

深入理解 USB 流量數據包的抓取與分析

發布於 2018-08-15 15:28:21 閱讀 2K

<https://cloud.tencent.com/developer/article/11862340>

0x01 問題提出

在一次演練中，我們通過 Wireshark 抓取了一個如下的數據包，我們如何對其進行分析？

No.	Time	Source	Destination	Protocol	Length	Bluetooth HCI Event	Info
1	0.000000	host	3.1.0	USBHUB	64		GET_STATUS Request [Port 8]
2	0.000008	3.1.0	host	USBHUB	68		GET_STATUS Response [Port 8]
3	0.000011	host	3.1.0	USBHUB	64		CLEAR_FEATURE Request [Port 8: PORT_SUSPEND]
4	0.042812	3.1.0	host	USBHUB	64		CLEAR_FEATURE Response [Port 8: PORT_SUSPEND]
5	0.042838	3.1.1	host	USB	66		URB_INTERRUPT in
6	0.042841	host	3.1.1	USB	64		URB_INTERRUPT in
7	0.086839	host	3.1.0	USBHUB	64		GET_STATUS Request [Port 8]
8	0.086846	3.1.0	host	USBHUB	68		GET_STATUS Response [Port 8]
9	0.102835	host	3.1.0	USBHUB	64		CLEAR_FEATURE Request [Port 8: C_PORT_SUSPEND]
10	0.102841	3.1.0	host	USBHUB	64		CLEAR_FEATURE Response [Port 8: C_PORT_SUSPEND]
11	0.102844	host	3.6.0	USB	64		GET_STATUS Request
12	0.103343	3.6.0	host	USB	66		GET_STATUS Response
13	0.103360	host	3.1.0	USBHUB	64		GET_STATUS Request [Port 8]
14	0.103365	3.1.0	host	USBHUB	68		GET_STATUS Response [Port 8]
15	0.103380	host	3.6.0	USB	64		GET_DESCRIPTOR Request DEVICE
16	0.103900	3.6.0	host	USB	63		GET_DESCRIPTOR Response DEVICE

> Frame 1: 64 bytes on wire (512 bits), 64 bytes captured (512 bits) on interface 0

> USB URB

> URB setup

bRequest: GET_STATUS (0x00)

wValue: 0x0000

wIndex: 8

wLength: 4

0000	c0 e6 89 b7 01 88 ff ff	53 02 00 01 03 00 00 3c S.....<
0010	36 34 2c 59 00 00 00 00	9d 58 08 00 8d ff ff ff	64,Y.... .X.....
0020	04 00 00 00 00 00 00 00	a3 00 00 00 08 00 04 00
0030	00 00 00 00 00 00 00 00	00 02 00 00 00 00 00 00

0x02 問題分析

流量包是如何捕獲的？

首先我們從上面的數據包分析可以知道，這是個 USB 的流量包，我們可以先嘗試分析一下 USB 的數據包是如何捕獲的。

在開始前，我們先介紹一些 USB 的基礎知識。USB 有不同的規格，以下是使用 USB 的三種方式：

- 1 USB UART
- 1 USB HID
- 1 USB Memory

UART 或者 Universal Asynchronous Receiver/Transmitter。這種方式下，設備只是簡單的將 USB 用於接受和發射數據，除此之外就再沒有其他通訊功能了。

HID 是人性化的接口。這一類通訊適用於交互式，有這種功能的設備有：鍵盤，鼠標，遊戲手柄和數字顯示設備。

最後是 USB Memory，或者說是數據存儲。External HDD, thumb drive / flash drive,等都是這一類的。

其中使用的最廣的不是 USB HID 就是 USB Memory 了。

每一個 USB 設備（尤其是 HID 或者 Memory）都有一個供應商 ID（Vendor Id）和產品識別碼（Product Id）。Vendor Id 是用來標記哪個廠商生產了這個 USB 設備。Product Id 用來標記不同的產品，他並不是一個特殊的數字，當然最好不同。如下圖

```
root@kali:~/桌面/usb/USB/UsbKeyboardDataHacker# lsusb
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 004: ID 0e0f:0008 VMware, Inc.
Bus 002 Device 003: ID 0e0f:0002 VMware, Inc. Virtual USB Hub
Bus 002 Device 002: ID 0e0f:0003 VMware, Inc. Virtual Mouse
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
```

上圖是我在虛擬機環境下連接在我電腦上的 USB 設備列表，通過 `lsusb` 查看命令。

例如說，我在 VMware 下有一個無線鼠標。它是屬於 HID 設備。這個設備正常的運行，並且通過 `lsusb` 這個命令查看所有 USB 設備，現在大家能找出哪一條是這個鼠標嗎？？沒有錯，就是第四個，就是下面這條：

```
Bus 002 Device 002: ID 0e0f:0003 VMware, Inc. Virtual Mouse
```

其中，ID `0e0f:0003` 就是 Vendor-Product Id 對，Vendor Id 的值是 `0e0f`，並且 Product Id 的值是 `0003`。Bus 002 Device 002 代表 usb 設備正常連接，這點需要記下來。

我們用 root 權限運行 Wireshark 捕獲 USB 數據流。但是通常來說我們不建議這麼做。我們需要給用戶足夠的權限來獲取 linux 中的 usb 數據流。我們可以用 `udev` 來達到我們的目的。我們需要創建一個用戶組 `usbmon`，然後把我們的賬戶添加到這個組中。

```
addgroup usbmon
```

```
gpasswd -a $USER usbmon
```

```
echo 'SUBSYSTEM=="usbmon", GROUP="usbmon", MODE="640"' > /etc/udev/rules.d/99-usbmon.rules
```

復制

接下來，我們需要 `usbmon` 內核模塊。如果該模塊沒有被加載，我們可以通過以下命令加載該模塊：

```
modprobe usbmon
```

復制

打開 `wireshark`，你會看到 `usbmonX` 其中 X 代表數字。下圖是我們本次的結果（我使用的是 root）：



欢迎使用 Wireshark

捕获

...使用这个过滤器: All interfaces shown

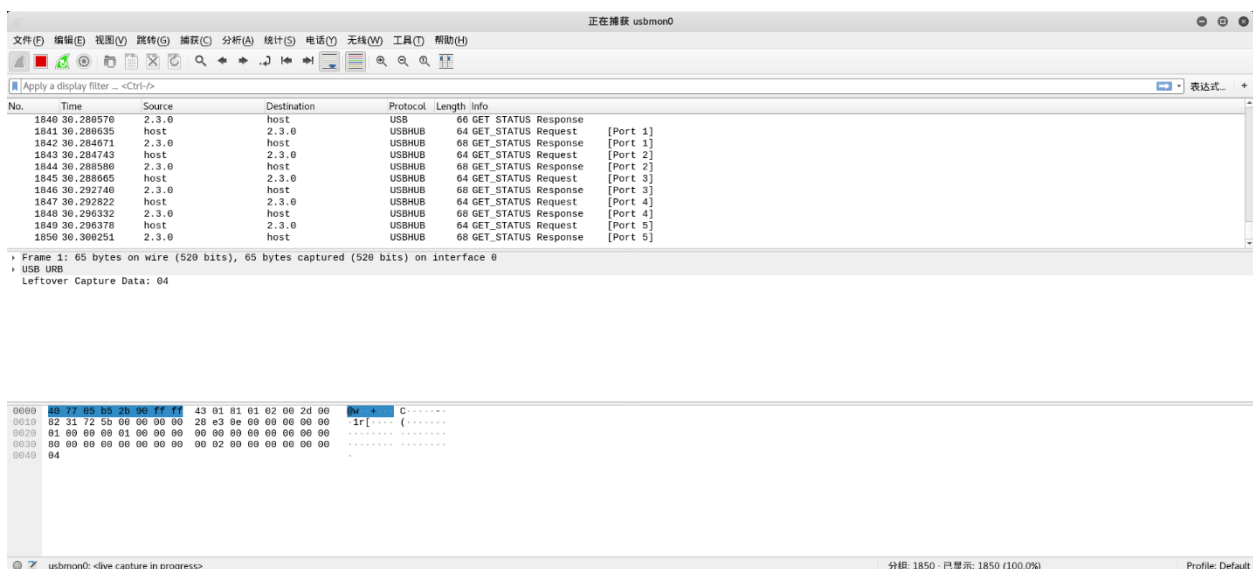


学习

[用户指导](#) · [Wiki](#) · [问题与解答](#) · [邮件列表](#)

正在运行 Wireshark2.6.1 (Git v2.6.1 packaged as 2.6.1-1).

如果接口處於活躍狀態或者有數據流經過的時候，wireshark 的界面就會把它以波形圖的方式顯示出來。那麼，我們該選那個呢？沒有錯，就是剛剛讓大家記下來的，這個 X 的數字就是對應這 USB Bus。在本文中是 `usbmon0`。打開他就可以觀察數據包了。



通過這些，我們可以瞭解到 usb 設備與主機之間的通信過程和工作原理，我們可以來對流量包進行分析了。

如何去分析一個 USB 流量包？

根據前面的知識鋪墊，我們大致對 USB 流量包的抓取有了一個輪廓了，下面我們介紹一下如何分析一個 USB 流量包。

USB 協議的細節方面參考 wireshark 的 wiki：<https://wiki.wireshark.org/USB>

我們先拿 GitHub 上一個簡單的例子開始講起：

No.	Time	Source	Destination	Protocol	Length	Bluetooth HCI Event	Info
1	0.000000	2.1.1	host	USB	35		URB_INTERRUPT in
2	0.137131	2.1.1	host	USB	35		URB_INTERRUPT in
3	0.299751	2.1.1	host	USB	35		URB_INTERRUPT in
4	0.399781	2.1.1	host	USB	35		URB_INTERRUPT in
5	0.838075	2.1.1	host	USB	35		URB_INTERRUPT in
6	0.968796	2.1.1	host	USB	35		URB_INTERRUPT in
7	1.184415	2.1.1	host	USB	35		URB_INTERRUPT in
8	1.316126	2.1.1	host	USB	35		URB_INTERRUPT in
9	1.599310	2.1.1	host	USB	35		URB_INTERRUPT in
10	1.934871	2.1.1	host	USB	35		URB_INTERRUPT in
11	2.054854	2.1.1	host	USB	35		URB_INTERRUPT in
12	2.067291	2.1.1	host	USB	35		URB_INTERRUPT in
13	2.384149	2.1.1	host	USB	35		URB_INTERRUPT in
14	2.484050	2.1.1	host	USB	35		URB_INTERRUPT in
15	3.000238	2.1.1	host	USB	35		URB_INTERRUPT in
Frame 1: 35 bytes on wire (280 bits), 35 bytes captured (280 bits) on USB URB							
Leftover Capture Data: 0000090000000000							

0000	1b 00 40 39 2d ac 89 b6 ff ff 00 00 00 09 00	..@-...
0010	01 02 00 01 00 81 01 08 00 00 00 00 09 00 00
0020	00 00 00	...

我們分析可以知道，USB 協議的數據部分在 Leftover Capture Data 域之中，在 Mac 和 Linux 下可以用 tshark 命令可以將 leftover capture data 單獨提取出來，命令如下：

```
tshark -r example.pcap -T fields -e usb.capdata //如果想導入 usbdata.txt 文件中，後面加上參數：>usbdata.txt
```

Windows 下裝了 wireshark 的環境下，在 wireshark 目錄下有個 tshark.exe，比如我的在 D:\Program Files\Wireshark\tshark.exe

此电脑 > Data (D:) > Program Files > Wireshark				搜索"Wireshark"
名称	修改日期	类型	大小	
reordercap.exe	2018/2/24 4:13	应用程序	322 KB	
services	2018/2/24 3:40	文件	949 KB	
smi_modules	2018/2/24 3:40	文件	1 KB	
task_AutoKey.pcapng	2018/8/10 9:03	Wireshark captu...	65 KB	
text2pcap.exe	2018/2/24 4:13	应用程序	344 KB	
text2pcap.html	2018/2/24 3:40	Chrome HTML D...	13 KB	
tshark.exe	2018/2/24 4:13	应用程序	565 KB	
tshark.html	2018/2/24 3:40	Chrome HTML D...	96 KB	
uninstall.exe	2018/2/24 4:13	应用程序	423 KB	

調用 cmd，定位到當前目錄下，輸入如下命令即可：

```
tshark.exe -r example.pcap -T fields -e usb.capdata //如果想導入 usbdata.txt 文件中，後面加上參數：>usbdata.txt
```

有關 tshark 命令的詳細使用參考 wireshark 官方文檔：<https://www.wireshark.org/docs/man-pages/tshark.html>

運行命令並查看 usbdata.txt 發現數據包長度為八個字節

```

root@kali:~/桌面/usb/USB/UsbKeyboardDataHacker# tshark -r example.pcap -T fields
-e usb.capdata
Running as user "root" and group "root". This could be dangerous.
tshark: Lua: Error during loading:
[string "/usr/share/wireshark/init.lua"]:32: dofile has been disabled due to ru
nning Wireshark as superuser. See https://wiki.wireshark.org/CaptureSetup/Captur
ePrivileges for help in running Wireshark as an unprivileged user.
00:00:09:00:00:00:00:00
00:00:00:00:00:00:00:00
00:00:0f:00:00:00:00:00
00:00:00:00:00:00:00:00
00:00:04:00:00:00:00:00
00:00:00:00:00:00:00:00
00:00:0a:00:00:00:00:00
00:00:00:00:00:00:00:00
20:00:00:00:00:00:00:00
20:00:2f:00:00:00:00:00
20:00:00:00:00:00:00:00
00:00:00:00:00:00:00:00
00:00:13:00:00:00:00:00
00:00:00:00:00:00:00:00
00:00:15:00:00:00:00:00

```

關於 USB 的特點應用我找了一張圖，很清楚的反應了這個問題：

性能	应用	特性
低速(1.5Mbps): ✓交互式设备 ✓10-100kbps	➤键盘，鼠标 ➤手写笔 ➤游戏手柄 ➤虚拟设备 ➤外设	•极低的成本 •易于使用 •热插拔 •同时使用多个外设
全速(12Mbps): ✓电话，音频类 ✓压缩的视频类 ✓500kbps – 10Mbps	➤话音 ➤宽带 ➤音频 ➤麦克风	•较低的成本 •易于使用 •热插拔 •同时使用多个外设 •可保证的带宽 •可保证的延迟
高速(480Mbps): ✓视频，大容量存储 ✓25 – 400Mbps	➤视频 ➤大容量存储 ➤图像 ➤宽带	•低成本 •易于使用 •热插拔 •同时使用多个设备 •可保证的带宽 •可保证的延迟 •高带宽

這裡我們只關注 USB 流量中的鍵盤流量和鼠標流量。

鍵盤數據包的數據長度為 8 個字節，擊鍵信息集中在第 3 個字節，每次 key stroke 都會產生一個 keyboard event usb packet。

鼠標數據包的數據長度為 4 個字節，第一個字節代表按鍵，當取 0x00 時，代表沒有按鍵、為 0x01 時，代表按左鍵，為 0x02 時，代表當前按鍵為右鍵。第二個字節可以看成是一個 signed byte 類型，其最高位為符號位，當這個值為正時，代表鼠標水平右移多少像素，為負時，代表水平左移多少像素。第三個字節與第二字節類似，代表垂直上下移動的偏移。

我翻閱了大量的 USB 協議的文檔，在這裡我們可以找到這個值與具體鍵位的對應關係：

http://www.usb.org/developers/hidpage/Hut1_12v2.pdf

usb keyboard 的映射表 根據這個映射表將第三個字節取出來，對應對照表得到解碼：

Table 12: Keyboard/Keypad Page

Usage ID (Dec)	Usage ID (Hex)	Usage Name	Ref: Typical AT-101 Position	PC- AT	Mac UNI X	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

我們寫出如下腳本：

```
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: "=",
0x2F: "[", 0x30: "]", 0x31: "\\\"", 0x32: "~", 0x33: ";", 0x34: "'", 0x36: ",",
0x37: "." }

nums = []

keys = open('usbddata.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 line[22]!='\0':

    continue

    nums.append(int(line[6:8],16))

    # 00:00:xx:....

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)

```

結果如下：

```

root@kali:~/桌面/usb/USB/UsbKeyboardDataHacker# python pwn.py
output :nFLAGPR3550NWARD5A2FEE6E0

```

我們把前面的整合成腳本，得：

```

#!/usr/bin/env python

import sys
import os

DataFileName = "usb.dat"

presses = []

normalKeys = {"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":"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>"}

def main():
    # check argv
    if len(sys.argv) != 2:
        print "Usage : "
        print "      python UsbKeyboardHacker.py data.pcap"
        print "Tips : "
        print "      To use this python script , you must install the tshark
first."
        print "      You can use `sudo apt-get install tshark` to install it"
        print "Author : "
        print "      Angel_Kitty <angelkitty6698@gmail.com>"
        print "      If you have any questions , please contact me by email."
        print "      Thank you for using."
        exit(1)

    # get argv
    pcapFilePath = sys.argv[1]

    # get data of pcap
    os.system("tshark -r %s -T fields -e usb.capdata > %s" % (pcapFilePath,
```



```

DataFileName))

# read data
with open(DataFileName, "r") as f:
    for line in f:
        presses.append(line[0:-1])
# handle
result = ""
for press in presses:
    Bytes = press.split(":")
    if Bytes[0] == "00":
        if Bytes[2] != "00":
            result += normalKeys[Bytes[2]]
        elif Bytes[0] == "20": # shift key is pressed.
            if Bytes[2] != "00":
                result += shiftKeys[Bytes[2]]
        else:
            print "[-] Unknow Key : %s" % (Bytes[0])
print "[+] Found : %s" % (result)

# clean the temp data
os.system("rm ./%s" % (DataFileName))

if __name__ == "__main__":
    main()

```

效果如下：

```

root@kali:~/桌面/usb/USB/UsbKeyboardDataHacker# python UsbKeyboardDataHacker.py
example.pcap
Running as user "root" and group "root". This could be dangerous.
tshark: Lua: Error during loading:
[string "/usr/share/wireshark/init.lua"]:32: dofile has been disabled due to ru
nning Wireshark as superuser. See https://wiki.wireshark.org/CaptureSetup/Captur
ePrivileges for help in running Wireshark as an unprivileged user.
显示应用程序 Key : 01
[-] Unknow Key : 01
[+] Found : flag{pr355 0nwards a2fee6e0}

```

另外貼上一份鼠標流量數據包轉換腳本：

```

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
if btn_flag == 1 :
    print posx , posy
keys.close()

```

鍵盤流量數據包轉換腳本如下：

```

nums=[0x66,0x30,0x39,0x65,0x35,0x34,0x63,0x31,0x62,0x61,0x64,0x32,0x78,0x33,0x38,0x
6d,0x76,0x79,0x67,0x37,0x77,0x7a,0x6c,0x73,0x75,0x68,0x6b,0x69,0x6a,0x6e,0x6f,0x70]
s=''
for x in nums:
    s+=chr(x)
print s
mappings = { 0x41:"A", 0x42:"B", 0x43:"C", 0x44:"D", 0x45:"E", 0x46:"F",
0x47:"G", 0x48:"H", 0x49:"I", 0x4a:"J", 0x4b:"K", 0x4c:"L", 0x4d:"M",
0x4e:"N",0x4f:"O", 0x50:"P", 0x51:"Q", 0x52:"R", 0x53:"S", 0x54:"T",
0x55:"U",0x56:"V", 0x57:"W", 0x58:"X", 0x59:"Y", 0x5a:"Z", 0x60:"0", 0x61:"1",
0x62:"2", 0x63:"3", 0x64:"4", 0x65:"5", 0x66:"6", 0x67:"7", 0x68:"8", 0x69:"9",
0x6a:"*", 0x6b:"+", 0x6c:"separator", 0x6d:"- ", 0x6e:". ", 0x6f:"/" }
output = ""
for n in nums:
    if n == 0 :
        continue
    if n in mappings:
        output += mappings[n]
    else:
        output += '[unknown]'
print 'output :\n' + output

```

上面這個例子的項目鏈接如下：<https://files.cnblogs.com/files/ECJTUACM-873284962/UsbKeyboardDataHacker.rar>

那麼對於我們開篇提到的問題，我們可以模仿嘗試如上這個例子：

首先我們通過 tshark 將 usb.capdata 全部導出：

```
tshark -r task_AutoKey.pcapng -T fields -e usb.capdata //如果想導入 usbdata.txt 文件中，後面加上參數：>usbdata.txt
```

```
root@kali:~/桌面/usb/USB# tshark -r task_AutoKey.pcapng -T fields -e usb.capdata >usbdata.txt
Running as user "root" and group "root". This could be dangerous.
tshark: Lua: Error during loading:
[string "/usr/share/wireshark/init.lua"]:32: dofile has been disabled due to running Wireshark as superuser. See https://wiki.wireshark.org/CaptureSetup/CapturePrivileges for help in running Wireshark as an unprivileged user.
root@kali:~/桌面/usb/USB# ls
task_AutoKey.pcapng  usbdata.txt  UsbKeyboardDataHacker.py
usb.dat              UsbKeyboardDataHacker
```

我們用上面的 python 腳本將第三個字節取出來，對應對照表得到解碼：

```
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:"=", 0x2F:"[",
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 line[22]!='0':
        continue
    nums.append(int(line[6:8],16))
# 00:00:xx:....
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)
```

運行結果如下：

```
root@kali:~/桌面/usb/USB# python pwn.py
output :n[unknown]A[unknown]UTOKEY''.DECIPHER'[unknown]MPLRVFFCZEYOUJFJKYBXGZVDG
QAURKXZOLKOLVTUFBLRNJESQITWAHXNSIJXPNMPLSHCJBTYHZEALOGVIAAISSPLFHLFSWFEHJNCRWHTI
NSMAMBVEXO[DEL]PZE[DEL]IZ'
```

```
output :n[unknown]A[unknown]UTOKEY''.DECIPHER'[unknown]MPLRVFFCZEYOUJFJKYBXGZV
DGQAURKXZOLKOLVTUFBLRNJESQITWAHXNSIJXPNMPLSHCJBTYHZEALOGVIAAISSPLFHLFSWFE
HJNCRWHTINSMAMBVEXO[DEL]PZE[DEL]IZ'
```

我們可以看出這是自動密匙解碼，現在的問題是在我們不知道密鑰的情況下應該如何解碼呢？

我找到了如下這篇關於如何爆破密匙：

<http://www.practicalcryptography.com/cryptanalysis/stochastic-searching/cryptanalysis-autokey-cipher/>

爆破腳本如下：

```
from ngram_score import ngram_score
from pycipher import Autokey
import re
from itertools import permutations

qgram = ngram_score('quadgrams.txt')
trigram = ngram_score('trigrams.txt')

ctext =
'MPLRVFFCZEYOUJFJKYBXGZVDGQAURKXZOLKOLVTUFBLRNJESQITWAHXNSIJXPNMPLSHCJBTYHZEALOGVIA
AISSPLFHLFSWFEHJNCRWHTINSMAMBVEXPZIZ'

ctext = re.sub(r'^A-Z','',ctext.upper())
# keep a list of the N best things we have seen, discard anything else

class nbest(object):
    def __init__(self,N=1000):
        self.store = []
        self.N = N

    def add(self,item):
```

```

        self.store.append(item)
        self.store.sort(reverse=True)
        self.store = self.store[:self.N]

    def __getitem__(self,k):
        return self.store[k]

    def __len__(self):
        return len(self.store)

#init
N=100
for KLEN in range(3,20):
    rec = nbest(N)
    for i in permutations('ABCDEFGHIJKLMNOPQRSTUVWXYZ',3):
        key = ''.join(i) + 'A'*(KLEN-len(i))
        pt = Autokey(key).decipher(ctext)
        score = 0
        for j in range(0,len(ctext),KLEN):
            score += trigram.score(pt[j:j+3])
        rec.add((score,''.join(i),pt[:30]))

    next_rec = nbest(N)
    for i in range(0,KLEN-3):
        for k in xrange(N):
            for c in 'ABCDEFGHIJKLMNOPQRSTUVWXYZ':
                key = rec[k][1] + c
                fullkey = key + 'A'*(KLEN-len(key))
                pt = Autokey(fullkey).decipher(ctext)
                score = 0
                for j in range(0,len(ctext),KLEN):
                    score += qgram.score(pt[j:j+len(key)])
                next_rec.add((score,key,pt[:30]))
            rec = next_rec
        next_rec = nbest(N)
    bestkey = rec[0][1]
    pt = Autokey(bestkey).decipher(ctext)
    bestscore = qgram.score(pt)

```

```
for i in range(N):
    pt = Autokey(rec[i][1]).decipher(ctext)
    score = qgram.score(pt)
    if score > bestscore:
        bestkey = rec[i][1]
        bestscore = score
print bestscore,'autokey,
klen',KLEN,' ':''+bestkey+', ',Autokey(bestkey).decipher(ctext)
```

跑出來的結果如下：

```
root@kali:~/桌面/usb# python usbpwn.py
-824.697138698 autokey, klen 3 : "YCI", ONDDICXCERSFORFKKSWPDHRNTDERUVXRPRUGCAZ
ZLSOYMESWPEESTJAPAXANPPYDSEGJPSYKMCBCEUGWGCWMNPTUWMYATGHQHVBPMBSTBATZMIWSPHTTG
-772.470967688 autokey, klen 4 : "SYNR", URYABOHCYQRMWTOXOFNASUIDOWSRD0FILXFGAY00
FDXDIGHPICMHSFLGADYRPKOYWITENTAUUGEGRICPRSYTBARSEHUNOPLRTUCLYCFIKLNEQBOROWBIUD
-803.48764464 autokey, klen 5 : "BCGKY", LNFHXUSXSEHWXRYFOBKZBLUTHPPAYDIKONHGBHGN
ZAELAKEOFIJSMCPAWHILNQIDHUMBYMEVYGOHTIPUTHADYWEFENJORBRYVWBAYMXHNUADFOBEUKLHV
-761.616653993 autokey, klen 6 : "KIDAHF", CHIROADVRNKKOROWAKKJSDVTWHIRWRBSGUOXKD
NAREBOAJNPUTNNTITZVWEHUNUPOAIWHEKHRITHEZEAHTETOPEMDSRDSTBPSKKYVSBYDURILDSKGHOFH
-743.720273262 autokey, klen 7 : "KIDEAFY", CHINVAHASWLTUCFRONIDEUEPTIXQXGIGGOURF
NNORHUMAWQBJOHWERWEEBNTYRILKFOESTIOCLAISGSTXASQMAWOFPVTSARZSOUKRFIBUTIVVEABLPUEE
Z
-674.914569565 autokey, klen 8 : "FLAGHERE", HELLOBOYSANDGIRLSYOUARESOSMARTTHATYO
UCANFINDTHEFLAGTHATIHIDEINTHEKEYBOARDPACKAGEFLAGISJHAWLZKEWXHNCDSLWBAQJTUQZDXZQ
PF
```

我們看到了 flag 的字樣，整理可得如下：

```
-674.914569565 autokey, klen 8 : "FLAGHERE",
HELLOBOYSANDGIRLSYOUARESOSMARTTHATYOU CANFINDTHEFLAGTHATIHIDEINTHEKEYBOA
RDPACKAGEFLAGISJHAWLZKEWXHNCDSLWBAQJTUQZDXZQPF
```

我們把字段進行分割看：

```
HELLO
BOYS
AND
GIRLS
YOU
ARE
SO
SMART
THAT
YOU
CAN
FIND
THE
FLAG
```

```
THAT
IH
IDE
IN
THE
KEY
BOARD
PACKAGE
FLAG
IS
JHAWLZKEWXHNCDSLWBAQJTUQZDXZQPF
```

最後的 flag 就是 `flag{JHAWLZKEWXHNCDSLWBAQJTUQZDXZQPF}`

0x03 資源下載

本文涉及到的所有項目鏈接全部放在 Github 上：

- <https://github.com/AngelKitty/UsbKeyboardDataHacker>

0x04 擴展閱讀

- https://blog.csdn.net/songze_lee/article/details/77658094
- <https://wiki.wireshark.org/USB>
- http://www.usb.org/developers/hidpage/Hut1_12v2.pdf
- <https://www.wireshark.org/docs/man-pages/tshark.html>
- <http://www.practicalcryptography.com/cryptanalysis/stochastic-searching/cryptanalysis-autokey-cipher/>
- <https://hackfun.org/2017/02/22/CTF%E4%B8%AD%E9%82%A3%E4%BA%9B%E8%84%91%E6%B4%9E%E5%A4%A7%E5%BC%80%E7%9A%84%E7%BC%96%E7%A0%81%E5%92%8C%E5%8A%A0%E5%AF%86/>