首届“网刃杯”网络安全大赛部分WP

原创

长球的鱼 于 2021-11-19 16:24:50 发布 1968 收藏

文章标签： 网络安全 wireshark python kali 安全

这是之前打比赛写的WP，供大家参考

藏在S7里的秘密

1.修复流量包

下载流量包打开后发现提示错报

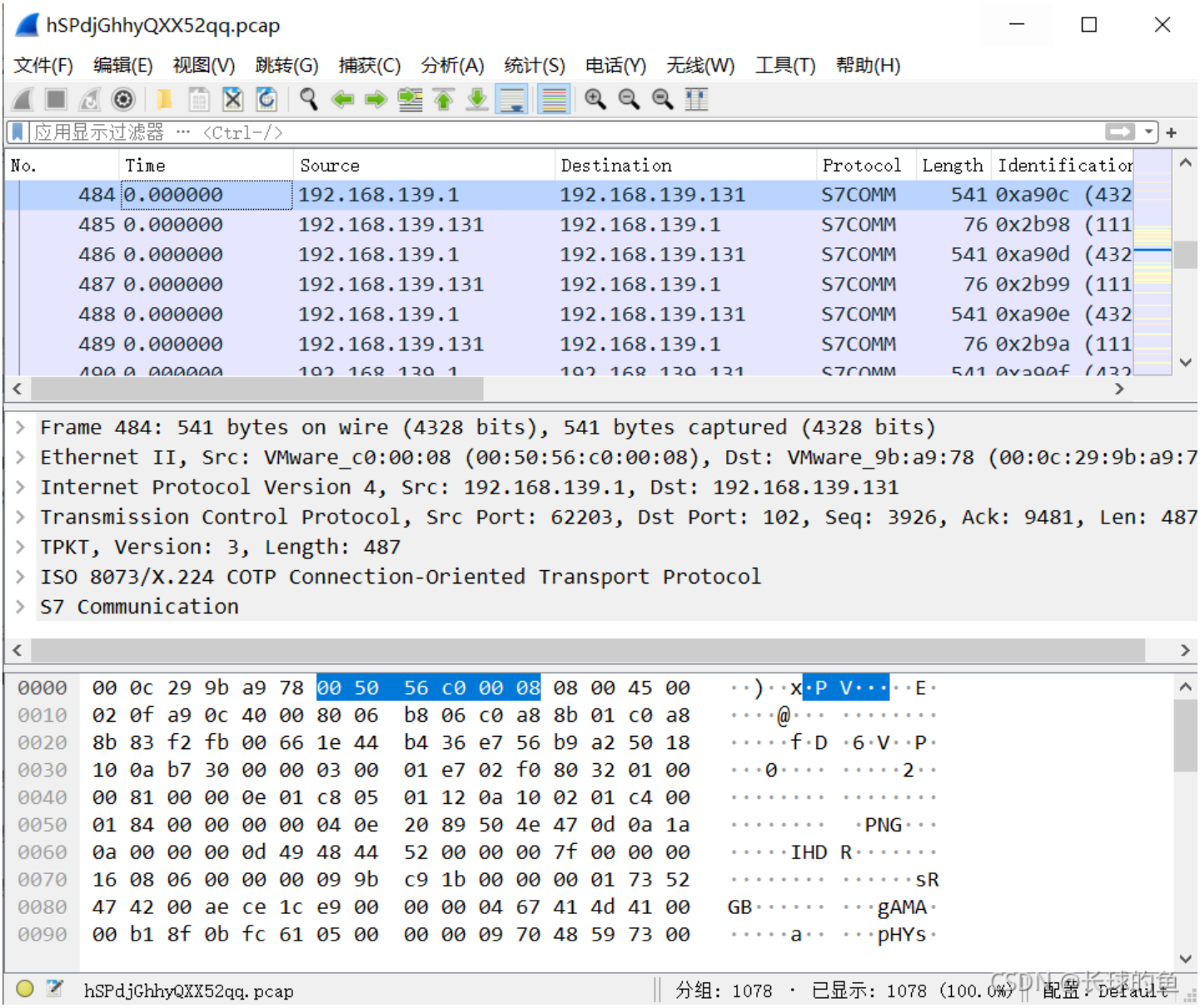
The file"S7.pcap" isn't a capture file in a format Wireshark understands.

所以我们找修复流量包的工具

http://f00l.de/hacking/pcapfix.php

修复完毕成功打开，审查流量包

藏在S7里的秘密



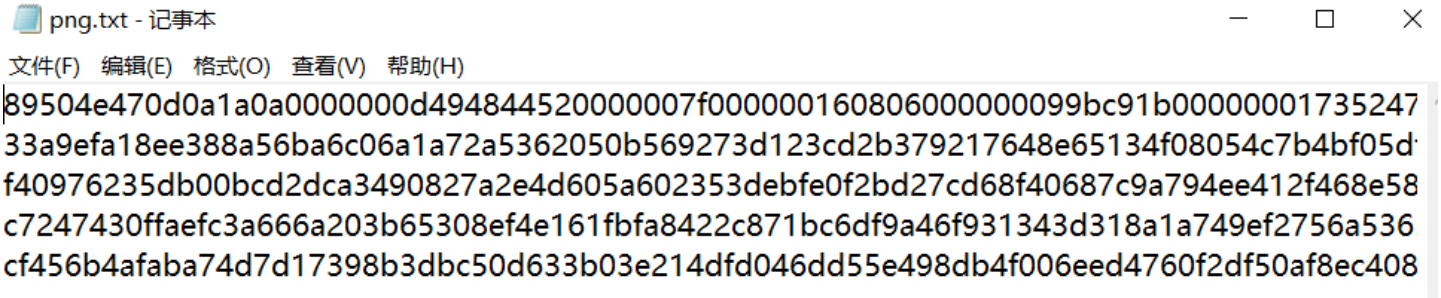
2.寻找有用信息

发现里面有PNG图片

尝试用tshark抓取出来

```
tshark -r C:\Users\XINO\Desktop\hSPdjGhhyQXX52qq.pcap -T fields -e s7comm.resp.data -Y "s7comm.param.func == 0x0
5 and ip.src==192.168.139.1" > png.txt
```

打开png.txt文件

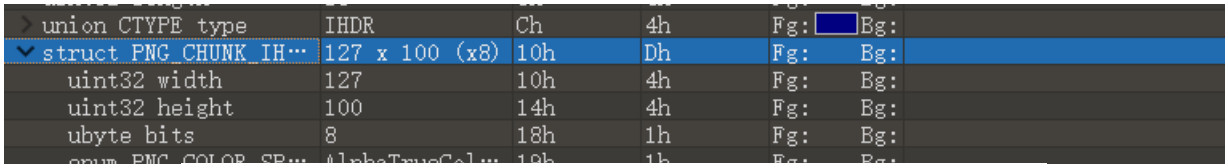


复制后将其放进010保存为PNG

不是 我flag呢?

3.修改图片高度

图片没有显示完整，我们接着用010修改其高度



`flag{FSfeQefjg}`

老练的黑客

分析问题查看流量并分析，猜测问题出自modbus协议中

1.过滤出modbus协议流量

No.	Time	Source	Destination	Protocol	Length	Identification	IRP information	Info
249	0.000000	192.168.3.130	192.168.3.164	Modbus…	66	0x2813 (102…		Query: Trans: 11008; Unit: 1, Func: 3: Read Holding Registers
250	0.000000	192.168.3.164	192.168.3.130	Modbus…	73	0x284c (103…		Response: Trans: 11008; Unit: 1, Func: 3: Read Holding Registers
253	0.000000	192.168.3.130	192.168.3.164	Modbus…	66	0x2815 (102…		Query: Trans: 11264; Unit: 1, Func: 3: Read Holding Registers
254	0.000000	192.168.3.164	192.168.3.130	Modbus…	73	0x284d (103…		Response: Trans: 11264; Unit: 1, Func: 3: Read Holding Registers
256	0.000000	192.168.3.130	192.168.3.164	Modbus…	66	0x2817 (102…		Query: Trans: 11520; Unit: 1, Func: 3: Read Holding Registers
257	0.000000	192.168.3.164	192.168.3.130	Modbus…	73	0x284e (103…		Response: Trans: 11520; Unit: 1, Func: 3: Read Holding Registers
259	0.000000	192.168.3.130	192.168.3.164	Modbus…	66	0x2819 (102…		Query: Trans: 11776; Unit: 1, Func: 3: Read Holding Registers
260	0.000000	192.168.3.164	192.168.3.130	Modbus…	73	0x284f (103…		Response: Trans: 11776; Unit: 1, Func: 3: Read Holding Registers
263	0.000000	192.168.3.130	192.168.3.164	Modbus…	66	0x281c (102…		Query: Trans: 12032; Unit: 1, Func: 3: Read Holding Registers

根据

2021CTF工业信息安全技能大赛-损坏的风机中找错误值的方法，我们对流量包进行审查

2.寻找被修改的错误值

我们审查流量包后，在1198行发现错误的data数据

```
> Frame 1198: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
> Ethernet II, Src: VMware_cb:84:a0 (00:0c:29:cb:84:a0), Dst: VMware_6d:a1:06 (00:0c:29:6d:a1:06)
> Internet Protocol Version 4, Src: 192.168.3.130, Dst: 192.168.3.164
> Transmission Control Protocol, Src Port: 12414, Dst Port: 502, Seq: 1321, Ack: 2042, Len: 12
> Modbus/TCP
v Modbus
    .000 0110 = Function Code: Write Single Register (6)
    Reference Number: 0
    Data: 22b8
```


因为答案是被修改的错误值和之前的转速，我们找修改之前的值

3.寻找被修改前的值

经过不断寻找发现在1209行有个4500的转数数据

```
> Internet Protocol Version 4, Src: 192.168.3.164, Dst: 192.168.3.130
> Transmission Control Protocol, Src Port: 502, Dst Port: 12414, Seq: 2054, Ack: 1345, L
> Modbus/TCP
v Modbus
    .000 0011 = Function Code: Read Holding Registers (3)
    [Request Frame: 1208]
    [Time from request: 0.000000000 seconds]
    Byte Count: 10
  v Register 0 (UINT16): 4500
      Register Number: 0
      Register Value (UINT16): 4500
  v Register 1 (UINT16): 262
```

```
0000  00 0c 29 cb 84 a0 00 0c  29 6d a1 06 08 00 45 00   ··)····· )m····E·
0010  00 3b 28 f0 40 00 80 06  49 56 c0 a8 03 a4 c0 a8   ·;(·@··· IV······
0020  03 82 01 f6 30 7e b9 7d  be 33 02 07 69 77 50 18   ····0~·} ·3··iwP·
0030  00 fb 40 80 00 00 76 00  00 00 00 0d 01 03 0a 11   ··@···v· ········
0040  94 01 06 00 3e 00 2e 00  49                       ····>·.· I
```


我们将其hex值拼接即为flag

```
flag{22b81194}
```

baby-usb

下载下来是两个文件，一个流量包，一个需要解密的文档，推测需要在流量包里找到密码解开

我们审查一下流量包

1.找有用信息

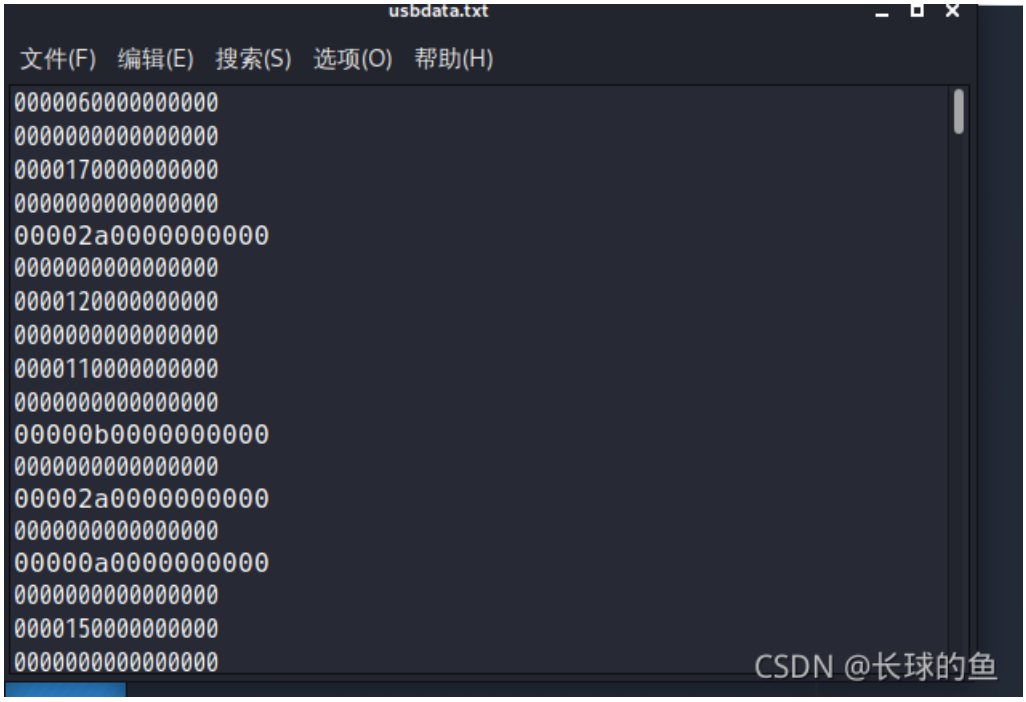
审查下来发现是键盘流量，键盘流量的数据长度为8个字节，其中有关击键的信息在第三字节（十六进制表示）

于是我们用tshark命令提取对应的Leftover Capture Data域中数据

```
tshark -r /home/kali/桌面/key.pcapng -T fields -e usb.capdata | sed '/^\s*$/d' > usbdata.txt
```

分出一个usbdata.txt

我们打开后是一串



2.脚本处理

网上找个脚本，为其加上冒号

```
f=open('usbdata.txt','r')
fi=open('out.txt','w')
while 1:
    a=f.readline().strip()
    if a:
        if len(a)==16: # 鼠标流量的话len改为8
            out=''
            for i in range(0,len(a),2):
                if i+2 != len(a):
                    out+=a[i]+a[i+1]+":"
                else:
                    out+=a[i]+a[i+1]
            fi.write(out)
            fi.write('\n')
    else:
        break

fi.close()
```

得到out.txt文件

3.提取出键盘流量后用脚本还原数据对应的信息

网上找的脚本

```
normalKeys = {
    "04":"a", "05":"b", "06":"c", "07":"d", "08":"e",
    "09":"f", "0a":"g", "0b":"h", "0c":"i", "0d":"j",
     "0e":"k", "0f":"1", "10":"m", "11":"n", "12":"o",
      "13":"p", "14":"q", "15":"r", "16":"s", "17":"t",
       "18":"u", "19":"v", "1a":"w", "1b":"x", "1c":"y",
        "1d":"z","1e":"1", "1f":"2", "20":"3", "21":"4",
         "22":"5", "23":"6","24":"7","25":"8","26":"9",
         "27":"0","28":"<RET>","29":"<ESC>","2a":"<DEL>", "2b":"\t",
```

```
         "2c":"<SPACE>","2d":"-","2e":"=","2f":"[","30":"]","31":"\\",
         "32":"<NON>","33":";","34":"'","35":"<GA>","36":",","37":".",
         "38":"/","39":"<CAP>","3a":"<F1>","3b":"<F2>", "3c":"<F3>","3d":"<F4>",
         "3e":"<F5>","3f":"<F6>","40":"<F7>","41":"<F8>","42":"<F9>","43":"<F10>",
         "44":"<F11>","45":"<F12>"}
shiftKeys = {
    "04":"A", "05":"B", "06":"C", "07":"D", "08":"E",
     "09":"F", "0a":"G", "0b":"H", "0c":"I", "0d":"J",
      "0e":"K", "0f":"L", "10":"M", "11":"N", "12":"O",
       "13":"P", "14":"Q", "15":"R", "16":"S", "17":"T",
        "18":"U", "19":"V", "1a":"W", "1b":"X", "1c":"Y",
         "1d":"Z","1e":"!", "1f":"@", "20":"#", "21":"$",
          "22":"%", "23":"^","24":"&","25":"*","26":"(","27":")",
          "28":"<RET>","29":"<ESC>","2a":"<DEL>", "2b":"\t","2c":"<SPACE>",
          "2d":"_","2e":"+","2f":"{","30":"}","31":"|","32":"<NON>","33":"\"",
          "34":":","35":"<GA>","36":"<","37":">","38":"?","39":"<CAP>","3a":"<F1>",
          "3b":"<F2>", "3c":"<F3>","3d":"<F4>","3e":"<F5>","3f":"<F6>","40":"<F7>",
          "41":"<F8>","42":"<F9>","43":"<F10>","44":"<F11>","45":"<F12>"}
output = []
keys = open('out.txt')
for line in keys:
    try:
        if line[0]!='0' or (line[1]!='0' and line[1]!='2') or line[3]!='0' or line[4]!='0' or line[9]!='0' or li
ne[10]!='0' or line[12]!='0' or line[13]!='0' or line[15]!='0' or line[16]!='0' or line[18]!='0' or line[19]!='0
' or line[21]!='0' or line[22]!='0' or line[6:8]=="00":
            continue
        if line[6:8] in normalKeys.keys():
            output += [[normalKeys[line[6:8]]],[shiftKeys[line[6:8]]]][line[1]=='2']
        else:
            output += ['[unknown]']
    except:
        pass

keys.close()

flag=0
print("".join(output))
for i in range(len(output)):
    try:
        a=output.index('<DEL>')
        del output[a]
        del output[a-1]
    except:
        pass

for i in range(len(output)):
    try:
        if output[i]=="<CAP>":
            flag+=1
            output.pop(i)
            if flag==2:
                flag=0
        if flag!=0:
            output[i]=output[i].upper()
    except:
        pass

print ('output :' + "".join(output))
```

稍微修改一下即可运行

```
> ct<DEL>onh<DEL>gratue<DEL>latioke<DEL><DEL>nsonfiny<DEL>dingmebutiwii<DEL>llns<DEL>ottellyouwheretqa<DEL><DEL>
hepz<DEL>asswordws<DEL><DEL>ox<DEL>fwe<DEL>od<DEL>rddoc<DEL>cumentisgoarfv<DEL><DEL><DEL>ndfinditagain
> output
> :congratulationsonfindingmebutiwillnottellyouwherethepasswordofworddocumentisgoandfinditagain
```

顺着输入一遍就会发现被删除的即为key

```
The key is qazwsxedcrfv
```

我们就可以打开word

```
flag{685b42b0-da3d-47f4-a76c-0f3d07ea962a}
```

mspaint

下载压缩包解压后发现是个.raw后缀的文件，于是我们用取证工具查看下有啥重要信息

1.用取证工具找有用信息

先看一下系统

```
volatility -f /root/桌面/mspaint.raw imageinfo
```

```
root@kali:~# volatility -f /root/桌面/mspaint.raw imageinfo
Volatility Foundation Volatility Framework 2.6
INFO    : volatility.debug    : Determining profile based on KDBG search...
          Suggested Profile(s) : Win7SP1x64, Win7SP0x64, Win2008R2SP0x64, Win2008R2SP1x64_24000, Win2008R2SP1x6
4_23418, Win2008R2SP1x64, Win7SP1x64_24000, Win7SP1x64_23418
                     AS Layer1 : WindowsAMD64PagedMemory (Kernel AS)
                     AS Layer2 : FileAddressSpace (/root/桌面/mspaint.raw)
                      PAE type : No PAE
                           DTB : 0x187000L
                          KDBG : 0xf8000403c0a0L
          Number of Processors : 1
     Image Type (Service Pack) : 1
                KPCR for CPU 0 : 0xfffff8000403dd00L
             KUSER_SHARED_DATA : 0xfffff78000000000L
           Image date and time : 2021-09-08 09:47:28 UTC+0000
     Image local date and time : 2021-09-08 17:47:28 +0800
root@kali:~#
```


找到版本后我们接着查看进程，看看是否有提示点

```
volatility -f /root/桌面/mspaint.raw --profile=Win7SP1x64 pslist
```

之后经过提示我们查看一下浏览器历史记录

```
volatility -f /root/桌面/mspaint.raw --profile=Win7SP1x64 iehistory
```

发现有个名字为key的图片，我们接着把它dump下来

```
Location: :2021090820210909: qiyue@file:///C:/Program%20Files/Common%20Files/key.png
Last modified: 2021-09-08 17:47:01 UTC+0000
Last accessed: 2021-09-08 09:47:01 UTC+0000
File Offset: 0x100, Data Offset: 0x0, Data Length: 0x0
**************************************************
```

```
volatility -f mspaint.raw --profile=Win7SP1x64 filescan | grep key.png
```

只有四个字符hack

猜测是某个解密密码，因为之前看浏览记录是有个百度网盘链接，于是我们试试这个

成功发现一个压缩包

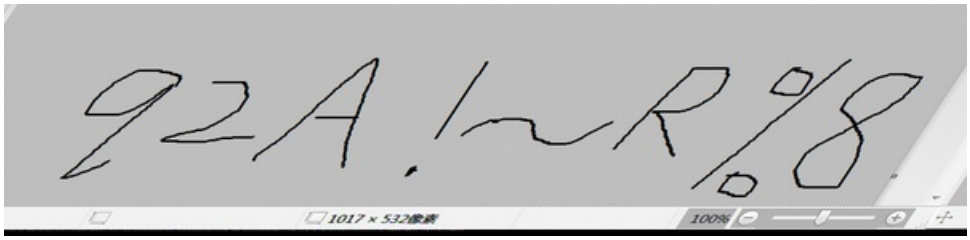
下载后是个flag.exe加密文件还是需要密码

返回虚拟机继续找有用信息

dump一个名为dump.exe的文件

```
volatility -f /root/桌面/mspaint.raw --profile=Win7SP1x64 memdump -p 1064 -D ./
```

之后发现可以用图像处理工具GIMP打开
之后反复测试找到信息



> q2A!~R%8

即为解压密码
我们打开flag.exe
2.逆写脚本
经过逆向得到

```
key = 'xxxxxxxxxxxxxx'
flag = 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx'
data = ''
for i in range(0, len(flag)):
    data += hex(ord(flag[i]) ^ ord(key[(i % 15)]))[2:].zfill(2)
else:
    print(data.upper())
    data = '120450142403436844505O6E5E1E1C165D045E6B52113C5951006F091E4F4C0C54426A52466A165B0122'
```

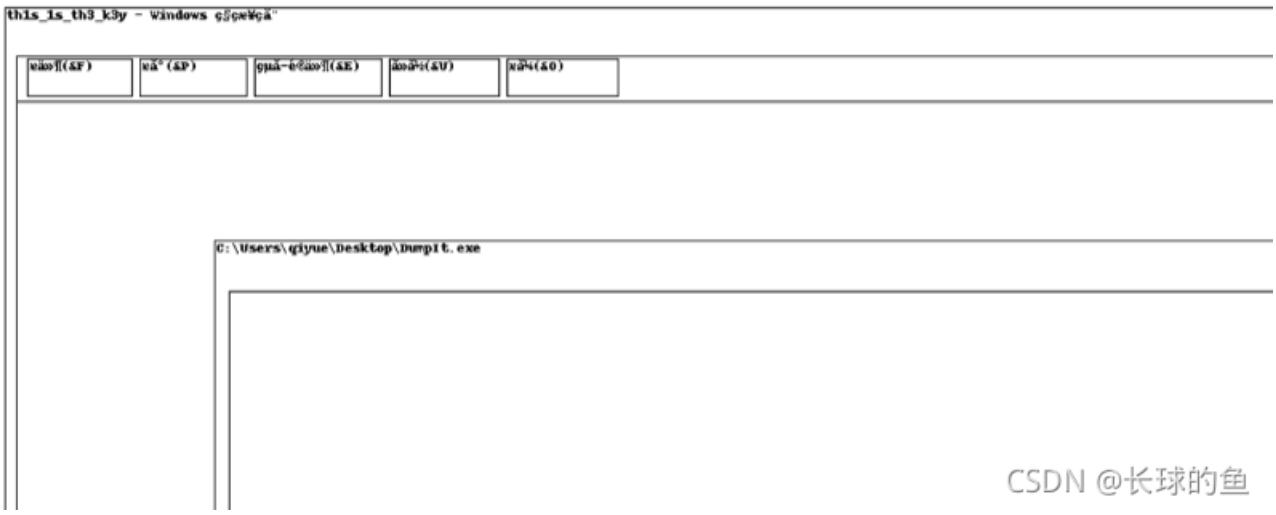
我们缺一个key于是回去看

```
volatility -f /root/桌面/mspaint.raw --profile=Win7SP1x64 cmdscan
```

```
root@kali:~# volatility -f /root/桌面/mspaint.raw --profile=Win7SP1x64 cmdscan
Volatility Foundation Volatility Framework 2.6
**************************************************
CommandProcess: conhost.exe Pid: 2684
CommandHistory: 0x1bfde0 Application: cmd.exe Flags: Allocated, Reset
CommandCount: 3 LastAdded: 2 LastDisplayed: 2
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0x5c
Cmd #0 @ 0x1bf430: I am a painter
Cmd #1 @ 0x159f70: My favorite thing is painting!!!
Cmd #2 @ 0x192c00: I usually have a good habit of saving screenshots
Cmd #37 @ 0x1b61c0:
Cmd #38 @ 0x140158:
**************************************************
CommandProcess: conhost.exe Pid: 1856
CommandHistory: 0x2afde0 Application: DumpIt.exe Flags: Allocated
CommandCount: 0 LastAdded: -1 LastDisplayed: -1
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0x5c
root@kali:~#
```


根据提示看截屏

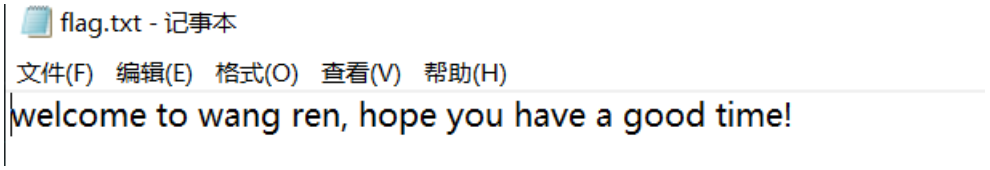
```
volatility -f /root/桌面/mspaint.raw --profile=Win7SP1x64 screenshot -D ./
```



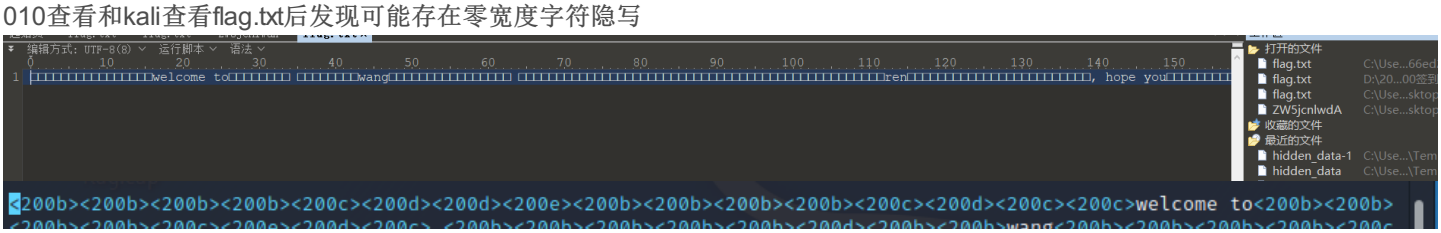

```
key = ['t','h','1','s','_','1','s','_','t','h','3','_','k','3','y']
flag = ''
for i in range(0,42):
    flag += chr(data[i] ^ ord(key[(i % 15)]))
print(flag)
//flag{20708c15-eb55-4cbc-930b-68de15c55b32}
```

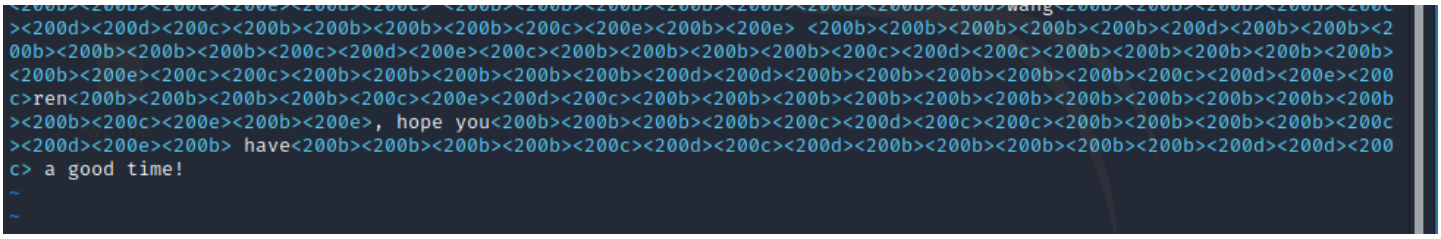
签到

两个txt文件，其中内容两个分别为

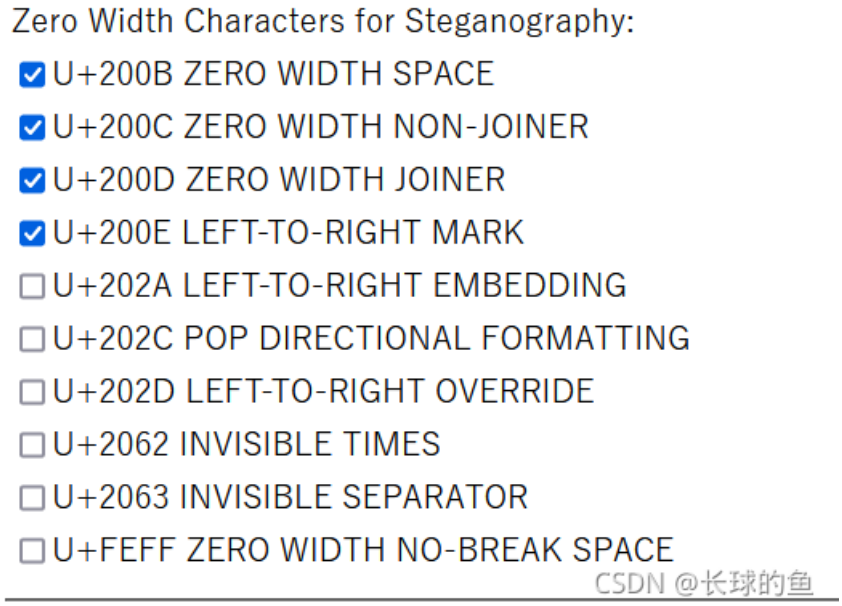


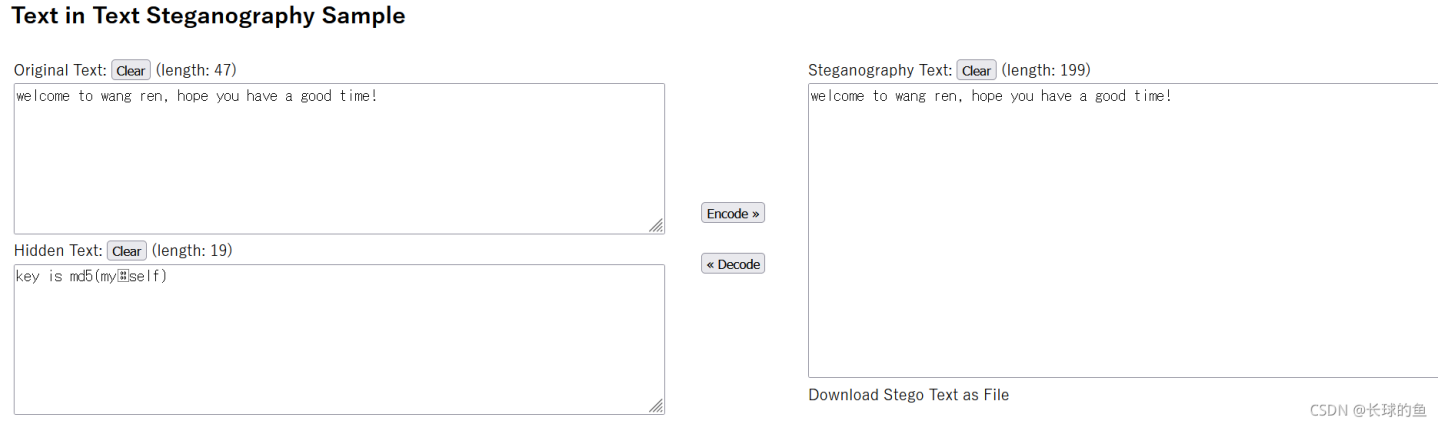
010查看和kali查看flag.txt后发现可能存在零宽度字符隐写





解码网站解密一下





the key is md5(myself)

接着我们md5加密一下flag.txt本身文件

```
f71b6b842d2f0760c3ef74911ffc7fdb
```


```
flag{WelY0me_2_bOl3an}
```