

# 【愚公系列】2022年01月 攻防世界-进阶题-MISC-81(传感器

1)

原创

愚公搬代码 于 2022-01-30 21:25:36 发布 6677 收藏

分类专栏: [#CTF-攻防世界-MISC](#) 文章标签: [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/aa2528877987/article/details/122755955>

版权



[CTF-攻防世界-MISC 专栏收录该内容](#)

98 篇文章 0 订阅

订阅专栏

## 文章目录

一、传感器1

二、答题步骤

1.差分曼切斯特编码

总结

## 一、传感器1

题目链接: [https://adworld.xctf.org.cn/task/task\\_list?type=misc&number=1&grade=1&page=5](https://adworld.xctf.org.cn/task/task_list?type=misc&number=1&grade=1&page=5)

题目描述: 已知ID为0x8893CA58的温度传感器的未解码报文为: 3EAAAAA56A69AA55A95995A569AA95565556 此时有另一个相同型号的传感器, 其未解码报文为: 3EAAAAA56A69AA556A965A5999596AA95656 请解出其ID, 提交格式为flag{xxx}

攻防世界 World of Attack&Defense

答题 竞赛 排行榜 队伍 商城 消息 愚公搬代码 中 / En

返回 本题用时: 1分1秒 misc 积分: 199分 本题金币: 4个

传感器1 最佳Writeup由系统战队 • admin提供 WP 建议

难度系数: ★★★★★ 4.0

题目来源: ciscn-2017

题目描述: 已知ID为0x8893CA58的温度传感器的未解码报文为: 3EAAAA56A69AA55A95995A569AA9556556 此时有另一个相同型号的传感器, 其未解码报文为: 3EAAAA56A69AA556A965A5999596AA95656 请解出其ID, 提交格式为flag{xxx}

题目场景: 暂无

题目附件: 暂无

实时消息

flag..

XCTF 高校网络安全专题挑战赛 @愚公搬代码

## 二、答题步骤

### 1.差分曼切斯特编码

```

#!/usr/bin/env python
#coding:utf-8

import re

#hex1 = 'AAAAA56A69AA55A95995A569AA95565556' # # 0x8893CA58
hex1 = 'AAAAA56A69AA556A965A5999596AA95656'

def bintohehex(s1):
    s2 = ''
    s1 = re.findall('.{4}',s1)
    print ('每一个hex分隔:',s1)
    for i in s1:
        s2 += str(hex(int(i,2))).replace('0x','')

    print ('ID:',s2)

def diffmqst(s):
    s1 = ''
    s = re.findall('.{2}',s)
    cc = '01'
    for i in s:
        if i == cc:
            s1 += '0'
        else:
            s1 += '1'
        cc = i # 差分加上cc = i

    print ('差分曼切斯特解码:',s1)
    bintohehex(s1)

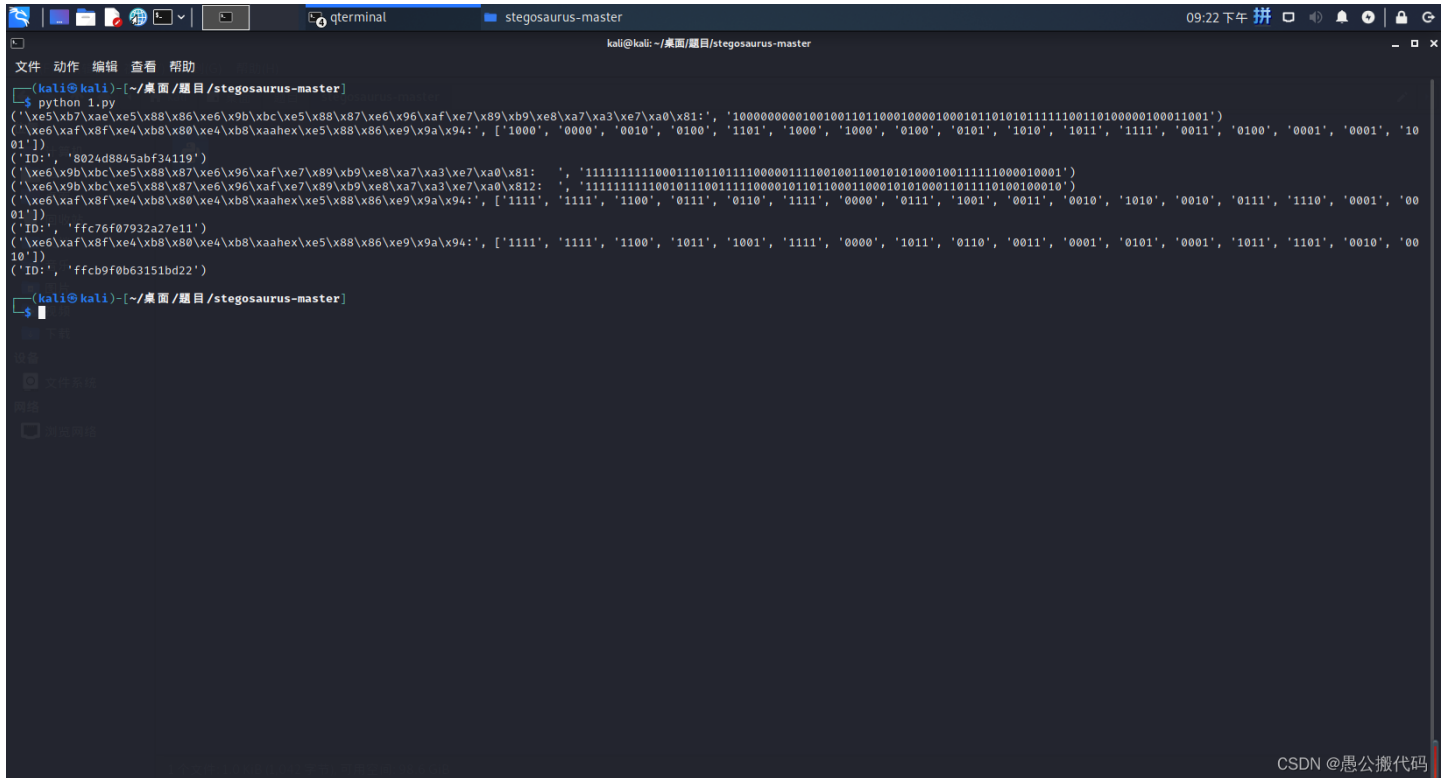
def mqst(s): #只能算曼切斯特编码,无法算差分
    mdict = {'5': '00', '6': '01', '9': '10', 'A': '11'}
    a1 = ''.join(mdict[i] for i in s)
    a2 = ''.join(mdict[i][::-1] for i in s)
    print ('曼切斯特解码: ',a1 )
    print ('曼切斯特解码2: ',a2)
    bintohehex(a1)
    bintohehex(a2)

if __name__ == '__main__':
    bin1 = bin(int(hex1,16))[2:]
    diffmqst(bin1)

    mqst(hex1)

```

用AAAAA56A69AA55A95995A569AA95565556(需要去掉3E前缀)运行, 结果是:



```
(kali@kali)~[桌面/题目/stegosaurus-master]
└─$ python 1.py
('\xe5\xb7\xe5\x88\x86\xe6\x9b\xbc\xe5\x88\x87\xe6\x96\xaf\xe7\x89\xb9\xe8\xa7\xa3\xe7\xa0\x81:', '100000000100100110110001000010001011010101111100110100000100011001')
('ID:', '8024d8845abf34119')
('\xe6\x9b\xbc\xe5\x88\x87\xe6\x96\xaf\xe7\x89\xb9\xe8\xa7\xa3\xe7\xa0\x81:', '111111111000111011011110000011110010010010101000100111111000010001')
('\xe6\x9b\xbc\xe5\x88\x87\xe6\x96\xaf\xe7\x89\xb9\xe8\xa7\xa3\xe7\xa0\x812:', '1111111110010110011110000101101100011000101010001101111010010010')
('\xe6\xaf\x8f\xe4\xb8\x80\xe4\xb8\xaaahex\xe5\x88\x86\xe9\x9a\x94:', ['1000', '0000', '0010', '0100', '1101', '1000', '1000', '0100', '0101', '1010', '1011', '1111', '0011', '0100', '0001', '1010', '0101', '1110', '0001', '0001', '1010', '0111', '1100', '0111', '0110', '1111', '0000', '0111', '1001', '0011', '0010', '1010', '0010', '0111', '1110', '0001', '0001', '1011', '1101', '0010', '0010'])
('ID:', 'ffc76f07932a27e11')
('\xe6\xaf\x8f\xe4\xb8\x80\xe4\xb8\xaaahex\xe5\x88\x86\xe9\x9a\x94:', ['1111', '1111', '1100', '1011', '1001', '1111', '0000', '1011', '0110', '0011', '0001', '0101', '0001', '1011', '1101', '0010', '0010'])
('ID:', 'ffc76f07932a27e11')
```

拿AAAAA56A69AA556A965A5999596AA95656去跑脚本, 得到差分曼切斯特编码为8024d8845abf34119, 左边去掉5个字符, 右边去掉4个字符, 换成大写就是flag: `flag{8845ABF3}`

## 总结

在曼彻斯特编码中, 每一位的中间有一跳变, 位中间的跳变既作时钟信号, 又作数据信号; 从高到低跳变表示"1", 从低到高跳变表示"0"。还有一种是差分曼彻斯特编码, 每位中间的跳变仅提供时钟定时, 而用每位开始时有无跳变表示"0"或"1", 有跳变为"0", 无跳变为"1"。