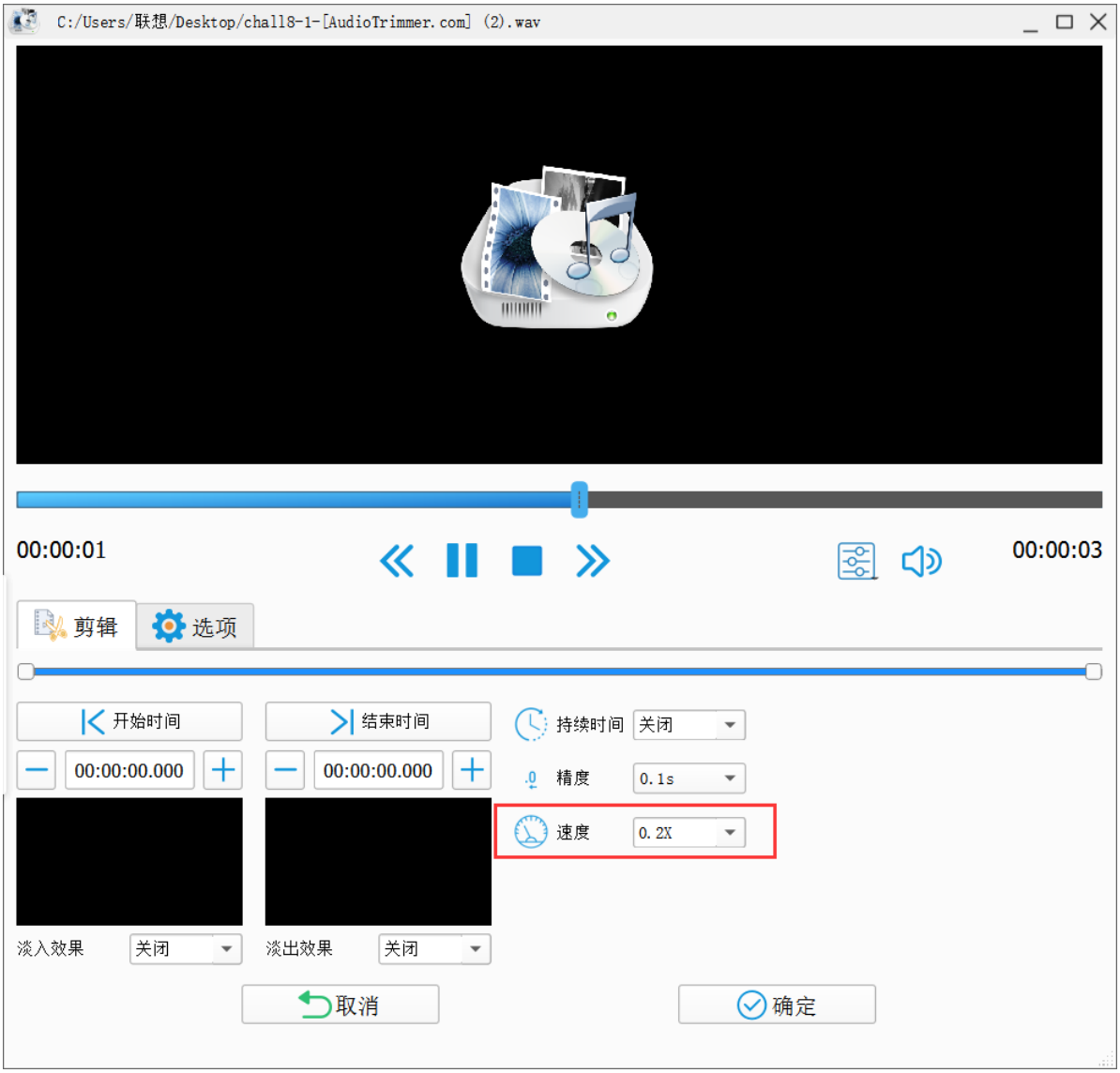
CTF1:

```
flag{this_is_different_from_hg}
```

step1: reverse it.

step 2: slow it with FormatFactory



And with the transform table below, I get the flag:

Symbol	Code Word
A	Alfa/Alpha
B	Bravo
C	Charlie
D	Delta
E	Echo
F	Foxtrot
G	Golf
H	Hotel
I	India
J	Juliett
K	Kilo
L	Lima
M	Mike
N	November
O	Oscar
P	Papa
Q	Quebec
R	Romeo
S	Sierra
T	Tango
U	Uniform
V	Victor
W	Whiskey
X	X-ray
Y	Yankee
Z	Zulu

CTF2:

```
pbctf{mechanical_keyboards_are_loud}
```

Step 1:Read the file and denote each segment.

```
from scipy.io import wavfile
samplerate, data = wavfile.read('./output.wav')

i = 0
countsilent = 0
samplefound = 0

avg = []
start = 0
end = 0
avg.append([0])
if __name__ == '__main__':
    f = open("avg", "a")
    for sample in data:
        i+=1
        if(sample[1]<100):
            countsilent += 1
        if countsilent > 10000 and sample[1]>100:
            countsilent = 0
            #print(str(i)+": sample found")
            start = i
            samplefound = 1
        if(countsilent > 8000 and samplefound==1):
            samplefound = 0
            #print(str(i)+": sample ended")
            end = i
            avg[len(avg)-1]=avg[len(avg)-1]/(end-start)
            print("avg: "+str(avg[len(avg)-1]))
            f.write("avg: "+str(avg[len(avg)-1]))
            f.write('\n')
            avg.append([0])
        if (samplefound == 1):
            avg[len(avg)-1] += sample[1]
    f.close()
```

Step 2: set a character for each segement, same segement will be sent with same character.

```
alphabet = "abcdefghijklmnopqr_stuvwxyz"
avg = {}

i=0

res=""
if __name__ == '__main__':
    with open("avg") as file:
        for line in file:
            value = line.rstrip()[6:-1]
            print(value)
            if str(value) not in avg:
                avg[str(value)]=alphabet[i]
                i+=1
            res+=avg[str(value)]
    f = open("pass", "a")
    f.write(res)
    f.close()
```

I get :

```
abcdefgdbfedhcijkdlkdfdhmnibadklacdabcdojkdpnkfqqgdrccqcedabmljibdabcdmfnkdplmda
bcdpnmoadan_cdnkdfdsccqdfkedabcdpqfidnodrhtapdlrckdhmftcd_ctbfkntfqdqcghlfmeodfm
cdqljedtqlocdhmftcdfkedabcdhnmeodscmcdonkinkidnkdnaodsfm_abdabcmcdsfodkldsfgdald
fkantnrfacdsbfadsfodfhljadaldbfrrck
```

Step 3: treat it as a coding message and decode it.

Substitution Solver - www.guballa.de

I get

```
thesdayshadsbegunsonsasbrightsnotesthesmunsfinallyspeeledsthroughsthesrainsforst
hesfirmtsti_esinsasweelsandsthesflagsimspbctfsopensbraces_echanicalsleyboardmsar
esloudsclomesbracesandsthesbirdmsweresmingingsinsitmswar_thsthereswamsnoswaystos
anticipateswhatswamsaboutstoshappen
```

Then I do a small modification:

```
if __name__ == '__main__':
    with open("decode") as file:
        for line in file:
            line=line.replace('s', ' ')
            line=line.replace("m","s")
            line=line.replace("_","m")
            print(line)
    file.close()
```

I get:

```
the day had begun on a bright note the sun finally peeled through the rain for
the first time in a weel and the flag is pbctf open brace mechanical leyboards
are loud close brace and the birds were singing in its warmth there was no way to
anticipate what was about to happen
```

the flag is pbctf open brace mechanical leyboards are loud close brace

reference :手写算法-声音密码学之识别监听键盘输入 - So who are you (kms.app)

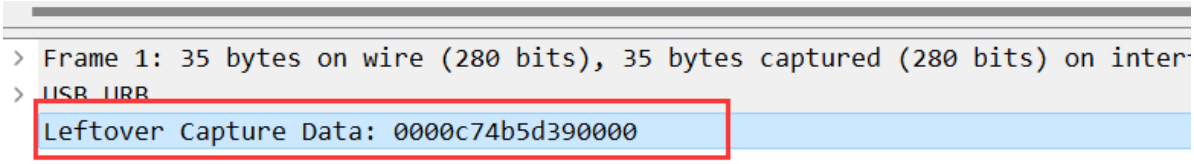
CTF3:

```
CTF{tHE_cAT_is_the_CULPRiT}
```

By searching , I get that:

The USB protocol mouse Data part is in the **Leftover Capture Data domain**, and the Data length is **four bytes**.

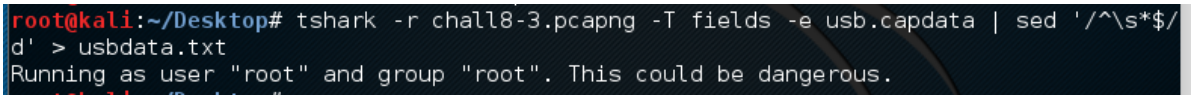
其中第一个字节代表按键，当取0x00时，代表没有按键、为0x01时，代表按左键，为0x02时，代表当前按键为右键。
第二个字节可以看成是一个signed byte类型，其最高位为符号位，当这个值为正时，代表鼠标水平右移多少像素，为负时，代表水平左移多少像素。
第三个字节与第二字节类似，代表垂直上下移动的偏移。
————————————
参考链接：https://blog.csdn.net/qq_43625917/article/details/107723635



Step 1:

use tshark in kali to extract data about the movement of mouse.

```
tshark -r usb2.pcap -T fields -e usb.capdata | sed '/^\s*$/d' > usbdata.txt
```



I get:

```
00:01:fe:00
00:01:ff:00
00:02:00:00
00:03:00:00
00:01:00:00
00:02:00:00
00:04:ff:00
00:01:ff:00
00:03:ff:00
00:03:fd:00
00:02:ff:00
00:01:ff:00
00:04:fd:00
00:01:fd:00
00:00:fd:00
00:ff:fc:00
00:fe:fd:00
00:fe:fc:00
00:fe:fd:00
00:ff:fc:00
00:ff:ff:00
00:00:ff:00
00:fe:fc:00
00:ff:fb:00
00:ff:fb:00
00:fc:fa:00
00:fe:fa:00
00:fd:f9:00
00:ff:fc:00
00:ff:fd:00
00:fd:fb:00
00:ff:fc:00
00:ff:fd:00
00:00:fd:00
00:fe:fc:00
```

Step 2: write a script to probe the hiden infomation.

```
nums = []
keys = open('out.txt','r')
f = open('xy.txt','w')
posx = 0
posy = 0
for line in keys:
    if len(line) != 12 :
        continue
    x = int(line[3:5],16)
    y = int(line[6:8],16)
    if x > 127 :
        x -= 256
    if y > 127 :
        y -= 256
    posx += x
    posy += y
    btn_flag = int(line[0:2],16)
    if btn_flag == 1 :
        print(str(posx))
    print(str(posy))
```

```
    f.write(str(posx))
        f.write(' ')
        f.write(str(posy))
        f.write('\n')
f.close()
```

if btn_flag == 1 : means the movement of left button.

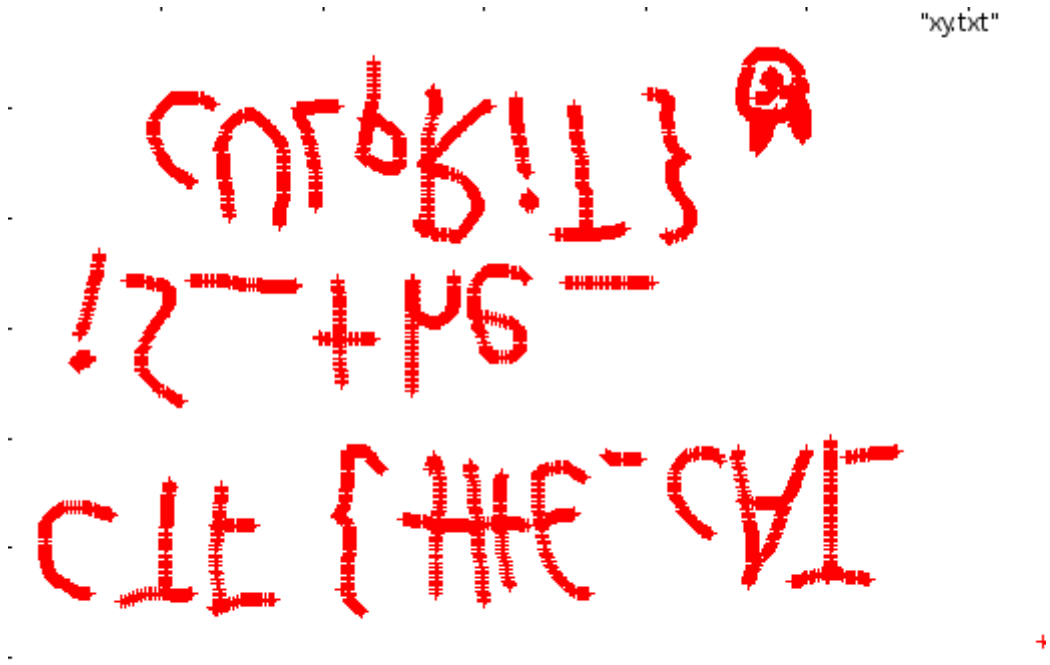
I get:

```
-899 -241
-901 -241
-902 -241
-904 -240
-906 -240
-907 -240
-909 -239
-910 -238
-911 -238
-912 -238
-913 -237
-914 -236
-915 -235
-916 -235
-917 -235
-919 -234
```

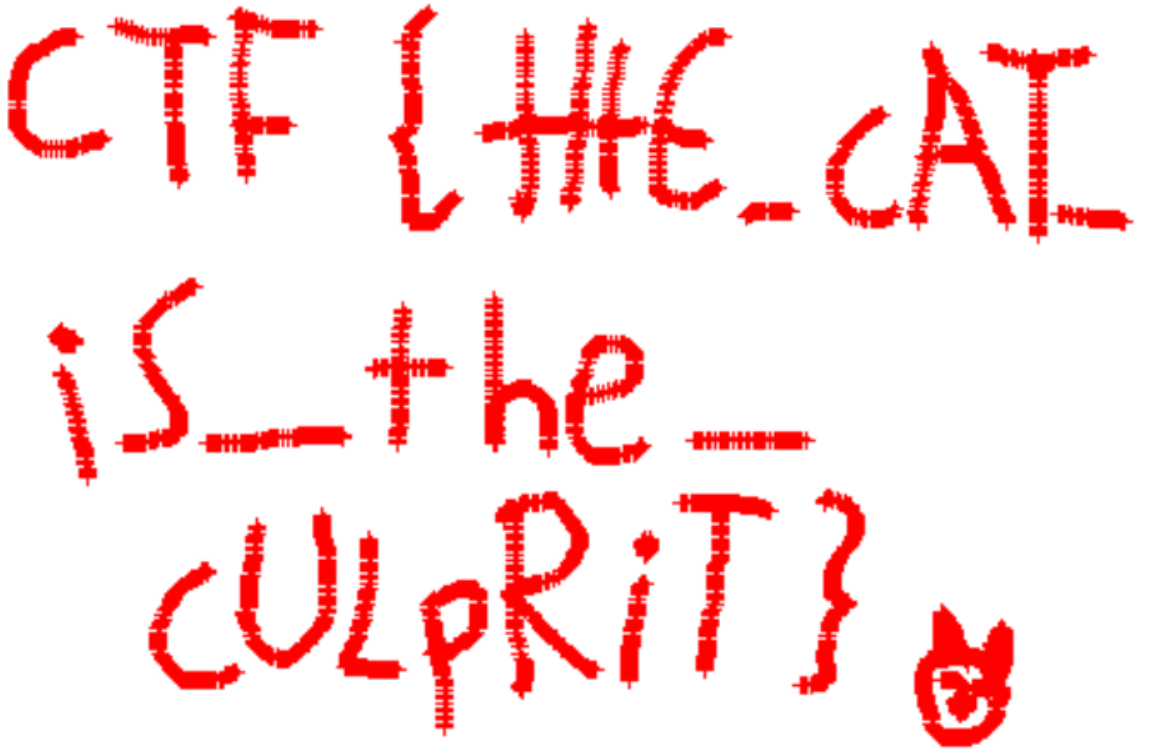
Step 3: plot it.

```
gnuplot
gnuplot>plot "xy.txt"
```

I get:



Finally, I overturn it.



Finish it ~~~~