

Wireshark例题-CTF

原创

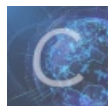
H3rmesk1t 于 2021-05-14 23:40:53 发布 1173 收藏 6

分类专栏: [安全学习](#) 文章标签: [wireshark ctf](#) [网络安全](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: <https://blog.csdn.net/LYJ20010728/article/details/116804672>

版权



[安全学习](#) 专栏收录该内容

21 篇文章 4 订阅

订阅专栏

Wireshark例题-CTF

搜索

文件提取

例题一

例题二

信息提取

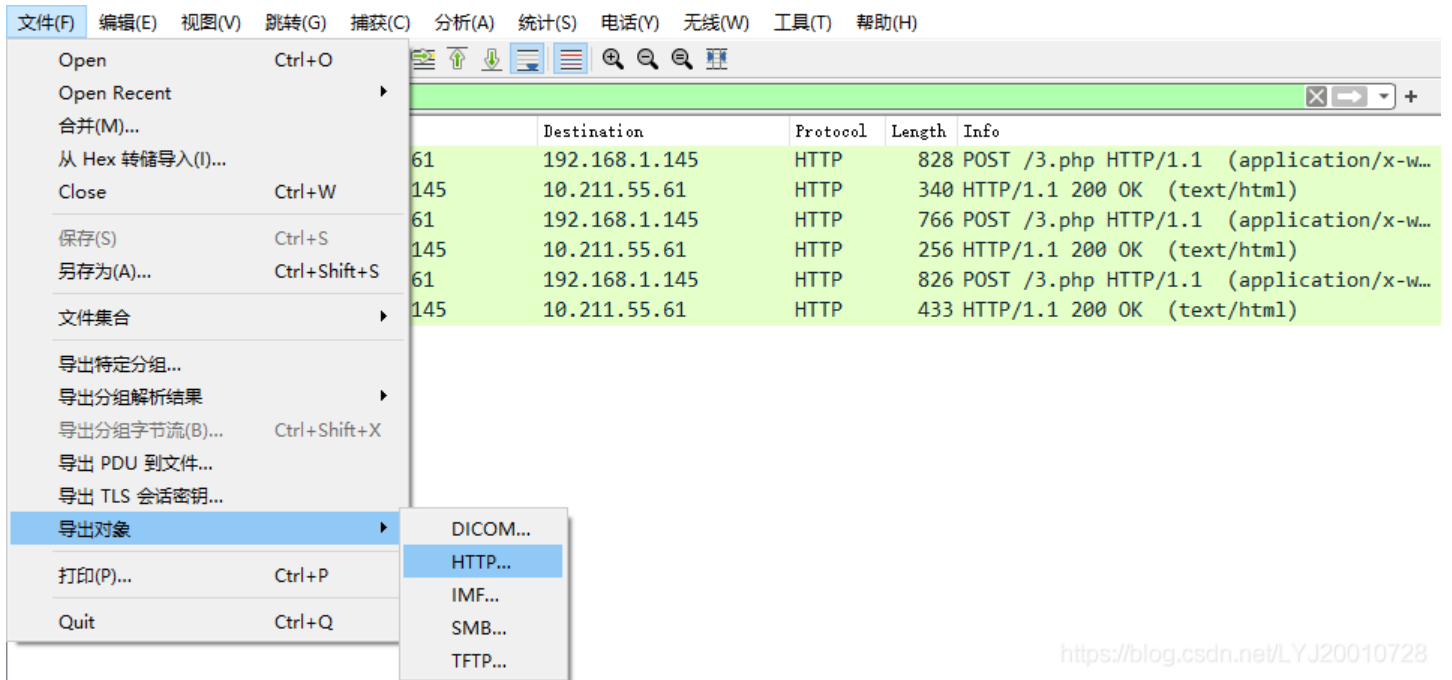
搜索

题目文件: [key.pcapng](#)

题目描述: flag被盗, 赶紧溯源!

题目题解:

- ①可以只将这个数据包当做文本文件打开, 比如用一些notepad++编辑器, 然后直接搜索
- ②用Wireshark自带的搜索功能找尝试查找一些关键词(比如key、flag、shell、pass等), 然后跟进可疑的数据包, 根据数据包特征, 很明显看出这是一个菜刀连接一句话木马的数据包, 然后往下找, 即可看到读取的flag



<https://blog.csdn.net/LYJ20010728>

No.	Time	Source	Destination	Protocol	Length	Info
5	0.184621	10.211.55.61	192.168.1.145	HTTP	828	POST /3.php HTTP/1.1 (application/x-w...
9	0.576743	192.168.1.145	10.211.55.61	HTTP	340	HTTP/1.1 200 OK (text/html)
18	21.139025	10.211.55.61	192.168.1.145	HTTP	766	POST /3.php HTTP/1.1 (application/x-w...
20	24.225688	192.168.1.145	10.211.55.61	HTTP	256	HTTP/1.1 200 OK (text/html)
30	48.763038	10.211.55.61	192.168.1.145	HTTP	826	POST /3.php HTTP/1.1 (application/x-w...
32	49.117671	192.168.1.145	10.211.55.61	HTTP	433	HTTP/1.1 200 OK (text/html)

这个时候就需要找到那个执行下载的数据包，找到数据传输的部分再导出，比如下面这个数据包大概是一个菜刀下载的过程，在最后一个包可以看到下载的文件，直接右键点击“导出分组字节流”，然后保存为.tar.gz文件

题目描述：抓到一只苍蝇！

题目题解：

首先用HTTP条件过滤一下；

右键第一个包，追踪流；

Wireshark · 追踪 HTTP 流 (tcp.stream eq 2) · misc_fly.pcapng

```
Accept-Language: zh-CN,zh;q=0.8
Cookie: ssuid=9979081647; ptui_loginuin=81101652; o_cookie=81101652; pgv_pvid=3703132940; newpt=2;
ptcz=4d9c0097882e7b9300db96ff8272c602d390465e21f8cc52d4a4719b7a73dfff; pt2gguin=o0081101652; uin=o0081101652; skey=@zT
p_skey=SMR2Xte-Sd3S2j-LIjK01POC1XsLYf6J5WR7DYTlx1s_; pt4_token=VqHcOafacF-cGmaNbpHVyg_; wimrefreshrun=0&; qm_flag=0;
sid=81101652&18c4549e039b41d8d5e73949a54d969a,qU01SM1h0Z51TZDNTWmotTElqS08xUE9DbFhzTF1mNko1V1I3RF1UbHgxc18.; qm_userna
qm_sid=18c4549e039b41d8d5e73949a54d969a,qU01SM1h0Z51TZDNTWmotTElqS08xUE9DbFhzTF1mNko1V1I3RF1UbHgxc18.; qm_domain=http:
qm_ptsk=81101652&@zTisQEDyk; foxacc=81101652&0; ssl_edition=mail.qq.com; edition=mail.qq.com; username=81101652&811016
new_mail_num=81101652&0; webp=1; ptisp=cn

{"path":"fly.rar","appid":"","size":525701,"md5":"e023afa4f6579db5becda8fe7861c2d3","sha":"ecccba7aea1d482684374b22e2e
```

可以看到一些基本信息，首先这是一个POST数据包，发送了一些文件相关信息，包括名称（fly.rar）和大小（525701）等。接下来应该就是文件实际上传的数据包，将过滤条件改为：http.request.method=="POST"；

http.request.method == "POST"

No.	Time	Source	Destination	Protocol	Length	Info
13	0.925023	192.168.1.101	14.17.42.24	HTTP	210	POST /cgi-bin/uploadunite?func=CreateF...
163	1.864990	192.168.1.101	59.37.116.102	HTTP	110	POST /ftn_handler/0b126a291df43b53f99c...
289	2.068360	192.168.1.101	59.37.116.102	HTTP	610	POST /ftn_handler/acbfc77208240d03e6af...
431	2.232611	192.168.1.101	59.37.116.102	HTTP	918	POST /ftn_handler/146b038670952f51f18d...
577	2.364839	192.168.1.101	59.37.116.102	HTTP	782	POST /ftn_handler/f6c7d6eef80795e03206...
729	3.102710	192.168.1.101	59.37.116.102	HTTP	391	POST /ftn_handler/1ffd8670a499bfb6e90c...
738	3.394152	192.168.1.101	14.17.42.24	HTTP	499	POST /cgi-bin/uploadunite?func=CheckFi...
767	5.751789	192.168.1.101	183.60.15.162	HTTP	867	POST /cgi-bin/getinvestigate?sid=x508Z...
781	6.103926	192.168.1.101	14.17.42.24	HTTP	801	POST /cgi-bin/compose_send?sid=x508ZuW...
1051	7.403270	192.168.1.101	183.60.15.162	HTTP	1042	POST /cgi-bin/getinvestigate?sid=x508Z...

从数据包的结构上看应该就是第二至第六个数据包是数据传输的过程。点开第二个可以看到MediaType的长度为131436=;

```
> [96 Reassembled TCP Segments (132317 bytes): #38(881), #39(1440), #41(1440), #42(1440), #43(1440), #44(1440), #46(1440)
> Hypertext Transfer Protocol
  ▾ Data (131436 bytes)
    Data: abcd9876000003ef00000000002015c01307cb0f705d906f7792b5023cb028ba442e6533...
    [Length: 131436]
```

第二到第五个都是一样的长度，第六个为1777，应该是剩余的最后一部分数据。但是 $131436*4+1777=527521!$ =525701，再看下第一个数据包;

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000030 C4 06 0B 3E 75 6C 14 21 2E F3 7E 7F 87 6F 7C 81 Ä..>ul.!.ó~.#o|.
00000040 5A F7 FC 04 63 DF 85 F9 21 A6 3E FB E4 B4 F1 4E Z÷ü.cb&...ú!|>ûä'ñN
00000050 E1 21 93 DE 81 CB 69 BE 1B 8B 4B 50 10 3F 68 AF á!"P.Ëi%.<KP.?h~
00000060 BD 39 58 A0 DF 19 BC 39 FB 52 F4 35 FD 4E 9D D0 %9X B.49úRó5ýN.Ð
00000070 A6 70 53 02 A2 5E F7 10 09 B8 73 48 94 7F 07 2E !pS.ç^÷...sH"...
00000080 8A 91 A7 48 8F B6 2B A9 2C C4 9C 4D 50 8E A4 87 Š'SH.¶+@,ÄMPŽH#
00000090 03 0D 7A 0D C0 6E A1 17 C4 20 41 FC 55 0A 1F 3B ..z.Äñ;.Ä ÄüU..;
000000A0 C5 D2 8D 63 20 65 59 FA F2 A3 BA A7 B9 FC 88 50 ÄÖ.c eYúó£°S'ü^P
000000B0 C1 2A 49 86 A7 B2 81 92 D0 78 56 DE 86 AB 51 F3 Ä*I†S°. 'ÐxV†«Qó
000000C0 1E 23 F9 E9 C8 3D 26 26 E0 9F 38 0B F9 6E 60 DD .#ùéÈ=ççãÿ8.ùn`Y
000000D0 F2 39 2A 84 43 C7 88 DF D2 C8 CC 76 63 28 89 04 ò9*„CÇ^BÖËivç(%
000000E0 17 ED 1D E7 AA AA 57 7B CB 9D 1C FB FA 1E FF 76 .i.ç*^W(È..úú.ÿv
000000F0 39 AC E4 4B BB 32 B8 B0 DB D9 A3 5B 06 07 47 FD 9-ãK»2,°ÜÜ£[.Gý
00000100 5D 10 79 C0 CA 5C 9D 37 1F E3 50 B8 F4 3C 7B 7A ].yÄË\..7.ãP,ó<{z
00000110 8E 07 FA 1F 80 03 32 68 69 76 8E FA 1E 77 02 1E Ž.ú.€.2hivŽú.w..
00000120 2D 88 11 F0 FC 24 D7 13 3D 48 F3 30 1B A1 57 40 -^..òú$*. =Hó0. ;W@
00000130 3B 8B BA B9 12 64 C5 C1 E3 E4 60 07 CB 4D 3E 8A ;<°².dÄÄää`.ÈM>Š
00000140 57 35 00 14 EC CC BA 7A EA 1D 48 26 84 37 4B 22 W5..iï°zè.Hä„7K"
00000150 E2 E7 AB AD 2B A8 67 49 00 08 05 85 00 00 00 00 äç«. +`gI.....
00000160 00 02 00 00 00 00 00 00 00 00 00 00 52 61 72 21 .....Rar!
```

都知道rar文件头应该是Rar，但是选中的数据部分前面却多出了很多，简单计算下一共多出了364，且 $364*5+525701=527521$ ，所以多出的也许是某种校验数据，在导出的时候将其忽略；
每个包都做同样的操作即可得出5个文件，再将这个文件按顺序拼接即可。拼接的话可以使用16进制编辑器手动拼接，也可以使用linux下cat命令，比如“cat 1 2 3 4 5 > fly.rar”；'这道题还设置了伪加密，需要修改加密位，将0x84位置改为0x80即可；

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000 52 61 72 21 1A 07 00 CF 90 73 00 00 0D 00 00 00 Rar!...i.s.....
00000010 00 00 00 00 F9 81 74 80 90 2D 00 3D 05 08 00 86 ....ù.t.[-.=...t
00000020 22 0F 00 02 25 2D ED F1 09 5C 59 46 1D 35 08 00 "...%-iñ.\YF.5..
00000030 20 00 00 00 66 6C 61 67 2E 74 78 74 00 F0 79 03 ...flag.txt.ðy.
00000040 4C 18 1E 15 15 0C 89 15 DC 16 1D EF A3 72 4B 90 L.....%.Ü..i£rK.
00000050 B0 90 08 24 3A 52 23 05 22 02 C8 41 C4 84 82 40 °..$:R#.".ÈÄÄ„,@
00000060 9D 04 3A EC 24 87 44 3A 58 A1 18 08 81 92 6F 60 ...i$+D:Xj...`o`
```

解压出来后是一个exe可执行文件，里面隐藏了一个png图片，是个二维码，扫描即可得到flag



已解码数据 1:

位置:(455.6,420.5)-(716.7,420.6)-(455.6,681.5)-(716.7,681.6)
颜色正常,正像
版本:3
纠错等级:H,掩码:3
内容:
flag{m1Sc_oxO2_Fly}

解码完成

信息提取

题目文件: sqlmap.pcap

题目描述: 数据库中的flag被偷走了, 好在全过程我们都有记录

题目题解:

数据包记录的是sqlmap获取flag的过程, 使用http && http contains"flag"过滤一下

```

No.    Time           Source            Destination       Protocol  Length  Info
-----
776    11.967929      10.0.0.101       10.0.0.201       HTTP      552     GET /message.php?id=1%20AND%20ORD...

> Transmission Control Protocol, Src Port: 42902, Dst Port: 80, Seq: 1, Ack: 1, Len: 521
  Hypertext Transfer Protocol
    GET /message.php?id=-3413%20UNION%20ALL%20SELECT%20NULL%2CCONCAT%280x7173636371%2CIFNULL%28CAST%28COUNT%28%60v...

0040  b9 f4 47 45 54 20 2f 6d 65 73 73 61 67 65 2e 70  -- GET /m message.p
0050  68 70 3f 69 64 3d 2d 33 34 31 33 25 32 30 55 4e  hp?id=-3 413%20UN
0060  49 4f 4e 25 32 30 41 4c 4c 25 32 30 53 45 4c 45  ION%20AL L%20SELE
0070  43 54 25 32 30 4e 55 4c 4c 25 32 43 43 4f 4e 43  CT%20NUL L%2CCONC
0080  41 54 25 32 38 30 78 37 31 37 33 36 33 36 33 37  AT%280x7 17363637
0090  31 25 32 43 49 46 4e 55 4c 4c 25 32 38 43 41 53  1%2CIFNU LL%28CAS
00a0  54 25 32 38 43 4f 55 4e 54 25 32 38 25 36 30 76  T%28COUN T%28%60v
00b0  61 6c 75 65 25 36 30 25 32 39 25 32 30 41 53 25  alue%60% 29%20AS%
00c0  32 30 43 48 41 52 25 32 39 25 32 43 30 78 32 30  20CHAR%2 9%2C0x20
00d0  25 32 39 25 32 43 30 78 37 31 36 66 37 35 37 33  %29%2C0x 716f7573
00e0  37 31 25 32 39 25 32 30 46 52 4f 4d 25 32 30 69  71%29%20 FROM%20i
00f0  73 67 2e 66 6c 61 67 73 25 32 33 20 48 54 54 50  sg.flags %23 HTTP
0100  2f 31 2e 31 0d 0a 41 63 63 65 70 74 2d 4c 61 6e  /1.1 -Ac cept-Lan

```

<https://blog.csdn.net/LYJ20010728>

将其payload解码一下是这样的，判断其ascii码是否大于64

id=1 AND ORD(MID((SELECTIFNULL(CAST(value AS CHAR),0x20) FROM isg.flags ORDER BY value LIMIT0,1),1,1))>64

然后一直到836个包判断第一位ascii码值大于72，然后开始从高到低递减，判断其ascii码不大于73，则第一位的ascii码值是73，对应的字符为l。以此类推，其flag为ISG{Blind_SQL_InJecTi0N_DeTEcTEd}。本题需要一定的耐心和SQL注入基础。但是这么做可能有些繁琐，其实pcap数据包可以直接用文本编辑器打开，就可以看到其中的http请求

```

3815  .枯-DCI兄等ET
      /message.php?id=-3413%20UNION%20ALL%20SELECT%20NULL%2CCONCAT%280x7173636371%2CIFNULL%28CAST%28COUNT
      %28%60value%60%29%20AS%20CHAR%29%2C0x20%29%2C0x716f757371%29%20FROM%20isg.flags%23 HTTP/1.1
3816  Accept-Language:
3817  Accept-Encoding:
3818  Host: 10.0.0.201
3819  Accept: text/html,
3820  User-Agent: sqlmap
3821  Accept-Charset: IS
3822  Connection: close
3823  Pragma: no-cache

```

所以可以使用字符串搜索的方式直接去查找其中的语句，然后判断flag，首先将原数据包中的http请求导出来，另存为sqli.pcap

```

OPEN FILES
x flag.py
flag.py
1  import re
2  import urllib
3
4  flag = ""
5  key = 1
6  pcap = open("sqli.pcap","rb")
7  lines = pcap.readlines()
8  for line in lines:
9      line = line.strip("\n")
10     line = urllib.unquote(line)
11     sql = re.search(r".*LIMIT 0,1\),(\d+),1\)\>*(\d*)",line)
12     response = re.search(r"Content-Length: (\d*)",line)
13     if response:
14         Content_Length = response.groups()[0]
15     if sql:
16         count,asc = sql.groups()
17         if str(key) == count:
18             asc_2 = asc
19         else:
20             if int(Content_Length)>150:
21                 flag += chr(int(asc)+1)
22             else:
23                 flag += chr(int(asc_2))
24         key += 1
25 print flag
26
ISG{Blind_SQL_InJecTi0N_DeTEcTEd}
[Finished in 0.4s]

```

<https://blog.csdn.net/LYJ20010728>