第五届蓝帽杯初赛 misc 赛后复现

原创

z.volcano 于 2021-05-01 08:10:29 发布 1182 收藏

分类专栏： # 比赛&复现

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/weixin_45696568/article/details/116305952

版权

比赛&复现 专栏收录该内容

4 篇文章 0 订阅

订阅专栏

挺可惜的，比赛当天正好是我这个月事情最多的一天，一共也就不到30分钟做题时间，那个图片题差一点就出来了...

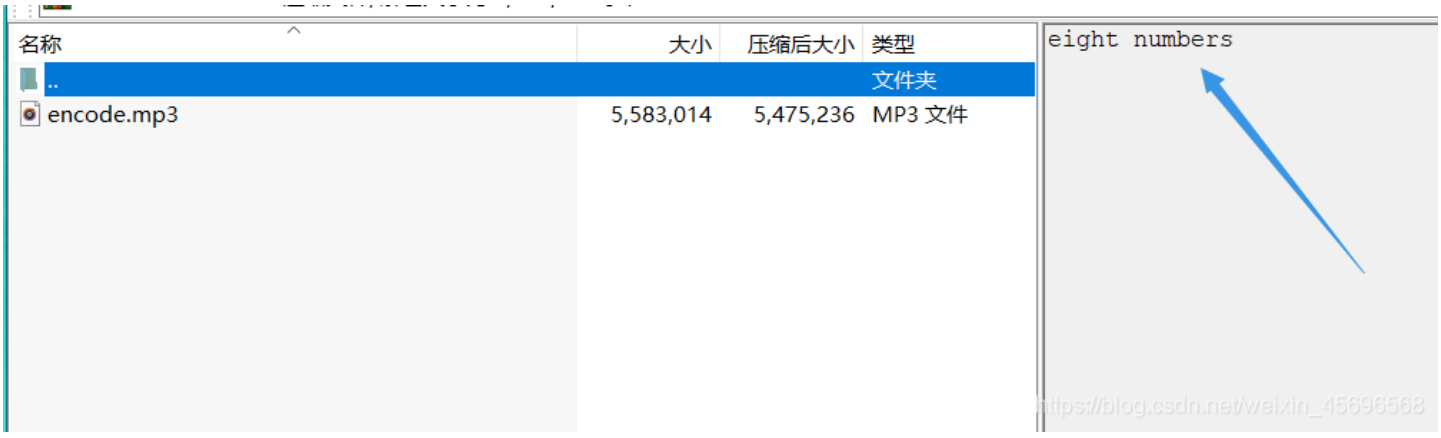
说到底还是自己太菜了，下次好好努力

题目附件我打包了一份：

> 链接：https://pan.baidu.com/s/1txp06OAZL9Ce-qN_VQhLtg
> 提取码：v9j7

冬奥会_is_coming

前面还是基础操作，binwalk分离出一个压缩包，发现**注释信息处有提示**



得到的mp3文件先用`Audacity`看一下频谱和波形，没有异常，那应该就是**mp3隐写**了，这里提示的`eight numbers`应该是密码的提示，冬奥会主题+8位数字，很容易联想到冬奥会举办日期：`20220204`

使用`MP3Stego`，命令如下，成功解密

```
C:\Users\ \Desktop\tools\MP3Stego_1_1_18\MP3Stego>Decode.exe -X -P 20220204 encode.mp3
MP3StegoEncoder 1.1.17
See README file for copyright info
Input file = 'encode.mp3'  output file = 'encode.mp3.pcm'
Will attempt to extract hidden information. Output: encode.mp3.txt
the bit stream file encode.mp3 is a BINARY file
HDR: s=FFF, id=1, l=3, ep=off, br=9, sf=0, pd=1, pr=0, m=0, js=0, c=0, o=0, e=0
alg.=MPEG-1, layer=III, tot bitrate=128, sfrq=44.1
mode=stereo, sblim=32, jsbd=32, ch=2
[Frame 13356]Frame cannot be located
Input stream may be empty
Avg slots/frame = 417.984; b/smp = 2.90; br = 128.008 kbps
Decoding of "encode.mp3" is finished
The decoded PCM output file name is "encode.mp3.pcm"
```

https://blog.csdn.net/weixin_45696568

解出的文本`encode.mp3.txt`内容如下，如果细心观察会发现，中间有一些**用于分割的空格**

encode.mp3.txt - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
\xe2\x9c\x8c\xef\xb8\x8e \xe2\x98\x9d\xef\xb8\x8e\xe2\x99\x93\xef\xb8\x8e\xe2\xa7\xab\xef\xb8\x8e
\xe2\x98\x9f\xef\xb8\x8e\xe2\x97\x86\xef\xb8\x8e\xe2\x99\x8c\xef\xb8\x8e \xe2\x9d\x92\xef\xb8\x8e
\xe2\x99\x8f\xef\xb8\x8e\xe2\x97\xbb\xef\xb8\x8e\xe2\x96\xa1\xef\xb8\x8e\xe2\xac\xa7\xef\xb8\x8e
\xe2\x99\x93\xef\xb8\x8e\xe2\xa7\xab\xef\xb8\x8e\xe2\x96\xa1\xef\xb8\x8e\xe2\x9d\x92\xef\xb8\x8e
\xe2\x8d\x93\xef\xb8\x8e \xe2\x96\xa0\xef\xb8\x8e\xe2\x99\x8b\xef\xb8\x8e\xe2\x9d\x8d\xef\xb8\x8e
\xe2\x99\x8f\xef\xb8\x8e\xe2\x99\x8e\xef\xb8\x8e \xf0\x9f\x93\x82\xef\xb8\x8e\xe2\x99\x8d\xef
\xb8\x8e\xe2\x99\x8f\xef\xb8\x8e\xf0\x9f\x8f\xb1\xef\xb8\x8e\xe2\x99\x8f\xef\xb8\x8e\xe2\x99\x8b\xef
\xb8\x8e\xf0\x9f\x99\xb5 \xe2\x99\x93\xef\xb8\x8e\xe2\xac\xa7\xef\xb8\x8e \xe2\x9d\x96\xef\xb8\x8e
\xe2\x99\x8f\xef\xb8\x8e\xe2\x9d\x92\xef\xb8\x8e\xe2\x8d\x93\xef\xb8\x8e \xe2\x99\x93\xef\xb8\x8e
\xe2\x96\xa0\xef\xb8\x8e\xe2\xa7\xab\xef\xb8\x8e\xe2\x99\x8f\xef\xb8\x8e\xe2\x9d\x92\xef\xb8\x8e
\xe2\x99\x8f\xef\xb8\x8e\xe2\xac\xa7\xef\xb8\x8e\xe2\xa7\xab\xef\xb8\x8e\xe2\x99\x93\xef\xb8\x8e
\xe2\x96\xa0\xef\xb8\x8e\xe2\x99\x91\xef\xb8\x8e\xf0\x9f\x93\xac\xef\xb8\x8e \xf0\x9f\x95\x88\xef
\xb8\x8e\xe2\x99\x92\xef\xb8\x8e\xe2\x8d\x93\xef\xb8\x8e \xe2\x96\xa0\xef\xb8\x8e\xe2\x96\xa1\xef
\xb8\x8e\xe2\xa7\xab\xef\xb8\x8e \xe2\xa7\xab\xef\xb8\x8e\xe2\x99\x8b\xef\xb8\x8e\xf0\x9f
\x99\xb5\xe2\x99\x8f\xef\xb8\x8e \xe2\x99\x8b\xef\xb8\x8e \xe2\x97\x8f\xef\xb8\x8e\xe2\x96\xa1\xef
\xb8\x8e\xe2\x96\xa1\xef\xb8\x8e\xf0\x9f\x99\xb5 \xe2\x99\x8b\xef\xb8\x8e\xe2\xa7\xab\xef\xb8\x8e
\xe2\x99\x93\xef\xb8\x8e\xe2\xa7\xab\xef\xb8\x8e\xe2\x9c\x8d\xef\xb8\x8e
```

第 1 行，第 1 列 100% Windows (CRLF) UTF-8

所以写脚本**去除\x**，然后得到的十六进制拿去转字符，得到

Recipe

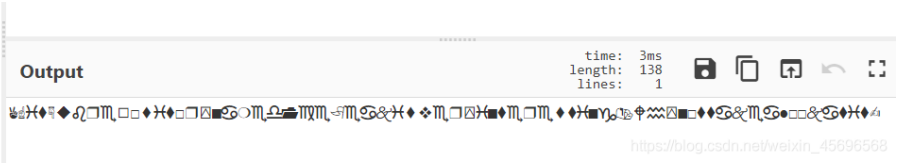
From Hex

Delimiter
Auto

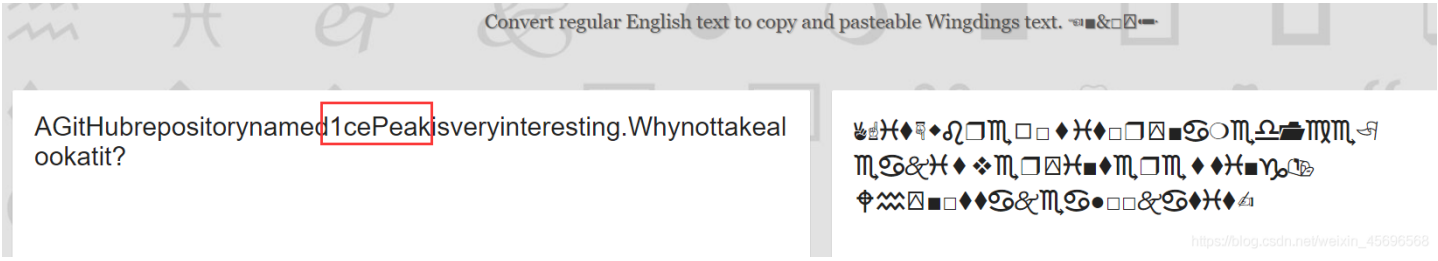
Input

length: 814
lines: 1

```
e29c8cefb88e e2989defb88ee29993efb88ee2a7abefb88ee2989fefb88ee29786efb88ee2998cefb88e
e29d92efb88ee2998fefb88ee297bbefb88ee296a1efb88ee2aca7efb88ee29993efb88ee2a7abefb88ee296a1efb88ee2
9d92efb88ee28d93efb88e e296a0efb88ee2998befb88ee29d8defb88ee2998fefb88ee2998eefb88e
f09f9382efb88ee2998defb88ee2998fefb88ef09f8fb1efb88ee2998fefb88ee2998befb88ef09f99b5
e29993efb88ee2aca7efb88e e29d96efb88ee2998fefb88ee29d92efb88ee28d93efb88e
e29993efb88ee296a0efb88ee2a7abefb88ee2998fefb88ee29d92efb88ee2998fefb88ee2aca7efb88ee2a7abefb88ee2
9993efb88ee296a0efb88ee29991efb88ef09f93acefb88e f09f9588efb88ee29992efb88ee28d93efb88e
e296a0efb88ee296a1efb88ee2a7abefb88e e2a7abefb88ee2998befb88ef09f99b5e2998fefb88e e2998befb88e
e2978fefb88ee296a1efb88ee296a1efb88ef09f99b5 e2998befb88ee2a7abefb88e
e29993efb88ee2a7abefb88ee29c8defb88e
```

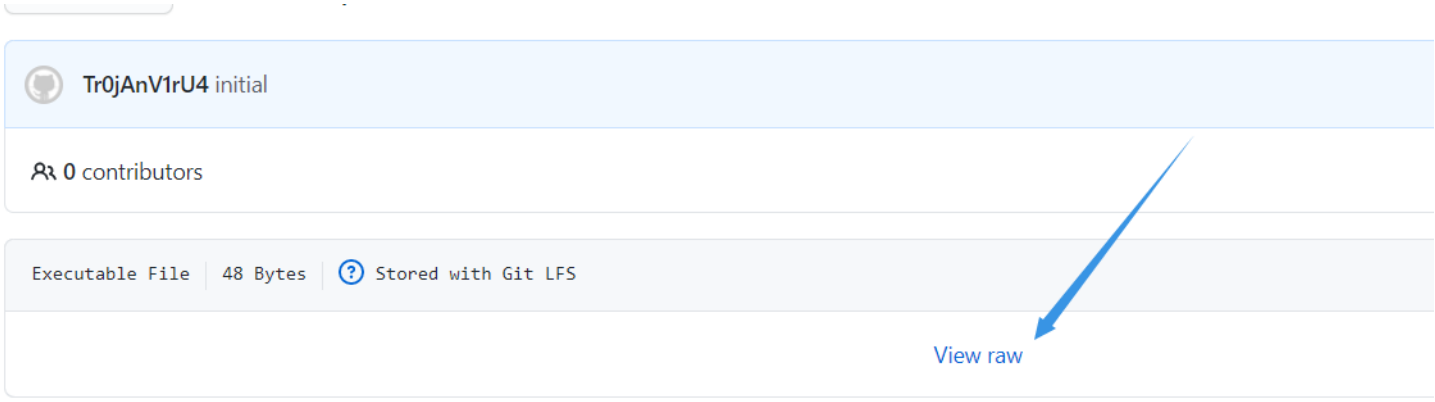


当时就卡在这里了，很可惜，当时没认出来这是什么字符，赛后发现这个考点我之前做过(淦)...它是`wingdings`，bugku的misc题目`闹酒狂欢`考到的就是这个字符，去在线网站解密



本来应该有空格分隔的，前面解码过程中把空格忽略了，不过稍微注意下能发现让我们去github上找一个名字叫`1cePeak`的项目，项目地址：https://github.com/Tr0jAnV1rU4/1cePeak/blob/main/A/post-checkout

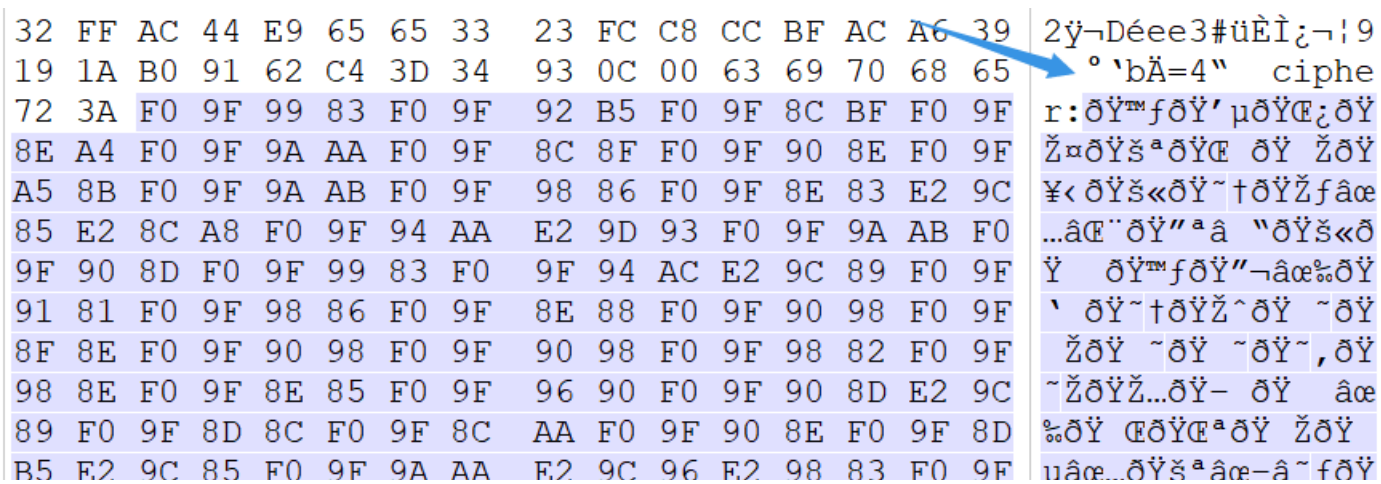
点这里把文件下载下来



用notepad打开发现内容如下

```
#!/bin/sh
echo How_6ad_c0uld_a_1cePeak_be? >&2
```

其中的`How_6ad_c0uld_a_1cePeak_be?`暂时不知道有什么用，转过头再看图片和mp3文件，发现mp3文件尾部有个`cipher`



91	A3	F0	9F	91	89	E2	84	B9	F0	9F	94	AA	F0	9F	8D
8E	F0	9F	94	84	F0	9F	91	A3	F0	9F	9A	AA	F0	9F	98
81	F0	9F	91	A3	F0	9F	92	B5	F0	9F	90	85	F0	9F	8D
B5	F0	9F	94	AC	F0	9F	9B	A9	F0	9F	98	87	F0	9F	96
90	F0	9F	96	90	F0	9F	8E	85	E2	9C	85	F0	9F	8F	8E
F0	9F	91	8C	F0	9F	9A	A8	F0	9F	98	86	F0	9F	8E	A4
F0	9F	8E	85	F0	9F	A6	93	F0	9F	8C	BF	F0	9F	A6	93
F0	9F	99	83	E2	9C	96	F0	9F	8D	8C	F0	9F	9B	A9	F0
9F	98	82	F0	9F	91	91	F0	9F	8C	8F	E2	98	83	F0	9F
98	87	F0	9F	98	8D	F0	9F	9B	A9	F0	9F	9A	B9	F0	9F
98	80	F0	9F	8D	8C	F0	9F	8E	88	F0	9F	92	A7	F0	9F
97	92	F0	9F	97	92										

‘£ðŸ‘‰â„¹ðŸ”ªðŸ Žð Ÿ”„ðŸ‘‰£ðŸšªðŸ˜ ðŸ‘‰£ðŸ’µðŸ …ðŸ µðŸ”¬ðŸ ›©ðŸ˜‡ðŸ ðŸ‘– ðŸŽ…âœ…ðŸ Ž ðŸ‘‰ŒðŸš¨ðŸ˜†ðŸ ›©ðŸ˜¡ðŸ ðŸ‘Ž…ðŸ¦ðŸ‘Œ™ƒâœ–ðŸ ŒðŸ ›©ðŸ˜‰,ðŸ‘‰ðŸŒ â˜ƒðŸ ‡ðŸ˜‰ðŸš¹ðŸ ðŸ˜€ðŸ ŒðŸŽ^ðŸ’§ðŸ —’ðŸ—’

https://blog.csdn.net/weixin_45696568

于是把后面的内容的十六进制值复制出来，拿去转字符，解出一堆emoji

Input length: 680 lines: 1

```
F09F9983F09F92B5F09F8CBFF09F8EA4F09F9AAAF09F8C8FF09F908EF09FA58BF09F9AABF09F9886F09F8E83E29C85E28C
A8F09F94AAE29D93F09F9AABF09F908DF09F9983F09F94ACE29C89F09F9181F09F9886F09F8E88F09F9098F09F8F8EF09F
9098F09F9098F09F9882F09F988EF09F8E85F09F9690F09F908DE29C89F09F8D8CF09F8CAAF09F908EF09F8DB5E29C85F0
9F9AAAE29C96E29883F09F91A3F09F9189E284B9F09F94AAF09F8D8EF09F9484F09F91A3F09F9AAAF09F9881F09F91A3F0
9F92B5F09F9085F09F8DB5F09F94ACF09F9BA9F09F9887F09F9690F09F9690F09F8E85E29C85F09F8F8EF09F918CF09F9A
A8F09F9886F09F8EA4F09F8E85F09FA693F09F8CBFF09FA693F09F9983E29C96F09F8D8CF09F9BA9F09F9882F09F9191F0
9F8C8FE29883F09F9887F09F988DF09F9BA9F09F9AB9F09F9880F09F8D8CF09F8E88F09F92A7F09F9792F09F9792
```

Output time: 2ms length: 164 lines: 1

https://blog.csdn.net/weixin_45696568

于是用emoji-aes解密,这里的key是刚才解出的字符，成功解出flag

Decrypt

To decrypt, select the agreed rotation (if custom), enter the emoji-aes string, and then the pre-shared encryption key.

Advanced

Message

Key

••••••••••••••••••••••••••

Decrypt

https://blog.csdn.net/weixin_45696568

I_will_but_not_quite

给了一个加密的py脚本和`Twin.vmem`，很明显要内存取证，拿到关键信息，再回来写解密脚本。

先分析镜像：`python vol.py -f Twin.vmem imageinfo`

知道系统版本是**Win7SP1x64**

然后分析一下进程：`python vol.py -f Twin.vmem --profile=Win7SP1x64 pslist`

```
Offset(V)          Name                    PID   PPID   Thds     Hnds   Sess  Wow64 Start                          Exit
------------------ -------------------- ------ ------ ------ -------- ------ ------ ------------------------------ ------------------------------
0xfffffa8000cbfb30 System                    4      0     84      515 ------      0 2021-03-29 09:44:08 UTC+0000

0xfffffa80012734f0 smss.exe                256      4      2       29 ------      0 2021-03-29 09:44:08 UTC+0000

0xfffffa8001e1c3d0 csrss.exe               332    324      9      416      0      0 2021-03-29 09:44:15 UTC+0000

0xfffffa80019e5b30 csrss.exe               384    376     11      310      1      0 2021-03-29 09:44:16 UTC+0000

0xfffffa8001ef3920 wininit.exe             392    324      3       77      0      0 2021-03-29 09:44:16 UTC+0000

0xfffffa8001ebe910 winlogon.exe            424    376      3      113      1      0 2021-03-29 09:44:16 UTC+0000

0xfffffa8001efeb30 services.exe            488    392      7      194      0      0 2021-03-29 09:44:17 UTC+0000

0xfffffa8001f35330 lsass.exe               496    392      7      592      0      0 2021-03-29 09:44:18 UTC+0000

0xfffffa8001f42b30 lsm.exe                 508    392     10      141      0      0 2021-03-29 09:44:18 UTC+0000

0xfffffa80022bf6f0 svchost.exe             600    488     11      347      0      0 2021-03-29 09:44:23 UTC+0000

0xfffffa80022ddb30 svchost.exe             672    488      8      275      0      0 2021-03-29 09:44:24 UTC+0000

0xfffffa800231a700 svchost.exe             764    488     20      459      0      0 2021-03-29 09:44:24 UTC+0000

0xfffffa8002328210 svchost.exe             800    488     16      368      0      0 2021-03-29 09:44:25 UTC+0000

0xfffffa8002357660 svchost.exe             824    488     33      948      0      0 2021-03-29 09:44:25 UTC+0000

0xfffffa800237fb30 svchost.exe             992    488     10      520      0      0 2021-03-29 09:44:26 UTC+0000

0xfffffa80023da390 svchost.exe             344    488     15      483      0      0 2021-03-29 09:44:27 UTC+0000

0xfffffa80024197d0 spoolsv.exe            1032    488     12      315      0      0 2021-03-29 09:44:29 UTC+0000

0xfffffa800242cb30 svchost.exe            1072    488     19      307      0      0 2021-03-29 09:44:29 UTC+0000

0xfffffa800228ab30 vmtoolsd.exe           1244    488      9      281      0      0 2021-03-29 09:44:31 UTC+0000

0xfffffa8002601b30 taskhost.exe           1508    488      9      206      1      0 2021-03-29 09:44:35 UTC+0000
```

```
0xfffffa8002610b30 dwm.exe                1724    800      3       68      1      0 2021-03-29 09:44:36 UTC+0000
0xfffffa800101bb30 TPAutoConnSvc.         1760    488     10      140      0      0 2021-03-29 09:44:36 UTC+0000
0xfffffa8002674b30 explorer.exe           1792   1636     44      879      1      0 2021-03-29 09:44:37 UTC+0000
0xfffffa80025c5b30 dllhost.exe            2024    488     13      186      0      0 2021-03-29 09:44:38 UTC+0000
0xfffffa8002526b30 TPAutoConnect.         1356   1760      5      118      1      0 2021-03-29 09:44:39 UTC+0000
0xfffffa8002713060 conhost.exe            1428    384      1       32      1      0 2021-03-29 09:44:39 UTC+0000
0xfffffa8002537b30 msdtc.exe              1744    488     12      144      0      0 2021-03-29 09:44:40 UTC+0000
0xfffffa8002840b30 vmtoolsd.exe           2160   1792      7      297      1      0 2021-03-29 09:44:45 UTC+0000
0xfffffa80028a6b30 SearchIndexer.         2416    488     11      656      0      0 2021-03-29 09:44:53 UTC+0000
0xfffffa80028c1b30 jusched.exe            2496   2200      6      377      1      1 2021-03-29 09:44:55 UTC+0000
0xfffffa80029d6680 svchost.exe            2748    488      7      110      0      0 2021-03-29 09:45:05 UTC+0000
0xfffffa80023ef990 svchost.exe             860    488     13      333      0      0 2021-03-29 09:46:36 UTC+0000
0xfffffa80019dfb30 WmiPrvSE.exe           1440    600      7      109      0      0 2021-03-29 09:48:34 UTC+0000
0xfffffa8002749b30 jucheck.exe            2960   2496      7      368      1      1 2021-03-29 09:50:24 UTC+0000
0xfffffa8001e38b30 javaws.exe              400   2960      0 --------      1      0 2021-03-29 09:50:24 UTC+0000
   2021-03-29 09:50:24 UTC+0000
0xfffffa8001d0d200 jp2launcher.ex         1932    400     27      439      1      0 2021-03-29 09:50:24 UTC+0000
0xfffffa8001fc9060 taskeng.exe            3044    824      4       83      1      0 2021-03-30 07:52:37 UTC+0000
0xfffffa800282eb30 SearchProtocol         2020   2416      8      321      0      0 2021-03-30 07:55:59 UTC+0000
0xfffffa8001fb22b0 SearchFilterHo         3024   2416      5       98      0      0 2021-03-30 07:55:59 UTC+0000
0xfffffa8001fbb990 WinRAR.exe             1696   1792     18      564      1      0 2021-03-30 07:56:21 UTC+0000
```

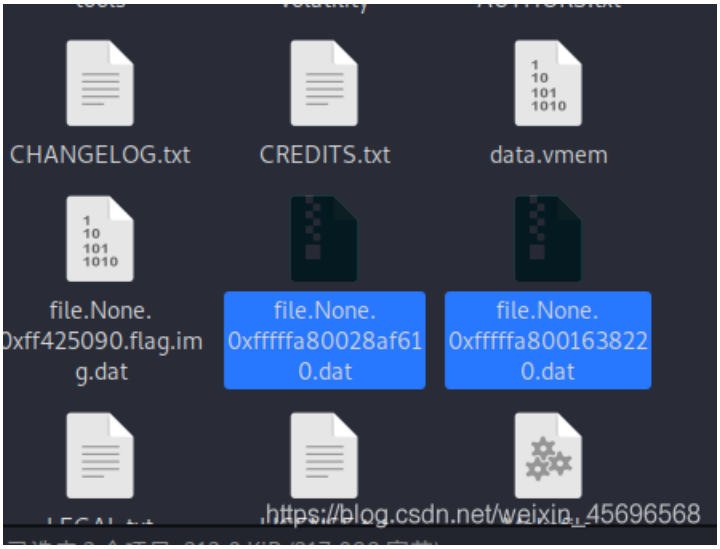
发现`WinRAR.exe`有点可疑，可能进行了压缩或者解压的操作，用`filescan`配合`grep`命令查一下rar和zip文件

```
┌──(volcano㉿kali)-[~/桌面/volatility-master]
└─$ python vol.py -f Twin.vmem --profile=Win7SP1x64 filescan | grep zip                    130 ×
Volatility Foundation Volatility Framework 2.6.1
0x000000003e23ab50     16      0 RW-rw- \Device\HarddiskVolume1\Users\Administrator\AppData\Roaming\Microso
ft\Windows\Recent\sea.zip.lnk
0x000000003e557990      2      0 RW---- \Device\HarddiskVolume1\Users\Administrator\Desktop\sea.zip
0x000000003e63a310      1      0 R-Dr-d \Device\HarddiskVolume1\$Recycle.Bin\S-1-5-21-3891451472-281351741-
2593777832-500\$IN5QJA1.zip
0x000000003e8ab810      2      0 -W---- \Device\HarddiskVolume1\$Recycle.Bin\S-1-5-21-3891451472-281351741-
2593777832-500\$IU8BK03.zip
0x000000003eceaa20     16      0 -W-rw- \Device\HarddiskVolume1\Program Files\WinRAR\zipnew.dat
0x000000003ecf1f20     15      0 R--r-d \Device\HarddiskVolume1\Windows\System32\zipfldr.dll
0x000000003ed15070      2      0 RW---- \Device\HarddiskVolume1\Users\Administrator\Desktop\倒影.zip
0x000000003ed3b070     10      0 R--r-d \Device\HarddiskVolume1\Program Files\Java\jre1.8.0_271\bin\sec569.dll
                                                                    https://blog.csdn.net/weixin_45696568
```

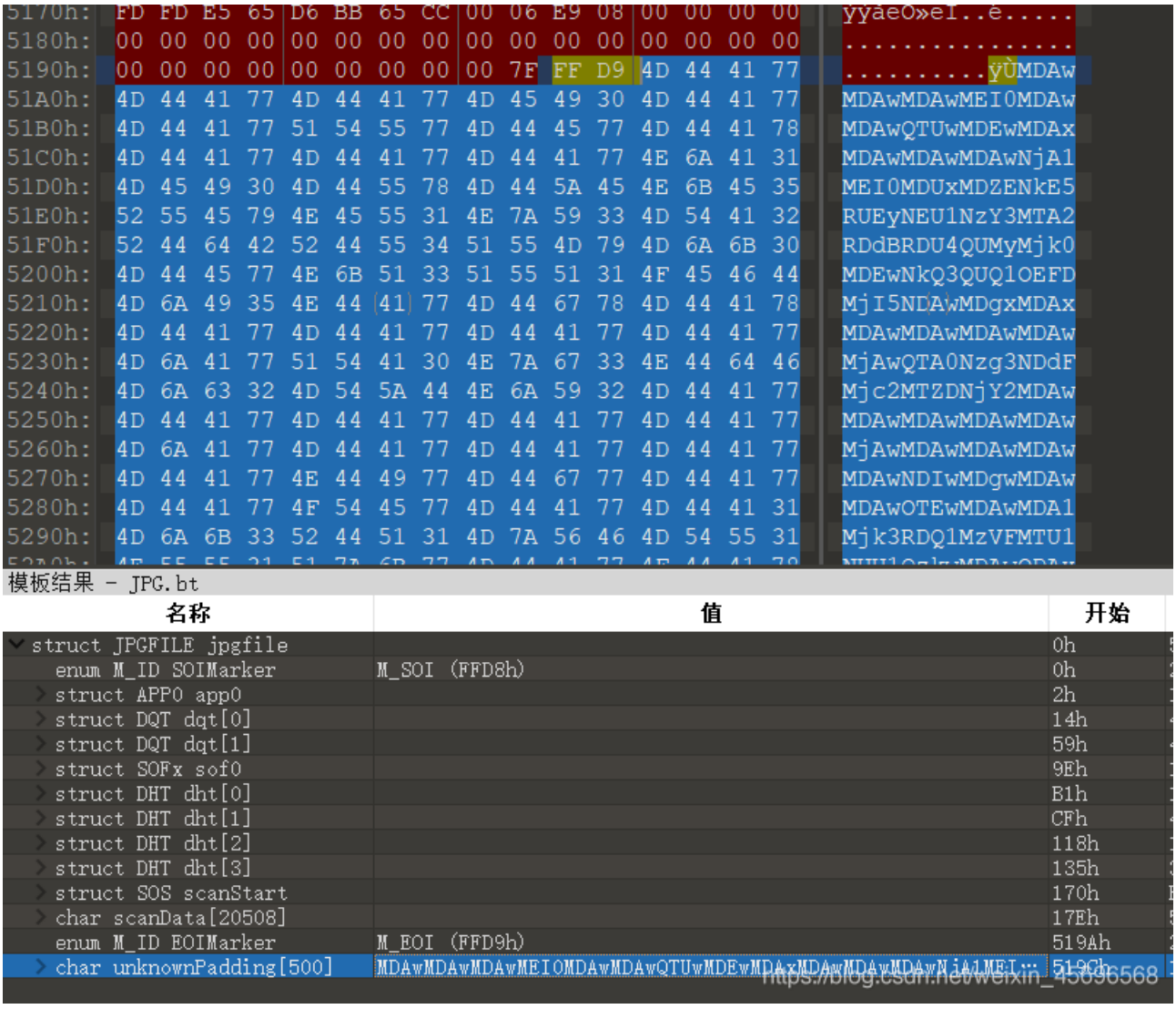
把这俩给dump出来

```
python vol.py -f Twin.vmem --profile=Win7SP1x64 dumpfiles -Q 0x000000003ed15070 --dump-dir=./
python vol.py -f Twin.vmem --profile=Win7SP1x64 dumpfiles -Q 0x000000003e557990 --dump-dir=./
```

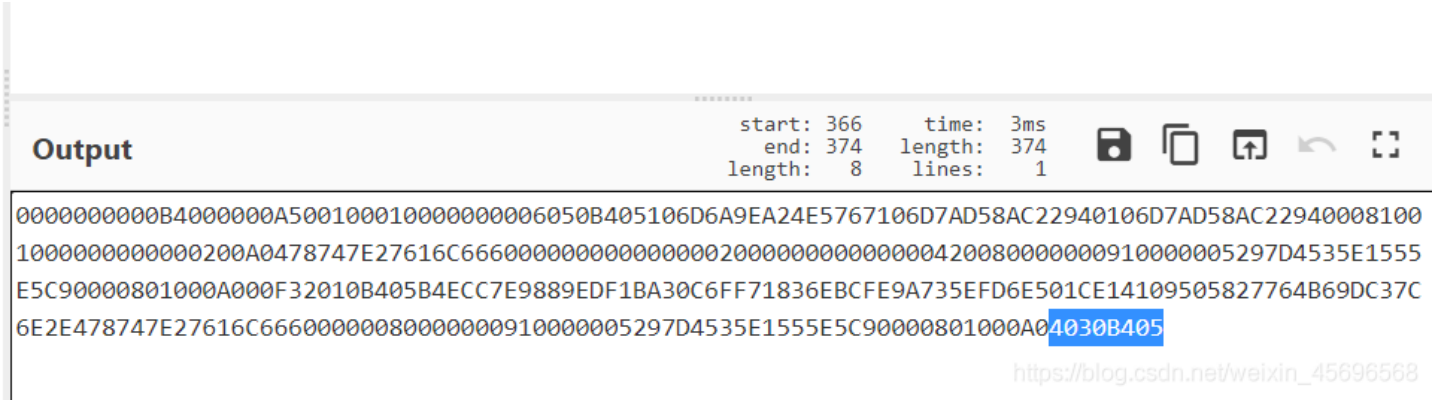
得到这两个文件，从kali拉回win10，改后缀然后分析



倒影.zip打开后把里面的后缀改成jpg，拖进winhex里分析发现尾部有一串base64

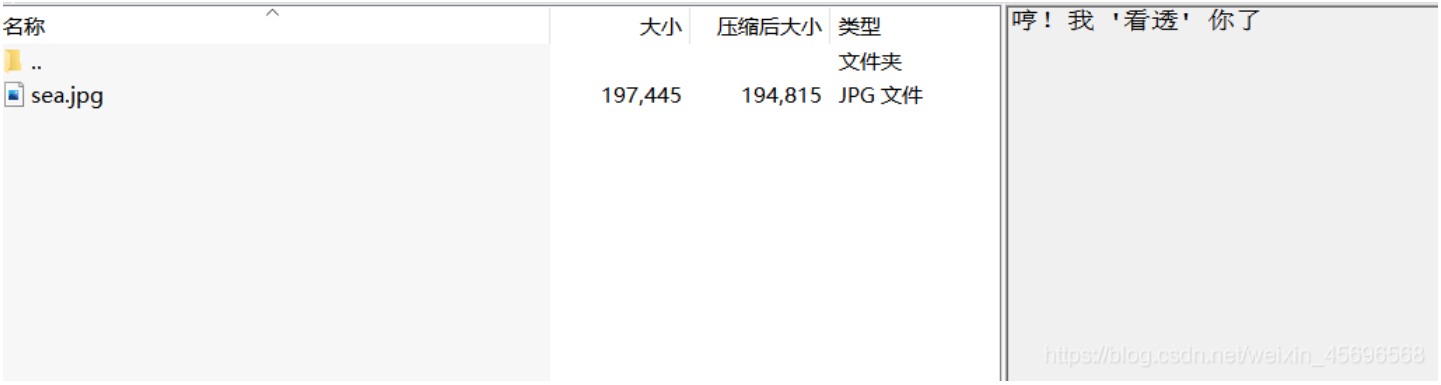


解码后看最后几个数字，结合文件名`倒影`，把数据倒过来，保存为一个`1.zip`

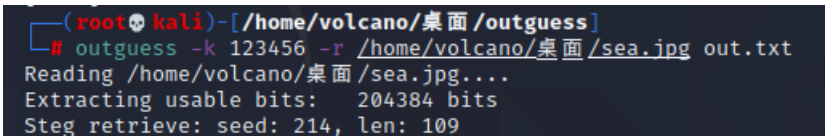


里面有个flag.txt，不过加密了打不开，也不知道是不是出题人故意搞人心态的

另一个压缩包sea.zip的注释信息中有提示



这里的`看透`其实是暗指`outguess`隐写，测试后发现密码为123456



解出的内容如下：

```
4266gj2zn17b2jo5b62k73g22xg6j658350r5771vd40h4bd2ns33q30651y57s6752su3q05881hs3h53nb3603co2mv40l58n3da3f61i5
```

直接十六进制转字符，发现解出的是乱码。其实这个是**双□六进制编码**(`twin-hex`)，没遇到过，但是题目中有提示，刚才内存取证分析的文件明就是`twin`，到在线网站解得

```
Vnw3HC07BDgbBWNRGTx2fSckf399V1Z9CxlvHVd6fHsaEnR8fX40NyQ7JhM8CWV5fgMNN24=
```

最后一步就是写解密脚本了，这里直接用套神mumuzi的，套神博客：https://blog.csdn.net/qq_42880719

```
import base64
import random
secret = "Vnw3HC07BDgbBWNRGTx2fSckf399V1Z9CxIvHVd6fHsaEnR8fX40NyQ7JhM8CWV5fgMNN24="
dec = base64.b64decode(secret).decode("utf-8")
# for i in range(len(dec)):
#  print(ord(dec[i]))
def r(s, num): #凯撒
 l=""
 for i in s:
  if(ord(i) in range(97,97+26)):
   l+=chr((ord(i)-97+num)%26+97)
  else:
   l+=i
 return l

for i in range(86,128):
 j = 1
 tmp = [""]*len(dec)
 tmp[-1] = chr(i)#爆破恢复最后一位，即可恢复所有
 while j != len(dec):
  tmp[-j-1] = chr(ord(dec[-j])^ord(tmp[-j])) #反着进行异或
  j += 1
 s = tmp[-1] #因为最后一位是最后一位和第一位异或，所以刚开始异或的其实是最后一位
 for i in range(len(tmp)-1):
  s += tmp[i]#这里即是将第2位至最后一位拼接起来加在第一位后面
 try:
  s = base64.b64decode(s).decode("utf-8")
  for i in range(1,26):#遍历凯撒
   flag = r(s,i)
   print(flag)
 except:
  pass
```

嫌疑人x的硬盘

我太菜了，照着大佬们的wp都没复现出来，先不写了…