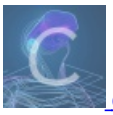
CTF总结之MISC

原创

醉等佳人归 于 2021-05-20 10:00:20 发布 987 收藏 15

分类专栏： ctf

本文链接：https://blog.csdn.net/qq_43647628/article/details/116697808



文章目录

1.文件操作与隐写

1.1.文件类型识别

（1）使用File命令识别文件类型，格式：file 文件名

（2）使用010Editor二进制查看文件头手动判断

1.2.文件头残缺|错误

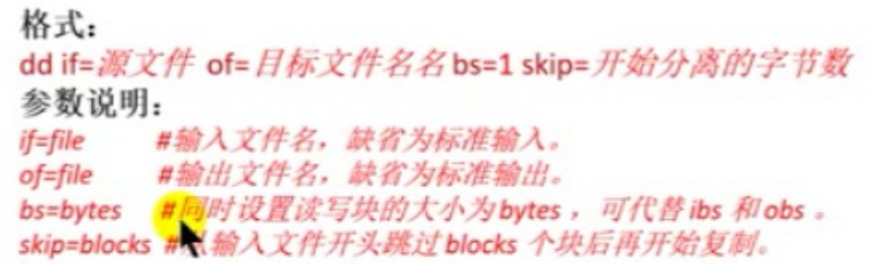
首先使用file判断文件类型，然后使用010Editor进行头部添加或者头部替换。

1.3.文件分离操作

（1）Binwalk工具
分析文件：binwalk filename
分离文件：binwalk -e filename

（2）foremost（推荐）
如果binwalk无法正确分离出文件，可以使用foremost。
用法：foremost 文件名 -o 输出目录名

（3）dd
当文件自动分离出错或者因为其他原因无法自动分离时，可以使用dd实现文件手动分离。
用法：



还有一个参数为count=块数

1.4.文件合并

cat 合并的文件 > 输出的文件

1.5.完整性检测

md5sum 文件名

1.6.文件内容隐写

直接将key以十六进制的形式写在文件中，通常在文件的开头或结尾部分，分析时通常重点观察文件开头和结尾部分，如果在文件中间部分，通常搜索关键字key或flag来查找隐藏内容。

2.图片隐写术

2.1.Filework

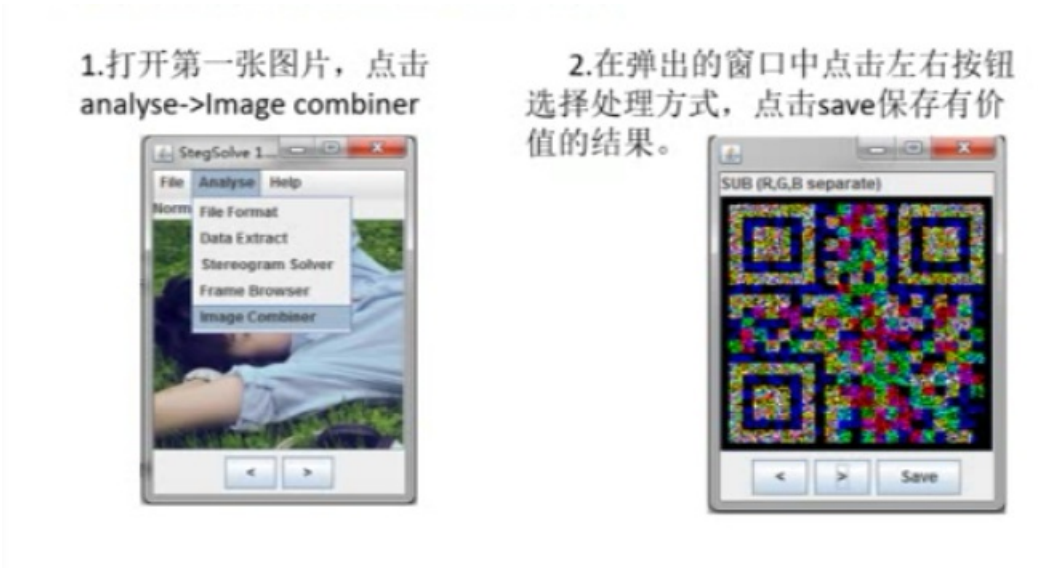
使用010Editor打开文件时会看到文件头部包含firework的标识，通过firework可以找到隐藏图片。
使用场景：查看隐写的图片文件

2.2.Exif

有时候信息会直接在exif中，图片右键属性，查看exif或者查看详细信息，在相关选项卡中查找flag信息。
使用exiftool filename也能查看。

2.3.Stegsolve

当两张jpg图片外观大小像素基本相同时，可以考虑结合分析，即对像素点进行XOR、ADD、SUB等在操作（记得比较的先后顺序不同也会导致操作的结果不同。）



作用有：通道分离、LSB分离、图像对比
得出二维码如果被反色了使用电脑图片编辑工具进行反色还原。

2.4.LSB（最低有效位Least Significant Bit）

1．像素三原色(RGB)
2．通过修改像素中最低位的1bit来达到隐藏的效果
3.工具: stegsolve（Data Extract ）、zsteg、wbstego4、python脚本

2.5.TweakPNG

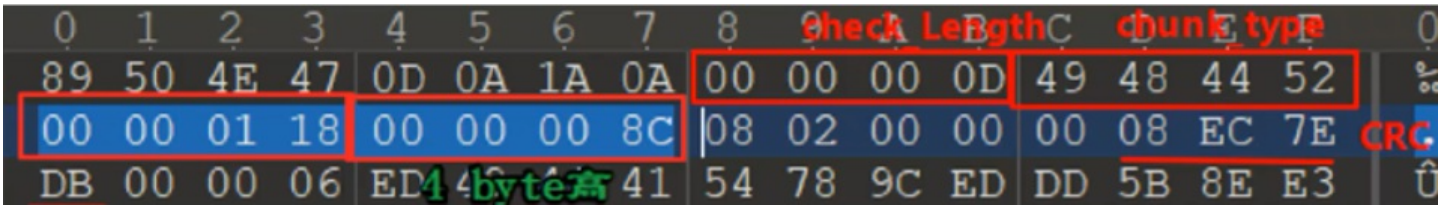
TweakPNG是一款简单易用的PNG图像浏览工具,它允许查看和修改一些PNG图像文件的元信息存储。

图像头部不止是文件类型的标识;还包含了长度(或者叫宽度)、高度以及校验值。
通常情况下，一个图像的产生，就会被赋予各种文件头部信息;一般情况下图像刚产生的时候，长宽高等信息会被校验，产生一个值X1。
如果这个图像被修改了，有可能产生图像无法打开或者显示不全：
（1）校验值被修改为X2:使用Tweak去检验的时候，通过长宽高算出来的值是X1跟现在的X2不一致，产生报错。
（2）高度被修改，比如减半:通过错误的长宽高，算出错误的校验值X3

使用场景:文件头正常却无法打开文件，利用TweakPNG修改CRC

例如当PNG文件头正常但无法打开文件，可能是CRC校验出错，可以尝试通过TweakPNG打开PNG，会弹出校验错误的提示，这里显示CRC是fe1a5ab6，正确的是b0a7a9f1。打开winhex搜索fe1a5ab6将其改为b0a7a9f1。

又例如有时cRc没有错误，但是图片的高度或者宽度发生了错误，需要通过CRc计算出正确的高度或者宽度。



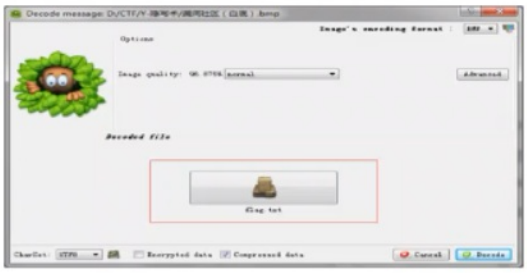
2.6.加密图片的解密

（1）BFtools工具

使用场景：在windows的cmd下，对加密过的图片文件进行解密
格式：
Bftools.exe decode braincopter *要解密的图片名称* –output *输出文件名*
Bftools.exe run *上一步输出的文件*

（2）SilentEye

silenteye是一款可以将文字或者文件隐藏到图片的加解密工具。
使用场景：windows下打开silentEye工具，对加密的图片进行解密
例：
1.使用silentEye程序打开目标图片，点击image->decode，点击decode，可以查看隐藏文件，点击保存即可

（3）Stegdetect工具探测加密方式

Stegdetect程序主要用于分析JPEG文件。因此用Stegdetect可以检测到通过JSteg、JPHide、OutGuess、Invisible Secrets、F5、appendX和Camouflage等这些隐写工具隐藏的信息。

stegdetect xxx.jpg
stegdetect –s *敏感度* *xxx.jpgexi*

```
thinking@ubuntu:~/Desktop$ stegdetect 123456.jpg
123456.jpg : f5(***)
thinking@ubuntu:~/Desktop$ stegdetect angrybird.jpg
angrybird.jpg : outguess(old)(*)
thinking@ubuntu:~/Desktop$ stegdetect Pcat.jpg
Pcat.jpg : negative
thinking@ubuntu:~/Desktop$ stegdetect -s 10.0 Pcat.jpg
Pcat.jpg : jphide(*)
thinking@ubuntu:~/Desktop$
```

windows使用

```
stegdetect.exe 123456.jpg
```

```
stegdetect.exe -s 10   Pcat.jpg
```

2.7.二维码处理

9.二维码处理

1.使用二维码扫描工具CQR.exe打开图片，找到内容字段

2.如果二维码某个定位角被覆盖了，该工具有时候也可以自动识别，如果识别失败，需要使用PS或画图工具将另外几个角的定位符移动到相应的位置，补全二维码。

3.压缩文件分析

3.1.伪加密

如果压缩文件是加密的，或文件头正常但解压缩错误，首先尝试文件是否为伪加密。zip文件是否加密是通过标识符来显示的，在每个文件的文件目录字段有一位专门标识了文件是否加密，将其设置为oo表示该文件未加密，如果成功解压则表示文件为伪加密，如果解压出错说明文件为真加密。

操作方法:使用winhex打开压缩文件，找到文件头（从一开始数 ）第九第十个字符，将其修改为0000。

使用winhex打开文件搜索16进制504B0102，可以看到每个加密文件的文件头字段。

3.2.暴力破解

通常我们可以使用ARCHPR.exe上具来破解zip文件使用场景: windows下加密过的zip文件

攻击类型选择暴力破解，在范围位置根据提示选择暴力破解范围选项设置暴力破解包含的类型、开始于和结束于选项具体范围，如果没有定义则全范制暴力破解。点击打开选择要破解的文件，点击开始进行破解。建议使用1~9位的数字密码，以及系统自带的英文字典作为密码字典。

明文攻击指知道加密的ZIP中部分文件的明文内容，利用这些内容推测出密钥并解密ZIP文件的攻击方法，相比于暴力破解，这种方法在破解密码较为复杂的压缩包时效率更高。

使用场景：已知加密的zip部分文件明文内容

例：假设一个加密的压缩包中有两个文件readme.txt和flag.txt，其中flag.txt的内容是我们希望知道的内容，而我们拥有readme.txt的明文文件，使用上述两个文件即可进行明文攻击。

操作：

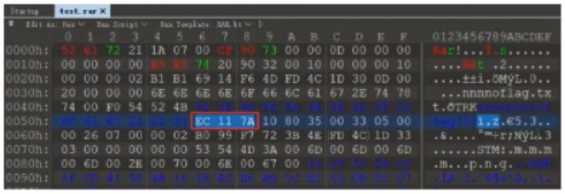
1、将readme.txt的明文文件进行压缩，变成readme1.zip。

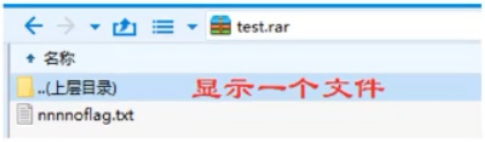
2、打开archpr，攻击类型选择明文，明文文件路径选择readme1.zip（即将明文文件不加密压缩后的文件），加密的文件

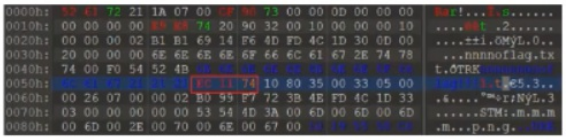
选择要破解的文件，点击开始，破解成功后会获得密码。

RAR文件格式

有时候给出的RAR文件的头部各个字块会故意给错导致无法识别。







HEAD_CRC 2 字节 所有块或块部分的CRC
HEAD_TYPE 1 字节 块类型
HEAD_FLAGS 2 字节 块标记
HEAD_SIZE 2 字节 块大小 #如果块标记的第一位被置1的话，还存在：
ADD_SIZE 4 字节 可选结构 - 增加块大小

那么，文件块的第3个字节为块类型，也叫头类型。
头类型是0x72表示是标记块
头类型是0x73表示是压缩文件头块
头类型是0x74表示是文件头块
头类型是0x75表示是注释头
.....

4.流量取证技术(WireShark过滤器)

利用wireshark本身的强大的报文过滤器，帮助我们筛选出想要的报文。

首先统计分级，可以将关注的数据包设置成过滤器

然后在关注的http数据报或tcp数据报中选择流汇聚，可以将HTTP流或TCP流汇聚或还原数据。

使用wireshark还可以自动提取通过http传输的文件内容。（文件->导出对象->HTTP）

4.1.流量包文件分析

1.过滤IP，如源IP或者目标 x.x.x.x

```
ip.src eq x.x.x.x or ip.dst eq x.x.x.x 或者 ip.addr eq x.x.x.x
```

2.过滤端口

```
tcp.port eq 80 or udp.port eq 80
tcp.dstport == 80  只显tcp协议的目标端口为80
tcp.srcport == 80  只显tcp协议的源端口为80
tcp.port >= 1 and tcp.port <= 80
```

3.过滤协议

```
tcp/udp/arp/icmp/http/ftp/dns/ip……
```

4.过滤MAC

```
eth.dst == A0:00:00:04:C5:84 过滤目标mac
```

5.包长度过滤

```
udp.length == 26 这个长度是指udp本身固定长度8加上udp下面那块数据包之和 。
tcp.len >= 7 指的是ip数据包(tcp下面那块数据),不包括tcp本身
ip.len == 94 除了以太网头固定长度14,其它都算是ip.len,即从ip本身到最后
frame.len == 119 整个数据包长度,从eth开始到最后
```

6.http模式过滤

```
http.request.method == "GET"
http.request.method == "POST"
http.request.uri == "/img/logo-edu.gif"
http contains "GET"
http contains "HTTP/1."
http.request.method == "GET" && http contains "User-Agent:"
http contains "flag"
http contains "key"
tcp contains "flag"
```

4.2.无限流量包跑密码

aircrack-ng 工具进行wifi 密码破解

1. 用aircrack-ng检查cap包：*aircrack-ng xxx.cap*

```
root@kali:~/Desktop# aircrack-ng shipin.cap
Opening shipin.cap
Read 16664 packets.

   #  BSSID              ESSID                     Encryption

   1  00:1D:0F:5D:D0:EE  0719                      WPA (1 handshake)

Choosing first network as target.

Opening shipin.cap
Please specify a dictionary (option -w).

Quitting aircrack-ng...
```

2. 用aircrack-ng跑字典进行握手包破解：*aircrack-ng xxx.cap –w pass.txt*

```
root@kali:~/Desktop# aircrack-ng shipin.cap -w pass.txt
Opening shipin.cap
Read 16664 packets.

   #  BSSID              ESSID                     Encryption

   1  00:1D:0F:5D:D0:EE  0719                      WPA (1 handshake)

Choosing first network as target.

Opening shipin.cap
Reading packets, please wait...

                                Aircrack-ng 1.2 rc4

      [00:00:00] 8/2492 keys tested (296.77 k/s)

      Time left: 8 seconds                                      0.32%

                            KEY FOUND! [ 88888888 ]


      Master Key     : B4 30 38 0F 24 7B 57 AC DE B5 3A 7F 2E FE 6B 45
                       0B 34 02 C3 89 F9 69 D5 B7 35 87 1B FB 4C EE 7F

      Transient Key  : 17 AE 23 D0 69 7C 0D 45 2B 40 F6 7D 06 C9 C5 6F
                       25 F0 B0 48 7A 6C 22 7C E2 73 50 71 46 FE 5D 0C
                       8F 59 01 BE 66 56 DF 1E 58 DD 34 DB BF A7 2D FD
                       2C 53 11 7F B2 E5 F0 16 7F 57 F5 6A 04 36 F5 71

      EAPOL HMAC     : 75 19 C5 F3 3E 33 58 23 CA 4B A1 85 FB 46 C0 2A
```

4.3.键盘流量包文件分析

USB协议的数据部分在Leftover Capture Data域之中。
右键leftover capture data –> 应用为列。

No.	Time	Source	Destination	Protocol	Length	Leftover Capture Data	Info
3	0.520044	3.10.1	host	USB	35	00fe0000feff0000	URB_INTERRUPT in
953	54.399801	3.10.1	host	USB	35	00fdff00fdffffff	URB_INTERRUPT in
947	54.351857	3.10.1	host	USB	35	00fdff00fdffffff	URB_INTERRUPT in
854	49.111879	3.10.1	host	USB	35	00fdff00fdffffff	URB_INTERRUPT in
691	36.423471	3.10.1	host	USB	35	00fdff00fdffffff	URB_INTERRUPT in

```
> Frame 3: 35 bytes on wire (280 bits), 35 bytes captured (280 bits)
> USB URB
  Leftover Capture Data: 00fe0000feff0000

0000  1b 00 f0 8a 42 83 03 97  ff ff 00 00 00
0010  01 03 00 0a 00 81 01 08  00 00 00 00 fe
0020  ff 00 00
```

展开子树(X) Shift+Right
展开全部(E) Ctrl+Right
收起全部(A) Ctrl+Left
应用为列

可以将该域的值在主面板上显示，键盘数据包的数据长度为8个字节，击键信息集中在第3个字节，每次key stroke都会产生一个keyboard event usb packet 。

USB键盘流量抓取分析

Leftover Capture Data中值与具体键位的对应关系，可以参考：
http://www.usb.org/developers/hidpage/Hut1_12v2.pdf

Table 12: Keyboard/Keypad Page

Usage ID (Dec)	Usage ID (Hex)	Usage Name	Ref: Typical AT-101 Position	PC-AT	Mac	UNIX	Boot
0	00	Reserved (no event indicated)[9]	N/A	√	√	√	4/101/104
1	01	Keyboard ErrorRollOver[9]	N/A	√	√	√	4/101/104
2	02	Keyboard POSTFail[9]	N/A	√	√	√	4/101/104
3	03	Keyboard ErrorUndefined[9]	N/A	√	√	√	4/101/104
4	04	Keyboard a and A[4]	31	√	√	√	4/101/104
5	05	Keyboard b and B	50	√	√	√	4/101/104
6	06	Keyboard c and C[4]	48	√	√	√	4/101/104
7	07	Keyboard d and D	33	√	√	√	4/101/104
8	08	Keyboard e and E	19	√	√	√	4/101/104
9	09	Keyboard f and F	34	√	√	√	4/101/104
10	0A	Keyboard g and G	35	√	√	√	4/101/104
11	0B	Keyboard h and H	36	√	√	√	4/101/104
12	0C	Keyboard i and I	24	√	√	√	4/101/104
13	0D	Keyboard j and J	37	√	√	√	4/101/104
14	0E	Keyboard k and K	38	√	√	√	4/101/104
15	0F	Keyboard l and L	39	√	√	√	4/101/104
16	10	Keyboard m and M[4]	52	√	√	√	4/101/104
17	11	Keyboard n and N	51	√	√	√	4/101/104
18	12	Keyboard o and O[4]	25	√	√	√	4/101/104
19	13	Keyboard p and P[4]	26	√	√	√	4/101/104
20	14	Keyboard q and Q[4]	17	√	√	√	4/101/104

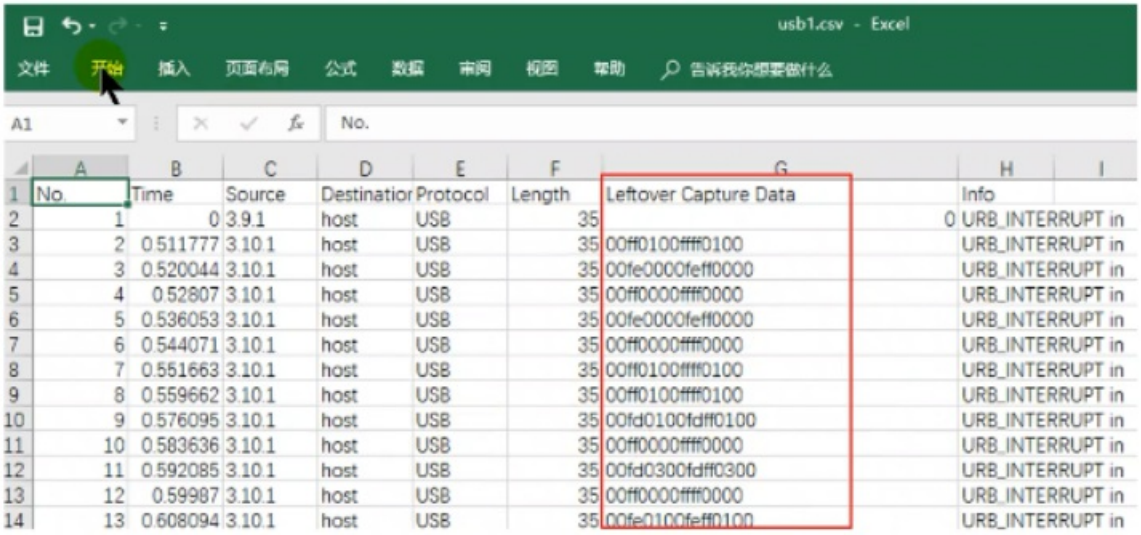
```
# a keyboard maping
mappings = { 0x04:"A",  0x05:"B",  0x06:"C", 0x07:"D", 0x08:"E",
0x09:"F", 0x0A:"G",  0x0B:"H", 0x0C:"I",  0x0D:"J", 0x0E:"K",
0x0F:"L", 0x10:"M", 0x11:"N",0x12:"O",  0x13:"P", 0x14:"Q",
0x15:"R", 0x16:"S", 0x17:"T", 0x18:"U",0x19:"V", 0x1A:"W",
0x1B:"X", 0x1C:"Y", 0x1D:"Z", 0x1E:"1", 0x1F:"2", 0x20:"3",
0x21:"4", 0x22:"5",  0x23:"6", 0x24:"7", 0x25:"8", 0x26:"9",
0x27:"0", 0x28:"\n", 0x2a:"[DEL]",  0X2B:"\t", 0x2C:" ",  0x2D:"
", 0x2E:"=", 0x2F:"-",  0x30:"]",  0x31:"\\", 0x32:"~", 0x33:";",
0x34:"'", 0x36:",",  0x37:".",0x82:"Casps Lock"}

nums = []
keys = open('usbdata.txt')
data=''
for line in keys:
    line = line.strip()
    if line[0]!='0' or line[1]!='0' or line[3]!='0' or
line[4]!='0' or line[9]!='0' or line[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':
        continue
    strs = str(line.split(":")[2])
    id = "0x"+strs
    if int(id,16) in mappings:
        data +=mappings[int(id,16)]
print data
keys.close()
```

USB键盘流量抓取分析

Leftover Capture Data 数据提取方式 1：

文件 → 导出分组解析结果 → 为CSV，导出保存成一个.csv文件。

在得到的csv文件的Leftover Capture Data可以将数据复制出来。此方式为通用复制数据方法。



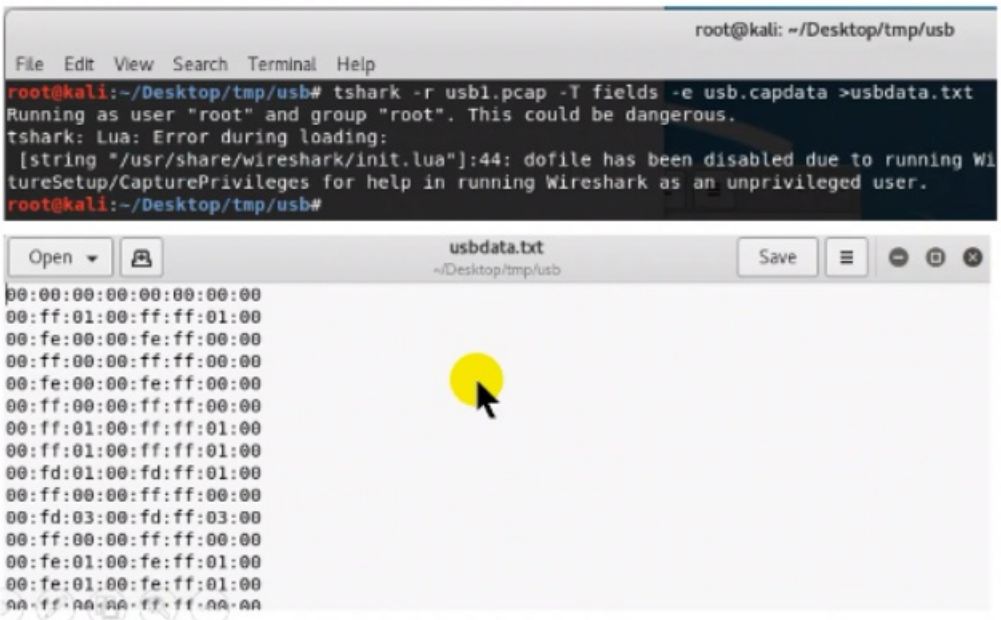
No.	Time	Source	Destination	Protocol	Length	Leftover Capture Data	Info
1	0	3.9.1	host	USB	35		0 URB_INTERRUPT in
2	0.511777	3.10.1	host	USB	35	00ff0100ffff0100	URB_INTERRUPT in
3	0.520044	3.10.1	host	USB	35	00fe0000feff0000	URB_INTERRUPT in
4	0.52807	3.10.1	host	USB	35	00ff0000ffff0000	URB_INTERRUPT in
5	0.536053	3.10.1	host	USB	35	00fe0000feff0000	URB_INTERRUPT in
6	0.544071	3.10.1	host	USB	35	00ff0000ffff0000	URB_INTERRUPT in
7	0.551663	3.10.1	host	USB	35	00ff0100ffff0100	URB_INTERRUPT in
8	0.559662	3.10.1	host	USB	35	00ff0100ffff0100	URB_INTERRUPT in
9	0.576095	3.10.1	host	USB	35	00fd0100fdff0100	URB_INTERRUPT in
10	0.583636	3.10.1	host	USB	35	00ff0000ffff0000	URB_INTERRUPT in
11	0.592085	3.10.1	host	USB	35	00fd0300fdff0300	URB_INTERRUPT in
12	0.59987	3.10.1	host	USB	35	00ff0000ffff0000	URB_INTERRUPT in
13	0.608094	3.10.1	host	USB	35	00fe0100feff0100	URB_INTERRUPT in

USB键盘流量抓取分析

Leftover Capture Data 数据提取方式 2：

使用wireshark提供的命令行工具 tshark，可以将Leftover Capture Data数据单独复制出来。

tshark -r usb1.pcap -T fields -e usb.capdata > usbdata.txt



4.4.鼠标流量包文件分析

USB鼠标流量抓取分析

鼠标流量与键盘流量不同，鼠标移动时表现为连续性，与键盘的离散性不一样。但是实际鼠标产生的数据是离散的。所以同样可以把数据抓取出来，构成二维坐标画出轨迹。

鼠标数据包的数据长度为4个字节，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。

文件(F) 编辑(E) 视图(V) 跳转(G) 捕获(C) 分析(A) 统计(S) 电话(Y) 无线(W) 工具(T) 帮助(H)

应用显示过滤器 ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Leftover Capture Data	Info
10	1.030704	2.3.1	host	USB	31	00030b00	URB_INTERRUPT in
11	1.030704	2.3.1	host	USB	31	00011100	URB_INTERRUPT in
12	1.051867	2.3.1	host	USB	31	00001800	URB_INTERRUPT in
13	1.052340	2.3.1	host	USB	31	00fd1e00	URB_INTERRUPT in
14	1.052340	2.3.1	host	USB	31	00fb2800	URB_INTERRUPT in
15	1.068194	2.3.1	host	USB	31	00fe2a00	URB_INTERRUPT in
16	1.083866	2.3.1	host	USB	31	00fd2d00	URB_INTERRUPT in
17	1.083866	2.3.1	host	USB	31	00fe2800	URB_INTERRUPT in
18	1.099498	2.3.1	host	USB	31	00002000	URB_INTERRUPT in
19	1.099498	2.3.1	host	USB	31	00000e00	URB_INTERRUPT in
20	1.330480	2.3.1	host	USB	31	00ff0100	URB_INTERRUPT in

第二个字节代表左右偏移；
当值为正时，代表右移多少像素。
当值为负时，代表左移多少像素。

同理，第三个字节代表上下偏移。

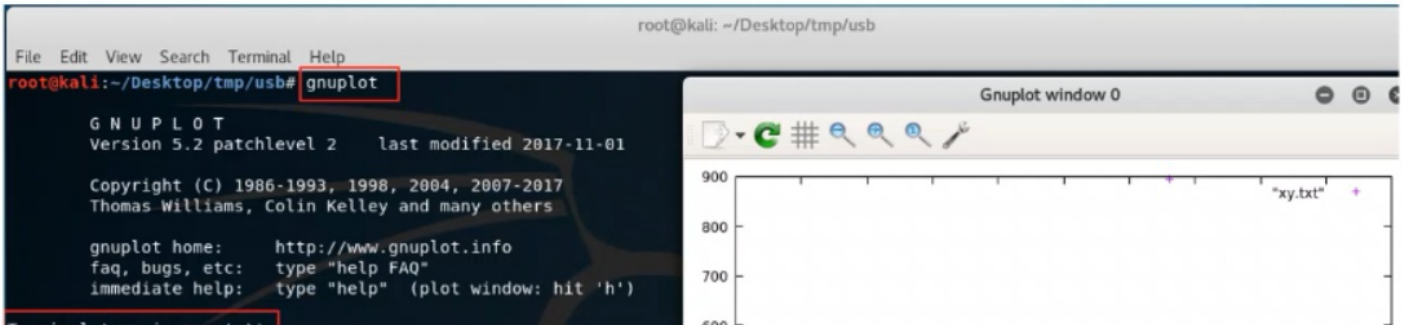
USB鼠标流量抓取分析

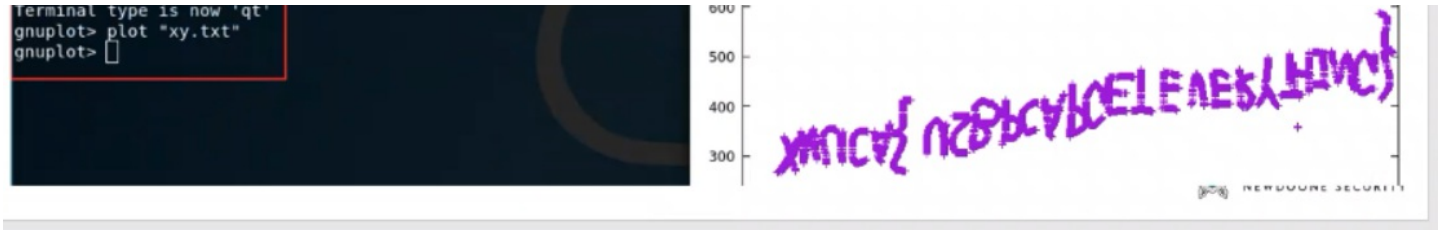
鼠标流量与键盘流量不同，鼠标移动时表现为连续性，与键盘的离散性不一样。但是实际鼠标产生的数据是离散的。所以同样可以把数据抓取出来，构成二维坐标画出轨迹。

```
nums = []
keys = open('usbdata.txt', 'r')
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)# 1 for left, 2 for right, 0 for nothing
    print btn_flag
    if btn_flag == 1:
        print posx, posy
keys.close()
```

USB鼠标流量抓取分析

用gnuplot工具把坐标画出来。
gnuplot
plot “xy.txt”





4.5.HTTPS流量包文件分析