杂项

原创

不会绑马尾的女孩 于 2021-01-08 23:39:24 发布 388 收藏

分类专栏： web

本文链接：https://blog.csdn.net/qq_45655136/article/details/112385883

这是大二上学期所学到的一些东西，分享一下

工具

Win Hex

WinHex是一款在Windows下的十六进制编辑器，在计算机取证，数据恢复，低级数据处理和IT安全领域特别有用。 日常和紧急使用的高级工具：检查和编辑各种文件，从文件系统损坏的硬盘驱动器或数码相机卡中恢复已删除的文件或丢失的数据。

010Editor

010Editor是一款快速且强大的十六进制编辑器。用来编辑二进制文件。有一个友好易于使用的界面，无限次的undo和redo操作。另外还可以打印十六进制的字节或者以书签的方式标出某些重要的字节。支持二进制模板（binary template）系统。

1、wireshark

and 逻辑与

or 逻辑或

xor ^^逻辑异或

not !逻辑非

snmp||dns||icmp显示SNMP或DNS或ICMP封包

ip.src==10.230.0.0/16

tcp.port==25

tcp.dstport==25

http.request.method==“POST”

http.host=="tracker.1ting.com"显示请求域名为tracker…的http封包

tcp contains “http” 显示payload中包含"http"字符串的tcp封包

http.requests.uri contains “online” 显示请求的uri包含“online"http的封包

其他

IO图形工具

Statistics统计

文件提取

2、Tshark

http.request.uri contains “flag” and ip.dst==192.168.173.134

常用参数：

-r: 设置读取本地文件

-R：包的读取过滤器

-Y：使用读取过滤器的语法，在单次分析中可以代替-R选项

-T：设置解码结果输出的格式，包括text,ps,psml和pdml，默认为text

-e:如果-T选项指定，-e用来指定输出哪些字段

盲注流量包

tshark -r hack.pcap -Y “http.requests.uri contains “flag” and ip.dst==192.168.173.134” -T fields -e http.requests.uri >1.txt

```
import re

count=1
oldch=0
flag=''

with open('1.txt','r') as f:
    for x in f.readlines():
        reg=re.match('/.*?,1\),(\d+).*?=(\d+)',a)
        #print reg.group(1)
        #print reg.group(2)

        if (reg):
            pos=int(reg.group(1))
            nch=int(reg.group(2))
            if pos>count:
                flag+=chr(oldch)
                count=pos
                oldch=nch
            else:
                oldch=nch
print flag
```

3、键盘流量

USB协议的数据部分在leftover Capture Data域之中

数据长度为八个字节，USB流量分为键盘流量和鼠标流量

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

可以用tshark命令(使用到wireshark里的工具tshark，这是wireshark工具的命令行程序，)可以将leftover capture data进行提取

tshark -r 流量包 -T filelds -e usb.capdata >usbdata.txt

提取出来后根据映射关系还原即可

题目：

【NSCTF】安全评测人员在对某银行卡密码输入系统进行渗透测试，截获了一段通过USB键盘输入6位数字密码的流量，其中也包含了一些其他无关的USB设备的流量，你能从中恢复出6位数字密码吗？最终提交的flag格式为flag

（1）使用tshark 命令把pcap的数据提取并去除空行到`usbdata.txt`

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

```
keyboard.py
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:"    ", 0x2C:" ",  0x2D:"-", 0x2E:"=", 0
x2F:"[",  0x30:"]",  0x31:"\\", 0x32:"~", 0x33:";",  0x34:"'", 0x36:",",  0x37:"." }
nums = []
keys = open('usbdata.txt')
for line in keys:
    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 l
ine[22]!='0':
        continue
    nums.append(int(line[6:8],16))
keys.close()
output = ""
for n in nums:
    if n == 0 :
        continue
    if n in mappings:
        output += mappings[n]
    else:
        output += '[unknown]'
print 'output :\n' + output
```

（2）提取出来的数据可能会带冒号，也可能不带（有可能和wireshark的版本相关），但是一般的脚本都会按照有冒号的数据来识别

> 有冒号时提取数据的`[6:8]`
> 无冒号时数据在`[4:6]`

```
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()
```

（3）提取出键盘流量后需要用脚本还原数据对应的信息。

python keyboard.py

BCFGIJGFEDCABACFEDCA7200[DEL]53[DEL]93
因为[DEL]是删除键,恢复出6位数字。所以flag: 720593

4、鼠标流量

USB协议鼠标数据部分在Leftover Capture Data域中，数据长度为`四个字节`。

> 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。

如图，数据信息为0x00002000，表示鼠标垂直向上移动20。

3.题目示例：

【NSCTF】这是一道鼠标流量分析题。
提取码：q6ro
（1）使用tshark 命令把pcap的数据提取并去除空行到`usbdata.txt`

```
tshark -r usb2.pcap -T fields -e usb.capdata | sed '/^\s*$/d' > usbdata.txt
1
```

（2）使用上面提到过的加冒号的脚本，并将脚本里提到的`16`改为`8`，得到

```
python3 maohao.py
```

1.题型：

flag隐藏在usb流量中，通过USB协议数据中的鼠标移动轨迹转换成flag。

2.解题思路：

1.使用kali linux中的tshark 命令把cap data提取出来，并去除空行

```
tshark -r usb2.pcap -T fields -e usb.capdata > usbdata.txt
tshark -r usb2.pcap -T fields -e usb.capdata | sed '/^\s*$/d' > usbdata.txt #提取并去除空行
12
```

2.根据usb协议鼠标数据还原鼠标移动轨迹，可写一个Python脚本进行快速还原。

（3）使用mouse.py测试信息隐藏位置

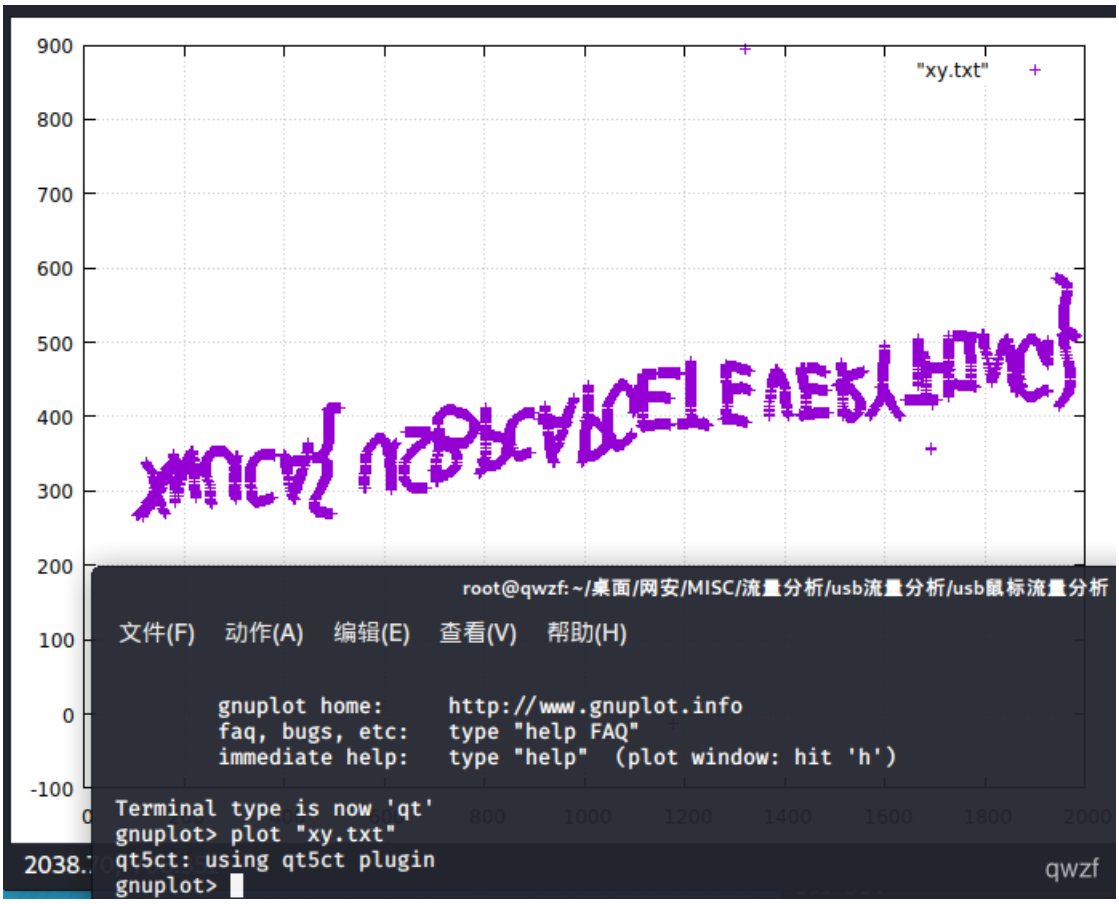
```
nums = []
keys = open('out.txt','r')
f = open('xy.txt','w')
posx = 0
posy = 0
for line in keys:
    if len(line) != 12 :
        continue
    x = int(line[3:5],16)
    y = int(line[6:8],16)
    if x > 127 :
        x -= 256
    if y > 127 :
        y -= 256
    posx += x
    posy += y
    btn_flag = int(line[0:2],16)  # 1 for left , 2 for right , 0 for nothing
    if btn_flag == 2 : # 1 代表左键
        f.write(str(posx))
        f.write(' ')
        f.write(str(posy))
        f.write('\n')

f.close()
12345678910111213141516171819202122324
```

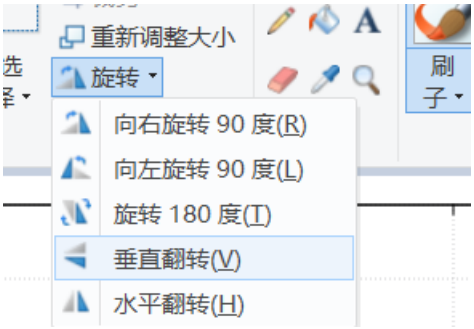
测试发现flag信息藏在右键中，即当脚本中btn_flag取2时可以得到一系列坐标

（4）用gnuplot将`xy.txt`里的坐标转化成图像

```
gnuplot
gnuplot>plot "xy.txt"
12
```



发现方向反了，使用windows上的"画图"垂直翻转一下即可。



最终得到flag

5、压缩包

zip

压缩源文件数据区0x50 4B 03 04

核心目录区0x50 4B 01 02、

目录结束标志0x50 4B 05 06

格式缺失：有的压缩包故意删去文件头、文件尾，可以用binwalk进行分析

zip伪加密：[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-c4mzt8Ez-1610120318772)(C:\Users\Tian\AppData\Roaming\Typora\typora-user-images\image-20201121124250329.png)]

zip爆破密码：archpr

zip明文攻击：手里有原压缩包和压缩包内部分文件，可以使用明文攻击方式，可以将压缩包内的文件进行压缩， 然后用archpr进行明文攻击

crc32爆破：[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-DfAKjx3Q-1610120318775)(C:\Users\Tian\AppData\Roaming\Typora\typora-user-images\image-20201121125051803.png)]

6、图片隐写

PNG文件

> 89 50 4E 47 0D 0A 1A 0A+数据块+数据块+数据块…

PNG 定义了两种类型的数据块，一种是称为关键数据块（critical chunk），这是标准

的数据块，另一种叫做辅助数据块（ancillarychunks），这是可选的数据块。关键数

据块定义了 4 个标准数据块，每个 PNG 文件都必须包含它们，PNG 读写软件也都必

须要支持这些数据块。

IHDR**（文件头数据块）**

文件头数据块 IHDR（HeaderChunk）：它包含有 PNG 文件中存储的图像数据的基本信息，由 13

字节组成，并要作为第一个数据块出现在 PNG 数据流中，而且一个 PNG 数据流中只能有一个文件

头数据块，其中我们只关注前8字节的内容

图片长宽信息隐藏

**经常会去更改一张图片的高度或者宽度使得一张图片显示不完整从而达到隐藏信息的目的。
Kali中不可以打开， 示文件头错误，而Windows自带的图片查看器可以打开，就 醒了我们IHDR被人篡改过

图片中直接追加内容

两张图片合并，图片尾追加文件，图片尾追加文字，都不影响，只要找到文件尾标识结构即可

图片元数据

元数据（Metadata），又称中介数据、中继数据，为描述数据的数据（Data about data），主要是描述数据属性（property）的信息，用来支持如指示存储位置、历史数据、资源查找、文件记录等功能。
元数据中隐藏信息在比赛中是最基本的一种手法，通常用来隐藏一些关键的 Hint 信息或者一些重要的比如password 等信息。
这类元数据可以 右键 -> 属性 查看

Llnux下直接使用exiftool工具查看

IDAT信息隐藏

IDAT：存储实际的数据，在数据流中可包含多个连续顺序的图像数据块。
储存图像像数数据。
在数据流中可包含多个连续顺序的图像数据块。
采用 LZ77 算法的派生算法进行压缩。
可以用 zlib 解压缩。

IDAT块只有当上一个块充满时，才会继续下一个新块。*
也可以使用Stegsolve -> Format Analysis有详细介绍

将二进制信息转化为二维码

```
import Image
img=Image.new('GRB',(25,25))
col= #二维码
i=0
for y in xrange(0,25):
    for x in xrange(0,25):
        if(col[i]=='1'):
            img.putpixel([x,y],(0,0,0))
        else:
            img.putpixel([x,y],(255,255,255))
        i=i+1
img.show()
```

file用来判断文件是什么类型，如果只返回data就说明可能文件头信息被删去了

7、LSB信息隐藏

LSB，最低有效位

PNG文件中的图像像数一般是由RGB三原色组成，每一种颜色占用8位，取值范围为

0×00~0xFF，即256种颜色，一共包含了256的三次方的颜色，即16777216（1千677W）种

颜色。人类的眼睛可以区分约1000万种不同的颜色，这就意味着人类的眼睛无法区分余下的

颜色大约有6777216（677W）种。

LSB隐写就是修改RGB颜色分量的最低二进制位（LSB），每个颜色都会有8bit，LSB隐写就

是修改了像数中的最低的1Bit，而人类的眼睛不会注意到这前后的区别，每个像数可以携带

3Bit的信息，这样就把信息隐藏起来了

GIF可能为动态图

所以可能是单张图片中有隐藏信息，也可能是图片的播放时间本身携带信息

可以使用identify命令去拆解GIF

8、其他杂项内容

NTFS流隐写

pyc或pyo中隐藏数据

识图

社工。。。

```
a=''#一串二进制数
b=int(a,2)
c=hex(b)#转换为十六进制数

print c[2:-1].decode('hex')
```

4、题解

EXIF信息，是可交换图像文件的缩写，是专门为数码相机的照片设定的，可以记录数码照片的属性信息和拍摄数据。

89 50 4E PE头是png照片的，就是说没有可能照片中嵌入了Exif信息,在查看PNG文件格式时，IHDR后面的八个字节就是宽高的值

将图片放在linux下，发现打不开，说明图片被截了

将图片的高改成和宽一样，然后再打开图片就有flag

眼见非实 zip

50 4B 03 04是压缩文件的头，还有.docx文件，应该压缩包里有一个文档，改后缀名为.zip

在winhax中发现.docx是一个.zip文件，改后缀名为.zip得到一个文件夹,然后打开文件发现flag

啊哒

拿到压缩包，解压后放在winhax中，发现有flag.txt文件

在linux中执行：binwalk ada.jpg

发现内含一个zip压缩包，分离文件 dd if=ada.jpg of=myzip skip=218773 bs=1

压缩包需要密码，查看原图片文件的详细信息，发现照相机型号，将其转换为字符串，得到解压密码

> if指定输入文件，of指定输出文件，skip是指定从输入文件开头跳过218773个块后在开始复制，bs设置每次读写块的大小字节为1字节
> foremost也可以分离

宽带信息泄露

conf.bin文件是路由器的备份配置文件，用工具router_Pass_view可以直接打开，由题目知道是宽带用户名，搜索username即可得到flag

隐写2

kali:fcrackzip -D -p evil.txt -u flag.zip -v

-D就是用字典模式 -p指定起始破解密码 -u这个参数是为了显示密码 用-v展示更多的信息

爆破出密码后，得到3.jpg,放在winhex中查看

做个游戏heiheihei.jar

放到jd-gui中得到java源码，审计代码，然后发现了flag，

压缩命令

> tar -xvzf 1.tar.gz
> x-解压文件
> v-在解压每个文件时打印出文件的名称
> z-该文件是一个使用gzip压缩的文件‘
> f-使用接下来的tar归档来进行操作

tar -xvjf archivefile.tar.bz2

> 具有bz2扩展名的文件是使用bzip算法进行压缩的，但是tar命令也可以对其进行处理，但是需要通过使用“j”选项来替换“z”选项。

将文件解压到一个指定的目录或路径 -C

tar -xvzf 1.tar.gz -C /opt/folder/

```
*.tar 用 tar -xvf 解压
*.gz 用 gzip -d或者gunzip 解压
*.tar.gz和*.tgz 用 tar -xzf 解压
*.bz2 用 bzip2 -d或者用bunzip2 解压
*.tar.bz2用tar -xjf 解压
*.Z 用 uncompress 解压
*.tar.Z 用tar -xZf 解压
*.rar 用 unrar e解压
*.zip 用 unzip 解压
```

想蹭网先解开秘密

> WIFI连接认证的重点在WPA的四次握手包(即采用WPA加密方式的无线AP与无线客户端进行连接前的认证信息包，也就是发送指令与验证指令俗称加密方式hash），即eapol协议(基于局域网的扩展认证协议)的包
> [外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-lt7G9YC6-1610120318776)
> (C:\Users\Tian\AppData\Roaming\Typora\typora-user-images\image-20201120185735088.png)]

Crunch字典生成工具

> crunch min max characterset -t pattern -o output filename

min=最小密码长度

characterset=用于生成密码的字符集

pattern=生成密码的指定模式

outfputfile=保存字典文件的路径

% 插入数字 @插入小写字母，插入大写字母 ^插入符号

例如生成缺位的手机号码

crunch 11 11 -t 1503453%%%% -o 1.txt或>>1.txt

aircrack进行爆破

```
aircrack-ng -a2 所下载文件的地址 -w password.txt
```

-a:爆破

-e:选择essid wifi的名称

解题：

crunch 11 11 1391040%%%% >>wifipass.txt

aircrack-ng -a2 'wifi.cap' -w wifipass.txt

选择的握手包是第三个

LInux2

[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-7BKlcdOJ-1610120318780)(C:\Users\Tian\AppData\Roaming\Typora\typora-user-images\image-20201120190808991.png)]

用notepad++打开，搜索key即可

细心的大象

这个图片的备注里有一段被base64加密的信息，应该是一段密码

```
正常的.jpg图像文件的FF D8 FF E0 00 10 4A 46 49 46 00 01
```

但是这个大象.jpg没有，图片特别大，使用binwalk分析一下，发现有一个rar压缩包，用dd命令分隔出来

```
dd if=1.jpg of=8.rar bs=1 skip=6391983
```

得到一个1.rar文件，但是没有办法z在linux中显示图片，在windows下解压图片可以正常显示，说明图片被截了，使用winhex打开，将宽和高改为一样的

爆照

附件下载发现是一张图片，首先用binwalk提取，发现有很多文件，用foremost分离，得到很多文件，分别用binwalk来分析，发现88,888,8888有修改的痕迹

88文件：改为.jpg

888文件：改为.jpg,发现备注里有base64加密的信息

8888文件：改为.zip，解压后为二维码文件

LSB在二进制中意为最低有效位

NTFS格式化磁盘

猫片

hint：LSB BRG NTFS

补上后缀名png,打开stegsolve的图片数据信息提取，选择Data Extract,选择LSB和BGR

发现了里面隐藏了一张图片png我们将其提取出来，保存的格式bin(txt格式用winhex和010Editor打开都是乱码)，然后我们修改文件的后缀为png，然后发现图片打不开，用winhex打开

发现了并不是我们预期的png文件头，PNG (png)的正常的文件头：89504E47，所以们将他前面的FFFF删掉保存退出。

额 半张二维码@@，CRC值出错，很明显是高度出错引起的

发现一个问题这个二维码和我们平时见到的不太一样，正常正方形中间应该是黑色的。所以还要用画图工具进行反色

mmp flag竟然不在里面，我无从下手了，开始参考网上大佬的wp

度娘搜了下，发现另一位老铁写的writeup

https://www.jianshu.com/p/abc44c54857a

最后根据hint里面的提示“NTFS”，根据大佬的说法，这是一种流隐写，需要用到工具

ntfstreamseditor，然而。。这里还有一个坑就是，这压缩文件一定要用winrar来解压才会产生这样的效果

得到一个pyc文件，也就是py编译后的文件，因此需要扔到网上去在线反编译一下

这里推荐一个网站，可以反编译py, https://tool.lu/pyc/

根据他这个加密的脚本再写出一个解密的脚本，运行一下就可以得到flag了