Baby Block

(由于本人没有什么密码学基础，所以一开始看到这道题时我是懵逼的。。。)

根据源代码的内容，它是使用了一种叫做AES-CTR的加密方法。根据网上的资料，得知这种方法将一种叫做"计数器"的东西先通过AES方法加密得到密钥，之后密钥和明文异或得到密文。

所以，如果想要破解得到明文，我们可以先得到加密后的密钥，之后和密文异或之后得到明文。我发现要点好像是在这个"计数器"这里。在这个加密程序中，计数器初始值是根据明文的每一个字节的数字加起来得到的，并且在AES加密的过程中只使用了一次。而明文的长度是已知的30位，所以这个和的最大可能也只有128*30，且明文的前5位一定为0ops{，所以完全可以通过爆破的方法得到这个计数器的初始值，从而计算出用于加密flag的密钥，异或后得到答案。

我们使用pwntools连接服务器，并且通过如下的python脚本得到计数器的初始值：

```
from pwn import *
import time

def int2hex(x):
    ans = ''
    while x>=0xff:
        ans+='ff'
        x-=0xff
    ans+=hex(x)[2:]
    if len(ans)<10:
        ans='0'*(10-len(ans))+ans
    if len(ans)%2==1:
        ans='0'+ans
    return ans

cur_sum = 0x20 * 30
while True:
    sh = remote(host='111.186.57.85',port='10080')
    print(sh.recvline())
    flag_enc = str(sh.recvline(),encoding='ascii')
    print(flag_enc)
    flag_enc = eval(flag_enc[flag_enc.find('flag: ')+6:-1])
    flag_first = b'0ops{'
    first_enc = bytearray()
    for i in range(5):
        first_enc.append(flag_enc[i]^flag_first[i])

    for i in range(30):
        cur_send = int2hex(cur_sum)
        cur_bytes = bytes.fromhex(cur_send)
        print(cur_send)
        sh.sendline(cur_send)
        recv_str = str(sh.recvline(),'ascii')
        recv_str = recv_str[recv_str.find('ciphertext: ')+12:-1]
        recv_str = eval(recv_str)
```

```
        print(recv_str)
        for i in range(min(5,len(recv_str))):
            if recv_str[i]^cur_bytes[i]!=first_enc[i]:
                print(i)
                print('error!')
                break
        else:
            print('something good found!')
            time.sleep(30)

        cur_sum += 1

    sh.close()

# fffffffffffffffffbe
```

得到了初始值之后，我们就可以知道密钥了，最后通过如下的Python脚本和得到的密文异或一下，得到这道题的flag。

```
from pwn import *

sh = remote(host='111.186.57.85',port='10080')
print(sh.recvline())
flag_enc = str(sh.recvline(),encoding='ascii')
print(flag_enc)
flag_enc = eval(flag_enc[flag_enc.find('flag: ')+6:-1])
to_send = 'fffffffffffffffffbe'
to_send += '0'*(60-len(to_send))
send_bytes = bytes.fromhex(to_send)
sh.sendline(to_send)
recv_str = str(sh.recvline(),'ascii')
recv_str = recv_str[recv_str.find('ciphertext: ')+12:-1]
recv_str = eval(recv_str)

for i in range(30):
    print(chr((send_bytes[i]^recv_str[i])^flag_enc[i]),end='')
```