CRC Forgery

这道题和其它两道题一样，也是采用了异或的加密方法。对于原文的每一位，首先判断其二进制是否为1，如果为1，则使用密钥对之后的内容进行异或，如果为0则移动到下一位。

这道题将明文分成了两段，我们需要从前和从后分别推到中间空缺的16个明文是什么。

对于前一段，由于密钥已知，我们直接使用加密方法，将前一段后面16个字符补0进行加密。对于后一段，使用倒推的方法，由于密钥的最后一位为1，所以根据现存的明文和密文可以判断出这一位的前64位是0还是1，所以可以由此循环倒推得到开头的16个字符。

最后将第一段的最后16个字符和后一段的开头16个字符异或一下，就是中间空缺的部分了。（由于时间原因，部分数学推导省略，通过仔细阅读代码便可领会。）

代码如下：

```
import binascii
import os
import random

def b2n(b):
    res = 0
    for i in b:
        res *= 2
        res += i
    return res

def n2b(n, length):
    tmp = bin(n)[2:]
    tmp = '0'*(length-len(tmp)) + tmp
    return [int(i) for i in tmp]

def s2n(s):
    return int(binascii.hexlify(s), 16)

def crc64(msg):
    msg = n2b(s2n(msg), len(msg)*8)
    msg += const
    print(msg)
    for shift in range(len(msg)-64):
        if msg[shift]:
            for i in range(65):
                msg[shift+i] ^= poly[i]
    res = msg[-64:]
    return b2n(res)

const = n2b(0xdeadbeeffeedcafe, 64)
poly  = n2b(0x10000000247f43cb7, 65)

ans_raw2 = n2b(0x1337733173311337,64)
```

```
src_raw1 = bytearray.fromhex(input())
src_raw2 = bytearray.fromhex(input())
src_raw1 = n2b(s2n(src_raw1),len(src_raw1)*8)
src_raw2 = n2b(s2n(src_raw2),len(src_raw2)*8)
src_raw1+=[0 for _ in range(64)]
src_raw2+=const

for shift in range(len(src_raw1)-64):
    if src_raw1[shift]:
        for i in range(65):
            src_raw1[shift+i] ^= poly[i]

for shift in range(len(src_raw2)):
    ans_raw2.insert(0,0)
    if src_raw2[-shift-1]!=ans_raw2[-shift-1]:
        for i in range(65):
            ans_raw2[i] ^= poly[i]

ans1 = src_raw1[-64:]
ans2 = ans_raw2[:64]

print(hex(b2n(ans1)^b2n(ans2)))
```

运行`nc 111.186.57.85 10081`后，将得到的空缺段落前后分别输入脚本，得到的答案即为中间的空缺部分。输入即可得到flag。