2017 信息安全大赛 crypto 传感器

原创

DDragon321 于 2019-07-26 12:04:16 发布 638 收藏 1

分类专栏： CTF 文章标签： CTF

本文链接：https://blog.csdn.net/qq_43165101/article/details/97377830

题目：

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flag{hex（不含0x）}。

一般采用的是差分曼切斯特编码,这里简要介绍一下曼切斯特编码和差分曼切斯特。
差分曼切斯特算法由开始时的变化决定编码，例如1001，表示0，差分曼切斯特变化为0不变为1，可以发现给出的编码一共有364*位，因为四位代表1个，所以对应的解码应该为36位，但是实际上对应的id只有84*位，推测有对应的取位方法，所以先根据样例进行破解。
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"。
因此可以知道，先将编码转化为2进制，然后再根据差分曼切斯特编码规则，如果前面跳变为0，不跳变为1.
python代码如下

```
from Crypto.Util.number import *
id1 = 0x8893CA58
#msg = 0x3EAAAAA56A69AA55A95995A569AA95565556
msg = 0x3EAAAAA56A69AA556A965A5999596AA95656
#注意前面的零会自动省略，有时需要控制数组的起始位置
s = bin(msg)
print s
r=""
tmp = 0
for i in xrange(len(s)/2):
    c = s[i*2]
    if c == s[i*2 - 1]:
        r += '1'
    else:
        r += '0'
print hex(int(r,2)).upper()
```

可以先测试一下题目给的样例

结果如图

```
0b1111101010101010101010101001010110101001101001101010100101010110101001010110011001010110100101011010011010101010010101010110010101010101010110
0XB0024D8893CA584181L
```

可以知道取的结果是7到14位

然后把要求的编码放入

计算得出结果为

```
================== RESTART: C:\python\python27\manqiesite.py ==================
0b1111101010101010101010101001010110101001101001101010100101010110101001010110011001010110100101011010011010101010010101010110010101010101010110
0XB0024D8845ABF34119L
```