i春秋 第二届春秋欢乐赛 ReCreators

原创

sps98 于 2019-11-08 23:11:45 发布 676 收藏 1

分类专栏： ctf Misc

ctf 同时被 2 个专栏收录

11 篇文章 0 订阅

订阅专栏

Misc

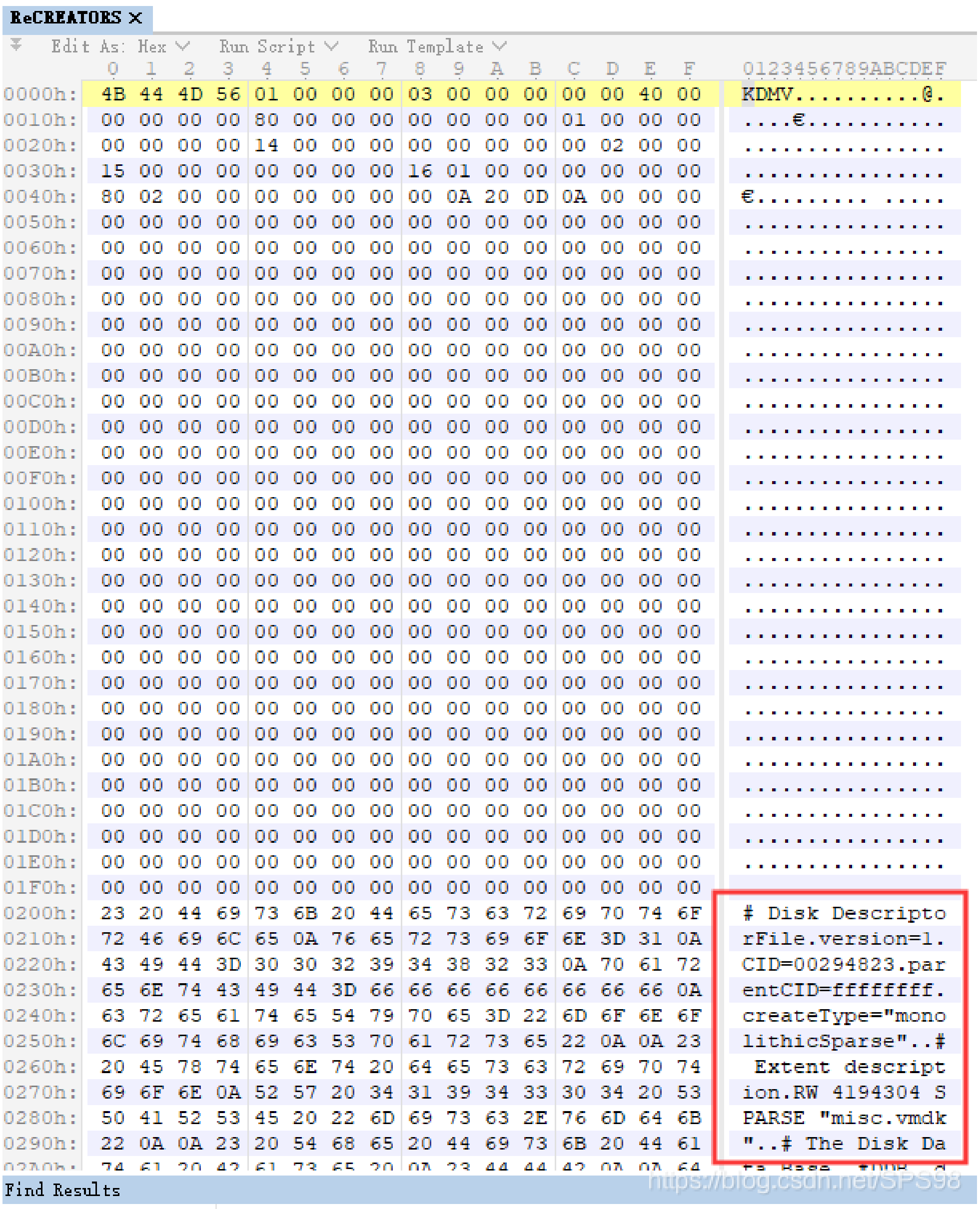
4 篇文章 0 订阅

订阅专栏

依旧不会做，在开头被卡住了，最后看了师傅们的题解跟着复现了下，

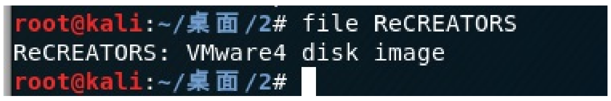
考点比较基础，但不得不说这题实在坑，套了好几层编码。

题目给了个27M的无扩展名文件，推测是程序，然鹅并不是，用010打开可以看到一些明文,



```
        0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   0123456789ABCDEF
0000h: 4B 44 4D 56 01 00 00 00 03 00 00 00 00 00 40 00  KDMV..........@.
0010h: 00 00 00 00 80 00 00 00 00 00 00 00 01 00 00 00  ....€...........
0020h: 00 00 00 00 14 00 00 00 00 00 00 00 00 02 00 00  ................
0030h: 15 00 00 00 00 00 00 00 16 01 00 00 00 00 00 00  ................
0040h: 80 02 00 00 00 00 00 00 00 0A 20 0D 0A 00 00 00  €........ .....
0050h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0060h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0070h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0080h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0090h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00A0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00B0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00C0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00D0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00E0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00F0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0100h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0110h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0120h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0130h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0140h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0150h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0160h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0170h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0180h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0190h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
01A0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
01B0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
01C0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
01D0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
01E0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
01F0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0200h: 23 20 44 69 73 6B 20 44 65 73 63 72 69 70 74 6F  # Disk Descripto
0210h: 72 46 69 6C 65 0A 76 65 72 73 69 6F 6E 3D 31 0A  rFile.version=1.
0220h: 43 49 44 3D 30 30 32 39 34 38 32 33 0A 70 61 72  CID=00294823.par
0230h: 65 6E 74 43 49 44 3D 66 66 66 66 66 66 66 66 0A  entCID=ffffffff.
0240h: 63 72 65 61 74 65 54 79 70 65 3D 22 6D 6F 6E 6F  createType="mono
0250h: 6C 69 74 68 69 63 53 70 61 72 73 65 22 0A 0A 23  lithicSparse"..#
0260h: 20 45 78 74 65 6E 74 20 64 65 73 63 72 69 70 74   Extent descript
0270h: 69 6F 6E 0A 52 57 20 34 31 39 34 33 30 34 20 53  ion.RW 4194304 S
0280h: 50 41 52 53 45 20 22 6D 69 73 63 2E 76 6D 64 6B  PARSE "misc.vmdk
0290h: 22 0A 0A 23 20 54 68 65 20 44 69 73 6B 20 44 61  "..# The Disk Da
```

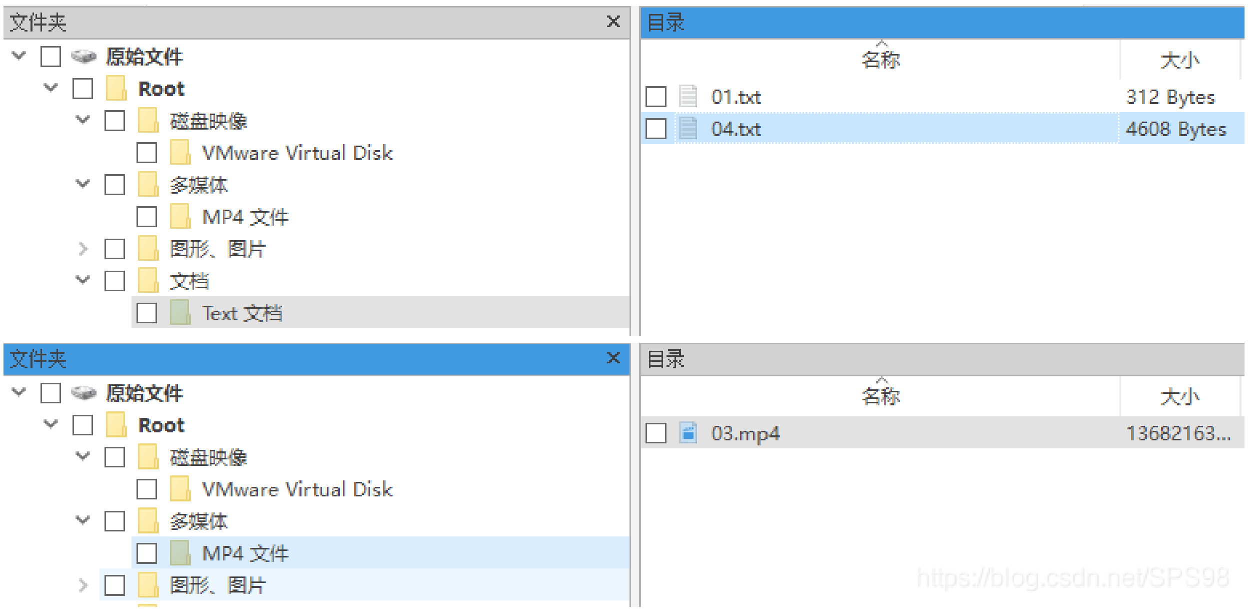
猜测是磁盘文件之类的, 用`file`查看了后发现是磁盘镜像文件.



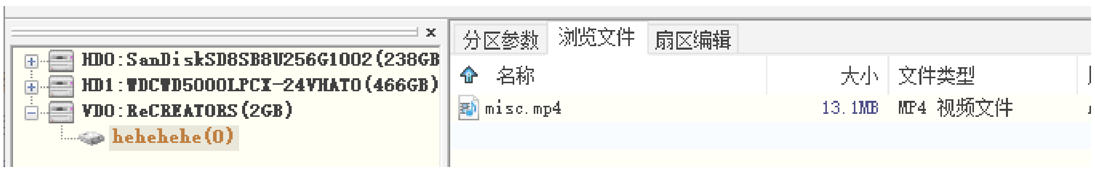
```
root@kali:~/桌面/2# file ReCREATORS
ReCREATORS: VMware4 disk image
root@kali:~/桌面/2#
```

然后有三种办法:

1. 使用加个 `.vmdk` 的扩展名(可选), 然后用vm挂载磁盘, 但是一开始不会操作, 就放弃了,
   写文的时候百度了下挂载方法, 成功在挂载给虚拟机了, 但是不知道系统怎么操作, 就又弃坑了, 头大.
2. 使用R-Studio打开, 然后点 `驱动器` - `扫描` , 勾上 `搜索已知文件类型` , 待扫描完成, 接着点开 `原始文件` - `文档` - `Text文档` , 就会看到两个文本, 而 `04.txt` 就是flag数据, 但是按照其他师傅以及复现结果来看, 提取到的flag数据是残缺的, 大概差了100多字节, 导致解不出来. 或者把 `misc.mp4` 恢复出来, 手动提取flag数据.

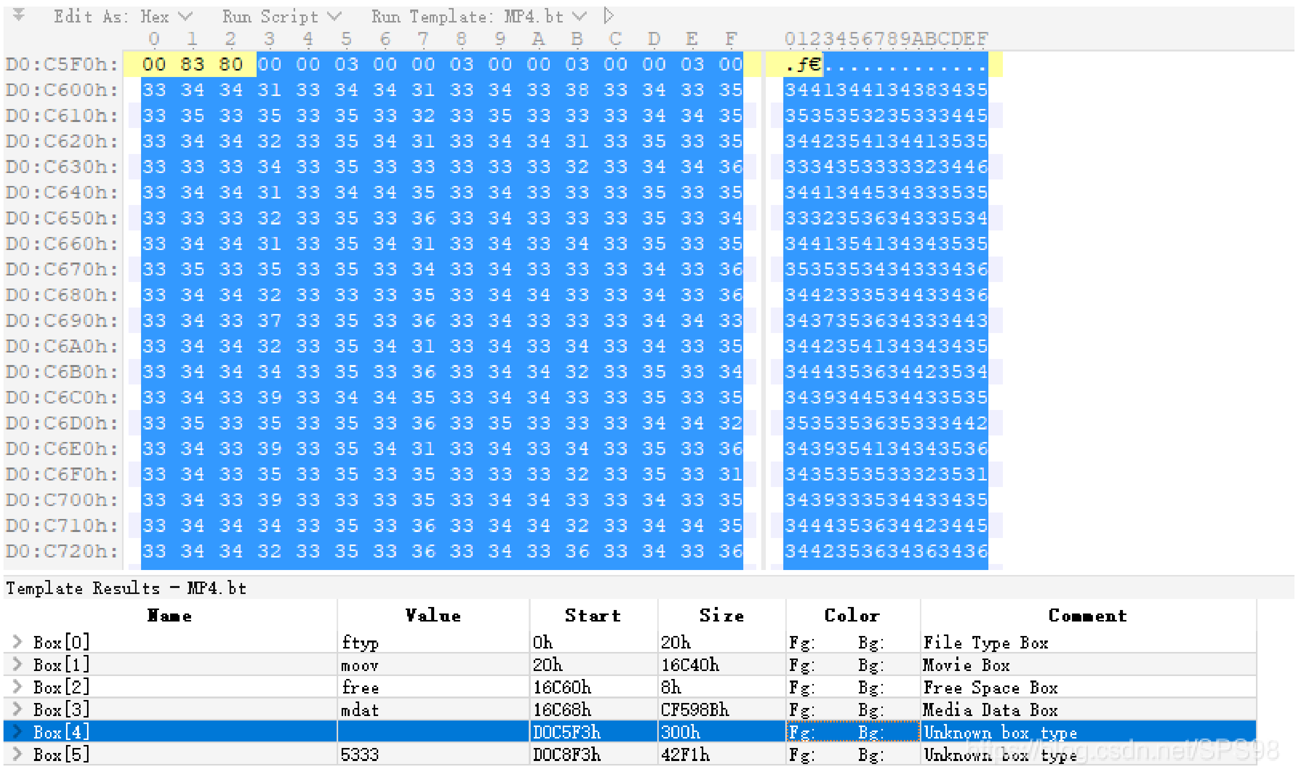


3. 使用DiskGenius打开镜像文件, 把 `misc.mp4` 提取出来:



按照套路, 先打开mp4看一看, 是个二次元动画,
不过居然能播放完(不知道哪个题解说会出错的, 可能是win10自己跳过错误部分了?)

嗯... 那就先不管这么多好了, 用010看看文件内容,
(其实这边应该也要去推测下有没有音频隐写的行为, 但是因为看了wp, 就绕过了)

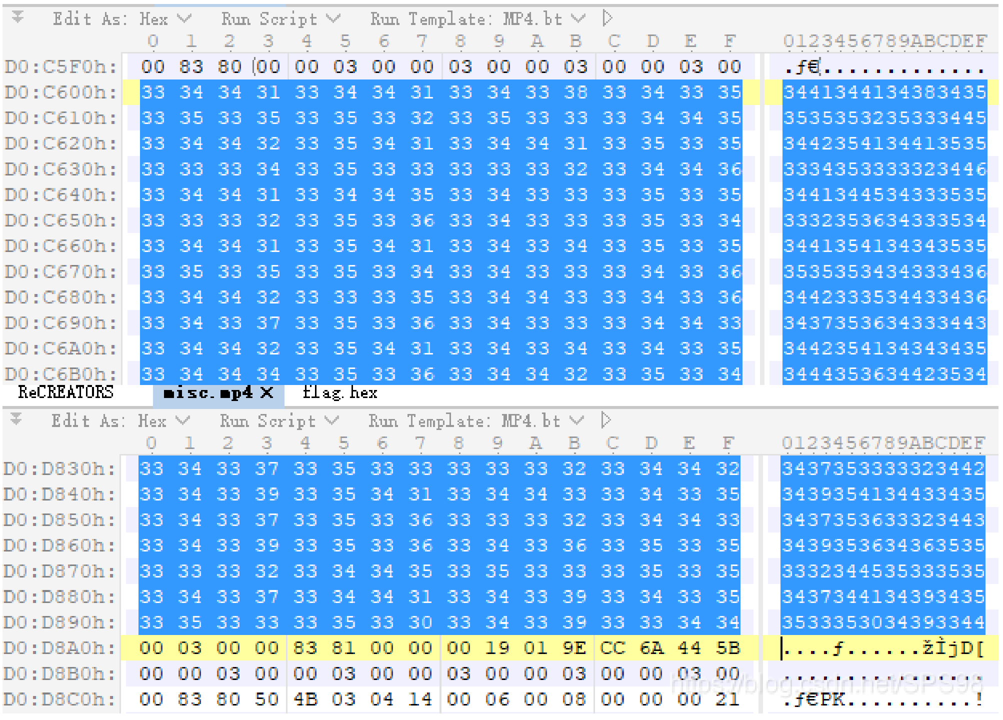


```
Edit As: Hex ∨   Run Script ∨   Run Template: MP4.bt ∨  ▷
            0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   0123456789ABCDEF
D0:C5F0h:  00 83 80 00 00 03 00 00 03 00 00 03 00 00 03 00   .ƒ€.............
D0:C600h:  33 34 34 31 33 34 34 31 33 34 33 38 33 34 33 35   3441344134383435
D0:C610h:  33 35 33 35 33 35 33 32 33 35 33 33 33 34 34 35   3535353235333445
D0:C620h:  33 34 34 32 33 35 34 31 33 34 34 31 33 35 33 35   3442354134413535
D0:C630h:  33 33 33 34 33 35 33 33 33 33 33 32 33 34 34 36   3334353333323446
D0:C640h:  33 34 34 31 33 34 34 35 33 34 33 33 33 35 33 35   3441344534333535
D0:C650h:  33 33 33 32 33 35 33 36 33 34 33 33 33 35 33 34   3332353634333534
D0:C660h:  33 34 34 31 33 35 34 31 33 34 33 34 33 35 33 35   3441354134343535
D0:C670h:  33 35 33 35 33 35 33 34 33 34 33 33 33 34 33 36   3535353434333436
D0:C680h:  33 34 34 32 33 33 33 35 33 34 34 33 33 34 33 36   3442333534433436
D0:C690h:  33 34 33 37 33 35 33 36 33 34 33 33 33 34 34 33   3437353634333443
D0:C6A0h:  33 34 34 32 33 35 34 31 33 34 33 34 33 34 33 35   3442354134343435
D0:C6B0h:  33 34 34 34 33 35 33 36 33 34 34 32 33 35 33 34   3444353634423534
D0:C6C0h:  33 34 33 39 33 34 34 35 33 34 34 33 33 35 33 35   3439344534433535
D0:C6D0h:  33 35 33 35 33 35 33 36 33 35 33 33 33 34 34 32   3535353635333442
D0:C6E0h:  33 34 33 39 33 35 34 31 33 34 33 34 33 35 33 36   3439354134343536
D0:C6F0h:  33 34 33 35 33 35 33 35 33 33 33 32 33 35 33 31   3435353533323531
D0:C700h:  33 34 33 39 33 33 33 35 33 34 34 33 33 34 33 35   3439333534433435
D0:C710h:  33 34 34 34 33 35 33 36 33 34 34 32 33 34 34 35   3444353634423445
D0:C720h:  33 34 34 32 33 35 33 36 33 34 33 36 33 34 33 36   3442353634363436
```

Template Results - MP4.bt

Name	Value	Start	Size	Color		Comment
Box[0]	ftyp	0h	20h	Fg:	Bg:	File Type Box
Box[1]	moov	20h	16C40h	Fg:	Bg:	Movie Box
Box[2]	free	16C60h	8h	Fg:	Bg:	Free Space Box
Box[3]	mdat	16C68h	CF598Bh	Fg:	Bg:	Media Data Box
Box[4]		D0C5F3h	300h	Fg:	Bg:	Unknown box type
Box[5]	5333	D0C8F3h	42F1h	Fg:	Bg:	Unknown box type

根据插件的解析结果, 发现有两个未知类型的数据段, 可以直接显示成文本, 那这边就把它先提取出来, 保存为 `flag.hex` .



```
Edit As: Hex ∨   Run Script ∨   Run Template: MP4.bt ∨  ▷
            0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   0123456789ABCDEF
D0:C5F0h:  00 83 80 00 00 03 00 00 03 00 00 03 00 00 03 00   .ƒ€.............
D0:C600h:  33 34 34 31 33 34 34 31 33 34 33 38 33 34 33 35   3441344134383435
D0:C610h:  33 35 33 35 33 35 33 32 33 35 33 33 33 34 34 35   3535353235333445
D0:C620h:  33 34 34 32 33 35 34 31 33 34 34 31 33 35 33 35   3442354134413535
D0:C630h:  33 33 33 34 33 35 33 33 33 33 33 32 33 34 34 36   3334353333323446
D0:C640h:  33 34 34 31 33 34 34 35 33 34 33 33 33 35 33 35   3441344534333535
D0:C650h:  33 33 33 32 33 35 33 36 33 34 33 33 33 35 33 34   3332353634333534
D0:C660h:  33 34 34 31 33 35 34 31 33 34 33 34 33 35 33 35   3441354134343535
D0:C670h:  33 35 33 35 33 35 33 34 33 34 33 33 33 34 33 36   3535353434333436
D0:C680h:  33 34 34 32 33 33 33 35 33 34 34 33 33 34 33 36   3442333534433436
D0:C690h:  33 34 33 37 33 35 33 36 33 34 33 33 33 34 34 33   3437353634333443
D0:C6A0h:  33 34 34 32 33 35 34 31 33 34 33 34 33 34 33 35   3442354134343435
D0:C6B0h:  33 34 34 34 33 35 33 36 33 34 34 32 33 35 33 34   3444353634423534
ReCREATORS    misc.mp4 ×    flag.hex
Edit As: Hex ∨   Run Script ∨   Run Template: MP4.bt ∨  ▷
            0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   0123456789ABCDEF
D0:D830h:  33 34 33 37 33 35 33 33 33 33 33 32 33 34 34 32   3437353333323442
D0:D840h:  33 34 33 39 33 35 34 31 33 34 34 33 33 34 33 35   3439354134433435
D0:D850h:  33 34 33 37 33 35 33 36 33 33 33 32 33 34 34 33   3437353633323443
D0:D860h:  33 34 33 39 33 35 33 36 33 34 33 36 33 35 33 35   3439353634363535
D0:D870h:  33 33 33 32 33 34 34 35 33 35 33 33 33 35 33 35   3332344535333535
D0:D880h:  33 34 33 37 33 34 34 31 33 34 33 39 33 34 33 35   3437344134393435
D0:D890h:  33 35 33 33 33 35 33 30 33 34 33 39 33 33 34 34   3533353034393344
D0:D8A0h:  00 03 00 00 83 81 00 00 00 19 01 9E CC 6A 44 5B   ....ƒ......žÌjD[
D0:D8B0h:  00 00 03 00 00 03 00 00 03 00 00 03 00 00 03 00   ................
D0:D8C0h:  00 83 80 50 4B 03 04 14 00 06 00 08 00 00 00 21   .ƒ€PK..........!
```

因为数据段可以显示成明文, 所以猜测是做了Hex(Base16)处理, 于是做一次解码.

> 以前都是用工具配合手动分割做半手解, 但这次数据实在太长了, 只好写代码了(说白了还是工具不行),
> 其实也有一些在线网站可以直接解未做处理的HEX, 但是保不齐哪天做题不让联网, 所以还是备一些离线工具(手写的也行)

因为已经解过一次了, 所以不想再手解一次, 转为~~轻松一点的~~写个脚本处理. (Python3)

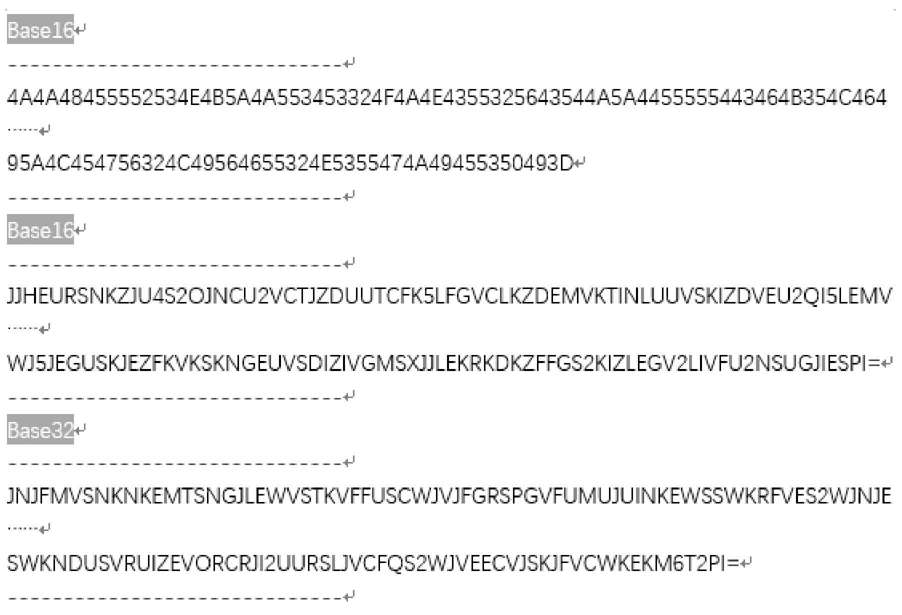
```python
import binascii
import base64


def get_continuous_asciis(ranges: list):
    """
    取连续的ASCII文本
    :param ranges: 范围数组, 例[['A', 'B']]
    :return:
    """
    return ''.join([''.join(map(chr, range(ord(r[0]), ord(r[1]) + 1))) for r in ranges])


with open('flag.hex') as f:
    enc = f.read()

while True:
    b16 = b32 = b64 = False
    if all([i.upper() in '0123456789ABCDEF' for i in enc]):
        b16 = True
    elif all([i.upper() in 'ABCDEFGHIJKLMNOPQRSTUVWXYZ2345678=' for i in enc]):
        b32 = True
    elif all([i.upper() in get_continuous_asciis([['a', 'z'], ['A', 'Z'], ['0', '9']]) + '+/=' for i in enc]):
        b64 = True
    else:
        print('End')
        break
    if b16:
        print('Base16')
        enc = binascii.a2b_hex(enc).decode('utf-8')
    elif b32:
        print('Base32')
        enc = base64.b32decode(enc).decode('utf-8')
    elif b64:
        print('Base64')
        enc = base64.b64decode(enc).decode('utf-8')
    print('-' * 30)
    print(enc)
    print('-' * 30)
```

效果如下(因为部分文本过长, 做了截取):



```
Base16↵
------------------------------↵
4A4A48455552534E4B5A4A553453324F4A4E4355325643544A5A4455555443464B354C464
······↵
95A4C454756324C49564655324E5355474A49455350493D↵
------------------------------↵
Base16↵
------------------------------↵
JJHEURSNKZJU4S2OJNCU2VCTJZDUUTCFK5LFGVCLKZDEMVKTINLUUVSKIZDVEU2QI5LEMV
······↵
WJ5JEGUSKJEZFKVKSKNGEUVSDIZIVGMSXJJLEKRKDKZFFGS2KIZLEGV2LIVFU2NSUGJIESPI=↵
------------------------------↵
Base32↵
------------------------------↵
JNJFMVSNKNKEMTSNGJLEWVSTKVFFUSCWJVJFGRSPGVFUMUJUINKEWSSWKRFVES2WJNJE
······↵
SWKNDUSVRUIZEVORCRJI2UURSLJVCFQS2WJVEECVJSKJFVCWKEKM6T2PI=↵
------------------------------↵
```

```
Base32
-------------------------------
KRVVMSTFNM2VKVSUJZHVMRSFO5KFQ4CTKJVTKRKVKRBE4YLMIUYVI3CWIZGVKOKFKZKF
……
GBKGWUTSJVKTC4KVLBUE6VSGIV4FIWDQJ5JFKMDXKVMHAU2RKQYDS===
-------------------------------
Base32
-------------------------------
TkVJek5UVTNOVFEwTXpSRk5EUTBNalE1TlVFMU9EVTROVFUxUVRRek5EVTBRVFZCTlRVME5
qUTFOVEkxTXpSQk5FWXpOVFJETlRVME9UVXpOVE0wT0RSRE5FRTFOelUyTkRVMFJUVXpO
VEkwUlRWQk5FTTBOelF6TlRVMU1qVXlORU0wT1RNeU5UWTBSak16TkVFMU5qUk5OVUUx
TXpVMU5FUTFRVE16TlRVkwTlRRMU16TTBNelU0TkRrMU1qUXhOVFExTXpOR
U0wUXpSQT09
-------------------------------
Base64
-------------------------------
NEIzNTU3NTQ0MzRFNDQ0MjQ5NUE1ODU4NTU1QTQzNDU0QTVBNTU0NjQ1NTI1MzRBN
EYzNTRDNTU0OTUzNTM0ODRDNEE1NzU2NDU0RTUzNTI0RTVBNEM0NzQzNTU1MjUyNE
M0OTMyNTY0RjMzNEE1NjRCNUE1MzU1NEQ1QTMzNUE0QjU2NEI0NTQ1MzM0MzU4NDk
1MjQxNTQ1MzNEM0QzRA==
-------------------------------
Base64
-------------------------------
4B355754434E4442495A5858555A43454A5A55464552534A4F354C55495353484C4A57564
54E53524E5A4C47435552524C4932564F334A564B5A53554D5A335A4B564B454533435849
524154533D3D3D
-------------------------------
Base16
-------------------------------
K5WTCNDBIZXXUZCEJZUFERSJO5LUISSHLJWVENSRNZLGCURRLI2VO3JVKZSUMZ3ZKVKEE3
CXIRATS===
-------------------------------
Base32
-------------------------------
Wm14aFozdDNhRFlwWDJGZmR6QnVaR1Z5Wm5VeFgyUTBlWDA9
-------------------------------
Base64
-------------------------------
ZmxhZ3t3aDR0X2FfdzBuZGVyZnUxX2Q0eX0=
-------------------------------
Base64
-------------------------------
flag{wh[redacted]y}
-------------------------------
End
```