IDF实验室-抓到一只苍蝇

原创

pcz_x 于 2015-05-30 23:40:18 发布 11094 收藏 3

分类专栏： ctf 文章标签： idf ctf writeup

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/ab748998806/article/details/46279849
版权

ctf 专栏收录该内容
7 篇文章 0 订阅
订阅专栏

感觉这道题对我已经有难度了

一、原题

报告首长！发现一只苍蝇。。

在哪？

here!

卧槽？！好大一坨苍蝇。。

文件地址： http://pan.baidu.com/s/1bnGWbqJ

提取码：oe6w

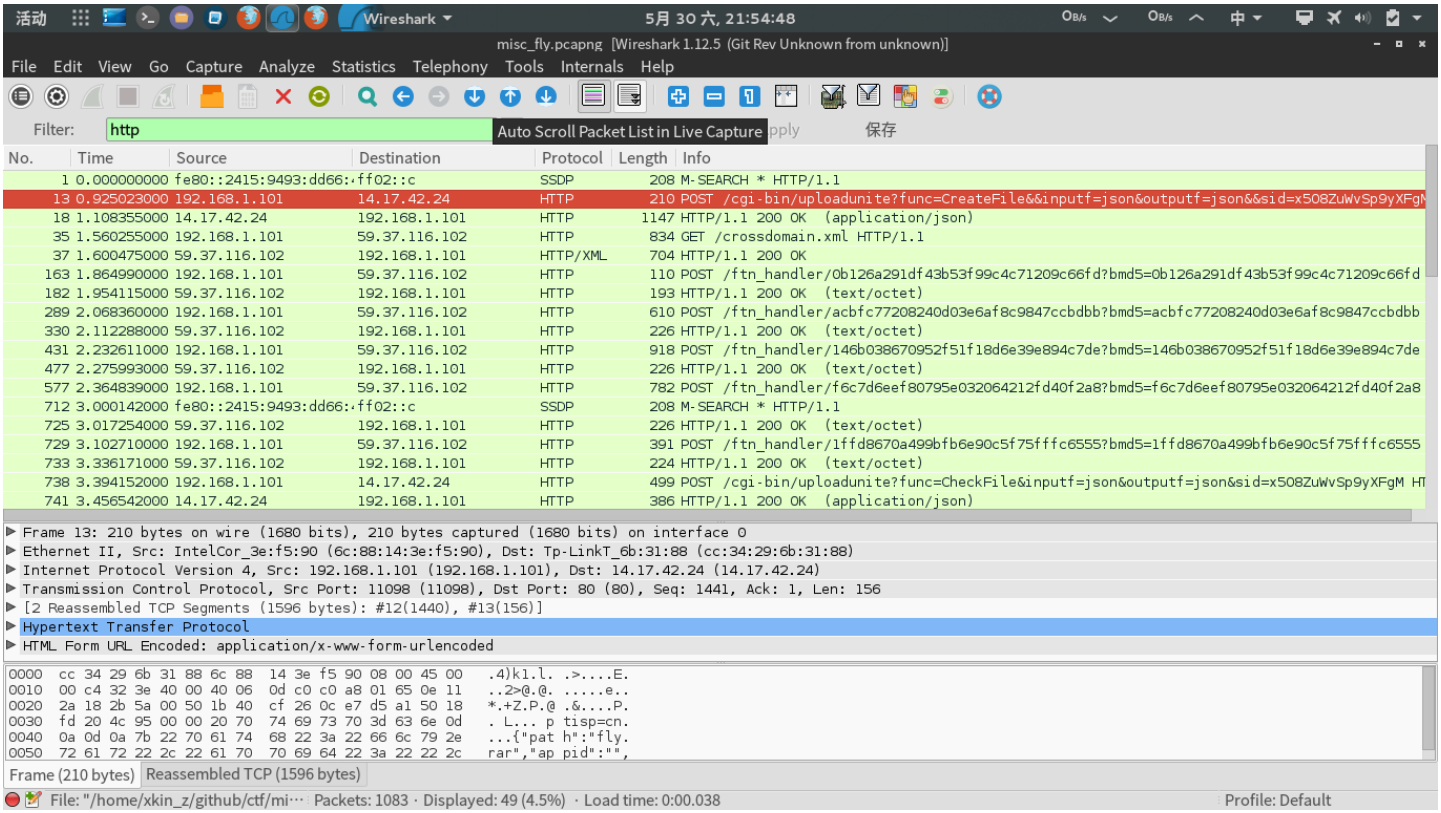
PS：flag写错了，太麻烦也懒得改了，格式还是wctf{...}，大家明白就好，不要在意这些细节。。

二、writeup

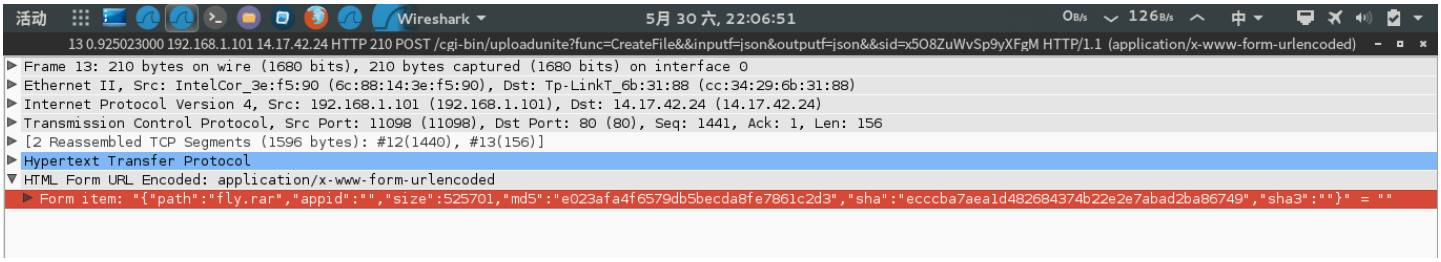
首先下载所给的文件misc_fly.pcapng，是一个抓包软件抓取的数据包，解题流程如下：

1.用wireshark分析数据包

打开数据包，发现是一堆TCP的包，先不理会底层的数据，只关心应用层。
应用层的协议只有HTTP，过滤出HTTP的包进一步分析：



请求都发往http://sz.mail.ftn.qq.com，猜测是qq邮箱的地址。
同时观察第一个请求



POST了一段这样的JSON：

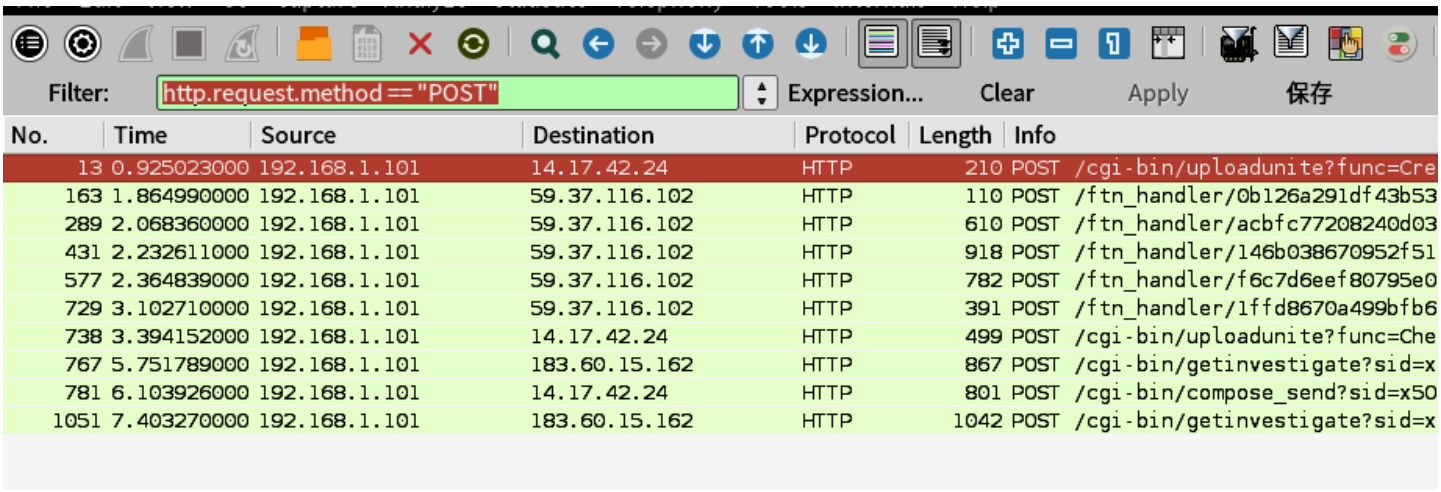
```
{
    "path":"fly.rar",
    "appid":"",
    "size":525701,
    "md5":"e023afa4f6579db5becda8fe7861c2d3",
    "sha":"ecccba7aea1d482684374b22e2e7abad2ba86749",
    "sha3":""
}
```

目标url为：

```
http://set2.mail.qq.com/cgi-bin/uploadunite?func=CreateFile&&inputf=json&outputf=json&&sid=x5O8ZuWvSp9y
```

这应该是一个上传文件的操作，文件名为`fly.rar`，文件大小为`525701`。
进一步分析以验证猜想，过滤出全部的POST数据包：

```
http.request.method == "POST"
```



Filter: http.request.method == "POST" Expression... Clear Apply 保存

No.	Time	Source	Destination	Protocol	Length	Info
13	0.925023000	192.168.1.101	14.17.42.24	HTTP	210	POST /cgi-bin/uploadunite?func=Cre
163	1.864990000	192.168.1.101	59.37.116.102	HTTP	110	POST /ftn_handler/0b126a291df43b53
289	2.068360000	192.168.1.101	59.37.116.102	HTTP	610	POST /ftn_handler/acbfc77208240d03
431	2.232611000	192.168.1.101	59.37.116.102	HTTP	918	POST /ftn_handler/146b038670952f51
577	2.364839000	192.168.1.101	59.37.116.102	HTTP	782	POST /ftn_handler/f6c7d6eef80795e0
729	3.102710000	192.168.1.101	59.37.116.102	HTTP	391	POST /ftn_handler/1ffd8670a499bfb6
738	3.394152000	192.168.1.101	14.17.42.24	HTTP	499	POST /cgi-bin/uploadunite?func=Che
767	5.751789000	192.168.1.101	183.60.15.162	HTTP	867	POST /cgi-bin/getinvestigate?sid=x
781	6.103926000	192.168.1.101	14.17.42.24	HTTP	801	POST /cgi-bin/compose_send?sid=x50
1051	7.403270000	192.168.1.101	183.60.15.162	HTTP	1042	POST /cgi-bin/getinvestigate?sid=x

其中倒数第二个包的内容是：

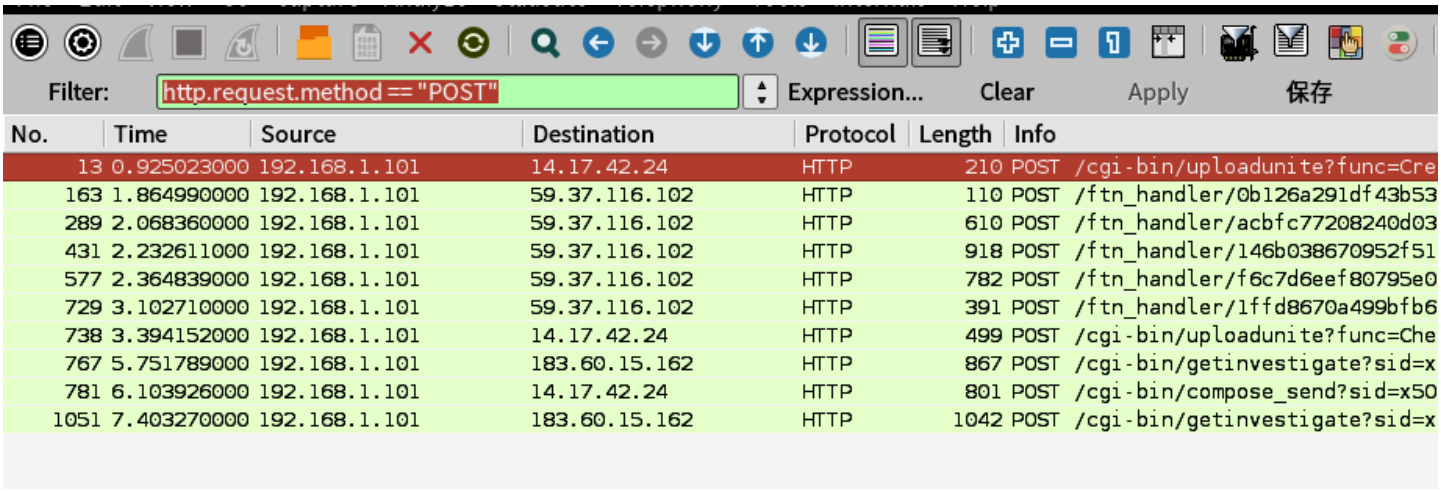
```
▼ Form item: "to" = ""king"<king@woldy.net>"
    Key: to
    Value: "king"<king@woldy.net>
▼ Form item: "subject" = "报告首长，发现一只苍蝇！"
    Key: subject
    Value: \346\212\245\345\221\212\351\246\226\351\225\277
▼ Form item: "content__html" = "<div>RT！</div>"
    Key: content__html
    Value: <div>RT\357\274\201</div>
▼ Form item: "sendmailname" = "81101652@qq.com"
    Key: sendmailname
    Value: 81101652@qq.com
▼ Form item: "savesendbox" = "1"
    Key: savesendbox
    Value: 1
▼ Form item: "actiontype" = "send"
    Key: actiontype
    Value: send
▼ Form item: "sendname" = "太虚邪灵"
    Key: sendname
    Value: \345\244\252\350\231\232\351\202\252\347\201\265
▼ Form item: "acctid" = "0 "
    Key: acctid
    Value: 0
▼ Form item: "separatedcopy" = "false"
    Key: separatedcopy
    Value: false
▼ Form item: "attachlist_log" = "fly.rar%C2%A0(513.4K)%2C1%
    Key: attachlist_log
    Value: fly.rar%C2%A0(513.4K)%2C1%2C0%7Cfly.rar%2Ccomple
```

至此已经可以确定：

```
数据包的内容：一封带附件的邮件
发件人：81101652@qq.com
收件人：king@woldy.net
附件：fly.rar
附件大小：525701 Bytes
```

接下来寻找附件数据在哪里。

2.寻找附件数据

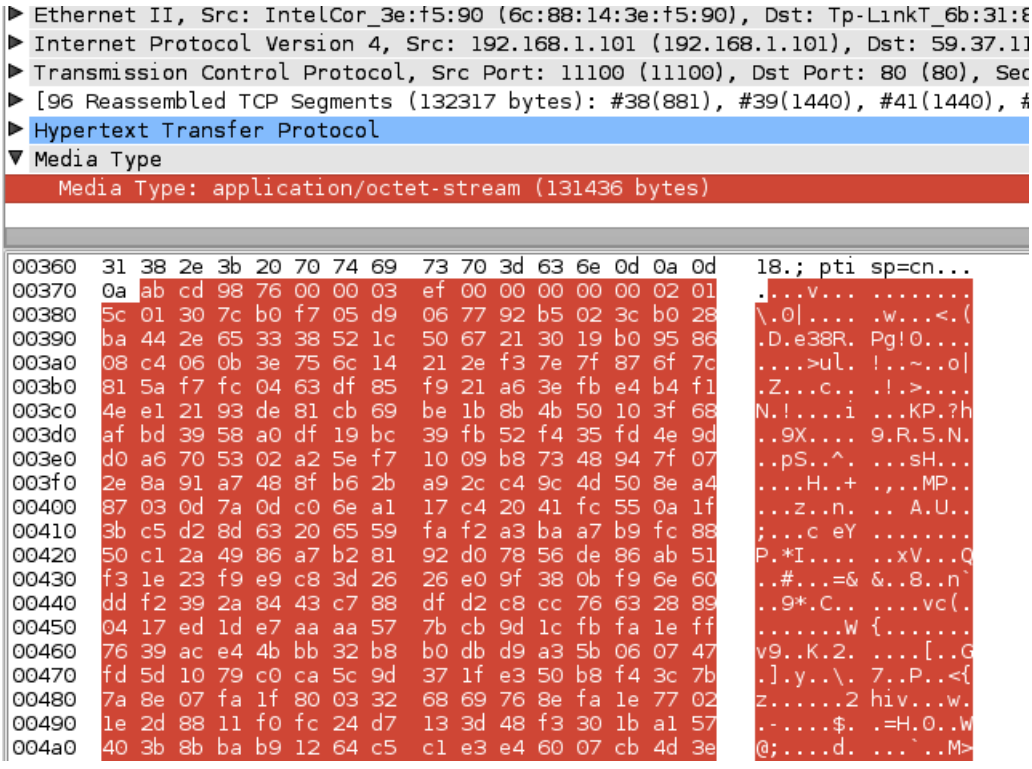


No.	Time	Source	Destination	Protocol	Length	Info
13	0.925023000	192.168.1.101	14.17.42.24	HTTP	210	POST /cgi-bin/uploadunite?func=Cre
163	1.864990000	192.168.1.101	59.37.116.102	HTTP	110	POST /ftn_handler/0b126a291df43b53
289	2.068360000	192.168.1.101	59.37.116.102	HTTP	610	POST /ftn_handler/acbfc77208240d03
431	2.232611000	192.168.1.101	59.37.116.102	HTTP	918	POST /ftn_handler/146b038670952f51
577	2.364839000	192.168.1.101	59.37.116.102	HTTP	782	POST /ftn_handler/f6c7d6eef80795e0
729	3.102710000	192.168.1.101	59.37.116.102	HTTP	391	POST /ftn_handler/1ffd8670a499bfb6
738	3.394152000	192.168.1.101	14.17.42.24	HTTP	499	POST /cgi-bin/uploadunite?func=Che
767	5.751789000	192.168.1.101	183.60.15.162	HTTP	867	POST /cgi-bin/getinvestigate?sid=x
781	6.103926000	192.168.1.101	14.17.42.24	HTTP	801	POST /cgi-bin/compose_send?sid=x50
1051	7.403270000	192.168.1.101	183.60.15.162	HTTP	1042	POST /cgi-bin/getinvestigate?sid=x

第一个请求向服务器POST附件信息，紧接着就应该是上传，结合数据包推断第2～6个，共5个数据包应为附件数据。

5个数据包中的`Media Type域`的大小各为

```
131436
131436
131436
131436
1777
```



合计`527521`，而前面附件信息里已知附件大小为`525701`，相差不多，多出来的部分应该是头部的信息之类。

3.还原附件数据

观察5个包`Media Type域`的内容，前面很大一部分内容是相同的，那么这一部分是通信时所需的头部的内容，不是附件本身的内容，通过计算将多余的数据去除。

> 已知：
> 附件被分成5个部分
> 5个子部分合计大小为`527521`
> 附件原大小为`525701`
> 求：
> 每个子部分头部多余的数据

容易求出，头部多余的部分：

```
                                                                    
```

将5个数据包的`Media Type域`分别导出为二进制文件：

然后使用`dd`命令分别将其前364个字节去除：

```
dd if=1 bs=1 skip=364 of=1.1
dd if=2 bs=1 skip=364 of=2.1
dd if=3 bs=1 skip=364 of=3.1
dd if=4 bs=1 skip=364 of=4.1
dd if=5 bs=1 skip=364 of=5.1
```

之后合并文件：

```
cat  1.1  2.1  3.1  4.1  5.1  >  fly.rar
```

校验md5值：

```
md5sum fly.rar
```

结果：

```
e023afa4f6579db5becda8fe7861c2d3
```

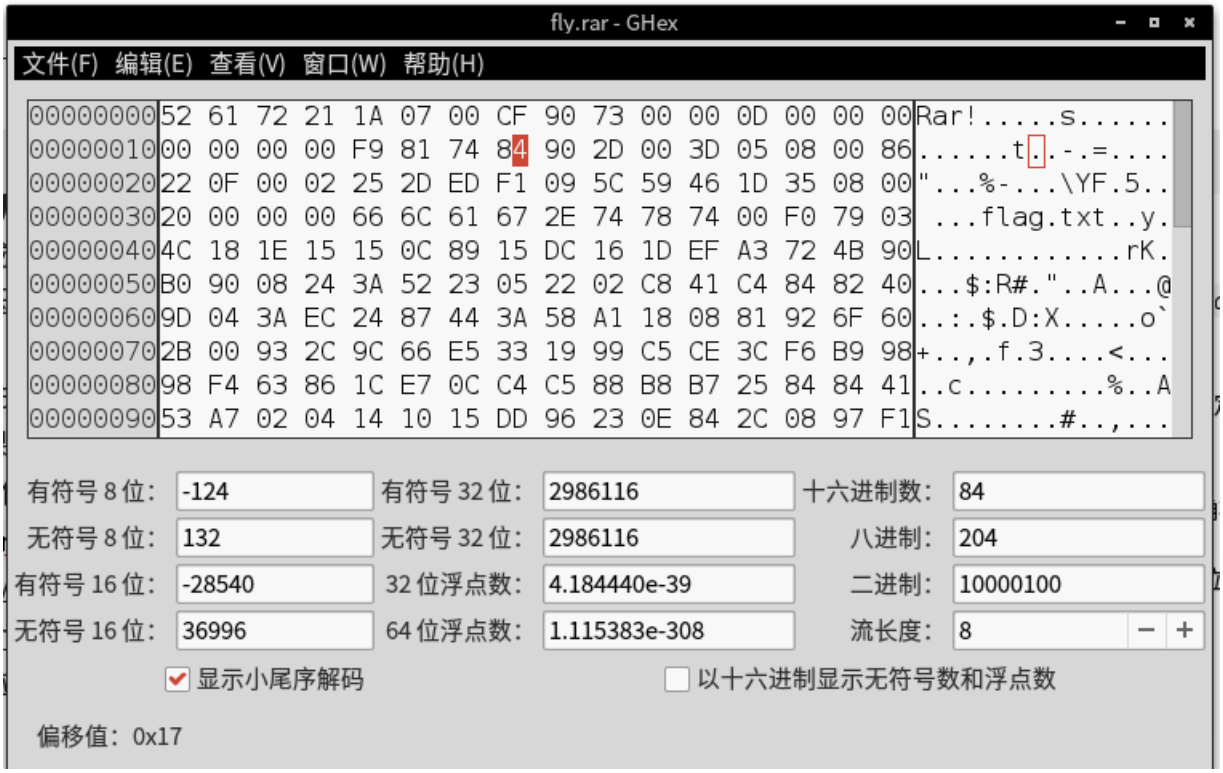
而第一步中得到的数据包中POST数据为：

```
{
    "path":"fly.rar",
    "appid":"",
    "size":525701,
    "md5":"e023afa4f6579db5becda8fe7861c2d3",
    "sha":"ecccba7aea1d482684374b22e2e7abad2ba86749",
    "sha3":""
}
```

二者一致。同理再校验sha值，同样一致。
至此，由网络包还原出了附件数据`fly.rar`：

4.处理附件数据

本以为到这里已经大功告成，解压`fly.rar`即可。
结果还有后招……竟然解压失败……
分析原因，
`fly.rar`文件通过了md5和sha校验，肯定是没问题的，自己的思路到这里断了，怎么想也没结果。
无奈搜答案，得出结果——伪加密。
即这是一个未加密过的`rar`文件，但是却将加密位置为了1，具体可参考
[rar文件格式描述]
只需将文件开头处`0x74`位后面的`0x84`位置改为`0x80`即可



修改后顺利解压，得到`flag.txt`。

5.处理`flag.txt`

这回答案该有了吧，打开`flag.txt`查看，这又是一个二进制文件。
无语。
继续二进制打开。
发现这是个`win32`的程序，`Linux`跑不了，转到`windows查看`。
是个满屏幕跑苍蝇的程序……还挺逼真的……
还是二进制打开分析。
文件内搜`PNG`、`Rar`、`JFIF`，文件尾有一个`PNG`，提取出来，是个二维码：

扫之，得到结果
`flag{m1Sc_oxO2_Fly}`

三、flag

`wctf{m1Sc_oxO2_Fly}`

四、知识点

这题够绕的……
虽然参考了部分答案，但做完感觉很好。