CTFHUB技能树-Misc-流量分析-ICMP

原创

valecalida 于 2021-01-13 11:58:35 发布 2285 收藏 8
分类专栏： CTF writeup python 文章标签： python CTFHub ICMP


CTF 同时被 3 个专栏收录
21 篇文章 0 订阅
订阅专栏


writeup
2 篇文章 0 订阅
订阅专栏


python
39 篇文章 1 订阅
订阅专栏

目录

Tips:代码仅供借鉴学习，还请大家多多思考

ICMP-Data：

ICMP-Length：

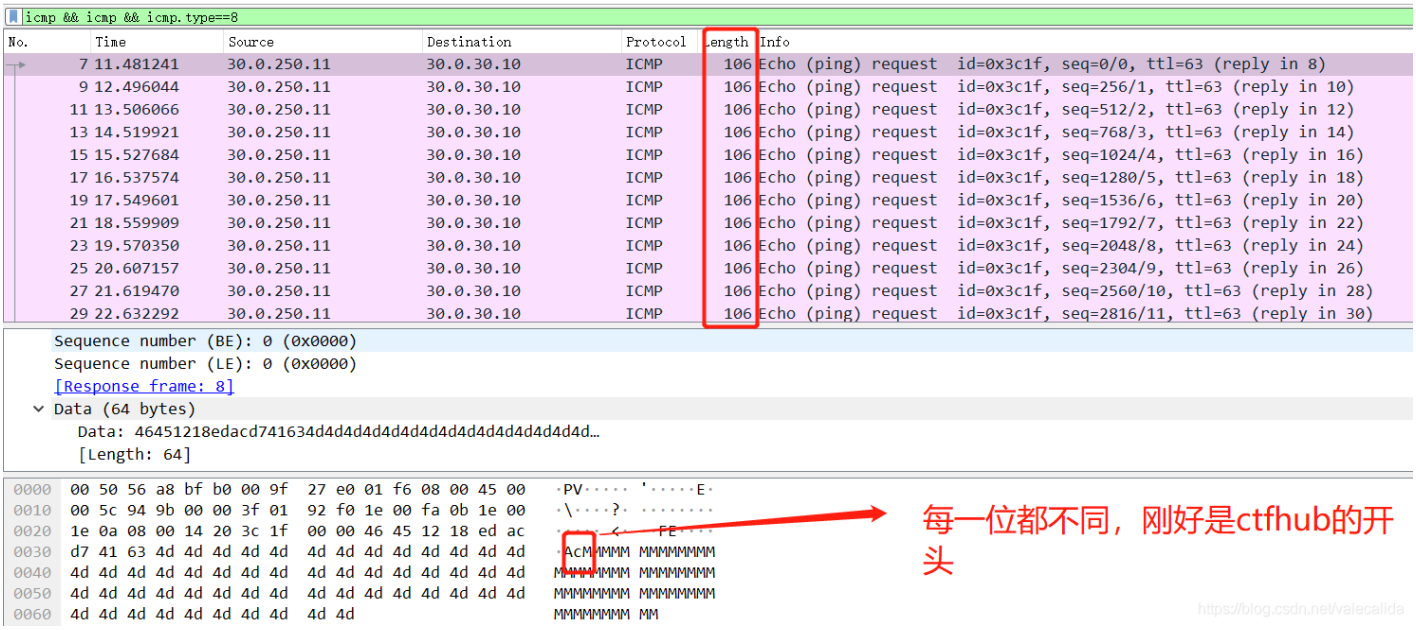
ICMP-LengthBinary

如博客有侵权请联系我，这边立马做处理，如果文章内容有问题，也请私信给我，我会纠正，本人是菜狗，拒绝喷子

Tips:代码仅供借鉴学习，还请大家多多思考

ICMP-Data：

根据题目给出的提示进行过滤显示，这里可以看到，ICMP协议Data部分的内容发生了变动，看后面的流量很容易就发现了ctfhub这个字符串，所以根据字符在data中的位置取值并转化为字符串即得flag




```
# coding = utf-8
# --author：valecalida--
from os import system as get_hex
# 调用tshark时需要将tshark加入环境变量，且脚本需要与流量包在一个路径下

get_hex("tshark -r icmp_data.pcap -Y \"icmp && icmp.type==8\" -T fields -e data > flag.txt")
f = open('flag.txt', 'r')
flag = ''
for line in f.readlines():
    flag += chr(int(line[16:18], 16))
print(flag)
f.close()
```

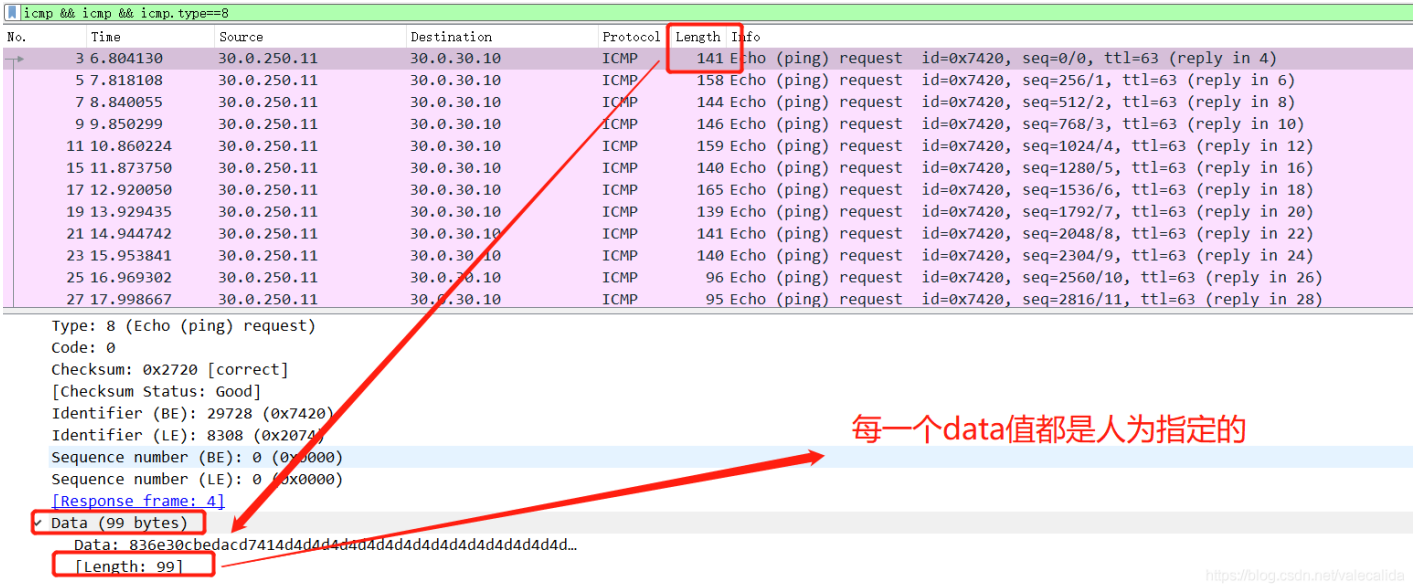
方法二：使用pyshark解析并获取flag

```
# coding = utf-8
# --author: valecalida--

import pyshark

cap = pyshark.FileCapture("icmp_data.pcap", display_filter="icmp && icmp.type==8")
flag = ''
for i in range(0, 25):
    flag += chr(int(str(cap[i].icmp.data_data)[24:26], 16))
print(flag)
cap.close()
```

依然是7行完成flag的获取

ICMP-Length：

根据题目给出的提示进行过滤显示，这里可以看到，ICMP协议Data部分的Length都是人为指定的，所以只需要把这一部分的值取出来，再转换成字符串即可获取flag



因为协议的字段都是层层递进的，所以这个部分根据字段头部取值即可，即data.length

```
# coding = utf-8
# --author: valecalida--
from os import system as get_code

# 调用tshark时需要将tshark加入环境变量，且脚本需要与流量包在一个路径下
get_code('tshark -r icmp_len.pcap -Y "icmp && icmp.type==8" -T fields -e data.len > flag.txt')
f = open('flag.txt','r')
flag = ''
for line in f.readlines():
    flag += (chr(int(line.strip())))
print(flag)
f.close()
```

方法二：使用pyshark来对流量进行解析

```
# coding = utf-8
# --author: valecalida--
In [1]: import pyshark

In [2]: cap = pyshark.FileCapture('icmp_len.pcap',display_filter="icmp && icmp.type==8")

In [3]: flag = ''

In [4]: for i in range(0,18):
   ...:     pkt = cap[i]
   ...:     flag += (chr(int(pkt.icmp.data_len)))
   ...:

In [5]: flag
Out[5]: 'ctfhub{acb659f023}'
In [6]: cap.close()
```

这样看起来不涉及到文件的创建以及读取处理，似乎更快一点，脚本稍后完善

```
# coding = utf-8
# --author: valecalida--
import pyshark
cap = pyshark.FileCapture('icmp_len.pcap', display_filter="icmp && icmp.type==8")
flag = ''
for i in range(0, 18):
    flag += (chr(int(cap[i].icmp.data_len)))
print(flag)
cap.close()
```

ICMP-LengthBinary

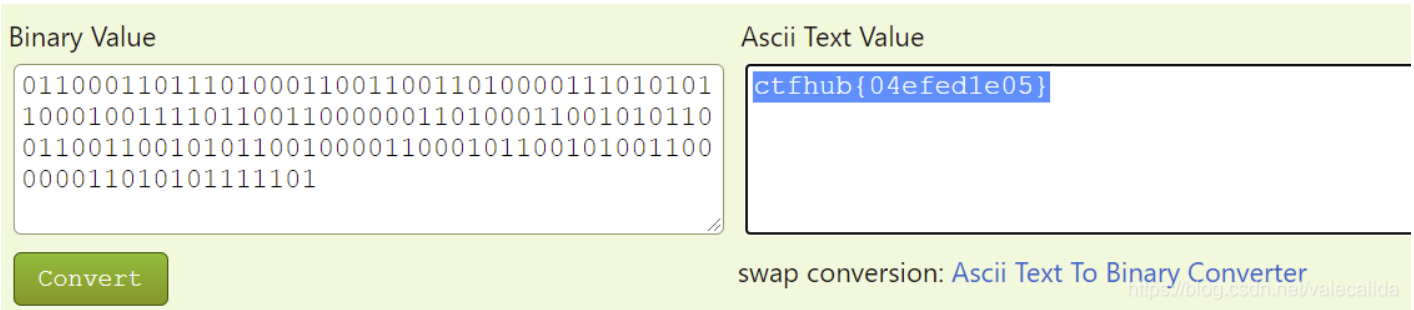
题目很直接的给了提示，就是二进制与length的关系，使用wireshark打开流量包查看，使用过滤器icmp&&icmp.type==8来进行过滤，查看每一条流量的length值，发现都是32或64，直接编写脚本

```
# coding = utf-8
# --author: valecalida--

import pyshark

cap = pyshark.FileCapture('icmp_len_binary.pcap', display_filter="icmp && icmp.type==8")
cap.load_packets()
flag = ''
con1 = ""
con2 = ""
for i in range(0, len(cap)):
    if cap[i].icmp.data_len == '32':
        con1 += '0'
        con2 += '1'
    elif cap[i].icmp.data_len == '64':
        con1 += '1'
        con2 += '0'
print(con1)
print(con2)
cap.close()
```

运行得到两串二进制字符串

```
01100011011101000110011001101000011101010110001001111011001100000011010001100101011001100110010101100100001

10011100100010111001100110010111100010101001110110000100110011111100101110011010100110011001101010011011110
```

直接在线解码，可以看到，直接得到了flag



或者直接用下面的脚本跑出flag

```
# coding = utf-8
# --author: valecalida--
import binascii
import pyshark

cap = pyshark.FileCapture('icmp_len_binary.pcap', display_filter="icmp && icmp.type==8")
cap.load_packets()
flag = ''
con1 = ""
con2 = ""
for i in range(0, len(cap)):
    if cap[i].icmp.data_len == '32':
        con1 += '0'
        con2 += '1'
    elif cap[i].icmp.data_len == '64':
        con1 += '1'
        con2 += '0'
print(binascii.a2b_hex(hex(int(con1, base=2))[2:]))
print(binascii.a2b_hex(hex(int(con2, base=2))[2:]))
cap.close()
```

运行得到flag

```
b'ctfhub{04efed1e05}'
b'\x9c\x8b\x99\x97\x8a\x9d\x84\xcf\xcb\x9a\x99\x9a\x9b\xce\x9a\xcf\xca\x82'
```

如博客有侵权请联系我，这边立马做处理，如果文章内容有问题，也请私信给我，我会纠正，本人是菜狗，拒绝喷子