }
1618   }
1619   }
1620   }
1621   }
1622   }
1623   }
1624   }
1625   }
1626   }
1627   }
1628   }
1629   }
1630   }
1631   }
1632   }
1633   }
1634   }
1635   }
1636   }
1637   }
1638   }
1639   }
1640   }
1641   }
1642   }
1643   }
1644   }
1645   }
1646   }
1647   }
1648   }
1649   }
1650   }
1651   }
1652   }
1653   }
1654   }
1655   }
1656   }
1657   }
1658   }
1659   }
1660   }
1661   }
1662   }
1663   }
1664   }
1665   }
1666   }
1667   }
1668   }
1669   }
1670   }
1671   }
1672   }
1673   }
1674   }
1675   }
1676   }
1677   }
1678   }
1679   }
1680   }
1681   }
1682   }
1683   }
1684   }
1685   }
1686   }
1687   }
1688   }
1689   }
1690   }
1691   }
1692   }
1693   }
1694   }
1695   }
1696   }
1697   }
1698   }
1699   }
1700   }
1701   }
1702   }
1703   }
1704   }
1705   }
1706   }
1707   }
1708   }
1709   }
1710   }
1711   }
1712   }
1713   }
1714   }
1715   }
1716   }
1717   }
1718   }
1719   }
1720   }
1721   }
1722   }
1723   }
1724   }
1725   }
1726   }
1727   }
1728   }
1729   }
1730   }
1731   }
1732   }
1733   }
1734   }
1735   }
1736   }
1737   }
1738   }
1739   }
1740   }
1741   }
1742   }
1743   }
1744   }
1745   }
1746   }
1747   }
1748   }
1749   }
1750   }
1751   }
1752   }
1753   }
1754   }
1755   }
1756   }
1757   }
1758   }
1759   }
1760   }
1761   }
1762   }
1763   }
1764   }
1765   }
1766   }
1767   }
1768   }
1769   }
1770   }
1771   }
1772   }
1773   }
1774   }
1775   }
1776   }
1777   }
1778   }
1779   }
1780   }
1781   }
1782   }
1783   }
1784   }
1785   }
1786   }
1787   }
1788   }
1789   }
1790   }
1791   }
1792   }
1793   }
1794   }
1795   }
1796   }
1797   }
1798   }
1799   }
1800   }
1801   }
1802   }
1803   }
1804   }
1805   }
1806   }
1807   }
1808   }
1809   }
1810   }
1811   }
1812   }
1813   }
1814   }
1815   }
1816   }
1817   }
1818   }
1819   }
1820   }
1821   }
1822   }
1823   }
1824   }
1825   }
1826   }
1827   }
1828   }
1829   }
1830   }
1831   }
1832   }
1833   }
1834   }
1835   }
1836   }
1837   }
1838   }
1839   }
1840   }
1841   }
1842   }
1843   }
1844   }
1845   }
1846   }
1847   }
1848   }
1849   }
1850   }
1851   }
1852   }
1853   }
1854   }
1855   }
1856   }
1857   }
1858   }
1859   }
1860   }
1861   }
1862   }
1863   }
1864   }
1865   }
1866   }
1867   }
1868   }
1869   }
1870   }
1871   }
1872   }
1873   }
1874   }
1875   }
1876   }
1877   }
1878   }
1879   }
1880   }
1881   }
1882   }
1883   }
1884   }
1885  
```

需要生成一个手机号的密码集合

```
#!/usr/bin/python
# -*- coding:utf8 -

s = '1391040'
s1 = ''
dic = open("pass.txt","a")
for a in range(9999):
    s1 = s+str(a).zfill(4)+'\n'
    dic.write(s1)
```

命令

```
aricrack -a 2-w pass.txt wifi.cap
```

```
C:\WINDOWS\system32\cmd.exe

Aircrack-ng 1.1

[00:00:04] 7688 keys tested (1565.15 k/s)

KEY FOUND! [ 13910407686 ]

Master Key      : C4 60 FE 8B 14 7D 58 00 91 D7 0A 9C 3C DE 44 69
                  0B E1 CD 81 07 F8 28 DB EA 76 1E ED 81 A3 FF FD

Transient Key   : 0D 88 B3 F4 BC A3 C9 D2 06 12 28 43 FF 5E 21 3E
                  F5 23 8E 0B 7A 9F 25 59 E9 7C 86 1E 7A 78 E4 D4
                  D3 62 CD DD 4D 87 80 EE B9 E1 16 91 4A 6E 3E 09
                  1E CE 5E 62 38 3C 05 35 34 A6 EB 16 31 D8 CE 96

EAPOL HMAC     : 1C E7 D0 96 DE 87 93 56 88 1D 08 C8 B9 AA B3 B0

https://blog.csdn.net/qq_40657585
```

18.Linux2

解压得到brave

binwalk, foremost



what? ? ? , 为啥咋交都不对


```
root@kali:~/Desktop/MISC# strings brave | grep KEY
KEY{24f3627a86fc740a7f36ee2c7a1c124a}
KEY{}
```

然后，这到底啥意思

19.账号被盗了

You are not an admin!

bp 抓包改包，false改成true

```
POST /cookieflag.php HTTP/1.1
Host: 123.206.87.240:9001
Content-Length: 0
Cache-Control: max-age=0
Origin: http://123.206.87.240:9001
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Referer: http://123.206.87.240:9001/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: isadmin=true
Connection: close
```

```
Date: Sun, 18 Nov 2018 12:42:57 GMT
Content-Type: text/html
Connection: close
Content-Length: 366

<!DOCTYPE html>
<html>
  <style>
    span {
      display: block;
      margin: auto;
      height: 25px;
      text-align: center;
      font-size: 30px;
    }
  </style>
<head>
  <title>bugku</title>
  <link href="style.css" rel="stylesheet" type="text/css">
</head>
<body>
  <span>http://120.24.86.145:9001/123.exe</span>
</body>
</html>
```

发现一个网址

<http://120.24.86.145:9001/123.exe>

下载下来后，发现



填好账号，密码

wireshark抓下包

```
220 smtp.qq.com Esmtpl QQ Mail Server
EHLO DELL
250-smtp.qq.com
250-PIPELINING
250-SIZE 73400320
250-STARTTLS
250-AUTH LOGIN PLAIN
250-AUTH=LOGIN
250-MAILCOMPRESS
250 8BITMIME
AUTH LOGIN
334 VXNlcm5hbWU6
YmtjdGZ0ZXN0QDE2My5jb20=
334 UGFzc3dvcmQ6
YTEyMzQ1Ng==
535 Error: .....: http://service.mail.qq.com/cgi-bin/help?subtype=1&&id=28&&no=1001256
QUIT
221 Bye
```

https://blog.csdn.net/qq_40657585

发现两个base64

解码发现是163邮箱账号和密码

bkctftest@163.com

a123456

登录即可

20.细心的大象

解压，查看属性

1.jpg 属性

常规 安全 详细信息 以前的版本

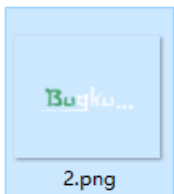
属性	值
说明	
标题	出题人已经跑路了
主题	出题人已经跑路了
分级	☆☆☆☆☆
标记	
备注	TVNEUzQ1NkFTRDEyM3p6
来源	
作者	Bugku
拍摄日期	2017/8/10 11:53
程序名称	sagit-user 7.1.1 NMF26X V8.2.26.0.NC...
获取日期	
版权	
图像	
图像 ID	
分辨率	3016 x 4032
宽度	3016 像素
高度	4032 像素
水平分辨率	72 dpi
垂直分辨率	72 dpi
位深度	24

[删除属性和个人信息 https://blog.csdn.net/qq_40657585](https://blog.csdn.net/qq_40657585)

base64解码 MSDS456ASD123zz

binwalk foremost

出现了压缩包，输入密码



眼熟。。改高度



BUGKU{a1e5aSA}

https://blog.csdn.net/qq_40657585

flag都没改，就是那张图

21.爆照(08067CTF)

一张图片

```
root@kali:~/Desktop/MISC/8# binwalk 8.jpg
DECIMAL      HEXADECIMAL  DESCRIPTION
-----
0            0x0          JPEG image data, JFIF standard 1.01
40499       0x9E33      Zip archive data, encrypted at least v2.0 to extract, compressed size: 8362, uncompressed size: 92278, name: 8
48892       0xBEFC      Zip archive data, at least v2.0 to extract, compressed size: 14906, uncompressed size: 15739, name: 88
63830       0xF956      Zip archive data, at least v2.0 to extract, compressed size: 11129, uncompressed size: 18479, name: 888
74992       0x124F0     Zip archive data, at least v2.0 to extract, compressed size: 10371, uncompressed size: 11782, name: 8888
85397       0x14D95     Zip archive data, at least v2.0 to extract, compressed size: 6945, uncompressed size: 92278, name: 88888
92377       0x168D9     Zip archive data, at least v2.0 to extract, compressed size: 6824, uncompressed size: 92278, name: 888888
99237       0x183A5     Zip archive data, at least v2.0 to extract, compressed size: 7076, uncompressed size: 92278, name: 8888888
106350      0x19F6E     Zip archive data, at least v2.0 to extract, compressed size: 8219, uncompressed size: 92278, name: 88888888
168452      0x29204     End of Zip archive
```

foremost 分离出8个文件和一个gif

用TXT打开，发现是jpg文件

第二张图发现二维码，第三张图备注发现base64，第四张图发现隐藏文件，根据题目提示，得出flag

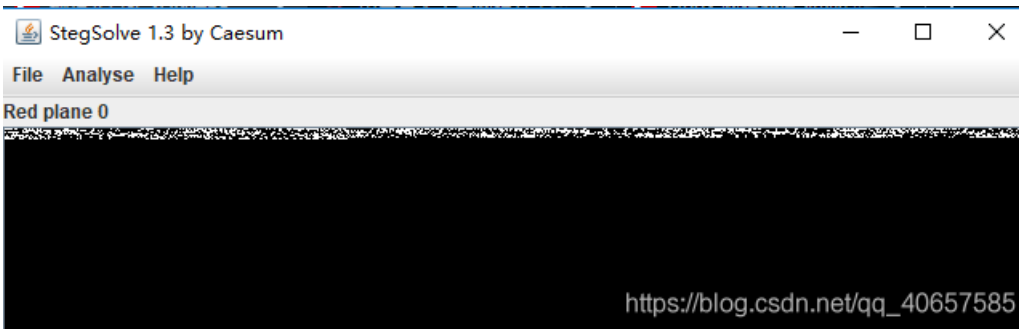
真的难。。。

22.猫片(安恒)

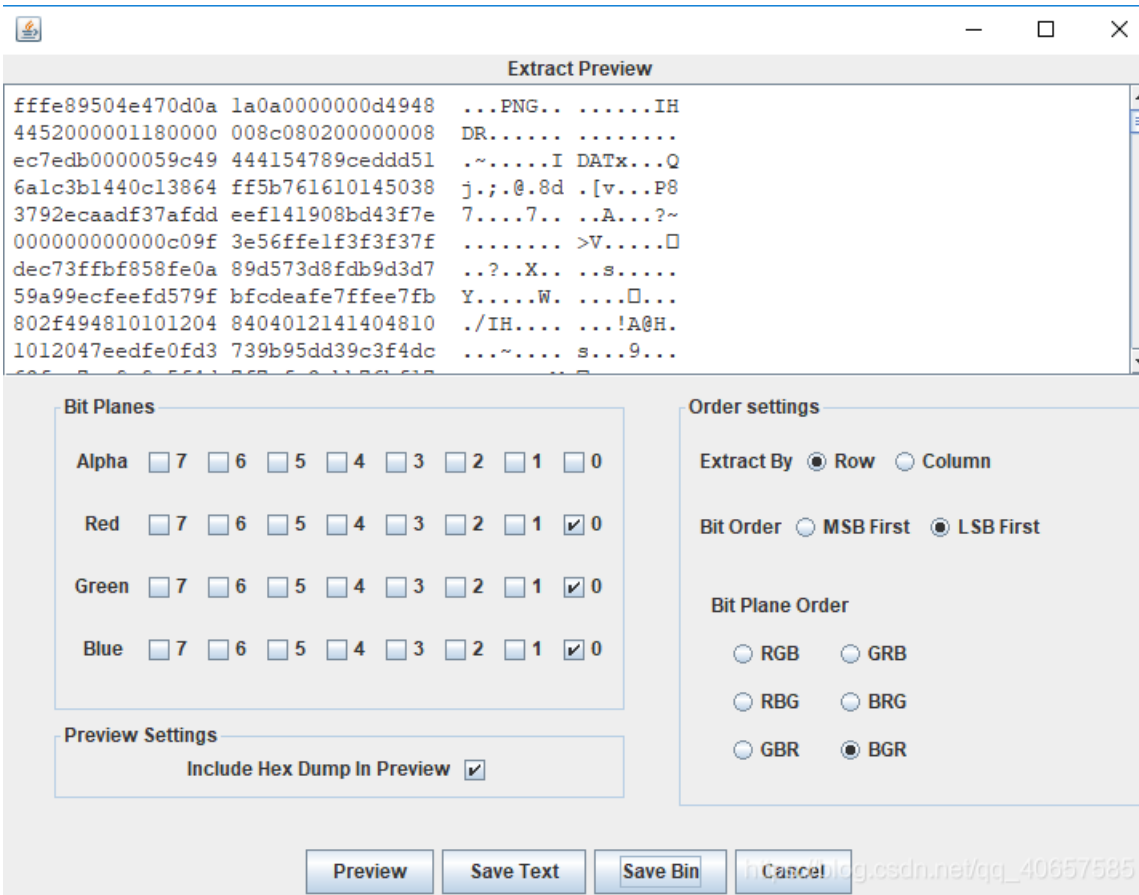


一波操作，没用，属性，binwalk，steghide，。。。都没用

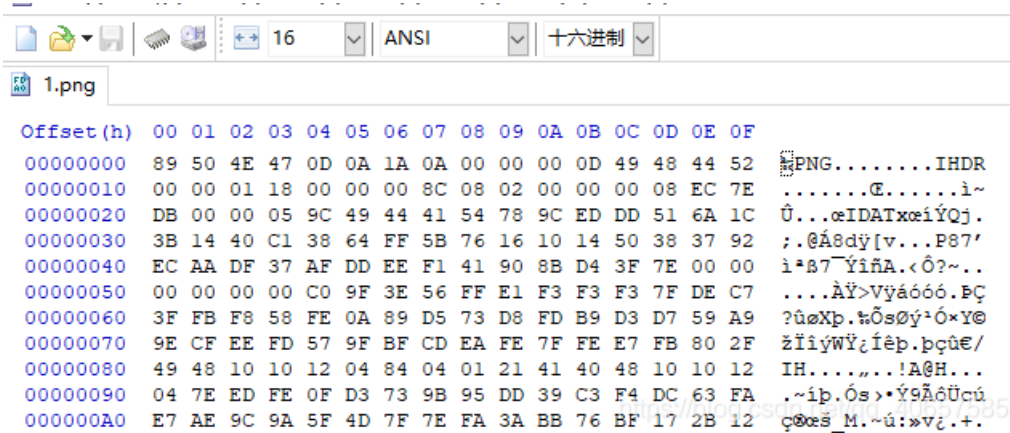
上网搜了一波，发现是StegSolve解决的



发现有东西，进入Extract Preview



发现是png格式的，保存为二进制。



png 图片开头是89 50 4E 47，删除多余的东西，保存



半张 二维码，改高度，老操作



扫描 二维码，得到

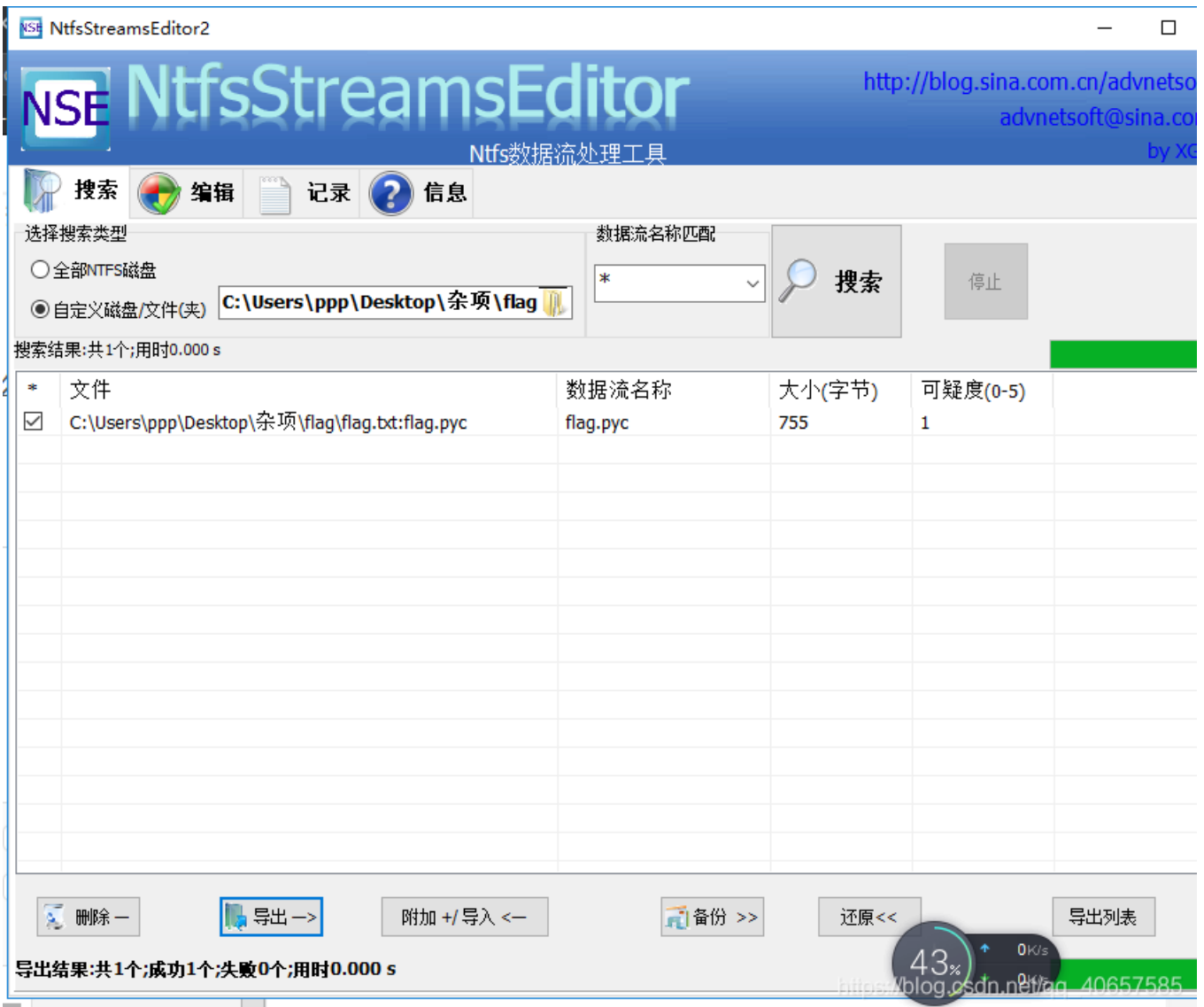


下载下来是个压缩包

解压



ntfs隐写，前几天刚看这个操作，然而怎么都不好使，看别人题解得知必须得能让人解码才可以读出字节流，果然好使



.pyc python逆向

uncompyle2 反编译一下

```
root@kali:~/uncompyle2# uncompyle2 '/root/Desktop/C!_Users_ppp_Desktop_杂项_flag_flag.txt!flag.pyc' >2.py
root@kali:~/uncompyle2#
```



```
# 2018.11.19 20:12:05 CST
#Embedded file name: flag.py
import base64

def encode():
    flag = '*****'
    ciphertext = []
    for i in range(len(flag)):
        s = chr(i ^ ord(flag[i]))
        if i % 2 == 0:
            s = ord(s) + 10
        else:
            s = ord(s) - 10
        ciphertext.append(str(s))

    return ciphertext[::-1]

ciphertext = ['96',
'65',
'93',
'123',
'91',
'97',
'22',
'93',
'70',
'102',
'94',
'132',
'46',
'112',
'64',
'97',
'88',
'80',
'82',
'137',
'90',
'109',
'99',
'112']
+++ okay decompiling /root/Desktop/C!_Users_ppp_Desktop_鐵佬]_flag_flag.txt!flag.pyc
# decompiled 1 files: 1 okay, 0 failed, 0 verify failed
# 2018.11.19 20:12:05 CST
```

写脚本解密

```

import base64

def decode():
    flag = ''
    ciphertext = ['96',
                  '65',
                  '93',
                  '123',
                  '91',
                  '97',
                  '22',
                  '93',
                  '70',
                  '102',
                  '94',
                  '132',
                  '46',
                  '112',
                  '64',
                  '97',
                  '88',
                  '80',
                  '82',
                  '137',
                  '90',
                  '109',
                  '99',
                  '112']

    a = []
    a = ciphertext
    a.reverse()    #数组逆序
    len1 = len(a)
    for i in range(len1):
        if i % 2 == 0:
            s = int(a[i]) - 10
        else:
            s = int(a[i]) + 10
        s=chr(i^s)
        flag += s

    return flag
if __name__ == '__main__':
    flag = decode()
    print flag[::-1]    #字符串逆序
    print flag

```

```

}!revE1C_Rev3lC_e@Y {galf
flag {Y@e_Cl3veR_ClEver!}

```

23.多彩

24.旋转跳跃

25.普通的二维码

打开一张二维码



```
已解码数据 1:
-----
位置:(16.0,16.0)-(132.0,16.0)-(16.0,132.0)-(132.0,132.0)
颜色正常,正像
版本:3
纠错等级:M,掩码:7
内容:
哈哈!就不告诉你flag就在这里!
-----
```

```
root@kali: ~/Desktop/bisai
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
root@kali:~/Desktop/bisai# binwalk misc100.bmp
DECIMAL      HEXADECIMAL  DESCRIPTION
-----
0            0x0         PC bitmap, Windows 3.x format,, 148 x 148 x 1

root@kali:~/Desktop/bisai# strings misc100.bmp
14615414114717311014116614513717106012513712017113716314316215116016413711716414
3137124157137124145156137101163143151151041175@xjseck!
root@kali:~/Desktop/bisai#
```

binwalk啥也没发现，strings发现一串字符，感觉像是8进制，因为最大是7，八进制转十进制，

```
12 49 44 12 9 39 15 25 8 12 9 54 12 41 31 15 8 48 10 41 31 10 1 57 11 57 51 12 25 50 13 9 48 14 33 31 9 57
```

emmm，不可能出来字母，试一下3位转一个

```
102 108 97 103 123 72 97 118 101 95 121 48 85 95 80 121 95 115 99 114 105 112 116 95 79 116 99 95 84 111 95
```

有希望，转字母

```
flag{Have_y0U_Py_script_0tc_To_Ten_Ascii!}
```

附上脚本

```
#!/usr/bin/python
# -*- coding:utf8 -
f = open("1.txt",'r')
s = ''
a = f.readline()
len1 = len(a)
for i in range(0,len1,3): #步长为3
    s = int(a[i:i+3],8) #切片长度为3
    print(chr(s)),
```

```
#!/usr/bin/python
# -*- coding:utf8 -
f = open("1.txt",'r')
s = ''
# a = f.read() #read()只能读一次，在read就读不出了
# len1 = len(a)
# for i in range(0,len1,3):
#     s = int(a[i:i+3],8) #使用切片，3位8进制转十进制
#     print(chr(s)),
len2 = 126//3
for i in range(len2):
    ss = eval('0'+f.read(3)) #3位八进制转十进制
    s +=chr(int(ss))
print s
```

26.乌云邀请码

您好：

这是来自于WooYun的一封邀请邮件，非常高兴你通过WooYun发布有价值的漏洞，很荣幸的邀请你成为WooYun白帽子中的一员，你可以通过如下的链接来注册

http://www.wooyun.org/user.php?action=register&code=b6d75821211e338dd56623c8825456ab&invite_email=504038236@qq.com&invite_type=0

WooYun会给你发送一封确认邮件，可以点击其中的链接完成注册，希望你继续支持WooYun

漏洞处理流程：<http://www.wooyun.org/help#bug>

白帽注意事项：<http://www.wooyun.org/help#whitehat>

本邮件由WooYun自动发送，请勿回复

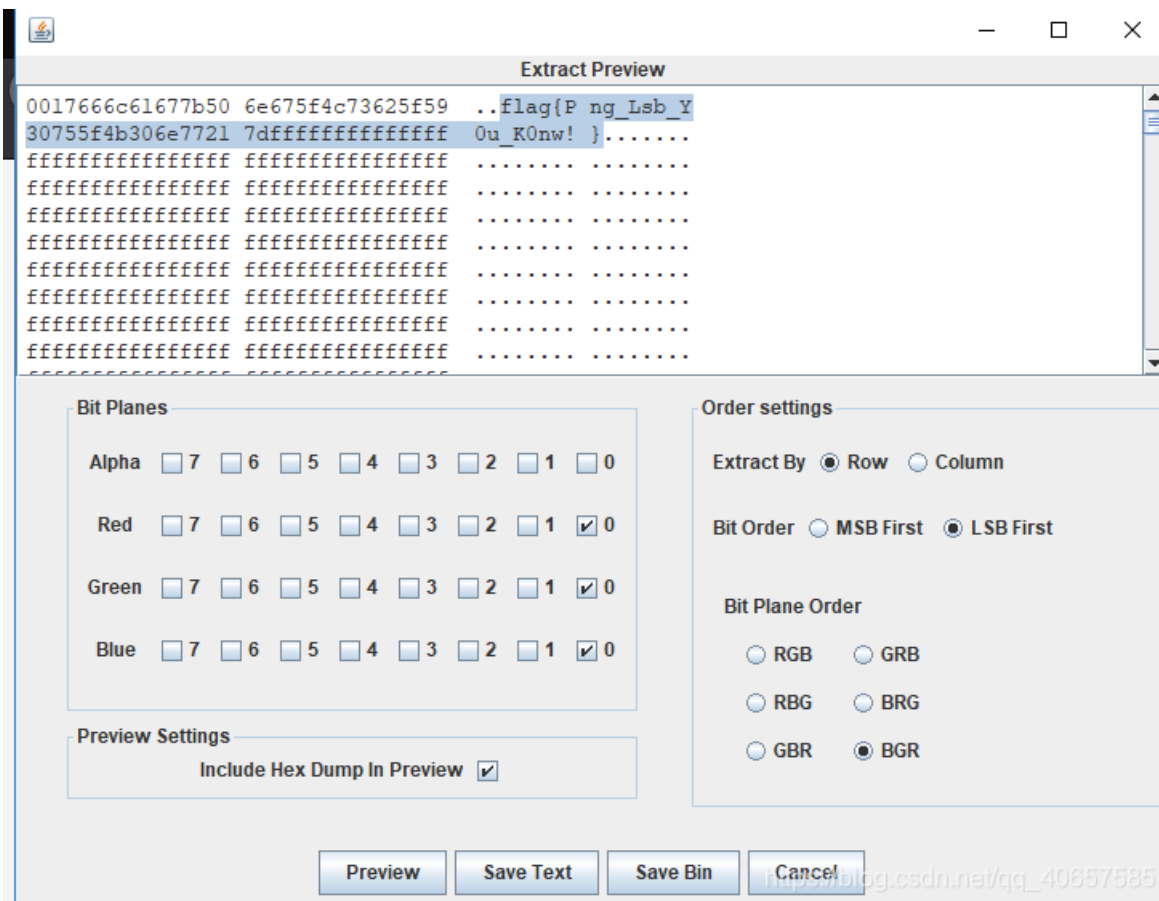
WooYun是一个自由平等的漏洞和安全信息报告平台

其他关于WooYun的更多详细信息请访问<http://www.wooyun.org/about.php>

谢谢!

https://blog.csdn.net/qq_40657585

属性 binwalk strings毫无发现，最后在stegSlove发现东西，flag，



27. 神秘的文件

将题目解压出来，题目压缩包里有logo.png和一个加密压缩包，很明显的明文破解，使用题目压缩包作为key和writeup压缩包进行明文破解(或者使用2345好压的标准压缩算法压缩logo.png)，得到密码：q1w2e3r4

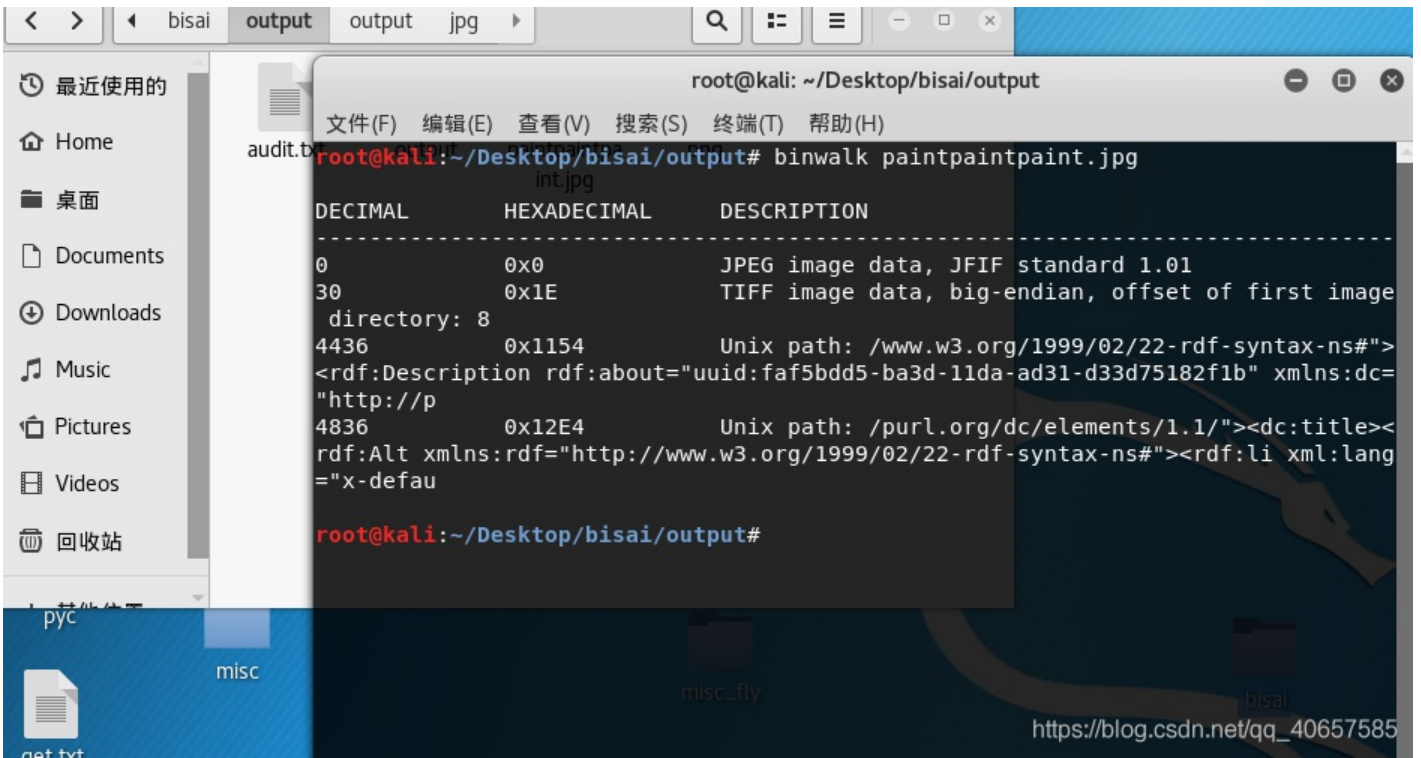
解压后得到doc文件，当成压缩包解密，能找到flag.txt中有一串base64编码的字符串，解码即可。

官方的，反正我是没爆破出来，谁做出来了，告诉我声。

28.图穷匕现



binwalk 发现有其他东西

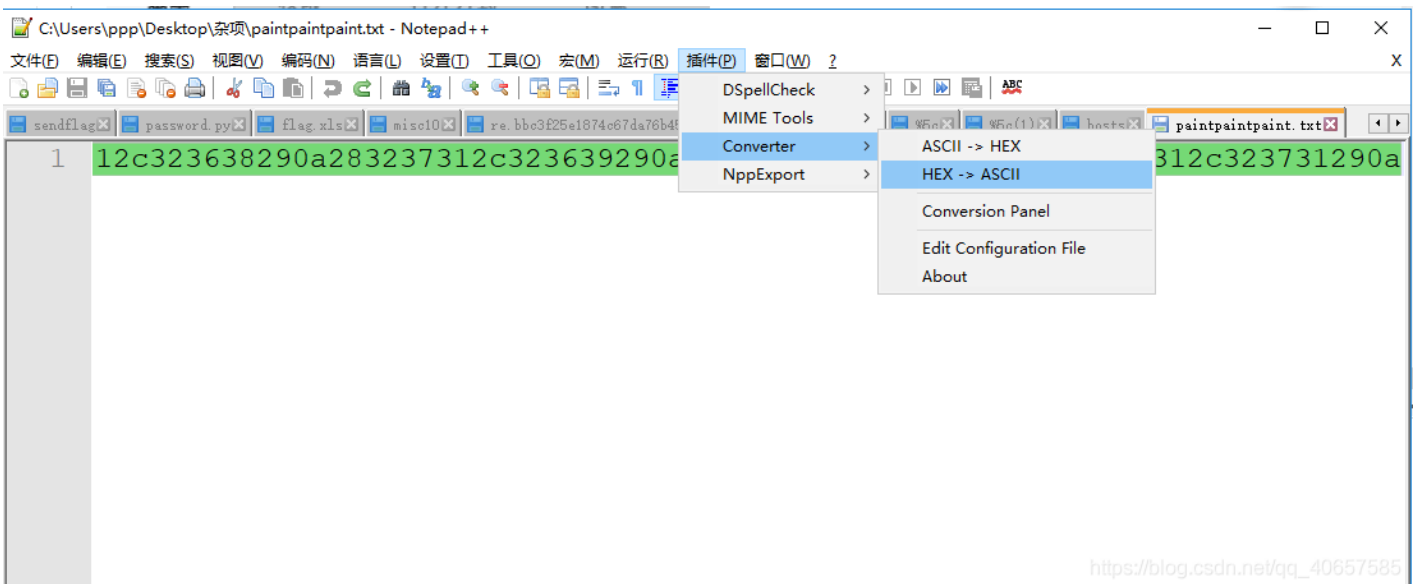


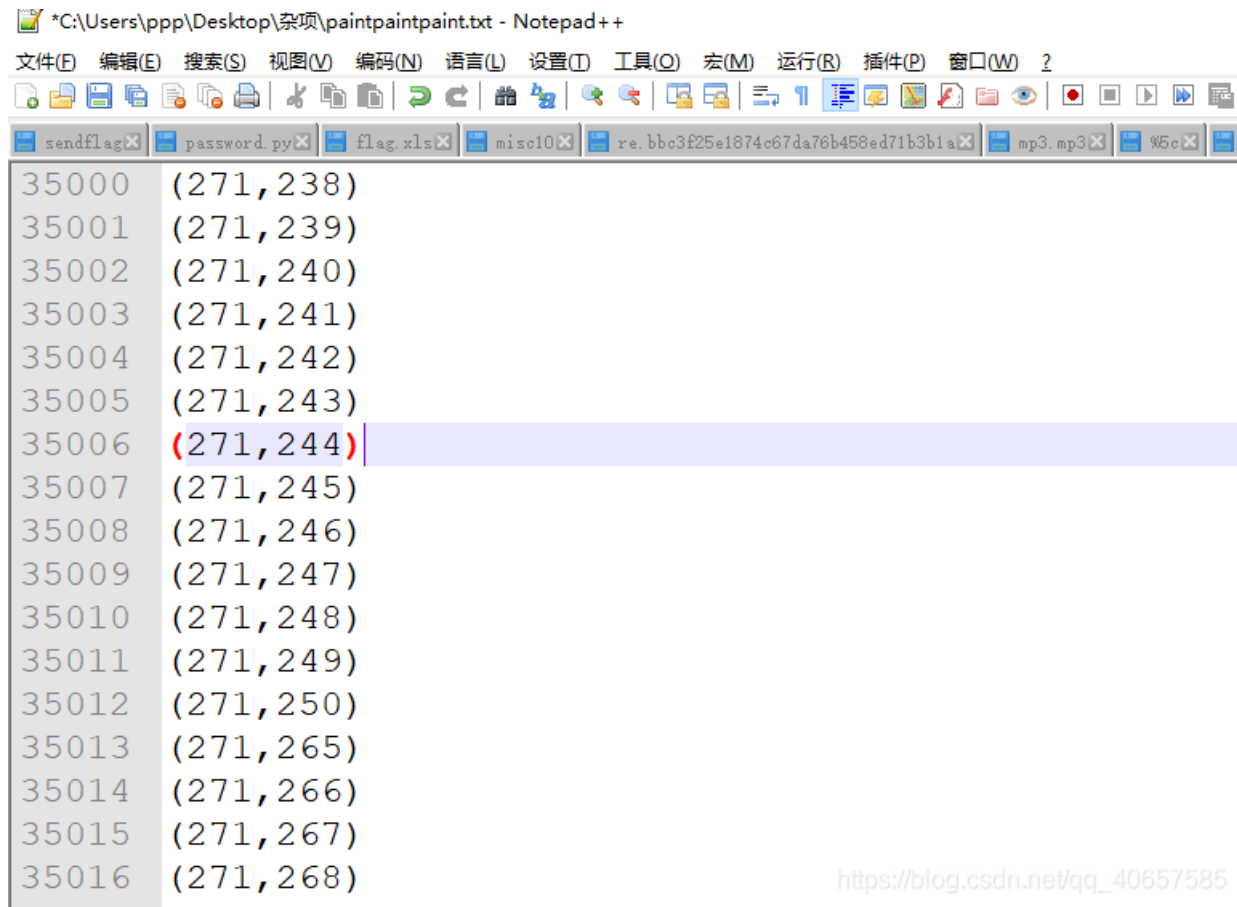
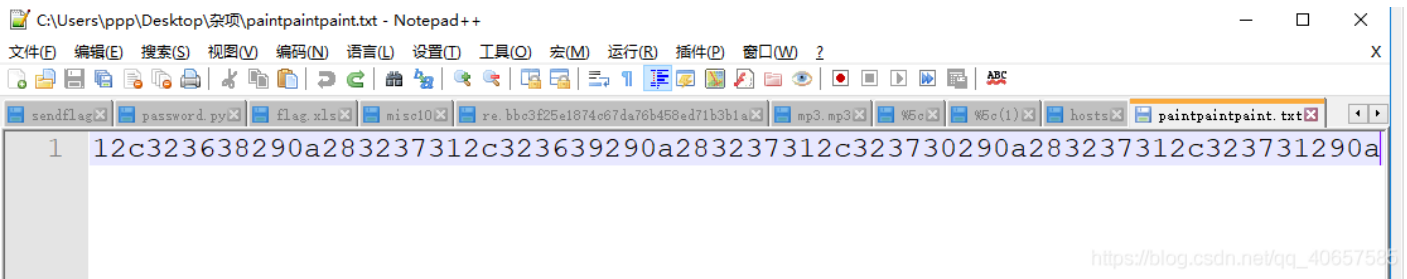
foremost 分离出来一张图片，和原图一样，但是只有21.2k，原图好几百k，emm，知道了吧；jpg文件头，FF D8 文件尾FF D9

用16进制编辑器打开，删除jpg文件部分，后缀名改为txt



16进制，转ascii码，notepad++就可以完成





接下来 画图就可以了，参考前面

29.convert

白哥的鸽子

jpg图片，查看图片属性，分一下层，binwalk，啥也没有。

strings一下



栅栏解码

```
fg2ivyo}|{2s3_o@aw__rcl@
```

结果:

得到因数(排除1和字符串长度):

2 3 4 6 8 12

第1栏: f2vo|23oa_r|giy}|s_@w_c@

第2栏: fio{3@_cgv}2_a_l2ylsowr@

第3栏: fv|3argy[_wc2o2o_li}s@_@

第4栏: fo3_g}__2lori{@cv2alysw@

第5栏: flag{w22_is_v3ry_cool}@@

第6栏: f3g_2oi@vaywo_)_lr{c2ls@

https://blog.csdn.net/qq_40657585

神秘的文件

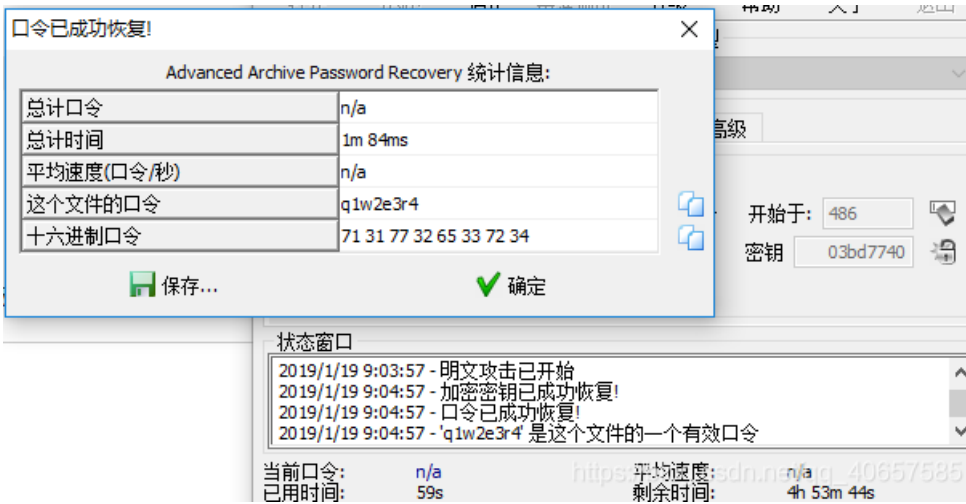
解压后得到一个flag压缩包和logo图片



压缩包含有密码, 并且里面也有一个相同的logo图片

名称	大小	压缩后大小	类型	修改时间	CRC32
..			文件夹		
2018山东省大...	272,070	259,726	Microsoft Word ...	2018/11/2 14...	6C5C9C...
logo.png *	27,870	27,405	PNG 文件	2018/10/15 1...	3E62BF64

知道了吧, zip明文爆破 (WinRAR加压才可以)



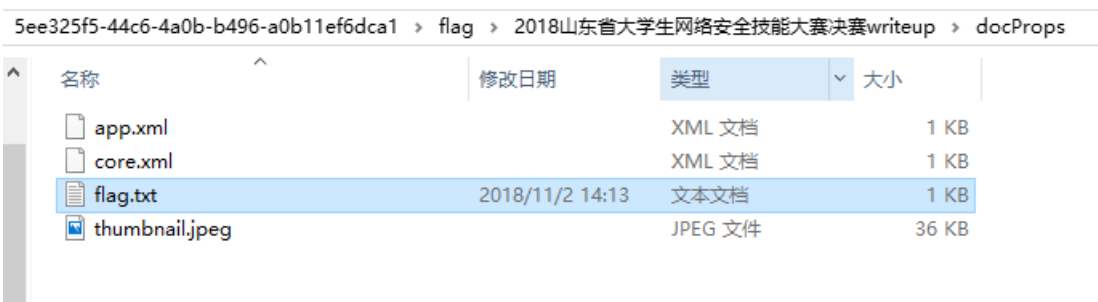
解压，打开文档



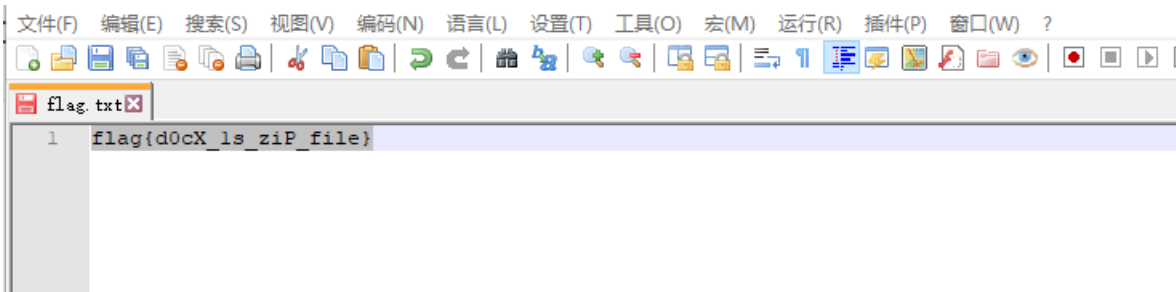
哪有什么 WriteUP，别想了，老老实实做题吧!

https://blog.csdn.net/qq_40657585

改后缀为zip，解压，在docProps里发现flag.txt



发现是base64，解密，我用的是notepad++，可以直接解密（插件里面的MIME Tools）



论剑

Challenge 19 Solves

论剑

100

剑客

十年磨一剑，霜刃未曾试。
今日把示君，谁有不平事。

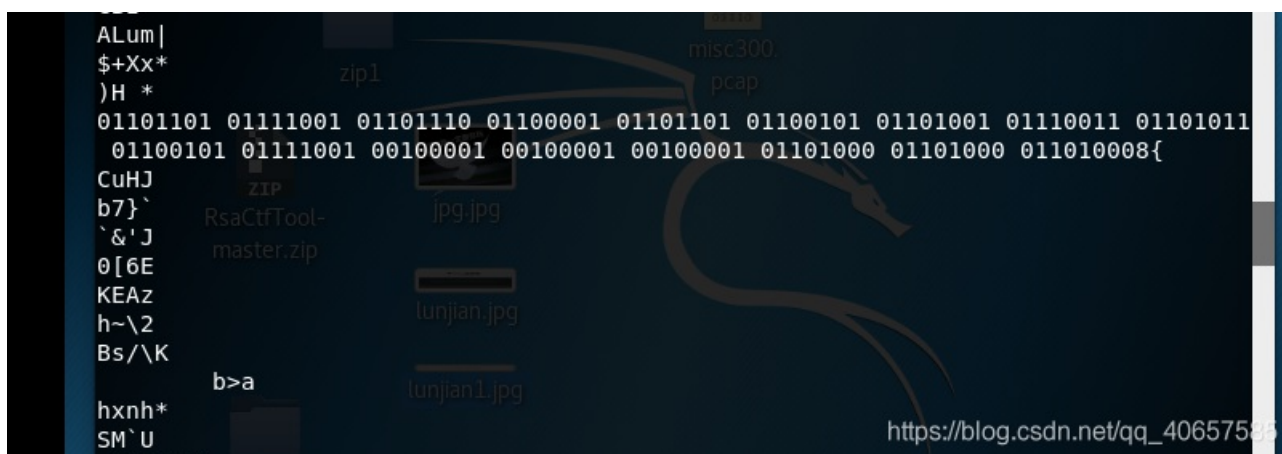
lunjian.jpg

Flag Submit

https://blog.csdn.net/qq_40657585

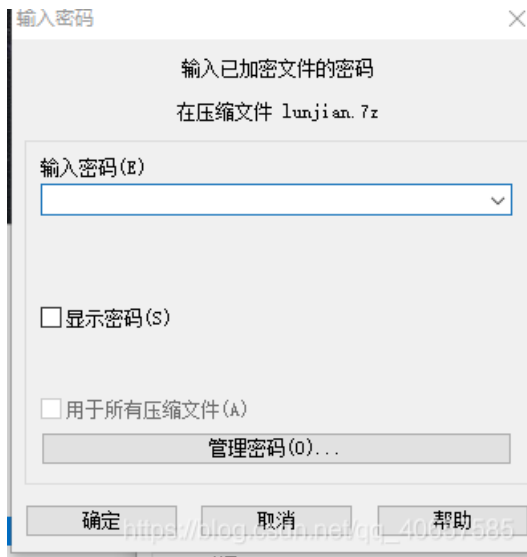
不知道这道题到底要干什么，试了好多种方法，不好使。

strings 发现一串二进制，转成ASCII为mynameiskey!!!hhh，没啥用

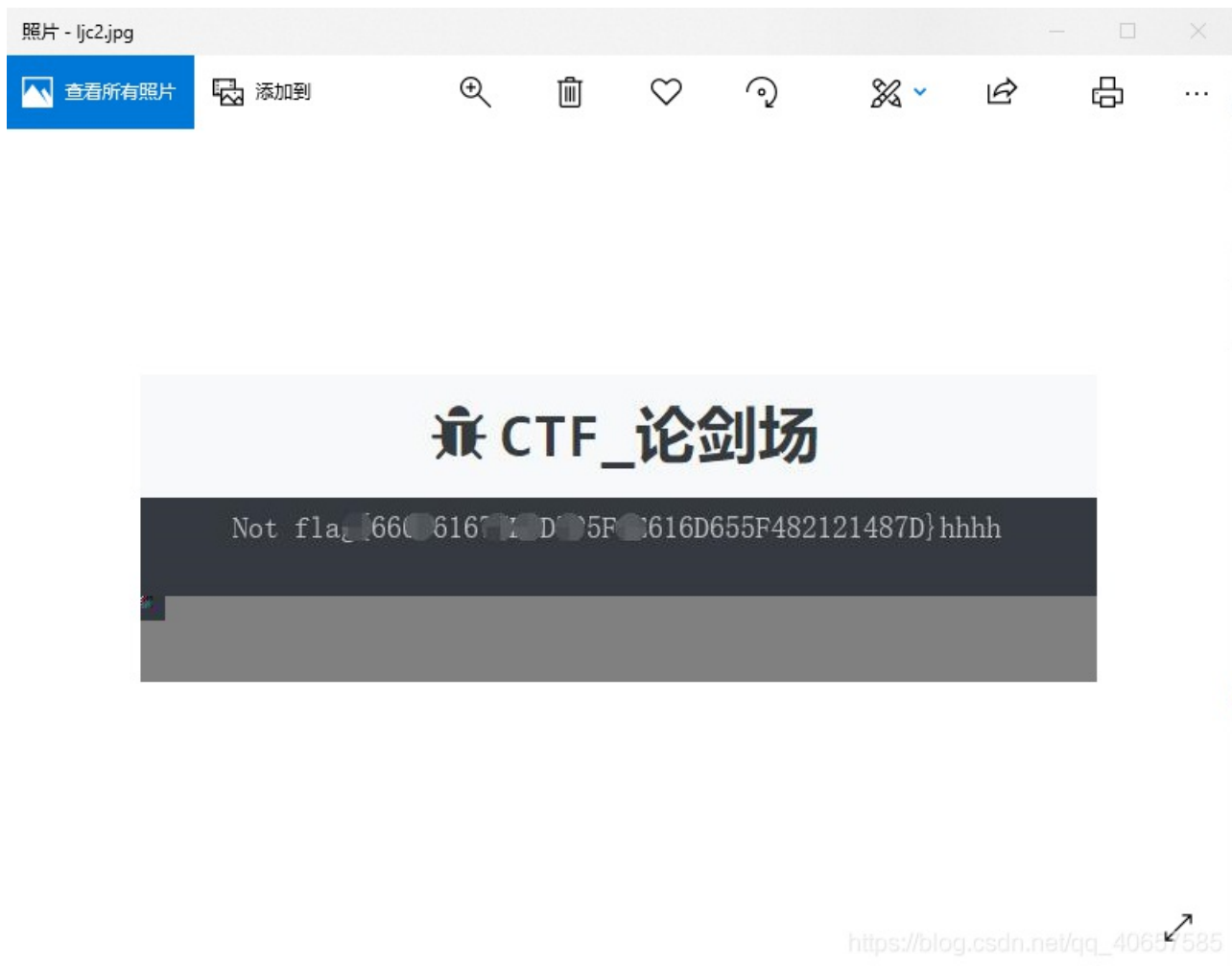


binwalk, foremost, 分离得到两张图片，各种尝试，最后发现修改高度，发现东西

解压的时候发现密码



尝试用 mynameiskey!!!hhh, 成功了



对比一下


```
#!/usr/bin/python
# -*- coding:utf8 -
from PIL import Image
x = 503    #x坐标 通过对txt里的行数进行整数分解
y = 122   #y坐标  x*y=行数

im = Image.new("RGB", (x, y))
file = open('1.txt')

for i in range(0, x):
    for j in range(0, y):
        line = file.readline().replace('(', '').replace(')', '') #获取一行rgb值, 并且把()都替换为空
        rgb = line.split(",") #逗号分割
        im.putpixel((i, j), (int(rgb[0]), int(rgb[1]), int(rgb[2]))) # (i,j) 为坐标, 后面的是像素点

im.save("f.png")
```



flag{ youc@n'tseeme }