2019工业信息安全技能大赛个人线上赛第一场(前5道)writeup

小蛤蟆99 / 2019-08-15 07:29:00 / 浏览数 4626 安全技术 CTF 顶(1) 踩(0)

1.1.Modbus协议分析

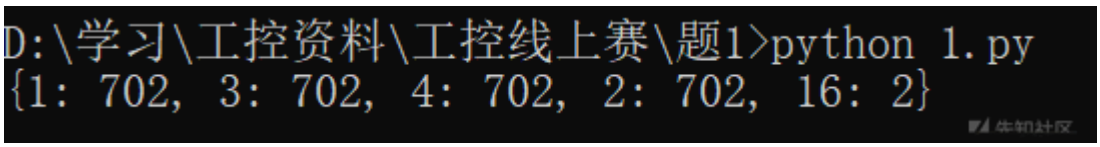
题目：黑客通过外网进入一家工厂的控制网络，之后对工控网络中的操作员站系统进行了攻击，最终通过工控协议破坏了正常的业务。我们得到了操作员站在攻击前后的网络
题目附件连接：https://pan.baidu.com/s/1jGu7-1EKc29HTQc-pCJZlw （提取码：8kqx）
解题步骤：

首先打开流量包，数据包都是关于Modbus/TCP的流量。

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.16.3.23	172.16.1.33	Modbus/TCP	66	Query: Trans: 3827; Unit: 1, Func: 1: Read Coils
2	0.001208	172.16.1.33	172.16.3.23	Modbus/TCP	64	Response: Trans: 3827; Unit: 1, Func: 1: Read Coils
3	0.001467	172.16.3.23	172.16.1.33	Modbus/TCP	66	Query: Trans: 4083; Unit: 1, Func: 3: Read Holding Registers
4	0.008287	172.16.1.33	172.16.3.23	Modbus/TCP	75	Response: Trans: 4083; Unit: 1, Func: 3: Read Holding Registers
5	0.008505	172.16.3.23	172.16.1.33	Modbus/TCP	66	Query: Trans: 4339; Unit: 1, Func: 4: Read Input Registers
6	0.016249	172.16.1.33	172.16.3.23	Modbus/TCP	75	Response: Trans: 4339; Unit: 1, Func: 4: Read Input Registers
7	0.016460	172.16.3.23	172.16.1.33	Modbus/TCP	66	Query: Trans: 4595; Unit: 1, Func: 2: Read Discrete Inputs
8	0.023129	172.16.1.33	172.16.3.23	Modbus/TCP	64	Response: Trans: 4595; Unit: 1, Func: 2: Read Discrete Inputs
9	0.122930	fe:54:00:f8:5c:21	Spanning-tree-(for-…	STP	60	Conf. Root = 32768/0/52:54:00:01:a6:a5 Cost = 0 Port = 0x8005
10	0.203113	172.16.3.23	172.16.1.33	TCP	54	1048 → 502 [ACK] Seq=49 Ack=63 Win=63796 Len=0
11	1.000154	172.16.3.23	172.16.1.33	Modbus/TCP	66	Query: Trans: 4851; Unit: 1, Func: 1: Read Coils
12	1.001309	172.16.1.33	172.16.3.23	Modbus/TCP	64	Response: Trans: 4851; Unit: 1, Func: 1: Read Coils
13	1.001590	172.16.3.23	172.16.1.33	Modbus/TCP	66	Query: Trans: 5107; Unit: 1, Func: 3: Read Holding Registers
14	1.007712	172.16.1.33	172.16.3.23	Modbus/TCP	75	Response: Trans: 5107; Unit: 1, Func: 3: Read Holding Registers
15	1.007987	172.16.3.23	172.16.1.33	Modbus/TCP	66	Query: Trans: 5363; Unit: 1, Func: 4: Read Input Registers
16	1.013149	172.16.1.33	172.16.3.23	Modbus/TCP	75	Response: Trans: 5363; Unit: 1, Func: 4: Read Input Registers
17	1.013411	172.16.3.23	172.16.1.33	Modbus/TCP	66	Query: Trans: 5619; Unit: 1, Func: 2: Read Discrete Inputs
18	1.014492	172.16.1.33	172.16.3.23	Modbus/TCP	64	Response: Trans: 5619; Unit: 1, Func: 2: Read Discrete Inputs

运行脚本，分析流量包中Modbus/TCP的协议功能码，脚本和运行结果如下：

```
import pyshark
def get_code():
    captures = pyshark.FileCapture("question_1564353677_modbus1.pcap")
    func_codes = {}
    for c in captures:
        for pkt in c:
            if pkt.layer_name == "modbus":
                func_code = int(pkt.func_code)
                if func_code in func_codes:
                    func_codes[func_code] += 1
                else:
                    func_codes[func_code] = 1
    print(func_codes)
if __name__ == '__main__':
get_code()
```



modbus常见功能码分析，分析结果我们可以知道1（读取线圈状态），3（读多个寄存器），4（读输入寄存器），2（读取输入内容），四个功能码都出现了702次，唯

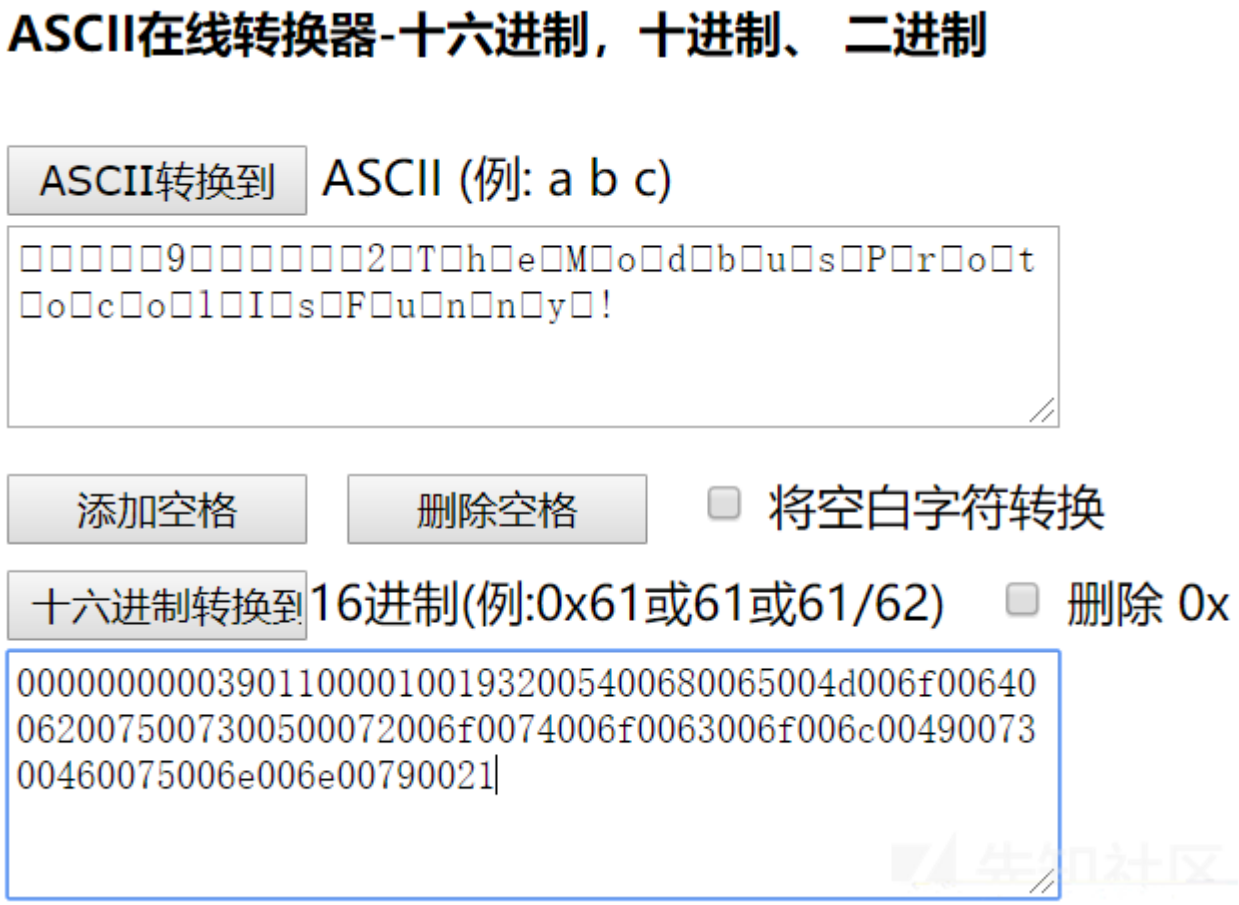
```
import pyshark
def find_flag():
    cap = pyshark.FileCapture("question_1564353677_modbus1.pcap")
    idx = 1
    for c in cap:
        for pkt in c:
            if pkt.layer_name == "modbus":
                func_code = int(pkt.func_code)
                if func_code == 16:
                    payload = str(c["TCP"].payload).replace(":", "")
                    print(hex_to_ascii(payload))
                    print("{0} *".format(idx))
        idx += 1
def hex_to_ascii(payload):
data = payload
```

```
flags = []
for d in data:
    _ord = ord(d)
    if (_ord > 0) and (_ord < 128):
        flags.append(chr(_ord))
return ''.join(flags)
if __name__ == '__main__':
find_flag()
```

```
D:\学习\工控资料\工控线上赛\题1>python 2.py
00000000003901100001001932005400680065004d006f0064006200750073005000720006f0074006f0063006f006c00490073004600750006e006e00790021
7506 #
000000000003019002
7508 #
```

提出的数据存在一个16进制字符串

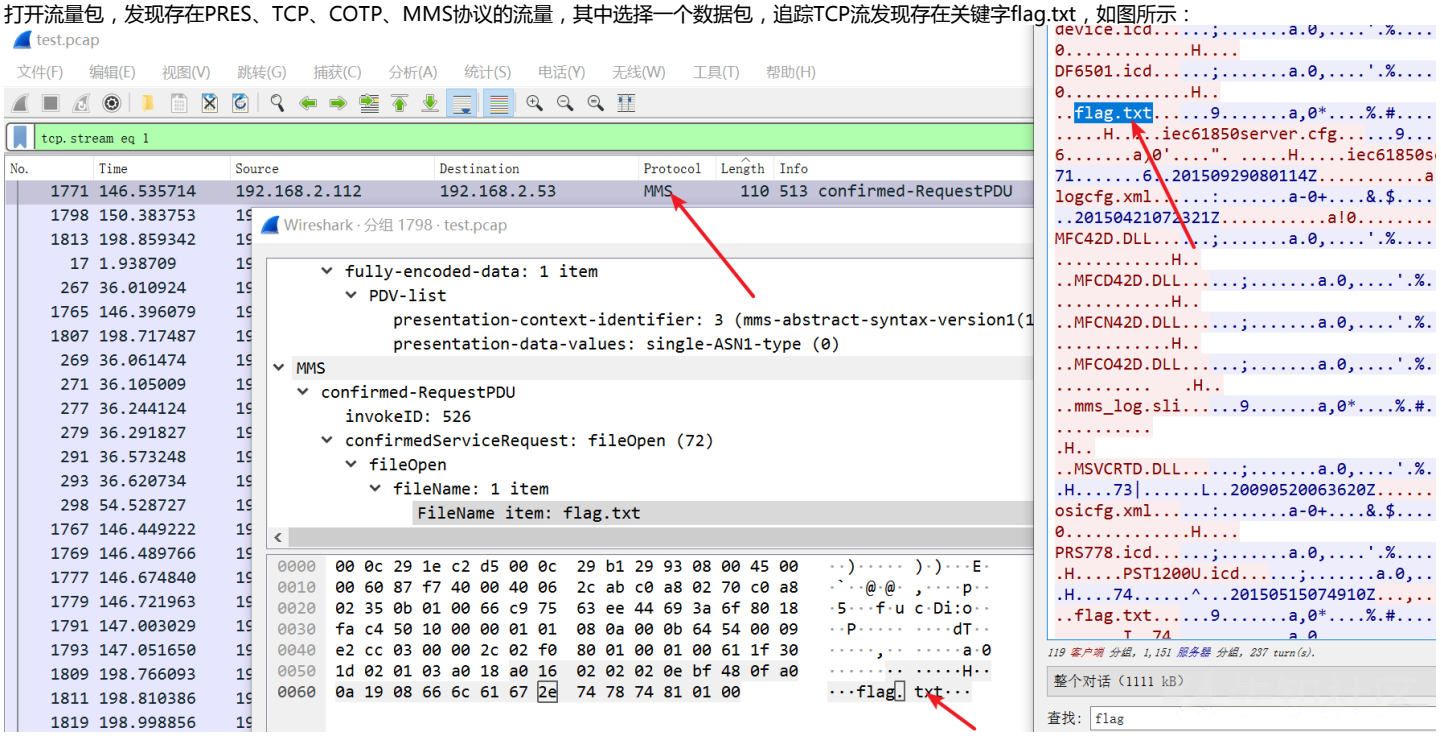
```
00000000003901100001001932005400680065004d006f0064006200750073005000720006f0074006f0063006f006c00490073004600750006e006e00790
```



1.2.工业协议分析1

题目：工业网络中存在异常，尝试通过分析PCAP流量包，分析出流量数据中的异常点，并拿到FLAG。
题目附件连接：https://pan.baidu.com/s/17jkHLBqcjxP0o9FpGIfObA （提取码：95ds ）
解题步骤：

打开流量包，发现存在PRES、TCP、COTP、MMS协议的流量，其中选择一个数据包，追踪TCP流发现存在关键字flag.txt，如图所示：



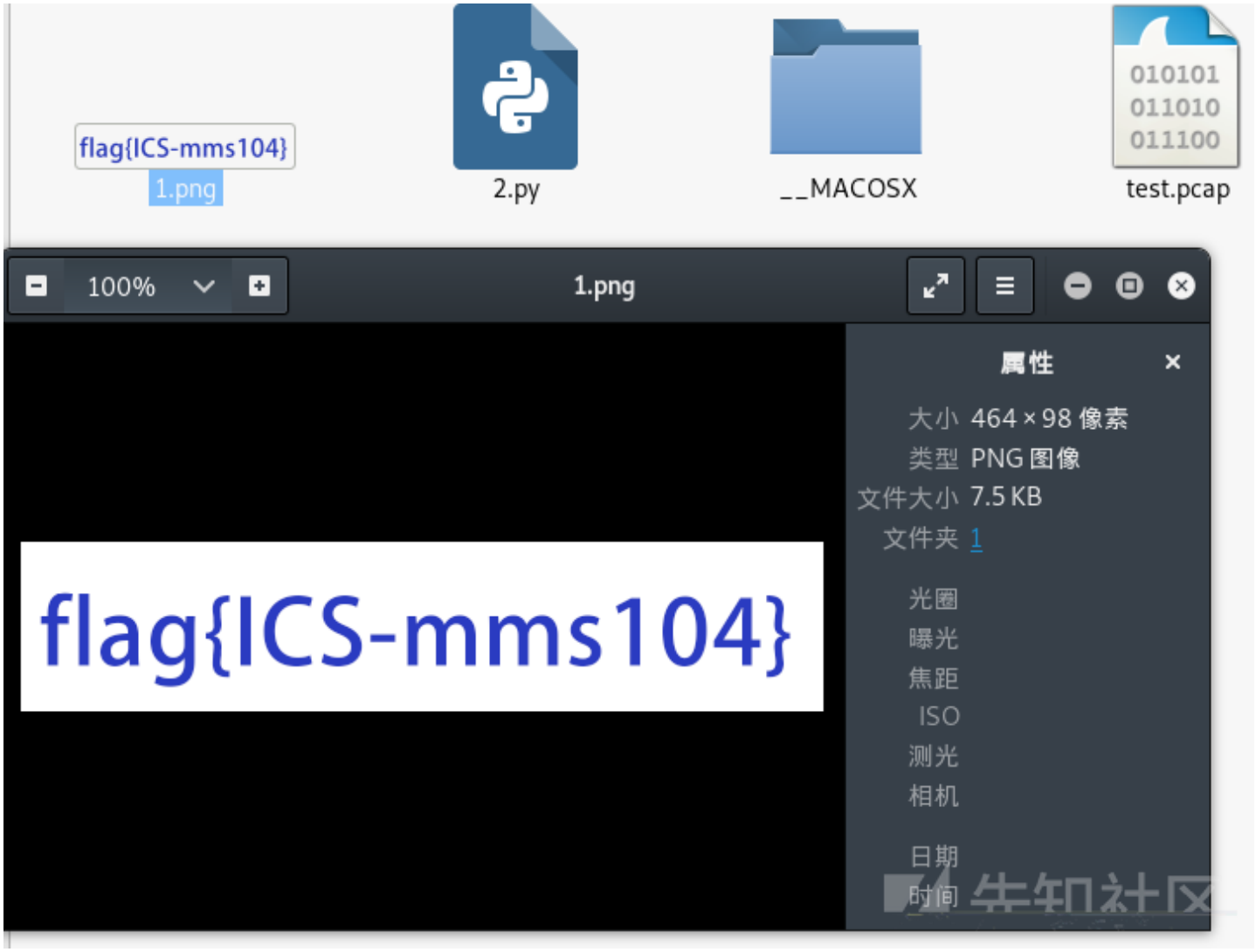
然而通过多次分析与flag.txt相对应的流量包中，没有发现flag.txt的内容，于是换一个思路，对流量包进行关键字（jpg、png、zip、rar、flag）搜索，查看是否存在其他

```
grep "flag" -a test.pacp
grep ".zip" -a test.pacp
grep ".jpg" -a test.pacp
grep ".png" -a test.pacp
```

最终，发现存在base64加密的png图片码，如图所示：

运行脚本，将图片码进行base64解码，解码后得到写有Flag的图片，Flag为flag{ICS-mm104}，脚本和原始图片如下：

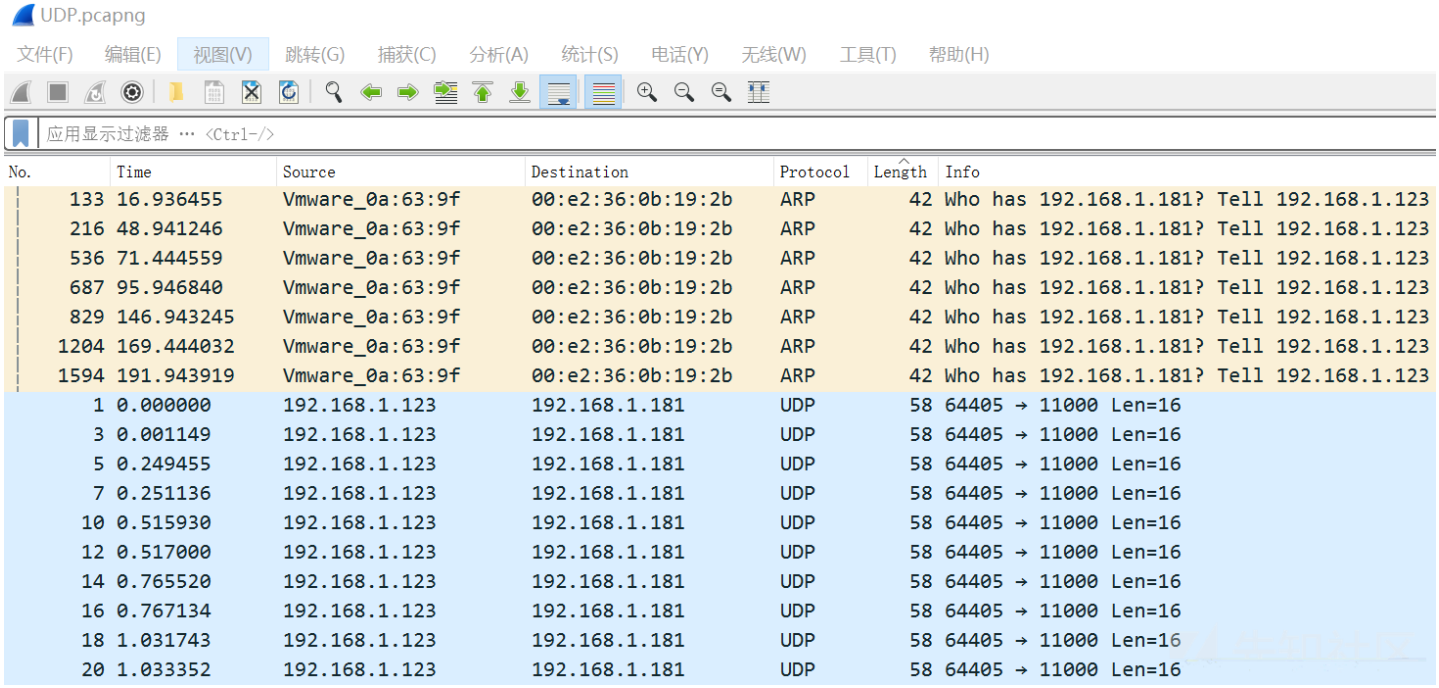
```
# coding=utf-8
import os, base64
img_str = 'iVBORw0KGgoAAAANSUhEUgAAAdAAAABiCAYAAADgKILKAAAAAXNSR0IArs4c6QAAAARnQU1BAACxjwv8YQUAAAAJcEhZcwAADsMAAA7DAcdvqGQA
img_data = base64.b64decode(img_str)
with open('1.png', 'wb') as f:
  f.write(img_data)
print 'successful'
```

3. 

1.3.工业协议分析2

题目：在进行工业企业检查评估工作中，发现了疑似感染恶意软件的上位机。现已提取出上位机通信流量，尝试分析出异常点，获取FLAG。
题目附件连接：https://pan.baidu.com/s/1efRIQfLXkXDwrMhJZC2ZOA （提取码：vxx2 ）
解题步骤：

打开流量包，发现存在关于ARP、UDP、SNA协议的流量包，其中存在大量的UDP流量，如图所示：



No.	Time	Source	Destination	Protocol	Length	Info
133	16.936455	Vmware_0a:63:9f	00:e2:36:0b:19:2b	ARP	42	Who has 192.168.1.181? Tell 192.168.1.123
216	48.941246	Vmware_0a:63:9f	00:e2:36:0b:19:2b	ARP	42	Who has 192.168.1.181? Tell 192.168.1.123
536	71.444559	Vmware_0a:63:9f	00:e2:36:0b:19:2b	ARP	42	Who has 192.168.1.181? Tell 192.168.1.123
687	95.946840	Vmware_0a:63:9f	00:e2:36:0b:19:2b	ARP	42	Who has 192.168.1.181? Tell 192.168.1.123
829	146.943245	Vmware_0a:63:9f	00:e2:36:0b:19:2b	ARP	42	Who has 192.168.1.181? Tell 192.168.1.123
1204	169.444032	Vmware_0a:63:9f	00:e2:36:0b:19:2b	ARP	42	Who has 192.168.1.181? Tell 192.168.1.123
1594	191.943919	Vmware_0a:63:9f	00:e2:36:0b:19:2b	ARP	42	Who has 192.168.1.181? Tell 192.168.1.123
1	0.000000	192.168.1.123	192.168.1.181	UDP	58	64405 → 11000 Len=16
3	0.001149	192.168.1.123	192.168.1.181	UDP	58	64405 → 11000 Len=16
5	0.249455	192.168.1.123	192.168.1.181	UDP	58	64405 → 11000 Len=16
7	0.251136	192.168.1.123	192.168.1.181	UDP	58	64405 → 11000 Len=16
10	0.515930	192.168.1.123	192.168.1.181	UDP	58	64405 → 11000 Len=16
12	0.517000	192.168.1.123	192.168.1.181	UDP	58	64405 → 11000 Len=16
14	0.765520	192.168.1.123	192.168.1.181	UDP	58	64405 → 11000 Len=16
16	0.767134	192.168.1.123	192.168.1.181	UDP	58	64405 → 11000 Len=16
18	1.031743	192.168.1.123	192.168.1.181	UDP	58	64405 → 11000 Len=16
20	1.033352	192.168.1.123	192.168.1.181	UDP	58	64405 → 11000 Len=16

首先对UDP流量包进行分析，分析发现UDP流量包的长度存在大量相同，一共出现的长度分别为16 17 12 14 10 18 19 20 22 25 32 89 95 104 105 116 131 137 524
528，在这些长度中仅12，89，104，105，131，137出现一次，其余长度多次出现，于是猜测这仅出现一次的流量包存在异常，于是分别分析12，89，104，105，13

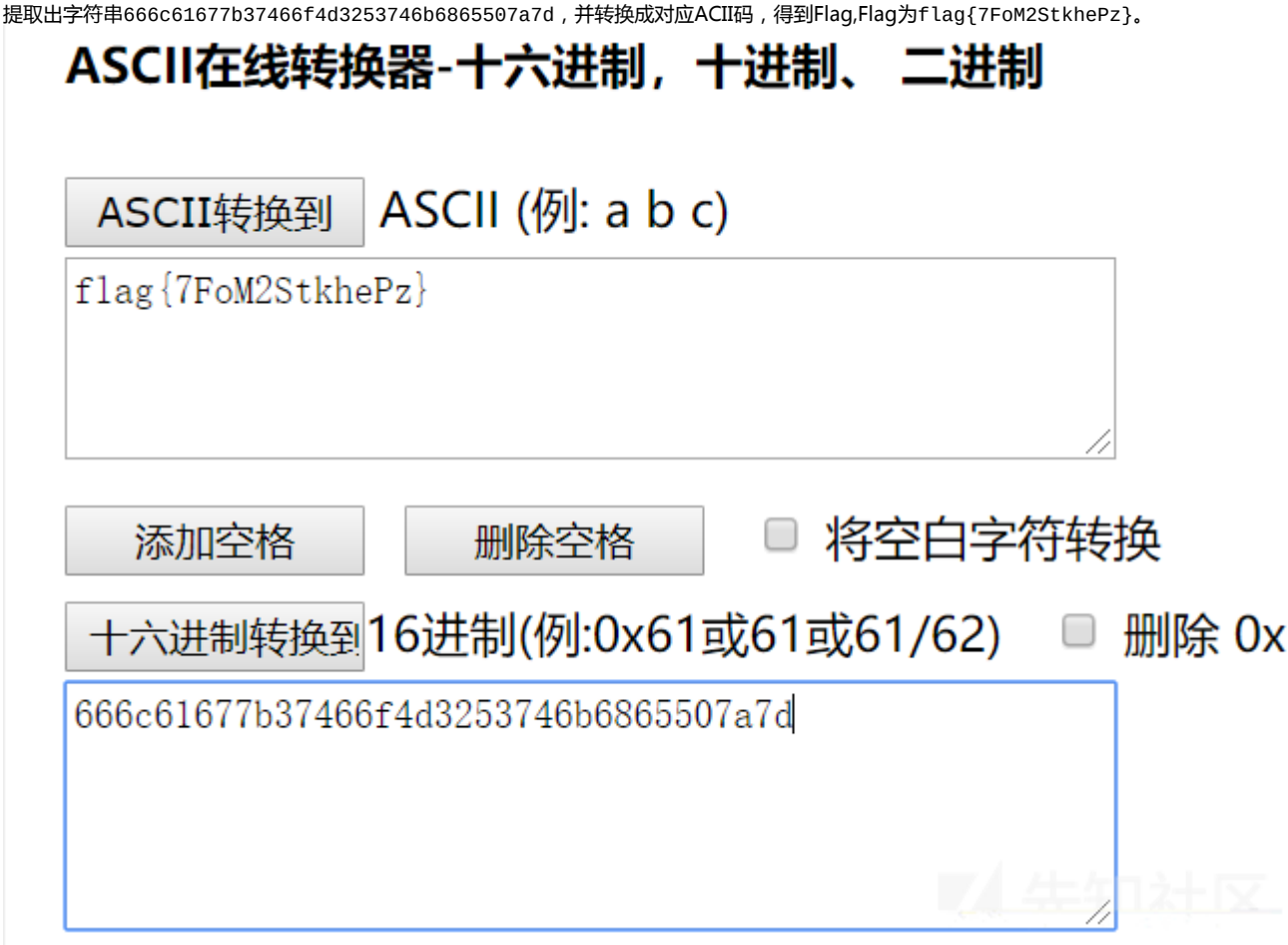
652	91.901650	192.168.1.123	192.168.1.181	UDP	64 64406 → 11000 Len=22
739	100.436232	192.168.1.181	192.168.1.123	UDP	64 11000 → 64406 Len=22
743	100.438565	192.168.1.181	192.168.1.123	UDP	64 11000 → 64406 Len=22
794	143.065734	192.168.1.123	192.168.1.181	UDP	64 64406 → 11000 Len=22
150	23.965514	192.168.1.123	192.168.1.181	UDP	67 64406 → 11000 Len=25
175	32.412085	192.168.1.181	192.168.1.123	UDP	74 11000 → 64406 Len=32
741	100.437598	192.168.1.181	192.168.1.123	UDP	74 11000 → 64406 Len=32
167	32.404322	192.168.1.181	192.168.1.123	UDP	131 11000 → 64406 Len=89
203	45.021168	192.168.1.123	192.168.1.181	UDP	137 64406 → 11000 Len=95
440	65.266055	192.168.1.123	192.168.1.181	UDP	137 64406 → 11000 Len=95
148	22.492885	192.168.1.123	192.168.1.181	UDP	146 64406 → 11000 Len=104
790	142.976251	192.168.1.123	192.168.1.181	UDP	147 64406 → 11000 Len=105
123	15.639029	192.168.1.181	192.168.1.123	UDP	158 11000 → 64406 Len=116
136	21.026137	192.168.1.181	192.168.1.123	UDP	158 11000 → 64406 Len=116
733	100.430392	192.168.1.181	192.168.1.123	UDP	173 11000 → 6440[illegible] Len=131
648	91.813108	192.168.1.123	192.168.1.181	UDP	179 64406 → 11000 Len=137
9	0.329000	192.168.1.243	239.255.255.250	SSDP	216 M-SEARCH * HTTP/1.1
26	1.330596	192.168.1.243	239.255.255.250	SSDP	216 M-SEARCH * HTTP/1.1
46	2.330743	192.168.1.243	239.255.255.250	SSDP	216 M-SEARCH * HTTP/1.1
63	3.331700	192.168.1.243	239.255.255.250	SSDP	216 M-SEARCH * HTTP/1.1
761	120.329607	192.168.1.243	239.255.255.250	SSDP	216 M-SEARCH * HTTP/1.1
762	121.330532	192.168.1.243	239.255.255.250	SSDP	216 M-SEARCH * HTTP/1.1
766	122.331482	192.168.1.243	239.255.255.250	SSDP	216 M-SEARCH * HTTP/1.1
767	123.332147	192.168.1.243	239.255.255.250	SSDP	216 M-SEARCH * HTTP/1.1
4	0.002151	192.168.1.181	192.168.1.123	UDP	566 11000 → 64405 Len=524
8	0.252397	192.168.1.181	192.168.1.123	UDP	566 11000 → 64405 Len=524
13	0.518054	192.168.1.181	192.168.1.123	UDP	566 11000 → 64405 Len=524

Wireshark · 分组 733 · UDP.pcapng

```
> Frame 733: 173 bytes on wire (1384 bits), 173 bytes captured (1384 bits) on interface 0
> Ethernet II, Src: 00:e2:36:0b:19:2b (00:e2:36:0b:19:2b), Dst: Vmware_0a:63:9f (00:0c:29:0a:63
> Internet Protocol Version 4, Src: 192.168.1.181, Dst: 192.168.1.123
> User Datagram Protocol, Src Port: 11000, Dst Port: 64406
v Data (131 bytes)
    Data: 4c004fa18300e3002527252711006200000000000000000…
    [Length: 131]
```

```
0000  00 0c 29 0a 63 9f 00 e2  36 0b 19 2b 08 00 45 00   ··)·c··· 6··+··E·
0010  00 9f 6b 5b 00 00 80 11  4a 72 c0 a8 01 b5 c0 a8   ··k[···· Jr······
0020  01 7b 2a f8 fb 96 00 8b  31 22 4c 00 4f a1 83 00   ·{*····· 1"L·O···
0030  e3 00 25 27 25 27 11 00  62 00 00 00 00 00 00 00   ··%'%'·· b·······
0040  00 00 00 00 00 00 00 00  00 00 00 20 00 00 00 18   ········ ··· ····
0050  00 00 00 38 00 00 00 2a  00 00 00 62 00 00 00 00   ···8···* ···b····
0060  00 00 00 62 00 00 00 00  00 00 00 00 02 14 00 4d   ···b···· ·······M
0070  41 49 4e 5f 28 d6 f7 b3  cc d0 f2 29 00 03 01 02   AIN_(··· ···)····
0080  00 c8 00 00 01 00 25 00  36 36 36 63 36 31 36 37   ·····%· 666c6167
0090  37 62 33 37 34 36 36 66  34 64 33 32 35 33 37 34   7b37466f 4d325374
00a0  36 62 36 38 36 35 35 30  37 61 37 64 00            6b686550 7a7d·
```

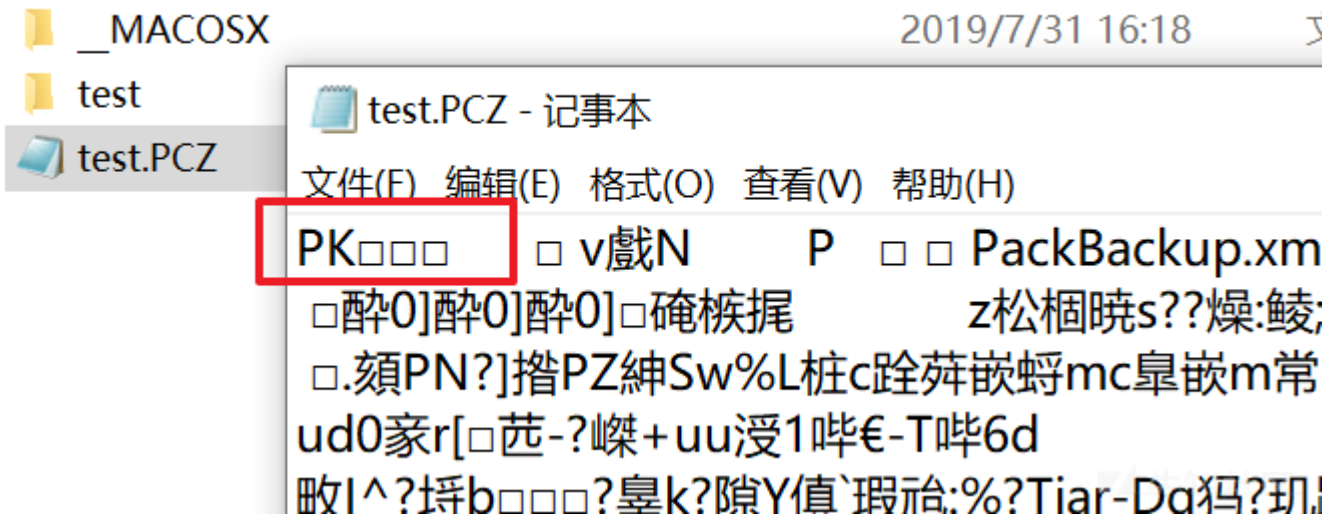
提取出字符串666c61677b37466f4d3253746b6865507a7d，并转换成对应ACII码，得到Flag,Flag为flag{7FoM2StkhePz}。

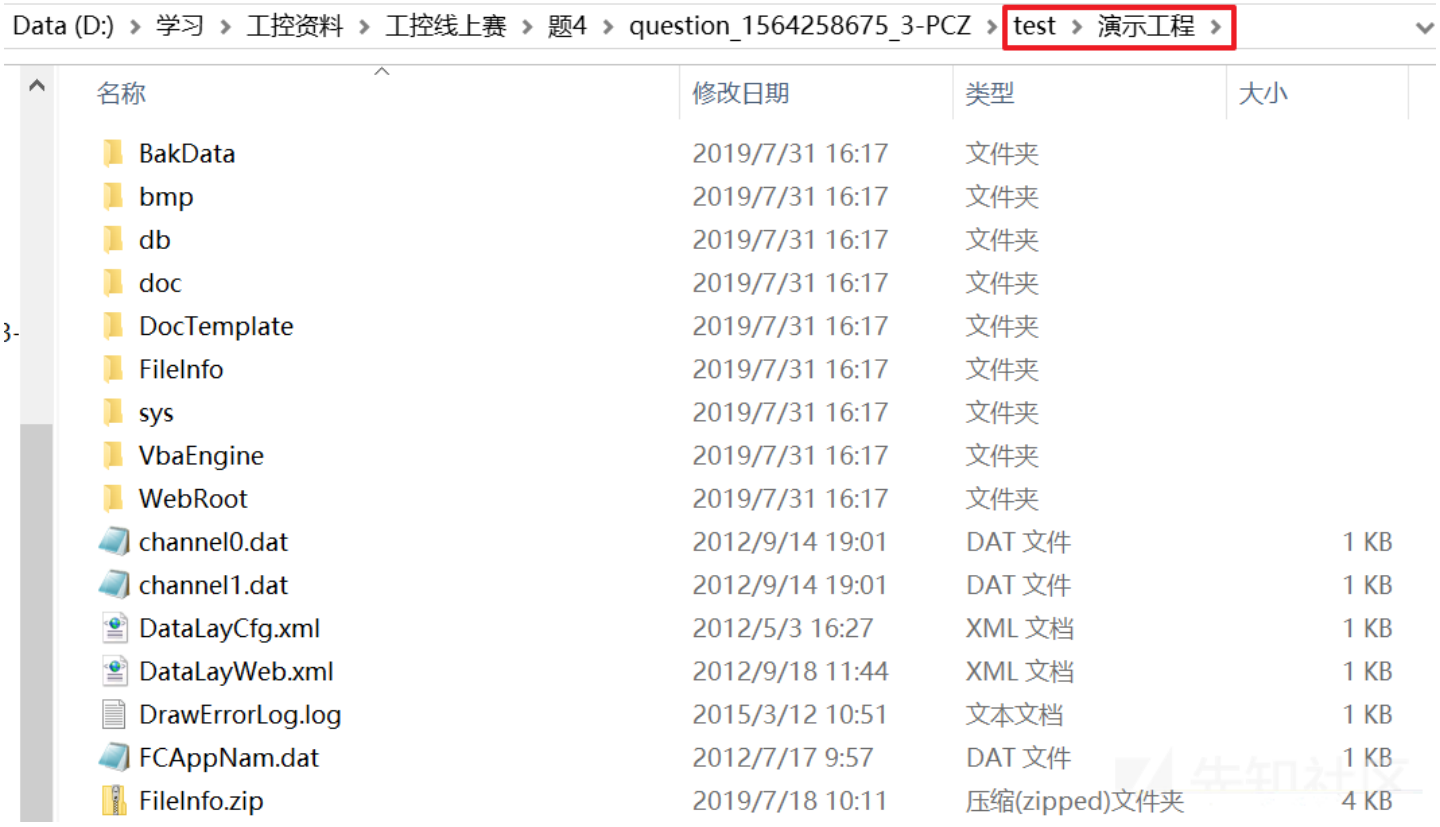


1.4.组态软件安全分析

题目：一些组态软件中进行会配置连接很多PLC设备信息。我们在SCADA工程中写入了flag字段，请获取该工程flag
题目附件连接：链接：https://pan.baidu.com/s/1LmaQpEJ-n3t654BhdjUrRg (提取码：xbuu)
解题步骤：

解压附件，发现得到一个.PCZ的文件，用记事本打开发现文件头为PK，于是将.PCZ的文件后缀改为.zip，解压后得到一个演示工程的文件夹，里面包含了很多文件，如图





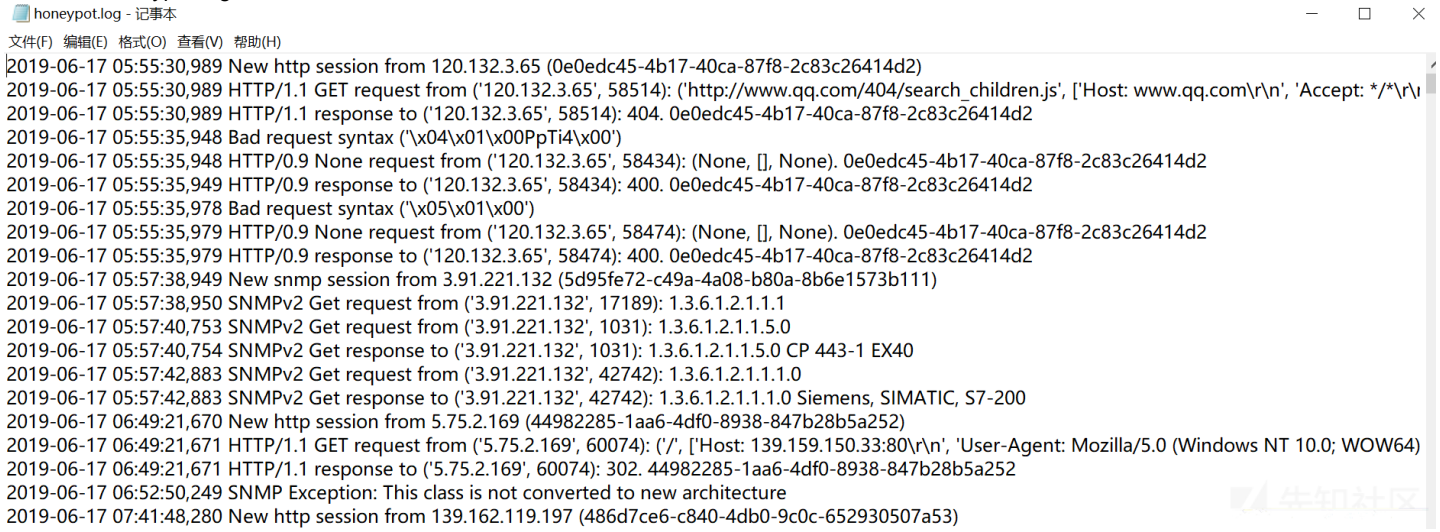
题目表明Flag就在文件夹中的某一个文件中，一个个打开审计过于麻烦，可以利用linux系统的grep指令，帮助我们在文件夹中查找指定关键字，在演示工程的文件夹中，-r "flag" ./进行搜索，最终得到Flag，Flag为flag{D076-4D7E-92AC-A05ACB788292}。

```
root@kali:/mnt/hgfs/S/test# grep -r "flag" ./
匹配到二进制文件 ./演示工程/doc/%path%/webroot/Http/AlarmCenter.dll
匹配到二进制文件 ./演示工程/doc/%path%/webroot/Http/HisDataCenter.dll
匹配到二进制文件 ./演示工程/doc/%path%/webroot/Http/Report.dll
匹配到二进制文件 ./演示工程/doc/%path%/webroot/Http/ReportFile/CellCtrl5u.ocx
匹配到二进制文件 ./演示工程/doc/%path%/webroot/Http/ReportFile/CellResChs.dll
匹配到二进制文件 ./演示工程/WebRoot/Http/AlarmCenter.dll
匹配到二进制文件 ./演示工程/WebRoot/Http/HisDataCenter.dll
匹配到二进制文件 ./演示工程/WebRoot/Http/Report.dll
匹配到二进制文件 ./演示工程/WebRoot/Http/ReportFile/CellCtrl5u.ocx
匹配到二进制文件 ./演示工程/WebRoot/Http/ReportFile/CellResChs.dll
./演示工程/演示Demo.eproj:<ProjInfo ProjectName="演示工程" ProjectGuid="{flag{D076-4D7E-92AC-A05ACB788292}}" ProjectDesc="" Resolution
="1024*768" CreateTime="04/06/11 14:33:05" LastModifyTime="04/06/11 14:33:05"/>
root@kali:/mnt/hgfs/S/test#
```

1.5.工控蜜罐日志分析

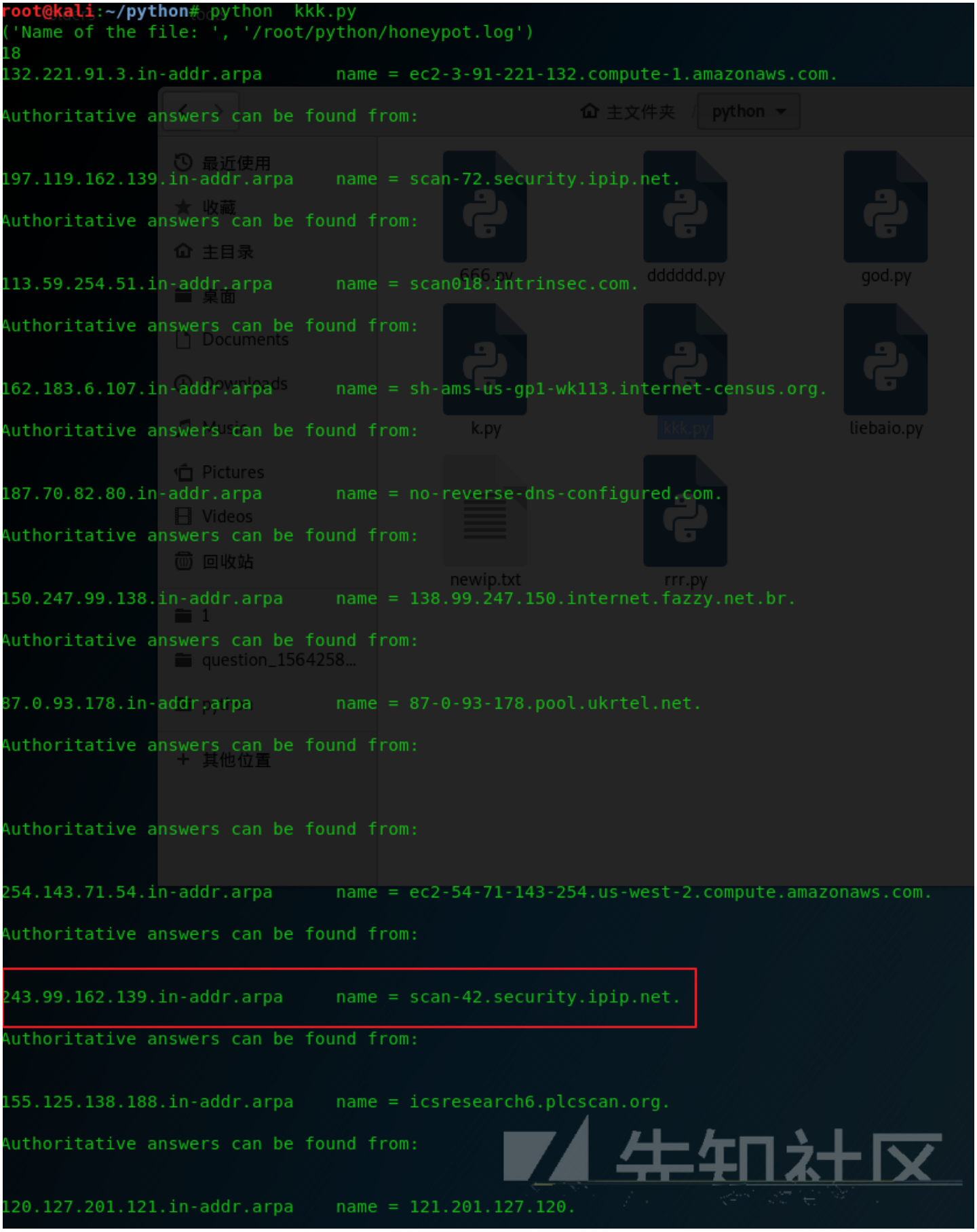
题目：工控安全分析人员在互联网上部署了工控仿真蜜罐，通过蜜罐可抓取并分析互联网上针对工业资产的扫描行为，将存在高危扫描行为的IP加入防火墙黑名单可有效减少
题目附件连接：https://pan.baidu.com/s/1WtuqaE64Zm2HqTPseIKUig （提取码：zu58）
解题步骤：

附件是一个henoypot.log，内容格式如图所示：



根据题目提示，Flag为某个IP对应的域名，于是可以编写脚本，首先提取出日志的IP，并且去重IP，然后再对每一个IP反查域名，寻找正确的域名，脚本和运行结果如下

```
#-*- coding:utf-8 -*-
import fileinput
import re
import os
import shutil
def readIp():
with open(r'/root/python/honeypot.log', 'r') as f:
     for line in f.readlines():
        result2 = re.findall('[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}',line) #■■ip■■■■■
        if not result2 == []:
            result = result2[0] + '\n'
        with open('/root/python/ip.txt', 'a+') as w:
            w.write(result)
def setIp():#■■
a=0
readDir = "/root/python/ip.txt"
writeDir = "/root/python/newip.txt"#new
lines_seen = set()
outfile = open(writeDir, "w")
f = open(readDir, "r")
for line in f:
  if line not in lines_seen:
    a+=1
    outfile.write(line)
    lines_seen.add(line)
print(a)
outfile.close()
def readDns():
with open(r'/root/python/newip.txt', 'r') as g:
    for i in g.readlines():
        com=os.popen('nslookup %s'%i)
        comm=com.read()
        if comm.find('NXDOMAIN')==-1:
            print comm
if __name__ == '__main__':
readIp()
setIp()
readDns()
```

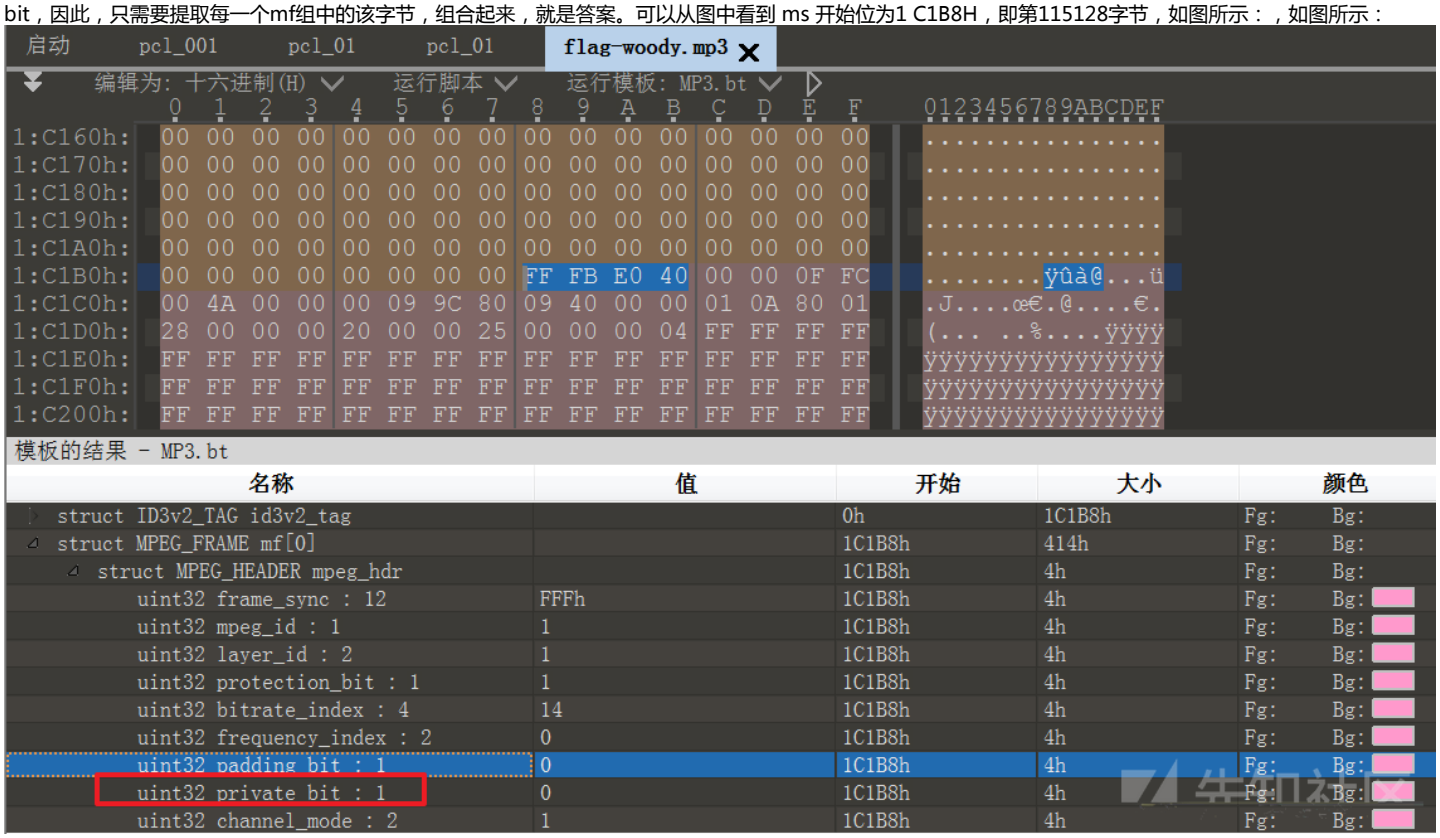


最终尝试域名，找到正确的域名为：`scan-42.security.ipip.net`，Flag为`scan-42.security.ipip.net`。

1.6.隐信道数据安全分析

题目：安全分析人员截获间谍发出的秘密邮件，该邮件只有一个mp3文件，安全人员怀疑间谍通过某种private的方式将信息传递出去，尝试分析该文件，获取藏在文件中的
题目附件连接：链接：https://pan.baidu.com/s/1IcP-kaKw02jHUOIgTEOh6g （提取码：kgqa ）
解题步骤：

1. 题目提示文件使用了private加密信息，在010Editor中打开mp3文件，发现存在private bit，因此，只需要提取每一个mf组中的该字节，组合起来，就是答案。可以从图中看到 ms 开始位为1 C1B8H，即第115128字节，如图所示：，如图所示：




```
uint32 frame_sync : 12
   uint32 mpeg_id : 1
   uint32 layer_id : 2
   uint32 protection_bit : 1
   uint32 bitrate_index : 4
   uint32 frequency_index : 2
   uint32 padding_bit : 1
   uint32 private_bit : 1
   uint32 channel_mode : 2
   uint32 mode_extension : 2
   uint32 copyright : 1
   uint32 original : 1
   uint32 emphasis : 2
```

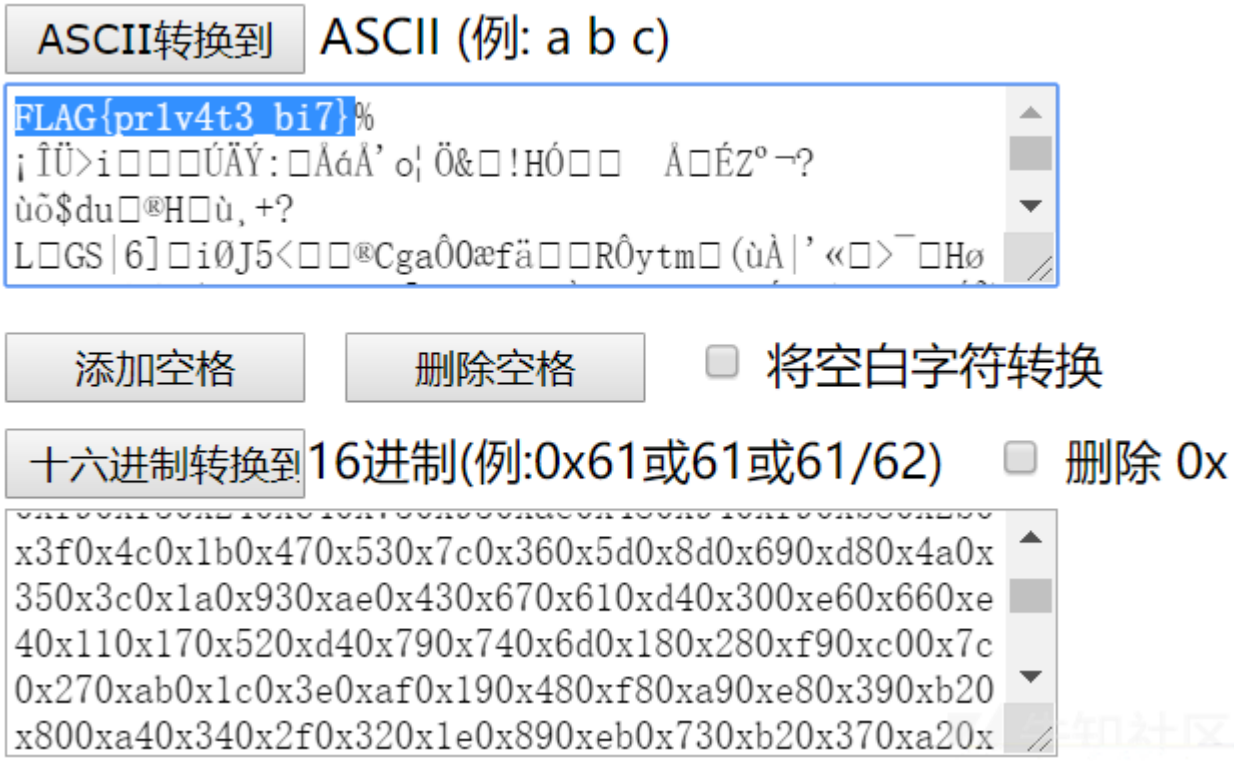
12+1+2+1+4+2+1+1+2+2+1+1+2=32，即总共4字节，private_bit
为24，所在的字节为第3个字节因此要从前一个，即第二个字节开始提取内容，该字节对应的地址为
115130观察每一个mf组，大小都为414h，即1044字节，因此可以得到以下脚本：

```
# coding:utf-8
import re
import binascii
n = 115130
result = ''
fina = ''
file = open('flag-woody.mp3','rb')
while n < 2222222 :
   file.seek(n,0)
   n += 1044
   file_read_result = file.read(1)
   read_content = bin(ord(file_read_result))[-1]
   result = result + read_content
textArr = re.findall('.{'+str(8)+'}', result)
textArr.append(result[(len(textArr)*8):])
for i in textArr:
   fina = fina + hex(int(i,2))[2:].strip('\n')
fina = fina#.decode('hex')
print (fina)
```

将得到的字符串

```
464c41477b707231763474335f6269377d25a1cedc3e69888894dac4dd3a87c5e1c5276fa6d626832148d39288a0c596c95abaac3f09f9f524647595ae4894
```

转换对应的ASCII码，得到Flag，Flag为FLAG{pr1v4t3_bi7}。



上一篇：缓冲区溢出攻击样例分析 下一篇：从 XSS Payload 学习浏...

1. 0 条回复
   - 动动手指，沙发就是你的了！