CTF 冬奥会_is_coming 2021 蓝帽杯 WriteUp

原创

baynk 于 2021-04-30 11:14:56 发布 824 收藏 4

分类专栏： # 蓝帽杯CTF Writeup 文章标签： CTF

本文链接：https://blog.csdn.net/u014029795/article/details/116295569

蓝帽杯CTF Writeup 专栏收录该内容

2 篇文章 0 订阅

订阅专栏

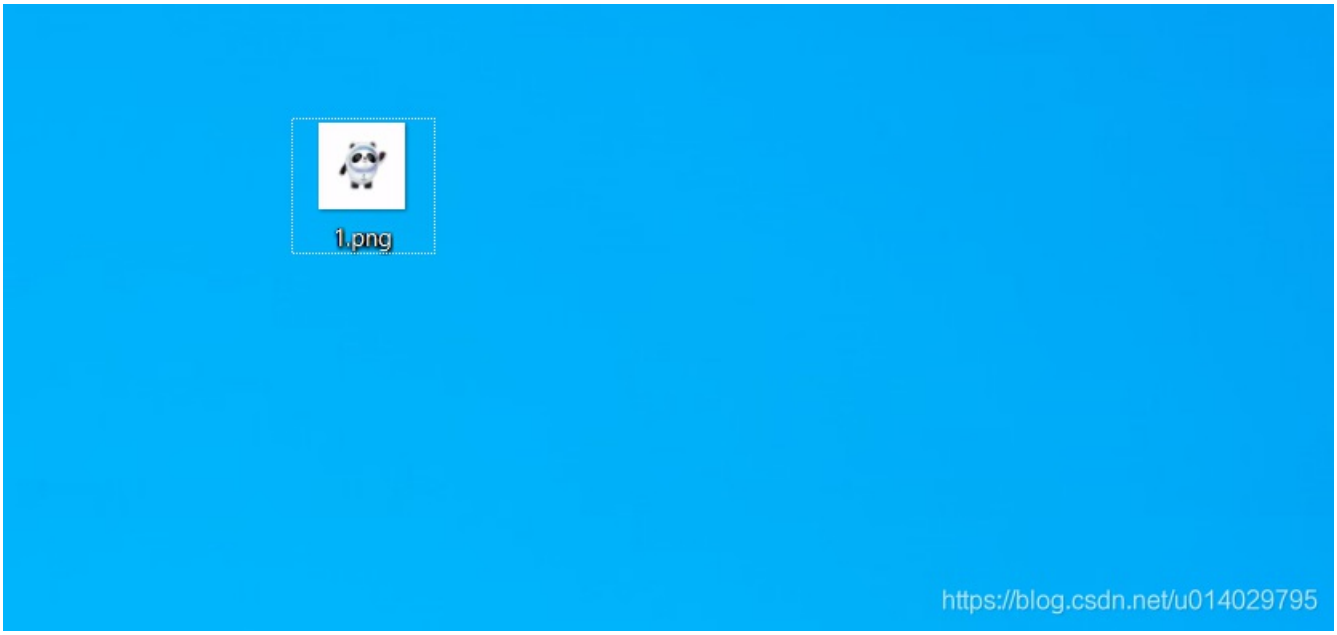
不想好好工作，就想学习，工作期间偷懒，和群友玩CTF，怎一个惨字了得。。。

一个Misc题，其实做的特别特别少，都是现学现卖，最后还是看群友的WP复现完成的。简单记录下复现过程。

下载地址放这，自行下载 ub97

自己肝的

只有一张图片，啥都没，看属性，改长宽啥的什么都没有。



丢Kali Binwalk看下，有文件，接着做了分离。

```
root@kali191a:~# binwalk 1.png
```

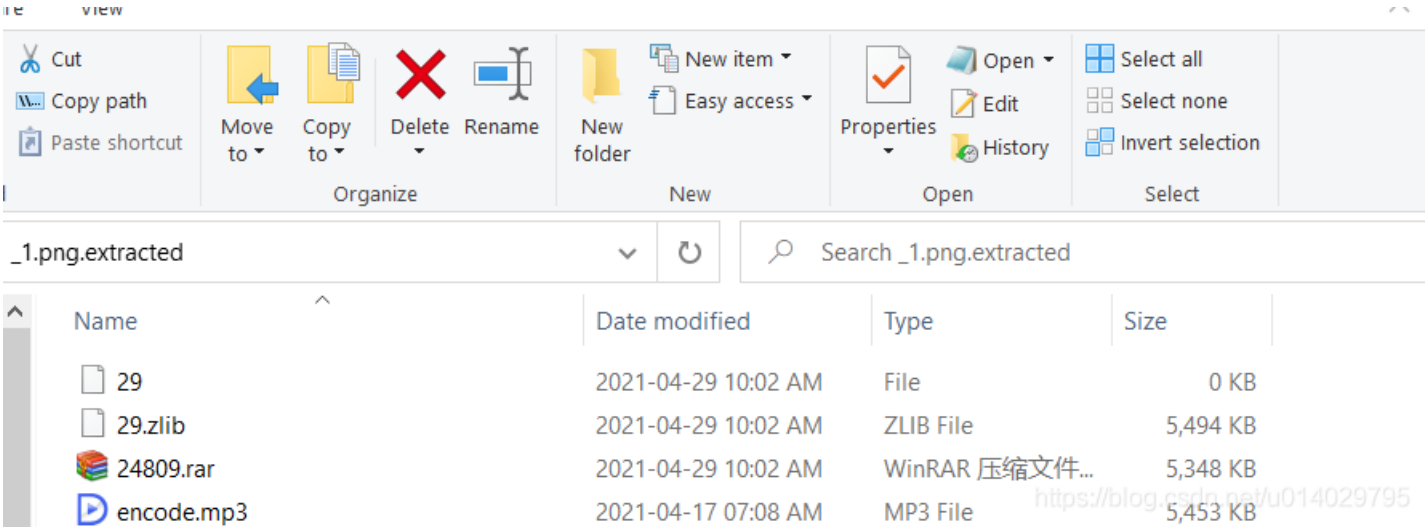
```
DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             PNG image, 657 x 657, 8-bit/color RGBA, non-interlaced
41            0x29            Zlib compressed data, default compression
149513        0x24809         RAR archive data, version 4.x, first volume type: MAIN_HEAD

root@kali191a:~# binwalk -e 1.png

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             PNG image, 657 x 657, 8-bit/color RGBA, non-interlaced
41            0x29            Zlib compressed data, default compression
149513        0x24809         RAR archive data, version 4.x, first volume type: MAIN_HEAD

root@kali191a:~# ls
1.png  _1.png.extracted  debug.log  Documents  Music  Public  Templates  user.list  w3af
```

得到了一个rar和一个zlip文件



这里通过听歌及查信息等方式都没发现啥，然后通过`audacity`看了下音乐频谱啥的也没东西，最后通过`MP3Stego`发现了是有加密的，但是需要密码。接下来就尴尬了，到处找密码。考虑了几个点，rar文件内有提示

另外zlib文件还没使用，于是折腾了半天zlib文件，啥也不是，最后没办法，试了下冬奥日期，坑爹，第一次打成了`20210204`。没错，日期错了，后来聊天才发现我打错了。。。应该是`20220204`，使用`MP3Stego`成功解密。

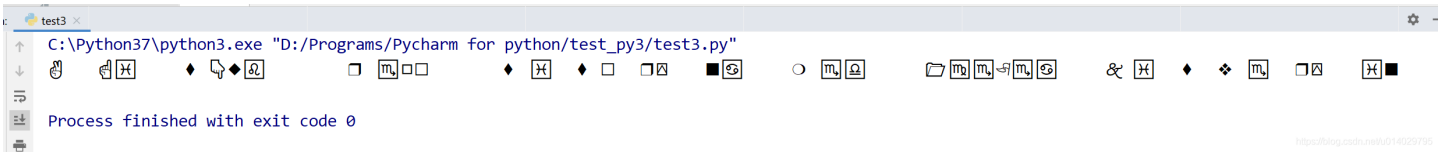
```
D:\Tools\01-Hack Penetration\10-CTF\MP3Stego_1_1_19\MP3Stego>Decode.exe -P 202202004 C:\Users\TuTuB\Desktop\_1.png.extracted\encode.mp3
MP3StegoEncoder 1.1.19
See README file for copyright info
Input file = 'C:\Users\TuTuB\Desktop\_1.png.extracted\encode.mp3'  output file = 'C:\Users\TuTuB\Desktop\_1.png.extracted\encode.mp3.pcm'
the bit stream file C:\Users\TuTuB\Desktop\_1.png.extracted\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 "C:\Users\TuTuB\Desktop\_1.png.extracted\encode.mp3" is finished
The decoded PCM output file name is "C:\Users\TuTuB\Desktop\_1.png.extracted\encode.mp3.pcm"
```

打开txt发现一堆十六进制，直接转有乱码，弄了个py转的，代码如下：

```
from urllib import parse
s = '\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\x
9f\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\xb
8\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\x9
9\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\x
a0\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\x8
e\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'
s = s.encode('unicode_escape')
#print(s)
ss = s.decode('utf-8').replace('\\x','%')
#print(s)
un = parse.unquote(ss)
print(un)
```

转完是看不懂的文字。。。



我先以为是emoji，解了半天无果，然后这里我就放弃了，实在不知道咋搞，先把这里称为`关键信息1`，后面还要用。

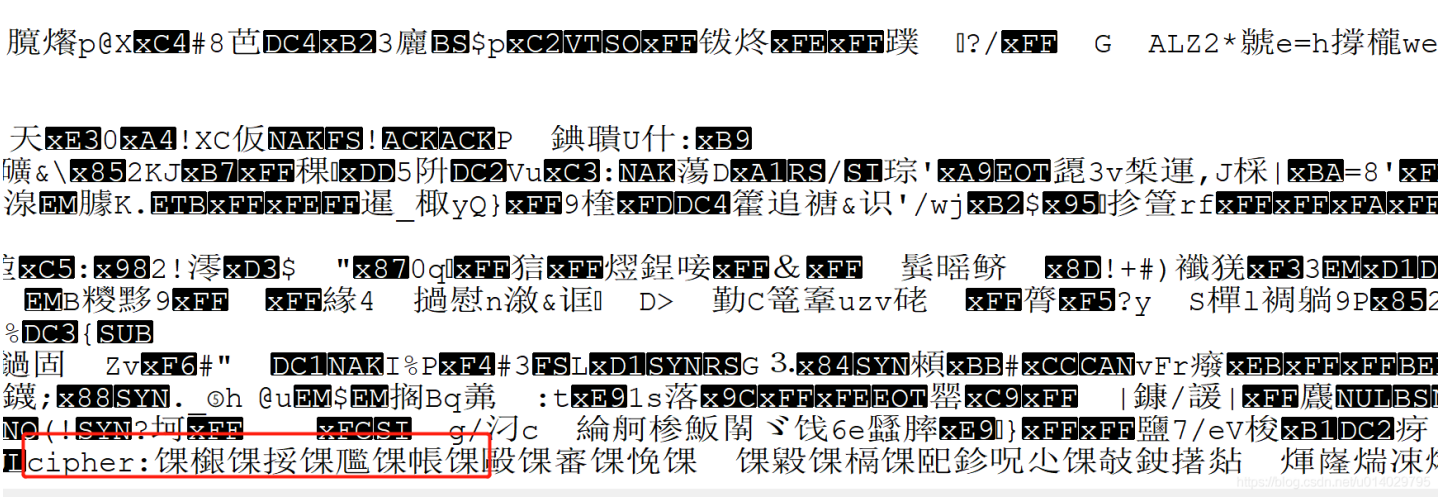
其实期间自己还发现了一些其它的信息，但是不知道要怎么利用。接下来再看群友的WP后进行复现的。

隔天复现

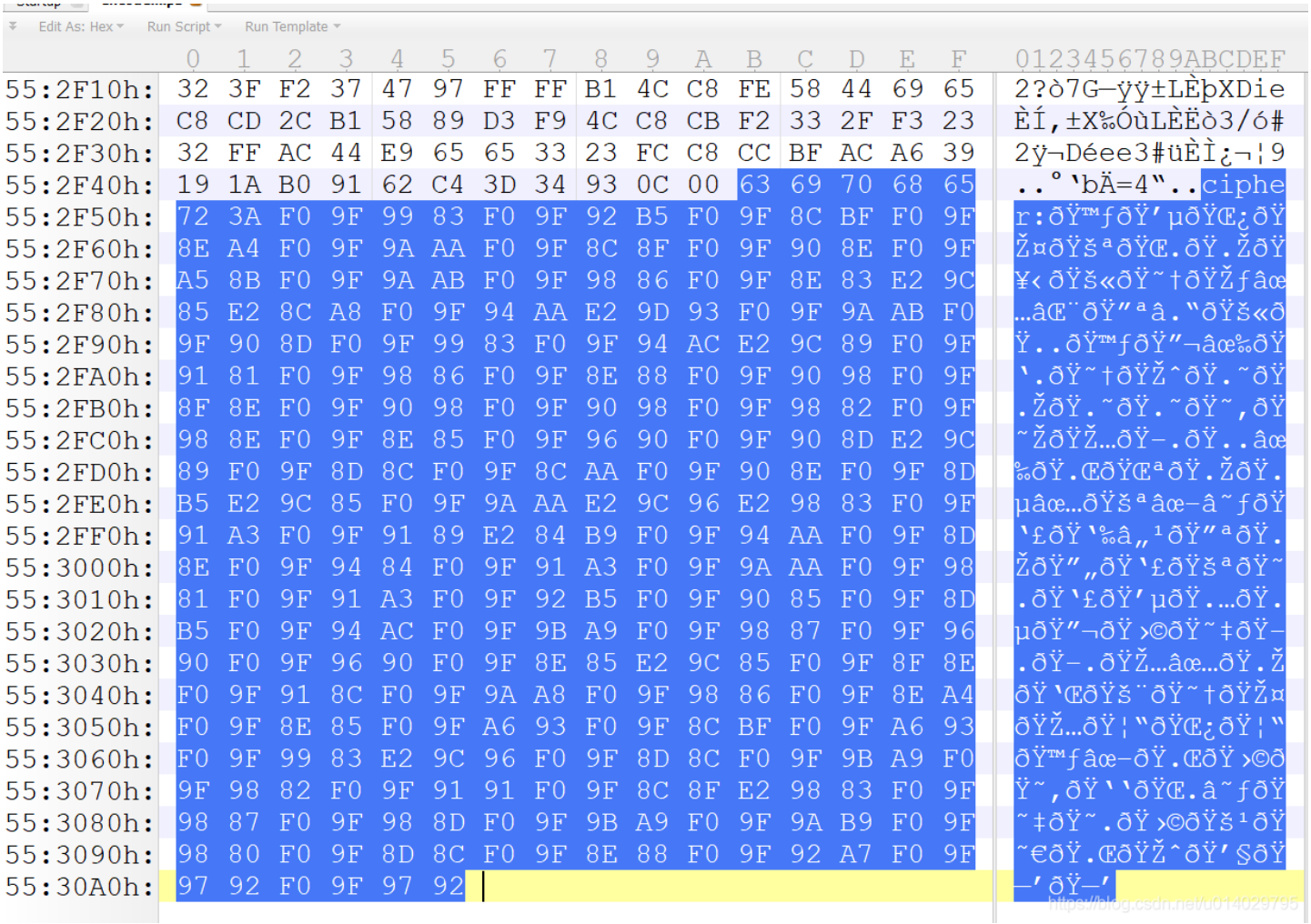
先说自己发现的一个不知道怎么利用的重要信息，期间找不到密码时，我使用了命令想直接去查解密的`password`，用了命令搜索pass。

```
root@kali191a:~/_1.png.extracted# strings * |grep pass
```

但是啥都没有，于是我用notepad++直接打开所有的文件，最终在mp3后面发现了有一段`cipher`开头的加密项



这里其实是关键，只是我弱小又无知，宝藏在面前我却不知道。。。接着用`010editor`把后面的加密部分以16进制形式导出成txt，内容如下。



接着用python把十六进制中间的空格全去掉，得到最后的字符串

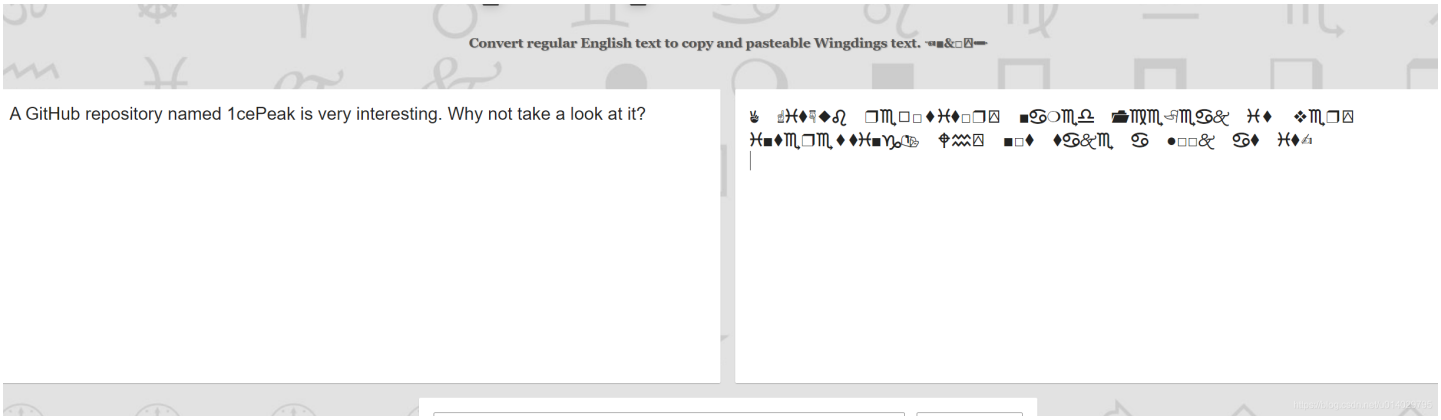
```
str = "72 3A F0 9F 99 83 F0 9F 92 B5 F0 9F 8C BF F0 9F8E A4 F0 9F 9A AA F0 9F 8C 8F F0 9F 90 8E F0 9FA5 8B F0 9F
 9A AB F0 9F 98 86 F0 9F 8E 83 E2 9C85 E2 8C A8 F0 9F 94 AA E2 9D 93 F0 9F 9A AB F09F 90 8D F0 9F 99 83 F0 9F 94
 AC E2 9C 89 F0 9F91 81 F0 9F 98 86 F0 9F 8E 88 F0 9F 90 98 F0 9F8F 8E F0 9F 90 98 F0 9F 90 98 F0 9F 98 82 F0 9F
98 8E F0 9F 8E 85 F0 9F 96 90 F0 9F 90 8D E2 9C89 F0 9F 8D 8C F0 9F 8C AA F0 9F 90 8E F0 9F 8DB5 E2 9C 85 F0 9F
9A AA E2 9C 96 E2 98 83 F0 9F91 A3 F0 9F 91 89 E2 84 B9 F0 9F 94 AA F0 9F 8D8E F0 9F 94 84 F0 9F 91 A3 F0 9F 9A
AA F0 9F 9881 F0 9F 91 A3 F0 9F 92 B5 F0 9F 90 85 F0 9F 8DB5 F0 9F 94 AC F0 9F 9B A9 F0 9F 98 87 F0 9F 9690 F0 9
F 96 90 F0 9F 8E 85 E2 9C 85 F0 9F 8F 8EF0 9F 91 8C F0 9F 9A A8 F0 9F 98 86 F0 9F 8E A4F0 9F 8E 85 F0 9F A6 93 F
0 9F 8C BF F0 9F A6 93F0 9F 99 83 E2 9C 96 F0 9F 8D 8C F0 9F 9B A9 F09F 98 82 F0 9F 91 91 F0 9F 8C 8F E2 98 83 F
0 9F98 87 F0 9F 98 8D F0 9F 9B A9 F0 9F 9A B9 F0 9F98 80 F0 9F 8D 8C F0 9F 8E 88 F0 9F 92 A7 F0 9F97 92 F0 9F 97
 92"
str1 = str.replace(" ","")
print(str1)

#723AF09F9983F09F92B5F09F8CBFF09F8EA4F09F9AAAF09F8C8FF09F908EF09FA58BF09F9AABF09F9886F09F8E83E29C85E28CA8F09F94A
AE29D93F09F9AABF09F908DF09F9983F09F94ACE29C89F09F9181F09F9886F09F8E88F09F9098F09F8F8EF09F9098F09F9098F09F9882F09
F988EF09F8E85F09F9690F09F908DE29C89F09F8D8CF09F8CAAF09F908EF09F8DB5E29C85F09F9AAAE29C96E29883F09F91A3F09F9189E28
4B9F09F94AAF09F8D8EF09F9484F09F91A3F09F9AAAF09F9881F09F91A3F09F92B5F09F9085F09F8DB5F09F94ACF09F9BA9F09F9887F09F9
690F09F9690F09F8E85E29C85F09F8F8EF09F918CF09F9AA8F09F9886F09F8EA4F09F8E85F09FA693F09F8CBFF09FA693F09F9983E29C96F
09F8D8CF09F9BA9F09F9882F09F9191F09F8C8FE29883F09F9887F09F988DF09F9BA9F09F9AB9F09F9880F09F8D8CF09F8E88F09F92A7F09
F9792F09F9792
```

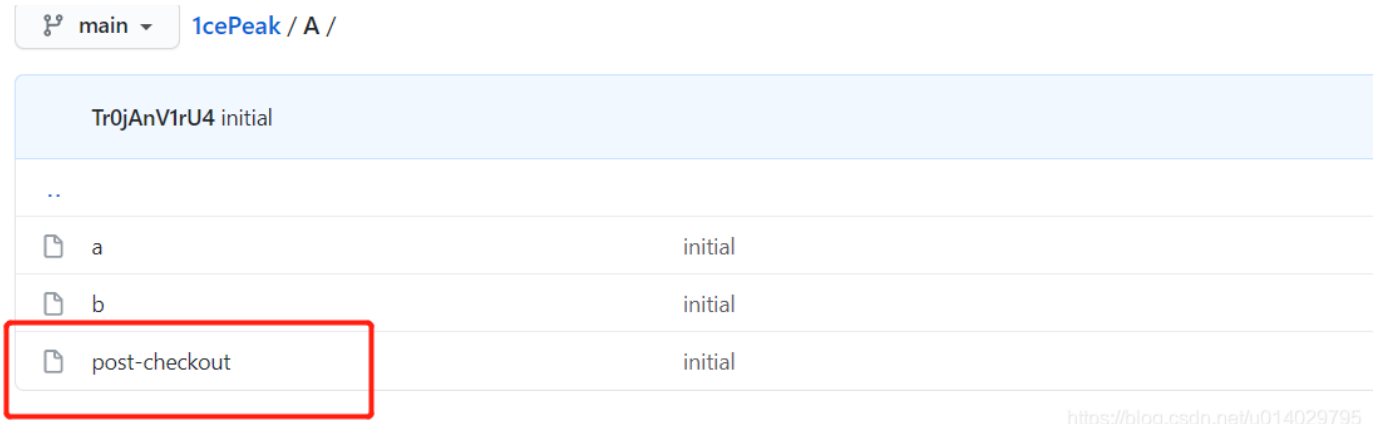
然后用hex进行解码，得到了新的emoji。

但是这个emoji也没办法直接解码(这段emoji称为关键信息2)，是啥emoji-sec，反正之前是没玩过，需要密码才能解，那密码是啥呢？其实就是之前那段看不懂的方字，也就是关键信息1，那段文字其实是是闹酒狂欢字体，真的是听过见过没用过，想不起来。。。

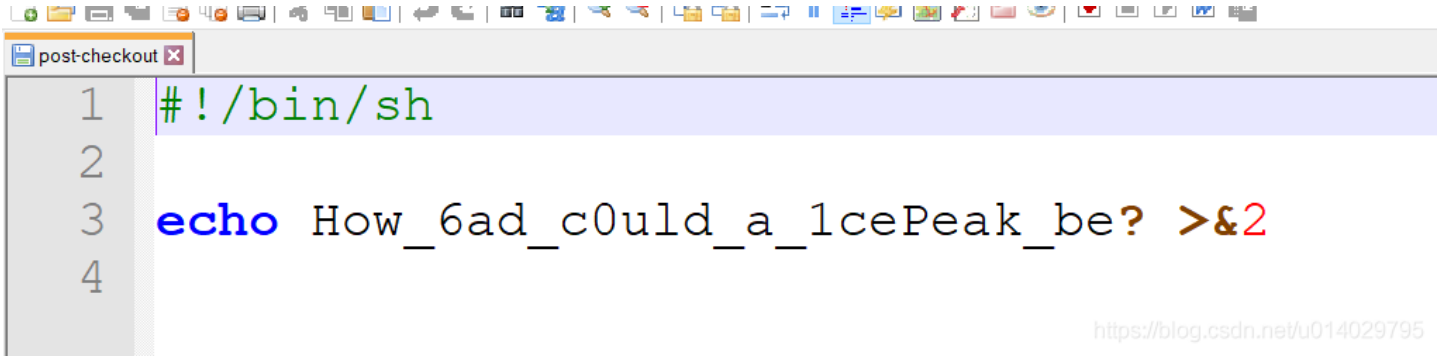
使用https://lingojam.com/WingdingsTranslator网站来解密。



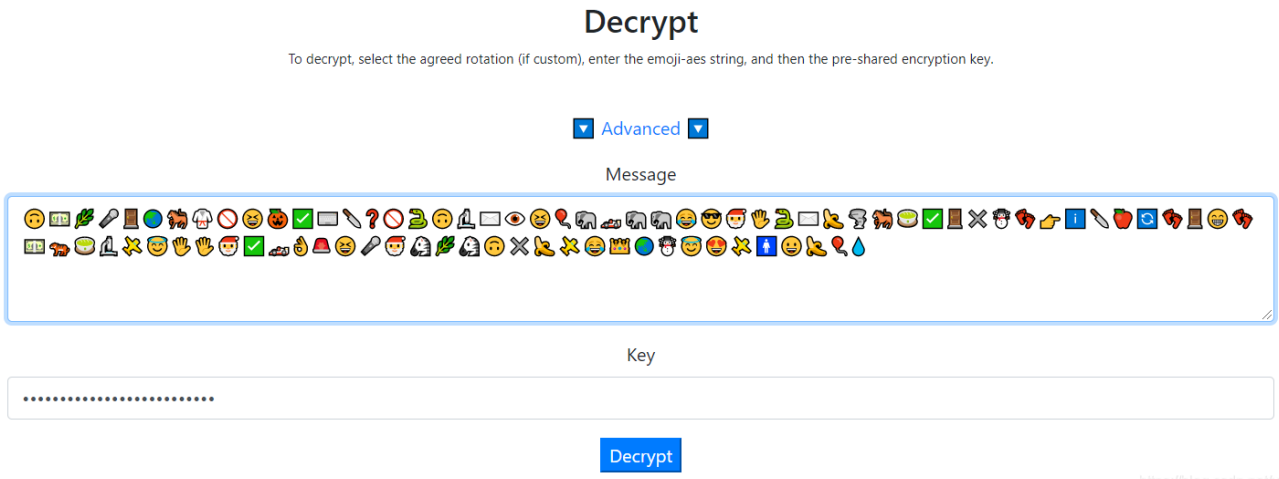
得到一段话，然后去github中搜索了1cePeak。



下载下来后，得到了一段shell，也不用运行了，就是`How_6ad_c0uld_a_1cePeak_be?`



这个信息干嘛的呢，显然这玩意就是`key`，用来解之前的那个`emoji-sec`的`关键信息2`的，在这个网站中进行解密即可`https://aghorler.github.io/emoji-aes/`



终于拿到`flag`，复现结束，还是有不少收获，学习了。。